Vision-based obstacle detection and avoidance

Cooper Bills, Arjun Prakash, and T.S. Leung

***Abstract*—** *We consider autonomously flying a miniature co-axial helicopter in various indoor environments. The architecture allows developers to write navigation algorithms for aerial robots without worrying about the underlying control features. Our work aims to provide a helicopter with a robust stabilization controller, and an obstacle detection and avoidance feature. We stabilize the helicopter using a combination of Optical Flow and Sonar Sensor data. Obstacle Avoidance allows us to detect and avoid obstacles using a support-vector machine on multiple segments of a live image streamed from the helicopter. We test the architecture using corridor-following and stair-following algorithms written as plug-ins into the system.*

I. Introduction

THE goal is to develop a controller for a helicopter to successfully execute directives from navigation routines. We also developed a learning algorithm to detect obstacles along corridors, and successfully avoid the obstacles. The input of the system is live image sequence captured by the camera mounted in front of an indoor helicopter. Stabilization should react to changes in the environment and keep the helicopter on-course. Obstacle detection should detect all the close range objects and suggests the best turning angles for the helicopter to avoid obstacles.

II. Introduction

The goal is to detect any obstacle along corridor with single onboard camera (Figure 1). We can make use of motion information such as optical flow to detect stationary obstacles in open environment or moving obstacles in confined environment. However it is very difficult to detect stationary objects in confined environment due to the fact those obstacles usually share same kind of motion information as the background.

Figure 1. Obstacle are detected and marked in live scene

III. Related Works

Lot of prior work has been done for stabilization, but they require bulky equipment and lot of processing power. Our problem required us to use minimal payload and no localization. Prior work in this field has been done where they use Optical flow to stabilize a similar helicopter [2].

For obstacle detection, lots of techniques have been explored in the past to detect moving obstacles with single camera. It can be done quite easily using optical flow because moving objects induce a unique motion vector compared to the surroundings. Researchers [2] used image stabilization to "freeze" the environment before detecting the areas which have a different motion as moving obstacle. Alternatively, they can simply perform some clustering method to separate different classes of motion generated by objects and background [3].

Detecting stationary obstacle is more challenging. This can be simplified if we assume that the helicopter can see the ground plane. The work in [4] assumes the entire camera footprint is always on the ground. Any sparse optical flow than can't satisfy homography transformation is declared as an obstacle boundary.

There are also works that do not use optical flow at all [5]. They use image segmentation technique to detect big rocks on the ground, based on low level features like color and edges. However, these methods are limited to ground robots moving on planar surface.

Therefore we designed our obstacle detection algorithm based on image structure. Our hypothesis is that the image energy spectrum provides important information for depth estimation as was studied by [6]

IV. Hardware Platform

In this work, we test our algorithm on the Blade CX2 Coaxial Micro Helicopter. The coaxial rotors limits the maneuverability of the helicopter, but it is more stable than the dual-rotor counterparts. The helicopter weighs approximately 200grams with the battery and has a payload capacity of about 70grams. Our algorithms run off-board on a laptop, and it gives control commands to the aerial platform using the Spectrum DX6i 2.4GHz transmitter. We use the Endurance PCTx interface as a link between the computer and the transmitter.

Our main sensor is a miniature KX141 camera; it is a 795×596 resolution camera that weighs just under 13grams. The images are transmitted back to the computer in real-time using a miniature 2.4GHz 10mW audio/video transmitter set. We sonar sensors (LV MaxSonar-EZ0) to detect height and distance to the walls. The sensor data is transmitted back using a XBee transmitter. Our robot platform is a co-axial hobby helicopter (the Blade CX2), fitted with custom sensors. We have software and hardware to allow a computer to control the helicopter remotely without human interaction. This helicopter control is achieved through a "PCTx" system, giving the computer the exact same controls available to a human pilot.

Figure 2: PCTx software setup.

V. Stabilization

It is very important to be able to successfully stabilize the helicopter during flight. We achieve this using the following individual tools:

A. Optical Flow:

Optical flow is used to correct for any drift and yaw in the helicopter. If the helicopter jerks strongly due to environment induced instability, Optical flow will pick it up. We can then use a Proportionality control to compensate for the jerk and dampen the instability. A differential component to the controller ins not necessary since Optical Flow is itself a differential of the position of the object (we can alternatively view it as a PD controller of position). This can be summarized by the following equations:

$$\dot{x}^* = \alpha \hat{\dot{x}}_{of} \qquad \dot{\theta}^* = \alpha \hat{\dot{\theta}}_{of}$$

where $\dot{x}^*$ and θ^* are the drift correction and yaw correction respectively.

B. Wall Avoidance:

We have sonar sensors mounted on the side of the helicopter. The sonar sensors feed into a Wall Avoidance algorithm that centers the helicopter. It does so by creating an exponential decay from the walls. Therefore, the push away from the wall increases exponentially as the helicopter gets closer to it. This ensures that the walls don't obstruct navigation when the helicopter is well positioned near the center of the corridor, while making sure the helicopter doesn't hit the wall at the same time.

C. Height:

To measure height, we have a vertical sonar sensor, aimed downwards on the bottom of the helicopter. Using this measurement over time, we tune the throttle of the helicopter using a PID controller to stabilize its height at a desired value. The PID controller can be described as:

$$Throttle(t) = Trim_{throttle} - [K_p * P(t) + K_i * I(t) + K_d * D(t)]$$

K_p, K_i, K_d are coefficients for the Proportionality, Integrative, and Derivative parts, respectively.
P(t), I(t), D(t) are Proportional, Integrative, and Derivative results on the measurements over time.
$Trim_{throttle}$ is the throttle required for the helicopter to hover.

The throttle needed to approach the desired height at the current time is now given by *Throttle(t)*. The PID controller uses the Proportional value *(P(t))* to help correct based on the current height measurement at time t. The Integrative value *(I(t))* helps correct over time, so if the Trim is set incorrectly this calculation will make up for it. Finally, the Derivative vale *(D(t))* helps stabilize the system, dampening the correction values as the helicopter approaches the correct height.

VI. Obstacle Detection and Avoidance

We have developed a vision-based algorithm to detect obstacles from single image based on support vector machine (SVM). The algorithm successfully identifies obstacles in a corridor, and an appropriate control routine helps the helicopter avoid the onstacle.

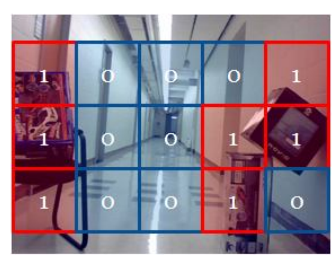


Figure 3. Each cell is manually labeled with 0 & 1 to represent free space and obstacle respectively.

A. Training

To train a SVM to understand what an obstacle is, we captured a training video of motion down a corridor. Some objects were placed along the path trajectory to represent obstacles as shown in Figure 3. A total of 560 training images were collected. We sub-divided each image into 3x5 grid cells. Each cell, with dimension 64x64 pixels, was manually labeled with 0 and 1 which represent free space and obstacle respectively as shown in Figure 1b. As was discovered by Torralba [7], the mean scene depth of an image can be represented by the global and local spectral signature extracted from the frequency domain of the image. Many applications have been developed to perform scene categorization [1, 8] and context-based object detection [9] by using this spectral signature as a global measure. However, to the best of our knowledge, we are the first group applying spectral signature to solve obstacle detection problem.

Each cell is represented by a feature vector of 384 elements which are the coefficients of the extracted spectral signature. We were attempted to add in more features that may also provide depth information such as edge orientation, color and time-to-collision. However extracting different classes of feature definitely increases the processing time during prediction and thus shorten the time available for the flight controller to react against obstacles.

Therefore we took another approach which captures more contextual information by including feature vectors from the four nearest neighboring cells as was done in [8]. The extended feature vector contains information from a larger portion of the image, and thus is more expressive than just local cell. This makes the feature vector of 5 x 384 = 1920 dimensional but only slightly increase the time for feature extraction. In training phase, we used a binary SVM classifier with radial basis kernel function, while tuning the SVM parameters to maximize the recall score during the training stage. Low miss detection rate for obstacle received higher priority than false alarm. If the SVM misclassifies a floor image segment as obstacles, it poses lesser risk to the helicopter.

B. Avoidance

Once the SVM detects an obstacle, we create a field around it to push the helicopter away from it. This is achieved by causing the helicopter to drift away from the obstacle, while continuing to point towards the goal (vanishing point in case of the corridor [1]). To make sure the helicopter doesn't move away from obstacles into walls, the algorithm gives preference to paths closer to the center of the image, since walls are usually along the sides. Figure 5 shows obstacle detection output during actual flight test and the corresponding control command. Green cells indicate free space and red ones mark detected obstacles.

VII. Experiments

A. Data

To try out the reliability in using local spatial signature in obstacle detection, we have coded a prototype of the classifier in Matlab using the open source spatial signature extraction [10] and SVMLight [11]. The trained SVM have been tested against four different sets of test images captured by web camera. All obstacles were manually marked in each image. The accuracy is measured by (1) & (2) and summarized in Table 1.

$$recall = \frac{TP}{TP + MI} \times 100 \tag{1}$$

$$precision = \frac{TP}{TP + FA} \times 100 \tag{2}$$

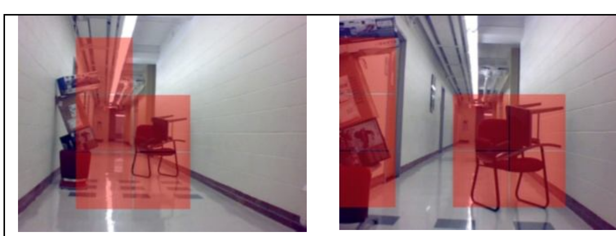
TP denotes true positive which is the total number of cells being labeled as obstacles. *MI* is the total number of cells being labeled as obstacles but misclassified as free space by our classifier. *FA* denotes false alarm which is number of cells being labeled as free space but misclassified as obstacles.

Test Set	Number of images	Recall(%)	Precision (%)
1	380	94.65	94.56
2	528	81.03	66.14
3	1091	83.14	86.37
4	662	73.30	75.31

Table 1. Accuracy of obstacle detection with the test images

B. Accuracy

From Table 1, the SVM classifier achieves at least 70% recall rate for all test sequences. It is very accurate for test set #1 as the obstacles are the same as the training data except being placed at different locations. The classifier still gives reasonable accuracy in detecting other obstacles with different class, size and orientation.



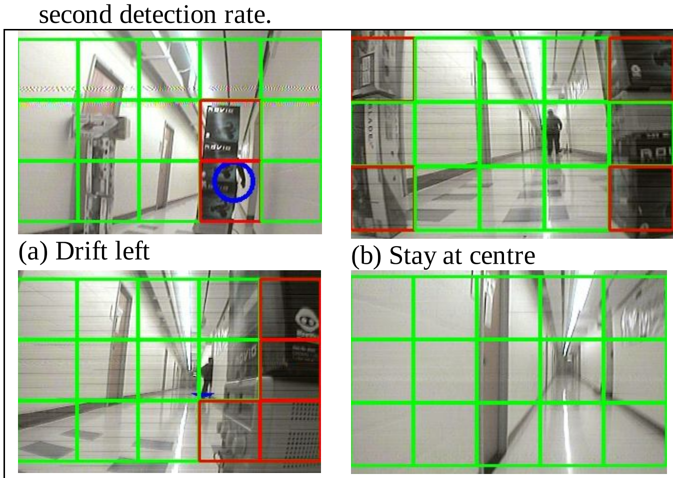
(a) frame #77 (b) frame #118

Figure 4. Obstacle detection output

Figure 4 shows some of the offline detection outputs when the camera approaches two obstacles in test set #2. Our SVM classifier marked the detected obstacles in red.

C. Real-time experiments with helicopter

In order to integrate the new obstacle detection with the existing helicopter platform, the classifier has been re-coded in C++ using OpenCV and spatial envelop source code available at [7]. Unfortunately, the trained SVM model in Matlab is not compatible with C++. Therefore we have coded the training functions in C++ as well. This implementation is quite different with the one used in the scene classifier of our previous work [12]. Instead of computing a spatial signature for entire frame, the new code has been optimized to extract 15 local spatial signatures per image and perform vectors concatenation from nearest neighbor grid cells. Despite of the new complexity, the classifier still can achieve 3 frame-per-second detection rate.



(a) Drift left (b) Stay at centre (c) Drift left (d) Stay at centre

Figure 5. Obstacle detection recorded during flight test

D. Real-time experiments of Stabilization

We tested the robustness of our stabilization by trying out 2 vision based algorithms on the helicopter. These algorithms [1] are meant to successfully detect corridors and stairs and provide a path towards the end of the corridor and stairs. We were able to successfully test these algorithms on our platform and our success rate has been documented in Table 2.

In addition to affirming the robustness of our stabilization routine, this experiment served to show how flexible our platform is. These 2 algorithms were integrated as direct plug-ins into the system.

Environment	Type of corridors	No. of experiments	Total length (meters)	Success rate (%)
Corridor	2	4	20m	75%
Staircase	2	10	4m	50%
Room	1	3	0m	100%
Open area	2	4	1.5m	75%

Table 2. Experimental performance in different settings.

VIII. Conclusion and Future Works

We have successfully implemented obstacle avoidance algorithm and integrated into the helicopter platform. Flight test results show that our system can detect and avoid obstacles along corridor in real-time. However more works are needed to improve the algorithm to handle different class of indoor environments.

References

[1] Yu-hin Joyce Chen, Arjun Prakash, Tung S. Leung and Ashutosh Saxena, “Autonomous Helicopter Navigation of Common Indoor Environments using Vision,” in *IROS 2010* (Submitted)

[2] [1] Demonceaux, C. and Kachi Akkouche, D., Robust obstacle detection with monocular vision based on motion,IVS04, Page(s): 527-532, 2004

[3] [2] Enkelmann, W.; Gengenbach, V.; Kruger, W.; Rossle, S.; Tolle, W., Obstacle Detection by Real-Time Optical Flow Evaluation, Intelligent Vehicles '94 Symposium, Proceedings of the Digital Object Identifier: 10.1109/IVS.1994.639480 1994 , Page(s): 97 - 102

[4] Manolis I. A. Lourakis, Stelios C. Orphanoudakis, Visual Detection of Obstacles Assuming a Locally Planar Ground, In Proc. of ACCV’98.

[5] Liana M. Lorigo , Rodney A. Brooks , W. E. L. Grimson , Visually-Guided Obstacle Avoidance in Unstructured Environments, IEEE Conference on Intelligent Robots and Systems , 1997

[6] Aude Oliva, Antonio Torralba, Modeling the Shape of the Scene: A Hlistic Representation of the Spatial Enevlop, International Journal of Computer Vision, Vol. 42(3): 145-175, 2001.

[7] Torralba, A., & Oliva, A., Depth estimation from image structure, IEEE Pattern Analysis and Machine Intelligence, 24,1226-1238, 2002

[8] A. Quattoni, and A.Torralba. Recognizing Indoor Scenes. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009.

[9] Contextual priming for object detection, A. Torralba, International Journal of Computer Vision, Vol. 53(2), 169-191, 2003.

[10] http://people.csail.mit.edu/torralba/code/spatialenvelope/

[11] T. Joachims, Making large-Scale SVM Learning Practical. Advances in Kernel Methods - Support Vector Learning, B. Schölkopf and C. Burges and A. Smola (ed.), MIT-Press, 1999.

[12] Lear’s GIST Implementation. http://lear.inrialpes.fr/software

[13] G. O. Young, “Synthetic structure of industrial plastics (Book style with paper title and editor),” in *Plastics*, 2nd ed. vol. 3, J. Peters, Ed. New York: McGraw-Hill, 1964, pp. 15–64.