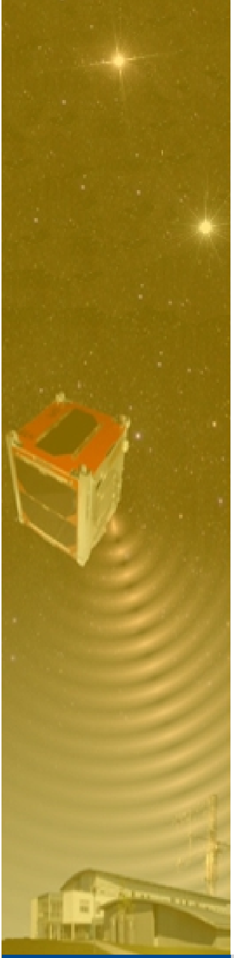


Institut für Informatik
Lehrstuhl für Robotik und Telematik
Prof. Dr. K. Schilling

Würzburger Forschungsberichte in Robotik und Telematik

Uni Wuerzburg Research Notes in Robotics and Telematics

Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Band 7

Christian Herrmann

Robotic Motion Compensation for Applications in Radiation Oncology

Die Schriftenreihe

wird vom Lehrstuhl für Informatik VII: Robotik und Telematik der Universität Würzburg herausgegeben und präsentiert innovative Forschung aus den Bereichen der Robotik und der Telematik.

Die Kombination fortgeschrittener Informationsverarbeitungsmethoden mit Verfahren der Regelungstechnik eröffnet hier interessante Forschungs- und Anwendungsperspektiven. Es werden dabei folgende interdisziplinäre Aufgabenschwerpunkte bearbeitet:

- Robotik und Mechatronik: Kombination von Informatik, Elektronik, Mechanik, Sensorik, Regelungs- und Steuerungstechnik, um Roboter adaptiv und flexibel ihrer Arbeitsumgebung anzupassen.
- Telematik: Integration von Telekommunikation, Informatik und Steuerungstechnik, um Dienstleistungen an entfernten Standorten zu erbringen.

Anwendungsschwerpunkte sind u.a. mobile Roboter, Tele-Robotik, Raumfahrtsysteme und Medizin-Robotik.

Lehrstuhl Informatik VII
Robotik und Telematik
Am Hubland
D-97074 Wuerzburg

Tel.: +49 (0) 931 - 31 - 86678
Fax: +49 (0) 931 - 31 - 86679

schi@informatik.uni-wuerzburg.de
http://www7.informatik.uni-wuerzburg.de

Universitätsbibliothek Würzburg
Am Hubland
D-97074 Würzburg

Tel.: +49 (0) 931 - 31 - 85906

opus@bibliothek.uni-wuerzburg.de
http://opus.bibliothek.uni-wuerzburg.de

ISSN (Internet): 1868-7474
ISSN (Print): 1868-7466
eISBN: 978-3-923959-88-4

Zitation dieser Publikation

HERRMANN, C. (2013). Robotic motion compensation for applications in radiation oncology. Schriftenreihe Würzburger Forschungsberichte in Robotik und Telematik, Band 7. Würzburg: Universität Würzburg.

Robotic Motion Compensation for Applications in Radiation Oncology

Dissertation zur Erlangung des
naturwissenschaftlichen Doktorgrades
der Julius-Maximilians-Universität Würzburg

vorgelegt von

Christian Herrmann

aus

Lohr am Main

Würzburg 2013

Eingereicht am: 08.03.2013
bei der Fakultät für Mathematik und Informatik
1. Gutachter: Prof. Dr. Klaus Schilling
2. Gutachter: Prof. Dr. Dominik Henrich
3. Gutachter: PD Dr. Klaus Bratengeier
Tag der mündlichen Prüfung: 22.07.2013

Danksagung

Mit der Fertigstellung der Dissertation ist es an der Zeit denjenigen zu danken, die mich während der Promotionsphase begleitet und unterstützt haben. Zuerst möchte ich mich herzlich bei meinem Doktorvater Prof. Dr. Klaus Schilling für die Möglichkeit bedanken, eine Promotion unter seiner Betreuung durchzuführen. Seine fortwährende Unterstützung in den letzten Jahren und richtungsweisenden Diskussionen und Anregungen haben mich befähigt, die Promotion erfolgreich abzuschließen. Auch für die abwechslungsreichen Aufgaben am Lehrstuhl in interessanten Forschungsprojekten und der Lehre möchte ihm danken, durch die ich viel Erfahrung sammeln konnte und die meine persönliche und berufliche Weiterentwicklung förderten. Besonderer Dank gilt ebenfalls Prof. Dr. Dominik Henrich und PD Dr. Klaus Bratengeier für die vielen hilfreichen Hinweise und Anregungen sowie für die rasche Erstellung der Gutachten über die Dissertation. Weiterhin möchte ich Prof. Dr. Rainer Kolla und Prof. Dr. Jürgen Albert danken, dass sie als Prüfer für die Disputation zur Verfügung standen.

Die vorliegende Arbeit ist aus dem Forschungsprojekt “Intra-fraktionelle Tumorverfolgung und Nachführung mittels eines automatisierten Bestrahlungstisches” hervorgegangen, welches von der Bayerischen Forschungsstiftung gefördert wurde. Da ohne diese Förderung meine Forschungsarbeit zur Bewegungskompensation in der Radioonkologie nicht möglich gewesen wäre, bin ich der Bayerischen Forschungsstiftung zu großem Dank verpflichtet. In dem interdisziplinärem Projekt war außerdem die Klinik und Poliklinik für Strahlentherapie des Universitätsklinikums Würzburg beteiligt. Allen voran möchte ich dort Dr. Jürgen Wilbert und Kurt Baier für die konstruktive und sehr fruchtbare Zusammenarbeit danken, sowie für viele Gespräche und Anregungen, die einen wichtigen Beitrag zu dieser Arbeit geleistet haben. An die unzähligen Stunden, die wir gemeinsam an Experimenten im Behandlungsraum der Strahlentherpie verbracht haben, erinnere ich mich gerne zurück. Weiterhin geht ein herzlicher Dank an Prof. Dr. Michael Flentje, Prof. Dr. Otto Sauer, Prof. Dr. Matthias Guckenberger, PD Dr. Klaus Bratengeier und Dr. Anne Richter für die angenehme Zusammenarbeit. Mein besonderer Dank gilt darüber hinaus noch Dr. Christian Groh, der mich vor allem in der Endphase meiner Dissertation bei Experimenten mit viel Engagement und Ausdauer unterstützt hat, auch wenn sich diese mal bis in die Nachtstunden gezogen haben. Außerdem sei dem industriellen Projektpartner Medical Intelligence GmbH, Schwabmünchen, für die Bereitstellung der Patientenliege HexaPOD gedankt. Die ständige Verfügbarkeit dieser hat meine Arbeit sehr beschleunigt und vereinfacht. Im besonderen möchte ich Dr. Michael Roth und Christian Janka für die Bereitstellung eines Manipulatorarms für Experimente und den oftmals in Anspruch genommenen Unterstützung im Fehlerfall des Manipulators danken. Eine sehr große Hilfe waren auch die zur Verfügung gestellten Interlocks, die das Experimentieren vereinfacht haben. Des Weiteren geht ein großer Dank an die Firma Guido Kübler GmbH, Bobingen, insbesondere an Torben Mersebach und Werner Sanktjohanser für ihre generelle Hilfsbereitschaft sowie das schnelle und unkomplizierte Bereitstellen einer Powerbox. Daneben sei Ihnen auch herzlich für die intensive Unterstützung bei den Vergleichsexperimenten mit dem Deutschen Krebsforschungszentrum in Heidelberg gedankt.

Großer Dank geht an alle aktuellen und auch ehemaligen Kollegen am Lehrstuhl für Robotik und Telematik sowie am Zentrum für Telematik. Es war mir immer eine besondere Freude mit Euch zusammenzuarbeiten und hat mir viel Spaß gemacht. Auch sind die vielen fachlichen und nicht so fachlichen Diskussionen während den Mittagspausen oder am Abend sehr bereichernd gewesen und gerade der persönliche Austausch hat auch einen wichtigen Beitrag zum Gelingen dieser Arbeit geleistet. Im besonderen danke ich Dr. Florian Zeiger für die Betreuung meiner Diplomarbeit, über die ich überhaupt erst an den Lehrstuhl gekommen bin. Prof. Dr. Lei Ma danke ich für die Einführung in das Forschungsprojekt und dafür, dass er mir immer mit Rat und Tat zur Seite stand. Für die schöne Zeit und das angenehme Arbeitsklima danke ich besonders Stephan Busch, Maximilian Drentschew, Dr. Frauke Driewer, Daniel Eck, Heidi Frankenberger, Andreas Freimann, Matthias Görs, Dr. Martin Hess, Robin Heß, Alexander Kleinschrodt, Markus Krauß, Florian Leutert, Rajesh Shankar Priya, Edith Reuter, Dr. Markus Sauer, Dr. Marco Schmidt und Lothar Stolz. Separat möchte ich mich auch bei allen Kollegen bedanken, mit denen ich in den letzten Jahren ein gemeinsames Büro teilte. Dazu zählen Maria Evita, Dr. Peter Hokayem, Vignesh Rammohan, Lakshminarasimhan Srinivasan und Kaipeng Sun. Ganz besonders möchte ich darunter Dr. Peter Hokayem und Lakshminarasimhan Srinivasan für zahlreiche Diskussionen jedweder Art, viele Gespräche, wichtige Anregungen und einen generellen Meinungs- und Gedankenaustausch danken. Dies hat die Zeit, die ich mit Euch verbringen durfte, sehr bereichert. Außerdem geht großer Dank an Dieter Ziegler für den Bau eines Bewegungsphantoms und die Erstellung verschiedener Platinen, die wichtige Aspekte in meiner Arbeit erst ermöglichten. Auch durfte ich während der Promotionszeit mit vielen Studenten zusammenarbeiten, von denen einige mittlerweile zu Kollegen geworden sind. Repräsentativ sei hier Theresa Eckert, Andreas Freimann, Robin Heß, Matthias Görs, Alexander Kleinschrodt, Oleksandr Kovalchuck, Markus Krug, Johannes Mühr, Thomas Niebler, Benjamin Ruderisch, Johannes Schnappauf, Christian Selbach und Thore Sünert für ihre Beiträge in Forschung und Lehre gedankt.

Der größte Dank gilt meinen Eltern Gertrud und Winfried Herrmann, deren bedingungsloser und uneingeschränkter Unterstützung ich mir schon mein ganzes Leben lang sicher sein konnte. Weiterhin bedanke ich mich bei meiner ganzen Familie und meinen Freunden für den Rückhalt und die Geduld mit mir. Meiner Frau Ute möchte ich meine Dankbarkeit bei der Erstellung von Videos ausdrücken, für die sie sich als atmendes Model zur Verfügung gestellt hat. Von ganzem Herzen danke ich ihr auch, dass sie mir stets Mut zugesprochen und mich in meiner Arbeit bestärkt und immer motiviert hat. Während der Anfertigung der Dissertation musste sie oft einsame Abende und Wochenenden verbringen. Für ihre grenzenlose Geduld, die nicht selbstverständlich ist, möchte ich ihr herzlich danken. Hätte sie mir nicht den Rücken freigehalten, wäre meine Arbeit in dieser Form nicht möglich gewesen. Danke für alles.

In Andenken an

meine Großmutter Margareta

und meinen Onkel Franz

Contents

Chapter 1

Introduction

Radiation therapy today allows precise treatment of static tumors inside the human body due to improvements in treatment procedures over the last 60 years. However, irradiation of moving tumors is still a challenging task as moving tumors often leave the treatment beam and the radiation dose delivered to the tumor reduces simultaneously increasing that in healthy tissue. In order to enable precise treatment of moving tumors, which benefits patients in various ways, this work aims to push the frontiers of radiation therapy by technical innovations to accommodate for that need. The focus of this work is to create a unique real-time system enabling active motion compensation through robotic means for the application of tumor motion compensation.

1.1 Historical Review

On November 8th, 1895, Prof. Dr. Wilhelm Conrad Röntgen made a discovery at the Department of Physics of the University of Würzburg, which was about to have a deep impact on history. In his publication “Über eine neue Art von Strahlen” (“About a new kind of radiation”) [1] he phrased the term x-rays for the newly discovered rays. His research triggered a revolution in medicine and medical technology for which he received the first Nobel Prize in Physics in 1901.

Apart from imaging and diagnostic capabilities of x-rays, their therapeutic effect was soon investigated by several scientists. However, applications were at first mainly limited to superficial treatments of skin diseases. The discovery of radioactivity by Henri Becquerel in 1896 and of radium by Marie Curie in 1898 lead to research about treating cancer and tumors by radiation. Inspired by these findings, Niels Finsen continued his work on the use of ultraviolet light for medical purposes. With further technical advances and the creation of linear accelerators, this ultimately resulted in the first radiation therapy using a linear accelerator in 1953. Apart from particle therapy, radiation therapy today is mainly carried out using photons, i.e. megavoltage x-rays.

Since the 1950’s, treatment delivery has been considerably improved by several innovations [2]. The availability of computed tomography (CT) revolutionized the planning of radiation therapy as tumors could be better visualized and separated from other tissue, thus moving from 2D to 3D planning. The introduction of 3D conformal radiation therapy allowed to shape the treatment beam to fit to the contours of the tumor by employing a multi-leaf collimator (MLC), thus enabling more precise treatments and improved dose distribution while reducing radiation toxicity in the surrounding tissue. Further improvements could be achieved with intensity-modulated radiation therapy where beam intensities are controlled on a fine-granular level, such that different parts of the beam deliver different doses. Together with 3D conformal radiation therapy and varying beam angles for each field, this allows even concave tumors to be irradiated while preserving the surrounding tissue as much as possible. Since treatment planning and delivery involve errors and uncertainties from various sources, image-guided radiation therapy tries to reduce them in order to achieve improved targeting accuracy. This is accomplished by first visualizing the tumor region by an imaging modality such as fluoroscopy or cone-beam CT [3], which is available in the treatment room. The acquired images are then registered to the planning CT and eventual corrections such as offsets can be accounted for just before the beginning of treatment or in between different radiation fields during one fraction.

1.2 Treatment Planning

Normally treatments are delivered during one session per day, which is referred to as a fraction for which a prescribed dose is to be delivered to the tumor according to the outcome of treatment planning. Each fraction may consist of multiple radiation fields for which, for example in intensity-modulated radiotherapy, the gantry (arm holding the linear accelerator) angle is varied in order to change the angle of the treatment beam relative to the patient. The isocenter refers to a fixed point in space of a linear accelerator through which the center of the treatment beam passes irrespective of the chosen gantry angle. The point is usually visualized in treatment rooms with an orthogonal setup of lasers in order to aid in patient setup. The isocenter serves also as a reference point and is defined within the tumor during treatment planning.

Another major step in treatment planning consists of the definition of different volumes according to ICRU report 50 [4] about “Prescribing, recording, and reporting photon beam therapy”. The gross tumor volume (GTV) consists of the tumor volume as seen in the planning images. The GTV is encompassed by the clinical target volume (CTV), which is rather an anatomical concept. It additionally includes possibly malignant tissue which should be irradiated as well to achieve the goal of radiotherapy. The planning target volume (PTV), being a geometrical concept, includes the CTV as well as all errors and uncertainties arising from patient and beam setup, patient movement and organ motion such as respiration, heart beat or fillings of the bladder. Essentially, the PTV is defined in such a way to ensure that the prescribed dose will be delivered to the CTV. Furthermore, the treated volume, containing at least the PTV, is defined using a certain isodose line (line of constant dose) if limitations of the irradiation hardware lead to application of prescribed doses in a larger volume than the PTV. Finally, the irradiated volume consists of all previously defined volumes and additionally contains the volume which will absorb significantly more dose than the tolerance dose of the affected tissue. With advancing technologies such as image-guided radiotherapy, the general definitions from ICRU report 50 were supplemented by ICRU report 62 [5] published in 1999. The concept of the internal target volume (ITV) was introduced, containing the CTV plus an internal margin. The internal margin is thereby chosen to specifically account for variations in size, position and shape of the CTV as a result of physiological processes such as organ motion. The definition of the PTV was altered to include the ITV in addition to setup margins, which occur due to positioning uncertainties from all related hardware devices.

Ultimately, the goal of radiation therapy is to deliver a prescribed dose to the CTV in order to reach a certain treatment goal, usually the destruction of malignant cells. However, in order to avoid side effects any healthy tissue and organs at risk should be simultaneously avoided as much as possible. With previously described improvements, state-of-the-art imaging technologies and also further advanced treatment variations, precise treatment delivery is already achievable for static tumors. With image-guided radiation therapy, even the effects of inter-fraction motions such as loss of weight or different filling levels of the bladder can be eliminated.

1.3 Treatment of Moving Tumors

However, difficulties arise in the treatment of moving tumors where the motion is considerably large during the course of a fraction. Especially lung tumors move regularly with large speeds and amplitudes, where their motion is mainly induced by respiration and heart beat. Also other tumors such as prostate or liver tumors are known to move intra-fractionally due to bowel movements or changing of filling levels of bladder or stomach [6]. Immobilization devices can help to reduce motions, however certain motion amplitudes will always remain, especially for lung tumors.

To illustrate the consequences of moving lung tumors on the treatment volumes, Figure 1.1 shows the CTV of a lung tumor as the red contour when the patient had completely inhaled. The image depicts the tumor inside the lung (large dark area) on three orthogonal slices, which were gained by reconstructing CT scans according to different respiratory phases [7]. Similarly, Figure 1.2 visualizes the CTV as the yellow contour when the patient had completely exhaled. As a reference, the contour of the other extreme location of the

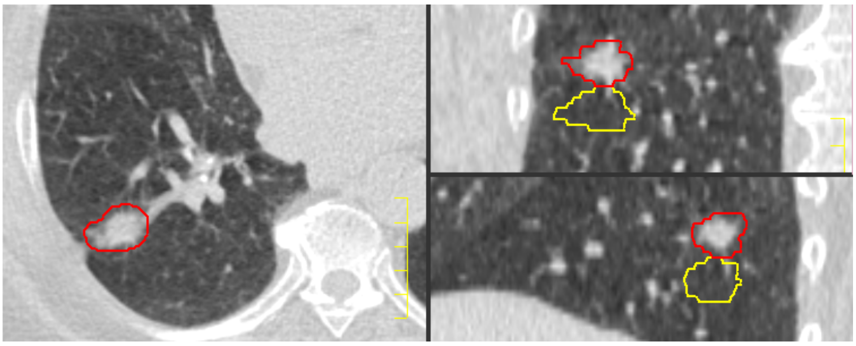


Figure 1.1: *CTV of lung tumor on three orthogonal slices in extreme position when patient had completely inhaled*

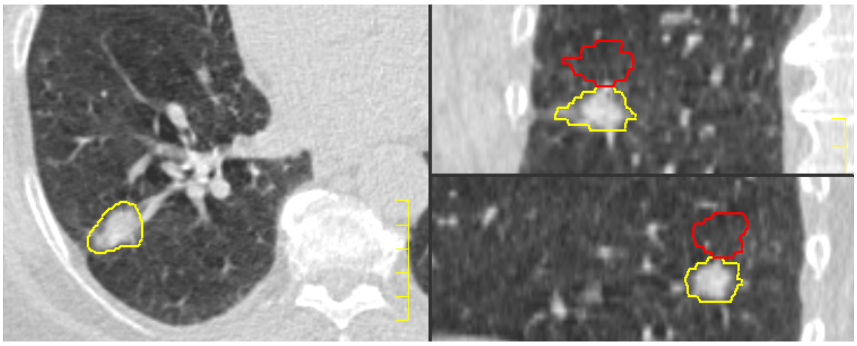


Figure 1.2: *CTV of lung tumor in extreme position when patient had completely exhaled*

CTV is additionally shown in both figures. Generally, the goal of radiation therapy in this example would ideally be to just irradiate the CTV and deliver a prescribed dose to this volume. If only that volume is treated without motion management, the actual dose delivered to the CTV would be significantly less than the prescribed dose because the beam only partially covers the CTV. In order to deal with this issue, the CTVs from different respiratory phases are combined to form the ITV, so that it includes the uncertainties due to organ motion according to its definition. Furthermore, the ITV is augmented by the security margin. In the example 5mm were added around the ITV, which finally forms the PTV shown in Figure 1.3. As can be seen, the PTV is considerably larger than the actual CTV, resulting in the irradiation of a large volume of healthy tissue. Taking into account all relevant physiological and technical factors including the choice of treatment type, beam form and field sizes, the CT-based planning results in a dose distribution, indicating the dose to be delivered to the patient's tissue. This is exemplified in Figure 1.4, showing the calculated isodose lines (lines of equal dose). The brown and blue isodoses in the form of tubes result from different treatment fields where the beam angle is varied. From this example, it can also be observed that the volume receiving a considerable amount of radiation is relatively large compared to the CTV and PTV, where the PTV is enclosed by the 37.5Gy isodose (orange).

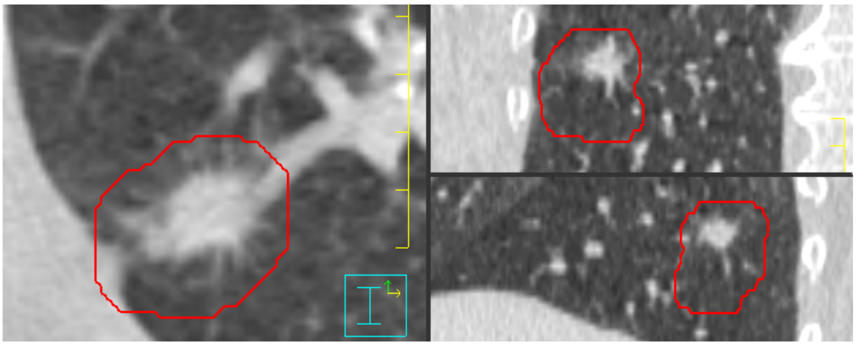


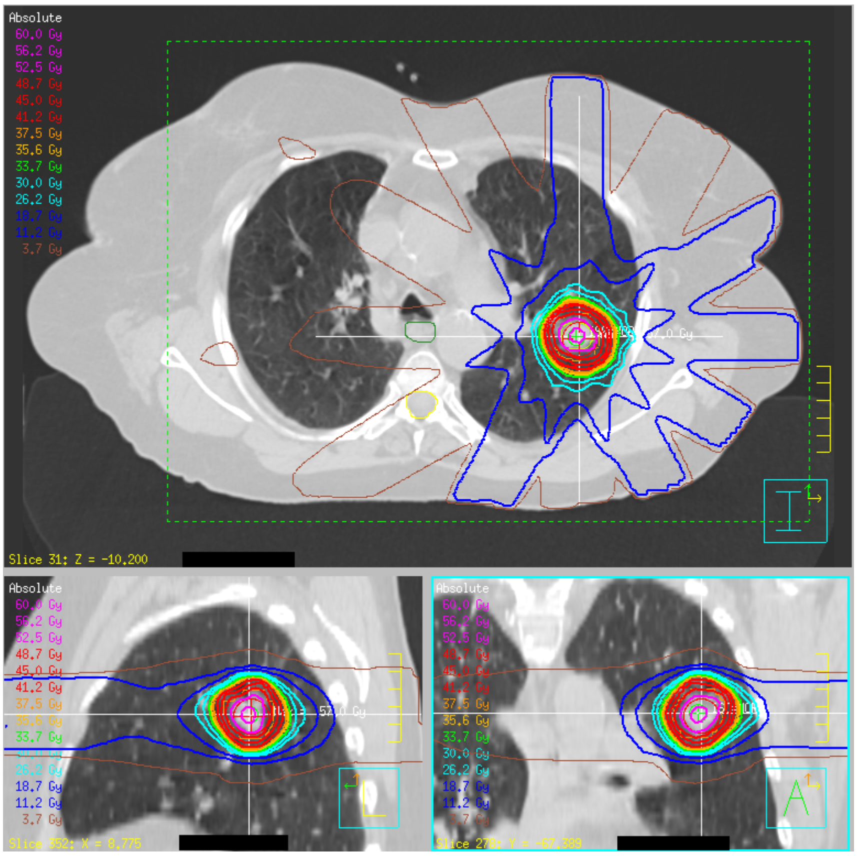
Figure 1.3: *PTV of lung tumor combined*

Due to the fact that healthy tissue is also irradiated, the prescribed dose is usually limited by tolerance dose of healthy tissue or organs at risk in order to reduce side effects and tissue toxicity. However, clinical studies suggest that improved tumor control can be achieved with higher treatment doses [8, 9]. These can only be applied if margins can be reduced, ideally limited to the CTV.

1.4 Approaches to Tumor Motion Compensation

Several approaches have been developed up to now in order to improve treatment delivery to moving tumors. They can be roughly distinguished into two groups. The first group summarizes procedures and systems which focus on the minimization of the impact of tumor motion implying a motion management strategy [10], whereas the second group deals with tumor motion compensation systems.

Respiration gating [11–15] and active or voluntary breathing control [16–19] belong to the group of tumor motion management systems. For respiratory gating, radiotherapy takes place only in certain breathing phases when the tumor passes a small window around



[redacted]
[redacted]
[redacted]

Figure 1.4: *Exemplified planning CT showing isodose lines and corresponding doses*

the center of the beam. Consequently this prolongs radiation treatment due to the fact that the beam is switched off for a considerable amount of time during a breathing cycle. Using active breathing control the patient's respiration is constrained through external devices, therefore it is not acceptable by many patients. Voluntary breath-hold requires the cooperation of patients which is not always feasible in such a way that patients are sometimes not capable of controlling their breathing to the required extend despite trying hard.

A compensation system which actively tries to keep the tumor in the isocenter of the radiation beam can be laid out in different ways. One method uses a multi-leaf collimator to adapt the beam parameters so that the beam is directed and displaced according to the tumor motion. MLC-based tumor tracking is still in the research phase and pursued by several groups worldwide [20–29]. Major issues in dynamic MLC tracking involve different tracking performances when the tumor is either moving into or perpendicular to the leafs travel direction and limitations due to leaf width and leakage effects of radiation [30].

Different from this strategy, robotic motion compensation monitors tumor motion during treatment and compensates for it in real-time using robotic motion. The most prominent example of this type of compensation system is the CyberKnife® system (Accuray, Inc., Sunnyvale, CA, USA) [31–39], which is currently the only robotic compensation system employed in clinical treatments so far. Within CyberKnife, a small linear accelerator is mounted onto an industrial manipulator (KUKA Roboter GmbH, Augsburg, Germany).

Tumor position information is usually acquired by continuous x-rays and tracking of the tumor (either with or without implanted markers). The manipulator is moved according to the tumor position in order to compensate for that motion.

A similar system is the VERO system from BrainLAB (BrainLAB AG, Feldkirchen, Germany) and MHI (Mitsubishi Heavy Industries, Tokyo, Japan) [40–45]. A small linear accelerator is mounted on a O-ring gantry with two orthogonal gimbals. The gimbals allow pan and tilt motion of the linear accelerator by which compensation of tumor motion is enabled due to their distance to the isocenter.

Unlike these systems, it is also possible to move the patient using a robotic treatment couch [46–48]. In order to achieve motion compensation, the treatment couch is directed into the opposite tumor motion direction. While [46, 49] uses a standard Elekta table stand as treatment couch by accessing the hardware of the joystick interface directly, [47, 50] and our group [48, 51] make use of the HexaPOD treatment couch. Further survey of these systems is given in [52].

1.5 Innovations

This work emerged from the research project “Intra-fractional tumor tracking and compensation using an automated treatment couch”, which was carried out in collaboration with the Department of Radiation Oncology of the University of Würzburg as clinical partner and industrial participation by Medical Intelligence GmbH, Schwabmünchen, Germany (now belonging to Elekta AB, Stockholm, Sweden). The project was supported by a grant from the Bavarian Research Foundation.

The main goal of the project was to develop an adaptive tumor tracking system (ATTS) to enable real-time motion compensation of intra-fractional tumor motion. It seeks to bring a number of innovations to the field of radiation therapy:

Reduction of Safety Margins and Treatment Time

By realizing real-time motion compensation, reduction of safety margins becomes possible, such that irradiated volumes can be further reduced compared to conventional techniques and healthy tissue can be spared as much as possible. Simultaneously, reduction of the margins allow more efficient adaptation of radiation doses, so that with higher radiation doses improved local tumor control and therapeutic ratios can be achieved.

By applying higher radiation doses, the desired dose distribution within the tumor can be reached earlier and treatment time can be reduced. Also, treatment time compared to other approaches such as gating is generally smaller. Thus, treatment becomes more cost-efficient and higher daily patient throughput can be achieved on the linear accelerator.

Non-invasive Therapy

A further goal of the research project was the usage of the treatment beam to derive the tumor position, allowing for completely non-invasive therapy. Compared to other methods which use markers implanted within the tumor followed by x-rays, no implementation of markers or additional radiation exposure is necessary. In addition to cost benefits by omitting implementation costs, little or no investment on more hardware and lower operating costs, better tolerance by patients can also be achieved. After all, in some cases, the position of a tumor is in-accessible to implement a marker.

Inherent Compensation of Baseline-Shifts

Physicians often require the compensation of baseline-shifts, i.e. the compensation of slow changes, which are inherently built-in into the ATTS. Compensation of fast changes (e.g. respiratory induced motion) includes the compensation of slow motions. Further applications of the ATTS are also possible, e.g. radiation of prostate cancer, where changes in the filling level of the bladder lead to slow shifts of the target volume which are already becoming noticeable during the course of a fraction.

Flexible and Cost-effective Treatment System

The presented approach makes use of only medically certified hardware, most of it being already standard in state-of-the-art treatment rooms. Since no modifications are introduced in any hardware device, a flexible system results which can be easily integrated into existing treatment rooms. Furthermore, the adoption of the ATTS imposes no constraints on existing treatment procedures of any kind in terms of compatibility. Also any type of radiation therapy remains possible irrespective of the employed technology such as photon and electron beam therapy or particle therapy. This stands in contrast to other approaches such as CyberKnife or VERO where the linear accelerators need to be compatible with the overall system. For these systems, the main constraint is the mass and size of the accelerator, especially within CyberKnife. This results in the limitation that only the treatment types supported by these systems can be used. Specifically, since an MLC is a heavy construction it is not supported by CyberKnife. However, with the ATTS, flexibility is even improved by adding the option of tumor motion compensation to an existing system.

Since no specialized hardware needs to be additionally developed for the ATTS, the costs when updating an existing treatment room are expected to be minor compared to fully integrated systems such as CyberKnife or VERO. More than 2500 linear accelerators are located in the USA and about 400 are used in Germany, thus making a certain economic potential of this approach apparent.

In order to realize these innovations and extended treatment possibilities, several technical innovations are necessary, which are the focus of this work. Simultaneously, these contributions form the outline of the work:

Real-Time Tumor Motion Compensation System

For achieving real-time tumor motion compensation, a properly designed real-time system is required, which is fully described in Chapter 2. Real-time aspects are thereby included in several ways. First of all, a real-time operating system is employed to achieve certain guarantees with respect to maximum scheduling latencies and uninterrupted execution of application code in order to adopt features not available in general purpose operating systems. More importantly, all data referring to physical quantities such as sensor measurements of respiratory or tumor position is associated with timestamps using a single clock source. For the adopted measurement devices, measurement latencies are determined with which timestamps are corrected to achieve best possible timing accuracy. Generally, the true time of all sensor data and generated data is taken into account throughout the whole system consistently. Furthermore, all sensor latencies as well as all software-induced latencies occurring during each calculation cycle are adaptively compensated for.

Chapter 2 further highlights the complete hardware and software setup of the ATTS, detailing all employed devices for this work as well as the most important aspects of the developed software. Including discussions about patient safety features, this chapter lays the foundation for the rest of the work.

Breathing and Tumor Motion Prediction

As with any mechanical system, the HexaPOD which actually enables motion compensation possesses certain dynamics including latencies. In order to deal with these effects, knowledge about tumor motion in the near future is required. Approaching the problem of motion prediction, an in-depth analysis of breathing and associated lung tumor motion as well as their correlation is first performed in Chapter 3 to acquire their general characteristics. This enables design and investigation of breathing and tumor motion prediction strategies. Tumor position determination based on the treatment beam as it is pursued in this work, imposes hard constraints on the quality of tumor position information. Several methods are developed to perform predictions even under these relatively harsh conditions.

Control Schemes for the HexaPOD

With the knowledge how the tumor will further move in the near future, a reference trajectory can be generated, specifying to which locations the HexaPOD needs to move at which time in order to realize motion compensation. To make the HexaPOD follow the reference trajectory, control schemes need to be employed ensuring that the HexaPOD maintains the reference trajectory as close as possible by applying appropriate inputs to the HexaPOD. The way how the HexaPOD responds to these inputs is a question of the dynamics. Hence, to learn more about the dynamics, dynamic models are presented and evaluated in the beginning of Chapter 4. Based on these models, a first controller is derived by optimization techniques. With an assumption about the HexaPOD, a second controller is proposed and its performance evaluated in hardware-in-the-loop tests. Combining both controllers to a third one to eliminate arising problems in some scenarios with the other ones, it is shown that tumor motion compensation is ultimately possible with the HexaPOD, not only by achieving sufficient tracking accuracy but also with decent patient comfort which is a major concern in this work. Patient comfort is crucial for a compensation system which affects patient's position as any harsh motion exhibited by the HexaPOD can have an impact on patient's breathing patterns. Irregularities might arise in these which can in turn challenge the compensation system. In order to avoid human-in-the-loop feedback effects, it is important for the compensation system to make the HexaPOD move on a smooth trajectory.

Finally, Chapter 5 concludes with a summary of the work and addresses future directions.

Chapter 2

Motion Compensation System Design

For performing robotic motion compensation, a technical system needs to be developed which is able to realize motion compensation in real-time. The main objective of the robotic system is to incorporate existing standard hardware devices already present in treatment rooms to allow flexible treatment possibilities and simple integration into existing treatment rooms. To approach this problem, several types of motion compensation modes are identified which are to be supported by the compensation system. The hardware and software designs of this system are explained in detail and insights on basic software features are given. The system will provide the general environment to allow the development of tumor prediction methods and the derivation of control methods for the treatment couch. Special emphasis is put on real-time aspects to allow true real-time operation of the whole system. Patient safety features of the compensation system are further discussed in order to enable a safe treatment.

2.1 Modes of Motion Compensation

Depending on the medical requirements of tumor motion compensation, the question that has to be answered for each patient and treatment is what mode of motion compensation should be applied. The distinct feature of each mode mainly affects the shape of the reference trajectory which defines the desired trajectory of the tumor leading to the requested compensation. Three different modes, their possible applications and technical requirements are identified and discussed in the following subsections, which cover a broad range of tumor motion compensation scenarios.

2.1.1 Recent Displacement Compensation

The recent displacement compensation (RDC) mode is the simplest mode. It uses the last known displacement from the tumor to its desired position in order to determine the control input for the HexaPOD to compensate for this displacement. So the reference trajectory boils down to a single value or a set-point which can be regarded as desired value in the current sampling instant. Of course, this mode cannot compensate for any latencies such as processing and transmission delays in the software and hardware as well as the measurement delay (occurring during the determination of the tumor displacement itself). But the advantage is that no predictions of future tumor positions are necessary for the generation of the reference trajectory. This type of motion compensation can be used, when an approximate tracking of tumor motion is sufficient and either no prediction is possible yet or no prediction is possible at all. The former case usually happens during the initialization of a prediction algorithm, e.g. the minimum number of samples have not been acquired yet or the learning phase is not finished yet. The latter case occurs if the tumor motion itself is unpredictable, for example with prostate motions or if the motion does not contain any repeatable pattern whose dynamics can be captured by a prediction model. In this mode, the compensation system tries to follow all measured displacements as fast as possible, even if a larger displacement persists only during a short time span.

2.1.2 Drift Motion Compensation

The drift motion compensation (DMC) mode can either be based solely on past values or in combination with predictions of tumor motion. Since drift is generally a slowly changing quantity, using past values in the calculation is usually sufficient. In any case, some method is used here to determine the drift or the baseline shift which is a displacement from the desired position. Depending on the shape of the tumor motion signal, this task can be accomplished by sliding window filtering or other filtering mechanisms which calculate a mean position of the motion. Similar to RDC, the current moving average is then used in the calculation of the reference trajectory which also becomes a desired value or a set-point. Drift compensation can be used when only the drift or the baseline shift of the tumor motion should be compensated. Compared to RDC, larger temporary displacements are damped due to filtering and compensation of only the long term trend of the signal. Possible applications of this mode include unpredictable tumor motions within prostates as well as quasi-periodic motions of lung tumors.

2.1.3 Full Motion Compensation

The full motion compensation (FMC) mode is the most advanced but also the most natural mode when one thinks of motion compensation. The goal here is to compensate for any displacement that occurs between current and desired positions of the tumor. The actuators of any motion compensation system possess a dynamic behavior including certain time delays of different orders between the input and the output of the system. Additionally there are usually other time delays in the motion compensation system such as those induced by measurement, transmission and processing. Because of all these delays, a full motion compensation is only feasible if the future behavior of the tumor motion is known up to a certain horizon whose minimum value is characterized by the sum of the delays. Thus, a prediction algorithm has to be employed for this task. But for this to work, the tumor motion has to be predictable which is the case, for example, with quasi-periodically moving lung tumors. From the predicted tumor motion, a reference trajectory can then be generated which should be followed by the actuators of the motion compensation system.

Basically, the RDC and FMC modes differ mainly in the data used for the generation of the reference trajectory, but have similar goals. Thus, FMC mode can be ideally combined with RDC mode during the beginning of a treatment as long as no predictions are possible.

2.2 Hardware and Software Setup

In order to realize the motion compensation modes mentioned in the previous section, a motion compensation system has been built. The adaptive tumor tracking and motion compensation system called ATTS only consists of standard hardware equipment and in-house made software packages which were developed at the Department of Radiation Oncology at the University Hospital in Würzburg and at the Department of Robotics and

Telematics at the University of Würzburg. A schematic overview depicting the information flow inside the system is sketched in Figure 2.1.

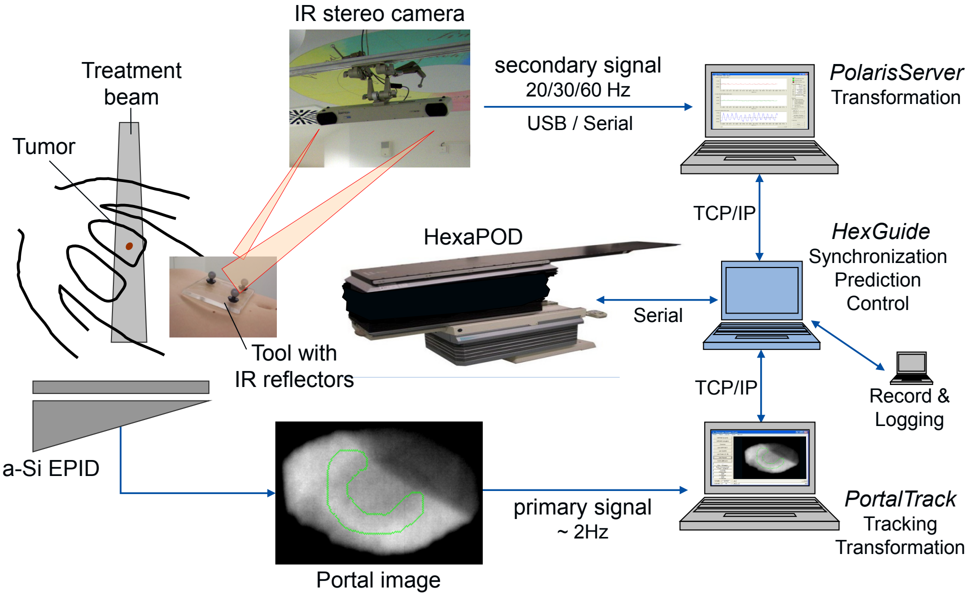


Figure 2.1: *Schematic overview over the complete tumor tracking and compensation system*

The main working principle is described in the following: Without compensation the tumor moves relative to the fixed treatment beam which is emitted by an Elekta Synergy® S linear accelerator with a multi-leaf collimator (the leaves are used to define the shape of the treatment beam). An image of the tumor, denoted as portal image, is captured by an electronic portal imaging device (EPID) using the treatment beam for visualization. The in-house made software package *PortalTrack* (developed by the Department of Radiation Oncology) tracks the tumor in subsequent images and delivers a 2D projection of the tumor position orthogonal to the treatment beam. Depending on the gantry angle (the gantry, carrying the collimator of the linear accelerator, can be rotated in order to irradiate tumors from different directions), a transformation of the tumor position takes place into a room fixed reference frame. Since this position information represents the true tumor position, it is referred to as the primary signal. The secondary signal in the system is measured by a stereo infrared camera which delivers the breathing motion of a patient. Using the in-house made software package *PolarisServer*, infrared tools carrying a number of infrared markers are localized by the stereo camera and the acquired coordinates of tools which are placed on the patient's abdomen and the HexaPOD are transformed into the room fixed reference frame.

Both the primary and the secondary signals are sent to the in-house made software package *HexGuide* where the intelligence of the system is concentrated. Based on these signals, predictions of future tumor positions are derived. Depending on the chosen motion compensation mode, a reference trajectory for the HexaPOD is generated. Using a control scheme with the reference trajectory as an input, the HexaPOD should follow this trajectory as close as possible in order to realize the mode-dependent goal of motion compensation. *HexGuide* also takes care of time synchronization between all software components and correct timing of incoming data. Incoming and generated data (e.g.

predictions) are stored in files for further analysis, evaluation or recording duties.

2.2.1 Hardware and Software Components

The following subsections explain each hardware and software component individually. The complete system described in the following was setup at the Department of Radiation Oncology at the University Hospital in Würzburg and is depicted in Figure 2.2. A subset of the components was installed in parallel at the Department of Robotics and Telematics at the University of Würzburg to serve as testbed for research and development of tumor prediction methods and control schemes of the HexaPOD which is the focus of this work.

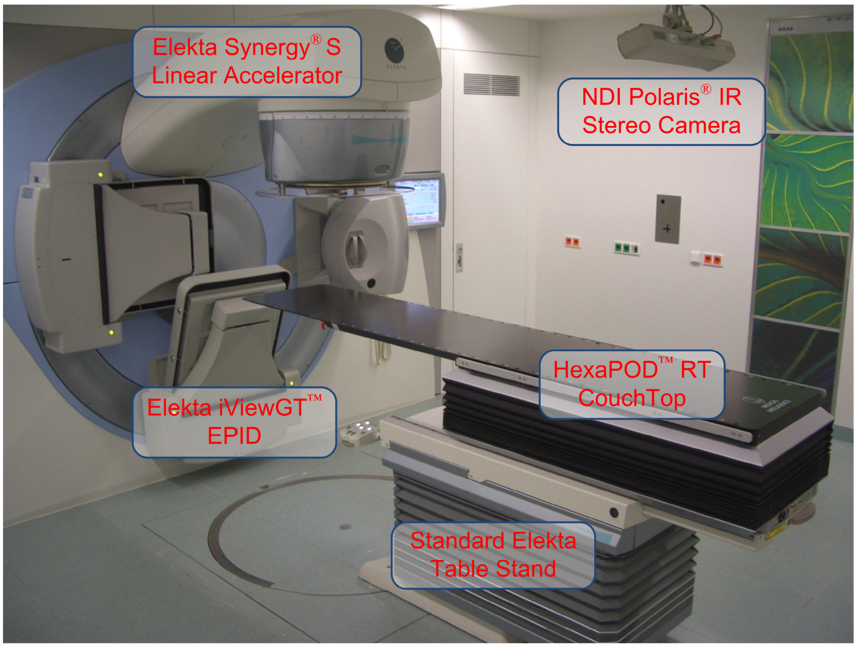


Figure 2.2: *Clinical setup of treatment room*

2.2.1.1 HexaPOD and HexaPOD evo

One of the main components of the tumor tracking system is the robotic treatment couch HexaPOD^{TM} RT CouchTop originally from Medical Intelligence GmbH, Schwabmünchen, Germany (now belonging to Elekta AB, Stockholm, Sweden), on which the patient lies during treatment. In normal clinical routine the HexaPOD is only used for correction of patient setup errors [53] and was initially not intended for dynamic and sustained motion during radiotherapy. Nevertheless, feasibility of using a HexaPOD for tumor motion compensation has been shown by preliminary studies [47, 54]. The original and clinically certified version of the HexaPOD showed a slow acceleration behavior which accounts for patient comfort during patient setup when the patient's tumor is moved into the isocenter either manually or automatically after the cone-beam CT. A special research firmware was installed along with the original firmware at the Department of Radiation Oncology by Medical Intelligence GmbH. This allowed both the certified clinical mode and a research mode where the ramps in the acceleration profile were reduced so that the HexaPOD could exhibit a better dynamic behavior, but still not exceeding the maximum speed of 8mm/s.

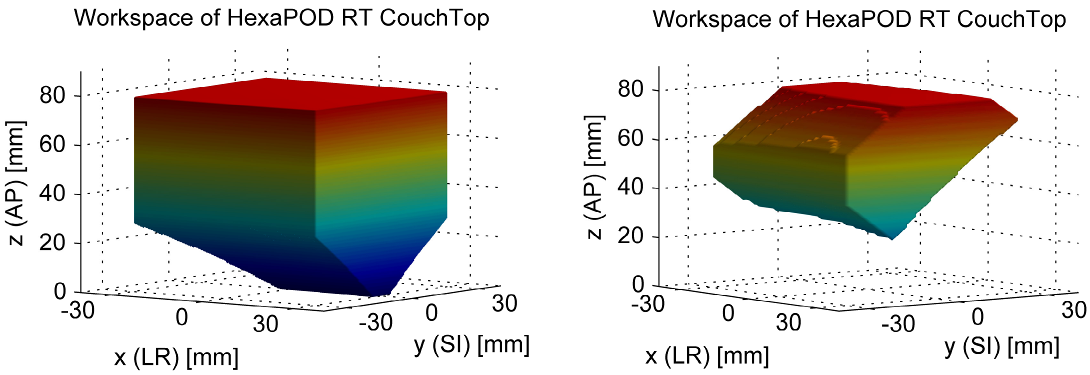
The HexaPOD is a parallel manipulator, alternatively called Stewart platform [55] with six degrees of freedom (three translational and three rotational). Six independently actuated prismatic legs determine position and orientation of an end-effector mounted on top of the legs. With this construction, the HexaPOD can perform highly accurate motion (positioning accuracy of 0.1mm) within a small range fulfilling the requirements of tumor motion compensation [56]. Besides, it also possesses high stiffness, low inertia and large payload capacity of 185kg. Actuation of the individual legs including inverse kinematics and speed control is realized by a built-in controller which is delivered by the manufacturer and cannot be affected directly.

The HexaPOD itself is mounted onto a standard Elekta table stand (cf. Figure 2.2) which can be rotated around a vertical axis (w.r.t. the room) and which can be moved along translational axes to account for required position displacements during patient setup which would be outside the workspace of the HexaPOD in its default position.

The HexaPOD is used in this setup to counteract tumor motion in order to keep the tumor fixed w.r.t. the beams-eye-view of the accelerator. For achieving tumor motion compensation within the ATTS, the treatment couch HexaPOD has to be controlled to appropriately adapt the tumor position during treatment delivery. Since the built-in controller is not directly accessible, the only way to control the HexaPOD is to use a predefined communication protocol which accepts positioning commands through a serial interface. Each positioning command “MOV” consists of a 6D position $(c_3, c_2, c_1, r_1, -r_2, -r_3)$ in the operational range for positions $c_j \in \mathbb{R}$ with $j = 1, 2, 3$ (corresponding to x-, y- and z-axis, respectively) of ±30mm, ±30mm and 0-80mm along the x-, y- and z-axis, respectively, and rotations $r_j \in \mathbb{R}$ with $j = 1, 2, 3$ of $\pm3°$ around all axes and of a normalized maximum speed $s_c \in \mathbb{R}$ in the range 0-1. The individual axes of the HexaPOD are aligned with the human body when a patient is lying on it. The x-axis denotes the lateral axis of a patient, i.e. left-right (LR). The y-axis is aligned longitudinally in the human body and describes the superior-inferior (SI)-direction ranging from feet to head. The z-axis complements the reference frame and is aligned in back to front direction, alternatively called anterior-posterior (AP)-direction.

Not all points within the operational range are accepted as target positions by the HexaPOD. Especially for c_3 along AP-direction between 0mm and 31mm, the positions along the other axes are constrained leading to a reduced workspace as shown in Figure 2.3(a). This low position range in height can be considered as an entry region, so that a patient can more easily climb onto the table. Because of this reduction, a new reference frame is defined in the developed software where the zero position of the HexaPOD is located at 55.5mm in AP- direction. Then the treatment range in AP-direction becomes ±24.5mm with maximized LR- and SI-range when the patient’s tumor has been placed in the isocenter and the HexaPOD is located at position $(55.5, 0, 0, 0, 0, 0)$. If orientation angles different from zero are used, then the workspace reduces depending on these angles which have to be considered in the developed software. Figure 2.3(b) visualizes the reduced workspace in one of the worst cases if all orientation angles are 3°. In this work only translational motion corrections with static orientation angles are used. So it is assumed that for each orientation angle the corresponding workspace limits are known.

During the course of this work the HexaPOD was exchanged at the Department of Radiation Oncology with a new version called “HexaPOD$^{\text{TM}}$ evo RT”. The requirements for the new version were: same control interface as the original HexaPOD, flatter design so that the HexaPOD occupies less space in height, larger operational range and support



(a) orientation angles set to zero

(b) orientation angles set to 3° (maximum)

Figure 2.3: *Workspace of the HexaPOD RT CouchTop in different orientation settings*

of higher loads. Additionally it was built with dynamics in mind which allows for faster reactions with double the speed of 16mm/s. The software interface specification of the HexaPOD evo highlights a dedicated entry channel from 0-40mm in AP-direction. Positions along all other axes, including orientations, have to be set to zero. This restricts the treatment range from 40mm to 100mm, effectively covering the range of ±30mm when the zero position is defined at 70mm along AP-direction at position $(70, 0, 0, 0, 0, 0)$. Table 2.1 provides a comparison of the main specifications of both HexaPOD types. Some parts of this work were performed using the original HexaPOD whereas other parts were done using the HexaPOD evo. Throughout this work, it is pointed out whenever the HexaPOD evo was used; otherwise the original HexaPOD is implied.

Type	Operational range		Max. speed	Max. load
HexaPOD	lateral (LR)	±30mm	8mm/s	185kg
	longitudinal (SI)	±30mm		
	vertical (AP)	0-80mm		
	treatment range in AP	±24.5mm		
	yaw, pitch, roll	±3°		
HexaPOD evo	lateral (LR)	±30mm	16mm/s	250kg
	longitudinal (SI)	±30mm		
	vertical (AP)	0-100mm		
	treatment range in AP	±30mm		
	yaw, pitch, roll	±3°		

Table 2.1: *Specifications of the HexaPOD and HexaPOD evo treatment couch*

2.2.1.2 Electronic Portal Imaging Device – EPID

Historically, x-ray sensitive film cassettes were employed for radiotherapy treatments to acquire portal images of irradiated regions with a mega-voltage treatment beam, also called mega-voltage imaging. Localization films were used at the beginning of each treatment for detection of patient setup errors using a low radiation dose, whereas with verification films the dose delivery during treatment was recorded. With the disadvantage that films had to be developed, which usually requires several minutes, it is a pure offline technology. For localization films the patient might move again while film development and information gained from it might become invalid. For verification imaging, the technology of film cassettes prohibits the assessment of targeting accuracy during treatment. This motivated the research on electronic portal imaging devices which already began in the late 1950's. First commercially available products using different imaging technologies emerged in the 1980's and have evolved to useful systems with adequate imaging capabilities allowing imaging in digitally available form, also in real-time [57, 58]. Apart from real-time imaging, this also opened new application possibilities, such as simplified quality assurance of radiation beams, geometric verification of field sizes and MLC positioning as well as tumor tracking in real-time [59]. With these increased possibilities in conjunction with an improved cost situation [60] as opposed to films, EPID has become widely accepted in radiation systems. Research in applying this technology is ongoing, e.g. in-vivo measurements of dose distributions [61] which is a main concern in radiotherapy.

The EPID installed at the Department of Radiation Oncology in Würzburg is an Elekta iViewGT$^{\text{TM}}$ consisting of an amorphous silicon (a-Si) flat panel detector which can be unfolded and rotated so that the panel is oriented perpendicular to the radiation beam in order to capture the exiting radiation which passes through the patient. This enables imaging of the irradiated region with the tumor inside as a 2D projection onto the plane of the panel. With knowledge of the gantry angle, this projection can be transformed into the room-fixed reference frame.

The research group at the Department of Radiation Oncology developed the software package *PortalTrack* [48] which reads every available portal image from the hardware and processes it to locate the tumor inside the image and track it in subsequent images. Previous research [62] demonstrated the feasibility of tracking moving objects with EPID by comparing different algorithms for finding an object in portal images. In the beginning of a treatment, a single shot image of the irradiated region is taken with *PortalTrack* using a lower radiation dose than during treatment delivery. Then a physician creates a mask by drawing an arbitrary formed shape into the portal image which should contain a certain area outside the tumor as well as an area inside the tumor. The masked area, shown as the green contour in the portal image in Figure 2.4, is then used for locating the relative pixel distance of the tumor mask in subsequent portal images. With knowledge of the pixel width, in case of the Elekta iViewGT$^{\text{TM}}$ it is 0.251mm, the relative pixel distance in relation to the reference image (with which the mask was defined) is then converted into length units. Applying the coordinate transformation matrix resulting from the gantry angle, the 2D projection of the tumor position onto the image plane is finally transformed into the room-fixed reference frame. As soon as this information becomes available, it is sent via transmission control protocol / internet protocol (TCP/IP) to the software package *HexGuide*.

However, due to hardware limitations, this version of the EPID only delivers a sampling rate of about 2Hz. Using this information alone for motion compensation of lung tumors is not enough for decent compensation, although the maximum moving frequency of lung tumors is with about 0.5Hz lower than the Nyquist frequency of 1Hz. But the moving frequency is close to the Nyquist frequency and sampling with EPID is not very equidistant resulting in temporarily lower effective sampling rates. Furthermore, a much higher

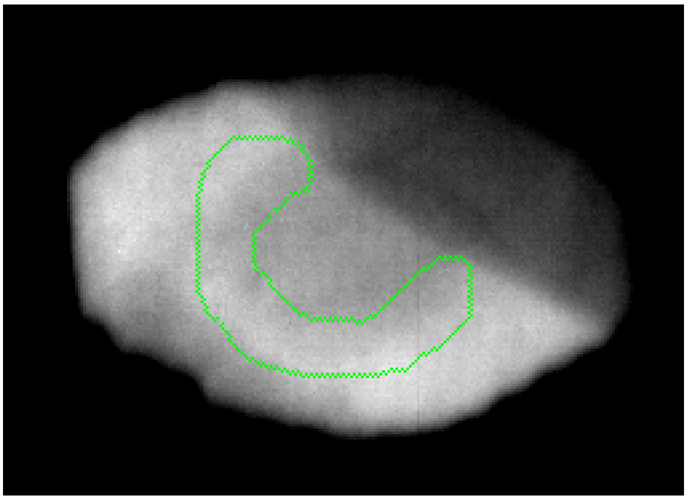


Figure 2.4: *Image with tumor acquired from EPID including mask for tumor tracking with* PortalTrack

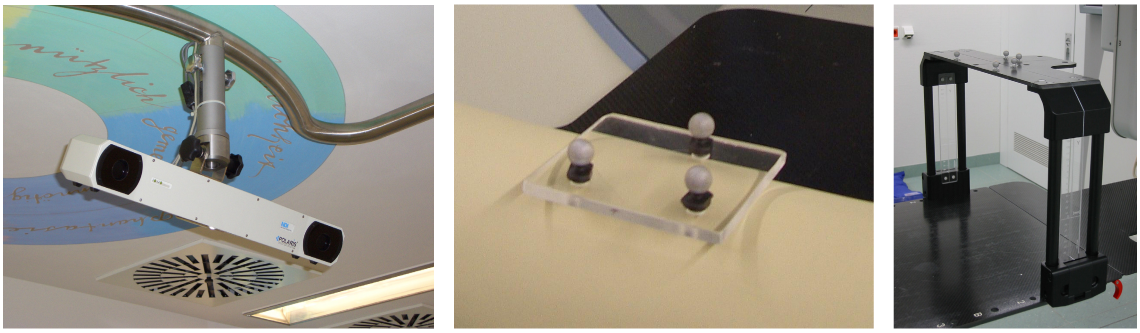
sampling rate is needed to provide good predictions of tumor motion, which is an essential part of the motion compensation system.

2.2.1.3 Infrared Positioning System

To overcome the limited sampling rate of the EPID in motion determination, an infrared (IR) positioning system with higher data rate is introduced to measure respiratory motion. Such a system is also standard equipment in treatment rooms. It usually aids in patient setup where a rough patient positioning is accomplished by placing a tool with IR-sensitive markers at a predefined position on the patient body and moving the patient with the standard table stand and the HexaPOD to the same coordinates of the IR tool which were determined during treatment planning.

For this work the positioning system NDI Polaris® depicted in Figure 2.5(a) was employed which is mounted on the ceiling in the treatment room. It is manufactured and distributed by Northern Digital Inc. located in Waterloo, Ontario, Canada. The positioning system is equipped with two arrays of IR light-emitting diodes that illuminate the surrounding environment and two CCD (charge coupled device) cameras which collect the reflected infrared light. Infrared markers (coated plastic spheres of a diameter of 1cm), which possess a high reflectance of infrared light, are located in the acquired images from both cameras and their 3D positions are determined by triangulation methods relative to a camera-specific reference frame. The geometric properties of the IR tools, i.e. the distances between each of the markers located on one tool, are uploaded to the system along with information of a local reference frame defined w.r.t. the tool. With this information the positioning system can robustly determine 3D positions (origin of the local reference frame) and 3D orientations of the local reference frame relative to the camera-specific reference frame. The system delivers transformations of up to 9 passive tools inside a pyramid-shaped measurement volume which ranges from a perpendicular distance of 0.95m up to about 2.4m.

The Polaris system can be connected to a PC via RS-232 or RS-485 with a maximum serial speed of 115200bps. During the course of this work, the Polaris system was replaced



(a) NDI Polaris stereo infrared camera mounted on the ceiling in the treatment room (b) IR tool for placement on patient's abdomen (c) C-shaped bridge with IR markers

Figure 2.5: *NDI Polaris IR positioning system and IR tools used for tracking the patient's abdomen and the HexaPOD*

by its successor NDI Polaris® Spectra® at the Department of Radiation Oncology. In contrast to the previous version, it is connected via a universal serial bus (USB) to serial interface, permitting higher serial transmission speeds of up to 1.2Mbps. Furthermore, the tracking accuracy was improved to less than 0.3mm RMS (according to the datasheet) and it comes with the option of an extended measurement volume ranging to a maximum distance of 3m.

The sampling rate of the system can be set to 20Hz, 30Hz or 60Hz. The sampling instants are distributed very equidistant with a low jitter. Using the maximum update rate of 60Hz (corresponding to a sampling period of 16.667ms) with the Polaris Spectra, the standard deviation of the sampling period was determined with a non-real-time operating system as 0.475ms. In tests with static IR tools, it was observed that the measurement noise of is approximately normally distributed with small standard deviations of less than 0.02mm along each axis and maximum deviations from the mean of about ±0.06mm.

Two IR tools are needed for this work: one is placed on the patient's abdomen (cf. Figure 2.5(b)) to get its position and the other is placed on the HexaPOD. The latter is a C-shaped bridge (cf. Figure 2.5(c)) under which the patient lies and which is spanned over the LR-direction of the HexaPOD. Since the HexaPOD is intended to move together with the patient during the course of treatment, the breathing motion of the patient is calculated by the difference between these tool positions.

2.2.1.4 Software Package *PolarisServer*

The software package *PolarisServer*, developed at the Department of Robotics and Telematics, interfaces directly with the Polaris positioning system through a serial port interface. It implements the full communication protocol in order to fulfill the following tasks:

- Initialization of the positioning system
    - Update rate setting (20Hz, 30Hz, 60Hz)
    - Upload of ROM-files containing tool definitions and their geometric properties
    - Activation of port handles to enable used tools
- Periodic requests of tool transformation with the selected update rate (using polling)

- Monitoring and adaptation of the timing to prevent over- and undersampling
- Calibration of a transformation between camera-specific reference frame and room-fixed reference frame
- Sending of 6D tool position information to *HexGuide* through a TCP/IP socket
- Reading diagnostic information (e.g. temperature too low)

The positioning system only supports a polling mode to retrieve tool transformations. In order to get these, a request has to be sent to the positioning system through the serial interface. Then the positioning system delivers the last acquired tool transformations at the next internal sampling interval. But the internal sampling interval is independent of the used sampling rate and fixed to 60Hz. This leads to the behavior that depending on the time of request in relation to the internal sampling intervals, the corresponding answer arrives within a response time of 16.667ms (1/60Hz), regardless of the selected update rate (20Hz, 30Hz or 60Hz). In the case of 20Hz or 30Hz, subsequent polls without any wait states in between would lead to an “oversampling” where the same positions would be returned two or three times, respectively. In the case of 60Hz, occasionally, a sampling interval would be missed leading to an “undersampling”. Due to small latencies (such as serial transmission time and processing delays) response times change and a request to get the information of the corresponding sampling interval is sometimes sent too late. Because of this behavior, the timing of the request has to be managed and monitored. Depending on the selected update rate, the request time has to be kept at a certain distance to the next internal sampling interval. This task is achieved in *PolarisServer* by using a simple proportional controller (P-controller) with a manually tuned gain where the response time can be specified as the desired value. The best value for the desired response time is, with 8ms, about half the internal sampling period. This setting enables robust compensation of unpredictable timing variations in both directions as the request time is furthest away from the preceding and succeeding answers. This procedure ensures best possible timing and fastest availability of sensor data which is immediately sent to *HexGuide* through a TCP/IP socket after transformation into the room-fixed reference frame.

Not only are the 6D positions of the tools sent, but also a timestamp, which is acquired as soon as the answer of the positioning system has fully arrived through the serial interface. To improve precision of timestamps, serial transmission time is subtracted from both timestamps which can be easily calculated from the serial speed, the number of data and stop bits and the amount of transferred data. Additionally the timestamps are corrected by measurement latencies in order to make the timestamps reflect the true time of physical measurement. Latency measurements of the Polaris system are discussed in Section 2.2.3.

Another important task is calibration of a transformation matrix (represented as a quaternion) to transform measurements within the camera-specific reference frame to the room-fixed reference frame. Details on this topic are covered in Section 2.2.2.5.

2.2.1.5 Software Package *HexGuide*

HexGuide is the heart of the ATTS. Its primary tasks are:

- Synchronization of incoming data,
- Peak detection in either the breathing or the tumor signal,

- Calculation of moving average of breathing and tumor signal,
- Calculation of tumor motion predictions,
- Generation of a reference trajectory for the HexaPOD based on predictions,
- Implementation of a control scheme to allow the HexaPOD to follow the reference trajectory and
- Logging of incoming and all generated data.

The whole functionality of *PolarisServer* has additionally been integrated into *HexGuide* as well to simplify the use of a real-time operating system (discussed later in Section 2.2.2.6) and to minimize software-induced latencies resulting from thread scheduling.

Figure 2.6 sketches the information flow and the major elements in *HexGuide* to fulfill aforementioned tasks in the standard configuration. Each processing cycle is started by an internal trigger, configurable to an arbitrary frequency which determines the fundamental sampling rate used in *HexGuide*. Samples of IR tool positions (either received from *PolarisServer* or acquired by the integrated Polaris interface) are linearly interpolated to past sampling instants of the trigger. Since no newer Polaris samples can be received than the current trigger sampling instant, the last Polaris sample can at best only be used to interpolate to the previous trigger sampling instant (if the timestamp in the last Polaris sample refers to a time later than the previous trigger sampling instant). If the Polaris sample should be older (i.e. when a sampling has been missed), then the interpolation is only performed up to the last trigger sampling instant before the timestamp of the Polaris sample. Ideally the trigger frequency and Polaris update rate should be set to the same value. Due to the equidistant nature of Polaris samples, *HexGuide* synchronizes the trigger

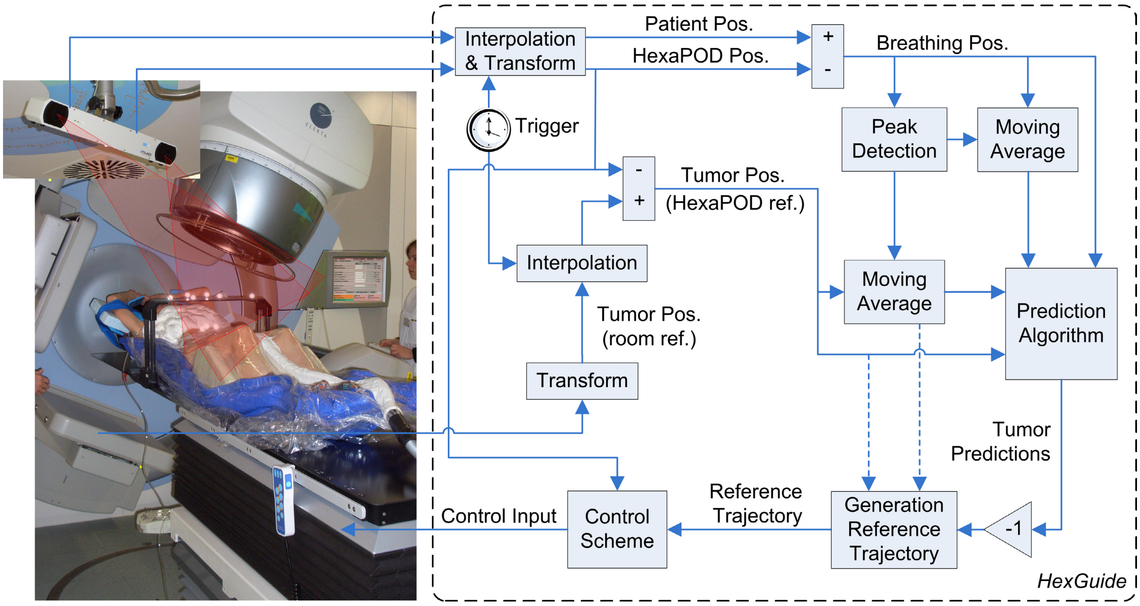


Figure 2.6: *Block diagram of the information flow and main elements of* HexGuide *in the standard configuration; dashed lines point out options depending on the compensation mode*

sampling instants to the arrival of Polaris samples in such a way that on each reception the trigger is instructed to fire, even if its timeout period has not expired. The timeout period is thereby set to the Polaris sampling period plus a short grace time of a few milliseconds. If no Polaris sample arrives in between the timeout period, the trigger fires. This scheme guarantees that even in case of missing samples, the trigger still fires and in case a valid sample arrives the trigger immediately fires which avoids unnecessary software-induced latencies. After the interpolation and transformation of the IR tool positions into the room-fixed reference frame, the breathing position is calculated by the difference between patient and HexaPOD tool positions. Then, peaks are detected in the breathing signal (or in the tumor signal, depending on the compensation mode) to enable the calculation of the moving average present in the breathing signal. Using information from the peak detector, the moving average of the tumor position w.r.t. the HexaPOD is also determined. The tumor position is determined by first transforming the tumor position coming from *PortalTrack* into the room-fixed reference frame (if a gantry angle different from zero is used), afterwards an interpolation to the trigger sampling instants is performed. Then the tumor position relative to the HexaPOD is calculated by subtracting the position of the HexaPOD. The moving averages of the breathing and tumor signals as well as the signals themselves are the inputs to a prediction algorithm. From current and past input data samples, future tumor position predictions are calculated. The predicted positions have to be negated, i.e. multiplied by -1 (given an adequate initialization in the zero position or the consideration of the proper offset), since compensation of tumor motion is targeted here. Depending on the motion compensation mode (RDC, DMC or FMC), a reference trajectory for the HexaPOD is calculated from the inverted predicted tumor position, from the moving average of the tumor position or from the tumor position signal directly. The reference trajectory and the position signal of the HexaPOD form the input of the control scheme whose task is to determine the control input to the HexaPOD allowing it to follow the reference trajectory. This calculation cycle is repeated when the trigger fires again.

In general, *HexGuide* takes great care of timing. Apart from the interpolation of sensor data to trigger sampling instants, several software-induced latencies are compensated within *HexGuide*. The gap between the latest interpolated breathing or tumor motion information of a minimum length of one sampling period is bridged by adaptively increasing the prediction horizon (cf. Section 2.2.2.2). Necessary calculations within *HexGuide* like transformations, interpolation, peak detection, moving average and the execution of the prediction scheme require a certain amount of time. In order to eliminate the effect of these processing latencies, an estimation of the “current” HexaPOD position is performed right before the control scheme is executed. The estimation is based on a linear extrapolation which takes into account the latest measured position of the HexaPOD, the time difference between that position information and “now”, and the last command sent to the HexaPOD, from which an estimation of the motion velocity is made. Finally, the estimated HexaPOD position refers to the point in time, right before the control scheme starts its work. Using the estimated HexaPOD position requires a time shift in the reference trajectory as well. This is accomplished by linear interpolation of the reference trajectory to appropriate timestamps relative to “now”. This requires the predictor to perform one additional forecast.

Possible realizations of the blocks “Prediction Algorithm” and “Control Scheme” are the main concern of this work and are discussed in Chapter 3 and Chapter 4, respectively. Details on interpolation, peak detection, calculation of moving average and reference frames and their calibration are explained in Section 2.2.2. These aspects found the basis of the rest of the work.

HexGuide and the included prediction algorithms were developed in a flexible way in order to additionally permit extended configuration possibilities. For example, they can also be applied in scenarios where only breathing motion is acquired, resulting in compensation of the breathing motion of a volunteer who is lying on the HexaPOD. This is especially useful for tests and evaluation purposes, which was frequently done in this work. Additionally, the prediction algorithms also work when no breathing motion is acquired, but only information about tumor motion is available. This can be the case if other tumor motion determination systems are used, e.g. the Calypso® system from Calypso Medical Technologies Inc., Seattle, Washington, USA. Then, the internal trigger directly drives the interpolation of the tumor motion signal as depicted in Figure 2.7. In contrast to previously discussed configuration, peak detection is then solely based on tumor motion information. Since no direct measurement of the HexaPOD's position is available in this configuration, the position information is directly retrieved from the HexaPOD using its communication protocol. This option also exists in the standard configuration, but it is not favored since the position has to be requested using a serial command and together with the answer about 50 Bytes have to transferred through the serial connection. This is running at a fixed rate of 57600bps (8,N,1) leading to a serial transmission time of 8.68ms. This time is compensated within *HexGuide* by interpolation of the predictions as described above. Nevertheless, the additional time spent for retrieving the HexaPOD's position restricts the overall sampling rate of the system and consequently this option can only be activated with lower sampling rates like 20Hz or 30Hz. This reduction occurs due to the fact that the duration from sending the "MOV" command (which is sent during every sampling instant) until arrival of the answer from the HexaPOD is in the range of several milliseconds (for the HexaPOD evo this is approx. 15ms), but the total time of the sampling period is not allowed to be exceeded (e.g. 16.667ms for a sampling rate of 60Hz)

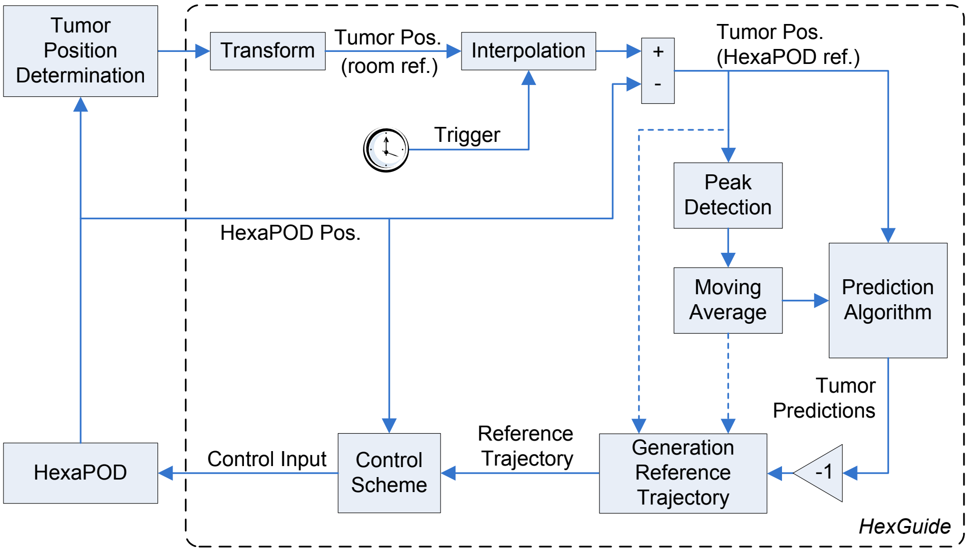


Figure 2.7: *Block diagram of the information flow and main elements of* HexGuide *in an alternative configuration with no determination of the breathing motion; dashed lines point out options depending on the compensation mode*

during the whole cycle. Of course, when this configuration is used, it has to be ensured that the information contained in the tumor motion signal is sufficient for reaching the mode-dependent goal of motion compensation.

2.2.1.6 Testing Software

For testing and evaluation purposes of single features or parts of presented system, several testing software packages were developed to simulate individual system components.

First, there is a simulator which simulates every hardware component used in the standard configuration. This program can be supplied with generated or pre-recorded breathing or tumor trajectories which are sent to *HexGuide* as primary and secondary signals. It also receives commands from *HexGuide* to reposition the simulated HexaPOD. The position of the HexaPOD is thereby superimposed on the primary and secondary signal to simulate the effect of a moving HexaPOD. The simulator program also takes care about timing issues so that pre-determined trajectories remain in sync with their time specification.

Another simulator program was developed which interfaces with the infrared positioning system and superimposes a pre-recorded breathing signal on the actual measurement of the IR camera which measures only the position of the HexaPOD. The superimposed signal is then sent as secondary signal to *HexGuide*. As an option of this simulator, a predefined tumor trajectory can also be supplied which is used for the primary signal. In this configuration, the real HexaPOD is used. So it is possible with this simulator to use pre-recorded breathing and tumor trajectories from real patients or pre-calculated artificial signals as a replacement for breathing and tumor signals, instead of having volunteers lying on the HexaPOD, but still using most of the real hardware.

2.2.1.7 Testbeds

All components of the new tumor tracking system previously described were developed during the course of this work and installed at the Department of Radiation Oncology in Würzburg. Here, the same hardware components are used for the ATTS which are currently used for actual treatment of patients. This fact emphasizes the simplicity of presented approach to the problem of tumor motion compensation, solely using existing hardware.

Comprehensive studies and tests have also been accomplished in diverse aspects including image acquisition and processing of the EPID subsystem [59, 62], EPID performance evaluation with real patient data [63], sensor data fusion of EPID and the infrared sensor for tumor position determination and prediction [64, 65]. In [48], integration and test results of the tracking system were presented with a 4D phantom (6-DoF manipulator arm) simulating the tumor motion with real pre-recorded data from lung tumor patients.

A subset of the hard- and software components have also been installed at the Department of Robotics and Telematics at the University of Würzburg for development and testing purposes. The testbed shown in Figure 2.8 consists of an earlier version of the HexaPOD$^{\text{TM}}$ RT CouchTop and of the successor of the Polaris IR positioning system named Polaris® Spectra®.

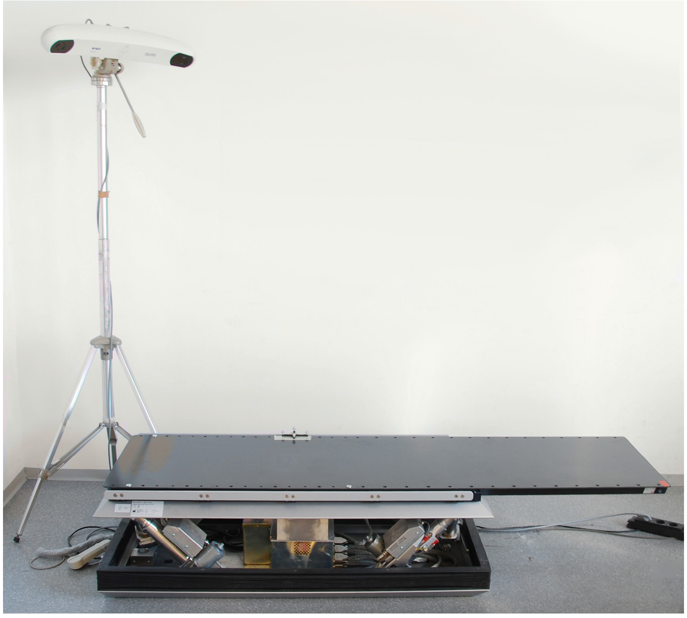


Figure 2.8: *Testbed setup consisting of early version of the HexaPOD RT CouchTop and Polaris Spectra IR camera*

2.2.2 Basic Software Features

Several basic aspects of the developed software are discussed in this section in order to provide common features supporting the creation of the tumor motion compensation system. After the considerations regarding sensor data timing, interpolation of tumor position is addressed. Then, the developed method of online peak detection for determination of breathing or tumor motion frequencies is presented which is the basis for the determination of breathing and tumor moving averages. Reference frames for sensors and actuators and their calibration based on an optimization technique are detailed as well as implementation specific issues concerning operating systems and their real-time capabilities.

2.2.2.1 Timing of Sensor Data

A very crucial issue in a motion compensation system is timing. Especially during sensor data acquisition, e.g. assessment of IR camera measurements or tumor position measurement, timing errors should be compensated. An overview of relevant points in time during sensor data acquisition using the example of a polled sensor is depicted in Figure 2.9. In this scenario, a sensor is connected to a computer or some kind of programmable device able to communicate with the sensor. The first step on the computer, starting at timestamp $\tau_{prepare}^{request} \in \mathbb{R}$, is to prepare the request (note: all following timestamps denoted in this form are elements of $\mathbb{R}$). That is transmitted at $\tau_{transmit}^{request}$ to the sensor where it is received at time $\tau_{received}^{request}$. Then it is processed on the sensor. Some sensors acquire measurements during a certain time interval, taking from time $\tau_{start}^{measure}$ to $\tau_{stop}^{measure}$. Then, measurements are usually processed on the sensor, requiring some more time until the measurement is transmitted to the computer at $\tau_{transmit}^{measure}$. There it is fully received at $\tau_{received}^{measure}$. Finally, parsing the data sent by the sensor requires a certain time, so that after $\tau_{parsed}^{measure}$ the sensor information is available for further processing.

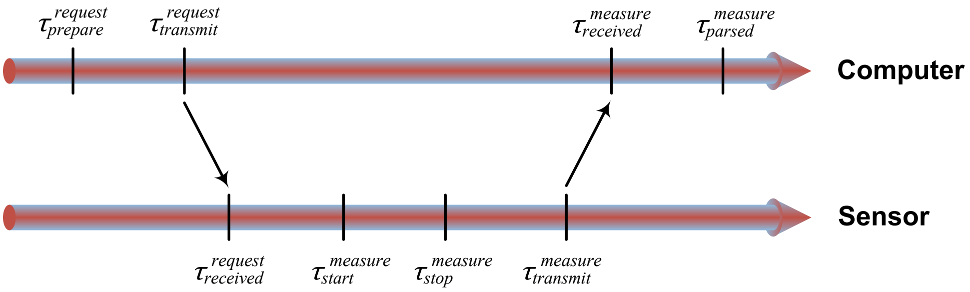


Figure 2.9: *Overview about timing between polled sensor devices and computer*

Depending on the capabilities of the individual devices, several possibilities arise concerning available timing information and its quality:

First, in the best case, measurements already contain a timestamp and an estimation of their age. The age is a time difference between the point in time when the physical measurement took place and the point in time when the timestamp was acquired. Then, timestamps can be corrected by the age and a good estimate of the timestamp of physical measurement can be gained which should lie in between $\tau_{start}^{measure}$ and $\tau_{stop}^{measure}$. Of course, a calibration of the offset between the sensor's clock and the clock of the receiving computer needs to be arranged before the acquisition process. If the acquisition process lasts for a longer time, checks of the calibration and an eventual recalibration should be performed in regular intervals. Modern techniques also allow to estimate a drift between the clocks which can be taken into account during runtime, so that timing accuracy can be improved without the necessity of too many recalibrations.

Second, measurements only contain a timestamp. Then the same considerations apply as before, but the age needs to be estimated. Here, the age is bounded by the duration of the last sampling period (assuming a serial procedure of sensor data acquisition, which is usually the case). A better estimation of the age can be eventually gained by knowing, how the measurement process works in detail. Then, this age can be used in the calculation of the timestamp or eventually neglected if it is very small.

Third, measurements consist of pure data without timestamps. Then, each measurement has to be timestamped by the software. It should be implemented in such a way that a timestamp is acquired at the earliest possibility to avoid or at least reduce program execution delays or scheduling latencies on the receiving computer. Also here, the upper bound of the age is usually the duration of the last sampling period.

In any of the previous groups, it can be possible to account for transmission delays (time difference between $\tau_{transmit}^{measure}$ and $\tau_{received}^{measure}$). When using for example a serial transmission, the lower bound on the transmission time can be calculated very accurately using the length of received data, number of data and stop bits, parity setting and serial transmission speed. When using a transmission control protocol (TCP) or user datagram protocol (UDP) connection, an estimation of the one-way delay is possible either using simple assumptions that the one-way delay is half the round-trip time or more sophisticated methods [66] can be used to estimate one-way delay. If the sensor is operating in polling mode, then the time for sending and transmitting measurement requests to the sensor can be considered as well (time difference between $\tau_{transmit}^{request}$ and $\tau_{received}^{request}$).

To summarize, all used sensors in motion compensation systems should be analyzed and appropriate time stamping mechanism selected in order to realize best possible timing information. For the used hardware in this work, the following can be stated: The Polaris IR stereo camera belongs to group three, but with the addition of an equidistant sampling rate. Acquired timestamps of arrived data are corrected by the serial transmission time as well as by the measured latency as discussed in Section 2.2.3. EPID processing is entirely done in software, so the acquisition software already determines a timestamp at the earliest possibility. Technically it belongs to group three, but *HexGuide* already receives timestamp information, so that from the perspective of *HexGuide* it rather belongs to the group two. The Calypso system for example belongs to group one. There, all timing information as well as the age are transmitted to *HexGuide*.

2.2.2.2 Interpolation of Sensor Data

Since most prediction algorithms and control methods require equidistant sampling intervals, *HexGuide* provides the option to interpolate the tumor position signal to a constant sampling rate. The choice between different interpolation types is thereby mainly influenced by the average sampling rate and the main characteristics of the tumor position signal. Furthermore, the positions of the patient's abdomen and HexaPOD, determined by the Polaris system, need to be interpolated as well to trigger sampling intervals as described before.

When continuous time signals are discretized using a certain sampling rate $f_s \in \mathbb{R}$, then sampling errors can occur. In case of quasi-periodically moving lung tumors, their trajectory can be roughly approximated with a sine. Motion frequencies of lung tumors, where their maximum main motion frequency is determined by the corresponding breathing frequency, usually do not exceed 0.6Hz (cf. Section 3.1.2). Hence, this can be regarded as an upper bound $f_{max} \in \mathbb{R}$ on breathing frequencies, leading to the tumor motion approximation

$$f(\tau) = a \sin^x(2\pi f_{max} \tau) \ ,$$

where $f(\tau) \in \mathbb{R}$ indicates the one-dimensional tumor position at time $\tau \in \mathbb{R}$ and $a > 0$ represents the motion amplitude. The exponent $x \in \mathbb{N}$ accounts for closer approximation of real motions by incorporating hysteresis compared to a pure sine. The first peak of $f(\tau)$ is located at one quarter of the signal's period $T_{max} = \frac{1}{f_{max}}$ at time

$$\tau_{peak} = \frac{1}{4} T_{max} = \frac{1}{4 f_{max}} \ .$$

Maximum sampling errors occur, if this signal is sampled in such a way that the sampling interval is equally distributed around τ_{peak} so that half of the interval is located on the left side of τ_{peak} and the other half on the right side. Then, the peak value is not captured by the sampling process and the sampling error maximized. Figure 2.10 illustrates this situation in case of $f_{max} = 0.5\text{Hz}$ and $f_s = \frac{1}{T_s} = 2\text{Hz}$ for a pure sine, i.e. $x = 1$. If linear interpolation is used with this sampling rate, then the interpolation error is equal to the sampling error between the value at the sampling points adjacent to the peak and the peak value which is cut off. These sampling points relative to the peak are given by

$$\begin{aligned} \tau_1 &= \tau_{peak} - \frac{T_s}{2} = \frac{1}{4 f_{max}} - \frac{1}{2 f_s} \text{ and} \\ \tau_2 &= \tau_{peak} + \frac{T_s}{2} = \frac{1}{4 f_{max}} + \frac{1}{2 f_s} \ . \end{aligned}$$

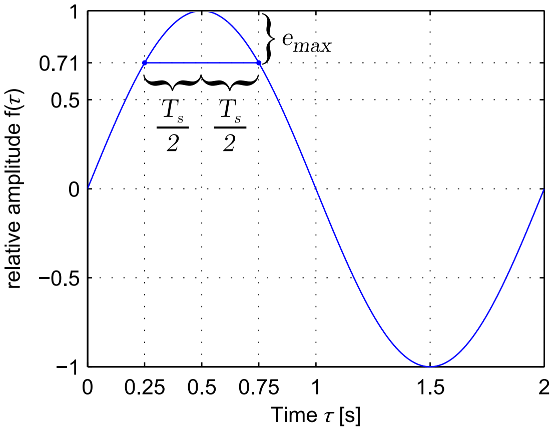


Figure 2.10: *Maximum sampling error e_{max} of a sine with frequency $f_{max} = 0.5Hz$ when sampling with $f_s = \frac{1}{T_s} = 2Hz$*

Thus, the maximum error e_{max} is given by

$$e_{max}(f_s) = |f(\tau_{peak}) - f(\tau_1)| = |f(\tau_{peak}) - f(\tau_2)| = a\left(1 - \cos^x\left(\pi\frac{f_{max}}{f_s}\right)\right) \ . \tag{2.1}$$

Its shape is mainly determined by the ratio between the signal's frequency f_{max} and the sampling rate f_s. From Eq. (2.1) the limits of e_{max} in terms of f_s are

$$\lim_{f_s \to \infty} e_{max}(f_s) = 0 \text{ and}$$
$$\lim_{f_s \to 2f_{max}} e_{max}(f_s) = a \ .$$

So the error decreases as the sampling rate increases and tends to zero as f_s approaches infinity. If f_s approaches its lower bound $2f_{max}$ (the Nyquist rate), the error increases and becomes as large as the absolute value of the amplitude $|a|$. In this case, the signal $f(\tau)$ would be sampled at sampling points τ with $f(\tau) = 0$, leading to the maximum error of $|a|$. Note that the Nyquist rate should not be used as sampling frequency because the sine cannot be reconstructed from the samples $f(\tau) = 0$ since the amplitude information is missing. This is coherent with the Nyquist-Shannon sampling theorem which states that the sampling frequency of a system has to be strictly greater and not equal to the bandwidth of the system or the main frequency component, so that the original signal can be reconstructed from samples without any information loss. Furthermore, the error is proportional to the amplitude a, which is set to 1 for the following considerations to deliver the error relative to the signal's amplitude. Figure 2.11 shows the interpolation error from Eq. (2.1) in the worst case ($f_{max} = 0.6$Hz) over the sampling rate relative to the amplitude a of the original signal $f(t)$ for $x = 1, 2, 4$. From that three major intervals can be distinguished from each other:

- Worst case error is between 1 (100% of the amplitude a) and 0.1 (10%) if low sampling rates up to about 4.1Hz for $x = 1$, 5.8Hz for $x = 2$ and 8.2Hz for $x = 4$ are used.
- Worst case error between 0.1 and 0.05 (5%) appears with medium sampling rates up to 5.9Hz for $x = 1$, 8.4Hz for $x = 2$ and 11.8Hz for $x = 4$.

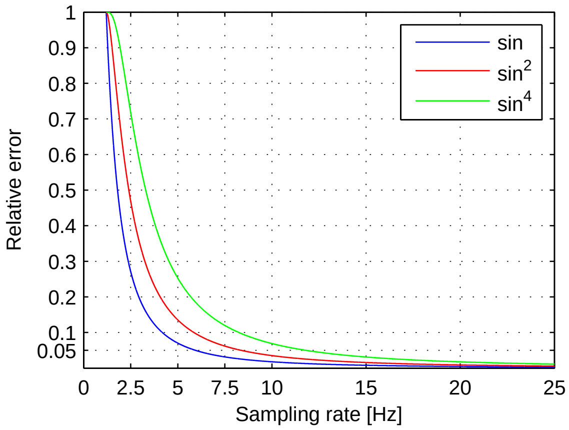


Figure 2.11: *Relative linear interpolation error when sampling* $\sin^x$ *with* $x = 1, 2, 4$ *of maximum breathing frequency of* $0.6Hz$ *in terms of sampling rate* f_s

- Worst case error is lower than 5% for higher sampling rates.

Note, that here the worst case error was observed. If the phase shift between the actual signal and the sampling process is smaller, then lower values of interpolation or sampling errors can be expected. Also note, that other measures of this error might be used as well for this consideration, e.g. integral error between original and linearly interpolated signal.

Several conclusions can be drawn from these considerations: If the sampling rate is large enough assuming a certain motion approximation (choice of x), then linear interpolation can be used for a quasi-periodic tumor motion resulting in a worst case sampling error smaller than a defined limit. If the sampling rate is lower, then higher-order interpolation techniques can be used such as quadratic or cubic polynomial interpolation, trigonometric interpolation or cubic spline interpolation in order to decrease interpolation errors compared to linear interpolation in some situations. These techniques help to better approximate the signal's shape in between given sampling points as they assume a certain shape by estimating parameters of an appropriate function in order to fit the function to the sampling points.

However, if the tumor signal does not contain any periodicity for which a higher-order interpolation would make sense because assumptions on the shape of the fitting functions are not applicable to the signal, then the linear interpolation method is preferable as this method only assumes a straight line in between two sampling points and thus can average possible interpolation errors. Linear and quadratic polynomial interpolation methods have been implemented in *HexGuide* which are described below:

Assume that successive measurements of a one-dimensional real-valued signal y are taken at arbitrary timestamps $\tau(t) \in \mathbb{R}$ with $t \in \mathbb{N}$, leading to a sequence of measurement samples $y(t) \in \mathbb{R}$. Let the latest measurement be the one with index t. Further let the timestamp to which the signal should be interpolated be $\tau_i \in \mathbb{R}$ with $\tau(t-1) \leq \tau_i \leq \tau(t)$ and $\tau(t-1) < \tau(t)$, where τ_i is determined by the trigger (cf. Section 2.2.1.5). The linear first-order interpolation function $f_1 : \mathbb{R} \mapsto \mathbb{R}$ to be fitted to the data points at $\tau(t)$ and $\tau(t-1)$ is

$$f_{i_1}(\tau) = c_1\tau + c_0 \ ,$$

where parameters $c_1 \in \mathbb{R}$ and $c_0 \in \mathbb{R}$ need to be estimated from the conditions $f_1(\tau(t-$

1)) = $y(t-1)$ and $f_1(\tau(t)) = y(t)$. The slope c_1 of the linear function can thus be easily calculated from sampling points at t and $t-1$:

$$c_1 = \frac{y(t) - y(t-1)}{\tau(t) - \tau(t-1)} \ . \tag{2.2}$$

Based on c_1, the parameter c_0 in the constant part of the interpolation function can be derived as

$$c_0 = y(t-1) - \tau(t-1)c_1 \ .$$

Function f_1 can finally be evaluated at τ_i to find the interpolated value corresponding to the timestamp.

The quadratic interpolation uses a second-order polynomial as fitting function $f_2 : \mathbb{R} \mapsto \mathbb{R}$

$$f_2(\tau) = c_2\tau^2 + c_1\tau + c_0 \ .$$

In order to estimate the parameters c_0, c_1 and $c_2 \in \mathbb{R}$, three sampling points must be taken into account with $\tau(t-2) < \tau(t-1) < \tau(t)$ and $\tau(t-2) \leq \tau_i \leq \tau(t)$. These parameters can be found from the equalities $f_2(\tau(t-2)) = y(t-2)$, $f_2(\tau(t-1)) = y(t-1)$ and $f_2(\tau(t)) = y(t)$ by solving a system of linear equations similar to first-order interpolation. Evaluation of function f_2 at τ_i delivers the interpolated value of quadratic interpolation. This method was chosen as a higher-order addition to the linear method because it shows good performance when applied to lung tumor trajectories and it requires only three samples until the first interpolation can be calculated, thus reducing startup time. For the initialization of polynomial-based interpolation methods, the number of required tumor samples is generally the order of the polynomial plus one.

Consecutive sampling timestamps $\tau(t-1)$ and $\tau(t)$ or $\tau(t-2)$ and $\tau(t-1)$ do not need to be equidistantly spaced for the interpolation to work. However, in order to avoid numerical problems in the implementation, a certain distance between them needs to be ensured so that the differences in the denominator in Eq. (2.2) and in the corresponding equations for quadratic interpolation do not grow too small.

The determination of the coefficients can also be done in more efficient ways, e.g. with Newton interpolation or Neville's algorithm [67] which compute the coefficients recursively. These are especially useful, if many coefficients need to be calculated. However, the interpolation here is of low order and is performed online, moving from one sampling instant to the next. During calculation, common intermediate results occur within one sampling instant which can be used in the next, so that the iterative computation of these equations becomes efficient enough.

Figure 2.12 provides a comparison between linear and quadratic interpolation on a tumor position signal which was recorded by the ATTS during a radiotherapy session using tumor tracking with the EPID. The original samples (represented by blue dots) possess an average sampling rate of 1.77Hz with a standard deviation of 0.37Hz. The blue curve depicts the linear interpolation and the red one shows the quadratic interpolation, both at a sampling rate of 20Hz. It can be estimated that in some situations a potential improvement over the linear interpolation is reached, e.g. at 9.2 seconds, whereas in other situations, e.g. at 3.1 and 41.8 seconds, larger or smaller overshoots can be observed. Generally, tumor position samples should be analyzed before the usage of an interpolation method is decided. Ideally, separate reference system with a larger sampling rate is required to quantitatively compare the outcomes of different interpolation methods.

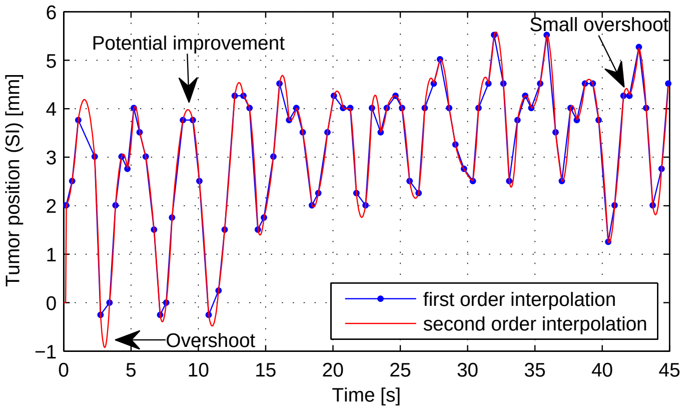


Figure 2.12: *Comparison of linear and quadratic interpolation on tumor position signal gained from the EPID of the ATTS*

Of course, when employing any interpolation method, a delay is introduced into the system until the last arriving data sample can be processed further. The worst case delay is thereby depending only on the interpolating sampling rate. This case becomes apparent, if right after the last interpolating sampling instant a new data sample arrives. Then it takes until the next interpolating sampling instant before that data sample can be used in an interpolation. Hence, the time delay $\Delta T_i \in \mathbb{R}$ in the worst case is bounded by the interpolating sampling rate $f_{trigger}$:

$$\Delta T_i \leq \frac{1}{f_{trigger}} = T_{trigger} \ , \tag{2.3}$$

where $T_{trigger} \in \mathbb{R}$ is the corresponding interpolating sampling period. The larger this sampling frequency, the smaller the time delay ΔT_i.

Table 2.2 subsumes the results from this section to aid in deciding which interpolation method is appropriate when taking into account the sampling rate of the tumor position signal and its general shape. Wherever quadratic interpolation is appropriate, higher-order methods are in general applicable as well.

Sampling Rate Signal Type	Low	Medium	High
Random, non-predictable e.g. prostate tumor	Linear	Linear	Linear
Quasi-periodic, predictable e.g. lung tumor	Quadratic Higher-Order	Linear Quadratic Higher-Order	Linear Quadratic Higher-Order

Table 2.2: *Recommendations for interpolation methods depending on sampling rate and shape of tumor position signal*

The previously discussed interpolation guarantees that an equidistant sampling rate of the tumor position signal can be reached regardless of the behavior of the underlying sensor. However, it does not handle the availability of the sensor data. Figure 2.13

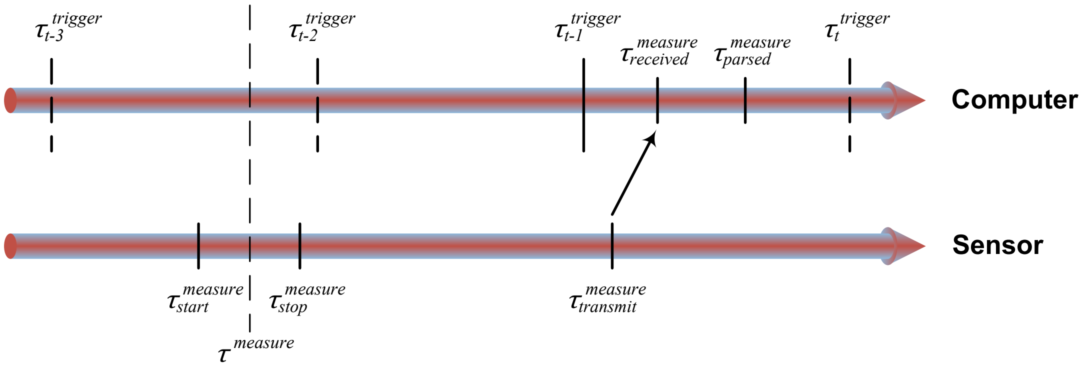


Figure 2.13: *Illustration of timing condition between sensor and computer which need to be handled properly*

exemplifies a problematic situation, where the time of sensor data acquisition $\tau^{measure}$ (depicted as the average time between $\tau_{start}^{measure}$ and $\tau_{stop}^{measure}$) is lying in the past with a time shift of more than one interpolating sampling instant relative to time $\tau_{received}^{measure}$ (when sensor data is fully received by the computer). Suppose that the sensor data contains an exact timestamp and an exact age, then the time point $\tau^{measure}$, when the physical sensor observation was performed, can be derived accurately. In this situation, an interpolation of tumor position can be performed up to $\tau_{t-3}^{trigger}$, of course using older tumor samples, too. However, the current sampling point, whose time is considered as "now" for the prediction and control algorithms, is $\tau_{t}^{trigger}$. This gap can be compensated by using prediction algorithms whose prediction horizon can be adjusted online. So if a tumor position sensor is used with which the delay between sensor data acquisition and availability in *HexGuide* is more than one sampling interval, then the prediction horizon for the current sampling instant needs to be increased. The amount of increase is determined by the number of past sampling instants during which no new tumor position information was available. This property of an adaptable prediction horizon will become one of the requirements for prediction algorithms discussed later in Section 3.2.2, so that prediction algorithms can be applied in any situation. The same procedure is also followed during interpolation of the Polaris samples. Due to high sampling rate, a linear interpolation is used for these kind of data.

2.2.2.3 Online Peak Detection

HexGuide is equipped with an online peak detector, which can detect peaks in either the interpolated breathing or the interpolated tumor positions, depending on the chosen configuration. Peak information aids in the estimation of current breathing or tumor motion frequency and in the detection of a phase shift. The calculation of breathing or tumor moving average also benefits from the knowledge of peaks.

Usually, an easy way to detect peak values of a function is to equate its derivative to zero and to solve for the free parameters. This general idea can also be applied to the discretized breathing and tumor position signal, but with the addition of several checks to gain a certain robustness against measurement noise, measurement errors and high frequency oscillations with small amplitude changes.

Let $y(t) \in \mathbb{R}$ be a discretized, eventually interpolated (cf. Section 2.2.2.2) breathing or tumor position signal of length $n \in \mathbb{N}$ with equidistant sampling timestamps $\tau(t)$ with $1 \leq t \leq n$. The discrete derivative in terms of time at a certain sampling instant t of the signal is obtained from the difference equation

$$\frac{\Delta y}{\Delta \tau}(t) = \frac{y(t) - y(t-1)}{\tau(t) - \tau(t-1)} \ .$$

In order to achieve a robust estimation of the time derivative in the presence of noise, it is averaged over a certain number w_P of derivatives in a sliding window manner. Hence, it becomes

$$\overline{\frac{\Delta y}{\Delta \tau}}(t) = \frac{1}{w_P} \sum_{i=1}^{w_P} \frac{\Delta y}{\Delta \tau}(t - i + 1) \ .$$

Since it is unlikely for a discretized derivative to become zero, the zero crossing of the derivative, which indicates a peak point in the signal, can be determined by comparing the signs of two consecutive averaged derivatives. Let the sign function be defined as

$$\operatorname{sgn}(x) = \begin{cases} -1 & \text{if } x < 0 \\ 0 & \text{if } x = 0 \\ 1 & \text{if } x > 0 \end{cases}$$

for $x \in \mathbb{R}$. If the condition

$$\operatorname{sgn}\left(\overline{\frac{\Delta y}{\Delta \tau}}(t-1)\right) \neq \operatorname{sgn}\left(\overline{\frac{\Delta y}{\Delta \tau}}(t)\right) \tag{2.4}$$

is fulfilled, then a zero-crossing either between or at $t-1$ and t has occurred.

However, this condition can be regarded as necessary for the presence of a peak, but it might not be sufficient. Let the actual peaks of the signal be denoted as the “global peaks”, whereas undesired peaks in between those are referred to as “local peaks” (global in the sense of unique during one breathing cycle). The local peaks can occur when in between two global peaks in the breathing or tumor signal, the patient eventually relaxes the breathing process, so that a plateau occurs (or even a temporary move in reverse direction) for a short time after which the patient continues breathing normally. Or they arise when the input signal is noisy, so that the sliding window filtering cannot filter the noise completely. Then, additional local peaks might be detected during a short interval which still belong to the same part of the breathing cycle. Therefore a procedure needs to be developed to distinguish between local and global peaks. Doing this in retrospect turns out to be quite easy. Then a properly sized sliding window can be used to filter most of the noise and adjacent peak candidates can be compared and checked for plausibility, whereby possible future peaks can also be taken into account. However, when implementing such a feature in an online fashion, the decision needs to be done as fast as possible without introducing too much delay and without knowledge from the future. The sliding window filtering with a filter size of w_P already introduces a delay of $\lceil w_P/2 \rceil$ sampling instants. Note that sliding window filters are in general suitable for this purpose since they are linear phase filters (the phase shift between output and input signal is a linear function of signal frequency) with a constant group delay of $w_P/2$ samples. From experiments using the previously described hardware, values for w_P between 3 and 5 offer a good trade-off between delay and filtering performance for noise suppression.

The rest of the decision process after applying condition (2.4) was therefore designed to be computable in constant time. For the process described in the following, two lists are built in the course of the peak detection process storing amplitude and timestamp of each peak (after the decision process is successfully completed). The list L_{max} contains the maximum peaks and L_{min} the minimum peaks. The type of the last peak is also stored. For each potential peak the following decision process is applied:

1. If a zero-crossing is detected according to condition (2.4), then proceed, else stop decision process.
2. If type of peak detected at last and type of potential peak are the same and if amplitude of potential peak is larger (in case of maximum peak) or is lower (in case of minimum peak) than amplitude of last peak of the same type, then overwrite the last peak information with the potential peak and proceed with step 6.
3. If the difference between peak time to time of last maximum / minimum peak is larger than or equal to T_{min}, then the potential peak is considered further. The minimum peak period T_{min} has to be chosen in such a way that it is not too small to let an early local peak be identified as global peak, nor too large to skip a global peak if the actual breathing frequency rises.
4. If step 3. failed, the potential peak is granted another chance: If difference between peak time to time of last maximum / minimum peak is smaller than T_{min}^{global}, then the potential peak is ultimately refused and the decision process is stopped. Otherwise the peak is checked further. T_{min}^{global} represents the minimum of all previously detected breathing periods gained by this process of peak detection. It brings in more robustness in case the duration of breathing cycles show a larger variation throughout a fraction.
5. If the difference between the last minimum / maximum peak value and the potential peak value is smaller than P_{min}, then the potential peak is ultimately discarded. P_{min} constrains the minimum amplitude difference between two different peak types.
6. Finally, when reaching here, the potential peak is considered a valid one. Now, the lists containing the minimum and maximum peaks are updated.
7. The peak detector contains an option to automatically adapt the parameters. If this option is enabled, then the value for P_{min} is updated in such a way that this is set to a configurable fraction between 0 and 1 of the amplitude difference between the current peak and the last peak of the opposite type if that value is smaller than the current value of P_{min}. T_{min} is updated in the same way where the reference is the length of the last detected breathing period. If the last breathing period including the peak is shorter than T_{min}^{global}, then T_{min}^{global} is set to the last breathing period (which is the shortest so far).

The flow chart in Figure 2.14 summarizes these steps and gives an overview of the individual decision steps during peak detection.

If the peak detector is running without the adaptation option, then step 4. can be omitted because it is not required. In any case, initial values for all conditions need to be found. Reasonable values can be extracted from extreme and average cases of breathing or

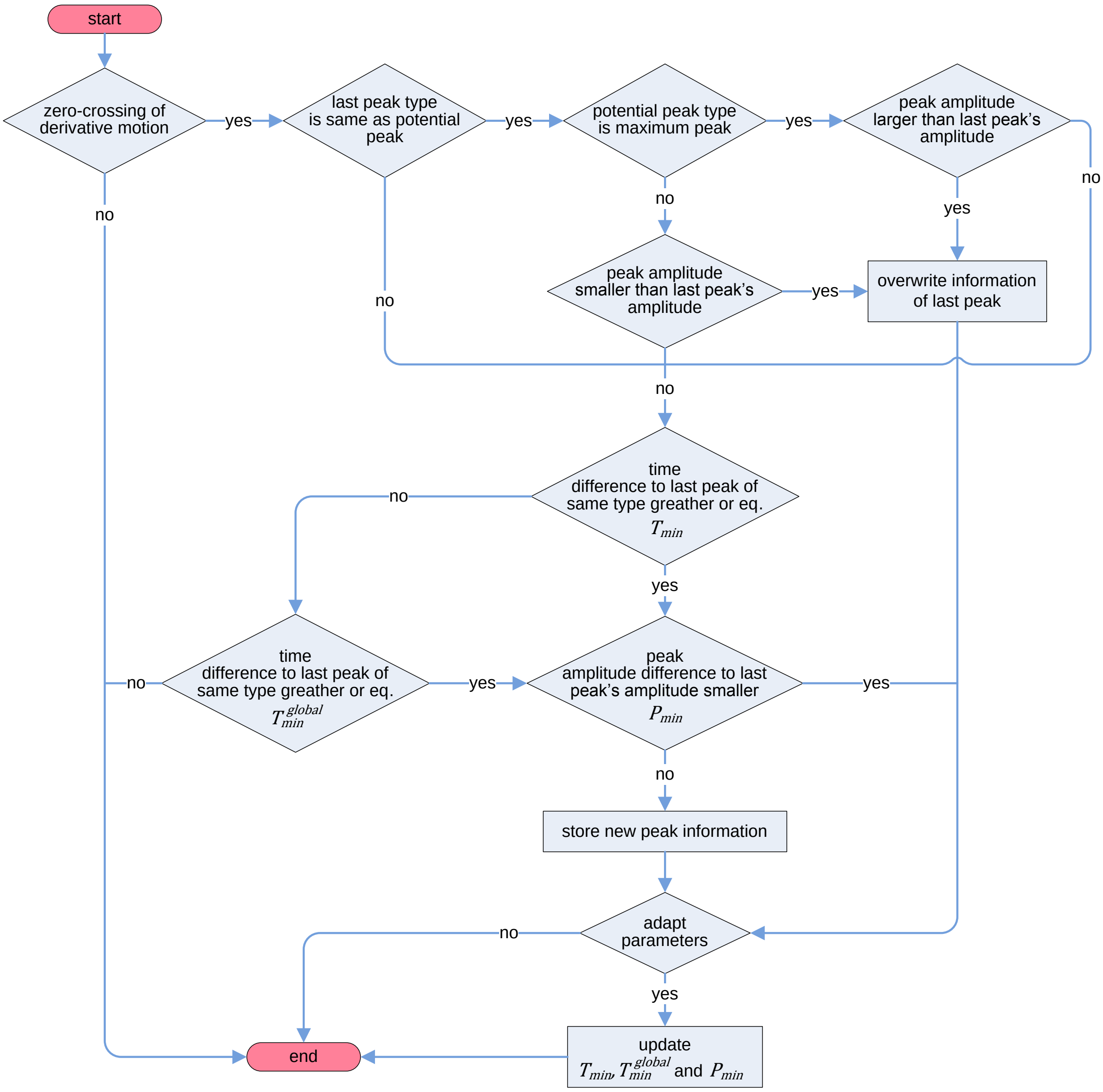


Figure 2.14: *Flow chart describing the individual decision steps during peak detection*

tumor motion, see also Section 3.1. In combination with the knowledge of the algorithm's working principle, initial values can be estimated. Then these values should be tested using as many as real patient datasets as possible. If some peak is skipped or a false peak identified, then these values should be tuned. However, with adaptation enabled, the need to find precise values is not critical as these values adapt to the current situation. In this case only the fractions of the breathing period need to be chosen with which the minimum times are calculated.

2.2.2.4 Determination of Moving Average

The moving average of either interpolated breathing or tumor positions can be determined within the ATTS for the realization of two functionalities: First, when using drift compensation mode (cf. Section 2.1.2) information about the current drift must be available, which can be derived from the moving average and second, some prediction algorithms can perform better predictions, if their input signals are centered around their mean. Here, the corresponding moving average is removed from the input signal, which is fed into the predictor. Then the predictor calculates a prediction from the mean-centered input signal and finally, the current moving average is added to the prediction to undo the initial removal of the moving average.

In general, the moving average can be derived using a sliding window of size $w_M \in \mathbb{N}$ applied to an equidistantly discretized breathing or tumor position signal $y(t)$ of length $n \in \mathbb{N}$ with sampling timestamps $\tau(t)$ with $1 \leq t \leq n$. The output $d(t) \in \mathbb{R}$ at sampling instant t of the sliding window can then be determined by

$$d(t) = \frac{1}{w_M} \sum_{i=1}^{w_M} y(t-i+1) \ , \tag{2.5}$$

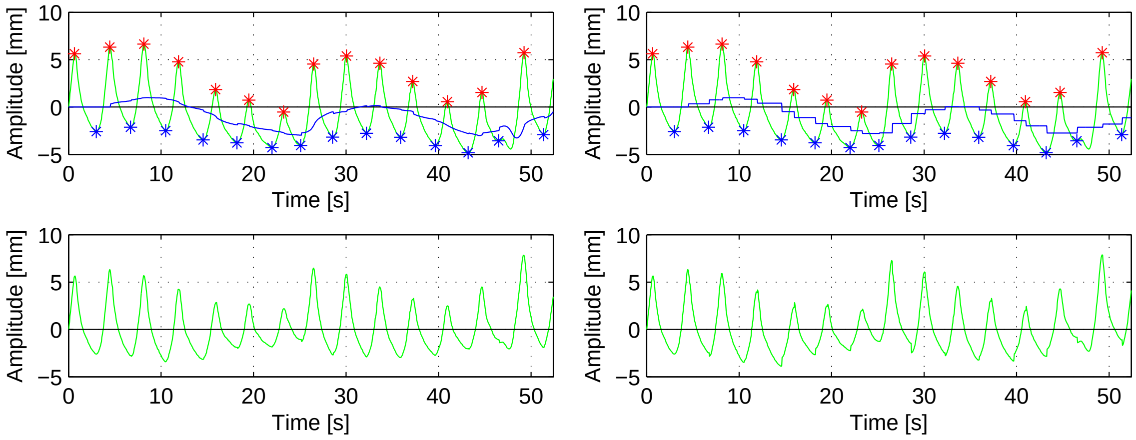
using the same weight of $\frac{1}{w_M}$ of each contributing sample of y.

In the case of a non-periodic signal, e.g. prostate motion, w_M should be chosen large enough to filter short-term peaks in the motion, but small enough to capture current signal trends. If it is too large, then older samples have a stronger impact than newer samples which might lead to over- or underestimation of the current trend.

For periodic signals, w_M should be set to the corresponding number of samples for the duration of one period to avoid estimation errors. Thereby it is not important to start and stop the calculation exactly at a peak or zero-crossing, but to use the correct number of samples. However, breathing motion in humans is subject to amplitude changes and phase shifts, so that the peak detection discussed in the previous section can help to choose the most up-to-date value for the window size w_M.

Two versions of moving average determination were implemented in *HexGuide*. The first version updates the moving average described by Eq. (2.5) in each sampling instant. This accommodates the fact that drift changes occur continuously during a breathing cycle. Nevertheless, the window size w_M is updated as soon as a peak is detected to account for phase changes in the breathing. The second version updates the moving average only during the sampling instant when a peak was detected. The moving average is then kept constant similar to a zero-order hold until the next peak is detected. A disadvantage compared to the first version is that this leads to more discontinuities in the moving average, which can have an effect on the smoothness of the prediction. However, this scheme does not introduce new dynamics during a breathing cycle because the moving average stays constant in this interval.

An example of the first variant is shown in Figure 2.15(a). The upper diagram shows a breathing trajectory together with detected maximum (red crosses) and minimum (blue crosses) peaks. The moving average (blue line) depicts a quite smooth behavior. A few slight discontinuities in the moving average can be observed when a peak is detected and changes in duration of the last breathing period compared to the previous one is larger. In the lower diagram, the modified breathing trajectory is shown in which the moving average was removed. One can observe, that it is much better centered around the baseline at zero. However, if a peak is offset due to breathing irregularities, e.g. occurring at second 47, then larger changes of the moving average can be observed afterwards which are compensated after a few seconds. Figure 2.15(b) depicts the results on the same breathing trajectory of the second version with updates of the moving average only performed during a peak.



(a) Moving average continuously updated at every sampling instant

(b) Moving average updated only when peak was detected

Figure 2.15: *Example of moving average determination: In the upper diagram green depicts a breathing signal with detected maximum (red crosses) and minimum (blue crosses) peaks; Moving average is shown as solid blue line; in lower diagram the breathing signal with eliminated moving average is shown*

This clearly introduces discontinuities which can only occur when a peak is detected. Of course, they also lead to more discontinuities in the modified breathing signal in the lower diagram, but this scheme can average errors present in the first version, for example the ones at second 47. The moving average stays constant and the height of it is less influenced by the offset peak as in the first version.

There is also another method [68] to track the mean position of respiratory motion based on ellipse fits and estimation of ellipse center. Unlike the baseline method against the method was compared, which also uses a moving average but with a fixed window size, the moving average method as presented here uses an adaptive window size, which is determined by robust detection of the peaks in the motion signal. For the purpose of this work, the presented method was deemed to deliver a satisfactory performance by visual inspection on all available datasets.

2.2.2.5 Reference Frames and Calibration

Central tasks for calibration are to find the mappings between the sensor reference frames to the HexaPOD reference frame, so that measurements given relative to the sensors' reference frames can be transformed into the HexaPOD frame. The choice of the HexaPOD frame as ultimate reference frame is thereby arbitrary, but natural for the given problem. If the individual axes of the HexaPOD frame are aligned perpendicular to the axes of the isocenter reference frame, then the calibration process finds the mapping simultaneously to the HexaPOD and the isocenter frame. This is assumed in the following.

The mapping between two frames can be described by a homogeneous transformation matrix $T_{dest}^{src} \in \mathbb{R}^{4\times 4}$ of the form

$$T_{dest}^{src} = \left[\begin{array}{ccc|c} & & & \\ & R & & \vec{p} \\ & & & \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

where $R \in \mathbb{R}^{3\times 3}$ is an orthonormal rotation matrix defining the orientation of the frame *src* in relation to the reference frame *dest* and $\vec{p} \in \mathbb{R}^{3\times 1}$ is the translational displacement of the origin of frame *src* to *dest*.

The calibration process needs to be performed for each sensor separately. In order to find the mapping between a sensor's frame S and the isocenter frame I several measurements of certain points $\vec{p}_i \in \mathbb{R}^{3\times 1}$ with $i = 1, 2, \ldots$ need to be taken which are known in both reference frames. Then, a procedure is applied to estimate the best homogeneous transformation matrix T_I^S fulfilling the equation

$$\begin{bmatrix} [\vec{p}_i]^I \\ 1 \end{bmatrix} = T_I^S \begin{bmatrix} [\vec{p}_i]^S \\ 1 \end{bmatrix} \tag{2.6}$$

for all pairs of acquired points $[\vec{p}_i]^I$ (relative to isocenter I) and $[\vec{p}_i]^S$ (relative to sensor frame S). In order to derive a solution for the transformation matrix, at least three non-collinear points must be measured. More points can be used to reduce the effect of measurement errors. Point pairs can be acquired by placing a measurable entity on the HexaPOD, then moving the HexaPOD to certain known locations w.r.t. the HexaPOD (delivering $[\vec{p}_i]^I$) and taking the corresponding measurement by the sensor (delivering $[\vec{p}_i]^S$). Using the HexaPOD, some points should also be acquired close to operational range limits in order to get a good coverage of the whole range and to decrease effects of positioning errors. For example, this would mean placing an IR tool on the HexaPOD for the Polaris system or fixing the beacons of the Calypso system on the HexaPOD.

Both systems actually also deliver an orientation in addition to a position measurement. In this case, the formulation of Eq. (2.6) can be extended to

$$T_{I,i}^{S_M} = T_I^S T_{S,i}^{S_M} \tag{2.7}$$

where $T_{I,i}^{S_M}$ describes the i-th measurement of position and orientation of the moveable entity as homogeneous transformation matrix relative to frame I, and $T_{S,i}^{S_M}$ is the i-th measurement of the moveable entity with respect to the sensor frame S as transformation matrix. Since this also involves orientation, another local reference frame S_M needs to be introduced, which coincides with S if the entity is located at the origin of S with zero orientation. If we apply the calibration process in the manner of Eq. (2.7) with an arbitrarily oriented measured entity, then we would perform the calibration with respect to the orientation of the entity's axes given by frame S_M which are present during the time of calibration. This means that an orientation of zero around all axes in frame I is measured (rotation matrix in $T_{I,i}^{S_M}$ becomes identity matrix), when the entity's axes are aligned with the same orientation as during calibration. If the orientation should also be correctly calibrated, then the axes of the measured entity (frame S_M) would need to be aligned coincident with the isocenter or HexaPOD frame during calibration. However, this is an inconvenient requirement because it can hardly be ensured that for example the axes of an IR tool are perfectly coinciding with the HexaPOD axes. Therefore, the calibration problem can be reformulated to make the approach more flexible.

The problem of calibration in this situation is that the transformation matrix $T^{I}_{S_M,0}$ is unknown which describes the position and orientation of the isocenter or HexaPOD frame I in relation to the local reference frame S_M of the measured entity during the time of calibration (index 0 indicates one measurement taken during calibration). Post-multiplying this transformation matrix to both sides of Eq. (2.7) results in

$$T^{S_M}_{I,i} T^{I}_{S_M,0} = T^{S}_{I} T^{S_M}_{S,i} T^{I}_{S_M,0} \ .$$

Relating this to a certain displacement of the HexaPOD during the calibration which is given by $T^{I_M}_{I,0}$ where I_M denotes the mobile frame moving along with the HexaPOD leads to

$$T^{S_M}_{I,i} T^{I}_{S_M,0} T^{I_M}_{I,0} = T^{S}_{I} T^{S_M}_{S,i} T^{I}_{S_M,0} T^{I_M}_{I,0} \ .$$

$T^{I_M}_{I,0}$ is completely determined by the position and orientation information in one specific HexaPOD's command during calibration (assuming the HexaPOD has moved already into the given position and orientation). Combining the transformations with index 0 yields

$$T^{S_M}_{I,i} T^{I_M}_{S_M,0} = T^{S}_{I} T^{S_M}_{S,i} T^{I_M}_{S_M,0} \ . \tag{2.8}$$

Since the effect of change of the HexaPOD's position and orientation on the measured entity at an arbitrary sampling instant i is already contained in $T^{S_M}_{I,i}$ and $T^{I_M}_{S_M,0} = T^{I_M}_{S_M,i}$ is not varying with time, the left-hand side of Eq. (2.8) can be reduced to $T^{I_M}_{I,i}$ describing the position and change w.r.t. the isocenter frame when the HexaPOD moved:

$$T^{I_M}_{I,i} = T^{S}_{I} T^{S_M}_{S,i} T^{I_M}_{S_M,0} \ . \tag{2.9}$$

This information is immediately known from the HexaPOD's command, whereas $T^{S_M}_{S,i}$ is directly measured by a sensor. $T^{I_M}_{S_M,0}$ can be determined using information gathered during calibration:

$$T^{I_M}_{SM,0} = T^{S}_{S_M,0} T^{I}_{S} T^{I_M}_{I,0} \ . \tag{2.10}$$

$T^{S}_{S_M,0} = \left(T^{S_M}_{S,0}\right)^{-1}$ is the inverse of a single sensor measurement $T^{S_M}_{S,0}$ of the moveable entity during calibration, $T^{I_M}_{I,0}$ is the corresponding position and orientation of the HexaPOD and T^{I}_{S} is the inverse of the calibrated transformation matrix T^{S}_{I} from Eq. (2.7).

The whole calibration procedure can be summarized as follows:

1. Place the measured entity on the HexaPOD with an arbitrary orientation of S_M.
2. Assume that $S_M = I_M$ in Eq. (2.7) and record at least three point pairs of the entity measured by the sensor ($T^{S_M}_{S,i}$) and by the HexaPOD ($T^{I_M}_{I,i}$).
3. Apply the calibration process to determine T^{S}_{I}, resulting in the calibration of the origin of I in the location of the origin of S_M as well as in the calibration of the orientation of I in the same orientation as S_M due to the assumption.
4. Use $T^{I}_{S} = \left(T^{S}_{I}\right)^{-1}$ and one point pair to calculate $T^{I_M}_{S_M,0}$ from Eq. (2.10).

$T^{I_M}_{S_M,0}$ can be regarded as an orientation correction matrix in order to determine the correct rotation. However, this does not apply to the position, since the origin of I is still located in the origin of S_M at the time of calibration. If the HexaPOD does not need to be rotated during motion compensation, this restriction does not impose any limitation on the feasibility of this approach. If rotations also need to be calibrated, basically the same approach still applies, however the HexaPOD needs to be rotated during calibration when the point pairs are acquired. Additionally, the calculation of the position in T^S_I needs to be extended by an approximation of the origin of the HexaPOD reference frame.

Having obtained the homogeneous transformation matrices T^S_I and $T^{I_M}_{S_M,0}$ through the calibration process, a new sensor measurement $T^{S_M}_{S,i}$ needs to be multiplied with these matrices according to Eq. (2.9) in order to be mapped into the isocenter reference frame I.

The mentioned calibration process used in this work is based on a method [69] for fitting two 3D point sets using a least squares approach for optimizing the mapping between the point sets with a quadratic error criterion summarized as follows: Suppose that $n \geq 3$ non-collinear measurements of the point pairs $[\vec{p}_i]^I$ and $[\vec{p}_i]^S$ at different locations have been acquired. Then the mapping of Eq. (2.6) between the two reference frames can be written using the orthonormal rotation matrix R^S_I and the origin displacement $\vec{p}^S_I$ between frames S and I as

$$[\vec{p}_i]^I = R^S_I \, [\vec{p}_i]^S + \vec{p}^S_I \; .$$

R^S_I and $\vec{p}^S_I$ are optimized by a least squares approach, minimizing the quadratic error

$$\frac{1}{n} \sum_{i=1}^{n} \left([\vec{p}_i]^I - R^S_I \, [\vec{p}_i]^S + \vec{p}^S_I \right)^T \left([\vec{p}_i]^I - R^S_I \, [\vec{p}_i]^S + \vec{p}^S_I \right) \tag{2.11}$$

between the point set $[\vec{p}_i]^I$ and the result of mapping the point set $[\vec{p}_i]^S$ into frame I. With the mean vectors

$$\overline{[\vec{p}]}^I = \frac{1}{n} \sum_{i=1}^{n} [\vec{p}_i]^I \quad \text{and} \quad \overline{[\vec{p}]}^S = \frac{1}{n} \sum_{i=1}^{n} [\vec{p}_i]^S \; ,$$

origin displacement can be eliminated from the minimization problem yielding

$$\frac{1}{n} \sum_{i=1}^{n} \left([\vec{p}_i]^I - \overline{[\vec{p}]}^I - R^S_I \left([\vec{p}_i]^S - \overline{[\vec{p}]}^S \right) \right)^T \left([\vec{p}_i]^I - \overline{[\vec{p}]}^I - R^S_I \left([\vec{p}_i]^S - \overline{[\vec{p}]}^S \right) \right) \; .$$

It can be shown that this is equivalent to maximizing

$$\frac{1}{n} \sum_{i=1}^{n} \left([\vec{p}_i]^I - \overline{[\vec{p}]}^I \right)^T R^S_I \left([\vec{p}_i]^S - \overline{[\vec{p}]}^S \right) = \operatorname{tr} \left(R^{S^T}_I C \right) \; , \tag{2.12}$$

where the matrix C is

$$C = \frac{1}{n} \sum_{i=1}^{n} \left([\vec{p}_i]^I - \overline{[\vec{p}]}^I \right) \left([\vec{p}_i]^S - \overline{[\vec{p}]}^S \right)^T \; .$$

Calculating the singular value decomposition of C given by

$$C = UWV^T$$

with orthogonal matrices U and V and a diagonal matrix W containing the singular values, Eq. (2.12) is maximized if $R_I^S = UV^T$. However, according to [70] and also confirmed after implementation of the calibration approach, sometimes maximization results in a reflection of the point sets, characterized by $\det\left(R_I^S\right) = -1$. But a rotation matrix needs to be orthonormal with a determinant of +1. To account for this, [70] proposes to modify the approach to

$$R_I^S = U \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det\left(UV^T\right) \end{bmatrix} V^T \ ,$$

where the smallest singular value is assumed to correspond to the last element of W.

Having obtained the rotation matrix, the origin displacement can be determined by

$$\vec{p}_I^S = \overline{[\vec{p}]}^I - R_I^S \overline{[\vec{p}]}^S \ ,$$

thus completing all required information for the calibration process. More sophisticated methods exist, some of them are compared in [71], which are better capable of dealing with ill-conditioned data. The approaches differ in the way, how Eq. (2.11) is minimized. However, since input data is well defined for this application, the described method produces very similar results in comparison to other methods such as [72].

2.2.2.6 Implementation Issues

All in-house made software packages were developed using C++. Algorithms were prototyped and tested primarily in MATLAB® whenever possible before they were implemented in C++. The programs are written in a platform-independent manner. This allows them to be compiled on Microsoft® Windows® operating systems (version 2000 or above) using the Microsoft® Visual Studio® development environment, on Linux-based operating systems with the GNU compiler collection GCC and on Linux-based operating systems using the hard real-time framework Xenomai.

The software package *HexGuide* needs to provide a stable and predictable environment with respect to timing, so that prediction algorithms and control methods can rely on equidistant sampling intervals to avoid timing errors. Furthermore, the application itself should constrain introduced latencies as much as possible. Therefore some general guidelines were followed during the development:

- No memory allocation during runtime: Memory allocation is quite a time consuming task and is performed completely during program initialization. This also enables the immediate compatibility with the Xenomai real-time system where no memory allocation is allowed in the real-time domain.

- Independent of the operating system, the program is run with higher scheduling priority in order to supersede the scheduling needs of normal applications. If *HexGuide* or a thread of it is runnable and the scheduler of the operating system decides which thread is granted some CPU time, then higher prioritized threads or processes are preferred. This improves the responsiveness of the application to external events, e.g. sensor data arrival.

- Faster code is favored over smaller code in order to constrain software-induced latencies to a minimum. All compiler settings and optimizations are configured to this end.

Each operating system possesses its own peculiarities which can affect the scheduling and timing of the application [73]. These are discussed below:

Windows Like any non-real-time operating system, Windows does not guarantee any timing or scheduling constraints. However, some improvements can be achieved by increasing the timer resolution. The standard timer resolution on most PC hardware is usually 10ms or 15ms. The highest resolution on PC systems is about 1ms. The exact value depends on the underlying hardware, where values of 0.9765ms, 0.9766ms and 1.0ms have been observed on different CPUs. The timer resolution determines the minimum time, the so-called quantum, for which a thread may execute without any interruptions by the scheduler (nevertheless hardware or software interrupts are still handled). Reducing the quantum to the minimum time (in which case the maximum resolution is achieved), makes the system more responsive to events because threads can be scheduled at a much finer timing granularity. This reduces the average latency between arrival of new sensor data and handling of that data. However, the system load system rises a little due to more frequent context switches between several other threads sharing one CPU.

Windows does not directly allow locking all current and future memory pages of a process to physical memory. However, it allows to define a process working set size which is the number of bytes of the process' memory the memory manager tries to keep in physical memory without swapping it out into the page file. This still gives no guarantees but at least reduces the chances of memory page faults whose handling negatively affects the timing of applications.

Linux In the same way as Windows, Linux per se also does not provide any scheduling or timing guarantees. Improvements can be gained in a similar way. The timer resolution can unfortunately not be influenced during run-time, but can be configured into the kernel (the resolution is given in terms of frequency, usually in the range between 100Hz (10ms) and 1000Hz (1ms). As a consequence of this, the kernel has to be manually compiled on a Linux system as distributions usually have a lower default frequency value set in the kernel.

The memory manager in Linux can be instructed to lock all current and future memory pages of a process to stay in physical memory rather than being moved to the swap filesystem. In contrast to Windows, this behavior is guaranteed in Linux.

Linux with Xenomai for Hard Real-Time Capabilities Xenomai[1] [74] is a free real-time framework, that brings hard real-time support to GNU/Linux. It is based on the ADEOS (Adaptive Domain Environment for Operating System) nanokernel which is injected into a standard Linux kernel by an appropriate kernel patch. It introduces multiple domains with different priorities which share the same hardware resources. The domains are used to

---

[1] `http://www.xenomai.org`

shield interrupts from being handled by the Linux kernel. This allows that the Linux kernel does not preempt real-time tasks, so that timing constraints can be met. Xenomai provides various interfaces, called skins, so that porting already existing real-time or non-real-time applications to Xenomai is simplified without rewriting the whole application. This feature made it attractive to adopt Xenomai here, since most of the existing code was already POSIX compliant which is a supported skin in Xenomai. Xenomai not only provides real-time support in kernel-mode, but also in user-space applications. Furthermore, it supports the real-time driver model which allows development of drivers for hardware interfaces that can be used in real-time applications. Up to now, a 16550A universal asynchronous receiver/transmitter (UART) serial driver, a hard real-time network protocol stack called RTnet[2] supporting popular network cards and real-time driver for interfacing with the CAN-bus have been implemented.

Within one application, real-time (RT) tasks and non real-time (NRT) tasks can coexist simultaneously. A real-time task is normally executed within the primary domain, meaning that it is scheduled by the real-time scheduler without any interference by the Linux kernel or its scheduler. If a real-time task issues a system call which cannot be handled by the primary domain (e.g. memory allocation, input or output of devices with non-real-time drivers behind), then the task is switched to the secondary domain, in which also the normal Linux kernel is executed. In this case, the task can be preempted by the Linux kernel and no scheduling guarantees can be met. In order to avoid that, it has to be ensured that only appropriate system calls are used and for the situation that a communication between RT and acNRT task needs to be established, appropriate methods for inter-process communication need to be used which are supported by the real-time driver model.

Within *HexGuide*, each complete cycle of the control loop is executed within one RT task (the task containing the trigger of the system). This involves the prediction of future tumor positions, generation of the reference trajectory and the calculation of the control input to the HexaPOD. The serial transmission of the control input to the HexaPOD is performed within the RT domain as well because of the RT-capable serial driver of Xenomai. The RT task is executed in a special mode, which locks the RT scheduler, guaranteeing that this task is not preempted by the scheduler for executing any other runnable task until the scheduler lock has been released or the task begins to wait for a condition. This specific feature enables *HexGuide*, once all available sensor data has been acquired for the current sampling instant, to perform a full control cycle without any interruptions by the scheduler (except the handling of hardware interrupts within the RT domain). Compared to standard Linux or Windows where the preemption is reduced due to above mentioned means, Xenomai can guarantee it, leading to a more predictable system in which timing latencies introduced by the operating system can be minimized.

The timing accuracy becomes apparent when a thread needs to be scheduled in regularly spaced intervals like the internal trigger in *HexGuide*. Therefore *HexGuide* was configured to use the internal trigger at different rates and executed on the various operating systems in the same way. During this experiment measurements of the internal trigger's interval time were acquired. Table 2.3 shows exemplary results of mean and standard deviations of the interval time during a test run on the same hardware. While Windows XP already provides good timing accuracy, Linux shows more precise timing. The mean is located closer to the desired value of either 50ms, $33.3\overline{3}$ms or $16.6\overline{6}$ms and the standard deviation (STD) is lower. With Xenomai the mean interval time possesses an accuracy in the

---

[2] `http://www.rtnet.org`

Operating system	Rate of internal trigger					
	20Hz		30Hz		60Hz	
	MEAN	STD	MEAN	STD	MEAN	STD
Windows XP	50.00033	0.39107	33.33312	0.33214	16.66688	0.24320
Linux	50.00003	0.13235	33.33336	0.15544	16.66672	0.18773
Linux+Xenomai	50.00000	0.00177	33.33333	0.00178	16.66666	0.00126

Table 2.3: *Timing comparison of interval time of internal trigger between different operating systems measured on the same hardware; all values are given in milliseconds*

nanosecond range and a standard deviation of less than two microseconds which is two orders of magnitude lower than for the NRT operating systems.

Because of the timing accuracy in combination with the scheduler lock, Xenomai provides a well-suited environment for the tasks of *HexGuide*. Thus, Linux with Xenomai is the preferred environment.

However, Xenomai currently does not possess a working support for USB. In order to make use of the full RT capabilities of Xenomai when using the Polaris Spectra IR camera, which comes only with USB support, an additional interface is required to convert from USB to a standard serial interface. Since the camera needs to be connected to a USB host controller, standard USB to serial adapters are not usable as they only possess a USB device controller. Therefore, a small Arduino-sized development board VNCLO-MB1A from Future Technology Devices International Ltd., Glasgow, United Kingdom equipped with Vinculum-II (VNC2), clocked at 48MHz, is employed (see Figure 2.16). Apart from an USB host controller integrated into the VNC2, it also possesses a standard UART interface up to 6Mbps to which a MAX13433E RS-485 transceiver from Maxim Integrated Products Inc., San Jose, California, USA, with a maximum communication

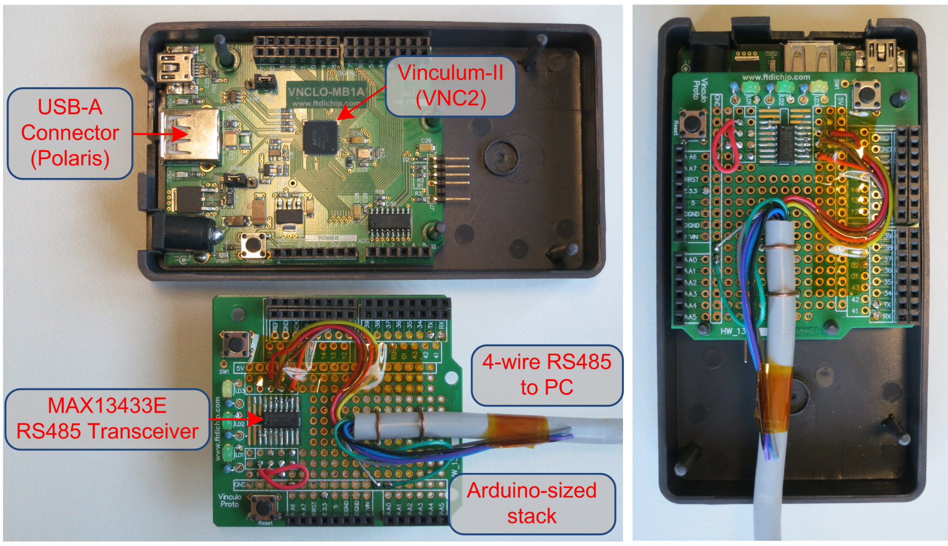


Figure 2.16: *Arduino-sized development board VNCLO-MB1A with Vinculum-II (VNC2) chip and customized stack with RS-485 transceiver*

speed of 16Mbps is connected. This combination enables connecting the Polaris camera using its maximum transfer rate of 1.2Mbps to the integrated USB host controller while the developed device itself can be connected to a RS-485 serial board located in a PC. For this work, the MOXA CP-134U serial card from Moxa Inc., Brea, California, USA is used on the PC side which can be accessed using the Xenomai RT serial drivers. Apart from two RS-485 ports, it supports two additional RS-232 ports. The maximum transfer rate of the MOXA card is, with 921.6Kbps, slightly lower than the one of the Polaris camera. The firmware, which was written for the VNC2 using an RT operating system specifically designed for VNC2, takes this into account by allowing interleaved sending and receiving on both the UART and USB host controller with different rates to minimize the effect of induced latencies.

2.2.3 Latency Determination and Compensation

In order to quantify the latency of the Polaris Spectra IR camera and the HexaPOD, several experiments were conducted. A microcontroller board (cf. Figure 2.17) based on a Crumb128 module from chip45.com, Linden, Germany was used, which was initially developed for a different project. The Crumb128 module carries an Atmel ATmega128 microcontroller from Atmel Corporation, San Jose, California, USA, running at 14.7456MHz and provides a MAX3221 RS-232 transceiver from Maxim Integrated Products Inc. for enabling RS-232 communication between the integrated UART and a serial port of a PC which is connected to the Sub-D-9 connector.

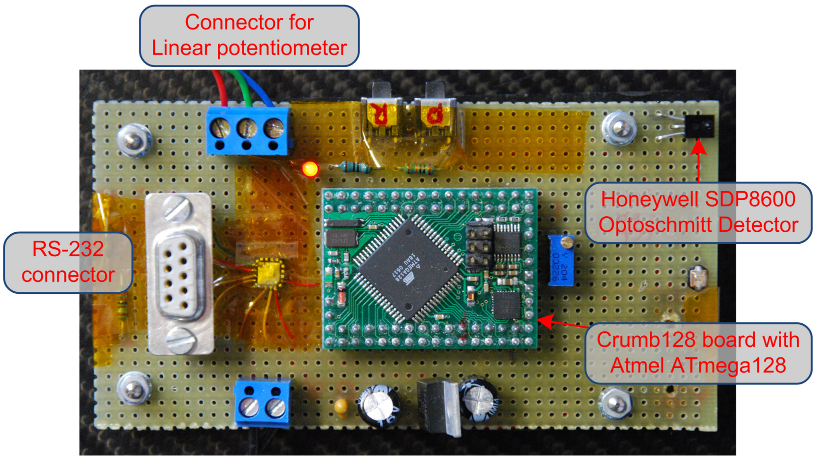


Figure 2.17: *Microcontroller board with Atmel ATmega128, IR light detector and connector for linear potentiometer used for latency measurements*

For measuring latency of the Polaris camera, the board was equipped with a Honeywell SDP8600 optoschmitt detector from Honeywell International Inc., Morristown, New Jersey, USA. The SDP8600, which is connected to an input pin of the microcontroller, consists of a photodiode sensitive to infrared light, a transistor and a Schmitt trigger.

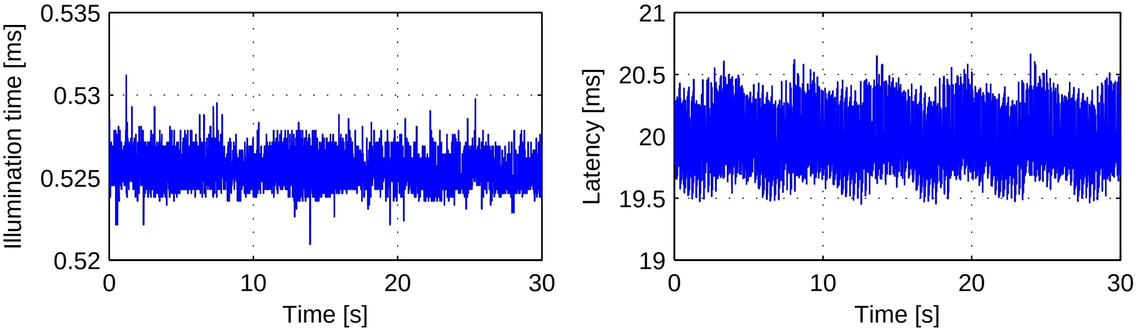
The latency of the HexaPOD was estimated using a linear sliding potentiometer which is connected to an analog to digital converter (ADC) input of the microcontroller.

All latency measurements were performed in the following setup: The microcontroller board was connected with a serial cable to one of the RS-232 ports of the MOXA serial card on the PC. In order to reduce transmission delays, the serial port was configured to operate at a rate of 460.8kpbs (as opposed to the maximum of 115.2kbps for a standard 16550A compatible UART normally found integrated into PCs), with which reliable RS-232 communication is still possible using a short cable. The software developed for the ATmega128 sends either a one byte packet (for Polaris latency determination) or two bytes (for HexaPOD latency estimation) to the PC when a measurement is complete. *HexGuide*, running with Xenomai on the PC, was extended in order to timestamp (using the most precise clock available) and to log the incoming measurements. Simultaneously, it interfaces with the Polaris camera to timestamp and log the tool transformations or it connects to the HexaPOD to send a series of predefined commands and to log the associated timestamps. For all timestamps acquired during latency determination, it is important to use the same clock source in order to enable comparisons of the timestamps.

2.2.3.1 Latency Measurement of the Polaris Spectra IR camera

The pin to which the SDP8600 optoschmitt detector is connected, triggers an interrupt either on falling or rising edge, enabling detection when the IR emitting diodes of the Polaris camera are switched on and off. Using an interrupt additionally guarantees the least possible latencies induced by the microcontroller. The total latency of the measurement system itself from detection of IR light to reception of the measurement packet from the microcontroller on the PC is as follows: The SDP8600 possesses a typical propagation delay of 5.0µs and an output rise time of 60ns (the output fall time is 15ns). The microcontroller triggers an interrupt when the pulse of the output of the SDP8600 is at least one clock cycle in duration. The execution of the code from the jump to the interrupt service routine to the start of the serial transmission of the one byte packet takes 111 clock cycles (determined by summing up the clock cycles of the instructions found in the assembler listing of the relevant code parts), corresponding to 7.53µs. The transmission of 10bits (one byte plus start and stop bit) at 460.8kbps is 21.70µs. The total latency of 34.36µs is assumed to be constant, by which the acquired timestamps on the PC are corrected.

Calculating the time difference between consecutive samples associated to IR light on and off delivers the illumination time depicted in Figure 2.18(a). With a mean value of 0.5253ms and a standard deviation of 0.0012ms the illumination time is quite constant. The measurement latency is the difference between the timestamp of transmission-start of the Polaris samples (calculated from the timestamp on reception subtracted by the serial



(a) Illumination time

(b) Latency between illumination and transmission of Polaris measurement

Figure 2.18: *Results for latency measurements of Polaris Spectra IR camera*

transmission time) and the timestamp of the physical measurement (the mean timestamp between IR on and off). The latency of the Polaris system (excluding serial transmission), exemplified in Figure 2.18(b), has a mean value of 19.9561ms in the range from 19.4510ms to 20.6638ms and a standard deviation of 0.2791ms. The latency variation of about 1ms is a result of the behavior of the USB host controller which reads data only in a timed fashion using a latency timer which defaults to 16ms. In this work it is set to the minimum value of 1ms, so that the smallest possible latency is introduced.

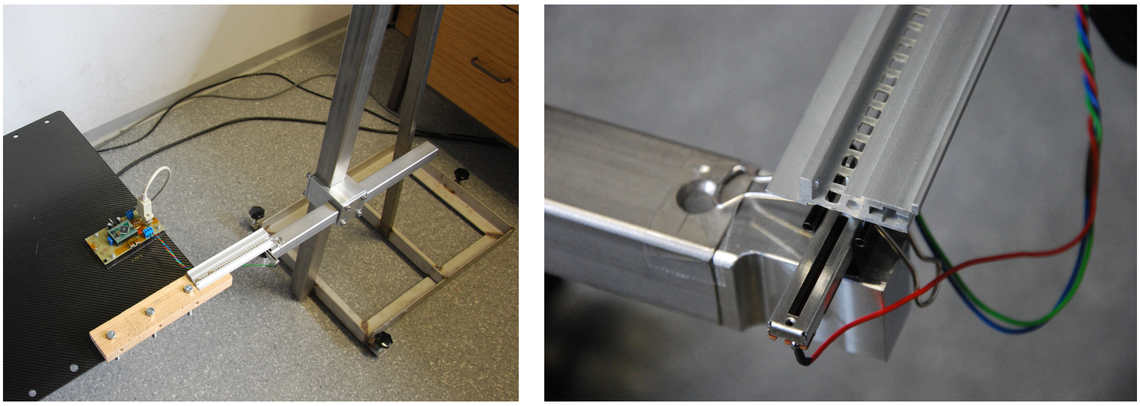
If the Polaris update rate is set to 20Hz or 30Hz, the latency measurements deliver unique values, which can additionally be verified when the tracking mode of the Polaris camera is immediately deactivated after one Polaris measurement. Since the average latency is larger than the smallest sampling period of 16.667ms, latency measurements are not unique for an update rate of 60Hz. In this setting, acquired latency values move in the range of 2.7843ms to 3.9971ms. Due to the fact that latencies are the same for 20Hz and 30Hz, it is assumed that they do not depend on the used update rate. Furthermore, when adding the duration of one sampling period of 16.667ms to the acquired latencies in the low millisecond range, the resulting latency values are in the same range as for an update rate of 20Hz or 30Hz. Hence, for 60Hz, the Polaris camera works in an interleaved mode in which the scene is illuminated already while the previous measurement is still processed. When assuming this operation mode, the Polaris camera possesses a constant measurement latency irrespective of the selected update rate (apart from the USB-induced variations of less than 1ms).

2.2.3.2 Latency Estimation of the HexaPOD

Precise measurements of the HexaPOD's latency are difficult to realize, especially with low-cost equipment. While the point in time of sending a command from a control PC can be determined accurately, estimation of point in time when the HexaPOD starts moving in response to that command is a challenging problem. This stems from the fact that sufficient resolution both in time and in terms of the measured quantity needs to be provided by the employed sensor. In order to achieve sufficient timing accuracy, time resolution of the sensor should be considerably smaller than 1ms. For position-based sensors, resolution in position should be at least in the µm range to reduce resolution-induced timing errors. Several types of sensors, such as inertial measurement units and acceleration sensors, have been considered for the purpose of latency estimation of the HexaPOD. They either suffered from insufficient timing resolution or insufficient sensitivity to recognize even sharp changes of motion. Hence, latency estimation was performed as described below.

The overview of the setup for latency estimation is shown in Figure 2.19(a). A wooden bar was rigidly attached to the HexaPOD's tabletop with three screws. An aluminum bar with quadratic holes was fixed to the wooden bar. The size of the holes is big enough to support the cone-shaped slider of the linear potentiometer without backlash as depicted in Figure 2.19(b). The potentiometer is held by a clamp screwed to a heavy metal stand. By rotating the complete stand or the wooden bar by 90° the latencies along all three translational axes can be measured.

The potentiometer EWAQ1 from Panasonic Corporation, Osaka, Japan, with a travel distance of 60mm is connected to a 10bit ADC input of the microcontroller. If the slider of the potentiometer is in one of the extreme positions, the smallest ADC value observed is 15. The largest ADC value of 1023 is reached if the slider is 3.5mm before the other



(a) Overview during measurements along the x-axis

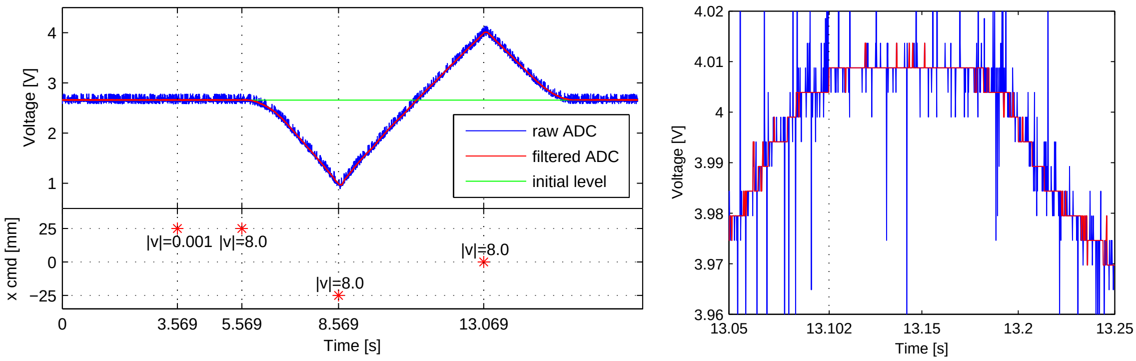
(b) Close view during measurements along the y-axis

Figure 2.19: *Setup for estimation of the HexaPOD's latency*

extreme position. This leaves an effective measurement range of 56.5mm which is covered by 1009 ADC values, hence resulting in a position resolution of 56.00µm.

On the microcontroller one ADC conversion takes 13 ADC clock cycles, where the ADC is clocked at 115.200kHz, corresponding to 112.85µs. Half of that duration (56.43µs) is taken into account for measurement latency. The execution of the code from the jump to the interrupt service routine of the ADC complete interrupt to the start of serial transmission of the two byte packet takes 151 clock cycles, corresponding to 10.24µs. The transmission of 20bits (two bytes plus start and stop bits) at 460.8kbps is 43.40µs. Also here, acquired timestamps on the PC are corrected by the total latency of 110.07µs. Each ADC conversion is started using a timer of the microcontroller which is executed at a rate of 6kHz in order to achieve a good time resolution of 166.667µs.

For determining latency of the HexaPOD along one axis, the HexaPOD was moved within the effective measurement range along the chosen axis by "MOV" commands. The timestamps, when these commands were sent, were logged as well as all digitized voltages of the potentiometer which correspond to a certain position of the HexaPOD. Both logged values are depicted in Figure 2.20(a) which exemplifies the measured results along the x-axis for latency estimation. The raw ADC measurements converted to voltage are shown in blue. Due to the high frequency polling, ADC delivers quite noisy measurements. However, these are represented by discrete values where most of the noisy measurements are characterized by outliers which only last for one sampling period, so that the discrete values before and after an outlier are equal. Exploiting this characteristic, a simple and effective filtering method can be established by comparing the current sample value to the previous and next ones. If the current and previous values differ from each other but the previous and next values are equal, the current one is set to that value. Applying this filter results in the curve shown in red in Figure 2.20. From this, the point in time was manually chosen when the HexaPOD was deemed to show a change in the filtered ADC values as depicted in Figure 2.20(b). The difference in time from the chosen timestamp to the logged timestamp of the command corresponds to the latency of the HexaPOD. Of course this manual process is subject to errors in judgment. Hence, each complete trial like the one shown was repeated three times and performed along all translational axes. Furthermore, two different set of trials were used. In the first set, the first command is one



(a) Logged ADC measurements converted to voltage for x-axis in upper part; commanded position for x-axis annotated with speed in lower part; time of command is shown as dashed vertical line

(b) zoomed in cut-out of ADC measurements, dashed vertical line shows manually chosen timestamp indicating the first effect of a command

Figure 2.20: *Examples of results for HexaPOD latency determination*

with minimum velocity of 0.0011mm/s as shown in Figure 2.20(a). The second command makes the HexaPOD move into the same direction but with a larger speed of 8mm/s. In the second set, the initial command with minimum velocity is not executed, so that the first command already sets the HexaPOD in motion.

The individual results for each of these trials on the HexaPOD are given in Table 2.4. Column C1 specifies the latency for the first command with larger velocity than the minimum one, i.e. the latency of the initial command with minimum velocity is not evaluated as it cannot be seen in the data (cf. Figure 2.20(a)). Column C2 gives the latency for the second command which makes the HexaPOD change direction while the HexaPOD is moving with maximum speed. Finally column C3 results as latency to the third command which makes the HexaPOD move into its zero position, also moving with maximum speed before. Column C1 specifies the latency for the first command with larger velocity than the minimum one, i.e. the latency of the initial command with minimum velocity is not evaluated as it cannot be seen in the data (cf. Figure 2.20(a)). Column C2 gives the

Axis	Trial	HexaPOD resting			HexaPOD minimum velocity		
		C1 [ms]	C2 [ms]	C3 [ms]	C1 [ms]	C2 [ms]	C3 [ms]
X	1	211.7	36.6	29.5	247.1	27.8	24.9
	2	213.9	27.3	26.6	242.7	30.9	26.9
	3	207.6	38.2	27.2	207.1	30.5	26.5
Y	1	145.1	24.0	24.5	137.1	23.2	24.0
	2	138.7	23.7	23.4	145.0	23.1	24.6
	3	137.8	23.2	22.4	140.9	23.9	23.1
Z	1	104.2	23.3	21.3	103.2	26.3	21.7
	2	116.0	23.9	22.9	121.3	24.7	23.1
	3	111.0	26.5	21.9	111.0	24.7	24.8

Table 2.4: *Results of HexaPOD latency estimation*

latency for the second command which makes the HexaPOD change direction while the HexaPOD is moving with maximum speed. Finally column C3 results as latency to the third command which makes the HexaPOD move into its zero position, also moving with maximum speed before.

Observed latencies reveal that there is no significant difference in the latencies when the HexaPOD is initially at rest or initially moving with minimum velocity as the latencies for C1 are located in a similar range. However, they are about an order of magnitude larger for the x-axis and about half an order of magnitude larger for the y- and z-axis compared to C2 and C3 when the HexaPOD is moving with maximum velocity. Of course, given a position resolution of 56µm it is harder to estimate from the data when the HexaPOD has started its motion if the HexaPOD is initially at rest or moving with minimum velocity, in comparison to the maximum velocity case. Nevertheless, the latencies are significantly different in both cases. Furthermore, the latencies for C1 also differ depending on the axes, where rising latencies are found for the z-, y- and x-axis in that order. For C2 and C3, the latencies are rather independent of the axis, with slightly larger latencies for C2 along the x-axis. Mainly, the latencies when the HexaPOD is already in motion are distributed in the range 21ms to 29ms.

A different situation is observed when the same measurements are performed on the HexaPOD evo. The results given in Table 2.5 indicate slightly larger latencies when the HexaPOD is in motion where most of them are distributed in the range between 25ms to 39ms for the x- and z-axis, whereas for the y-axis different latencies of about 52ms are observed for C2 alone. However, when comparing the latencies with and without initial command the most noticeable difference is seen for C1. Without initial commands, latencies of about 1490ms are observed for the x- and y-axes, with latencies of about 606ms for the z-axis. With initial commands, latencies reduce to about 66ms for x-axis, 71ms for y-axis and 48ms for z-axis. The reason for this huge difference is originating from brakes which were introduced into the HexaPOD evo. Without an initial command with slow velocity, the brakes have to be released before the HexaPOD can start moving, taking 0.5s up to 1.5s. However, with an initial command the latencies when the HexaPOD evo is moving with minimum velocity are significantly smaller than for the previous HexaPOD version. As a consequence of this, if the HexaPOD evo is used, an initial command

Axis	Trial	HexaPOD evo resting			HexaPOD evo minimum velocity		
		C1 [ms]	C2 [ms]	C3 [ms]	C1 [ms]	C2 [ms]	C3 [ms]
X	1	1488.8	32.5	35.2	64.0	34.1	38.5
	2	1485.1	36.9	39.3	67.3	32.4	35.8
	3	1492.0	34.8	39.7	68.3	32.8	37.7
Y	1	1491.5	52.2	29.2	71.8	53.8	31.6
	2	1493.4	52.1	29.1	70.4	50.1	29.5
	3	1492.8	52.3	28.2	72.2	51.3	28.9
Z	1	605.9	27.3	26.2	46.9	24.6	27.4
	2	606.3	29.3	27.2	49.3	27.0	25.4
	3	606.5	29.4	28.4	49.1	29.6	26.9

Table 2.5: *Results of HexaPOD evo latency estimation*

with minimum velocity must be sent to the HexaPOD evo prior to any other positioning command. The same command can be used for the HexaPOD as well, however, this does not influence the latencies.

To summarize, latency estimation of the HexaPOD was successfully performed. General latencies when the HexaPOD is in motion fall into the range between 21ms to 39ms depending on the actuated axis and version of the HexaPOD. Increased latencies need to be expected, when the HexaPOD is moving with minimum velocity (or if the HexaPOD is resting with zero velocity).

2.2.3.3 Latency Compensation

In order to realize a true real-time system, the known latencies should be compensated for. Therefore the following procedure is applied within *HexGuide* to compensate for almost all soft- and hardware-induced latencies: The timestamp of each sensor measurement for either breathing or tumor position determination is corrected by the measured or estimated age of the measurement (if the timestamp does not include this information already). In case of the Polaris IR camera, the timestamp acquired when a measurement has been fully received on the computer is corrected by 19.9561ms plus the serial transmission time (cf. Section 2.2.3.1). After reception of the sensor data, several processing steps are undertaken such as interpolation, peak detection, moving average determination. The duration of these as well as the computational time for the selected prediction method (computational times are given in Section 3.7) is measured with the most precise clock available. After the predictor has finished its work, this latency is compensated for by an estimation of the current (where current refers to the actual point in time right when the estimation is performed) position of the HexaPOD since the HexaPOD has moved on in between the physical measurement of its position and the actual time. Hence, the whole latency from physical sensor measurement to computations of predictions is compensated for. The following computation time of the control scheme (which is usually less than 1ms) is ignored. Furthermore, the latency of the HexaPOD is not compensated for due to its varying characteristics, although it can be smaller and larger than the duration of one sampling time. Variations of the treatment and consequences of latencies are also depending on the requirements and flexibility of controllers. These are covered during the discussion of the individual controllers presented in Section 4.2.

2.3 Patient Safety Features

In order to ensure safe treatment for the patient, the ATTS can be equipped with safety features as described below:

Phase-dependent Tumor Location Region The ATTS requires monitoring of tumor positions using some sensor system such as EPID or Calypso during the course of a whole fraction. With this information available, a region can be defined in which the tumor is allowed to move. If the tumor leaves this region, i.e. the error compared to the desired position of the tumor becomes too large, then irradiation can be shut off. However, since any motion compensation method requires a certain time for convergence, the region should be defined in terms of time. One approach is to define four phases of motion compensation in which the maximum allowed error is reduced so that medical requirements are still met. The first phase starts with the beginning of a fraction at a time where no tumor position measurement is available. As long as this measurement is not available, the motion compensation system cannot perform any action. This phase should last only for a very short time, ideally a fraction of a second and it ends when the first measurement has arrived. In the second phase, the initialization of prediction algorithms is started, but no prediction is available yet, so that no reference trajectory can be generated for use by a control method. But the negated tumor position measurements can be treated as set-points for the HexaPOD and the selected control method can be instructed to move the HexaPOD to this set-point. As long as the system is in the second phase, this can be repeated each time a new measurement arrives. The HexaPOD then follows a trajectory which neglects the system latencies and dynamics of the HexaPOD, but still allows the best possible compensation strategy with a significant reduction of tumor motion given the available information. In the third phase, starting when a tumor prediction becomes available for the first time (initialization or learning phase of prediction algorithm is finished), a transition takes places depending on the convergence properties of the prediction algorithm and of the control method. After the transition phase, the final phase is entered in which the algorithms have converged (except when phase or amplitude changes occur which might require a short re-convergence time).

Obviously, in the first phase no constraints on the maximum allowed error of tumor motion can be stated. During the second phase, the maximum errors can be significantly constrained, however transition errors when the HexaPOD starts to move have to be taken into account here. In the third phase, the maximum allowed errors can be further reduced, whereas the maximum allowed errors are minimal during the fourth phase. In this way, a quality assurance method of the tumor motion compensation system can be realized which is constantly monitored during the course of a treatment field. Thus, the treatment can be interrupted any time the defined tumor location region is violated.

Status Monitoring Using a watchdog mechanism, status of the activated sensors and HexaPOD can be monitored. A timeout is defined for each sensor after which a new sample needs to arrive. This can be performed using hard constraints where a single violation of the timeout leads to an error or using soft constraints which need to be violated a certain number of times before an error condition is raised. The HexaPOD can be handled in the same way as it gives a response to each command after a certain amount of time. If the HexaPOD fails to respond with an answer before the timeout expires, *HexGuide* can enter an error state.

Actually, the whole processing in *HexGuide* can be monitored and timeouts can be defined between important steps like the prediction or the computation of the control input. It is thereby important that the watchdog, usually realized as a separate process, is running with a higher priority than all other processes in the system in order to avoid starvation of the watchdog process even when other processes put the system under more load.

Plausibility Checks All incoming sensor data as well as all generated data like tumor prediction and control input should be checked for plausibility. This can be achieved by defining constraints on maximum movement velocities of breathing or tumor motion. In this respect, from one sampling instant to the next only a certain limited change of each variable can be expected. This can be realized using either hard or soft constraints, where the latter can be violated a certain number of times consecutively.

2.4 Summary

The ATTS is a classical robotic system [75]. It possesses sensors to characterize the environment (acquisition of breathing and tumor motion information). It contains a processing unit which interprets and processes sensor data (motion prediction). Based on this, closed-loop control decides on the control input which is applied to an actuator (HexaPOD) which interacts with the physical world in order to achieve a defined task (motion compensation).

By inclusion of the HexaPOD, EPID and an IR camera, the architecture of the presented tumor motion compensation system allows using standard hardware devices instead of specialized expensive designs as in other fully integrated approaches. This enables flexible treatments by imposing no constraints on established treatment procedures and beam types. Simple and cost-efficient updates of existing treatment rooms are possible with this system, where the important functionalities for motion compensation and the intelligence of the system is realized through software components. As real-time operation of the system is a key feature for successful motion compensation, a real-time operating system is employed to realize timing guarantees within the software applications. The measurement latencies of the sensor systems were determined to acquire the true time of the physical measurement. This enables precise timing of all data acquired and generated by the system. In turn, this allows compensation of software-induced latencies to further improve timing conditions. This is especially important for a motion compensation system since the software interacts with the physical world by actuating the HexaPOD. Any non-compensated latency can therefore induce positioning errors thereby reducing overall system performance. Several important aspects of the software were discussed for supporting the development of prediction methods and control algorithms. Patient safety features complement the system to enable safe treatment procedures.

Chapter 3

Breathing and Tumor Motion Prediction

As any mechanical robotic system such as the HexaPOD possesses a dynamic response including latencies, successful motion compensation can only be achieved if the near future of the motion trace to be compensated is known. Therefore predictors are developed to accommodate for that need. Furthermore, this knowledge also enables the adoption of advanced control methods for the HexaPOD which explicitly take into account future horizons to perform optimizations.

In order to approach breathing and tumor motion prediction, breathing and tumor motions are characterized in terms of amplitudes, frequencies and speeds based on a large number of clinical datasets. The correlation between breathing and tumor motion is analyzed, laying the foundation for tumor motion prediction in the presence of an undersampled tumor motion signal. The prediction problem is stated along with a detailed description of several metrics for prediction algorithms serving as performance measures. Three different classes of prediction algorithms are identified, where each class refers to available inputs in terms of breathing and/or tumor motion signals. It is shown, how the correlation analysis can be exploited to setup prediction algorithms making use of both signals. Several prediction algorithms are presented along with a detailed description and analysis how their free parameters can be chosen in order to achieve reliable predictions. Where applicable, extensions to prediction algorithms are proposed and evaluated fusing both breathing and tumor motions to enable improved tumor predictions compared to the developed correlation schemes. This chapter concludes with a comparison of the presented prediction algorithms detailing the main characteristics of achieved predictions.

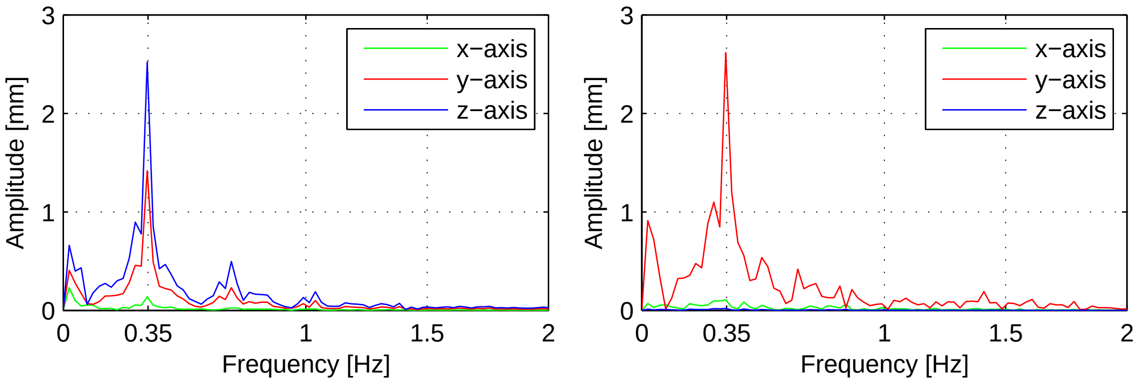
3.1 General Properties of Breathing and Tumor Trajectories

Apart from sources like heartbeat and spontaneous muscular or patient motions, lung tumor motion is mainly determined by breathing, which appears as a quasi periodic pattern [36, 76]. Tumor motion amplitudes reported in literature are usually below 10mm [77], in extreme cases up to 50mm [78] with breath-by-breath frequencies between 6 to 31 breaths per minute [79] (corresponding to 0.1Hz and 0.52Hz, respectively). The shape of motion traces often depend on the relative location of the tumor to internal organs [80].

During a period of about two years, breathing and tumor motions were simultaneously recorded by the sensors of the ATTS described in Section 2.2.1 in order to gain knowledge about motion properties and to elaborate its characteristics. Live data from 24 patients were recorded during normal treatments, usually with several fractions where each fraction consists of several fields (for each field the gantry angle was varied). Respiratory data was gathered by the Polaris IR camera from an IR tool which was placed on an arbitrarily chosen free space on the patient's abdomen and patients were asked to perform abdominal breathing. Tumor motion data was acquired by recording portal images and performing automated tracking of manually defined masks with *PortalTrack*. Obtained 2D projections of tumor motion were then rotated by the gantry angle to match the same reference system as the calibrated reference system for respiratory data. Thus, tumor motion data originates from 2D projections, although it is given in a 3D reference system. Note that all tumor motion data used in this work does not contain information about tumor motion in the beam's direction. Further note, that this data was acquired before this work started, so that no influence on the measurement and recording process could be taken. Hence, this work uses the data as it is. A total number of 200 breathing and tumor motion datasets were obtained, whereby for three patients only one field is available. The duration of these datasets varies between 16.9s and 124.7s with an average of 54.2s.

Typical examples of breathing and tumor motions are depicted in Appendix A, in Figures A.1 – A.4. From visual inspection, periodicity of breathing motion - and with it also tumor motion - becomes obvious, but with hysteresis effects [80]. These result from the fact that resting humans possess longer exhalation than inhalation phases. Additionally, the beginning of exhalation progresses faster than its ending and inhalation is usually faster than exhalation with different speeds.

From amplitude spectra, obtained by the discrete fast Fourier transform (DFFT) and exemplified in Figure 3.1 for one dataset (taken from Figure A.1), a main frequency component of 0.35Hz can be observed which corresponds to the main breathing frequency (cf. Figure 3.1(a)). This is also the main frequency component in the tumor amplitude spectrum (cf. Figure 3.1(b)), leading to the conclusion of a close relationship between breathing and tumor motion. Additionally, a lower contribution of about 0.025Hz found in both spectra suggests slower periodic changes which might be caused by whole body motion or other anatomical motions apart from breathing. Compared to the larger breathing frequencies, these slow motions appear as drift or baseline-shift overlaid on the periodic breathing or tumor signal. An example of such a motion with larger drift can be observed in Figure A.1.



(a) Breathing amplitude spectrum

(b) Tumor amplitude spectrum

Figure 3.1: *Breathing and tumor amplitude spectrum of patient 1, fraction 081205, gantry angle* 190°

3.1.1 Maximum Amplitudes

Regarding breathing amplitudes along different axis, the z-axis (AP) can be inferred from spectrum as dominant axis with largest amplitudes, followed by the y-axis (SI) with slightly lower amplitudes. The x-axis (LR) contributes the least. A different picture arises from the tumor amplitude spectrum, where the y-axis possesses the largest amplitudes compared to x- and z-axis. This is a general property among most patient data supported by the amplitude metrics of all datasets summarized in Tables 3.1 and 3.2. For each dataset, the maximum amplitude range (including an eventual drift which causes larger amplitude values) was calculated along each axis as the difference between the largest and smallest sample values. From them, the average maximum amplitude, its standard deviation, root-mean-square and maximum were determined. With an average maximum amplitude of about 5.87mm and a maximum of 16.76mm, the z-axis is also identified as dominant axis for breathing motion from the amplitude metrics. Similarly, for tumor motions these metrics show the y-axis to be the dominant axis with a mean amplitude of 6.71mm and a maximum of 29.06mm. On average, the maximum tumor amplitudes are larger than the maximum breathing amplitudes. But when evaluating this individually for each dataset, it turns out that this only occurs for 93 datasets. For the remaining 107 datasets, maximum breathing amplitudes exceed maximum tumor amplitudes (regardless of axes). In contrast to the maximum, the root-mean-square (RMS) value, calculated from the mean-centered signal, is a measure of all amplitudes over the signal's duration, proportional to the signal's integral. It is a useful metric since with ideal motion compensation, the RMS value would become zero. The ratios of the RMS values along the different axes are similar to the ratios of the maximum amplitudes, so that the latter already give a good idea of the overall amplitudes to be compensated.

	MEAN	STD	MAX	MIN	RMS
x (LR) range [mm]	0.805	0.623	3.901	–	0.180
y (SI) range [mm]	1.941	1.308	7.459	–	0.465
z (AP) range [mm]	5.873	3.097	16.759	–	1.406
Frequency [Hz]	0.313	0.072	0.582	0.172	–

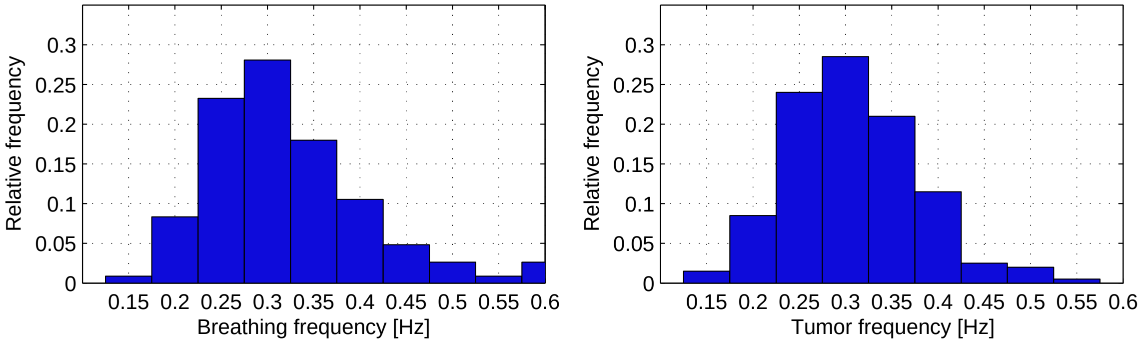
Table 3.1: *Main properties of breathing motion*

	MEAN	STD	MAX	MIN	RMS
x (LR) range [mm]	2.578	1.736	7.819	–	0.580
y (SI) range [mm]	6.705	4.564	29.056	–	1.714
z (AP) range [mm]	2.262	1.441	7.203	–	0.517
Frequency [Hz]	0.309	0.070	0.548	0.170	–

Table 3.2: *Main properties of tumor motion*

3.1.2 Main Frequencies

Statistical metrics of the main frequency were determined by the DFFT applied to the mean-centered breathing and tumor signal of each dataset. The frequency with largest amplitude from the resulting amplitude spectrum was selected as main frequency. In accordance to the values from literature, the minimum frequency is about 0.17Hz, the maximum about 0.58Hz for breathing motion and 0.55Hz for tumor motion. The average frequency is approximately 0.31Hz (cf. Tables 3.1 and 3.2). Note that the maximum frequency resolution Δf_{max} of the DFFT, which is the expected single-sided error during determination of the main frequency, is constrained by the duration T_{data} of the input signal in such a way that $\Delta f_{max} = 1/T_{data}$. For the recorded patient data, Δf_{max} consequently varies between 0.059Hz and 0.008Hz. The distribution of the main frequencies in terms of the relative frequency (in relation to a total of 200 datasets) is visualized in Figure 3.2. The distributions for breathing and tumor frequencies are similar to each other, having lower relative frequencies near the minimum and maximum breathing frequencies and the majority in the range between 0.225Hz and 0.375Hz.



(a) Breathing frequency distribution (b) Tumor frequency distribution

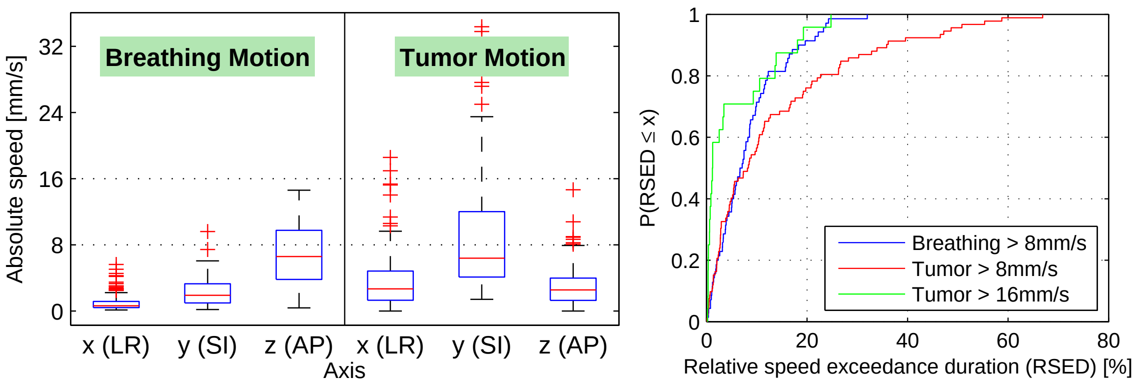
Figure 3.2: *Breathing and tumor frequency distribution among 200 patient datasets*

3.1.3 Maximum Speeds

Aside of maximum amplitudes and frequencies, but closely connected to them, the maximum speed of breathing and tumor motion is also of great importance for tumor compensation systems. Therefore recorded patient data were analyzed to find the maximum absolute breathing and tumor speed of each dataset. The raw signal data was discretized to 50ms intervals. Since it is subject to noise effects, the problem of outliers during speed determination occurs for which discrete derivatives are necessary. In order to reduce this effect and to prevent overestimation of maximum speeds, breathing and tumor signals were filtered with a low-pass filter. The filter was designed with approximately 0dB attenuation in the passband so that amplitudes of pure breathing and tumor motion are not distorted. A Butterworth filter is especially suitable for that purpose since its magnitude response is maximally flat in the passband. A 10th-order Butterworth filter with a cut-off frequency of 1.2Hz was found to be feasible for the patient data. The magnitude response is shaped very flat for frequencies up to 1Hz and for frequencies smaller than 0.8Hz the absolute magnitude is well below 0.005dB. However, a drawback of this filter is its large phase response even at smaller frequencies. To account for this, the filter is applied in forward and reverse direction, effectively resulting in a zero-phase filter. Additionally, initial conditions of the filter's delay elements are estimated before each filtering process to reduce transient effects at the beginning and at the end of the data sequence [81]. With this method the filtered sequence is of the same length as the input sequence (with correct transients) which is important for velocity estimation. If the filter is applied only in forward direction as it is usually done, then the filtered sequence is cut by a number of samples equivalent to the filter order, thus omitting the last samples. After the filtering process using the zero-phase filtering method, the discrete-time derivative of the resulting output sequence is calculated to deliver the signal speed.

Maximum absolute speeds of breathing and tumor motion of all recorded patient datasets are summarized in the box plot in Figure 3.3(a) for each axis. All box plots in this work stick by the following convention: The lower and upper limits of each box are determined by the 25th and 75th percentiles, where the central mark is the median value. The lower and upper whisker extend at maximum by 1.5 times the inter quartile range (IQR), the range between the 25th and 75th percentiles. Data points outside the whiskers are considered outliers and are marked individually. Since breathing and tumor motion occurs synchronously along different axes with the same frequency, a similar observation for speed can be made as for maximum amplitudes. For breathing, the dominant axis is clearly the z-axis (AP) whereas for tumor motion larger maximum speeds can be generally observed for the y-axis (SI). For both motions the median speed of the dominant axis is always larger than for the minor axes. Additionally the lower quartiles of the dominant axis are either larger than the upper quartiles of the minor axes or approximately in the same region, so that at least 50% of the maximum speeds along the dominant axes are larger than 50% of the minor axes speeds. Comparing the speed distributions of the dominant axis between breathing and tumor motion, it can be observed on the one hand, a similar median, and on the other, much larger maximum speeds. While the lower quartiles are located closely together, the upper quartile of the tumor motion is larger than the one for breathing, suggesting that a significant number of datasets possess larger tumor motion speeds than breathing motion speeds. However, an individual analysis of each dataset reveals that for only 107 out of 200 datasets the maximum tumor motion speed is faster than breathing. Averaging this information with the same information from amplitudes (93 out of 200 datasets had larger tumor than breathing amplitudes), it follows that 50% of the datasets have either larger tumor amplitudes or larger tumor speeds, whereas for the remaining 50% breathing motion is either larger or faster.

Maximum tumor speed is one of the main factors constraining the performance of the



(a) Maximum speeds

(b) CDF of speed exceedance duration

Figure 3.3: *Maximum breathing and tumor speeds*

ATTS from the perspective of full motion compensation. The speed limits of the used HexaPOD versions are either 8mm/s (HexaPOD RT) or 16mm/s (HexaPOD evo). For breathing motion, it can be observed from Figure 3.3(a), that the speed limit of the HexaPOD evo is never reached, regardless of the axes, so that the performance of breathing motion compensation would not be constrained by the speed limit. The limit of 8mm/s is not exceeded along the minor x-axis, only for one dataset along the y-axis and for 70 datasets (35% of the datasets) along the dominant z-axis. Note that this exceedance only occurs during the progression from inhalation to exhalation or vice versa and is taking place only during a certain amount of time during breathing cycles, so that exceedance duration is usually significantly less than the duration of the dataset. In order to get a better picture of the quality of exceedance, the cumulative distribution function (CDF) of the relative speed exceedance duration (RSED) (the ratio between the number of samples with exceedance of the limit and the total number of dataset samples) is shown in Figure 3.3(b). The determination of the RSED for each dataset is thereby performed in such a way that it is first calculated along each axis separately and then the maximum of these values is used as RSED in order to consider the worst case. The blue line represents the CDF of RSEDs for breathing over all datasets. The maximum RSED is found to be about 32% (here the value of the CDF becomes one) and the average RSED is 8.3%, which means that if breathing speeds exceed 8mm/s, the average duration of exceedance is 8.3%. In the worst case, the maximum exceedance duration that can be expected is about one third of the dataset's length. Due to the high slope of the CDF for lower values of the RSED, most datasets with exceedance have a relatively low exceedance duration. For example the CDF reveals that 95% of the datasets have a RSED of less than 23.3% or that 66% of the datasets have a RSED of less than 9.2%. This suggests, that exceedance caused by breathing is not considered severe, although a certain performance degradation is to be expected if breathing motion is compensated by the ATTS using the HexaPOD RT. If the HexaPOD evo is used, then performance of breathing motion compensation is not constrained by its' speed limit.

A different picture arises for the tumor speeds. Here, the limit of 8mm/s is exceeded for 92 datasets (46%) and the limit of 16mm/s for 24 datasets (12%). Among the last, only five datasets (2.5%) exceed 24mm/s with a maximum of 34.4mm/s. With y being the dominant tumor axis, the largest and most frequent exceedance can be observed for this axis. Along the x- and z-axes, exceedance of 8mm/s is only observed for some datasets (along z-axis a lesser number than along x-axis) and 16mm/s is slightly exceeded for two datasets along the x-axis, but not any more for the z-axis. As it is obvious from Figure 3.3(a), tumor motion exceedance is larger than breathing exceedance and is of a different quality, as seen in Figure 3.3(b). The initial slope and values of the red line in the CDF for the 8mm/s limit are similar to breathing motion (blue line), especially until a CDF value of 0.5 which is reached for RSED $\leq$ 8.1% (half the datasets with exceedance have a RSED of less than 8.1%). However, since the slope reduces until the maximum RSED of 66.9%, the 8mm/s limit cannot be considered appropriate for a high performance tumor motion compensation system. Nevertheless, significant reduction of tumor amplitudes, eventually clinically sufficient, might be possible with the HexaPOD RT even in such a scenario. But for the limit of 16mm/s (green line), the slope is initially very large (70% of the datasets have a RSED of less than 3.5%) and approximately approaches the one for breathing with the limit at 8mm/s. In this case the maximum RSED is 24.8%.

To summarize, with the HexaPOD RT (8mm/s) breathing and tumor motion compensation is in general constrained by the speed limit for less than half of the datasets but still feasible (according to RSED). A better performance can be expected for breathing than for tumor motion compensation. With the HexaPOD evo (16mm/s), breathing motion compensation is not constrained by the limit and tumor motion compensation is affected in only 12% of the datasets with low RSEDs, so that a decent compensation performance can be expected.

3.1.4 Breathing to Tumor Correlation

For the development of breathing-induced tumor motion prediction algorithms, information about correlation between between breathing and tumor motion is required. Previous considerations on breathing and tumor motion already suggested the existence of a close relationship between these motions for lung tumors, which is analyzed in more detail below.

The first step was the calculation of the correlation coefficient between breathing and tumor motions along different axes. Let a sample of a breathing signal at sampling instant $t \in \mathbb{N}$ be denoted by $p_j^B(t) \in \mathbb{R}$ (assuming equidistantly spaced samples) and the corresponding tumor sample be denoted by $p_j^T(t) \in \mathbb{R}$ where $j = 1, 2, 3$ corresponds to the x-, y- and z-axis, respectively. Then the empirical correlation coefficient between breathing axis l and tumor axis m is defined as

$$R_{l,m} = \frac{\sum_{t=1}^{n} \left(p_l^B(t) - \overline{p_l^B}\right)\left(p_m^T(t) - \overline{p_m^T}\right)}{\sqrt{\sum_{t=1}^{n} \left(p_l^B(t) - \overline{p_l^B}\right)^2} \sqrt{\sum_{t=1}^{n} \left(p_m^T(t) - \overline{p_m^T}\right)^2}} , \tag{3.1}$$

where n is the total number of samples and

$$\overline{p_l^B} = \frac{1}{n}\sum_{t=1}^{n} p_l^B(t) \qquad \text{and} \qquad \overline{p_m^T} = \frac{1}{n}\sum_{t=1}^{n} p_m^T(t)$$

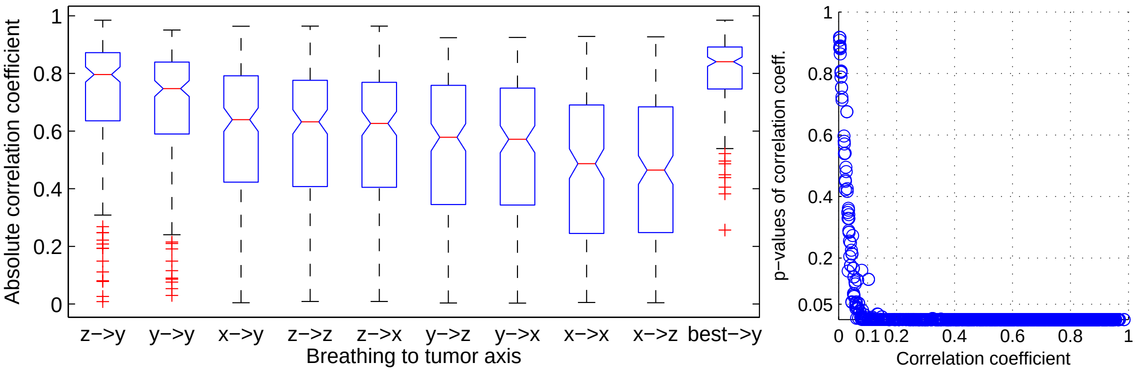
are the arithmetic means of the breathing and tumor signal, respectively.

For all available datasets the correlation coefficient $R_{l,m}$ between breathing and tumor motion was calculated for all combinations of axes l and m. Resulting statistical metrics of the correlation coefficient are given in Table 3.3. One might intuitively assume that the correlation between breathing and tumor motion of the same axis is stronger than for different axes. However, when analyzing the correlation coefficients it turns out that this is only partially true. The strongest correlation can be found for the z↦y axes (this notation first states the breathing axis and after the arrow the tumor axis is given). Since these axes are the dominant axes of the breathing and tumor motion, respectively, this result is expected given previous considerations regarding amplitudes and frequencies of breathing and tumor motions. The less-dominant y-axis of breathing also correlates well with the dominant y-axis of tumor motion. An interesting observation from Table 3.3 is, that the largest three correlations (both in median and mean) are comprised of one of the breathing axis to only the dominant y-axis of tumor motion, in the order of maximum breathing amplitudes per axis. Hence, the significance of breathing motion observed along an axis mainly determines the correlation of this axis with the tumor y-axis. For all correlations, the median is always larger than the corresponding mean, where the difference is approximately proportional to the median or mean value. This results in correlation coefficients

	MEDIAN	MEAN	STD	MAX	MIN	# in best
$R_{3,2}$: z↦y	0.7964	0.7265	0.2131	0.9847	0.0083	85
$R_{2,2}$: y↦y	0.7476	0.6829	0.2157	0.9507	0.0296	45
$R_{1,2}$: x↦y	0.6397	0.5837	0.2500	0.9637	0.0046	24
$R_{3,3}$: z↦z	0.6322	0.5817	0.2477	0.9642	0.0087	14
$R_{3,1}$: z↦x	0.6266	0.5736	0.2458	0.9642	0.0087	10
$R_{2,3}$: y↦z	0.5784	0.5398	0.2574	0.9241	0.0036	6
$R_{2,1}$: y↦x	0.5717	0.5283	0.2541	0.9249	0.0032	6
$R_{1,1}$: x↦x	0.4868	0.4619	0.2634	0.9285	0.0054	5
$R_{1,3}$: x↦z	0.4648	0.4604	0.2581	0.9271	0.0045	5
best↦y	0.8370	0.8015	0.1289	0.9847	0.2564	-

Table 3.3: *Correlation coefficients of breathing to tumor motion over all patient datasets*

being distributed in left-skewed distributions (more occurrences at larger values) if generally larger correlation coefficients arise or being uniformly distributed for correlations with smaller coefficients. This can also be inferred from the box plot in Figure 3.4(a) which gives a quick overview of the distributions of correlation coefficients. Noticeable is also



(a) Correlation overview (b) Correlation significance

Figure 3.4: *Correlation analysis*

that for the z↦y and y↦y correlation, only a few datasets possess correlation coefficients smaller than 0.3. However, all correlations possess similar values for minimum coefficients (mostly in the vicinity of 0.08, suggesting no correlation between breathing and tumor motion) as well as for maximum coefficients (between 0.9241 and 0.9847, implying a strong relationship of breathing and tumor motion).

All this leads to the conclusion, that in general for each patient the best matching correlation needs to be determined individually, since only expectations can be given from this analysis. However, this determination could already be assessed during treatment planning, if for example 4D-CT is used with which a reconstruction of CT images in relation to the position in the breathing cycle is possible. Having once obtained the best correlating axes of breathing and tumor motion, this information can then be used in the configuration of tumor motion prediction methods which are based on breathing information. The last row in Table 3.3 gives the statistical metrics of the correlation coefficient if only the correlation with largest coefficient is taken into account for the tumor y-axis. The last column represents the number of datasets of the corresponding correlation contributing to best↦y. Against the background of previous discussion, it is of no surprise, that most of the datasets with best correlation originate from z↦y and y↦y. Clearly, all metrics of the best correlation supersede the other correlations. As can be seen from Figure 3.4(a), more than 75% of the datasets among best↦y possess a correlation coefficient larger than 0.75. Only 8 outliers with coefficients lower than 0.53 are found, so that for most of the dataset a correlation can be found with reasonably large correlation coefficient.

A still open question concerns significance of the derived correlation coefficients. The significance can be tested with a t-test on the null hypothesis that two datasets are not correlating, i.e. possess a correlation coefficient of 0. The probability to accept the null hypothesis over all computed correlation coefficients is shown in Figure 3.4(b) by a scatter plot. Introducing a significance level of 95%, i.e. a probability of at maximum 5%, it can be seen that all correlation coefficients larger than 0.1 are to be treated as significant (except for one), since the p-values for these coefficients are well below the significance limit. With this limit it turns out that only a few correlation coefficients (74 out of 1800) are not significant, but anyway with a very low correlation coefficient. If the significance level would be set to 99%, then only 89 out of 1800 correlation coefficients would be not significant. This demonstrates that most of the computed correlation coefficients are significant. If the best matching correlation is chosen, then all correlation coefficients are significant (minimum correlation coefficient for best↦y is 0.256 and all coefficients larger than 0.1 are significant).

The correlation coefficient from Eq. (3.1) assumes a linear relationship between breathing and tumor motion, described by the first-order linear model

$$p_m^T(t) = c_{1_{l,m}} p_l^B(t) + c_{0_{l,m}}$$

with $1 \leq t \leq N$ and coefficients $c_{1_{l,m}}$ and $c_{0_{l,m}}$. In this case it is equivalent to the square root of the coefficient of determination R^2. In the linear regression problem to find appropriate coefficients (i.e. fitting a straight line to the data), the coefficient of determination is the proportion of variance which is explained by the model and is a useful measure for the goodness of fit. The general regression problem described by

$$p_m^T(t) = f\left(p_l^B(t)\right) + e(t) \tag{3.2}$$

with $f : \mathbb{R} \mapsto \mathbb{R}$ where f includes q parameters $c_{i_{l,m}}$ with $0 \leq i \leq q$ (linearly or nonlinearly) is to determine these parameters so that errors $e(t)$ are minimized for $1 \leq t \leq n$. In this scenario the coefficient of determination is defined as

$$R_{l,m}^2 = 1 - \frac{\sum_{t=1}^n \left(f\left(p_l^B(t)\right) - p_m^T(t)\right)^2}{\sum_{t=1}^n \left(p_m^T(t) - \overline{p_m^T}\right)^2} = 1 - \frac{\sum_{t=1}^n (e(t))^2}{\sum_{t=1}^n \left(p_m^T(t) - \overline{p_m^T}\right)^2} ,$$

where the numerator represents the residual sum of squares and the denominator is the total sum of squares (which is proportional to the sample variance). Since neither the correlation nor the coefficient of determination by itself reveal how exactly the relationship between breathing and tumor motion should be formed using f, one possibility is to fit the parameters of various functions f with a different shape and to evaluate R^2 using f. This was done extensively with a lot of linear and nonlinear functions f (linearity in parameters). Major improvements of R^2 or its square-rooted value could be observed for polynomial functions of the form

$$f_{l,m}\left(p_l^B(t)\right) = \sum_{i=0}^{q} c_{i_{l,m}} \left(p_l^B(t)\right)^i \ . \tag{3.3}$$

The corresponding square-rooted value of R^2 is depicted in Figure 3.5 for polynomial orders up to 5 when using best$\mapsto$y. This conversion was chosen to have it scaled to previously discussed correlation coefficients for better comparison. It can be observed that for orders higher than one the median of $\sqrt{R^2}$ becomes larger. With this, the interquartile range reduces and shifts slightly to larger values whose lower quartile value rises significantly. Especially for orders higher than three, outliers with lowest values are eliminated and the lower whisker value rises. As a consequence, the fitting function between breathing and tumor motion can be better approximated with higher order polynomials than with a simple first order linear polynomial. Nevertheless the gain for orders larger than three reduces, so that a fitting polynomial of order three can be considered sufficient for the used data.

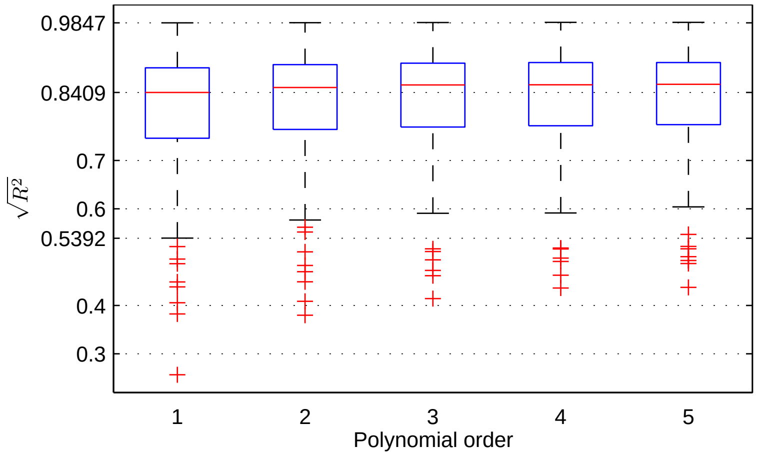


Figure 3.5: *Square-rooted coefficient of determination for best$\mapsto$y when polynomial relationship between breathing and tumor motion is used*

In general, one might expect larger correlation coefficients or coefficients of determination between breathing and tumor motion since the main part of tumor motion is caused by breathing and lung tumors physically move according to breathing. One limiting factor in the calculation of the correlation coefficient is a possible phase shift between breathing and tumor motion. Phase shifts can occur due to anatomical reasons or be caused by timing differences or variations during sensor data acquisition (here breathing and tumor motion was recorded using two independent systems). Phase shifts are not reflected by the correlation coefficient and can reduce it, although there might be a very strong correlation.

To illustrate that, consider two periodic functions of the form

$$x(\tau) = a_1 \sin(2\pi f \tau) + x_0 \quad \text{and} \quad y(\tau) = a_2 \sin(2\pi f \tau + \phi) + y_0 \ ,$$

where $\tau \in \mathbb{R}$ represents continuous time, $a_1 \in \mathbb{R}$ and $a_2 \in \mathbb{R}$ are the amplitudes of the sines, $x_0 \in \mathbb{R}$ and $y_0 \in \mathbb{R}$ represent an offset or DC gain, $f \in \mathbb{R}$ is their common frequency and $\phi \in \mathbb{R}$ is the phase shift between the sines $x(\tau) \in \mathbb{R}$ and $y(\tau) \in \mathbb{R}$. If there is no phase shift, the scatter plot of $x(\tau)$ over $y(\tau)$ shows a straight line through the point $(x_0, y_0)^T$, where the slope is determined by the ratio of the amplitudes $\frac{a_2}{a_1}$. In this case, the value of the correlation coefficient is one and the correlation can be perfectly described by a linear first-order model identified with the parameters of the straight line. Thus, the correlation coefficient is invariant with respect to changes of the amplitudes and the offset. However, it is not invariant under changes of the phase shift ϕ. A phase shift of 90° leads to a correlation coefficient of 0 and a phase shift of 180° results in -1. In fact, the correlation coefficient for phase shifted sinusoidal signals is given by $\cos(\phi)$.

A phase shift ϕ different from zero causes the plot to change from a straight line to an ellipse which can be described in parametric form in the x-y-plane by

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x_c \\ y_c \end{bmatrix} + R(\alpha) \begin{bmatrix} a\cos\varphi \\ b\sin\varphi \end{bmatrix} \tag{3.4}$$

with

$$R(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} .$$

$x_c = x_0$ and $y_c = y_0$ locate the center of the ellipse in the x-y-plane, $a \in \mathbb{R}$ is the length of the semi-major axis, $b \in \mathbb{R}$ denotes the length of the semi-minor axis, $\alpha \in [0, 2\pi]$ determines the tilt angle of the ellipse w.r.t the x-axis and $\varphi \in [0, 2\pi]$ is the free parameter in the ellipse equation. Enlarging the phase shift ϕ from 0° to 180° effects the following changes of the ellipse:

- The tilt angle α can vary in the interval $\left[\arctan\left(\frac{a_2}{a1}\right), -\arctan\left(\frac{a_2}{a1}\right)\right]$, becoming zero for $\phi = 90°$. If and how α varies depends on the amplitude ratio. If the amplitude ratio is one, α takes only the value zero or the limits of the interval, which are in this case 45° and $-45°$.

- Starting from the maximum

  $$a_{max} = \frac{\max(|a_1|, |a_2|)}{\cos\left(\arctan\left(\frac{\min(|a_1|,|a_2|)}{\max(|a_1|,|a_2|)}\right)\right)} ,$$

  the length of the semi-major axis a decreases to its minimum given by $a_{min} = \max(|a_1|, |a_2|)$, which is reached for $\phi = 90°$. After that it approaches the maximum again which is reached for $\phi = 180°$.

- The length of the semi-minor axis b increases from $b_{min} = 0$ to its maximum defined by $b_{max} = \min(|a_1|, |a_2|)$ for $\phi = 90°$ and approaches zero again as ϕ tends to 180°.

- With varying a and b, the numerical eccentricity

  $$e = \frac{\sqrt{a^2 - b^2}}{a}$$

  changes as well. Starting with a maximum eccentricity of 1, e reaches the minimum $e_{min} = \frac{1}{a_{min}}\sqrt{a_{min}^2 - b_{max}^2}$ at $\phi = 90°$. An eccentricity of zero, resulting in a circle instead of an ellipse, can only be achieved with equal amplitudes a_1 and a_2.

In the extreme case when the semi axes are a_{max} and $b_{min} = 0$ exactly, ellipse equation (3.4) reduces to the representation of a straight line with slope α. So the ellipse equation is a generalization of the straight line, given in the case of no phase shift. To summarize, phase shifts of two sinusoidal signals of the same frequency determine the shape of resulting ellipse when both signals are plotted against each other. Fitting an ellipse to the given data allows then to deal with the phase shift and establishes a perfect correlation between two ideal phase-shifted sinusoids. This can even be extended to sinusoids of the form

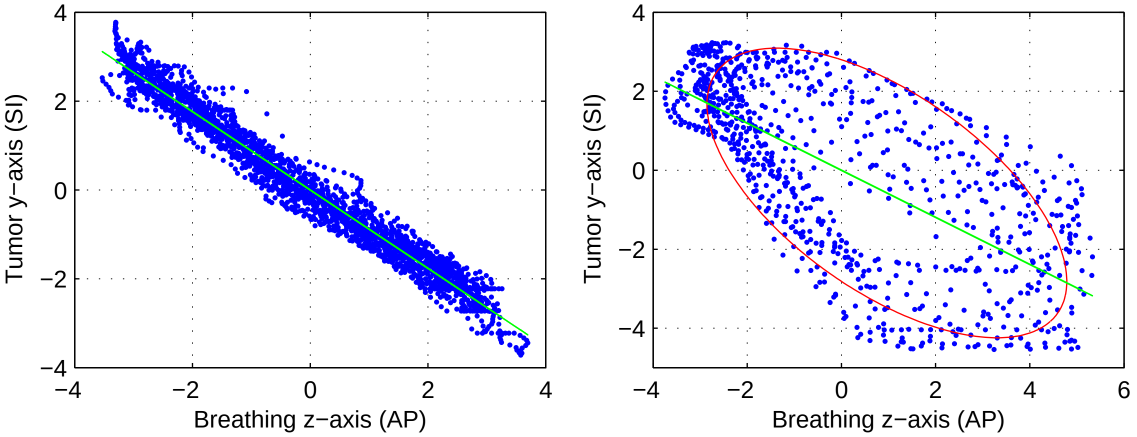
$$x(\tau) = a_1 \sin^2(2\pi f \tau) + x_0 \quad \text{and} \quad y(\tau) = a_2 \sin^2(2\pi f \tau + \phi) + y_0$$

or

$$x(\tau) = a_1 \cos^2(2\pi f \tau) + x_0 \quad \text{and} \quad y(\tau) = a_2 \cos^2(2\pi f \tau + \phi) + y_0 \ .$$

which can better approximate hysteresis effects as discussed before. The only necessary change is attributed to the correlation coefficient which becomes $\cos(2\phi)$.

A detailed visual inspection of scatter plots between breathing and corresponding tumor motion of all possible combinations of axes revealed that for a significant number of datasets an elliptic correlation between breathing and tumor motion was observed rather than a pure linear correlation. Examples of both cases from the datasets are visualized in Figure 3.6(a) for a linear correlation and in Figure 3.6(b) for an elliptic correlation. It can be seen from Figure 3.6(a) that the fitted straight line (green) is well suited to describe linear correlation between breathing along z-axis and tumor motion along y-axis. But this is not the case in Figure 3.6(b), where the residuals between data points (blue dots) and fitted ellipse (shown in red) are lesser than to the fitted line (in green color). In such a case, which occurs due to phase shifts between breathing and tumor signals, the correlation can be better described by an ellipse.



(a) Example of linear correlation

(b) Example of elliptic correlation

Figure 3.6: *Examples of different correlation types*

Note that residuals of an ellipse can be defined in different ways. For example a residual could be defined as the orthogonal distance from a data point to the closest point on the ellipse, yielding the geometric distance further denoted as d_g. Alternatively it could be seen as the vertical distance d_v (along the y-axis) between the y-component of the data point and the closest point on the ellipse having the same x-component as the data point. The latter is the same as calculating the points of intersection between the ellipse and a vertical line defined by the x-component. If two intersections result, the intersection point with minimum distance to the data point is used. If no intersection is found, a point of the ellipse with maximum or minimum (whichever lies closer to the data point) x-component is taken. This residual type is chosen here since it also better reflects the purpose of the correlation: Given a breathing sample (x-component), derive the corresponding tumor sample (y-component).

Fitting ellipses to sampled data is a research topic of its own. For this work, several fitting algorithms described and compared in [82] and [83] were considered in a preliminary analysis. Fitting ellipses can either be performed by minimizing the algebraic distance d_a in a least-squares sense or the geometric distance d_g (Euclidean distance to closest point on ellipse). The former means minimizing an algebraic quantity included in the representation of an ellipse in conic form (as opposed to parametric form) and results in a constrained linear least-squares problem which can be solved efficiently. Methods minimizing the geometric distance d_g need to solve nonlinear least-squares problems whose computational effort is one to two orders of magnitude larger. Since observed results on methods performing the algebraic fit produce useful results on the given data, these methods were favored here. Several available methods were compared by the number of ellipses found and by statistical metrics of the residuals. The method which was found to perform well on the given data was developed by Bookstein [84]. It is based on eigenvalue decomposition in combination with solving a system of linear equations. It is computationally inexpensive, enabling the online usage of this method.

In order to quantify the goodness of fit for ellipse fits, which is comparable for different ellipses and also invariant under Euclidean transformations, a similar measure like the coefficient of determination is required. Several measures [85] exist, but none of them can be used for generalized comparisons. Thus, the following procedure was performed to quantify the improvement of elliptic correlation: On each dataset of length n, an attempt is made to fit ellipses to breathing and tumor signals along each combinations of axes. If an ellipse can be fitted, the residuals $d_v\left(p_l^B(t), p_m^T(t)\right)$ are computed for all discrete sampling instants $t \in \mathbb{N}$ according to previous description. Then a linear first-order fit according to Eqn. (3.2) and (3.3) with $q = 1$ was performed on the same data and the residuals $e(t) = f\left(p_l^B(t)\right) - p_m^T(t)$ between fitted model and data points were determined. Based on these residual measures, the residual ratio

$$r_r = \sqrt{\frac{SSR_e}{SSR_l}} = \sqrt{\frac{\sum_{t=1}^{n}\left(d_v\left(p_l^B(t), p_m^T(t)\right)\right)^2}{\sum_{t=1}^{n} e^2(t)}}$$

was defined, where SSR_e is the sum of the squared residuals of the ellipse and SSR_l is the sum of the squared residuals of the straight line. If r_r is between zero and one, the elliptic correlation leads to lesser residuals than the linear correlation. The distribution of the residual ratios are visualized in Figure 3.7, in which 14 outliers with residual ratios above 3.0 (most of them occurring along axis with weaker correlation) are omitted for better visibility. From the totally possible 1800 axis combinations for 200 datasets, 81 are neglected because either the projection on the x-axis or the z-axis of the tumor motion is zero due to the used gantry angle, leading to a total number of 1719 combinations. From these, ellipses could be fitted in 1325 cases (77%). It can be observed that for almost all

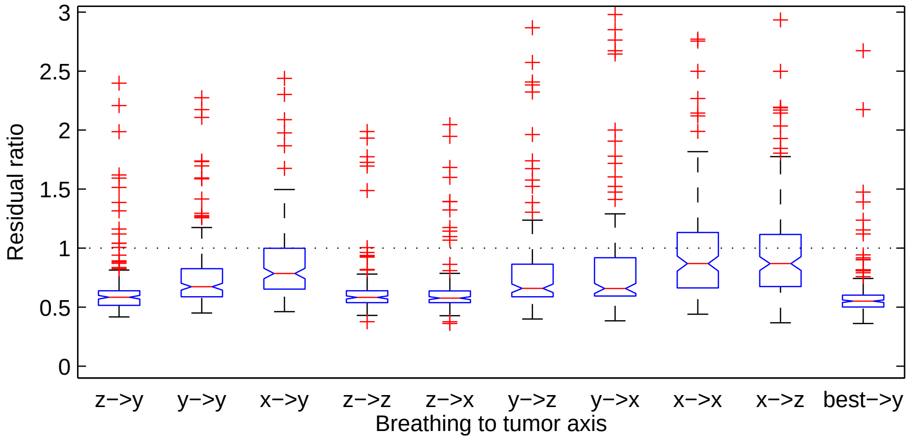


Figure 3.7: *Improvement of elliptic correlation in relation to linear first-order correlation. A few outliers with residual ratios above 3 are omitted for better visibility.*

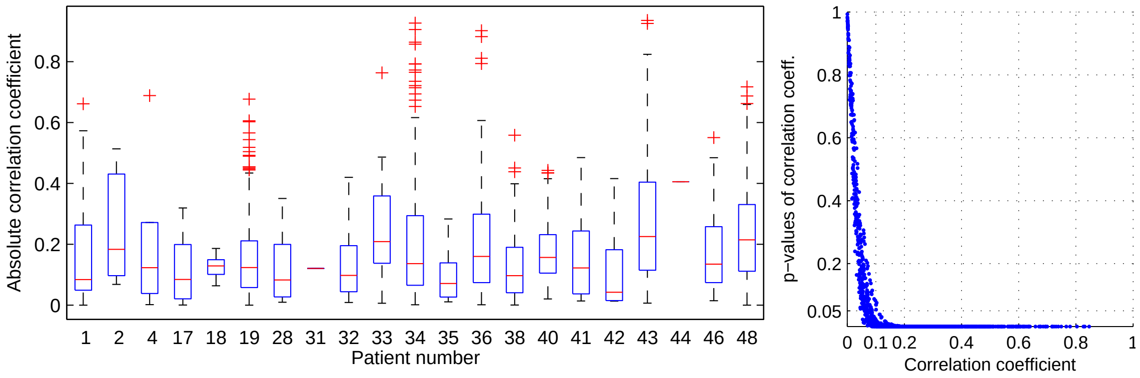
axis combinations, the residual ratio improves substantially using the elliptical correlation. In total, 81.8% (1084 out of 1325) of the residual ratios are below 1.0. Note, that in this analysis only one ellipse was fitted to a complete dataset, thereby performing a kind of averaging over the dataset. As can be seen in Figure 3.6(b), the path through the scatter plot approximately follows the fitted ellipse but also deviates from it as the underlying breathing and tumor motion progresses with time. This is due to variability in breathing patterns and could be accounted for by continuously updating the ellipse fit as time moves on. Nevertheless, even for the static case using one ellipse, the improvement is already significant.

However, some cases remain, where neither the linear correlation nor the elliptic correlation is useful. Most of these cases occur along the axes which anyway possess only small amplitudes. This can be seen from scatter plots, showing a rather random distribution for some datasets. Reasons for this are subject to limited sensor resolution both in position and time, e.g. the used EPID possesses a resolution of 0.251mm with a sampling frequency of about 2Hz. In contrast to the higher sampling rate of the breathing signal, this leads to interpolation errors (cf. Section 2.2.2.2) which can further reduce the correlation coefficient or residual ratios. Considering these effects, the achieved correlation between breathing and tumor motion can be considered large enough to assume a useful relationship, either as elliptic or linear correlation whichever performs better. This can be exploited during the development of tumor prediction algorithms which partially rely on breathing signals in addition to tumor motion. In extreme cases, tumor motion prediction could be solely based on breathing signals in such a way that breathing serves as a surrogate. However, larger deviations between predictions and true values should be expected in this case.

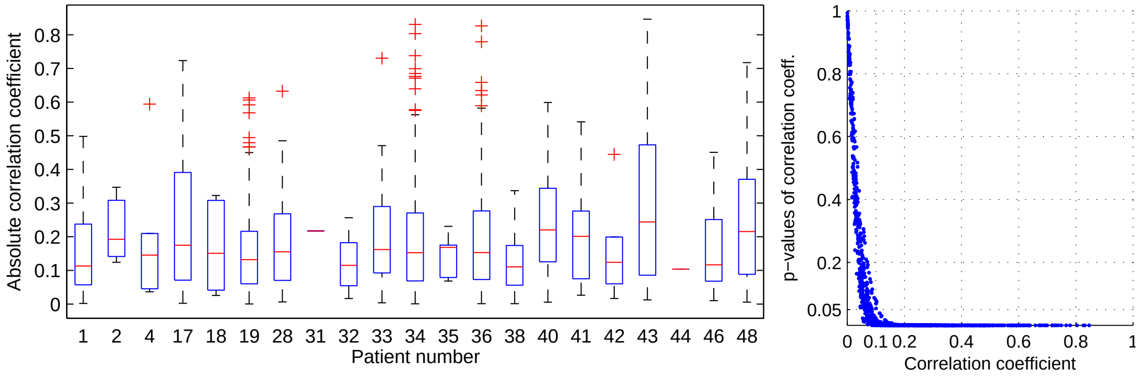
Inter-Field and Intra-Fraction Correlation An interesting question arises about variability of breathing patterns of a single patient in between several fields. This can be analyzed by comparing either each available field (resulting in the inter-field correlation) or all fields of a single fraction (leading to the intra-fraction correlation). Since recording of patient data started at an arbitrary point in the breathing cycle, the start time of each dataset has to be synchronized for the comparison of a pair of fields by the correlation coefficient. If this is omitted, then an unknown phase shift in the data could lead to useless correlation coefficients. This was achieved here using the peak detection where either the first maximum or minimum peak was chosen, whichever occurred first in both datasets.

For the inter-field correlation, correlation coefficients were calculated for each available pair of fields from a single patient. This was done individually for breathing motion with data from the dominant z-axis (AP) and for tumor motion with data from the y-axis (SI). The corresponding absolute correlation coefficients for each patient (three patients with only one field available are not shown) for breathing motion are depicted in the box plot in Figure 3.8(a) and their p-values are visualized in Figure 3.8(b). In general the correlation coefficients take small values below 0.5 with their median being usually less than 0.2. This is also reflected by the overall statistics given in the first part of Table 3.4. Mean and median correlation coefficient are quite small and the coefficients themselves have a large spread (see STD) in the same magnitude as the median or mean value. Only a few outliers with larger values suggest a good correlation. But in general a useful correlation between different fields of a patient cannot be established in a reliable way by using the correlation coefficient. A similar picture emerges fory tumor motion. The associated box plot in Figure 3.8(c) and corresponding scatter plot of p-values in Figure 3.8(d) are similar to the respective plots for breathing, resulting in the conclusion that inter-field correlation of breathing and tumor motion along their dominant axes are similar to each other.

Since inter-field comparisons include data from several fractions recorded on different days, one might assume that comparisons of fields belonging to the same fraction yield



(a) Inter-field correlation of breathing motion along dominant z-axis (b) Correlation significance



(c) Inter-field correlation of tumor motion along dominant y-axis (d) Correlation significance

Figure 3.8: *Inter-field correlation analysis*

	MEDIAN	MEAN	STD	MAX	MIN	#
Inter-field, breathing (AP)	0.1426	0.1867	0.1659	0.9357	0.0001	1292
Inter-field, tumor (SI)	0.1461	0.1868	0.1567	0.8464	0.0005	
Intra-fraction, breathing (AP)	0.1728	0.2340	0.2035	0.9269	0.0018	307
Intra-fraction, tumor (SI)	0.1788	0.2214	0.1752	0.8309	0.0007	

Table 3.4: *Statistics of inter-field and intra-fraction correlation coefficients*

substantially larger correlation coefficients because fields of the same fraction are close together in time. However, statistical results stated in the lower part Table 3.4 about the intra-fraction correlation coefficient suggest a slightly improved correlation compared to the inter-field correlation, but still with small values in general. A conclusion can be drawn from this that even for fields within one fraction, the variability of the main breathing and tumor motion is too large for a linear first-order relationship to be derived. This is supported when individual motions are compared visually between several fields. It can be seen that even during one field, breathing and tumor motions are subject to amplitude and phase changes which do not occur again in the same way, either in the field itself or in another field. Despite these nonlinear changes, the shape of observed hysteresis during each breathing cycle seems similar for each patient. But this alone does not lead to a large correlation coefficient. In general, there is too much variability in breathing, significantly reducing the possibility of reaching a good correlation.

3.2 Prediction Problem Formulation

3.2.1 Goal

The goal pursued by a predictor is to overcome the bounds imposed by causality in order to gain information about the future. This information can be used in the present to influence a process, so that future behavior can be determined to a certain extent. If a process is subject to unknown influences for which no mathematical formulation is given or is hard to derive, one can use past outputs and/or inputs to the process to approximate the underlying process dynamics and assume that the dynamics will still hold in the near future. Naturally, this scheme is only feasible, if input to output behavior is not purely random but is based on either a trend or some recurring pattern. Clearly, breathing and lung tumor motion fulfills this condition, so that, for example in contrast to erratic and spontaneous prostate motions, these should be reasonably predictable.

For tumor motion prediction, the prediction problem can be stated as follows: Denote the 3D breathing position along all translational axes at a discrete sampling instant $t \in \mathbb{N}$ as $\vec{p}^B(t) = \left[p_1^B(t), p_2^B(t), p_3^B(t)\right]^T$ and the corresponding tumor position as $\vec{p}^T(t) = \left[p_1^T(t), p_2^T(t), p_3^T(t)\right]^T$. Further denote the timestamps, which are acquired from a single clock, associated with each discrete sampling instant t with $\tau(t) \in \mathbb{R}$. Given n_B past samples of breathing motion $\vec{p}^B(t), \dots, \vec{p}^B(t-n_B+1)$ and n_T past samples of tumor motion $\vec{p}^T(t), \dots, \vec{p}^T(t-n_T+1)$ at instant t, the predictor f computes a k-step prediction of the 3D tumor position $\hat{\vec{p}}^T(t+k|t)$:

$$\hat{\vec{p}}^T(t+k|t) = f\left(\vec{p}^B(t), \dots, \vec{p}^B(t-n_B+1), \vec{p}^T(t), \dots, \vec{p}^T(t-n_T+1)\right). \tag{3.5}$$

The notation $(t+k|t)$ explicitly shows that an item was computed at sampling instant t but refers to a future sampling instant $t+k$. k is usually referred to as the prediction horizon. For any predictor in the context of this work, it is assumed that all inputs and outputs are equidistantly spaced in time with a constant sampling period $\Delta\tau$, so that $\tau(t+k) = \tau(t) + k\Delta\tau$ for any $t, k \in \mathbb{N}$. Depending on available data or constraints (e.g. certain correlations between different axes or limitations of the prediction method), the inputs to the predictor from Eq. (3.5) might be reduced. For example only certain axes of breathing or tumor motion might be used or the calculation might be solely based on either the breathing or the tumor motion. Similarly, the output might be restricted to one or two axes of tumor motion, if only motions along these axes should be compensated.

In discrete systems, in which measurements of signal values are available at every sampling instant, more up-to-date predictions can be achieved by performing multistep predictions. That is, in each sampling instant t, not only is the k-step ahead prediction calculated, but also the predictions from $t+1$ to $t+k-1$ are determined which were already calculated at earlier sampling instants but with larger prediction horizon. In this way, the most recent input values are used by the predictor, so that the effective prediction horizon (between age of the input data and current sampling instant to be predicted) is varying from one to k. In doing so can serve to achieve higher prediction accuracy because accuracy degrades with rising prediction horizon.

3.2.2 Requirements and Metrics of Prediction Algorithms

Certain demands arise on how predictors should work and how their output should be shaped in order to achieve high overall system performance. Where applicable, corresponding metrics are given with which a requirement can be quantified. These are used in the remainder of this chapter for comparison between various prediction algorithms.

Adaptivity From previous discussion about the nature of breathing and tumor data and also stated in [86], especially in terms of variability, it is obvious that predictors should be laid out in an adaptive way so that they can deal with changing breathing or tumor motion patterns.

Accuracy In order to achieve a high-performance tumor motion compensation, the predictor should deliver accurate predictions: Given a k-step ahead prediction $\hat{\vec{p}}(t|t-k)$ of either the breathing $\hat{\vec{p}}^B(t|t-k)$ or tumor motion $\hat{\vec{p}}^T(t|t-k)$, the 3D k-step prediction error

$$\vec{e}_k(t) = \vec{p}(t) - \hat{\vec{p}}(t|t-k) \tag{3.6}$$

between the true value $\vec{p}(t)$ in sampling instant t (assuming that the true value can be precisely determined) and the predicted position should be as small as possible over all sampling instants $k < t \leq n_k$, where $n_k \in \mathbb{N}$ is the latest sampling instant for which a prediction and an actual position are available simultaneously. In the following it is assumed that each time series of prediction errors is of length $n \in \mathbb{N}$ and starts with sampling instant $t = 1$.

The worst case error along a single axis j is obtained by the maximum absolute error

$$\mathrm{MAX}\left(e_{j,k}\right) = \max\left\{\, |e_{j,k}(t)| \quad | \quad 1 \leq t \leq n \right\} \ .$$

A unified measure for this along several axes is given by taking the maximum of the maximum absolute errors along the individual axes

$$\mathrm{MAX}\left(\vec{e}_k\right) = \max\left\{ \mathrm{MAX}\left(e_{j,k}\right) \quad \middle| \quad 1 \leq j \leq 3 \right\} \ . \tag{3.7}$$

Since maximum errors do not occur permanently for all sampling instants, a similar measure is of interest, which delivers an upper limit of error for a certain amount of sampling instants. This can be seen in statistical terms as a single-sided confidence interval such that absolute prediction errors for a given percentage of sampling instants are below the upper limit of the interval. The 95% confidence interval (CI95) is adopted here, yielding an upper limit of absolute errors so that 95% of the prediction errors belonging to a dataset are less than that limit. Practically this metric is easily obtained by first sorting the prediction errors $e_{j,k}(t)$ along axis j with $1 \leq t \leq n$ in ascending order into a list and then choosing the element at list index $\lfloor n \cdot 95\% \rfloor$ as the upper limit. Along several axes the maximum of this metric along the individual axes is chosen in the same way as for the absolute maximum error in Eq. (3.7), that is

$$\mathrm{CI95}\left(\vec{e}_k\right) = \max\left\{ \mathrm{CI95}\left(e_{j,k}\right) \quad \middle| \quad 1 \leq j \leq 3 \right\} \ . \tag{3.8}$$

Note that the 100% confidence interval is equivalent with the maximum absolute error.

Another common error measure is the mean-square error (MSE) defined by

$$\mathrm{MSE}\left(e_{j,k}\right) = \frac{1}{n} \sum_{t=1}^{n} \left(e_{j,k}(t)\right)^2$$

along an axis j with $j = 1, 2, 3$. The MSE can be seen as the average quadratic error per sample. For better interpretation, it is converted back to the same units as the measured items with the RMS

$$\mathrm{RMS}\left(e_{j,k}\right) = \sqrt{\frac{1}{n} \sum_{t=1}^{n} \left(e_{j,k}(t)\right)^2} \ ,$$

yielding a measure of the average error per sample. For multiple axes, the maximum of the RMS errors along the individual axis is chosen as unified metric:

$$\mathrm{RMS}\left(\vec{e}_k\right) = \max\left\{ \mathrm{RMS}\left(e_{j,k}\right) \quad \middle| \quad 1 \leq j \leq 3 \right\} \ . \tag{3.9}$$

In contrast to the RMS, the mean absolute error (MAE)

$$\mathrm{MAE}\left(e_{j,k}\right) = \frac{1}{n} \sum_{t=1}^{n} \left|e_{j,k}(t)\right| \ ,$$

which is another measure of the average error per sample, does not scale the errors according to their magnitude. In the RMS error, errors larger than one contribute more to the measure, whereas errors smaller than one contribute less. The maximum of the metric along all axes is used as a measure for more axes in order to deliver the worst case performance:

$$\mathrm{MAE}\left(\vec{e}_k\right) = \max\left\{ \mathrm{MAE}\left(e_{j,k}\right) \quad \middle| \quad 1 \leq j \leq 3 \right\} \ . \tag{3.10}$$

Based on the RMS, an intuitively interpretable measure, called the prediction ratio

(PR), is defined. Along a single axis j it is obtained from

$$\mathrm{PR}\left(p_j, e_{j,k}\right) = \left(1 - \frac{\mathrm{RMS}\left(e_{j,k}\right)}{\mathrm{RMS}\left(p_j\right)}\right) \cdot 100\% \ , \tag{3.11}$$

relating the RMS of the prediction error to the RMS value of the signal p_j to be predicted. If no prediction errors are present, the PR takes the value of 100%. Note that the PR might become negative in case prediction fails or is unstable. Failure is thereby characterized by smaller negative values down to about -100%, whereas instabilities lead to larger negative values. In these cases, the RMS of prediction errors in the numerator of Eq. (3.11) is larger than the RMS of actual signal values in the denominator. Deriving the PR along several axis simultaneously is achieved with a weighted average

$$\mathrm{PR}\left(\vec{p}, \vec{e}_k\right) = \frac{\sum_{j=1}^{3} \mathrm{RMS}\left(p_j\right) \mathrm{PR}\left(p_j, e_{j,k}\right)}{\sum_{j=1}^{3} \mathrm{RMS}\left(p_j\right)} \ , \tag{3.12}$$

where PRs of the individual axes are weighted according to the RMS value of the desired signal for the corresponding axes. This takes into account different overall amplitudes along the single axes, so that the prediction errors are scaled accordingly. Substituting Eq. (3.11), Eq. (3.12) can be simplified to

$$\mathrm{PR}\left(\vec{p}, \vec{e}_k\right) = \frac{\sum_{j=1}^{3} \mathrm{RMS}\left(p_j\right) - \mathrm{RMS}\left(e_{j,k}\right)}{\sum_{j=1}^{3} \mathrm{RMS}\left(p_j\right)} \cdot 100\% \ .$$

Smoothness An important property of predictions, but sometimes ignored in related works, is smoothness. A sequence of predictions, especially if multistep predictions are used, should be as smooth as possible because they are used for the generation of a reference trajectory which constitutes the input to the control system. If smooth trajectories can be generated, the control effort for following the given trajectory usually reduces. This in turn enhances patient comfort and might also help to achieve better tracking accuracy.

Smoothness can be measured in different ways. However, to the knowledge of the author no “optimal” smoothness measure quantifying what is intuitively understood by smoothness has been found. Hence, several measures are used, defined as follows.

Especially when considering position signals along an axis j, smoothness can be defined using jerk, the time derivative of acceleration. Using the difference quotient (in forward orientation)

$$\hat{p}_j^{'}(t|t-k) = \frac{\Delta \hat{p}_j(t|t-k)}{\Delta \tau} = \frac{\hat{p}_j(t|t-k) - \hat{p}_j(t-1|t-k-1)}{\tau(t) - \tau(t-1)}$$

or the notationally simplified but equivalent form

$$\hat{p}_j^{'}(t) = \frac{\Delta \hat{p}_j(t)}{\Delta \tau} = \frac{\hat{p}_j(t) - \hat{p}_j(t-1)}{\tau(t) - \tau(t-1)} \ ,$$

where $\tau(t)$ represents the timestamp in seconds of sampling instant t and j is the translational axis, the third-order difference equation, yielding jerk, can be determined recursively for discretized signals:

$$\frac{\Delta^3 \hat{p}_j(t|t-k)}{\Delta\tau^3} = \frac{\Delta^3 \hat{p}_j(t)}{\Delta\tau^3} = \frac{\Delta^2 \hat{p}_j^{'}(t)}{\Delta\tau^2} = \frac{\Delta \hat{p}_j^{''}(t)}{\Delta\tau} \ . \tag{3.13}$$

If all samples of $\hat{p}_j$ are perfectly equidistant with sampling period $\Delta\tau$, then all denominators in Eq. (3.13) become $\Delta\tau$ and the equation can be simplified to

$$\frac{\Delta^3 \hat{p}_j(t)}{\Delta\tau^3} = \frac{\hat{p}_j(t) - 3\hat{p}_j(t-1) + 3\hat{p}_j(t-2) - \hat{p}_j(t-3)}{\Delta\tau^3} \ .$$

Various jerk-based smoothness measures have been proposed so far (e.g. integrated squared jerk or root mean square jerk; see e.g. [87] for an overview). For this application the following smoothness measure S_J is deemed to be useful:

$$S_J\left(\hat{p}_j\right) = \log\left(1 + \frac{a}{(n-2)\left(\max\left(\hat{p}_j\right) - \min\left(\hat{p}_j\right)\right)} \sum_{t=3}^{n} \left| \frac{\Delta^3 \hat{p}_j(t|t-k)}{\Delta\tau^3} \right| \right) \ . \tag{3.14}$$

Basically it consists of the mean absolute jerk which is scaled to an amplitude range of one, so that the measure is independent of signal amplitudes. $\max\left(\hat{p}_j\right)$ and $\min\left(\hat{p}_j\right)$ determine the maximum and minimum of $\hat{p}_j$, respectively. Instead of the often used squared jerk, the absolute jerk is adopted here to make the measure less sensitive to larger jerk values, as the jerk itself is already very sensitive. In order to further mitigate this effect, the logarithm is applied to the resulting values plus one (to avoid negative values). Constant a is introduced to balance Eq. (3.14). Thus it should have the same unit as $\Delta\tau^3$ with value of one.

Another method purely based on time series data for smoothness determination [88] is relating differences between two consecutive samples to differences between a sample value and its mean $\overline{\hat{p}_j}$ by evaluating

$$S_{TS}\left(\hat{p}_j\right) = \frac{\frac{1}{n-1}\sum_{t=2}^{n} \left|\hat{p}_j(t|t-k) - \hat{p}_j(t-1|t-k-1)\right|}{\frac{1}{n}\sum_{t=1}^{n} \left|\hat{p}_j(t|t-k) - \overline{\hat{p}_j}\right|} \ . \tag{3.15}$$

The index TS is short for "time series" and denotes the formulation on which the calculation of smoothness is based.

A different approach is motivated from frequency domain by realizing that smoothness is influenced by high-frequency contributions in signals in relation to low-frequency contributions. Following that idea, a measure of smoothness can be derived similar to the signal-to-noise ratio (SNR), given an appropriate definition of "signal" and "noise". For the application of breathing or tumor motion prediction, it can be safely assumed (cf. Section 3.1) that frequencies below 0.8Hz can be devoted to the signal whereas frequencies above that limit are considered noise. The limit is chosen to be slightly larger than the observed maximum breathing or tumor motion frequency. Then the smoothness measure S_{SNR} along axis j can be defined in units of dB by

$$S_{SNR}\left(\hat{p}_j\right) = 10 \log \frac{P_{signal}\left(\hat{p}_j\right)}{P_{noise}\left(\hat{p}_j\right)} \ , \tag{3.16}$$

where $P_{signal}\left(\hat{p}_j\right)$ is computed from the DFFT of $\hat{p}_j$ as sum of contributions to the power spectrum whose frequencies are below 0.8Hz and $P_{noise}\left(\hat{p}_j\right)$ is determined accordingly from the same power spectrum but using frequencies larger than 0.8Hz.

The last smoothness measure discussed here is based on digital filters. When low-pass filters are applied to signals, the goal is to remove contributions with larger frequencies from the signal in order to smooth the signal. When comparing the filtered and unfiltered signals, the difference between them can be considered a measure of smoothness. This difference is what the filter has removed from the original signal. If the overall difference is small, then the filtered and unfiltered signal are similar and consequently the unfiltered signal is smooth. If the difference is larger, the filter has removed more contributions from the signal, leading to the conclusion that the unfiltered signal is less smooth. However, filtered and unfiltered signals can only be compared if filters introduce no phase shift and if the discretized signals are of the same length. To account for this constraint, zero-phase filters based on low-pass Butterworth filters (see Section 3.1) are used here. However, the filter has to be properly designed in terms of order and cut-off frequency so that it behaves in the desired way. Having once determined the filter parameters, this smoothness measure cannot be applied generically as it is possible with S_J or S_{TS}, but it is limited to certain input signals. However, the properties of breathing and tumor signals are restricted to a narrow range in terms of amplitudes and frequencies, allowing for the design of a unique filter which enables the feasibility of the smoothness measure based on filtering. For the determination of the smoothness of tumor predictions (sampled at a rate of 20Hz) a 9-th order Butterworth filter with a cut-off frequency at 2Hz yielded a flat response in the passband and sufficiently large dampening in the stopband. If the filtered position prediction signal along axis j is denoted by $\hat{f}_j$, the smoothness measure can be defined by

$$S_F\left(\hat{p}_j\right)=\frac{1}{n\left(\max\left(\hat{p}_j\right)-\min\left(\hat{p}_j\right)\right)}\sum_{t=1}^{n}\left|\hat{f}_j(t)-\hat{p}_j(t|t-k)\right| \ . \tag{3.17}$$

Similar to S_J, the smoothness measure S_F is scaled to an amplitude of one and incorporates the mean absolute difference between the filtered and the unfiltered signal.

If smoothness needs to be determined for more than one axis simultaneously, then several possibilities arise: If the worst case is of interest, one could use the maximum (for S_{TS}, S_J and S_F) or the minimum (for S_{SNR}) of the measures applied to each axis individually. However, this might lead to unfair comparisons among the individual axes due to different overall signal amplitudes. To account for this, a unified smoothness measure S for multiple axes is derived in the same way as the multidimensional prediction ratio in Eq. (3.12). The one dimensional smoothness measures for each individual axis are weighted by the RMS value of the signal:

$$S\left(\hat{\vec{p}},\vec{p}\right)=\frac{\sum_{j=1}^{3}\mathrm{RMS}\left(p_j\right)S\left(\hat{p}_j\right)}{\sum_{j=1}^{3}\mathrm{RMS}\left(p_j\right)} \ , \tag{3.18}$$

Smoother input signals are indicated either by larger values of S_{SNR} or by smaller values of S_{TS}, S_J and S_F. All considered smoothness measures have the drawback that no absolute limit for ideal values can be stated, as their magnitude highly depends on the signal shape, even for smooth signals such as linear combinations of ideal sine or cosine signals of different frequencies. Additionally they yield different changes in values for phase shifted or biased signals. Appendix B demonstrates the sensitivity of the smoothness measures and infers a rating of them. In general the smoothness measure S_F is the least sensitive to phase shifts or drifts in the signal compared to non-smooth signals. Nevertheless, all measures are used to have a profound basis for the assessment of smoothness.

Short Learning Time Any predictor starting with no knowledge about the process to predict needs a certain learning time, similar to a dead time, until the first prediction is available. This time is usually used either to capture the initial dynamics of the underlying process or to perform some kind of learning process. Naturally, it is desirable to keep this time as short as possible, so that predictions can be gained as early as possible in the treatment process. Especially for short fields with durations of 30 seconds or less, it is important to reach short learning times of not more than a few seconds. Normally, the learning time depends on one or more parameters of prediction methods. Since working parameter sets will be determined for the application of tumor motion compensation for each method, the learning time can either be precisely pre-computed or at least constrained to a narrow range.

Fast Convergence After the learning time, the predictor should exhibit fast convergence behavior so that predictions approach true values as fast as possible in order to achieve high accuracy. Since predictors should work adaptively, convergence also plays a role after changes in breathing or tumor motion patterns. In this case, fast convergence can help to limit the temporary loss in accuracy.

Usually convergence is quantified using the prediction error in such a way that convergence is reached if the prediction error tends to approach zero. A useful measures can be derived, for example, which defines a limit on the first-order derivative of the prediction error. If the absolute value of the derivative falls below that limit the first time, then convergence can be assumed. However, such measures cease to work with periodic signals. Then the prediction error is also usually periodic and its absolute value or the one of its derivative become small only during short time intervals but reaches larger values in between. One way to alleviate this problem is to perform averaging in a sliding window manner, both on the prediction error and the predictions. Given a window size w, the "convergence coefficient" $C\left(p_j, \hat{p}_j, t\right)$ at sampling instant t can be defined as the ratio between the averaged prediction errors and the averaged tumor amplitudes inside the sliding window:

$$\begin{aligned} C\left(p_j, \hat{p}_j, t\right) &= \frac{\frac{1}{w}\sum_{i=0}^{w-1} \left|\hat{p}_j(t-i|t-i-k) - p_j(t-i)\right|}{\frac{1}{w}\sum_{i=0}^{w-1} \left|p_j(t-i)\right|} \\ &= \frac{\sum_{i=0}^{w-1} \left|e_{j,k}(t-i)\right|}{\sum_{i=0}^{w-1} \left|p_j(t-i)\right|} \;. \end{aligned} \tag{3.19}$$

Essentially, Eq. (3.19) represents the windowed version of the absolute percent error

$$\left|\frac{e_{j,k}(t-i)}{p_j(t-i)}\right| \cdot 100\% \;.$$

Thus, the decision about convergence is not evaluated based on the data of a single sampling instant alone but over a larger horizon which results in a more robust criterion, especially if it is applied to periodic signals. Of course, robustness is mainly influenced by the choice of the window size w which is given in terms of the used sampling rate. If w is chosen smaller, then it is similar to the evaluation of the first-order derivative and the current error gains more impact in the coefficient which might lead to robustness problems. If w is chosen larger, then the coefficient (regarded as signal over time) becomes smoother and hence more robust. On the other hand, errors from the past have a larger influence and cause the coefficient to decay slower (in case convergence takes place). Due to these facts, w should be chosen in accordance with signal properties, especially w.r.t its dominant frequency. From empirical investigations it was concluded that w should not be less than 1/7 of the period of the main frequency (corresponding to the length of one breathing cycle) but also not larger than one average breathing cycle. Given a sampling rate of 20Hz (corresponding to a period of 50ms), $w = 15$ was chosen here which corresponds to about 1/4.44 of the period of an average breathing frequency of 0.3Hz.

Based on the convergence coefficient, the convergence time can be expressed by

$$t_{j,c_{max}} = \min \left\{ t \mid C\left(p_j, \hat{p}_j, t\right) \leq c_{max} \right\} \ , \tag{3.20}$$

where c_{max} is the limit in percentage and $t_{j,c_{max}}$ represents the sampling instant when the convergence coefficient first dropped below the limit for axis j. This can be converted into time with $\tau_{j,c_{max}} = \tau(t_{j,c_{max}})$. In this work, two limits were chosen as 25% and 10% for c_{max} and both of them evaluated for each prediction algorithm. Of course, choosing different limits will lead to different convergence times, so that the use of the convergence time has to be treated in such a way that it gives a rough idea.

The unified measure for convergence time along multiple axes simultaneously is derived by taking the maximum of the measures of the individual axes:

$$t_{c_{max}} = \max \left\{ t_{j,c_{max}} \quad \middle| \quad 1 \leq j \leq 3 \right\} \ . \tag{3.21}$$

Computationally Fast The tumor motion compensation system is triggered with a certain sampling rate which highly depends on the capabilities of the sensors and actuators. Given the system described in Section 2.2.1, possible sampling rates fall into the range from 10Hz to 60Hz (smaller rates than 10Hz might lead to larger sampling and tracking errors, cf. Section 2.2.2.2). Consequently, all computations have to be finished within one sampling period in the range from 16.667ms to 100ms. Depending on the computational time of other components (e.g. moving average, peak detection, control scheme), the predictor can only use a certain fraction of the complete duration. In order to build a motion compensation system being able to operate in real-time, the predictor should only use a certain maximum computation time of about several milliseconds. The maximum allowable time depends on the selected sampling rate and the computational times of the other system components. The computational time is assessed for this work by measuring the execution time of the prediction algorithm on a standard state-of-the-art PC.

Flexible Prediction Horizons As already discussed before, the predictor should provide multistep predictions to shorten effective prediction horizons and to improve accuracy. Simultaneously the ability of multistep predictions brings flexibility to the predictor, which is especially important if the tumor position signal needs to be interpolated. Flexible horizons guarantee the ability to calculate the required number of predictions, even when tumor motion is sampled with a different sampling rate (cf. Section 2.2.2.2). In such a case, the prediction horizon can be increased or decreased according to the difference in the number of samples between the predictors update rate and the tumor sampling rate.

3.2.3 Prediction Schemes

Depending on the capabilities of prediction algorithms, they can be grouped into two different classes. The algorithms of class I are constrained in such a way that they can only perform predictions of a single input signal (cf. Figure 3.9). Most of the commonly employed prediction algorithms fall into this class. In that respect, they can either output breathing or tumor motion predictions which are based on either breathing or tumor signals, respectively. Therefore the prediction problem according to Eq. (3.5) becomes

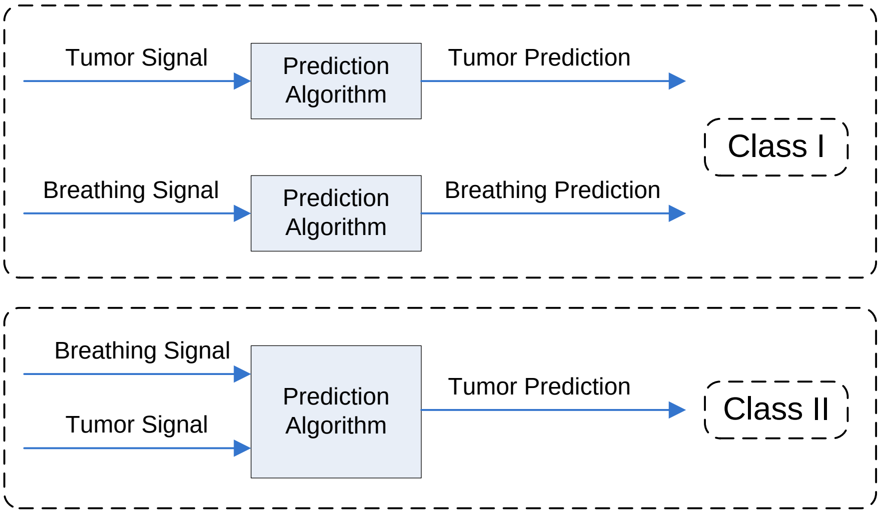


Figure 3.9: *Basic classes of prediction algorithms*

$$\hat{\vec{p}}^T(t+k|t) = f_k\left(\vec{p}^T(t), \ldots, \vec{p}^T(t-n_T+1)\right)$$

for tumor motion prediction and

$$\hat{\vec{p}}^B(t+k|t) = f_k\left(\vec{p}^B(t), \ldots, \vec{p}^B(t-n_B+1)\right)$$

for breathing motion prediction. For tumor motion compensation, the algorithms in class I can consequently only be used (without modification) in the alternative configuration of HexGuide (cf. Section 2.2.1.5 and Figure 2.7), which purely relies on direct measurements of tumor motion (without additional breathing information). As discussed in Section 2.2.2.2, a sufficient sampling rate of the tumor motion sensor needs to be used in order to reduce sampling errors which can seriously decrease system performance. For example, this condition is fulfilled for the Calypso system but not for the EPID. It is assumed that an equidistant sampling rate is either provided by the sensor or achieved by interpolation.

Algorithms in class II can make use of direct and indirect measurements of the tumor position, where the latter is represented by the breathing signal in this work (see Figure 3.9). The indirect measurements are assumed to be sampled with a sufficient and equidistant sampling rate (which is the case for the breathing signal acquired by the Polaris system), whereas the sampling rate of the direct measurement can be lower. If the sampling rates differ, then the direct measurements needs to be interpolated to the rate of the indirect measurement using the interpolation techniques described in Section 2.2.2.2.

The fact that signals interpolated at a larger sampling rate are still less informative than those sampled at a larger rate in the first place (even if higher order interpolations are used), does not bring any advantage to class I algorithms when applied to interpolated EPID data. Delays introduced by interpolation aggravate the situation (see Eq. (2.3)). To overcome the limitations of class I algorithms, a cascaded prediction scheme sketched in Figure 3.10 is proposed. A class I algorithm is thereby employed to deliver a breathing position prediction which is fed as input to a regression algorithm. Given a regression function and corresponding parameters describing the relationship between breathing and tumor motion based on correlation, the tumor position prediction can be derived from the breathing position prediction. A prediction algorithm of class I extended by a regression scheme will be referred to as class III algorithm. This is essentially an extrapolation as the determined relationship from the past is assumed to be valid for the near future as well. Based on past measurements of breathing and tumor motion, the regression algorithm continuously updates regression parameters so that the adaptivity requirement is fulfilled. For realizing regression, the correlation analysis performed in Section 3.1.4 is exploited, in which a useful correlation was found to be either represented by a linear polynomial or by an ellipse.

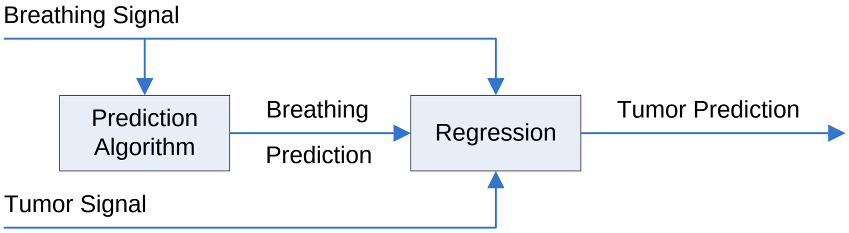


Figure 3.10: *Cascaded prediction scheme, Class III*

Linear Regression According to Eq. (3.2), the linear regression problem to be solved online in each sampling instant t is as follows: Given w_R pairs of past breathing and tumor motion samples

$$\left\{\left(p_l^B(t-w_R+1), p_m^T(t-w_R+1)\right), \ldots, \left(p_l^B(t), p_m^T(t)\right)\right\} , \tag{3.22}$$

the parameters $c_{u_{l,m}} \in \mathbb{R}$ need to be determined from the equation

$$p_m^T(t-i) = f_{l,m}\left(p_l^B(t-i)\right) + e(t-i) = \sum_{u=0}^{q} c_{u_{l,m}} \left(p_l^B(t-i)\right)^u + e(t-i) \tag{3.23}$$

for all $i \in \{0, \ldots, w_R - 1\}$ with $w_R \geq q+1$, while minimizing the errors $e(t-i)$ in a least-squares sense. This can be achieved efficiently using various numerical algorithms (see e.g. [89]) for larger q. If $q = 1$, the least-squares optimization of the linear first-order function possesses a closed form solution and can thus be easily performed. Having obtained the parameters of the function $f_{l,m}$, it can be used with a breathing prediction $\hat{p}_l^B(t+k|t)$ to estimate the corresponding tumor prediction $\hat{p}_m^T(t+k|t)$ as:

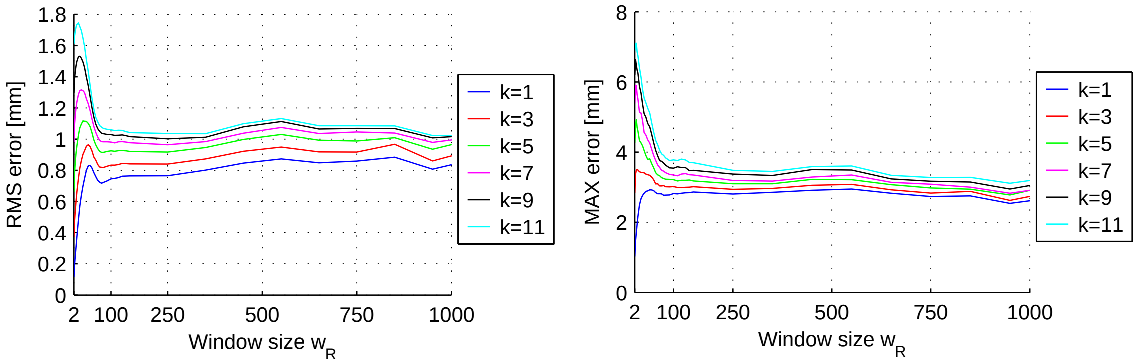
$$\hat{p}_m^T(t+k|t) = f_{l,m}\left(\hat{p}_l^B(t+k|t)\right) \ . \tag{3.24}$$

If the data points are badly conditioned, for example if the resulting regression line would be almost parallel to the y-axis (associated with tumor motion), then the linear regression fails as the underlying system of linear equations is close to a singularity. This case is handled by a fallback solution, replacing Eq. (3.24) with

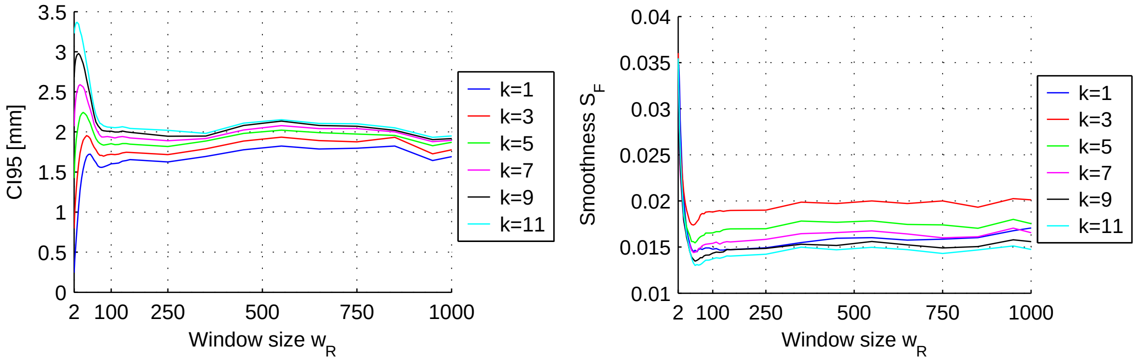
$$\hat{p}_m^T(t+k|t) = p_m^T(t) + k\left(p_m^T(t) - p_m^T(t-1)\right) \ , \tag{3.25}$$

that is, performing a linear extrapolation which is purely based on tumor motion data, where the difference between the last two samples of tumor motion is assumed to hold for future values as well.

In order to assess an optimal window size w_R, the following procedure was performed on all available datasets with best$\mapsto$y axes using different prediction horizons $k \in \{1,3,5,7,9,11\}$: Denote the number of samples in a dataset with n. In each iteration t, starting from w_R up to $n-k$, the linear regression according to Eq. (3.23) is performed and the resulting regression parameters are used in $f_{l,m}$ to calculate the prediction with Eq. (3.24) using $p_l^B(t+k)$. Since future values are available in pre-recorded datasets, future breathing positions $p_l^B(t+k)$ as well as future tumor positions $p_m^T(t+k)$ are known. This reflects the case when perfect breathing predictions are available, which is applicable for this analysis. Hence, the prediction error can be derived by the difference between $\hat{p}_m^T(t+k|t)$ and $p_m^T(t+k)$. Finally, the RMS and maximum of the prediction error as well as the CI95 confidence interval and the smoothness measure S_F were determined and averaged over all datasets for several w_R in the range 2-1000. These metrics are depicted in Figure 3.11 for selected values of k. Note that some datasets contain less than 1000 samples. These are excluded from the average metrics if the length of the dataset is smaller than the current value of w_R. The RMS and CI95 metrics possess a peak for window sizes in the range between 15 to about 50 which result when the window size is in the range of the number of samples corresponding to about one fourth to one breathing cycle, where the peak is reached for smaller k at smaller w_R. If $k < 7$, minimum RMS and CI95 metrics are achieved for $w_R = 2$, whereas for $k \geq 7$ minimum values result from $w_R \geq 75$. In general errors approach a stable value for increasing w_R where the absolute errors become larger for increasing k. This gives rise to the conclusion that regression should be performed at least over the length of a breathing cycle. Similar observations can be made on the maximum errors as well, except that the minimum for $k \geq 3$ is found for larger w_R and the maximum errors slightly decrease continuously with increasing w_R. The averaged smoothness metric S_F in Figure 3.11(d) indicates that regardless of the prediction horizon k, best smoothness is reached for $50 \leq w_R \leq 250$. Furthermore, this justifies the use of a larger w_R for $k = 1$ where lowest RMS and maximum (MAX) errors are found for $w_R = 2$. To summarize, in order to achieve both small errors and smoother trajectories, it is recommended to use a value of w_R in the range 75-250 for any prediction horizon k. Choosing larger values for w_R results in similar error metrics with slightly decreased smoothness. Additionally, large w_R means more averaging over a past horizon of data, limiting the adaptivity of the regression in case of changing motion patterns. Therefore it is useful to keep w_R as low as possible given small errors and small smoothness metrics.



(a) Average RMS errors (b) Average MAX errors

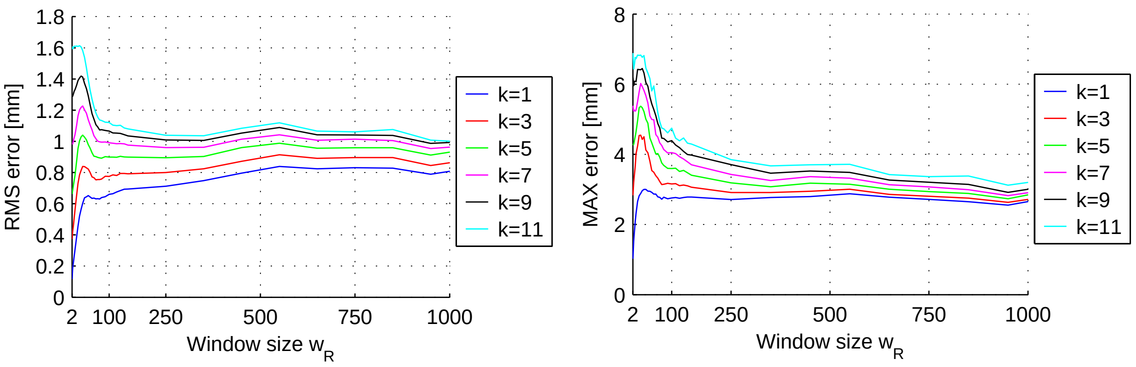


(c) Average CI95 (d) Average smoothness S_F

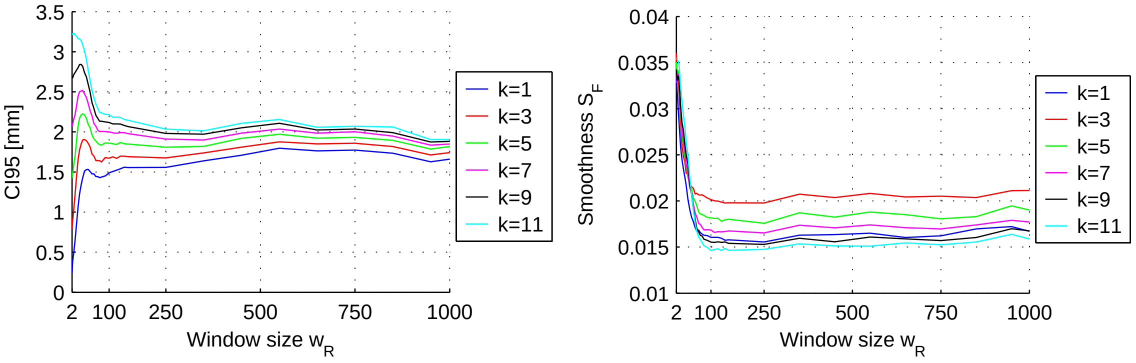
Figure 3.11: *Online linear regression depending on window size* w_R

The shown results were achieved using the polynomial order $q = 1$ in Eq. (3.23). The analysis in Figure 3.5 suggested an improved correlation for $q = 3$. This is also evaluated with the following modifications to the previous first-order regression: At first an attempt is made to fit a cubic polynomial to the given w_R data points. If a singularity occurs, a quadratic polynomial is tried next. If that also leads to singularity, the first-order polynomial is used. If there is a singularity as well, finally the fallback solution from Eq. (3.25) is taken. In polynomial regression the minimum value of w_R is given by $q + 1$. Comparing the corresponding results in Figure 3.12 to the previous case, similar results are obtained. Cubic polynomial regression leads to slightly smaller average RMS errors whereas average maximum errors are larger especially for $w_R < 500$. While a performance increase on CI95 can be observed for $k \leq 5$, first-order regression performs better on that metric for $k > 5$. The metric S_F clearly shows an increase of smoothness values for the whole range of w_R and for any k.

Due to the fact that during normal operation, usually larger k are of interest (because of the time difference between the latest breathing and tumor position information), the first-order linear regression is favored here. Despite the improved correlation using cubic polynomials, the first-order online regression shows a slightly improved performance. One reason is that the correlation analysis reflects the static state in which the whole dataset is already known and in which the residuals between the data points and the regression curve is taken as a metric. In online regression, the dataset is only known up to the current sampling instant and as opposed to the static case, an extrapolation according to



(a) Average RMS errors (b) Average MAX errors



(c) Average CI95 (d) Average smoothness S_F

Figure 3.12: *Online cubic polynomial regression depending on window size w_R*

the prediction horizon k is performed and the residuals between the extrapolated point and the "future" data point is used as a metric. Therefore the results from the static analysis can differ to the online regression using extrapolation.

If the extrapolation based on linear regression is tested with two ideal sines or cosines of the same frequency as surrogate for breathing and tumor motion, then perfect results with no error can be achieved for arbitrary $w_R \geq 2$ and arbitrary $k \geq 1$ as long as there is no phase shift between both signals. However, in the presence of phase shifts, errors arise in any case whose magnitude depends on the phase angle.

Elliptical Regression In order to deal with these effects resulting from phase shifts, the regression scheme based on ellipse fitting, as already pointed out in Section 3.1.4, is evaluated in an online version. Similar to linear regression, an ellipse is fitted in each sampling instant t given w_R pairs of past breathing and tumor motion samples, where $w_R \geq 5$ as the fitting algorithm requires at least five data points. The resulting ellipse is then located in the x-y plane, in which the x-axis refers to breathing motion and y-axis to tumor motion. A k-step ahead tumor prediction $\hat{p}^T_m(t+k|t)$ for a corresponding breathing prediction $\hat{p}^B_l(t+k|t)$ is then determined in the following way: At first, the intersection points between the ellipse Eq. (3.4) and the straight line

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \hat{p}^B_l(t+k|t) \\ 0 \end{bmatrix} + \lambda \begin{bmatrix} 0 \\ 1 \end{bmatrix} , \tag{3.26}$$

parametrized by $\lambda \in \mathbb{R}$ and aligned parallel to the y-axis, are determined by equating both the ellipse and line equation, solving for λ and back-substituting into Eq. (3.26). The solution can be obtained more easily if a reference system is used in which the ellipse center is located at the origin and to which the ellipse is not tilted. The ellipse can be expressed conveniently in canonical form (as opposed to the parametric form in Eq. (3.4)) w.r.t. the transformed reference system by

$$\frac{\tilde{x}^2}{a^2} + \frac{\tilde{y}^2}{b^2} = 1 \ . \tag{3.27}$$

The same transformation can then be applied to the straight line to transform the representation of the line to the same reference system, that is first shifting the line by the ellipse center $(-x_c, -y_c)^T$ and then rotating with $R^T(\alpha)$, yielding

$$\begin{aligned} \begin{bmatrix} \tilde{x} \\ \tilde{y} \end{bmatrix} &= R^T(\alpha) \left(\begin{bmatrix} x \\ y \end{bmatrix} - \begin{bmatrix} x_c \\ y_c \end{bmatrix} \right) = R^T(\alpha) \left(\begin{bmatrix} \hat{x}^B - x_c \\ -y_c \end{bmatrix} + \lambda \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right) \\ &= R^T(\alpha) \left(\begin{bmatrix} \hat{x}^B - x_c \\ 0 \end{bmatrix} + \tilde{\lambda} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right) = \begin{bmatrix} (\hat{x}^B - x_c)\cos\alpha \\ -(\hat{x}^B - x_c)\sin\alpha \end{bmatrix} + \tilde{\lambda} \begin{bmatrix} \sin\alpha \\ \cos\alpha \end{bmatrix} \\ &= \begin{bmatrix} \tilde{x}_l \\ \tilde{y}_l \end{bmatrix} + \tilde{\lambda} \begin{bmatrix} \sin\alpha \\ \cos\alpha \end{bmatrix} \ , \end{aligned} \tag{3.28}$$

where $\hat{x}^B = \hat{p}_l^B(t+k|t)$ and $\tilde{\lambda} = \lambda - y_c$. Eq. (3.28) can be split up into two equations, thereby solving for $\tilde{x}$ and $\tilde{y}$, and inserted into Eq. (3.27), leading to the quadratic equation

$$\underbrace{\left(\frac{\sin^2\alpha}{a^2} + \frac{\cos^2\alpha}{b^2} \right)}_{\tilde{a}} \tilde{\lambda}^2 + \underbrace{2 \left(\frac{\tilde{x}_l \sin\alpha}{a^2} + \frac{\tilde{y}_l \cos\alpha}{b^2} \right)}_{\tilde{b}} \tilde{\lambda} + \underbrace{\frac{\tilde{x}_l^2}{a^2} + \frac{\tilde{y}_l^2}{b^2} - 1}_{\tilde{c}} = 0$$

which needs to be solved for $\tilde{\lambda}$. Introducing

$$D = \tilde{b}^2 - 4\tilde{a}\tilde{c} \ ,$$

a unique solution exists if $D = 0$. The solution is then given by

$$\tilde{\lambda} = \frac{-\tilde{b}}{2\tilde{a}} \ .$$

Because $\lambda = \tilde{\lambda} + y_c$ and $y = \lambda$ (which follows from Eq. (3.26)), the y-component in the original reference frame can be determined which corresponds to the tumor prediction $\hat{p}_m^T(t+k|t)$:

$$\hat{p}_m^T(t+k|t) = y = \lambda = \tilde{\lambda} + y_c \ ,$$

thereby implicitly reversing the transformations on the reference frame.

If $D > 0$, two solutions exist which are

$$\tilde{\lambda}_{1,2} = \frac{-\tilde{b} \pm \sqrt{D}}{2\tilde{a}} \ .$$

In this case, a procedure needs to be found which selects a correct solution. A correct solution is characterized as being close to a point $(x_k, y_k)^T$ on the ellipse which would result when the ellipse is traversed according to the prediction horizon k into the direction leading to future estimates and starting from the point on the ellipse which is closest to the most recent data point at sampling instant i. In order to generally determine points on the ellipse which are closest to given points, an iterative search algorithm is employed which uses the parametric form of the ellipse and determines the corresponding ellipse parameter φ. The algorithm starts with an initial guess calculated from the coordinates of the given point and converges to a solution usually within a few steps. In this way, the ellipse parameters

$$\{\varphi(t-w_R+1), \varphi(t-w_R+2), \ldots, \varphi(t)\} \tag{3.29}$$

are determined, each of them corresponding to a point on the ellipse lying closest to the data points (3.22), respectively. In the ideal case, e.g. when using ideal but phase-shifted sines sampled in equidistant intervals, the differences between the angles φ in (3.29) are constant, where the difference corresponds to a progression of the angle from one sampling instant to the next. Hence, the averaged angle differences

$$\overline{\Delta\varphi} = \frac{1}{w_R-1}\sum_{i=1}^{w_R-1}\Delta\varphi(t-i+1) = \frac{1}{w_R-1}\sum_{i=1}^{w_R-1}\left(\varphi(t-i+1)-\varphi(t-i)\right)$$

can be used as an estimate of the average change in the angle in between two sampling instants. In order to numerically assess the angle differences correctly in the presence of switches from $-\pi$ to π or vice versa, this can be modified to

$$\overline{\Delta\varphi} = \frac{1}{w_R-1}\sum_{i=1}^{w_R-1}\operatorname{atan2}\left(\sin\Delta\varphi(t-i+1), \cos\Delta\varphi(t-i+1)\right) \ ,$$

where $\operatorname{atan2}(y,x)$ is the four-quadrant version of $\arctan\left(\frac{y}{x}\right)$. With this information available, the point $(x_k, y_k)^T$ can finally be determined by substituting

$$\varphi(t+k) = \varphi(t) + k\overline{\Delta\varphi}$$

into the ellipse Eq. (3.4). Then the appropriate solution $\tilde{\lambda}_1$ or $\tilde{\lambda}_2$ can be selected: If

$$d_g\left(\begin{bmatrix}\hat{x}^B\\ \tilde{\lambda}_1+y_c\end{bmatrix}, \begin{bmatrix}x_k\\ y_k\end{bmatrix}\right) < d_g\left(\begin{bmatrix}\hat{x}^B\\ \tilde{\lambda}_2+y_c\end{bmatrix}, \begin{bmatrix}x_k\\ y_k\end{bmatrix}\right)$$

the solution

$$\hat{p}_m^T(t+k|t) = \tilde{\lambda}_1 + y_c$$

is chosen, otherwise

$$\hat{p}_m^T(t+k|t) = \tilde{\lambda}_2 + y_c \ .$$

If $D < 0$ there is no intersection between the straight line and the ellipse. This can for example happen due to variability in breathing or tumor motion. Several options are available to deal with this case. One might select the point on the ellipse which is closest to the straight line Eq. (3.26). However, that point would not be determined using the prediction horizon k and thus can be far away from $(x_k, y_k)^T$, so that this option is discouraged. Another option might be to select the point $(x_k, y_k)^T$ itself as solution, with the justification being the fact that this could be an appropriate extrapolation according to the ellipse fit. On the other hand, this point is definitely far away from the straight line as their x-components differ. In tests it was found that the best option in this case is to fall back to linear regression. For that, $w_R = 250$ is chosen here according to previous discussion of useful ranges for w_R in first-order linear regression. Since regression is performed online, there are not enough data points for the first 250 samples in case the linear fallback solution needs to be taken. In this case, the linear regression is computed over the number of available data points up to the maximum of $w_R = 250$.

Testing elliptical regression with ideal phase shifted sines yields optimal results with no error. This supersedes the linear regression which is not able to handle phase shifts. However, if signals do not possess a phase shift, the elliptical regression method needs to be adapted to handle this as a special case. This can be done easily by using linear regression instead. Linear regression is also incorporated if no ellipse can be fitted at all. Then the data points are usually arranged anyway in a line-shape.

While elliptical regression with these updates works very well on ideal data, either phase-shifted or not, some problems arise when applying it to the pre-recorded datasets which are not perfectly conditioned such as the ideal data. Applying the same test procedure as with linear regression discussed before, results in the situation depicted in Figures 3.13 and 3.14. At first, in Figure 3.13 the relative frequency (frequency of occurrence) of the solution type due to two fallback possibilities other than the ellipse solution is analyzed. With low values of the window size w_R, ellipses cannot be fitted frequently as fewer data points are usually arranged in a way that fits better to linear solutions (red line). With increasing w_R, the number of ellipse solutions (blue line) rises until about 750. This is due to the fact that with more data points, the chance of fitting an ellipse rises. At about a window size of 60, corresponding to the length of an average breathing cycle, the number of ellipse solutions converges. Including more data points in the fit consequently does not increase the chance of more ellipse fits. In fact, for $w_R > 750$ the relative number of ellipse fits reduces to approximately 0.5, which can be attributed to a non ellipse-like distribution of the regression points, making it impossible for the chosen fitting algorithm to find an appropriate ellipse. In about 1% of the cases the last fallback (green line) is used (see Eq. (3.25)) which occurs even when the linear first-order fallback solution fails. The results of the error metrics shown in Figures 3.14(a) – 3.14(c) generally follow the same pattern as polynomial regression. The errors for smaller w_R are less than the ones for linear regression because the linear fallback solution is used more frequently with the maximum available data points (up to 250). Comparing the RMS error between first-

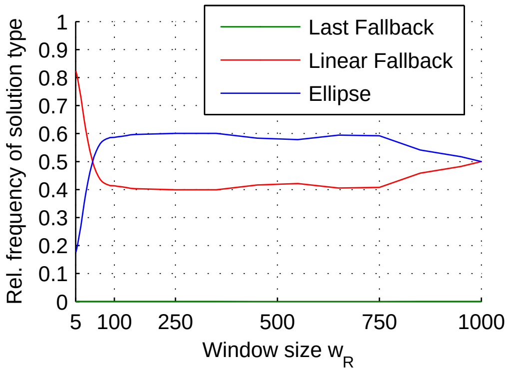
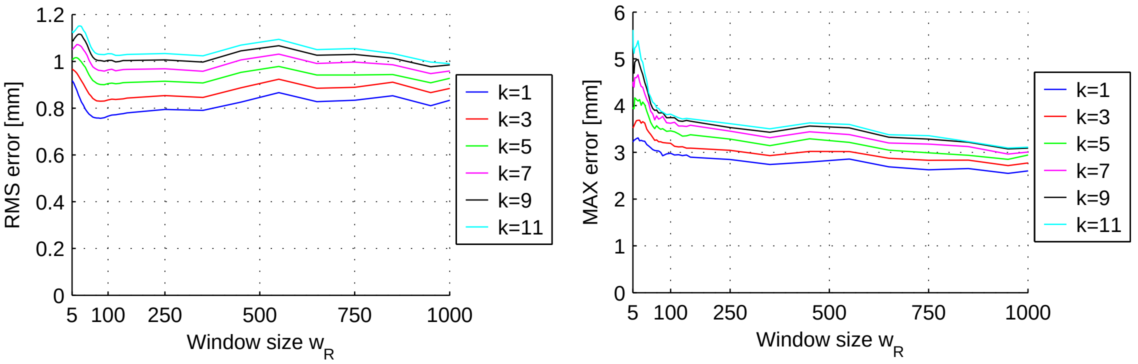
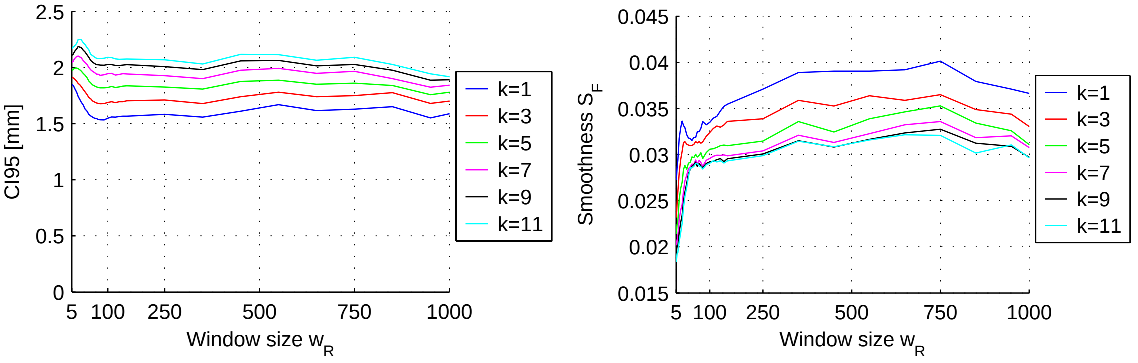


Figure 3.13: *Elliptical regression solution types*



(a) Average RMS errors (b) Average MAX errors



(c) Average CI95 (d) Average smoothness S_F

Figure 3.14: *Online elliptical regression depending on window size w_R*

order linear and elliptical regression, it turns out that linear regression performs better only for $k = 1$ and $w_R < 350$, otherwise the elliptical regression leads to less average RMS errors. However, the performance increase is minor as can be seen by comparing absolute values of the RMS errors. Opposed to that, maximum errors are in general smaller for linear regression, especially for $w_R = 250$ which is in the range of recommended values for linear regression. The CI95 metric yield slightly improved results on the elliptical regression for $k \leq 5$ and degraded results for $k > 5$. The smoothness metric for the elliptical regression clearly shows larger smoothness values, i.e. less smoothness than for linear regression. The reason is that often ellipses cannot be fitted on the given data and the fallback solution is taken instead (cf. Figure 3.13). During switching it is not guaranteed that two consecutive solutions lie closely together, so that better smoothness can be achieved when no switching occurs as for example in linear regression alone. Even when switching does not occur during several sampling instants, ellipses fitted to the real data are not very stable as time progresses. Often the center of the ellipses as well as their semi axes a and b are changing significantly from one sampling instant to the next, leading to strongly changing extrapolations and consequently to degraded smoothness. However, as already outlined before, an alternative approach needs to be used in conjunction with elliptical regression in order to provide a solution in every sampling instant.

Choice of Regression Method for Class III Algorithms Given the improved smoothness for first-order linear regression, the generally similar performance of linear and elliptical regression on the error metrics but with better performance on the maximum errors for the linear regression, the first-order linear regression with $w_R = 250$ is chosen for the subsequent prediction algorithms which are formulated as class III. Note that this choice is based on the given datasets. Using other datasets having a different characteristics might lead to different conclusions on the regression method yielding the best results. This can occur due to the fact that available datasets of breathing and tumor motions are not completely synchronized. Apart from unknown timing offsets, they may possess unknown temporary phase shifts caused by varying latencies in the acquisition systems. Additionally, the extremely low sampling rate of the EPID system introduces further difficulties in establishing a good correlation, which can be used in online regression.

To summarize the issue of discussed online regression methods used to derive tumor predictions from breathing predictions: Given ideal data, the elliptical regression is preferred because it can naturally deal with phase-shifted as well as with non phase-shifted data. Linear regression works well on non phase-shifted data but fails on phase-shifted data. However, applying these methods to real pre-recorded data in an online fashion, that is, only a certain window reaching into the past of the data history is considered, it turns out that among these methods the first-order linear regression slightly supersedes the others especially when smoothness is taken into account. Polynomial as well as elliptical regression applied statically on the whole dataset suggested an improvement over the first-order linear regression. However, this improvement is present merely in mild form in the online regression due to the performed extrapolation on which the error criteria is based instead of the residuals between the data points from which the fit is determined in the static case. Additionally, the online linear regression supersedes the static version, leading to the conclusion that given the pre-recorded datasets the relationship between breathing and tumor motion has to be continuously updated. Therefore it is also useful to limit the size of the sliding window in order to adapt to changing motion patterns within an appropriate time.

Other correlation models are also available. [90] uses several linear models whose parameters are fitted offline. Consequently, as presented, the model can only be used in retrospect. [91] proposes a physical model with a several springs and dashpots to model the correlation. The CyberKnife system uses an approach similar to the linear method presented here, with adjustable polynomial orders and a distinction between inhalation and exhalation phase [92]. As will become clear during the course of this work, the correlation model based on regression will mostly serve as baseline method to which others will be compared.

3.2.4 Prediction Methods

Various prediction methods for different purposes ranging from industrial applications, e.g. [93], to respiratory or tumor motion prediction have been proposed already in literature. Among model-free (in the sense of no dynamical model describing the motion in physical terms) prediction methods, often autoregressive (AR) models are applied in different variants, where their parameters are updated by various algorithms. AR models are formulated over a certain past signal history, where current or future values can be predicted by a linear combination of past observations with a parameter vector. For example, [94–96] apply the least-mean-square (LMS) algorithm [97] to update the parameters of a linear AR model from which breathing predictions are determined. [50, 96] additionally apply the recursive least-squares (RLS) algorithm for parameter updates, which is found to provide a higher prediction accuracy and faster convergence than the LMS algorithm. Furthermore, [96] applies a wavelet-based LMS algorithm for updating parameters. Other works incorporate autoregressive moving average (ARMA) models [98] together with the RLS algorithm, where the model additionally includes an estimation of future prediction error. [99] based the breathing prediction onto a seasonal autoregressive integrated moving average model, which is deemed more suitable to deal with instationarities of breathing motion. Rather than incorporating past observations directly into the model, it is regressed over discrete differentiations of the breathing signal. [100] introduces a periodic ARMA model for prediction, which estimates both periodic and non-periodic components of breathing motion.

Other model free prediction methods include simple linear extrapolation filters as in [28, 101], usually showing larger prediction errors when the prediction horizon increases. Often they are merely used as baseline methods for comparison with other prediction methods. Quite often, prediction methods based on artificial neural networks (ANNs) [102] are found in literature with various structures or variations. For example, [94] compares the LMS algorithm to an ANN. [103] uses an ANN approach with offline training of a single layer perceptron with linear and nonlinear output functions. [29, 101] propose multilayer perceptrons with a single layer of hidden neurons and a linear output function. [104] compares several variants of ANNs with different training algorithms to other prediction methods for respiratory motion prediction. Furthermore, combinations of ANNs and fuzzy logic methods [105] as well as pure fuzzy logic approaches [106] are found. One of the main drawbacks of the ANN methods is the need for a relatively long training phase, either online or offline. Newer model free prediction methods make use of a support vector machine, usually employed for classification tasks. These can be reformulated to perform support vector regression, which also updates parameters of an AR-like model or other nonlinear models based on the chosen kernel. Predictors based on support vector regression are often found to supersede others, as in [107, 108]. Often Gaussian kernels are employed instead of linear kernels. Another interesting approach [109] introduces a family of linear predictors based on the assumption that the difference between original and delayed breathing signal is constant. By expanding the prediction error to account for changes of higher order derivatives, predictions can be improved, when the assumption does not hold anymore.

Similar to that idea, [110] proposes a local dynamic model formulated in an augmented space. Since a dynamic model is introduced describing the motion in physical terms, this method belongs to the class of model-based predictors. In this method, the original breathing signal and the delayed signal are used to span an augmented space in which the respiratory motion describes a circular or elliptical path. An extended Kalman filter estimates the states of the dynamic model which includes the angular velocity of the circular motion into the states apart from position and linear velocities. A more simplistic approach is given in [104, 111], which assume a constant velocity and constant acceleration model to describe tumor motion. The models are formulated as state-space models where the states are estimated by a linear Kalman filter. An interacting multiple model approach is used to fuse both filters to improve prediction accuracy. [112] models the motion with a spring-mass model, where the states are estimated with an extended Kalman filter. In [101] an initial guess of a state transition matrix of a state-space model is used to describe the motion. The state transition matrix is then continuously updated by an expectation maximization algorithm. Efforts have also been done to fit a sinusoidal model [113] in time domain representation to past measurements. Predictions are hereby determined from the fitted sinusoidal model.

Different from these methods, in [114–116] breathing stages like exhalation and inhalation are modeled by states of a finite state machine. Transitions of states are then analyzed using a hidden Markov model. The states are associated with observables like tumor position and velocity from which predictions are generated. Motivated from frequency domain, [117, 118] propose a Fourier linear combiner for breathing prediction which builds a Fourier series model and continuously updates the Fourier coefficients. This was applied for real-time compensation of tremor during percutaneous needle insertion in robot-assisted microsurgery. [119] introduces a kernel density estimation approach for breathing prediction, in which probability distributions of future breathing positions are estimated.

Other methods are based on the idea to find current breathing patterns in past ones and to determine the prediction from best matching patterns. [120] proposes the concept of subsequence similarity matching where piecewise linear parts of breathing motion are associated with finite states. [121] constructs an augmented space from past respiratory motion measurements and additionally performs a local regression in the augmented space. [92, 122] introduce a pattern matching approach based on sliding windows of past data, which is integrated into the Synchrony® Respiratory Tracking System, a subsystem of CyberKnife. [123] use the concept of embedding vectors to search for best matching breathing patterns in past data which are defined by the embedding vectors. This method was successfully employed for motion compensation in beating heart surgery.

This work investigates several of the aforementioned prediction methods in the following sections where preliminary results are given in [64] and [124].

3.3 Adaptive Filters

Assuming a real-valued signal y is sampled at equidistantly spaced sampling points, where the current one is denoted with index t, the resulting time series $y(t)$, $y(t-1)$, ..., $y(t-n_a)$ is said to be a realization of an AR process [125] of order n_a if it fulfills the model equation

$$A(q)y(t) = e(t) \tag{3.30}$$

in which $y(t) \in \mathbb{R}$ is considered the current output of the model in sampling instant t. q^{-1} is the backward shift operator

$$q^{-1}y(t) = y(t-1) \ ,$$

$A(q)$ is a polynomial of degree n_a of the form

$$A(q) = 1 + a_1 q^{-1} + a_2 q^{-2} + \cdots + a_{n_a} q^{-n_a}$$

with model parameters $a_i \in \mathbb{R}$, $i = 1, 2, \ldots, n_a$ and $e(t) \in \mathbb{R}$ is a disturbance drawn from white noise. Rearranging Eq. (3.30) to

$$y(t) = -\sum_{i=1}^{n_a} a_i y(t-i) + e(t) \tag{3.31}$$

relates the current output $y(t)$ to past samples of signal y by a linear combination with the model parameters a_i plus the noise term $e(t)$. Since the model is expressed in the form of a regression model and $y(t)$ is regressed on past values of itself, the model is called "autoregressive".

Assuming the parameters of the model in Eq. (3.31) are already known from information up to time $t-1$, the undisturbed version of this model can serve as a one-step predictor in the case of unmeasured disturbances, yielding

$$\hat{y}(t|t-1) = -\sum_{i=1}^{n_a} a_i y(t-i) \; . \tag{3.32}$$

Shifting by one sampling instant and using the latest measurement $y(t)$, Eq. (3.32) can be rewritten to

$$\hat{y}(t+1|t) = -\sum_{i=1}^{n_a} a_i y(t-i+1) \; . \tag{3.33}$$

Introducing the parameter vector $\theta = [a_1, a_2, \ldots, a_{n_a}]^T$ and the vector of observations $\varphi(t) = [-y(t), -y(t-1), \ldots, -y(t-n_a+1)]^T$, Eq. (3.33) can be simplified to

$$\hat{y}(t+1|t) = \theta^T \varphi(t) = \varphi^T(t)\theta \; . \tag{3.34}$$

Based on the one-step prediction, the general k-step predictor with $k \in \mathbb{N}$ can be derived by iteratively replacing the newest element in $\varphi(t)$ with its prediction performed in the previous step. Since a prediction is "plugged-in" into the observation vector instead of an observation which is not available yet, the predictor is commonly referred to as plugin-predictor [126] and can be described in general by

$$\hat{y}(t+k|t) = \varphi^T(t)\Theta^{k-1}\theta \tag{3.35}$$

with

$$\Theta = \left[\begin{array}{c|c} \theta & \begin{array}{c} I_{n_a-1} \\ 0^T_{n_a-1} \end{array} \end{array} \right]$$

and $\Theta^0 = I_{n_a}$, where I_{n_a} is the $n_a \times n_a$ identity matrix and $0^T_{n_a-1}$ is a row vector of $n_a - 1$ zeros.

Instead of iterating over the AR model, it is also possible to acquire a k-step predictor in a direct approach similar to Eq. (3.34) by

$$\hat{y}(t+k|t) = \varphi^T(t)\theta_k \; , \tag{3.36}$$

where θ_k refers to a parameter vector with which only k-step predictions can be performed. In order to perform multistep predictions with the direct predictor, it is therefore necessary to use separate parameter vectors θ_i for $i = 1, 2, \ldots, k$.

When modeling either breathing or tumor motion by means of an AR model, $y(t)$ represents the current one-dimensional measurement (or interpolation) of either the breathing or tumor motion signal, respectively. Due to the variability in motion patterns, model parameters a_i need to be updated continuously in order to adapt to changes in momentary frequency and amplitude, essentially making the parameter vector θ and the AR model time-variant. Several methods exist in order to estimate the model parameters from past samples of signal y, which will be outlined in the following subsections.

3.3.1 Least-Mean-Square Algorithm

At first, the class I formulation is presented which can be applied to a single breathing or tumor motion signal. The following subsection discusses the influence of the free parameters of the least-mean-square algorithm on various performance metrics when using breathing signals and provides insights on how optimal parameter settings can be selected. With known parameters, the next subsection shows the achievable tumor prediction performance when employing the developed regression scheme based on breathing predictions, i.e. the class III variant. Next, an extension is proposed to achieve a class II formulation, thereby fusing breathing and tumor motions. Value ranges of the free parameters of the class II extension are derived and the resulting tumor prediction metrics are compared to the ones from the class III variant.

3.3.1.1 Class I Formulation

One popular method to estimate model parameters of an AR model online is the LMS algorithm originally developed by Widrow and Hoff [127]. It is the stochastic analogue of the method of steepest descent which solves an optimization problem by minimizing the mean square error. The algorithm [125] can be summarized for the direct predictor according to Eq. (3.36) as follows: Starting from an initial estimate of the model parameters $\hat{\theta}_k(0) \in \mathbb{R}^{n_a}$ at time $t = 0$, the estimated parameters $\hat{\theta}_k(t)$ (as opposed to the true but unknown model parameters $\theta_k(t)$) are recursively updated in each sampling instant by

$$\hat{\theta}_k(t) = \hat{\theta}_k(t-1) + \mu \left(y(t) - \hat{\theta}_k^T(t-1)\varphi(t-k) \right) \varphi(t-k) \ . \tag{3.37}$$

For the computation of the prediction error

$$e_k(t) = y(t) - \hat{\theta}_k^T(t-1)\varphi(t-k) \tag{3.38}$$

in Eq. (3.37) at sampling instant t, the latest measurement $y(t)$ is compared with the k-step ahead predicted value computed from the estimated parameters $\hat{\theta}_k^T(t-1)$ from the previous sampling interval and n_a past observations $\varphi(t-k) = [-y(t-k), -y(t-k-1), \ldots, -y(t-k-n_a+1)]^T$ shifted by k sampling intervals. The shift is necessary since with $\hat{\theta}_k$, the k-step prediction for the current sampling interval t needs to be computed, which can only be achieved using $\varphi(t-k)$. The increment in model parameters in Eq. (3.37) is proportional to the prediction error modified by a small step-size parameter $\mu > 0$ which can be used to control the amount of adaption in each update in order to make the algorithm more robust to noise or to avoid instabilities. For the "direction" of the adaption in terms of the steepest-descent idea of the algorithm, $\varphi(t-k)$ is chosen. The term $\varphi(t-k)e_k(t)$ actually is the simplest form of an unbiased estimator for $\mathrm{E}\left[\varphi(t-k)e_k(t)\right]$ (with $\mathrm{E}\left[\cdot\right]$ being the expectation operator), which minimizes the cost function underlying in the LMS algorithm.

Having obtained the updated parameter vector, the k-step prediction is computed by

$$\hat{y}(t+k|t) = \hat{\theta}_k(t)\varphi(t) \ . \tag{3.39}$$

The standard LMS algorithm suffers from one major deficiency known as the gradient noise amplification problem which occurs especially for large $\varphi(t-k)$. Additionally the choice of μ is not invariant to scaled input data. Therefore the normalized LMS was proposed by Nagumo and Noda [128], in which the adaptation increment is scaled by the 1-norm of $\varphi(t-k)$. Later this was extended to a general p-norm [129]. Here the usually employed 2-norm was chosen, leading to the parameter update

$$\hat{\theta}_k(t) = \hat{\theta}_k(t-1) + \frac{\mu}{\alpha + \|\varphi(t-k)\|^2} \Big(y(t) - \hat{\theta}_k^T(t-1)\varphi(t-k) \Big) \varphi(t-k) \ . \tag{3.40}$$

Normalization introduces a numerical problem when $\varphi(t-k)$ becomes small. In order to avoid division by zero, a small constant $\alpha > 0$ is introduced in the denominator with a value of 10^{-5} in this work.

Convergence of the LMS algorithm (either standard or normalized) depends on the choice of the step-size parameter μ and the correlation matrix of the input vector $\varphi(t-k)$. For stationary input signals, the coefficients $\hat{\theta}_k$ converge to the optimal weights of the Wiener filter [125]. In the standard LMS algorithm, an approximate upper bound for the step-size parameter μ can be found from the inverse power of the input signal, whereas in the normalized LMS algorithm, it can be shown that $0 < \mu < 2$ [130, 131] in order for it to converge in the mean square error. The smaller μ is chosen, the smaller the adaptation increment becomes, which in turn reduces the convergence rate. Since a feedback of the measured output $y(t)$ is introduced in Eq. (3.40) for the update of the parameters, these can become unstable if the input signal changes too fast in comparison with the convergence rate. In such a case, μ can be reduced in an attempt to mitigate this problem, thereby constraining convergence at the same time.

The dead time or "learning time" of the LMS algorithm before the first prediction can be performed is at least lower-bounded by the number of samples n_a which need to be acquired for the construction of $\varphi(t)$ in prediction equation (3.39). However, the first parameter updates can be performed at $t = n_a + k$ because $\varphi(t-k)$ needs to be defined in Eq. (3.40), resulting in a different dead time depending on k for the direct predictor. For the plugin predictor only a one-step prediction is necessary, from which a k-step prediction is iteratively determined, leading to a dead time of $t = n_a + 1$ sampling intervals being independent of k.

For the LMS prediction of breathing or tumor motion, the choice of

$$\hat{\theta}_k(0) = \begin{bmatrix} -1 & 0 & 0 & \dots & 0 \end{bmatrix}^T$$

as initial estimate of the parameter vector $\hat{\theta}_k(0)$ is suitable as it brings the early predictions close to the recent value, especially for smaller values of k, thus enabling fast convergence.

3.3.1.2 Class I Parameter Selection

Applying the LMS algorithm on an AR model of order n_a while having an one-dimensional input signal, two design parameters n_a and μ need to be found. Therefore exhaustive simulations on 200 pre-recorded breathing datasets were performed, which were interpolated to a sampling time of 50ms (20Hz). The parameter n_a was varied between 2 and 150 with different step sizes of 1, 2, 5 and 10, depending on the range of n_a itself. Values for the parameter μ were: 0.001, 0.005, 0.01, 0.025, and the range between 0.05 and 1.0 in steps of 0.05. For each parameter setting, the LMS predictor determined predictions up to a horizon of $k = 11$ (corresponding to 550ms) in all three translational axes. For each dataset, the following 3D criteria were computed: prediction ratio (PR), 95% confidence interval (CI95), mean absolute error (MAE), root-mean-square error (RMS), maximum error (MAX), smoothness criteria (S_F, S_J, S_{TS} and S_{SNR}) and convergence time $\tau_{c_{max}}$ with $c_{max} = 25\%$ (CT25) and $c_{max} = 10\%$ (CT10).

At first, instabilities were analyzed for the plug-in and for the direct predictor. Instabilities were determined by negative prediction ratios. Note that if the negative value is small, then there is technically no instability, but the prediction failed delivering errors large enough so that the RMS value of the errors is larger than the RMS of signal to be predicted. Even if the LMS predictor was able to successfully predict one or two axes without instabilities, if an instability occurred on the remaining axes, the complete trial is thereby counted as unstable. For $k = 1$, no instabilities occurred in both predictors. However with increasing prediction horizon, the number of instabilities rises as can be inferred from Figure 3.15 for the direct predictor and from Figure 3.16 for the plug-in predictor. Naturally, a dependency between instabilities and parameter settings can be observed. For small values of n_a more instabilities occur as for larger values. A similar observation can be made for μ as well, however with a weaker impact. Especially for larger prediction horizons a “valley” emerges for values for μ between 0.005 to 0.01 with a low number of instabilities for both predictors. The location of the valley changes with

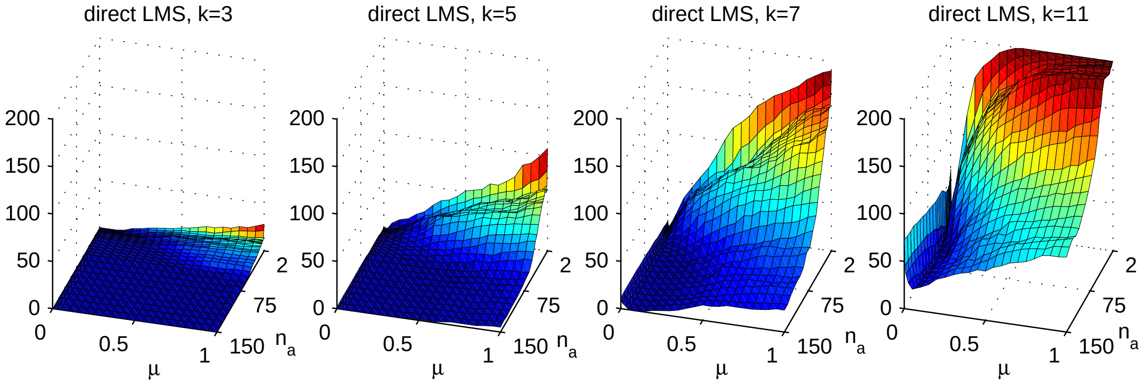


Figure 3.15: *Number of unstable trials among 200 datasets of the direct LMS predictor in relation to parameters n_a and μ*

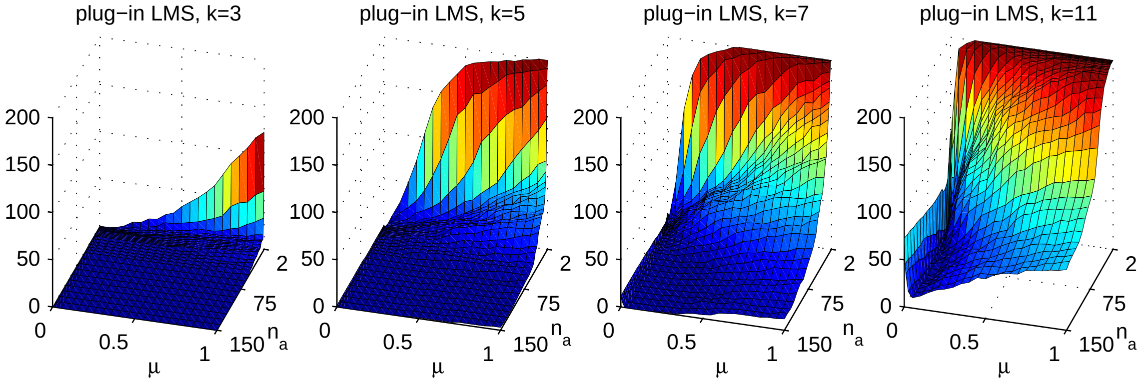


Figure 3.16: *Number of unstable trials among 200 datasets of the plug-in LMS predictor in relation to parameters n_a and μ*

increasing n_a, moving to values for μ between 0.025 to 0.1. It can be seen that choosing μ too small (i.e. less than 0.005 for small n_a or less than 0.025 for larger n_a) is not beneficial in terms of instabilities. Furthermore, the choices of parameters leading to none or only a few instabilities become confined to an increasingly narrow region with increasing prediction horizon. This behavior can be attributed to "instationarities" in the signal caused by changing breathing patterns, such that the model in Eq. (3.36), which is applied to the latest past of the signal described by $\varphi(t)$, matches lesser than several sampling instants before, leading to the necessity of adapting the parameters even more, thereby increasing the chances of instabilities. Comparing the plug-in to the direct predictor, it turns out that even for lower prediction horizons, instabilities occur more frequently for small values of n_a, but for larger n_a the surface in Figure 3.16 stay flatter at a lower level. Therefore the parameters for the plug-in predictor can be chosen more freely for low prediction horizons without leading to instabilities. For larger prediction horizons, the constraints equal the ones found for the direct predictor.

In order to find suitable parameter settings which work well on the majority of the prerecorded datasets, the average prediction ratio (averaged over the results for each dataset while neglecting unstable trials) is considered as a major criterion, which is depicted in Figure 3.17 for the direct predictor and in Figure 3.18 for the plug-in predictor. As already stated before, the plug-in predictor equals the direct predictor for $k = 1$, which can also be seen by comparing the result plots. In this case, the effect of both n_a and μ on the average prediction ratio is minor, especially for the region given by $n_a \geq 50$ and $\mu \geq 0.025$, in which large average prediction ratios of 90.0% to 91.29% are reached irrespective of the parameter values. For small values of $\mu < 0.025$, the step size in the LMS adaption is in general too small, resulting in slower convergence of parameters and thus in degraded overall prediction performance. Similarly, for $n_a < 50$ (which is slightly less than the length of an average breathing cycle) and especially for $n_a \leq 5$, the order

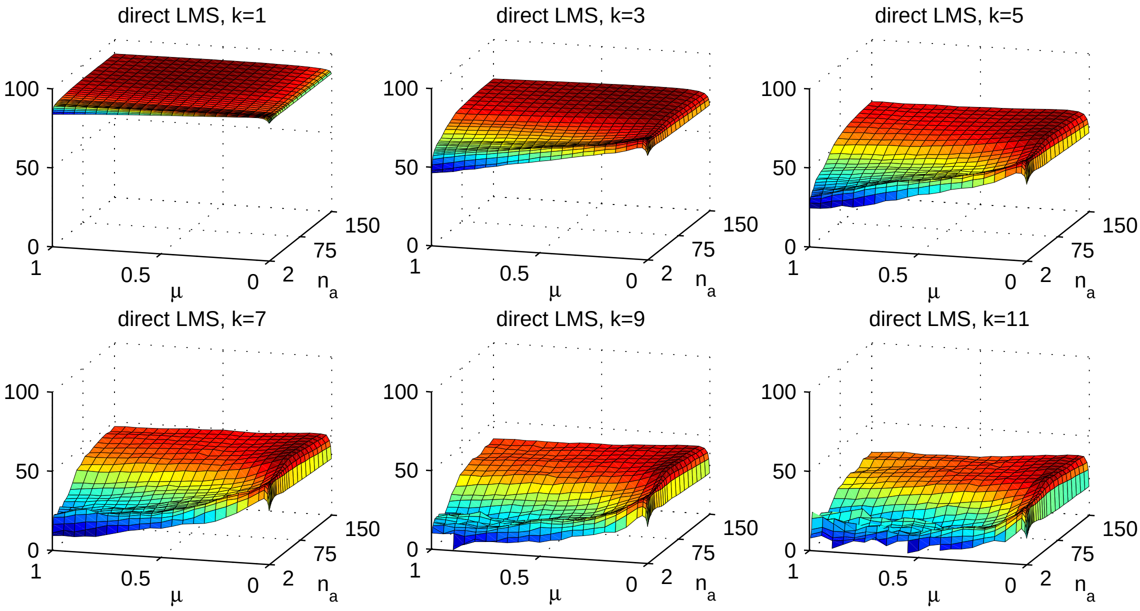


Figure 3.17: *Mean prediction ratio of 200 datasets of the direct LMS predictor in relation to parameters n_a and μ*

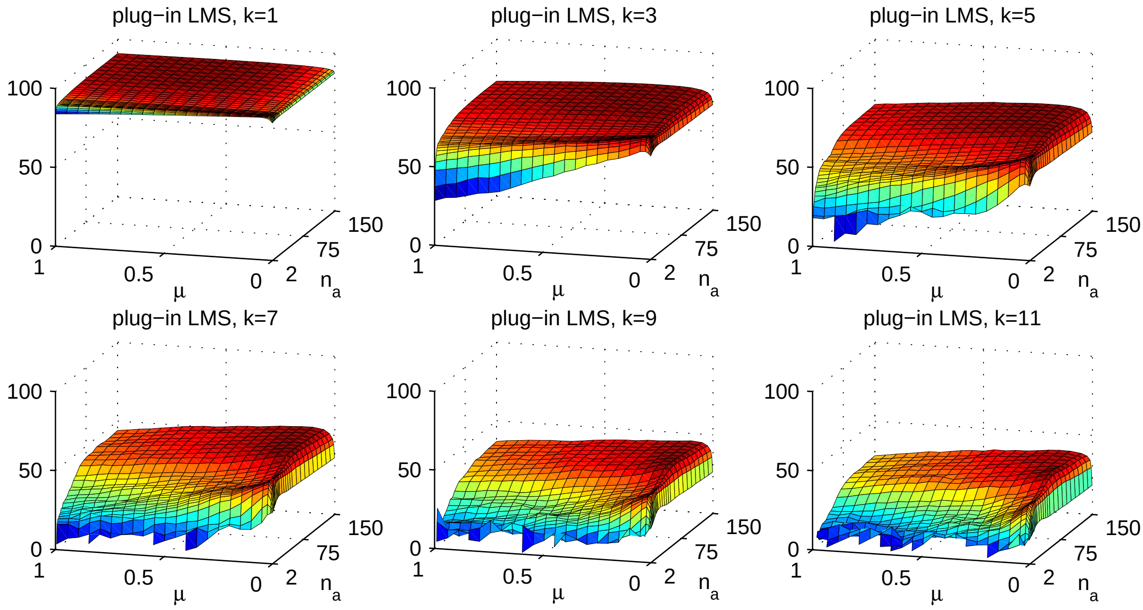


Figure 3.18: *Mean prediction ratio of 200 datasets of the plug-in LMS predictor in relation to parameters n_a and μ*

of the AR process is chosen too small in order for the AR model to sufficiently capture the dynamics of the motion. For prediction horizons $k > 1$, the region of parameters with maximum PRs gradually move to a narrow band with $0.025 \leq \mu \leq 0.1$ and $60 \leq n_a \leq 100$. Comparing the maximum PRs for each depicted prediction horizon, it turns out that the maximum average PR decreases approximately linearly starting from $k = 1$ with 91.29% to $k = 11$ with 40.49% using the direct predictor and 42.25% using the plug-in predictor. This performance degradation as the prediction horizon increases is expected. Given information about a signal until time t, it is more difficult for a predictor to come close to a future value at time $t + k$ as k is increased because signal characteristics can change. Additionally, when using a linear predictor as in the case of an AR model, it is especially hard to acquire good predictions in the vicinity of peaks when direction of the motion changes.

A closer inspection of the regions with maximum PRs and the regions with a small number of instabilities reveals that these regions have a huge overlap, leading to the conclusion that useful parameter sets can be found which lead to only a few instabilities and for which the average PR is maximized. However, an individual parameter set should be chosen for each k, as the region of maximum PRs changes with k for both the direct and plug-in predictor. This can be accommodated for the direct predictor easily by running each instance for a separate k with different parameter settings. However, this necessitates the plug-in predictor to run separate instances with appropriate parameter settings whereas normally only one instance would be sufficient.

Analyzing plots of the remaining averaged prediction accuracy criteria (MAE, RMS, MAX, CI95) over the parameter settings, a similar (but inverted) “landscape” can be observed as in the PR plots in Figures 3.17 and 3.18 in which minimum criteria values can be found for the same regions as in the PR plot, hence supporting the choice of parameters from the corresponding regions. This is further supported when analyzing these criteria in terms of the standard deviation instead of the average, which is a measure of the spread of the criteria values among the datasets in relation to the parameter settings. In these plots, lower values for standard deviations of the criteria can be observed for the same regions as well.

Representative for the various smoothness criteria considered, Figures 3.19 and 3.20 depict the smoothness measure S_F in relation to the LMS parameters for the direct and plug-in predictor, respectively. The smoothness measures S_{TS} and S_J yield similar plots,

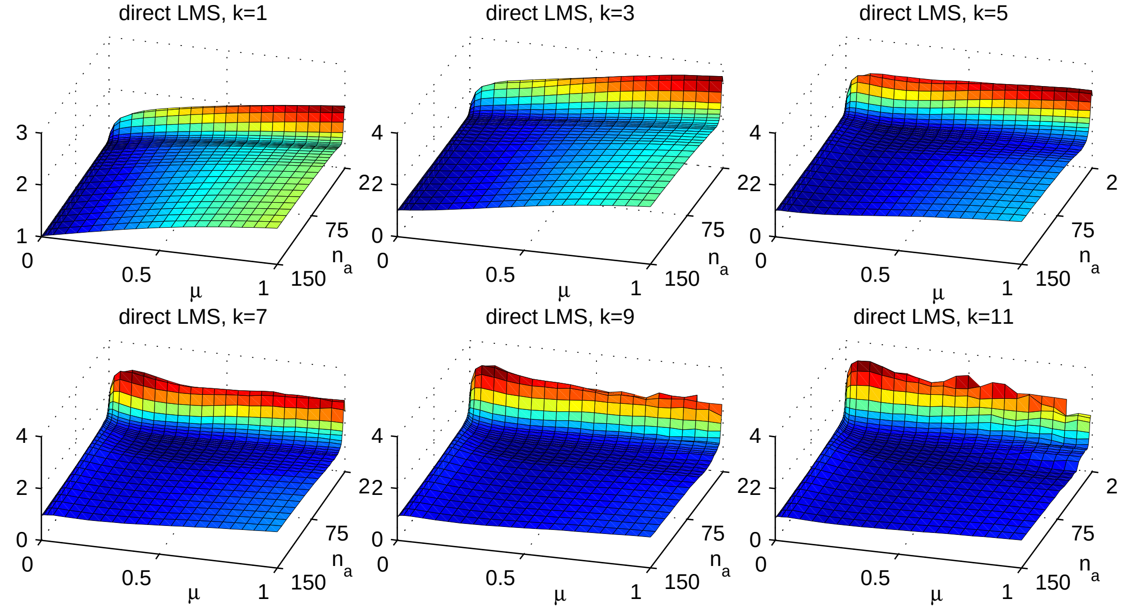


Figure 3.19: *Mean smoothness metric S_F of 200 datasets of the direct LMS predictor in relation to parameters n_a and μ*

whereas S_{SNR} results in a logarithmic version of these plots due to its definition. In general, small values for μ lead to smoother predictions, especially for small k. This is attributed to the fact that a small μ decreases the amount of adaptation in the parameter update equation of the LMS and consequently predictions become smoother. Using the plug-in predictor, smoothness is approximately linearly dependent on μ, where small values for μ lead to similar smoothness measures for each k. For larger μ, the smoothness value rises with increasing k. The behavior is quite different with the direct predictor, especially for $k > 3$ where the smoothness surface is generally flatter with slightly smaller values. This suggests that the direct predictor leads to smoother predictions than the plug-in predictor. Additionally, there is a significant decrease in smoothness in both predictors for small values of n_a irrespective of μ, where the limit of n_a when the smoothness value starts to increase is in the range 8-10. This behavior results from the low-pass filtering characteristics of the underlying AR model for which larger orders of n_a lead to more filtering and consequently to smoother predictions.

Plots of the convergence time (not shown here) have a structure similar to the ones of the smoothness measure S_F for both plug-in and direct predictors. The major difference between them is that a significant increase in average convergence time is observed for $n_a < 60$ whereas the dependence of μ approximately follows the inverted PR plots. As the convergence time measure is based on errors, the smallest convergence times can be

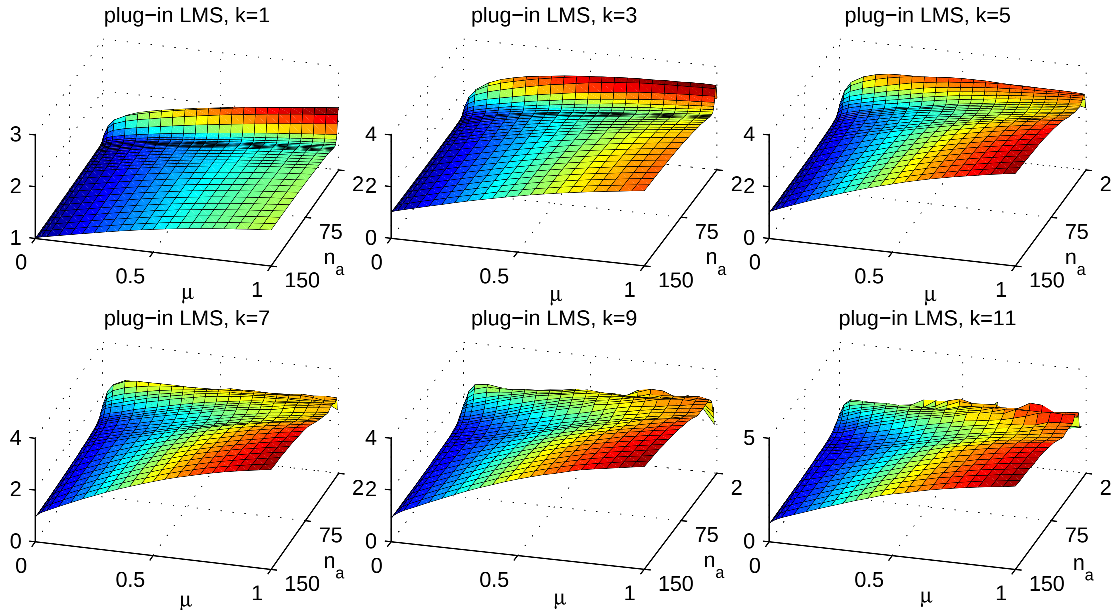


Figure 3.20: *Mean smoothness metric S_F of 200 datasets of the plug-in LMS predictor in relation to parameters n_a and μ*

observed for regions of parameter settings with maximum PR.

The results discussed so far were based on data averaged over all datasets. However, the question remains if the selection of best parameters based on average criteria is also useful for each individual dataset. In order to approach this question, the parameter settings leading to the 20 largest PRs were determined for each dataset individually and then accumulated over all datasets, so that for each k the distribution of the 20 best parameter settings can be visualized as in Figure 3.21 for the direct predictor. Since the plot for the plug-in predictor is quite similar, it is omitted here. Comparing the region in the vicinity of peaks to the plots of maximum PR for each k separately, it turns out that the largest

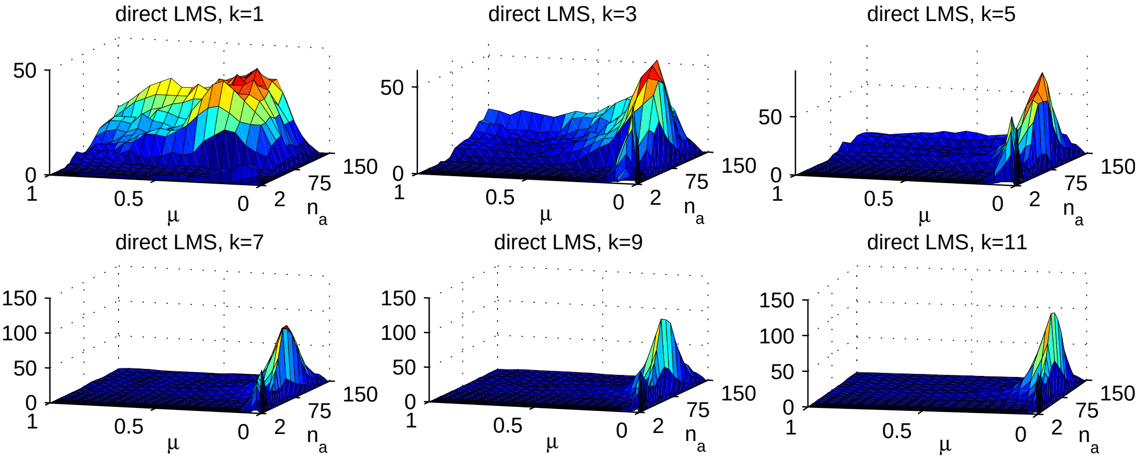


Figure 3.21: *Distribution of 20 parameter settings of the direct predictor with largest PR for each dataset and accumulated over all datasets*

parts of the regions are overlapping. Thus it can be stated that choosing the parameters from the averaged PR is feasible in order to simultaneously reach satisfactory performance for each individual dataset as well.

To summarize, best parameter settings for the LMS predictor should be chosen for each prediction horizon individually. It is thereby feasible to acquire useful parameter values from regions with maximum PR. As discussed before, the remaining mean metrics and especially the averaged standard deviation of the PR also showed best performance in these regions. Since the 200 breathing datasets were acquired from 24 patients which show a broad range of different characteristics, the results can be generalized in statistical terms so that for the same regions of best parameter settings, similar performance can be expected when predicting other patient's breathing. Judging from the prediction accuracy metrics given the best parameter settings, the plug-in predictor possesses a slightly higher performance than the direct predictor. But in terms of instabilities and smoothness, the plug-in predictor lags behind. Since also smooth trajectories are an important requirement for predictions, the direct predictor will be considered further.

In order to deal with the imminent problem of instabilities, the idea from [95] seems reasonable if the LMS predictor is employed. Parallel instances of LMS predictors are executed with different parameter settings simultaneously, where the parameter settings are determined by a rule based on initial settings. Based on prediction accuracy, the predictor with the smallest errors is then chosen to deliver the final predictions.

Having obtained parameter settings for each k by choosing the one with maximum PR for each k, it is interesting to determine the prediction performance of the direct LMS predictor applied to all datasets. Figure 3.22 depicts box plots of all criteria, each of them showing the distribution of criteria values for prediction horizons between 1 and 11 in steps of 2 among all 200 datasets. In order to fuse the various metrics together in one single plot,

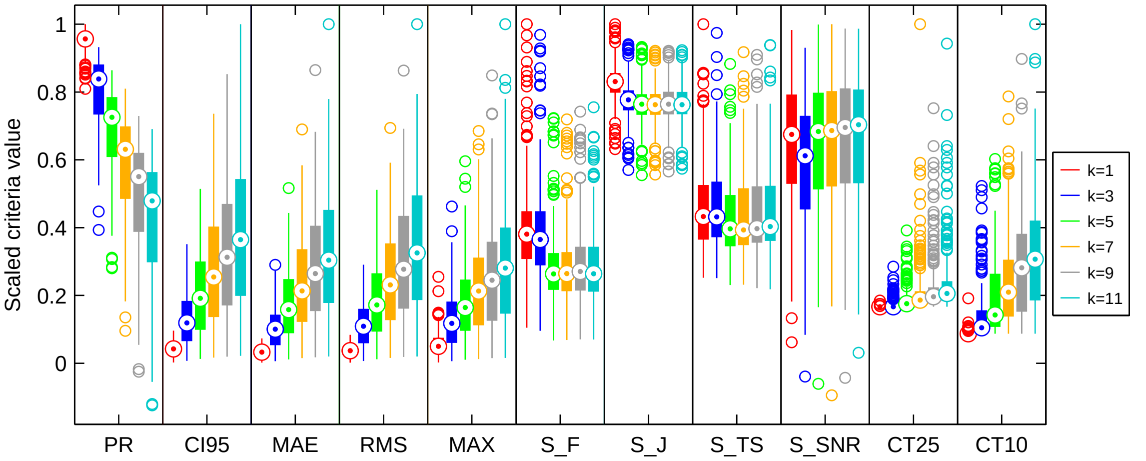


Figure 3.22: *Scaled breathing prediction metrics of parameter setting with largest average PR determined for each k separately using the direct LMS predictor; metric values are visualized by box plots for different k*

their values are scaled by the maximum of each criteria (irrespective of k). The absolute values of the main characteristics visualized qualitatively in the figure, can be drawn from Table 3.5, which lists the minimum, mean, median and maximum value for each metric and for each prediction horizon. The standard deviation is omitted here, since the box

Prediction horizon	Type	PR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]	S_F	S_J	S_{TS}	S_{SNR}	CT25 [s]	CT10 [s]
1	min	78.9	0.01	0.00	0.00	0.02	0.35	1.82	0.07	1.48	8.75	8.75
	mean	92.4	0.19	0.06	0.09	0.52	1.34	2.39	0.13	15.59	8.76	9.04
	median	93.3	0.18	0.06	0.09	0.45	1.27	2.40	0.12	16.14	8.75	8.75
	max	97.5	0.41	0.14	0.20	2.28	3.34	2.88	0.27	23.51	9.70	19.20
3	min	38.3	0.03	0.01	0.02	0.06	0.32	1.64	0.07	-0.93	8.75	8.75
	mean	78.6	0.56	0.21	0.28	1.17	1.29	2.24	0.12	14.08	9.15	14.43
	median	81.7	0.51	0.20	0.26	1.05	1.22	2.24	0.12	14.64	8.75	10.50
	max	90.9	1.50	0.57	0.69	4.13	3.23	2.71	0.26	22.25	14.95	52.45
5	min	27.3	0.05	0.02	0.03	0.09	0.22	1.60	0.06	-1.45	8.75	8.75
	mean	66.8	0.90	0.35	0.45	1.59	0.96	2.20	0.12	15.57	9.93	19.28
	median	70.7	0.82	0.31	0.41	1.46	0.88	2.20	0.11	16.35	9.22	14.28
	max	84.2	2.20	1.02	1.22	5.32	2.42	2.69	0.24	23.89	20.55	60.45
7	min	9.3	0.07	0.03	0.04	0.11	0.23	1.61	0.06	-2.25	8.75	8.75
	mean	56.1	1.21	0.47	0.61	2.04	0.95	2.20	0.12	15.56	11.30	24.41
	median	61.5	1.09	0.42	0.55	1.90	0.88	2.20	0.11	16.41	9.77	21.05
	max	78.9	3.14	1.36	1.66	6.11	2.40	2.66	0.25	23.92	52.45	78.95
9	min	-2.4	0.08	0.03	0.04	0.13	0.24	1.63	0.06	-1.03	8.75	8.75
	mean	47.5	1.43	0.58	0.73	2.32	0.99	2.21	0.12	15.81	11.92	29.46
	median	53.7	1.33	0.52	0.66	2.19	0.90	2.20	0.11	16.63	10.30	28.25
	max	71.1	3.64	1.71	2.06	7.57	2.48	2.66	0.24	23.61	39.45	90.05
11	min	-12.1	0.09	0.04	0.05	0.14	0.23	1.65	0.06	0.74	8.75	8.75
	mean	40.4	1.64	0.66	0.84	2.64	0.98	2.20	0.12	15.84	12.87	32.34
	median	46.7	1.56	0.60	0.78	2.51	0.88	2.20	0.11	16.82	10.80	30.80
	max	67.3	4.27	1.97	2.38	8.92	2.52	2.66	0.25	23.60	49.45	100.30

Table 3.5: *Minimum, mean, median and maximum breathing prediction metrics of parameter setting with largest average PR determined for each k separately; metrics are given for different k using the direct LMS predictor*

plots give a clearer picture of the distribution. From all prediction accuracy metrics (PR, CI95, MAE, RMS, MAX), the performance decrease with increasing prediction horizon k becomes obvious. With decreasing median, the IQR simultaneously increases, leading to a larger spread of the metric values among the 200 datasets. That is, the prediction accuracy becomes more and more different for the datasets as k increases. As can be seen from the whiskers of the PR box plots for $k \geq 9$, there are some datasets which yield a low prediction performance, eight of them resulting in a negative PR down to -12.1% (two for $k = 9$ and six for $k = 11$). That means, these datasets possess at least one axis where the LMS prediction failed. Given the PR value and from inspection of the individual dataset, it can be seen that there is no instability but the prediction did not converge to the true signal, with predictions being either far away or phase-shifted. Due to the fact that the CI95 is significantly less than MAX errors (whereas the difference between them are only 5% of the largest errors), it can be seen that most of the absolute errors are on average less than 1mm for $k \leq 5$ and less than 1.7mm for $k \leq 11$. All smoothness metrics (S_F, S_J, S_{TS}, S_{SNR}) indicate smoother predictions for $k > 3$. The reason for this behavior originates from the choice of parameters which differ for each k, where the difference between the parameter settings is lesser for increasing k, leading to the observed results. From the convergence time metrics (CT25 and CT10), an increase of median convergence time as well as broader spread of it can be observed with increasing k.

Instead of computing statistics of the direct LMS predictor from parameter settings with largest PR which were determined for each k individually, they can also be computed from parameter settings determined for each dataset and each k with largest PR, yielding the best achievable performance per dataset with the LMS predictor (of course given certain constraints on the parameters such as maximum μ and step sizes of the parameters). Given these individual parameter choices for each dataset, the metrics given in Table 3.6 result. Obviously negative PRs do not occur anymore, nevertheless there are still a few datasets with poor prediction performance, especially for larger k. When comparing this table of best achievable performance to Table 3.5, it becomes evident that there is a visible performance increase of 2%-4% in the mean PR when choosing the parameter setting with largest PR for each dataset individually. On the other hand the reduction in maximum errors is smaller than one might expect. Therefore choosing a separate parameter setting for each k and keeping it constant for that k is a feasible approach.

Prediction horizon	Type	PR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]	S_F	S_J	S_{TS}	S_{SNR}	CT25 [s]	CT10 [s]
1	min	80.1	0.01	0.00	0.00	0.02	0.38	1.96	0.06	1.61	8.75	8.75
	mean	92.6	0.18	0.06	0.09	0.51	1.26	2.37	0.12	15.81	8.76	9.04
	median	93.4	0.18	0.06	0.09	0.45	1.16	2.36	0.11	16.33	8.75	8.75
	max	97.7	0.40	0.13	0.19	2.29	2.98	2.80	0.30	23.94	9.65	19.20
3	min	47.1	0.03	0.01	0.02	0.05	0.55	1.75	0.07	-0.85	8.75	8.75
	mean	80.3	0.51	0.19	0.25	1.12	1.34	2.28	0.13	14.45	9.09	14.30
	median	82.6	0.49	0.18	0.24	1.01	1.16	2.26	0.12	15.14	8.75	10.53
	max	91.2	1.24	0.45	0.60	3.85	3.58	2.89	0.31	23.16	14.20	56.10
5	min	31.1	0.05	0.02	0.03	0.09	0.35	1.68	0.07	-2.60	8.75	8.75
	mean	69.5	0.82	0.31	0.40	1.54	1.33	2.30	0.13	13.81	10.43	21.09
	median	73.0	0.78	0.30	0.38	1.45	1.13	2.29	0.11	14.81	9.22	15.13
	max	86.9	2.04	0.77	0.98	4.84	3.69	2.91	0.34	22.61	35.10	90.05
7	min	19.1	0.06	0.03	0.03	0.10	0.23	1.61	0.06	-0.04	8.75	8.75
	mean	59.3	1.10	0.44	0.55	1.91	1.24	2.27	0.13	14.08	13.38	27.26
	median	63.2	1.04	0.40	0.52	1.83	1.08	2.26	0.11	15.15	10.22	23.15
	max	82.1	2.68	1.09	1.36	5.82	4.06	2.75	0.32	23.19	57.30	105.60
9	min	8.4	0.07	0.03	0.04	0.11	0.22	1.62	0.06	0.49	8.75	9.00
	mean	51.1	1.32	0.53	0.67	2.23	1.21	2.25	0.13	14.31	15.28	30.62
	median	55.6	1.25	0.49	0.63	2.11	1.03	2.25	0.11	15.31	10.75	27.47
	max	77.0	3.19	1.29	1.60	6.94	4.47	2.94	0.46	23.19	90.05	105.60
11	min	2.4	0.08	0.03	0.04	0.12	0.23	1.63	0.06	0.67	8.75	8.75
	mean	44.5	1.50	0.61	0.77	2.48	1.12	2.23	0.13	14.78	16.35	32.22
	median	49.0	1.45	0.57	0.73	2.44	0.98	2.23	0.11	15.60	11.50	28.80
	max	72.4	3.65	1.49	1.84	8.45	4.03	2.83	0.45	23.58	90.05	105.60

Table 3.6: *Minimum, mean, median and maximum breathing prediction metrics of parameter setting with largest PR determined for each dataset and each k individually; metrics are given for different k using the direct LMS predictor*

3.3.1.3 Class III Evaluation

As outlined in Section 3.2.3, the direct LMS predictor in class I formulation with different parameter settings for each prediction horizon k, can be extended to a class III algorithm by including the first-order linear regression with a sliding window size $w_R = 250$. For this evaluation, 11-step (multi-step) predictions are determined for the breathing motion from which corresponding tumor predictions are determined according to the mapping given by the linear regression function. This is updated as soon as new tumor position information becomes available. For each dataset the breathing axes with largest correlation coefficient to the tumor y-axis was determined, so that the best↦y correlation was used in the regression. Metrics of the results on tumor prediction based on online regression are given in Table 3.7.

Prediction horizon	Type	PR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]	S_F	S_J	S_{TS}	S_{SNR}	CT25 [s]	CT10 [s]
1	min	-80.2	0.33	0.10	0.15	0.55	0.28	1.79	0.01	1.36	8.75	8.75
	mean	34.6	2.19	0.89	1.12	3.70	1.12	2.31	0.12	15.65	15.19	27.93
	median	36.6	1.48	0.61	0.76	2.39	1.10	2.32	0.11	16.51	10.92	23.80
	max	81.7	15.84	5.45	6.53	23.69	2.93	2.78	0.31	25.06	49.50	107.40
3	min	-82.8	0.32	0.10	0.15	0.55	0.24	1.70	0.01	1.17	8.75	8.75
	mean	33.4	2.22	0.90	1.14	3.69	0.91	2.16	0.11	15.46	15.23	27.98
	median	35.5	1.58	0.65	0.80	2.46	0.87	2.16	0.11	16.44	10.97	23.05
	max	79.3	16.06	5.52	6.60	23.94	2.92	2.64	0.29	25.15	56.05	101.85
5	min	-84.0	0.31	0.10	0.15	0.54	0.23	1.68	0.01	1.66	8.75	8.75
	mean	32.1	2.24	0.91	1.15	3.67	0.86	2.16	0.11	15.77	15.36	27.99
	median	33.4	1.68	0.70	0.85	2.61	0.80	2.16	0.10	16.81	11.15	24.85
	max	75.1	16.20	5.53	6.62	24.18	2.65	2.65	0.29	26.07	56.05	107.40
7	min	-86.3	0.31	0.10	0.14	0.55	0.22	1.66	0.01	2.13	8.75	8.75
	mean	29.8	2.30	0.94	1.19	3.72	0.86	2.16	0.11	15.85	15.68	28.28
	median	31.8	1.76	0.72	0.90	2.65	0.81	2.16	0.10	16.76	11.20	24.38
	max	69.7	16.34	5.53	6.66	24.41	2.67	2.64	0.29	26.68	55.85	91.95
9	min	-86.2	0.30	0.10	0.14	0.54	0.21	1.64	0.01	2.36	8.75	8.75
	mean	27.9	2.33	0.96	1.21	3.71	0.89	2.17	0.11	16.13	16.52	30.57
	median	29.2	1.91	0.77	0.97	2.72	0.82	2.17	0.10	16.96	11.28	27.08
	max	66.8	16.11	5.54	6.63	24.66	3.04	2.65	0.29	27.64	56.05	107.40
11	min	-88.0	0.30	0.10	0.14	0.55	0.19	1.62	0.01	2.65	8.75	8.75
	mean	25.2	2.43	0.99	1.25	3.78	0.88	2.17	0.11	16.19	16.89	30.22
	median	25.7	2.00	0.81	1.02	2.73	0.81	2.17	0.10	16.99	11.72	27.57
	max	64.4	15.99	5.53	6.63	24.89	2.98	2.64	0.29	28.13	56.05	101.85

Table 3.7: *Minimum, mean, median and maximum tumor prediction metrics given for different k of the direct LMS class III predictor based on breathing predictions*

In general, minimum and maximum metrics are relatively far away from each other resulting in a large spread of each metric's values. As a consequence, there are some datasets yielding good metric values while there are also other datasets with worse metric values. Median and mean values lie closer together than for the LMS class I predictor applied on breathing motion. As can be inferred by comparing Table 3.7 to Table 3.5 all performance metrics of class I algorithms and breathing motion result in significantly improved metric values than for the class III version, although this is directly based on the class I algorithm applied on breathing motion. One reason is that the regression does not always fit well all the time and its goodness-of-fit is changing with time. Another reason is that the effective prediction horizon in terms of the latest tumor prediction information is usually much larger than k. Depending on the characteristics of the test datasets, the size of the gap between latest tumor and latest breathing information varies between one and 44 sampling instants with an average maximum of 17 per dataset, which needs to be added to k to get the effective prediction horizon. Therefore the average time from the latest tumor information with which the regression was updated to the actual k is usually much larger than k. Then the correlation might also have changed already, resulting in larger errors here. Nevertheless, these results will serve as a baseline to which results, e.g. from the following class II extension will be benchmarked.

3.3.1.4 Class II Extension

An AR model for predicting breathing motion can be naturally extended to perform tumor motion prediction based on available breathing predictions by employing an autoregressive model with exogenous input (ARX) model

$$A(q)y(t) = B(q)u(t-h) + e(t)$$

which extends the AR model by an external input signal $u(t) \in \mathbb{R}$. This can be shifted by h sampling instants to account, for example, for a certain dead time to pass between the input shows an effect on the output. In analogy to $A(q)$, $B(q)$ is a polynomial of degree n_b

$$B(q) = b_0 + b_1 q^{-1} + b_2 q^{-2} + \cdots + b_{n_b-1} q^{-n_b+1}$$

with model parameters $b_i \in \mathbb{R}$, $i = 0, 1, \ldots, n_b - 1$. Extending the parameter vector to

$$\theta = [a_1, a_2, \ldots, a_{n_a}, b_0, b_1, \ldots, b_{n_b-1}]^T$$

and the vector of observations to

$$\varphi(t) = [-y(t), \ldots, -y(t-n_a+1), u(t-h), u(t-h-1), \ldots, u(t-h-n_b+1)]^T \ , \quad (3.41)$$

the predictor takes the same form as in Eq. (3.34) for the pure AR model. The model parameters in θ can be determined by the LMS algorithm without any modification if the additional inputs $u(t-h), u(t-h-1), \ldots, u(t-h-n_b+1)$ are known.

For tumor motion prediction, the output signal $y(t)$ of the ARX model describes the tumor motion to be predicted. If the input signal $u(t)$ represents respiratory motion, the resulting ARX model fuses both tumor and respiratory motion in order to perform tumor motion predictions without the need of using regression models based on correlation analysis. Therefore the structure is consistent with class II algorithms. Especially when using the EPID as source of tumor motion information, there is often a large gap in the recent past between respiratory and tumor motion signals, in which the latest tumor position information is several sampling instants older than the latest respiratory position information. As already discussed before, this problem is circumvented by adaptively changing the prediction horizon k according to the size of the gap. So from the perspective of the latest tumor position information, there are usually measurements of the respiratory motion available originating from newer sampling instants. Assuming a correlation between respiratory and tumor motion, especially the respiratory information which is newer than tumor position information can help improve tumor motion prediction by including this information into the ARX model. Essentially, this means to have $h < 0$ in Eq. (3.41) assuming that sampling instant t refers to the time of the latest tumor position information. Note that normally $h < 0$ leads to an acausal model if t refers to the latest sampling instant. The size of the gap in units of sampling instants imposes a lower bound on h (with $h < 0$). However, the size of the gap varies during runtime which would necessitate an adaptation of the model structure, order and parameters, rendering this approach infeasible. But if in a prior step the breathing motion is predicted by an AR model, the missing values of breathing motion for a fixed h, for which no measurements are available yet, can be replaced by its prediction. Similar to the k-step predictor in Eq. (3.39), the predictor of the ARX model can be stated by

$$\hat{y}(t+k|t) = \hat{\theta}_k(t)\varphi_k(t) \ ,$$

where the choice of the observation vector $\varphi_k(t)$ additionally includes either measurements of breathing motion or breathing predictions, depending upon their availability. In the case of $k=1$, the one-step prediction is to be computed for tumor motion. Since either the breathing position information $u(t)$ or its one-step prediction $\hat{u}(t+1|t)$ is already available for $t+1$, it can be included into $\varphi_1(t)$:

$$\varphi_1(t) = [-y(t),\dots,-y(t-n_a+1),\hat{u}(t+1|t),u(t),u(t-1),\dots,u(t-n_b+2)]^T \ ,$$

which is equivalent to letting $h=-1$. For the two-step prediction, $\varphi_2(t)$ can be defined by

$$\varphi_2(t) = [-y(t),\dots,-y(t-n_a+1),\hat{u}(t+2|t),\hat{u}(t+1|t),u(t),\dots,u(t-n_b+3)]^T \ ,$$

resulting in $h=-2$. Therefore, let $h=-k$ for arbitrary k which includes breathing position information up to sampling instant $t+k$ for the prediction of tumor position to the same sampling instant. Thus in general, $\varphi_k(t)$ becomes

$$\varphi_k(t) = [-y(t),\dots,-y(t-n_a+1),\hat{u}(t+k|t),\dots\hat{u}(t+1|t),u(t),\dots,u(t-n_b+k+1)]^T \ ,$$

with a dimension of n_a+n_b. If $n_b<k+1$, then only predictions $\hat{u}$ are included in $\varphi_k(t)$.

3.3.1.5 Class II Parameter Selection

The class II extension of the direct LMS predictor requires breathing predictions using the class I direct predictor, where its parameters n_a and μ were determined for each prediction horizon separately. The predictor using the ARX model necessitates choice of model parameters n_a, n_b and μ. The parameter n_a was selected to equal the one from the class I predictor, thereby using as many tumor position samples in the ARX model as breathing position samples in the AR model. The remaining parameters n_b and μ are determined by exhaustive simulations over the same domain as for the class I predictor.

Similar to that, instabilities can occur in the ARX model as well, as depicted in Figure 3.23. In contrast to the LMS predictor applied on breathing motion, a small number of instabilities are found for $k=1$, especially with increasing μ and decreasing n_b. Generally the number of instabilities increases as k becomes larger. There are also several trials with very small values of μ which result in a small negative PR, shown here as instabilities. As the prediction horizon increases, a valley of a small number of instabilities emerges which is located at values of μ between 0.025 to 0.1. For $k=11$ the minimum is found for n_b in the range 16-60, thereby constraining the choice of adequate parameters. For smaller prediction horizons, the valley broadens so that n_b and μ can be chosen from a wider range.

As can be inferred from Figure 3.24 which displays the mean prediction ratio for the direct LMS predictor of the ARX model applied on all datasets, the regions with maximum PR are covered by the regions with lowest instabilities. In particular, these are found for μ between 0.025 to 0.1 and for n_b between 6 to 60, for larger k the parameter n_b can reach up to 100. All other metrics, RMS, CI95, MAX, MAE, S_F, S_{TS}, S_{SNR}, S_J, CT25 and

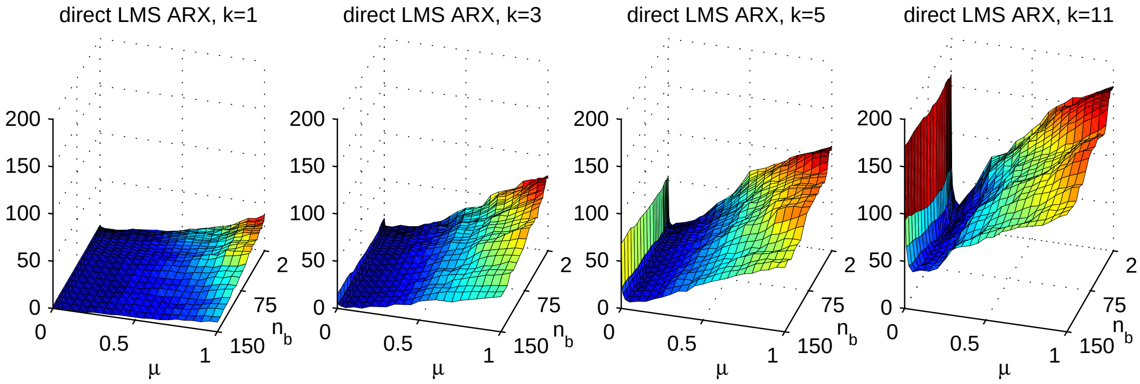


Figure 3.23: *Number of unstable trials among 200 datasets of the direct LMS ARX predictor in relation to parameters n_b and μ*

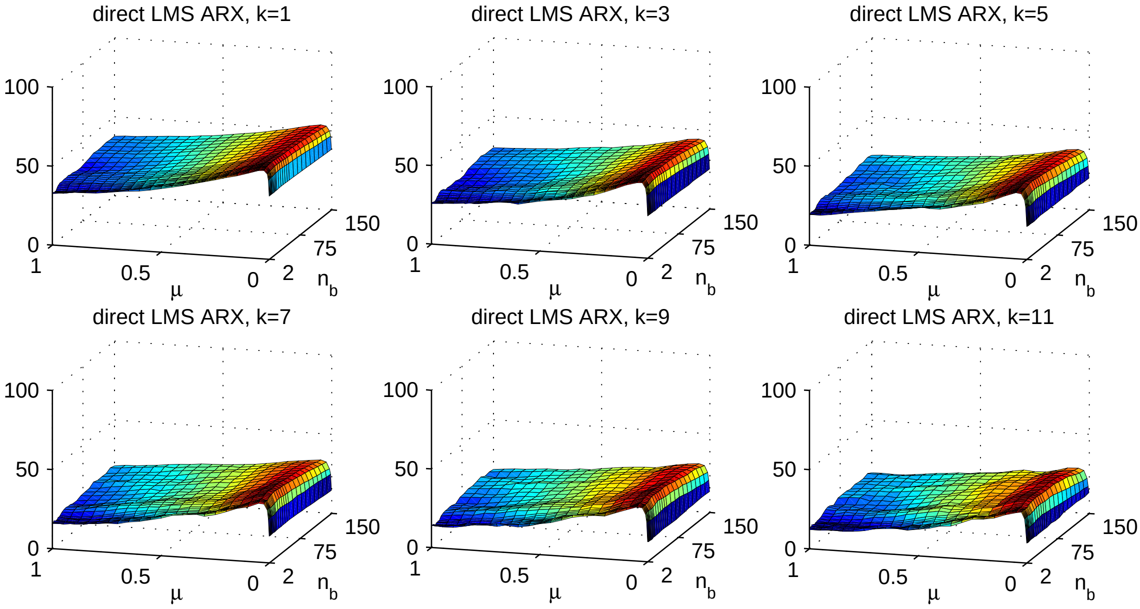


Figure 3.24: *Mean prediction ratio of 200 datasets of the direct LMS ARX predictor in relation to parameters n_b and μ*

CT10, show best metric values in exactly the same region. It is interesting to note that this choice is especially supported by the smoothness criteria, since in the LMS predictor, smoothness needed to be traded-off with the prediction accuracy criteria. Additionally, the averaged standard deviations possess smallest values in a similar region. Thus, the choice of parameters for the direct LMS ARX predictor can be narrowed to a small region in which best possible metrics on all datasets are located.

Choosing the parameter setting with best PR for each prediction horizon k individually (according to Figure 3.24 they lie closely together) and applying this on every dataset using the dominant y-axis of tumor motion, yield the results summarized in Table 3.8. Comparing these results to Table 3.7, it can be seen that there is a significant performance increase in all prediction accuracy and convergence time criteria in contrast to the class III algorithm. Opposed to that, the ARX model parameters of the class II algorithm are

Prediction horizon	Type	PR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]	S_F	S_J	S_{TS}	S_{SNR}	CT25 [s]	CT10 [s]
1	min	-5.7	0.33	0.08	0.14	0.56	0.74	2.19	0.05	2.06	8.75	8.75
	mean	57.2	1.40	0.47	0.67	2.97	2.08	2.63	0.14	12.23	9.87	14.61
	median	59.0	1.10	0.38	0.54	2.16	1.98	2.63	0.13	12.57	9.30	11.55
	max	82.1	5.00	1.65	2.34	11.78	4.36	3.02	0.27	20.22	19.65	52.90
3	min	-23.7	0.38	0.11	0.17	0.53	0.71	2.22	0.05	1.53	8.75	8.75
	mean	48.8	1.63	0.60	0.80	3.13	2.09	2.65	0.15	11.88	10.49	18.68
	median	52.0	1.29	0.48	0.64	2.24	2.00	2.66	0.14	12.36	9.58	15.10
	max	80.2	6.00	2.12	2.83	13.33	3.87	3.00	0.30	20.29	29.30	55.90
5	min	-28.4	0.39	0.12	0.18	0.65	0.72	2.22	0.05	1.75	8.75	8.75
	mean	43.0	1.81	0.69	0.90	3.28	2.09	2.66	0.14	11.99	11.57	22.38
	median	45.9	1.42	0.56	0.74	2.44	2.01	2.66	0.14	12.56	10.10	17.05
	max	77.4	6.61	2.35	3.09	15.40	4.21	3.03	0.30	19.93	35.50	91.30
7	min	-30.6	0.42	0.14	0.20	0.59	0.70	2.21	0.06	1.56	8.75	8.75
	mean	38.2	1.96	0.76	0.99	3.55	1.99	2.65	0.15	11.97	11.92	25.61
	median	41.3	1.58	0.63	0.82	2.58	1.94	2.65	0.14	12.56	10.40	21.90
	max	76.6	7.08	2.95	3.66	18.80	3.73	3.00	0.32	20.21	35.55	110.00
9	min	-24.8	0.42	0.15	0.20	0.63	0.69	2.15	0.06	1.88	8.75	8.75
	mean	33.9	2.11	0.82	1.06	3.71	1.91	2.58	0.15	12.16	12.52	25.41
	median	36.9	1.74	0.68	0.90	2.78	1.82	2.58	0.14	12.74	11.05	22.33
	max	75.2	8.28	3.33	4.10	20.85	4.03	2.99	0.31	20.38	37.95	60.65
11	min	-20.7	0.38	0.15	0.19	0.72	0.70	2.16	0.06	1.88	8.75	8.75
	mean	30.4	2.24	0.88	1.13	3.83	1.87	2.57	0.15	12.21	13.01	27.55
	median	33.0	1.79	0.72	0.95	2.86	1.76	2.57	0.14	12.69	11.25	25.42
	max	72.2	9.38	3.66	4.58	22.87	4.11	2.99	0.32	20.09	39.70	110.00

Table 3.8: *Minimum, mean, median and maximum tumor prediction metrics of parameter setting with largest average PR determined for each k separately; metrics are given for different k using the direct LMS ARX predictor*

updated based on more information about the breathing motion, which is not the case for the class III algorithm. There, the determination of tumor motion is only based on regression which is only updated to the latest available tumor motion information. Introducing newer breathing information (either measured or determined by the LMS AR breathing predictor) enables the ARX predictor to achieve more precise tumor position predictions. Naturally, overall performance decreases with increasing prediction horizon k and the difference between class II and class III results become smaller as well. When comparing the absolute values of the prediction accuracy metrics to the ones from the class I predictor of breathing motion in Table 3.6, a noticeable drop in performance can be observed for tumor predictions. This behavior emerges from a larger effective prediction horizon for tumor motion because the varying gap between latest available tumor position information and latest available breathing position information needs to be bridged (cf. Section 3.3.1.3). Although class III predictions are smoother, the improved prediction accuracy and convergence time performance of the class II predictor outweigh the smoothness, so that the class II predictor in form of the LMS ARX predictor should be preferred over the class III predictor.

3.3.2 Recursive Least-Squares Algorithm

The presentation of the recursive least-squares algorithm is organized in the same way as the LMS algorithm: class I formulation and parameter selection, class III evaluation and class II extension including parameter selection. Throughout the following evaluations, the performance metrics are gradually compared to the ones of the LMS algorithm.

3.3.2.1 Class I Formulation

Given an autoregressive model, its parameters can also be updated by the recursive RLS algorithm [125]. In contrast to the LMS algorithm, it solves a linear least-squares problem to estimate the model parameters. The RLS algorithm requires an initial estimate of the model parameters $\hat{\theta}_k(0) \in \mathbb{R}^{n_a}$ and an initial estimate of a matrix $P(0) \in \mathbb{R}^{n_a \times n_a}$ at time $t = 0$. For the direct predictor, the RLS algorithm can be summarized according to Eq. (3.36) for an arbitrary sampling instant t as follows:

Given the vector of n_a past observations

$$\varphi(t-k) = [-y(t-k), -y(t-k-1), \ldots, -y(t-k-n_a+1)]^T \ ,$$

where k is the prediction horizon, at first a gain vector $K \in \mathbb{R}^{n_a}$ is computed from the matrix P at the previous sampling instant $t-1$ by

$$K = \frac{\lambda^{-1} P(t-1)\varphi(t-k)}{1 + \lambda^{-1}\varphi^T(t-k)P(t-1)\varphi(t-k)} \ . \tag{3.42}$$

Using K, an update of P for the current sampling instant t is derived from the previous sampling instant with

$$P(t) = \lambda^{-1}P(t-1) - \lambda^{-1}K\varphi^T(t-k)P(t-1) \ . \tag{3.43}$$

Since P is updated recursively, the RLS predictor contains a memory carrying information from the time of initialization of the algorithm through the whole prediction process. A parameter $\lambda \in \mathbb{R}$ is usually chosen from the interval $]0;1]$ to be close to the nominal value of one. Due to the fact that P is divided by λ in every sampling instant, it has the effect of an exponential forgetting factor, limiting the influence from past updates and increasing emphasis on newer information. With the gain vector K, the model parameters are recursively updated by

$$\hat{\theta}_k(t) = \hat{\theta}_k(t-1) + K\left(y(t) - \hat{\theta}_k^T(t-1)\varphi(t-k)\right) \ . \tag{3.44}$$

Equivalent to the LMS algorithm, the parameter update is proportional to the (a priori) prediction error $e_k(t)$ from Eq. (3.38) and to the observation vector $\varphi(t-k)$, which is included in K. The major contrast to the LMS algorithm is the adoption of P in the numerator and denominator of K. Since K partially takes over a similar function as μ in the LMS algorithm, the RLS algorithm continuously adapts the step-size parameter according to P.

From the derivation of the RLS algorithm, P can be identified as the inverse of the autocorrelation matrix of the observation vector. This inverse is used in the minimization of a cost function (which is formulated over the squared prediction error) to solve for the optimal parameter vector $\hat{\theta}_k$. Therefore $P(0)$ needs to be chosen in such a way to render the initial autocorrelation matrix nonsingular. If there are no data samples available at the start of the RLS algorithm to compute the inverse autocorrelation matrix, it can be initialized with $P(0) = \delta^{-1}I_{n_a}$, where δ is a positive constant. This variant of initialization is known as soft constrained initialization [132]. According to [133], δ should be chosen depending on the SNR of the observations. For medium and high SNR, δ should be small and for low SNR larger. For initialization of the RLS algorithm, the initial parameter vector needs to be chosen as well. It is customary to start with parameter values of zero. Indeed, in preliminary analysis using the available datasets, no other initial parameter values could be found performing better than the zero vector as opposed to the LMS algorithm.

The RLS algorithm was formulated for the direct predictor. Equivalent to the LMS algorithm, the plug-in predictor is derived in the same way by letting $k = 1$ and iteratively substituting unavailable observations by predictions in order to achieve multi-step predictions. The considerations for dead time hold for the RLS in the same way.

3.3.2.2 Class I Parameter Selection

In order to find suitable values for the parameters n_a, λ and δ, simulations were performed on all pre-recorded datasets similar to the parameter selection procedure of the LMS algorithm. The parameter n_a was varied between 2 and 150 with different step sizes of 2, 4, 10 and 20, depending on the range of n_a itself. Values for the parameter λ were: 0.8, 0.9, 0.95, 0.975, 0.99, 1.0, 1.01, 1.025, 1.05 and 1.1 with smaller steps in the vicinity of 1.0. Simultaneously the parameter δ was varied between 10^{-3} and 10 in powers of 10. The analysis on the results was performed in exactly the same way as for the class I LMS algorithm.

At first, instabilities were analyzed which are depicted in Figure 3.25 for the direct predictor whereas Figure 3.26 shows the instabilities for the plug-in predictor. Since the RLS algorithm incorporates three parameters, a reduction of dimensions is necessary to enable visual analysis with 3D plots. For the shown graphs, the dimension for parameter δ was reduced in such a way that for fixed values of n_a and λ, the maximum number of instabilities was used which was determined from all available values of the parameter δ. For both types of RLS predictors, it can be seen that the choice of the exponential forgetting factor λ mainly determines the number of instabilities, which increase the most with decreasing λ but also if λ is chosen larger than 1.0. Generally, more instabilities arise if the prediction horizon k is increased, constraining suitable choices for λ to a small valley around 1.0. There is also a significant dependency on the parameter n_a, which

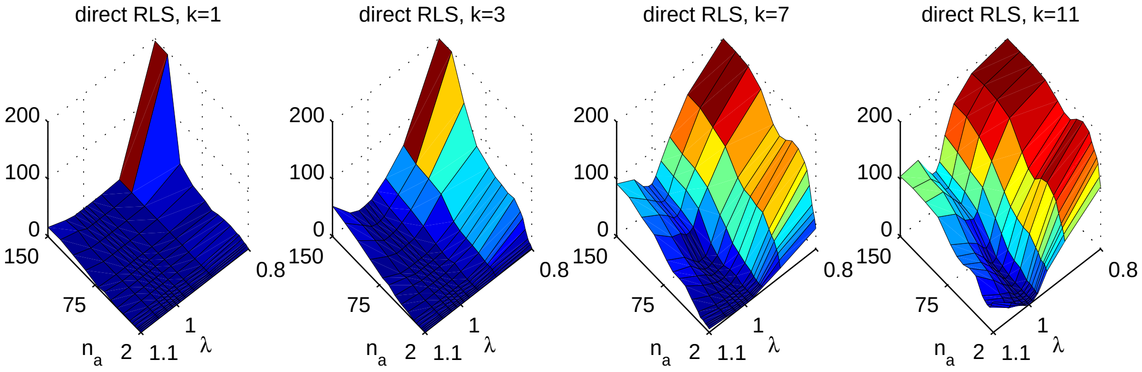


Figure 3.25: *Number of unstable trials among 200 datasets of the direct RLS predictor in relation to parameters n_a and λ*

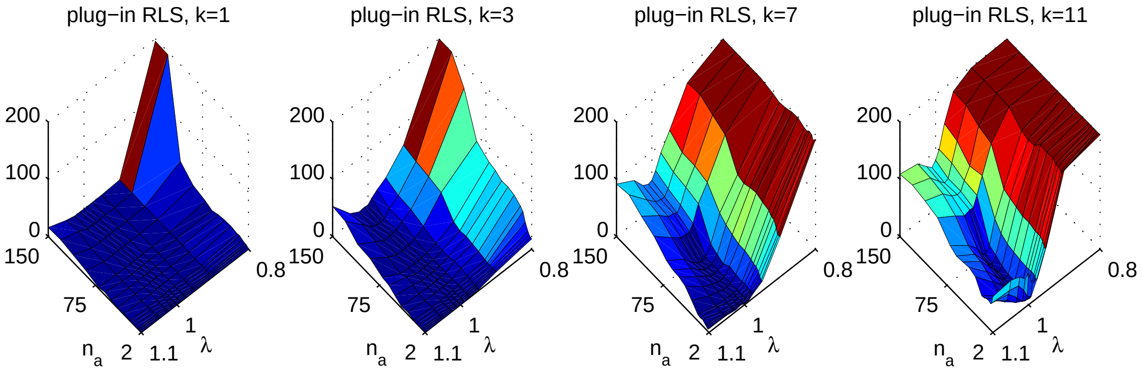


Figure 3.26: *Number of unstable trials among 200 datasets of the plug-in RLS predictor in relation to parameters n_a and λ*

becomes apparent for large k. Therefore, n_a should be chosen less than 60 for the direct predictor. In fact, no instabilities arise for all $k \leq 11$ if $n_a \leq 30$ and $\lambda = 1$. For the plug-in predictor the choices of parameters with no instabilities are slightly more constrained with an additional lower bound of $n_a \geq 8$. This improves upon the LMS algorithm for which no parameter setting could be found for $k = 11$ with no instabilities.

To further study the influence of parameters, the average prediction ratio of the plug-in RLS predictor is shown in Figure 3.27, in which the reduction of the dimension for δ was performed by averaging over that dimension. The same plot for the direct predictor is

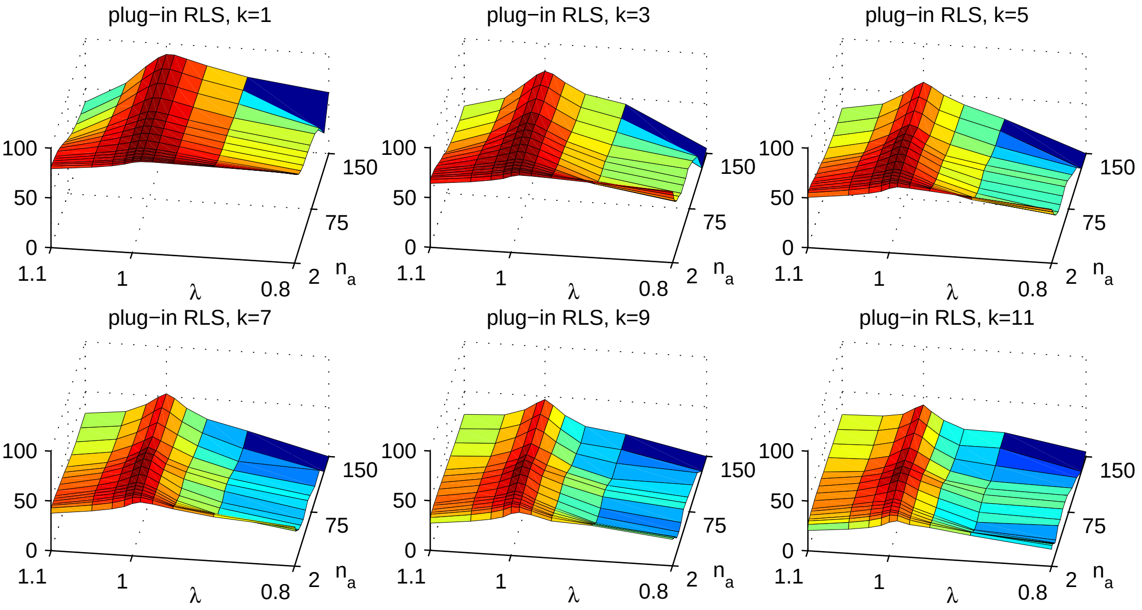


Figure 3.27: *Mean prediction ratio of 200 datasets of the plug-in RLS predictor in relation to parameters n_a and λ*

quite similar except that slightly less PRs are achieved; therefore it is omitted here. For $k = 1$, maximum PRs can be found for $\lambda = 0.99$ where the influence of n_a plays only a minor role. Nevertheless, the peak value is reached for $n_a = 8$. For $k > 1$, maximum PRs result only for $\lambda = 1$, where the best choices for n_a move to larger values for increasing k

up to $n_a = 60$. While for smaller k parameter settings can be found with maximum PRs and no instabilities, a trade-off is required for larger k. However, choosing n_a between 8 and 30 to avoid instabilities still results in PRs close to the maximum ones.

In order to visualize the influence of the parameter δ, the dimension for n_a was eliminated from the data by reducing it with its average as n_a showed only a minor impact on the PRs. Figure 3.28 shows the resulting PRs of the plug-in predictor in relation to the remaining parameters λ and δ. It can be inferred that δ has only a weak impact, given the logarithmic scaling. This is consistent with theory, as δ is only used for initializing the inverse autocorrelation matrix $P(0)$ and the fast convergence of the RLS algorithm constrains an inappropriate choice of δ to only a short time until P is similar compared to a case if δ is chosen more appropriately. However, from these plots slightly improved PR result when choosing δ in between 0.01 and 0.1 irrespective of the prediction horizon k.

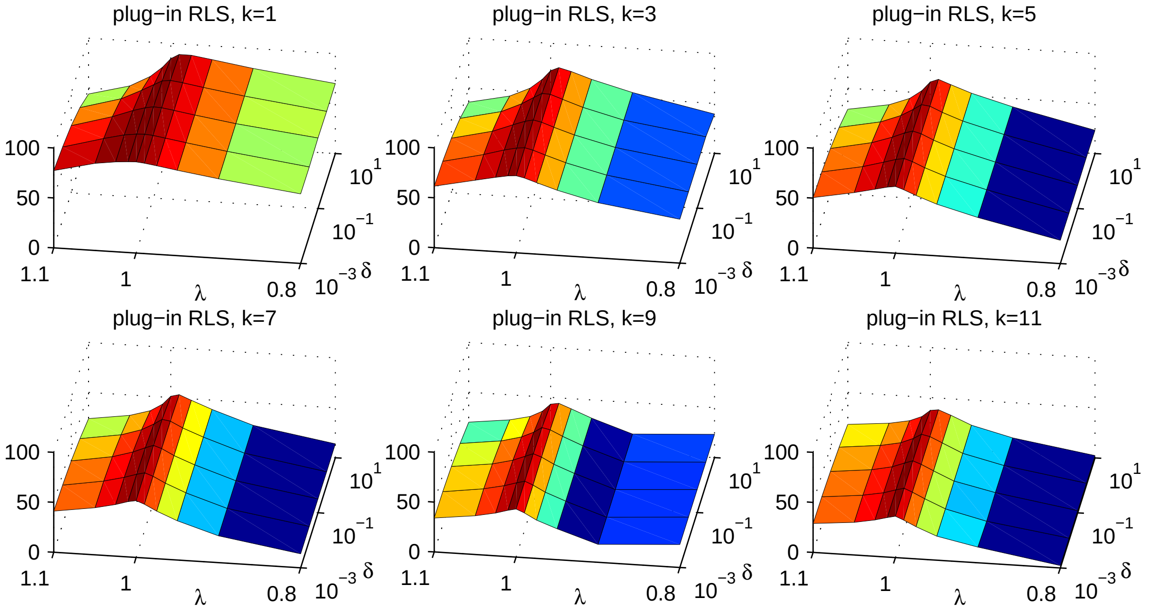


Figure 3.28: *Mean prediction ratio of 200 datasets of the plug-in RLS predictor in relation to parameters λ and δ*

In order to get a full picture of all parameters of the RLS algorithm, similar plots with a reduction along the dimension for λ would be required. When analyzing these, they do not bring any new information and so are omitted here. Furthermore, the remaining prediction accuracy metrics like RMS, CI95, MAX and MAE support the results discussed so far on the PR, so plots of these are omitted here, too.

However, interesting observations can be made from the smoothness metrics, represented by the metric S_F shown in Figure 3.29, where the data was reduced by the dimension along δ. Note that the unstable trails are not included in these plots. Therefore the smoothness values for $\lambda \leq 0.9$ have to be excluded completely from analysis because for these values, almost all datasets showed instabilities, especially for $k > 3$. Similarly, the region for $\lambda \geq 1.025$ showed a considerable amount of instabilities, where these increased with increasing n_a. For $k \leq 3$ and $\lambda \approx 1.0$ a small improvement of smoothness can be observed for smaller values of n_a. For larger k and $\lambda \approx 1.0$, larger values of n_a are beneficial for the average smoothness. However, the smoothness values for $\lambda \approx 1.0$

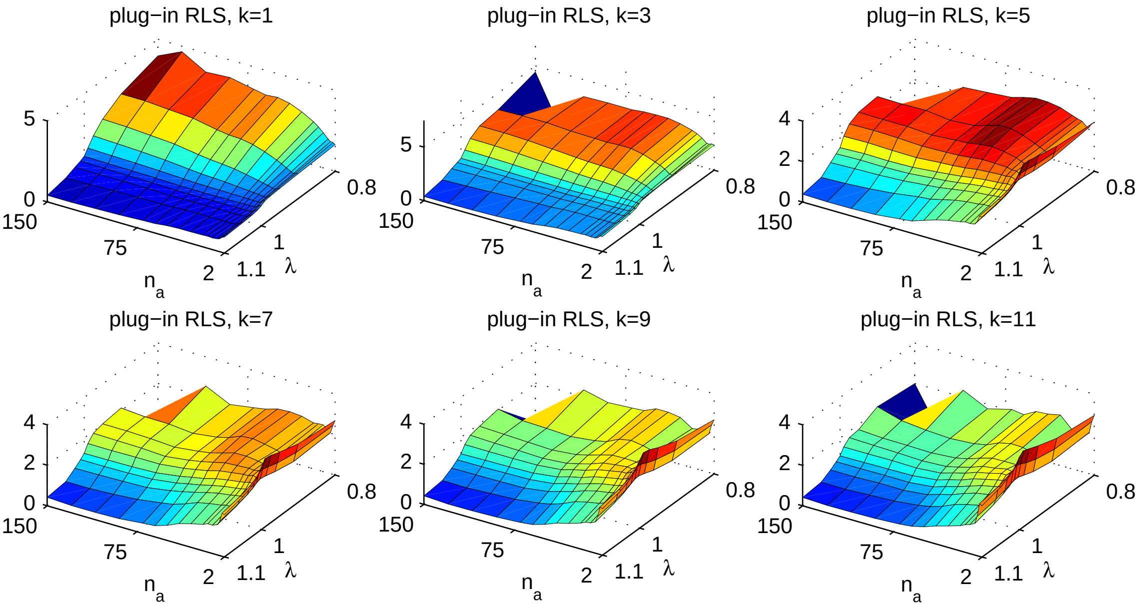


Figure 3.29: *Mean smoothness metric S_F of 200 datasets of the plug-in RLS predictor in relation to parameters n_a and λ*

are located closer to the minimum for smaller k than for larger values. Nevertheless, the regions with maximum PR still lead to the best possible smoothness values given the constraints due to instabilities. When comparing smoothness metrics between direct and plug-in predictors, it turns out that the plug-in predictor generally delivers better smoothness values. Together with larger PRs, the RLS plug-in predictor is hence favored over the direct predictor.

So far, the analysis was based on metric values which were averaged over all datasets. When selecting the 20 best parameter settings with maximum PR for each dataset individually and accumulating them over all datasets, the distribution of parameters with respect to λ and n_a shown in Figure 3.30 results for the plug-in predictor (while reducing

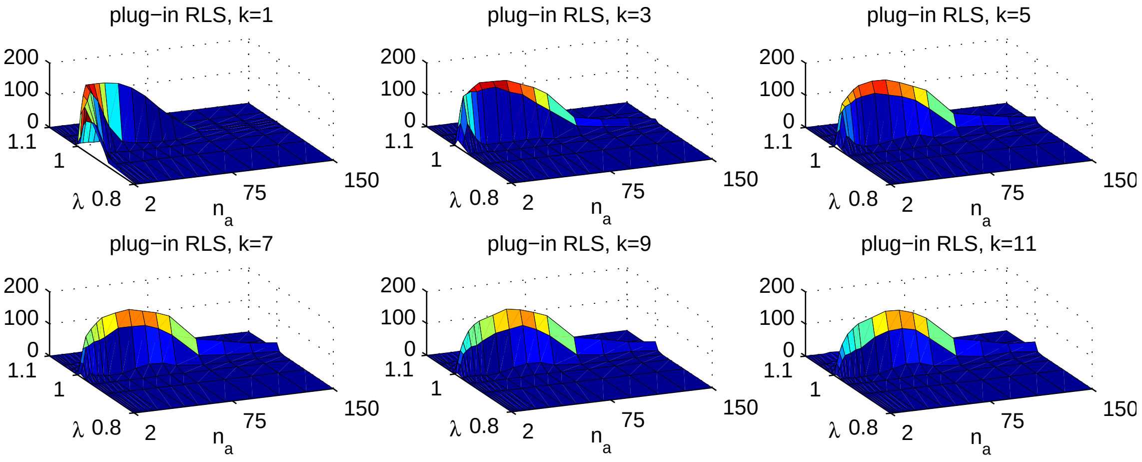


Figure 3.30: *Distribution of 20 parameter settings of the plug-in RLS predictor with largest PR for each dataset and accumulated over all datasets*

the dimension for δ by averaging). The region of parameters with maximum occurrence is the same as the one which resulted when averaging over all datasets (cf. Figure 3.27). Also here, the shift to larger values of n_a for increasing k can be observed as well as the sharp increase of occurrences for $\lambda = 1$. Comparing the ranges for n_a with maximum occurrence to the ones for the LMS predictor (c.f. Figure 3.21), it becomes obvious that the LMS predictor generally requires higher orders of n_a in the AR model in order to reach maximum PRs. Hence, the RLS algorithm can start its predictions earlier than the LMS algorithm (when choosing parameter settings with maximum PR) as n_a determines the lower bound of samples needed to fill the observation vector before one calculation cycle of the predictors based on AR models can be completed.

When choosing the parameter setting with maximum PR for each k separately from the averaged metrics over all datasets and displaying all the prediction metrics as a scaled box plot (under the same conditions as for the LMS predictor in Figure 3.22), the plot in Figure 3.31 results, showing the distribution of each metric for all datasets and for k from 1 to 11 in steps of 2. The corresponding minimum, maximum and median values for each k and for each metric are given in Table 3.9. Similar to the box plot of the LMS algorithm, the one for the RLS algorithm shows degradation of prediction accuracy as prediction horizon increases, associated with an increasing spread of the metric values. In contrast to the LMS predictor, the smoothness decreases here with increasing k for all smoothness metrics. $k = 9$ and $k = 11$ show a small negative PR as minimum metric value. A detailed inspection revealed that these occurred only for one of the 200 datasets. Compared to the LMS, this is a significant improvement as eight datasets possessed a negative PR there with an even lower value. In general, when comparing the prediction accuracy metrics to the ones of the LMS given in Table 3.5, a small improvement can be observed for $k = 1$ and major improvement is found for the RLS algorithm for $k > 1$. The absolute values of mean PR increases by 2.7%-7.8%, the mean CI95 reduces by 0.11mm-0.21mm and the mean RMS reduces by 0.05mm-0.12mm. Although the minimum, mean and median of the maximum error metric are smaller for RLS, the maximum of it is larger. Since no averaging is performed on the maximum of the MAX metric, higher values can be caused by just

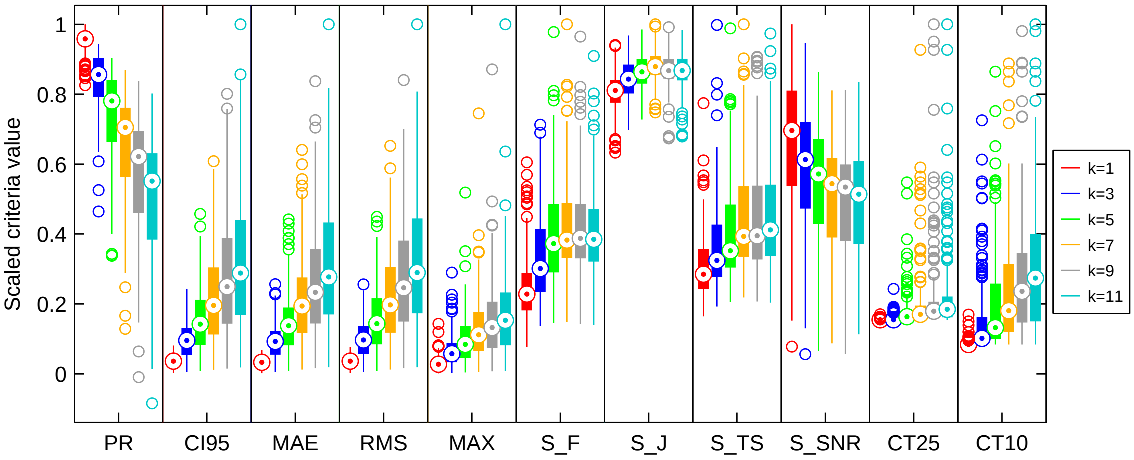


Figure 3.31: *Scaled breathing prediction metrics of parameter setting with largest average PR determined for each k separately using the plug-in RLS predictor; metric values are visualized by box plots for different k*

Prediction horizon	Type	PR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]	S_F	S_J	S_{TS}	S_{SNR}	CT25 [s]	CT10 [s]
1	min	80.8	0.01	0.00	0.00	0.02	0.37	1.95	0.07	1.85	8.75	8.75
	mean	93.1	0.17	0.06	0.08	0.49	1.21	2.49	0.13	15.92	8.76	9.13
	median	93.9	0.16	0.06	0.08	0.44	1.12	2.50	0.12	16.53	8.75	8.75
	max	97.9	0.37	0.12	0.17	2.30	2.97	2.90	0.32	23.74	9.65	17.65
3	min	45.5	0.02	0.01	0.01	0.04	0.67	2.15	0.08	1.33	8.75	8.75
	mean	81.9	0.45	0.17	0.23	1.07	1.67	2.60	0.15	14.01	9.02	15.47
	median	83.9	0.43	0.16	0.22	0.93	1.48	2.60	0.13	14.55	8.75	10.60
	max	92.4	1.10	0.45	0.57	4.65	3.50	2.99	0.41	22.45	13.75	75.55
5	min	33.1	0.04	0.02	0.02	0.07	0.71	2.24	0.08	1.54	8.75	8.75
	mean	72.5	0.72	0.27	0.36	1.59	1.99	2.67	0.17	12.76	10.09	19.89
	median	76.5	0.64	0.24	0.32	1.35	1.83	2.66	0.15	13.57	9.25	13.75
	max	88.5	2.07	0.77	1.00	8.32	4.81	3.04	0.41	20.49	31.00	90.05
7	min	12.6	0.05	0.02	0.03	0.10	0.73	2.31	0.09	2.07	8.75	8.75
	mean	63.0	1.00	0.37	0.50	2.06	2.09	2.72	0.19	11.93	11.23	25.19
	median	69.0	0.89	0.34	0.44	1.80	1.88	2.71	0.16	12.92	9.65	18.83
	max	85.2	2.76	1.12	1.46	11.95	4.92	3.08	0.41	19.25	52.45	92.55
9	min	-0.9	0.07	0.03	0.04	0.12	0.70	2.08	0.09	1.34	8.75	8.75
	mean	54.9	1.24	0.47	0.62	2.43	2.04	2.69	0.18	11.64	12.17	28.61
	median	60.9	1.13	0.41	0.55	2.12	1.91	2.67	0.16	12.68	10.15	24.60
	max	82.0	3.63	1.47	1.88	13.97	4.74	3.06	0.37	19.27	56.60	102.15
11	min	-8.3	0.08	0.03	0.04	0.13	0.69	2.09	0.08	2.69	8.75	8.75
	mean	48.2	1.45	0.54	0.72	2.72	1.99	2.67	0.19	11.63	12.65	31.46
	median	54.0	1.31	0.49	0.65	2.46	1.89	2.68	0.17	12.20	10.45	28.55
	max	78.5	4.53	1.75	2.23	16.05	4.47	3.03	0.40	19.81	56.60	104.20

Table 3.9: *Minimum, mean, median and maximum breathing prediction metrics of parameter setting with largest average PR determined for each k separately; metrics are given for different k using the plug-in RLS predictor*

one dataset with a short-term but large error. Judging from the smoothness metrics, RLS results in generally lower smoothness metrics, except for $k = 1$. The majority of the metric values for convergence time, especially the ones for CT25, are slightly smaller for the RLS predictor. In stationary environments, RLS is known to converge about an order of magnitude faster than the LMS algorithm. However, given the differentiated picture of the CT25 and CT10 metrics, the difference in convergence time when applied to breathing motions is not that large. A reason is the rather non-stationary nature of breathing motion slowing down the convergence of the RLS algorithm.

Choosing the parameter setting with largest PR for each dataset and for each k individually results in the metrics given in Table 3.10. No negative PR occur anymore and the increase in mean PR lies in between 0.2%-3.2%. Despite this small improvement of the PR, the maximum of the MAX error metric is cut down by half for larger k. That is, there is at least one dataset for which the parameter setting with largest PR, determined by averaging over all datasets for a single k, leads to degraded performance compared to the parameter setting with largest PR chosen for that dataset specifically. However, the CI95 metrics lie closer together as well as the minimum, mean and median of MAX, so that the difference in overall performance is constrained as only the maximum of MAX would indicate. Additionally, the impact on smoothness and convergence time is minor. As a consequence, it is still feasible to select a single parameter setting for each k from the averaged PR metric, however accepting some outliers on the maximum errors for a few datasets.

Prediction horizon	Type	PR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]	S_F	S_J	S_{TS}	S_{SNR}	CT25 [s]	CT10 [s]
1	min	81.8	0.01	0.00	0.00	0.02	0.37	1.96	0.07	1.61	8.75	8.75
	mean	93.3	0.16	0.06	0.08	0.49	1.18	2.45	0.13	16.03	8.76	9.06
	median	94.0	0.16	0.06	0.08	0.43	1.08	2.46	0.12	16.66	8.75	8.75
	max	98.0	0.38	0.12	0.17	2.25	2.75	2.86	0.29	23.88	9.70	18.45
3	min	52.3	0.02	0.01	0.01	0.04	0.60	2.12	0.08	-0.20	8.75	8.75
	mean	82.6	0.44	0.16	0.22	1.05	1.56	2.55	0.14	14.38	9.04	15.56
	median	84.7	0.41	0.15	0.20	0.96	1.42	2.54	0.13	15.08	8.75	10.47
	max	92.5	1.07	0.40	0.52	3.93	3.50	2.95	0.31	22.71	13.25	57.70
5	min	33.1	0.04	0.01	0.02	0.06	0.67	2.14	0.08	-1.41	8.75	8.75
	mean	73.8	0.68	0.26	0.34	1.48	1.91	2.61	0.16	13.07	10.35	21.40
	median	77.3	0.63	0.24	0.31	1.34	1.71	2.59	0.14	13.76	9.22	13.65
	max	89.3	1.70	0.67	0.84	5.72	4.11	3.02	0.36	21.89	31.00	99.35
7	min	24.9	0.05	0.02	0.03	0.09	0.82	1.75	0.08	-1.62	8.75	8.75
	mean	64.9	0.93	0.36	0.47	1.89	2.00	2.64	0.17	12.40	12.35	25.22
	median	70.0	0.88	0.32	0.43	1.79	1.79	2.64	0.15	13.56	9.70	18.17
	max	86.7	2.26	0.95	1.20	5.27	4.63	3.12	0.45	20.97	52.75	105.60
9	min	17.7	0.07	0.03	0.03	0.13	0.30	1.57	0.09	-2.62	8.75	8.75
	mean	57.4	1.15	0.44	0.58	2.19	1.97	2.61	0.18	12.13	13.93	28.91
	median	62.3	1.07	0.40	0.54	2.07	1.80	2.65	0.15	13.34	10.03	24.10
	max	84.2	2.82	1.19	1.46	6.42	5.05	3.13	0.44	20.03	71.60	105.60
11	min	12.7	0.08	0.03	0.04	0.15	0.11	1.03	0.08	-0.59	8.75	8.75
	mean	51.4	1.33	0.51	0.67	2.44	1.90	2.56	0.18	12.14	14.61	31.03
	median	56.1	1.27	0.46	0.63	2.30	1.81	2.64	0.15	12.90	10.53	27.77
	max	81.9	3.16	1.38	1.70	7.82	5.12	3.13	0.45	21.11	71.55	105.60

Table 3.10: *Minimum, mean, median and maximum breathing prediction metrics of parameter setting with largest PR determined for each dataset and each k individually; metrics are given for different k using the plug-in RLS predictor*

When analyzing the distribution of the best parameter setting for each dataset individually (using a similar plot as in Figure 3.30), most occurrences are located at $\lambda = 1$ but with a varying n_a, so that for some datasets small values of n_a lead to best performance whereas for other datasets larger values of n_a up to about 60 yield better results. The fact that $\lambda = 1$ generally leads to best achievable prediction performance for most of the datasets, means that the exponential forgetting feature is not required. However, it is interesting to note that for a few datasets, best performance is observed for $\lambda > 1$, which is not complying with theory as λ should be one at maximum. This finding was also reported in [134]. A closer inspection revealed that for these datasets, the well-known problem of estimator windup [135] is present in a mild form, which can occur if the input to the RLS algorithm, in this case the breathing motion, is not properly excited. This happens for some datasets in which the breathing is only changing very slowly and is almost constant for several seconds. Since P is divided in each update recursion in Eq. (3.43) by the exponential forgetting factor λ, it can grow exponentially if $\lambda < 1$ and the input is not exciting enough. However, if $\lambda > 1$ is chosen in such a case, it can effectively avoid the problem of estimator windup since then P cannot become unstable anymore. There are also other methods like conditional updating or constant-trace algorithms [135] in order to prevent estimator windup. The latter of them was also evaluated on all datasets by scaling the matrix P in each update recursion by its trace. However, an overall performance degradation can be observed compared to previously discussed results on the RLS algorithm with exponential forgetting, so this scheme is not further pursued.

3.3.2.3 Class III Evaluation

Based on the class I RLS predictor, performing breathing predictions, the class III form is analyzed in the following, which makes use of linear regression to establish a relationship between breathing and tumor motion. The evaluation is performed under exactly the same conditions as for the LMS predictor from Section 3.3.1.3. The corresponding results are given in Table 3.11 using the plug-in RLS predictor for which a single parameter setting was determined for each k separately by choosing the one with largest PR on average over all datasets. The performance metrics of the underlying breathing predictions can be drawn from Table 3.9.

Prediction horizon	Type	PR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]	S_F	S_J	S_{TS}	S_{SNR}	CT25 [s]	CT10 [s]
1	min	-80.5	0.33	0.10	0.15	0.55	0.27	1.73	0.02	1.33	8.75	8.75
	mean	34.8	2.18	0.88	1.12	3.68	1.01	2.34	0.12	15.90	15.30	26.88
	median	36.9	1.48	0.61	0.76	2.40	0.93	2.35	0.11	16.77	10.97	22.90
	max	81.9	15.83	5.45	6.53	23.70	2.90	2.76	0.32	25.07	55.70	92.05
3	min	-81.9	0.32	0.10	0.15	0.55	0.31	1.74	0.02	0.57	8.75	8.75
	mean	33.5	2.21	0.90	1.14	3.76	1.36	2.43	0.14	14.51	15.53	30.52
	median	35.4	1.55	0.63	0.78	2.44	1.19	2.42	0.12	15.31	11.13	25.72
	max	80.6	15.72	5.51	6.59	23.93	3.66	2.89	0.40	24.13	55.85	107.40
5	min	-83.4	0.32	0.10	0.15	0.54	0.35	1.78	0.02	1.03	8.75	8.75
	mean	31.9	2.24	0.91	1.16	3.87	1.63	2.50	0.15	13.32	16.08	30.95
	median	33.6	1.62	0.66	0.83	2.56	1.47	2.50	0.13	13.96	11.70	27.20
	max	78.3	15.82	5.54	6.64	24.17	3.75	2.93	0.38	23.30	55.85	107.40
7	min	-84.8	0.32	0.10	0.15	0.56	0.36	1.79	0.02	2.30	8.75	8.75
	mean	30.2	2.29	0.93	1.18	3.95	1.71	2.55	0.16	12.64	16.29	32.09
	median	32.2	1.71	0.70	0.88	2.81	1.54	2.56	0.15	13.40	11.85	28.05
	max	74.9	15.86	5.52	6.67	24.42	4.24	2.98	0.37	22.83	55.85	103.45
9	min	-86.3	0.31	0.10	0.14	0.55	0.36	1.79	0.02	3.44	8.75	8.75
	mean	28.3	2.34	0.96	1.21	3.96	1.71	2.54	0.16	12.34	16.58	33.16
	median	29.9	1.78	0.74	0.91	2.87	1.60	2.55	0.15	13.01	11.97	30.00
	max	70.7	15.90	5.48	6.70	24.79	4.16	3.00	0.34	22.48	55.85	107.40
11	min	-88.0	0.30	0.10	0.14	0.56	0.35	1.78	0.02	3.78	8.75	8.75
	mean	26.4	2.40	0.98	1.24	4.00	1.67	2.53	0.16	12.33	16.17	32.84
	median	27.5	1.90	0.75	0.95	2.95	1.62	2.55	0.15	12.61	12.03	29.33
	max	66.6	15.80	5.46	6.71	24.90	3.96	3.01	0.35	22.31	55.85	107.40

Table 3.11: *Minimum, mean, median and maximum tumor prediction metrics given for different k of the plug-in RLS class III predictor based on breathing predictions*

Comparing tumor motion prediction results from the class III RLS predictor to the class III LMS predictor, it turns out that the metric values, especially the prediction accuracy metrics, are similar with a slight improvement for RLS, i.e. the impact of the improvement observed between the class I breathing predictors on the class III predictors is minor. Additionally, the major effects on smoothness are inherited from the class I predictor in that RLS predictions are generally less smoother than LMS predictions. Furthermore, a minor increase in convergence time can be observed for the class III RLS predictor.

3.3.2.4 Class II Parameter Selection

In order to enable direct tumor motion predictions, the class I RLS predictor using an AR model can be extended to incorporate an ARX model in the same way as for the LMS predictor as discussed in Section 3.3.1.4. In this scheme, at first a class I RLS predictor performs breathing predictions. These are input to the ARX model which is regressed over tumor position samples. Note that the effective prediction horizon is usually larger than k in the class I predictor. However, in the following discussion about parameter settings and results of the class II RLS predictor, k denotes the effective prediction horizon.

The ARX model incorporates the parameters n_a, n_b, λ and δ for which suitable values need to be found. Due to the similar characteristics of breathing and tumor motion, the parameter n_a is chosen to equal n_a from the class I predictor, which is determined by the parameter setting with largest PR on average for each k separately. For the parameters n_b (denoting the number of breathing position samples or predictions fed as external input into the ARX model), the exponential forgetting factor λ and δ, simulations were performed. The ranges for n_b and λ were selected to be the same as the ranges of n_a and λ for the class I predictor, respectively. The range for δ was changed to 10^{-1} to 10^4 since preliminary analysis revealed a different dependency than for the class I predictor.

At first, instabilities were analyzed in terms of the parameters n_b and λ (while reducing the dimension for δ by taking the maximum along that dimension). Judging from Figure 3.32, the lowest number of instabilities arise for $\lambda = 1$ and for $n_b \leq 50$. If the prediction horizon increases, the upper bound on n_b decreases, so that the choice for n_b becomes more constrained. The effect of δ (not shown here) on instabilities is minor since δ is only used during initialization of the RLS algorithm.

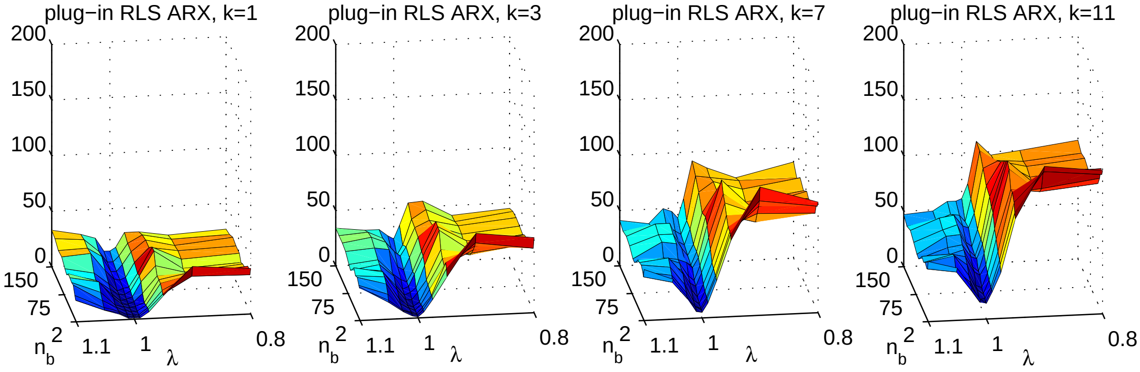


Figure 3.32: *Number of unstable trials among 200 datasets of the plug-in RLS ARX predictor in relation to parameters n_b and λ*

Next, the prediction ratio is analyzed in view of the parameters n_b and λ. Here, the reduction of the dimension δ is performed by averaging over that dimension. The resulting plots in Figure 3.33 clearly indicate that $\lambda = 1$ yields maximum PRs. Irrespective of the prediction horizon, the optimum value for n_b is found at 30, about half the duration of an average breathing cycle. These results fit well to the instabilities which are still low in this region of λ and n_b. In order to acquire the dependence on the parameter δ, reduction of the results was performed along the dimension for n_b due to the smaller impact of n_b compared to λ. The influence of δ and λ can be observed in Figure 3.34. Maximum PRs are found for $\delta = 100$ irrespective of the prediction horizon k. As a consequence, δ should

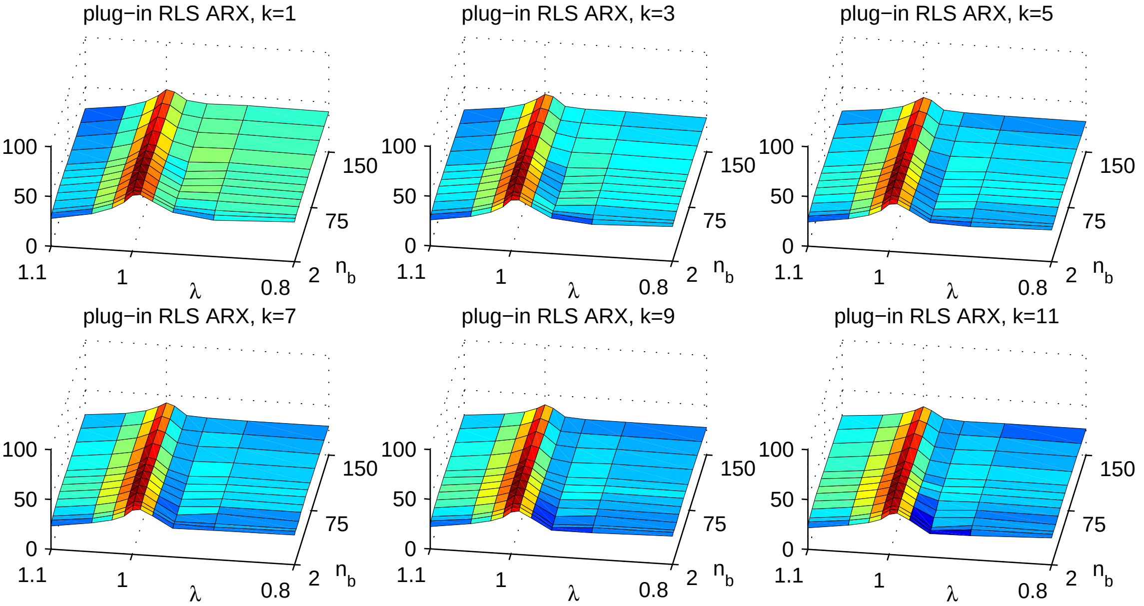


Figure 3.33: *Mean prediction ratio of 200 datasets of the plug-in RLS ARX predictor in relation to parameters n_b and λ*

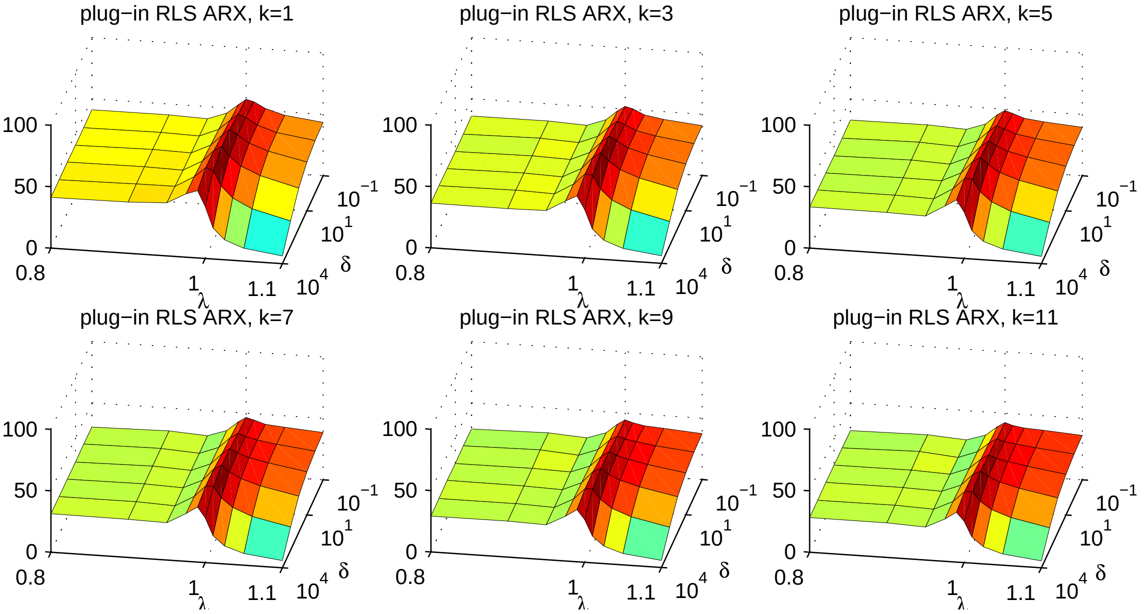


Figure 3.34: *Mean prediction ratio of 200 datasets of the plug-in RLS ARX predictor in relation to parameters λ and δ*

be selected about three to four magnitudes larger than for the class I predictor. The other prediction accuracy metrics like CI95, MAE, RMS and MAX follow the same parameter dependency as the PR in such a way that they result in smaller metric values in the same regions when PR is maximized.

Having obtained separate parameter settings for the ARX-based plug-in RLS predictor for each k individually by finding the parameter setting with largest PR averaged over all datasets, the metrics presented in Table 3.12 result. Comparing these to results of the LMS ARX predictor given in Table 3.8, generally an improvement on all metrics can be observed. All mean and median prediction accuracy metrics thereby improve more with increasing prediction horizon. The mean PR shows absolute increases in the range 7.5%-15.5%, mean CI95 reduces by 0.38mm-0.51mm, mean RMS drops by 0.12mm-0.26mm and mean MAX reduces by 0.29mm-0.53mm. Except for two metrics for $k = 1$ (maximum S_F and minimum S_{SNR}), all smoothness metrics show a considerable improvement. Additionally in improvement is visible for the convergence time criteria with a few exceptions especially for $k = 1$ and $k = 3$. Despite the general performance increase in PR, one dataset results in worse PR values for $k \geq 9$ than the LMS ARX predictor. However, the RLS ARX predictor clearly shows better general performance.

Prediction horizon	Type	PR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]	S_F	S_J	S_{TS}	S_{SNR}	CT25 [s]	CT10 [s]
1	min	27.1	0.22	0.07	0.11	0.41	0.56	2.05	0.05	-0.31	8.75	8.75
	mean	64.7	1.12	0.39	0.55	2.70	1.57	2.47	0.13	13.90	9.95	14.67
	median	67.2	0.89	0.33	0.45	1.80	1.38	2.45	0.13	14.25	9.15	12.67
	max	85.6	5.21	1.55	2.19	16.58	4.39	2.97	0.27	23.01	22.65	50.60
3	min	10.0	0.26	0.08	0.13	0.45	0.33	1.91	0.03	2.88	8.75	8.75
	mean	59.3	1.26	0.47	0.63	2.66	1.16	2.36	0.12	16.28	10.61	17.03
	median	61.6	1.02	0.39	0.54	1.84	1.03	2.34	0.11	16.55	9.30	14.08
	max	83.5	5.12	1.86	2.46	11.78	2.64	2.81	0.23	27.50	68.35	91.95
5	min	-5.1	0.27	0.08	0.13	0.46	0.36	1.95	0.03	3.64	8.75	8.75
	mean	55.5	1.39	0.52	0.70	2.78	1.17	2.38	0.12	16.23	11.01	18.41
	median	58.2	1.13	0.44	0.60	2.01	1.02	2.37	0.12	16.72	9.50	14.83
	max	82.3	5.77	2.01	2.65	12.01	3.31	2.88	0.23	28.30	68.30	91.95
7	min	-18.7	0.27	0.08	0.13	0.47	0.38	1.91	0.03	4.50	8.75	8.75
	mean	52.1	1.51	0.57	0.76	2.92	1.21	2.40	0.13	15.96	11.00	20.34
	median	55.5	1.24	0.48	0.67	2.11	1.07	2.40	0.12	16.52	9.70	15.63
	max	81.6	6.64	2.31	3.05	12.04	3.51	2.91	0.24	28.20	29.80	93.65
9	min	-41.8	0.27	0.08	0.13	0.48	0.35	1.74	0.03	5.32	8.75	8.75
	mean	48.5	1.62	0.62	0.82	3.14	1.22	2.41	0.13	15.51	11.49	22.88
	median	52.4	1.38	0.53	0.70	2.27	1.08	2.41	0.12	16.25	9.78	16.45
	max	81.5	6.81	2.55	3.28	15.14	3.06	2.86	0.27	26.80	37.30	105.30
11	min	-40.7	0.26	0.08	0.13	0.51	0.39	1.94	0.03	4.73	8.75	8.75
	mean	45.9	1.73	0.66	0.87	3.25	1.28	2.44	0.14	15.19	11.90	24.40
	median	50.4	1.45	0.56	0.75	2.35	1.17	2.44	0.12	16.04	10.20	19.02
	max	81.2	7.15	2.90	3.62	17.97	3.19	2.87	0.30	24.98	37.40	110.00

Table 3.12: *Minimum, mean, median and maximum tumor prediction metrics of parameter setting with largest average PR determined for each k separately; metrics are given for different k using the plug-in RLS ARX predictor*

3.3.3 Extended Recursive Least-Squares Algorithm

According to Haykin [125], the standard RLS algorithm with exponential forgetting can be reformulated as a special case of a Kalman filtering problem [136]. The standard formulation for Kalman filters can be based on a discrete state-space model. The following single-input single-output (SISO) state-space model is considered here:

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t) + w(t) \\ y(t) &= Cx(t) + v(t) \end{aligned} \tag{3.45}$$

with the state $x(t) \in \mathbb{R}^n$, system matrix $A \in \mathbb{R}^{n\times n}$, input matrix $B \in \mathbb{R}^n$, output matrix $C \in \mathbb{R}^{1\times n}$, input vector $u(t) \in \mathbb{R}$, output $y(t) \in \mathbb{R}$ where n is the model order. The random variable $w(t) \in \mathbb{R}^n$ models process noise, assumed to be white and following a normal distribution with zero mean and variance of $Q \in \mathbb{R}^n$, whereas the random variable $v(t) \in \mathbb{R}$ is the measurement noise, also white with zero mean and variance of $R \in \mathbb{R}$. Furthermore, both noise terms are assumed to be independent of each other.

The correspondence between the RLS algorithm and a Kalman filter can be setup by employing the state-space model [137]

$$\begin{aligned} x(t+1) &= \frac{1}{\sqrt{\lambda}}x(t) \\ y(t) &= \varphi^T(t-1)x(t) + v(t) \end{aligned} \tag{3.46}$$

which describes an unforced dynamic motion. The model is called "unforced" because of the lack of an input $u(t)$. The system matrix is a diagonal matrix with element values of $1/\sqrt{\lambda}$, where λ represents the exponential forgetting factor. Furthermore, no process noise $w(t)$ is present. In contrast to the assumption on the measurement noise $v(t)$ of the general model (3.45), $v(t)$ is drawn from a normal distribution with zero mean and unit variance. The output matrix C is replaced here by the observation vector $\varphi(t-1)$, which is multiplied to the state vector $x(t)$. Comparing this to Eq. (3.34), $x(t)$ can be associated with the parameter vector $\theta(t)$ of an AR model. Thus the order n can be identified with the order n_a of the AR model. Note that by using $\varphi(t-1)$ instead of $\varphi(t)$, the model is expressed as a prediction model performing a one-step prediction by multiplying with the parameters contained in $x(t)$.

By inserting this model into the common Kalman filter equations (described in Section 3.4), the same equations as the ones for the RLS algorithm result on simplification, which is possible due to the special structure of the unforced dynamical model. Since in the resulting equations the covariance matrix is propagated from one sampling instant to the next, this variant of the Kalman filter is called covariance filtering algorithm [125]. The benefit of this rewriting emerges when adapting the unforced dynamical model in order to generate extended versions of the RLS algorithm. For example, a process noise $w(t)$ could be introduced into that model. Since there is no obvious way to determine the process noise given the unforced dynamical model in the context of breathing or tumor motion modeling, this approach is neglected here. As can be seen from the output equation of the model (3.46), the standard RLS algorithm already includes the measurement noise $v(t)$, but it is assumed to be normally distributed with zero mean and unit variance. Since the distribution of measurement noise can actually be determined because breathing or tumor position is measurable, an extended recursive least-squares (ERLS) algorithm can be derived using Kalman filter theory which introduces a known variance R of the measurement noise. After performing all the required calculations, it turns out that the only change compared to the standard RLS equations is found in Eq. (3.42) which becomes

$$K = \frac{\lambda^{-1}P(t-1)\varphi(t-k)}{R + \lambda^{-1}\varphi^T(t-k)P(t-1)\varphi(t-k)} \ . \tag{3.47}$$

That is, the augend in the denominator with a value of one is identified with the variance R of the measurement noise, which coincides with the assumption of unit variance of the measurement noise.

As already mentioned, it is possible to determine the noise of breathing or tumor motion. This was performed in preliminary experiments showing small improvements on the prediction metrics. However, treating the variance R as a parameter, thereby introducing an additional degree of freedom into the algorithm, enables tuning of all parameters including the measurement noise R. Due to temporary or permanent deviations from assumptions like normally distributed noise or due to time-varying distributions, a value of R different from the determined one can lead to improved results. In addition to parameters λ, n_a and δ, R was varied between 10^{-3} and 10^{1} in simulations with the ERLS algorithm (class I). When choosing the parameter setting with largest average PR for each k individually, the metrics summarized in Table 3.13 result. In comparison with the corresponding table of the standard class I RLS algorithm (cf. Table 3.9) it can be seen that almost all prediction accuracy metrics are equal (up to the given precision), except for small improvements for min. PR for $k = 1$, max. PR for $k = 11$ and max. CI95 for $k = 7$ as well as small degradations for max. CI95 for $k = 9$ and $k = 11$. Additionally, for almost all MAX values for $k \geq 5$, the ERLS algorithm shows slightly better results. Despite these small improvements in prediction accuracy, all smoothness metrics clearly demonstrate improvements over all values. Furthermore, there is also a small improvement on some convergence time values.

Prediction horizon	Type	PR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]	S_F	S_J	S_{TS}	S_{SNR}	CT25 [s]	CT10 [s]
1	min	80.9	0.01	0.00	0.00	0.02	0.37	1.85	0.07	1.85	8.75	8.75
	mean	93.1	0.17	0.06	0.08	0.49	1.20	2.48	0.13	15.93	8.76	9.17
	median	93.9	0.16	0.06	0.08	0.44	1.11	2.49	0.12	16.53	8.75	8.75
	max	97.9	0.37	0.12	0.17	2.30	2.96	2.89	0.32	23.76	9.65	17.65
3	min	45.5	0.02	0.01	0.01	0.04	0.58	1.99	0.08	1.34	8.75	8.75
	mean	81.9	0.45	0.17	0.23	1.07	1.65	2.59	0.15	14.03	9.01	15.31
	median	83.9	0.43	0.16	0.22	0.93	1.47	2.59	0.13	14.58	8.75	10.42
	max	92.4	1.10	0.45	0.57	4.65	3.50	2.98	0.41	22.49	13.75	75.55
5	min	33.1	0.04	0.02	0.02	0.07	0.66	2.00	0.08	1.58	8.75	8.75
	mean	72.5	0.72	0.27	0.36	1.58	1.95	2.64	0.17	12.82	10.02	19.80
	median	76.5	0.64	0.24	0.32	1.35	1.81	2.64	0.14	13.59	9.25	13.45
	max	88.5	2.07	0.77	1.00	8.30	4.75	3.01	0.40	20.60	31.00	90.05
7	min	12.6	0.05	0.02	0.03	0.10	0.72	2.06	0.08	2.10	8.75	8.75
	mean	63.0	1.00	0.37	0.50	2.05	2.05	2.70	0.18	12.01	11.17	24.73
	median	69.0	0.89	0.34	0.44	1.79	1.85	2.68	0.16	13.06	9.60	17.92
	max	85.2	2.75	1.12	1.46	11.92	4.87	3.06	0.41	19.37	52.45	92.55
9	min	-0.9	0.07	0.03	0.04	0.12	0.64	1.97	0.08	1.38	8.75	8.75
	mean	54.9	1.24	0.47	0.62	2.42	2.02	2.67	0.18	11.70	12.05	27.99
	median	60.9	1.13	0.41	0.55	2.13	1.87	2.66	0.16	12.72	10.17	23.75
	max	82.0	3.64	1.47	1.88	13.92	4.71	3.04	0.37	19.30	56.60	102.15
11	min	-8.2	0.08	0.03	0.04	0.13	0.66	1.98	0.08	2.71	8.75	8.75
	mean	48.2	1.45	0.54	0.72	2.71	1.97	2.65	0.18	11.69	12.68	31.39
	median	54.0	1.31	0.49	0.65	2.45	1.87	2.66	0.17	12.28	10.45	28.55
	max	78.6	4.55	1.75	2.23	15.97	4.43	3.02	0.39	19.83	56.60	104.20

Table 3.13: *Minimum, mean, median and maximum breathing prediction metrics of parameter setting with largest average PR determined for each k separately; metrics are given for different k using the plug-in ERLS predictor*

As a consequence, by extending the RLS algorithm in order to include a specific noise variance instead of assuming unit variance as in the standard RLS algorithm, it is possible to achieve the same prediction accuracy with an improved smoothness. This result can also be intuitively understood by realizing that in making an algorithm aware of improved knowledge about noise, it inherits the ability to reduce noise effects in predictions.

3.4 Kalman Filters

Kalman filters, often employed for smoothing or filtering, are especially suitable for performing predictions by virtue of the fact that they already include a one-step prediction. As already mentioned, the formulation of the Kalman filters in the context of this work is based on the state-space model (3.45), including process and measurement noise with known variances of Q and R, respectively.

For the purpose of breathing or tumor motion prediction the state-space model should somehow describe the dynamics of that motion for a single axis. Hence, the output $y(t)$ of the model refers either to breathing or tumor motion. Before several motion models are introduced in the following subsections, the update recursion of the linear Kalman filter is summarized, which is common to all linear formulations irrespective of the underlying state-space model.

Essentially the Kalman filter performs an estimation $\hat{x} \in \mathbb{R}^n$ of the true state $x \in \mathbb{R}^n$ of the state-space model and its covariance matrix $P \in \mathbb{R}^{n \times n}$ based on past observations of the output y. Thereby, the estimated state $\hat{x}$ is the mean of the state distribution and the covariance matrix P is associated with the variance of the state distribution, which is normally distributed if the process noise and measurement noise are normally distributed with zero mean.

As starting point, the Kalman filter requires an initial guess of $\hat{x}(0)$ and $P(0)$. The first step in the Kalman filter recursion, referred to as the time update, consists of the projection of the estimated state at the previous sampling instant one step ahead to the current sampling instant using the state equation

$$\hat{x}_{-}(t) = A\hat{x}(t-1) + Bu(t-1) \ , \tag{3.48}$$

where A and B are the system and input matrices of the state-space model. This delivers the a priori state estimate $\hat{x}_{-}(t)$ for sampling instant t which is computed using information up to time $t-1$. Similarly, the covariance matrix $P(t-1)$ is also projected one step ahead to the current sampling instant based on the state transition matrix A and the variance Q of the process noise:

$$P_{-}(t) = AP(t-1)A^T + Q \ . \tag{3.49}$$

In order to proceed with the measurement update step, a measurement $y(t)$ needs to be obtained for the current sampling instant t. Then, the Kalman gain K is computed by

$$K = \frac{P_{-}(t)C^T}{CP_{-}(t)C^T + R} \ . \tag{3.50}$$

Note the correspondence with Eq. (3.47) from the ERLS algorithm when substituting $P_{-}(t)$ by $\lambda^{-1}P(t-1)$, which is a direct result from Eq. (3.49) obtained by inserting $1/\sqrt{\lambda}$ as system matrix A and setting $Q = 0$. The Kalman filter recursion proceeds by determining the a priori error $e_{-}(t)$, also referred to as innovation,

$$e_{-}(t) = y(t) - C\hat{x}_{-}(t) \ ,$$

by using the output equation from the state-space model applied to the a priori state estimate $\hat{x}_-(t)$. Together with the Kalman gain K, the innovation is used to derive the a posteriori (after obtaining a noisy measurement of the system's output) state estimate $\hat{x}(t)$:

$$\hat{x}(t) = \hat{x}_-(t) + Ke_-(t) \ . \tag{3.51}$$

Note the equivalent structure of Eq. (3.51) with the parameter update Eq. (3.44) of the RLS algorithm. The update of the covariance matrix $P(t)$ with

$$P(t) = P_-(t) - KCP_-(t) \ .$$

completes the measurement update step and the update recursion of the Kalman filter.

Substituting the a posteriori state estimate $\hat{x}(t)$ into the output equation of the (undisturbed) state-space model, the filtered output of the system can be obtained, showing the filtering property of this approach. However, the goal here is to generate predictions which are readily available by continuing with time update equation (3.48). Shifting this by one sampling instant, the a priori state estimate $\hat{x}_-(t+1)$ for the next sampling instant $t+1$ is obtained by

$$\hat{x}_-(t+1) = A\hat{x}(t) + Bu(t) \ , \tag{3.52}$$

assuming that the input $u(t)$ is known. Using the output equation, the one-step prediction $\hat{y}(t+1|t)$ is derived from Eq. (3.52) with

$$\hat{y}(t+1|t) = C\hat{x}_-(t+1) \ . \tag{3.53}$$

Further iterations over Eqn. (3.52) – (3.53) enable multi-step predictions. These occur in the form of a plug-in predictor since further predictions are derived based on predicted states. Note that in general, direct predictors cannot be developed when relying on state-space models since the state equation only establishes a relationship between two consecutive sampling instants.

3.4.1 Linear Kalman Filter with Constant Velocity Model

One popular linear model, which is often used in conjunction with Kalman filters, is the constant velocity model [138]. It can be applied if the variable to be predicted can be stated in terms of the physical properties position and velocity, which is feasible for breathing or tumor motions. The corresponding state-space model with a similar structure as in Eq. (3.45) is:

$$\begin{aligned} x(t+1) &= \begin{bmatrix} 1 & \Delta\tau \\ 0 & 1 \end{bmatrix} x(t) + \begin{bmatrix} \frac{\Delta\tau^2}{2} \\ \Delta\tau \end{bmatrix} a(t) \\ y(t) &= \begin{bmatrix} 1 & 0 \end{bmatrix} x(t) + v(t) \end{aligned} \ ,$$

where the state $x(t) = (x_p(t), x_v(t))^T$ is composed of the position $x_p(t) \in \mathbb{R}$ and velocity $x_v(t) \in \mathbb{R}$. $\Delta\tau$ denotes the sampling time in seconds in between two consecutive sampling instants. The inclusion of the sampling time introduces an interesting feature into the model, which is not available with AR models, in that it can seamlessly deal with variable sampling times, if required. Generally, the model assumes a constant velocity in between two sampling instants which can be seen from the second state equation involving $x_v(t+1) = x_v(t) + \Delta\tau a(t)$. Following the idea of [139], the process noise $a(t) \in \mathbb{R}$ allows for a change of velocity, given in units of acceleration. Furthermore, it reduces the number of design parameters as two would normally be required for the process noise (one for each state). The first state equation for $x_p(t+1)$ simply integrates the velocity once and adds the change in position due to the acceleration given by $a(t)$. The output equation selects the first state since the position refers to the breathing or tumor position, to which the measurement noise $v(t) \in \mathbb{R}$ is added.

The process noise $a(t)$ is not directly added to each state variable as in Eq. (3.45). When substituting $w(t) = \left(\Delta\tau^2/2, \Delta\tau\right)^T a(t)$ and assuming that $a(t)$ is normally distributed with zero mean and variance $S \in \mathbb{R}$, it follows from the displacement theorem that the variance Q of $w(t)$ is

$$Q = \begin{bmatrix} \frac{\Delta\tau^2}{2} \\ \Delta\tau \end{bmatrix} S \begin{bmatrix} \frac{\Delta\tau^2}{2} & \Delta\tau \end{bmatrix} = \begin{bmatrix} \frac{\Delta\tau^4}{4} & \frac{\Delta\tau^3}{2} \\ \frac{\Delta\tau^3}{2} & \Delta\tau^2 \end{bmatrix} S \ ,$$

which can then be included into the Kalman filter equations.

From preliminary analysis it was found that fast convergence is observed when setting the first state variable $x_p(0)$ of the initial state to the first measurement and the second state variable $x_v(0)$ to the linear velocity in between the second and first position measurement. In an online system this would require two measurements which is usually at least an order of magnitude smaller than for AR models, where the minimum number of samples required to start the first prediction cycle is lower-bounded by n_a. With this nearly exact initialization of the initial state, the choice of the initial covariance matrix $P(0)$ plays only a minor role. For convenience, it can be initialized with the identity matrix I_2.

Employing the constant velocity (CV) model in a Kalman filter, further denoted as Kalman CV, leaves two design parameters which need to be selected prior to operation: the variance R of measurement noise and the variance S of the acceleration. In analogy to the ERLS algorithm, R can be determined from the data, however treating it as a design parameter allows for tuning and observing the influence of this parameter on prediction metrics. For the simulation trials in the class I formulation of this approach, R was varied between 10^{-3} and 10^2 with different step sizes and S was varied between 10^{-8} and 10^8 in powers of 10.

At first, instabilities were analyzed which are depicted in Figure 3.35 for several prediction horizons k. For $k = 1$ the valley with no instabilities is constrained by the ratio between acceleration noise S and measurement noise R. If $10^2 \leq \frac{S}{R} \leq 10^9$, then no in-

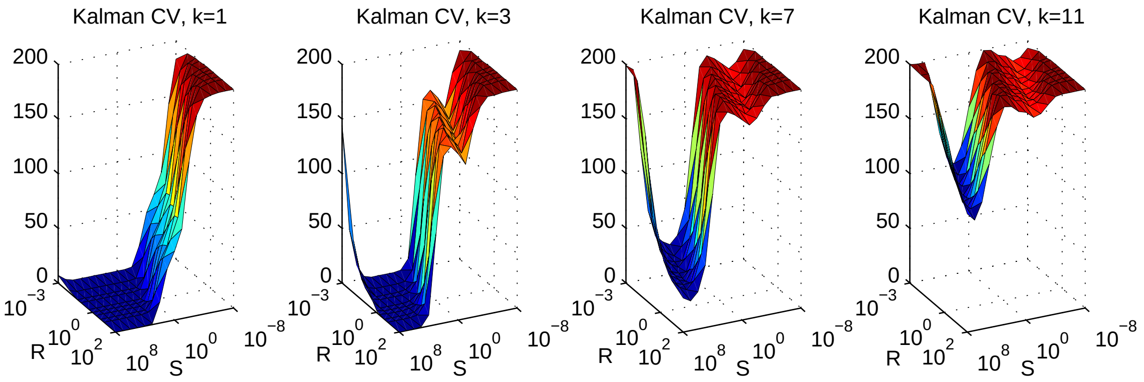


Figure 3.35: *Number of unstable trials among 200 datasets of the Kalman CV predictor in relation to parameters R and S*

stabilities are found on any of the datasets. For increasing prediction horizon, the valley becomes smaller. For $k = 3$, still no instabilities are possible with $10^4 \leq \frac{S}{R} \leq 10^7$. But for $k = 5$ (not shown in the figure), the minimum number of instabilities is found to be three and constrained to $\frac{S}{R} \approx 10^5$. This constraint is maintained at least until $k = 11$, but as can be seen from the figure with a strongly increasing minimum number of instabilities: 27 for $k = 7$, 56 for $k = 9$ and 127 for $k = 11$. It is interesting to note that the number of instabilities is exactly the same if the ratio $\frac{S}{R}$ is kept constant. Comparing the number of instabilities to the previously discussed approaches, a large increase of instabilities can be observed, especially for $k \geq 5$.

The dependency on the noise ratio is also revealed by plots of the average PR as shown in Figure 3.36. Note that white regions in the plots are caused by parameter settings which lead to instabilities (or negative PRs) for all 200 datasets as depicted in Figure 3.35. Consequently there is no value available for these regions which are to be removed from the allowed parameter ranges. The ratios of $\frac{S}{R}$ with maximum PR are found to approximately

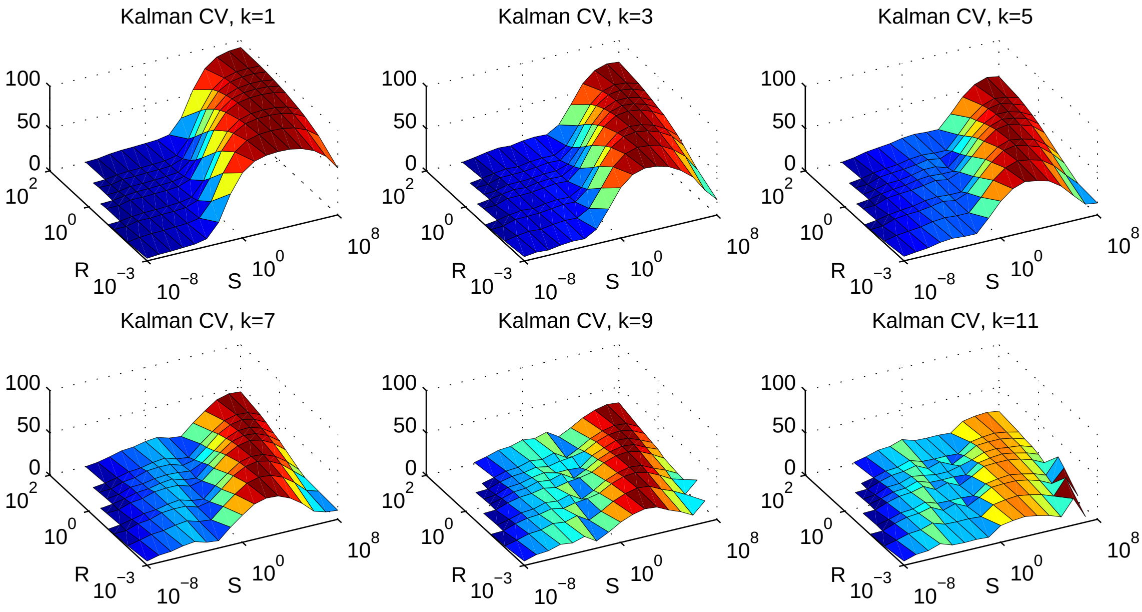


Figure 3.36: *Mean prediction ratio of 200 datasets of the Kalman CV predictor in relation to parameters R and S*

be: 10^6 for $k = 1$ and 10^5 for $k \geq 3$. Hence, the parameter settings leading to best average PRs are nearby. Unlike other prediction methods, one can observe a quite strong impact of the parameter S on the PR. If it is chosen too small (given in relation to the parameter R), PRs near zero or smaller results. The reason is that the acceleration enters the model in form of process noise and a certain minimum acceleration, which is determined by the noise variance, needs to be present, driving the motion of breathing or tumor signals. Equivalent to the number of instabilities, exactly the same PR value can be found for each k for a constant ratio $\frac{S}{R}$. That is, from a given variance for one of the noises, the variance of the other one can be determined which leads to maximum average PRs (and simultaneously to least number of instabilities). A reason for the dependency on this ratio can also be given from Eqn. (3.49) and (3.50), showing that the Kalman gain K (from which the state update is determined) is approximately proportional to this ratio. Since Q

(which is directly depending on S), enters mainly in the numerator, a too small S effects a too small Kalman gain. Therefore, the state update is too small. When observing the predictions on individual datasets directly, the amplitudes of the predicted motion are always considerably smaller if S is chosen too small.

However, in this case, quite smooth predictions result, which can be seen from the corresponding smoothness plots, represented by the ones for S_F shown in Figure 3.37. The regions with poor PR (cf. Figure 3.36) are found to yield an exceptionally good smoothness metric S_F close to zero, irrespective of the prediction horizon k. A too small variance in these regions simultaneously leads to smooth predictions because the maximum change in velocity and position is upper-bounded by the maximum acceleration, which is specified as variance. Similar to the previous plots, a constant ratio $\frac{S}{R}$ results in the same smoothness values, which is another indication for yielding the same predictions as long as the noise ratio is the same. The line with maximum PR is approximately coinciding with the line for which the change in smoothness is the largest, i.e. where the hill in the smoothness plot is the steepest. This line can be also seen as a contour line of a topographic map.

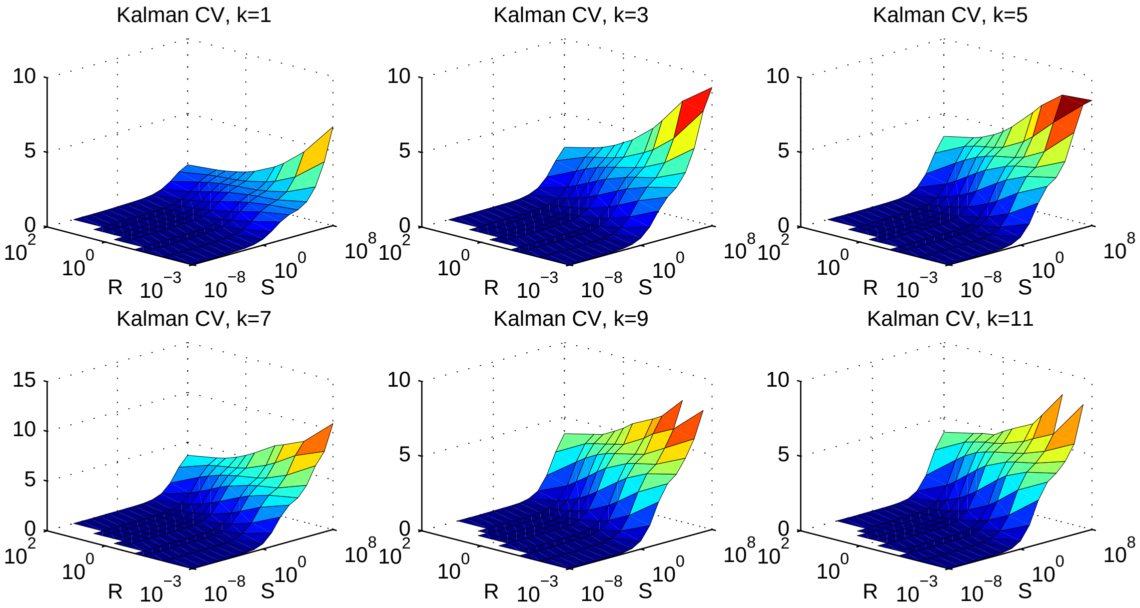


Figure 3.37: *Mean smoothness metric S_F of 200 datasets of the Kalman CV predictor in relation to parameters R and S*

Table 3.14 summarizes the prediction metrics when choosing the parameter setting with largest average PR for each k individually. In comparison with Table 3.9 including the metrics of the plug-in RLS predictor, it becomes obvious that comparable results are only achieved for $k = 1$, although the RLS predictor is superior on all metrics except for maximum S_{TS}. With increasing prediction horizon, the metrics show an increasing difference to the ones of the RLS predictor, especially for the prediction accuracy metrics. The only exception is the metric S_J which slightly improves with Kalman CV predictor for $k \geq 3$. However, the remaining smoothness metrics generally show lesser smoothness for the Kalman CV predictor. When inspecting the predictions of individual datasets, an overshoot in the vicinity of the peaks of the breathing motion can be observed, where the amount of overshoot increases with rising prediction horizons. This increase is also present

Prediction horizon	Type	PR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]	S_F	S_J	S_{TS}	S_{SNR}	CT25 [s]	CT10 [s]
1	min	77.1	0.01	0.00	0.00	0.02	0.42	1.97	0.07	1.66	8.75	8.75
	mean	91.8	0.20	0.06	0.10	0.54	1.60	2.53	0.14	15.24	8.76	9.15
	median	92.8	0.20	0.06	0.09	0.48	1.54	2.54	0.13	15.80	8.75	8.75
	max	97.6	0.47	0.14	0.21	2.70	4.09	3.02	0.31	22.69	9.65	19.25
3	min	15.7	0.03	0.01	0.02	0.05	0.61	1.96	0.08	-0.15	8.75	8.75
	mean	75.7	0.63	0.22	0.30	1.32	2.25	2.52	0.16	12.57	9.17	14.87
	median	79.3	0.61	0.22	0.30	1.22	2.19	2.54	0.15	13.28	8.78	10.75
	max	89.7	1.64	0.48	0.70	5.82	5.45	2.93	0.36	19.72	19.25	107.95
5	min	-46.6	0.06	0.03	0.03	0.10	0.81	2.08	0.10	-0.99	8.75	8.75
	mean	58.5	1.15	0.37	0.53	2.35	2.86	2.63	0.20	10.51	10.66	26.88
	median	64.5	1.08	0.36	0.51	2.19	2.84	2.66	0.19	11.29	9.35	22.25
	max	82.0	3.13	0.91	1.32	10.00	5.99	3.00	0.47	17.37	55.55	107.95
7	min	-111.2	0.09	0.04	0.05	0.16	0.95	2.15	0.11	-1.50	8.75	9.00
	mean	36.6	1.79	0.59	0.82	3.41	3.23	2.69	0.23	9.14	13.92	38.76
	median	45.8	1.70	0.57	0.79	3.21	3.30	2.71	0.22	10.02	10.50	36.50
	max	72.6	4.40	1.43	2.00	14.18	6.13	3.04	0.55	16.12	100.50	107.95
9	min	-171.3	0.12	0.06	0.07	0.23	0.72	1.94	0.11	-0.75	8.75	9.85
	mean	12.0	2.57	0.86	1.18	4.22	2.59	2.51	0.19	9.52	16.86	44.72
	median	22.4	2.44	0.82	1.12	4.03	2.64	2.53	0.18	10.30	11.75	43.80
	max	61.1	5.84	2.18	2.77	13.11	5.09	2.90	0.41	16.84	107.95	124.75
11	min	-234.3	0.16	0.07	0.08	0.29	0.95	2.10	0.12	-1.31	8.75	10.35
	mean	-12.9	3.22	1.13	1.51	5.42	3.19	2.64	0.23	8.29	22.88	49.71
	median	-0.9	3.06	1.08	1.44	5.25	3.27	2.66	0.22	8.97	17.10	48.12
	max	48.6	7.12	2.85	3.44	19.31	5.70	3.01	0.53	15.69	107.95	124.75

Table 3.14: *Minimum, mean, median and maximum breathing prediction metrics of parameter setting with largest average PR determined for each k separately; metrics are given for different k using the Kalman CV predictor*

in other predictors, however the effect is more profound with the Kalman CV predictor. When the breathing motion is more linear (in between inhalation and exhalation), the prediction accuracy is better than during the peaks where the motion is rather nonlinear. The adoption of a linear model in the form of the CV model, cannot deal with the nonlinear effects in the vicinity of peaks. Hence, there are temporary deviations between predictions and true values, which are growing with increasing prediction horizon.

Of course, the situation improves when selecting the parameter setting with largest PR for each dataset and each k separately, but the resulting metrics (not shown here) still show lesser performance than the RLS metrics with parameter settings chosen from average PR. Therefore, even when choosing the parameter setting with best possible PR, the performance of the RLS predictor is not reached by the Kalman CV approach on any metric.

Furthermore, this approach cannot be extended to a class II predictor like how an AR model was extended to an ARX model. Although it would be possible from the structure of state-space models to include an input through the term $Bu(t)$ in the state equation, to the knowledge of the author there is no obvious way available how to select or derive the input matrix B which would describe the physical influence of (future) breathing position predictions on tumor motion position and velocity, leading to an acausal model. However it could be fused together with an ARX model in such a way that at first breathing predictions are performed with the Kalman CV approach and then, these are fed as input to an ARX model. The parameter vector of the ARX model can then be updated by the RLS or ERLS algorithm in order to perform tumor predictions. Since the breathing prediction performance of the Kalman CV approach does not reach the one of the RLS predictor, this step was omitted. Instead, an extension to the CV model was tested, which is followed up in the next subsection.

3.4.2 Linear Kalman Filter with Constant Acceleration Model

The constant acceleration model [138] is a direct extension of the constant velocity model. As the name suggests, it assumes a constant acceleration between two sampling instants. It can be derived in a straight forward way by augmenting the state of the constant velocity model with an additional state variable $x_a(t) \in \mathbb{R}$ representing acceleration. Performing the required adaptions from physical relationships on how acceleration affects position and velocity, the following state-space model results:

$$\begin{aligned} x(t+1) &= \begin{bmatrix} 1 & \Delta\tau & \frac{\Delta\tau^2}{2} \\ 0 & 1 & \Delta\tau \\ 0 & 0 & 1 \end{bmatrix} x(t) + \begin{bmatrix} \frac{\Delta\tau^2}{2} \\ \Delta\tau \\ 1 \end{bmatrix} a(t) \\ y(t) &= \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} x(t) + v(t) \end{aligned} ,$$

where the state $x(t)$ is defined as $x(t) = (x_p(t), x_v(t), x_a(t))^T$. The process noise in the form of acceleration $a(t)$ still affects the first two state variables (position and velocity) in the same way as in the constant velocity model. It is directly added to the last state variable as this is already an acceleration, thereby respecting the physical units in the overall model. The variance Q can be derived in analogy to the constant velocity, yielding

$$Q = \begin{bmatrix} \frac{\Delta\tau^2}{2} \\ \Delta\tau \\ 1 \end{bmatrix} S \begin{bmatrix} \frac{\Delta\tau^2}{2} & \Delta\tau & 1 \end{bmatrix} = \begin{bmatrix} \frac{\Delta\tau^4}{4} & \frac{\Delta\tau^3}{2} & \frac{\Delta\tau^2}{2} \\ \frac{\Delta\tau^3}{2} & \Delta\tau^2 & \Delta\tau \\ \frac{\Delta\tau^2}{2} & \Delta\tau & 1 \end{bmatrix} S \, .$$

The same simulations as for the CV model were performed on the Kalman filter with the constant acceleration (CA) model, further denoted as Kalman CA. The arising instabilities in terms of the noise variances R and S are depicted in Figure 3.38. The structure of the

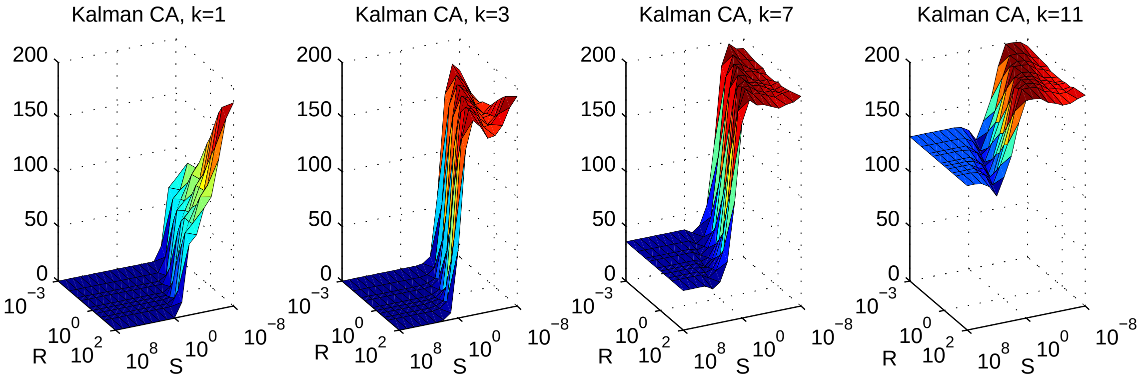


Figure 3.38: *Number of unstable trials among 200 datasets of the Kalman CA predictor in relation to parameters R and S*

instability plots in terms of the noise ratio $\frac{S}{R}$ is approximately similar to the CV model except that there is no increase in instabilities for larger noise ratios (at least not in the chosen intervals for R and S). For $k = 1$, no instabilities are found for $\frac{S}{R} \geq 10^{-2}$. For $k = 3$ there are parameter settings available leading to no instabilities if $\frac{S}{R} \geq 10^{1}$. For $k = 5$ the minimum is found to be 3 at $\frac{S}{R} \approx 10^{2}$, which is equal to Kalman CV. With instabilities of 32, 66 and 115 for prediction horizons of 7,9, and 11, respectively, the Kalman CA predictor results in slightly more instabilities than Kalman CV. In these cases, the minimum number of instabilities is reached if $\frac{S}{R} \approx 10^{2}$, which is constant for $3 \leq k \leq 11$.

The average PR is displayed in Figure 3.39. In these plots, the dependency on the noise ratio $\frac{S}{R}$ is also clearly visible with the same properties as for the CV model. However, a major difference is that if the maximum value is reached by a sufficiently large ratio, a certain height of the resulting plateau is kept which is located at slightly smaller value than the maximum. The difference in height between the plateau and the maximum increases moderately with increasing prediction horizon. The noise ratios with maximum PR, extracted from the figure result approximately in: 10^{3} for $k = 1$, 10^{2} for $3 \leq k \leq 9$ and 10^{1} for $k = 11$. Thus, the noise ratios should be chosen about three to four orders of magnitude smaller than for the CV model.

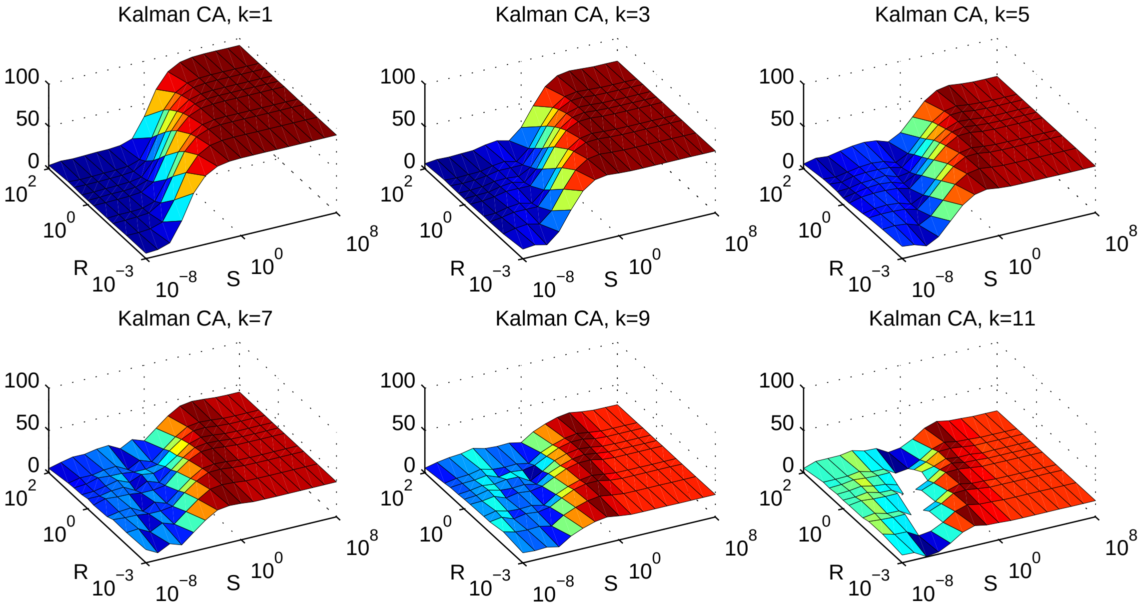


Figure 3.39: *Mean prediction ratio of 200 datasets of the Kalman CA predictor in relation to parameters R and S*

Figure 3.40 shows the metric S_F as a representative for smoothness measures. Similar to the CV model, the general structure of this plot is equivalent to the PR plot. However, regions with maximum PR are located in the upper third before reaching the plateau. The considerations for smoothness can be analogously transferred from the CV model. The corresponding numerical metrics are given in Table 3.15. It is interesting to compare this table to the equivalent table of the CV model (cf. Table 3.14). As can be seen, the CA model leads to small improvements on some prediction accuracy metrics (mainly PR) for $k = 1$ and a few improvements for the maximum of MAX for $k > 1$. However, on the

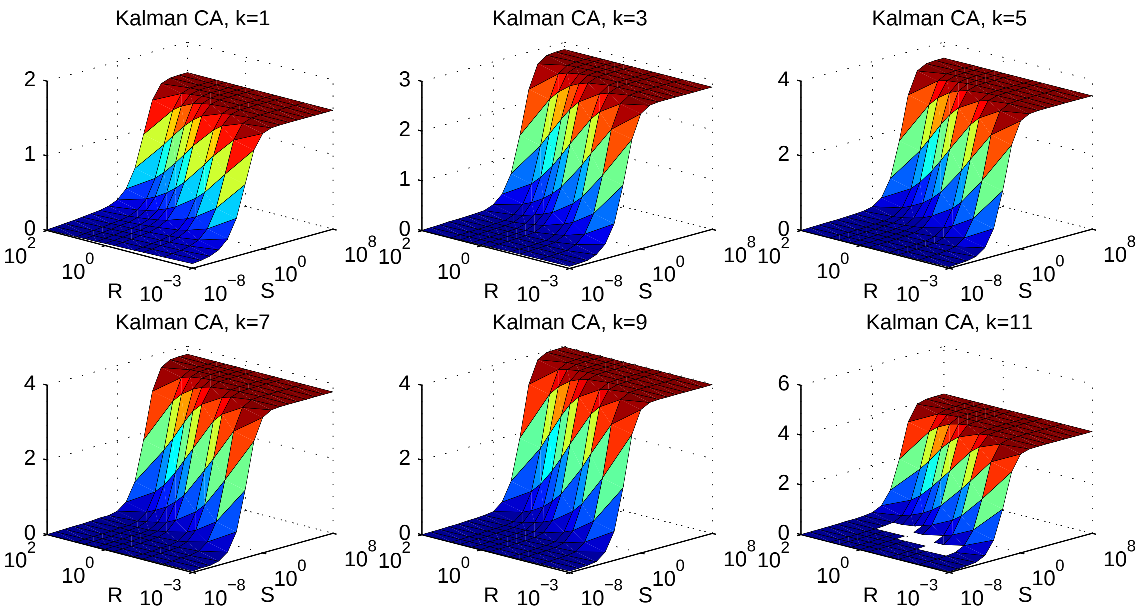


Figure 3.40: *Mean smoothness metric S_F of 200 datasets of the Kalman CA predictor in relation to parameters R and S*

Prediction horizon	Type	PR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]	S_F	S_J	S_{TS}	S_{SNR}	CT25 [s]	CT10 [s]
1	min	77.1	0.01	0.00	0.00	0.02	0.40	1.96	0.07	1.64	8.75	8.75
	mean	91.9	0.20	0.06	0.10	0.54	1.56	2.52	0.14	15.26	8.76	9.12
	median	92.9	0.19	0.06	0.09	0.47	1.50	2.53	0.12	15.82	8.75	8.75
	max	97.6	0.46	0.13	0.20	2.63	4.02	3.01	0.30	22.79	9.65	19.25
3	min	13.8	0.03	0.01	0.02	0.05	0.61	1.96	0.08	-0.18	8.75	8.75
	mean	75.2	0.65	0.22	0.31	1.35	2.25	2.53	0.16	12.47	9.24	15.75
	median	78.9	0.63	0.22	0.31	1.24	2.19	2.54	0.15	13.16	8.80	10.90
	max	89.5	1.70	0.49	0.72	6.00	5.37	2.93	0.36	19.59	19.25	107.95
5	min	-53.3	0.06	0.03	0.03	0.11	0.70	1.98	0.09	-0.88	8.75	8.75
	mean	56.6	1.23	0.39	0.56	2.37	2.55	2.54	0.18	10.77	10.83	27.38
	median	63.1	1.14	0.38	0.54	2.20	2.53	2.57	0.17	11.49	9.35	21.05
	max	81.6	3.32	0.99	1.42	9.04	5.31	2.92	0.41	17.71	55.65	107.95
7	min	-122.8	0.10	0.04	0.05	0.17	0.82	2.05	0.11	-1.38	8.75	9.30
	mean	32.9	1.92	0.63	0.88	3.47	2.89	2.60	0.21	9.45	14.13	40.66
	median	42.5	1.83	0.61	0.84	3.23	2.93	2.62	0.19	10.25	10.65	39.43
	max	71.0	4.82	1.57	2.16	13.14	5.47	2.97	0.47	16.57	100.50	110.70
9	min	-194.4	0.13	0.06	0.07	0.25	0.91	2.09	0.12	-1.32	8.75	10.30
	mean	6.3	2.68	0.91	1.24	4.70	3.09	2.64	0.23	8.57	19.30	47.85
	median	18.6	2.51	0.87	1.18	4.42	3.15	2.65	0.21	9.33	13.73	47.45
	max	58.8	6.36	2.26	2.95	17.40	5.59	3.01	0.52	15.93	107.95	124.75
11	min	-258.4	0.17	0.07	0.09	0.31	0.59	1.82	0.10	0.13	8.75	10.35
	mean	-26.8	3.64	1.32	1.72	5.52	2.15	2.40	0.17	9.49	25.01	49.66
	median	-15.1	3.42	1.24	1.64	5.35	2.17	2.41	0.16	10.17	18.23	48.12
	max	43.4	7.88	3.46	4.08	15.67	4.22	2.81	0.35	16.99	107.95	124.75

Table 3.15: *Minimum, mean, median and maximum breathing prediction metrics of parameter setting with largest average PR determined for each k separately; metrics are given for different k using the Kalman CA predictor*

majority of the metrics, a degradation of performance is visible except for most of the smoothness metrics, where the CA model generally results in an improved performance.

Consequently, in a trade-off between the CV and CA model, one would rather choose the CV model given the resulting performance. However, as demonstrated in [138] and [139], an improvement can be realized by combining both predictors into an interacting multiple model approach [140, 141]. By using a Markov chain, transitions between different models, in this case CV and CA models, can be realized with a certain probability, which is updated according to the likelihood of the Kalman filters, which itself is derived from the covariances matrices computed by the underlying Kalman filters. The final estimated state is formed by a linear combination of all the model states with weights proportional to each model's probability. Since the prediction effectively varies in between the predictions from the CV and CA models, the performance increase by adopting an interacting multiple model approach is not expected to outperform the RLS predictor, especially for large prediction horizons.

3.4.3 Extended Kalman Filter with Spring-Mass Model

Another interesting approach based on a physical model in order to perform predictions of quasi periodic motions is based on a simple spring-mass model [112], in which a mass m is attached to a spring with constant k. The spring is assumed to be massless and fixed on the remaining side. Additionally, the effects of gravity are neglected and the mass can only perform a one-dimensional motion. In continuous time τ, the unforced spring-mass model can be expressed by the differential equation

$$\frac{d^2}{d\tau^2}p(\tau) = -\frac{k}{m}p(\tau) \ ,$$

where $p(\tau)$ is the deviation in position of the mass m from the equilibrium.

From the solution of the differential equation in the form

$$p(\tau) = A\cos(\omega\tau) + B\sin(\omega\tau) \ ,$$

ω can be identified as the natural frequency of the spring-mass system given by $\omega = \sqrt{k/m}$. The coefficients A and $B \in \mathbb{R}$ would need to be determined from given initial conditions. With ω, the differential equation can be simplified to

$$\frac{d^2}{d\tau^2}p(\tau) = -\omega^2 p(\tau) \ .$$

Further substituting $d = \omega^2(\tau)$ yields

$$\frac{d^2}{d\tau^2}p(\tau) = \ddot{p} = -dp(\tau). \tag{3.54}$$

With $\dot{p} = v$, where v is the velocity of the mass and the state $x(\tau) = (p, v)^T$, the second-order continuous time state-space model of the spring-mass system is obtained from Eq. (3.54):

$$\begin{aligned} \dot{x}(\tau) &= \begin{bmatrix} 0 & 1 \\ -d & 0 \end{bmatrix} x(\tau) \\ p(\tau) &= \begin{bmatrix} 1 & 0 \end{bmatrix} x(\tau) \end{aligned} \tag{3.55}$$

with an initial state $x(0) = (y_0, v_0)$ where y_0 is the initial position and v_0 denotes initial velocity of the mass.

In order to incorporate this model into a discrete Kalman filter, the continuous time model needs to be discretized. Based on the analytical solution of the state equation (from the continuous time model) incorporating the system matrix A_c, the discrete time equivalent of the system matrix A_d is generally given by the matrix exponential

$$A_d = e^{A_c T_s} \ ,$$

where T_s denotes the sampling time of the discrete system [142]. There are several ways (e.g. rectangular method, trapezoidal method) [143] to approximate the matrix exponential, leading to system matrices of different complexities. In order to keep the complexity low, the polynomial method [144] is chosen, which is essentially a Taylor expansion of the matrix exponential

$$A_d = e^{A_c T_s} = I + \sum_{i=1}^{n} \frac{T_s^i}{i!} A_c^i \ ,$$

where n denotes the order of the expansion. Unlike [112], a higher order of $n = 3$ is chosen here. To simplify the resulting model further, $T_s = 1$ is used. Then, physical quantities of the discrete time system including the notion of time cannot be interpreted in terms of the original units any more, but in terms of normalized time. For example, the velocity included in the state vector of a continuous time model is converted into increments in position during one sampling period. Performing the required calculations, the following system matrix results for the discrete model:

$$A_d = e^{A_c} = I + \sum_{i=1}^{3} \frac{1}{i!} A_c^i = \begin{bmatrix} 1 - \frac{d}{2} & 1 - \frac{d}{6} \\ \frac{1}{6} d^2 - d & 1 - \frac{d}{2} \end{bmatrix}$$

with A_c being the system matrix of the state-space model (3.55).

The model is intended to follow the quasi-periodic breathing or tumor motion. To describe the dynamics of motion, the parameter d, associated with the ratio between the spring constant k and mass m, is the only choice to accommodate for the dynamics; specifically it represents the square of the motion frequency, whereas the amplitude is determined by the initial condition. Due to variability and hysteresis effects, the parameter d needs to be constantly adapted to make the model follow the motion trace as close as possible. Hence, d is further denoted explicitly depending on time. Furthermore, the first state variable, i.e. the position $p(t)$, was defined as the deviation of the mass's position from the equilibrium when the spring is at rest. If the signal to be predicted is initially not free from the mean position or if the baseline (the position of the equilibrium) changes as time progresses, an estimation of the level of the baseline is required. Augmenting the state vector by the unknowns $d(t)$ and the baseline $b(t)$, a Kalman filter can estimate these unknowns. In total the state vector of the discrete system is finally given by $x(t) = (p(t), v(t), d(t), b(t))^T$. The fact that the system matrix A_d depends on the third state variable $d(t)$ renders the initially linear state-space model nonlinear. The complete model including process $w_P(t) \in \mathbb{R}^4$ and measurement noise $w_M(t) \in \mathbb{R}$ is as follows:

$$
\begin{aligned}
x(t+1) = \begin{bmatrix} p(t+1) \\ v(t+1) \\ d(t+1) \\ b(t+1) \end{bmatrix} &= \underbrace{\begin{bmatrix} \left(1 - \frac{d(t)}{2}\right) p(t) + \left(1 - \frac{d(t)}{6}\right) v(t) \\ \left(\frac{1}{6} d^2(t) - d(t)\right) p(t) + \left(1 - \frac{d(t)}{2}\right) v(t) \\ d(t) \\ b(t) \end{bmatrix} + w_P(t)}_{f(x(t), w_P(t))} , \qquad (3.56) \\
y(t) &= \underbrace{\begin{bmatrix} 1 & 0 & 0 & 1 \end{bmatrix} x(t) + w_M(t)}_{h(x(t), w_M(t))}
\end{aligned}
$$

with $f(x(t), w_P(t))$ representing the nonlinear state transition matrix. The first two rows of $f(x(t), w_P(t))$ are given by the product A_d and the reduced state $(p(t), v(t))$. The last two rows contain the previous values for $d(t)$ and $b(t)$, thus the undisturbed model assumes them to be constant. However, in presence of process noise, they are allowed to change. $h(x(t), w_M(t))$ is the output function, which for this model remains linear. As indicated in the output function, the final output is derived by the sum of the baseline $b(t)$ and deviation from that baseline given by the position $p(t)$.

In order to adopt this model into a standard Kalman filter, linearization around the current working point is necessary, which is achieved by the Jacobian matrices for $f(x(t), w_P(t))$ and $h(x(t), w_M(t))$, leading to the extended Kalman filter (EKF) [145]. Since the realization of the noise terms is not known at each sampling instant, it is substituted with zero for the Jacobian matrices, which is justified by assuming their distribution to be of zero mean. Computing the Jacobian matrices, the system matrix $A(t)$ for the linear discrete system results in

$$
A(t) = \frac{\partial}{\partial x} f(x(t), 0) = \begin{bmatrix} 1 - \frac{d(t)}{2} & 1 - \frac{d(t)}{2} & \frac{v(t)}{6} - \frac{p(t)}{2} & 0 \\ \frac{1}{6} d^2(t) - d(t) & 1 - \frac{d(t)}{2} & \frac{d(t)p(t)}{3} - p(t) - \frac{v(t)}{2} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} , \qquad (3.57)
$$

which becomes time varying due to the dependence on the first three state variables. Since the output function is linear, the resulting output matrix

$$
C = \frac{\partial}{\partial x} h(x(t), 0) = \begin{bmatrix} 1 & 0 & 0 & 1 \end{bmatrix}
$$

is time invariant, which can directly be used in the Kalman filter Eqn. (3.50) – (3.53). Note that the Jacobian matrices of the noise terms become identity matrices, hence the same noise terms appear as additives in the linearized model. The process noise $w_P(t)$ and the measurement noise $w_M(t)$ are assumed to be normally distributed with zero mean and variance $Q \in \mathbb{R}^{4\times4}$ and variance $R \in \mathbb{R}$, respectively. The system matrix $A(t)$ from Eq. (3.57), which needs to be updated at each sampling instant according to the current state is inserted into Eq. (3.49) for the projection of the covariance matrix during the time update step, whereas the projection of the state given in Eq. (3.48) and the one-step prediction from Eq. (3.52) is completely replaced by the state equation of the undisturbed nonlinear model (3.56) (i.e. substituting the noise terms with zero).

The EKF with spring-mass model requires in total five parameters to be selected: one variance of the measurement noise R and four process noise variances q_p, q_v, q_d and $q_b \in \mathbb{R}$ associated with each of the four state variables. The measurement noise was determined from the datasets to reduce the number of free parameters to four by the following procedure: All datasets were filtered using a properly designed Butterworth filter with a cut-off frequency of 1.2Hz together with a zero-phase filter in order to eliminate phase delay. The difference between the filtered and original signal then delivers the noise signal. Histograms of the noise signals revealed that the noise is approximately normally distributed with a mean close to zero. This procedure was applied to all axes separately. For each dataset and each axis, the variance of the noise signal was determined and averaged over all datasets, resulting in 3.69×10^{-4}, 1.86×10^{-3} and 1.279×10^{-2} given in mm^2 for the x-, y-, and z-axis, respectively. These values were substituted for the parameter R for the corresponding axes. The remaining four parameters describing the process noise were varied in the range 10^{-6} to 10^{6} in steps of 10^{2}.

Due to the dimensionality, the influence of the parameters cannot be shown similar to previously discussed predictors. Instead, common values of individual parameters were determined, leading to the maximum PRs. It was observed that for almost all k, the variance parameter q_b was 1.0 for the parameter setting with maximum average PR. In a further attempt, the assumption was tested if there are constant parameter ratios, which result in maximum PR. For that purpose, the 20 parameter settings which lead to maximum PR were scaled by q_b, resulting in the parameter ratios between the q_p, q_v and q_d in relation to q_b. The number of occurrences were determined for each parameter ratio over all k from 1 to 11 in steps of 2, leading to a total of 120 occurrences. These are depicted in Figure 3.41. Naturally, q_b possesses the maximum number of occurrences with a value of 1 (shown in green color). The narrow distribution of the parameters q_v and q_d between 10^{-2} to 1 and 10^{-6} to 10^{-4} shows that among the 20 best parameter settings the ratio between q_v, q_d and q_b is approximately constant. Conversely, the distribution of q_p is rather uniform but constrained to about 10^{-6} to 1 (there are no occurrences for $q_p \geq 10^{2}$). Consequently, when choosing the ratio between q_v and q_d to q_b as indicated in the figure, q_p can be selected from a larger range between 10^{-6} to 1.

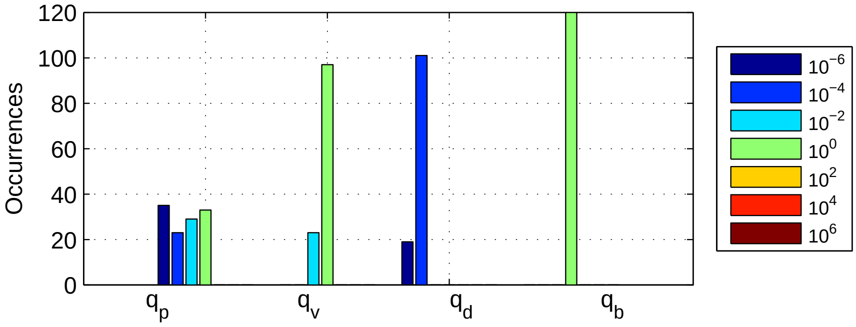


Figure 3.41: *Number of occurrences of parameter values among the 20 parameter settings with largest average PR shown in relation to q_b*

The resulting metrics when choosing the parameter setting with largest average PR for each k individually are given in Table 3.16. When comparing this to the results of the CV and CA model, a considerable increase in almost all metrics can be observed. The increment in PR rises approximately quadratic with increasing k from 0.5% to 39.1%, so

Prediction horizon	Type	PR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]	S_F	S_J	S_{TS}	S_{SNR}	CT25 [s]	CT10 [s]
1	min	77.5	0.01	0.00	0.01	0.02	0.35	1.81	0.07	1.42	8.75	8.75
	mean	92.3	0.19	0.06	0.09	0.52	1.33	2.37	0.13	15.47	8.76	9.01
	median	93.2	0.18	0.06	0.09	0.44	1.26	2.38	0.12	16.08	8.75	8.75
	max	97.4	0.40	0.13	0.19	2.34	3.41	2.87	0.28	23.53	9.70	12.10
3	min	31.4	0.03	0.01	0.02	0.05	0.56	2.03	0.08	-0.63	8.75	8.75
	mean	78.7	0.53	0.19	0.26	1.18	2.12	2.59	0.17	12.58	9.07	15.79
	median	81.7	0.51	0.19	0.26	1.06	2.02	2.61	0.15	13.25	8.75	10.53
	max	91.1	1.29	0.43	0.58	6.52	4.92	3.02	0.38	20.04	12.20	65.60
5	min	-7.0	0.06	0.02	0.03	0.09	0.71	2.13	0.09	-1.89	8.75	8.75
	mean	66.1	0.86	0.32	0.42	1.93	2.67	2.70	0.21	10.28	10.40	28.05
	median	70.9	0.82	0.31	0.42	1.78	2.65	2.72	0.19	10.92	9.22	22.17
	max	85.2	2.17	0.77	1.00	12.06	5.56	3.07	0.49	17.14	38.60	107.95
7	min	-33.9	0.08	0.04	0.05	0.14	0.76	2.09	0.10	-2.90	8.75	8.85
	mean	51.7	1.25	0.48	0.62	2.48	2.74	2.64	0.22	9.00	14.07	39.06
	median	57.9	1.16	0.46	0.60	2.29	2.74	2.64	0.20	9.67	10.37	35.35
	max	78.9	2.98	1.10	1.39	15.06	5.70	3.08	0.54	16.27	62.70	107.95
9	min	-50.3	0.11	0.05	0.06	0.20	0.91	2.16	0.12	-4.09	8.75	10.10
	mean	37.2	1.60	0.63	0.81	3.22	3.13	2.69	0.27	7.32	19.83	47.37
	median	44.0	1.49	0.60	0.78	2.96	3.15	2.70	0.25	7.89	12.45	45.40
	max	71.9	3.74	1.38	1.72	24.10	6.35	3.14	0.67	14.40	100.45	122.65
11	min	-53.6	0.13	0.06	0.07	0.26	0.84	2.12	0.12	-2.21	8.75	10.30
	mean	26.2	1.87	0.77	0.96	3.66	2.96	2.67	0.27	7.22	22.64	49.92
	median	32.4	1.76	0.73	0.92	3.46	2.99	2.68	0.25	7.62	16.00	48.30
	max	65.2	4.83	1.77	2.19	30.15	5.97	3.11	0.63	14.28	100.55	122.80

Table 3.16: *Minimum, mean, median and maximum breathing prediction metrics of parameter setting with largest average PR determined for each k separately; metrics are given for different k using the EKF predictor with spring-mass model*

that especially for large prediction horizons, the spring-mass model delivers a significantly improved prediction accuracy. Also the other prediction accuracy metric like the CI95 improve in a similar way with decrements between 0.01mm for $k = 1$ and 1.35mm for $k = 11$. The only exceptions are a few maximum MAX errors for $k \geq 5$ which are slightly larger for the spring-mass model. The smoothness metrics also show an increase in smoothness except S_{TS} for $k \geq 9$. When comparing the differences between the individual smoothness measures between mass-spring and CV model, it additionally turns out that S_{TS} shows smaller differences than the other smoothness metrics, so that S_{TS} seems to be less reliable. Furthermore, the convergence time criteria show smaller times for the spring-mass model, except for some CT10 values. Despite the overall improvements compared to the CV and CA models, the spring-mass model does not outperform the RLS or ERLS predictor, but comes close. There are a few individual values of some metrics which indicate an improvement for only a few datasets. For smaller prediction horizons the general performance difference is smaller but increases with the prediction horizon.

3.5 Nonlinear Prediction Algorithm

Firstly, the class I formulation of a nonlinear prediction algorithm based on a fundamental theorem from nonlinear time series analysis is presented followed by a detailed analysis of the free parameters' influence. Then, it is shown how the algorithm can be rewritten to a class II formulation to enable breathing-based tumor motion predictions. For both variants, performance metrics are given in a comparative manner in relation to previously discussed prediction methods.

3.5.1 Class I Formulation

A very interesting strategy for organ-motion prediction has been proposed in [123] in the context of minimally invasive robotic cardiac surgery. The methodology is motivated by nonlinear time series analysis based on Takens' Reconstruction Theorem [146]. The reconstruction theorem is stated in literature in a more general form [147], but can be adapted quite well to the formulation of nonlinear dynamical systems. In the following it is assumed that a single axis of breathing or tumor motion is observed in regular time intervals, resulting in a sequence of observations $y(t) \in \mathbb{R}$ with $0 \leq t \leq n$. Furthermore it is assumed that the breathing or tumor motion is caused by an unforced nonlinear dynamic system of the form

$$\begin{aligned} x(t+1) &= f(x(t)) \\ y(t) &= g(x(t)) \end{aligned} \tag{3.58}$$

with state transition function $f(x(t))$ and output function $g(x(t))$. Since the model's state is not directly observable, the model is said to possess an inner state $x(t) \in \mathbb{R}^d$, where d is the dimension of the state vector. The functions f and g are further considered to be unknown. However, it is known that the time series of observations

$$\begin{bmatrix} y(0) & y(1) & \ldots & y(n) \end{bmatrix} = \begin{bmatrix} g(x(0)) & g(x(1)) & \ldots & g(x(n)) \end{bmatrix}$$

is generated by the model (3.58) based on the output function $g(x(t))$, the corresponding state $x(t)$ and the state transition $f(x(t))$ which describes how the state for the next sampling instant evolves from the current one. Takens' reconstruction theorem states that the map

$$x \mapsto \Big(g(x), g(f(x)), \ldots g(f^{p-1}(x))\Big) \tag{3.59}$$

with $x \in \mathbb{R}^d$ is an embedding of the state-space $\mathbb{R}^d$ into the space $\mathbb{R}^p$ for the pair (f, g) provided that f and g are continuously differentiable and that

$$p > 2d \; . \tag{3.60}$$

Given the dynamical model (3.58), the elements of the map (3.59) can be simply identified as the observations $y(t)$ of the output of the dynamic process during p consecutive sampling instants. Using this map, a unique relationship between the unknown d-dimensional state and the p-dimensional embedding vector can be established just by incorporating corresponding observations into the embedding vector. However, this relationship is not expressed explicitly, but according to the reconstruction theorem, the evolution of the system according to the dynamics in state-space as time progresses is equivalent to the evolution of the map in $\mathbb{R}^p$. In other words, having enough knowledge about the history of the system, it is possible to reconstruct the dynamics of the nonlinear system based on past measurements, where the number of past measurements needs to be sufficiently large according to condition (3.60). An embedding vector, also called reconstruction vector, is defined for the current sampling instant t similar to the map (3.59) by incorporating p past observations with a time delay of h sampling instants between subsequent components

[123]:

$$D(t) = \Big[y(t) \quad y(t-h) \quad \dots \quad y(t-h(p-1))\Big]^T \ . \tag{3.61}$$

This variant of an embedding vector uses uniformly distributed sampling points, whereas also non-uniform embeddings are possible [148, 149].

The prediction algorithm proceeds as follows: In each sampling instant t with $t \geq hp$ an embedding vector $D(t)$ is acquired. For accomplishing predictions, the current embedding vector $D(t)$ is compared to all previously collected embeddings $D(i)$ with $hp \leq i < t$ by determining the Euclidean distances $\delta(i) = \|D(t) - D(i)\|_2$. The M best matching embeddings (i.e. with smallest distances) $D(\tilde{t}_1)$, $D(\tilde{t}_2)$, $\dots$ $D(\tilde{t}_M)$ from the sampling instants $\tilde{t}_j$) with $1 \leq j \leq M$ are used to calculate a k-step prediction $\hat{y}(t+k|t)$:

$$\hat{y}(t+k|t) = \sum_{j=1}^{M} w_j y(\tilde{t}_j + k) \tag{3.62}$$

with the weights

$$w_j = \frac{1}{N} \frac{1}{\delta(\tilde{t}_j)} \tag{3.63}$$

and

$$N = \sum_{j=1}^{M} \frac{1}{\delta(\tilde{t}_j)} . \tag{3.64}$$

Because $y(\tilde{t}_j + k)$ is required to be present already, it is necessary to demand $\tilde{t}_j \leq t - k$ for $1 \leq j \leq M$. That is, depending on k eventually some $\tilde{t}_j$ need to be excluded if they do not fulfill this condition and the search needs to be continued until M valid samples have been acquired. To summarize, first the algorithm searches the points in time with the best matching embedding vectors. Then a k-step prediction is determined as a linear combination of the observations at these points in time advanced by k sampling instants. The rationality of this procedure lies in the assumption that points in time with similar embedding vectors lead to similar k-step predictions because the state of the underlying state-space model is also assumed to be similar. The weighting for each sampling instant depends on the Euclidean distance, so that samples with a larger distance to the current embedding vector have a smaller contribution to the prediction.

Generally, p defines the number of components in an embedding vector. The choice of p has to be seen in terms of condition (3.60), such that p needs to be at least twice as large as the order d of the underlying state-space model driving the breathing or tumor motion. If p is much larger than d it adds redundancy to the approach and causes unnecessarily high computational effort. If p is too small, the method may not be able to cover enough information of the signal's history because the order of the dynamics can be larger than $p/2$. However, d is unknown and therefore p will be treated as a parameter of this approach for which suitable values will be derived. h decides the time period between the components of an embedding vector. If h is too small, the components are almost identical and not much new information is gained. On the other hand, a large h might deliver uncorrelated data in the embeddings. According to the reconstruction theorem, the nominal value for h is one. However, like p, also h is given in terms of the sampling rate for which other values than the nominal ones might deliver better performance of the prediction method. Hence, h is also treated as a parameter. Furthermore, there is also the parameter M which decides on the number of best matching embeddings to be included into the prediction. The main purpose of M is to introduce an additional degree of freedom to be less prone to noisy data by averaging the effects of noise from different samples.

3.5.2 Class I Parameter Selection

In preliminary evaluations, the parameter ranges were initially set larger but then confined to a smaller region because performance gradually degraded with increasing parameter values. Also $p = 1$ was excluded which actually shows a severe performance degradation, complying with theory because p would need to be at least 2 in order to cover first-order dynamics. Finally, the parameter ranges were selected as follows: $p \in 2, 3, 5, 7, 9, 11$, h was selected from 1 to 11 in steps of 2 and M was chosen from 2 to 30 in steps of 2.

From the prediction equation (3.62), it is obvious that the prediction method is inherently stable (if the input signal remains bounded) because predictions are directly derived from linear combinations of previous samples without any feedback. Conversely, all previously discussed prediction methods adapted their parameters based on errors between predictions and observations, thus introducing feedback bearing potential of instabilities. In this respect, these methods behave like infinite impulse response filters whereas the Takens predictor can be associated with a finite impulse response filter. If one of the Euclidean distances in Eq. (3.63) or Eq. (3.64) is close to zero, then numerical problems can arise, effectively leading to instabilities. However, this can be easily accounted for in the algorithm by checking each distance against a small limit which should be a few orders of magnitude larger than the smallest representable number of the floating point unit, e.g. 10^{-10}. If the distance is smaller than the limit, then the prediction can be directly acquired from the corresponding sample alone. It was observed that this situation never occurred in practice so far, but in any case the algorithm should be protected against this unnecessary instability.

Similar to the previously discussed prediction methods, the average PR is analyzed at first. Since there are three parameters, a reduction of one dimension is necessary to enable visualization. When reducing the dimension along M by averaging, the influence of p and h can be seen from Figure 3.42. Compared to other prediction methods, the parameters generally have a small influence on the PR. Maximum average PR is achieved for small values of both p and h except for $p = 2$ and $h = 1$. With increasing prediction horizon, the quite small region of maximum PR shifts to slightly larger values for p and h, but always less than 5. The maximum average PR for $k \leq 11$ is always found for $p = 2$ with $h \in 1, 3, 5$. When reducing the dimension along h, the influence of the parameters M and p can be analyzed in Figure 3.43. As can be seen, the impact of M is relatively weak compared to h and p. However, there is a small performance decrease if M is too small. It can be inferred that M should be larger than the prediction horizon k. Optimal values for M increase approximately linearly with rising prediction horizon from 4 for $k = 1$ to 28 for $k = 11$. Consequently there is a certain minimum averaging necessary which depends on the prediction horizon in order to reach a more optimum performance in terms of the prediction ratio. The influence of p can be observed in the same way as in the previous figure. The parameters M, p and h have the same influence on the remaining prediction accuracy metrics.

The smoothness measure S_F is depicted representatively for the smoothness in Figure 3.44, showing the results when reducing the dimension along M. p has generally a stronger impact than h, in such a way that smoothness improves with increasing p and

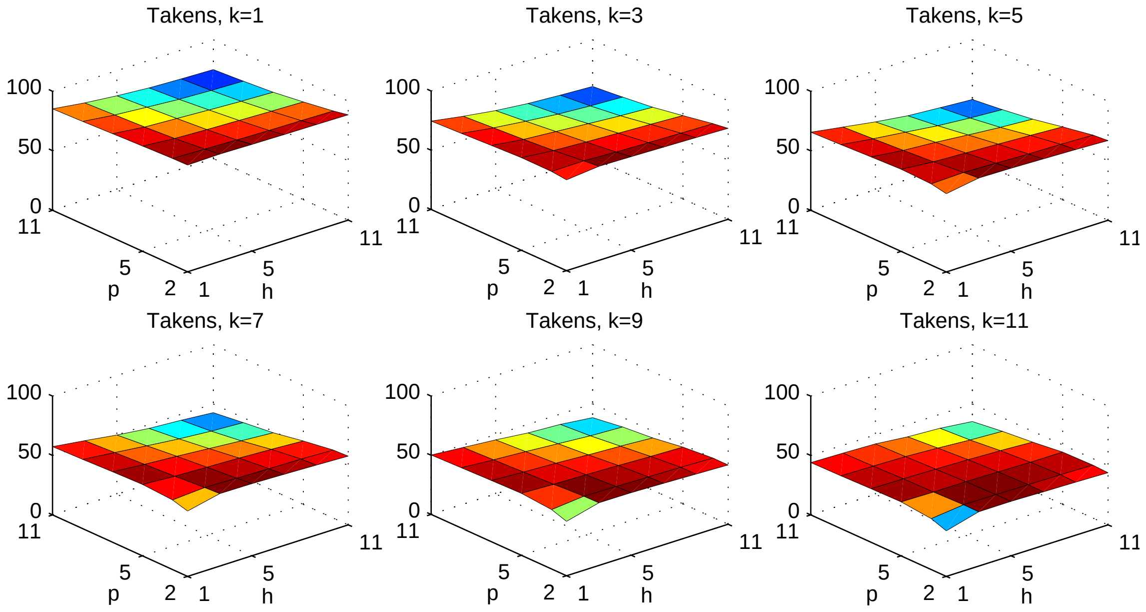


Figure 3.42: *Mean prediction ratio of 200 datasets of the Takens predictor in relation to parameters p and h*

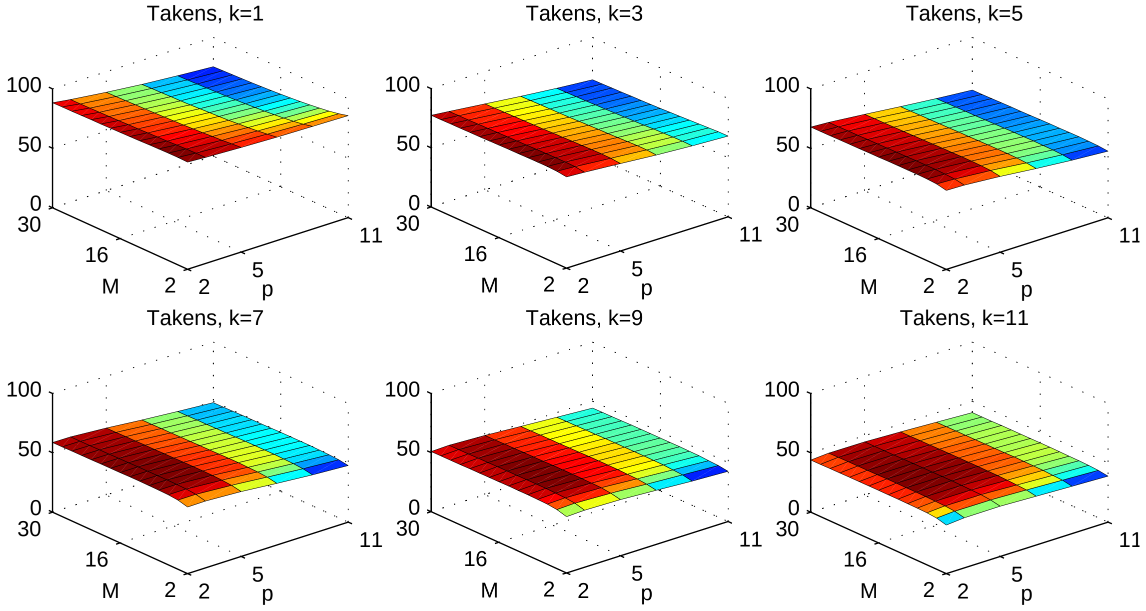


Figure 3.43: *Mean prediction ratio of 200 datasets of the Takens predictor in relation to parameters p and M*

slightly with increasing h. Since p is the dimension of the embedding vector, with increasing p, more past observations are taken into account. When these are compared to previous embedding vectors, a small Euclidean distance of a past embedding vector means that the current embedding vector fits well to the past one. If that is the case, it is more likely that predictions which are based on that will also fit better and will lead to smaller acceleration in the predicted signal and consequently to smoother predictions.

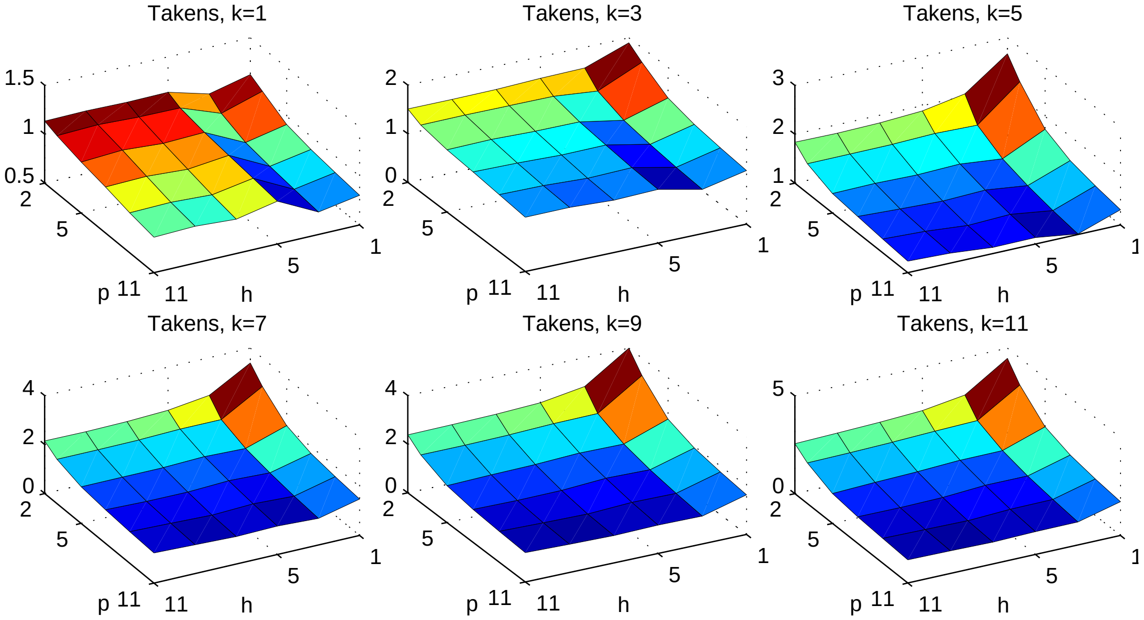


Figure 3.44: *Mean smoothness metric S_F of 200 datasets of the Takens predictor in relation to parameters p and h*

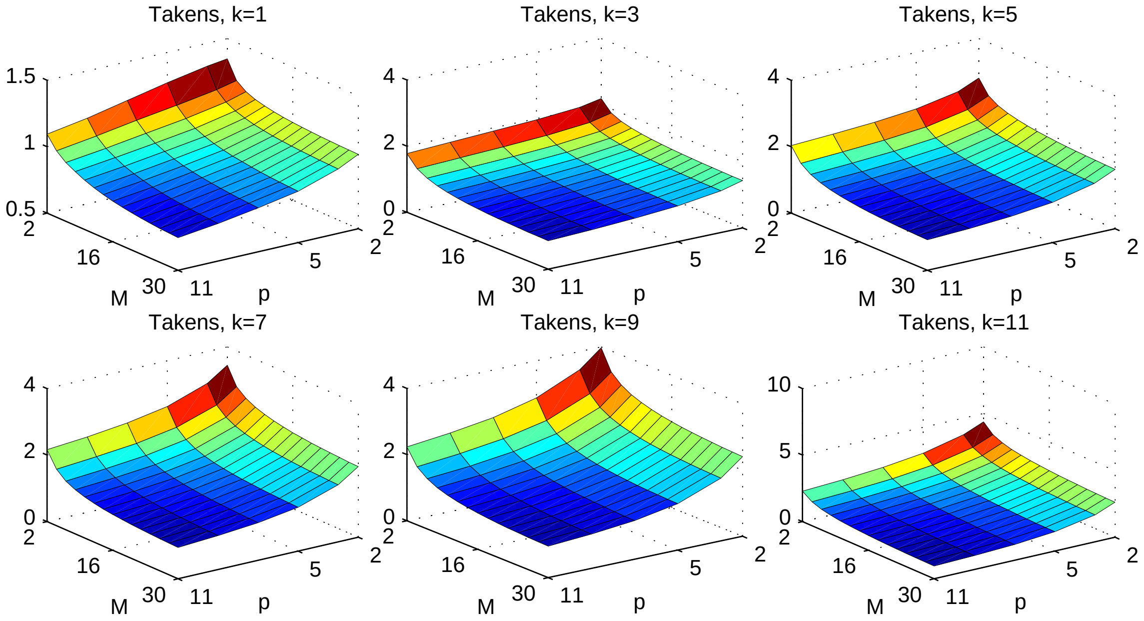


Figure 3.45: *Mean smoothness metric S_F of 200 datasets of the Takens predictor in relation to parameters p and M*

Small smoothness values are generally found for $h = 2$ especially for small prediction horizon. As can be inferred from Figure 3.45, showing the smoothness in terms of M and p, the parameter M has a stronger impact on smoothness than p. When increasing M, more samples enter the prediction equation (3.62) so that noise effects and differences in sample values average and lead to smoother predictions. Comparing the influences of M

and p on both smoothness and PR, it turns out that with these parameters, a trade-off between tracking accuracy and smoothness can be achieved. Since p has a stronger impact on the PR than M, first p should be chosen to achieve a good PR. Then, smoothness can be tuned using M without a severe loss of PR. Of course, smoother predictions can be realized if p is treated as a free parameter, but due to its influence on the prediction accuracy, this trade-off is required.

When determining the 15 parameter settings with maximum PR for each dataset and displaying the parameters' distribution, Figures 3.46 and 3.47 result, showing the distribution in terms of p and h as well as in terms of M and p (with reduction of the correspondingly missing dimension). The regions with the most frequent occurrences can be identified to be the similar to the ones which possess maximum average PR in Figures 3.42 and 3.43, demonstrating that the parameter setting chosen from maximum average PR generally also leads to good performance for each dataset. The regions with most frequent occurrences are found for small p and $h \leq 5$. For increasing prediction horizons, the landscape of occurrences flattens but leaving a peak at $p = 3$ and $h = 7$. Regions for M with maximum occurrences move from 2-12 for $k = 1$ to 18-30 for $k = 11$.

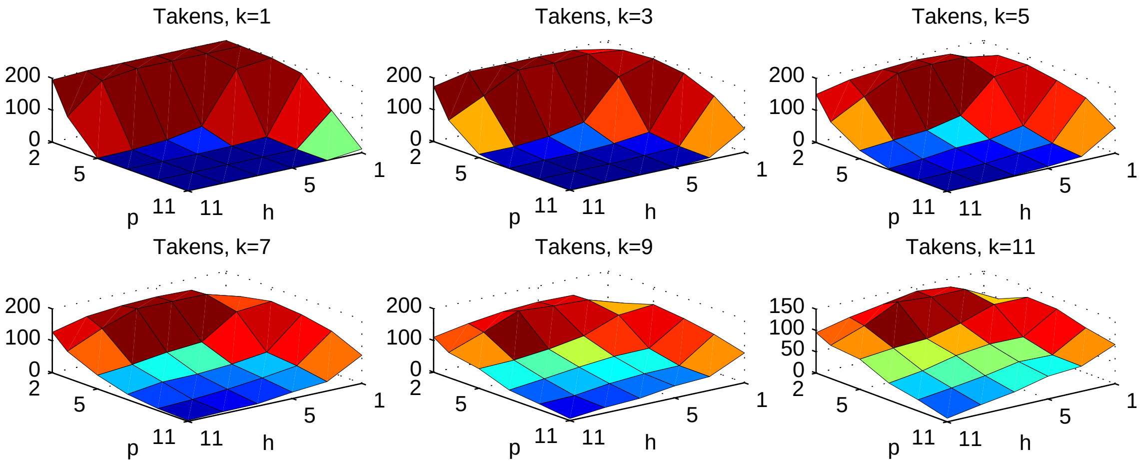


Figure 3.46: *Distribution of 15 parameter settings of the Takens predictor with largest PR for each dataset and accumulated over all datasets in relation to parameters p and h*

When choosing the parameter setting with maximum average PR for each k separately, the metrics given in Table 3.17 result for breathing motion. Comparing these to previously presented prediction methods, it turns out that there is a considerable improvement on all prediction metrics for the Takens predictor as opposed to the model-based approaches and the LMS algorithm. However, the RLS and ERLS algorithm perform slightly better in terms of prediction accuracy, but the performance difference is minor. When comparing the individual metric values to the results of the ERLS algorithm given in Table 3.13, the following small performance differences exist on the mean prediction accuracy metrics where the ERLS algorithm performs slightly better: difference of mean PR is less than 2.1%, difference of mean CI95 is less than 0.13mm, difference in mean RMS is less than 0.03mm and difference in mean MAX is less than 0.37mm. Given these differences, it is quite interesting to note that the MAE is generally smaller for the Takens predictor if $k \geq 5$. The difference in mean MAE then varies between 0.02mm and 0.04mm for

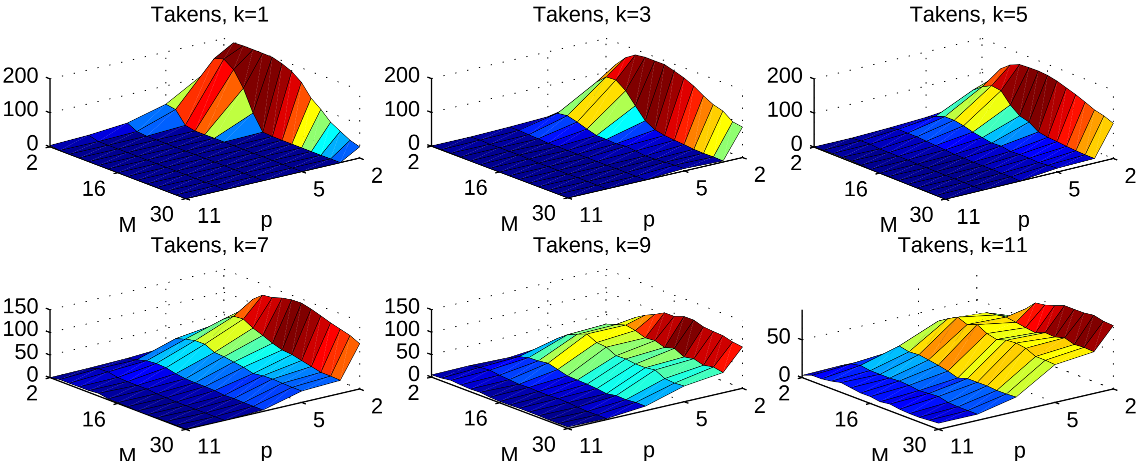


Figure 3.47: *Distribution of 15 parameter sSettings of the Takens predictor with largest PR for each dataset and accumulated over all datasets in relation to parameters p and M*

Prediction horizon	Type	PR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]	S_F	S_J	S_{TS}	S_{SNR}	CT25 [s]	CT10 [s]
1	min	79.6	0.01	0.00	0.01	0.03	0.34	1.97	0.07	1.64	8.75	8.75
	mean	91.1	0.22	0.07	0.11	0.66	1.27	2.48	0.13	16.20	8.77	9.33
	median	92.1	0.22	0.08	0.11	0.58	1.18	2.48	0.12	16.78	8.75	8.95
	max	95.9	0.84	0.20	0.32	3.62	3.24	2.97	0.26	23.68	9.65	22.10
3	min	57.2	0.03	0.01	0.02	0.06	0.58	2.06	0.07	1.23	8.75	8.75
	mean	79.8	0.53	0.17	0.26	1.44	1.58	2.53	0.13	14.76	9.05	12.83
	median	82.0	0.48	0.17	0.24	1.28	1.39	2.50	0.12	15.47	8.75	10.22
	max	91.1	1.99	0.56	0.92	5.23	4.18	3.07	0.27	21.83	12.85	57.35
5	min	39.6	0.06	0.02	0.03	0.10	0.76	2.17	0.08	1.05	8.75	8.75
	mean	70.5	0.82	0.25	0.38	1.92	2.01	2.63	0.15	13.29	9.59	16.66
	median	73.1	0.74	0.23	0.35	1.71	1.78	2.61	0.13	14.18	9.00	11.55
	max	88.8	2.70	0.83	1.31	6.29	5.48	3.16	0.33	20.10	31.05	107.95
7	min	28.7	0.08	0.03	0.04	0.14	0.89	2.27	0.08	-0.59	8.75	8.75
	mean	61.4	1.10	0.34	0.51	2.35	2.31	2.70	0.16	12.15	10.25	21.01
	median	63.8	1.00	0.32	0.47	2.13	2.07	2.67	0.15	12.89	9.30	13.70
	max	85.2	3.62	1.06	1.65	8.63	5.63	3.17	0.40	18.99	39.55	107.95
9	min	15.9	0.11	0.04	0.05	0.18	0.98	2.29	0.07	-3.40	8.75	8.75
	mean	53.4	1.37	0.43	0.63	2.73	2.24	2.66	0.15	11.71	10.03	21.29
	median	55.2	1.25	0.39	0.57	2.51	2.06	2.63	0.14	12.43	9.40	14.42
	max	81.9	4.57	1.29	1.97	10.98	5.25	3.15	0.37	19.84	24.95	107.95
11	min	8.1	0.12	0.04	0.06	0.19	1.01	2.34	0.08	-5.61	8.75	8.75
	mean	46.5	1.58	0.50	0.73	3.02	2.37	2.69	0.16	11.14	10.48	22.23
	median	48.5	1.42	0.46	0.65	2.81	2.18	2.67	0.14	11.75	9.50	15.67
	max	80.4	4.71	1.52	2.16	12.18	5.55	3.19	0.38	19.34	39.55	107.95

Table 3.17: *Minimum, mean, median and maximum breathing prediction metrics of parameter setting with largest average PR determined for each k separately; metrics are given for different k using the Takens predictor*

different k. There are also a few other minimum and maximum metrics for which the Takens predictor shows an improved performance: min. PR for $k \geq 3$, max. PR for $k = 5, 7, 11$ and max. MAX for $k \geq 5$. The improvement, especially on the max. MAX error, demonstrates that the Takens predictor can better handle difficult cases in which large prediction errors can be expected, especially for increasing prediction horizon. This can be partially attributed to the stable nature of the Takens predictor, which will always yield stable predictions, even for larger prediction horizons.

Looking at the remaining metrics for smoothness and convergence time, a considerable performance increase can be observed for the Takens predictor on most of these metrics (the only exception is S_F for $k \geq 7$). That is, the Takens predictor generally produces smoother measurements which additionally exhibit a faster convergence time (in terms of the CT25 and CT10 metrics). Additionally, from the influence of the tuning parameters p, h and M on the PR and smoothness depicted in Figures 3.42 – 3.45 it can be seen that the PR is rather insensitive to a change of these parameters. But it possesses a more profound impact on smoothness. Together with the fact that the metric values in Table 3.17 all resulted from $p = 2$ which is located at rather large smoothness values, a further improvement in smoothness can be achieved by increasing p without much loss in prediction accuracy. Hence, the Takens predictor provides more degrees of freedom to tune the predictions, allowing a reasonable trade-off between prediction accuracy and smoothness.

The metrics of the best achievable performance in terms of the PR (i.e. selecting the parameter setting with largest PR for each dataset and each k individually) are given in Table 3.18. As already estimated from the distribution of the 15 best parameter settings, the metrics compared to the case when choosing the parameter setting with largest average PR for each k individually (given in Table 3.17) are similar, however with a visible increase

Prediction horizon	Type	PR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]	S_F	S_J	S_{TS}	S_{SNR}	CT25 [s]	CT10 [s]
1	min	81.0	0.01	0.00	0.01	0.03	0.40	2.07	0.07	1.56	8.75	8.75
	mean	91.3	0.22	0.07	0.11	0.63	1.16	2.45	0.12	16.34	8.77	9.36
	median	92.3	0.22	0.08	0.11	0.58	1.06	2.44	0.12	16.78	8.75	8.90
	max	96.0	0.64	0.21	0.29	3.57	2.76	2.86	0.24	24.62	9.65	30.05
3	min	60.3	0.03	0.01	0.02	0.05	0.65	2.05	0.07	0.79	8.75	8.75
	mean	80.5	0.51	0.17	0.25	1.37	1.55	2.53	0.13	14.95	9.07	12.68
	median	82.3	0.48	0.16	0.24	1.23	1.38	2.50	0.12	15.55	8.75	10.33
	max	91.4	1.70	0.55	0.87	4.53	3.50	3.00	0.25	22.88	12.90	49.15
5	min	44.1	0.05	0.02	0.03	0.10	0.81	2.20	0.07	0.40	8.75	8.75
	mean	71.9	0.78	0.25	0.37	1.86	1.83	2.60	0.14	13.79	9.44	15.71
	median	73.8	0.70	0.23	0.34	1.63	1.66	2.59	0.13	14.75	9.00	11.55
	max	88.8	2.70	0.83	1.31	6.28	4.11	2.97	0.27	22.01	16.65	81.05
7	min	29.7	0.07	0.03	0.03	0.13	0.68	2.16	0.07	-0.17	8.75	8.75
	mean	63.7	1.04	0.33	0.49	2.25	1.99	2.63	0.15	13.03	9.77	18.62
	median	66.1	0.91	0.30	0.43	1.95	1.89	2.64	0.14	13.81	9.30	13.05
	max	86.6	3.70	1.06	1.65	8.70	4.76	3.05	0.31	21.84	17.95	71.20
9	min	18.5	0.08	0.03	0.04	0.12	0.63	2.10	0.07	-4.18	8.75	8.75
	mean	56.9	1.26	0.40	0.59	2.57	2.03	2.62	0.15	12.73	10.08	20.03
	median	58.5	1.07	0.36	0.51	2.28	1.93	2.64	0.14	13.31	9.47	13.62
	max	85.2	4.12	1.26	1.91	10.75	5.06	3.09	0.37	21.51	22.45	92.75
11	min	11.4	0.08	0.03	0.04	0.12	0.37	1.93	0.07	-4.10	8.75	8.75
	mean	51.2	1.44	0.46	0.67	2.80	1.97	2.60	0.15	12.78	10.43	19.62
	median	52.8	1.22	0.41	0.58	2.53	1.91	2.63	0.14	13.29	9.60	14.13
	max	84.3	4.37	1.42	2.04	12.56	4.56	3.02	0.35	21.77	28.65	92.75

Table 3.18: *Minimum, mean, median and maximum breathing prediction metrics of parameter setting with largest PR determined for each dataset and each k individually; metrics are given for different k using the Takens predictor*

in prediction accuracy. For example the increment in mean PR rises from 0.2% for $k = 1$ to 4.7% for $k = 11$, which is approximately comparable (though slightly better) to the increase observed for the RLS algorithm, which was 0.2%-3.2%. Comparing the results (Table 3.18) to the ERLS algorithm with the parameter setting chosen from average PR for each k individually (cf. Table 3.13), a performance increase can be observed for the Takens predictor in several prediction accuracy metrics (especially minimum and mean PR, a few MAX PRs, some CI95 values, most of the MAE values, all RMS values for $k \geq 9$) and in almost all smoothness and convergence time metrics. On the other hand, when comparing the best achievable performance between the Takens predictor and the RLS algorithm (c.f. Table 3.10), the improvements of the Takens predictor can be observed on almost the same metric values when comparing the parameter setting determined from average PRs for both methods.

3.5.3 Class II Extension

Another interesting feature of the Takens predictor is that it can naturally be modified to seamlessly perform tumor motion predictions based on breathing predictions. Assuming the correlation $l \mapsto m$ between breathing axis l and tumor axis m, the Takens predictor first determines the elements of the embedding vector from breathing samples p_l^B:

$$D(t) = \begin{bmatrix} p_l^B(t) & p_l^B(t-h) & \dots & p_l^B(t-h(p-1)) \end{bmatrix}^T ,$$

with which the Euclidean distances $\delta(i)$ are determined with $hp \leq i < t$, where t refers to the latest available interpolated tumor position information. The sampling instants $\tilde{t}_j$ of the M best matching embeddings with $1 \leq j \leq M$ (which are determined based on breathing axis l) are used in the prediction equation

$$\hat{p}_m^T(t+k|t) = \sum_{j=1}^{M} w_j p_m^T(\tilde{t}_j + k) ,$$

where the weights w_j are determined according to Eq. (3.63).

3.5.4 Class II Parameter Selection

For the following analysis, the gap between latest available tumor position and latest available breathing position (which is considered as the current sampling instant) is bridged by adapting the prediction horizon according to the gap size. Hence, the prediction horizon k represents the effective prediction horizon relative to the current sampling instant.

In the class II variant, the Takens predictor always yields stable predictions. As can be inferred from Figures 3.48 and 3.49, showing the influence of the parameters p and h as well as M and p on the average PR, the general influence of the parameters are the same as in the class I variant applied to breathing motion. However, the regions with maximum PR are found at slightly larger values for all of M, p and h. For all depicted prediction horizons, the maximum PR in terms of p and h can be found for $p = 3$ and $h = 7$, whereas optimal values for M vary between 22 to 30, increasing with growing prediction horizon. This is consistent with the trend already discovered in the class I formulation, where M increased with k. Since the effective prediction horizon in the class

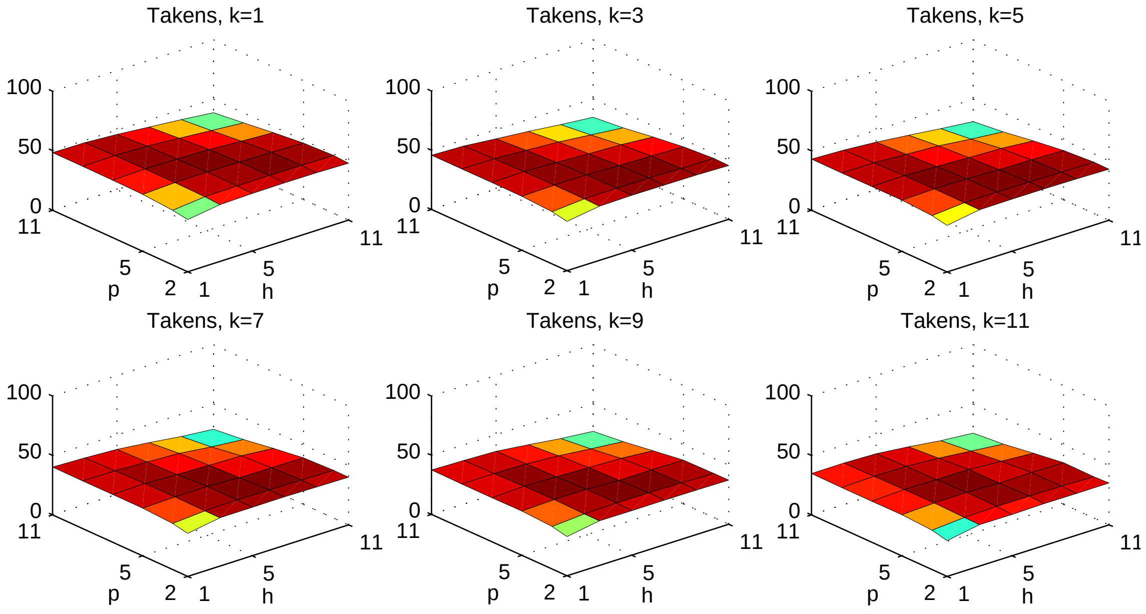


Figure 3.48: *Mean tumor prediction ratio of 200 datasets of the Takens predictor in relation to parameters* p *and* h

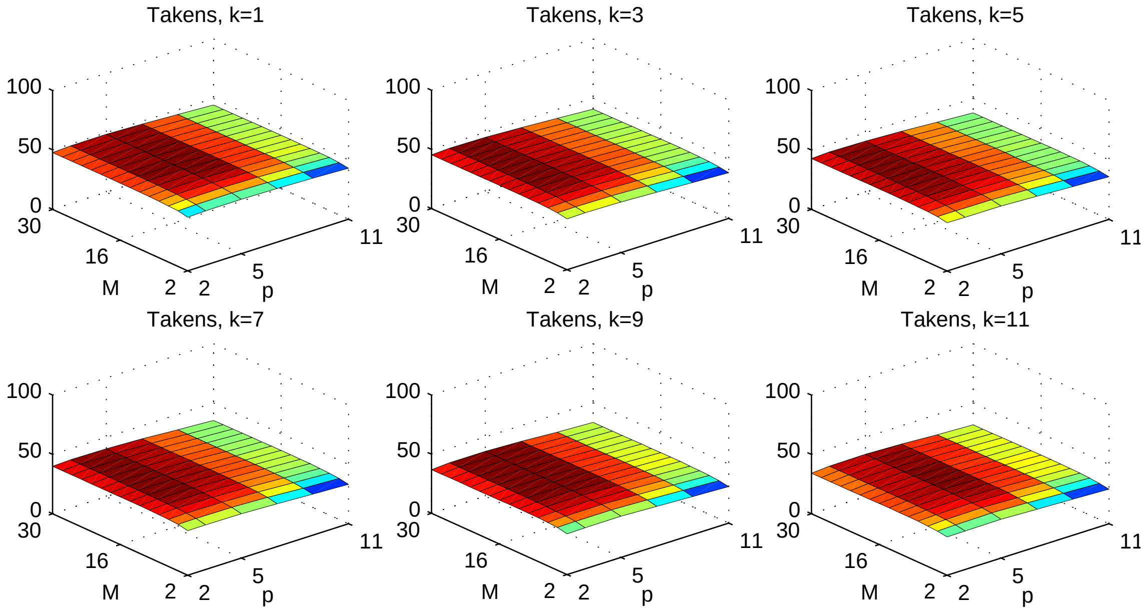


Figure 3.49: *Mean tumor prediction ratio of 200 datasets of the Takens predictor in relation to parameters* M *and* p

II formulation is always larger than the prediction horizon relative to the current sampling instant, M should consequently be larger in the class II formulation.

The parameters' influence on the remaining prediction accuracy metrics is the same as on the PR. Similarly, the influence on smoothness and convergence time metrics is analogous to the class I formulation.

When choosing the mentioned parameter settings leading to the maximum average PR, the metrics in Table 3.19 result for tumor motion predictions. Comparing these to the results of the class III LMS and RLS predictors (cf. Tables 3.7 and 3.11), an obvious performance increase can be observed for the Takens predictor on almost all prediction accuracy metrics and on all convergence time metrics. The increment in PR ranges between 7.9% to 14.1%. However, the class II Takens predictor generally delivers less smooth predictions. When comparing with the class II LMS ARX predictor (cf. Table 3.8), a different picture emerges. For small prediction horizons, the Takens predictor results in less prediction accuracy whereas for larger prediction horizons, the Takens predictor generally performs better, especially on the mean, median and max. PR metric. However, on most values of MAX, the LMS ARX predictor shows smaller maximum errors even for larger k. On the other hand, the class II Takens predictor leads to smoother predictions with generally lesser convergence time. Comparing the class II Takens predictor to the RLS ARX predictor with the metrics given in Table 3.12, the latter performs better on almost all prediction accuracy metrics (except min. PR for $k = 9, 11$) as well as on almost all smoothness metrics. The increment in the mean PR moves in between 11.2% and 16.0%, where the increment becomes smaller with increasing prediction horizon. Despite these improvements for the RLS ARX predictor, the Takens predictor possesses a smaller convergence time, especially for increasing prediction horizons. From the trend that the performance increase becomes smaller as the prediction horizons becomes larger, it can be estimated that for some value of the prediction horizon, the Takens predictor will deliver approximately similar prediction accuracy. This can be attributed to the stable nature of

Prediction horizon	Type	PR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]	S_F	S_J	S_{TS}	S_{SNR}	CT25 [s]	CT10 [s]
1	min	-14.7	0.30	0.07	0.13	0.51	0.55	1.97	0.06	4.05	8.75	8.75
	mean	48.7	1.73	0.60	0.84	3.44	1.62	2.44	0.13	14.04	10.19	16.05
	median	52.2	1.34	0.48	0.66	2.26	1.35	2.39	0.12	14.45	9.33	11.85
	max	82.9	14.15	4.01	5.90	18.60	4.91	3.00	0.26	21.69	29.80	55.85
3	min	-17.8	0.30	0.07	0.13	0.51	0.57	1.93	0.06	3.84	8.75	8.75
	mean	45.8	1.84	0.63	0.88	3.55	1.71	2.46	0.13	13.62	10.39	16.86
	median	48.3	1.43	0.51	0.72	2.40	1.49	2.42	0.12	13.97	9.53	12.30
	max	82.4	14.86	4.23	6.12	20.45	5.20	3.02	0.26	21.15	29.80	55.85
5	min	-19.2	0.30	0.07	0.13	0.54	0.64	1.99	0.06	4.07	8.75	8.75
	mean	43.0	1.96	0.67	0.93	3.67	1.79	2.48	0.13	13.29	10.59	17.57
	median	45.6	1.57	0.54	0.76	2.66	1.62	2.45	0.12	13.82	9.65	12.67
	max	81.3	15.77	4.41	6.27	21.14	4.93	3.00	0.26	20.50	29.80	55.85
7	min	-22.2	0.32	0.08	0.13	0.52	0.73	2.09	0.05	2.36	8.75	8.75
	mean	40.2	2.08	0.71	0.98	3.84	1.91	2.51	0.14	12.67	11.06	18.71
	median	41.1	1.62	0.57	0.79	2.77	1.77	2.50	0.12	13.04	9.65	13.03
	max	78.8	14.90	4.44	6.32	22.13	5.00	3.03	0.26	20.38	50.60	91.95
9	min	-25.5	0.32	0.08	0.13	0.65	0.67	2.03	0.06	4.59	8.75	8.75
	mean	37.3	2.19	0.75	1.03	3.95	1.86	2.50	0.13	12.96	10.89	18.82
	median	39.0	1.73	0.61	0.86	2.93	1.73	2.49	0.12	13.34	9.75	12.92
	max	76.8	15.67	4.79	6.56	22.51	5.07	3.02	0.28	19.93	29.80	91.95
11	min	-30.8	0.32	0.08	0.13	0.61	0.50	1.89	0.05	4.95	8.75	8.75
	mean	34.3	2.32	0.77	1.07	4.03	1.65	2.45	0.13	13.31	10.92	17.56
	median	35.6	1.87	0.67	0.90	2.97	1.58	2.46	0.12	13.36	9.72	12.10
	max	76.7	15.79	4.65	6.55	23.49	3.85	2.88	0.23	20.68	29.80	55.85

Table 3.19: *Minimum, mean, median and maximum tumor prediction metrics of parameter setting with lowest PR applied to all datasets for different k using the Takens predictor*

the Takens predictor, whereas the RLS ARX predictor is challenged by more instabilities with increasing prediction horizon.

3.6 Nonlinearly extended Recursive Least-Squares Algorithm

Essentially the Takens predictor performs a pattern matching by comparing the latest pattern, i.e. an embedding vector, to previously acquired patterns from the signal's past. A prediction is then generated using the best matching patterns by a linear combination of corresponding sample values. When inspecting individual datasets in terms of the prediction error when applying the RLS or ERLS predictor, a certain pattern arises in such a way that the evolution of the errors approximately follows the breathing motion. That is, prediction errors from a certain breathing phase are sometimes similar to the same breathing phase of another breathing cycle. With this in mind, the ERLS can be improved by estimating the future prediction error based on past prediction errors, and correcting for them during current predictions. The Takens predictor is especially useful for this purpose as it can estimate the prediction error based on best matching breathing patterns.

Therefore, the fusion of the ERLS predictor and the Takens predictor is proposed in the following way for the class I formulation, resulting in the nonlinearly extended recursive least-squares (NERLS) predictor: At first the ERLS predictor performs multi-step predictions up to the prediction horizon k according to Eqn. (3.35), (3.43), (3.44) and (3.47). The predictions of the ERLS predictor will further be noted by $\hat{y}_{RLS}(t+k|t)$. In each sampling instant $t > k$, the prediction errors

$$e_k(t) = \hat{y}_{RLS}(t|t-k) - y(t)$$

are determined for each k from the k-step prediction for the current sampling instant (performed k sampling instants before) and the latest observation $y(t)$. Then, the Takens predictor is used to estimate the prediction error as follows. Since the prediction error depends on the breathing phase, the embedding vector $D(t)$ is acquired from $y(t)$ according to Eq. (3.61). Also the determination of the M best matching embeddings according to the Euclidean distances $\delta_(i)$ remains unchanged. However, the estimates for the prediction error are now derived from past prediction errors $e_k(t)$:

$$e_k(t+k|t) = \sum_{j=1}^{M} w_j e_k(\tilde{t}_j + k) \ ,$$

where the weights w_j are given by Eqn. (3.63) and (3.64). Since the prediction errors differ with respect to k, a separate error estimate is calculated for each k. This is removed from RLS predictions to produce the final predictions

$$\hat{y}(t+k|t) = \hat{y}_{RLS}(t+k|t) - \gamma e_k(t+k|t) \ .$$

The parameter $\gamma \in [0; 1]$ is introduced to adjust the contribution of the error estimate on the final predictions due to uncertainties in the error estimation. If $\gamma = 0$, the presented approach reduces to the ERLS algorithm.

When performing simulations over the whole parameter range of γ, it is possible to determine if the approach improves upon the RLS algorithm if $\gamma > 0$ is found to result in maximized PRs. Apart from γ, simulations were extended to the parameters for the error estimator, i.e. M, p and h. Since the ERLS predictor is included without modification, the parameter setting for the ERLS algorithm was selected to lead to maximized average PRs. For the variable parameters of the NERLS predictor it was found that maximized average PRs are reached with $M = 15$ for all inspected prediction horizons, whereas p and h range from 5 to 9. However, the product of p and h varies in between 25 and 45, depending on the prediction horizon. Values for γ leading to maximized average PR are found in between 0.4 and 0.5, hence resulting in an improvement of the NERLS approach over the ERLS predictor. The corresponding breathing metrics are given in Table 3.20. Comparing these results to the ones of the ERLS predictor summarized in Table 3.9, a performance increase can be observed on almost all prediction accuracy metrics except for some max. CI95, max. MAE and mean and median MAX values with only a small difference in between. The mean PR improves by 0.1%-1.6%, where the increase becomes larger for rising prediction horizons. While smoothness slightly degrades except for most of the maximum smoothness metrics, a clear benefit is observed for convergence time.

Prediction horizon	Type	PR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]	S_F	S_J	S_{TS}	S_{SNR}	CT25 [s]	CT10 [s]
1	min	81.9	0.01	0.00	0.00	0.02	0.39	1.98	0.07	1.81	8.75	8.75
	mean	93.2	0.16	0.06	0.08	0.49	1.25	2.51	0.13	15.95	8.76	9.14
	median	94.0	0.16	0.06	0.08	0.43	1.16	2.52	0.12	16.54	8.75	8.75
	max	98.0	0.36	0.12	0.17	2.30	3.01	2.92	0.32	23.59	9.70	17.50
3	min	51.4	0.02	0.01	0.01	0.04	0.69	2.15	0.08	1.01	8.75	8.75
	mean	82.6	0.44	0.16	0.22	1.07	1.69	2.60	0.15	14.05	9.07	14.10
	median	84.4	0.41	0.15	0.20	0.91	1.55	2.60	0.13	14.61	8.75	10.80
	max	92.7	1.09	0.44	0.56	4.65	3.40	2.96	0.40	22.19	13.45	57.35
5	min	34.7	0.04	0.02	0.02	0.06	0.73	2.24	0.09	1.28	8.75	8.75
	mean	73.5	0.70	0.25	0.35	1.58	1.97	2.67	0.17	12.84	9.85	19.20
	median	77.1	0.62	0.23	0.31	1.34	1.83	2.66	0.14	13.59	9.25	13.70
	max	88.9	2.02	0.80	1.04	8.24	4.67	3.03	0.39	20.39	29.75	90.05
7	min	16.0	0.05	0.02	0.03	0.09	0.73	2.31	0.09	1.33	8.75	8.75
	mean	64.3	0.98	0.35	0.48	2.07	2.07	2.72	0.18	12.10	10.87	24.27
	median	69.8	0.86	0.32	0.42	1.78	1.88	2.70	0.16	13.12	9.83	18.72
	max	85.9	2.78	1.19	1.53	11.84	4.83	3.08	0.38	19.25	29.85	104.20
9	min	-1.7	0.07	0.03	0.03	0.12	0.72	2.28	0.10	1.22	8.75	8.75
	mean	56.1	1.24	0.44	0.60	2.49	2.04	2.71	0.19	11.81	11.48	27.12
	median	61.9	1.08	0.38	0.52	2.15	1.92	2.70	0.17	12.85	10.45	23.45
	max	83.1	3.67	1.48	1.92	13.83	4.59	3.06	0.36	18.49	28.20	99.55
11	min	-7.4	0.08	0.03	0.04	0.12	0.72	2.25	0.10	3.11	8.75	8.75
	mean	49.8	1.44	0.51	0.69	2.80	2.02	2.70	0.19	11.84	11.63	30.11
	median	55.1	1.26	0.45	0.61	2.44	1.93	2.70	0.17	12.34	10.70	27.55
	max	81.1	4.51	1.77	2.28	15.96	4.38	3.03	0.35	19.71	25.05	105.60

Table 3.20: *Minimum, mean, median and maximum breathing prediction metrics of parameter setting with largest average PR determined for each k separately; metrics are given for different k using the nonlinearly ERLS predictor*

The class II RLS ARX predictor can also be extended by the error estimator based on the Takens predictor in a similar way. Since in the class II variant a complete set of breathing predictions is performed first, the NERLS approach can be adopted in the same way in order to deliver breathing predictions which are corrected by breathing prediction error estimates. These breathing predictions are then fed as external input into an ARX model, where its parameters are also updated by an RLS algorithm. The ARX model delivers tumor predictions for which a separate instance of the error estimator is additionally incorporated, estimating the tumor prediction error. Finally, the tumor prediction error is removed from the tumor predictions of the RLS ARX predictor. The achieved results when compared with the metrics of the RLS ARX predictor given in Table 3.12 are similar to the ones of the NERLS predictor applied to breathing motion: the PR and most of the prediction accuracy and convergence time metrics show a slight improvement, while the smoothness drops a little.

To summarize, the NERLS approach improves upon the ERLS algorithm in terms of prediction accuracy and convergence time of breathing predictions, where the ERLS algorithm has so far delivered the most precise predictions among all discussed predictors. A similar improvement can also be achieved for tumor predictions by employing an additional tumor prediction error estimator based on Takens predictor.

3.7 Comparison of Prediction Algorithms

While the analysis of previously discussed prediction methods was based on quantitative measures, it is worthwhile to extend the analysis also in a qualitative manner as this can reveal general characteristics of predictions which cannot be extracted from the metrics. Therefore two patient datasets were selected for which 3D breathing predictions are shown in the following for each prediction method (except the LMS predictor). The first dataset of patient 1, depicted in Figure A.1, is a rather regular one, whereas the second dataset of patient 40, visualized in Figure A.4, contains some irregularities in the form of baseline shifts and amplitude variations. For each prediction method, the parameter setting was selected as the one maximizing the average PR for the displayed prediction horizons, i.e. one of them which was used in the tables giving the quantitative metrics.

Figures 3.50 – 3.51 show the predictions of the plug-in ERLS predictor for patient 1 and 40, respectively. Each figure contains the 3D breathing predictions while for each axis the

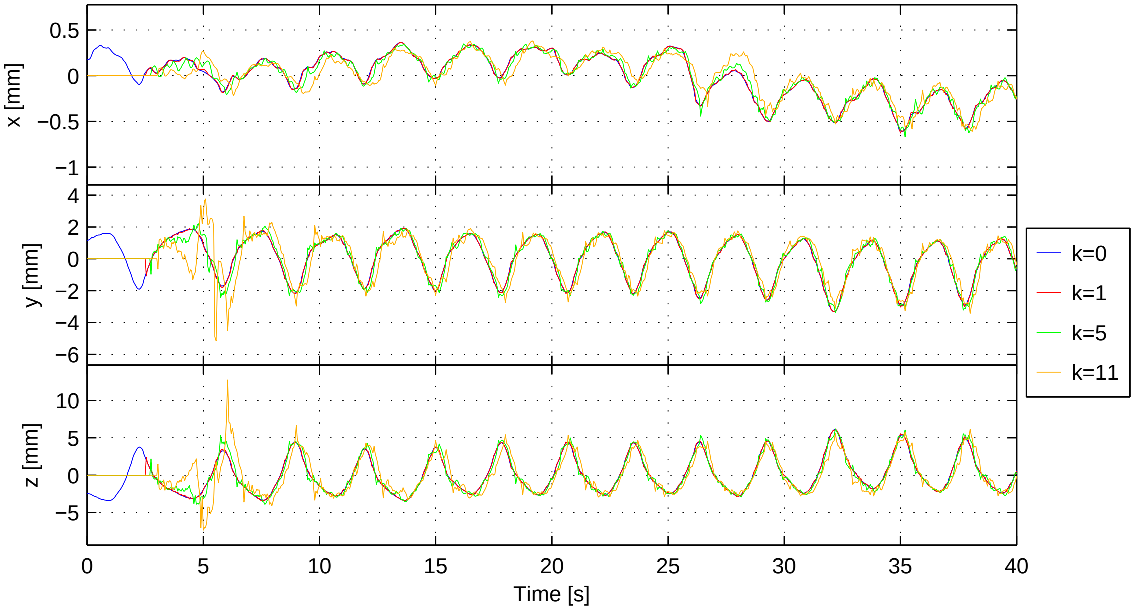


Figure 3.50: *Multi-step predictions of plug-in ERLS predictor for patient 1*

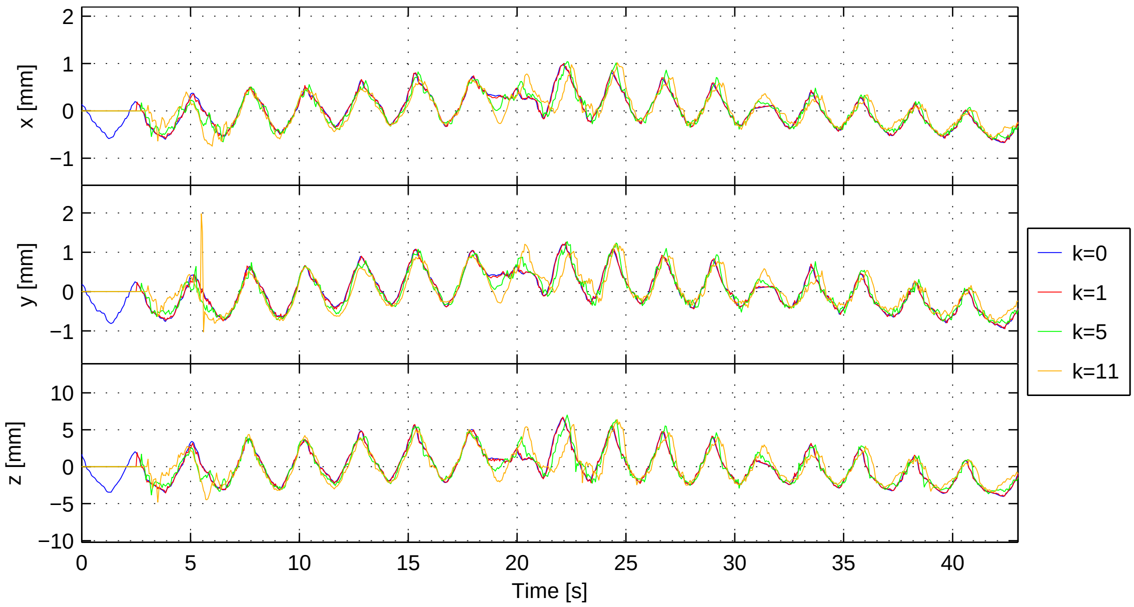


Figure 3.51: *Multi-step predictions of plug-in ERLS predictor for patient 40*

one-, five- and eleven-step predictions (displayed in red, green and orange color, respectively) are overlaid on the true signal in blue color. Additionally, each figure shows the complete predictions as time progresses including the dead time for which no predictions can be performed (prediction values are set to zero during this time) and the transition phase during which the predictions converge.

The dead time of the ERLS predictor is clearly visible in the figures and depends on the order n_a of the underlying AR model in addition to the prediction horizon. The predictions in the figures were gained with $n_a = 50$, corresponding to a dead time of 2.5s plus the prediction horizon. It can be observed that the one-step prediction converges within one sampling instant to the true signal along all axes and stays close to it, even during the irregularities of patient 40 at time 20s and 32s. With increasing prediction horizon, the transition phase prolongs which can be seen by larger deviations from the true signal. Especially for patient 1, temporary deviations extend by more than one amplitude for several seconds. It is also interesting to observe the behavior during irregularities of patient 40. While the one-step predictions are close to the true signal, deviations become larger with increasing prediction horizon. The eleven-step predictions are shaped in a way that would be expected if the signal had continued regularly as before. After the irregularities, deviations slowly converge again in a time frame of several seconds, depending on the severeness of the irregularity. This is illustrated by the second irregularity for which the transition phase is much shorter than for the first one which is longer in duration. In terms of smoothness, it can be seen that predictions become less smooth as the prediction horizon increases which is supported by the smoothness metrics given in Table 3.13. When comparing both patients, it turns out that patient 1 can be predicted more accurately than patient 40 because patient 1 possesses a more regular breathing pattern whereas patient 40 shows more variability.

The predictions of the Kalman CV predictor presented in Figures 3.52 and 3.53 are quite different from the ones of the ERLS predictor. While the one-step predictions are

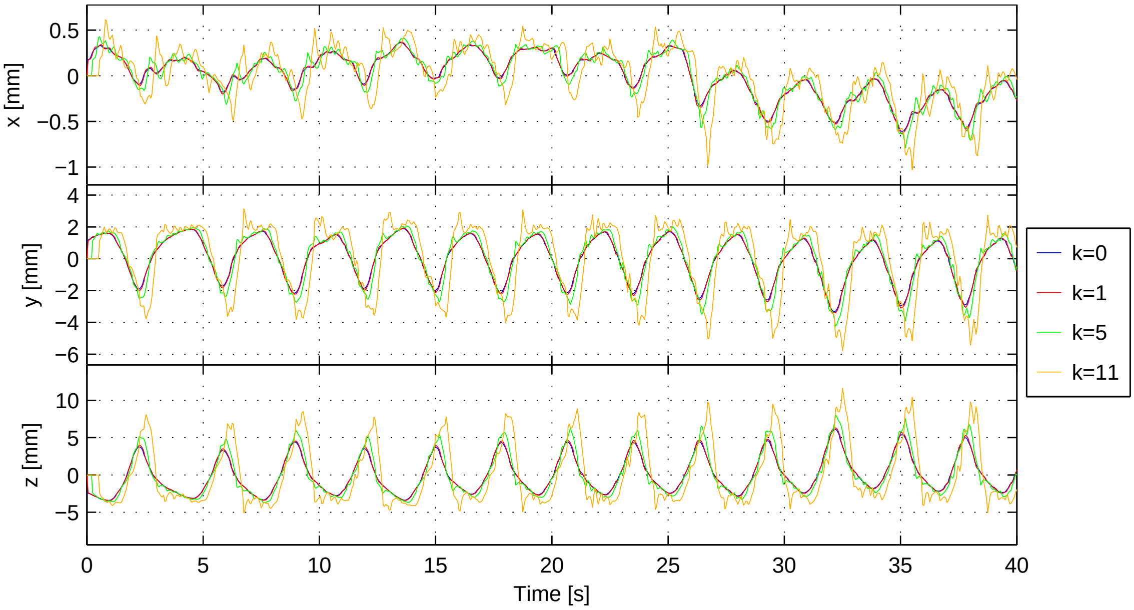


Figure 3.52: *Multi-step predictions of Kalman CV predictor for patient 1*

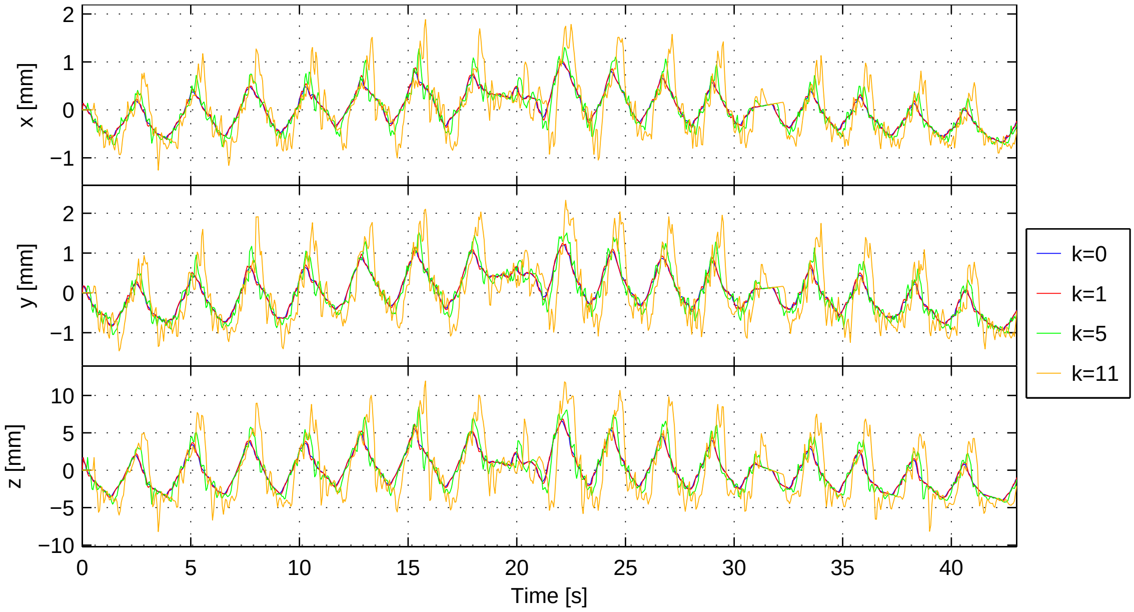


Figure 3.53: *Multi-step predictions of Kalman CV predictor for patient 40*

still close to the true signal, larger deviations are found for increasing prediction horizons. The largest deviations occur during the peaks of breathing motion which originate from the assumption of constant velocity. In the vicinity of the peaks, the signal dynamics become nonlinear, thus violating the linear assumption. Consequently, deviations are larger than for the RLS predictor. Additionally, the predictions of patient 40 are less smooth which can be attributed to a suboptimal setting of the process noise variance S, which needs to account for the acceleration present in the true signal. Hence, the Kalman CV predictor is sensitive to the choice of the process noise, where the optimal setting may vary depending on the dynamics of the breathing pattern. However, a clear advantage of the method is that all multi-step predictions can be performed as soon as a first sample is available, a feature which is not found in other methods. Corresponding predictions for the Kalman CA predictor are similar, hence their display is omitted.

During the comparison of metrics between the Kalman CV predictor and the EKF with spring-mass model, a performance increase was observed, which is exemplified by inspecting Figures 3.54 and 3.55, showing the predictions of EKF predictor. It can be seen that

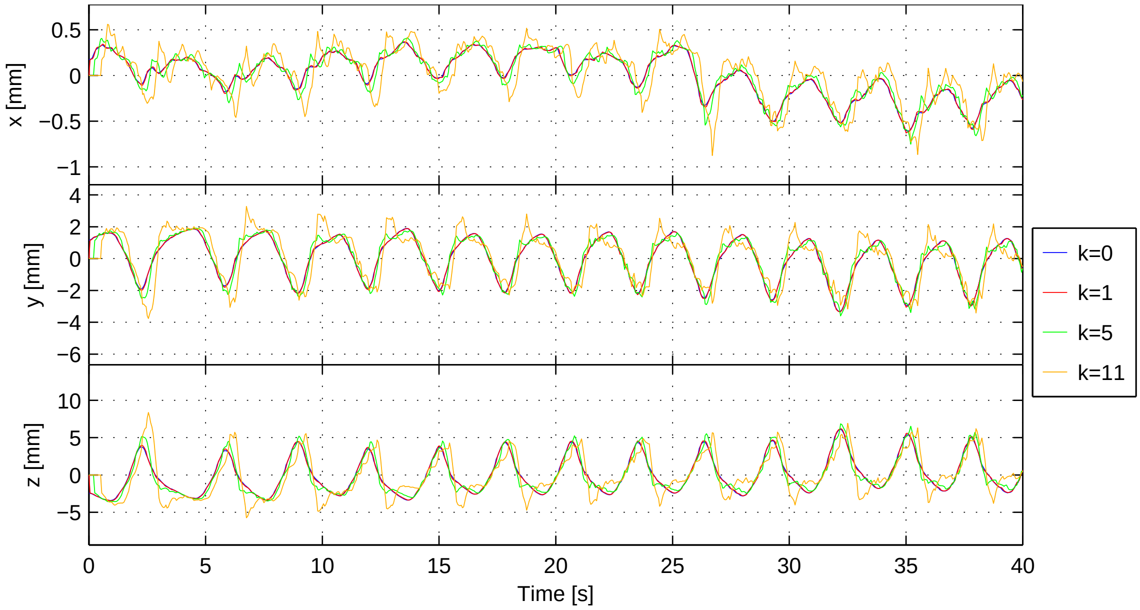


Figure 3.54: *Multi-step predictions of EKF predictor for patient 1*

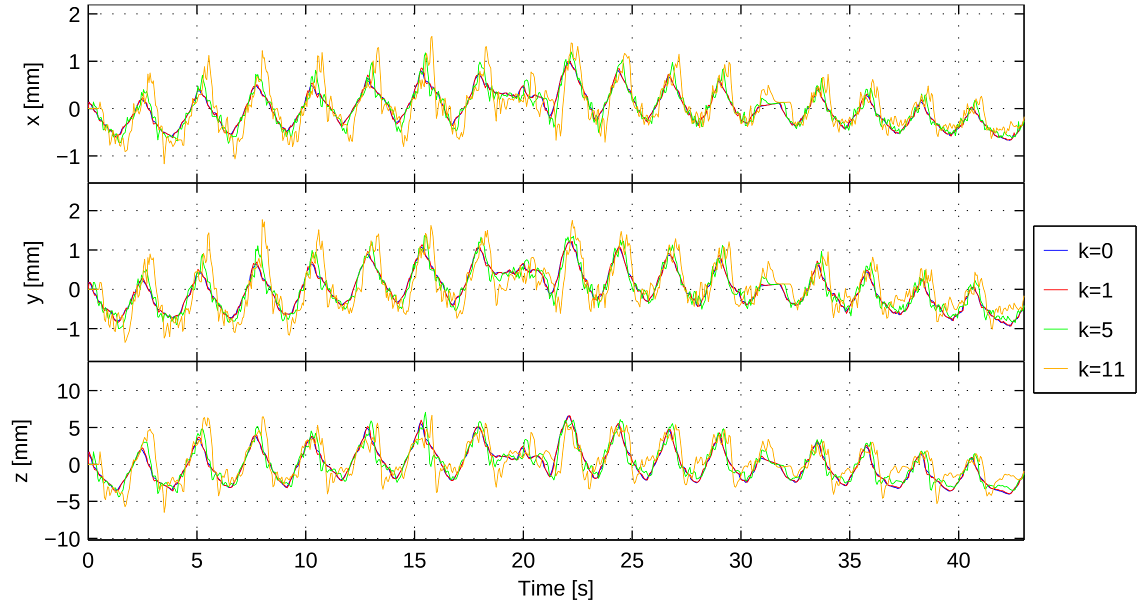


Figure 3.55: *Multi-step predictions of EKF predictor for patient 40*

predictions are closer to the true signal compared to the Kalman CV or CA predictor, especially during the peaks for smaller prediction horizons. Due to the underlying nonlinear model, which is linearized to the latest past of the signal, the predictor can deal better with the nonlinear effects in the vicinity of peaks. However, as the prediction horizon increases, linearization does not hold anymore, leading to larger deviations. Nevertheless, performance is generally better than the Kalman CV predictor but less than the RLS predictor.

The multi-step predictions of the Takens predictor are shown in Figures 3.56 and 3.57. The dead time depends on the product of the parameters h and p. Since moderate values of h and p were identified as optimal ones, it can usually be kept rather short as opposed to the LMS or RLS predictors. However, at least one complete breathing cycle is necessary, for patient 40 even two to three, in order to get reliable predictions. Nevertheless, prediction amplitudes stay constrained during the transition phase as opposed to the RLS predictor. Compared to the previously discussed predictors, the predictions are smoother even for larger prediction horizons, which was also observed on the quantitative metrics. Unlike other predictors, sometimes one-step predictions deviate from the true signal, for example during the peak at time 32s for patient 1 on the y- and z-axis. Since the predictor searches for a breathing pattern (the embedding vector) in the signal's past and performs a linear combination of the best matching embeddings to generate a prediction, it can only predict something which already occurred in the past. When comparing the height of this peak to the previous ones present in the signal, it can be seen that the amplitude was never reached in the signal before. Hence, the resulting predictions deviate from the true signal. This is also observable for patient 40 where the variability challenges the predictor. Although predictions are relatively close to the true values, the variability of breathing patterns results in less smoothness compared to patient 1. Due to the pattern matching principle of the Takens predictor, difficulties arise during the irregularities of patient 40. Because of the inherent stability of the predictor, the predictions stay constrained but oscillating

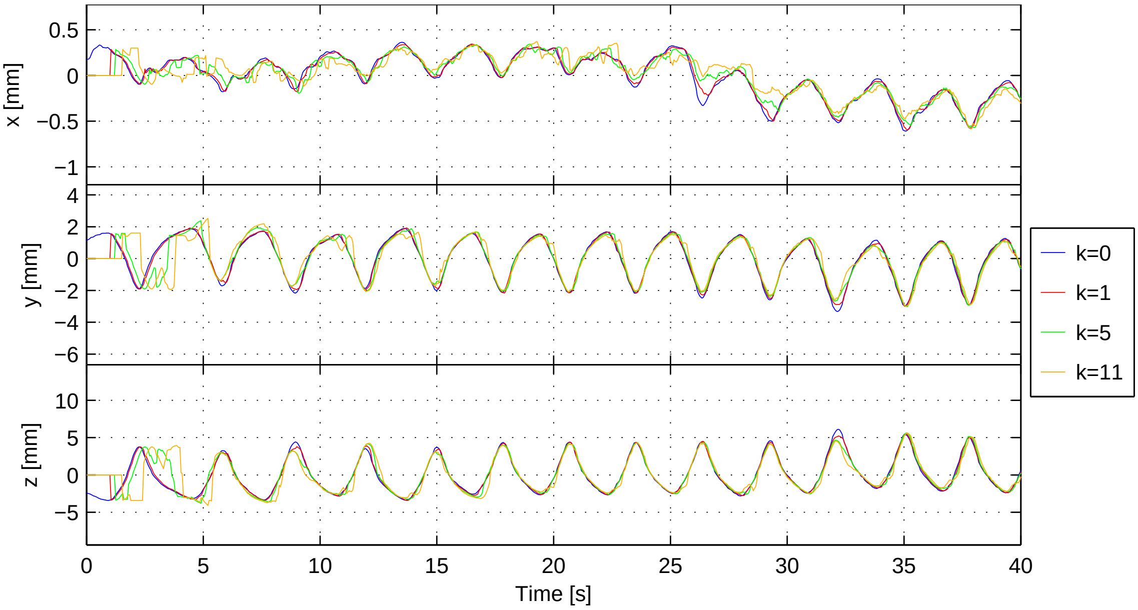


Figure 3.56: *Multi-step predictions of Takens predictor for patient 1*

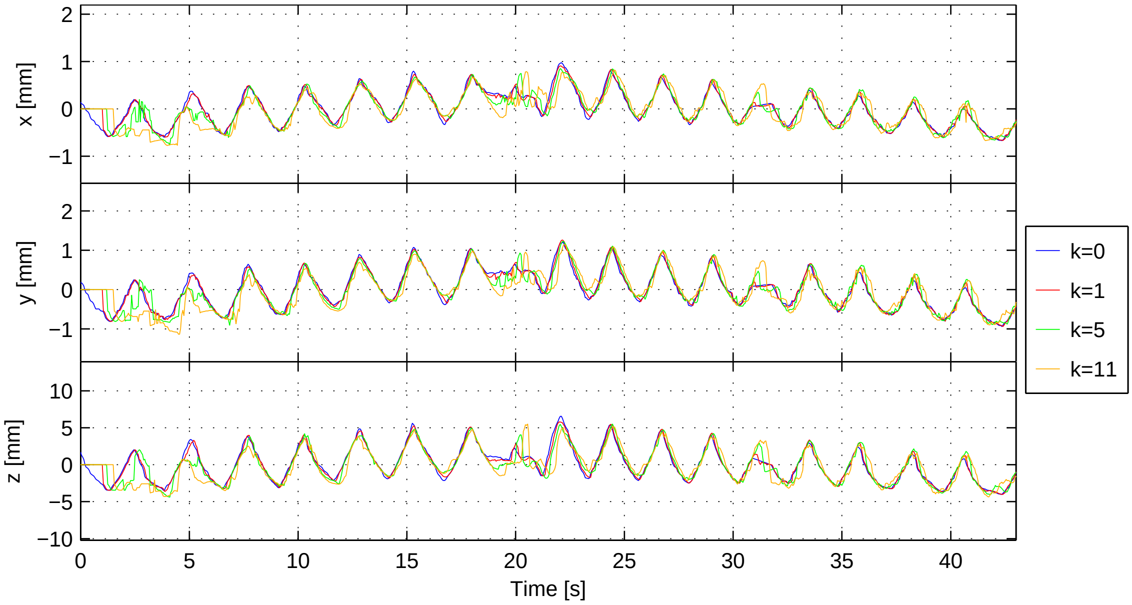


Figure 3.57: *Multi-step predictions of Takens predictor for patient 40*

around the true value. However, an advantage of this predictor when compared to the others is the short convergence time after irregularities which result from the pattern matching principle, in which no parameters need to converge again.

The performance benefit of the NERLS predictor is rather difficult to estimate from the figures, therefore they are omitted. The resulting figures are very similar to the ones of the ERLS predictor, where the predictions are slightly closer to the true signal during some phases of the breathing motion.

Table 3.21 summarizes the most important metrics of the various prediction methods in a qualitative way. An assessment of the predictor's accuracy is given which is based on the mean PRs. Furthermore, the smoothness is relatively categorized based on the mean smoothness metrics. The dead time is given in multiples of sampling instants and in terms of parameters of the corresponding prediction methods. The transition phase is a qualita-

Prediction method	Accuracy (PR)	Smooth-ness	Dead time	Transition phase	Convergence time CT10	Computation time [µs]
LMS	−	++	n_a	+/−	+	52.398±3.886
RLS	+	+	n_a	−−	+	1946.079±28.582
ERLS	+	+	n_a	−−	+	1946.079±28.582
Kalman CV	−−	−−	0	+	−−	45.515±3.664
Kalman CA	−−	−−	0	+	−−	48.169±3.315
Kalman EKF	−	−−	0	+	−−	48.361±3.532
Takens	+	++	$h \cdot p$	−	++	326.121±98.698 increase by 9.808µs/s
NERLS	++	+	n_a	−−	+	2309.237±125.496 increase by 9.893µs/s

Table 3.21: *Qualitative comparison of class I prediction methods*

tive assessment on how the predictions are shaped after the dead time until the predictions have converged, i.e. how reliable they are, which can be derived from Figures 3.50 – 3.57. The mean convergence time metric CT10 is used to assess the convergence time additionally to how predictions converge during the transition phase. The computation time was measured using the C++ implementation of each prediction method on an Intel® Pentium 4 processor with 3.0GHz running the hard real-time framework Xenomai as described in Section 2.2.2.6. The computation time is given as average time in microseconds of a calculation cycle during one sampling instant and its standard deviation. For each method, the parameter setting was chosen from the ones with maximum average PR, which leads to the largest computation time. So the computation time can be regarded to be close to the worst case. All prediction methods were configured to perform multi-step predictions in each sampling instant with a horizon of 1-11.

As can be inferred from Table 3.21, each prediction method possesses its own strength and weaknesses. While the LMS predictor is less accurate than the RLS, ERLS, Takens and NERLS predictor, it shows one of the best results in terms of smoothness. Additionally, the computation time is low, even with larger model orders n_a. Since the order needs to be selected much larger for the LMS predictor (60-150) than for the RLS variants (8-30), the dead time is usually 2-5 times as large. The RLS variants show an improved prediction accuracy associated with a slight loss in smoothness, where the smoothness of the ERLS predictor is improved compared to the RLS predictor. The cost for this improvement is a relatively large computation time, where most of the time is spent during update of the inverse autocorrelation matrix $P(t)$. Another drawback of the RLS variants is unreliable predictions during the transition phase, especially for larger prediction horizons. The model-based Kalman predictors are less accurate and smooth as other prediction methods, however they possess no dead time, relatively reliable predictions during the transition phase and the lowest computation times. The Takens predictor, delivering acceptable accuracy and good smoothness, has a relatively small dead time (since optimal parameter values for p and h are small compared to the model orders n_a of the LMS and RLS predictors) and a fast convergence time. The computation time is moderate (due to a highly optimized implementation), where most of the time is required for the search for the best matching embeddings. Unlike other predictors where the computation time is constant (given constant parameters), the computation time here is increasing linearly with time because in each sampling instant a new embedding vector is added. If the average increase of about 10µs/s should lead to a loss of the real-time property of the whole compensation system for exceptionally long datasets, the history of embedding vectors can be limited. Then the Takens predictor also possesses a constant computational time. Note that the Takens predictor is the only one of the discussed predictors which is inherently stable. Since the NERLS predictor is based on the ERLS and Takens predictor, its computation time is larger than both together and also increases with time. It showed the most accurate predictions with smoothness comparable to the other RLS variants, while inheriting the properties for transition and convergence time of the RLS predictors.

In order to reduce the transition phase, several of the discussed predictors can be run in parallel and their predictions used for a certain time. In the beginning of a treatment fraction, the predictions of EKF predictor with spring-mass model can be used for all prediction horizons to bridge the dead time of other predictors, since the EKF predictor possesses no dead time and produces better predictions than the Kalman CV or CA predictors. As soon as the dead time of the ERLS or NERLS predictor has passed, their predictions can be used. However, due to the irregular behavior of predictions with larger prediction horizons during the transition phase, it is advisable to restrict the adoption of predictions from ERLS or NERLS predictors to small prediction horizons, e.g. smaller than five. Usually these predictions converge after a few breathing cycles, so that, depending on a defined number of breathing cycles (determined by the online peak detector discussed in Section 2.2.2.3), the NERLS predictions can be used also for larger prediction horizons. Since the NERLS predictor also incorporates the Takens predictor, breathing predictions can be acquired by the Takens part of that predictor almost without additional computation time. Due to the stable nature of the Takens predictor, its predictions can serve as fallback in case the underlying ERLS predictions become unstable for larger prediction horizons. To summarize, this scheme exploits the advantages of each predictor to serve several goals simultaneously by eliminating dead time, realizing reliable predictions during the transition phase and achieving accurate predictions after convergence while retaining the possibility of always stable predictions.

3.8 Summary

Based on analysis of the general characteristics of breathing and tumor motion, several correlation models were developed to establish a relationship between these motions, which is important for compensation of respiration-induced motion of lung tumors. Especially in the context of the poor quality of EPID-based tumor position information, the correlation enables more reliable predictions of tumor motions than without breathing information. Several prediction methods were presented to enable both breathing and tumor motion predictions. Ranges of the free parameters for each prediction method were determined by comparing their performances on a large number of pre-recorded clinical datasets from real patients using 11 performance metrics representing prediction accuracy, smoothness and convergence time. It was shown that for all prediction methods narrow ranges of the parameters can be found which are able to deal with a wide variety of patient motion characteristics.

Unique extensions to the prediction methods in the form of ARX models were proposed which considerably improve prediction performance by fusing both breathing predictions and past tumor motion information. Simultaneously, they implicitly incorporate the correlation in the model eliminating the use of explicit correlation schemes. Despite the goal of tumor motion compensation being pursued, results of breathing motion prediction was also presented. These results can be generalized to tumor motion prediction if tumor positions are determined by a sensor with higher update rates (e.g. the Calypso system) than the EPID-based system. Therefore, the prediction methods and developed prediction schemes are flexible in terms of the choice of the underlying sensors. The obtained results based on the EPID for tumor position determination can be seen as a worst case which can be improved upon by the availability of other sensor systems. However, since predictions even under these challenging conditions are feasible, they lay the foundation for non-invasive tumor position determination.

By fusing several predictors together, it was shown that the individual advantages of each can be exploited to realize reliable multi-step predictions right from the beginning of the first sampling instant throughout the whole treatment process. Thus, the dead time of predictors can be eliminated completely and transition phases can be improved, which is especially useful for intensity-modulated radiotherapy where treatment fractions usually range from about 40 seconds to two minutes. While other predictors implemented for tumor motion prediction require a considerable amount of training time (e.g. the predictor based on support vector regression used in [30] requires 30 seconds), the benefit of elimination of dead times is obvious.

The obtained multi-step predictions can now be used to generate a reference trajectory for the HexaPOD (cf. Section 2.2.1.5), serving as input to a control scheme which makes the HexaPOD follow that trajectory. The development of appropriate control schemes is the goal of the following chapter.

Chapter 4

Modeling and Control of the HexaPOD

For realizing motion compensation, the actuator of the presented robotic system, i.e. the HexaPOD, needs to be controlled to follow the reference trajectory generated from tumor motion predictions. For the sake of model-based control schemes, a dynamic model of the HexaPOD is required which describes the position response in terms of the commands sent to the HexaPOD, i.e. the control input. Hence, dynamic modeling of the HexaPOD is discussed. A simple linear model is proposed to describe the dynamic response. The model's parameters are gained by system identification methods. A more realistic nonlinear extension to the linear model is presented in order to deal with nonlinear effects in the HexaPOD's position response.

Based on these models, a model predictive controller is derived, enabling the HexaPOD to follow a given desired reference trajectory. A second controller is proposed based on one assumption of the working principle of the internal controller of the HexaPOD. Finally, a third controller is derived as combination of the first and second one. For each of these controllers, comparative results with real hardware experiments and humans in the loop as well as choices of free parameters are presented and discussed.

4.1 Modeling the HexaPOD

The control interface of the HexaPOD uses positions as set points in commands (cf. Section 2.2.1.1). In response to a positioning command, the HexaPOD exhibits a certain dynamic behavior which determines how the desired position given in a command is reached by exerting appropriate accelerations which are integrated over time to velocities and positions. Thus, a dynamic model needs to describe the position-position response. This deviates from the usual problem of dynamic modeling where normally the input is given as torque or voltage which results in a certain velocity and position response. To the knowledge of the author there is no description available which models the position-position response in physical terms. Hence, modeling the dynamic response of the HexaPOD proceeds by assuming a certain model structure and acquiring its parameters by system identification methods.

4.1.1 Linear ARX Models

As a first approach, the HexaPOD is considered a black box and its response will be modeled by an ARX model. For a single translational axes $j = 1, 2, 3$, the input $u_j(t) \in \mathbb{R}$ to the model can be identified with the desired position present in the current command to the HexaPOD whereas the output $y_j(t) \in \mathbb{R}$ represents the actual position of the HexaPOD. Since each axis can be actuated independent of the others, a separate model is setup for each axis.

For convenience of presentation, the model is stated as a SISO system representing the dynamics of the HexaPOD treatment couch for a single axis. Generally the model is given by

$$A_j(q)y_j(t) = B_j(q)u_j(t-n_k) + e_j(t) \ . \tag{4.1}$$

q is the delay operator with

$$\begin{aligned} A_j(q) &= 1 + a_{1,j}q^{-1} + \cdots + a_{n_a,j}q^{-n_a} \ , \\ B_j(q) &= b_{0,j}q^{0} + b_{1,j}q^{-1} + \cdots + b_{n_b-1,j}q^{-n_b+1} \ , \end{aligned}$$

such that

$$A_j(q)y_j(t) = y_j(t) + a_{1,j}y_j(t-1) \ + \cdots + a_{n_a,j}y_j(t-n_a) \ , \tag{4.2}$$

$$B(q)_j u_j(t-n_k) = b_{0,j}u_j(t-n_k) + b_{1,j}u_j(t-n_k-1) \ + \cdots + b_{n_b-1,j}u_j(t-n_k-n_b+1) \ , \tag{4.3}$$

include the information from the past n_b samples of the input shifted backward by n_k samples as well as the past n_a samples of the output signal. Thus, the order of the model is given by the tuple (n_a, n_b, n_k).

Rearranging Eqn. (4.1) – (4.3) to

$$y_j(t) = \varphi_j^T(t)\theta_j + e_j(t)$$

with the vectors

$$\varphi_j(t) = [y_j(t-1), \ldots, y_j(t-n_a), \ u_j(t-n_k), \ldots, u_j(t-n_k-n_b+1)]^T \ , \tag{4.4}$$

$$\theta_j = [a_{1,j}, \ldots, a_{n_a,j}, \ b_{0,j}, \ldots, b_{n_b-1,j}]^T \ , \tag{4.5}$$

the problem statement of modeling is to find the best parameter vector θ_j minimizing the error $e_j(t)$ from input/output samples of an identifying dataset. Having once obtained the parameters, the model can be used for simulation using a test dataset for model validation. Starting from an initial condition $\varphi_j(0)$, the undisturbed model

$$y_j(t) = \varphi_j^T(t)\theta_j \tag{4.6}$$

is consecutively fed with a new input sample $u_j(t-n_k)$ and the last model output $y_j(t-1)$ in each sampling instant by updating $\varphi_j(t)$.

4.1.1.1 System Identification

System identification [150] refers to the procedures carried out to find the parameters of the model. Given a set of equidistantly sampled inputs and outputs of the ARX model, its parameters are identified by the least-squares method, which finds the minimum of the squared errors $e_j(t)$ with respect to the model parameters θ_j. Naturally, the choice of the input signal affects the dynamic response which can be observed on the output. That is, the dynamic response varies with different input signals. Hence, for a certain application of the system to be modeled, it is important to design the input to the system in such a way to make the system exhibit all the dynamics, especially the ones which are required for the application, so that these dynamics are reflected by the identified parameters.

In the context of this work, the HexaPOD will be used to compensate for quasi-periodic breathing and tumor motions. In this case, the HexaPOD will perform approximately the same motion in the opposite direction. The main characteristics of these motions were analyzed in Section 3.1. It was found that main frequencies range from 0.17Hz to about 0.6Hz, where most of the patients show dominant motion frequencies in between 0.225Hz and 0.375Hz. Maximum amplitudes differ depending on the axis. On average, the maximum amplitude range (peak to peak) moves in between 0.805mm to 6.705mm, where its maximum value lies in between 3.901mm and 29.056mm. This amplitude limit has been reported to be exceeded only in rare cases [78]. Furthermore, it was discussed that for tumor motion compensation the HexaPOD will have to operate, during some phases, at its maximum velocity with good coverage of the main phases still possible.

Since patients have different and varying motion characteristics, the design of input signals suitable for system identification needs to take the full range of these characteristics into account. Chirp signals with a constant amplitude can cover the whole frequency range because frequencies are varied with time. With linear chirp signals, the instantaneous frequency $f(\tau) \in \mathbb{R}$ is varied linearly with time τ according to

$$f(\tau) = f_0 + c\tau \ ,$$

where $f_0 \in \mathbb{R}$ is the starting frequency being active at time $\tau = 0$ and $c \in \mathbb{R}$ is the chirp rate, which determines how fast the frequency rises with time. A specific value of the linear chirp signal is given by

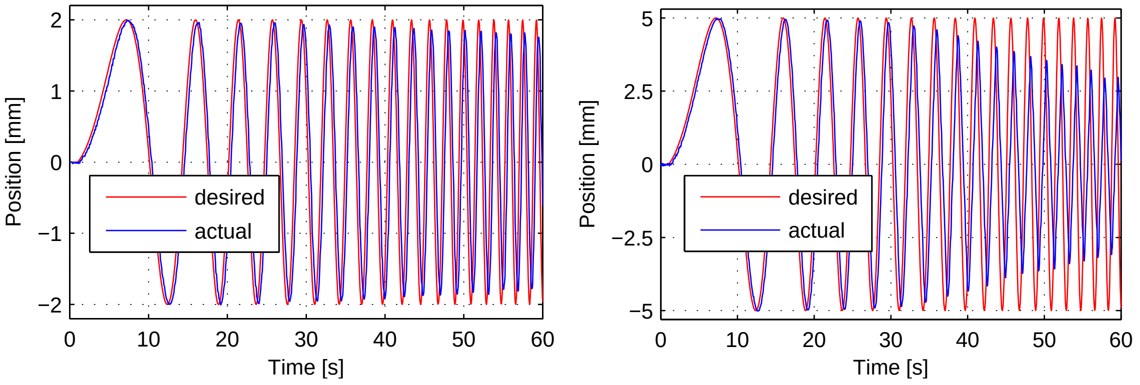
$$u_j(\tau) = a \sin\left(2\pi \int_0^\tau f(\tau')d\tau'\right) = a \sin\left(2\pi \left(f_0\tau + \frac{c}{2}\tau^2\right)\right) , \tag{4.7}$$

where a is the amplitude of the chirp. To cover the frequency range in a single identifying dataset, the starting frequency was chosen to be $f_0 = 0.01$Hz, the chirp rate was set to $c = 0.01$, so that at a maximum time $\tau = 60$s the instantaneous frequency of 0.6Hz is reached. Three datasets with different amplitudes were created, where the amplitude was constant in each dataset. The datasets are referred to as C1, C2 and C3, for which the amplitude a was chosen as 1mm, 2mm and 5mm, respectively (leading to peak-to-peak amplitudes of 2mm, 4mm and 10mm, respectively) to cover the main range of amplitudes. The datasets were generated by sampling Eq. (4.7) with a rate of 33Hz and using the corresponding values for all translational axes.

Apart from the ability of chirps to cover necessary frequency and partially also amplitude ranges, they additionally fulfill the persistent excitation condition [151] which is essential for system identification. The order of persistent excitation of an input signal generally imposes an upper limit on the order of the model to be identified. If the order of the model is n, then the input needs to be at least persistently exciting of the same order n to reliably identify its parameters. Essentially this condition guarantees that the autocorrelation matrix stays non-singular throughout the identification process. This is important because its inverse is required for the least-squares solution of the model's parameters. Since step inputs are persistently exciting of order 1, they are suitable for identification of first-order systems only. Sine signals with one frequency component are persistently exciting of order 2. For each frequency component added to a sine signal which produces a separate spectral line, the order increases by two. However, a chirp signal combines many frequency components together so that the design of an input signal remains straightforward. The order of persistent excitation was evaluated with MATLAB for the generated discrete datasets. Using the standard threshold of 10^{-9} for the algorithm which determines the order of persistent excitation, the chirp sampled at 33Hz shows an order of 12 while the same chirp sampled at 20Hz is persistently exciting of order 14. Thus, it will be possible to identify models with orders at least up to 10.

The experiments for system identification were carried out at the Department of Radiation Oncology in Würzburg. For this experiment, the HexaPOD was loaded with 100kg and equipped with a C-shaped bridge (cf. Figure 2.5(c)) with IR markers mounted on it. With this IR tool, the actual position of the HexaPOD was measured by the Polaris Spectra infrared camera at a rate of 20Hz. On the control PC the testing software package (cf. Section 2.2.1.6) was instructed to send the samples of the datasets as desired positions to the HexaPOD. It was ensured that the timing accuracy of the application is well below 1ms to avoid timing errors being the reason for position displacements. The control PC also interfaces with Polaris IR camera, so that a common clock source can be used for timestamping both the Polaris measurements and sending of desired positions to the HexaPOD. For evaluation, both datasets were interpolated to a common sampling rate of 20Hz.

Figures 4.1(a) and 4.1(b) depict the results along the z-axis for the chirp signals C2 and C3, respectively. The red line shows the desired positions for the z-axis which were sent to the HexaPOD using a command. The blue line depicts the actual position of the HexaPOD which was measured by the Polaris camera. As time progresses, i.e. the instantaneous frequency of the chirp signal rises, the actual position of the HexaPOD does not reach the desired one any more for the chirp C2 in Figure 4.1(a). This behavior occurs due to the velocity limit of the HexaPOD. The chirp C3 possesses a larger amplitude and the same frequencies as C2, so that the linear velocities of C3 are larger. Consequently velocity saturation has a stronger impact on the actual position of the HexaPOD, which can be clearly observed in Figure 4.1(b). The plots exemplify the results obtained for the z-axis. For the x- and y-axes, similar results were obtained. Table 4.1 gives an overview of all chirp signals and a qualitative assessment, if velocity saturation occurred, that is, if the desired amplitude was not reached on the output. This data consisting of input samples originating from the chirp datasets and output samples obtained by measurements of the



(a) Chirp signal C2

(b) Chirp signal C3

Figure 4.1: *Exemplary chirp signals C2 and C3 used for model identification, red: desired position (input, command), blue: actual position (output, measurement)*

ID	a [mm]	f_0 [Hz]	c	f_e [Hz]	saturation
C1	1	0.01	0.01	0.6	no
C2	2	0.01	0.01	0.6	minor
C3	5	0.01	0.01	0.6	yes

Table 4.1: *Chirp datasets with starting frequency f_0, chirp rate c and end frequency f_e used for system identification*

HexaPOD's position is used to perform parameter identification of ARX models.

Furthermore, several test datasets with two superimposed sine signals of different frequencies f_1 and f_2 and amplitudes a_1 and a_2 were used for validating already identified models. They were generated by sampling

$$u_j(\tau) = a_1 \sin(2\pi f_1 \tau) + a_2 \sin(2\pi f_2 \tau)$$

with a rate of 33Hz. These are summarized in Table 4.2. While the sine datasets S1 to S4 show no velocity saturation because the linear velocity of the superimposed sines is smaller than the HexaPOD's velocity limit, an almost permanent velocity saturation is observed for the sine datasets S5 and S6, where the HexaPOD continuously moves at maximum speed.

ID	f_1 [Hz]	a_1 [mm]	f_2 [Hz]	a_2 [mm]	saturation
S1	0.1	1	0.4	2	no
S2	0.1	2	0.4	1	no
S3	0.05	4	0.25	3	no
S4	0.05	3	0.25	4	no
S5	0.1	10	0.4	20	yes
S6	0.1	20	0.4	10	yes

Table 4.2: *Sine datasets with frequencies f_1, f_2 and amplitudes a_1, a_2 used for system identification validation tests*

The criterion for model validation along an axis j in this analysis is the RMS error

$$\mathrm{RMS}(y_j^M, y_j) = \sqrt{\frac{1}{n}\sum_{t=1}^{n}\left(y_j^M(t) - y_j(t)\right)^2}\,, \tag{4.8}$$

where $y_j(t) \in \mathbb{R}$ is the measured position of the HexaPOD along axis j and $y_j^M(t) \in \mathbb{R}$ is the output of the simulated model (4.6). Before the RMS error can be determined, both time series need to be interpolated to the same sampling times, resulting in a common length $n \in \mathbb{N}$. Since all datasets start at position zero along all axis with the HexaPOD initially resting, the initial vector of observations and commands $\varphi_j(0)$ is set to the zero vector. Note that for the validation tests, the model is completely simulated. No updates with actual positions of the real HexaPOD are performed. Instead when moving from one sampling instant to the next, φ_j is updated accordingly with the previous output of the model. As a consequence of this, errors can accumulate in the model due to parameter mismatch as opposed to the real HexaPOD.

4.1.1.2 Model Order

The orders of the ARX model, i.e. the tuple (n_a, n_b, n_k), are the free parameters in the problem of system identification. They will be estimated by a grid search in a range of suitable order values using the identifying datasets C1 and C2. In addition to the datasets S1 to S4, C1 or C2 (the one which is not used for identification) is also used for model validation. These datasets have in common that no or only minor velocity saturation occurs, so that they result in linear behavior. If C3 is used for identification, the resulting model would capture the saturation present in the output of the identifying dataset and would lead to large errors during model validation using datasets with no velocity saturation. Due to the order of persistent excitation of the chirp datasets, n_a, n_b and n_k were varied between 1 and 10. Note that the ARX model is expressed in terms of the backward shift operator, so that for $n_b > n_a$ the resulting model is still causal even if n_k is increased. For each model order (n_a, n_b, n_k) the following procedure was applied: An ARX model of the chosen order was identified using the dataset C1. Then the model was simulated with the datasets C2 and S1 to S4 and for each simulation trial the RMS error according to Eq. (4.8) was recorded. After that, the procedure was repeated with C2 as identifying dataset and C1 was included in the model validation datasets instead of C2. Finally, the mean value was determined from all acquired RMS errors.

These average RMS errors are depicted in Figure 4.2 for the z-axis. Each plot shows the results for a single n_k in the range 1-6 over n_a and n_b. Almost identical plots are found for the x- and y-axis as well, so that following considerations can be applied to these axes, too. Additionally the location of the three smallest RMS errors is highlighted with a red dot. In general it can be concluded irrespective of n_k that smallest RMS errors are found

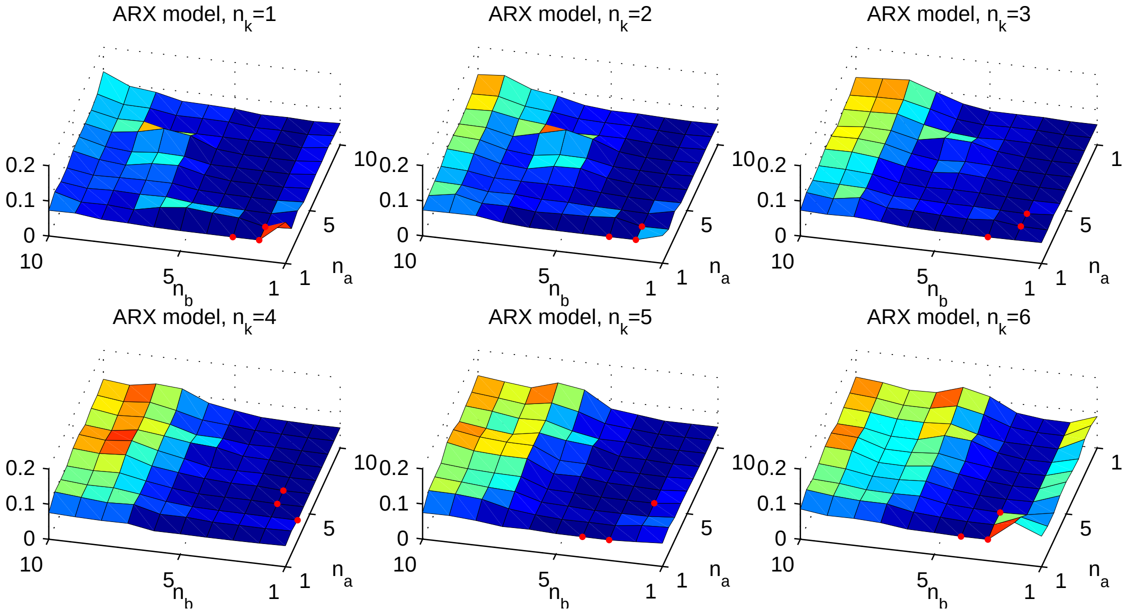


Figure 4.2: *Average RMS error along z-axis when identifying ARX models with C1 and C2 and applying them to test datasets C1, C2, S1 to S4*

for $n_b \leq 5$. In this case n_a has a minor impact, however RMS errors are slightly increasing with rising n_a. For most values of n_k it can be seen that $n_a = 1$ and $n_b = 1$ results in increased errors, so that this order should not be chosen. For $n_k \leq 4$, the most frequent occurrences of the three smallest RMS errors are found for $n_b = 2$ or $n_b = 3$ with $n_a \leq 2$. This suggests that the HexaPOD can be modeled as a second-order system for one axis. For rising model orders, measurement errors and disturbances which are of higher order are also incorporated into the model leading to larger RMS error. Hence, a smaller model order simultaneously acts as a low pass filter, rendering the model less sensitive to noise.

Figure 4.3 visualizes the average of the three smallest RMS errors for each n_k individually in order to elaborate the dependency on n_k. Generally small values of $n_k \leq 4$ lead to the smallest RMS errors where the minimum is found at $n_k = 3$ for the x- and y-axis and at $n_k = 4$ for the z-axis. However, choosing smaller values is also a feasible approach as RMS errors are still close to the ones for $n_k = 3$. On the other hand, increasing n_k to values larger than 6 results in rising RMS errors (they were included in the simulation but are not shown here). While the best model orders are similar for the different axes, the RMS errors show a small difference. Part of this difference is subject to noise effects of the Polaris system, where the noise power is different along each axis (cf. Section 3.4.3). Another part of this error can be the caused by small deviations during calibration (which is performed with noisy measurements of the Polaris system), rendering the axes of the Polaris IR camera and the HexaPOD system to not perfectly coincide. Furthermore, slightly different dynamics of the HexaPOD along each axis can also be a reason. During the estimation of the HexaPOD's latency (cf. Table 2.4), latencies between 103ms and about 240ms were observed when the HexaPOD was initially at rest or moving with minimum velocity, where the largest latencies were found for the x-axis. When the HexaPOD was moving faster, then latencies drop down to approximately 24ms to 38ms. In the identifying chirp datasets both slow and fast motions are present. Combining both observations of the latencies approximately matches the choice for $n_k = 3$ which corresponds to 150ms. If the HexaPOD is mostly moving fast, also the choice of $n_k = 1$ is valid, which comes closer to the reduced latencies.

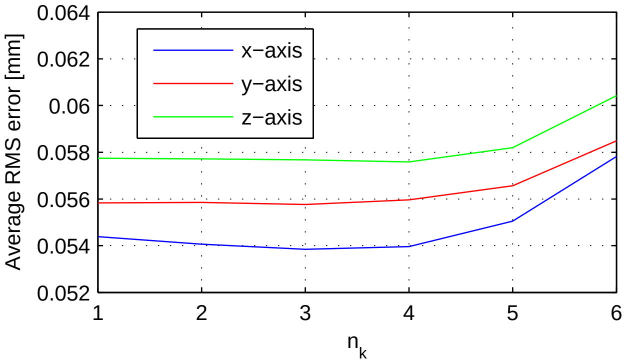


Figure 4.3: *Average RMS error along the translational axes of the three parameter sets (n_a,n_b) with smallest RMS errors shown for each n_k*

When the HexaPOD is instructed to change its position given a certain command, it accelerates, then reaches a phase with constant velocity and finally decelerates again until the desired position has been reached. Using step inputs, the typical second-order behavior with two delay elements can be observed on the actual position. Hence, the model order should be selected as two (for both n_a and n_b), consistent with the results from Figure 4.2.

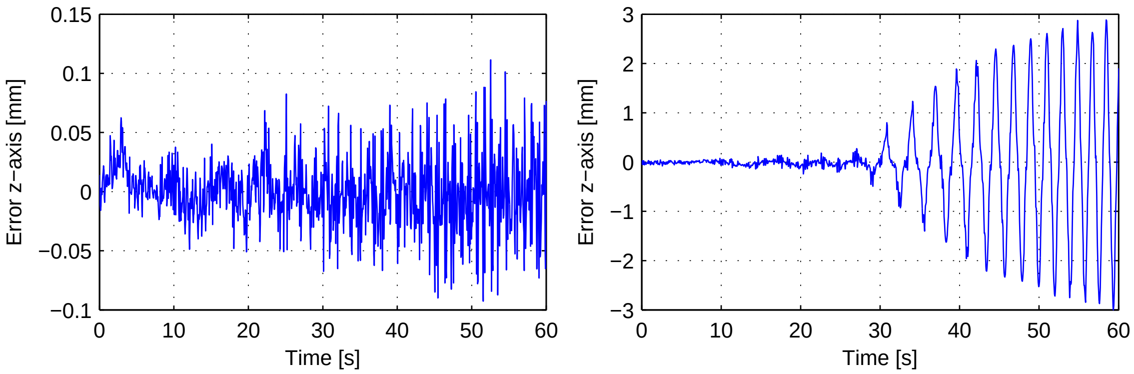
4.1.1.3 Model Validation

For model validation, three models of order (2,2,3) were identified with datasets C1, C2 and C3, respectively. Each model was simulated with the inputs from datasets S1 to S4 and C1 to C3 excluding the one used for identification. The resulting RMS errors obtained by averaging over the translational axes are summarized in Table 4.3. It is seen from the first two columns (except the row for C3) that the models identified with C1 and C2 show similar and small RMS errors. These are the datasets with almost no velocity saturation. In contrast to that, the model identified with C3 (with velocity saturation) in the third column shows RMS errors more than one magnitude larger. The same behavior results for the models identified with C1 or C2 and applied to C3 (found in row for C3). Consequently, the identification dataset should contain no velocity saturation to allow small errors when applied to test datasets without velocity saturation. Of course, if the test dataset shows saturation, then larger errors are to be expected because the dampening induced by velocity saturation is not captured by the model. Conversely, if the identifying dataset contains saturation, then large errors are found for test datasets without saturation. In this case the model parameters reflect the damped response from the identifying dataset which is different for datasets without saturation possessing no dampening. Additionally, when identifying with a dataset showing saturation (e.g. C3) and applying this to a dataset also containing saturation but with different amplitudes, the dampening will change because saturation is a nonlinear effect. Due to this reason, large errors will occur in this situation. Hence, useful results can only be achieved when identifying with a dataset without saturation. Since the RMS errors are smallest for the identifying dataset C2 except for test dataset S4, C2 is used for identification in the following.

Test dataset	RMS ID C1 [mm]	RMS ID C2 [mm]	RMS ID C3 [mm]
S1	0.0614	**0.0607**	0.5819
S2	0.0418	**0.0416**	0.5578
S3	0.0653	**0.0648**	1.6018
S4	**0.0774**	0.0794	1.2802
C1	-	**0.0343**	0.2557
C2	**0.0540**	-	0.5055
C3	1.0068	**0.9977**	-
Sum	1.3068	**1.2786**	4.7829

Table 4.3: *Model validation results of ARX model identified with datasets C1, C2 and C3 and applied to test datasets S1 to S4 and C1 to C3; smallest RMS errors are typeset in bold for each test dataset*

Figure 4.4 exemplifies these results for the z-axis when identifying the ARX model with C2. Figure 4.4(a) shows the error when applying the model to C1, where both datasets do not contain saturation. Most of the errors stay below 0.1mm, however with slight increase when the HexaPOD moves faster towards the end of the dataset. Figure 4.4(b) depicts



(a) Applied to dataset C1 (without velocity saturation) (b) Applied to dataset C3 (with velocity saturation)

Figure 4.4: *Error along z-axis of linear ARX model identified with C2 (without velocity saturation) and applied to two different datasets*

the errors when applied to C3 containing large saturation. It can be observed that as soon as velocity saturation occurs at around 28s, model errors start to increase rapidly.

From previous discussion it becomes evident that linear ARX model are sufficient to capture the dynamics of the HexaPOD as long as it operates below its speed limit. However, if the limit is reached, the system becomes nonlinear and the dynamics do not follow the ARX model any more. As a consequence of this, linear ARX models can merely be used as a first approximation to the problem.

4.1.2 Nonlinear Extension of ARX-Model

The nonlinear velocity saturation gives rise to an extension of the linear ARX model which explicitly takes saturation into account. The extension is based on the idea that the HexaPOD's dynamics follow the ARX model as long as no saturation takes place, so that the output of the ARX model can be used as it is. If the occurrence of saturation is detectable, then the output of the ARX model can be modified accordingly to reflect saturation.

In order to accomplish this, the preliminary output $\tilde{y}_j^M(t)$ along axis j of the model is calculated in accordance with Eq. (4.6) by

$$\tilde{y}_j^M(t) = \varphi_j^T(t)\theta_j.$$

Given a constant and sufficiently small sampling time $\Delta\tau$ and a position $y_j^M(t-1)$ of the simulated HexaPOD at the previous sampling instant, the linear velocity $\tilde{v}_j^M(t)$ can be derived as

$$\tilde{v}_j^M(t) = \frac{\tilde{y}_j^M(t) - y_j^M(t-1)}{\Delta\tau}$$

which would be applied according to the model as a result of a certain input value $u_j(t)$. Note, that the acceleration can be neglected for small $\Delta\tau$.

The potential velocity $\tilde{v}_j^M(t)$ is then bounded by the maximum velocity v_{max} of the

HexaPOD, yielding the constrained velocity

$$v_j^C(t) = \begin{cases} \tilde{v}_j^M(t), & \text{if } |\tilde{v}_j^M(t)| \le v_{max} \\ \operatorname{sgn}(\tilde{v}_j^M(t)) v_{max}, & \text{if } |\tilde{v}_j^M(t)| > v_{max} \end{cases}.$$

Since $v_j^C(t)$ is always less or equal to the maximum velocity, it can be realized by the HexaPOD. To complete the calculation cycle, $v_j^C(t)$ is integrated to the final output position $y_j^M(t)$ with respect to time as

$$y_j^M(t) = y_j^M(t-1) + v_j^C(t)\Delta\tau.$$

In case the HexaPOD operates below the maximum velocity, the output of the extended model is the same as the one from the original ARX model. Thus, it behaves according to the dynamics captured by its model parameters obtained during system identification. As soon as the velocity limit would need to be exceeded, the constraint bounds the velocity to its allowed range, so that only realizable velocities are exhibited by the extended model. In total, this scheme can incorporate linear behavior as long as the maximum velocity is not exceeded and is simultaneously able to deal with the nonlinearity imposed by velocity saturation.

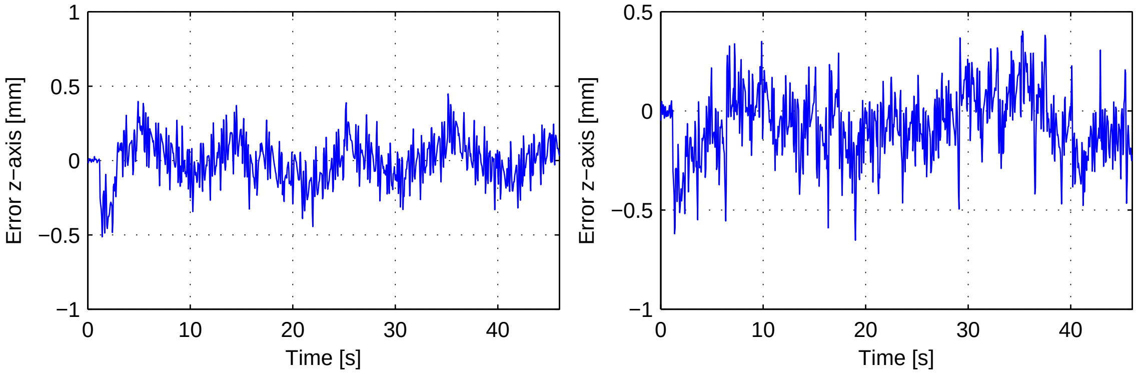
Model Validation Since nonlinearly extended ARX model contains the linear ARX model, all considerations with respect to model order are valid here, too.

The same model validation tests as for the linear ARX model was performed on the extended ARX model. Additionally, the test datasets S5 and S6 are included in this analysis. The resulting RMS errors are given in Table 4.4. The RMS errors of the datasets S1 to S4, C1 and C2 with almost no velocity saturation (and identified with C1 or C2) are either the same or slightly smaller than the ones of the linear ARX model given in Table 4.3. For these datasets the saturation component of the extended model plays only a minor role because the velocity limit is mostly not reached. In this case, the solutions of

Test dataset	RMS ID C1 [mm]	RMS ID C2 [mm]	RMS ID C3 [mm]
S1	0.0609	**0.0605**	0.5819
S2	0.0418	**0.0416**	0.5578
S3	0.0651	**0.0647**	1.6018
S4	**0.0770**	0.0790	1.2312
S5	0.1947	**0.1556**	7.1588
S6	0.2590	**0.2001**	3.4432
C1	-	**0.0343**	0.2557
C2	**0.0540**	-	0.5055
C3	0.1068	**0.0974**	-
Sum	0.8593	**0.7333**	15.3359

Table 4.4: *Model validation results of nonlinearly extended ARX model identified with datasets C1, C2 and C3 and applied to test datasets S1 to S6 and C1 to C3; smallest RMS errors are typeset in bold for each test dataset*

the linear ARX model are used for the final model output. However, the RMS error for the test dataset C3 with velocity saturation improved by an order of magnitude and is almost in the same range as the other test datasets. Also the test datasets S5 and S6, representing extreme cases with a large amplitude range of 57.3mm on the input, show relatively small RMS errors (compared to the input amplitudes) which would not be realizable with the linear model alone. The errors, shown in Figures 4.5(a) and 4.5(b) are found to be two orders of magnitude smaller than the maximum amplitudes in the desired signal.



(a) Applied to dataset S5 (with extreme velocity saturation)

(b) Applied to dataset S6 (with extreme velocity saturation)

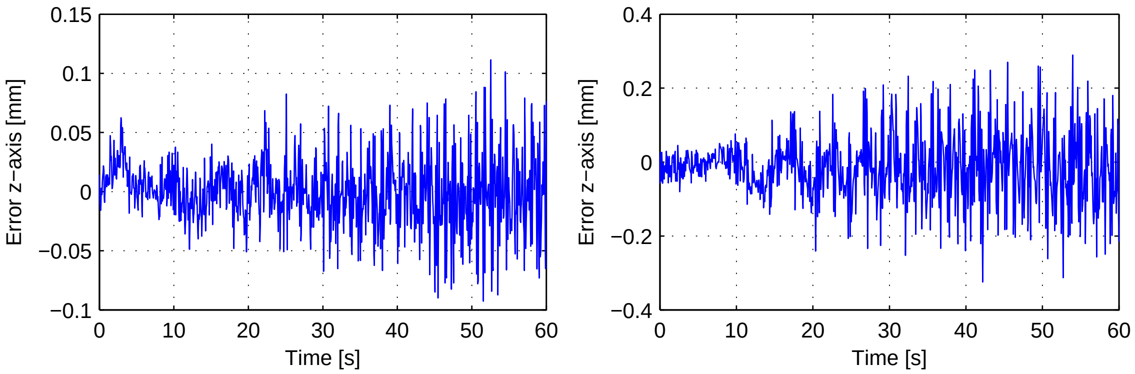
Figure 4.5: *Error along z-axis of nonlinearly extended ARX model identified with C2 (without velocity saturation) and applied to datasets S5 and S6*

Of course, identification has to be performed using datasets containing no saturation. As can be seen from the last column of Table 4.4, large RMS errors result if identification is done with C3. Thus, datasets with velocity saturation such as C3 should not be used for identification of the underlying linear ARX model as the saturation part is covered by the nonlinear extension. In analogy to the results for the linear model, C2 should be used as identifying dataset since the RMS errors are the smallest for each test dataset except for S4.

Figure 4.6 shows the results for the z-axis when identifying the ARX model with C2. Comparing this to the case of the linear ARX model (cf. Figure 4.4), the error of the test dataset C1 depicted in Figure 4.6(a) is very similar. However, most of the errors of the test dataset C3 shown in Figure 4.6(b) are less than 0.3mm which is an order of magnitude smaller than the errors of the linear model. Thus, the nonlinearly extended ARX model is capable of dealing with the HexaPOD's dynamics even in the case of velocity saturation. Furthermore, these results, also given in [56], demonstrate that the velocity saturation is the nonlinearity which dominates the response of the HexaPOD.

4.1.3 Related Modeling Approaches

To the knowledge of the author, so far one different modeling attempt has been done to setup a dynamic model for the HexaPOD. In [152], a second-order dynamic model was selected for the treatment couch where its model parameters were identified by using a single step input as position command. Then an internal model control scheme was adopted for motion compensation. Depending on the dead time embedded in the model, the controller has shown different performance in simulation compared to hardware tests



(a) Applied to dataset C1 (without velocity saturation)

(b) Applied to dataset C3 (with velocity saturation)

Figure 4.6: *Error along z-axis of nonlinearly extended ARX model identified with C2 (without velocity saturation) and applied to two different datasets*

and it was found that the HexaPOD behaved even worse than motion compensation without feedback control, in which desired positions were sent to the HexaPOD as positioning commands resulting in open-loop control.

After the presented modeling approach discussed in this work, the observed behavior can be caused by the following reasons: At the time of the writing of [152] inherent dead time of the HexaPOD was unknown and could not be compensated for by the internal model controller, in which an inverse model of the HexaPOD is used to eliminate the HexaPOD's dynamics. Another reason is shown by the nonlinearly extended model, demonstrating that the HexaPOD's dynamics become nonlinear if the maximum speed of the HexaPOD is reached. The modeling of the HexaPOD revealed that a linear second order model can be used for representing the dynamics, but larger errors are to be expected when a test signal is faster than the speed limit. Furthermore, the model parameters were identified using a single step input which does not fulfill the persistent excitation condition, implying higher order dynamics of the HexaPOD cannot be stimulated by this input and therefore are not identified. To aggravate this even more, it was found in tests with the HexaPOD that step inputs deliver different responses depending on the current position, desired position and also the normalized velocity specification in the HexaPOD's command. These differences prohibited the use of a single dynamic model which is not able to deal with all tested step inputs simultaneously. Additionally, no conclusive relationship between the normalized velocity specification and the responses to step inputs could be identified. For this reason, identification was based on position-position responses while using the maximum velocity specification in combination with appropriately designed input signals covering similar behavior as during compensation.

4.2 Control of the HexaPOD

A control scheme is applied to the HexaPOD in order to enable tumor motion compensation within the ATTS. The main task of a controller is to provide appropriate control input to the HexaPOD which makes the HexaPOD follow the given reference trajectory. Since the ATTS is a discrete system, controllers are implemented in software in a discrete way. After the problem formulation of controlling the HexaPOD has been stated, several metrics are discussed with which control methods are compared and their performance assessed. The remainder of this section discusses three developed controllers and evaluates their performance in several scenarios using hardware- and human-in-the-loop experiments.

4.2.1 Problem Formulation

Depending on the target application, usually three different goals are to be realized by a controller:

- Disturbance rejection: Any disturbance acting on the system or on sensor measurements should be canceled as fast as possible by the controller so that the system remains at a predefined operation point.
- Set-point regulation: When the set-point or the desired value changes, the system output should be regulated to the set-point, usually obeying some performance specification such as settling time and maximum overshoot.
- Reference tracking: The system output should follow a given reference trajectory as close as possible; this can be viewed as permanently changing the “set-point” with time.

The problem of lung tumor motion compensation clearly aims at reference tracking since lung tumors are continuously changing their position intra-fractionally, which requires appropriate adaptation of the desired position of the HexaPOD. This is generally the case in all compensation modes of RDC, DMC and FMC (cf. Section 2.1) when compensating lung tumor motions. However, with DMC the rate of change is smaller, so that the desired position of the HexaPOD also needs to change at a slower rate. When compensating prostate tumors in RDC or DMC, the problem rather shifts to set-point regulation as changes in the desired position occur more infrequently. In this situation disturbance rejection becomes more important than with reference tracking as the impact of noise effects increases due to a larger ratio to the traveled distance of the HexaPOD. It has to be ensured that disturbances and noise are sufficiently damped by the controller, so that they are not amplified thereby rendering the closed-loop system instable. Nevertheless, a certain disturbance rejection is also required within reference tracking to account for patient comfort as any harsh motions by the HexaPOD are directly transmitted to the patient.

In any compensation mode it is assumed in the following that for each translational axis j a future reference trajectory $r_j(t+k|t) \in \mathbb{R}$ with $1 \leq k \leq n_r$ is available at each sampling instant t. The finite horizon length n_r depends on the requirements of the chosen controller. The main task of a controller is to make the HexaPOD follow the reference trajectory as close as possible. Furthermore, the actual trajectory of the HexaPOD should be smooth enough so that patients feel comfortable.

It is further assumed that the calibration of the HexaPOD and all sensor systems was performed such that the origin of the HexaPOD’s reference frame coincides with the isocenter of the treatment beam. In FMC mode, the reference trajectory is obtained by negating corresponding predictions acquired from the chosen prediction method from Chapter 3. Negation is necessary for compensation to make the HexaPOD move into the opposite direction as the tumor. In DMC mode, the reference trajectory is generated by using the current negated value of the moving average. If a controller requires more than one future reference, i.e. $n_r > 1$, successive values of the reference are kept constant. The same procedure is followed in RDC mode where the reference trajectory is formed by the last negated measurement of the tumor position.

In the following sections, several controllers are presented to achieve the goal of motion compensation. These are possible realizations of the block “control scheme” depicted in Figures 2.6 and 2.7.

4.2.2 Performance Metrics

Generally, the same performance metrics as for the predictors are of interest for assessing the performance of controllers. Since reference tracking is one of the main goals to be achieved by a controller, the 3D tracking error

$$\vec{e}(t) = \vec{r}(t|t-1) - \vec{p}^{H}(t) \tag{4.9}$$

between the reference position $\vec{r}(t|t-1) = [r_1(t|t-1), r_2(t|t-1), r_3(t|t-1)]^T$ and the actual position $\vec{p}^{H}(t)$ of the HexaPOD at sampling instant t is an important measure to describe tracking accuracy. Based on the tracking error, the following metrics are used in the remainder of this chapter to assess tracking accuracy: maximum absolute tracking error $\mathrm{MAX}(\vec{e})$ (Eq. (3.7)), 95% confidence interval of absolute tracking error $\mathrm{CI95}(\vec{e})$ (Eq. (3.8)), root-mean-square tracking error $\mathrm{RMS}(\vec{e})$ (Eq. (3.9)) and mean absolute tracking error $\mathrm{MAE}(\vec{e})$ (Eq. (3.10)).

Equivalently to the prediction ratio from Eq. (3.11), compensation ratio (CR) is defined for a single axis j as

$$\mathrm{CR}(r_j, e_j) = \left(1 - \frac{\mathrm{RMS}(e_j)}{\mathrm{RMS}(r_j)}\right) \cdot 100\%$$

and along multiple axes as

$$\mathrm{CR}(\vec{r}, \vec{e}) = \frac{\sum_{j=1}^{3} \mathrm{RMS}(r_j)\,\mathrm{CR}(r_j, e_j)}{\sum_{j=1}^{3} \mathrm{RMS}(r_j)} = \frac{\sum_{j=1}^{3} \mathrm{RMS}(r_j) - \mathrm{RMS}(e_j)}{\sum_{j=1}^{3} \mathrm{RMS}(r_j)} \cdot 100\% \;.$$

The smoothness metrics S_F, S_J, S_{TS} and S_{SNR} from Eqn. (3.14) – (3.17) are also adopted. Since the smoothness of the resulting trajectory exhibited by the HexaPOD is of interest, all the smoothness measures are determined from the actual positions of the HexaPOD $\vec{p}^{H}(t)$. The multiple axes pendant in Eq. (3.18) is consequently computed as $S(\vec{p}^{H}(t), \vec{r}(t|t-1))$, thus weighting the smoothness metrics along the individual axes according to the RMS value of the corresponding reference trajectory (where the one-step trajectory $r_j(t|t-1)$ is the desired trajectory for the HexaPOD along axis j).

Finally, the convergence time metrics according to Eqn. (3.20) and (3.21) are employed in the CT25 and CT10 variant. The difference in computation is that the convergence coefficient is determined as $C\left(r_j(t|t-1), p_j^H\right)$ for an axis j.

Depending on suitability to the control method, additional metrics are also introduced during the following discussion of control schemes.

4.2.3 Model Predictive Control of ARX-Model

Generally, the availability of a model describing the HexaPOD's dynamics is a prerequisite to enable model-based control of the HexaPOD. The nonlinearly extended ARX model which takes into account the velocity limit of the HexaPOD, demonstrated that this model supersedes the linear ARX model. However, to the knowledge of the author, so far no convincing way has been determined to build a controller based on the nonlinearly extended model due to saturation. However, it is possible to use the underlying linear ARX model in combination with a model predictive controller (MPC) [153]. Of course, performance reduction is then to be expected in the presence of speed limit violations.

The linear ARX model with identified parameters can be transformed to a discrete state-space model which is common to the formulation of the MPC problem and used in the remainder of this work. MPC of the identified ARX model is further denoted as IMPC. MPC was chosen here for the following reasons: First, it explicitly takes into account a model of the plant, i.e. the HexaPOD, which is already available. Second, it derives the control input based on optimization of the future system output and control effort given a future reference trajectory, thus allowing to optimize for both tracking accuracy and patient comfort. And third, it simultaneously respects actuator limits. Since the underlying model incorporates the position-position response, only the operational range of the HexaPOD can be used as actuator limits. Especially the optimizing character is useful in this context because velocity saturation cannot be taken into account explicitly and therefore deviations between the model and the real HexaPOD are to be expected. Hence, MPC in combination with the model results in a straightforward way to answer the question which positioning commands need to be sent to the HexaPOD in order to enable reference tracking.

4.2.3.1 MPC Formulation

The basis of a model predictive controller in this work is a linear, time-invariant, discrete and proper SISO state-space model of the form

$$\begin{aligned} x_j(t+1) =& Ax_j(t) + B(u_j(t-1) + \Delta u_j(t)) \\ y_j(t) =& Cx_j(t) \end{aligned} \tag{4.10}$$

with state $x_j(t) \in \mathbb{R}^n$, system matrix $A \in \mathbb{R}^{n\times n}$, input matrix $B \in \mathbb{R}^n$, output matrix $C \in \mathbb{R}^{1\times n}$, input vector $u_j(t) = u_j(t-1) + \Delta u_j(t) \in \mathbb{R}$, output $y_j(t) \in \mathbb{R}$ where n is the model order and j represent a translational axis. In contrast to the standard formulation of a state-space model, the control increment $\Delta u_j(t)$ appears explicitly in model (4.10). The advantage of this formulation will become apparent soon.

This SISO model can be applied to only one of the translational axes. Since the desired positions for the individual axes in the HexaPOD's command are independent from each other, the MPC is applied to each axis separately. The matrix A and vectors B and C are derived from the ARX model, specifically from the elements of the parameter vector θ_j from Eq. (4.5) for a single axis j. The resulting model order n of the state-space model is generally given by $n = \max(n_a, n_b + n_k - 1)$ because for increasing n_k additional states have to be introduced to realize an appropriate number of zero-order hold elements for the propagation of the delayed input.

Based on the goal of MPC (disturbance rejection, set-point regulation, reference tracking), a quadratic cost function incorporating future control inputs $\hat{u}_j(t+k|t)$ or future control increments $\Delta\hat{u}_j(t+k|t)$ with $0 \leq k \leq N_u - 1$ (computed at time t) up to the control horizon N_u is chosen. It will be minimized to deliver a sequence of optimal control increments

$$\Delta\hat{U}_j^* = \{\Delta\hat{u}_j(t|t), \Delta\hat{u}_j(t+1|t), \ldots, \Delta\hat{u}_j(t+N_u-1|t)\}$$

while constraints on the output $y_j(t)$ and input $u_j(t)$ are fulfilled at all times $t \geq 0$:

$$y_j^{min} \leq y_j(t) \leq y_j^{max} \tag{4.11}$$

$$u_j^{min} \leq u_j(t) \leq u_j^{max} \tag{4.12}$$

Since the output $y_j(t)$ and input $u_j(t)$ are both positions in terms of the HexaPOD, their limits are coherent with the operational range defined in Section 2.2.1.1.

The MPC makes use of the so-called receding horizon strategy. This means that the minimization of the cost function is carried out in each sampling instant t, but only the optimal control input $\hat{u}_j(t|t) = u_j(t-1) + \Delta\hat{u}_j(t|t)$ using the first element $\Delta\hat{u}_j(t|t)$ from the sequence $\Delta\hat{U}_j^*$ is applied to the physical system, thereby becoming the current control input $u_j(t)$. In the next sampling instant this control input becomes $u_j(t-1)$ which is needed for the calculation of next control input. The remaining control increments in the sequence $\Delta\hat{U}_j^*$ are ignored. However, they can be used in case of failures such as when the MPC problem is infeasible in the next sampling instant or when a sensor for breathing or tumor position was temporarily not able to perform a measurement.

Since lung tumor motion compensation is a reference tracking problem, the following cost function was chosen for this purpose:

$$J(t) = \frac{1}{2}\left(\sum_{k=1}^{N_p}\left(\hat{y}_j(t+k|t) - r_j(t+k|t)\right)^2 Q(k) + \sum_{k=0}^{N_u-1}\Delta\hat{u}_j(t+k|t)^2 T(k)\right). \tag{4.13}$$

The first sum in Eq. (4.13) penalizes deviations of the predicted system output $\hat{y}_j(t+k|t)$ at time t from the reference trajectory $r_j(t+k|t)$ up to the prediction horizon N_p in order to achieve the goal of reference tracking. The first sum in Eq. (4.13) penalizes deviations of the predicted system output $\hat{y}_j(t+k|t)$ at time t from the reference trajectory $r_j(t+k|t)$ up to the prediction horizon N_p in order to achieve the goal of reference tracking. The second sum keeps the control effort low as larger deviations from the control input at the previous sampling instant have a stronger influence on the cost function. Frequently the control input $\hat{u}_j(t+k|t)$ is used in the cost function instead of the control increments. Since the control input refers to a position rather than voltage or forces, using $\hat{u}_j(t+k|t)$ for the determination of control effort would result in an undesirable dependence of it from the location in the operational range of the HexaPOD. Thus, only the difference between two control inputs, i.e. the control increment, is useful in the definition of the cost function as it is independent from the location. For that reason, the formulation of model (4.10) was chosen, making the control increment $\Delta u_j(t)$ explicit. Both sums incorporate weighting factors $Q(k) > 0$ and $T(k) \geq 0$ to influence the impact of each sum to the value of the cost function.

Combining the state-space model (4.10), the constraints (4.11), (4.12) and the cost function (4.13), the optimization problem can be formulated as

$$\begin{aligned}
&\min_{\Delta\hat{U}_j \triangleq \{\Delta\hat{u}_j(t),\ldots,\Delta\hat{u}_j(t+N_u-1)\}} \quad \{J(t)\} \\
\text{s.t.}\quad & \hat{x}_j(t+k+1|t) = A\hat{x}_j(t+k|t) + B\left(\hat{u}_j(t+k-1|t) + \Delta\hat{u}_j(t+k|t)\right),\; k \geq 0, \\
& \hat{y}_j(t+k|t) = C\hat{x}_j(t+k|t),\; k \geq 0, \\
& \hat{x}_j(t|t) = x_j(t), \\
& \hat{u}_j(t-1|t) = u_j(t-1), \\
& \hat{u}_j(t+k|t) = \hat{u}_j(t+k-1|t),\; N_u \leq k < N_p, \\
& u_j^{min} \leq \hat{u}_j(t+k|t) \leq u_j^{max},\; k = 0,\ldots,N_p, \\
& y_j^{min} \leq \hat{y}_j(t+k|t) \leq y_j^{max},\; k = 1,\ldots,N_p\;.
\end{aligned} \tag{4.14}$$

The minimization of the cost function w.r.t. the sequence of control inputs in Eq. (4.14) is carried out obeying the constraints on the output and input defined by y_j^{min}, y_j^{max} and u_j^{min}, u_j^{max}, respectively. It is assumed that full knowledge of the state $x_j(t)$ in the current sampling instant t is available, represented by the constraint $\hat{x}_j(t|t) = x_j(t)$, which is needed as the start point in the optimization. If the state is not known completely, a state observer can be used to estimate the current state. The start point of the control input $\hat{u}_j(t-1|t)$ is set to the last control input $u_j(t-1)$ at the previous sampling instant which was applied to the system. Furthermore, the state-space model is directly included into the optimization problem in the form of two constraints. One of these results from the state equation of the model, which computes predictions of the state $\hat{x}_j(t+k+1|t)$ given future control increments, whereas the other originates from the output equation determining the predicted output $\hat{y}_j(t+k|t)$. Naturally, the control horizon N_u in the cost function cannot exceed the prediction horizon N_p, but it might be chosen smaller. So for $N_u \leq k < N_p$, the future control inputs $\hat{u}_j(t+k|t)$ are held constant resulting in the constraint $\hat{u}_j(t+k|t) = \hat{u}_j(t+k-1|t)$.

4.2.3.2 MPC Computation

In order to make the MPC problem (4.14) computable it has to be transformed into a quadratic program which is shown next.

The state-space model incorporated into the MPC problem can be iterated over multiple sampling instants starting from a known state $\hat{x}_j(t|t)$ and a known control input $\hat{u}_j(t-1|t)$ to calculate predictions of the state and of the output. This is expressed by the state propagation equation

$$\hat{x}_j(t+k|t) = A^k\hat{x}_j(t|t) + \sum_{h=0}^{k-1} A^{k-1-h}B\left(\hat{u}_j(t-1|t) + \sum_{i=0}^{h}\Delta\hat{u}_j^i(t+i|t)\right). \tag{4.15}$$

Using the output equation of the state-space model (4.10), the system output can be directly calculated by substituting the state with Eq. (4.15):

$$\hat{y}_j(t+k|t) = CA^k\hat{x}_j(t|t) + C\sum_{h=0}^{k-1} A^{k-1-h}B\left(\hat{u}_j(t-1|t) + \sum_{i=0}^{h}\Delta\hat{u}_j(t+i|t)\right). \tag{4.16}$$

Introducing the augmented vectors

$$\hat{Y}_j = \begin{bmatrix} \hat{y}_j(t+1|t) & \hat{y}_j(t+2|t) & \dots & \hat{y}_j(t+N_p|t) \end{bmatrix}^T , \tag{4.17}$$

$$R_j = \begin{bmatrix} r_j(t+1|t) & r_j(t+2|t) & \dots & r_j(t+N_p|t) \end{bmatrix}^T , \tag{4.18}$$

$$\Delta\hat{U}_j = \begin{bmatrix} \Delta\hat{u}_j(t|t) & \Delta\hat{u}_j(t+1|t) & \dots & \Delta\hat{u}_j(t+N_u-1|t) \end{bmatrix}^T \tag{4.19}$$

and applying Eq. (4.16), the cost function (4.13) can be rewritten in matrix form as

$$J(t) = \frac{1}{2}\left(\left(\hat{Y}_j - R_j\right)^T Q(\hat{Y}_j - R_j) + \Delta\hat{U}_j^T T \Delta\hat{U}_j\right) , \tag{4.20}$$

where Q and T are positive (semi-)definite matrices of proper dimensions and contain the weighting factors $Q(k)$ and $T(k)$ on their diagonals. Using Eq. (4.16) the output prediction vector $\hat{Y}_j$ can be formulated as

$$\hat{Y}_j = S\hat{x}_j(t|t) + V_u\hat{u}_j(t-1|t) + V_\Delta \Delta\hat{U}^j \tag{4.21}$$

with

$$S = \begin{bmatrix} CA \\ \vdots \\ CA^{N_p} \end{bmatrix}, V_u = \begin{bmatrix} CA^0B \\ \vdots \\ \sum_{h=0}^{N_p-1} CA^hB \end{bmatrix},$$

$$V_\Delta = \begin{bmatrix} CA^0B & \cdots & 0 \\ \vdots & \ddots & 0 \\ \sum_{h=0}^{N_p-1} CA^hB & \cdots & \sum_{h=0}^{N_p-N_u} CA^hB \end{bmatrix},$$

separating the predicted control increments $\Delta\hat{U}_j$ from the already known state $\hat{x}_j(t|t) = x_j(t)$ and last input $\hat{u}_j(t-1|t) = u_j(t-1)$. Defining

$$E_j(t) = R_j - S\hat{x}_j(t|t) - V_u\hat{u}_j(t-1|t) ,$$

the cost function (4.20) can be rewritten (omitting the argument t) to:

$$\begin{aligned} J(t) &= \frac{1}{2}\left(\left(V_\Delta\Delta\hat{U}_j - E_j\right)^T Q\left(V_\Delta\Delta\hat{U}_j - E_j\right) + \Delta\hat{U}_j{}^T T \Delta\hat{U}_j\right) \\ &= \frac{1}{2}\left(\left(\Delta\hat{U}_j^T V_\Delta^T - E_j^T\right) Q\left(V_\Delta\Delta\hat{U}_j - E_j\right) + \Delta\hat{U}_j^T T\Delta\hat{U}_j\right) \\ &= \frac{1}{2}\left(\underbrace{E_j^T Q E_j}_{\text{const}} + \Delta\hat{U}_j^T\left(V_\Delta^T Q V_\Delta + T\right)\Delta\hat{U}_j - 2\Delta\hat{U}_j^T V_\Delta^T Q E_j\right) . \end{aligned} \tag{4.22}$$

Since the first part in Eq. (4.22) is independent of $\Delta\hat{U}_j$, it is constant for the optimization with respect to $\Delta\hat{U}_j$. Thus, the reduced cost function

$$\tilde{J}(t) = \frac{1}{2}\Delta\hat{U}_j^T G \Delta\hat{U}_j + \Delta\hat{U}_j^T g_0 \tag{4.23}$$

can be used instead for the optimization with

$$\begin{aligned} G &= V_\Delta^T Q V_\Delta + T \ , \\ g_j(t) &= -V_\Delta^T Q E_j \\ &= V_\Delta^T Q S \hat{x}_j(t|t) + V_\Delta^T Q V_u \hat{u}_j(t-1|t) - V_\Delta^T Q R_j \ . \end{aligned}$$

The matrix G is constant for a time-invariant state-space model and constant weighting matrices Q and T and can be precomputed for the solution of the MPC problem if these values are not changed during runtime. $g_j(t)$ depends on the current state $\hat{x}_j(t|t)$, the last control input $\hat{u}_j(t-1|t)$ and the future reference trajectory R_j. Thus, it must be recomputed in each sampling instant. With all these computations, the state-space model is now directly incorporated into the reduced cost function (4.23) of the optimization problem, rather than still present as part of the constraints. In that way, no explicit computations need to be made for the future system states or outputs. Instead, these are resolved in the cost function.

During the optimization process using the reduced cost function the constraints on the input and output must still be obeyed. The output constraint

$$y_j^{min} \leq \hat{y}_j(t+k|t) \leq y_j^{max}$$

must be fulfilled for all output predictions over the whole prediction horizon for $k = 1, \ldots, N_p$. Using Eq. (4.17) these constraints can be combined into matrix form

$$\underbrace{\begin{bmatrix} -1 & 0 & 0 & \cdots & 0 \\ 1 & 0 & 0 & \cdots & 0 \\ 0 & -1 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \cdots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & \cdots & -1 \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}}_{C_y \in \mathbb{R}^{2N_p \times N_p}} \hat{Y}_j \leq \underbrace{\begin{bmatrix} -y_j^{min} \\ y_j^{max} \\ -y_j^{min} \\ y_j^{max} \\ \vdots \\ -y_j^{min} \\ y_j^{max} \end{bmatrix}}_{Y_j^{lim} \in \mathbb{R}^{2N_p}} . \tag{4.24}$$

Inserting Eq. (4.21) into Eq. (4.24) leads to

$$C_y \left(S\hat{x}_j(t|t) + V_u \hat{u}_j(t-1|t) + V_\Delta \Delta\hat{U}_j \right) \leq Y_j^{lim},$$

which can be rearranged to

$$\underbrace{-C_y V_\Delta}_{C_{y\Delta} \in \mathbb{R}^{2N_p \times N_u}} \ \Delta\hat{U}_j \underbrace{-C_y S \hat{x}_j(t|t) - C_y V_u \hat{u}_j(t-1|t) + Y_j^{lim}}_{C_{yj} \in \mathbb{R}^{2N_p}} \geq 0, \tag{4.25}$$

where $C_{y\Delta}$ is multiplied with the predicted control increments $\Delta\hat{U}_j$ and C_{yj} is constant in the optimization problem as it is independent of $\Delta\hat{U}_j$. For convenience Eq. (4.25) is rewritten as

$$C_{y\Delta} \Delta\hat{U}_j + C_{yj} \geq 0 \tag{4.26}$$

which describes all constraints on the predicted outputs over the prediction horizons.

The constraints on the control inputs can be handled in a similar way. Here, the input constraint

$$u_j^{min} \leq \hat{u}_j(t+k|t) \leq u_j^{max}$$

must be fulfilled for all future control inputs over the whole control horizon for $k = 0, \ldots, N_u - 1$ which can be written in matrix form

$$\begin{bmatrix} -1 & 0 & 0 & \cdots & 0 \\ 1 & 0 & 0 & \cdots & 0 \\ 0 & -1 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \cdots & \ddots & \ddots & 0 \\ 0 & 0 & 0 & \cdots & -1 \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \hat{U}_j \leq \underbrace{\begin{bmatrix} -u_j^{min} \\ u_j^{max} \\ -u_j^{min} \\ u_j^{max} \\ \vdots \\ -u_j^{min} \\ u_j^{max} \end{bmatrix}}_{U_j^{lim} \in \mathbb{R}^{2N_u}} \tag{4.27}$$

with

$$\hat{U}_j = \begin{bmatrix} \hat{u}_j(t|t) & \hat{u}_j(t+1|t) & \ldots & \hat{u}_j(t+N_u-1|t) \end{bmatrix}^T .$$

$\hat{U}_j$ can be formulated in terms of $\Delta\hat{U}_j$:

$$\hat{U}_j = \begin{bmatrix} \hat{u}_j(t-1|t) \\ \hat{u}_j(t-1|t) \\ \vdots \\ \hat{u}_j(t-1|t) \end{bmatrix} + \begin{bmatrix} \sum_{k=0}^{0} \Delta\hat{u}_j(t+k|t) \\ \sum_{k=0}^{1} \Delta\hat{u}_j(t+k|t) \\ \vdots \\ \sum_{k=0}^{N_u-1} \Delta\hat{u}_j(t+k|t) \end{bmatrix} = \begin{bmatrix} \hat{u}_j(t-1|t) \\ \hat{u}_j(t-1|t) \\ \vdots \\ \hat{u}_j(t-1|t) \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & & \vdots \\ 1 & 1 & 1 & \cdots & 1 \end{bmatrix} \Delta\hat{U}_j. \tag{4.28}$$

Inserting Eq. (4.28) into Eq. (4.27) leads to

$$\underbrace{\begin{bmatrix} -1 & 0 & 0 & \cdots & 0 \\ 1 & 0 & 0 & \cdots & 0 \\ -1 & -1 & 0 & \cdots & 0 \\ 1 & 1 & 0 & \cdots & 0 \\ \vdots & \cdots & \ddots & \ddots & 0 \\ -1 & -1 & -1 & \cdots & -1 \\ 1 & 1 & 1 & \cdots & 1 \end{bmatrix}}_{C_{u_0} \in \mathbb{R}^{2N_u \times N_u}} \Delta\hat{U}_j + \underbrace{\begin{bmatrix} -\hat{u}_j(t-1|t) \\ \hat{u}_j(t-1|t) \\ -\hat{u}_j(t-1|t) \\ \hat{u}_j(t-1|t) \\ \vdots \\ -\hat{u}_j(t-1|t) \\ \hat{u}_j(t-1|t) \end{bmatrix} - U_j^{lim}}_{C_{u_1} \in \mathbb{R}^{2N_u}} \leq 0 . \tag{4.29}$$

With $C_{u\Delta} = -C_{u_0}$ and $C_{uj} = -C_{u_1}$, Eq. (4.29) becomes

$$C_{u\Delta}\Delta\hat{U}_j + C_{uj} \geq 0$$

which has the same form as the output constraints described by Eq. (4.26). Thus, both constraints can be combined into the inequality

$$\underbrace{\begin{bmatrix} C_{u\Delta} \\ C_{y\Delta} \end{bmatrix}}_{C_\Delta \in \mathbb{R}^{2(N_p+N_u) \times N_u}} \Delta\hat{U}_j + \underbrace{\begin{bmatrix} C_{uj} \\ C_{yj} \end{bmatrix}}_{C_j \in \mathbb{R}^{2(N_p+N_u)}} \geq 0. \tag{4.30}$$

C_Δ is only depending on constant parameters (for an underlying time-invariant state-space model) whereas C_j needs to be recalculated in each sampling instant from known values such as $\hat{x}_j(t|t)$ and $\hat{u}_j(t-1|t)$.

Eqn. (4.23) and (4.30) represent a strictly convex quadratic program (QP) with a set of inequality constraints and no equality constraints. This can be solved online through interior point methods or active-set dual methods ([154]), thus delivering the optimal control increments $\Delta\hat{U}_j$, where the first one is used to determine the control input which is sent to the HexaPOD.

4.2.3.3 Experimental Tests and Results

The goal of the following experimental tests was to determine useful ranges of the free parameters within the IMPC approach. The setup for the experimental tests was as follows: The Polaris IR camera measured the position of two calibrated IR tools, both attached rigidly to the HexaPOD. One of the tools was used to acquire the actual position of the HexaPOD. The measurements of the other one were modified by the simulator program (cf. Section 2.2.1.6) which overlays predefined trajectories to simulate a breathing patient. The software package *HexGuide* was configured to predict that breathing motion using the Takens predictor. The prediction horizon k in the predictor was chosen equal to the prediction horizon N_p used in MPC. Predictions along each translational axis were negated at each sampling instant to deliver the reference trajectory R_j (cf. Eq. (4.18)) for axis j. The system is hardware-triggered at a rate of 20Hz. All considerations in terms of specific values of horizon parameters have to be seen w.r.t. the corresponding sampling time of 50ms.

For each translational axis, a separate instance of the MPC was executed. The derived control input from each instance, $u_j(t) = u_j(t-1) + \Delta\hat{u}_j(t|t)$ was applied to the HexaPOD by sending the appropriate command where $u_j(t)$ is the position for the corresponding axis in the command.

For these experiments, the ARX model along one translational axis of the HexaPOD was derived as discussed in Section 4.1.1. The order of the ARX model was chosen as $(2, 2, 1)$, leading to a second-order state-space model. Note that this identified model of the HexaPOD is of non-minimum phase as a zero is located outside of the unit circle in the z-plain. Thus, the MPC is not always feasible for prediction horizon $N_p = 1$. Hence, $N_p = 1$ was excluded from the experimental tests. Since only the first state of the two states in $x_j(t)$ is directly measurable, an observer is employed to estimate the other state.

The cost function incorporates two horizon parameters N_p and N_u which have an impact on the performance. For optimization, the absolute value of the cost function is not of interest, so that $Q = I_{N_p}$ (identity matrix) is fixed and T, the influence of the control increments, is used as tuning parameter.

For all the free parameters N_p, N_u and T, exhaustive tests in a defined parameter range were run on four different "breathing" trajectories (annotated with maximum speed): three sines (±3mm, 0.2Hz ($3.77\frac{\text{mm}}{\text{s}}$); ±5mm, 0.2Hz ($6.28\frac{\text{mm}}{\text{s}}$); ±2.5mm, 0.4Hz ($6.28\frac{\text{mm}}{\text{s}}$)) and one recorded patient dataset ($> 8\frac{\text{mm}}{\text{s}}$). Note that the patient dataset already exceeds the maximum speed of the HexaPOD in parts of each breathing cycle. The last two sines are also close to the maximum speed whereas the first sine is an example of a slow dataset compared to the maximum speed.

Since the impact of the parameters on the controller performance is the objective in these following tests, the RMS metric was considered representative of the tracking accuracy metric and the smoothness metric S_F was used to assess the smoothness.

Horizon Parameters Figure 4.7 depicts the RMS of the tracking error of combinations (N_p, N_u) up to 10 (corresponds to 0.5 seconds) of all four test datasets (with $T = 0.1$). For each dataset the RMS errors are located in a narrow band (except for $N_p = 2$) where the "height" of the band or the average RMS error correlates with the maximum speed of the dataset in such a way that a slower maximum speed results in a smaller RMS error. An inspection of individual trials (not shown here) reveals that the maximum errors either occur in the vicinity of peaks (for slower datasets) or (for faster datasets) in regions with too high velocities where reference tracking is constrained by the maximum speed of the HexaPOD. For each dataset an increasing N_p (irrespective of N_u) generally improves tracking error because with an increased prediction horizon the IMPC can adjust the current control input with respect to future behavior of the reference. The RMS errors for $N_p \geq 7$ show only a minor difference, so that an increase above 10 does not provide a justifiable advantage in contrast to a higher computational load.

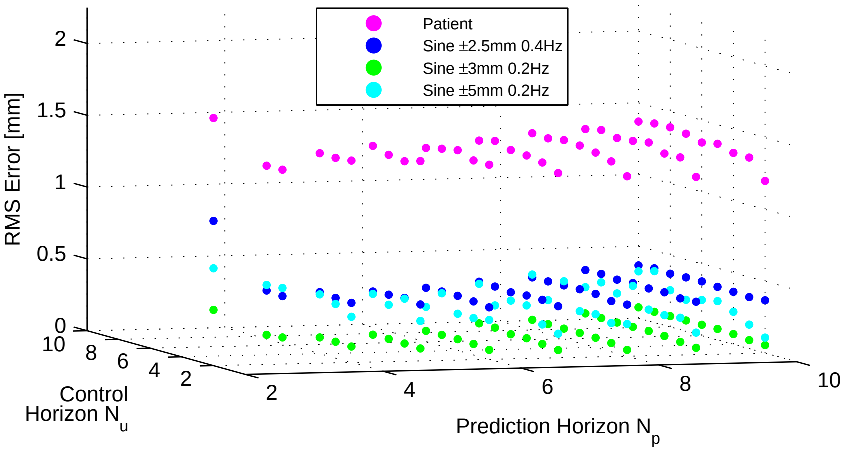


Figure 4.7: *RMS errors of four test datasets at various settings of horizon parameters N_p and N_u of the IMPC scheme*

N_u plays only a minor role as the RMS errors of different settings of N_u for one dataset lie closely together. However, one can observe a slight reduction of the RMS errors for smaller values of N_u. In order to find a useful range of N_u, Figure 4.8 shows the occurrence of the 10 best combinations (N_p,N_u) in terms of smallest RMS tracking error (note that each combination can occur at maximum four times). The most frequent combinations are clearly observed for $N_p \geq 7$ and $2 \leq N_u \leq 3$. Larger values for N_u can also be used (cf. Figure 4.7), but then a slight increase of the tracking error occurs.

Control Increment Weighting The influence of control increment weighting T was studied on the best combination of (N_p, N_u) from the previous test for each dataset. For almost all datasets, it can be observed in Figure 4.9(a) that with rising T, the error norm reaches a minimum and then slightly increases again. The increase of the error norm at higher values of T is caused by the stronger weighting of the control increments in contrast to the tracking error contribution inside the cost function, thus decreasing the tracking ability. The location of the minimum and the slope of the increase depends on the maximum speed

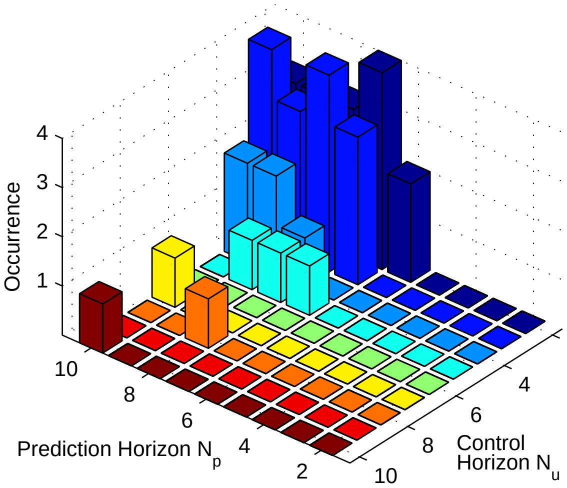
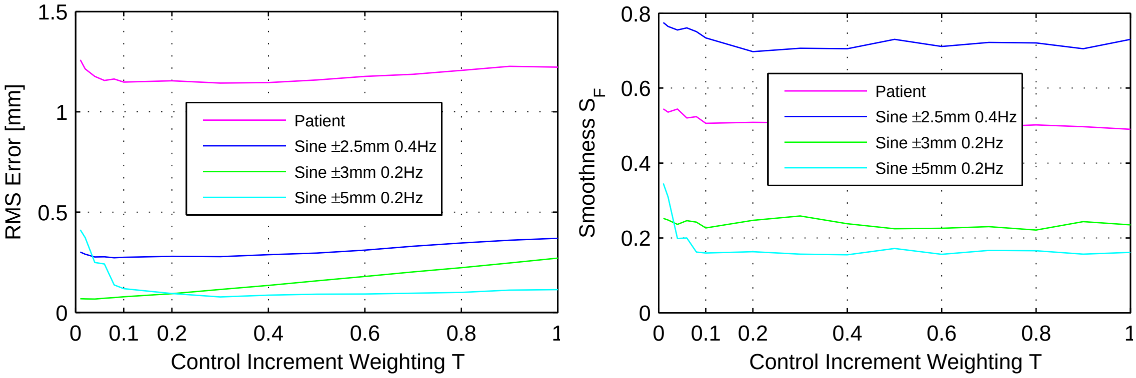


Figure 4.8: *Occurrence of 10 best settings of horizon parameters among four test datasets of the IMPC scheme*



(a) Influence on RMS of tracking error

(b) Influence on smoothness S_F

Figure 4.9: *Influence of control increment weighting T on tracking error and smoothness in the IMPC scheme*

of the dataset. For slower datasets, the minimum can be found at smaller values, but the slope of increase is slightly larger. For faster datasets, the opposite is true. However, $T = 0.1$ is a good compromise for all four datasets.

A rising T also has the effect of smoothing the control inputs by penalizing larger deviations from the last control input. With small values of T, one can actually hear the HexaPOD changing direction several times in a short interval (this occurs especially in the vicinity of peaks) and these sharp changes dramatically decrease patient comfort. Therefore, smoothness of the control input is an important aspect of the system. Figure 4.9(b) gives an impression of the smoothness metric S_F of the resulting HexaPOD's trajectory in terms of the control increment weighting T. With increasing T, smoothness improves until it reaches an almost constant value at about $T = 0.1$. Similar observations can be made using other smoothness metrics as well. Generally, any of the tracking accuracy metrics and smoothness metrics can be used to trade-off a value for T, taking into account patient comfort, too.

For $T = 0.1$, the maximum errors and the maximum errors in relation to the amplitude of the best (N_p, N_u) of each dataset are: sine ±3mm: 0.19mm (3.2%), sine ±5mm: 0.29mm (2.9%), sine ±2.5mm: 0.52mm (10.5%), patient: 3.49mm (35.2%). Judging from the error norms and maximum errors, a significant compensation of motion can be achieved using the IMPC approach.

These results are published as well in similar form in [155]. It is interesting to note in this context, that within the same session on the same conference, [156] published their results with a different MPC approach using the standard Elekta table stand for motion compensation in radiotherapy.

The results obtained demonstrate the general feasibility of MPC in combination with a model obtained by system identification methods to achieve motion compensation using the HexaPOD. Further results of this method will be given in the next section, where the results of the MPC approach are compared to a different controller.

4.2.4 Velocity-Based Position Control

As mentioned in Section 2.2.1.1, the HexaPOD can be controlled by positioning commands sent through a serial connection. Each command consists of a 6D-position within the operational range and a normalized speed specification $s_c(t)$. In Section 4.1.3 it was stated that no conclusive relationship between $s_c(t)$ and the response of the HexaPOD could be found. However, it is assumed for the following considerations, that the allowed range of $s_c(t)$ between 0 and 1 maps linearly into the range between 0 and the maximum speed v_{max} of the HexaPOD. The consequence of this assumption is that the response to step inputs can still not be described in terms of $s_c(t)$, but this is anyway not the intended purpose. For tumor motion compensation it is rather important to allow reference tracking, requiring permanent adaptation of the speed specification $s_c(t)$. Essentially, this summarizes the main idea of the velocity-based position control (VPC) scheme, in which $s_c(t)$ is computed in such a way that the HexaPOD exhibits appropriate velocities to match those on the reference trajectory.

Given a momentary speed $s(t) > 0$ along the reference trajectory, the appropriate $s_c(t)$ in the HexaPOD's commands is derived by

$$s_c(t) = \frac{1}{v_{max}} \underset{v_{max}}{\operatorname{sat}} \left(s(t)\right) \ , \tag{4.31}$$

where

$$\underset{v_{max}}{\operatorname{sat}} \left(s(t)\right) = \begin{cases} |s(t)| & \text{if } |s(t)| \leq v_{max} \\ v_{max} & \text{if } |s(t)| > v_{max} \end{cases} \ , \tag{4.32}$$

constrains the speed $s(t)$ to the maximum speed v_{max}. This actuator saturation renders the controller nonlinear.

For the following considerations, it is assumed for now that the HexaPOD shall be moved along a single axis j only. For reference tracking the goal of a controller is to adjust the actual speed of the HexaPOD so that the desired position at the next sampling instant is reached (if the maximum speed limit is not already reached). Since the speed specification does not carry direction information, the upper limit l_j^U or the lower limit l_j^L of the operational range for axis j can be used in the 6D-position of the command for guiding the HexaPOD in the right direction. Given the current position $p_j(t)$ along axis j at sampling instant t and the desired position at the next sampling instant $p_j(t+1) = r_j(t+1|t)$, the position $c_j(t)$ for axis j in the command can be determined as

$$l_j(t) = \begin{cases} l_j^U & \text{if } p_j(t+1) \geq p_j(t) \\ l_j^L & \text{if } p_j(t+1) < p_j(t) \end{cases} , \tag{4.33}$$

so that $c_j(t) = l_j(t)$. Note that the choice for $c_j(t)$ has to be made in such a way to keep a certain minimum distance to the reference. If the actual position of the HexaPOD comes too close to the desired one specified in the command, the HexaPOD either slows down or even stops at the desired position, preventing the HexaPOD from maintaining the necessary speed on the reference trajectory (unless the reference trajectory becomes constant in which case the HexaPOD should stop). Choosing the operational limit always guarantees the constraint on distance. The speed $s(t)$ has to be calculated in such a way that using the remaining time $\Delta\tau$ (the time difference between the next sampling instant and the current time captured right before this calculation, see also the latency analysis in Section 2.2.3.3)

$$s(t) = \frac{|p_j(t+1) - p_j(t)|}{\Delta\tau} \tag{4.34}$$

is the absolute value of the linear speed which has to be linearly normalized into the range 0 and 1 by Eq. (4.31). The normalized value $s_c(t)$ is then sent together with the 6D position to the HexaPOD as a new command. The case when only motions along axis j are used but currently no motion is required, i.e. $p_j(t+1) = p_j(t)$, has to be handled separately. Then $s(t)$ in Eq. (4.34) would become 0, but this would be rejected by the HexaPOD with an error response. Instead, $s(t)$ has to be set to the minimum allowed speed of 1.008µm/s and the position for axis j in the command is one of the limits of the operational range; here l_j^U was already chosen in Eq. (4.33). This procedure results in a very slow movement of the HexaPOD towards l_j^U. For this application the traveled distance with this slow speed is very small and can be neglected since this condition should last only for a few moments, if at all, and the minimum speed is more than two magnitudes smaller than in normal operation mode.

The approach discussed so far is applicable to one axis at a time as only one speed specification is provided in the HexaPOD's command. For realizing reference tracking along two or all three translational axes simultaneously, the only solution is to choose one speed and somehow adapt the position specifications in the command. The basic idea is to calculate a 3D-vector

$$\Delta\vec{p}(t) = \vec{p}(t+1) - \vec{p}(t) = \begin{bmatrix} \Delta p_1(t) \\ \Delta p_2(t) \\ \Delta p_3(t) \end{bmatrix}$$

specifying the movement direction of the HexaPOD from the current position $\vec{p}(t)$ and the desired position $\vec{p}(t+1)$ in the next sampling instant $t+1$. If $\Delta\vec{p}(t)$, starting from $\vec{p}(t)$, is extended, it will at some point $\vec{c}(t)$ intersect with exactly one of the six limiting planes of the cuboid spanned by the upper limits l_1^U, l_2^U, l_3^U and lower limits l_1^L, l_2^L, l_3^L of the operational range along all three translational axes. This situation is exemplified in Figure 4.10. From the plane of intersection, the axis with the most significant motion, the *main* axis, can be determined as the axis where its upper or lower limit was hit by the extended vector. Then, projections of the intersection point on each axis are calculated. These projections are directly used in the command for the HexaPOD as c_j with $j = 1, 2, 3$.

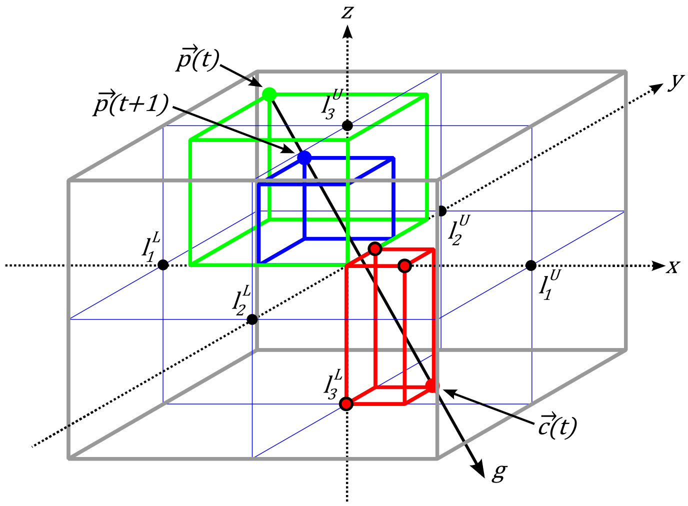


Figure 4.10: *Geometric representation of position control approach*

For calculating the intersection point $\vec{c}(t)$, it is convenient to introduce the vector form of a straight line

$$g: \quad \vec{a} = \vec{p}(t) + K\,\Delta\vec{p}(t) \tag{4.35}$$

with $\vec{a} \in \mathbb{R}^3$ representing a point on the line for a certain scalar $K \in \mathbb{R}$. Finding the plane of intersection can be simplified by using Eq. (4.33) as this takes into account the direction along each axis, so that only three planes need to be checked for intersection instead of six. Let the planes P_j with $j = 1, 2, 3$ be given by the Hesse normal form

$$P_j: \quad \vec{a}\,\vec{i}_j - l_j(t) = 0 \tag{4.36}$$

where $\vec{a} \in \mathbb{R}^3$ is a point on the plane and $\vec{i}_j$ is the j-th unit vector used as a normal vector of the plane. Substituting Eq. (4.35) into Eq. (4.36), solving for K and inserting that value in Eq. (4.35) determines the point of intersection of g with P_j. Let K_j denote the resulting K when calculating the intersection of g with P_j, then

$$K_j = \frac{l_j(t) - p_j(t)}{\Delta p_j(t)}\ . \tag{4.37}$$

If $\Delta p_j(t) = 0$ in Eq. (4.37) then g is parallel to the plane P_j and no intersection occurs, i.e. no motion should occur along axis j. This case has to be handled as a special case in a similar way as for the considerations for only one axis above. Assuming that $\Delta p_j(t) \neq 0$ for $j = 1, 2, 3$, then three K_j exist, each of them representing an intersection with the plane P_j, either inside or outside the cuboid defined by the operational range. Since the movement direction for each axis is already covered in Eq. (4.33), K_j is always positive. As the cuboid is convex and both points $\vec{p}(t)$ and $\vec{p}(t+1)$ are located inside, g can only intersect one of the planes P_j. The plane P_m of intersection is determined by the minimum of K_j:

$$m = \arg\min_{j \in \{1,2,3\}} K_j\ .$$

Inserting K_m into Eq. (4.35), the point of intersection $\vec{c}(t)$ can finally be calculated by

$$\vec{c}(t) = \begin{bmatrix} c_1(t) \\ c_2(t) \\ c_3(t) \end{bmatrix} = \vec{p}(t) + K_m \, \Delta\vec{p}(t) \ ,$$

where its projections on the translational axes are depicted in Figure 4.10 as filled circles in red with black borders. These are sent to the HexaPOD as part of the MOV command.

For the determination of the speed in the command, two possible methods were identified which the HexaPOD could use to reconstruct the desired velocities along each axis from the information given in a single command:

1. Speed is to be specified as linear speed along the path to travel:

$$s(t) = \frac{\|\Delta\vec{p}(t)\|}{\Delta\tau} = \frac{\sqrt{\Delta p_1(t)^2 + \Delta p_2(t)^2 + \Delta p_3(t)^2}}{\Delta\tau} \tag{4.38}$$

   The assumption on the behavior of the internal controller of the HexaPOD would be that the desired velocities along each axis are reconstructed from the specified speed in the command and from the ratio of the corresponding element of $\Delta\vec{p}(t)$ and the length of $\Delta\vec{p}(t)$.

2. The speed $s(t)$ is given as the desired speed along the main axis m:

$$s(t) = \frac{|\Delta p_m(t)|}{\Delta\tau} \tag{4.39}$$

   Then, the controller of the HexaPOD could reconstruct the desired velocities along each axis from the speed $s(t)$ and from the ratio of the projection for the corresponding axis to the projection for the main axis.

The main difference between these two methods is that with method 1 the observable speed along each axis might be less than v_{max} if more than one axis is actuated. Let all three elements of $\Delta\vec{p}$ (implicitly depending on time) be of the same magnitude $d = \Delta p_1 = \Delta p_2 = \Delta p_3$, i.e. the difference between current and desired position is the same along each axis, and let them be just as large to result in $s = v_{max}$, then the velocity along each axis j is reconstructed in the internal controller as

$$v_j = \frac{d}{|\Delta\vec{p}|}s = \frac{d}{\sqrt{3d^2}}s = \pm\frac{1}{\sqrt{3}}s = \pm\frac{1}{\sqrt{3}}v_{max} \approx \pm 0.5774 v_{max} \ .$$

This means that if the internal controller works with method 1, the maximum achievable speed along each axis when actuating all three translational axes in the same way is constrained to less than $0.5774v_{max}$. If the controller uses method 2, then the choice of the main axis is arbitrary in this situation with $s = v_{max}$ and the reconstruction of the velocity along each axis j in the controller is

$$v_j = \frac{d}{|d|}s = \pm v_{max}.$$

Thus, the HexaPOD would be able to exhibit a speed of v_{max} along each axis. The speed reconstruction method the controller follows can be determined experimentally by using a test signal with maximum speeds between $0.5774v_{max}$ and v_{max}. If the controller uses method 2, reference tracking with this test signal should be successful; otherwise a saturation of the speed would be observable and a deviation from the reference would occur.

Irrespective of the speed reconstruction method, a speed gain λ with $0 < \lambda \leq 1$ is introduced into Eq. (4.31) to influence the control effort, e.g. for adjusting the aggressiveness of the controller:

$$s_c(t) = \frac{1}{v_{max}} \underset{v_{max}}{\operatorname{sat}} \left(\lambda s(t)\right) \ , \tag{4.40}$$

which trades-off tracking accuracy and patient comfort.

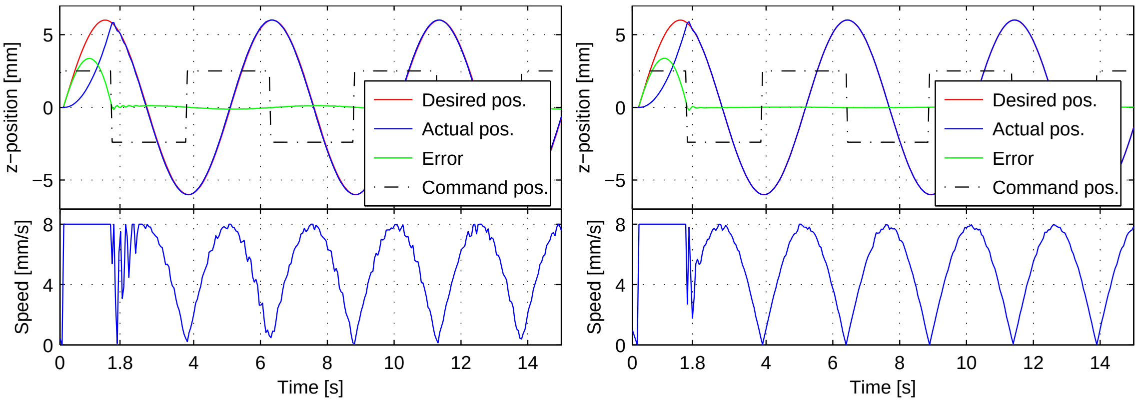
In this approach, position control is achieved by deriving the velocity specification in the HexaPOD's command from the desired velocity along the reference trajectory and appropriately adapting the position specifications in the command. Hence, this approach is further referred to as velocity-based position control (VPC).

4.2.4.1 Experimental Tests and Results

Several experimental tests were conducted to assess the performance of the VPC approach. With the first experiment, the velocity reconstruction method is determined which will be used for the following experiments. The second experiment demonstrates the influence of speed gain λ on tracking performance, so that adequate parameter values can be found. The following three experiments compare control performance between IMPC and VPC approaches using several metrics in three distinct scenarios: ideal sinusoidal trajectories, pre-recorded breathing trajectories and human volunteers lying on the HexaPOD whose breathing motion was compensated.

Experiment 1: Velocity Reconstruction in VPC The first experiment aims at determining which method the HexaPOD uses for reconstructing the desired velocities along each axis, given a certain command for the velocity-based position control approach. To determine this method, a predetermined breathing trajectory in the form of a sinusoidal signal with a frequency of 0.2Hz and an amplitude of ±6mm was injected into the system using $\lambda = 1$. The sinusoidal signal (possessing a maximum speed of about 7.54mm/s which exceeds the required limit of about $0.5774v_{max} \approx 4.62$mm/s but is still lower than the maximum of $v_{max} = 8$mm/s) was used as the desired position along all translational axes simultaneously. The reconstruction method can be determined by inspection of the tracking error. If the controller uses method 2, a minor tracking error would result, whereas with method 1, a significant tracking error would be observable as the maximum achievable velocity along three axis simultaneously would be constrained to a maximum of 4.62mm/s.

The results for the decision of the type of velocity reconstruction for the position control approach is depicted in Figures 4.11(a) and 4.11(b) along the z-axis for methods 1 and 2, respectively. For the x- and y-axes, similar plots are obtained. In the upper part of both figures, the desired position (red) is shown as the sinusoidal signal with ±6mm and 0.2Hz. Also, the actual position (blue) of the HexaPOD can be seen which approaches the desired position after a short transition phase with a duration of about 1.8s. After that, the controller keeps tracking the desired position and enters a steady state. When applying method 1, Figure 4.11(a) shows a maximum periodic tracking error of 0.122mm. For method 2, the tracking error observed in Figure 4.11(b) is less than 0.022mm, which



(a) Application of reconstruction method 1 leading to maximum steady state tracking error of 0.122mm where actual position precedes desired position

(b) Application of reconstruction method 2 leading to maximum steady state tracking error of 0.022mm

Figure 4.11: *Results of velocity reconstruction methods: Upper part: Actual position of the HexaPOD (blue) reaches desired position (red) after transition phase with a steady state tracking error (green); commanded position (scaled by 1/10) and speed (lower part) form the control input sent to the HexaPOD*

is well within the measurement noise range of the IR camera. Judging from these tracking errors, the controller clearly uses method 2 for velocity reconstruction. Otherwise larger tracking errors caused by the potential velocity limitation to 4.62mm/s would have resulted. Furthermore, it is interesting to note that the actual position for method 1 precedes the desired position. Because the internal controller uses method 2 for velocity reconstruction but the commands are determined with method 1, the speed $s_c(t)$ originating from Eq. (4.38) is always too large. Hence, a phase shift between actual and desired position occurs. This can also be seen by comparing the lower parts of Figures 4.11(a) and 4.11(b), showing the speed $s(t)$. For method 1, the speed profile contains more high frequency oscillations than for method 2. These result from the closed-loop behavior when the controller tries to reduce the tracking error in presence of the too large speed $s(t)$.

The control input in VPC is split up into a commanded position part (shown as the dash-dot line in the upper parts of the figures and scaled by 1/10 to fit into the diagram) and a speed part (lower part of the figures). This additionally demonstrates the working principle of the controller. The commanded position only takes minimum and maximum values of the operational range along the z-axis and thus defines the movement direction while speed controls the position to be reached.

For the remaining experiments, velocity reconstruction method 2 was used for the VPC approach.

Experiment 2: Influence of Speed Gain λ The second experiment determines the influence of speed gain λ introduced in Eq. (4.40). For that purpose, λ was varied inside its allowed range $]0;1]$ starting from 0.1 in steps of 0.1. Several pre-calculated sinusoidal trajectories covering slow to fast motions with small and large amplitudes as well as sinusoidal trajectories which exceed the maximum speed of 8mm/s were used for each value of λ. The following sines were selected with maximum speeds given in parentheses: ±3mm with 0.2Hz (3.77mm/s), ±5mm with 0.2Hz (6.28mm/s), ±2.5mm with 0.4Hz (6.28mm/s), ±7mm with 0.2Hz (8.80mm/s) and ±5mm with 0.3Hz (9.42mm/s). These sinusoidal trajectories were injected into the system as desired positions for the HexaPOD along all three translational axes simultaneously. The evaluation was performed with respect to the maximum absolute tracking error to deliver the worst case performance.

The speed gain λ in Eq. (4.40) influences the control effort by reducing the speed in commands which would be necessary to achieve best possible tracking. This effects an increasing maximum tracking error with decreasing λ as seen in Figure 4.12. The determination of the maximum tracking error started after a transition time of 2.5s after which the transient behavior (cf. Figure 4.11) is almost zero for all used datasets. The advantage of a lower control effort is the ability to avoid small overshoots of the controller which are caused by unknown timing variations in the system, measurement noise and non-smoothness of predicted tumor positions. Smaller values of λ help to reduce frequent direction changes of the HexaPOD which can degrade patient comfort.

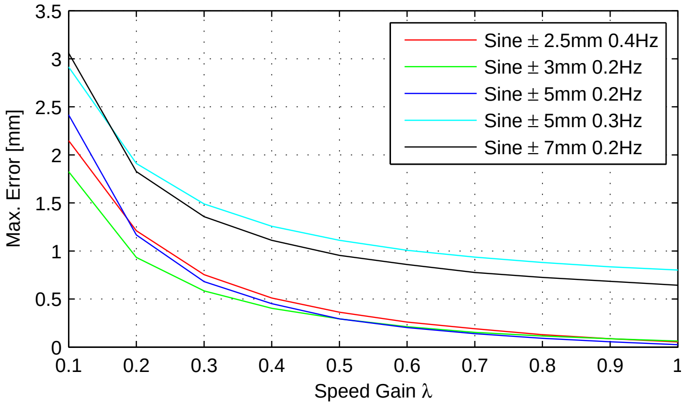


Figure 4.12: *Maximum tracking errors for several ideal sine reference trajectories for different speed gains* λ *between* 0 *and* 1 *in steps of* 0.1

A strong dependency between the maximum tracking error and maximum speed of the desired trajectory can also be observed from Figure 4.12. For the sine trajectories below 8mm/s (±2.5mm with 0.4Hz, ±3mm with 0.2Hz and ±5mm with 0.2Hz), the maximum errors are located in the same region and become very small for $\lambda = 1$. The maximum errors for faster trajectories lie apart from the slower ones. Here, the maximum error is determined by the amount and duration of speed exceedance of the trajectory compared to the HexaPOD's speed limit. In this case, the HexaPOD cannot keep the same tracking performance but still tries to compensate as much as possible.

Independent of the maximum speed, the shapes of the exponentially decreasing curves of the maximum tracking error are similar, where the slope decreases with increasing λ. For $\lambda > 0.5$, the slope decreased in such a way, that differences to λ near one are lesser compared to smaller values for λ. Therefore choosing $\lambda > 0.5$ seems reasonable.

Experiment 3: IMPC vs. VPC with Ideal Trajectories In order to get a clear picture of the performance of IMPC in comparison to VPC, experiment 3 was carried out on ideal trajectories using the sines from experiment 2. Also here, the trajectories were injected through the simulator program which overlays them on the actual IR measurements along all three translational axes. For IMPC, *HexGuide* performed a prediction of the trajectory up to a horizon of 7 (because of the prediction horizon of $N_p = 7$) whereas for the position control approach, a horizon of 1 is sufficient. Note that the prediction algorithm was configured to deliver a multi-step prediction, i.e. in sampling instant t all predictions for $t + 1$ up to $t + N_p$ are calculated, so that the most up-to-date information is used for predictions.

The evaluation aims at a comparison of IMPC and VPC under almost ideal circumstances with respect to important metrics of the tracking error such as compensation ratio (CR), 95% confidence interval (CI95), mean absolute error (MAE), root-mean-square (RMS) error and maximum absolute error (MAX). Table 4.5 shows the results of these metrics of the tracking error using ideal sine trajectories as desired positions for both IMPC and VPC (with $\lambda = 1$). Although IMPC exhibits an acceptable performance, especially for slower trajectories, VPC outperforms it on all metrics. The maximum tracking error is reduced by an order of magnitude to almost zero for trajectories below the speed limit of the HexaPOD. This performance is further emphasized by the fact that even the maximum errors are within the measurement noise range of the IR camera. Consequently, CI95, MAE and RMS errors are also significantly smaller for VPC. A value of 100% for CR would result in perfect tracking with no tracking errors, which is almost reached for VPC on the slower trajectories. However, in the presence of measurement errors, this value cannot be reached, so the achieved CRs can be seen as near perfect. Also for faster trajectories, VPC possesses a significantly larger CR. In general, VPC can follow the trajectories much closer than IMPC. Since the ideal input trajectories are smooth, no reduction of λ was needed here to trade-off patient comfort. The resulting trajectories of the HexaPOD in this experiment were smooth, as seen quantitatively from Figure 4.11 and heard qualitatively from the noises of the HexaPOD's motors during motion.

	max.	IMPC					VPC				
Dataset	speed	CR	CI95	MAE	RMS	MAX	CR	CI95	MAE	RMS	MAX
	[mm/s]	[%]	[mm]	[mm]	[mm]	[mm]	[%]	[mm]	[mm]	[mm]	[mm]
Sine ±3mm 0.2Hz	3.77	96.2	0.145	0.079	0.088	0.193	98.3	0.050	0.032	0.035	0.052
Sine ±5mm 0.2Hz	6.28	90.2	0.709	0.273	0.347	0.841	99.7	0.015	0.008	0.009	0.024
Sine ±2.5mm 0.4Hz	6.28	79.9	0.692	0.297	0.360	0.841	98.7	0.036	0.020	0.022	0.074
Sine ±7mm 0.2Hz	8.80	76.2	2.110	0.988	1.219	2.293	94.0	0.617	0.190	0.298	0.712
Sine ±5mm 0.3Hz	9.42	65.6	2.140	1.180	1.343	2.282	88.4	0.783	0.285	0.409	0.822

Table 4.5: *Comparison of IMPC and VPC in terms of various metrics of the tracking error when using ideal sine trajectories as reference trajectories where two trajectories possess maximum speeds above the speed limit of the HexaPOD*

Experiment 4: IMPC vs. VPC with Pre-recorded Patient Trajectories In the same setup as for experiment 3, this experiment uses 30 pre-recorded breathing trajectories from 18 different patients (for some patients, several fractions were used) along all three translational axes to estimate the performance of both control methods under more realistic conditions. Almost all patient datasets show the z-axis (AP) to be the dominant axis, the less dominant y-axis (SI) and the x-axis (LR) with minor motion amplitudes. Note that within tumor trajectories, the dominant axis is usually aligned with the superior-inferior direction. However, since control algorithms are capable of handling each axis independent of the others in the same way, a switch of the dominant axis does not affect the tracking performance of the HexaPOD.

Pre-recorded breathing motions from real patients were used to drive the system in the fourth experiment. For VPC, a value of $\lambda = 1$ would cause significant and frequent direction changes for many datasets which would be unacceptable for patients. Therefore this experiment was conducted using the values 0.6, 0.7 and 0.8 for λ in VPC.

Apart from the tracking accuracy metrics (CR, CI95, MAE, RMS and MAX), the metrics S_F, S_J, S_{TS} and S_{SNR} as well as CT25 and CT10 were determined to assess smoothness and convergence time. Except for the convergence time metrics, the metrics were determined after a startup time of 15s for each dataset. Figure 4.13 shows the distribution of the metric values as compact box plots for each controller setting. Before the box plots were drawn, the data for each metric was scaled to the maximum of that metric (over all controller settings), so that all metrics fit into one diagram. The corresponding value ranges associated with each box plot and the mean and median values are summarized in Table 4.6.

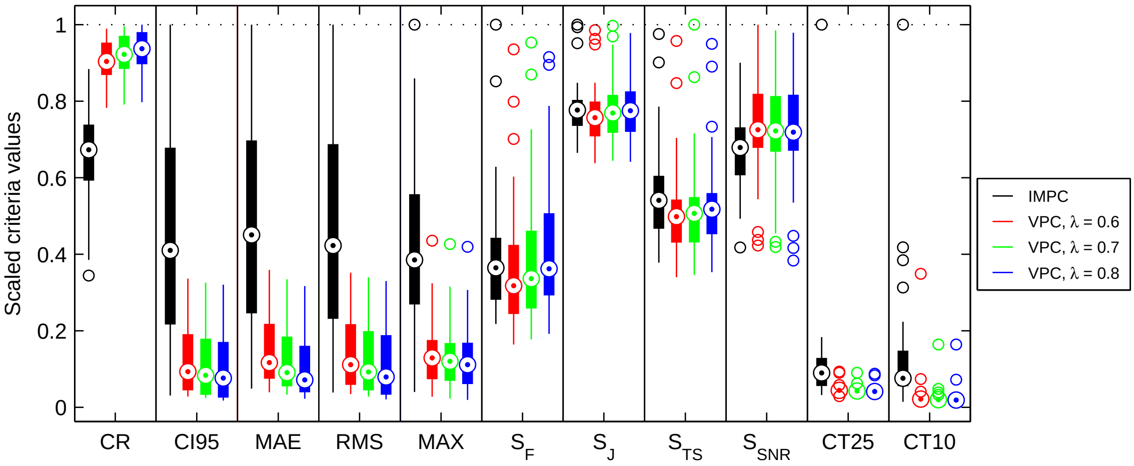


Figure 4.13: *Qualitative comparison of IMPC and VPC performance during compensation of 30 pre-recorded patient trajectories*

In analogy to the results of experiment 3, any of the VPC variants outperform IMPC on all metrics simultaneously. The huge performance increase in terms of tracking accuracy can be observed on the CR metric. The minimum CR for any of the VPC variants is larger than the upper quartile of the CR for IMPC. Additionally, the median values are larger than the upper whisker of the IMPC box plot. Also the spread of the CR values is constrained to a more narrow region between 76.7% to 98.0%. This is supported by the remaining tracking accuracy metrics in a similar way. While most of CI95 values for IMPC are larger than 1mm, all VPC variants show CI95 values of usually less than 1mm. Furthermore, the maximum CI95 values for the VPC variants are smaller than the corresponding mean and median values for IMPC. The same is true for the MAE, RMS and MAX metrics. Typical RMS errors for the VPC variants are about 0.20mm with average maximum errors of about 0.92mm. Thus, VPC allows precise reference tracking where most of the worst case errors are smaller than 1mm.

In terms of smoothness, the performance difference between IMPC and VPC is smaller. According to S_F and S_{TS}, smoother trajectories of the HexaPOD are obtained only for VPC with $\lambda = 0.6$ (VPC6) whereas VPC with $\lambda = 0.7$ (VPC7) leads to similar smoothness

Controller	Type	CR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]	S_F	S_J	S_{TS}	S_{SNR}	CT25 [s]	CT10 [s]
IMPC	min	33.7	0.12	0.05	0.06	0.26	0.42	1.85	0.08	10.38	1.15	1.40
	mean	64.6	1.73	0.47	0.73	2.78	0.80	2.18	0.13	16.80	4.46	13.94
	median	65.9	1.63	0.45	0.69	2.54	0.71	2.16	0.12	16.87	3.22	7.34
	max	86.6	3.98	1.00	1.62	6.59	1.95	2.78	0.22	22.37	35.96	96.73
VPC $\lambda = 0.6$	min	76.7	0.11	0.04	0.06	0.18	0.32	1.77	0.08	10.50	1.05	1.20
	mean	88.7	0.47	0.15	0.22	0.92	0.72	2.15	0.11	18.00	1.72	3.37
	median	88.5	0.37	0.12	0.18	0.85	0.62	2.10	0.11	18.03	1.60	2.07
	max	96.9	1.34	0.36	0.57	2.87	1.82	2.74	0.21	24.85	3.35	33.76
VPC $\lambda = 0.7$	min	77.5	0.10	0.03	0.05	0.15	0.34	1.79	0.08	10.40	1.00	1.20
	mean	90.3	0.43	0.12	0.20	0.89	0.77	2.17	0.12	17.85	1.58	2.55
	median	90.3	0.33	0.09	0.15	0.79	0.66	2.14	0.11	17.95	1.52	1.92
	max	97.5	1.29	0.34	0.55	2.81	1.86	2.77	0.22	24.47	3.25	15.88
VPC $\lambda = 0.8$	min	78.1	0.07	0.02	0.03	0.13	0.37	1.78	0.08	9.53	0.95	1.20
	mean	91.5	0.39	0.10	0.18	0.85	0.82	2.17	0.12	17.79	1.57	2.55
	median	91.8	0.30	0.07	0.13	0.73	0.70	2.16	0.12	17.86	1.48	1.85
	max	98.0	1.27	0.32	0.53	2.76	1.78	2.72	0.21	24.32	3.15	15.88

Table 4.6: *Quantitative comparison of IMPC and VPC performance during compensation of 30 pre-recorded patient trajectories*

as IMPC. VPC with $\lambda = 0.8$ (VPC8) results in less smooth trajectories. S_J shows similar smoothness for IMPC and VPC6 (with a preference for VPC6) and a larger spread of the metrics for VPC7 and VPC8 towards larger metric values. According to S_{SNR}, the smoothness of the VPC variants is similar and better than in IMPC. Generally, it can be inferred that VPC6 results in improved smoothness compared to the other controllers. Most of the smoothness metrics indicate that VPC8 leads to less smoothness than IMPC. With increasing λ, smoothness generally deteriorates within VPC as was intended by the introduction of λ.

In opposition to deterioration of smoothness, tracking accuracy improves with increasing λ according to all tracking accuracy metrics. The medians as well as the upper and lower whiskers of these metrics shift to improved metric values. The spread of metrics slightly reduces, too. Between VPC6 and VPC8, improvements of the average metrics are as follows: 2.8% for CR, 0.08mm for CI95, 0.05mm for RMS and 0.07mm for MAX. Hence, tracking accuracy and smoothness can be traded-off using the parameter λ.

Along with improved tracking accuracy for VPC variants, convergence time significantly improves for the VPC variants both in terms of absolute metric values and spread. With increasing λ, convergence time slightly reduces further.

Further analysis of these results in terms of a per dataset comparison using a subset of the tracking accuracy metrics is detailed in [157].

Figure 4.14 exemplifies the tracking errors of a dataset when using IMPC, VPC6 and VPC8. The CR metrics of this dataset resulted in 38.8%, 86.5% and 88.7%, respectively. The associated RMS errors were 1.21mm, 0.28mm and 0.24mm. Each diagram shows the amplitudes along z-axis (dominant) against the y-axis (less dominant) with and without compensation as well as the histograms for these axes in these situations depicting the relative frequency of the amplitudes. The amplitudes without compensation (blue) differ slightly between each trial because they are tumor predictions (corresponding to the negated desired position in the reference tracking problem) which can change due to measurement noise and feedback effects. The shown dataset has high slopes especially

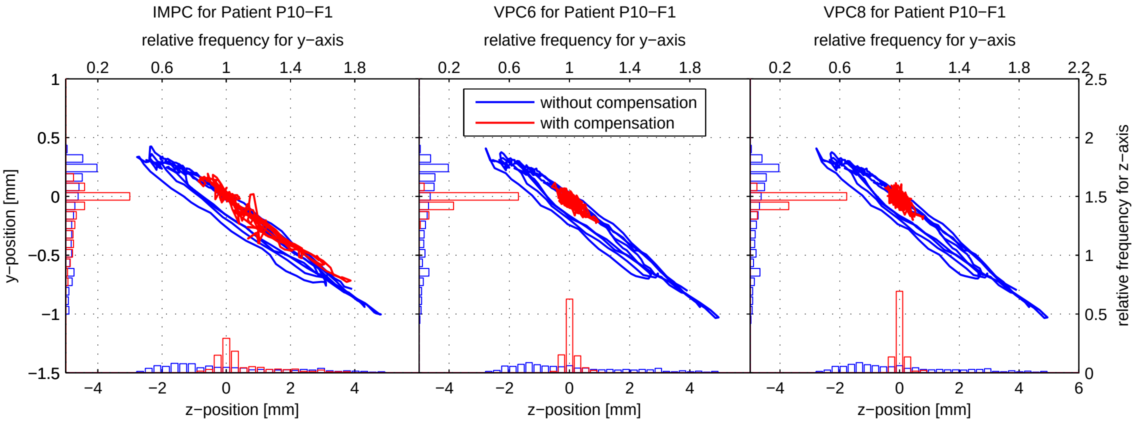


Figure 4.14: *Results of experiment 4 for a pre-recorded patient dataset: diagrams show the amplitudes of dominant z-axis against less dominant y-axis along with histograms depicting the relative frequencies of the amplitude ranges for both z- and y-axes. Each diagram depicts the situation without compensation (blue) and with active compensation (red) using one method of IMPC, VPC6 and VPC8.*

during inhalation phases which exceed the maximum speed of the HexaPOD. For IMPC, a considerable reduction of amplitudes can be observed in this case. However, VPC6 and VPC8 further decrease the residual amplitudes due to their ability to accelerate faster and to follow the desired trajectory more closely.

The histograms provide an intuitive comparison between the methods. Without compensation, amplitudes are distributed according to a quasi-periodic signal with a hysteresis due to different paths during inhalation and exhalation. This can be seen from the blue histograms in Figure 4.14 which show a higher frequency of the amplitudes during exhalation than inhalation because the exhalation phase is usually longer than the inhalation phase, especially when approaching the end of the exhalation phase. Thus, amplitudes at the end of the exhalation phase have similar values and occur during a longer period of time, as a consequence of which the frequency of these amplitudes rises. With active compensation, the red histograms change their shape. For all three variants, the largest frequency can be observed for amplitudes in the vicinity of zero, which means that the goal of motion compensation is generally fulfilled. The VPC variants show a significant improvement over IMPC for this dataset. The corresponding distributions approximately become normal distributions with zero mean and small standard deviation. For VPC8 the amplitudes of the z-axis are located closer to zero in a more narrow region than for VPC6 which demonstrates a slight but measurable improvement concerning tracking error.

Experiment 5: IMPC vs. VPC with Volunteers The last experiment was performed for both control methods using 17 healthy volunteers who were lying unconstrained on the HexaPOD. An IR tool, attached to a belt, was placed on their abdomen as in Figure 4.15 to measure breathing position. Another IR tool was rigidly attached to the HexaPOD to acquire its position. Compensation was performed along the dominant z-axis and less dominant y-axis.

Volunteers were asked to breathe normally. On startup of the system, the first complete

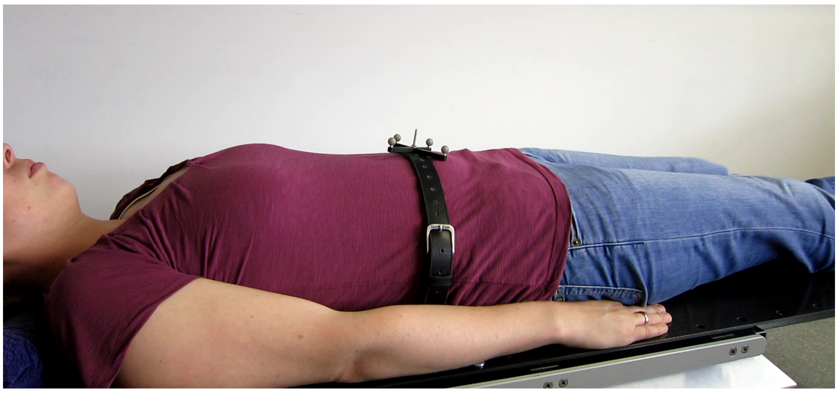


Figure 4.15: *Volunteer with attached IR tool lying unconstrained on the HexaPOD*

breathing cycle was used to determine the desired set-point position for motion compensation as the mean position between the last minimum and maximum peak (This procedure is necessary since the system is started at an arbitrary point during a breathing cycle but for better comparison the desired position should be similar). With the volunteer on the HexaPOD, the IMPC, VPC6 and VPC8 methods were used for breathing compensation in an arbitrary and varying order. Between each method the system was reconfigured for the next method which resulted in a small pause of less than one minute. Since the results of this experiment do not rely on the same trajectory for each volunteer, they are not directly comparable in contrast to the previous experiments. In fact, there are some volunteers whose breathing trajectories for the methods differ quite a lot in terms of amplitude and frequency.

However, the same metrics as in the previous experiment were used. Qualitative comparison of the metrics for the controller settings are shown in Figure 4.16 as box plots scaled to the maximum value for each metric. The corresponding minimum, maximum, mean and median values for each metric and controller setting is given in Table 4.7.

Due to the unconstrained positioning of the volunteers on the HexaPOD during this

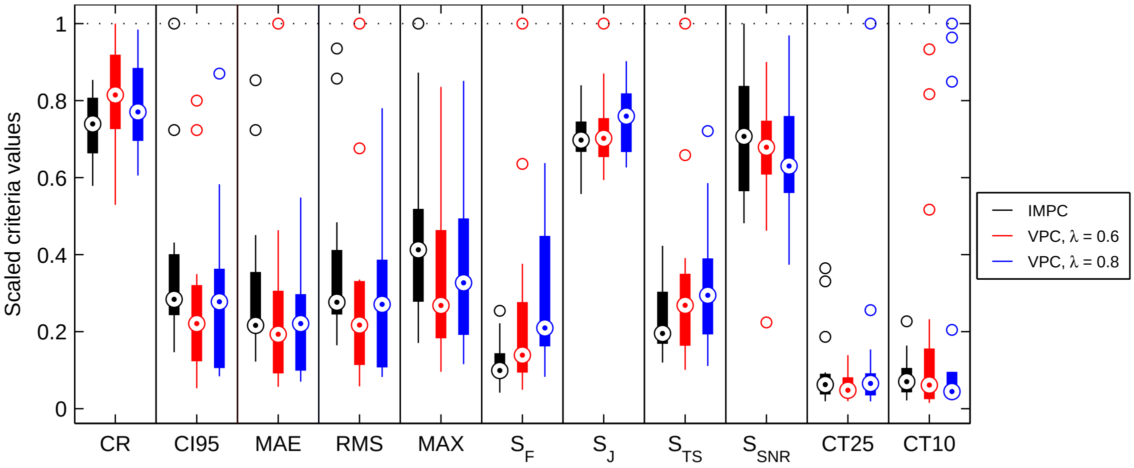


Figure 4.16: *Qualitative comparison of IMPC and VPC performance during compensation of 18 volunteer's breathing trajectories*

Controller	Type	CR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]	S_F	S_J	S_{TS}	S_{SNR}	CT25 [s]	CT10 [s]
IMPC	min	53.9	1.39	0.46	0.73	2.00	0.20	1.52	0.05	14.94	0.70	2.25
	mean	67.7	3.41	1.14	1.61	5.20	0.54	1.91	0.10	22.17	3.55	8.45
	median	68.9	2.69	0.81	1.23	4.83	0.47	1.91	0.08	21.94	2.30	7.39
	max	79.6	9.47	3.18	4.16	11.70	1.21	2.30	0.17	31.02	13.38	23.97
VPC $\lambda = 0.6$	min	49.3	0.50	0.21	0.26	1.12	0.23	1.62	0.04	6.94	0.70	1.60
	mean	75.0	2.52	0.92	1.26	3.86	1.13	1.96	0.13	20.38	2.20	19.88
	median	75.9	2.09	0.72	0.97	3.14	0.66	1.92	0.11	21.06	1.75	6.49
	max	93.2	7.58	3.72	4.44	9.78	4.75	2.73	0.41	27.92	5.10	98.58
VPC $\lambda = 0.8$	min	56.4	0.80	0.26	0.37	1.35	0.39	1.71	0.05	11.59	0.70	2.35
	mean	72.9	2.72	0.87	1.28	4.39	1.42	2.07	0.13	19.93	4.72	21.95
	median	71.8	2.63	0.82	1.20	3.82	1.00	2.08	0.12	19.55	2.40	4.70
	max	91.8	8.24	2.04	3.47	9.96	3.03	2.47	0.30	30.07	36.70	105.62

Table 4.7: *Quantitative comparison of IMPC and VPC performance during compensation of 18 volunteer's breathing trajectories*

experiment, their breathing patterns are distributed over a wide spectrum in terms of frequencies and amplitudes. While some volunteers showed peak-to-peak amplitudes of 8mm, others exhibited 20mm. For these volunteers, the HexaPOD continuously operated with its maximum speed while sill leaving larger residual amplitudes. Hence, compensation for volunteers with fast and/or deep breathing resulted in smaller CRs and increased tracking errors. A detailed table enabling direct comparison of each volunteer's results is given in [157].

As can be seen from the distribution of tracking accuracy metrics shown in Figure 4.16, the performance increase for VPC compared to IMPC is less with volunteers than with pre-recorded patient trajectories. Major improvements according to the CR can be observed for VPC6 and VPC8 compared to IMPC. However, there are a few datasets yielding worse tracking accuracy than with IMPC since the lower whisker for the CR of VPC6 extends further than the one for MPC. Following the CI95, MAE, RMS and MAX metrics, VPC6 generally performs better than IMPC. According to the upper whiskers of VPC8 for these metrics, there are also some datasets which perform worse compared to IMPC. From all tracking accuracy metrics it can be further inferred that VPC8 shows less tracking accuracy with a much wider spread of the metrics.

All smoothness metrics show the best smoothness for IMPC. With VPC smoothness deteriorates, especially with increasing λ the smoothness metrics spread more with worse median metric values. The convergence time metrics deliver a differentiated picture. According to the mean and median of CT25 metric, there is a small benefit for VPC6 compared to IMPC and VPC8, whereas for CT10, VPC8 possesses smaller convergence times (apart from the larger outliers).

Comparing the results of this experiment with volunteers to the previous one with pre-recorded patient trajectories, in which VPC outperformed IMPC, a different overall performance results with volunteers. The main difference in the experiment setup is the placement of the IR tool which measures breathing motion. For pre-recorded patient trajectories, the IR tool was rigidly attached to the HexaPOD whereas with volunteers it was placed on the soft part of the abdomen underneath the diaphragm. In combination with unconstrained volunteers on the HexaPOD, this setup allows the IR tool to oscillate. When the HexaPOD changed its movement direction along one axis relatively sharp, oscillations of the abdominal area could be observed during the compensation experiments.

These oscillations can be triggered by another effect which is more likely to happen in VPC than in IMPC, especially with increased λ. This was observed during a closer inspection of the individual datasets which revealed that the oscillations usually occurred in the vicinity of peaks where the HexaPOD needs to slow down. Thus, oscillations are actually small overshoots where the desired position is reached but the HexaPOD could not slow down fast enough, so that it needs to change direction at least for the next sampling instant. Because of the closed-loop system, a small oscillation can introduce feedback effects which can sustain for several sampling instants. However, it is interesting to note that the trials with volunteers having a higher breathing velocity and with a shorter exhalation phase suffered less from these oscillation effects. In these cases, the HexaPOD always lags behind the reference trajectory which is not completely reached, effectively preventing oscillations. This is also the reason for the observation that oscillations occurred more frequently along the minor axes, especially the y-axis, which has only a peak-to-peak amplitude of less than 2mm for some volunteers. Additionally, measurement noise of the IR camera plays a more significant role and can be another cause for oscillations apart from non-compensated system latencies.

The experiment setup with human volunteers lying completely unconstrained on the HexaPOD is more challenging than in normal clinical routine, where the patient is constrained by a fixation system. Such a system, for example the BodyFIX® system constrains the patients motions relative to the HexaPOD so that practically no whole-body motions are possible. Furthermore, the evacuation of the system puts pressure on the whole human body which can significantly damp or even prevent oscillations [158].

As could be seen, the oscillation effects can cause worse performance of VPC8 in comparison to VPC6. This situation is exemplified in Figure 4.17. Even though the amplitudes of breathing motion are slightly smaller along the z- and y-axis for VPC8 as opposed to VPC6, the residual amplitudes after compensation are larger for VPC8 than for VPC6, especially along the y-axis where the peak-to-peak breathing amplitude range is about

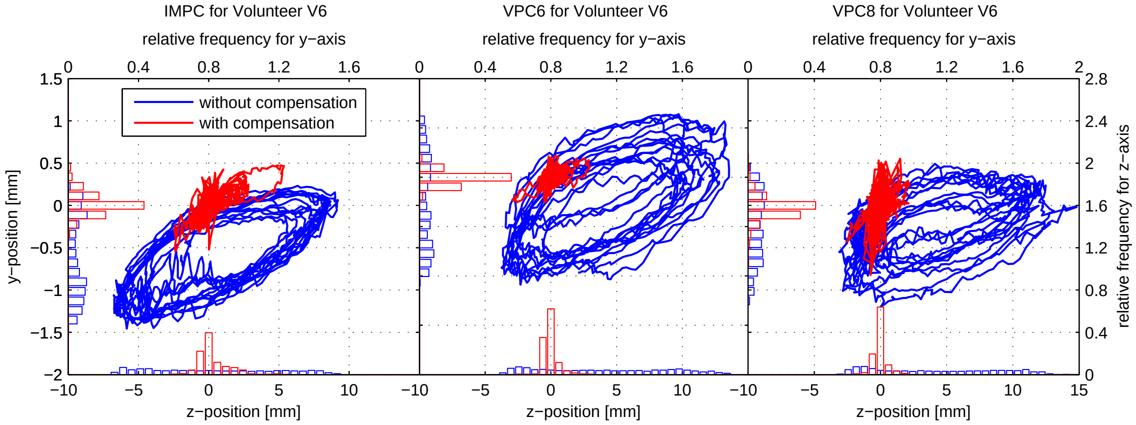


Figure 4.17: *Results of experiment 5 for a volunteer: diagrams show the amplitudes of dominant z-axis against less-dominant y-axis along with histograms showing the relative frequencies of certain amplitude ranges for both z- and y-axes. Each diagram depicts the situation without compensation (blue) and with active compensation (red) using one method of IMPC, VPC6 and VPC8.*

1.5mm. This is also reflected by the smoothness metric S_F: for VPC6 it is 0.458 whereas it doubled for VPC8 to 0.997. Nevertheless, the performance metrics of tracking error still show an improvement for VPC8 over the ones for IMPC which can also be seen by more narrow histograms along the dominant z-axis. The breathing pattern of VPC6 shows a more varying character than for VPC8 or IMPC (there is a small baseline-shift along the y-axis), however residual amplitudes are lower for the y- and z-axes and the distribution is more narrow.

4.2.4.2 Discussion

During the tests with VPC, no instability has been observed so far. The controller always tries to follow the reference trajectory as close as possible by computing the necessary control input to reach the desired position at the next sampling instant and to simultaneously keep the velocity of the reference. If the reference is faster than the speed limit, larger deviations from the reference trajectory occur, but the resulting velocity of the HexaPOD is then at maximum, thereby still trying to follow the trajectory as much as possible. Due to the way the positions in the command are derived, they will never exceed the operational range limits, so that the position of the HexaPOD is constrained within that range, effectively resulting in a stable control loop as long as the reference is constrained to the operational range. Since the non-compensated latencies as discussed in Section 2.2.3.3 are small, they can be assumed to pose no threat to stability.

In IMPC, stability is achieved by appropriate choices of the free parameters. Also for this controller, during all preliminary tests as well as during all experiments, no instability has been observed. However, due to constraints, the MPC could become infeasible especially for non-minimum phase models. Therefore the implementation checks at each sampling instant if MPC was infeasible and reports this condition. During all the tests no infeasibility has occurred (except if $N_p = 1$). If an infeasibility occurs, the second control input from the sequence of optimal control increments is chosen, which was computed during the previous sampling instant. This procedure can be repeated $N_u - 1$ times in case of persistent infeasibilities. If no more control increments are available, treatment would need to be stopped. However, this condition never occurred so far.

In Experiment 1, a sine trajectory with an appropriate maximum speed was used as a reference trajectory along all three translational axes in order to make the internal controller of the HexaPOD exhibit the type of velocity reconstruction by observing its output trajectory. Results demonstrated that the velocity specification in the HexaPOD's command has to be chosen as the absolute value of the desired velocity along the main axis, the axis with the largest difference between desired and actual position.

Experiment 2 pointed out the influence of speed gain λ in VPC with useful settings for trading-off tracking performance and patient comfort.

Using ideal sine trajectories as reference trajectories the third experiment compared the tracking performance of IMPC and VPC with $\lambda = 1$. If the maximum speed of trajectories is smaller than the speed limit of the HexaPOD, an already large tracking performance can be achieved with IMPC whereas with VPC near perfect tracking with almost no tracking error is reached. These results additionally validate the negligence of the non-compensated latencies. Faster trajectories result in larger tracking errors but VPC still outperforms IMPC.

A more realistic scenario is to use real pre-recorded patient trajectories which are overlaid on actual measurements of the IR camera as in Experiment 4. Depending on the maximum speed of the reference, larger tracking errors can be observed. Still, VPC clearly outperformed IMPC for every patient dataset whereby larger speed gains λ resulted in less tracking errors but also in less smooth trajectories.

If the system is used with real humans in the loop, the closed-loop nature of the system can lead to feedback effects degrading performance in terms of both tracking and smoothness as becomes apparent in the fifth experiment. Nevertheless, VPC with a sufficiently low λ provides a better tracking performance than the IMPC variant. The generally lesser performance of IMPC can be attributed to several aspects: Firstly, there are modeling uncertainties in the parameters, arising in any model based on system identification methods. Possibly, a change of the identifying datasets for example to real patient data could lead to improved models. Secondly, measurement noise of the IR camera can further worsen the situation during identification. And thirdly, since IMPC is based on a model describing the position-position response, it only adapts the position parameters in the HexaPOD's command, whereas the velocity parameter is set to the constant 1. In VPC, control is realized over velocity, allowing a fine granular adaptation of position.

Due to unconstrained positioning of volunteers on the HexaPOD and the placement of the IR tool on soft tissue, one sharp change in direction can cause the IR tool to oscillate for a short moment which can induce more direction changes. As the HexaPOD system is itself inherently stable and the occurring oscillations of the IR tool are quickly damped by the human body, instability is of no concern here but patient comfort is. Despite considerable tracking performance achieved with both controllers (with improved tracking errors for VPC), some volunteers experienced a “shaky ride” on the HexaPOD. Some even felt uncomfortable during the compensation of their breathing motion. Both controllers possess means of reducing control effort leading to smoother trajectories of the HexaPOD, however at the cost of tracking performance. Hence, the question arose if VPC can be combined with MPC to further improve upon the VPC approach, especially in terms of smoothness. The combined approach is discussed in the following section.

4.2.5 Model Predictive Control of Velocity Model

In VPC, control is mainly achieved by determining the appropriate velocity required for reference tracking along the main axis using Eq. (4.34). The non-absolute version of this in the form

$$s(t) = \frac{p_j(t+1) - p_j(t)}{\Delta\tau}$$

can be regarded as the inverse of a simple linear velocity model along axis j given by

$$p_j(t+1) = p_j(t) + \Delta\tau s(t) \ . \tag{4.41}$$

Identifying the position $p_j(t)$ with the state $x_j(t)$ of a first-order state-space model with $n = 1$ of the form Eq. (4.10), the elements of it can be chosen as $A = 1$, $B = \Delta\tau$ and $C = 1$ in order to reflect Eq. (4.41). Note that $\Delta\tau$ represents the uniform sampling time unlike in Eq. (4.34). The input $u_j(t)$ corresponds to the linear velocity (which is specified in normalized form in the HexaPOD's command) whereas the output $y_j(t)$ is the actual position of the HexaPOD along one axis j. Essentially, the model is expressed as an undisturbed constant velocity model, where its disturbed version was already used in Section 3.4.1 for motion prediction.

This state-space model, originating from the VPC approach, can be incorporated into a MPC similar to the identified model of the HexaPOD from Section 4.2.3. The output $y_j(t)$ still represents the position of the HexaPOD. Thus, the constraint on the output

$$l_j^L \leq y_j(t) \leq l_j^U \tag{4.42}$$

ensures the position of the HexaPOD lies within the operational range limits l_j^L and l_j^U for an axis j. The constraint on the input $u_j(t)$ forces the velocity of the HexaPOD to be smaller than v_{max}, that is

$$-v_{max} \leq u_j(t) \leq v_{max} \ . \tag{4.43}$$

In that respect, the constant velocity model in combination with MPC enables the controller to simultaneously take care of the operational limits of the HexaPOD as well as of the velocity saturation which was first observed during model identification.

With the state-space model (4.10), the output constraint (4.42), input constraint (4.43) and cost function (4.13) which is adopted in the same way from the IMPC approach, the optimization problem can be written as

$$\begin{aligned}
&\min_{\Delta\hat{U}_j \triangleq \{\Delta\hat{u}_j(t),\ldots,\Delta\hat{u}_j(t+N_u-1)\}} \{J(t)\} \\
\text{s.t.}\quad & \hat{x}_j(t+k+1|t) = A\hat{x}_j(t+k|t) + B\left(\hat{u}_j(t+k-1|t) + \Delta\hat{u}_j(t+k|t)\right),\ k \geq 0, \\
& \hat{y}_j(t+k|t) = C\hat{x}_j(t+k|t),\ k \geq 0, \\
& \hat{x}_j(t|t) = x_j(t), \\
& \hat{u}_j(t-1|t) = u_j(t-1), \\
& \hat{u}_j(t+k|t) = \hat{u}_j(t+k-1|t),\ N_u \leq k < N_p, \\
& -v_{max} \leq \hat{u}_j(t+k|t) \leq v_{max},\ k = 0,\ldots,N_p, \\
& l_j^L \leq \hat{y}_j(t+k|t) \leq l_j^U,\ k = 1,\ldots,N_p \ .
\end{aligned}$$

Since a full measurement of the state $x_j(t)$ is available ($x_j(t)$ equals the current position of the HexaPOD along axis j measured by the IR camera), no state observer needs to be employed.

In effect, one instance of MPC including the velocity model determines a desired velocity increment $\Delta v_j(t) = \Delta\hat{u}_j(t|t)$ for a single axis j as the input refers to the HexaPOD's velocity along an axis. If more axes need to be actuated, additional instances of the MPC can be executed in parallel to deliver the velocity increments for the remaining axes.

Adding the desired velocity increments to the previous control inputs $u_j(t-1)$ for corresponding axis j, the 3D vector $\vec{v}(t)$ of desired velocities is formed:

$$\vec{v}(t) = \begin{bmatrix} u_1(t-1) + \Delta v_1(t) \\ u_2(t-1) + \Delta v_2(t) \\ u_3(t-1) + \Delta v_3(t) \end{bmatrix} \ ,$$

which the HexaPOD is supposed to realize until the next sampling instant. Since the HexaPOD can be supplied only with one velocity specification, the VPC approach is expressed in terms of velocities in the following to accommodate for that need. Inserting

$$\Delta\vec{p}(t) = \vec{v}(t)\Delta\tau$$

into Eq. (4.35) and substituting

$$K' = K \Delta\tau$$

yields

$$g : \vec{a} = \vec{p}(t) + K' \, \vec{v}(t) \ .$$

Then, Eq. (4.37) becomes

$$K'_j = \frac{l_j - p_j(t)}{v_j(t)}$$

from which the point of intersection is derived as

$$\vec{c}(t) = \vec{p}(t) + K'_m \, \vec{v}(t)$$

in analogy to the VPC approach. Eq. (4.39) simplifies to

$$s(t) = |v_m(t)|,$$

completing the MPC approach including the velocity model, further referred to as VMPC.

Prediction accuracy normally degrades with the prediction horizon N_p. Therefore parameters $Q(k)$ in the cost function are chosen as $Q(k) = 2^{N_p - k}$ for $k = 1, \dots, N_p$, reducing the contribution of tracking errors for higher prediction horizons in the cost function. Future control increments should be weighted equally, so that $T(k) = T(j), \forall k, j \in \{1, \dots, N_u\}$. The value of the cost function is not of interest, but the ratio γ of both sums is. Using

$$\gamma = \frac{\sum_{i=1}^{N_u} T(i)}{\sum_{i=1}^{N_p} Q(i)} = \frac{N_u T}{2^{N_p} - 1}$$

the tuning parameter T can be derived from a given γ by $T = \left(2^{N_p} - 1\right) \gamma / N_u$. Setting $N_u = N_p$ for this work, VMPC possesses only two tuning parameters N_p and γ.

4.2.5.1 Experimental Tests and Results

The same experimental tests as for the comparison between IMPC and VPC were performed to compare VPC to VMPC. Therefore the same experimental setup applies as described in Section 4.2.4.1.

From experience with the IMPC approach, where the model of the HexaPOD was obtained by system identification methods, useful values for N_p were observed in the range of 5-10, where performance gain reduces exponentially with larger prediction horizons. Thus, $N_p = 7$ was used for the VMPC approach with $N_u = N_p$. Various values for γ were tested in preliminary tests. It was found that the values 0.01, 0.025, 0.05, 0.075 and 0.1 cover an adequate range from emphasis on tracking accuracy to patient comfort.

Experiment 1: VPC vs. VMPC with Pre-recorded Patient Trajectories In the first experiment, the simulator program used 30 pre-recorded breathing trajectories from 18 different patients (the same were already used in experiment 4 comparing IMPC to VPC). Compensation using real hardware was performed along all three translational axes for both control methods with all specified parameter settings.

The analysis of results is performed in terms of the previously used prediction accuracy, smoothness and convergence time metrics. Additionally, both VPC and VMPC methods give rise to the adoption of another metric representing the control effort (CE). The CE can be computed from the saturated control input $s(t)$ according to Eq. (4.32) with

$$\mathrm{CE} = \sum_{t=2}^{n} \frac{1}{\Delta\tau} \left(\underset{v_{max}}{\mathrm{sat}} \left(s(t)\right) - \underset{v_{max}}{\mathrm{sat}} \left(s(t-1)\right) \right) = \frac{v_{max}}{\Delta\tau} \sum_{t=2}^{n} s_c(t) - s_c(t-1) \ ,$$

which represents the desired acceleration given in mm/s^2 to follow the reference trajectory (if not saturated). The results in terms of applied metrics (calculated after a startup time of 15s for each dataset) of the tracking error, smoothness and convergence time are depicted in Figures 4.18 and 4.19 as box plots, each of them showing the distribution of a criterion for a specified controller setting. Before the box plots were drawn, data for each criterion was first scaled to the maximum of that criterion, so that all criteria fit into

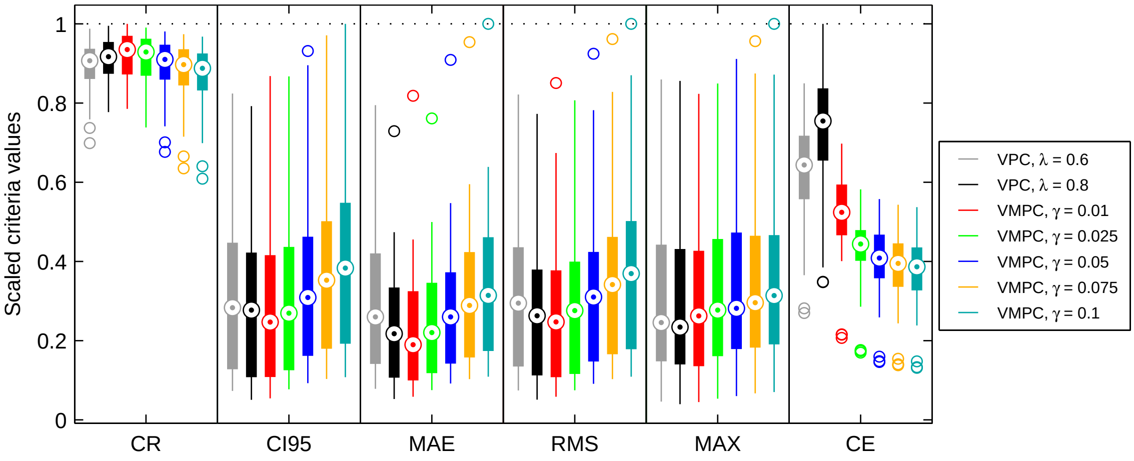


Figure 4.18: *Qualitative comparison of VPC and VMPC performance on scaled tracking accuracy and CE metrics during compensation of 30 pre-recorded patient trajectories*

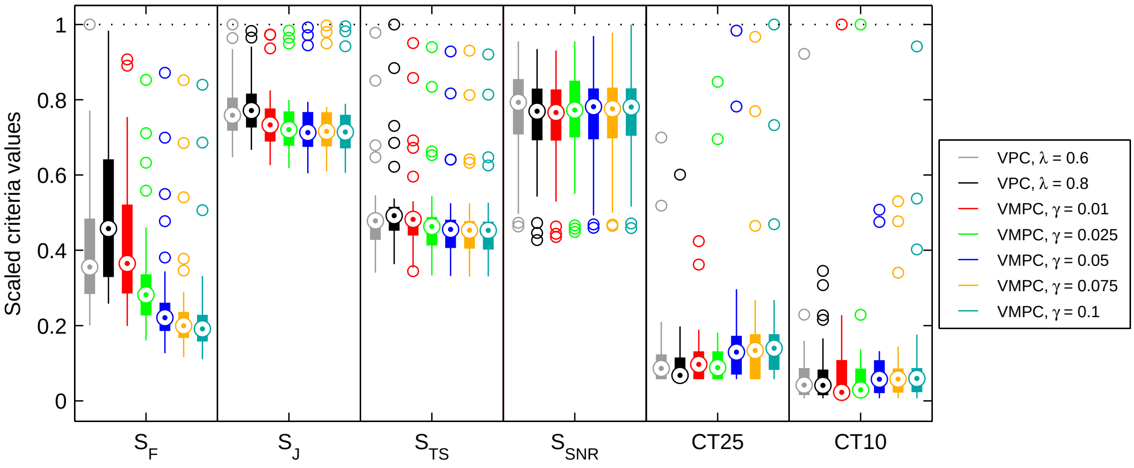


Figure 4.19: *Qualitative comparison of VPC and VMPC performance on scaled smoothness and convergence time metrics during compensation of 30 pre-recorded patient trajectories*

one diagram. The corresponding value ranges associated with each box plot as well as the mean and median values are summarized in Table 4.8.

Controller	Type	CR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]	CE [$\frac{\text{mm}}{\text{s}^2}$]	S_F	S_J	S_{TS}	S_{SNR}	CT25 [s]	CT10 [s]
	min	66.7	0.17	0.06	0.08	0.24	6.92	0.45	1.82	0.08	10.55	0.70	0.70
VPC	mean	84.5	0.73	0.24	0.35	1.54	16.02	0.91	2.17	0.12	17.65	1.55	8.36
$\lambda = 0.6$	median	86.6	0.65	0.21	0.33	1.28	16.52	0.80	2.13	0.12	18.06	1.05	4.17
	max	94.3	1.88	0.64	0.92	4.49	21.81	2.24	2.81	0.24	21.73	8.49	91.29
	min	74.2	0.12	0.04	0.06	0.21	8.92	0.58	1.87	0.09	9.73	0.70	0.70
VPC	mean	86.5	0.67	0.19	0.31	1.51	18.70	1.12	2.20	0.13	16.96	1.30	7.31
$\lambda = 0.8$	median	87.6	0.63	0.18	0.29	1.22	19.38	1.02	2.16	0.12	17.51	0.82	4.07
	max	95.0	1.81	0.59	0.86	4.47	25.67	2.20	2.76	0.25	21.28	7.29	34.21
	min	75.0	0.12	0.05	0.06	0.23	5.31	0.45	1.76	0.08	9.91	0.70	0.70
VMPC	mean	87.7	0.66	0.18	0.30	1.55	13.08	0.93	2.10	0.12	16.95	1.38	8.71
$\gamma = 0.01$	median	89.3	0.56	0.15	0.28	1.37	13.45	0.82	2.06	0.12	17.43	1.17	2.25
	max	95.5	1.99	0.66	0.95	4.29	17.91	2.03	2.73	0.23	21.19	5.15	99.03
	min	70.5	0.18	0.06	0.08	0.28	4.35	0.36	1.73	0.08	10.22	0.70	0.70
VMPC	mean	86.5	0.72	0.19	0.32	1.63	10.92	0.74	2.08	0.12	17.27	1.72	8.52
$\gamma = 0.025$	median	88.7	0.62	0.18	0.31	1.44	11.40	0.63	2.02	0.11	17.58	1.07	2.90
	max	94.6	1.98	0.62	0.90	4.43	14.94	1.91	2.76	0.23	21.74	10.29	99.03
	min	64.6	0.21	0.07	0.10	0.31	3.75	0.28	1.70	0.08	10.47	0.70	0.70
VMPC	mean	84.6	0.80	0.22	0.36	1.72	10.01	0.62	2.06	0.12	17.31	2.14	8.49
$\gamma = 0.05$	median	86.9	0.71	0.21	0.35	1.47	10.49	0.49	2.00	0.11	17.79	1.57	5.69
	max	93.7	2.13	0.74	1.03	4.75	14.31	1.95	2.78	0.23	22.06	11.94	50.29
	min	60.6	0.24	0.08	0.11	0.35	3.54	0.26	1.71	0.08	10.57	0.70	0.70
VMPC	mean	83.2	0.86	0.25	0.39	1.79	9.60	0.56	2.06	0.12	17.37	2.24	9.27
$\gamma = 0.075$	median	85.7	0.81	0.23	0.38	1.54	10.14	0.45	2.01	0.11	17.67	1.62	5.74
	max	93.0	2.22	0.77	1.07	4.99	13.94	1.91	2.80	0.23	22.27	11.74	52.49
	min	58.1	0.25	0.09	0.12	0.36	3.37	0.25	1.70	0.08	10.45	0.70	0.70
VMPC	mean	82.0	0.91	0.27	0.42	1.86	9.36	0.53	2.06	0.12	17.41	2.30	11.58
$\gamma = 0.1$	median	84.8	0.88	0.25	0.41	1.64	9.92	0.43	2.00	0.11	17.77	1.70	5.89
	max	92.4	2.29	0.81	1.12	5.22	13.79	1.88	2.79	0.23	22.77	12.13	93.24

Table 4.8: *Quantitative comparison of VPC and VMPC performance during compensation of 30 pre-recorded patient trajectories*

The metrics resulting from VPC for this experiment (cf. Table 4.8) and the ones for experiment 4 (comparing VPC to IMPC, cf. Table 4.6) differ although they represent the same experiment with the same datasets. This difference stems from a different implementation of the VPC methods. While in experiment 4, the actual position of the HexaPOD was derived from the HexaPOD directly (by sending a command to the HexaPOD to acquire its position), the actual position of the HexaPOD was acquired by measurements of an IR tool. When acquiring the position from the HexaPOD directly, the resulting trajectories are smoother because no measurement noise of the IR camera is introduced. However, the change in implementation was done to allow for a fair comparison between VPC and VMPC, which only makes use of the positions acquired by the IR camera.

As can be seen in Figure 4.18, improvements of tracking accuracy according to the CR can be achieved with VMPC for smaller values of γ. With increasing γ, the spread of the CR metric increases, even beyond the range of CR values for VPC6 and VPC8. This is supported in the same way by the CI95, MAE, RMS and MAX metrics. For small γ, emphasis in the cost function is placed on tracking accuracy rather than on control increments. With increasing γ, emphasis shifts towards small control increments. Hence, tracking accuracy decreases with increasing γ.

According to the CE metric displayed in Figure 4.18, control effort decreases with increasing γ due to the fact that smaller control increments are favored by the optimization in VMPC. It is interesting to note that the control effort is smaller for any of the VMPC variants than for VPC, despite improved tracking accuracies for small γ in VMPC compared to VPC. This behavior can be attributed to the optimizing character of VMPC, taking into account a future horizon rather than just the next sampling instant as in VPC. The same results are furthermore supported by other smoothness metrics S_F, S_J, S_{TS} and S_{SNR}.

Similar to the decrease in tracking performance with increasing γ, the convergence time metric CT25 shows an increase in convergence time. While similar convergence times are observed for VPC and VMPC with small γ, VPC8 possesses the smallest average convergence times. With CT10 the picture changes with smallest average convergence times for VMPC with $\gamma = 0.01$, however associated with a larger spread.

Generally, the best overall performance can be observed for $\gamma = 0.01$. If $\gamma \geq 0.025$ the improvement in smoothness is considerably larger than the average loss in tracking accuracy (cf. median CRs of box plot and mean CRs in Tab 4.8). Hence, suitable values for γ range in between 0.01 to 0.05. Average compensation ratios of 84.6% to 87.7% can be expected with average CI95 values of less than 0.8mm and average RMS errors of less than 0.36mm. Simultaneously, smoothness and with it patient comfort is considerably improved compared to VPC.

As an example, Figure 4.20 shows the results for a patient dataset when using VPC8, VMPC with $\gamma = 0.01$ and VMPC with $\gamma = 0.05$. While with VPC8, tracking errors (shown in red) are considerably reduced as opposed to the motion to be compensated (shown in blue), VMPC with $\gamma = 0.01$ improves tracking errors further which can be seen by the red histograms. Even when using VMPC with $\gamma = 0.05$, resulting in a smoother trajectory, yields smaller tracking errors than VPC8, especially along the y-axis.

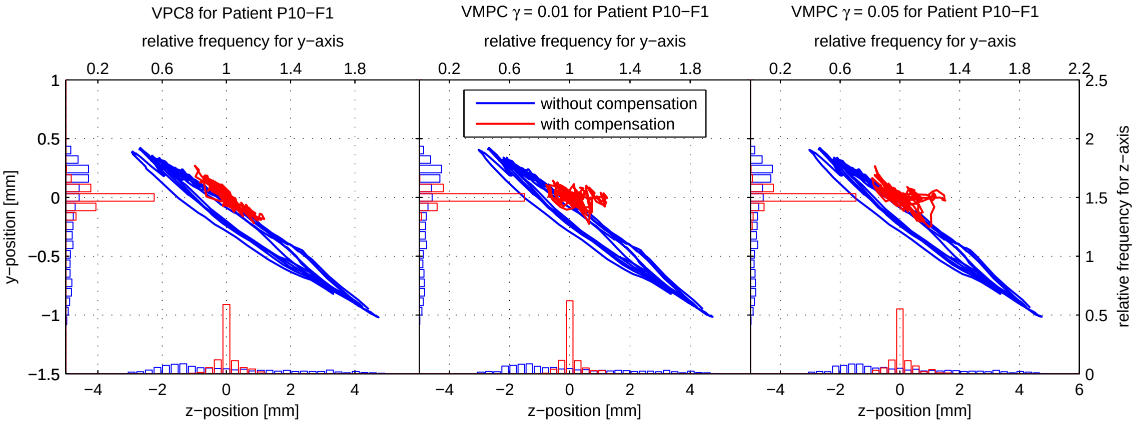


Figure 4.20: *Results of experiment 1 for a pre-recorded patient dataset: diagrams show the amplitudes of dominant z-axis against less dominant y-axis along with histograms depicting the relative frequencies of the amplitude ranges for both z- and y-axes. Each diagram depicts the situation without compensation (blue) and with active compensation (red) using VPC8 and VMPC.*

Experiment 2: VPC vs. VMPC with Volunteers The second experiment was also performed for both control methods along all three translational axes (note that in experiment 4 comparing IMPC to VPC only motions along the y- and z-axis were compensated) using 18 healthy volunteers who were lying unconstrained on the HexaPOD. An IR tool, attached to a belt, was placed on their abdomen to measure the 3D breathing position (using a second IR tool rigidly attached to the HexaPOD). The volunteers were asked to breathe normally.

The same evaluation as for experiment 1 was performed on the last 60s of each trial. Although the results for each volunteer are not directly comparable as for experiment 1, the box plots in Figures 4.21 and 4.22 show the distribution of scaled metric values. The corresponding minimum, maximum, mean and median values are given in Table 4.9.

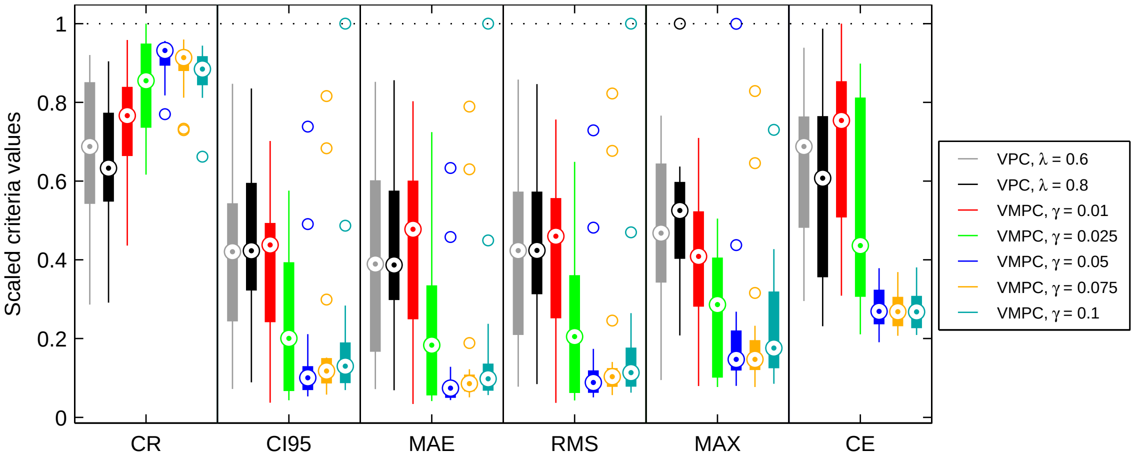


Figure 4.21: *Qualitative comparison of VPC and VMPC performance on scaled tracking accuracy and CE metrics during compensation of 18 volunteer's breathing trajectories*

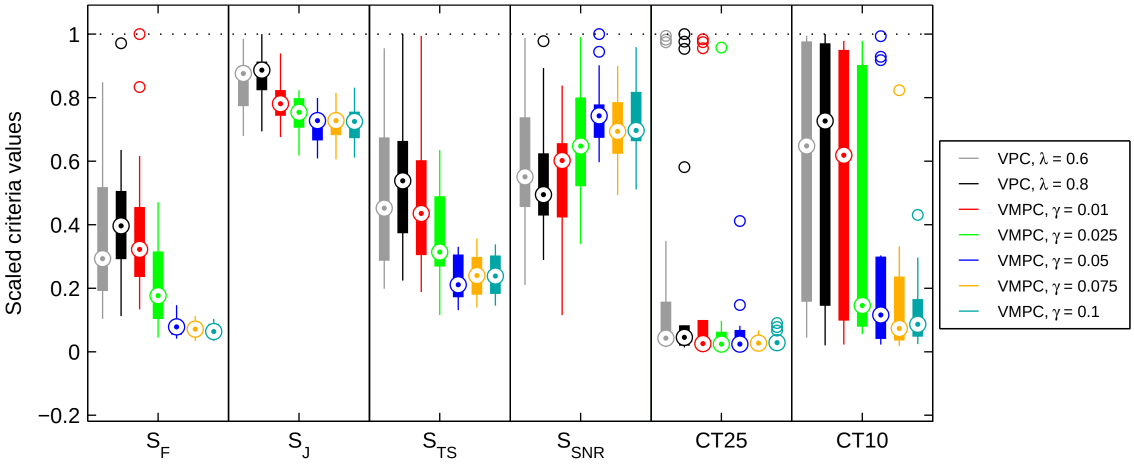


Figure 4.22: *Qualitative comparison of VPC and VMPC performance on scaled smoothness and convergence time metrics during compensation of 18 volunteer's breathing trajectories*

Controller	Type	CR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]	CE [$\frac{\text{mm}}{\text{s}^2}$]	S_F	S_J	S_{TS}	S_{SNR}	CT25 [s]	CT10 [s]
	min	27.9	0.35	0.13	0.18	0.70	10.54	0.68	1.86	0.09	5.45	1.30	4.25
VPC	mean	66.1	1.95	0.72	0.96	3.39	22.27	2.38	2.32	0.22	15.27	20.94	54.17
$\lambda = 0.6$	median	67.0	2.01	0.70	1.00	3.44	24.53	1.93	2.40	0.20	14.32	4.07	61.72
	max	89.7	4.05	1.53	2.02	5.64	33.49	5.58	2.70	0.41	25.69	94.23	94.79
	min	28.4	0.42	0.12	0.20	1.53	8.26	0.74	1.90	0.10	7.52	1.30	1.90
VPC	mean	62.7	2.11	0.76	1.03	3.76	21.35	2.64	2.37	0.22	14.37	21.17	54.81
$\lambda = 0.8$	median	61.7	2.02	0.69	1.00	3.86	21.67	2.61	2.43	0.23	12.86	4.32	69.22
	max	88.1	3.99	1.54	1.99	7.35	35.21	6.38	2.74	0.43	25.43	94.79	95.28
	min	42.5	0.18	0.06	0.09	0.58	11.02	0.88	1.85	0.08	2.99	1.10	2.15
VMPC	mean	73.2	1.85	0.76	0.95	2.97	25.29	2.50	2.16	0.21	14.32	22.97	51.31
$\gamma = 0.01$	median	74.7	2.09	0.86	1.08	3.01	26.89	2.12	2.14	0.19	15.65	2.42	58.93
	max	93.4	3.35	1.44	1.78	5.22	35.67	6.57	2.58	0.43	21.79	93.24	93.24
	min	60.1	0.21	0.07	0.10	0.56	7.51	0.30	1.69	0.05	8.84	1.00	5.29
VMPC	mean	81.6	1.13	0.40	0.54	2.00	18.44	1.44	2.04	0.16	16.87	8.28	34.53
$\gamma = 0.025$	median	83.3	0.96	0.33	0.48	2.11	15.57	1.16	2.07	0.14	16.84	2.27	13.88
	max	97.4	2.75	1.30	1.53	3.71	32.05	3.09	2.26	0.28	25.77	90.74	93.24
	min	75.0	0.25	0.08	0.12	0.59	6.81	0.27	1.67	0.06	15.52	1.70	2.15
VMPC	mean	88.3	0.74	0.22	0.34	1.56	9.91	0.52	1.94	0.10	19.53	5.72	24.68
$\gamma = 0.05$	median	90.8	0.48	0.13	0.21	1.08	9.60	0.51	2.00	0.09	19.31	2.27	11.04
	max	93.2	3.53	1.14	1.71	7.35	13.51	0.96	2.19	0.14	26.01	39.00	94.63
	min	71.1	0.27	0.09	0.13	0.57	7.39	0.23	1.66	0.06	12.82	1.45	1.70
VMPC	mean	86.7	0.89	0.28	0.41	1.61	9.92	0.48	1.96	0.11	18.50	3.08	14.99
$\gamma = 0.075$	median	89.0	0.56	0.15	0.24	1.08	9.57	0.47	2.00	0.10	18.04	2.55	6.99
	max	93.5	3.90	1.42	1.93	6.10	13.16	0.74	2.24	0.15	23.40	6.34	78.45
	min	64.5	0.33	0.10	0.15	0.63	7.45	0.23	1.68	0.06	13.28	1.40	2.30
VMPC	mean	84.9	0.96	0.31	0.44	1.65	9.76	0.44	1.96	0.10	18.89	3.39	12.10
$\gamma = 0.1$	median	86.2	0.62	0.18	0.27	1.29	9.56	0.42	1.99	0.10	18.11	2.70	8.24
	max	92.0	4.78	1.79	2.35	5.37	13.58	0.68	2.28	0.15	24.94	8.59	41.05

Table 4.9: *Quantitative comparison of VPC and VMPC controller performance during compensation of 18 volunteer's breathing trajectories*

This experiment includes VPC6 and VPC8 when performing motion compensation for all three translational axes whereas the results in experiment 5 for IMPC vs. VPC covered compensation along the y- and z-axis only. As can be inferred by comparing the performance metrics for VPC in these two cases (cf. Tables 4.7 and 4.9), tracking accuracy and smoothness are considerably reduced in this experiment. This further demonstrates the problems of the VPC approach where induced oscillations reduce overall performance. Since compensation along the minor x-axis was additionally performed in this experiment, the oscillation behavior became worse compared to experiment 5, which is well described by the smoothness metrics. The results from experiment 5 can be observed here as well, that is, VPC6 delivers both improved tracking accuracy and smoothness compared to VPC8. However, for some volunteers, performance with VPC was not acceptable in terms of comfort.

As can be seen from the tracking accuracy metrics shown in Figure 4.21, tracking accuracy for VMPC with $\gamma = 0.01$ improves compared to the VPC trials, however the metrics still show a large spread comparable to VPC. A similar situation can be observed in this case for the smoothness metrics depicted in Figure 4.21. In terms of CE, this controller setting exhibits even a larger control effort than the VPC variants.

All the performance metrics improve simultaneously if γ is increased to 0.025. Along with an improvement of the mean and median values of each metric, the spread of each criteria reduces. Among the trials with this controller setting, there were already some for which no oscillations were introduced so that the volunteers felt comfortable (supported by metric values). However, this was not reached for all of them.

Increasing γ to 0.05 finally solved the comfort issues as can be seen by all tracking accuracy and smoothness metrics. This setting not only results in the best tracking accuracy metrics but also shows the smallest spread of these metrics. Simultaneously the smoothness metrics result in improved metric values with a small spread compared to the previously discussed controller settings. As can be inferred from the CE metric, overall improvement is achieved with considerably less control effort. This fact substantiates the origin of the oscillation problem, which is caused by a too large control effort making the HexaPOD change directions sharply and thus introducing oscillations which are amplified by the closed-loop system.

Increasing γ further slightly reduces tracking accuracy as can be seen by all tracking accuracy metrics. Then, the cost function emphasizes control effort stronger than tracking accuracy. Simultaneously, some of the smoothness metrics indicate slight improvements. However, the smoothness gain is considerably less than the loss in tracking accuracy, so that γ should be selected as small as possible.

During inspection of the individual datasets, it turned out that in all datasets for VPC and VMPC with $\gamma = 0.01$ and in some datasets for VMPC with $\gamma = 0.025$, strong oscillations with large amplitudes on the x-axis and smaller ones on the y-axis were introduced, which was already observed for VPC along two axes (experiment 5). Nevertheless, most of the amplitude along the dominant z-axis was still compensated. In these settings the controllers are too aggressive and cause the HexaPOD to change direction too often and too fast, so that the patients and the soft part of the abdomen on which the IR tool is placed starts oscillating, thereby strongly degrading patient comfort and tracking accuracy especially along the x- and y-axis. VMPC with $\gamma \geq 0.05$ is able to deal with these effects in order to account for patient comfort while maintaining good CR average of 88.3% with average RMS errors of 0.34mm and CI95 of less than 0.74mm when applying the system on real humans for 3D compensation of breathing motion. The oscillation effects could be reduced in VPC as well by decreasing λ, but then average CR would be less than the ones achievable with VMPC. Another important aspect also visible in the box plots is the spread of the criteria values depicted by the upper and lower whiskers which is very small for VMPC with $\gamma \geq 0.05$ compared to the other controller settings. These settings resulted in similar criteria values for all volunteers, suggesting that when applied to other volunteers or patients, the range of criteria values will be similar as well, so that a somewhat patient-independent controller setting can be found. Most of the volunteers actually enjoyed the tests with $\gamma \geq 0.05$, some of them were almost falling asleep, so that also according to the volunteers judgment after the tests, these controller parameters are acceptable.

Figure 4.23 depicts the individual compensation results of a volunteer in terms of tracking error (shown in red) and motion to be compensated (shown in blue) for VPC6, VMPC with $\gamma = 0.05$ and VMPC with $\gamma = 0.1$. Despite motion amplitudes are smaller for VPC6 than during the other trials, tracking errors are larger. As can be seen from the motion to be compensated, the trace in the y-z plane is more irregular than for the trials with VMPC. This stems from induced oscillations which the volunteer was exposed to during compensation with VPC6. Furthermore, feeling uncomfortable during motion compensation can also change the breathing pattern of a volunteer. In sum, this leads to larger

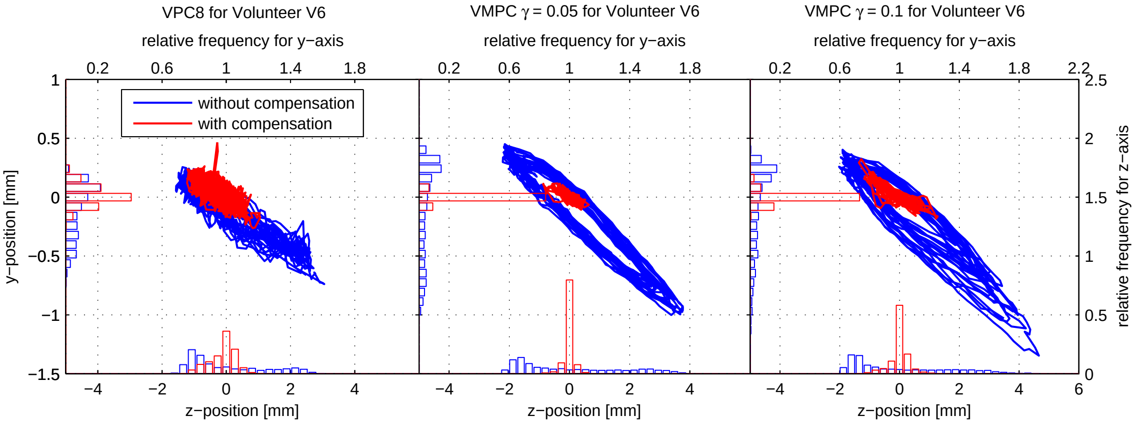


Figure 4.23: *Results of experiment 2 for a volunteer: diagrams show the amplitudes of dominant z-axis against less-dominant y-axis along with histograms showing the relative frequencies of certain amplitude ranges for both z- and y-axes. Each diagram depicts the situation without compensation (blue) and with active compensation (red) using VPC6 and VMPC.*

residual tracking errors for this volunteer. For VMPC with $\gamma = 0.05$ the motion to be compensated follows a regular trace. As indicated by the histograms, only small tracking errors remain while the HexaPOD followed a smooth trajectory, having no negative impact on the motion. For VMPC with $\gamma = 0.1$, the resulting trajectory became smoother, however with a small loss in tracking accuracy as indicated by the red histograms. Both VMPC variants are possible for this volunteer, where γ can be used to trade-off patient comfort and tracking accuracy.

4.2.5.2 Discussion

Comparative results from hardware-in-the-loop tests of VPC and VMPC approaches were given in which 3D breathing motion of 30 pre-recorded patient trajectories and 18 volunteers was compensated, which is also found in [159]. The test scenario was more difficult than in normal treatment sessions, in which patients are constrained by a fixation system. This makes it harder for them to move, thus also damping oscillations unlike when lying unconstrained on the HexaPOD.

In VPC patient comfort can be accounted for by reducing the speed gain λ, however at the cost of tracking accuracy. While for ideal trajectories or even pre-recorded breathing trajectories, performance in terms of smoothness is still acceptable, it becomes worse when VPC is used with humans-in-the-loop. Then, oscillation effects seriously reduce patient comfort and often even tracking performance.

The adoption of VPC by MPC allows to better account for patient comfort while simultaneously preserving the tracking capabilities. Especially, with volunteers lying on the HexaPOD, tracking performance could be even improved with VMPC. Note that the achieved results reported in Table 4.9 were achieved by the HexaPOD with a speed limit of 8mm/s. Most of the volunteers' trajectories possess maximum speeds exceeding the limit (especially during exhalation), effectively constraining tracking performance. Hence, the results with already large compensation ratios have to be interpreted in terms of this constraint. Thus, most of the remaining tracking errors are actually caused by the constraint. Using a faster HexaPOD, such as the HexaPOD evo, can further reduce tracking errors.

Neglecting the prediction and control horizon to be chosen for VMPC, the parameter γ is the only one to realize the trade-off between tracking accuracy and patient comfort, i.e. smoothness of the HexaPOD's trajectory. It was found that with this parameter the trade-off can be controlled on a fine-granular level. Similar to IMPC, stability is achieved by appropriate choice of the MPC parameters, specifically γ. Since supplied reference trajectories have constrained characteristics in terms of amplitude and frequency variations (cf. Section 3.1), a sufficiently large range can be expected to be found γ, to avoid instabilities of the constrained MPC problem for any breathing or tumor motion. In fact, quantitative comparisons revealed that a single choice for γ is applicable to all volunteers in order to reach a certain performance in terms of smoothness and tracking accuracy. Infeasibilities were not encountered in all tests using VMPC. In case of infeasibilities they can be handled as discussed in the IMPC approach. If that fails, too, VPC could be used any time as fallback solution, rendering the control scheme robust. This additionally makes the method more appealing and easy to apply.

4.3 Overall Performance Analysis

Overall performance analysis in this section focuses on the overall error, comprised of prediction and tracking errors. The relationship between these errors is established and exemplified in experiments with volunteers. Furthermore, it is shown how the transition phase after system startup can be reduced to allow for a short startup time with small tracking and overall errors. The timing within the software package *HexGuide* demonstrates real-time aspects of the ATTS. Appropriate timing bounds are stated for the sampling time to allow operation of the ATTS in hard real-time. Additionally, an outlook using the HexaPOD evo treatment couch is given.

4.3.1 Overall Error Analysis

So far, analysis of control methods has been performed in terms of tracking error. However, the error of interest for characterizing the whole system is the overall error, i.e. the non-compensated residual motion amplitudes.

Both prediction and tracking errors contribute to overall error, where their relationship is derived as follows: Denote a measurement of the actual tumor or breathing position as $\vec{p}^{T}(t) \in \mathbb{R}^3$, which is measured in relation to a room-fixed reference frame with the coordinate axes coinciding with the HexaPOD reference frame. In order to predict the tumor or breathing motion, it needs to be stated relative to the HexaPOD as the HexaPOD changes tumor position with respect to the room-fixed reference frame. Hence, the actual position $\vec{p}^{H}(t) \in \mathbb{R}^3$ of the HexaPOD is removed from tumor or breathing position, yielding the relative position $\Delta\vec{p}^{T}(t) \in \mathbb{R}^3$ given by

$$\Delta\vec{p}^{T}(t) = \vec{p}^{T}(t) - \vec{p}^{H}(t) \ . \tag{4.44}$$

Based on the relative position, the chosen predictor computes k-step predictions $\Delta\hat{\vec{p}}^{T}(t+k|t)$ of the tumor or breathing position relative to the actual position of the HexaPOD.

According to Eq. (3.6), the k-step prediction error is

$$\vec{e}_k^{\,P}(t) = \Delta\vec{p}^{\,T}(t) - \Delta\hat{\vec{p}}^{\,T}(t|t-k) \ , \tag{4.45}$$

since the desired value for the predicted position is given by the measurement and the actual value is the prediction. The predictions are negated to deliver the reference trajectory $\vec{r}(t|t+k) \in \mathbb{R}^3$ of the HexaPOD

$$\vec{r}(t|t+k) = -\Delta\hat{\vec{p}}^{\,T}(t|t+k) \ , \quad k = 1, 2, \ldots, n_r \ . \tag{4.46}$$

Negation is required to make the HexaPOD move in direction opposite to tumor or breathing motion in order to achieve motion compensation. The reference trajectory, specifically the reference position $\vec{r}(t|t+1)$ at the next sampling instant, defines the desired position of the HexaPOD. It's difference from the actual position of the HexaPOD yields the tracking error

$$\vec{e}^{\,T}(t) = \vec{r}(t|t-1) - \vec{p}^{\,H}(t) \tag{4.47}$$

according to Eq. (4.9). Thus, tracking error is determined by negated one-step predictions of the tumor or breathing position.

Subtracting Eq. (4.45) with $k = 1$ from Eq. (4.47) yields

$$\vec{e}^{\,T}(t) - \vec{e}_1^{\,P}(t) = \vec{r}(t|t-1) - \vec{p}^{\,H}(t) - \Delta\vec{p}^{\,T}(t) + \Delta\hat{\vec{p}}^{\,T}(t|t-1) \ . \tag{4.48}$$

By inserting Eq. (4.46), Eq. (4.48) reduces to

$$\vec{e}^{\,T}(t) - \vec{e}_1^{\,P}(t) = -\Delta\vec{p}^{\,T}(t) - \vec{p}^{\,H}(t) \ . \tag{4.49}$$

In terms of overall error, describing the residual positions after compensation, the first term $-\Delta\vec{p}^{\,T}(t)$ of the right-hand side of Eq. (4.49) defines the desired position of the HexaPOD which would be necessary to achieve full compensation in accordance with Eq. (4.46). Since the second term describing the actual position of the HexaPOD is subtracted, the right-hand side of Eq. (4.49) states the overall error $\vec{e}^{\,O}(t)$. Thus, the overall error is determined by the difference between tracking and prediction errors. Expanding

$$\vec{e}^{\,O}(t) = \vec{e}^{\,T}(t) - \vec{e}_1^{\,P}(t) = -\Delta\vec{p}^{\,T}(t) - \vec{p}^{\,H}(t) \tag{4.50}$$

by inserting Eq. (4.44), yields

$$\vec{e}^{\,O}(t) = -\vec{p}^{\,T}(t) \ .$$

Hence, overall error is alternatively represented by the negated tumor or breathing position measurement. As this measurement is performed with respect to a room-fixed reference frame, this quantity can be observed from room-fixed reference point, thus matching overall error.

The relationship between these errors can also be seen by following the information flow within *HexGuide* (cf. Figures 2.6 and 2.7). Note that the sum of absolute prediction and tracking errors is always an upper limit of the absolute overall error. Thus, if prediction and tracking error point into the same direction, they can reduce overall error.

The analysis of results from experiment 2 (comparison of VPC and VMPC with volunteers) was extended to the overall error for VMPC with $\gamma = 0.05$. All accuracy metrics for each volunteer's trial are given in Table 4.10. The overall compensation ratio (OCR)

Volunteer	Overall errors					Prediction errors					Tracking errors				
	OCR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]	PR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]	CR [%]	CI95 [mm]	MAE [mm]	RMS [mm]	MAX [mm]
V1	71.6	0.85	0.30	0.40	1.56	84.9	0.52	0.19	0.25	0.99	79.6	0.49	0.16	0.22	1.10
V2	71.5	0.78	0.29	0.40	1.79	80.0	0.51	0.19	0.26	1.27	84.2	0.50	0.15	0.22	0.96
V3	81.7	0.76	0.24	0.34	1.37	86.0	0.49	0.17	0.23	0.97	92.3	0.39	0.10	0.16	0.74
V4	83.7	0.51	0.17	0.24	1.40	88.0	0.33	0.11	0.16	1.07	91.0	0.33	0.09	0.15	0.89
V5	78.8	0.77	0.21	0.34	1.85	83.6	0.55	0.16	0.25	1.40	92.1	0.28	0.08	0.13	0.96
V6	82.2	0.62	0.21	0.29	1.39	87.1	0.43	0.16	0.21	0.86	90.9	0.34	0.10	0.15	0.87
V7	72.3	2.67	1.08	1.36	3.66	87.2	1.05	0.37	0.50	2.22	79.7	2.36	0.82	1.13	3.22
V8	82.4	0.60	0.23	0.29	1.20	85.8	0.44	0.17	0.22	0.88	93.0	0.27	0.09	0.12	0.59
V9	80.9	0.89	0.29	0.40	1.97	85.6	0.58	0.20	0.28	1.37	91.4	0.47	0.12	0.20	1.12
V10	82.4	1.12	0.40	0.55	2.53	87.0	0.73	0.27	0.37	1.65	92.3	0.62	0.16	0.28	1.73
V11	66.5	4.21	1.50	2.05	7.35	85.8	1.53	0.52	0.72	2.92	75.0	3.53	1.14	1.71	7.35
V12	81.7	0.86	0.28	0.38	1.50	89.1	0.40	0.15	0.20	0.86	88.7	0.60	0.16	0.25	1.07
V13	75.8	0.79	0.28	0.38	1.60	81.6	0.54	0.19	0.26	1.19	89.2	0.41	0.12	0.19	1.28
V14	78.8	0.73	0.27	0.36	1.73	85.9	0.47	0.17	0.23	1.09	87.3	0.50	0.14	0.22	1.16
V15	77.7	0.40	0.13	0.19	0.70	84.4	0.24	0.09	0.11	0.46	87.0	0.28	0.08	0.12	0.61
V16	77.1	1.49	0.44	0.64	3.45	83.0	0.97	0.32	0.46	2.50	90.7	0.65	0.18	0.28	1.62
V17	83.9	1.52	0.52	0.71	3.09	89.1	0.81	0.33	0.43	2.19	91.8	1.01	0.23	0.41	1.97
V18	84.8	0.49	0.18	0.25	1.23	87.9	0.35	0.13	0.18	0.98	93.2	0.25	0.08	0.12	0.76
min	66.5	0.40	0.13	0.19	0.70	80.0	0.24	0.09	0.11	0.46	75.0	0.25	0.08	0.12	0.59
mean	78.5	1.11	0.39	0.53	2.19	85.7	0.61	0.22	0.30	1.38	88.3	0.74	0.22	0.34	1.56
med.	79.9	0.79	0.28	0.38	1.66	85.8	0.52	0.18	0.25	1.14	90.8	0.48	0.13	0.21	1.08
max	84.8	4.21	1.50	2.05	7.35	89.1	1.53	0.52	0.72	2.92	93.2	3.53	1.14	1.71	7.35

Table 4.10: *Quantitative comparison of overall, prediction and tracking performance during compensation of 18 volunteer's breathing trajectories of VMPC with* $\gamma = 0.05$

is derived similarly to the CR, except that the RMS of the desired trajectory is computed from negated breathing positions stating the desired values in terms of overall error.

When multiplying the achieved PRs and CRs for each volunteer, the result expressed in percent is found to be slightly smaller than the corresponding OCR. In fact, for all volunteers from Table 4.10, the relationship between PR and CR to OCR is found to be

$$\mathrm{OCR} = \mathrm{PR} \cdot \mathrm{CR} + \varepsilon \ ,$$

where $\varepsilon \in [1.7\%; 4.2\%]$. This indicates that an approximate relationship between the ratios may exist in such a way that the product of PR and CR constitutes a lower bound for OCR where the difference between the bound and the actual OCR is small. However, more trials are necessary to consolidate this potential relationship. Nevertheless, it can provide a thumb rule to estimate the average OCR when average PRs are known for a predictor and average CRs are available for a control method.

As can be seen in Table 4.10, RMS prediction errors are smaller or equal than 0.5mm except for volunteer V11. The associated CI95 limits are mostly smaller than 1mm (except for V7 and V11) and absolute MAX errors are smaller than 3mm. As seen by the median MAX errors, half of the maximum absolute errors are smaller than 1.14mm. On average, CI95 limits of 0.61mm and RMS errors of 0.30mm result for the prediction errors. Thus, predictions of the Takens predictor (which was used in the experiments) lead to small one-step prediction errors with PRs in the range 80.0% to 89.1%. As became apparent during comparison of predictors, slightly increased prediction accuracy can be achieved with the ERLS or NERLS predictor, however with a loss in smoothness.

CI95 limits of tracking errors are smaller than 1.1mm with RMS errors smaller than 0.5mm except for volunteers V7 and V11. Absolute maximum tracking errors are smaller than 2mm in these cases. On average, CI95 limits of 0.74mm and RMS errors of 0.34mm with MAX errors of 1.56mm are found. Thus, tracking errors are also small for most volunteers with CRs in the range 75.0% to 93.2%.

Both tracking and prediction errors result in the overall error according to Eq. (4.50). Since the absolute overall error is always less than the sum of absolute prediction and tracking errors, the same is true for each of the error metrics shown in Table 4.10. CI95 limits of the overall error are smaller than 1.6mm with RMS errors smaller than 0.71mm and absolute maximum errors smaller than 3.45mm except for V7 and V11. On average, CI95 limits of 1.11mm, RMS errors of 0.53mm and MAX errors of 2.19mm result in terms of the overall error. Hence, both prediction and tracking errors contribute to the overall error, however their contribution depends on their sign. Because of that, they partially reduce the overall error as becomes obvious from all error metrics.

As seen in this error analysis, errors for volunteers V7 and V11 are larger than for the others. In order to derive the reason for this result, several characteristics of each volunteer's breathing trajectory are given in Table 4.11. The first four columns state the CI95, MAE, RMS and MAX metric values of observed speeds along breathing trajectories. The speeds were obtained by first performing a zero-phase filtering with a Butterworth filter as described in Section 3.1.3 and then computing the discrete derivative of filtered output. Speeds from multiple axes were combined by taking the maximum metric along each translational axis. The following three columns show the amplitude ranges along each translational axis of breathing motion, including drifts and baseline shifts, so that the effective periodic amplitude range is usually smaller than the indicated range. In the last

Volunteer	CI95 [mm/s]	MAE [mm/s]	RMS [mm/s]	MAX [mm/s]	Range x [mm]	Range y [mm]	Range z [mm]	Remarks
V1	6.43	3.61	4.08	8.10	3.63	1.18	6.69	
V2	5.46	2.57	3.09	8.12	0.42	1.16	7.06	
V3	6.00	3.42	3.80	7.14	0.44	1.79	8.13	
V4	5.66	3.40	3.74	5.89	0.39	0.78	6.90	
V5	4.59	2.49	2.84	4.93	1.10	1.52	9.09	
V6	5.97	2.85	3.47	6.48	0.67	1.52	6.19	
V7	10.10	5.63	6.30	11.54	2.05	3.10	16.61	fast, deep
V8	4.78	2.12	2.66	5.23	0.47	1.31	6.89	
V9	5.24	2.57	2.97	4.95	0.76	0.92	7.10	
V10	5.98	3.61	3.89	7.01	0.83	1.60	12.15	
V11	10.02	5.26	6.11	13.22	2.89	2.29	19.91	fast, irreg.
V12	7.99	4.24	4.77	9.45	0.47	1.06	7.82	
V13	3.80	1.98	2.25	4.66	1.10	3.16	6.82	
V14	5.92	3.95	4.25	6.55	0.49	1.06	7.77	
V15	3.38	2.08	2.27	3.69	0.68	0.39	4.10	
V16	6.23	3.70	4.13	9.85	1.17	5.66	14.63	irregularity
V17	7.54	4.20	4.73	8.62	1.28	3.29	15.97	regular, deep
V18	5.31	3.27	3.57	6.11	0.73	0.73	7.04	

Table 4.11: *Speed metric results and amplitude ranges along three translational axes from 18 volunteer's breathing trajectories during compensation by VMPC with $\gamma = 0.05$*

column qualitative remarks about the motion traces are given for volunteers with either large speed metrics or large ranges. The CI95, MAE and RMS speed metrics for volunteer V7 are the largest among all volunteers, where the MAX speed metric value comes close to the one of volunteer V11. Especially the large CI95 metric indicates generally larger speeds throughout the whole dataset. Additionally, the amplitude range of V7 belongs to one of the largest. As verified by visual inspection of the dataset, V7 exhibits fast and deep breathing motion traces. Similarly, volunteer V11 shows fast motions as well with a large amplitude range. However, an irregularity is found in the motion trace with larger amplitudes than during normal breathing. This irregularity stems from the curiosity of the volunteer who was interested to see the system's reaction to a single deeper and faster breath. V16 shows rather moderate motion speeds and amplitudes during normal breathing but also exhibits an intended irregularity being the reason of the large amplitude range along the y- and z-axes. V17 is an example of a volunteer with regular but deep breathing with moderate speeds.

When comparing speed metrics from Table 4.11 to metrics of the tracking error given in Table 4.10, a correlation between tracking error metrics and speed metrics is obvious. This is especially prominent for V7 and V11 in such a way that exceptionally large tracking errors result when the largest CI95 speeds which are larger than the HexaPOD's speed limit, are observed. To support this, each scatter plot in Figure 4.24 connects one of the speed metrics RMS, MAE, CI95 and MAX to the corresponding tracking error metric for each volunteer. From each plot, the same observation can be made: Small speed metric values lead to small corresponding error metric values, where approximately a linear correlation can be fitted. After a certain limit for each speed metric, the associated tracking error metric value is much larger. The outliers in the scatter plots correspond to volunteers V7 and V11. Thus, the tracking error to be expected is mainly determined by the speed of the trajectory to be compensated. If the speed is above the HexaPOD's speed limit, larger tracking errors will result.

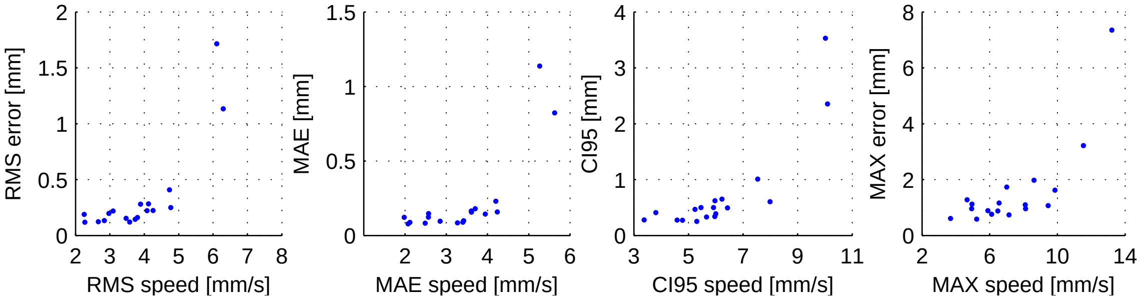


Figure 4.24: *Correlation between speed metrics of breathing motion to tracking error metrics of 18 volunteer's breathing trajectories during compensation by VMPC with* $\gamma = 0.05$

Apart from V7 and V11, larger MAX overall errors are found for V16 and V17. While V17 is a regular but deep motion trace with large amplitudes, the PR at 89.1% is relatively high because the large MAX error in relation to the large absolute motion range of 15.97mm along the z-axis appears reduced. Because of the regular nature of the trajectory and moderate speeds, the OCR for V17 still belongs to one of the largest among all volunteers. With V16, an irregularity is found, which moderately decreased prediction performance and to a certain extent also tracking performance. However, the speeds of the motion trace are mainly below the HexaPOD's speed limit so that the irregularity can be partially compensated for. In contrast to V16, the speed introduced by the irregularity of V11 was too large to make the HexaPOD track the reference trajectory, resulting in a large MAX tracking error.

Since irregularities were partially predictable, maximum prediction errors were still constrained to less than 3mm and decent prediction performance of more than 80% PR was achieved. If speed of the trajectory to be compensated for is below the HexaPOD's speed limit, tracking errors remained small even in the case of irregularities. If speed increases above the limit, then larger tracking errors are to be expected. Nevertheless, even with a speed limit of 8mm/s with which the experimental test was performed, most of the predicted motion could be compensated by the ATTS. If a faster HexaPOD is employed, tracking errors will reduce further.

Note that tracking errors in VMPC are not only influenced by one-step predictions but by all predictions up to the prediction horizon N_p due to the optimization using the future reference trajectory defined by predictions. Hence, the tracking errors cannot be seen to be completely independent from prediction errors, so that overall errors result as difference between tracking and prediction errors with an interaction between the tracking errors and predictions as well as associated prediction errors.

4.3.2 Transition Phase and Real-Time Considerations

The transition phase when the system is switched on was not discussed yet. To illustrate the overall error including the transition phase when the system is started, an experiment was performed using a custom-made phantom (cf. Figure 4.25) which was placed on top of the HexaPOD. The phantom possesses a plexiglass plate which is connected by six dampers including springs to a wooden base plate. Up to three servo motors are used to

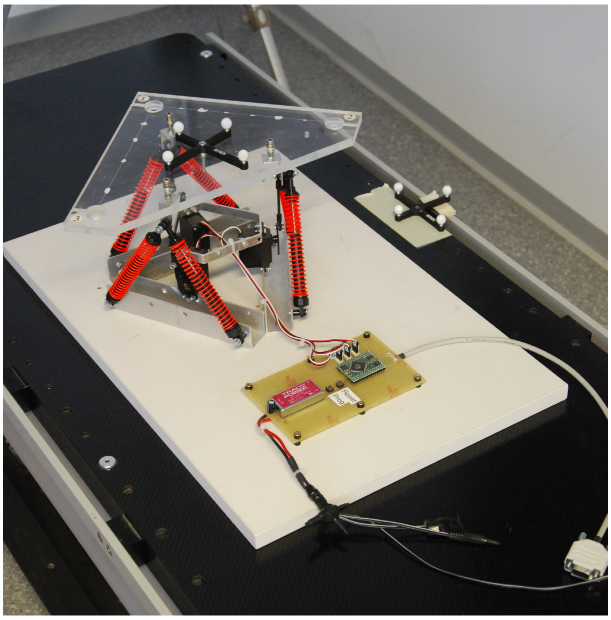


Figure 4.25: *Custom-made breathing phantom for exhibiting regular 3D trajectories including hysteresis is placed onto the HexaPOD*

displace the plexiglass plate while the springs counteract that motion. An IR tool was attached to the phantom to measure the displacement. If only two servos are used, the IR tool follows the motion trace depicted by a red line in Figure 4.26. The resulting 3D trajectory is regular and includes hysteresis similar to real breathing or tumor motion. A second IR tool was fixed to the HexaPOD for acquiring its position. The very first position measurement received from the IR camera after system startup defines timestamp 0. The sequence of applied predictors within *HexGuide* followed the recommendations given in Section 3.7. At first, the EKF predictor with spring-mass model is employed which is able to deliver a full set of multi-step predictions right from the beginning. Simultaneously, acquired measurements are used to initialize the NERLS predictor. With the detection of the peak at time 11.25s, the predictions from the NERLS predictor are used to generate the reference trajectory which is fed to the VMPC. Thus, the initialization and transition phase of the NERLS predictor can be completely bridged. Due to the fact that predictions are available from the beginning of compensation, VMPC enables the HexaPOD to quickly converge to the desired position shown as a red line in Figure 4.26 (which is the negated difference between the measurements of the IR tool on the phantom and on the HexaPOD, i.e. the motion to be compensated relative to the HexaPOD). The blue line represents the actual position measurement of the IR tool attached to the HexaPOD. The difference between desired and actual position forms the overall error shown in green. In the depicted situation, convergence is reached at approximately 1.6s after system startup. This time depends on initial displacement at system startup, speed and direction of the trajectory, accuracy of predictions as well as on maximum acceleration of the HexaPOD. However, it can be safely estimated that for most common breathing and tumor trajectories, convergence takes place in a matter of a few seconds given accurate short-term predictions and trajectory speeds less than the HexaPOD's speed limit. Under these conditions, the example additionally demonstrates achievable overall performances by the combination of

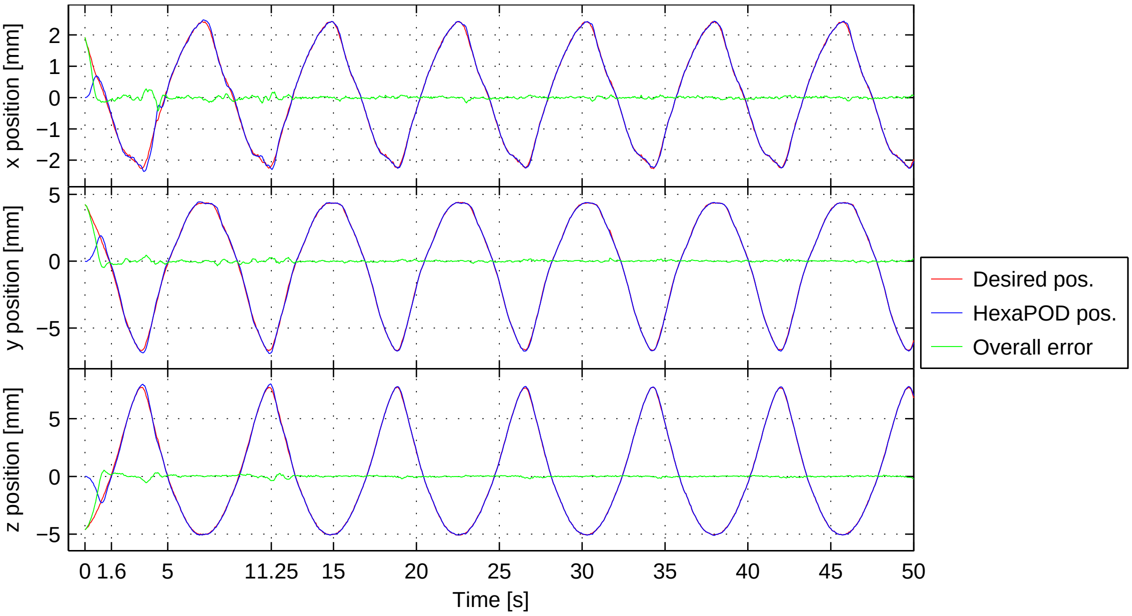


Figure 4.26: *3D compensation of phantom motions showing early-startup capabilities by combination of several predictors and VMPC*

several predictors and VMPC.

In order to ensure overall system performance, real-time conditions have to be respected within the ATTS. Therefore, timings are constantly monitored within *HexGuide* by acquiring timestamps at certain processing steps. Figure 4.27 exemplifies the resulting timings of the compensation test shown in Figure 4.26. The first diagram depicts the measurement latency of 19.99ms (including serial transmission time) of the Polaris IR camera, which was determined in Section 2.2.3.1. Upon arrival of a measurement sample, acquired timestamps are corrected for measurement latency. After arrival of the measurements, some preprocessing steps such as determination of moving average and peak detection are un-

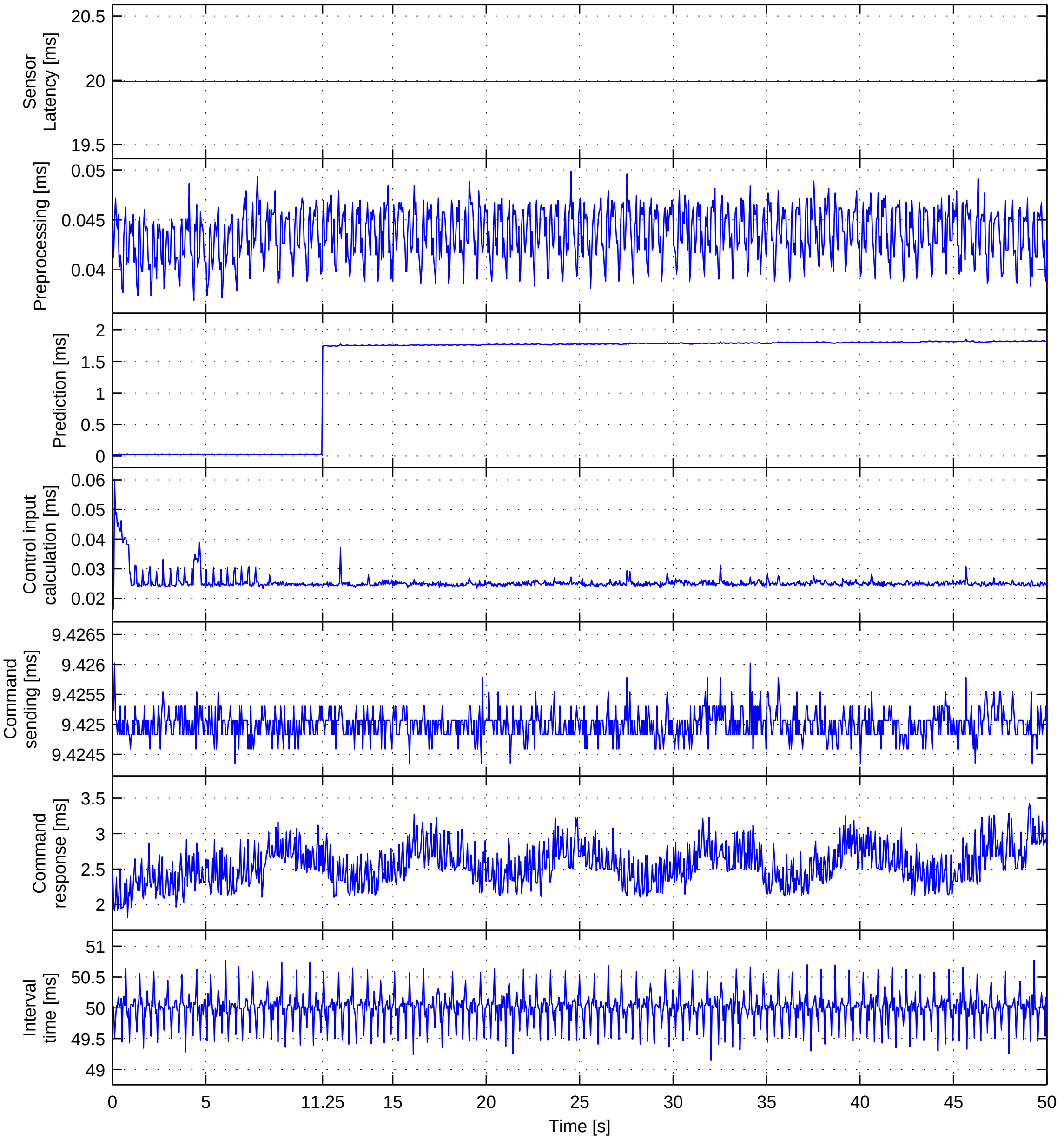


Figure 4.27: *Timing diagrams of important processing steps within* HexGuide *demonstrating real-time operation*

dertaken, requiring less than 50µs according to the second diagram. After that, the chosen predictor is executed to deliver multi-step predictions. Initially, the EKF with spring-mass model is used, taking about 40µs. At the detected peak at time 11.25s the NERLS predictor is used instead of the EKF, requiring about 1.80ms. Since this includes the Takens predictor, the computational time is slightly rising with time as seen in the third diagram. Note that all these latencies until the predictor finished its work is compensated within *HexGuide* as outlined in Section 2.2.3.3. The next diagram shows the computational time of the chosen control method, in this case VMPC was used. The average computational time is, at about 26µs, quite small for solving three MPC problems online (one for each translational axis). When the calculation of the control input is finished, it is sent to the HexaPOD via a serial connection. Due to the limited transmission speed of 57600bps, this takes about 9.425ms, thus, being a major timing limit as opposed to other computational times of the ATTS. Sending the control input, technically concludes each calculation cycle, however, the HexaPOD responds to each command. As can be seen in the sixth diagram, this takes about 2ms to 3ms, where the response time seems to periodically vary along with the HexaPOD's trajectory. Reception of the response is handled asynchronously, so that a new calculation cycle can already begin right after the control input has been sent. The sums of computational delays from reception of a breathing or tumor position measurement sample (triggering the start of a calculation cycle) until the control input has been sent, are well below 16ms (even for the HexaPOD evo). Even the response time of the HexaPOD could be included into this limit. Thus, the system can still be operated safely at a rate of 60Hz. In case of a longer treatment session, the history of the Takens predictor eventually needs to be limited to achieve a constant prediction time. Once a calculation cycle has been started, it will never be interrupted by other tasks in the real-time system, as the scheduler is locked until the calculation cycle finishes. After a calculation cycle is complete, the system waits until the next sampling instant. Given a specific setting of the overall sampling rate, the real-time system has to ensure that a new calculation cycle is started. This is performed in a protected manner as described in Section 2.2.1.5, such that a calculation cycle is started after a configurable timeout period in case no measurement has been received from sensors within that timeout. The corresponding interval times are depicted in the last diagram of Figure 4.27. Ideally, the time difference between two sampling instants would be 50ms due to a sampling rate setting of 20Hz. However, there are deviations from this, mainly within ±0.5ms which are caused by the internal working principle of the USB host controller on the PC (cf. Section 2.2.3.1). Nevertheless, the mean value of the interval time is 50.000ms, thus demonstrating that in each sampling instant, a calculation cycle is started. During all tests in the context of this work, the timing limit of 16ms was never violated (even though 50ms is enough for the employed update rate of 20Hz), so that the overall system can operate in a safe way, guaranteeing a proper response in the required amount of time.

The computational times of control methods IMPC, VPC and VMPC running on the real-time operating system are given in Table 4.12 in terms of average time and standard deviation when performing 3D breathing motion compensation. For IMPC and VMPC, computational time is exemplified for the prediction horizons $N_p = 5$ and $N_p = 10$ with $N_u = N_p$. Since the number of constraints for both IMPC and VMPC is $2(N_p + N_u)$, computational time increases with horizon parameters. Computational time additionally depends on the current state and on the reference trajectory, such that varying computational times need to be anticipated. This can, for example, be seen during the first

Control Method	Computational time [µs]	
	Horizon $N_p = 5$	Horizon $N_p = 10$
IMPC	37.616±4.830	49.432 ±11.381
VPC	11.789 ±0.688	
VMPC	26.155±1.701	33.536 ±1.040

Table 4.12: *Computational time of control methods*

few seconds in the previous example with VMPC, where computational times are larger during the transition phase. However, for any IMPC and VMPC variant, computational time never exceeded 500µs. Since VMPC deals with a one-dimensional problem for one axis and IMPC includes two dimensions, the computational times for VMPC are smaller than for MPC based on an identified second-order model. The computational time for VPC is constant as it does not depend on any parameter or input data. With an average of 11.789µs, computational time is small enough to be neglected in the overall system. The same is true for VMPC and IMPC with $N_p = 5$, where the average computational times are about two to three times larger than for VPC. The standard deviations become larger as computational time increases, however they remain constrained to a few microseconds.

4.3.3 Outlook on HexaPOD evo

The HexaPOD evo possesses the same control interface as the HexaPOD but its speed limit is, with 16mm/s, twice as large. Therefore, a significant increase in tracking accuracy can be expected in situations where tracking accuracy is constrained by the speed limit of 8mm/s of the HexaPOD.

To illustrate the possibilities, the following preliminary experiment was conducted with the HexaPOD evo. A small 6-DoF robotic manipulator depicted in Figure 4.28 was placed on the HexaPOD evo. An extended arm was attached to the robot's flange carrying an IR tool (not seen in the figure). A separate IR tool was attached to the HexaPOD. The robot was instructed to perform a sinusoidal motion of ±3mm along the x-axis (LR) and ±6mm along the y-axis (SI, pointing towards the gantry) with a frequency of 0.25Hz. The maximum speed of 9.42mm/s exceeds the limit of the HexaPOD. The Takens predictor was used for this test in combination with VMPC. Figure 4.29 shows the relevant quantities during compensation. The desired position (red line) is the negated difference between measurements of the IR tool on the manipulator and the IR tool on the HexaPOD (i.e. the displacement relative to the HexaPOD to be compensated). The blue line shows the actual position of the HexaPOD and the green line resembles the overall error between desired and actual positions which could not be compensated for.

Compensation was started at time 0s, although no motion was exhibited by the manipulator. Until motion begins, the controller sufficiently damps measurement noise, keeping the HexaPOD evo at the desired location. After the manipulator starts moving at time 3.7s, a phase shift between actual and desired positions is observed persisting until time 8.8s. This stems from the Takens predictor, which needs at least one full motion period to produce better matching predictions. The transition phase can be eliminated by employing additional predictors as discussed in the previous section. After that, the controller maintains tracking with small overall residual errors on both axes. When the manipulator

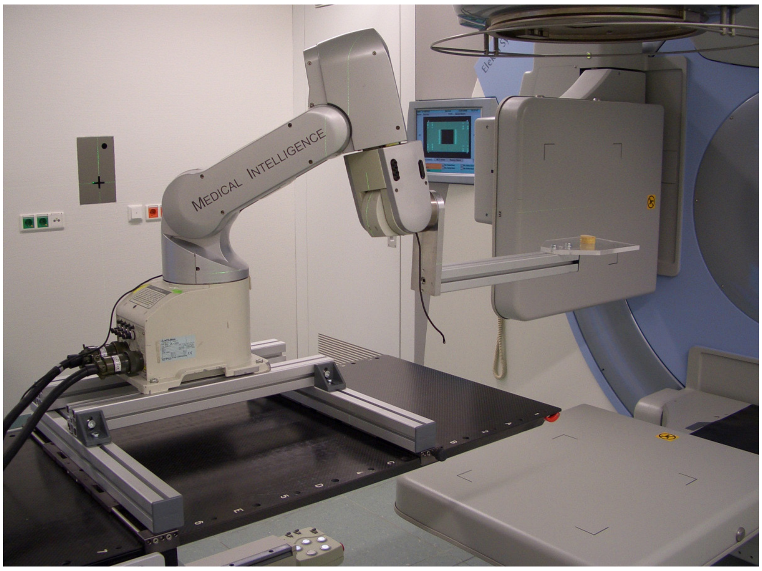


Figure 4.28: *6-DoF robotic manipulator with an extended arm including tumor phantom standing on the HexaPOD evo*

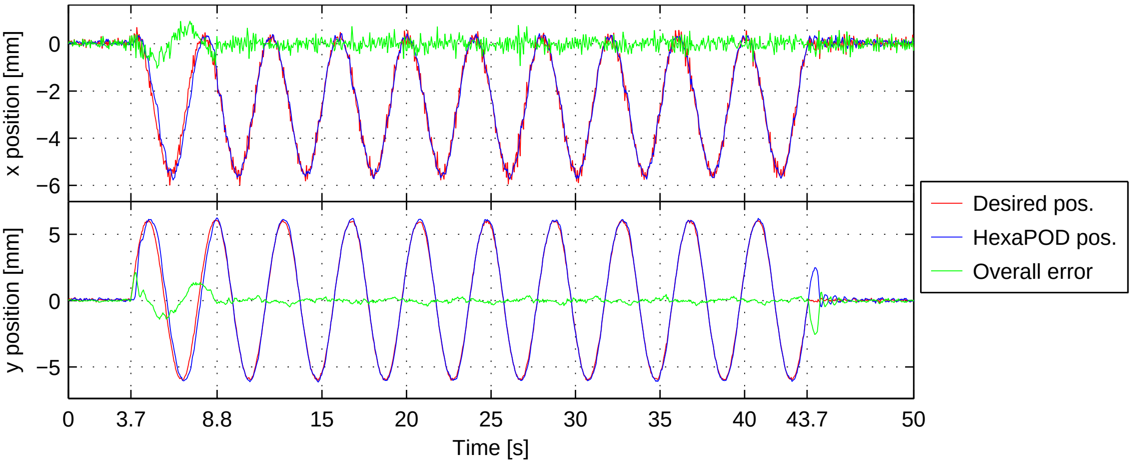


Figure 4.29: *2D motion compensation result of a fast sinusoidal trajectory with HexaPOD evo where motion was exhibited by a small robotic manipulator*

stops moving at time 43.7s, a short-term increase of the error occurs. The duration of this increase is determined by the amount of time the predictor takes to adapt to the new situation without motion.

Although the manipulator follows a smooth sinusoidal trajectory along both axes, high frequency oscillation can be observed on the red line along the x-axis, i.e. the difference between the measurement of both IR tools. Since these oscillations are not present on the direct measurements of the HexaPOD (represented by the blue line), they originate from the IR tool attached to the manipulator. As stiffness of serial manipulators is known to be less than for parallel manipulators, this manipulator showed a significant backlash in the last three joints which mainly acts along the x-axis. Hence, a moving HexaPOD evo introduces these oscillations. However, as demonstrated by the achieved results, VMPC does not amplify these oscillations in its feedback loop, thus maintaining tracking. When the manipulator stopped its motion, small oscillations can be observed in Figure 4.29 also along the y-axis. These oscillations decay within a few seconds, thus demonstrating a stable controller even under the rather difficult condition of a manipulator with backlash.

4.4 Summary

Initial modeling of the HexaPOD's dynamic response was performed using linear ARX models. It was seen that these models, when identifying their parameters from appropriately designed input signals, are capable of describing the dynamics as long as the velocity of the HexaPOD is not saturated. If that is the case, then the nonlinearly extended ARX model is able to cover resulting nonlinearities. The importance of properly performed system identification procedures in order to gain successful models became obvious in comparison to related modeling approaches.

As a first approach for controlling the HexaPOD, the identified ARX model was incorporated into an MPC with which reference tracking was shown to be feasible. However, the VPC approach outperformed the MPC approach in several scenarios. With VPC, near perfect tracking of ideal sinusoidal trajectories was realized. With pre-recorded breathing trajectories tracking performance was still acceptable, however with limited smoothness and patient comfort. With compensation of breathing motions of volunteers lying unconstrained on the HexaPOD, patient comfort became more constrained for some of the volunteers. While tracking performance was still better than with IMPC, the achieved smoothness in VPC was not acceptable for some volunteers. To further improve the VPC approach, it was realized that the approach already contained a simple inverse model of the HexaPOD. This was rewritten as a state-space model, enabling the adoption of it into an MPC which allows optimization over a future horizon. The resulting VMPC approach was then evaluated in the same scenarios as the VPC approach. It was observed that with VMPC, tracking accuracy and smoothness can be traded-off at a fine granular level. While sufficient smoothness could be achieved to make all volunteers feel comfortable, VMPC additionally showed improved tracking performance compared to other methods.

To summarize, with the development of the VMPC approach a unique and powerful control method is available for the HexaPOD, solving patient comfort issues of other methods. Test scenarios were chosen to be more challenging than during normal treatments, where fixation systems help to reduce oscillation and feedback effects. Even under these more difficult conditions, VMPC showed exceptionally good performance so that motion compensation by the HexaPOD is ultimately realized with small residual tracking errors. The adoption of the HexaPOD evo promises further reduction of tracking errors due to a larger speed capabilities covering almost all observable breathing and tumor motions.

Chapter 5

Conclusions

This work faced three major challenges: the realization of a simple, cost-efficient and flexible real-time motion compensation system, predictors for gaining breathing and tumor motion information about the near future and appropriate control methods to actuate the HexaPOD in order to achieve motion compensation.

The simple, cost-efficient and flexible architecture of the real-time tumor motion compensation system was realized by adopting standard hardware device often found in treatment rooms. The intelligence of the system is concentrated in the developed software, which interfaces with all necessary hardware devices for breathing and tumor position determination. The software fuses the hardware devices in a new way for enabling motion compensation. Real-time capabilities were achieved by employing a real-time operating system, guaranteeing that all required calculations during each sampling time are finished before the next sampling instant. Furthermore, the actual time of all acquired and generated data is respected and taken into account in the adaptive tumor tracking system (ATTS). Additionally, system-induced latencies are completely compensated for. This allows for proper interaction and timing between the software and physical world in order to lay the foundation for real-time motion compensation.

The ATTS is not restricted to the employed devices used for breathing and position determination in this work. It is built in a flexible way to be open to other devices as well. For example during the work for comparing our approach to a MLC-based tracking system as outlined in [30], the Calypso system was used to acquire tumor position information. With an appropriate time stamping mechanism, generally any tumor position determination system can be adopted by the ATTS which delivers measurements regularly or quasi-regularly.

The supported compensation modes (recent displacement compensation, drift motion compensation and full motion compensation) allow compensation for any moving tumor, whether its motion is predictable (e.g. for lung tumors) or unpredictable (e.g. for prostate tumors). The only requirement is some means to acquire tumor position information online. Hence, the ATTS in its current implementation possesses all the necessary prerequisites to compensate motions of tumors other than lung tumors as well. With the ATTS our group demonstrated recently in [160], that motion compensation of prostate tumors is possible with submillimeter geometric accuracy and high dosimetric accuracy. Nevertheless, breathing-induced lung tumor motion compensation is considered to be the most challenging problem due to relatively large speeds and amplitudes found for lung tumors.

In order to tackle this problem, short-term predictions of breathing and tumor motion are required. Therefore, several prediction methods were investigated and their performance compared under equal conditions by various metrics. It was shown that each predictor possesses different advantages and disadvantages in terms of prediction accuracy, smoothness, dead time and transition phases, making each of them applicable in different situations or in different phases during the course of a treatment field. It was discussed which sequence of predictors can be used to eliminate the dead time completely, that is to have a full set of predictions already available within the very first sampling instant. This is especially useful for the relatively short treatment fields encountered in the treatment of lung tumors. Furthermore, it was shown how some of the predictors can be combined to deliver improved and stable predictions. A unique way of fusing breathing and tumor position measurements without the need of correlation models was presented to allow for tumor motion predictions especially under the difficult conditions of EPID-based tracking.

To complete the ATTS, the last challenge was the creation of control methods to make the HexaPOD follow the reference trajectory which is formed by tumor motion predictions. With the IMPC controller based on a model gained by system identification methods, considerable tracking performance could already be achieved. The major breakthrough was the development of the VPC method with which near-perfect tracking of ideal reference trajectories is possible. However, when compensating motions of humans lying on the HexaPOD, severe problems in terms of patient comfort were encountered, which simultaneously constrained tracking accuracy. By realizing that the VPC approach actually contained a simple inverse model of the HexaPOD, VPC could be incorporated into a MPC, resulting in the VMPC method. With the optimization characteristics of IMPC, the VMPC approach ultimately solved the oscillations and feedback problems. Thus, precise and smooth tracking of human motion was achieved even in difficult scenarios where human volunteers lay unconstrained on the HexaPOD. It was found that VMPC can be used with a single parameter setting to reach these goals, so that an easily applicable controller resulted. As could be seen, tracking performance is mainly limited by the maximum speed of the HexaPOD. But even with the HexaPOD variant (which was initially not intended to perform continuous motion), considerable motion compensation could be achieved. With the HexaPOD evo, this speed limitation became less severe, so that for almost all observed tumors the HexaPOD evo is able to follow their trajectories.

However, the main limitation in this approach as well as in others are still prediction accuracies. While short-term predictions of a few sampling times are achievable with high accuracy, prediction performance degrades with increasing prediction horizon. Since in this work, the controller computes the current control input to the HexaPOD also based on mid-term predictions, an improvement of these would further benefit overall system performance. Hence, the topic of more advanced predictors is subject of future work.

The integration of EPID-based real-time tumor tracking is currently pursued by the Department of Radiation Oncology of the University of Würzburg. While offline tracking is already possible, the online version would complement the ATTS to have a mean of tumor position determination available which allows for completely non-invasive treatments without implanted markers or additional x-rays for imaging the tumor. Furthermore, current activities also include investigations towards ultrasound-based tumor tracking systems applicable to prostate tumors. If they become available, they can be easily integrated into the ATTS to open up further treatment possibilities.

Further steps also include the adoption of motion compensation in treatment planning by physicians, to complete the work flow for treatments. This will enable phantom as well as clinical studies to evaluate the effectiveness of the approach regarding achievable reduction of margins and resulting dose distributions in the treatment volume.

Initial investigations [158] with the ATTS in various scenarios using breathing-correlated as well as breathing-uncorrelated motions indicate that continuous motion is generally tolerated well by volunteers and irregular breathing patterns are limited. However, more investigations need to be performed to elaborate on the impact of continuous HexaPOD motions on patient's breathing patterns, which is a concern [30, 36] often raised in the context of couch-based tumor motion compensation. Especially, the advanced controllers developed in this work need to be evaluated in depth in terms of patient tolerance on a larger scale.

Despite the increase of the absolute five year survival rate of patients suffering from lung tumors between 1984 and 2008 from 7% to 14% [161], [2] describes the survival rate of patients suffering from lung tumors as “still frustrating”. In a study [162] from 2012 carried out in Germany, relative five year survival rates for 2007-2008 are specified as 19% for females and 15% for males. In 2009, 225000 females and 256000 males fell ill with cancer in Germany in total [163]. From them, lung cancer was newly diagnosed for 15000 females and 34000 males. Apart from breast cancer in females and prostate cancer in males, lung cancer is the third most frequent cancer diagnosis in Germany. The importance is further supported by the high mortality rates. From 100000 females and 112000 males dying from cancer in 2010 in Germany, 14000 females and 29000 males died from lung cancer [163]. Most recent European mortality predictions [164] for 2013 state that “in a few years lung cancer will likely become the first cause of cancer mortality in women as well, overtaking breast cancer”.

Despite an increasing number of cancer patients is surviving due to improved and more precise treatments [165], further improvements are necessary – and possible – especially for moving tumors. This work paved the way for realizing a tumor motion compensation system which is easy to implement into existing treatment rooms due to sole use of standard hardware, thus providing a competitive system as an alternative to other fully integrated treatment systems.

Appendix A

Patient Data Examples

The following figures exemplify four breathing and tumor motion datasets to visualize certain characteristics of these datasets. The data was acquired by the ATTS from patients while they were undergoing regular radiation therapy treatments. In each figure the first three plots depict breathing motion along the x- (LR), y- (SI) and z- (AP) axes. The maximum amplitude range in the data is annotated in each plot's title. The last three plots correspond to the linearly interpolated tumor motion acquired by the EPID and transformed to the same reference system as the breathing motion by appropriate rotation with the gantry angle.

- Figure A.1 shows a rather regular breathing and tumor motion trace with only little drift included.
- The traces seen in Figure A.2 are similarly regular. However, tumor motion data at seconds 10-13 and 30-33 show the effect of the low (and not equidistant) sampling rates. The linearly interpolated signal then follows a different path as the actual tumor (as suggested by previous motion paths and assumed correlation to breathing).
- Figure A.3 depicts one of the longer datasets. Motion traces are found to be more irregular with several short-term phase and amplitude variations.
- The traces given in Figure A.4 show a varying and slowly oscillating drift in combination with larger amplitude variations before and after second 27.

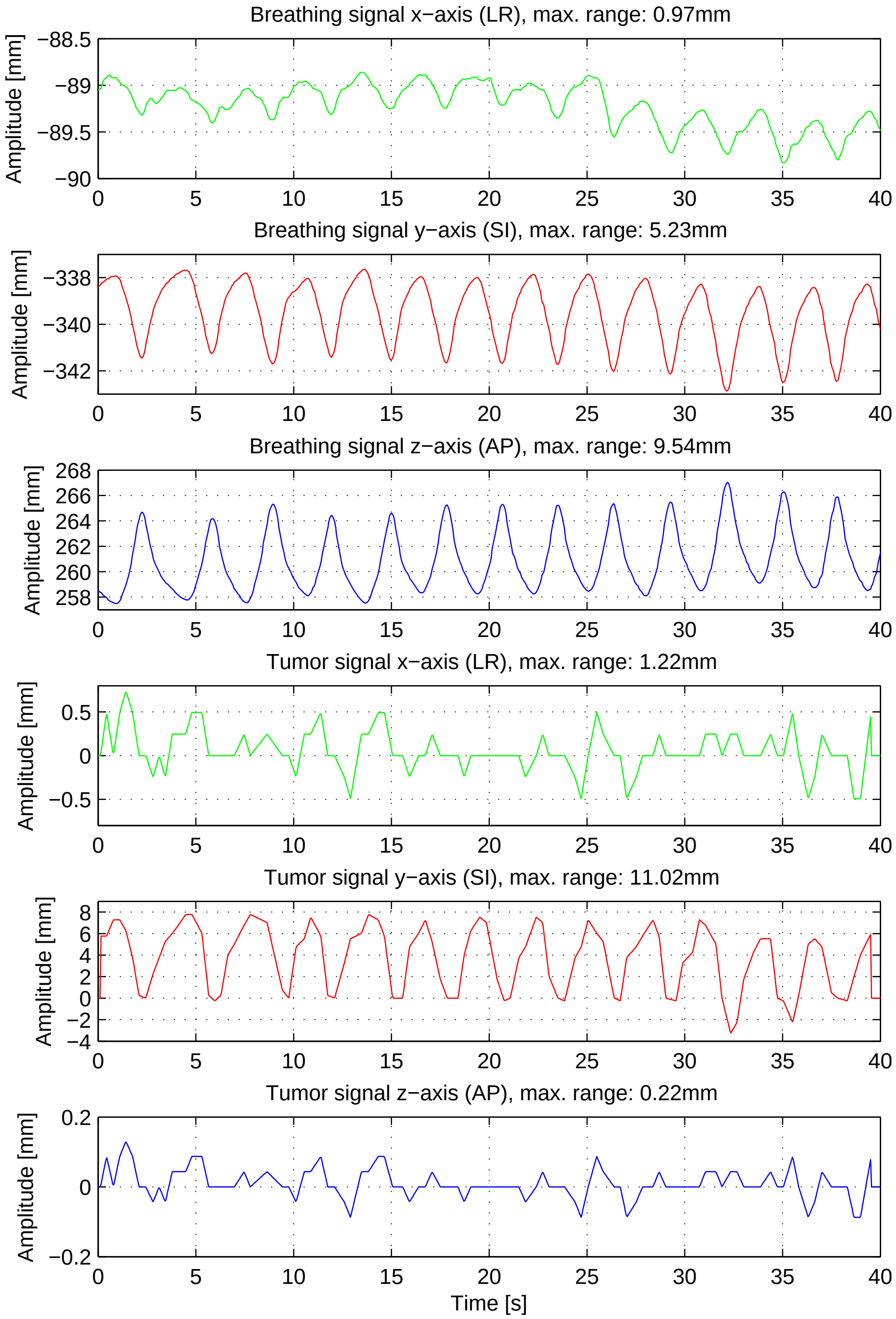


Figure A.1: *Breathing and tumor motion of patient 1 during one field with gantry angle of 190°*

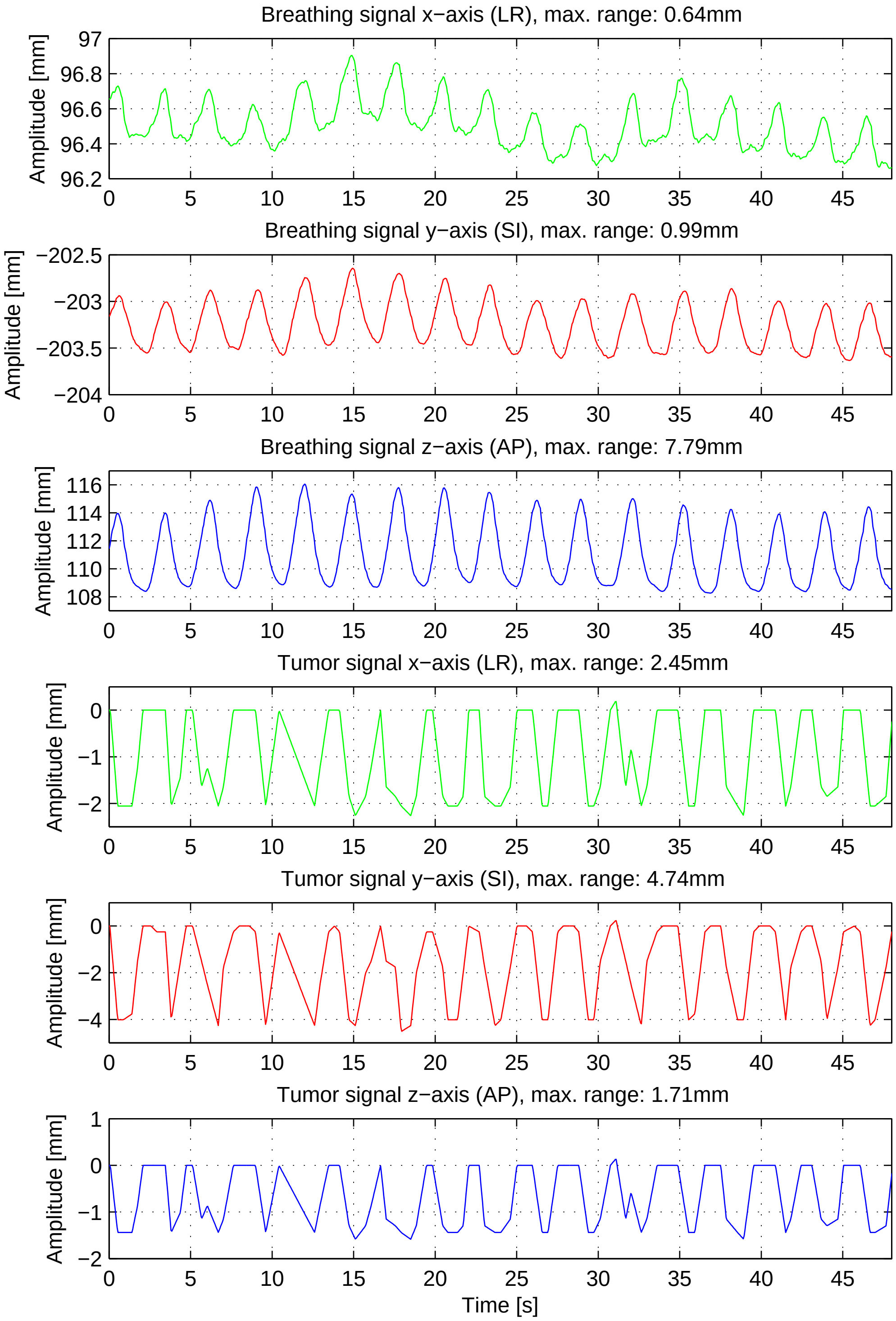


Figure A.2: *Breathing and tumor motion of patient 4 during one field with gantry angle of 35°*

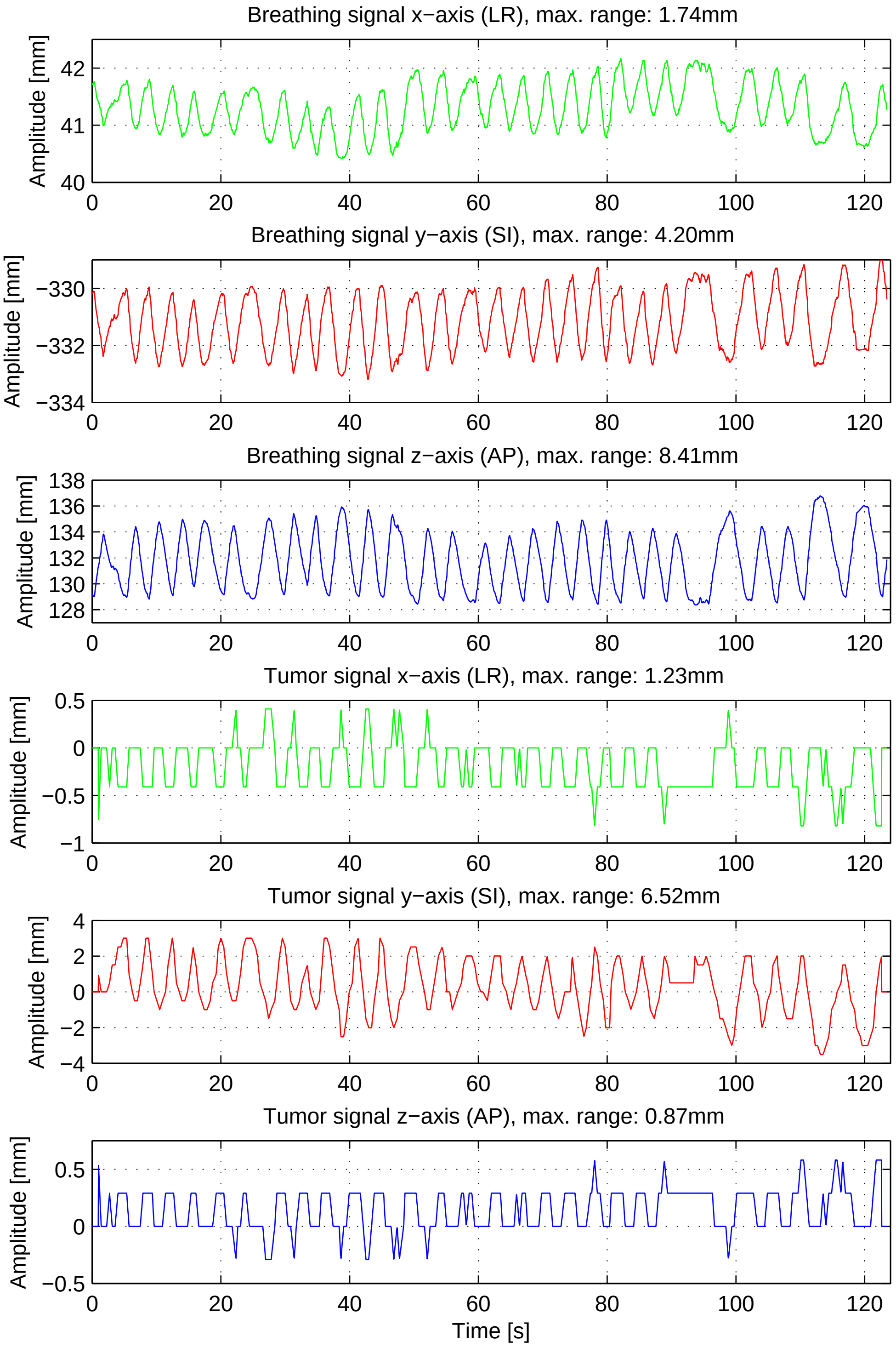


Figure A.3: *Breathing and tumor motion of patient 32 during one field with gantry angle of 215°*

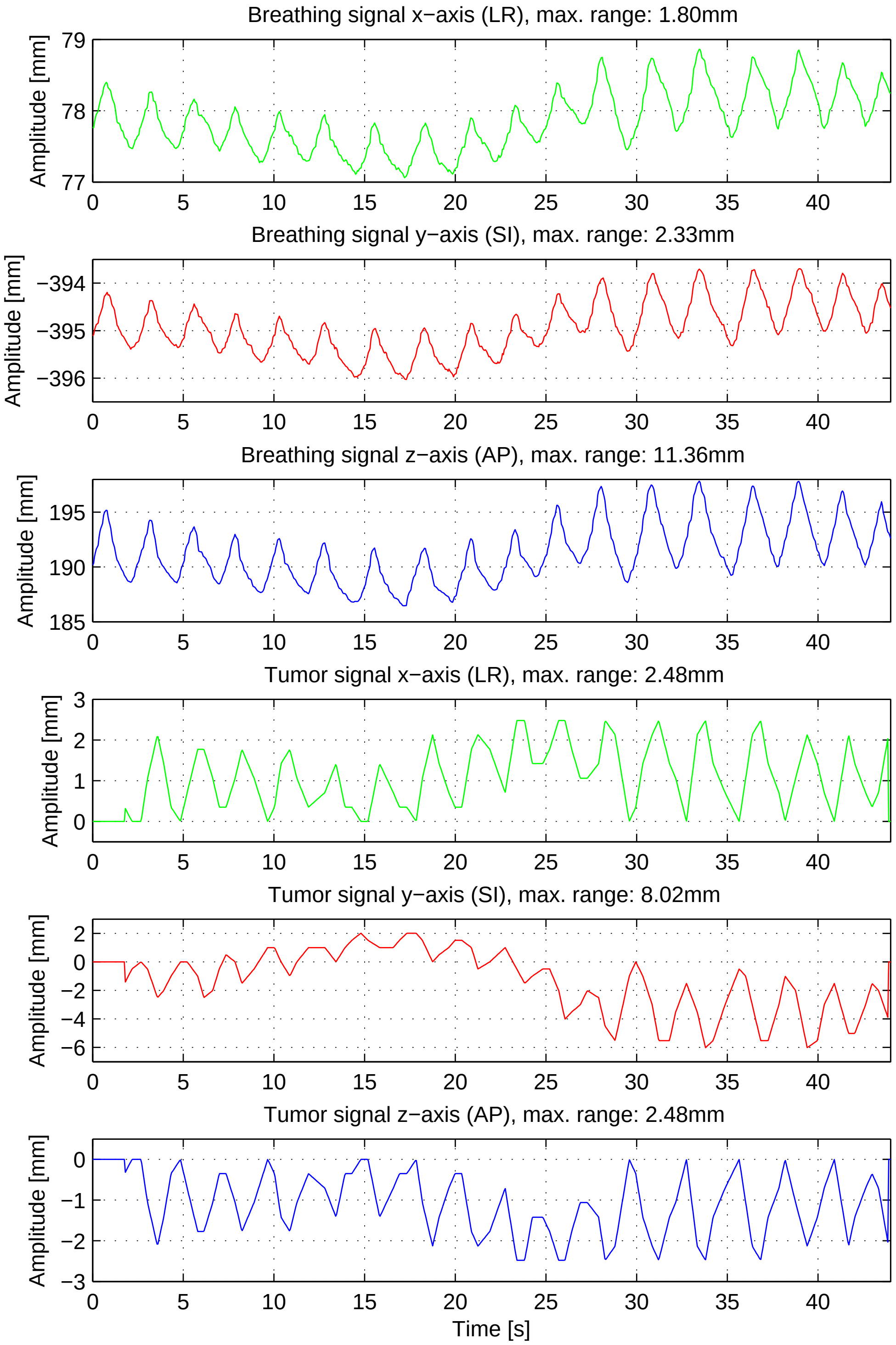


Figure A.4: *Breathing and tumor motion of patient 40 during one field with gantry angle of 225°*

Appendix B

Smoothness Measures

The smoothness measures defined in Section 3.2.2 are compared in various scenarios below. All the considered smoothness measures are sensitive to phase shifts and certain kinds of drifts, i.e. the measures result in different values when applied to an ideal smooth trajectory and either a phase shifted or drifting version of it. In order to avoid misjudgment of smoothness, the influence of phase changes and drifts is studied here and compared to the case of a noisy signal, thus considering the non-smooth case. The forthcoming analysis assumes a smooth cosine signal

$$y_s(\tau) = a\cos\left(2\pi f\tau\right)$$

defined over the finite time $0 \leq \tau \leq 60$. The phase-shifted version uses the same amplitude and frequency but introduces a phase shift of 45°:

$$y_p(\tau) = a\cos\left(2\pi f\tau + \frac{\pi}{4}\right) \ .$$

A small linear drift is superimposed on the smooth signal, yielding

$$y_{d_1}(\tau) = a\cos\left(2\pi f\tau\right) + 0.1\tau \ ,$$

whereas a periodic drift is incorporated by adding a cosine with a low frequency:

$$y_{d_2}(\tau) = a\cos\left(2\pi f\tau\right) + 2\cos\left(2\pi \cdot 0.01 \cdot \tau\right) \ .$$

The noisy reference signal $y_n(\tau)$ is gained by adding a pseudo-random noise $n(\tau)$ drawn from the standard normal distribution:

$$y_n(\tau) = a\cos\left(2\pi f\tau\right) + 0.5n(\tau) \ .$$

For two different settings, both of the amplitude a and frequency f, all smoothness criteria were evaluated on the discretized test signals $y_s(\tau)$, $y_p(\tau)$, $y_{d_1}(\tau)$ and $y_{d_2}(\tau)$, yielding certain smoothness values $S\left(y_s\right)$, $S\left(y_p\right)$, $S\left(y_{d_1}\right)$ and $S\left(y_{d_2}\right)$. These values were then set in relation to the noisy case $S\left(y_n\right)$ by calculating

$$s = \frac{S\left(y\right)}{S\left(y_n\right)} \cdot 100\% \ , \tag{B.1}$$

where y is one of y_s, y_p, y_{d_1} or y_{d_2}. Note that for the calculation of s for S_{SNR}, the numerator and denominator in Eq. (B.1) have to be exchanged because larger values indicate smoother signals here. Ideally, the ratio s should be zero or at least as small as possible, meaning that the measure is not sensitive to phase shifts or drifts. Table B.1

shows s for the considered measures and scenarios. From each measure applied on the ideal signal y_s it is obvious, that none of the criteria can be used for comparing different signals, as varying amplitudes or frequencies lead to significantly different smoothness measures even in the case of already smooth signals. A similar situation arises for phase shifted signals or signals with drifts. All measures show varying smoothness values which also depend on amplitude and frequency of the underlying signal. Nevertheless, the measures exhibit ratios in a certain range of values. As can be seen, the average ratios decrease from S_{SNR} to S_{TS} to S_J and to S_F, in which the ratios for S_F are very small compared to the rest of the measures. This means that S_F is insensitive to phase shifts or drifts, so that this measure has the potential to describe smoothness in a way which comes closest to what is intuitively understood by smoothness.

Measure	a [mm]	f [Hz]	Ideal [%]	Phase Shift [%]	Linear Drift [%]	Periodic Drift [%]	Rating
S_{SNR} (Eq. (3.16))	4	0.49	68.4	54.9	44.6	76.1	+
	4	0.28	52.4	60.2	49.6	64.7	
	8	0.49	86.2	69.2	71.7	92.6	
	8	0.28	71.1	81.6	50.0	81.1	
S_{TS} (Eq. (3.15))	4	0.49	58.6	58.3	53.2	54.8	++
	4	0.28	37.7	37.1	34.7	35.0	
	8	0.49	83.8	83.4	81.7	82.4	
	8	0.28	64.2	63.3	63.1	63.0	
S_J (Eq. (3.14))	4	0.49	32.5	32.5	25.9	27.5	+++
	4	0.28	14.0	14.0	9.8	10.7	
	8	0.49	35.1	35.1	31.0	32.1	
	8	0.28	15.2	15.1	12.5	13.1	
S_F (Eq. (3.17))	4	0.49	0.2	0.2	0.1	0.1	++++
	4	0.28	0.1	0.1	0.0	0.0	
	8	0.49	0.3	0.3	0.2	0.3	
	8	0.28	0.1	0.1	0.1	0.1	

Table B.1: *Comparison of different smoothness measures in terms of signal variations*

List of Abbreviations

ADC	analog to digital converter	43
ANN	artificial neural network	86
AP	anterior-posterior	13
AR	autoregressive	85
ARMA	autoregressive moving average	86
ARX	autoregressive model with exogenous input	100
ATTS	adaptive tumor tracking system	6
CA	constant acceleration	124
CE	control effort	196
CDF	cumulative distribution function	58
CI95	95% confidence interval	70
CR	compensation ratio	168
CT	computed tomography	1
CTV	clinical target volume	2
CV	constant velocity	120
DFFT	discrete fast Fourier transform	54
DMC	drift motion compensation	10
EKF	extended Kalman filter	129
EPID	electronic portal imaging device	11
ERLS	extended recursive least-squares	116
FMC	full motion compensation	10
GTV	gross tumor volume	2
IMPC	model predictive controller including identified ARX model	169
IQR	inter quartile range	57
IR	infrared	16
ITV	internal target volume	2
LMS	least-mean-square	85
LR	left-right	13
MAE	mean absolute error	70
MAX	maximum	78
MLC	multi-leaf collimator	1

MPC	model predictive controller	169
MSE	mean-square error	70
NERLS	nonlinearly extended recursive least-squares	143
NRT	non real-time	41
OCR	overall compensation ratio	204
PR	prediction ratio	70
PTV	planning target volume	2
QP	quadratic program	175
RDC	recent displacement compensation	9
RLS	recursive least-squares	86
RMS	root-mean-square	55
RSED	relative speed exceedance duration	58
RT	real-time	41
SI	superior-inferior	13
SISO	single-input single-output	115
SNR	signal-to-noise ratio	72
STD	standard deviation	41
TCP	transmission control protocol	24
TCP/IP	transmission control protocol / internet protocol	15
UART	universal asynchronous receiver/transmitter	41
UDP	user datagram protocol	24
USB	universal serial bus	17
VPC	velocity-based position control	178
VMPC	model predictive control of velocity model	195

List of Figures

List of Tables

References

[1] W.C. Röntgen. **Über eine neue Art von Strahlen**. Sitzungsberichte der physikalisch-medizinischen Gesellschaft zu Würzburg, 1896.

[2] R. Sauer. ***Strahlentherapie und Onkologie***. Elsevier GmbH, Munich, 5th edition, 2010.

[3] M. Guckenberger, J. Meyer, J. Wilbert, K. Baier, G. Müller, J. Wulf, and M. Flentje. **Cone-beam CT based image-guidance for extracranial stereotactic radiotherapy of intrapulmonary tumors**. *Acta Oncologica*, 45(7):897–906, 2006.

[4] International Commission on Radiation Units and Measurements. ***ICRU Report 50: Prescribing, Recording, and Reporting Photon Beam Therapy***. ICRU, Bethesda, MD, USA, 1993.

[5] International Commission on Radiation Units and Measurements. ***ICRU Report 62: Prescribing, Recording, and Reporting Photon Beam Therapy (Supplement to ICRU Report 50)***. ICRU, Bethesda, MD, USA, 1999.

[6] K.M. Langen and D.T.L. Jones. **Organ motion and its management**. *International Journal of Radiation Oncology Biology Physics*, 50(1):265–278, 2001.

[7] M. Guckenberger, J. Wilbert, J. Meyer, K. Baier, A. Richter, and M. Flentje. **Is a single respiratory correlated 4D-CT study sufficient for evaluation of breathing motion?** *International Journal of Radiation Oncology Biology Physics*, 67(5):1352–1359, 2007.

[8] M.K. Martel, R.K. Ten Haken, M.B. Hazuka, M.L. Kessler, M. Strawderman, A.T. Turrisi, T.S. Lawrence, B.A. Fraass, and A.S. Lichter. **Estimation of tumor control probability model parameters from 3-D dose distributions of non-small cell lung cancer patients**. *Lung Cancer*, 24(1):31–37, 1999.

[9] J. Willner, K. Baier, E. Caragiani, A. Tschammler, and M. Flentje. **Dose, volume, and tumor control prediction in primary radiotherapy of nonsmall-cell lung cancer**. *International Journal of Radiation Oncology Biology Physics*, 52(2):382–389, February 2002.

[10] P. J. Keall, G. S. Mageras, J. M. Balter, R. S. Emery, K. M. Forster, S. B. Jiang, J. M. Kapatoes, D. A. Low, M. J. Murphy, B. R. Murray, C. R. Ramsey, M. B. Van Herk, S. S. Vedam, J. W. Wong, and E. Yorke. **The management of respiratory motion in radiation oncology report of AAPM task group 76**. *Medical Physics*, 33(10):3874–3900, 2006.

[11] K. Ohara, T. Okumura, M. Akisada, T. Inada, T. Mori, H. Yokota, and M.J.B. Calaguas. **Irradiation synchronized with respiration gate**. *International Journal of Radiation Oncology Biology Physics*, 17(4):853–857, 1989.

[12] H.D. Kubo and B.C. Hill. **Respiration gated radiotherapy treatment: A technical study**. *Physics in Medicine and Biology*, 41(1):83–91, 1996.

[13] H. Shirato, S. Shimizu, T. Kunieda, K. Kitamura, M. van Herk, K. Kagei, T. Nishioka, S. Hashimoto, K. Fujita, H. Aoyama, K. Tsuchiya, K. Kudo, and K. Miyasaka. **Physical aspects of a real-time tumor-tracking system for gated radiotherapy**. *International Journal of Radiation Oncology Biology Physics*, 48(4):1187–1195, 2000.

[14] G.S. Mageras, E. Yorke, K. Rosenzweig, L. Braban, E. Keatley, E. Ford, S.A. Leibel, and C.C. Ling. **Fluoroscopic evaluation of diaphragmatic motion reduction with a respiratory gated radiotherapy system**. *Journal of Applied Clinical Medical Physics*, 2(4):191–200, 2001.

[15] E.C. Ford, G.S. Mageras, E. Yorke, K.E. Rosenzweig, R. Wagman, and C.C. Ling. **Evaluation of respiratory movement during gated radiotherapy using film and electronic portal imaging**. *International Journal of Radiation Oncology Biology Physics*, 52(2):522–531, 2002.

[16] J. Hanley, M.M. Debois, D. Mah, G.S. Mageras, A. Raben, K. Rosenzweig, B. Mychalczak, L.H. Schwartz, P.J. Gloeggler, W. Lutz, C.C. Ling, S.A. Leibel, Z. Fuks, and G.J. Kutcher. **Deep inspiration breath-hold technique for lung tumors: The potential value of target immobilization and reduced lung density in dose escalation**. *International Journal of Radiation Oncology Biology Physics*, 45(3):603–611, 1999.

[17] J.W. Wong, M.B. Sharpe, D.A. Jaffray, V.R. Kini, J.M. Robertson, J.S. Stromberg, and A.A. Martinez. **The use of active breathing control (ABC) to reduce margin for breathing motion**. *International Journal of Radiation Oncology Biology Physics*, 44(4):911–919, 1999.

[18] M.J. Murphy, D. Martin, R. Whyte, J. Hai, C. Ozhasoglu, and Q.T. Le. **The effectiveness of breath-holding to stabilize lung and pancreas tumors during radiosurgery**. *International Journal of Radiation Oncology Biology Physics*, 53(2):475–482, 2002.

[19] H. Onishi, K. Kuriyama, T. Komiyama, S. Tanaka, N. Sano, Y. Aikawa, Y. Tateda, T. Araki, S. Ikenaga, and M. Uematsu. **A new irradiation system for lung cancer combining linear accelerator, computed tomography, patient self-breath-holding, and patient-directed beam-control without respiratory monitoring devices**. *International Journal of Radiation Oncology Biology Physics*, 56(1):14–20, 2003.

[20] P.J. Keall, V.R. Kini, S.S. Vedam, and R. Mohan. **Motion adaptive x-ray therapy: A feasibility study**. *Physics in Medicine and Biology*, 46(1):1–10, 2001.

[21] P.J. Keall, H. Cattell, D. Pokhrel, S. Dieterich, K.H. Wong, M.J. Murphy, S.S. Vedam, K. Wijesooriya, and R. Mohan. **Geometric accuracy of a real-time target tracking system with dynamic multileaf collimator tracking system**. *International Journal of Radiation Oncology Biology Physics*, 65(5):1579–1584, 2006.

[22] L. Papiez. **DMLC leaf-pair optimal control of IMRT delivery for a moving rigid target**. *Medical Physics*, 31(10):2742–2754, 2004.

[23] L. Papiez and D. Rangaraj. **DMLC leaf-pair optimal control for mobile, deforming target**. *Medical Physics*, 32(1):275–285, 2005.

[24] L. Papiez, D. Rangaraj, and P. Keall. **Real-time DMLC IMRT delivery for mobile and deforming targets**. *Medical Physics*, 32(9):3037–3048, 2005.

[25] T. Neicu, H. Shirato, Y. Seppenwoodle, and S.B. Jiang. **Synchronized moving aperture radiation therapy (SMART): Average tumor trajectory for lung patients**. *Physics in Medicine and Biology*, 48(5):587–598, 2003.

[26] Y. Liu, C. Shi, B. Lin, C.S. Ha, and N. Papanikolaou. **Delivery of four-dimensional radiotherapy with trackbeam for moving target using an accuknife dual-layer MLC: Dynamic phantoms study**. *Journal of Applied Clinical Medical Physics*, 10(2):2926, 2009.

[27] M. Tacke, S. Nill, and U. Oelfke. **Real-time tracking of tumor motions and deformations along the leaf travel direction with the aid of a synchronized dynamic MLC leaf sequencer**. *Physics in Medicine and Biology*, 52(22):N505–N512, 2007.

[28] M. Tacke, S. Nill, A. Krauss, and U. Oelfke. **Real-time tumor tracking: Automatic compensation of target motion using the Siemens 160 MLC**. *Medical Physics*, 37(2):753–761, 2010.

[29] A. Krauss, S. Nill, M. Tacke, and U. Oelfke. **Electromagnetic real-time tumor position monitoring and dynamic multileaf collimator tracking using a siemens 160 MLC: Geometric and dosimetric accuracy of an integrated system**. *International Journal of Radiation Oncology Biology Physics*, 79(2):579–587, 2011.

[30] M.J. Menten, M. Guckenberger, C. Herrmann, A. Krauß, S. Nill, U. Oelfke, and J. Wilbert. **Comparison of a multileaf collimator tracking system and a robotic treatment couch tracking system for organ motion compensation during radiotherapy**. *Medical Physics*, 39(11):7032–7041, November 2012.

[31] J.R. Adler, M.J. Murphy, S.D. Chang, and S.L. Hancock. **Image-guided robotic radiosurgery**. *Neurosurgery*, 44(6):1299–1306, 1999.

[32] A. Schweikard, G. Glosser, M. Bodduluri, M.J. Murphy, and J. Adler. **Robotic motion compensation for respiratory movement during radiosurgery**. *Computer Aided Surgery*, 5(4):263–277, 2000.

[33] S.D. Chang and J.R. Adler. **Robotics and radiosurgery - the cyberknife**. *Stereotactic and Functional Neurosurgery*, 76:204–208, 2001.

[34] A. Schweikard, H. Shiomi, and J. Adler. **Respiration tracking in radiosurgery**. *Medical Physics*, 31(10):2738–2741, 2004.

[35] R.I. Whyte, R. Crownover, M.J. Murphy, D.P. Martin, T.W. Rice, M.M. Jr. DeCamp, R. Rodebaugh, M.S. Weinhous, and Q.T. Le. **Stereotactic radiosurgery for lung tumors: Preliminary report of a phase I trial**. *The Annals of Thoracic Surgery*, 75(4):1097–1101, 2003.

[36] M.J. Murphy. **Tracking moving organs in real time**. *Seminars in Radiation Oncology*, 14(1):91–100, 2004.

[37] P.C. Gerszten, C. Ozhasoglu, S.A. Burton, W.J. Vogel, B.A. Atkins, S. Kalnicki, and W.C. Welch. **Cyberknife frameless stereotactic radiosurgery for spinal lesions: Clinical experience in 125 cases**. *Neurosurgery*, 5(1):89–98, July 2004.

[38] M. Hoogeman, J. Prévost, J. Nuyttens, J. Pöll, P. Levendag, and B. Heijmen. **Clinical accuracy of the respiratory tumor tracking system of the cyberknife: Assessment by analysis of log files**. *International Journal of Radiation Oncology Biology Physics*, 74(1):297–303, 2009.

[39] W. Kilby, J.R. Dooley, G. Kuduvalli, S. Sayeh, and C.R. Maurer. **The cyberknife robotic radiosurgery system in 2010**. *Technology in Cancer Research and Treatment*, 9(5):433–452, October 2010.

[40] Y. Kamino, K. Takayama, M. Kokubo, Y. Narita, E. Hirai, N. Kawawda, T. Mizowaki, Y. Nagata, T. Nishidai, and M. Hiraoka. **Development of a four-dimensional image-guided radiotherapy system with a gimbaled x-ray head**. *International Journal of Radiation Oncology Biology Physics*, 66(1):271–278, September 2006.

[41] Y. Kamino, S. Miura, M. Kokubo, I. Yamashita, E. Hirai, M. Hiraoka, and J. Ishikawa. **Development of an ultrasmall C-band linear accelerator guide for a four-dimensional image-guided radiotherapy system with a gimbaled x-ray head**. *Medical Physics*, 34(5):1797–1808, May 2007.

[42] N. Linthout, D. Verellen, K. Tournel, T. Reynders, M. Duchateau, and G. Storme. **Assessment of secondary patient motion induced by automated couch movement during on-line 6 dimensional repositioning in prostate cancer treatment**. *Radiotherapy and Oncology*, 83(2):168–174, May 2007.

[43] M. Speiser, P. Medin, W. Mao, L. Papiez, F. Gum, and T. Solberg. **First assessment of a novel IGRT device for stereotactic body radiation therapy**. In O. Dössel and W.C. Schlegel, editors, *World Congress on Medical Physics and Biomedical Engineering*, volume 25/1 of *IFMBE Proceedings*, pages 266–269. Springer Berlin Heidelberg, Munich, Germany, September 7-12, 2009.

[44] K. Takayama, T. Mizowaki, M. Kokubo, N. Kawada, H. Nakayama, Y. Narita, K. Nagano, Y. Kamino, and M. Hiraoka. **Initial validations for pursuing irradiation using a gimbals tracking system**. *Radiotherapy and Oncology*, 93(1):45–49, October 2009.

[45] T. Depuydt, D. Verellen, O. Haas, T. Gevaert, N. Linthout, M. Duchateau, K. Tournel, T. Reynders, K. Leysen, M. Hoogeman, G. Storme, and M. De Ridder. **Geometric accuracy of a novel gimbals based radiation therapy tumor tracking system**. *Radiotherapy and Oncology*, 98(3):365–372, March 2011.

[46] O. Haas, G. Bueno, R. Spriestersbach, H. Himmler, K. Burnham, and J. Mills. **Imaging and control for adaptive radiotherapy**. In *Proceedings of the 16th IFAC World Congress*, pages 25–30, Prague, July 3-8, 2005.

[47] W.D. D'Souza, S.A. Naqvi, and C.X. Yu. **Real-time intra-fraction-motion tracking using the treatment couch: A feasibility study**. *Physics in Medicine and Biology*, 50(17):4021–4033, 2005.

[48] J. Wilbert, J. Meyer, K. Baier, M. Guckenberger, C. Herrmann, R. Hess, C. Janka, L. Ma, T. Mersebach, A. Richter, M. Roth, K. Schilling, and M. Flentje. **Tumor tracking and motion compensation with an adaptive tumor tracking system (ATTS): System description and prototype testing**. *Medical Physics*, 35(9):3911–3921, September 2008.

[49] R. Spriestersbach, O.C.L. Haas, and K.J. Burnham. **Modelling and control of patient support system for radiotherapy**. In *Proceedings of the Control 2004 Conference*, U.K., 2004. University of Bath.

[50] P. Qiu, W.D. D'Souza, T.J. McAvoy, and K.J. Ray Liu. **Inferential modeling and predictive feedback control in real-time motion compensation using the treatment couch during radiotherapy**. *Physics in Medicine and Biology*, 52(19):5831–5854, 2007.

[51] C. Herrmann, L. Ma, K. Schilling, J. Wilbert, and K. Baier. **Tumorverfolgung und Nachführung mittels eines automatisierten Bestrahlungstisches**. In *Proceedings of the 8th Würzburger Medizintechnik-Kongress*, pages 77–80, Würzburg, Germany, May 21-23, 2007.

[52] L. Ma, C. Herrmann, and K. Schilling. **On challenges of robot assisted radiotherapy for lung tumors**. In *Proceedings of the IEEE International Conference on Robotics and Biomimetics (ROBIO 2010)*, pages 773–778, Tianjin, China, December 10-14, 2010.

[53] G. Baroni, G. Ferrigno, R. Orecchia, and A. Pedotti. **Real-time three-dimentional motion analysis for patient positioning verification**. *Radiotherapy and Oncology*, 54(1):21–27, 2000.

[54] J. Meyer, J. Wilbert, K. Baier, M. Guckenberger, A. Richter, O. Sauer, and M. Flentje. **Positioning accuracy of cone-beam computed tomography in combinationwith a hexapod robot treatment table**. *International Journal of Radiation Oncology Biology Physics*, 67(4):1220–1228, 2007.

[55] J.P. Merlet. ***Parallel Robots***, volume 128 of *Solid Mechanics and Its Applications*. Springer, 2006.

[56] C. Herrmann, L. Ma, and K. Schilling. **Modeling a hexapod for tumor motion compensation in robot assisted radiotherapy**. In *Proceedings of the joint conference of 41st International Symposium on Robotics and 6th German Conference on Robotics*, pages 1019–1025, Munich, Germany, June 2010.

[57] L.E. Antonuk. **Electronic portal imaging devices: A review and historical perspective of contemporary technologies and research**. *Physics in Medicine and Biology*, 47(6):R31–R65, 2002.

[58] M. Herman, J. Balter, D. Jaffray, K. McGee, P. Munro, S. Shalev, M. Van Herk, and J. Wong. **Clinical use of electronic portal imaging: Report of AAPM radiation therapy committee task group 58**. *Medical Physics*, 28(5):712–737, 2001.

[59] K. Baier and J. Meyer. **Fast image acquisition and processing on a TV camera-based portal imaging system**. *Zeitschrift Medizinische Physik*, 15(2):122–125, 2005.

[60] K. Kesteloot, A. Dutreix, and E. van der Schueren. **A model for calculating the costs of in vivo dosimetry and portal imaging in radiotherapy departments**. *Radiotherapy and Oncology*, 28(2):108–117, 1993.

[61] M. Aristophanous. **Epid-guided 3D dose verification of lung SBRT**. *Medical Physics*, 38(1):495–503, 2011.

[62] J. Meyer, A. Richter, K. Baier, J. Wilbert, M. Guckenberger, and M. Flentje. **Tracking moving objects with megavoltage portal imaging: A feasibility study**. *Medical Physics*, 33(5):1275–1280, 2006.

[63] A. Richter, J. Wilbert, K. Baier, M. Flentje, and M. Guckenberger. **Feasibility study for markerless tracking of lung tumors in stereotactic body radiotherapy**. *International Journal of Radiation Oncology Biology Physics*, 78(2):618–627, 2010.

[64] L. Ma, C. Herrmann, and K. Schilling. **Modeling and prediction of lung tumor motion for robotic assisted radiotherapy**. In *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2007)*, pages 189–194, San Diego, CA, USA, October 29 - November 2, 2007.

[65] L. Ma, C. Herrmann, K. Schilling, J. Wilbert, and K. Baier. **Real-time tumor tracking with a robotic treatment couch**. In *Proceedings of the 2nd International Conference on Computer Aided Medical Interventions: Tools and Applications (SURGETICA)*, pages 215–221, Chambery, France, 2007.

[66] J. Choi and C. Yoo. **One-way delay estimation and its application**. *Computer Communications*, 28(7):819–828, 2005.

[67] R. Freund and R. Hoppe. ***Stoer/Bulirsch: Numerische Mathematik 1***. Springer-Verlag Berlin Heidelberg, 10th edition, 2007.

[68] D. Ruan, J.A. Fessler, and J.M. Balter. **Mean position tracking of respiratory motion**. *Medical Physics*, 35(2):782–792, 2008.

[69] K.S. Arun, T.S. Huang, and S.D. Blodstein. **Least-squares fitting of two 3-D point sets**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(5):698–700, 1987.

[70] J.H. Challis. **A procedure for determining rigid body transformation parameters**. *Journal of Biomechanics*, 28(6):733–737, 1995.

[71] A.B. Carman and P.D. Milburn. **Determining rigid body transformation parameters from ill-conditioned spatial marker co-ordinates**. *Journal of Biomechanics*, 39(10):1778–1786, 2006.

[72] F.E. Veldpaus, H.J. Woltring, and L.J. Dortmans. **A least-squares algorithm for the equiform transformation from spatial marker coordinates**. *Journal of Biomechanics*, 21(1):45–54, 1988.

[73] A. Silberschatz, P.B. Galvin, and G. Gagne. ***Operating System Concepts***. John Wiley and Sons, Inc., 7th edition, 2005.

[74] K. Yaghmour, J. Masters, G. Ben-Yossef, and P. Gerum. ***Building Embedded Linux Systems***. O'Reilly, 2nd edition, 2008.

[75] J. Angeles. ***Fundamentals of Robotic Mechanical Systems: Theory, Methods, and Algorithms***. Mechanical Engineering Series. Springer, 2nd edition, 2003.

[76] A. Hugelin and J. Vibert. **Is the respiratory rhythm multistable in man?** In G. Swanson and F. Grodins, editors, *Respiratory Control: A modeling perspective*, pages 353–359. Plenum Press, New York, 1989.

[77] Y. Seppenwoolde, H. Shirato, K. Kitamura, S. Shimizu, M. van Herk, J.V. Lebesque, and K. Miyasaka. **Precise and real-time mesurement of 3D tumor motion in lung due to breathing and heartbeat: Measured during radiotherapy**. *International Journal of Radiation Oncology Biology Physics*, 53(4):822–834, 2002.

[78] P. Giraud, Y. De Rycke, B. Dubray, S. Helfre, D. Voican, L. Guo, J.C. Rosenwald, K. Keraudy, M. Housset, E. Touboul, and J.M. Cosset. **Conformal radiotherapy (CRT) planning for lung cancer: Analysis of intrathoracic organ motion during extreme phases of breathing**. *International Journal of Radiation Oncology Biology Physics*, 51(4):1081–1092, 2001.

[79] G. Benchetrit. **Breathing pattern in humans: Diversity and individuality**. *Respiration Physiology*, 122(2–3):123–129, 2000.

[80] S. Moria, M. Endob, S. Komatsuc, T. Yashiroc, S. Kandatsuc, and M. Baba. **Four-dimensional measurement of lung tumor displacement using 256-multi-slice CT-scanner**. *Lung Cancer*, 56(1):59–67, 2007.

[81] F. Gustafsson. **Determining the initial states in forward-backward filtering**. *IEEE Transactions on Signal Processing*, 44(4):988–992, 1996.

[82] W. Gander, G.H. Golub, and R. Strebel. **Least-squares fitting of circles and ellipses**. *BIT Numerical Mathematics*, 34(4):558–578, 1994.

[83] W. Gander, G.H. Golub, and R. Strebel. **Fitting of circles and ellipses – least squares solution**. Technical Report 217, Institut für Wissenschaftliches Rechnen, ETH Zürich, June 1994.

[84] F. Bookstein. **Fitting conic sections to scattered data**. *Computer Graphics and Image Processing*, 9(1):56–71, 1979.

[85] P.L. Rosin. **Analysing error of fit functions for ellipses**. *Pattern Recognition Letters*, 34(14):1461–1470, 1996.

[86] C. Ozhasoglu and M.J. Murphy. **Issues in respiratory motion compensation during external-beam radiotherapy**. *International Journal of Radiation Oncology Biology Physics*, 52(5):1389–1399, 2002.

[87] N. Hogan and D. Sternad. **Sensitivity of smoothness measures to movement duration, amplitude, and arrests**. *Journal of Motor Behavior*, 41(6):529–534, 2009.

[88] R. Dahlhaus, J. Kurths, P. Maass, and J. Timmer, editors. ***Mathematical Methods in Signal Processing and Digital Image Analysis***. Springer-Verlag Berlin Heidelberg, 2008.

[89] H.R. Schwarz and N. Köckler. ***Numerische Mathematik***. Vieweg + Teubner, 7th edition, 2009.

[90] J. Meyer, K. Baier, J. Wilbert, M. Guckenberger, A. Richter, and M. Flentje. **Three-dimensional spatial modelling of the correlation between abdominal motion and lung tumour motion with breathing**. *Acta Oncologica*, 45(7):923–934, 2006.

[91] P.L Wilson and J. Meyer. **A spring-dashpot system for modelling lung tumour motion in radiotherapy**. *Computational and Mathematical Methods in Medicine*, 11(1):13–26, March 2010.

[92] S. Sayeh, J. Wang, W.T. Main, W. Kilby, and C.R. Maurer. **Respiratory motion tracking for robotic radiosurgery**. In H.C. Urschel, J.J. Kresl, J.D. Luketich, L. Papiez, and R.D Timmerman, editors, *Treating Tumors that Move with Respiration*, pages 15–29. Springer Berlin Heidelberg, 2007.

[93] R. Zamani and A. Afsahi. **Adaptive estimation and prediction of power and performance in high performance computing**. *Computer Science - Research and Development*, 25(3–4):177–186, September 2010.

[94] M.J. Murphy, M. Isaakson, and J. Jalden. **Adaptive filtering to predict lung tumor motion during free breathing**. In *16th International Congress and Exhibition of Computer Assisted Radiology and Surgery*, Paris, France, 2002.

[95] F. Ernst, A. Schlaefer, S. Dieterich, and A. Schweikard. **A fast lane approach to LMS prediction of respiratory motion signals**. *Biomedical Signal Processing and Control*, 3(4):291–299, October 2008.

[96] F. Ernst and A. Schweikard. **A survey of algorithms for respiratory motion prediction in robotic radiosurgery**. In S. Fischer, E. Maehle, and R. Reischuk, editors, *Informatik 2009: Im Focus das Leben, Beiträge der 39. Jahrestagung der Gesellschaft für Informatik e.V.*, volume 154 of *LNI*, pages 1035–1043, Lübeck, Germany, September 28 - October 2, 2009.

[97] B. Widrow. **Adaptive filters I: Fundamentals**. Technical Report SU-SEL-66-126, Stanford Electronics Laboratory, Stanford University, CA, USA, 1966.

[98] Q. Ren, S. Nishioka, H. Shirato, and R.I. Berbeco. **Adaptive prediction of respiratory motion for motion compensation radiotherapy**. *Physics in Medicine and Biology*, 52(22):6651–6661, 2007.

[99] N. Homma, M. Sakai, and Y. Takai. **Time series prediction of respiratory motion for lung tumor tracking radiation therapy**. In *Proceedings of the 10th WSEAS International Conference on Neural Networks*, pages 126–131, 2009.

[100] K.C. McCall and R. Jeraj. **Dual-component model for respiratory motion based on the periodic autoregressive moving average (periodic ARMA) method**. *Physics in Medicine and Biology*, 52(12):3455–3466, 2007.

[101] G. Sharp, S.B. Jiang, S. Shimizu, and H. Shirato. **Prediction of respiratory tumor motion for real-time image-guided radiotherapy**. *Physics in Medicine and Biology*, 49(3):425–440, 2004.

[102] E.A. Wan. **Time series prediction by using a connectionist network with internal delay lines**. In A.S. Weigend and N.A. Gershenfeld, editors, *Time Series Prediction*, volume XVII of *Santa Fe Institute Studies in the Sciences of Complexity*, pages 195–217. Addison-Wesley, 1994.

[103] J.H. Goodband, O.C. Haas, and J.A. Mills. **Neural network approaches to predicting attenuator profiles for radiation therapy**. In M.N. Sahinkaya and K.A. Edge, editors, *Proceedings of the Control 2004 Conference*, University of Bath, U.K., September 2004.

[104] A. Sahih, O.C. Haas, J.H. Goodhand, D. Putra, A. Mills, and J. Burnham. **Respiratory motion prediction for adaptive radiotherapy**. In *DVD-ROM IAR & ACD Conference. German-French Institute for Automation and Robotics*, Nancy, France, 2006.

[105] M. Kakar, H. Nystrom, L.R. Aarup, T.J. Nottrup, and D.R. Olsen. **Respiratory motion prediction by using the adaptive neuro fuzzy inference system (ANFIS)**. *Physics in Medicine and Biology*, 50(19):4721–4728, 2005.

[106] M. Kakar. **Respiratory motion prediction: A fuzzy logic approach**. *Frontiers in Artificial Intelligence and Applications*, 227:108–121, 2011.

[107] F. Ernst and A. Schweikard. **Forecasting respiratory motion with accurate online support vector regression (SVRpred)**. *International Journal of Computer Assisted Radiology and Surgery*, 4(5):439–447, September 2009.

[108] A. Krauss, S. Nill, and U. Oelfke. **The comparative performance of four respiratory motion predictors for real-time tumour tracking**. *Physics in Medicine and Biology*, 56(16):5303–5317, July 2011.

[109] F. Ernst and A. Schweikard. **Predicting respiratory motion signals for image-guided radiotherapy using multi-step linear methods (MULIN)**. *International Journal of Computer Assisted Radiology and Surgery*, 3(1–2):85–90, June 2008.

[110] S.M. Hong, B.H. Jung, and D. Ruan. **Real-time prediction of respiratory motion based on a local dynamic model in an augmented space**. *Physics in Medicine and Biology*, 56(6):1775–1789, February 2011.

[111] D. Putra, P. Skworcow, O.C. Haas, K.J. Burnham, and J.A. Mills. **Output-feedback tracking for tumour motion compensation in adaptive radiotherapy**. In *Proceedings of the American Control Conference (ACC '07)*, pages 3414–3419, New York, USA, July 11-13, 2007.

[112] J. Jalden and M. Isaksson. **Temporal prediction and spatial correlation of breathing motion by adaptive filtering**. Technical report, Stanford University, Stanford, CA, USA, June 2001.

[113] S.S. Vedam, P.J. Keall, A. Docef, D.A. Todor, V.R. Kini, and R. Mohan. **Predicting respiratory motion for four-dimensinal radiotherapy**. *Medical Physics*, 31(8):2274–2283, 2004.

[114] H.M. Wu, G.C. Sharp, B. Salzberg, D. Kaeli, H. Shirato, and S.B. Jiang. **A finite state model for respiratory motion analysis in image guided radiation therapy**. *Physics in Medicine and Biology*, 49(23):5357–5372, 2004.

[115] H. Wu, G.C. Sharp, B. Salzberg, D. Kaeli, H. Shirato, and S.B. Jiang. **SU-DD-A3-06: Model-based probabilistic prediction of tumor respiratory motion**. *Medical Physics*, 32(6):1894, 2005.

[116] A. Kalet, G. Sandison, H. Wu, and R. Schmitz. **A state-based probabilistic model for tumor respiratory motion prediction**. *Physics in Medicine and Biology*, 55(24):7615–7631, November 2010.

[117] C.N. Riviere, A. Thakral, I.I. Iordachita, G. Mitroi, and D. Stoianovici. **Predicting respiratory motion for active canceling during percutaneous needle insertion**. In *Proceedings of the 23rd Annual EMBS International Conference*, Istanbul, Turkey, October 25-28, 2001.

[118] K.C. Veluvolu, U.X. Tan, W.T. Latt, C.Y. Shee, and W.T. Ang. **Adaptive filtering of physiological tremor for real-time compensation**. In *Proceedings of the IEEE International Conference on Robotics and Biomimetics (ROBIO 2008)*, pages 524–529, Bangkok, Thailand, February 22-25, 2008.

[119] D. Ruan. **Kernel density estimation-based real-time prediction for respiratory motion**. *Physics in Medicine and Biology*, 55(5):1311–1326, February 2010.

[120] H. Wu, B. Salzberg, G.C. Sharp, S.B. Jiang, H. Shirato, and D. Kaeli. **Subsequence matching for tumor respiratory motion analysis**. In *Proceedings of the ACM SIGMOD*, pages 682–693, Baltimore, MD, USA, June 2005.

[121] D. Ruan, J.A. Fessler, and J.M. Balter. **Real-time prediction of respiratory motion based on local regression methods**. *Physics in Medicine and Biology*, 52(23):7137–7152, 2007.

[122] P. Skworcow, D. Putra, A. Sahih, J. Goodband, O. C. Haas, K. J. Burnham, and J. A. Mills. **Predictive tracking for respiratory induced motion compensation in adaptive radiotherapy**. In *Proceedings of the UKACC Control Conference 2006*, pages 203–210, Universities of Glasgow and Strathclyde, U.K., August 30 - September 1, 2006.

[123] T. Ortmaier, M. Gröger, D. Boehm, V. Falk, and G. Hirzinger. **Motion estimation in beating heart surgery**. *IEEE Transactions on Biomedical Engineering*, 52(10):1729–1740, October 2005.

[124] R. Heß. **Methoden zur Vorhersage von Atem- und Tumorbewegung beim Menschen**. Master's thesis, Computer Science VII: Robotics and Telematics, University of Würzburg, Germany, August 2009.

[125] S. Haykins. ***Adaptive Filter Theory***. Prentice Hall, 3rd edition, 1996.

[126] C.K. Ing. **Multistep prediction in autoregressive processes**. *Econometric Theory*, 19(2):254–279, April 2003.

[127] B. Widrow and M.E. Hoff. **Adaptive switching circuits**. In *1960 IRE WESCON Convention Record, Part 4*, pages 96–104, New York, August 1960.

[128] J. Nagumo and A. Noda. **A learning method for system identification**. *IEEE Transactions on Automatic Control*, 12(3):282–287, June 1967.

[129] S.C. Douglas. **A family of normalized LMS algorithms**. *IEEE Signal Processing Letters*, 1(3):49–51, March 1994.

[130] A. Weiss and D. Mitra. **Digital adaptive filters: Conditions for convergence, rates of convergence, effects of noise and errors arising from the implementation**. *IEEE Transactions on Information Theory*, 25(6):637–652, November 1979.

[131] T. Hsia. **Convergence analysis of LMS and NLMS adaptive algorithms**. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '83)*, volume 8, pages 667–670, Boston, MA, USA, April 14-16, 1983.

[132] N.E. Hubing and S.T. Alexander. **Statistical analysis of the soft constrained initialization of recursive least squares algorithms**. In *International Conference on Acoustics, Speech, and Signal Processin (ICASSP '90)*, pages 1277–1280, Albuquerque, NM, USA, April 3-6, 1990.

[133] G.V. Moustakides. **Study of the transient phase of the forgetting factor RLS**. *IEEE Transactions on Signal Processing*, 45(10):2468–2476, October 1997.

[134] F. Ernst and A. Schweikard. **Prediction of respiratory motion using a modified recursive least squares algorithm**. In *7. Jahrestagung der Deutschen Gesellschaft für Computer- und Roboterassistierte Chirurgie*, pages 157–160, Leipzig, Germany, September 24-26, 2008.

[135] K.J. Åström and B. Wittenmark. ***Adaptive Control***. Addison-Wesley, 2nd edition, 1995.

[136] R.E. Kalman. **A new approach to linear filtering and prediction problems**. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[137] S. Haykin. **Adaptive tracking of linear time-variant systems by extended RLS algorithm**. *IEEE Transactions on Signal Processing*, 45(5):1118–1128, May 1997.

[138] D. Putra, O.C. Haas, J.A. Mills, and K.J. Bumham. **Prediction of tumour motion using interacting multiple model filter**. In *Proceedings of the 3rd IET International Conference on Advances in Medical, Signal and Information Processing (MEDSIP)*, pages 1–4, Glasgow, U.K., 2006.

[139] D. Putra, O.C. Haas, J.A. Mills, and K.J. Burnham. **A multiple model approach to respiratory motion prediction for real-time IGRT**. *Physics in Medicine and Biology*, 53(6):1651–1663, 2008.

[140] H. Blom and Y. Bar-Shalom. **The interacting multiple model algorithm for systems with markovian switching coefficients**. *IEEE Transactions on Automatic Control*, 33(8):780–783, August 1988.

[141] E. Mazor, A. Averbuch, Y. Bar-Shalom, and J. Dayan. **Interacting multiple model methods in target tracking: A survey**. *IEEE Transactions on Aerospace and Electronic Systems*, 34(1):103–123, January 1998.

[142] A.K. Mandal. ***Introduction to Control Engineering: Modeling, Analysis and Design***. New Age International (P) Ltd. Publishers, December 2008.

[143] R. Tóth, P.S.C. Heuberger, and P.M.J. Van den Hof. **Discretisation of linear parameter-varying state-space representations**. *IET Control Theory and Applications*, 4(10):2082–2096, October 2010.

[144] H. Hanselmann. **Implementation of digital controllers – A survey**. *Automatica*, 23(1):7–32, January 1987.

[145] G. Welch and G. Bishop. **An introduction to the kalman filter**. Technical Report TR 95-041, University of North Carolina at Chapel Hill, 2006.

[146] F. Takens. **Detecting strange attractors in turbulence**. In D. Rand and L. Young, editors, *Dynamical Systems and Turbulence*, volume 898 of *Lecture Notes in Mathematics*, pages 336–381. Springer Berlin / Heidelberg, 1981.

[147] H. Broer and F. Takens. ***Dynamical Systems and Chaos***, volume 172 of *Applied Mathematical Sciences*. Springer-Verlag New York Inc., 2010.

[148] L.C. Uzal, G.L. Grinblat, and P.F. Verdes. **Optimal reconstruction of dynamical systems: A noise amplification approach**. *Physical Review E*, 84(1):016223, July 2011.

[149] J.C. Quintana-Duque. **Non-linear dynamic invariants based on embedding reconstruction of systems for pedaling motion**. In *Proceedings Sportinformatik 2012 of the 9th dvs-Symposium*, University of Konstanz, Germany, September 12-14, 2012.

[150] R. Isermann and M. Münchhof. ***Identification of Dynamic Systems***. Advanced textbooks in control and signal processing. Springer, 2011.

[151] L. Ljung. ***System Identification: Theory for the User***. Prentice Hall, Upper Saddle River, NJ, USA, 2nd edition, 1998.

[152] W.D. D'Souza and T.J. McAvoy. **An analysis of the treatment couch and control system dynamics for respiration-induced motion compensation**. *Medical Physics*, 33(12):4701–4709, 2006.

[153] J. Maciejowski. ***Predictive Control with Constraints***. Prentice Hall, Harlow, England, 2001.

[154] D. Goldfarb and A. Idnani. **A numerically stable dual method for solving strictly convex quadratic programs**. *Mathematical Programming*, 27(1):1–33, 1983.

[155] C. Herrmann, L. Ma, and K. Schilling. **Model predictive control for tumor motion compensation in robot assisted radiotherapy**. In *Proceedings of the 18th IFAC World Congress 2011*, pages 5968–5973, Milano, Italy, September 2011.

[156] O. C. Haas, D. Paluszczyszyn, M. Ruta, and P. Skworcow. **Motion prediction and control for patient motion compensation in radiotherapy**. In *Proceedings of the 18th IFAC World Congress 2011*, pages 5985–5990, Milano, Italy, September 2011.

[157] C. Herrmann, L. Ma, J. Wilbert, K. Baier, and K. Schilling. **Control of a hexapod treatment couch for robot-assisted radiotherapy**. *Biomedical Engineering / Biomedizinische Technik*, 57(5):333–351, October 2012.

[158] J. Wilbert, K. Baier, A. Richter, C. Herrmann, L. Ma, M. Flentje, and M. Guckenberger. **Influence of continuous table motion on patient breathing patterns**. *International Journal of Radiation Oncology Biology Physics*, 77(2):622–629, June 2010.

[159] C. Herrmann and K. Schilling. **Improving patient comfort using model predictive control in robot-assisted radiotherapy**. In *Proceedings of the 2013 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5426–5432, Karlsruhe, Germany, May 6-10, 2013.

[160] J. Wilbert, K. Baier, C. Herrmann, M. Flentje, and M. Guckenberger. **Accuracy of real-time couch tracking during 3-dimensional conformal radiation therapy, intensity modulated radiation therapy, and volumetric modulated arc therapy for prostate cancer**. *International Journal of Radiation Oncology Biology Physics*, 85(1):237–242, January 2013.

[161] R. Albers, U. Bartholomäus, C. Gottschling, P. Klammer, M. Odenwald, J. Reinhard, and S. Sanides. **Die neue Krebs-Medizin**. *FOCUS*, 2013(4):76–83, January 13, 2013.

[162] **Krebs in Deutschland 2007/2008**. 8. Ausgabe, Robert Koch-Institut (Hrsg) und die Gesellschaft der epidemiologischen Krebsregister in Deutschland e.V. (Hrsg), 2012.

[163] Gesellschaft der epidemiologischen Krebsregister in Deutschland e.V. (GEKID). **GEKID Atlas** [last checked 30.01.2013]. URL: `http://www.ekr.med.uni-erlangen.de/GEKID/Atlas/CurrentVersion/Inzidenz/atlas.html`.

[164] M. Malvezzi, P. Bertuccio, F. Levi, C. La Vecchia, and E. Negri. **European cancer mortality predictions for the year 2013**. *Annals of Oncology*, 00:1–9, 2013.

[165] B. Strassmann. **Im Überleben alleingelassen**. *Die Zeit*, 67(5):29–30, January 26, 2012.