COMPLEMENTARY VISION BASED DATA FUSION FOR ROBUST POSITIONING AND DIRECTED FLIGHT OF AN AUTONOMOUS QUADROCOPTER

Nils Gageik[1], Eric Reinthal[2], Paul Benz[3]and Sergio Montenegro[4]

Chair of Computer Science 8, University of Würzburg, Germany

ABSTRACT

The present paper describes an improved 4 DOF (x/y/z/yaw) vision based positioning solution for fully 6 DOF autonomous UAVs, optimised in terms of computation and development costs as well as robustness and performance. The positioning system combines Fourier transform-based image registration (Fourier Tracking) and differential optical flow computation to overcome the drawbacks of a single approach. The first method is capable of recognizing movement in four degree of freedom under variable lighting conditions, but suffers from low sample rate and high computational costs. Differential optical flow computation, on the other hand, enables a very high sample rate to gain control robustness. This method, however, is limited to translational movement only and performs poor in bad lighting conditions. A reliable positioning system for autonomous flights with free heading is obtained by fusing both techniques. Although the vision system can measure the variable altitude during flight, infrared and ultrasonic sensors are used for robustness. This work is part of the AQopterI8 project, which aims to develop an autonomous flying quadrocopter for indoor application and makes autonomous directed flight possible.

KEYWORDS

Autonomous UAV, Quadrocopter, Quadrotor, Vision Based, Positioning, Data Fusion, Directed Flight

1. INTRODUCTION

In spite of the fact that nowadays exist solutions for vision based positioning to enable autonomous flight of UAV's (Unmanned Aerial Vehicles), these solutions suffer from different inherent drawbacks. Main drawbacks are high computational burden [1, 2] and low sample rate [3],limitations to translational movement [11, 12], and bad robustness to varying lighting conditions[4]. Despite the considerable rise of computing power during the last decade, computational burden is still a concern for autonomous UAVs. High computing capacity results in a significant increase in weight as well as power consumption of a UAV. Flight-time is thereby reduced by a noticeable factor. This lead to several scientific works on UAVs, where computation for the positioning system is done on external hardware. These approaches, however, break the requirements for a fully autonomous system. Our proposed positioning system focuses on computational efficiency with acceptable hardware load to deploy relatively lightweight on-board-hardware with low power consumption. The system therefore provides fully autonomous positioning without the need of external systems. Hence fully autonomous directed flight becomes possible, that is required for passing through narrow openings such as doors and windows with an on-Board collision avoidance system having preferential direction. This is an alternative to laser based solutions [16-17].

DOI : 10.5121/ijaia.2014.5501

The following study deals with the concept, implementation and evaluation of merging two vision based methods for positioning of an autonomous UAV. First two sections describe the two methods used here, explaining each on its own. The fourth section describes the overall concept of merging the two systems with focus on the data fusion and is followed by a section about its implementation. The paper ends with sections on evaluation and discussion.

2. FOURIER TRACKING (METHOD 1)

2.1. Overview

Fourier-transform based image registration (short: Fourier Tracking)is the process of determining the geometric correspondence of two images. With a camera mounted on the UAV and pointing vertically to the ground, the motion of the UAV can be computed by continuous registration of succeeding images. With the constraint that the focus of the camera stays in the plane of surface, succeeding images are affine transformations with respect to translation, rotation and scaling. By registering the images, the parameters of the transformation are gained, which permit to conclude the movement of the UAV. Image registration is done in Fourier space to reduce computational costs and to improve robustness against changing light conditions.

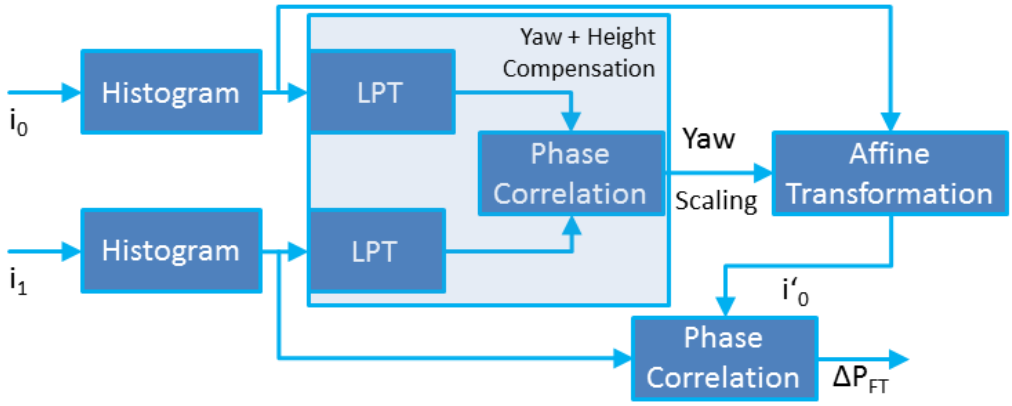

Figure 1.Fourier-transform based image registration

Figure 1 pictures the simplified concept of the implemented Fourier Tracking. To gain robustness under variable lighting conditions, a histogram equalization is applied to each image before processing. This gives the positioning system the possibility to function under bad lighting conditions as well as with surfaces having very low contrast.

Next Log-Polar-Transformation (LPT) and Phase Correlation are executed to gain the yaw angle and scaling factor. The scaling factor, which corresponds to the height or change in height, together with the yaw angle is required for the affine transformation, which transforms the first image i_0 to i_0', so that it fits to the second image i_1. Now both images correspond in height and yaw angle to another and the second phase correlation can determine the translation between both images. This translation corresponds to the position change of the quadrocopter.

Each LPT block consists of a FT (Fourier Transform), a Log-Polar-Transformation(section 2.3). and a high pass filter. The coordinate-change to logarithmic polar coordinates induces distortion

in the transformed image since the transform is non-uniform. To overcome bad performing of the phase correlation, a high-pass filter is applied after the transform. [6]

Each Phase correlation (section 2.2) requires three Fourier transforms (i_0, i_1, inverse).

2.2. Fourier Transform and Cross-Power-Spectrum (CPS)

The concept uses extended phase correlation [5] for registering images which are translated, rotated and scaled with respect to each other.

Let $i[m,n]$ be an image in spatial domain and $I[s_1,s_2]$ the corresponding image in Fourier domain thus

$$\mathcal{F}[i] = I \tag{2.1}$$

Two images $i_1$and i_2, which are translated by $\langle m_0, n_0 \rangle$, own the following dependence (neglecting border-effects):

$$i_2[m,n] = i_1[m+m_0, n+n_0] \tag{2.2}$$

Due to the shift-theorem of the Fourier transform, this dependence can be written in Fourier space as:

$$I_2[s_1,s_2] = e^{-i2\pi(s_1 m_0 + s_2 n_0)} I_1[s_1,s_2] \tag{2.3}$$

The phase correlation allows to compute this translation by using the cross-power-spectrum (CPS) of the two images. The CPS is defined as:

$$CPS = \frac{I_1[s_1,s_2] I_2^*[s_1,s_2]}{|I_1[s_1,s_2] I_2[s_1,s_2]|} \tag{2.4}$$

where$I_2^*[s_1,s_2]$ denotes the complex conjugate of $I_2[s_1,s_2]$. By inserting formula (2.3) in (2.4), one can show that the CPS corresponds to the translational difference of the two images:

$$CPS = e^{-i2\pi(s_1 m_0 + s_2 n_0)} \tag{2.5}$$

The CPS is technically the convolution of the two images in Fourier space. By taking the inverse Fourier transform of the CPS, the translation $\langle m_0, n_0 \rangle$ is obtained, because the exponential term of equation (2.3) corresponds to the delta-function.

$$\mathcal{F}^{-1}\{CPS\} = \mathcal{F}^{-1}\left\{e^{-i2\pi(s_1 m_0 + s_2 n_0)}\right\} = \delta[m-m_0, n-n_0] \tag{2.6}$$

For two overlapping images, equation (2.2) is only satisfied in the overlapping area. The non-corresponding areas add distortion to the CPS, which therefore differs from an exact delta-impulse. The real CPS describes a correlation surface of the images, with its peak at the delta-impulse from equation (2.6). This peak gives the translation between$i_1$and i_2:

$$\langle m_0 | n_0 \rangle = \max_{m,n} \mathcal{F}^{-1}\{CPS\} \tag{2.7}$$

With this concept only horizontal translated images can be registered. A preceding rotation and scaling compensation is therefore required.

2.3. Polar-Coordinates

By applying a coordinate-change to logarithmic polar-coordinates$i[\log r, \theta]$, rotation and scaling can be computed using phase correlation as well. This concept is derived from Reddy and Chatterji[6].

Let $i_2[u, v]$ be rotated by $\theta_0$and translated to $i_1[m, n]$. Their variables are related by:

$$\begin{aligned} u &= m\cos\theta_0 + n\sin\theta_0 - m_0 \\ v &= -m\sin\theta_0 + n\cos\theta_0 - n_0 \end{aligned} \tag{2.8}$$

Applying the Fourier transform, we get:

$$I_1[s_m, s_n] = e^{-i2\pi(s_m m_0 + s_n n_0)} I_2[s_u, s_v] \tag{2.9}$$

Due to the rotation property of the Fourier transform, the variable relation is given by the following equation:

$$\begin{aligned} s_u &= s_m\cos\theta_0 + s_n\sin\theta_0 \\ s_v &= -s_m\sin\theta_0 + s_n\cos\theta_0 \end{aligned} \tag{2.10}$$

The magnitudes of the Fourier transforms differ only by rotation with a rotation centre in the middle of the images. This corresponds to a translation in polar coordinates, since rotation around the image centre only affects the θ-coordinate. Applying a coordinate-change to polar coordinates, the magnitudes of the Fourier transforms are related by:

$$|I_1[s_r, s_\theta]| = |I_2[s_r, s_\theta - \theta_0]| \tag{2.11}$$

If i_2 is also scaled to $i_1$by α, which only affects the r-coordinate in polar coordinates, equation (2.11) extends to:

$$|I_1[s_r, s_\theta]| = \frac{1}{\alpha}|I_2[\alpha s_r, s_\theta - \theta_0]| \tag{2.12}$$

Assuming a minimal scale change between two images, the factor α^{-1} may be neglected. Switching to logarithmic polar coordinates gives:

$$|I_1[log\, s_r, s_\theta]| = \frac{1}{\alpha}|I_2[log\, s_r - log\, \alpha, s_\theta - \theta_0]| \tag{2.13}$$

By substituting $x = \log s_r$and $c = \log\alpha$, we obtain the following formula:

$$|I_1[x, s_\theta]| = |I_2[x - c, s_\theta - \theta_0]| \tag{2.14}$$

These translations c and θ_0 can be obtained by applying phase correlation, where $\alpha = e^c$ corresponds to the scaling and θ_0 to the rotation of the two input images. Once rotation and scale change is computed, the second image is scaled and rotated by the computed values. Thus, the transformed image and the first input image differ merely in translation. This translation is obtained by applying another phase correlation.

2.3. Implementation details

Once two images are registered, their translation, scaling and rotation are transformed to the coordinate system of the UAV, which leads to the change in position. Let i_0 be taken at time t_0 and i_1 slightly thereafter at time t_1. The position of the UAV in the surface plane at time $t_0$equals $\langle x_0, y_0 \rangle$, the altitude is h_0 and the heading is given by φ_0.

The rotation of the images corresponds directly to the change in heading of the UAV, leading to

$$\varphi(t_1) = \varphi_0 + \theta \tag{2.15}$$

The scaling of the images represents the relative change in altitude.

$$h(t_1) = \frac{h_0}{\alpha} \tag{2.16}$$

The translation of the image registration is obtained in pixels. Therefore, a calibration factor α has to be determined to translate the translation to meters. This factor α is constant for a given camera and resolution. The ratio of pixel to meter is also affected by the altitude of the UAV. The field of view of the camera changes linearly with the distance to the camera scene. Thus the translation in meters is given by:

$$\langle x_1, y_1 \rangle = a h_0 \langle m_0, n_0 \rangle \tag{2.17}$$

This translation is aligned in camera-coordinates and has to be rotated in UAV-coordinates with the angle of heading at time t_0:

$$\langle x(t_1), y(t_1) \rangle = \begin{bmatrix} \cos\varphi_0 & \sin\varphi_0 \\ -\sin\varphi_0 & \cos\varphi_0 \end{bmatrix} \langle x_1, y_1 \rangle \tag{2.18}$$

By continuous registration of images, the movement in the surface plane, the altitude and the heading of the UAV can be iteratively measured. Each registration with the given concept (Fig. 1)requires eight Fourier transforms, representing the most computational load of the positioning system. To reduce computational costs, Fourier transform is implemented using Fast Fourier Transform. For a square image of $n \times n$ pixels, the transform can be computed in $\mathcal{O}(n \log n)$. [7]

In order to reduce computation time further, the symmetry of the Fourier transform is exploited. Since Fourier transform is designed for complex input, real input sequences such as images lead to complex-conjugated symmetry in Fourier space. Therefore it is sufficient to compute merely the upper half of the CPS. To gain further precision, the peak location of the CPS is computed with a weighted average around the absolute peak. This gives sub-pixel-accuracy for the translation, which also improves the robustness due to higher accuracy in the rotation-and-scale transform.

Since images are finite, Fourier transform induces border effects leading to miscalculations in the CPS. Applying a Hanning-window before each transform minimizes these effects by blackening out edge regions of the images. This increases the reliability of the phase correlation significantly. [8]

3. DIFFERENTIAL OPTICAL FLOW COMPUTATION (METHOD 2)

3.1. Overview

A common method for differential optical flow computation is the Lucas-Kanade-Method [9]. It can be realized with the algorithm of Srinivasan [10]. For simplification it is presumed, that the change in the picture is only a translation and at most one pixel per frame. Under these constraints, the searched optical flow values u and v in the x- and y-axis between two pictures can then be determined by equation 3:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum_x \sum_y (P_t - P_{t-1}) * (P_2 - P_1) \\ \sum_x \sum_y (P_t - P_{t-1}) * (P_4 - P_3) \end{bmatrix} * \begin{bmatrix} \sum_x \sum_y (P_2 - P_1)^2 & \sum_x \sum_y (P_2 - P_1) * (P_4 - P_3) \\ \sum_x \sum_y (P_2 - P_1) * (P_4 - P_3) & \sum_x \sum_y (P_4 - P_3)^2 \end{bmatrix}^{-1} \quad (3.1)$$

with $P_x(i,j)$, $P_y(i,j)$ and $P_t(i,j)$ being the partial intensity derivatives of point P(i, j) in the x- or y-axis or after time t, respectively:

$$\begin{aligned} P_x(i,j) &= P_2 - P_1 \\ P_y(i,j) &= P_4 - P_3 \\ P_t(i,j) &= P_t - P_{t-1} \end{aligned} \quad (3.2)$$

The partial intensity derivatives are computed from the neighboring pixels to the left (P_2), right (P_1), up (P_4) and down (P_3) and the previous picture (Fig. 2).

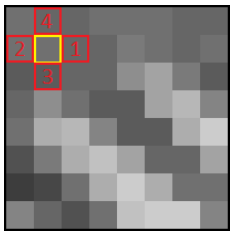

Figure 2.Centre Pixel P_t or P_{t-1} in yellow and neighbours in red [11]

3.3. Implementation details

In this work the ADNS 3080, an optical flow or optical mouse sensor, is used together with a proper lens [12]. This sensor works with an internal sample rate of up to 6400fps and provides good results under good lighting conditions [4], but it cannot handle rotations. Furthermore the sensor fails, if there is too high (as in case of outdoors) or too less light. To overcome these two drawbacks, the Fourier Tracking has been added to the system.

Since there is no information in the data sheet, which method is implemented on this sensor, this is not absolutely clear. But because of its characteristics and drawbacks, the high frame rate, the mentioned problems and limitation to translations it is considered to be either method 2 or a method with similar characteristics.

4. CONCEPT

4.1. Overview

The main motivation for this work was the limitation of method 2 (optical flow) to translations. Any yaw rotation is interpreted as a translation and therefore cannot be executed without accumulating high position error. After any yaw rotation with position hold, the system would hold position on a different location. Therefore method 1 (Fourier Tracking) was designed in a way which can handle yaw rotations, but suffers from a high computational burden.

Since computation power on-Board the quadrocopter is very much limited and is required also for other applications like mapping, object and obstacle detection, as well as method 2 is sufficient in most cases, method 1should only be activated for error correction after rotating. This concept idea is called rotation compensation (4.2).

Besides this, method 2 also fails in bad lighting conditions. In this case method 1 is also activated, because it can handle bad lighting conditions. The data of both methods are then merged, which is called dynamic complementary data fusion (4.3). It is called dynamic, because method 1 is automatically activated, if method 2 produces bad results.

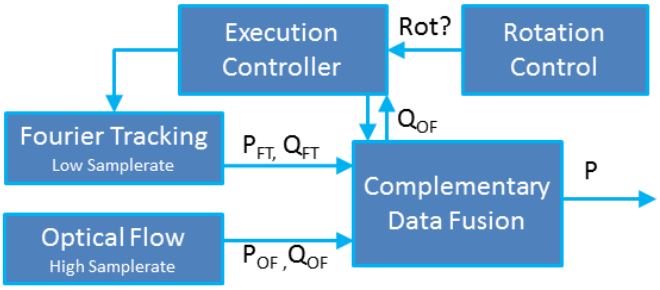

Figure 3. Concept

The overall concept of the system is illustrated in Figure 3. Both sensor systems send its data, the last measured position P and the quality Q of the last measurement to the complementary data fusion, where these data are merged. Depending on the quality data Q_{OF} of the optical flow system or the rotation state, the execution controller then activates or deactivates the Fourier Tracking and the dynamic complementary data fusion. The rotation state can be either rotating or not rotating.

The Fourier Tracking uses a space-fixed frame, while the optical flow and the quadrocopter with its position control can only operate in a body-fixed frame. That's why the results have to be transformed from one frame to another. Furthermore both methods are performed on different processors, because the Fourier Tracking needs high computation power and the ADNS uses SPI and therefore is best connected to the microcontroller, which also performs the control loop of the system and drives the motors. This means, the results of the Fourier Tracking also have to be sent to the microcontroller. To release the microcontroller, the data fusion is performed on the computer with high computation power.

4.2. Rotation Compensation

The idea of the rotation compensation is to start the Fourier Tracking whenever the quadrocopter is going to perform a yaw rotation and stop the Fourier Tracking after the rotation is finished. Then the position error, which occurs while rotating, is corrected. In this case the Data Fusion simply discards the erroneous optical flow measurements and uses only the measurements from the Fourier Tracking.

The advantage of this method is, that it is easy to implement and enables the quadrocopter to perform yaw rotations and correct errors after rotating. It can be improved by updating the position during the rotation, but then the transformation as well as the correction has to be executed more often. This means more communication and work load for the microcontroller, but it could also lead to instability of the control, because of actual value jumps through position corrections.

4.3. Dynamic Complementary Data Fusion

The idea of the dynamic complementary data fusion is to activate and use the Fourier Tracking only when required and to complementary incorporate both measurements depending on each quality (formula 4.1). To ensure erroneous measurements from the optical flow sensor are not used during rotation, α is set to zero in this case.

$$P = P + \alpha \cdot \Delta P_{OF} + (1 - \alpha) \cdot \Delta P_{FT} \tag{4.1}$$

Three stats can be divided depending on the quality of the optical flow Q_{OF}. In the first state the quality Q_{OF} is so low, that only the Fourier Tracking is used, so α is set zero. In the second state the quality Q_{OF} is so high, that the Fourier tracking is deactivated so $\alpha$is set to one. In the third state α is computed by the relationship of the previously normalized and scaled qualities (formula 4.2).

$$\alpha = \frac{Q_{OF}}{Q_{OF} + Q_{FT}} \tag{4.2}$$

4.4. Directed Flight

Directed Flight means, that the nose of the quadrocopter is always directed into flight direction, as it is known from airplanes. This becomes necessary, if the quadrocopter is no longer symmetrical, but has a preferential direction, because of a fix-mounted PMD camera or stereo vision system for collision avoidance. With this configuration, to fly through a narrow opening like a window or a door; the flight direction, the preferential direction and the yaw set value of the quadrocopter have to be the same.

To realize this, the quadrocopter can simply be rotated over the yaw axis. For directed flight from space-fixed frame Position $P_1 = (x_1,y_1)$ to $P_2 = (x_2,y_2)$ the yaw set value γ can be computed using formula 4.3:

$$\gamma = \arccos\left(\frac{x_2 - x_1}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}\right) \cdot sign\,(y_2 - y_1) \tag{4.3}$$

5. IMPLEMENTATION

5.1. Hardware Design

The overall Hardware Design of the full system is shown in figure 4. The red dashed line separates the components connected to the AVR (upper) from those connected to the LP-180 Pico-ITX board. The minimum components required for this study have a red bordering line.

The system uses the IMU3000 for orientation computation, while the MinIMU-9 v2 is for backup. Two infrared sensors, one ultrasonic sensor, a pressure sensor and the IMU are fused for height over ground computation[13]. The 6 DOF (degree of freedom) control of the autonomous quadrocopter is executed on the AVR32 UC3A1512 microcontroller with 10ms sample time[4].

Method 1 is implemented on the LP-180 and the C920 webcam from Logitech is used as Fourier Tracking sensor. The C270 webcam has also been tested for this application, but showed very disappointing performance under dynamic movements.

The second position sensor is the ADNS 3080.The dynamic complementary data fusion for position computation of both methods and the navigation is executed on the LP-180. The fused position, if Fourier Tracking is active, and the position set value are sent to the UC3A1515 via USART/RS232. This applies to the set value for yaw also.

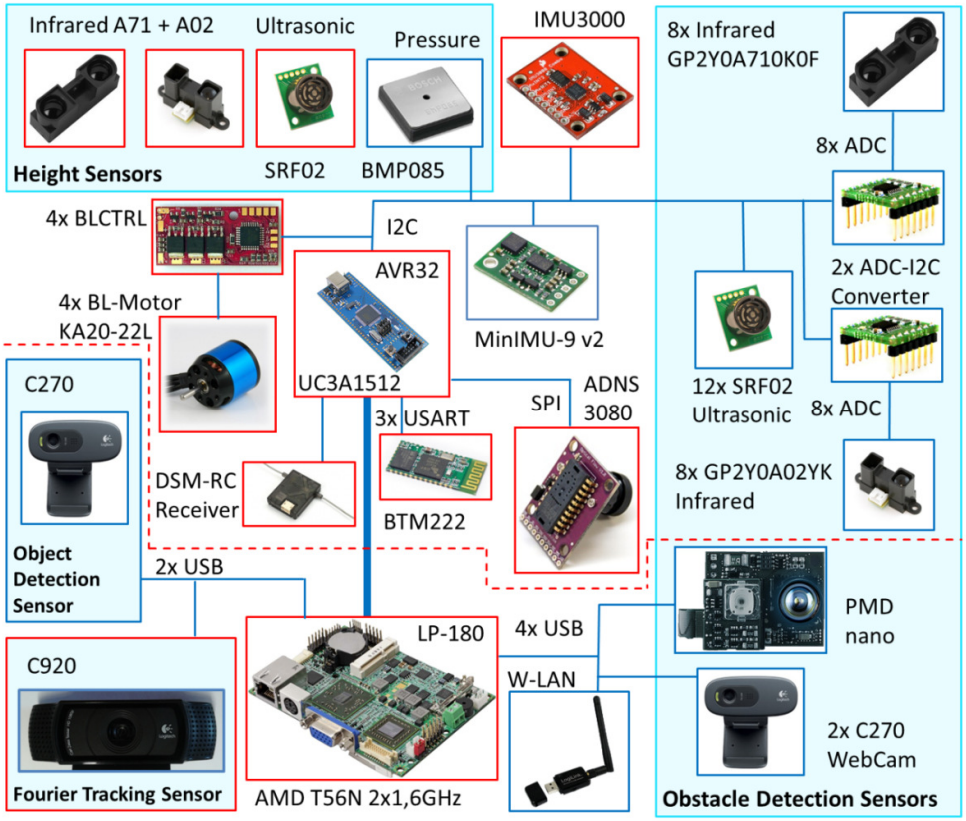

Figure 4. Hardware Design

In this work the object detection and the obstacle detection is not used, thus these sensors (except the C270 webcams) are not connected to the evaluated quadrocopter for this paper. The other sensors are attached to the system, though they are not used here.

5.1. Software Design

Figure 5 shows the simplified software design of the software on the LP-180 for directed flight. The position P_{OF} measured by the optical flow sensor, its quality Q_{OF} and the current orientation q in quaternion are sent from the AVR to the LP-180 every 10ms. This information is processed by the Pose Receiver method, which then execute the Execution Controller.

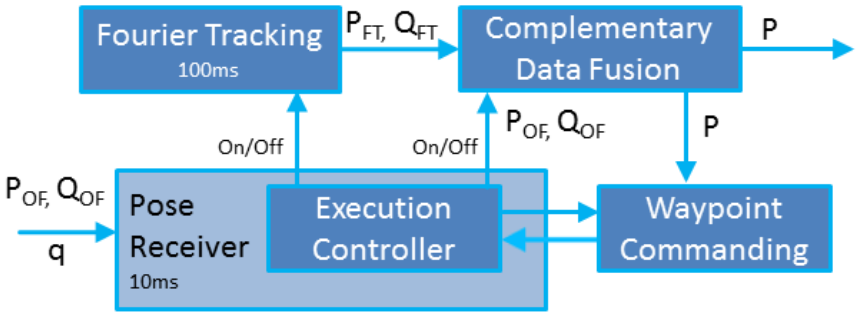

Figure 5. Software Design

The Execution Controller activates or deactivates the Fourier Tracking depending on the quality Q_{OF}. It also updates the Complementary Data Fusion by incorporating the new position P_{OF} every 10ms. Subsequently the Waypoint Commanding function is executed, which realizes the directed navigation. While doing this, the quadrocopter always first yaws to face the next waypoint and then approaches it. Hence, it rotates, translates, rotates, and translates, and so on, until the last waypoint has been reached.

The Waypoint Commanding function consists of four states and a Boolean flag rotation, indicating whether a translation or a rotation has to be performed next:

- *Navigation Off:*

  This is the initial state and the directed navigation is not possible. The rotation flag is set to true. User commands are necessary to set up the waypoint list and to switch to the *Navigation On* state.

- *Navigation On:*

  In this state navigation is generally possible. If the last waypoint has not been reached, the next waypoint or rotation is processed. This means the next state of Waypoint Commanding is *Waypoint Control* or *Rotation Control*, depending on the rotation flag. The according set values are also sent to the AVR, which changes the set value of the according controller. In case of a rotation a signal is also send to the Execution Controller to activate the Fourier Tracking.

- *Waypoint Control:*
  The waypoint control state checks, whether the current waypoint is reached or not. If yes, it switches back to the *Navigation On* state. After every 3 seconds, when a waypoint is not reached, the set values are sent again to the AVR. This is important in case, that a command via the USART/RS232 communication link gets lost. Then the LP-180 would wait (forever) for the AVR to reach a certain waypoint and the AVR will not be able to react properly. It would control to a different position and the navigation would stick. This procedure is also much more simple and robust than using acknowledgements. For safety all commands are sent twice and together with a checksum, so that invalid commands can be discarded.

- *Rotation Control:*

  By analogy to the *Waypoint Control* the *Rotation Control* state checks, if the rotation is finished. If so, it sends a signal to the Execution Controller to deactivate the Fourier Tracking. Then the position is corrected using the Fourier Tracking and it is updated on the AVR.

The Fourier Tracking is the implementation of method 1 (chapter 2). The Complementary Data Fusion is the implementation of the already mentioned concept in chapter 4.3. It is executed after every Fourier Tracking sample, but not during rotation. Position updates are sent to the AVR every 100ms or 500ms, depending on the quality Q_{OF}. This is so, because too many position updates disturb the controller because of jumps, feedback and delay issues. The software on the LP-180 uses Qt [14], Open CV and FFT libraries and is implemented in C++. The software on the AVR is implemented using AVR32 Studio and C.

6. EVALUATION

The system has been extensively evaluated. In three different scenarios the system has been tracked with the Optical Tracking System OptiTrack from Natural Point (referenced as OTS) with five Flex 3 cameras [15] to get a reference position, while different autonomous flight scenarios have been executed. For position estimation and control exclusively the mentioned on-Board optical sensors with the described data fusion has been used (referenced as EST). In total 70 experiments (trials) has been documented and the most representative results were selected to picture the behaviour here

Table 1. Final Position Errors for P_1 and P_2

Position Error	P_1			P_2		
	E_x [cm]	E_y [cm]	ΔF_p [cm]	E_x [cm]	E_y [cm]	ΔF_p [cm]
Trial 1	16.6	5.9	17.6	-15.4	-6.0	16.5
Trial 2	-13.1	-20.0	23.9	-6.1	-3.8	7.2
Trial 3	-22.0	-15.5	26.9	-26.3	-0.7	26.3
Trial 4	-8.4	-17.1	19.1	-10.8	-13.2	17.1
Trial 5	-11.6	-17.4	20.9	-11.9	-4.2	12.6
Mean	-7.7	-12.82	21.68	-14.1	-5.58	15.94

5.1. Single Rotation Compensation (Headed Flight)

In this setup the quadrocopter is tracked while performing a directed flight containing of a single waypoint. It flies from the initial position P_0 = (0m, 0m) to P_1 = (0m, 1.5m) or P_2 = (1.5m, 1.5m). Each experiment was repeated five times with the same settings and all results showed very similar behaviour. The position was measured before the rotation and after 20s (P_1) or 10s (P_2), when the rotation and translation were already finished. The position error is the difference between the position change of OTS and EST and can be seen in Table 1. E_x and E_y are the errors in the x-axis and y-axis, respectively. ΔF_p is the total 2D position error. From this data it can be concluded that the error in the position system after the mentioned manoeuvre is in the range of 12-27cm with a mean of about 19cm. This is already quite high after such a short period of time, but many effects such as wrong scaling and misalignment of OTS and EST increase this error. Therefore more data has to be taken into account to come to a conclusion.

As all experiments showed similar results, two experiments from both scenarios are illustrated in Figure 6 and Figure 7 to discuss this setup more in detail. The graphene show, that after about 2s the rotation is finished and after about 5s the set point is reached. The green line shows, that the Fourier Tracking is activated before the rotation and is deactivated after the rotation is finished. Then the translation is executed and the system reaches the set position with a first overshoot of about 30 - 50cm. This overshoot is caused by the PID controller and its parameters, but also by the fact, that the EST measures a smaller position then the OTS. Both graphene taken into account together, it can be derived, that the overshoot is also affected by the happenings on the other axis.

The system has a 2° x-axis mechanical misalignment towards the ground level (over pitch). That is the reason, why the system does not lift straight, but flies forward. This explains a 25cm position error over the x-axis after starting (Fig. 7).

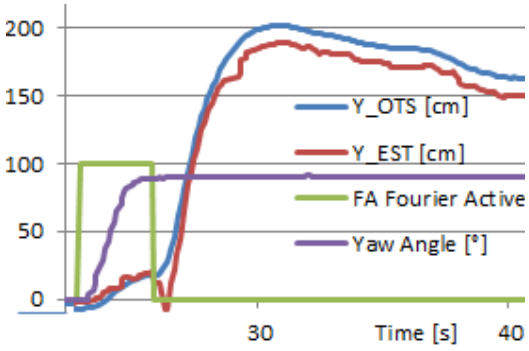

Figure 6.P_1Trial 5
Fourier Tracking Off: FA = 0
Fourier Tracking On: FA = 100

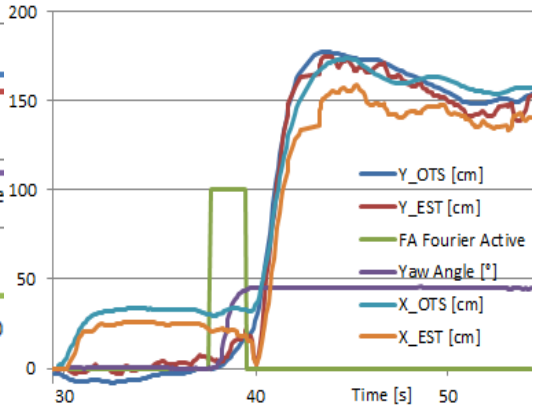

Figure 7.P_2Trial 4

5.2. Multiple Rotation Compensation (Headed Flight)

In this setup the quadrocopter is tracked while performing a directed flight containing of a set of waypoints. Compared to the first setup the quadrocopter now performs multiple iterative rotations and directed translations. As waypoint set a square (Fig. 8)comprising of 4 waypoints and a Nikolaus house (Fig. 10) comprising of 8 waypoints has been used.

Again the position error and the time have been documented (Table 2). The position error is in the range of 12-34 cm, with one exception of 55cm. The mean is about 28cm for square and about 31cm for Nikolaus house. This means with 4 or 8 times more waypoints resulting in as many additional translations and rotations, the error increases, but not proportional to the time or amount of waypoints.

Table 2. Final Position Errors for Square and Nikolaus House Flight

Position Error	Square				Nikolaus House			
	E_x [cm]	E_y [cm]	t [s]	ΔF_p [cm]	E_x [cm]	E_y [cm]	t [s]	ΔF_p [cm]
Trial 1	11,5	-21,8	25	24,7	-31.5	-11.2	75	33.4
Trial 2	11,6	-22,6	25	25,4	21.1	10.5	55	23.5
Trial 3	-13,6	-18,2	20	22,8	28.3	-11.5	140	30.6
Trial 4	8,7	-32,9	35	34,0	-7.4	10.5	70	12.9
Trial 5	18,8	-25,4	25	31,6	-53.3	-13.,4	50	55.0
Mean	7,4	-24,8	26	27,7	-8,6	-3,0	78	31,0

The big error of 55cm in trial 5 of the Nikolaus house made a closer analyzation necessary. The data indicate that the error occurred during a fast movement. The reason for this and a solution need to be found in a further investigation.

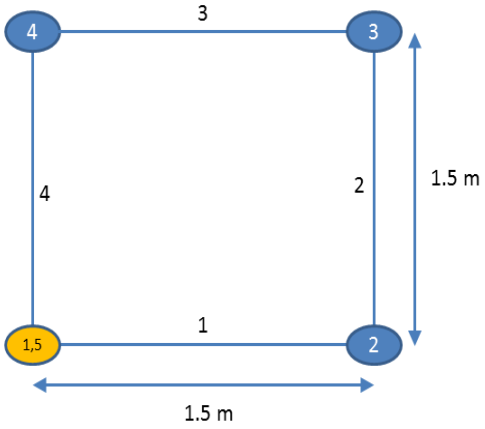

Figure 8. Square Waypoints

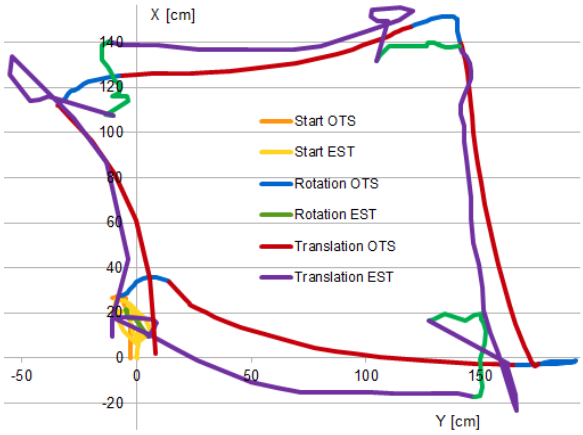

Figure 9. Directed Square Flight: Position (Trial 2) tracked with on-Board estimation (EST) and OTS during different phases start, rotation and translation

Again two trials are illustrated to explain the behaviour of the system in more details. Figure 9 shows the tracked position (EST, OTS) of the quadrotor during the square flight. The tracks are coloured in three different colours for the start, rotation and translation phase. It can clearly be seen, that during rotation a position error occurs, which is corrected afterwards, so that the system approaches correctly to the next waypoint.

Figure 11 shows the tracked position (OTS) during the flight of a Nikolaus house. Again the greatest variations from the set values occur during orientation. Figure 12 shows the corresponding yaw angle. The system adopts the closer yaw angle fastly(within a few seconds). The rotation speed is limited because of the fact, that the Fourier Tracking sensor and system cannot handle faster rotations. During translation the yaw angle is constant and therefore the Fourier Tracking can be deactivated.

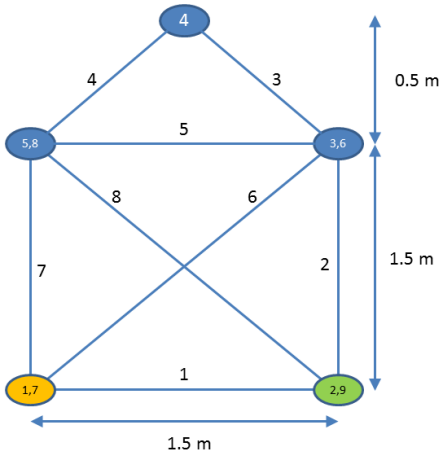

Figure 10. Nikolaus House Waypoints

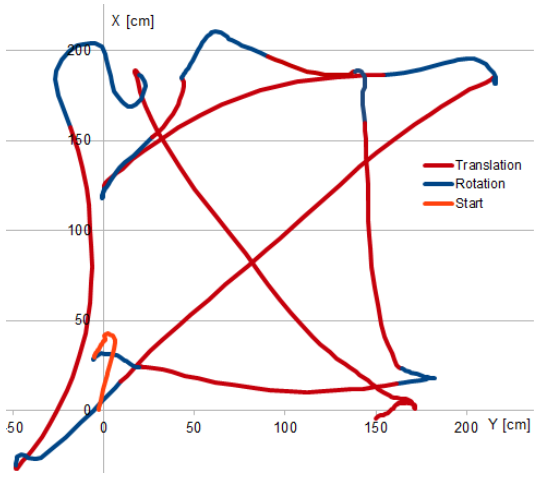

Figure 11. Directed Nikolaus House Flight: Position (Trial 2) tracked with OTS during different phases start, rotation and translation

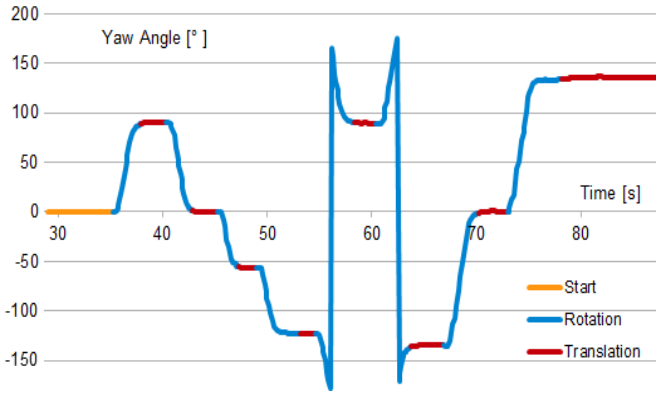

Figure 12. Directed Nikolaus House: Corresponding Yaw Angle (Trial 2)

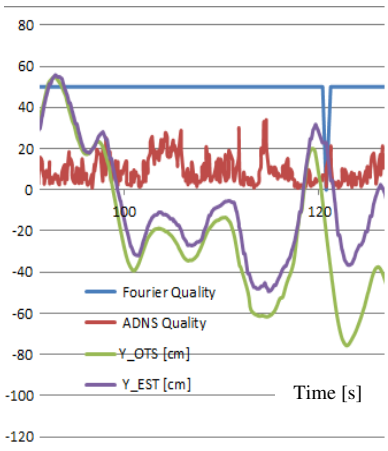

Figure 13. Effect of a Sensor Failure on Position Error under critical lighting conditions

5.1. Dynamic Weighted Data Fusion

In the last setup the system has been evaluated under varying lighting conditions to investigate and demonstrate this effect and the systems behaviour on changes. Figure 14 illustrates the four different investigated lighting conditions. Good conditions are those, which are normal operating conditions for the ADNS-3080with relative much but not too much light. This is the light you

normally have in a bright room. Bad conditions are those, which make the ADNS fail sometimes, though there is still much light in the room. In our case we switched off the direct top light and there was only indirect light left. For the experiments with Very Bad Light conditions, we switched off most lights, but kept on some lights, so in the test section there was also a brightness bridge from dark to bright. The other time, for Total Bad conditions, all lights were switched off and only some daylight came through the curtains.

Figure 13 shows the effect of a sensor failure because of bad lighting conditions. After about 120s even the Fourier Tracking failed resulting in a great position error. The relationship between position error and sensor failure because of low quality can be clearly seen here, though it is also shown, that the system does not totally fail, but follows the further movement correctly with about constant error.

During all experiments of this setup the quadrotor is on position hold and again 5 trials have been documented for each lighting conditions. During the experiment, it could be seen, that the quadrotor became more dynamic under worse lighting conditions. That is why beside the position error this time also the standard deviation is computed after 60s.

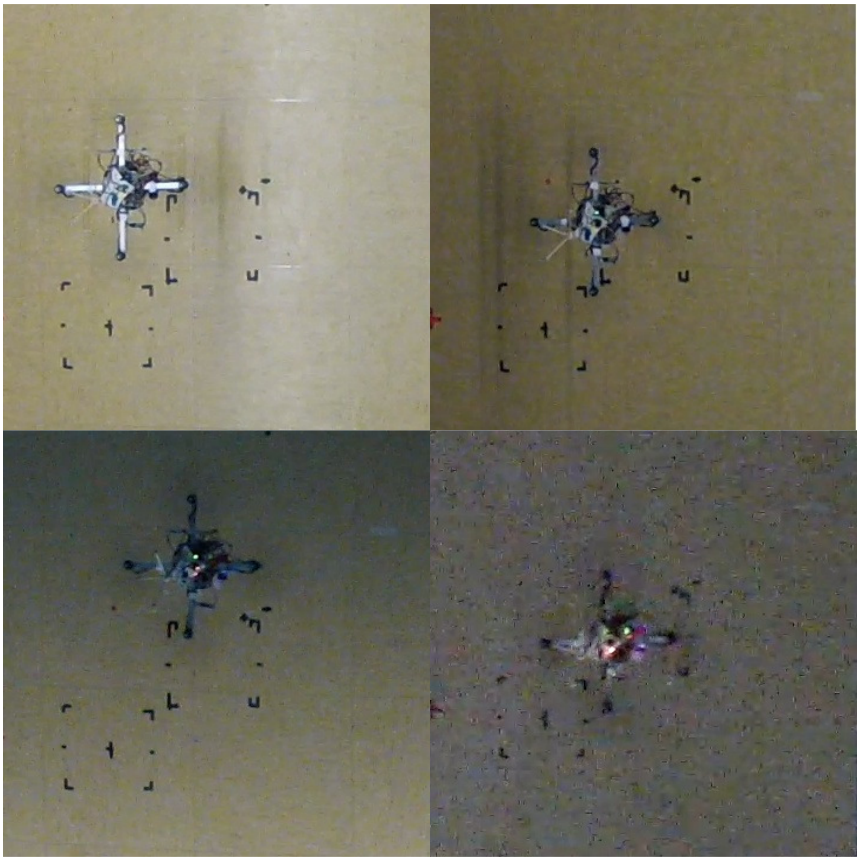

Figure 14. Different Lighting Conditions Top: Good (left), Bad (right)
Bottom: Very Bad (left), Total Bad (right)

Figure 15 summarizes the results of this setup (illustration of Table 3 &4).Figure 15 and Table 3 proves that the position error is significantly higher under worse lighting conditions compared to good conditions. The position error under worse conditions is about 4 times higher. Rather unexpected is the result, that the position error under bad conditions is higher than the error under

very bad or total bad conditions. Under bad conditions the Fourier Tracking is often activated and deactivated and both positions are fused following formula 4.1 and 4.2. It can be concluded, that a problem with the data fusion still exists and the parameters need to be tuned further. It might be, that wrong ADNS sensor values are used or that an error is incorporated into the system because of a wrong synchronisation of both systems, as both sensors are running on different processors with a different sample time and delay before their data fusion. This problem becomes more sophisticated, as the Fourier Tracking system dynamically activates and deactivates. These effects were taken into account, but further tuning of the data fusion under changing light conditions is required or the presented solution is not fully suitable. Despite this error the system performed better under bad lighting conditions compared to the worse lighting conditions (very bad & total bad), because of the following two facts.

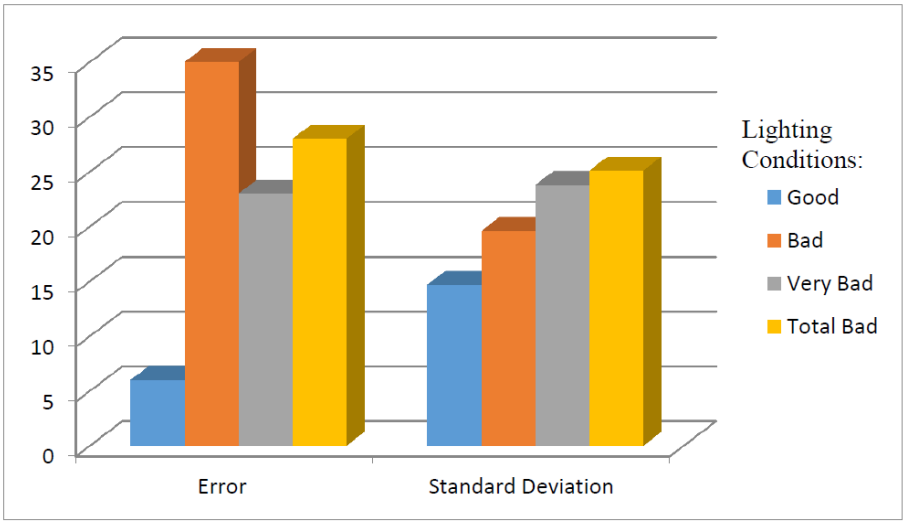

Figure 15.Effect of Different Lighting Conditions on Position Error and Standard Deviation

The entries of table 3 containing an 'X' represent experiments, which could not finished properly, because the position system failed in a way, that the quadrotor left the test section. This happened one time under very bad and total bad conditions. In these cases the Fourier Tracking also failed (compare Fig.13). The experiment was then repeated and it worked in all other cases.

Table3.Position Errors under different (difficult) lighting conditions

$$P = \sqrt{X^2 + Y^2}$$

Errors [cm]	Trial 1			Trial 2			Trial 3			Trial 4			Trial 5			$\bar{P}$
	X	Y	P	X	Y	P	X	Y	P	X	Y	P	X	Y	P	
Good	-1	-13	13	-3	-3	4	1	-2	2	-5	0	5	2	5	6	6
Bad	-47	-44	64	-31	-6	31	-48	-32	58	-28	-32	43	-2	-15	15	35
Very Bad	X -8	X -17	X 18	- 11	-8	14	-4	-67	67	-12	-21	24	-6	-11	13	23
Total Bad	8	-35	36	2	-8	8	X -8	X 19	X 21	-16	13	20	-43	-67	80	28

Besides this, the quadrotor had more problems to fly calm as the lighting conditions became worse. This is shown in Table 4 with the increase in the standard deviation of the position (OTS). The reason for this difficulty is the fact, that with worse conditions, the ADNS failed more and more until totally and the controller had to rely on the low-sampled Fourier Tracking only. Thereby, to prevent against instability, the control parameters for both cases are set very similar and this might be tuned further.

Table 4. Standard deviation under different (difficult) lighting conditions

Standard deviation [cm]	Trial 1		Trial 2		Trial 3		Trial 4		Trial 5		Mean
	X	Y	X	Y	X	Y	X	Y	X	Y	
Good	6	19	16	18	12	13	21	18	11	13	14.7
Bad	19	29	12	14	23	24	18	20	26	15	19.6
Very Bad	20	26	23	20	24	46	15	25	16	23	23.8
Total Bad	28	36	19	19	24	22	28	22	24	29	25.1

Furthermore a switch in the lighting conditions also incorporates difficulties to the control, which is shown on figure 16. In this experiment the conditions have been changed every 10-20 seconds in the following procedure: good - bad – very bad – bad – good

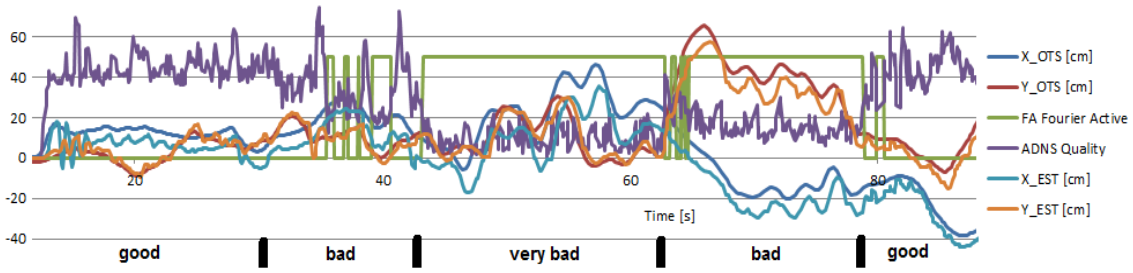

Figure 16. System behaviour and performance under changing lighting conditions Fourier Tracking Off: FA = 0; Fourier Tracking On: FA = 50

It can clearly be seen, that the system becomes fitful under worse changing conditions and even more after a clarification. This experiment also shows that the position system can handle these critical changes in the lighting conditions.

6. CONCLUSION AND PERSPECTIVE

The evaluation proved that the system is capable of a fully autonomous directed flight and the presented complementary vision based data fusion is sufficient for controlling a quadrotor in an autonomous flight. Even autonomous position hold under very bad lighting conditions is possible.

In spite of this, there is still space for optimization and the accuracy as well as the reliability of the system need to be further improved. The control accuracy can be improved by updating the position during rotation. Position errors can be reduced by optimizing the data fusion, its parameters or the concept, but probably by reducing the weight of the ADNS under erroneous

conditions. Still it has been shown, that even under bad lighting conditions the ADNS improves the control behaviour.
However, the reliability of the system can only be significantly improved, by adding other, non-optical sensors for positioning like radar, or ultrasonic, since optical sensors depend inherently on light. For outdoor applications, GPS would also be possible.

The directed flight can now be combined together with other optical systems - like pmd camera and camera based stereo-optical distance determination – to fly fully autonomously through narrow openings like windows or doors.

ACKNOWLEDGEMENTS

The author would like to thank Diana Baeva and Qasim Ali for reviewing this paper. This work was funded by the IHK Würzburg-Schweinfurt and the Universitätsbund Würzburg. This publication was funded by the German Research Foundation (DFG) and the University of Wuerzburg in the funding program Open Access Publishing.

REFERENCES

[1] Kendoul F. et al, Optical-flow based vision system for autonomous 3D localization and control of small aerial vehicles, Robotics and Autonomous Systems 2009, Elsevier
[2] Herisse B. et al, Hovering flight and vertical landing control of a VTOL Unmanned Aerial Vehicle using Optical Flow, 2008 IEEE International Conference on Intelligent Robots and Systems
[3] Reinthal E., Positionsbestimmungeinesautonomen Quadrokopters durchBildverarbeitung, 2014, BA Thesis, University of Wuerzburg
[4] Gageik, N., Autonomous UAV with Optical Flow Sensor for Positioning and Navigation, 2013, International Journal of Advanced Robotic Systems, INTECH
[5] Averbuch A. and Keller Y., A Unified Approach to FFT Based Image Registration, 2002, Tel Aviv University
[6] Reddy B. S. and Chatterji B. N., An FFT-Based Technique for Translation, Rotation, and Scale-Invariant Registration, 1996, IEEE Transactions on Image Processing vol 5 no 8
[7] Arens T. et al, Mathematik, 2008, Heidelberg SpektrumAkademischerVerlag
[8] Jähne B., Practical Handbook on Image Processing for Scientific Applications, 1997, Boca Raton CRC Press LLC
[9] Lucas, B. and Kanade, T. 1981. An iterative image registration technique with an application to stereo vision. In Proceedings of the International Joint Conference on Artificial Intelligence, pp. 674–679.
[10] Srinivasan M., An image-interpolation technique for the computation of optic flow and egomotion, Biological Cybernetics, 1994, Springer-Verlag
[11] Strohmeier M., Implementierung und EvaluierungeinerPositionsregelungunterVerwendung des optischenFlusses, Würzburg 2012, BA Thesis
[12] ADNS-3080 High-Performance Optical Mouse Sensor, Data Sheet, Avago Technologies, http://www.avagotech.com
[13] Gageik N., Rothe J., Montenegro S., Data Fusion Principles for Height Control and Autonomous Landing of a Quadrocopter, UAVveek 2012
[14] Qt Project, http://qt.digia.com
[15] Natural Point, OptiTrack, www.naturalpoint.com/optitrack/
[16] Ascending Technologies, Research Price List, 2013, Krailling, Germany, www.asctec.de
[17] Shen, S., Autonomous Multi-Floor Indoor Navigation with a Computationally Contrained MAV, International Conference on Robotics and Automation, 2011, Shanghai, IEEE

AUTHORS

Dipl.-Ing. Nils Gageik is working as a research assistant and PhD student at the Chair Aerospace Information Technology at the University of Wuerzburg. He received his diploma from the RWTH Aachen University2010 in Computer Engineering.

B. Sc. Eric Reinthal is a Master Student in the international Spacemaster program. He received his bachelor degree in 2014 at the University of Wuerzburg.

B. Sc. Paul Benz is a Master Student at the University of Wuerzburg. He received his bachelor degree in 2013 at the University of Wuerzburg.

Prof.Dr. Sergio Montenegro is holder of the Chair Aerospace Information Technology at the University of Wuerzburg.