Proceedings of the EuroGNC 2013, 2nd CEAS Specialist Conference
on Guidance, Navigation & Control, Delft University of Technology,
Delft, The Netherlands, April 10-12, 2013

Stereo Vision based Obstacle Avoidance on Flapping Wing MAVs

S Tijmons, GCHE de Croon, BDW Remes, C de Wagter, R Ruijsink, E van Kampen, QP Chu

Abstract One of the major challenges in robotics is to develop a fly-like robot that can autonomously fly around in unknown environments. State-of-the-art research on autonomous flight of light-weight flapping wing MAVs uses information such as optic flow and appearance variation extracted from a single camera, and has met with limited success. This paper presents the first study of stereo vision for onboard obstacle detection. Stereo vision provides instantaneous distance estimates making the method less dependent than single camera methods on the camera motions resulting from the flapping. After hardware modifications specifically tuned to use on a flapping wing MAV, the computationally efficient Semi-Global Matching (SGM) algorithm in combination with off-board processing allows for accurate real-time distance estimation. Closed-loop indoor experiments with the flapping wing MAV DelFly II demonstrate the advantage of this technique over the use of optic flow measurements.

1 Introduction

Autonomous flight of flapping wing MAVs (FWMAVs) is a considerable challenge. The main reason for this is that their light weight prevents the use of heavy and energy-consuming laser scanners that are successful on heavier MAVs such as quad rotors [1] [2]. Still, there have been several attempts at achieving autonomous flight with FWMAVs. Hines at al. [3] describes an FWMAV design that is currently not

S Tijmons, B D W Remes, C de Wagter, R Ruijsink, E van Kampen, Q P Chu
Control and Operations Division, Faculty of Aerospace Engineering, Delft University of Technology, Kluyverweg 1, 2629HT Delft, The Netherlands, e-mail: s.tijmons@student.tudelft.nl, microuav@gmail.com

G C H E de Croon
Advanced Concepts Team, European Space Agency, 2200 AJ Noordwijk, The Netherlands, e-mail: guido.de.croon@gmail.com

able to fly on its own, but experiments show it is able to control its pitch and roll angle by using actuators that change the wing shape. Lin et al. [4] shows the altitude control of the 10 gram FWMAV called *Golden Snitch*. No onboard processing or sensing is used for this task. Using an external stereo camera the position of the vehicle is determined, and further control is performed by a ground station. Duhamel et al. [5] presents an experiment with a 101 milligram flapping wing microrobot called *RoboBee*. Using an onboard optic flow sensor and a well textured screen, the altitude is successfully controlled offboard in a closed-loop experiment, with only small oscillations and a slight drift. Baek et al. [6] performs closed-loop altitude control on a 12 gram ornithopter by using an external camera. In a follow up on this research [7], a 13 gram ornithopter is presented that is able to fly autonomously to a target, using an onboard infrared sensor for target tracking and 3-axis gyroscopes for attitude estimation. During 20 trials a success rate of 85% is reached. Garcia Bermudez et al. [8] performes optic flow measurements on a 7 gram ornithopter. Heavily down-sampled onboard camera images are stored onboard during flight, and uploaded to a computer afterwards to compute optic flow. The main finding is a strong coupling between body motion and the sensed optic flow. Tedrake et al.[9] shows autonomous flight of an ornithopter with a 2-meter wingspan. Only pitch control has been tested successfully using an IMU.

With DelFly II several autonomy experiments have been performed dealing with various control tasks [10]. These tests range from height control with an external camera to height control and path following with an onboard camera and offboard processing. Also a novel appearance cue for obstacle avoidance is introduced [11] [12]. It is based on the principle that when an object is approached, its colors and detailed texture become more and more visible, while other objects move out of sight. It is shown that this cue is a useful complement to optic flow for detecting obstacles with the DelFly.

This experiment showed that optic flow is still not sufficient to perform obstacle avoidance on FWMAVs. To perform good optic flow measurements the camera images should be noiseless and rotation rates should be known, requiring three gyroscopes that can measure the rotational speeds of the vehicle. Measurements should be performed onboard, but the amount of onboard processing power is currently too limited. Therefore the video signal is sent to a ground station, which implies a low frame rate. The frame rate of 30 FPS and line-by-line recording of the camera result in large image distortions that affect the optic flow quality.

In this paper the use of stereo vision is proposed to circumvent these problems. For optic flow image sequences are used, while stereo vision uses images taken at the same time. Vehicle motion has therefore a smaller influence on the quality of the measurements and the video frame rate is of no importance on the quality of individual measurements. Furthermore, it gives an instantaneous overview of obstacles in sight of the camera.

In Section 2 a description is given of the DelFly system including stereo cameras and ground station. Section 3 discusses stereo vision and the algorithm used in this study. The performance of the stereo vision system is presented in Section 4. Closed-

loop autonomy experiments and their results are discussed in Section 5. Finally a summary of the conclusions is given in Section 6.

2 System Design

Since the research in this study focused on FWMAVs, tests were performed with the DelFly II. Its design is shown in Figure 1. The most defining feature of the DelFly is that there is always a camera and transmitter onboard (in this study two cameras). The current version of DelFly II is also equipped with gyrometers, a pressure meter, and onboard processing for these high-frequency measurements. Additional defining features are its biplane wing model and its tail. For more details, the interested reader is referred to [12]. Figure 2 shows an overview of all system components and their interactions.

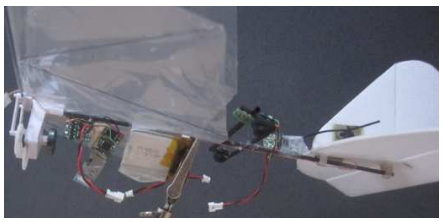

Fig. 1 Side-view of DelFy II including stereo vision cameras

For communication with the ground station a Bluetooth transceiver is used. This system operates at the same frequency as the NTSC transmitter of the stereo system: 2.4GHz. Wi-Fi networks normally operate around this frequency as well. As a result the images received on the ground can become noisy, as is illustrated in figure 3. The ground station is a 2.30GHz dual-core system running on Windows 7. The system is prone to several types of delay. It takes around 60ms to receive the stereo images on

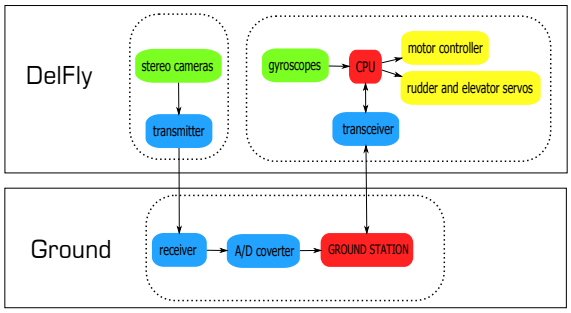

Fig. 2 Diagram of the interaction between all system components

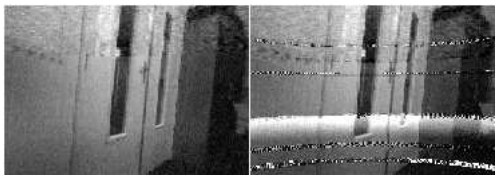

Fig. 3 Example of noise due to an interfering source. Left is noise free, right contains severe noise

the ground. Processing is then performed in real-time (40 ms) and control signals are then send via Bluetooth. This is the slowest step, which at least takes around 60ms. However, because of interference from the other systems operating around the same frequency, this delay varies over time and can become more than 200ms in some cases.

The main feature of interest is the stereo vision camera, which will be discussed in more detail. Due to the stereo camera system the weight of the DelFly in this current configuration is more than usual. Normally the total weight including sensors and batteries is under 17 gram. However, the stereo vision system, including a separate battery, accounts for 5.2 gram. The total weight of the DelFly in this configuration is 21.1 gram.

The selected configuration of the DelFly for this study is for slow-forward flight because of the purpose of indoor obstacle avoidance. In this configuration the speed can still be increased to several meters per second, but it can also fly stable with only 0.6m/s. Hovering is not possible due to the heavy weight of the configuration. The speed is controlled by the tail elevator. The rudder can be used to make turns. The turn speed can be controlled accurately with a servo. However, there is variation in the response of the DelFly to a rudder input. The turns are therefore not strictly circular. Furthermore, giving too much rudder input will result in a fast spiral motion. Still, with sufficient rudder input the turn diameter is less than 1m.

2.1 Stereo Vision Camera

The stereo camera system is the main sensor on the DelFly. Its components can be seen in more in detail in figure 4. The setup consists of two synchronized CMOS 720x240 cameras (with an offset of 7.6 cm) running at 25 Hz and a 2.4 GHz NTSC transmitter. The cameras have a field of view of ± 60 degrees horizontally. Because there is only one transmitter, the video streams from both cameras have to be combined as one. In the initial setup this was done as follows: an NTSC frame consist of an even field and an odd field. To combine two synchronized NTSC cameras, the even lines of the first camera are scanned first, and then the camera source is switched and the uneven lines of the second camera are scanned. This image-based scheme results in frames which consist of image lines from the left and right camera alternately. The resulting frame size is still the same (720x480) but the resolution for each camera has now been reduced to 720x240 pixels.

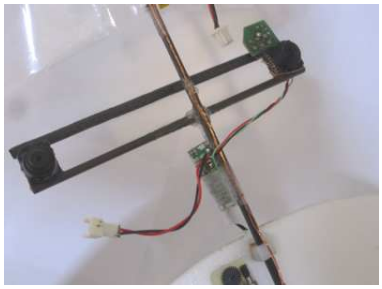

Fig. 4 Stereo camera system. The base line of the cameras is 7.6cm

During early tests with the camera system a shortcoming of this setup was noticed. The result from the stereo matching process was strongly affected by the motion of the camera. During static tests the results were promising and proved to be reliable, but during motion the results would become distorted. Since all even lines are scanned before the uneven lines in this image-based scheme, there is a time difference between the scan lines from the left and the right image. The first line of the 'transmitted' image comes from the right camera, the second line comes from the left camera. When the camera is at rest, it can be roughly assumed that these two lines are observing the same features. When the camera is in motion, this assumption does not hold anymore because of the time difference of approximately 20ms (half the time between two frames) between the lines. During this time the cameras might have changed orientation and the left and right image lines cover different view directions. As a result, the output from the stereo matching process becomes distorted.

The hardware of the camera system was changed such that each time after a scan line has been scanned, the system switches to the other camera. As a result of this line-based scheme the frames sent by the transmitter now consist of two sets of two images that have been taken at different times. This is illustrated in figure 5. Two images (one from the left camera and one from the right camera) are captured on the even lines first (light colors), and after that another set of stereo images is captured on the uneven lines (dark colors). The images on the uneven lines are always the most recent stereo images, and these are used for stereo processing. Each individual image now has a resolution of 720x120 pixels. The benefit of this approach is that the time difference between the stereo images has been reduced significantly. Instead of switching between cameras after 240 lines have been scanned, switching is now done after each single scanline.

So by changing the hardware synchronization from an image-based scheme to a line-based scheme, the time difference has been reduced with a factor 1/240 to roughly 83μs. For the purpose of stereo matching it is assumed that two consecutive uneven image lines (which always contain image lines from both cameras) cover the same image areas.

The impact of this modification is shown in figure 6. A small test was performed where the stereo camera setup was positioned at a fixed height above a large chessboard (to assure texture). A record was made of the camera stream while during the

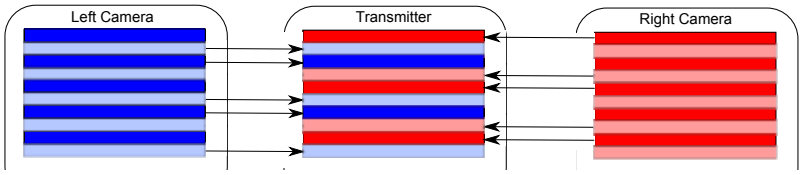

Fig. 5 Line-based synchronization scheme designed for FWMAV stereo vision. The 'transmitted' image consists of image lines from the left and right camera's. The even lines (light) are scanned first and consist of image lines from the left (blue) and right (red) camera alternately. It takes about 83μs to scan one image line. After all even lines have been scanned, the uneven lines (dark) are scanned from the left and right camera alternately.

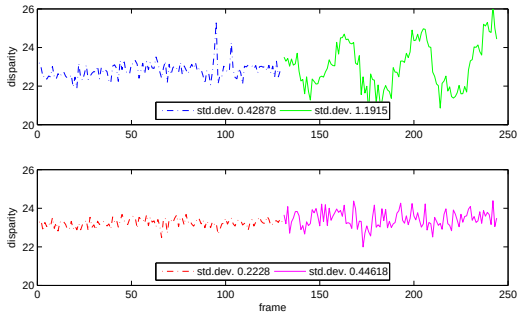

Fig. 6 Comparison between the camera reading methods. **Top** initial method **Bottom** implemented method. During the first 135 frames there is no motion (dash-dotted lines), further on there is a relative motion between the camera and the chessboard (solid lines).

first few seconds the scene was static. After a few seconds, the chessboard was slid back and forth (left-right in the camera view) to introduce motion. The disparity was then computed for both types of camera implementations to see the effect of motion on the output. From the figure it is clear that the 'initial' system (top plot) performs significantly worse as soon as the scene starts to move. From the data one can see the left-right motion of the chessboard. When the chessboard slides to the left, the images appear to move towards each other. Hence a smaller disparity is measured. When sliding in the other direction, larger disparities are measured. From the top plot it can be seen that this motion is not visible from the measurements. But it should be noted that the measurements show smaller deviations during the first seconds of the experiment when there was no motion.

In this setup the effective resolution is reduced to a quarter of the original resolution. However, this is not an issue since the images are sub-sampled to a resolution of 160x108 to perform stereo processing at 25Hz. As noted before the camera images can be subject to noise. Furthermore, in the current setup both cameras make use of the same intensity calibration parameters, which only apply to one of them. As a result there is a major difference in sensitivity to bright image features. The

cameras are also very sensitive to direct and reflected sunlight. This can blind the cameras. Also high frequency light sources can have a disturbing effect.

3 Stereo Vision methods

Computer stereo vision is the extraction of 3D information from digital images. In general this implies that images from two or more cameras are evaluated by an algorithm that tries to compute which pixels correspond to the same physical object. When this matching is done, it is known for each pixel how large it is shifted in other images. By knowing the characteristics of the cameras, these shifts (denoted as 'disparities'), can be converted to real xyz-coordinates. By using all image pixels together a 3D reconstruction of the scene can be obtained.

A considerable amount of research has been done for decades on the problem of computational stereo vision. This research is still ongoing with focuses on quality and computational efficiency. These are conflicting aspects. A concise overview of computer stereo vision methods that have been developed over the years is beyond the scope of this paper. Interested readers are referred to the Middlebury taxonomy of Scharstein and Szeliski [13] and the evaluation of Tombari et al. [14] for overview articles. For stereo vision on a flapping wing MAV, the main requirement is implementability in real time systems. Real-time performance can be obtained in two ways: by using efficient algorithms or by using special hardware implementations. In this study the focus lies on efficient algorithms. Using for example a Graphical Processing Unit(GPU), Field Programmable Gate Array (FPGA), Application-Specific Integrated Circuit (ASIC) or Digital Signal Processor (DSP) allows the use of optimized computation strategies that are very specific and have a limited applicability. Since the aim of this study is to converge to full autonomy, onboard processing is also a topic of interest. It is believed that if algorithms cannot be implemented on a CPU in real-time, they will also be no candidate for on board processing in future systems. The focus in this study is therefore further limited to methods that enable real-time performance on CPUs.

Comparison

Stereo vision algorithms can be divided in four groups depending on the optimization strategy they are based on: Winner-Takes-All, One-Dimensional Optimization, Multi-Dimensional Optimization and Global Optimization. Figure 7 shows a comparison between the these types of optimization. Global Optimization is left out of this comparison because of its computational complexity. From each of the other three types an example from the OpenCV library was taken to demonstrate the most important differences. The figure shows the result of each type of optimization method for the same image. The stereo images were sub-sampled such that

each method had real-time performance. The parameters were tuned to obtain the best result.

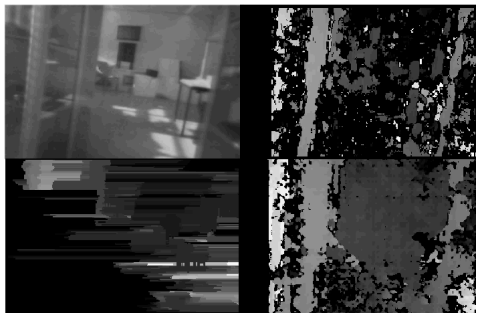

Fig. 7 Comparison of three different types of stereo vision methods. **Top-Left** test image **Top-Right** Block Matching (Winner-Takes-All) **Bottom-Left** Dynamic Programming (One-Dimensional Optimization) **Bottom-Right** Semi-Global Block Matching (Multi-Dimensional Optimization)

The Block Matching method shows a relatively sparse result. Dominant features, such as vertical lines, are matched quite well, but in between these features a lot of unknown regions are left empty (black pixels). Even the shadows on the ground apparently do not provide enough texture for good matching. The information from this method is partly useful, in that it provides information on obstacles close by. But this information would be much more useful if the method would be able to indicate that the center zone of the image contains only obstacles far away. Note that the center zone even contains blobs of white pixels that indicate non-existing close objects.

The Dynamic Programming method performs even worse. The main structures in the image can not even be distinguished. This result might not be fully representative for dynamic programming algorithms since these perform better than winner-takes-all methods in general. In the top-left corner of the image the streaking effect is visible: the image lines appear as if they are a little bit randomly shifted horizontally (typical effect of Dynamic Programming). The bottom-left part of the image is almost empty (no reliable matches) and the right part of the image does not show clear objects. This illustrates the short-coming of Dynamic Programming: matching errors influence the results for the remainder of the image lines. The bad matching results in the left part of the image spoil the results in the right part of the image. The fact that this implementation uses pixel-to-pixel matching costs might have a negative influence of the final result.

Compared to the other two methods, Semi-Global Block Matching gives significantly better results. The main structure of the scene is clearly visible in the disparity map: two cabinets close by on both sides and in between there is space with obstacles much further away. Also here some regions are left empty but the amount of known disparities is substantially larger. False matches are also visible but their number is also small. This method gives the most useful information, and is potentially useful enough for obstacle detection. The result is also notable because the method relies on simple pixel-to-pixel matching costs.

According to literature Semi-Global Matching represents a good trade-off between computational efficiency and performance [15] [16] [17].

Based on the findings from literature and the above results that support these findings it was decided to use the Semi-Global Matching [18] method for implementation in the obstacle avoidance strategies that were developed and tested in this study.

4 System Performance

The performance characteristics in terms of distance measurements accuracy are discussed in this section. These are based on static and flight tests.

4.1 Static accuracy measurements

An important performance measure for the stereo vision system is its accuracy of measuring distances to objects. To measure its actual performance without the influence of platform vibrations, a static test was done. For this test the camera was fixed at several distances (100,150,200,250,300,400,500 and 600cm) from a screen. The screen was a chess mat that was hanging vertically in the field of view of the camera. The stereo vision system was used in the same way as it is during flight. Disparity maps were computed from 1100 frames per measurement point. From each disparity map a small patch of 10x10 pixels was taken from the center of the map to compute the mean disparity. This disparity was used for calculating the distance from the camera to the screen. The results are shown in figure 8. From the results it can be observed that, at least for the static case, the stereo camera system is capable of measuring the distance to obstacles up to 500cm with a mean error of less than 50cm. For the task of obstacle avoidance this can be regarded as an acceptable performance. Obstacles that are even farther away will be detected with a lower distance accuracy. The mean error is larger than 140cm in these cases.

4.2 Accuracy Measurements during flight

The accuracy of the stereo vision system has also been measured in flight. The experiment was performed using a free-flying DelFly at a speed of approximately 60cm/s. The DelFly was flying in the direction of the chess mat. Two external cameras were used to track the position of the DelFly. Tracking was performed as follows: two video cameras were positioned such that the chess mat would be in their field of view and also the area in front of the chess mat (around 5m). The cameras were positioned on both sides of the flight path of the DelFly. By using a power-

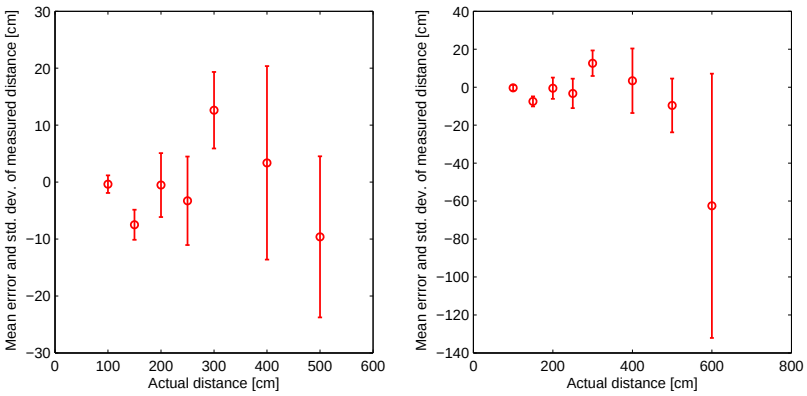

Fig. 8 Distance measurement accuracy for the static case. The left plot is a detailed version of the right plot.

ful background subtraction routine [19] and blob tracking, a special small marker positioned under the DelFly could be tracked. By using triangulation routines from OpenCV, the three-dimensional flight path (w.r.t the chess mat) of the DelFly was determined. The measurements from the onboard camera and the external cameras were synchronized by looking for specific features in the recorded videos.

Figure 9 shows the result from the first flight test. The blue points in the left plot indicate the distance between the DelFly and the mat, based on measurements from the external cameras. At small distances the blue points show some discontinuities. This is a result from the background subtraction. At small distances the DelFly flies between the cameras and the mat. The white marker on the DelFly will at some points not be noticed when it is in front of a white chess board field. The tracking routine will then find another point on the DelFly, leading to triangulation errors. These measurement errors should therefore be ignored.

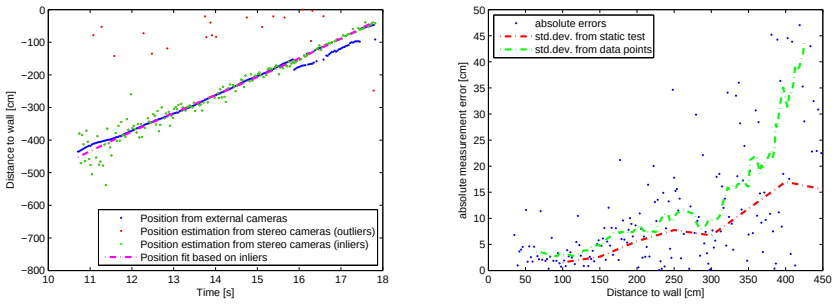

Fig. 9 Distance measurement accuracy for the flight test. The left plot shows the actual distance and estimated distance over time. The right plot shows the estimation error with reference to the actual distance.

The red and green dots are onboard distance measurements. As can be seen from the plot, most of these points are concentrated around the blue points. However, some very clear outliers (red dots) are visible. These measurements result from a hardware problem. In some cases the video frames received by the ground station are mixed-up. The order of the scan-lines is then different from the normal case and the left-right images going to the stereo processing routine contain wrong combinations: two images from the same camera, or swapped left-right images. This results in corrupt disparity maps. Another problem is a typical haze effect which results in images that are a mixture of two images.

These bad results (red dots) were left out by detecting and omitting corrupt frames. The curve fit is based on the good measurements (green dots). The right plot in the figure shows the deviation of the measurement points based on the curve fit. A running average (green dashed line) was computed based on the average error with a windows size of 21. Also the standard deviation for the static case is shown in the figure for comparison.

From the left plot it can be observed that the tracked distances and the measured distances show a very good correspondence. The main observation from the right plot is that the onboard measurements have a larger standard deviation than those obtained during the static test. For distances larger than 350 cm the error seems to grow rapidly, but this at a moment that the DelFly is still turning towards the mat.

5 Obstacle Avoidance

This section discusses the results from tests with two different obstacle avoidance strategies.

5.0.1 Direct Yaw control

The turn logic for this strategy is straightforward. From the disparity map obtained by the stereo vision algorithm it is computed how many pixels belong to obstacles that are on short range (less than 1.1m). These pixels are summed separately for the left and right halves of the image, forming so-called 'obstacle-signals'. If the left obstacle-signal reaches a threshold a turn to the right is initiated, and vice versa. If both obstacle-signals reach the threshold at the same time, a right turn is initiated. The threshold value has been chosen such that image noise and computational errors do not induce unnecessary turns. The turn is initiated by giving a predefined step input to the rudder. This rudder input is a fixed value that can be set separately for left and right turns. Its value was chosen such that the turns are steady and symmetrical (around 40 cm radius) and the turn speed is not too fast to avoid spiral motions. The turn will end only as soon as both obstacle-signals become lower than another (and smaller) threshold. As soon as the lower threshold has been reached, the rudder will go back to its trim position. However, if one of the obstacle-signals

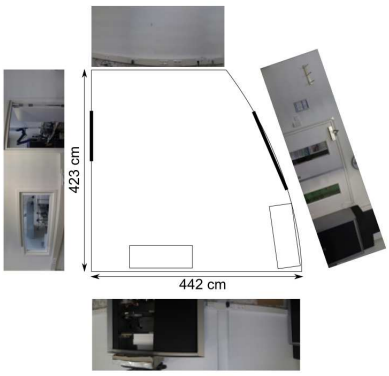

Fig. 10 Floor plan of the test room. The images around show the walls, doors and cabinets in the room

reaches the higher threshold again within a predefined safety-time, the DelFly will continue its previous turn, regardless of which of the two 'obstacle'-signals reached the threshold. This will prevent the DelFly to turn back into the direction it just turned away from, since this is most likely not a safe maneuver.

The experiment was conducted in a room of ∼4.23x4.42m. Figure 10 shows a floor plan of the room. The images on the sides give a good impression of its appearance. Except for the walls the main obstacles are two black cabinets. The door on the left was closed during the experiments, and part of the window on the left was covered to prevent window collisions. It should be noted that the images in the figure only show a part of the scene (mainly the top part) while the onboard cameras of the DelFly could see more of the lower parts of the room. The lights were most of the time switched off during the experiments since they resulted in a flickering effect in the stereo cameras. During the experiments the 'obstacle'-signals were logged, as well as turn events. Furthermore, an onboard image was captured at the moment a turn event (left/right turn or end of turn) occurred. The elevator was given a constant input such that the speed would be around 0.6m/s during the test.

This experiment was repeated several times and resulted in various observations. As a general result it can be stated that the obstacle detection performed well. The obstacle avoidance strategy showed some expected flaws. This will be illustrated by data recorded during one of the flights.

Figure 11 shows a situation during the first seconds of one of the test flights. The sketch on the right indicates the position of the DelFly at the start of the flight and during the first turn. During the first seconds, it flies close to the wall. But, as can be seen in the left onboard image, the wall on the right is outside the field of view. The left bottom plot shows the 'obstacle'-signals during the last seconds before the turn. In this case these values are initially zero because the obstacle detection was not activated yet. From the plot it can be seen that the cabinet, (mainly) on the left side in the image, lets the left obstacle signal increase faster than the right signal, as expected. When the left signal exceeds the threshold (in this experiment set at 200), a turn to the right is initiated successfully. As a result, the DelFly turned into the direction of the wall. It was prevented from colliding manually.

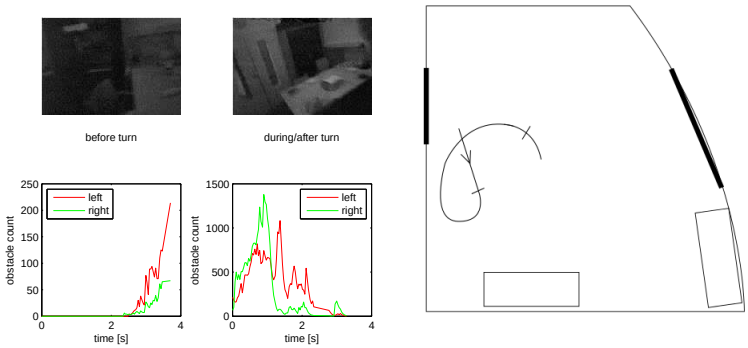

Fig. 11 Example of a turn decision. The first image (top left) is an onboard image from the moment the turn decision threshold (200) was reached. The corresponding obstacle-signals (up to turn decision) are shown in the bottom left image. The other onboard image (top middle) was taken at the moment the lower threshold (50) was reached. The corresponding obstacle-signals (from turn initiation until end of turn are shown in the bottom middle image. The figure on the right shows the flight path during the turn. For dimensions, see figure 11.

The middle bottom plot shows the amount of obstacles detected during the turn. It can be seen that during the first half second of the turn, the amount of right 'obstacles' increases first. This is because the wall now enters the field of view. After one second the DelFly has turned around and the right obstacle signal decreases. Since the wall is now in the left side of the view, the left obstacle signal is now very high. While turning away from the wall, the left obstacle signal decreases. It can be observed that it takes fairly long before a safe flight direction was found during the turn. First it takes two seconds before the left obstacle signal decreases below the threshold (set at 50). If this threshold would have been the same (also 200) the turn would have been ended approximately one second earlier. However, earlier experiments showed that for a threshold value of 200, turns would very frequently be ended too early (and then continued immediately, but with some delay). Also note that at the end of the turn, the left obstacle signal decreases below 50, but at the same time the right signal increases again. From the right onboard image in figure 11 it can be observed that the DelFly rolls while making a turn. The table in the image appears to be shifted up in the right side of the image. Apparently it is then detected as an obstacle.

In the sketch on the right it is indicated at which points the turn was initiated and ended. The end point corresponds to the location where the right onboard image (see figure) was taken. In the sketch it is indicated that after this moment the DelFly continued its turn a bit longer. This is a result from the delay between the ground station and the DelFly.

Figure 12 shows how the DelFly continued after the first turn. It is flying into the direction of the same cabinet as before, but now it is in the right side of the camera field of view (top left image). The left bottom plot indicates that indeed an obstacle

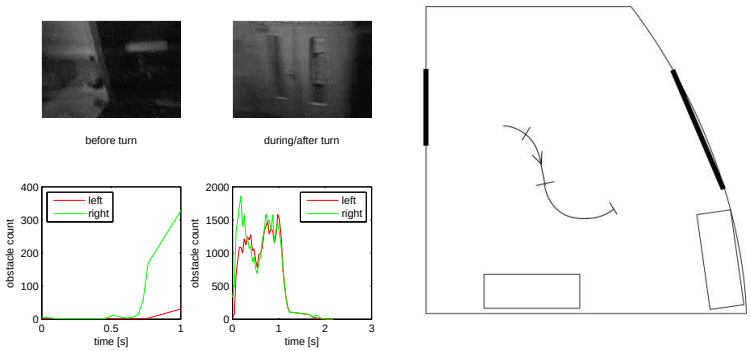

Fig. 12 Example of a turn decision. The first image (top left) is an onboard image from the moment the turn decision threshold (200) was reached. The corresponding obstacle-signals (up to turn decision) are shown in the bottom left image. The other onboard image (top middle) was taken at the moment the lower threshold (50) was reached. The corresponding obstacle-signals (from turn initiation until end of turn are shown in the bottom middle image. The figure on the right shows the flight path during the turn. For dimensions, see figure 11.

is detected on the right side. During the first 0.2s after turn initiation (middle bottom plot), the 'obstacle'-signals increase quickly since the DelFly approaches the cabinet. Then, after some delay, the turn command is received onboard and the DelFly starts to turn to the left. Note that around one second later, both signals drop quickly. However, it takes another second before the signals drop below the threshold value. Apparently this is caused by the other cabinet in the corner. It should be noted that during this turn significant noise occurred. As a result, no obstacle detection was performed between 1.23s and 1.66s after turn initiation. This is also the case in the left bottom plot. In that case there are no measurements between 0.77s and 1.0s after the previous turn.

These examples show that the DelFly successfully detects obstacles in its field of view at sufficient range to perform obstacle avoidance. Also during the turns the obstacle detection provides reliable information which makes it possible to decide at which point the turn can be ended safely.

Situations as described in the first example can occur because of the direct nature of the turn strategy in combination with the limited field-of-view of the stereo cameras. During some of the experiments these situations occurred rarely and the DelFly could fly autonomously for longer than 1 minute.

An important observation during the tests is the endurance of the DelFly in its current configuration. As discussed earlier, almost full throttle needs to be applied right from the start of the flight. Within one minute, full throttle is required. Within 2-3 minutes the batteries cannot deliver sufficient power to keep the DelFly at a constant height anymore.

5.0.2 Direct Yaw and Pitch control

The second experiment is a follow-up of the first one, and it was done the same way. The only difference is the addition of a simple pitch control rule. During unobstructed flight, the elevator is in its fixed position such that the DelFly will fly at a speed of around 0.6m/s. As soon as an obstacle needs to be avoided, a turn is initiated the same way as in the first experiment. At the same time the elevator input is changed such that the DelFly will loose its speed and start to hover. As a result the DelFly will change its heading (by yawing) while it keeps its position. Obstacles can be avoided without the risk of making a turn and colliding with another object out of the camera field of view.

Before this test was conducted, it was already known that the DelFly in its current configuration is too heavy for hovering. It will definitely loose height at the turning points. However, the experiment can be useful in demonstrating that this simple avoidance strategy is suitable for an (FW)MAV as long as it is able to hover. Future designs of the DelFly might be able to hover more efficiently and could use this strategy for maneuvering in small spaces.

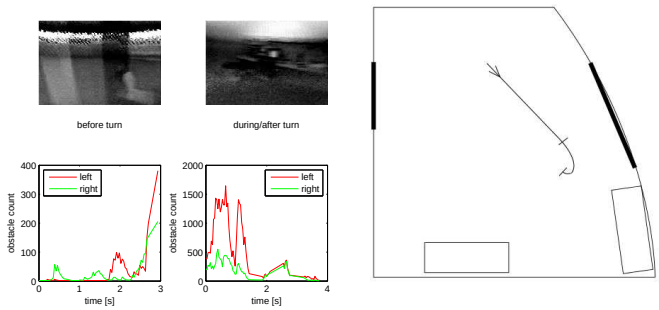

Fig. 13 Example of a turn decision. The first image (top left) is an onboard image from the moment the turn decision threshold (200) was reached. The corresponding obstacle-signals (up to turn decision) are shown in the bottom left image. The other onboard image (top middle) was taken at the moment the lower threshold (50) was reached. The corresponding obstacle-signals (from turn initiation until end of turn are shown in the bottom middle image. The figure on the right shows the flight path during the turn. For dimensions, see figure 11.

As explained this second avoidance strategy is meant to demonstrate the benefit of making turns without forward speed. An example situation is shown in figure 6. The DelFly approaches the cabinet and at some point a turn is initiated. From the bottom middle plot it can be seen that initially the amount of left detected obstacles increases because of control delay and initial forward speed. The DelFly turns to the right and the obstacle-signals decrease. From the right onboard image (top middle image) it was observed that the DelFly has lost some height and is now flying slightly above table height. As discussed earlier, this is an expected (but unwanted) result due to the bad hover performance of this specific configuration of the DelFly.

Furthermore this results in the additional problem that there are now other obstacles detected, such as those that are on the tables. Also partly because of noise (and bad obstacle detection measurements for that reason) the turn is ended at a point much further than one would expect.

It was observed that with this strategy obstacle detection and avoidance could be performed successfully without the problem of colliding with out-of-sight obstacles. This demonstrates the advantage of stereo vision over optic flow measurements.

5.0.3 Look-Ahead Yaw control

The previous experiments demonstrated that the DelFly is not able to hover. Furthermore it was demonstrated that responding directly to obstacles in the field-of-view will result in collisions with obstacles that are outside the field-of-view. Therefore a third turn strategy is discussed here.

In this new strategy the DelFly continuously flies with a constant speed (fixed elevator setting). A turn is initiated when too many obstacles (pixels with a large disparity value) are detected in the safety region. The safety region is defined such that it covers an area large enough for the DelFly to turn around 360 degrees. Because of the limited field-of-view of the camera, this turn area will lie ahead of the current position of the DelFly. Figure 14 shows this safety region. The region is defined in the camera reference frame, with the x-direction positive to the right, the y-direction positive up and the z-direction positive in the direction of flight. Starting at the origin (position of the camera), two oblique lines define the camera-field of view. The dashed line is the trajectory the DelFly will follow as soon as too many obstacles are detected. After 225cm a right turn will be initiated. During the turn the same safety region is used to detect a new safe flight direction. As soon as it is found, the turn will be terminated. Because a turn might be terminated by mistake or an overshoot can occur due to delays, the turn will be continued immediately if the new direction of flight is not regarded safe anymore. This is only possible within a fraction of a second after turn termination. This type of turn recovery has been taken into account in the safety region definition. This is why the outer circle has been drawn. Around this outer circle extra safety margin has been included to accommodate for the width of the DelFly and inaccuracies in range estimations. Note that the turn area has been centered in the image in order to minimize the size of the safety region. As a result, the flight trajectory towards the turn area is drawn as a slanted line. For this reason the stereo vision cameras were mounted on the DelFly with an offset angle to align the drawn flight trajectory with the flight direction of the DelFly. In other words, the cameras are pointed a little bit to the right.

In this strategy only rudder commands are used. Because the obstacle measurements are sensitive to noise, filtering is required to increase robustness. For this reason a logical diagram was developed as shown in figure 15 (left) which decides upon rudder inputs. In this logic each turn is divided in phases. During Phase 1 the DelFly flies straight with 1m/s (faster than during earlier experiments to increase flight endurance). A threshold is used for the turn decision based on the amount of

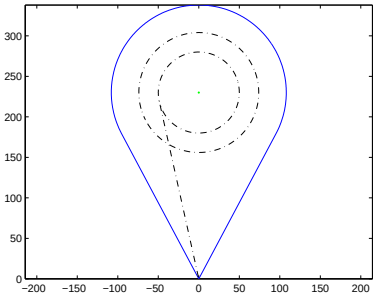

Fig. 14 Turn strategy using continuous turns. The dashed line is the DelFly flight trajectory. The area between the blue line is the obstacle-free region.

pixels that exceed the disparity constraint defined by the safety region. To suppress noisy measurements, filtering is applied as follows. Each time the threshold is exceeded (250 pixels), the current time is stored. It can then be checked if the threshold is exceeded ten times within one second. If this is the case, it is concluded that there is an obstacle. The earliest detection time of these ten detections is then used as reference time for the second phase. In Phase 2, the DelFly still flies straight and waits until it has reached the point in figure 14 where the turn needs to be started. This time to turn is just over 2 seconds. However, because the turn response of the DelFly to rudder inputs is sluggish initially, and because of communication delays, the time to turn was tuned experimentally and set a a value of 1500ms. After this time has elapsed the turn is initiated in Phase 3. During the turn it is checked if the current direction of flight is obstacle free. As soon as a lower threshold of 200 pixels is reached, Phase 4 starts. No filtering is used here because this will result in unwanted delay. The turn speed of the DelFly is around 1 rad/s and small delays result in large flight direction differences. To compensate for the quick decision making and turn overshoots, it is checked in Phase 4 if the new flight direction is indeed a safe direction to fly. If within one second after turn termination the obstacle threshold of 250 pixels is exceeded again, the turn is resumed in Phase 3. Otherwise the new flight direction is regarded as safe and Phase 1 starts again. In Phase 1 also another check is performed to detect obstacles at short range. The avoidance region defined as in the first experiment is used here. A threshold of 500 pixels is used. If it is exceeded three times in row, Phase 3 is activated to start turning immediately. The main reason for including this rule is the sensitivity of the DelFly to wind disturbances which changes the flight trajectory to such an extent that obstacles initially out of the field-of-view will result in collisions. This rule is used to prevent unexpected collisions but does not guarantee that the DelFly will be able to continue safely. Tests with this turn strategy were performed in a larger test room because of the size of the safety region. In the test room from the first experiment the DelFly would keep turning continuously. The test room is visualized in figure 15 (right).

Figure 16 shows the result from the test with the best result that has been obtained. During this test the DelFly flew around for 72.6s without hitting any object. The experimenter did not have to pull the DelFly up to keep it at a constant height.

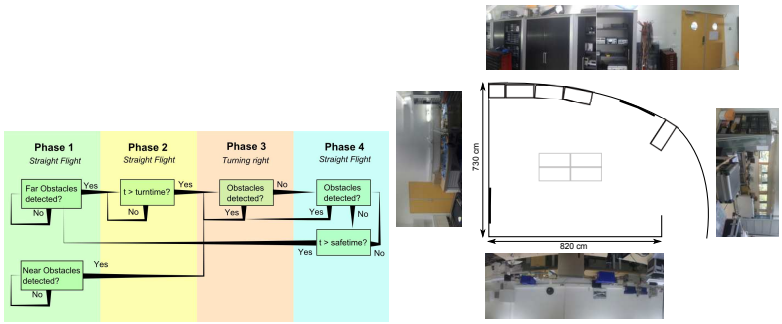

Fig. 15 Left: Flight phase diagram for rudder control. Right: Floor plan of the test room.

It should be noted that the experiment was ended without reason. The DelFly was still performing autonomous flight and the total successful test time could have been longer than the reported length.

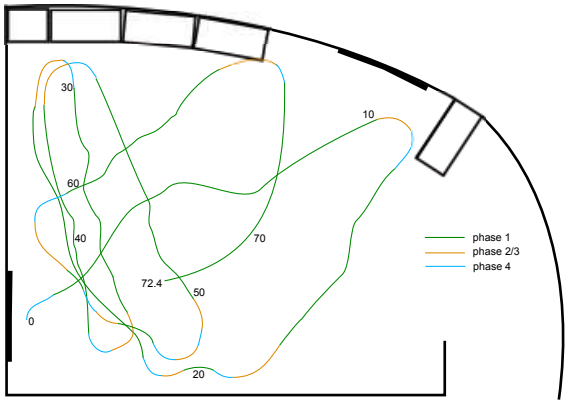

Fig. 16 Flight track of the DelFly during the experiment. The numbers indicate the flight time, the colors represent the flight phases.

The experiment starts at a point where the DelFly is coming out of a turn that was performed early after start up. This is point t=0 in figure 16. The track colors indicate the flight phases the controller is in according to figure 15 (left). During start of the experiment Phase 4 is active where the DelFly is ending its turn. During the next ten seconds Phase 1 is active and the DelFly should perform straight flight. From the flight track it can be observed that the flight is far from straight. Due to non-zero wind speeds in the test room, caused by ventilation and air conditioning systems, the DelFly swerves significantly. At this time this does not result in avoidance problems. When the DelFly approaches the upper wall after 10 seconds, Phase 2 and 3 are activated. These phases are combined in the figure. The flight track during the subsequent Phase 4 goes right past the cabinet. However, at this point the DelFly was flying above cabinet height and for that reason it was not considered to be an

obstacle. Only the wall needed to be avoided at this point. The flight is then continued in the direction of the lower wall. Note that in all cases where Phase 4 is active, the turn continues. In many cases the new flight direction is around 90 degrees further to the right. This is a result from all system delays, including video reception delay, processing time and radio control delay. After 20 seconds in the experiment Phase 1 is active again. Note that the flight track unexpectedly deflects to the left. As a result a new turn is triggered which directs the DelFly back to the upper wall. This sudden left turn can be explained by yaw/roll instability and is unpredictable. The cabinets in the top then force the DelFly to go back (at around 30 seconds). Note that back at the bottom wall the DelFly preserves a larger distance to the wall compared to other turns. Again the DelFly goes to the top cabinets and back. Just after 50s the lower wall is approached again. In this case an early turn is initiated which is ended too early. As a result Phase 4 is activated while the DelFly still continuous in the direction of the wall. Because the wall is detected again Phase 3 is active again after 684ms. The turn is then continued till the flight direction is now in the direction of the cabinets again. Again the DelFly unexpectedly turns quickly to the left and flies in the direction of the doors on the left. These are detected early and a slight turn follows immediately. Another turn is then initiated 1517ms later to avoid the left wall. At t=60 the DelFly is performing a straight flight in the direction of the top wall at a height above the cabinets. The wall behind the cabinets is then detected and avoided successfully. The flight ends after 72.6s without colliding with any obstacle. Note that during the last part of the flight the DelFly gradually but severely makes a turn to the right. Near obstacles (see bottom of Phase 1 in figure 15 (left) were never detected. This means that the DelFly never tried to avoid obstacles that were detected late.

6 Conclusions

From the results presented in this paper it can be concluded that stereo vision can be applied successfully for obstacle detection and avoidance on FWMAVs. It was shown that real-time stereo vision can provide accurate and sufficient obstacle information. By making use of suitable camera hardware the flapping motion of FWMAVs has a minor influence on the stereo vision algorithm. In this respect this method outperforms optic flow techniques.

The small camera system is capable of giving distance estimates with a standard deviation of 20cm up to 5m. Even for texture-poor areas the accuracy is still adequate. The weight of the camera system and extra required battery leads to a reduced flight endurance and a reduced flight envelope i.e. hovering is not possible.

Closed-loop experiments showed that stereo vision can provide robust and reliable obstacle information that allows the DelFly to perform successful obstacle avoidance. An autonomous flight time of 72.6 seconds has been obtained as the best result.

One of the focuses of future research will be on the camera design. Lighter cameras with a wider field of view should result in better performance. Another important focus will be on onboard image processing. This will eliminate communication delays and the need for a ground station within communication range.

References

1. Griffiths S, Saunders J, et al. (2007) Obstacle and Terrain Avoidance for Miniature Aerial Vehicles. In: Advances in unmanned Aerial Vehicles, doi: 10.1007/978-1-4020-6114-1_7
2. Kownacki C (2009) Guidance and obstacle avoidance of MAV in uncertain urban environment. http://www.emav2009.org/EMAV_final_papers/EMAV2009_papers.html. Cited 1 Oct 2012
3. Hines L, Arabagi V, Sitti M (2011) Free flight simulations and pitch and roll control experiments of a sub-gram flapping-flight micro aerial vehicle. In: IEEE ICRA, 1-7.
4. Lin S, Hsiao F et al. (2009) Altitude control of flapping-wing mav using vision-based navigation. In: IEEE ICRA, 3644 - 3650.
5. Duhamel PE, Perez-Arancibia N et al. (2012) Altitude feedback control of a flapping-wing microrobot using an on-board biologically inspired optical flow sensor. In: IEEE ICRA, 4228-4235.
6. Beak S, Fearing R (2010) Flight forces and altitude regulation of 12 gram i-bird. In: IEEE RAS and EMBS Int Conf on Biomedical Robotics and Biomechatronics (BioRob), 454-460.
7. Beak S, Garcia Bermudez F, Fearing R (2011) Flight control for target seeking by 13 gram ornithopter. In: IEEE/RSJ Int Conf on Intelligent Robots and Systems.
8. Garcia Bermudez F, Fearing R (2009). Optical flow on a flapping wing robot (Tech. Rep.). http://www.eecs.berkeley.edu/∼fgb/pubs.shtml. Cited 1 Oct 2012
9. Tedrake R, Jackowski Z et al. (2006) Learning to fly like a bird (Tech. Rep.). http://groups.csail.mit.edu/robotics-center/public_papers/Tedrake09.pdf. Cited 1 Oct 2012
10. de Croon G, de Clerq K, et al. (2009) Design, aerodynamics, and vision-based control of the Delfly. In: the Int J on MAVs, Volume 1, Number 2, 71-97.
11. de Croon G, de Weerdt E et al. (2012) The appearance variation cue for obstacle avoidance. In: IEEE Transactions on Robotics, Volume 28, Issue 2, 529-534.
12. de Croon G, Groen M et al. (2012). Design, aerodynamics, and autonomy of the Delfly. In: Bioinspiration and Biomimetics, Volume 7, Issue 2.
13. Scharstein D, Szeliski R (2002). A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. In: International Journal for Computer Vision, 47 (1- 3), 7-42.
14. Tombari F, Mattoccia S, Di Stefano L (2008). Classification and evaluation of cost aggregation methods for stereo correspondence. In: IEEE Conference on Computer Vision and Pattern Recognition, 1-8.
15. Christian B, Hesselbarth S, Flatt H, Blume H, Pirsch P. (2010) Real-Time Stereo Vision System using Semi-Global Matching Disparity Estimation: Architecture and FPGA-Implementation. In: International Conference on Embedded Computer Systems, 93-101.
16. Gehrig S K, Everli F, Meyer T (2010) Real-Time Semi-Global Matching on the CPU. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 85-92.
17. Chen B, Chen H. (2011) A realization of semi-global matching stereo algorithm on GPU for real-time application. MIPPR 2011: Pattern Recognition and Computer Vision
18. Hirschm*ü*ller H (2008) Stereo processing by Semi-Global matching and mutual information. In: IEEE Trans. Pattern Anal. Machine Intell., 30 no. 2, 328-341.
19. Barnich O, van Droogenbroeck M (2011). Vibe: A universal background subtraction algorithm for video sequences. In: IEEE Transactions on Image Processing, 20, Issue: 6, 1709 - 1724.