Obstacle Detection using Binocular Stereo Vision in Trajectory Planning for Quadcopter Navigation

Albert Bugayong and Manuel Ramos, Jr
Electrical and Electronics Engineering Institute, University of the Philippines Diliman, Quezon City, Philippines
Email: albert.paolo.bugayong@eee.upd.edu.ph, manuel@eee.upd.edu.ph

Abstract. Quadcopters are one of the most versatile unmanned aerial vehicles due to its vertical take-off and landing as well as hovering capabilities. This research uses the Sum of Absolute Differences (SAD) block matching algorithm for stereo vision. A complementary filter was used in sensor fusion to combine obtained quadcopter orientation data from the accelerometer and the gyroscope. PID control was implemented for the motor control and VFH+ algorithm was implemented for trajectory planning. Results show that the quadcopter was able to consistently actuate itself in the roll, yaw and z-axis during obstacle avoidance but was however found to be inconsistent in the pitch axis during forward and backward maneuvers due to the significant noise present in the pitch axis angle outputs compared to the roll and yaw axes.

1. Introduction
Mobile robotics deals with the creation of robots capable of locomotion through land, water and air. It also deals with intelligent navigation using data obtained from sensors for self-localization, mapping and path planning [1], [2].

Sonars, IR sensors, laser range finders, and multiple cameras have been used in mobile robots to obtain distance information from its environment for obstacle avoidance and navigation. Sonars, IR sensors and laser range finders, however, have the disadvantage of obtaining information only at a specific point in space. As a result, they are used in large quantities to form sensor matrices or are mounted onto motors to perform a sensor sweep on the surroundings to obtain more information from the environment. Some researches, notably from Saxena, et.al, used monocular vision and structured lighting to obtain distance information from its environment [3]. Stereo vision is ideal as it is passive and is less computationally expensive compared to monocular vision.

This project focuses on indoor navigation for reconnaissance in closed environments such as buildings and mining areas.

2. Methodology

2.1. Sensor Fusion
Gyroscopes are susceptible to drifts or accumulations of angle measurement errors while accelerometers are prone to high frequency noise due to vibrations. A high-pass filter is applied to the gyroscope to cancel out drift while a low-pass filter is applied to the accelerometer to minimize noise from vibrations [4]. A complementary filter was used to fuse gyroscope and accelerometer data to
obtain accurate pitch, yaw and roll angle estimations. The equation for the complementary filter is given by:

\[
\theta = (\alpha) \ast \left(\theta + \dot{\theta}_{\text{gyro reading}} \cdot dt\right) + (\alpha - 1) \ast \left(\theta_{\text{accel reading}}\right)
\] (1)

where \(dt\) is the sampling period and \(\alpha\) and is defined by:

\[
\alpha = \frac{\tau}{\tau + t_s}
\] (2)

where \(\tau\) is the time constant which determines both the high pass and the low pass filter cut off frequency and \(t_s\) is the sampling period [4], [5].

2.2. Stereo Vision

For cost matching, the sum of absolute difference (SAD) is the most commonly used matching criteria due to its low complexity and good performance [6]. The equation is given by:

\[
SAD = \sum_{(i,j)\in W(x,y)} \left| I_L(i,j) - I_R(i,j+d) \right|
\] (3)

where \(I_L\) is the left pixel intensity, \(I_R\) is the right pixel image intensity, \(W\) is a square window surrounding pixel \((x, y)\) and \(d\) is the disparity value [6].

The sum of absolute difference takes the absolute difference between the reference window from the first image and the target window in the second image. The reference window is moved pixel by pixel across all target windows until the associated cost is minimized [6]. The depth of an object is measured from the baseline between the camera centers. The disparity of an object is the difference of its projections from the two images [7]. The depth can be computed using the equation:

\[
Z = \frac{bf}{d}
\] (4)

where \(Z\) is the depth calculated, \(b\) is the baseline or the distance between the two cameras, \(f\) is the camera focal length and \(d\) is the disparity [7].

A vector slice was obtained from the center row of the disparity map. The depth from the camera’s baseline for each disparity was computed using equation 7. This distance was then used to obtain obstacle locations from the environment and served as a basis for the execution of a proper obstacle avoidance maneuver.

2.3. Motor Control

A PID controller was applied separately to the Altitude, Pitch, Yaw and Roll controllers [8], [9].

\[
PID_{\theta} = k_p(\theta_{\text{error}}) + k_i(\theta_{\text{error, accumulated}}) + k_d(\theta_{\text{error, predicted}})
\] (5)

where \(\theta_{\text{error}} = \theta_{\text{current}} - \theta_{\text{desired}}\) [8].

The pitch, roll and yaw and altitude each have their own set of PID controllers. Each PID controller is represented by:

\[
duty_n = PID_{\text{pitch}} \pm PID_{\text{roll}} \pm PID_{\text{yaw}} \pm PID_z \pm duty_{\text{base, throttle}}
\] (6)

where \(duty_n\) is the input PWM signal to the ESC for each motor \(n\), \(duty_{\text{base, throttle}}\) is the base duty or constant throttle required for quadcopter elevation [5], [9].

2.4. Trajectory Planning

To construct the Cartesian Histogram Grid, a 1D depth vector slice from the 2D disparity map is obtained. The VFH algorithm [2], [10], [11] was modified to suit the scope of this research. Due to the absence of position tracking, once the desired heading is obtained, the current map information is disposed of in preparation for the next sampling. The first stage of this process is to create the
occupancy grid. 5 meters was the set maximum distance wherein obstacles are detected. Consequently, the grid dimensions were set to 5 by 5 meters and divided into 1089 cells, equivalently, 33 by 33 cells. The resolution of each cell is effectively 15 centimeters. The cartesian coordinates for each occupied cell were then obtained through the following equations [12]:

\[ x = z \cos(\theta) \]  
\[ y = z \sin(\theta) \]

where \( x \) and \( y \) are the coordinates of an occupied cell, \( z \) is the depth from the depth vector, and \( \theta \) is the depth's corresponding angle computed by:

\[ \theta = pixel(i) \times \left( \frac{fov}{nop} \right) \]

where \( i \) is the pixel row coordinate, \( fov \) is the field of view and \( nop \) is the number of pixels in the row.

For the second stage of this process, the obstacle magnitude vectors \( m_{i,j} \) and direction vectors \( \beta_{i,j} \) were computed for each cell obstacle using equations 12 and 13.

\[ m_{i,j} = (c_{i,j})(d_{\text{max}} - d_{i,j}) \]
\[ \beta_{i,j} = \tan^{-1}\left( \frac{y_{i}-y_{o}}{x_{i}-x_{o}} \right) \]

where \( c_{i,j} \) is the certainty value set to 1 assuming that the depth obtained is the same as the actual environment, \( d_{i,j} \) is the distance of the robot from the cell in centimeters and \( d_{\text{max}} \) is the maximum distance possible which is 5 meters or 500 centimeters. The obstacles were then grouped into sectors to form the primary polar histogram using equation 14 to 16 where \( \alpha = 5^\circ \) is the angular sector resolution and \( k \) is the discrete quantized multiples of \( \alpha \) and \( H^P \) is the polar obstacle density (POD).

\[ k = \text{INT}\left( \frac{\beta_{i,j}}{\alpha} \right) \]
\[ \alpha = \frac{360}{n} \]
\[ H^p_k = \sum_{i,j \in c_a} m_{i,j} \]

The sectors were labeled in the opposite order as the map generated was a mirror image of the actual environment. For the third stage, a binary polar histogram \( H^b \) was derived from the primary polar histogram by filtering the POD’s through a threshold was experimentally obtained.

\[ H^k = 1, \text{ if } H^p_k > \tau \]
\[ H^k = 0, \text{ otherwise} \]

At the fourth stage, the candidate valleys were obtained from the resulting free sectors from the binary polar histogram. The valleys were then labeled as free or blocked if too narrow. At the last stage, the center sector location of each candidate valley was obtained and the valley selected is the valley closest to the center or current heading of the quadcopter.

3. Hardware Implementation

The Raspberry Pi Compute Module with IO Board v.1 was selected as the single board computer (SBC) for this project as it comes with two Camera Serial Interface (CSI) ports. This was essential to enable the use of a stereo camera pair [13]. The Raspberry Pi Model B+ was used for motor control and trajectory planning blocks. Two lightweight Raspberry Pi camera modules were used for implementing stereo vision [14]. OpenCV, also known as Open Source Computer Vision Library, is an open source image processing software library used in implementing Stereo Vision in this research [15]. Adafruit’s 10 DOF IMU containing a L3GD20H gyroscope, LSM303 compass and the LSM303 accelerometer in a single breakout board was used for this research. A HC-SR04 ultrasonic sensor was
used for altitude estimation. The DJI F450 quadcopter frame used has a range between 600g - 1200g. The total weight of all components was found to be 955.5g ensuring the payload of the quadcopter was within its limits.

4. Results and Analysis

4.1. Stereo Vision

It can be seen on figure 1 that the errors in distance measurement from the stereo vision sensor were 20 cm to 50 cm away from the actual distance which is sufficient for the application in this project. The discontinuity errors in the depth map slice was taken into account in the VFH block of the system by marking them as blocked during the sector clustering stage.

![Figure 1](image1.png)

**Figure 1:** Depth slice with subject at 1 meter (a) and 3 meters (b) from the baseline.

4.2. Motor Control

![Figure 2](image2.png)

**Figure 2:** Altitude control flight response at 100cm setpoint

![Figure 3](image3.png)

**Figure 3:** Trajectory Planning – Command vs Response – Left Entry

![Figure 4](image4.png)

**Figure 4:** Trajectory Planning – Command vs Response – Right Entry

![Figure 5](image5.png)

**Figure 5:** Trajectory Planning – Command vs Response – Reverse Maneuver

4.2.1. PID Controller. The Single Loop Altitude PID was used for altitude control. It could be seen at figure 2, that the setpoint of 100cm during hover was reached with minimal to no oscillations.

The cascaded PID controller consists of two loops. The inner control loop or the rate loop controls the angular rate while the outer loop or the stabilize loop controls the angle. A cascaded PID controller was used to obtain angular velocity data from the gyroscope which was not affected by high frequency vibrations to reduce noise entering the system from the D controller. The resulting cascade is a P-PID
control loop. The current refresh rate of the control loop was 500Hz to 540Hz.

4.2.2. Trajectory Planning

![Figure 6: Obstacle Set-up](image)

In the tests conducted using the set-up in figure 6, consistent correct steering along the yaw axis inflight was observed given the appropriate velocity was attained when approaching the obstacle. A fast approach caused the obstacle not to be detected because of a slow frame rate of 6Hz. This framerate entailed a delay of 16ms in average thus an ideal traveling velocity was required to compensate. However, when an obstacle was detected during a fast approach, the quadcopter might be too near causing oscillations in the steering decision requiring a reverse maneuver to be performed. The oscillation was caused by the low sector resolution set at 5° which was equivalent to only 12 sectors. This was a seldom problem given the correct velocity was maintained. It can also be noted that if too many commands on each axis overlap such as when the quadcopter was oscillating between two obstacles, sudden jumps in altitude were observed.

In the data observed, the pitch and roll axis readings from the complementary filter were very noisy with ±3° in the roll axis and ±8° error in the pitch axis when hovering due to motor vibrations which caused the readings to be unreliable with commands less than the magnitude of the noted errors. However, it was found that using commands greater than 5° on the pitch axis caused fast approaches and doing the same with the roll axis caused severe banking leading to overshoot, at worse cases overpowering the pitch command impeding forward movement. Thus, a lower range of commands was selected, ±3° for both the pitch and the roll. Despite the problem with the readings, there was a visible response in the pitch and roll axes while given a command inflight. This was most apparent with the roll axis which was enabled consistent banking. The default pitch command for slow forward travel however was not consistent.

4.2.2.1. Left Entry. During an entry attempt at the left side of the set-up, Figure 3 details the response of the quadcopter in each axis given a command. It could be observed that due to the noise in the pitch and roll axes, it was difficult to attain the roll and, more so, the pitch axis setpoints. The yaw, however, was less prone to noise influence due to the large angle value setpoints which was much greater than the noise from the complementary filter output. The yaw was able to steer correctly, albeit, with a delay approximately 1.5 seconds. The PID for the yaw axis was tuned for this smooth slow response as a fast response would cause the quadcopter to jump along the z-axis. The first yaw command issued was positive to turn the quadcopter counter clockwise together with a pitch command of 0.6° to slow down and -0.7 to gain speed, and a roll command of -3° and -2° to bank left. As the last positive yaw command was issued, the roll performs a counter bank to avoid drifting along the horizontal and the pitch axis resumes with the default pitch command as it approaches the next obstacle. The second yaw command was negative to turn the quadcopter to clockwise as it approaches the left obstacle and performs the same pitch maneuvers as described before but performing the reverse of the previous banking maneuvers.
4.2.2.2. Right Entry. Figure 4 details the response of the quadcopter in each axis as an entry in the right side of the set-up was performed. It can be observed that due to the noise in the pitch and roll axes, it was difficult to attain the pitch and the roll axis setpoints. To avoid the first obstacle, the quadcopter this time turned clockwise with a negative angle on the yaw axis, and banked to the right as seen by the positive roll angle commands between 3 and 4 seconds. Simultaneously, the quadcopter pitched upward to slow down as shown in the present positive angle command and then proceeds to decrease its banking angle as it avoids the obstacle. The opposite yaw and roll maneuvers were executed at the 5-second mark as the quadcopter approaches the second obstacle at the right side of the set-up. The same pitch maneuvers were executed simultaneously. The commands after the 6-second mark were to be disregarded as they were data collected as the quadcopter was stowed in preparation for the next trials.

4.2.2.3. Reverse. Figure 5 shows a case wherein oscillation in steering was observed when the quadcopter approaches too fast thus performing the decision making at a distance too near the obstacle. The quadcopter was set to pitch back whenever it encounters a large change in yaw angle commands, a change greater than 45˚. The pitch was set to 1.25˚. Pulses of this magnitude could be seen between the 5- to 7-second mark in the plot of the pitch axis. This eventually caused the quadcopter to pitch back to gain distance from the obstacle and then attempts again to avoid the obstacle heading towards the right side.

5. Conclusion
The data obtained from the stereo vision sensor was reasonably accurate up to 5 meters. However, the current SBC used in this research has proven to be insufficient to get a standard frame rate of 30 frames per second for a sufficiently fast update rate to provide more current information for the navigation block.

The Raspberry Pi was sufficient for motor control due as it was able to operate the PID Control loop at a rate of 500Hz to 540Hz resulting to sufficient corrections to maintain stable flight. The system, however, requires a fixed velocity when travelling forward to compensate for the delay in obtaining depth information due to the slow frame rate as well as the slow yaw response which was necessary to avoid sudden jumps along the z-axis when turning. Take-off and landing were not implemented requiring the quadcopter to be released at a height above the setpoint during operation. There exist lateral drifts on the X and Y axes with the X axis minimal while the Y axis has a tendency to pitch forward due to the inherent bias of the quadcopter. Inconsistencies in maintaining travelling velocity can be attributed to the significant noise present in the pitch axis angle outputs compared to the roll and yaw axes.

References
[1] G. Cook, Mobile robots: navigation, control and remote sensing. John Wiley & Sons, 2011.
[2] Louenell B. Gueta and Manuel C. Ramos, Jr. Mobile robot navigation using binocular stereo vision MS thesis, University of the Philippines-Diliman, 2005.
[3] Saxena, S. H. Chung, and A. Y. Ng, “3-d depth reconstruction from a single still image,” International journal of computer vision, vol.76, no.1, pp.53–69, 2008.
[4] S. Colton and F. Mentor, “The balance filter,” Presentation, Massachusetts Institute of Technology, 2007.
[5] Carissa Norielle L. Cruz, Motor Control using a Smartphone for Fixed Path Quadcopter Flight, Undergraduate Student Project, University of the Philippines-Diliman, June 2015.
[6] R. H. Thaher and Z. K. Hussein, “Stereo vision distance estimation employing sad with canny edge detector,” International Journal of Computer Applications, vol. 107, no. 3, 2014.
[7] N. Oon-Ee, V. Ganapathy, and S. Ponnambalam, “Modular stereo vision: Model and implementation,” Depth Map and 3D Imaging Applications: Algorithms and Technologies: Algorithms and Technologies, p. 245, 2011.
[8] Zul Azfar, A.; Hazry, D., "A simple approach on implementing IMU sensor fusion in PID controller for stabilizing quadrotor flight control," in Signal Processing and its Applications (CSPA), 2011 IEEE 7th International Colloquium on , vol., no., pp.28-32, 4-6 March 2011

[9] Bouabdallah, S. "Design and control of quadrotors with application to autonomous flying." PhD diss., École Polytechnique federale de Lausanne, 2007.

[10] Borenstein, J.; Koren, Y., "The vector field histogram-fast obstacle avoidance for mobile robots," in Robotics and Automation, IEEE Transactions on , vol.7, no.3, pp.278-288, Jun 1991

[11] Ulrich, I.; Borenstein, J., "VFH+: reliable obstacle avoidance for fast mobile robots," in Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on , vol.2, no., pp.1572-1577 vol.2, 16-20 May 1998

[12] A. Guevara, “Using The Sensor Kinect For Landmark Recognition At Escuela De Ingeniería De Antioquia”, Escuela De Ingeniería De Antioquia, Undergraduate Project, 2011

[13] Raspberry Pi Foundation (2014, April 7). Raspberry Pi CMIO [Online]. Available: https://www.raspberrypi.org/blog/tag/compute-module-io-board/

[14] Raspberry Pi Foundation (2013, May 14). Raspberry Pi Camera [Online]. Available: https://www.raspberrypi.org/documentation/hardware/camera/

[15] opencv.org