Development and Analysis of a Floor-Based Obstacle Detection Method for Autonomous Micro Air Vehicles

Group 2
Abdelhakim Amer, Prawien Kanhai, Luc Kloosterman, Sebastian Nicolay*, Jean-Luc Overkamp, Barry van Leeuwen
Technical University of Delft, Delft

Abstract

Obstacle avoidance is a challenging task for autonomous MAVs in cluttered environments. In this paper, a new vision algorithm is developed which is able to run safely on the Parrot bebop drone inside the Cyberzoo environment. Different strategies are investigated from which it can be concluded that the application of a color filter for floor segmentation is the most optimal approach in the given environment. The developed algorithm distinguishes the green floor of the Cyberzoo from the obstacles and the area outside it with an on-board runtime smaller than 0.04 seconds on the Parrot Bebop drone. Objects from which the size exceeds a certain threshold as seen by the drone's vision will be avoided. For this it was found that stopping and then searching for a safe heading is the most promising approach. In order to optimize the performance of the drone, multiple tests are conducted. This includes tuning of the obstacle threshold, Cyberzoo border threshold, Cyberzoo border detection robustness and the speed parameters. Finally, a maximum speed and heading rate of 1.5 m/s and 60 deg/s, respectively, is found to give the most optimal performance.

1 Introduction

During the Autonomous Flight of Micro Air Vehicles course, students develop an efficient, purely vision-based approach to autonomously navigate through obstacles in a controlled environment, which is also known as the Cyberzoo. In the scope of this work, a floor-based vision algorithm was developed which allows the Parrot Bebop drone to successfully fly trough the Cyberzoo while avoiding obstacles. First, a short literature study on several relevant monocular based detection and avoidance methods are presented. First, a short literature study on several relevant monocular based detection and avoidance methods is presented. Eventually, the algorithm's performance is tested and analyzed.

2 Literature study

This short literature study will give an overview of some existing monocular vision based obstacle detection and avoidance methods that are potentially viable onboard the Parrot Bebop inside the Cyberzoo environment. The explored methods are: Optical flow using feature extraction, Hough line detection using Canny edges, and ground segmentation.

Optical flow is a popular method for obstacle avoidance that is inspired from insects. Takeda [1] used optical flow vectors of image points around the Focus of Expansion to estimate the time to collision with an obstacle. Another approach adopted by Souhila [2] used the difference between the sum of magnitudes of the optical flow vectors on the left side of the image versus the right side to determine the direction from which the obstacle is approaching. De Croon [3] combined optical flow with appearance variation cues for obstacle detection. Another approach by Sagar et al. [4] classified objects in the image as foreground and background, and then used optical flow on the features in the foreground. Limitations of optical flow include its dependence on texture as well as the aperture problem. When tested with sample images from the Cyberzoo, using different feature detection algorithms, these limitations showed that optical flow is unreliable in detecting common obstacles, such as orange poles.

Another obstacle detection method is to extract them based on the surfaces and shapes of features. The discrete features in the Cyberzoo make it a good environment to use an edge and/or corner detection method to extract features of obstacles. This method does not track features but extracts them per frame in order to identify an object based on certain predefined thresholds. Two commonly used methods are the Canny edge detector and the Harris corner detector. Harris [5] mentions that the downside of a pure Canny edge detector is the difficult detection of curvy edges and textures of varying scales. Harris corner detectors combine the edge detection with the extraction of feature-points or corners. This theoretically allows the corner detection to better distinguish textured surfaces as well as identify objects and irregular shapes. However, the Cyberzoo does not have many curved edges nor highly textured patches. It was confirmed by simple testing that the Canny edge detection is much better at extracting features from the Cyberzoo. Using the Canny edge detector, a lot of noise and ground-texture is extracted. To extract obstacles from the detected features/edges, a probabilistic Hough line detector was implemented. In order to filter out objects far away, the image was eroded. This caused 'thin' objects (far away) to not be extracted and thicker objects to remain as a feature. The Hough line detector was shown to be promising but computationally too heavy and was thus aborted.

*Email address: s.a.nicolay@student.tudelft.nl

The third approach is inspired by a combination of sky segmentation [6] and appearance-based obstacle detection [7]. Ulrich et al. [7] uses the nearby ground as a reference and compares all pixels to that reference. Obstacles are then identified by difference in appearance from the ground. This approach works for flat ground and obstacles with a different appearance than the ground, but not for overhanging obstacles. These three assumptions are true for the Cyberzoo where the green ground can be used as a reference to compare obstacles in the field of view. Ulrich et al. [7] mentions that the strategy is most promising when combined with some method for distance measure. The next challenge is to classify an object as an obstacle based on either the distance to the subject, the location, or the shape. Aguilar et al. [6] shows that treating protrusions from the horizon as an obstacle is a suitable and low cost method to obstacle avoidance. The Cyberzoo deals with a very different environment, in which the 'horizon' is actually a box. However, the principle can be applied. The orange pillars extend through the horizon and cause a non-continuity of the horizon. This can be used in combination with appearance-based obstacle detection to identify the obstacles below the horizon line.

The choice for the avoidance manoeuvre was to either move around the obstacle or yaw and evade the obstacle. However, due to the small size of the Cyberzoo, strictly moving along an avoidance path may lead the drone into small corners, wasting valuable time. Therefore, it was concluded that the most optimal navigation approach to obstacle avoidance was to stop and search for a safe heading.

The insights gained from the short literature study provide a basis from which a feasible and optimal object detection and avoidance algorithm was designed.

3 Method and Implementation

The implementation of the obstacle avoidance method was distributed over two modules. The first module contains the vision algorithm which processes the current frame from the on-board camera and provides a simple navigation command. The second module uses this command to set up a confidence level based on which an appropriate speed and heading rate are determined.

3.1 Vision module

In the scope of the present work, a new vision algorithm was developed based on the main requirements of being robust and efficient. The basic idea of the algorithm is to make use of the green Cyberzoo floor that stands out in color with respect to potential obstacles as well as to the Cyberzoo borders. The vision algorithm consists of three main units including a color filter, an analyzer and an interpreter. The analyzer processes the outcome of the filtered image whereas the interpreter translates the results from the analyzer into a command that is subsequently sent to the navigation module.

The green color range of the Cyberzoo floor is defined by its minimum and maximum values in the YUV color space. The ranges of the values are chosen large enough to cover various lighting conditions. These values are used by the color filter which is part of a repeatedly called vision algorithm. The color filter consists of two nested loops. The first loop goes through the horizontal pixel axis of the current frame while the second loop goes through the vertical pixel axis. A scale factor of a power of two can be used to skip pixels and thus reduce computational effort. Only the bottom half of the image is processed, since the floor rarely exceeds half of the image height. As the image received exists in a UYVY pixel color format, it is first transformed into a YUV color format. Subsequently, the values of the individual pixels are compared to the predefined color value ranges.

The outcome is stored in a one-dimensional array, here referred to as green array. Every entry of the green array represents one pixel column of the image. If at least one pixel within a pixel column is within the predefined green color range, a 1 is stored in the green array. Otherwise, a 0 is stored. By ignoring the vertical location of the detected green pixels, mats on the ground are not seen as an obstacle as long as a green area either in front of or behind the mat is visible.

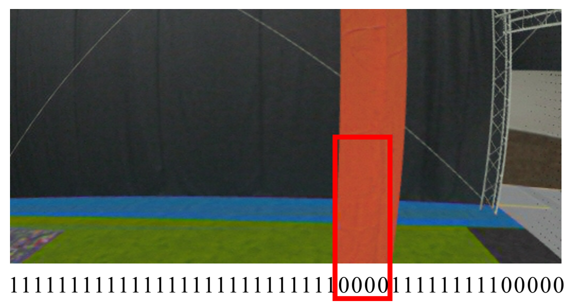

Figure 1: Non-green pixel columns within the green area are considered an obstacle. If the obstacle width exceeds a threshold, the drone is commanded to turn.

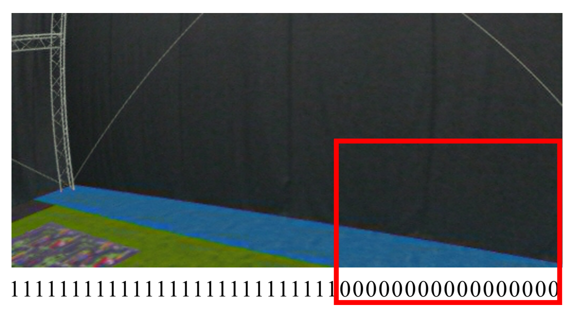

Figure 2: Non-green pixel columns outside the green area are considered a Cyberzoo border. If the border width exceeds a threshold, the drone is commanded to turn.

The analyzer extracts the information from the green array that is necessary to detect and avoid obstacles and borders of the Cyberzoo. At the beginning, a filter is applied that removes single and double green entries in the green array, as they can be considered background noise. Then, the minimum and the maximum horizontal indices of the green area are determined which give information about the horizontal spatial extension of the green floor. Using this information, the algorithm is able to determine the following two cases:

1. Any non-green pixel column within the green area is considered an obstacle.
2. Any non-green pixel column outside the green area is considered out of bounds.

This principle is further illustrated in Figures 1 and 2 where the numbers below the images represent the green array.

Subsequently, the algorithm determines the horizontal centroid of both the obstacle and the green area which will later be used to decide on the turning direction. Thereby, the green centroid is stored as a global variable and is only recalculated if the total amount of green pixels is at least 1000. In this way, the algorithm makes sure that the calculated centroid of the green area is reasonable and can not be manipulated due to image distortion. The centroid of the green area will later be used to decide on the turning direction when approaching a border. Completing the analysis part of the vision algorithm, the width of the obstacle is divided by the width of the image in terms of pixels. This ratio will later be compared with a predefined threshold.

Eventually, the interpreter unit of the vision algorithm translates the previously gained information into a navigation command. This command is an integer which can have three different values. A 0 means that no close obstacle or border was detected so the drone can move straight ahead. A 1 and a 2 mean that the drone shall turn right or left, respectively, regardless the cause for the maneuver. The consecutively checked conditions and the according generated commands are listed in Table 1. The obtained command is eventually sent to the navigation module via the ABI messenger system which is part of the paparazzi environment. The total on-board runtime of the vision algorithm on a Parrot Bebop is less than 0.04 s.

Finally, the algorithm was extended to also detect green obstacles which are similar to the color of the floor. This is accomplished by counting the total amount of green pixels in the upper area of the image and comparing them with a predefined threshold.

3.2 Navigation module

The navigation algorithm is based on the existing Orange Avoider Guided algorithm. A flow diagram of the model is shown in Appendix A. The initialization of this module first chooses a random heading and then establishes contact with the vision module. The periodic function receives a navigation command from the vision algorithm at 10 Hz. A confidence level, which increases or decreases when the navigation command is zero or non-zero, respectively, is introduced to control the forward velocity. Thus, if the vision module detects an obstacle, the drone will decelerate based on the decreasing confidence level. If the confidence level becomes zero, the drone will stop and search for a new safe heading. If the navigation command is zero for a longer period the confidence level will increase to its maximum value of 5 which enables the drone to fly at maximum speed.

Table 1: Conditions and navigation commands overview

	Condition	**Comm.**
1.	Maximum green index is at less than 70% of image width → border on the right	left
2.	Minimum green index is at more than 30% of image width → border on the left	right
3.	Total of green pixels is less than 100 → out of bounds → check green area centroid	left/ right
4.	Obstacle threshold is not fulfilled	straight
5.	Obstacle centroid is right of image center	left
6.	Obstacle centroid is left of image center	right
7.	None of the previous applies	right

4 Tests and Analysis

Several tests were conducted with the aim to optimize the drone's performance. Thereby, various parameters were adjusted in order to obtain a maximum flight distance within a given time period. The main tests worth mentioning at this point include two tests regarding the vision algorithm as well as a final, extensive test and analysis of the achieved flight distance including adjustments of the drone's speed and heading rate. Eventually, a Receiver Operating Characteristic (ROC) for the utilized color filter is presented.

4.1 Obstacle threshold tuning

Objective: The optimum obstacle width threshold shall be determined. A too high threshold may cause the drone to detect the obstacle too late. However, a too low threshold may cause the drone to do unnecessary turns.
Task: Change the threshold during the simulation via the settings in the Paparazzi Ground Control Station (GCS). Use different obstacles and observe the drone's behaviour. Measure total covered distance over given time period.
Observation: An obstacle width threshold of more than 12 % of the image width causes the drone to stop too late. A threshold of 8 % or less leads to unnecessary turns.
Result: An obstacle width threshold of 10 % was found to be a reasonable trade-off between safety and performance, and led to the largest covered distance within a given time period.
Iterations: Initially, this test led to a threshold reduction from 20 % to 15 %. At a later stage, the test was redone, finding that a 10 % threshold is more reasonable.

4.2 Cyberzoo border threshold tuning and robustness

Objective: The optimum threshold of the image share that is covered by the Cyberzoo border shall be determined. This threshold serves as an indicator for a close border based on which the drone decides to turn (cp. Figure 2). If the threshold is set too large, the drone may detect the border too

late and fly out of bounds. However, if it is too low, the drone will turn very early and not exploit the entire Cyberzoo.
Task: Change the border threshold during the simulation via the settings in the GCS. Let the drone approach all Cyberzoo borders from several angles.
Observation: Setting the threshold to 25 % led to early turns of the drone. Furthermore it was observed that obstacles that exceed the left or right edge of the image are treated as a border and not as obstacles. A relatively high threshold of 40 % thus caused the drone to crash into obstacles when they covered e.g. the left 35 % of the image. Moreover, when approaching the borders with transparent nets, the drone sometimes detected false obstacles due to background noise. The amount of those falsely counted green pixels in the background could be shown to be always less than 100.
Result: A border threshold of 30 % was found to be a reasonable trade-off between a good exploitation of the Cyberzoo and a low chance of crashing into obstacles. An additional condition was added that makes the drone turn as soon as the total amount of green pixels falls below 100. Thereby, the turning direction is based on the latest centroid of the green floor (cp. section 3). As a result, all borders are detected reliably and the most efficient turning direction is chosen.
Iterations: The knowledge gained from this test led to the objective to never get too close to any obstacle in order to avoid the above mentioned issue of overlooking obstacles that exceed the left or right edge of the image.

4.3 Performance tests with a variation of speed parameters

Objective: Maximize the covered distance by altering the max. speed (m/s) and heading rate (deg/s).
Task: Simulate the flight for two simulation minutes for various parameter settings during the whole range of max. speed (0.5 to 6 m/s) and heading rate (20 to 120 deg/s).
Observation: High speed and high heading rates both show unstable flight performance which could lead to more crashes. The region between 1.1 and 1.5 m/s is most promising.
Result: The region between 1.1 and 1.5 m/s is the most promising and is shown in Figure 3. Still, the drone crashes during some of the simulations in this region. Nevertheless, these measurements are still valid since crashes are detrimental for high covered distance.
Iterations: The first iteration showed which region of parameter values is promising. The second iteration consisted of a deeper analysis of the promising region. This led to a max. speed and heading rate of 1.3 m/s and 60 deg/s, respectively.

4.4 Color filter ROC

As a measure to assess the quality of the developed vision algorithm, a ROC was generated. Based on five images from the drone's on-board camera, that represent common situations during the flight in the Cyberzoo, the floor detection ability of the green color filter described in section 3 was evaluated. The results in terms of true and false positively detected green pixel columns is depicted in Figure 4. It

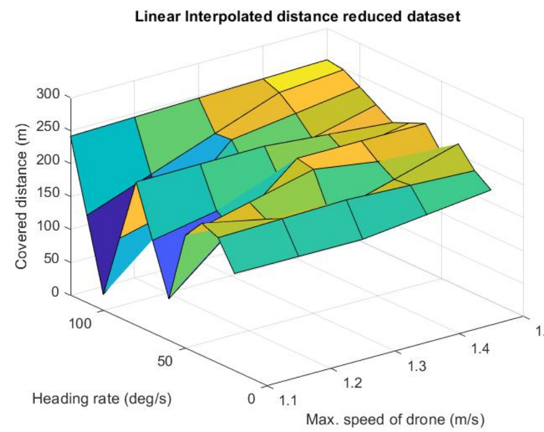

Figure 3: Reduced dataset of simulation results

was shown that very high true positive ratios can be achieved while the false positive ratio is minimal. Thus, the floor detection part of the vision algorithm works with great reliability.

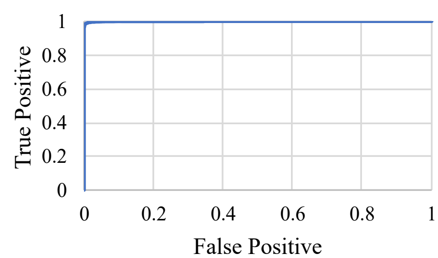

Figure 4: ROC of the utilized green color filter

5 Conclusion

This paper describes the development of an algorithm which allows drones to successfully fly autonomously through the Cyberzoo while avoiding obstacles. First of all, a literature study is conducted regarding different relevant algorithms. In this section, it is concluded that an appearance-based obstacle detection is the most suitable. For obstacle avoidance it was found that the most optimal approach is to stop and search for a safe heading. In the second section, the theory behind the new vision algorithm which is developed is described. It uses the green color of the Cyberzoo floor to detect obstacles. In the last section, the performance of the algorithm is analysed, using three different test setups. The outcome of these tests was used to optimise the parameters of the algorithm. Setting various thresholds and a maximum operating velocity of 1.3 m/s and 60 deg/s, respectively, the most optimal performance was found. Limitations of the algorithm include the requirement of floors consisting of similar color values such as the Cyberzoo. Furthermore, it does not detect objects which are very thin, e.g. panels from the side. In summary, a robust and efficient obstacle detection and avoidance algorithm was successfully developed and proven in simulation.

Acknowledgements

We would like to thank Professor Guido De Croon and all the Teachers and T.As for their continuous guidance and motivation throughout the semester which helped us to finish this project successfully. Even with the special circumstances with the onset of the Corona pandemic, they managed to keep the course going remotely and ending it with an online competition that was really enjoyable.

References

[1] N. Takeda, M. Watanabe, and K. Onoguchi. Moving obstacle detection using residual error of foe estimation. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS '96*, volume 3, pages 1642–1647 vol.3, 1996.

[2] K. Souhila and Achour Karim. Optical flow based robot obstacle avoidance. *International Journal of Advanced Robotic Systems*, 4, 03 2007.

[3] Guido Croon, Emma van de Weerdt, Christophe De Wagter, Bart Remes, and Rick Ruijsink. The appearance variation cue for obstacle avoidance. *IEEE Transactions on Robotics*, 28:529–534, 04 2012.

[4] Jyotsna Sagar and Arnoud Visser. Obstacle avoidance by combining background subtraction, optical flow and proximity estimation. 2014.

[5] Christopher G. Harris and Mike Stephens. A combined corner and edge detector. In *Alvey Vision Conference*, 1988.

[6] G. C. H. E. de Croon, C. De Wagter, B. D. W. Remes, and R. Ruijsink. Sky segmentation approach to obstacle avoidance. In *2011 Aerospace Conference*, pages 1–16, 2011.

[7] Iwan Ulrich and Illah Nourbakhsh. Appearance-based obstacle detection with monocular color vision. *Proceedings of the International Conference on AAAI/IAAI*, 06 2000.

Appendix A Flowchart Navigation Module