A Novel Stereo based Obstacle Avoidance System for Unmanned Aerial Vehicles

Arjun Chakraborty
Georgia Institute of Technology
School of Electrical and Computer Engineering
Atlanta, USA

Bhargav Srinivasan
Georgia Institute of Technology
School of Electrical and Computer Engineering
Atlanta, USA

ABSTRACT
The use of autonomous unmanned aerial vehicle (UAV) has been on the rise. They are used to replace an ever-increasing amount of labor. There is a need for unmanned aerial systems to operate safely in the environment around them. The work in this paper aims at creating an obstacle avoidance system using Stereo Vision. The work uses standard block matching algorithms. OpenCV and the KTTI Vision Benchmark suite is used. The ArduPilot SITL simulator is used for running the algorithms and displaying the results. The droneapi is an application that is used to access the UAV's information and describe new kinds of flight behavior. The application created is known as STOBA (Stereo Based Obstacle Avoidance), which was created to run within the ArduPilot SITL, in order to provide the mentioned obstacle avoidance capability

Keywords
Stereo Vision, Obstacle avodiance, UAV,OpenCV

1. INTRODUCTION
In this work, a vision based obstacle avoidance algorithm for an unmanned aerial vehicle is presented (UAV). When an UAV moves in a real environment, the perception of the environment around it is crucial. Stereo Vision [4] in itself is a computationally demanding process and hence use of complex methods can make the algorithm slow and difficult to implement in real time.

The proposed method in this project is divided into three separate modules. The first one is the stereo vision algorithm,. It is responsible for providing reliable depth maps, also known as disparity maps of the scenery in the form of frames suitable for the UAV to move autonomously. The second module is the decision-making algorithm. It analyzes the disparity map and finds the most appropriate direction in order for the UAV to avoid the obstacle. The implementation is done using OpenCV[11]. The third module is the use of the ArduPilot Software in the Loop simulator to visualize the environment.

The [1] APM: Copter in the ArduPilot suite is being used to simulate a full range of flight requirements and can be used for various purposes. The Arducopter provides a stable, well-designed advanced autopilot technology, which can be used on a variety of hardware that is supported by the APM (ArduPilotMega). The work was done entirely done on [2] droneapi, which is an interface that can be plugged into ArduPilot system as Ground Control System (GCS) or on board as companion computer.

The API, which is a stereo-based obstacle avoidance system, is designed to run over the [1] ArduPilot autopilot system. The system runs real time at a pre-set frequency and analyzes the environment using stereo cameras mounted on the copter. The camera feeds the images into an image analysis engine. The images are passed onto a stereo module, which creates a disparity map, then applies a gray color threshold in order to filter out the objects that are close to the camera (obstacles), and then applies a histogram equalization technique to figure out a path to traverse.

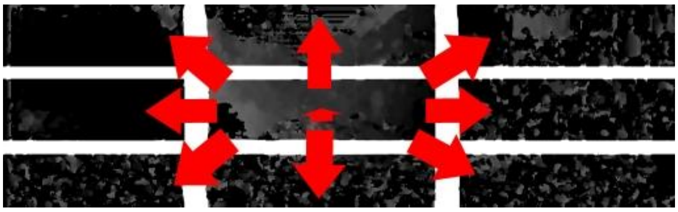

Figure 1:Deciding which path to traverse based on processed stereo image

The dataset used in this work is the [10] KTTI vision benchmark suite. It makes use of an autonomous driving platform to develop real world computer vision benchmark suites. The procurement for datasets for UAVs is difficult. The KTTI vision benchmark suite is suitable for computing disparity images for a low flying quadcopter. Hence, it is suitable for use in this work.

2. BACKGROUND AND RELATED WORK

A. Stereo Vision
The central problem that is being solved in this work is to enable the UAV to perceive the world and increase it's autonomy. [5] A set of two eyes is used to form a visual system that perceives the world in a certain direction. The objects are correlated to form a perceived depth of the world. Stereo Vision is based on this very principle and is a fitting solution to the problem that is being faced.

B. Epipolar Geometry
[6] Correlating two images could be difficult as potentially every pixel in one image could correspond to every pixel in the second image. A 2D search would be a strain on the resources and impractical. Instead, epipolar geometry is used. Each point in either camera must lie on an epipolar line on another camera. This reduces the search to one dimension.

C. Disparity Map
The disparity map [9] is a picture produced by correlating pictures of the same scene. It is intuitive in nature. The close objects are brighter than objects that are further back. Corresponding point that have no distance between them are said to be at infinity and the given disparity is said to be 0. Hence, a disparity map is created.

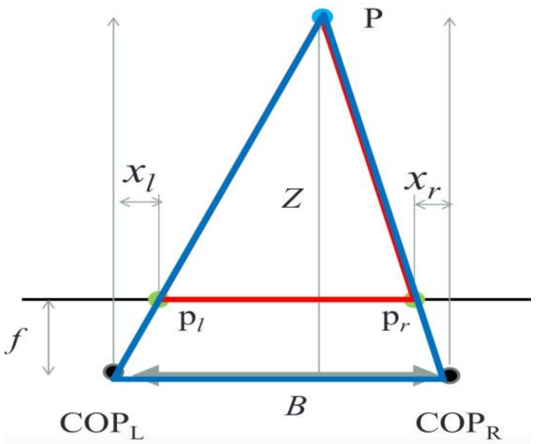

Figure 2: Epipolar view (Kalva 2014)

Let point P project into left and right image. The distance is positive in the left and negative in the right image. It is assumed that the optical axes are parallel, known as camera parameters. CoPl and CoPR are the right and left camera optical centers. The point P is at a distance Z in the camera coordinate system. B is the baseline and f is the focal length.

Using the properties of similar triangles in (pl,P,pr) and *(Cl,P,Cr)*, the expression comes out to be,

$$(B-x_l+x_r) / (Z-f)=(B/Z)$$

$$Z=f * (B/x_l-x_r)$$

(x_l-x_r) is the disparity and Z is the depth. Depth and disparity are inversely proportional to each other.

D. ArduPilot and droneapi

The system architecture of the [3] ArduPilot SITL simulator is shown in the figure. The application leverages the interface of the droneapi. Certain custom actions are predefined, which the copter executes according to the situation. The figure shows that the control module using [2] droneapi plugs into the MAVProxy module, to generate MAVLink (Micro Air Vehicle Communication Protocol) message in order to control the SITL simulator module. [1]MAVLink has a protocol header similar to the CAN (Controller Area Network) bus standards, it has eight fields and includes a check-sum. The SITL module generates both a visual output and a physics simulation of the entire aircraft, which can be viewed.

E. Stereo matching Algorithms

Block Matching Algorithms: Matching pixel for pixel usually gives more than one result and does not give a good result. Methods of block matching algorithms are sum of absolute differences and mean square error.

Semi Global Block Matching: [8] It is the computation along several paths, symmetrically from all directions through the image. Eight paths from all the directions meet at every pixel. Each path carries the information about the cost for reaching a pixel with a certain disparity. For each pixel and disparity, the costs are summed over the eight paths. The disparity with the lowest cost is chose.

F. Obstacle avoidance

To perform obstacle avoidance, the UAV needs to know the distances to the objects around it. [6] Depth information is extracted using stereovision. One of the more popular methods is the threshold estimation method. It involves assuming a threshold T, and then dividing the disparity map into 3 windows. The number of pixels greater than the threshold is enumerated. If it is less than 20 percent of the total number of pixels, the UAV observes the disparity value of the other two windows and chooses the window with a smaller value.

3. ALGORITHM

[11] OpenCV is an open source computer vision library, which is used for computer vision applications of the work. OpenCV inbuilt functions were used for reading and displaying the images. The main task using OpenCV involved creating the disparity images using stereo matching algorithms. The functions used were [11] *StereoBM* and *StereoSGBM*. The inbuilt functions allow the user to vary parameters such as no. of disparities, window size, minimum disparity and speckle window size. The module is a python interface with the MAVProxy protocol which helps us craft MAVLink messages designed to control the standard APM: Copter. The location of the copter, it's orientation *(x,y,z,pitch,roll, yaw)*, velocity and various other parameters can be fetched using these messages. The messages can also be used to send commands to the copter, set waypoints and also provide direct control of the copter using its RC channels.

The decision-making module of our system is based upon the concept of histogram comparison. The *compareHist* function is used. After the disparity map is created, thresholding is done on the disparity map using a programmed range of pixel values. This range helps divide the objects that are closer to the ones that are farther away. The next step involves the divison of the disparity map into 9 equally sized segments. A mask is created for the cooler black and the histograms are matched with each of the 9 segments. The computation is done using the Bhattacharya distance. The segments with the lowest score were discarded. This automatically leads the vehicle to a direction where the number of obstacles is low. This is how obstacle detection works.

4. DETAILS OF IMPLEMENTATION

This is the core of our system; it contains the logic for image processing, obstacle detection and avoidance. In this section, the application used to override control of the quad rotor when it encounters an obstacle. The entire application is written in python and it has seven modules. Additionally, module tests were implemented in each module before they were integrated to the application.

Main: This module is the one that is executed first and links all of the other modules together, and it dictates the flow of control between the modules.

Drone Control: The droneapi library is used in this function to create a Drone class, which contains all of the functions needed to arm, takeoff, and maintain control of the drone. In addition, it has cases to traverse a selected path, when it detects an obstacle.

Vision Module: In this module, a class is written to handle the display of the HUD (Heads Up Display) which shows what the copter is currently observing. The module contains computer

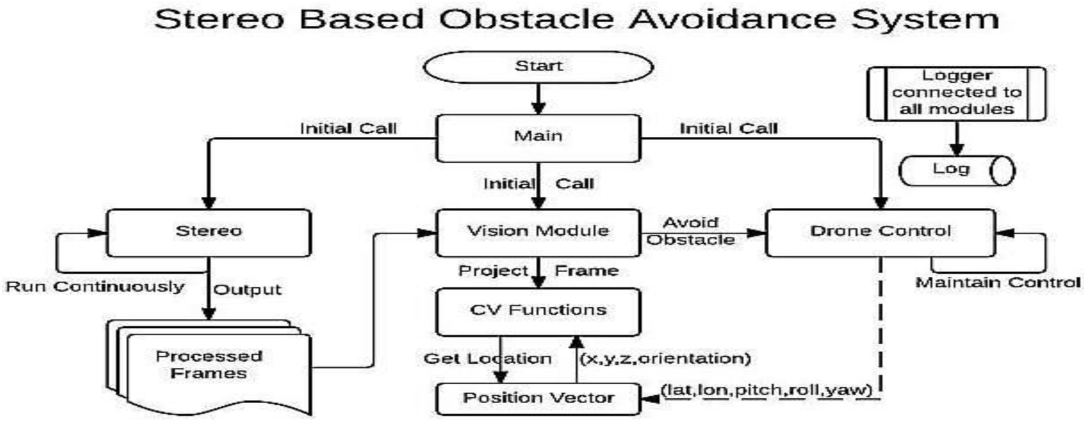

Figure 3:System Flowchart

vision functions, which are used for the projection of the image into a virtual 3D space in the HUD, and rotation matrices are applied corresponding to the movement of the copter, in order to simulate a realistic view of what the copter is observing.

CV Functions: The vision module makes use of the CV functions module, in order to obtain the processed frames in real time from the camera. The OpenCV functions are implemented in this module, on each frame fetched from the camera.

Get Frame: This function fetches a frame from the camera and figures out the projection parameters required to display the image and passes these to the visualize target function.

Visualize Target: This function converts the image from the camera's view to the world view. The functions *getPerspectiveTransform* and *warpPerspective* are used to achieve this effect.

Rotate Image: The functions *getRotationMatrix2D* and *warpAffine* are applied to the image using the orientation of the copter.

Stereo: The Stereo module, has three main functions namely, *createDisparity*, *colorFinder* and *compHist*. The first function creates a disparity map using the stereo images produced and the color finder thresholds the image to filter out obstacles (by removing high intensity colors which correspond to nearby objects.) Then the compare histogram function selects the portion of the image which is the most appropriate path for the copter to traverse.

Position Vector: This module was ported from the ArduPilot Balloon Finder application. This module enables the user to convert the latitude, longitude NEU(north east up) GPS co-ordinates to our frame x,y,z co-ordinates.

Logger: The module is implements a python logger, which logs both to the console and to a file.

5. RESULTS

The performance of the system was measured by supplying images to the stereo module and it was checked to see whether the path chosen by the copter was devoid of any obstacles.

True Positive rate: Twenty sets of images with obstacles were fed into the stereo module, which were randomly selected from the KTTI vision benchmark suite. Eighteen out of the twenty
sets were classified correctly i.e. the algorithm chose a safe path to traverse 90% of the time.

False positive rate: Fifty sets of images devoid of any obstacles were fed into the dataset and the algorithm was expected to not deviate from the current path. Erroneous results were
encountered but a very good false positive rate of 2% was achieved.

6. CONCLUSION

In this work, a selection of stereo matching algorithms has been implemented for the creation of a collision avoidance module. ArduPilot and OpenCV have been used for the purpose. The stereo images were imported from the [10] KTTI Vision benchmark suite and were used for metric tests.

There were various difficulties encountered. The integration of the OpenCV with the droneapi module was very difficult. This required the establishment of control of the copter and the design of a possible flow of control through the various modules. Initially, the dataset proved to be an issue. Photographs around the Georgia Tech campus were captured. Unfortunately various calibration issues were faced, after which the [10] KTTI Vision benchmark suite was used.

The STOBA currently works on a dataset. There is scope for integration of the module into real time. This involves the use of a stereo camera. This would help in real time obstacle. The testing can be done on a real quadcopter. This is because droneapi and ArduPilot have been configured to work on hardware. Improvements can be made on the robustness of the obstacle avoidance technique. It sometimes struggles to detect obstacles on images, which have a lot of clutter.

7. REFERENCES

[1] Mackay,Randy."ArdupilotBalloonFinder."2015. *https://github.com/rmackay9/ardupilot-balloon-finder*

[2] KevinHester,”DronePythonDoc”. 2015.*http://python.dronekit.io/automodule.html*

[3] CentMesh, NCSU Centennial Research Project. “Software in the loop (SITL).”2010. *http://centmesh.csc.ncsu.edu/trac/MeshBed/wiki/HardWare/Drones/Autopilot*

[4] J. Cieśluk, Z. Gosiewski, "A Stereovision System for Real Time Obstacle Avoidance by Unmanned Aerial Vehicle", Solid State Phenomena, Vol 198, pp. 159-164, Mar. 2013.

[5] Kostavelis, Ioannis, Lazaros Nalpantidis, and Antonios Gasteratos. "Comparative Presentation of Real Time Obstacle Avoidance Algorithms Using Solely Stereo Vision.", IARP/EURON International Workshop on Robotics for risky intervention and Environmental Surveillance-Maintenace, Jan. 2010.

[6] Kostavelis, Ioannis, Lazaros Nalpantidis, and Antonios Gasteratos."Real –Time Algorithm for Obstacle avoidance using a Stereoscopic Camera", Thrid Panhellenic Scientific Student Conference on Informatics, 2009

[7] Viet CN, Marshall IW,"Vision Based obstacle avoidance for a small Low-cost Robot", International Conference on Advanced Robotics (ICAR), 2007

[8] Heiko Hirschmuller,"Semi- Global Matching-Motivation,Development and Applications", Photogrammetric week, pp. 173-184, Sep. 2011.

[9] Kalva, Andreas. "Collision Detection System Using Computer Vision on Low Power Devices", LAP Lambert Academic Publishing,2014

[10] Greiger Andreas, Lenz Philip, Stiller Christoph and Urtasun Racquel, "Vision meets Robotics: The KTTI Dataset", International journal of Robotics Research (IJRR), 2013

[11] OpenCV Documentation. 2015. http://opencv.org.