McNair Scholars Program Report

Spring 2017

**Collision Avoidance System for Unmanned Aerial Systems using Stereoscopic Vision**

Erwin Perez

Dr. Subodh Bhandari



McNair Scholars Program

California State Polytechnic University, Pomona

**Table of Contents**

[Table of Figures 1](#_Toc482736791)

[1.0 Introduction 2](#_Toc482736792)

[2.0 Theory 3](#_Toc482736793)

[2.1 Image Rectification 3](#_Toc482736794)

[2.2 Camera Calibration 4](#_Toc482736795)

[2.3 Disparity Map Generation 5](#_Toc482736796)

[3.0 Hardware 6](#_Toc482736797)

[3.1 Airframe 6](#_Toc482736798)

[3.2 Flight Computer 7](#_Toc482736799)

[3.3 Stereo Cameras 8](#_Toc482736800)

[3.4 Autopilot 9](#_Toc482736801)

[3.5 Ground Control Station 11](#_Toc482736802)

[4.0 Software 13](#_Toc482736803)

[4.1 Camera Calibration and Image Rectification 13](#_Toc482736804)

[4.2 Disparity Map Generation 13](#_Toc482736805)

[4.3 Collision Avoidance 15](#_Toc482736806)

[5.0 Results 18](#_Toc482736807)

[5.1 Ground and Static Tests 18](#_Toc482736808)

[5.2 Flight Test- Post Processing 19](#_Toc482736809)

[5.3 Flight Test- Real Time Processing 20](#_Toc482736810)

[6.0 Conclusion and Future Work 21](#_Toc482736811)

[7.0 Acknowledgements 22](#_Toc482736812)

[8.0 References 23](#_Toc482736813)

**Table of Figures**

**Section 2:**

[Figure 2.1 - Stereovision System 2](#_Toc474966114)

[Figure 2.2 - Epipolar Geometry 3](#_Toc474966115)

[Figure 2.3 - Disparity Map Generation 4](#_Toc474966116)

**Section 3:**

[Figure 3.1 - DJI Spreading Wings Hexacopter 6](#_Toc477982094)

[Figure 3.2 - Intel NUC Skull Canyon NUC6i7KYK 7](#_Toc477982095)

[Figure 3.3 - Mounted Point Grey Chameleon Cameras 8](#_Toc477982096)

[Figure 3.4 - Pixhawk Autopilot Module 9](#_Toc477982097)

[Figure 3. 5 - Mission Planner Snapshot 11](#_Toc477982098)

**Section 4:**

[Figure 4.1 - Left Image and Figure 4.2 - Right Image 1](#_Toc474966122)4

[Figure 4.3 - Blended of Both Camera Perspectives 1](#_Toc474966123)5

[Figure 4.4 - Ground Truth Disparity Map of Figure 4.3 1](#_Toc474966124)6

[Figure 4.5 - Grid Generation With Selected Quadrant 1](#_Toc474966126)7

[Figure 4.6 - Collision Avoidance Process](#_Toc474966127) 17

**Section 5:**

[Figure 5.1 - BM and SGBM Disparity Maps 19](#_Toc482736716)

[Figure 5.2 - Disparity Map Generated Using BM 20](#_Toc482736717)

[Figure 5.3 - Post-Processed Disparity Map Using 3-D Viewing Algorithm 21](#_Toc482736718)

[Figure 5.4 - Former Team's Disparity Map of Similar Region 21](#_Toc482736719)

1. **Introduction**

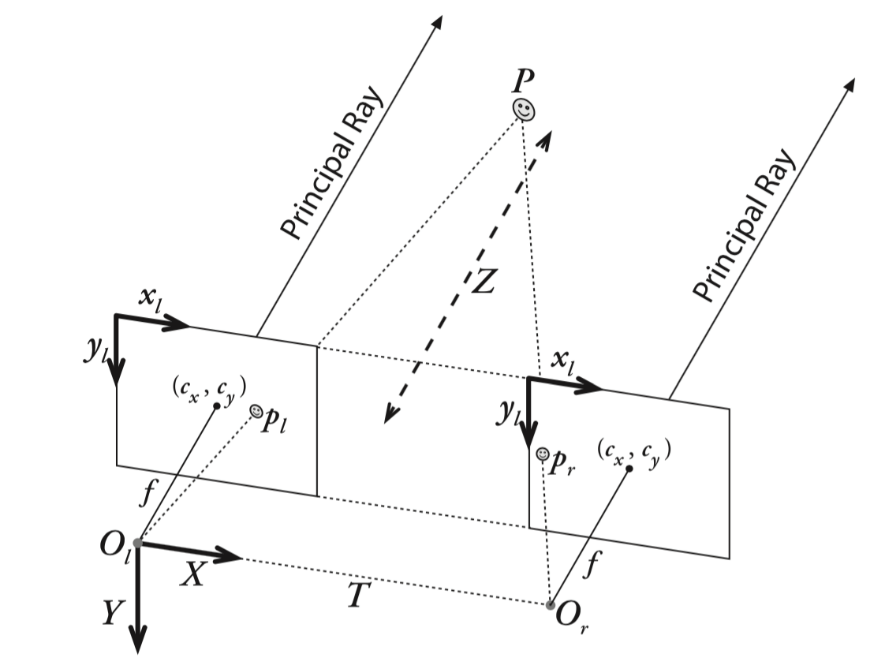
Unmanned aerial systems (UAS) have become the fastest growing sector if aerospace. Their increased use can be attributed to their ability to remove humans from dull, dirty, and dangerous missions. The increased popularity of UAS has made the need for a collision avoidance system more apparent in order to implement them into the Nation Airspace System (NAS). The Federal Aviation Association (FAA) has been pushing for safety among these unmanned aerial vehicles (UAV) which includes collision avoidance systems. There have been many proposed and researched ideas that have tried to solve the problem of collision avoidance for safety. One of these is use of automatic dependent surveillance- broadcast (ADS-B) sensors that would be implemented into each unmanned aerial system (UAS) to be able detect other ADS-B sensors from other UAS. Another method uses Lidar sensors to map out an area, making it possible to detect objects on a third dimensional plane. The aforementioned methods are useful, but relatively more expensive to a third method known as stereoscopic vision. This paper proposes the use of stereoscopic vision as a collision avoidance system for UAS. This method will implement the use of a C++ library and an open source library, OpenCV, to implement the computer algorithms which implement camera vision techniques similar to artificial intelligence. Stereoscopic vision uses two cameras to generate two dimensional images and uses a computer algorithm known as Block Matching to generate a sense of depth perception in a blended image known as a disparity map. Another reason why stereoscopic vision is a formidable option is that it can act as a redundant system to the other methods of collision avoidance mentioned and it can also be used as a standalone system that is more affordable, making it more available to the UAV enthusiast. The project will involve use of commercial-off-the-shelf items to reduce the project cost and application time.

In this paper, we will discuss the application the team has made to improve the previous algorithm for the stereoscopic vision and we will discuss the current results for the flight test and static ground tests. Future work and recommendations will then be discussed as well.

# **2.0 Theory**

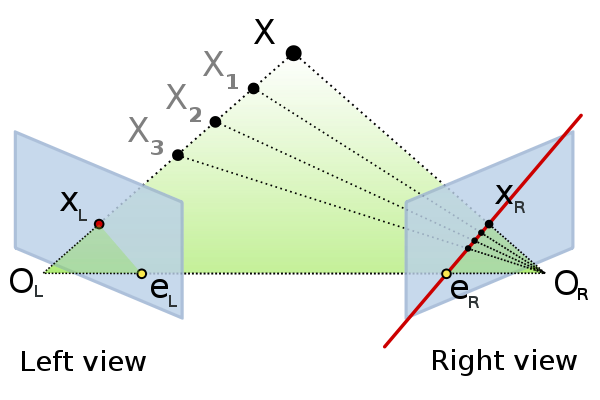
## **2.1 Image Rectification**

Stereoscopic vision works by the use of two cameras located a set distance apart. For the current setup, the cameras are placed 10 inches apart while on the unmanned aerial vehicle (UAV). The ideal stereovision system, shown in Figure 2.1, shows how the two image planes are coplanar to each other and the projection points, and, are on the same row. In stereoscopic vision, it is important that the projection points are leveled on the same pixel row, otherwise an error in data may propagate. Currently, in order to alleviate this error there is an implementation of image rectification. The purpose of image rectification is row-alignment, which means making the corresponding projection points to be on the same pixel level by introducing epipolar geometry.



**Figure 2.1 - Stereovision System**

In epipolar geometry, shown in Figure 2.2, the two rectangles represent the image plane of left camera and right camera respectively. From this image we see that the image planes do not need to be perfectly aligned since there will be a correction made from the use of epipolar geometry. In the image, X represents the points captured by both cameras and and are optical centers. X projects on left image plane on point and projects on right image plane on point . Drawing a line from to we obtain a line called the baseline. The intersection points of baseline and two image planes are called epipole, and . The lines formed by and is called epipolar line, and and are the same. The goal of rectification is making the two epipolar lines parallel in order to make and on the same pixel row, and thus reducing the propagation of error.



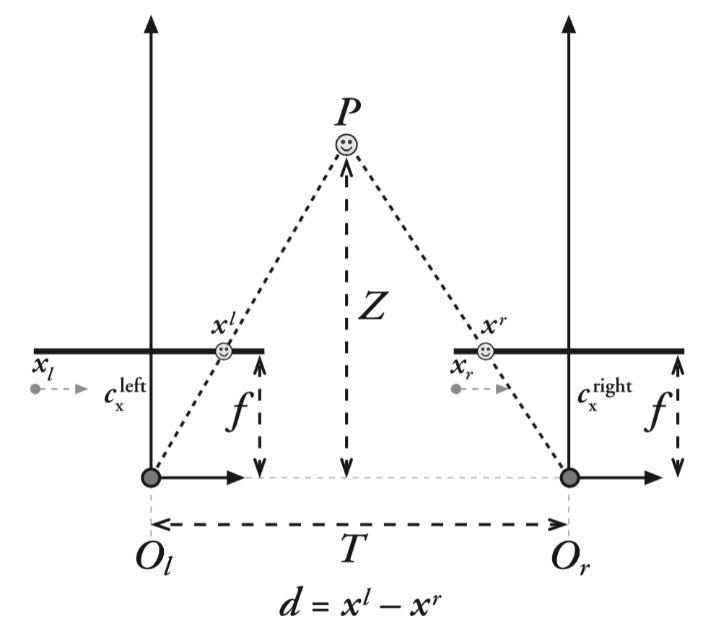
**Figure 2.2 - Epipolar Geometry**

## **2.2 Camera Calibration**

An important part if data collection is camera calibration. Camera calibration is used before any ground or flight tests in order to minimize the error in the generated disparity maps. Camera calibration produces parameters that will be used for image rectification. These parameters consist of intrinsic matrix and extrinsic matrix. Intrinsic matrix contains the information of the property of the camera and the relationship between the camera coordinate and image coordinate. Extrinsic matrix describes the relationship between the world coordinate and camera coordinate. Generating the distortion parameters and doing stereo calibration are also required in this step. The camera lens generates two types of distortion, radial distortion and tangent distortion. Stereo calibration is the process of calculating the geometrical relationship between the two cameras in physical world.

## **2.3 Disparity Map Generation**

Given an object located at point P in physical world, the projection of this point on each of the two camera planes will appear to be in a different location. These two projection points, and , and the disparity between these two projection points is called disparity, as shown in Figure 2.3. Depending on the distance of the observed point to camera plane, the disparity projected on both camera planes will be different. If the observed point is closer to the image planes, the disparity will be smaller. Hence, by using this method, disparity is utilized for determining the relative distance between objects and the cameras.



**Figure 2.3 - Disparity Map Generation**

# **3.0 Hardware**

## **3.1 Airframe**

The airframe used for this project is the DJI S900 Spreading Wings Hexacopter which is shown in Figure 3.2. The hexacopter was chosen due to its light weight and great stability and maneuverability. Another key reason why this hexacopter is a great option for collision avoidance is the fact that it has a payload capacity of over 9 lbs, which is used for the on-board hardware. The UAV weighs around 7 lbs without payload and its six rotors allow it to fly for about 20 minutes while fully loaded, allowing enough time to generate the disparity maps.

[](http://www.google.com/url?sa=i&rct=j&q=&esrc=s&source=images&cd=&cad=rja&uact=8&ved=0ahUKEwiE36_jiZTSAhVC82MKHd8mCYMQjRwIBw&url=http://store.dji.com/product/spreading-wings-s900&psig=AFQjCNGT8piQikNtHrcxys0WIOLclm7jaA&ust=1487315923212033)

**Figure 3.1 - DJI Spreading Wings Hexacopter**

## **Flight Computer**

The on-board flight computer used is a powerful mini-computer designed by Intel, the Intel NUC Skull Canyon NUC6i7KYK, seen in Figure 3.2. This particular NUC incorporates a 6th generation Intel Core i7-6770HU processor, which provides for 32 GB of DDR4 RAM. This kit comes with four USB 3.0 ports and an Ethernet port. For use of external peripherals, a simple wireless mouse and keyboard are used. There are two ways to access this NUC, the first way is to connect it to an external monitor that supports either Mini-HDMI or HDMI; the second way is to use an external 10/100 MBPS desktop switch which we would then connect to the NUC and a personal computer or laptop using Ethernet cords and then open a Remote Desktop Connection. The cameras are connected to the NUC by use of two USB 3.0 slots.

[](https://www.google.com/url?sa=i&rct=j&q=&esrc=s&source=images&cd=&cad=rja&uact=8&ved=0ahUKEwiY97O-0aDSAhUW-2MKHbNeD9oQjRwIBw&url=https://www.newegg.com/Product/Product.aspx?Item%3DN82E16856102166&psig=AFQjCNH5kiYf4sbj34eTyccb_NjFcU_pVQ&ust=1487747489208383)

**Figure 3.2 - Intel NUC Skull Canyon NUC6i7KYK**

## **3.3 Stereo Cameras**

The UAS in this project uses two Point Grey Chameloeon3 cameras. They are ideal for this project as they are minimal in terms of their low profile and low weight. The dimensions of these cameras are 44mm x 35mm x 19.5mm with a mass of 54.9 grams. Special software and API are also required for these cameras to function. These cameras can capture color images at 1.3 MP using a USB 3.0 with resolutions up to 1280x1024 and up to 149 FPS. These cameras also require a lens to be attached before use and the Fujinon YV2.8×2.8SA-2 their low-profile design and low weight of 50 grams. These lenses, however, must be manually focused for the desired range thus are slightly more challenging to properly set up. The USB 3.0 cables used were the ACC-01-2300 USB 3.0 Type-A to Micro-B cable found on the Point Grey website. In order to safely mount the cameras onto the UAS, a 3-D printed mount was created. The two cameras fit inside the mount, which has an opening for each of the camera lenses, seen in Figure 3.3. This allows the cameras to mounted at a set distance on the UAS while minimizing camera movement during flight.



**Figure 3.3 - Mounted Point Grey Chameleon Cameras**

## **3.4 Autopilot**

The UAS needs a method of detecting collisions and proceed to avoid them. The onboard cameras and sensor must work in unison with an on-board autopilot to avoid such threats. Collision threats include, but are not limited to, ground based structures and other aircraft. On this project, a Pixhawk autopilot module is used, seen in Figure 3.4, which comes with pre-loaded open source firmware that allows the hexacopter full autonomous capability. The firmware provides advanced functions such as support for hundreds of three-dimensional waypoints, automatic take-off and landing, as well as sophisticated mission planning and camera controls. The Pixhawk is compatible with the ground station software used in this project, the Mission Planner software. For full autonomy, the Pixhawk requires an external compass away from magnetic interference as well as a GPS unit. As a failsafe method, it also has a built-in hardware failsafe processor that will enable the aircraft to return to the launch site if it experiences radio loss.



**Figure 3.4 - Pixhawk Autopilot Module**

Communication between the autopilot and the ground control station is achieved through the use of two XBee radios. The XBee radios wirelessly transmit and receive the packets of data which can be viewed and analyzed in the ground station. In addition, a 3DR GPS with magnetometer is connected to the Pixhawk in order for the autopilot to receive positioning data from the GPS. This GPS with magnetometer ensures that the autopilot will be fully functional and will give the aircraft the capability of being fully autonomous. The benefit of having a direct connection compatibility of the GPS with the autopilot is that it provides precise navigational data which are shown as waypoints on the ground control station. The orientation of the aircraft is also much more precise with this direct compatibility and ensures that there is minimal error when determining the current aircraft orientation. Because the purpose of this project is to have obstacle avoidance capabilities, accuracy of the autopilot is crucial and having a precise orientation and navigation data is required. The autopilot is being powered through the same power source as the NUC on board which is a separate power source from the propeller’s electric motor. The NUC and the autopilot are directly connected together so the collision avoidance algorithm stored can be implemented on board the aircraft. Having the NUC on board and directly connected to the autopilot enables the aircraft to detect an obstacle and immediately complete the necessary actions to change the current flight path. The autopilot will follow the waypoints generated through the user interface on the ground control station and this will allow the plane to autonomously fly the path necessary for the current mission. If the aircraft senses an obstacle in its path, it will use the collision avoidance algorithm to generate a new route to maneuver around the object and continue on its planned path which was initiated. If no object is sensed in its path, it will continue its route at the same desired elevation and orientation as it was programmed to.

## **3.5 Ground Control Station**

The ground control station is where the main operations of the autonomous aircraft are implemented and this includes the input of the initial waypoints for the desired mission as well as the updated waypoints required for collision avoidance that are changed in real time. In this case Mission Planner was chosen as the ground station due to its full featured application already compatible for the Pixhawk. Mission Planner can be used as a configuration utility or as a dynamic control supplement for the autonomous aircraft. Mission Planner is free to download and quick to understand so it was installed and calibrated fairly quickly based off on the current location at Cal Poly Pomona. Once the firmware is configured on the Pixhawk, it is possible to setup, configure, and tune the aircraft for optimum and desired performance. Also, it is possible to plan, save, and load autonomous missions into the autopilot straight from Mission Planner by simple point and click waypoint entry on interfaces such as Google Maps or any other sort of mapping tools. Between the capabilities of the autopilot and the ground station hundreds of three-dimensional waypoints can be supported for any given mission. The use of this ground control station can be seen in the snapshot of the Mission Planner in Figure 3.5 below.



**Figure 3. 5 - Mission Planner Snapshot**

With the telemetry hardware on the Pixhawk it is possible to monitor the aircraft’s status while it is in operation. One major component is the ability to record the telemetry logs which contain much more information regarding the flight details and this will allow to further view and analyze telemetry logs. It is also possible to operate the aircraft in first person view if this is desired. Some other useful instruments available with Mission Planner are that it is possible to view the planes battery, altitude, speed, orientation, servo functions, real time waypoint generation, and the strength of the signal from telemetry.

# **4.0 Software**

Special computer algorithms are used for various parts of the project. The algorithms used were in C++ while using a main library in Open Source Computer Vision (Open CV), which is used to develop the codes.

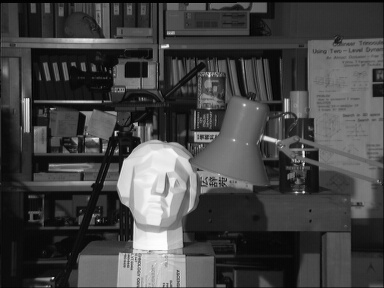
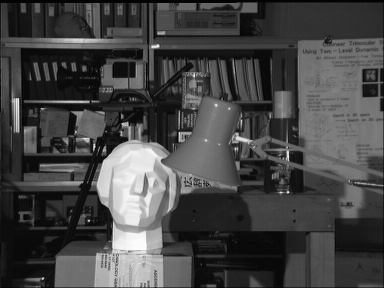
## **4.1 Camera Calibration and Image Rectification**

The camera calibration is conducted for each individual camera by using a chessboard series. This process is necessary in order to calibrate each camera with OpenCV. Using the calibration code, the corners of the chessboard will be detected and utilized to generate parameters, which will determine the accuracy of the camera calibration. While conducting the stereo calibration, the two cameras need to mount still. During this process, the cameras are placed in their housing for testing preparedness. As the calibration code is ran, the two cameras capture a series of chessboard images simultaneously.

## **4.2 Disparity Map Generation**

Disparity map generation is done by conducting a series of processes that mimic the human eyesight. The computer algorithm begins by choosing a pixel from the left image as a reference point and finds a corresponding point on the right image. Once this matching is done, the location of each pixel on each camera plane will be in a different location. Based on this distance between the two matched pixels, the algorithm can determine the distance to the object. This process is repeated for the entire image until the product is a blended image known as a disparity map. This process can be seen in Figure 4.1 and Figure 4.2. The images show the perspective of each camera on the same focal frame. Once these two images are blended, seen in Figure 4.3, we can see that matching pixels will appear in different locations in their respective camera planes. Based on the shift of the matched pixels we can see a pattern between this shift, or disparity, and the distance to the objects. The more an object shifts in the blended image, the closer the object is to the cameras. This is analogous to focusing on an object with our eyes and remaining focused on the object. By closing one eye we will see that the object will appear in a specific location. By then switching the closed eye, we will see that this object will appear to move. The close the object we focused on is to our eyes, the more of a shift that will be perceived. This is precisely what is happening with the computer algorithm. This approach uses a method known as Sum of Absolute Difference (SAD). This process compares two image boxes of left and right images by calculating the similarity of them; the center of the image boxes are reference pixel of left image and target pixel of right image. Choosing the most similarity boxes, the distance of the centers, reference pixel and target pixel, would be the value of the disparity.

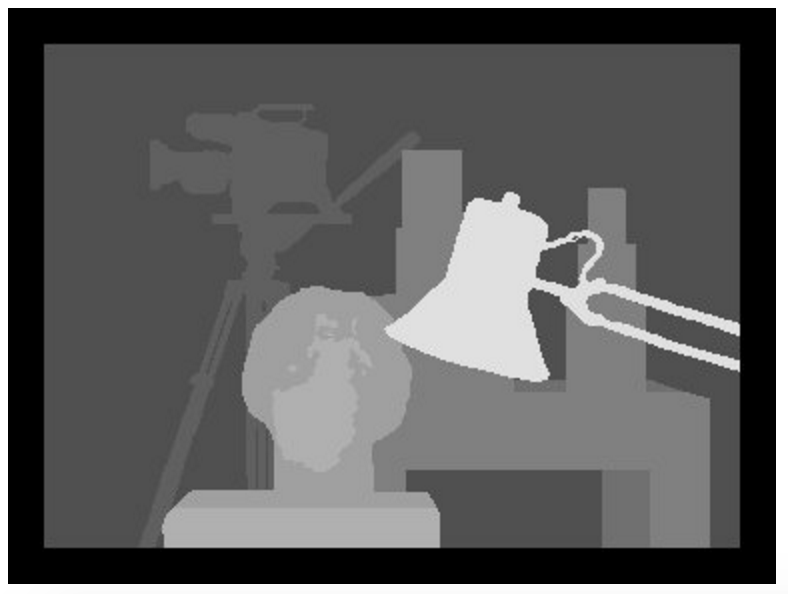
To simplify the computational difficulty, the images are converted to gray scale images which also make it easier to visualize distance to objects. Using an algorithm known as Stereo Block Matching, the pixels in the image are appointed a value from 0 to 255; in this scale 0 represent black and 255 represents white. The value that is appointed to the pixel represents how much the specific pixel shifted. The more an object shifted, a higher numerical value is appointed, thus representing a pixel that is closer to the reference frame. This pixel will then appear a lighter shade in the disparity map. This is visually represented in Figure 4.4. Furthermore, because the pair of images is rectified, the matching direction is only on the horizontal pixel row.



**Figure 4.1 - Left Image Figure 4.2 - Right Image**



**Figure 4.3 - Blended Image of Both Camera Perspectives**

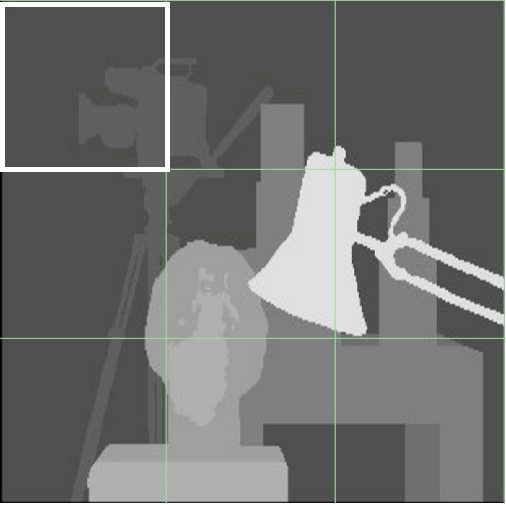


**Figure 4.4 - Ground Truth Disparity Map of Figure 4.3**

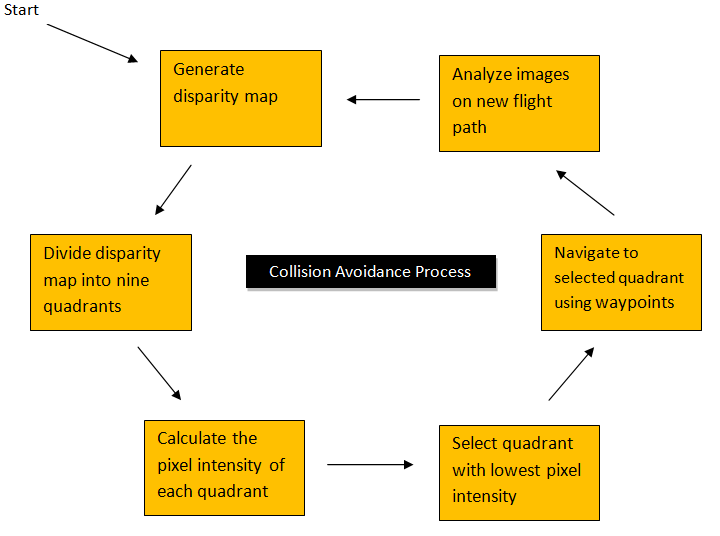
**4.3 Collision Avoidance**

The collision avoidance part of the project involves the avoidance of potential threats during the UAS flight. The objects would be encountered during flight and by executing a maneuver, the UAS must first avoid the obstacle and then return to the original flight path. This process can be broken down into sub-steps: sense, detect, and avoid.

The current collision avoidance algorithm that was used was developed by the previous team under the guidance of Dr. Subodh Bhandari. This team carried out most of the algorithm development on a fixed-wing aircraft and the codes were adapted on the current hexacopter airframe. The proposed collision avoidance algorithm works by developing a nine-quadrant layout of the disparity map, seen in Figure 4.5. Each quadrant is analyzed by recording the pixel intensity, which refers back to the appointing of numerical values to pixels, and comparing them to one another. Based on the quadrant that is determined to be most pixel intense, the quadrant with the most dark objects, the algorithm will select that quadrant. Said quadrant will be the quadrant that is least likely to host a collision. Once the safest quadrant is selected, the UAV will execute a maneuver by flying toward the selected quadrant through waypoint navigation. A typical autonomous collision avoidance process in outlined in Figure 4.6.



**Figure 4.5 - Grid Generation with Selected Quadrant**

****

**Figure 4.6 - Collision Avoidance Process**

**5.0 Results**

**5.1 Ground and Static Tests**

The generation of the disparity maps can be done by the use of two similar algorithms: block matching (BM) and semi-global block matching (SGBM). Block matching generates one disparity map in 0.093 seconds by using a 3 pixel by 3 pixel searching box scheme. Semi-global black matching uses a 7 pixel by 7 pixel scheme and produces a disparity map in 0.18 seconds. This difference in pixel size region dictates the accuracy of the disparity map. By using SGBM, the disparity map generation is slower but yields more accuracy disparity map. This minute difference in output can be seen in Figure 5.1.

During the generation of the disparity maps, some conditions cause error on the disparity map, such as texture-less areas, half-occlusions and regions near depth discontinuities. OpenCV provides a function to minimum the error, and it requires two disparity maps from a set of left and right image. The first disparity map is same as usual, using left image as reference and searching on right image. And the second disparity map is using right image as reference image and searching left image. The filter function compares these two different disparity maps and minimize the errors.

1. **Left image (b) Right image**

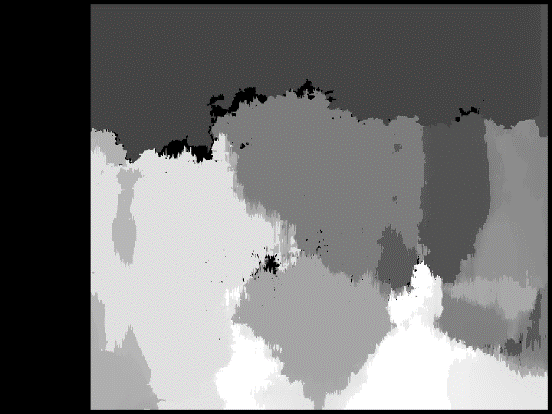


**(c) BM disparity map (d) SGBM disparity map**

**Figure 5.1 - BM and SGBM Disparity Maps**

**5.2 Flight Test- Post Processing**

All data and disparity maps are post processed in this step in the project. Currently, the disparity maps generated are also stored in the NUC and once the manned flight is over, the images are collected and analyzed for accuracy and clarity. The last flight test generated images that are acceptable but still have some imperfections. These inaccuracies are due to the shadowing of the object analyzed. Figure 5.2 shows a disparity map that was generated of a tree line. The center of the disparity map shows a light gray region that should be displayed as a white object as it was the closest tree to the UAS, but instead it was shown as a light gray. This was likely due to the cameras not interpreting the shadows correctly and interpreting this region as a separate and further region. This is the only minor error in our data as the camera calibration is being conducted before flight tests and this has yielded only minute calibration offsets of less than 0.1%.

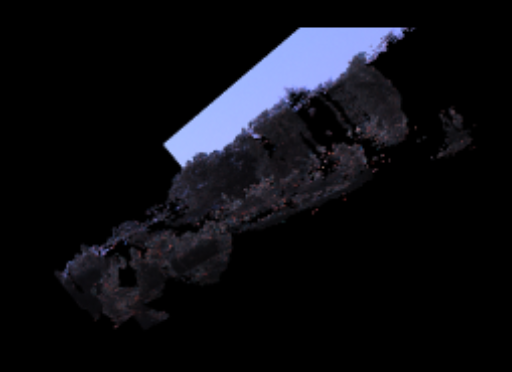
****

**Figure 5.2 - Disparity Map Generated Using BM**

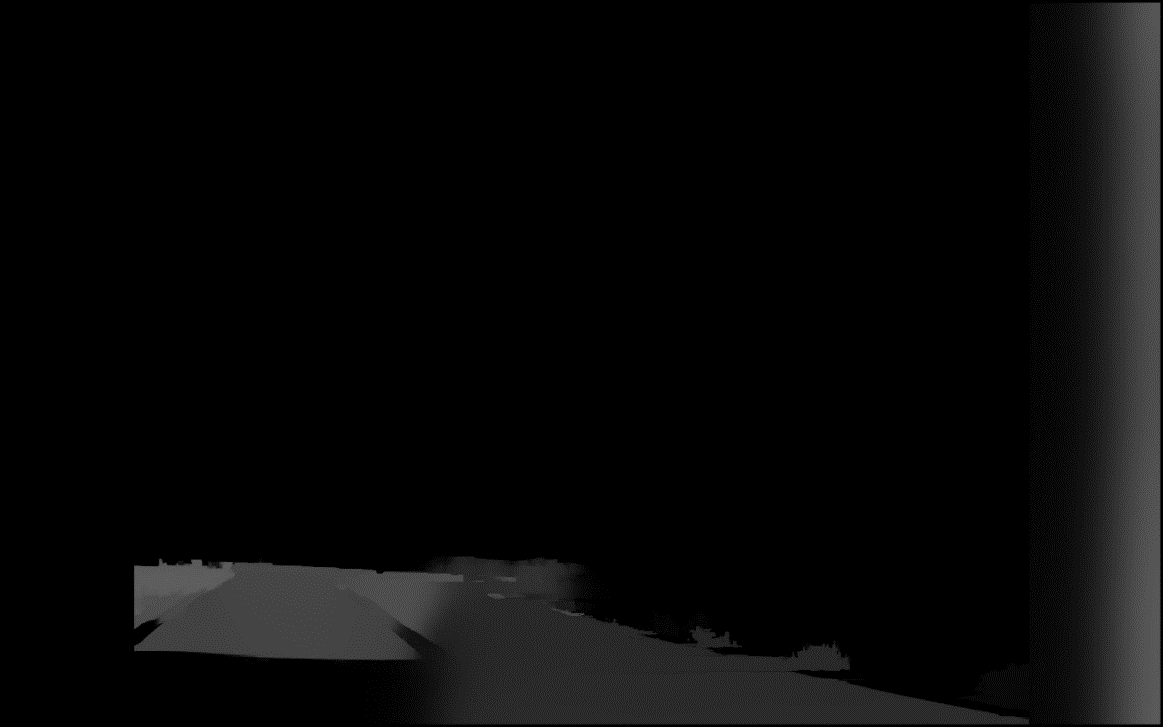
These results indicated that for a distance of to about 50 feet, the disparity maps are accurate enough to use and move on to the next step in the project, which is waypoint navigation. Further testing will be made to ensure that shadow regions will not interfere and yield faulty disparity maps which may hinder the UAS mission. Optimizing the algorithms further will provide faster processing speed which will allow the UAS to read further distance to help avoid collisions, since the UAS will move relatively faster during full speed flight.

**5.3 Flight Test- Real Time Processing**

The restriction of this flight testwas that the NUC was not able to record the actual flight image since imaging processing itself took up most of the processing power. Most of the processing capability is used for the disparity map generation which produces about 5 disparity maps per second. In Figure 5.3 we see a post-processed tree line as seen in Figure 5.2. This disparity map was modified using a 3-D viewing algorithm which allows the user to turn a disparity map and turn a 2-D image into a 3-D option. From this image, we see that the tree line was accurately collected by the disparity map. The shadow error region previously discussed can also be seen in this 3-D image, but as far as the overall accuracy of the image we can see that the disparity map generation is in a favorable and acceptable step in the project. Comparing these results with the former team’s disparity map generation, seen in Figure 5.4, it is clear that the advancements in the computer algorithm have reached a point where waypoint navigation is attainable.



**Figure 5.3 - Post-Processed Disparity Map Using 3-D Viewing Algorithm**

****

**Figure 5.4 - Former Team's Disparity Map of Similar Region**

**6.0 Conclusion and Future Work**

One major notion to keep into account is that the current algorithm is being developed from a code that is mainly geared towards static disparity map generation. Because of this, the movement of the aircraft must be accounted for and vibrations in the UAS must be minimized to avoid errors in the data. Further testing will need to be made in a dynamic aspect to ensure that full speed UAS flight can be handled with the current algorithm. At this point, all hardware is integrated and the base algorithm has been developed enough to begin with waypoint generation, which will introduce the final step to the autonomous collision avoidance aspect of the project. Improvements of the disparity map generation has seen a vast improvement from the previous team’s progress as seen in Figure 5.4.

Future work for the project includes testing with real-time collision avoidance. And further improving the BM algorithm to attain more accurate disparity maps. Further testing with shadow regions can yield to improvements that will avoid such errors. Due to safety limitations, real-time collision avoidance should only be conducted at a point where are systems are running completely efficiently and free of error. Future work will include the use of a faster onboard processor which will allow for more accurate disparity maps and faster algorithm processing. This could be attained by a faster processor or testing with two onboard processors instead of one. Another possible method and direction for the project could result in the use of a single camera that is specifically made for stereoscopic vision. This could yield to improved image quality as the camera is specifically made for this technique of stereoscopic vision. If the project takes this direction, the computer algorithms may need to be modified or replaced completely depending on compatibility.

**7.0 Acknowledgements**

I would like to express my gratitude to my mentor, Dr. Bhandari, and especially the former team members that extended me their work and progress. I would like to especially thank Kishan Patel who was available for questions regarding the progress of the previous team. Finally, I would like to thank the McNair Scholars program for their guidance and opportunity to allow me to discover my potential and capabilities. Thank you.

**8.0 References**

1. Alvarez, H., L.M. Paz, J. Sturm, and D. Cremers. "Collision Avoidance for Quadrotors with a Monocular Camera." Print.
2. Anthony Tan, Jake Banuelos, Kishan Patel, Minwoong Chae, Jonathan Cratsley, Muhammad Moiz Khanqadri, Daniel Boebinger, Andrew Nipp, Jeff Wang, Subodh Bhandari, and Daisy Tang, “*Flight Test Results of the Collision Avoidance System for Unmanned Aerial Systems using Stereoscopic Vision”,* American Institute of Aeronautics and Astronautics. Print.
3. Bhandari, S., Srinivasan, T., Gray, J., Torstenbo, M., Corral, J., Brown, N., Mcdorman, N., Wood, J., and Raheja, A., “Collision Avoidance System using Stereoscopic Vision for Unmanned Aerial Systems,” *AIAA Infotech @ Aerospace*, 2016.
4. Blin, Richards, et al. “Obstacle Avoidance System for UAVs using Computer Vision.” Proceedings ofAIAA Invotech@Aerospace Conference, FL, 5-9 January, 2015.
5. Hrabar, S. "3D Path Planning and Stereo-based Obstacle Avoidance for Rotorcraft UAVs." *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems* (2008).
6. Humenberger, M., Zinner, C., Weber, M., Kubinger, W., and Vincze, M., "A Fast Stereo Matching Algorithm Suitable for Embedded Real-time Systems," *Computer Vision and Image Understanding*, Vol. 114, No. 11, pp. 1180-1202, November 2010.
7. Schaeffer, R., “A Standards-Based Approach to Sense-and-Avoid Technology,” *AIAA 3rd Unmanned Unlimited Technical Conference, Workshop and Exhibit*, September, 2004.
8. Lacher, A. R., Maroney, D. R., and Zeitlin, A. D., “Unmanned Aircraft Collision Avoidance - Technology Assessment and Evaluation Methods,” *The MITRE Corporation*, 2007.
9. Park, Jongho, and Youdan Kim. "Obstacle Detection and Collision Avoidance of Quadrotor UAV Using Depth Map of Stereo Vision." *AIAA Guidance, Navigation, and Control (GNC) Conference* (2013).
10. Saneyyoshi, K., “Stereo Vision System on Automobile for Collision Avdoiance,” *IAPR Conference on Machine Vision Applications*, June, 2011.
11. Tzovaras, Dimitrios, Nikos Grammalidis, and Michael G. Strintzis. "Disparity Field and Depth Map Coding for Multiview 3D Image Generation." *Signal Processing: Image Communication* 11.3 (1998): 205-30.