INTECH
open science | open minds

OPEN ACCESS
ARTICLE

International Journal of Advanced Robotic Systems

An Autonomous UAV with an Optical Flow Sensor for Positioning and Navigation

Regular Paper

Nils Gageik[1,*], Michael Strohmeier[1] and Sergio Montenegro[1]

1 Chair of Computer Science 8: Aerospace Information Technology, University of Würzburg, Würzburg, Germany
* Corresponding author E-mail: gageik@informatik.uni-wuerzburg.de

Received 14 Jan 2013; Accepted 05 Jul 2013

DOI: 10.5772/56813

Abstract A procedure to control all six DOF (degrees of freedom) of a UAV (unmanned aerial vehicle) without an external reference system and to enable fully autonomous flight is presented here. For 2D positioning the principle of optical flow is used. Together with the output of height estimation, fusing ultrasonic, infrared and inertial and pressure sensor data, the 3D position of the UAV can be computed, controlled and steered. All data processing is done on the UAV. An external computer with a pathway planning interface is for commanding purposes only. The presented system is part of the AQopterI8 project, which aims to develop an autonomous flying quadrocopter for indoor application. The focus of this paper is 2D positioning using an optical flow sensor. As a result of the performed evaluation, it can be concluded that for position hold, the standard deviation of the position error is 10cm and after landing the position error is about 30cm.

Keywords Autonomous UAV, Quadrocopter, Quadrotor, Optical Flow, Positioning, Navigation

1. Introduction

Due to the progress in sensor, actuator and processor technology and the associated reduced costs, combined with the improved performance of such parts, the construction of semi-autonomous and fully autonomous quadrocopters is possible today [1-4]. As there are different definitions of an autonomous system, this phrase is used here for a system that can operate completely independently from any external devices for all of its functionality. In comparison to systems using GPS [2-4] or Optical Tracking Cameras [5] for positioning, which would better be called semi-autonomous, autonomous systems are capable of operating in unknown and GPS-denied environments such as in houses, caves, tunnels or any other places where no accurate GPS signal is available, since they are not dependent on an external device or signal. There exist many different approaches for an autonomous system using ultrasonic sensors [6], infrared sensors [3], laser scanners [7, 8], stereo cameras or the Kinect camera from Microsoft [9]. Each suffers from its own drawbacks, which are reliability, price, weight and size. For reliability

reasons a multi-sensor system is mandatory and video camera based systems benefit from low weight, price and size.

Though approaches using vision-based position sensors for autonomous navigation exist [10-16], those are either not suitable or not adaptable to our system, since a low-cost, quick, accurate, reliable and simple solution was required. Complex solutions suffer from a high computational and implementation burden as well as higher risk of failure because of unknown system behaviour. A vision based SLAM (simultaneous localization and mapping) algorithm is presented in [12], but the high computational burden is done on an external CPU, breaking our definition of autonomy. This is also the case for the optical flow computations of [14] and [15], which must be performed on an external computer because of the complexity of the design and the high computational burden. Therefore, this paper presents a simple design and a quick solution for autonomous quadrocopter navigation using only the principle of optical flow for positioning. The presented solution can easily be fused with other solutions and is adaptable to any system as well as extendable. This paper explains in detail a realization of this approach for autonomous flight, its capability and its drawbacks.

In [16] an approach for collision avoidance of a UAV using optical flow is discussed, which was evaluated by simulations only. In contrast to that this paper shows reliable empirical data from autonomous experimental flights under real indoor conditions and evaluated with an independent optical tracking system.

2. Optical Flow for Positioning

The most common methods for optical flow computation are differential, matching, energy-based and phase-based. Matching and phase-based methods have a high computational burden. Since the differential method of Lucas and Kanade [17] is widespread and shows acceptable computational burden and good performance [18], it was decided to implement this method on different hardware to compare the result with the ADNS-3080 optical flow sensor.

2.1 Lucas-Kanade Method

Lucas and Kanade [17] assume that the change between two pictures is small and constant. This means that the transformation is valid within the neighbourhood M and a translation in x-y-plane is assumed, not taking rotations and translations in the z-axis into account. Comparing two iterative pictures, this leads to the following least-squares optimization solution [19]:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum_{(i,j)\epsilon M} P_x(i,j) * P_t(i,j) \\ \sum_{(i,j)\epsilon M} P_y(i,j) * P_t(i,j) \end{bmatrix} * \begin{bmatrix} \sum_{(i,j)\epsilon M} P_x^2(i,j) & \sum_{(i,j)\epsilon M} P_x(i,j) * P_y(i,j) \\ \sum_{(i,j)\epsilon M} P_x(i,j) * P_y(i,j) & \sum_{(i,j)\epsilon M} P_y^2(i,j) \end{bmatrix}^{-1} \quad (1)$$

Here $P_x(i,j)$, $P_y(i,j)$ and $P_t(i,j)$ are the partial intensity derivatives of point P(i, j) in the x- or y-axis or after time t and u and v are the searched optical flow values in the x- and y-axis.

2.2 Algorithm of Srinivasan

This optimization can be implemented by applying the algorithm of Srinivasan [19]. It can be simplified by comparing the intensities of Pixels [Figure 1] and reducing M to 1. The current valid pixel P_t is compared with the previous pixel P_{t-1} and is scaled within the left and right neighbour P_2 and P_1 as well as the upper and lower neighbour P_4 and P_2. With Equation 2 the optical flow can then be computed.

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

A derivation of Equation 2 can be also found in [20], but another explanation is that it follows as a result of the simple substitution of the partial derivative (Equation 1) by the potential difference (Equation 3) with respect to the axis or time.

$$\begin{aligned} P_x(i,j) &= P_2 - P_1 \\ P_y(i,j) &= P_4 - P_3 \\ P_t(i,j) &= P_t - P_{t-1} \end{aligned} \quad (3)$$

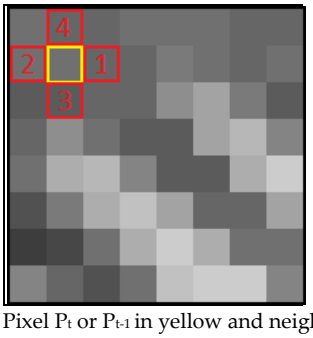

Figure 1. Centre Pixel P_t or P_{t-1} in yellow and neighbours in red

Srinivasan [19] also presents a solution that takes rotations into account, but for our simple implementation a solution for 2D position change is sufficient for now.

One advantage of this algorithm is that it can easily be implemented using one loop through all pixels. Furthermore, the inversion of a 2x2 matrix presents no difficulty, so the algorithm is for real time execution. [21] presents a speed-up version of the Lucas-Kanade-Method and [22] provides an open-source implementation of this algorithm.

3. Optical Flow Sensors

3.1 Implementation (Optical Flow)

The goal was to implement an optical flow sensor pointing to the ground in order to compute the relative position of a quadrocopter flying at a height of about 1m. To add this to our autonomous quadrocopter for navigation, different solutions were investigated. Because of the high data volume and rate of visual systems as well as the limited memory and computational power on board the quadrocopter, picking the right camera sensor is not trivial. Constraints on the camera are a simple interface to a microprocessor, frame rate, resolution, price and weight. A closer look was taken at the Centeye Vision Chip Tam2 [22], the 4D USART µCAM [23], the CMUCam 4 [24], the OV7670 from OmniVision [25] and the ADNS-3080 [26].

3.2 Sensor System Comparison

The Tam2 has a resolution of 16x16 (black-white) with 25fps. Centeye provides a breakout-board with an implementation of the optical flow algorithm. This is a good starting point and the system showed good results for detecting the movement of nearby objects, but because of the fixed lens and low-resolution, more distant objects were blurred.

The µCAM is easy to connect using USART, but we only achieved a maximal sample rate of 13fps, which is too slow for the differential method and our application. Reasons for this were the slow USART communication interface and a processing delay of the camera.

The CMUCam 4 is a microcontroller-board for computer vision with an open source firmware and 32kB memory. The processor has eight cores and is programmed in SPIN. Therefore, it has a high initial training effort and the low memory is a problem. Furthermore, the price and size of this system are not suitable for our application.

The OV7670 is a VGA (colour) camera chip with a resolution of 640x480. For data processing the STM32 F4 Discovery Board [27] was used and its 8bit parallel data interface was connected to the sensor for reading pictures. The SCCB, an I²C like interface, is for sending commands to the camera (configuration) only. Because of the limited memory of the microprocessor, QVGA instead of VGA format was used, which is also supported by the camera. From this 320x240 QVGA picture only a 64x64 pixel sub-window is processed, also because of the limited memory (192kB RAM) of the microprocessor. With this system 30fps was achieved.

The ADNS-3080 is an optical mouse sensor and is available on a breakout-board with SPI interface as a ready optical flow sensor for two degrees of freedom. The sensor has a resolution of 30x30 pixels and achieves 6400fps. For a shorter shutter speed and better results under bad lighting conditions, the frame rate was set to 2000fps. The board cost about $40 and is small (2x3cm). Its biggest disadvantage is that the software is not open source and its functionality therefore is unknown, not adaptable and not extendable. Also, dust and dirt are a big problem. The lens is replaceable and it was switched to a focal length of f = 16mm for a better focus on objects with a distance of about 1m.

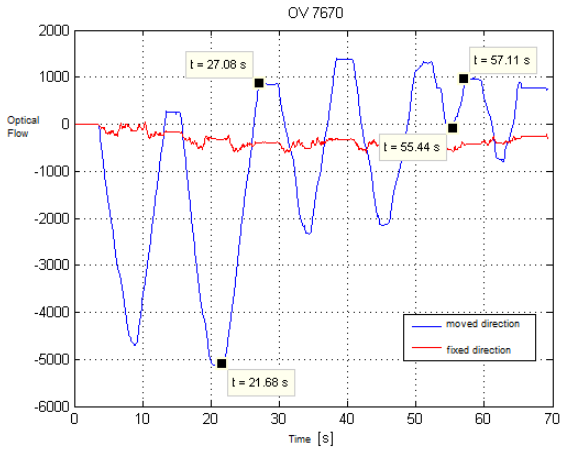

Figure 2. Optical Flow of OV7670 with 30fps at different speed

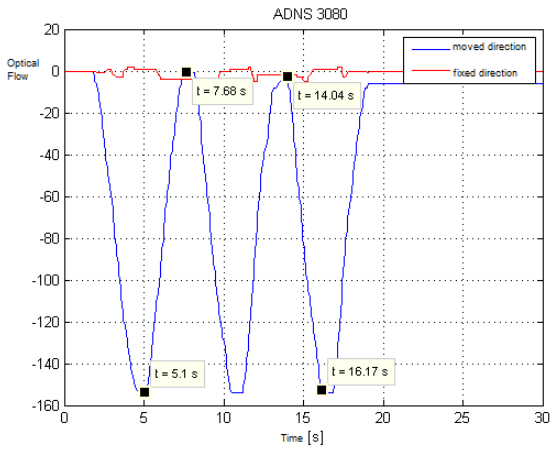

Figure 3. Optical Flow of ADNS-3080 with 2000fps at different speed

3.3 Evaluation (Optical Flow)

Only the OV7670 and the ADNS-3080 seemed to be suitable for our requirements, so a closer evaluation of both systems was made. Both sensors were moved about 20cm back and forth in one direction with different speeds and the results were compared [Figure 2-3]. Both systems are capable of detecting the movement and its direction with the principle of optical flow. Furthermore, it showed clearly that the implementation of the OV7670 with 30fps dependents on the speed, while this is not the case for the ADNS-3080.

3.4 Conclusion (Optical Flow)

The implemented differential method assumes that the transformation between two pictures is not higher than one pixel. Therefore, only speeds according to the frame-rate are measurable. For the OV7670 this means that 30 pixel translations per second are completely detectable. This could be an explanation for the poor performance of the OV7670, since the ADNS-3080 provides 6400fps.

Because the ADNS-3080 showed a good performance, it was implemented in the UAV for autonomous navigation.

4. Implementation (Autonomous UAV)

4.1 System Implementation

The hardware-design of the quadrocopter is shown in Figure 4. The total price for all hardware components is about €500. The fusion of the infrared, ultrasonic, pressure and inertial sensor for height estimation is described in [28].

The position estimation is carried out, integrating the optical flow measurements. The used parameters of the ADNS-3080 are 2000fps, 30x30 pixel resolution, 400 CPI and automatic shutter speed. These are the default parameters of the sensor. A higher frame rate achieved no better results under normal light conditions.

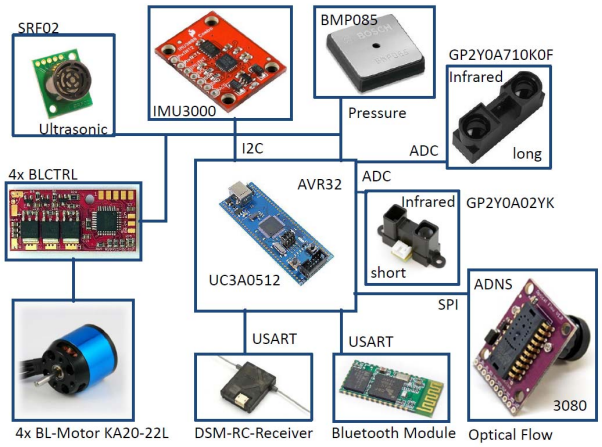

Figure 4. Hardware Design

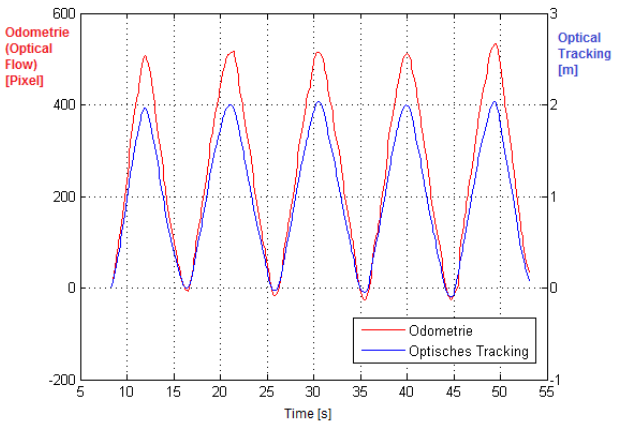

Figure 5. Scaling Factor F_s Calibration

The result needs to be scaled according to height. To find the correct scaling factor, F_s, the sensor was moved several times 2m along one direction at a height of 1m and its translation was tracked with the optical tracking system PPT X4 [29] as the true reference value [Figure 5].

The measured distances were then optimally fitted to the optical tracking measurements using three different methods:

- F_s is assigned by Method of least squares (MLS)
- F_s is the ratio of Mean of both curves
- F_s is the averaged ratio of the difference between neighboured minima and maxima

The resulting scaling factors F_s are 259.51, 257.64 and 259.21. Because the minima-maxima-method is the simplest method and showed the same results as MLS, this method can be used for scaling the sensor into metres.

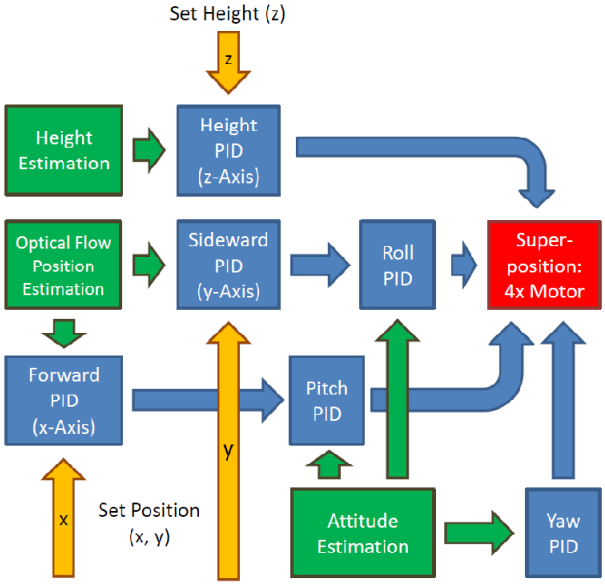

Figure 6. System-Core Design: Sensors & Signal Processing (Green), Commands & External Set Points (Yellow), Internal Control Data Processing (Blue), Superposition & Output (Red)

4.2 Control Design

For each degree of freedom an empiric optimized PID controller was implemented. This cumulates for a six DOF system to six cascading PID controllers [Figure 6]. The height estimation and the set height are the inputs of the height control, which determine the total voltage of all four motors and regulate in this way the lift of the system. One fourth of the voltage corresponds to an 8bit value and the total voltage is distributed to the four motors according to the outputs of the attitude control. This fusion is done by superposition and the restrictions of the motors have to be considered here.

The optical flow estimation is the input (measurement) of the two position controllers (forward, sideward). The 2nd input for the forward or sideward controller is the set point for position, x or y respectively, which can be changed remotely. This enables, together with the changeable height, an autonomous 3D flight. The differences between the set points and optical estimations are the errors, which are the inputs of the two PID position controllers (x- and y-axis). The outputs of the position controllers are the set point of the roll and pitch axis attitude controllers.

4.3 Control Software

For debugging and evaluation purposes, as well as to control the quadrocopter, a control program was developed using Qt [30]. The program was used to display pictures of the OV7670 and the ADNS-3080 and to trace and change parameters. The position of the quadrocopter, scaled in metres, can be tracked on a 2D map [Figure 7], using either the optical tracking system as a reference or the optical flow sensor. The first one corresponds to the true position and the second one to the assumed position used by the quadrocopter for position control and autonomous flight. Using mouse clicks, set points for positions can be changed in real time and sent to the quadrocopter remotely (Bluetooth).

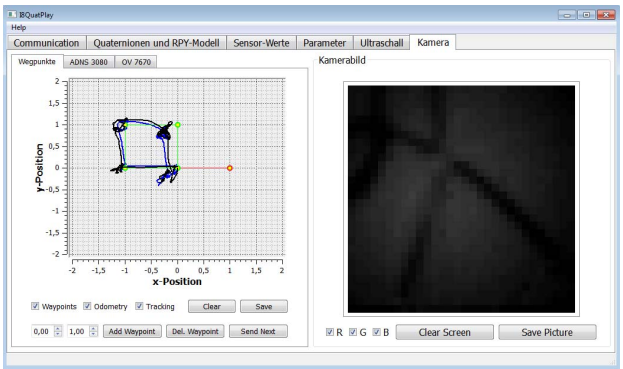

Figure 7. Qt Control-Software

5. Evaluation (Autonomous UAV)

The capabilities of the system were extensively evaluated [20]. The evaluation consists of the static position hold, a simple position change as an easy dynamic test case and two complex dynamic test cases performing a fully autonomous flight.

5.1 Static Position Hold

In this experiment the quadrocopter was manually started and then the position holding was activated. The position controller uses only the optical flow sensor and the optical tracking is for evaluation of the system. The quadrocopter stood in the same position in the air 1m above ground for six minutes. Its position was tracked during the whole flight [Figure 8-9]. For both optical tracking and odometry (optical flow) the standard deviation from the centre is about 0.1m and the double standard deviation is about 0.16m, implying that 95% of the time a quadrocopter with a spread of 64cm (end of propeller to end of propeller) is within a circle of 96cm (64 + 2x16) diameter.

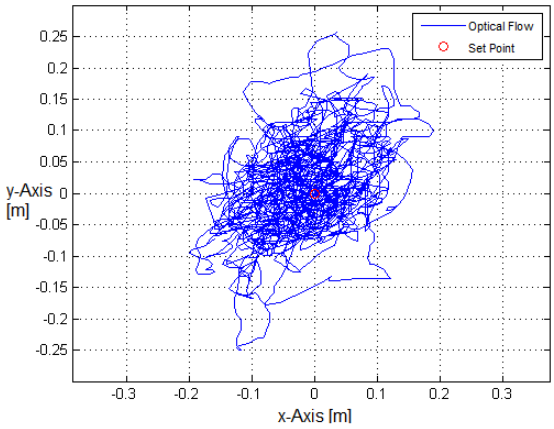

Figure 8. Position Hold: Odometry (Optical Flow)

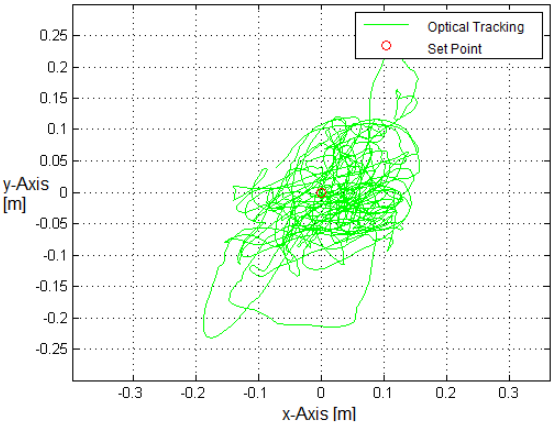

Figure 9. Position Hold: Optical Tracking

Figure 10. Position at start (left) and after 365s (right)

Comparing both figures [Figure 8-9], a small difference can be detected, which corresponds slightly to a yaw transformation. This becomes clearer looking at Figure 10, which shows two pictures of the quadrocopter from above, made at the beginning of the experiment and after 365s. During the experiment the quadrocopter rotated about 13° around the yaw axis.

The explanation for this is the integration drift of the gyroscope. Since the system does not use a magnetometer for yaw compensation, this axis is not compensated and drifts. Since the optical flow computation assumes only translational transformations and no rotations, these rotations cause an error in the optical flow computation. This can also be proven by rotating the quadrocopter manually around the yaw axis. The optical flow sensor interprets this as a position change and the controller tries to compensate for this incorrectly measured position error and flies the quadrocopter to a nearby wrong position.

5.2 Dynamic Control

To investigate the control behaviour of the system, an experiment was performed, where the quadrocopter had to react to a step response. The quadrocopter was in position hold at p_0 = (x = 0, y = 0) and a new set point p_1 = (x = 2m, y = 0) was set remotely. Figure 11 shows the reaction of the system.

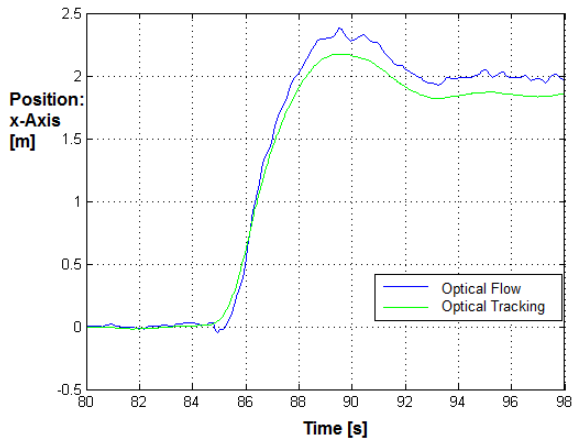

Figure 11. Step Response: (0,0) to (2,0)

The figure proves the stability of the controller. The new set point is reached within about 3s (rise time). The overshoot is about 15% (ca. 0.3m) and the settling time is between 7s and 9s.

The final position error is about 0.15m. This could be a scaling error, but then a linear transformation would exist, which converts one line into the other. Taking a closer look at the graph, this is not the case [Figure 11]. The source of error must be somewhere else. Since the quadrocopter rotates around its pitch axis to achieve a new position on the x-axis and pitch rotations also cause incorrectly measured position changes through the optical flow sensor, this could be an explanation. This is most likely, since Figure 11 shows that this offset corresponds to a similar difference between both curves at 85s. At 85s the quadrocopter received the new set point and pitched. This pitching (pitch axis rotation) causes an incorrect optical flow measurement, which can be seen in the figure. Though the optical tracking shows that the quadrocopter is moving forward, the odometry indicates incorrectly the opposite at the beginning of the manoeuvre.

5.3 Autonomous Flight

In this experiment an autonomous flight was performed. The quadrocopter had to fly the *Nikolaus*-house. This is a child's game, where a house has to be drawn in one run without withdrawing the pen, so that each wall is drawn only once. This drawing was the flight path of the quadrocopter [Figure 12]. On the map of the control software the drawing of the *Nikolaus*-house was displayed, which was executed by the autonomous quadrocopter and corresponds to its flight path [Figure 13]. There is a fixed calibration offset $p_o \approx$ (x = 0.2m, y= 0.05m), which transforms the optical tracking measurements into optical flow measurements. This is because the quadrocopter was manually flown to the zero point of the optical tracking coordinate system and then the autonomous flight was activated, setting the current point as the zero point of the odometry coordinate system. Aside from the calibration offset the evaluation showed further position errors, which cannot be solved easily. The figure illustrates these already mentioned problems, such as the yaw drift problem and the pitch and roll problem, all resulting in a position offset between odometric and optical tracking (true) position. Nevertheless, the evaluation proves the potential of the system. Therefore, it was shown that the quadrocopter is capable of flying autonomously and able to draw *Nikolaus*-houses.

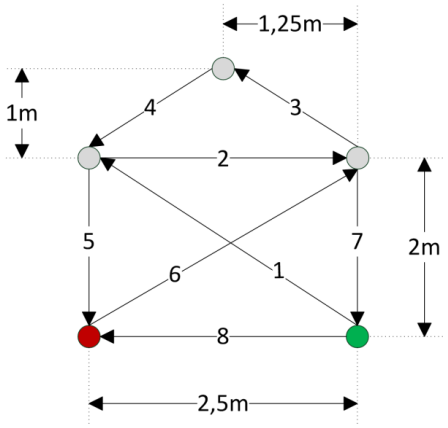

Figure 12. Flight plan

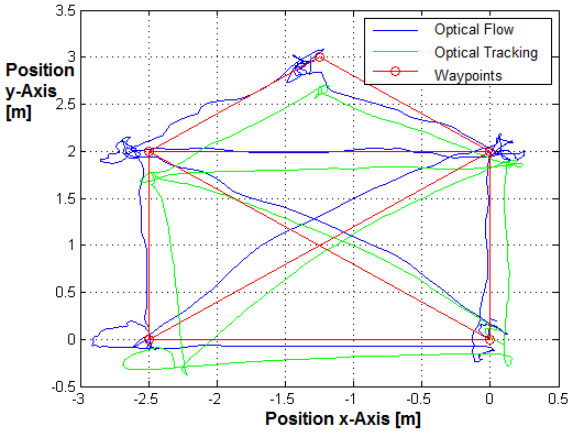

Figure 13. Autonomous Flight: Position from Optical Flow Sensor (blue), Position from Optical Tracking (green), Waypoints (red circles)

5.4 Autonomous Manoeuvres

In the last experiment the concept of an autonomous flight was extended by an autonomous starting and landing manoeuvre. In this experiment the quadrocopter is given a list of six waypoints and then the start command is sent by a PC remotely. The quadrocopter, still standing on the ground, started, proceeded to follow the waypoints list and landed at the last waypoint autonomously.

The waypoints in metres are w_1 = (x = 0, y = 0), w_2 = (x = -0.4, y = 0), w_3 = (x =-0.8, y = 0), w_4 = (x = -1.2, y = 0), w_5 = (x = -1.2, y = 0) and w_6 = (x = -2.0, y = 0).

The manoeuvre was realized with a flight control state machine containing three states: *Starting*, *Flying* and *Landing*. In the *Starting* state the quadrocopter controls its height and keeps its position until a flying height of 1m is achieved. Then it switches to the *Flying* state in which it processes the waypoint list. After the last waypoint is reached, the quadrocopter switches to the *Landing* state and performs an autonomous landing. Figure 14 and Figure 15 show the measured positions of the quadrocopter using the optical flow sensor and the waypoints of two performed flights.

The landing completes the manoeuvre and the position error can be measured by comparing the final set position with the position of the quadrocopter on the ground after landing. This experiment was performed several times and the results prove that the quadrocopter is capable of performing this manoeuvre. The quadrocopter can also fly autonomously over an obstacle such as a chair placed on the ground. Table 1 demonstrates eight resulting measurements of the position in centimetres. Furthermore, the elapsed time between sending the remote command and landing on the ground is shown for several runs.

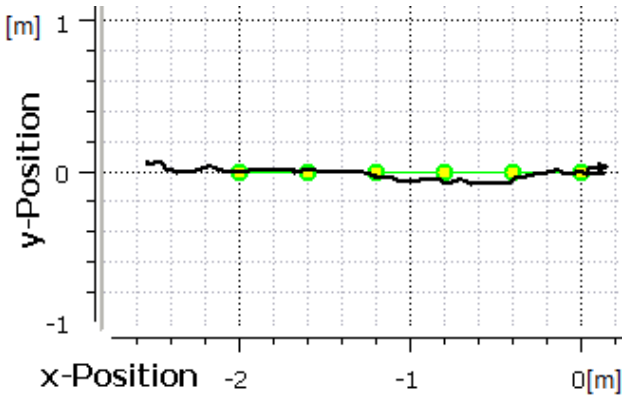

Figure 14. Flight Path of Fly 3: Waypoints (Green Dots) and Optical Flow Computed Position (Black Line)

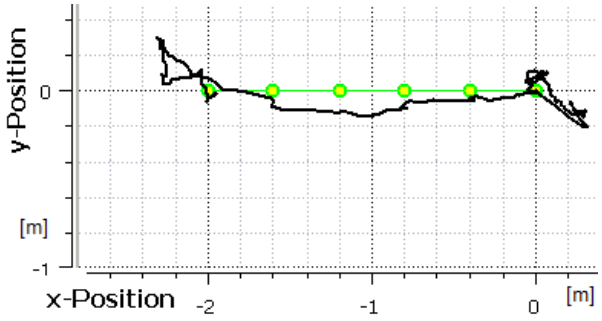

Figure 15. Flight Path of Fly 7

Final Position Error [cm]	X-Error [cm]	Y-Error [cm]	Time [s]
37	26	27	-
44	34	27	-
42	37	21	-
42	42	6	-
28	20	19	< 48
22	16	14	< 27
11	1	11	< 53
33	32	6	27

Table 1. Results of Autonomous Manoeuvres: Final Position Error in Centimetres, the Position Error in x- and y-axis and spent time for whole manoeuvre

The evaluation demonstrates that in this experiment the final position error after landing on the ground is about 0.3m and the elapsed time for the whole manoeuvre is about 30s. The quadrocopter always landed less than 45cm from the goal and all errors have the same sign and a similar amount. Therefore, it can be supposed that there is a systematic error that needs to be investigated. However, there is also an unpredictable error, which cannot be completely overcome with this approach. The system measures and controls the optical flow while changing the height (translation in z) for starting and landing. This is one explanation for the position error. Furthermore, the implemented algorithm supposes waypoints to be reached with a tolerance of 14cm in the air (before landing). The final position error can also be reduced by reducing this tolerance and improving the starting and landing manoeuvres, which are also sources of error. A more complex waypoint list like the flight plan of Figure 12 was also executed and showed similar results. This confirms the assumption that errors during the starting and landing manoeuvres are the main reason for final position errors.

6. Conclusion and Perspective

It could be proved that an autonomous flight with an optical flow sensor is possible. The presented system is capable of performing a complete flight in all three geometrical dimensions from starting till landing without human interaction. Hence no external reference system such as optical tracking or GPS is required.

The position error, after an autonomous flight over eight set points, is about 20cm, while on position hold the standard deviation of the error over six minutes is only 10cm. This error is mainly caused by accumulated system, sensor and control noise. There is still potential for a higher accuracy of the system, since the evaluation demonstrated different drawbacks of the current implementation. This is especially the case for dynamic position changes and autonomous flight.

The yaw drift problem can be addressed using a magnetometer as a reference and compensated for such drifts. In addition to this, we are working on a gimbal mounted bottom plate, which is regulated by two servo motors according to the attitude of the quadrocopter. This means the pitch and roll angle of the bottom plate with the integrated optical flow sensor is constantly zero and the attitude of the optical flow sensor is independent of the rotations of the quadrocopter. The system could also be improved by fusing orientation information with the optical measurements to compensate for incorrect measurements due to rotation.

Another improvement can be achieved by replacing the sensor and implementing a very high fps solution taking rotations and translations along the z-axis into account. The final position error after an autonomous start, pathway flight and landing is about 30cm. This can be reduced with a more accurate starting and landing procedure, which takes changes of height for the optical flow position computation into account.

7. Acknowledgments

This publication was funded by the German Research Foundation (DFG) and the University of Wuerzburg in the funding program Open Access Publishing. The author also would like to thank Michael Ruffer and Daniel Hayden for their support on writing this paper.

8. References

[1] Nonami K., Autonomous Flying Robots, Springer, ISBN-10: 4431538550, 2010
[2] Microdrones GmbH, www.microdrones.com
[3] ArduCopter, http://code.google.com/p/arducopter
[4] HiSystems GmbH, www.mikrokopter.de
[5] Mellinger D. et al., Trajectory Generation and Control for Precise Aggressive Maneuvers with Quadrotors, The International Journal of Robotics Research, Vol. 31 No. 5, 2012
[6] Gageik N., Mueller T., Montenegro S., Obstacle Detection and Collision Avoidance Using Ultrasonic Distance Sensors for an Autonomous Quadrocopter, UAVveek 2012
[7] Grzonka S. et al., A Fully Autonomous Indoor Quadrotor, IEEE Transactions on Robotics, Vol. 28 No. 1, February 2012
[8] Gronzka S., Mapping, State Estimation, and Navigation for Quadrotors and Human-Worn Sensor Systems, PhD Thesis, Uni Freiburg, 2011
[9] Lange S., Sünderhauf N. Neubert P., Drews S., Protzel P., Autonomous Corridor Flight of a UAV Using a Low-Cost and Light-Weight RGB-D Camera, Advances in Autonomous Mini Robots, ISBN: 978-3-642-27481-7, 2012
[10] Ding W. et al., Adding Optical Flow into GPS/INS Integration for UAV navigation, International Global Navigation Satellite Systems Society, IGNSS 2009
[11] Wang J. et al., Integration of GPS/INS/VISION Sensor to Navigate Unmanned Aerial Vehicle, The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. XXXVII Part B1, Beijing 2008
[12] Blösch M. et al., Vision Based MAV Navigation in Unknown and Unstructured Environments, Robotics and Automation (ICRA), 2010 IEEE International Conference, 21-28, 2010
[13] Corke P., An Inertial and Visual Sensing System for a Small Autonomous Helicopter, Journal of Robotic Systems, Vol. 21, No. 2, 43-51, 2004

[14] Kendoul F. et al., Optical-flow based vision system for autonomous 3D localization and control of small aerial vehicles, Robotics and Autonomous Systems, Elsevier, 2009
[15] Herisse B. et al., Hovering flight and vertical landing control of a VTOL Unmanned Aerial Vehicle using Optical Flow, IEEE International Conference on Intelligent Robots and Systems, 2008
[16] Zing S. et al., MAV Navigation through Indoor Corridors Using Optical Flow, 2010 IEEE International Conference on Robotics and Automation, 2010
[17] Lucas, B. and Kanade, T. An iterative image registration technique with an application to stereo vision. In Proceedings of the International Joint Conference on Artificial Intelligence, 674–679, 1981
[18] Barron et al., Performance of Optical Flow Techniques, International Journal of Computer Vision, Vol. 12, No. 1, 43-77, 1994
[19] Srinivasan M., An image-interpolation technique for the computation of optic flow and egomotion, Biological Cybernetics, Springer-Verlag, 1994
[20] Strohmeier M., Implementierung und Evaluierung einer Positionsregelung unter Verwendung des optischen Flusses, Würzburg, BA Thesis, 2012
[21] Rav-Acha A., Peleg S., Lucas-Kanade without iterative Warping, International Conference on Image Processing, IEEE, 1097-1100, 2006
[22] Centeye Inc., http://centeye.com/
[23] μCAM Serial JPEG Camera Module, Data Sheet, 4D Systems, www.4dsystems.com.au
[24] CMUcam: Open Source Programmable Embedded Color Vision Sensors, http://www.cmucam.org
[25] OV7670/OV7171 CMOS VGA (640x480) Camera Chip, Implementation Guide, OmniVision, http://www.ovt.com/
[26] ADNS-3080 High-Performance Optical Mouse Sensor, Data Sheet, Avago Technologies, http://www.avagotech.com
[27] STM32 F4 DISCOVERY, Data Brief, September 2011, STMicroelectronics, http://www.st.com
[28] Gageik N., Rothe J., Montenegro S., Data Fusion Principles for Height Control and Autonomous Landing of a Quadrocopter, UAVveek, 2012
[29] WorldViz, www.worldviz.com
[30] Qt Project, http://qt.digia.com