UNSYS digital

Journal of Unmanned System Technology

# Vision-Based UAV Line Formation

Sutthiphong Srigrarom, Yilin Niu, Eugene Tan Ren Jie

University of Glasgow Singapore

***Abstract***— Conventional Unmanned Aerial Vehicle (UAV) formation requires radio communication between the UAVs and the ground control station. However, noise and interruption in the communication links may result in severe consequences. In this paper, vision-based line formation is introduced to eliminate the problems mentioned earlier. Individual UAV with a camera on-board can detect its neighboring UAVs by implementing the concept of the Histogram of Oriented Gradients (HOG) descriptors and cascade classifiers. Image processing such as removal of shadow is applied to improve the detection rates.

Upon successful detection, the Leader-Follower formation theory is applied to allow the follower UAV to distinguish and follow the path of the leader UAV. Lastly, to control the movement of the UAVs, a flight controller is designed and implemented on Simulink, based on the line formation requirements.

***Keywords***— Unmanned Aerial Vehicle, vision-based, Histogram of Oriented Gradients, cascade classifier, Leader-Follower, flight controller.


DOI: 10.21535/just.v5i3.989

## I. INTRODUCTION

VARIOUS visual object detection methods were proposed for object detection of different sizes and shapes. The most well-known and widely used method was proposed by Paul Viola and Michael Jones [1] due to its rapid and high detection rates. This method is based on Haar feature and uses the integral image, AdaBoost and combine complex classifiers in a cascade structure. By implementing the integral image, the calculation speed for Haar feature is improved. AdaBoost is used to achieve an accurate and high detection rate classifier by cascading the weak classifiers using a weightage voting learning algorithm. Rainer Lienhart and Jochen Maydt [2] extends [1] in two areas, they extended the Haar-like feature by adding a set of 45° rotated feature. Another area is about the different types of AdaBoost and proves empirically [3] that Gentle AdaBoost outperformed the other two boosting variants which are Discrete AdaBoost and Real AdaBoost.

On the other hand, the Local binary pattern (LBP) [4] is an effective tool to describe the image texture features of an image with shorter training time, the computation is simpler than Haar feature. However, LBP is not sensitive to small changes in the localization which results in poor performance under lighting changes in the environment [5].

Navneet Dalal and Bill Triggs [6] proposed Histogram of Oriented Gradients for human detection. It calculates the gradient magnitude and direction of each pixel, generate a histogram of a small patch of pixels (cells) and sort them into 9 bins, and form a histogram for a big patch of pixels (blocks). To achieve illuminance invariance, the histogram undergoes a contrast normalization. In this paper, Histogram of Oriented Gradients was used as it was proven to be effective at capturing the overall shape of an object and insensitive to light changes as compared to the two methods discussed earlier on.

Line formation became one of the leading research areas in the last two decades. Line formation represents the scenario of leader and follower. There are many papers that had been studied on achieving line formation with limited information. Luca Consolini *et.al* [7] developed a line formation by controlling the relative position and orientation with using an omnidirectional camera. However, this line formation provides the estimation of the leader's velocity and require communication between the robots and requires a large bandwidth. The method proposed by Noah Cowan *et al.* [8] is that leader's linear and angular velocity is unknown, however it can be estimated by communication.

To achieve the Vision-based Line formation, the requirements for the line formation must be simple. H. J, Min *et al.* [9] developed an algorithm that only required the relative length and angle from the follower to leader, without any form of communication in between the UAVs or markers on the floor. The information required by this algorithm can be extracted easily using a 2D camera. This algorithm was subsequently tested on ground robots. Therefore, in this paper, the algorithm will be implemented into the UAV together with the HOG detection to achieve the formation.

Computer vision is about the extraction and analysis of informative features that are present in an image. The existence of strong fluctuations in illumination across an image such as shadows has been proven to be problematic for many computer vision algorithms. For example, by applying edge detection on two similar images extracted from [10], with and without the presence of shadow as seen in Figure 1. It could be seen that the shadow-less image produces a better result as more edges were detected.

Corresponding author: Sutthiphong Srigrarom
(e-mail: spot.srigrarom@glasgow.ac.uk)
This paper was submitted on November 6, 2017;
revised on December 21, 2017; and accepted on December 30, 2017.

There are many methods proposed to remove the shadow, however, not all methods are applicable to the UAV context which requires real-time processing and robustness in different backgrounds.

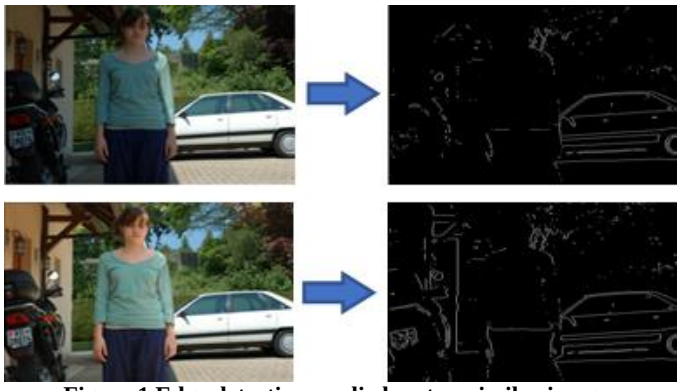

**Figure 1 Edge detection applied on two similar images**

Y Matsushita *et al.* [11] proposed a method that was very effective in removing the shadow, however it requires a collection of images taken from a fixed point which make it unsuitable for this project. On the other hand, GD Finlayson *et al.* [12] proposed a method which depends on the camera model used. By projecting the image colors in the 2D log-chromaticity space at the direction that is orthogonal to the lighting direction (dependent on the camera model), the result is a 1D illumination invariance image which makes the shadow stands out. Thus, further image processing could be done to remove the shadow since its location is known.

Image recovering algorithms are commonly used for different applications such as image cloning. In an image cloning process, a small image is added into a big image seamlessly. The main concept of this cloning process is that the gradients around the small image, when pasted onto the big image, are set to zero. Thus, making a smooth transition.

This concept could be applied to remove the shadow in an image by taking the shadow like a small image, setting the gradients at the shadow edge to zero. An image can be presented by a Laplacian equation or a Poisson equation. To solve these equations, different methods as such using finite-difference approximations to discretize these equations, and to restore the image with the new set of the gradients.

Thus, this paper will look deeper into GD Finlayson *et al.* [12] method as the algorithm is not computational extensive since the UAV doesn't have the capability to hold a large processor onboard. Lastly, this paper will explore different recovery methods to determine each method's effectiveness and computational time.

## II. Histogram of Oriented Gradients (HOG)

The HOG describes an object by the gradients of each pixel in the image. Each pixel consists its own direction and magnitude, and the gradient defined the change in the intensity between pixels. This section will introduce briefly on the feature extraction process to obtain the HOG descriptor.

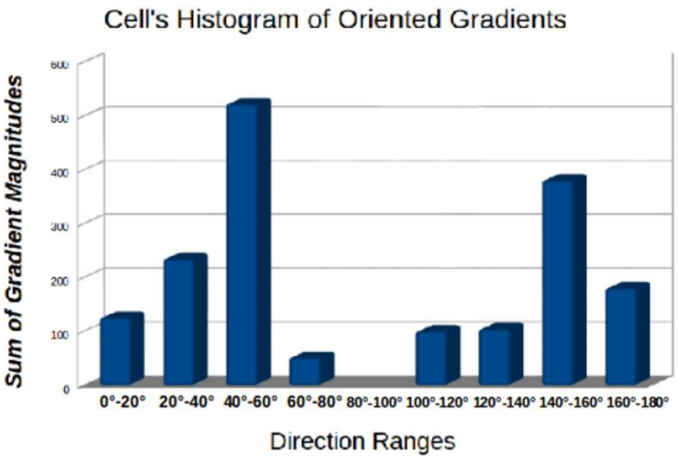


**Figure 2 Cell's Histogram of Oriented Gradients**

### A. *Gradient Computation*

The first step is to calculate the $x$-direction gradient by convolution with the vector $[-1\ 0\ 1]$ to obtain $X'$ and $y$-direction gradient $Y'$ by convolution with $[-1\ 0\ 1]'$ of the image.

The next step is to find the gradient magnitude and direction for each pixel $(i, j)$ given by:

$$\text{Gradient magnitude}(i,j) = \sqrt{\left(X'_{ij}\right)^2 + \left(Y'_{ij}\right)^2} \quad \textbf{(1)}$$

$$\text{Gradient direction}(i,j) = \arctan\left(\frac{Y'_{ij}}{X'_{ij}}\right) \quad \textbf{(2)}$$

### B. *Spatial/Orientation Binning*

To compute the HOG vector, divide an input image into many small sections (blocks) as shown in Figure 3 which contain an even smaller section (cells) shown in Figure 4. Each pixel has its own gradient orientation element and uses it to calculate a weighted vote, the votes are accumulated into orientation bins over local spatial regions is called as cells.

Take the gradients magnitude and direction of the cell to compute 9 bins histogram between 0 – 180 degrees which 1 bin is 20 degrees away as shown in Figure 5, extracted from [13]. Navneet Dalal and Bill Triggs [6] proves that as the number of orientation bins increasing, the performance also improves expressively up to around 9 bins, improvements become insignificantly from 9 bins onwards.

### C. *HOG visualization*

The next step is to compute the 9 bins histogram for the other 8 cells in the block. Concatenate 1 block's histogram into 81-sized vector, by normalization to achieve illumination invariance. Overlapping of the blocks improve performance, but it also increases the descriptor size. Keep the overlapping of the blocks at 50% and repeat steps shown as above, a complied HOG vector will be obtained for the image. The HOG descriptor visualization to demonstrate the result after all the steps can be seen in Figure 3.

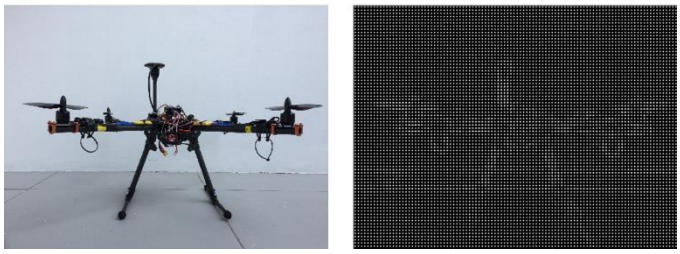

**Figure 3 HOG visualization of a quadcopter in an image**

Basically, all the dominant pixels in each cell are presented in the HOG visualization.

### *D. Cascade classifier with gentle AdaBoost*

AdaBoost is a learning algorithm and is an approach to machine learning. It is kind of weightage voting algorithm. By combining all the weak and inaccuracy learners hence construct a high accuracy prediction rule [14]. Rainer Lienhart *et al.* [3] compared the differences between Discrete AdaBoost, Real AdaBoost and Gentle AdaBoost. Experiment results show that Gentle AdaBoost is more stable and robust when comes to the noisy data and give the best performance. Each unique feature is considered as a weak classifier [15]. The weak classifier learning corresponds to the detection of threshold and parity which can maximize the chances of isolating the positive and negative samples. Furthermore, Gentle AdaBoost applies regression by using the weighted least squares method which can improve the performance.

A weak classifier is trained to detect nearly all the objects in the image at every stage, a cascade classifier is trained to reject some images that do not include the desired object. The concept is shown in Figure 4 where h is the hit rate, and f is the false alarm rate. As mentioned earlier, a weak classifier is only better than randomly guess since one weak classifier contains one feature of the object, furthermore, the feature based classifier is added at each stage of boosting so that it can achieve the best hit rate. When the number of stages increase, more weak classifiers are needed, to obtain the desired false alarm rate at give hit rate [2][3].

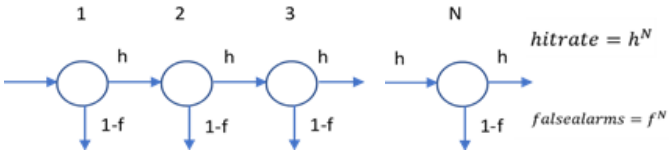


**Figure 4 Cascade classifiers with N stages**

Number of stages indicates the number of cascade stages to train the detector. The accuracy of results increases with the increase in the number of stages, however, this requires more samples and longer training time.

### *E. Dataset sample selection*

Selection of dataset is important to ensure good detection rate. Traditionally, the training dataset is huge, however, it may not produce a good detection rate [16] as it may cause the HOG visualization looks unclear and not able to extract the features well. It is better that the negative images have the same background as the positive images but without the UAV, and too many of negative samples may not achieve a good detection rate. Experiments were carried out with different sample sizes and the results can be found in Table 1. It was found that having 1492 positive sample and 5306 negative samples resulted in the best practical detection rate. Also, due to the vibration of the UAV and it is difficult to align the follower perfectly with the leader, the number of positive samples were taken in misalign slightly.

To produce robust HOG features, both positive samples and negative samples were taken in the simple and complex background, *e.g.*, lab environment and grass patch with trees. As leader-follower formation is a real-time application, and image processing is considered as dense calculation for the system, so all the training and real-time testing image resized to 640×480 pixels.

### *F. Non-Maximum Suppression (NMS)*

At the test stages, one UAV may be detected as several UAVs and that caused problems in the line formation as the bounding box is an important factor to calculate the length and angle from follower to leader. NMS has been used widely in computer vision to help in object detection as an intermediate step. NMS is used to locate the interest object from the group of detections which near the true position. NMS can be achieved by setting the overlapping threshold of each bounding boxes [17].

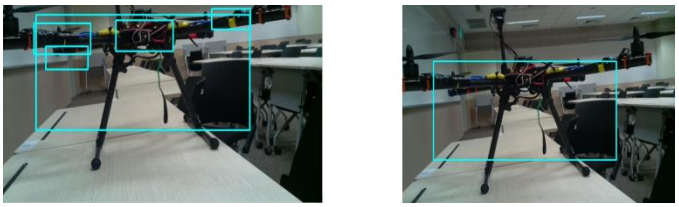

**Figure 5 (a) Detection without NMS (b) Detection with NMS**

**Table 1**

| | Settings | | | | | Results | | |
|---|---|---|---|---|---|---|---|---|
| No. | No. of positive sample | No. of negative sample | False detection rate | Stage | True positive rate | Detection rate | Practical false detection rate | Failure detection |
| 1 | 1500 | 5000 | 0.1 | 7 | 0.995 | 78.9 | 15.8 | 5.26 |
| 2 | 1500 | 5000 | 0.1 | 8 | 0.995 | 73.7 | 26.3 | 0 |
| 3 | 1500 | 5000 | 0.1 | 7 | 0.997 | 78.9 | 15.8 | 5.26 |
| **4** | **1500** | **5300** | **0.1** | **7** | **0.997** | **84.2** | **10.5** | **5.26** |
| 5 | 1500 | 5300 | 0.1 | 7 | 0.998 | 73.7 | 15.8 | 10.5 |
| 6 | 1500 | 18000 | 0.1 | 7 | 0.997 | 57.9 | 0 | 42.1 |
| 7 | 1600 | 5300 | 0.1 | 7 | 0.997 | 63.2 | 31.6 | 5.26 |
| 8 | 1800 | 5300 | 0.1 | 7 | 0.997 | 84.2 | 5.26 | 10.5 |
| 9 | 2000 | 5000 | 0.1 | 7 | 0.995 | 52.6 | 36.8 | 10.5 |
| 10 | 2000 | 5000 | 0.2 | 7 | 0.995 | 63.2 | 31.6 | 10.5 |
| 11 | 2000 | 18000 | 0.1 | 7 | 0.995 | 68.4 | 0 | 31.6 |

### G. *Final UAV detection results*

The UAV detection results can be seen in Figure 6. In Figure 6, screenshots are captured at different interval of the real-time video stream. The UAV were placed slightly tilted during the test to ensure robustness during real-time flight. For more details, see the full clip at https://youtu.be/ydV8jZE8-d4.

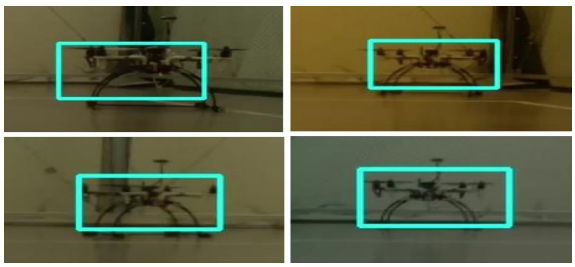

**Figure 6 Screenshots of real-time detection at different interval**

## III. Selection of Leader UAV in Formation

### A. *Working principles of leader selection*

It is necessary to recognize the leader UAV in a formation as the follower UAV shall strictly follow the leader UAV only. To differentiate between the follower and leader, a number card was allocated to each UAV as their new identification. Three classifiers are applied in this part of the algorithm and designed in such a way that leader (number) identification detection only occurs within the UAV bounding box.

Leader selection algorithm distinguished individual leader UAVs and unidentified UAVs. Furthermore, the concept of leader selection algorithm is based on the position of the number identification bounding box. Referring to Figure 7, if the coordinate of number bounding box $(x_5, y_5)$ is within the range of $(x_1, y_1)$ and $(x_2, y_2)$ for both $x$ and $y$ axis, number identification of 1 will issues to the UAV 1 on the left, and same concept apply to UAV 2. Besides, a UAV will be defined as unidentified UAV if no number identification attached to it.

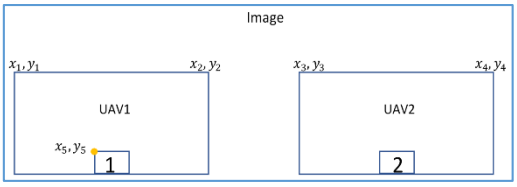


**Figure 7 Coordinate system of bounding box for leader selection**

### B. *Results from the leader selection algorithm*

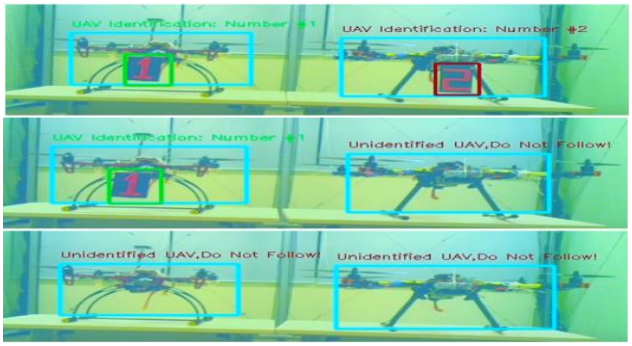


**Figure 8 Leader selection algorithm assigning identification number**

## IV. Shadow Removal in An Image

When capturing images in a different environment, it was observed that the intensity across the whole image changes. These changes were partly due to the lighting source such as the direction of sunlight) or that the object was blocked from the lighting source which makes the object look darker. Therefore, a method to remove the intensity in an image is required to differentiate between an object which itself is dark in color and an object which appeared to be dark in color (e.g. shadow cast over an object). With the HOG detection method which uses gradient, presence of shadow might deduce the probability of reliable detection.

GD Finlayson *et al.* [12] shown that a mathematical equation that consists the intensity term for each color channel could be derived based on certain assumptions. Thus, by dividing two color channels, the result of this division is an intensity invariance image.

### A. *Mathematical equation for 2D camera*

The camera response of each color channel can be modelled as shown below [10].

$$p_k = \int E(\lambda) L(\lambda) S_k(\lambda) d\lambda \quad \textbf{(3)}$$

$k$ = {Red, Green, Blue}
$E$ = Energy spectrum of the light source
$S_k$ = Sensitivity of the camera sensor
$L$ = Illumination of the object

It is difficult to solve the camera response due to the integral term. Therefore, under the assumption that the camera response of each channel acts like a delta function, equation (3) can be simplified to the following.

$$p_k = E(\lambda_k) L(\lambda_k) S_k(\lambda_k) \quad \textbf{(4)}$$

### B. *Energy spectrum of the light source*

To introduce the intensity term in the camera response equation, the light source is assumed to be within the Planckian locus. It means that the light source is assumed to be a blackbody. A blackbody has a specific intensity value depending on its body's temperature. The Planckian locus provides a good approximation for typical light sources such as daylight since these sources' temperature falls within the locus.

By using the Planck radiation formula, $E(\lambda)$ can be approximated as

$$E(\lambda, T) = I \frac{8\pi hc}{\lambda^5} \frac{1}{e^{\frac{hc}{\lambda k T}} - 1} = I k_1 \lambda^{-5} e^{-\frac{k_2}{\lambda T}} \quad \textbf{(5)}$$

$I$ = Intensity of the light source
$\lambda$ = Wavelength of the light source
$T$ = Temperature of the black body
$k_1$ and $k_2$ are constants

Energy spectrum of the light source, $E(\lambda, T)$ is now a function of temperature to consider different type of light sources.

### C. *Intensity invariance image*

The final camera response equation is given by:

$$p_k = Ik_1{\lambda_k}^{-5}e^{-\frac{k_2}{\lambda_k T}}L(\lambda_k)S_k(\lambda_k) \quad (6)$$

Now, by dividing two color channels:

$$X_1 = \frac{Red\ (p_1)}{Green(p_2)} = \frac{Ik_1{\lambda_1}^{-5}e^{-\frac{k_2}{\lambda_1 T}}L(\lambda_1)S_1(\lambda_1)}{Ik_1{\lambda_2}^{-5}e^{-\frac{k_2}{\lambda_2 T}}L(\lambda_2)S_2(\lambda_2)} = \frac{{\lambda_1}^{-5}e^{-\frac{k_2}{\lambda_1 T}}S_1(\lambda_1)}{{\lambda_2}^{-5}e^{-\frac{k_2}{\lambda_2 T}}S_2(\lambda_2)} \quad (7)$$

From the equation above, the intensity term had been removed, making $X_1$ intensity invariance. Note that $L(\lambda_1) = L(\lambda_2)$ as $L$ is the illumination of the object.

However, D. Finlayson *et al.* [13] suggested that the intensity invariance image is susceptible to noise. For example, when dividing the red and green channel, if the image depicts a red dominant environment, the value of green channel would be significantly low making the result undesirable. The solution was to make the mean of all three channels as the denominator.

By projecting two intensity invariance vectors, $Y_1$ & $Y_2$, in a certain direction in the logarithmic space, the result is an illumination & intensity (I&I), $Z$, invariance greyscale image [13].

$$X_1 = \log\left[\frac{Red\ (p_1)}{\sqrt[3]{Red(p_1)+Green(p_2)+Blue(p_3)}}\right] \quad (8)$$

$$X_2 = \log\left[\frac{Green\ (p_2)}{\sqrt[3]{Red(p_1)+Green(p_2)+Blue(p_3)}}\right] \quad (9)$$

$$X_3 = \log\left[\frac{Blue(p_3)}{\sqrt[3]{Red(p_1)+Green(p_2)+Blue(p_3)}}\right] \quad (10)$$

$$Z = (X_1 - X_2)\cos\theta + (X_1 + X_2 - X_3)\sin\theta \quad (11)$$

$\theta$ = Projection angle in the logarithmic space

For different types of RGB camera, the projection angle would be different. Therefore, trial & error was carried on different sample images with different projection angle. Ideally, by using the projection angle found during the trial & error process, the shadow in the images should stand out from other objects. In this paper, the sample images used for testing can be found in Figure 9.

**Figure 9 Sample images with shadow used for processing**

From the results seen in Figure 10, it was observed that for different projection angles, the color profile of the shadow is different (light to dark) and there were unwanted objects as seen in (d) where the man was detected too. The ideal projection angle was 2π which lead to in the ideal processed image (e) where only the shadow was present in the image.

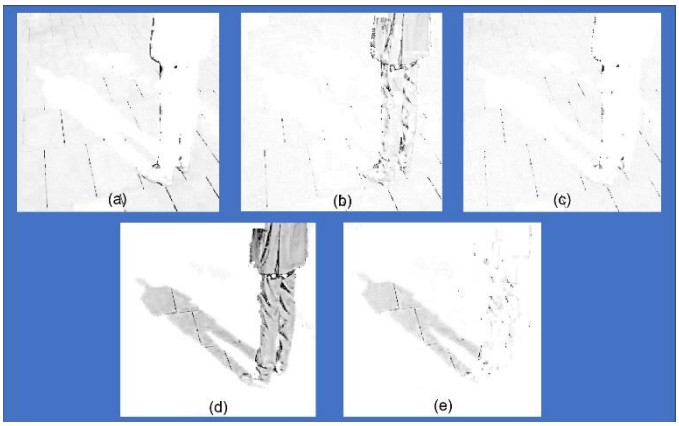


**Figure 10 Processed images with different projection angles**

### D. *Shadow detection from processed invariance image*

By human eyes, it is easy to recognize the presence of the shadow in the image. However, in computer vision, a method is required to tell the computer about the presence of shadow.


**Figure 11 Greyscale values of image in & outside of shadow**

In Figure 11, the shadow appeared to have a lower greyscale value (a) as compared the greyscale value in the rest of the image (b). With this information, the shadow could be masked using binary thresholding in MATLAB together with some simple image processing to remove noise.

**Figure 12 Shadow masked through binary threshold**

An accurate masking of the shadow is vital to ensure that when the binary image is applied in the image recovering algorithm, other objects that are masked as seen in Figure 11(a) will not be removed during the process.

### E. *Image recovering algorithm*

Before moving to the image recovering algorithm, a change in certain parameters of an image is required to remove the shadow effectively. Recall that a shadow results in a change in intensity and illumination between pixels. Therefore, if the gradients at the edge of the shadow mask are set to zero, it indicates to the computer that there is no change in pixel values which means no presence of shadow. Once the gradients are modified. The processed image is ready to be recovered. Discrete Laplacian equation was selected to represent the image because it consists both $x$-axis and $y$-axis gradient terms. Given a square sized image, the Laplacian equation above can be simplified to:

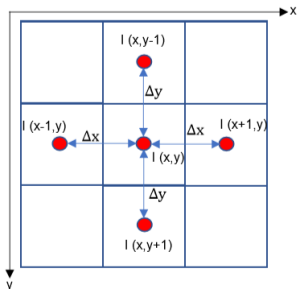


**Figure 13 Image (I) consisting of 3×3 pixels**

$$I(x,y) \approx \frac{1}{4}[I(x+1,y) + I(x-1,y) + I(x,y+1) + I(x,y-1)] + \Delta x^2[\nabla^2 I] \quad \textbf{(12)}$$

To achieve a unique solution, the Laplacian equation must be defined at the boundary. As such, by using the Dirichlet boundary conditions, pixels value around the outer perimeter of the image were set to zero. Lastly, the Laplacian equation with the new gradient dataset can be solved by using the Jacobi Iteration method.

### F. *Recovered Image*

By using the Jacobi Iteration method, it requires a long computational time since it is running on a loop, making it unsuitable for real-time applications. Thus, another method was used to recover another sample image. With the masked shadow binary image, by increasing the pixels' values in the masked area such that these pixels' values were matched closely with its surrounding.

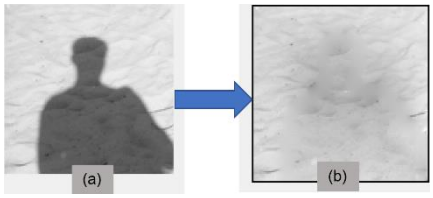


**Figure 14 (a) Original Image (b) Recovered Image**

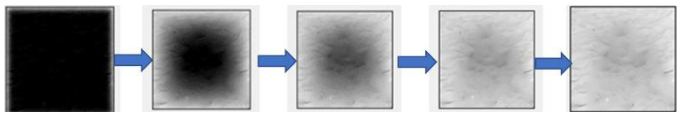

**Figure 15 Output at different iterations**


**Figure 16 Amplification of the masked area (a)without median filter (b) with median filter**

## V. Line Formation Algorithm

### A. *Working principle of 2D camera*

By implementing the focal length formula for the 2D camera, it is possible to acquire the actual distance from the camera on-board the follower UAV to the leader UAV. Since, the focal length of the on-board camera is unknown, a simple experiment was carried out to obtain the focal length using the following equation:

$$F = \frac{(P \times D)}{W} \quad \textbf{(13)}$$

$F$ = Focal length of the camera
$P$ = Width of the object as seen in the image (pixels)
$D$ = Actual distance away from camera to object
$W$ = Actual width of the object

According to the experiment carried out, the field of view (FOV) of the camera is approximated to be around 54 degrees. Therefore, the angle between the follower and leader UAV is given by:

$$\alpha = \frac{\text{centroid of bounding box}}{0.5 \text{ of the width of image}} \times 27° \quad \textbf{(14)}$$

### B. *Line formation algorithm*

Before moving into the equations, two important assumptions are made. Firstly, all the required information used for the formation is obtained only from a monocular camera. Secondly, the formation is achieved only by visual information without any forms of communication between UAVs.

With the relative distance, $l$ and angle, $\alpha$ obtained from the monocular camera as mentioned earlier, it is possible to express the coordinate position of the leader with reference from the follower.

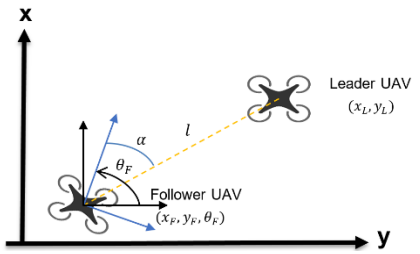


**Figure 17 Initial position for leader and follower UAV in 2D**

From Figure 17, the relative position of the leader with respect to the follower can be represented as $X_r = l\cos\alpha$ and $Y_r = l\sin\alpha$ where $X_r$ and $Y_r$ is the position of the leader in the coordinate frame of follower.

Similarly, the follower velocities can be approximated as $\dot{x}_F = v_F \cos\theta_F$ , $\dot{y}_F = v_F \sin\theta_F$ , and $\dot{\theta}_F = w_F$ . $v_F$ is the linear velocity of follower, $w_F$ is the angular velocity of follower, and $\theta_F$ is the pose of the follower.

By using the concepts above, the position of the leader is shown below as:

$$\begin{pmatrix} x_L \\ y_L \end{pmatrix} = \begin{pmatrix} x_F \\ y_F \end{pmatrix} + \begin{pmatrix} \cos\theta_F & -\sin\theta_F \\ \sin\theta_F & \cos\theta_F \end{pmatrix} \begin{pmatrix} x_r \\ y_r \end{pmatrix} \quad \textbf{(15)}$$

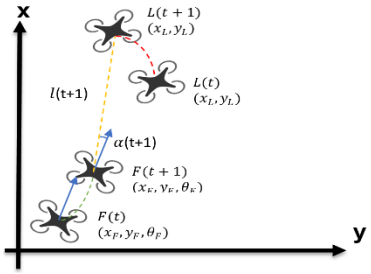


**Figure 18 Leader & Follower formation at time t+1**

Referring to Figure 18, the position of follower and leader is changing from time $t$ to time $t+1$ accordingly. As a follower, it is essential to estimate the velocities of the leader to keep a constant distance with it.

$$\dot{x_L} \cong \frac{1}{\Delta t}(x_L(t) - x_L(t-1)) \quad \textbf{(16)}$$

$$\dot{y_L} \cong \frac{1}{\Delta t}(y_L(t) - y_L(t-1)) \quad \textbf{(17)}$$

Regarding Figure 20, the relative distance from follower to leader is $l$ which is defined as:

$$l = \sqrt{(x_L - x_F)^2 + (y_L - y_F)^2} \quad \textbf{(18)}$$

$$\alpha = \arctan(y_L - y_F, x_L - x_F) - \theta_F \quad \textbf{(19)}$$

Then, the output of the closed loop control systems is $y_l = [l\ \alpha]^T$. Taking the derivative of it, $\dot{y_l} = G_1 u_l + r_l$,

$$G_1 = \begin{pmatrix} -\cos\alpha & 0 \\ \frac{1}{l}\sin\alpha & 1 \end{pmatrix}$$

$$r_l = \begin{pmatrix} \dot{x}_L \cos(\theta_F + \alpha) + \dot{y}_L \sin(\theta_F + \alpha) \\ \frac{1}{l}(\dot{y}_L \cos(\theta_F + \alpha) - \dot{x}_L \sin(\theta_F + \alpha)) \end{pmatrix}$$

$$u_l = [v_F\ w_F]^T$$

As this is a non-linear system, feedback linearization able to apply to the system, the velocity for follower is

$$u_l = {G_1}^{-1}(y_a - r_l)$$

$$y_a = \begin{pmatrix} k_1(l^d - l) \\ k_2(\alpha^d - \alpha) \end{pmatrix}$$

$y_a$ = auxiliary control input
$k_1$ and $k_2$ are desired controller gain
$l^d$ = desired length to keep with the leader in the formation
$\alpha^d$ = desired angle between the leader in the formation

One assumption was made that $|\alpha|$ is smaller than the field of view of the monocular camera onboard. As it is close loop system, the output $y_l$ will ensure that the system is more stable since there is error signal feedback into system to do correction.

## VI. Quadcopter Dynamics Control Model

The outputs of the line formation are the commanded flight velocity which is dependent on the desired distance to keep from its leader & the desired yaw rate which depends on the desired angle to keep from its leader. Lastly, the line formation works in 2D for now. Therefore, individual UAV needs to maintain at the same altitude during formation.

Thus, in this section, equations would be derived to make the quadcopter be able to control its velocities and hover at the required altitude.

### *A. Rotational Equation of Motion*

Quadcopter's rotational moments can be generated by creating differential thrust between its four propellers. Thrust produced by each propeller $= K_T . \omega^2$, with

$K_T = \rho \cdot D^4 \cdot C_T \cdot \left(\frac{1}{2\pi}\right)^2$,

ρ = Density of Air,
$n$ = Propeller rotation speed (rev/s),
$D$ = Propeller diameter,
$C_T$ = Thrust coefficient,
$\omega$ = Angular velocity (rad/s)

The summary of Force and Moments are shown by (20).

$$\begin{bmatrix} T^b \\ I_{xx}\dot{p} \\ I_{yy}\dot{q} \\ I_{zz}\dot{r} \end{bmatrix} = \begin{bmatrix} -K_T & -K_T & -K_T & -K_T \\ -K_T.\iota_{arm} & K_T.\iota_{arm} & K_T.\iota_{arm} & -K_T.\iota_{arm} \\ K_T.\iota_{arm} & -K_T.\iota_{arm} & K_T.\iota_{arm} & -K_T.\iota_{arm} \\ K_T.\iota_{arm} & K_T.\iota_{arm} & -K_T.\iota_{arm} & -K_T.\iota_{arm} \end{bmatrix} \begin{bmatrix} {\omega_1}^2 \\ {\omega_2}^2 \\ {\omega_3}^2 \\ {\omega_4}^2 \end{bmatrix} \quad \textbf{(20)}$$

$\iota_{arm}$ = Motor arm length from C.G.
$I$ = Moment of Inertia about the respective axis.

By inverting the matrix above, it determines the angular speeds required for each propeller to generate the desired thrust force and moments inputs.

$$\begin{bmatrix} {\omega_1}^2 \\ {\omega_2}^2 \\ {\omega_3}^2 \\ {\omega_4}^2 \end{bmatrix} = \begin{bmatrix} -K_T & -K_T & -K_T & -K_T \\ -K_T.\iota_{arm} & K_T.\iota_{arm} & K_T.\iota_{arm} & -K_T.\iota_{arm} \\ K_T.\iota_{arm} & -K_T.\iota_{arm} & K_T.\iota_{arm} & -K_T.\iota_{arm} \\ K_T.\iota_{arm} & K_T.\iota_{arm} & -K_T.\iota_{arm} & -K_T.\iota_{arm} \end{bmatrix}^{-1} \begin{bmatrix} T^b \\ I_{xx}.\dot{p} \\ I_{yy}.\dot{q} \\ I_{zz}.\dot{r} \end{bmatrix} \quad \textbf{(21)}$$

Using the derived inversed matrix, it is possible to control its movement in all three axes. Now, by applying a random input to the matrix, the results are shown below:

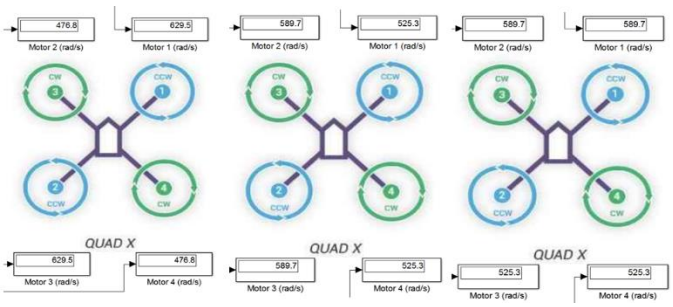


**Figure 19 (a) Yaw right (b) Roll right (c) Pitch backwards**

### B. *Velocities and Altitude hold control algorithm*

$$\begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix}^b = \begin{bmatrix} T^b sin\theta \\ 0 \\ T^b cos\theta \end{bmatrix}$$

$F_x$ is the force that accelerates the Quadcopter along the $x$-axis. By assuming the pitch angle to be small,

$$F_x = T^b sin\theta \approx T^b \theta$$

The $x$-axis velocity in the body frame can be controlled by a proportional control law.

$$F_x = m_{UAV} K_p \left(Velocity_{x(desired)} - Velocity_{x(actual)}\right)$$

By equating both equations together,

$$\theta = \frac{m_{UAV}}{T^b} K_p \left(Velocity_{x(desired)} - Velocity_{x(actual)}\right) \quad \textbf{(22)}$$

Similarly, for $y$-axis velocity control:

$$\phi = \frac{m_{UAV}}{T^b} K_p \left(Velocity_{y(desired)} - Velocity_{y(actual)}\right) \quad \textbf{(23)}$$

Lastly, for altitude hold, the equation is given by:

$$\begin{aligned} Thrust\ Required = {} & K_p (Z_{desired} - Z_{actual}) \\ & + K_I \left( \int (Z_{desired} - Z_{actual}) dt \right) \\ & + Hover\ Thrust \end{aligned} \quad \textbf{(24)}$$

$K_p$ = Proportional gain,
$K_I$ = Intergal gain,
$Z_{desired}$ = Desired Altitude,
$Z_{actual}$ = Measured Altitude

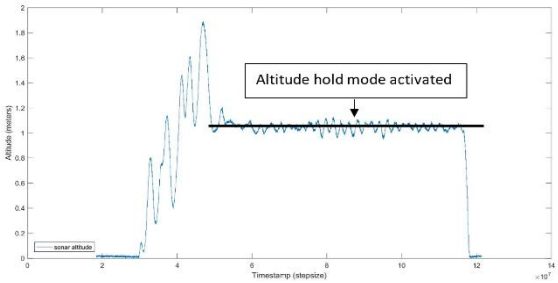


**Figure 20 Quadcopter performing an altitude hold at 1m.**

### C. *Overview of the quadcopter flight controller*

With all the required equations for the line formation, the flight controller was subsequently built on Simulink.

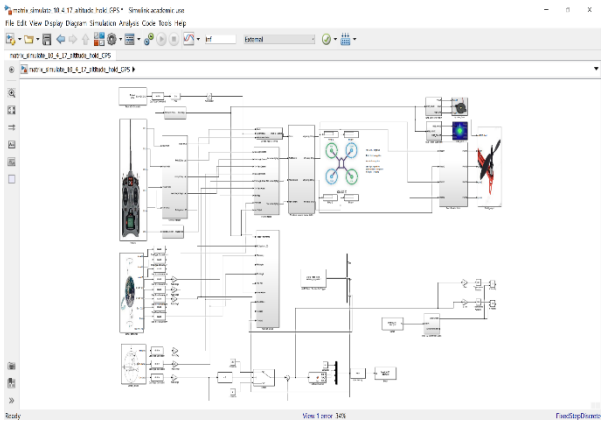

**Figure 21 Overview of Flight controller using Simulink**

## VII. Simulation Results

Tests were carried on individual algorithms and the results as seen in Figure 22 had shown the desired outputs, therefore, these algorithms were integrated onto the quadcopter.


**Figure 22 (a) Success detection of UAV with the corresponding distance and angle (b) Line formation output the desired velocities to maintain formation.**

## VIII. Concluding Remarks

For future development, code optimization of the flight controller is the main priority which could be the reason for sluggish performance. Installation of a flow sensor and laser range finder to achieve better altitude holding. With the hardware integration fixed, a flight test will be carried out.

As the HOG detection rate is at around 84.2%, to improve the detection rate, more research will be done on the implementation of HOG with Support Vector Machine (SVM) instead of cascade classifier to determine if SVM could provide a higher detection rate. Lastly, due to the high computational time required for the shadow removal, thus it wasn't implemented into the UAV. Therefore, methods such as Fast Fourier Transform could replace the Jacobi iterations.

## Acknowledgement

This research work has been partially funded by the Singapore Institute of Technology and the University of Glasgow Singapore. Their generous support is greatly appreciated.

## References

[1] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, I-511 - I-518, 2001.

[2] R, Lienhart and J, Maydt. An extended set of haar-like features for rapid object detection. In Proceedings of the 2002 International Conference of IEEE on image processing, pp. I-900-I-903 Vol. 1, 2002.

[3] R. Lienhart, A. Kuranov and V. Pisarevsky. Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In Joint Pattern Recognition Symposium (pp. 297-304). Springer Berlin Heidelberg 2003.

[4] T. Ojala, M. Pietikainen and T. Maenpaa, Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. In IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 7, pp. 971-987, Jul 2002.

[5] Ms. Varsha Gupta and Mr. Dipesh Sharma. A Study of Various Face Detection Methods. In International Journal of Advanced Research in Computer and Communication Engineering vol.3, no. 5, May 2014.

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Vol. 1, pp. 886-893, 2005.

[7] L. Consolini, F. Morbidi, D. Prattichizzo and M. Tosques. Leader–follower line formation of nonholonomic mobile robots with input constraints. Automatica, 44(5), 1343-1349. 2008.

[8] N. Cowan. O. Shakerina. R. Vidal and S. Sastry. Vision-based follow-the-leader. In Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vol. 2, pp. 1796-1801, 2003.

[9] H. J, Min. A. Drenner and N. Papanikolopoulos. Vision-based leader-follower formations with limited information. In 2009 IEEE International Conference on Robotics and Automation, pp. 351-356. 2009.

[10] C. Fredembach and G. Finlayson, "Simple Shadow Removal", in 18th International Conference IEEE, 2006.

[11] Y.Matsushita, K. Nishino, K. Ikeuchi and M. Sakauchi, "Illumination Normalization with Time-Dependent Intrinsic Images for Video Surveillance," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, no. 10, pp. 1336 – 1347, 2004.

[12] G. D. Finlayson, S.D. Hordley, C. Lu and M.S. Drew, "On the Removal of Shadow in Image," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.28, no.1, pp. 59-58, 2006.

[13] T. Hakim and C. David. Implementation of HOG for Human Detection. Implementation of HOG for Human Detection, 2015, www.geocities.ws/talh_davidc/#cst_extract. Accessed 3 April. 2017.

[14] R. E, Schapire. Explaining adaboost. In Empirical inference (pp. 37-52). Springer Berlin Heidelberg. 2013.

[15] N. Hamdi, K. Auhmani, M. M. R. Hassani and O. Elkharki. An Efficient Gentle AdaBootst-based Approach for Mammograms Clasification. Journal of Theoretical and Applied Information Technology, 81(1), 138. 2005.

[16] X, Zhu, C. Vondrick, D. Ramanan and C. C. Fowlkes. Do We Need More Training Data or Better Models for Object Detection? In BMVC. Vol. 3, p. 5. September 2012.

[17] A. Rosebrock. (Faster) Non-Maximum Suppression in Python – PyImageSearch. PyImageSearch, 2015. [Online]. Available: http://www.pyimagesearch.com/2015/02/16/faster-non-maximum-suppression-python/. [Accessed: 10- Apr- 2017].