MAV Navigation through Indoor Corridors Using Optical Flow

Simon Zingg, Davide Scaramuzza, Stephan Weiss, Roland Siegwart
Autonomous Systems Lab
ETH Zurich

***Abstract*—Safe navigation through corridors plays a major role in the autonomous use of Micro Aerial Vehicles (MAVs) in indoor environments. In this paper, we present an approach for wall collision avoidance using a depth map based on optical flow from on board camera images. An omnidirectional fisheye camera is used as a primary sensor, while IMU data is needed for compensating rotational effects of the optical flow. The here presented approach is designed for safely maneuvering a helicopter through an indoor corridor. Results based on real images taken in a corridor with textured walls are shown at the end of this paper.**

I. Introduction

THE past years showed an increasing interest in MAV applications in different environments. Surveillance and reconnaissance for military as well as for civil purposes are main tasks of Unmanned Aerial Vehicles (UAV). Currently, UAVs are used mainly in open sky, away from ground obstacles penetrating their flight-paths. Based on GPS data, geological obstacles such as hills or mountains can be bypassed. However, growing interests in smaller airborne vehicles flying autonomously in near-earth or even indoor environments call for other methods for obstacle avoidance. In urban applications or indoor use, collision avoidance cannot be achieved based on GPS-data alone. Too many unforeseen obstacles may cross the way path of the MAV. Especially in autonomous use, reliable obstacle avoidance is indispensable for ensuring the MAV's survivability.

In order to ensure collision avoidance, new approaches are used, mainly relying on onboard sensors that are able to scan the instant environment and to provide the controller with reliable data to safely maneuver the MAV in an unknown environment.

Different sensors have been used for detecting obstacles. Kumar and Ghose[1] and Kwag and Kang [2] implemented radar based navigation and obstacle avoidance. Saunders et al. [3] used a forward looking laser range finder for path planning. However, this two approaches lack in heavy weight or high power consumption. For a flying platform it is important to keep weight light and to save energy wherever possible. Therefore, having a lightweight sensor with a small power-consumption is of great interest. Both targets are reached using a camera. Since it is a passive sensor, power requirements are low while having a light weight. These arguments declare the camera to one of the most suitable sensors for obstacle avoidance. Based on the optical flow achieved by camera images, it is possible to estimate the distance to surrounding objects. Computing this distance is highly expensive in computation, though, which is the major disadvantage of using camera images for obstacle avoidance.

The here presented approach uses camera images as primary sensor data for estimating the distance of surrounding obstacles. The camera is a 190° Field-Of-View (FOV) fisheye camera pointing downwards. This allows us to get distance information from all around the MAV. The flying platform is a quadrotor helicopter having a diameter of 53cm, equipped with an Inertial Measurement Unit (IMU). Computations are done on an external computer. This algorithm is optimized for navigating a MAV through an indoor corridor.

Using image feature tracking, the optical flow of two images, taken with a short time interval, is calculated. This optical flow is caused by translation and rotation of the helicopter. Depth estimation can be achieved only from optical flow caused by translation. Using data provided by the onboard IMU, optical flow effects caused by rotation can be compensated.

Based on this optical flow a depth map is created, containing depth information from the current environment of the helicopter. Since the MAV is placed in a corridor, it detects a wall on either side of the helicopter, while the median lateral distances to the walls on both sides are measured. Using this information, the error towards the center of the corridor can be calculated. Normalized with the overall measured width of the corridor, this error can then be used as input of a PD controller steering the helicopter through the corridor.

The normalized error allows us the use of this algorithm without having information about the amount of speed or the corridor width, which is very useful since no speed information of the helicopter are available and applications in different environments are possible. However translational movement of the MAV is needed to detect surrounding obstacles.

The paper is organized as follows: in Section II, we review the work in this area; in Section III, we present our equipment and in Section IV we describe our approach. Finally, in Sections V and VI we present the experimental results and

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n. 231855 (sFly). Simon Zingg is currently a Master student at the ETH Zurich (email: szingg@student.ethz.ch). Davide Scaramuzza is currently senior researcher and team leader at the ETH Zurich (email: davide.scaramuzza@ieee.org). Stephan Weiss is currently PhD student at the ETH Zurich (email: stephan.weiss@mavt.ethz.ch). Roland Siegwart is full professor at the ETH Zurich and head of the Autonomous Systems Lab (email: r.siegwart@ieee.org).

draw the conclusions.

II. Related Work

The use of optical flow for obstacle avoidance is a widespread approach. Many different obstacle avoidance strategies rely on this phenomenon, which is often presented as biologically inspired. Tammero et al. [4] showed that fruit flies avoid obstacles when turning away from regions with high optical flow, while Srinivasan et al. [5] found out that honeybees flying through a tunnel try to balance the amount of optical flow on both sides in order to maintain equidistance to the flanking walls. Adapting these behaviors, Serres et al. [6] developed an autopilot for lateral obstacle avoidance of a hovercraft. Two linear cameras pointing $\pm$ 90° to the side provided optical flow. By balancing the optical flow on both sides, they made the hovercraft navigate in the middle of a corridor.

Hrabar [7] used a similar method for lateral obstacle avoidance of a rotorcraft flying in an urban environment. For depth estimation, stereo cameras were used. However, stereo cameras require heavier payload to the MAV, which actually should be prevented. Nevertheless, single camera collision avoidance for frontal obstacle is possible. Zufferey et al. [8], [9] implemented such a system on a 10g microflyer. Using two linear cameras for measuring optical flow, he computed the divergence of the optical flow on the left and right side of the direction of travel. Increasing divergence indicates a frontal obstacle which can be safely avoided with proportional rudder deflection. This system was successfully tested in an indoor environment that was properly modified by adding bare-code-like texture on the walls.

Similarly, Muratet et al. [10] used the optical flow field of a perspective camera facing the direction of travel. In case of a divergence of the field from one point, a frontal obstacle could be detected. The point of divergence is called focus of expansion. This situation allows us to compute the time to impact onto the obstacle. If the time to impact falls below a threshold, the controlled helicopter stops and executes a 180° turn.

Besides lateral and frontal obstacle avoidance, altitude control is another application of optical flow for controlling MAVs. Ruffier and Franceschini [11] regulated the altitude of a helicopter using two downward optical flow sensors. Similar implementations have been done by Zufferey [9] and Green et al. [12] who additionally implemented an autonomous landing strategy. While keeping optical flow constant, speed is reduced successively, causing the MAV to approach the ground and finally touch down.

All the above mentioned approaches use optical flow as a primary input. However, it is possible to use optical flow for computing a depth map containing obstacles surrounding the MAV. Based on this map, the desired waypath of the MAV can be planned taking the detected obstacles into account. Call et al. [13], [14] presented a method to detect obstacles using a forward looking onboard camera. Distances were measured based on optical flow amplitude and GPS data. A three dimensional map provided a rough estimation of the obstacle locations. The fixed wing MAV then used a sliding mode control law to avoid obstacles.

Fig. 1. The Hummingbird quadrotor helicopter provided by Ascending Technologies.

Fig. 2. The μEye camera with a 190° lens recording monochrome pictures from the surrounding of the helicopter. It is pointing downwards onto the ground.

From this brief literature review we can see that many different techniques for obstacle avoidance based on optical flow have been realized so far. The focus is laid on approaches using optical flow as control input. Such systems can be applied in special environments only. Whereas, depth map based navigation would allow to navigate flying robots in a more complex indoor environment.

III. Equipment

A. Flying Platform

Our MAV is the Ascending Technologies Hummingbird[1](see Fig. 1), a quadrotor helicopter having an overall diameter of 53 cm and a payload of 200 grams. The operational flying time varies between 23 minutes without payload and 12 minutes with full payload. Additionally, the MAV is equipped with a fully working Inertial and Measurement Unit (IMU) providing information about the pitch, roll, and yaw angle of the helicopter. Furthermore, its built-in controller allows us to regulate the overall thrust, angular positions of pitch and roll angles, and the angular speed of the yaw angle. For communication purposes, a ZigB communication board allows us to send control inputs to the helicopter and to grab IMU data.

B. Camera

As for the camera, we used the μEye camera (see Fig. 2) from IDS[2]. The resolution of this monochrome camera is

[1]http://www.asctec.de

[2]http://www.ids-imaging.com/

752×480 pixels, while the maximal possible frame rate lays at 87 frames per second. The mounted fisheye lens provides a field of view of up to 190°. The camera is attached to the center of the helicopter pointing downwards. In this orientation, the recorded images contain information from the surrounding of the MAV.

C. Computer

The overall computation was done off-board on a 2GHz Dual Core laptop. Connection to the helicopter IMU and controller was ensured via the ZigB wireless communication board while images of the on-board camera were streamed via WiFi. The overall algorithm ran at 20 Hz.

IV. Approach

A. Optical Flow

Optical flow is an often used method for non stereo-vision based collision avoidance. It is a visual phenomenon experienced daily when observing an object moving at a different speed than the observer. The motion of the observed obstacle depends on the distance between observer and obstacle and their relative speed. Optical flow can therefore be used for estimating relative distances. It is measured in units of angular velocity such as radians per second or degrees per second. If optical flow is extracted from images of a video camera, as it is done in our approach, optical flow is the motion of a particular pixel from one video frame to another.

In the here presented method for obstacle avoidance, the camera is fixed on the helicopter. Since the surroundings of the robot is assumed to be stationary, optical flow is only caused by the motion of the MAV. Optical flow caused by near obstacles is bigger than optical flow caused by obstacles farther away. This phenomenon can be used for estimating the relative distance from the camera to the obstacles.

The calculation of optical flow is one of the most demanding aspects concerning computational power in this application. Several different methods have been developed [15]. Differential methods call upon spatio-temporal intensity derivatives, correlation approaches rely on feature matching, while frequency-based methods use velocity-tuned filters in the Fourier-domain. Lucas and Kanade[16] introduced an algorithm that allows us calculation of sparse optical flow, i.e. optical flow is not computed for every pixel but only for certain previously determined pixels that are selected in the first frame. For pixel motion larger than the search-window, a pyramidal approach for the Lucas-Kanade method is used [17], where the standard algorithm is applied recursively to resized versions of the image. This allows us to scan a bigger area and to find optical flow with magnitudes larger than the ones detected by the standard approach.

The Pyramidal Lucas-Kanade optical flow detector has been applied in our algorithm. The pixels in the first frame are selected using the Shi and Tomasi's corner finder[18].

B. Depth Estimation in Straight Flight

Based on the optical flow magnitude, it is possible to estimate the distance of the detected object. Assuming linear motion of the MAV, optical flow is a function of the robot's forward velocity v, the distance to the obstacle D and the angle between the direction of travel and the obstacle α (see Figure 3).

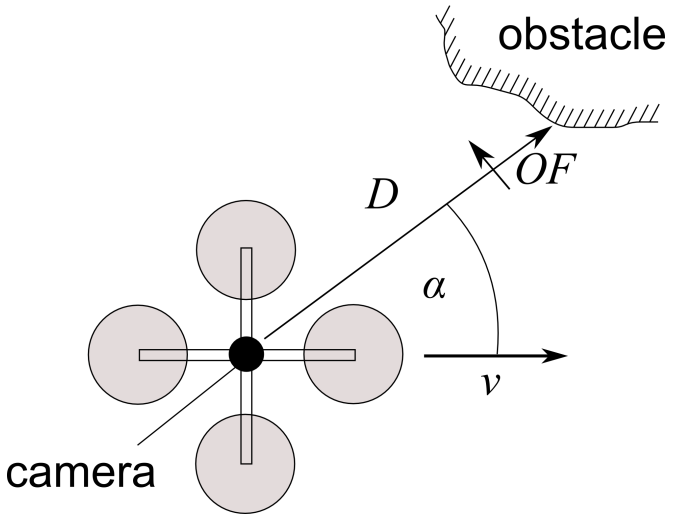

Fig. 3. Optical flow during the translational flight of the MAV.

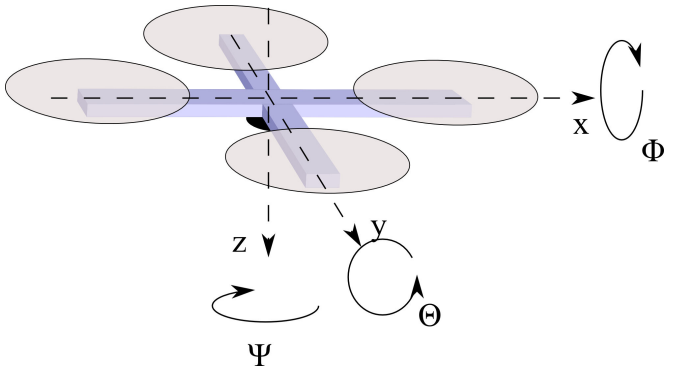

Fig. 4. The quadrotor helicopter navigates in six degrees of freedom and can therefore perform motion in x,y and z direction as well as rotation around Θ, Φ and Ψ

$$OF = \frac{v}{D} \cdot sin\alpha \tag{1}$$

Solving this equation for D gives:

$$D = \frac{v}{OF} \cdot sin\alpha \tag{2}$$

Therefore, when speed and optical flow data are available, it is relatively easy to compute the distance towards a detected object, as long as pure translational movement of the robot can be taken for granted.

The quadrotor helicopter, however, is a system navigating in six degrees of freedom. Besides translation in x, y, and z direction, it is able to rotate around pitch Θ, roll Φ and yaw Ψ angles as shown in Fig. 4. In a general case, pure translation cannot be assumed. Rotations of the MAV cause additional optical flow which is not usable for distance-estimation. Compensating this rotational effects is therefore indispensable.

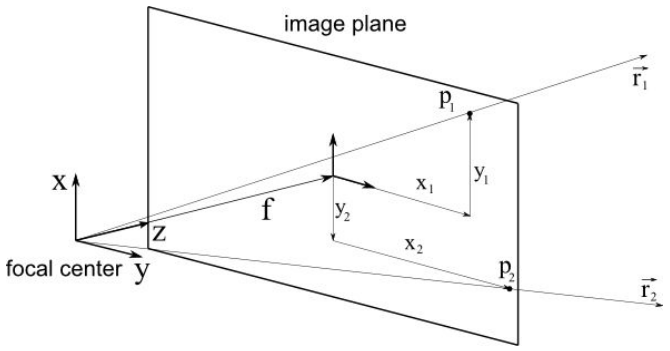

Fig. 5. Pinhole camera model with tracked point p_1 and matching point p_2 found by the Lucas-Kanade algorithm.

C. Compensation of Rotation-based Effects

Using a pinhole camera model, the compensation for the rotation is explained. Assuming two frames f_1 and f_2 are acquired at times t_1 and t_2. From t_1 to t_2 the robot performs a motion consisting of translation and rotation. The angular changes from t_1 to t_2 are $\Delta\Theta$, $\Delta\Phi$, and $\Delta\Psi$ for pitch, roll, and yaw angles. The corresponding angular changes can be extracted from IMU data grabbed at times t_1 and t_2. Optical flow is now calculated using the above mentioned Shi-Tomasi corner finder for selecting pixel p_1 in frame f_1 and the Pyramidal Lucas-Kanade approach for finding the matching pixel p_2 in frame f_2. Relying on the pinhole camera model, it is possible to find for each pixel p_1 and p_2 the corresponding ray $\vec{r_1}$ and $\vec{r_2}$ in real world coordinates. Assuming the pixels p_1 and p_2 have image coordinates x_1, y_1 and x_2, y_2 respectively, the corresponding rays in camera coordinate system are

$$\vec{r_1} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + \lambda_1 \cdot \begin{pmatrix} x_1 \\ y_1 \\ f \end{pmatrix} \tag{3}$$

and

$$\vec{r_2} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + \lambda_2 \cdot \begin{pmatrix} x_2 \\ y_2 \\ f \end{pmatrix} \tag{4}$$

while f is the focal length and the origin of the image coordinate system is placed in the center of the image as shown in Fig. 5.

Since the camera is mounted close to the helicopter's center of gravity, it can be assumed that the coordinate systems of the camera and the helicopter match under the condition that the axis of both coordinate systems are collinear. This implies that rotations around the x, y, and z axis of the helicopter can be adopted as rotations around the x, y, and z axis of the camera coordinate system.

Because for depth estimation only translational movement of the helicopter can be taken into account, pixel p_2 has to be placed at the position it would be if there was no rotation. To achieve this, the ray $\vec{r_2}$ is transformed into the ray $\vec{r_2'}$ that is placed in the coordinate system x', y', and z' having no rotation, see Figure 6. Using the well-known Euler angles

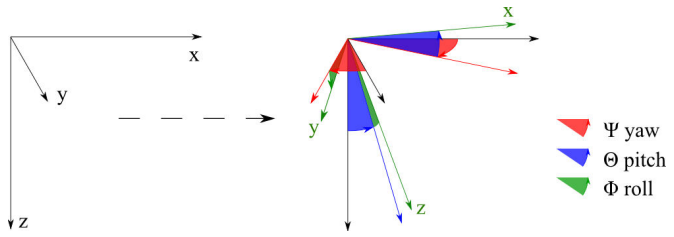

Fig. 6. Motion consisting of translation and rotation

the following transformation is achieved:

$$\vec{r_2'} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\Delta\Phi & -\sin\Delta\Phi \\ 0 & \sin(\Delta\Phi) & \cos(\Delta\Phi) \end{pmatrix} \cdot \begin{pmatrix} \cos(\Delta\Theta) & 0 & \sin(\Delta\Theta) \\ 0 & 1 & 0 \\ -\sin(\Delta\Theta) & 0 & \cos(\Delta\Theta) \end{pmatrix} \cdot \begin{pmatrix} \cos(\Delta\Psi) & -\sin(\Delta\Psi) & 0 \\ \sin\Delta\Psi & \cos(\Delta\Psi) & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \vec{r_2} \tag{5}$$

$\vec{r_2'}$ now points into the direction the feature would have at time t_2 if the helicopter would not have performed rotation. Using $\vec{r_2'}$ allows us to calculate reliable optical flow for depth estimation either by calculating the corresponding pixel p_2' or by applying the scalar product:

$$\vec{r_1} \cdot \vec{r_2'} = |r_1| \cdot |r_2'| \cdot \cos\gamma \tag{6}$$

where γ is the angle between r_1 and r_2'. Since optical flow is measured in rad/s, optical flow can be described as follows:

$$OF = \frac{\gamma}{\Delta t} = \frac{\arccos(\frac{\vec{r_1} \cdot \vec{r_2'}}{|r_1| \cdot |r_2'|})}{\Delta t} \tag{7}$$

with $\Delta t = t_2 - t_1$.

Applying a pinhole model, however is not appropriate when using a 190° fisheye camera. Therefore, calculating the rays r_1 and r_2 from image coordinates of the pixels p_1 and p_2 is not as intuitive as described above. Nevertheless, it is not impossible. A toolbox calibrating omnidirectional cameras has been developed by Scaramuzza [19] and can be downloaded from [20]. It allows us an easy calculation of rays corresponding to image pixel coordinates. The resulting ray $\vec{r}_{final}$ is assumed to have the following structure:

$$\vec{r}_{final} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + \lambda \cdot \begin{pmatrix} x_{final} \\ y_{final} \\ g(\rho) \end{pmatrix} \tag{8}$$

where x_{final} and y_{final} are the x and y coordinates of the pixel in the image frame and $\rho = \sqrt{x_{final}^2 + y_{final}^2}$. $g(\rho)$ has the form of a polynomial and is computed during a calibration procedure based on images taken by the used camera. ρ is the distance between the observed image point and the image center.

After computing $\vec{r_1}$ and $\vec{r_2}$ using the calibration toolbox, the application of the Euler-angles and optical flow calculation is the same as described above.

D. Depth Map from Optical Flow

The ability of calculating accurate optical flow depending only on the linear motion of the robot allows us to calculate

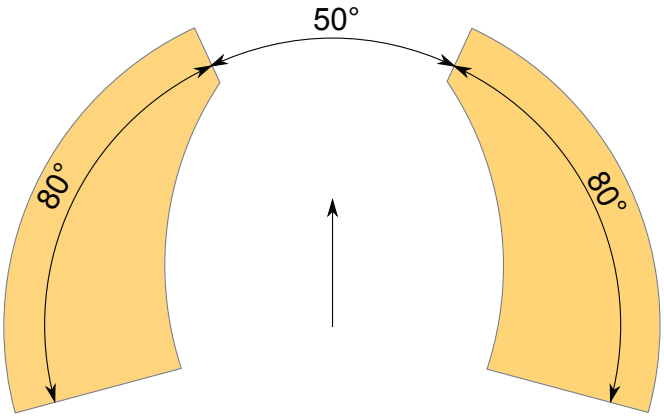

Fig. 7. The regions of interest in which optical flow is calculated. The central arrow indicates the direction of travel.

a map containing the depth of the MAV's surroundings. For each optical flow element in the image, the distance to the real-world feature causing the optical flow can be estimated as mentioned above. Using this depth information, a temporary map can be built containing the distances towards obstacles in all directions from which optical flow is measured. This depth map is the heart-piece for the helicopter navigation with obstacle avoidance.

In the here presented case, the test environment is an indoor corridor. Using the depth map, both walls of the corridor on either side of the helicopter are detected. Ensuring that the main portion of optical flow is caused by the wall, optical flow is calculated in certain regions of interest within the image only. This region is shaped similar to the wall appearance in the image. This sections contain two segments of 80°, one on each side of the MAV as shown in Figure 7. A frontal section of 50° is not taken into account, since depth estimations close to the direction of travel causes numerical problems.

In average, approximately 500 optical flow elements are detected, about 250 on each side. Since wrong matches in the Lucas-Kanade algorithm giving wrong distance estimations cannot be avoided, the calculated optical flow is inspected with a filter for removing outliers.

A simple threshold filter removes too large optical flow amplitudes. In a further step, an angular criterion is checked. Optical flow has to be tangential to a circle with its center at the center of the image with a deviation of up to 50°. This threshold seems to be large, it has been chosen experimentally and proved to work though. It is necessary, since the optical flow has not an exact circular shape. If the threshold is chosen smaller, optical flow close to the direction of travel might be filtered out, even if it was not wrongly matched. A remarkable amount of wrong matches could be extracted using these two criteria.

E. Error Estimation

For safe navigation through the corridor, the position error of the MAV towards the center of the corridor is of significant importance. Error calculation can be done based on the previously computed depth map. Each depth estimation based on an optical flow element can be written in the form of a vector pointing from the camera towards the detected feature on the wall as seen in Fig. 8:

$$\vec{D} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \tag{9}$$

Where the magnitude of $\vec{D}$ is the absolute distance to this particular feature. Since we focus on the error towards the center of the corridor, it is useful to project the map onto a two dimensional plane which is defined by the x and y axis of the helicopter coordinate system. Therefore, the z component of the distance vector $\vec{D}$ is neglected. On this two-dimensional map, the walls can be detected easily. The median sideways distance of the MAV towards the right wall is corresponding to the median of all y_R values of the distance vectors $\vec{D}_R$ pointing to the right, whereas the left-hand distance is the median of the absolute value of all y_L values of the distance vectors $\vec{D}_L$ pointing left. The error e towards the corridor's center can now be calculated as the difference between the distance to the right and the distance to the left divided by two:

$$e = \frac{|\tilde{y_R}| - |\tilde{y_L}|}{2} \tag{10}$$

where $\tilde{y_R}$ and $\tilde{y_L}$ indicates the median values of all y_R and y_L components, respectively. Using the median distance on both sides is useful to detect upcoming obstacles that may narrow or widen the corridor. However, because we only consider the absolute angles yielded by the on-board IMU (neither velocity nor its direction), the distance toward the detected walls can only be estimated relatively to the MAV's speed. Facing this fact, a way had to be found to eliminate the unknown speed from the error to convert it into an appropriate value for the later use as control input. The wished effect can be achieved when normalizing the above found error e with half of the overall width of the corridor, which is the sum of $|\tilde{y_L}|$ and $|\tilde{y_R}|$. The normalized error e_n then has the following form:

$$e_n = \frac{|\tilde{y_R}| - |\tilde{y_L}|}{|\tilde{y_R}| + |\tilde{y_L}|} \tag{11}$$

The normalized error varies from -1, if the MAV is located at the right wall, to $+1$, if the MAV is located at the left wall. This error e_n can be used as input signal for a controller navigating the helicopter through the corridor.

F. Implementation

The flow chard of the control system is shown in Fig. 10. In a first step, two camera images are acquired with corresponding IMU data. The time between two consecutive image acquisitions is 100 ms. The most time-consuming elements are the image and IMU data acquisition as well as optical flow calculation. The built-in IMU of the helicopter has to be polled each time for data acquisition. Furthermore, it sends by default a larger data set than needed for angular informations only. Those properties induce a high time consumption of up to 0.1s for a single data transmission from the IMU.

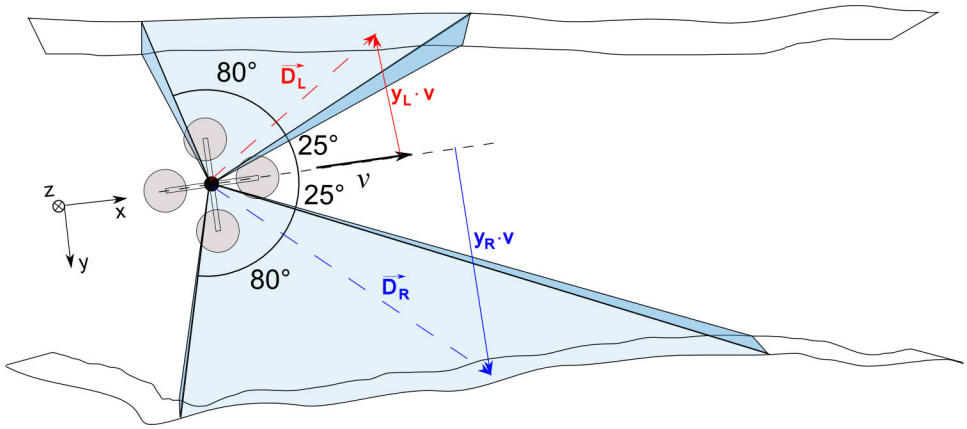

Fig. 8. The MAV within the corridor. Depth estimation is done on both sides. Based on this, the error towards the corridor's center is calculated. On each side approximately 250 distances are estimated. $\vec{D}_R$ indicates all distance vectors pointing right, while $\vec{D}_L$ indicates all distance vectors pointing left.

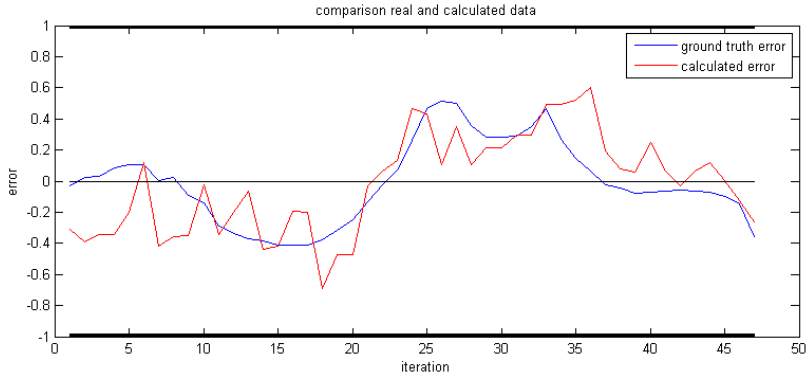

Fig. 9. Comparing the error of the MAV towards the center of the corridor (marked with a thin black line), computed by the depth map based algorithm and using ground truth measurements.The walls of the corridor are marked with thick black lines.

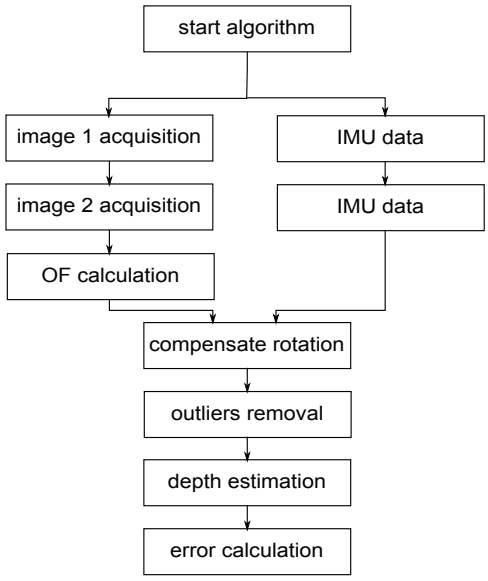

Fig. 10. Flow chard of the algorithm.

V. Experiments

To check the functionality of the introduced error estimation, tests have been performed using real data.

A. Test Setup

Since the algorithm is optimized for use in a corridor, the chosen test-area is an indoor corridor having a width of 2.5m and a height of 3m. The walls were already heavily textured. No additional features or special illumination were used to improve the performance of feature tracking and optical flow calculation.

The helicopter was flown manually by remote control, while the proposed algorithm was used for successive error estimation. The estimated error, computed by our algorithm, was then compared with the real ground truth error.

B. Test Results

C. Controller Simulation

The error based on ground truth measurements and calculated by our algorithm can be seen in Fig. 9. At a first glance, we can notice a correlation between the ground truth error and the error based on depth map calculation. However, the computed error seems to be very noisy and can have a deviation of up to a quarter of the corridor width.

Having a closer look, the following parameters can be observed:

- average deviation: $0.146 \approx 7.3\%$

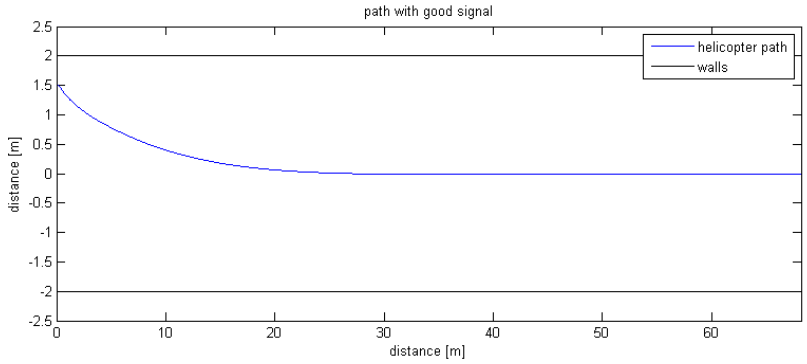

Fig. 11. Way-path of the simulated helicopter in a 4m wide corridor using a PD controller and exact input signal.

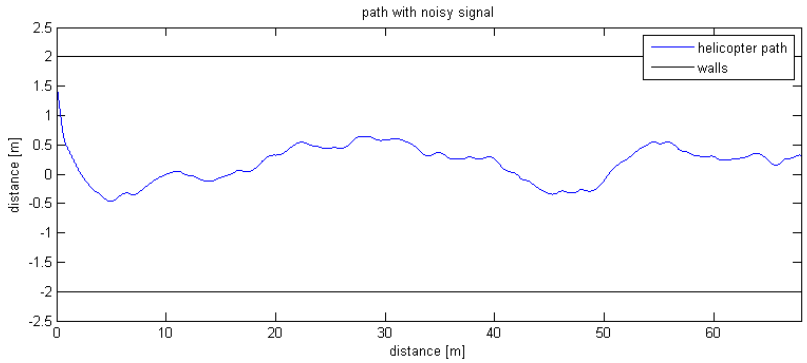

Fig. 12. Way-path of the simulated helicopter in a 4m wide corridor using a PD controller. The input signal has deviations similar to the error computed by the depth map algorithm.

- median deviation: $0.188 \approx 9.4\%$
- maximum deviation: $0.54 \approx 27\%$

This deviations can be explained in this way. If the IMU data are noisy, and therefore not precise enough, the compensation for rotational effects cannot work properly and produces wrong results. Especially, inaccurate information of the yaw angle Ψ can cause wrong sideways depth estimations, since its effect is a decrease of the optical flow magnitude on one side of the MAV and an increase of the optical flow on the other side.

Nevertheless, it will be shown that the computed error is sufficiently accurate to be used as input argument for a controller.

Since the helicopter is an unstable platform, all three angles (pitch, roll, and yaw) have to be controlled. The helicopter's on-board controller regulate the angular position of pitch and roll and the angular speed of the yaw angle.

Using the computed error, lateral changes in the position of the MAV should be achieved. To reach this, the controller-outputs change the angular position of the roll angle and the angular speed of the yaw angle. The pitch angle is held at a constant value of 0.5° to reach a forward speed parallel to the x-direction.

Since height control cannot be achieved using the error-input, the thrust is kept constant, what causes an approximated constant height.

For controlling the roll and yaw angles, a PD controller is chosen. Using a MATLAB Simulink based point mass model of the quadrotor helicopter, a discrete time controller $C(z)$ working with a sampling time of $T = 0.5\text{s}$ has been designed:

$$C(z) = \frac{(k_p + k_D) \cdot z - k_p}{z} \tag{12}$$

The tuning parameters k_p and k_D are chosen as follows:

$$k_p = 0.2 \tag{13}$$

and

$$k_D = 0.009 \tag{14}$$

The resulting way-path of the helicopter in a corridor with a width of 4m is shown in Fig. 11 with exact input argument. As initial conditions, the MAV is positioned 1.5m to the left of the center of the corridor and the instant yaw angle is chosen such that the x-axis is parallel to the walls of the corridor.

In Fig. 12 the input signal is provided with noise, similar to the noise achieved by the error estimation from the depth map algorithm. As can be observed, the MAV stabilizes around the center of the corridor. Therefore, it can be assumed that the input computed by the above presented algorithm is sufficient for ensuring save navigation of the MAV.

VI. Conclusion

In this study, we addressed the problem of obstacle avoidance using vision based methods, where the navigation of a miniature quadrotor helicopter within an indoor corridor was the main task. Relying on an optical flow based depth map, the walls of the corridor could be detected and the error toward

the center of the corridor computed. The depth map is built on optical flow measurements caused by translational movement of the helicopter. The rotational effects in the optical flow were compensated using IMU data. Test results based on real images showed that very accurate IMU data are required for reliable depth informations. A simulated PD controller showed a centering behavior of the helicopter with input arguments as provided by the algorithm.

We believe that depth-map based collision avoidance may evolve as a very powerful tool, since it might be applied in unknown environments with a high density of objects, and therefore be ideal for use in indoor environments.

VII. Acknowledgements

The authors would like to thank Marco Zamponi for his great support and valuable contribution.

References

[1] B. Ajith Kumar and D. Ghose, "Radar-assisted collision avoidance, guidance strategy for planar flight," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 37, pp. 77–90, Jan 2001.

[2] Y. Kwag and J. Kang, "Obstacle awareness and collision avoidance radar sensor system for low-altitude flying smart uav," *Digital Avionics Systems Conference, 2004. DASC 04. The 23rd*, vol. 2, pp. 12.D.2–121–10 Vol.2, Oct. 2004.

[3] J. B. Saunders, O. Call, A. Curtis, A. W. Beard, and T. W. Mclain, "Static and dynamic obstacle avoidance in miniature air vehicles," in *in Proceedings of the Infotech@Aerospace Conference*, pp. 2005–6950, 2005.

[4] L. F. Tammero, V. L. S. Building, and D. Of, "The influence of visual landscape on the free flight behavior of the fruit fly drosophila melanogaster," *Journal of Experimental Biology*, vol. 205, pp. 327–343, 2002.

[5] M. V. Srinivasan, S. W. Zhang, M. Lehrer, and T. S. Collett, "Honeybee navigation en route to the goal: Visual flight control and odometry," *Journal of Experimental Biology*, vol. 199, pp. 237–244, 1996.

[6] J. Serres, D. Dray, F. Ruffier, and N. Franceschini, "A vision-based autopilot for a miniature air vehicle: joint speed control and lateral obstacle avoidance," *Auton. Robots*, vol. 25, no. 1-2, pp. 103–122, 2008.

[7] S. E. Hrabar, *Vision-Based 3D Navigation for an Autonomous Helicopter*. PhD thesis, USC, Jan 2006.

[8] J.-C. Zufferey, *Bio-inspired Flying Robots: Experimental Synthesis of Autonomous Indoor Flyers*. Lausanne: EPFL/CRC Press, 2008.

[9] J.-C. Zufferey, A. Klaptocz, A. Beyeler, J.-D. Nicoud, and D. Floreano, "A 10-gram Vision-based Flying Robot," *Advanced Robotics*, vol. 21, no. 14, pp. 1671–1684, 2007. Special issue on IROS2006.

[10] L. Muratet, S. Doncieux, Y. Briere, and J.-A. Meyer, "A contribution to vision-based autonomous helicopter flight in urban environments.," *Robotics and Autonomous Systems*, vol. 50, no. 4, pp. 195–209, 2005.

[11] F. Ruffier and N. H. Franceschini, "Aerial robot piloted in steep relief by optic flow sensors.," in *IROS*, pp. 1266–1273, IEEE, 2008.

[12] W. E. Green, P. Y. Oh, K. Sevcik, and G. Barrows, "Autonomous landing for indoor flying robots using optic flow," in *ASME Int. Mech. Eng. Congress and Expo*, pp. 1341–1346, Mech, 2003.

[13] B. R. Call, "Obstacle avoidance for unmanned air vehicles," Master's thesis, Brigham Young University, December 2006.

[14] B. Call, R. Beard, C. Taylor, and B. Barber, "Obstacle avoidance for unmanned air vehicles using image feature tracking," in *AIAA Guidance, Navigation, and Control Conference*, August 2006.

[15] J. L. Barron, D. J. Fleet, S. S. Beauchemin, and T. A. Burkitt, "Performance of optical flow techniques," *International Journal of Computer Vision*, vol. 12, no. 1, pp. 43–77, 1994.

[16] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision (ijcai)," in *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI '81)*, pp. 674–679, April 1981.

[17] J. Y. Bouguet, "Pyramidal implementation of the lucas kanade feature tracker: Description of the algorithm," 2002.

[18] J. Shi and C. Tomasi, "Good features to track," *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 593–600, June 1994.

[19] D. Scaramuzza, A. Martinelli, and R. Siegwart, "A toolbox for easy calibrating omnidirectional cameras." In Proceedings of IEEE International Conference on Intelligent Robots and Systems, 2006.

[20] D. Scaramuzza, "Omnidirectional camera calibration toolbox for matlab, first release 2006, last update 2009, google for "ocamcalib"."