Flinders
UNIVERSITY
ADELAIDE AUSTRALIA

2017

Autonomous Take-off and Landing of a Small Drone from an Autonomous Surface Vessel

Visual Guidance

By: Wahyu Adi Prasetyono

ID: 2114725

Supervisors: A. Prof Karl Sammut

Dr. Greg Ruthenbeck

Mr. Jonathan Wheare

Submitted to the School of Computer Science, Engineering and Mathematics in the Faculty of Science and Engineering in partial fulfilment of the requirements of Bachelor of Engineering (Electronics) (Honours) at Flinders University- Adelaide, Australia

Date: October2017

Flinders University

School of Computer Science, Engineering and Mathematics (CSEM)

DECLARATION

I certify that this work does not incorporate without acknowledgment any material previously submitted for a degree or diploma in any university; and that to the best of my knowledge and belief, it does not contain any material previously published or written by another person except where due reference is made in the text.

_____ Author: Wahyu Adi Prasetyono

Signature 10/17/2017

ACKNOWLEDGEMENTS

In the name of Allah, the Most Gracious and the Most Merciful

Alhamdulillah, all praises to Allah for the strengths and His blessing in completing this thesis. Special appreciation goes to my Academic Supervisors, Prof Karl Sammut, Mr. Jonathan Wheare and Dr. Greg Ruthenbeck for their supervision and constant support. Their suggestions and comments throughout this year have contributed to the success of this research. Not forgotten, my appreciation to PAPUAN GOVERNMENT for providing me a chance to study in Australia. Thanks for your support and care through my time in Australia. I also would like to thank all people in Engineering Service for their support and help on the technical department for kindly providing me the requested parts and components; as well as, Ben Chartier from Nova Systems for sharing his experience regarding autonomous system.

In addition, I would like to thank my team mates, Stephen Keen and David Swincer, for their support and help throughout this years. It is been an amazing time. You guys are the best team mate I have ever had. Moreover, all my friends and family in Adelaide specially Jennifer Queen Korwa, Brian Alexander Unani, Alfred Mayor, Elisabeth Jakarimilena, Brendha Rumambi, Meiselman Buntulobo, and other for their kindness and moral support during my study. Thanks for the friendship and memories.

The last but not least, my deepest gratitude to my beloved parents: Mr. Sriyono and Mrs. Gerarda Boseren, and to my brother, Tommy Frenlie Boseren, and to my sister, Anissa Fitri Amalia for their endless love, prayers and encouragement.

Wahyu Adi Prasetyono, Flinders University, 10/17/2017

ABSTRACT

This project presents the autonomous landing of a UAV (Unmanned Aerial Vehicle) on a small platform. The project was divided into three parts, mission control, visual guidance, and navigation control. The objective of this research was to find a suitable method for landing a UAV using visual guidance. The chosen method was a combination of long-range and short-range guidance. The long-range guidance was handled by a combination of background subtraction and adaptive thresholding. It was used to guide the UAV from seven meters above the vessel down to one meters using a camera looking upward. After that, the system would switch to the short-range guidance that is handled by infrared detection using the IR LOCK sensor. The short-range guidance is used for guiding the UAV from one meter above to the landing pad. The research focussed on two things: research and implementation. The chosen method was decided after reviewing some existing methods. The guidance software was developed in the Robot Operating System (ROS) to match the existing platform of Flinders ASV (Autonomous Surface Vessel). The integration system test found that the IR lock sensor for the short range was not detected by the flight controller. As a result, all the tests were done by maximising the long-range guidance only and forcing the UAV to land when the altitude was less than 0.5 meter. The resultant test showed that with the absence of the short-range guidance, the long-range guidance was still able to perform the full landing process with a success rate of 30– 60% (middle range). The long-range guidance was relatively accurate and fast at correcting the position of the UAV when the UAV was above one meter, but very poor when the UAV is less than one meter. The overall project aim which is to have a fully autonomous UAV to take-off and land on small platform is partially done. However, the objective of this study which design and implement a visual guidance system for take-off and land of UAV has been achieved. Further research is definitely necessary to complete the task especially on the tracking side.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

AR TAG : Augmented Reality

ASV : Autonomous Surface Vehicle

CCD : Charge-Coupled Device

CMECI : Centre for Maritime Engineering, Control, and Imaging

CMOS : Complementary Metal-Oxide Semiconductor

DSP : Digital Signal Processing

DT-ATLS : Dual Thread-Automatic Take-off and landing System

FIR : Far Infrared

FOV : Field Of View

FPV : First Person Video

GPS : Global Positioning System

IBVS : Image-based Visual Servoing

IMU : Inertial Measurement Unit

IR : Infrared

LIDAR : Light Detection and Ranging

MIR : Mid-Infrared

NIR : Near Infrared

NLOG : Negative Laplacian of Gaussian

OPATS : Object Position and Tracking System

POSE : Position and Orientation

RGB : Red-Green-Blue

ROS : Robot Operating System

RTK : Real Time Kinematic

UAV : Unmanned Aerial Vehicle

UCARS : UAV Common Automatic Recovery System

UKF : Unscented Kalman Filter

USV : Unmanned surface Vehicle

WAM-V : Wave Adaptive Modular Vessel

1. INTRODUCTION

The overall aim of this project was to allow an Unmanned Aerial Vehicle (UAV) to autonomously take-off and landing from a small platform. This project was divided into three parts, which were mission control, visual guidance, and system navigation. This thesis focused on visual guidance for landing the UAV. The aim of this thesis was to find a suitable method for guiding the landing process of the UAV autonomously and to implement the method in the system.

Flinders University is developing Autonomous Surface Vessels (ASV) called TopCat under the auspices of the Centre for Maritime Engineering, Control, and Imaging (CMECI) as a research platform for maritime autonomy, situational awareness, and environmental monitoring. As a regular contestant of the Maritime RobotX Challenge, the TopCat vessel is developed as generalised system, which is not only limited to for a research platform, but also for the competition (Sammut et al., 2016)

The TopCat vessel is equipped with several cameras for visual feedback and monitoring; however, due to the low structure of the TopCat vessel, horizontal visibility from the vessel is limited. This limitation is also directly impact the long distance manual navigation that relies on the visions. Information about aerial perspective over the environment can be used to solve this problem. A simple way to obtain the information from aerial perspective over the environment is from satellite. However, most data from satellites are suffering from outdated unless the provider can guarantee that the data is updated periodically or pay licenses fee.

Flinders researchers resolve that problem by teaming up the TopCat vessel with an aerial vehicle called CAT'SEYES. This aerial vehicle will act as a surveillance aircraft that will take-off, fly, and collect up-to-date data from an aerial perspective. This combination of both vehicles is great, but most aerial vehicles are limited by their flight time, which results in a reduction of the coverage range if deployed from the shoreline. To extend the operating time and expand the coverage area, the aerial vehicle can be piggybacked by the ASV. The ASV will act as a carrier vehicle, which not only transports the aerial vehicle, but also provides a base station and extra power to the aerial vehicle. As a result, the aerial vehicle can be deployed at anytime and anywhere multiple times. In addition, by piggybacking the ASV, both the aerial vehicle and the ASV can do several tasks concurrently; hence, the functionality and the efficiency are increased.

To enable such ability, the aerial vehicle is required to be semi-autonomous to allow auto and manual deployment and controlled when needed. However, the UAV has to be autonomous specifically on take-off and landing to remove the dependence on a human pilot since everyone has a different ability to operate aerial vehicles using remote.

Due to the safety, size, and payload of the TopCat, hobby-sized aircrafts (remote control aircraft) such as fix-wing (plane), single rotor (helicopter), and multirotor (drone) are the most suitable size. For the project, our group decided to use a multirotor type quadrotor as the UAV (Keen, 2017).

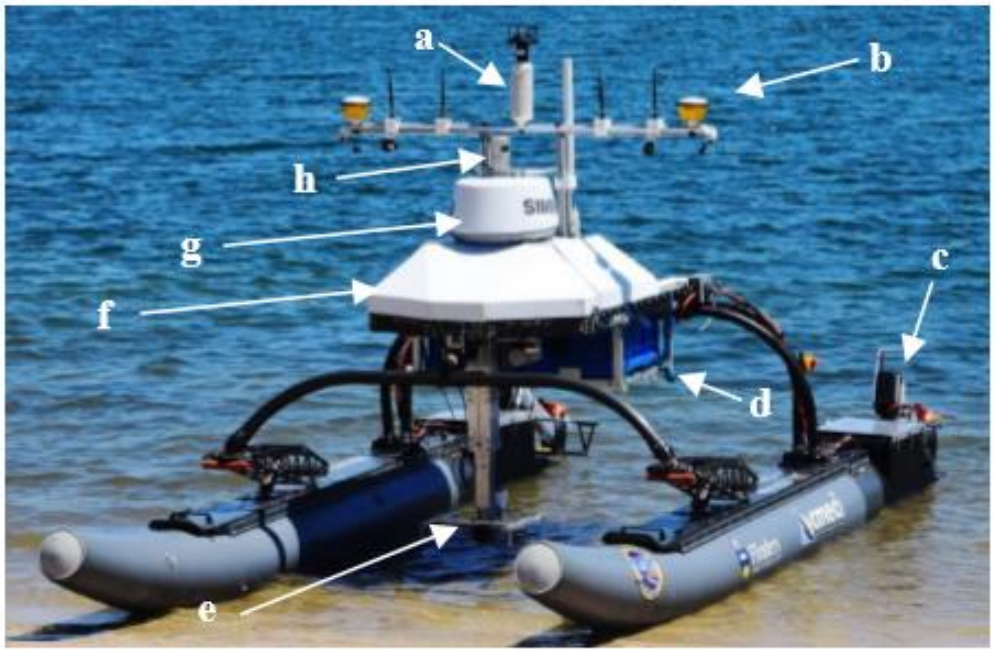
1.1. System overview of QUAD

The following Figure 1.1 shows the Quadrotor design that is used for this project. The aircraft that is going to be used in this project is a 400mm size quadrotor with weighting approximately 1.4kg including 5200mAh battery. The UAV is capable of carrying a payload up to 400grams with a flight time around 12 minutes. The flight controller is the PIXHAWK flight controller. The quadrotor is also equipped with the Ublox Neo-7M DIGITAL GPS with built in compass with a refresh rate of 10Hz. Moreover, for communication, the UAV is equipped with two 915MHz telemetry radio for communication, a 5.6GHz transmitter for streaming live video, and a 2.4GHz receiver remote controller. This configuration is adequate for the UAV to talk to TopCat vessel (USV) and broadcast its condition to the base station concurrently. In the same time on separate frequency, the UAV will transmit live streaming First Person View (FPV) video for observation. This functionality is reserved for future development. The 2.4GHz receiver remote control is available for emergency. Further detail on vehicle design can be found on navigation control report (Keen, 2017).

Figure 1.1: CAT'SEYES' design

1.2. System overview of TopCat

The Flinders Autonomous Surface Vehicle known as TopCat uses innovative Wave Adaptive Modular Vessel (WAM-V) as the base platform. As shown in Figure 1.2, the system is powered by two 3.88kWh LiIon batteries. Excluding the engine system and only running the sensor, the batteries can last about 2 days from fully charged, but it is reduced to approximately 5 hours when all the sensors and the engine are running. The engine system consists of a pair of Torqeedo Cruise 2R electric motors driven by Actuator Control Module (ACM), which consists of a Diligent Cerebot MC7, a 24V DC/DC converter, and RS4485 PMOD. The ASV also equipped with dual antennas Trimble BX982 GPS for navigation system, a Simrad 4G maritime navigation radar, and a Velodyne HDL-32 LIDAR for obstacle detection system. Besides, the ASV also equipped with several cameras for vision task under water and above water (Sammut et al., 2016). The overall system is developed in Robot Operation System (ROS) Environment (DirkThomas. 2017).



Key:	
a: central mast	e: underwater sensor deployment system
b: sensor and comms array	f: computing and electronics
c: propulsion system	g: marine radar
d: battery system	h: velodyne lidar

Figure 1.2: Current TopCat's design (Sammut et al., 2016)

1.3. Autonomous landing Problem

Take-off, climb, cruise, descent, and landing are the phases of flying aircraft. Of all these phases, it is undoubtable that landing is the most difficult task (Khithov et al., 2017, Gui et al., 2013). Pilots of all types of aircraft spend hours to practices and to perfect their landing skill. One of the major problems with landing an aircraft is there is only short period of time for the pilot to make a decision. Moreover, factors such as landing location, visibility, type of terrain, and wind disturbance increase dramatically the difficulty of making the decision. Those factors are difficult to predict because they vary from each flight (Gautam et al., 2014).

The same problems are also found in the autonomous landing of UAV. However, the problems become more complex due to the absence of a pilot. Accurately detecting the position and orientation of the landing platform is critical to land a UAV successfully especially when the landing platform has a narrow size. Global Positioning System (GPS) is an inexpensive sensor for tracking the position of an object. However, many consumer GPS sensors are inadequate for precisely tracking the position of the UAV (Yi-ChengLu, 2017).

In addition, limited payload was another major factor that has to be considered. Having multiple sensors on the UAV can deliver robust result, but it increases the weight of the UAV reducing the flight time dramatically. Besides, this extra weight could affect the dynamics of the UAV, which resulted in reducing the agility of the UAV.

The existing LIDAR sensor from the TopCat is expected will struggle to continuously tracking the drone while descending because the LIDAR has 1.25º gap between its lasers (HDL-32E. 2017. HDL-32E) which opens the possibility of the drone will slip through between the lasers. In addition, the lidar also has several blind spots on the backside due to the antenna structure of the TopCat.

1.3.1. Advance system for Autonomous Landing of UAV

A number of researchers have investigated autonomous landing of UAV using various approaches. The most advanced methods for autonomous landing are currently being developed by the military for defence purposes such as Object Position and Tracking System (OPATS), UAV Common Automatic Recovery System (UCARS) and Dual Thread-Automatic Take-off and landing System (DT-ATLS). Those methods rely on modern sensors such as laser pointers, millimeter wave track radar, and combination of millimeter wave track radar and differential GPS (Kong et al., 2014).

Object Position and Tracking System (OPATS) is an automatic UAV landing system that uses a laser pointer sensor to measure the dynamic position of the object of interest continuously (RUAG Group, 2017). The UAV Common Automatic Recovery System (UCARS) is a ground based radar system that locates and tracks the position of aircrafts that have been fitted with an airborne transponder. This method allows aircraft to take-off and landing in any weather conditions such as night, fog, rain, and most sea conditions (SIERRA NEVADA CORPORATION, 2017a). Dual Thread-Automatic Take-off and Landing System (DT-ATLS) is a combination system between radar-based millimeter wave, Take-off and Landing System (TALS) and Differential GPS (DGPS). The system provides sub-meter accuracy to the object of interest and redundancy system configuration for GPS denied environment (SIERRA NEVADA CORPORATION, 2017b and Harker, R. & Gilligan, J., 2006).

Although the advanced systems above have been proven that they can provide great performance and are highly reliable, their methods are not targeted for hobbyist size quadrotors because the equipment that is used tends to be bulky, heavy, power hungry and expensive. The weight of an airborne transponder of DT-ATLS or UCARS is around 1.4 – 1.8kg and it requires an 18-28VDC power source to run (SIERRA NEVADA CORPORATION, 2017a; 2017b).

1.3.2. Classical method for autonomous landing of UAV

Besides using advanced sensors above, there are many other researchers investigating inexpensive solutions of this problem. Many of them are using visual based detection method. The idea is to use cameras to visually detect the position of the UAV in the camera's field of view (FOV). The classical method of detecting is to use feature extraction and detection. Weaver et al. proposed landing platform detection using colour detection to detect the edge of a landing pad, which had been pained with a specific colour, using camera looking downward. For tracking, they used standard Kalman filter to recover the UAV from short occlusion and two staged PID controllers to interact with the controller to stabilise the UAV position with respect to the landing platform (Weaver et al., 2013).

A similar method was also proposed by Kim et al. (2014). The authors approached this problem using colour filtration to detect the colour of the whole landing pad. In addition, they added a filter, which removes any detected objects with size less than the requirement, to remove the possibility of getting multiple detected objects if there are objects that had similar colour. They used a smartphone as a downward looking camera and the processing unit for image processing. In addition, to reduce the possibility of losing the target when descending, they

modified the normal smartphone camera lens to be an omnidirectional fish-eye lens. On the top of that, they applied an Unscented Kalman Filter (UKF) to track the detected object continuously.

Unlike previous colour filtering, which is only filtering visible colours, Wenzel et al. (2011), Gui et al. (2013), and Khithov et al. (2017) designed non-visible colour filtering. The colour is emitted by infrared LEDs (IR beacon).

Wenzel et al. used a Wii remote infrared (IR) camera to detect IR LEDs beacon, which were located on the landing pad. The method allowed a UAV to land on a moving carrier. However, the method was not applicable to an outdoor environment due to interference from the sun

Gui et al. (2013) conducted slightly similar research with better equipment. A Digital Signal Processing (DSP) camera was used for object detection. The camera was modified to be sensitive to 940nm (NIR) wavelength light only. With this modification, they were able to detect an IR beacon in outdoor environment under sunlight with less interferences.

Furthermore, infrared detection method was demonstrated by Khithov et al (2017). Their method was to detect beacons with higher infrared wavelength around 3 – 8μm (Mid-Infrared (MIR)). The simulation that was conducted showed that the method was functional, but there was no actual real world testing yet.

Coming from a different point of view, Saripalli et al. (2002) conducted research using an "H" sign as the marker on the landing pad which will be detected by downward looking camera from the UAV. Moment inertia comparison was used to find the region in the image which was closest to the marker. The method was proven practical in outdoor environment.

Similar pattern detection was shown by Lange et al. (2009). The pattern was a collection of concentric white circles on the black background. Each ring is made with the same ratio between the inner and the outer sides; thus, the ring can be detected independently. As a result, if some of the rings are not visible, the other which visible will allow the algorithm to work. In addition, by increasing the number of the circles, the detection can be done in higher altitude.

Other pattern style is augmented reality (AR) tag. It was done by Lee et al. (2012) and Chandra et al. (2016). For detecting this pattern, they use build in library, which available on open source program such as ROS and OpenCV.

Different approach for landing an autonomous aerial vehicle was taken by Pinto et al. (2014) and Yi-ChengLu (2017). While many people use on board camera to detect the position of the landing pad from the UAV, they used camera looking upward from the landing pad to detect the position of the drone from the landing pad. Both methods have been proven works in the real time testing.

As describe in the previous works above, visual based method is the most suitable method for this project because they are tend to be cheaper and easy to implement while maintaining good performance. There are many different approaches of visual based method for autonomous landing as describe before. The research question that is need to be answer is what are the most suitable methods for our system and how do they are implemented them in the given time constraints?

2. LITERATURE REVIEW

OpenCV and Matlab are two of a few tools that can be used for simplifying image processing. These two platforms have a built in library, which has been optimised for image processing. OpenCV is an open source library, which can be used in programing language such as Java, C, C++ and Python. On the other hand, Matlab is a non-free software, which is commonly used in the engineering field. Both tools implement image-processing function such as image thresholding, morphological operation, and other mathematical operation associate with matrices.

Regarding image processing, every image is represented as a matrix in which each element of the matrix represents a pixel, the value of pixel represents the intensity of the pixel, and the size of the matrix represents the resolution of the image. Coloured images have a three-dimension matrix compare to grayscales images, which only have one dimension. As known, images from a camera are sometimes subjected to some noise due to camera sensor quality, camera's lens, different lighting conditions, blurring, etc. For example, the colour of an object is affected by the amount of the light received by the object. Less light will make the object appear dark while too much light makes the object appear white. Due to this inconsistency, correction or manipulation must be made before doing any image processing or object recognition. The following section outlines several techniques to correct and manipulate images.

2.1. Image correction and manipulation

2.1.1. Brightness and contrast adjustment using Constant multiplication

Enhancing the contrast of the image can correct the inconsistency of an image. Correcting image brightness and contrast can be done by multiplying and adding each pixel value with some constant. The implementation is done by creating three nested "for loops" which visit every pixel value on every dimension and perform the calculation accordingly. The mathematical expression (Changing the contrast and brightness of an image! — OpenCV 2.4.13.4 documentation. 2017) of this method is shown below.

$$g(x) = \alpha f(x) + \beta$$

Where:

- α is value bigger than zero. It is called gain, which controls the contrast of the image.
- β is real number. It is called the bias, which controls the image brightness

2.1.2. Brightness and contrast adjustment using Histogram equalisation

Enhancing the contrast can also be done through histogram equalisation of the image. This process is known as global equalisation. Through this process, the intensity of image pixels will be redistributed uniformly across the entire intensity range that results in contrast amplification on the regions of the image that has low contrast. This calculation is implemented in the OpenCV function called "equalizeHist(Mat src, Mat dst,)" (OpenCV: Histograms - 2: Histogram Equalization. 2017). The math expression of this process is:

$$y = (L-1) * \int_0^x p_n(w)dw$$

Where:

- y is new pixel value,
- $(L-1)$ is range of the pixel intensity,
- $p_n = \frac{number\ of\ pixels\ with\ intensity\ n}{total\ number\ of\ pixels} \quad n = 0,1,2,3 \ldots, L-1$

Besides using global equalisation, there is also adaptive histogram equalisation (AHE). Unlike global equalisation, which applies globally to the whole image, AHE applies histogram equalisation in small region scale that results in a much smoother contrast enhancement. However, some AHE method such as general AHE and contrast limited AHE (CLAHE) are very computationally expensive due to the fact that the equalisation calculation is done on each pixels in the image.

Figure 2.1 shows the difference between the global equalisation and adaptive equalisation. As can be seen the result from adaptive equalisation is more natural than the global equalisation.

A B C

Figure 2.1: comparison between global equalisation with adaptive equalisation. (A) Original image, (B) Global equalisation, (C) adaptive equalisation (OpenCV: Histograms - 2: Histogram Equalization. 2017)

2.1.3. Noise reduction using Gaussian blurring

If the previous techniques are to sharpen the image, this process is to smooth the image. Small blobs with high contrast can be a problem when detecting the shape of the object through contour detection because most of those noises will easily be detected. The Gaussian filter is one of the most used filter for performing this type of operation. It works by convolving each point in the input array with a Gaussian kernel, and then summing them all to produce the output array. The mathematical operation of this filter is shown as below where μ is the peak of Gaussian kernel and σ is the variance of each inputs x and y (Smoothing Images — OpenCV 3.0.0-dev documentation. 2017).

$$G_0(x,y) = Ae^{\frac{-(x-\mu_x)^2}{2\sigma_x^2}+\frac{-(y-\mu_y)^2}{2\sigma_y^2}}$$

Example of Gaussian blur on image is shown in the Figure 2.2 below. Setting the σ as 500 and applying different kernel size result in difference blurring affect. The most left side is the original image and as it slides to the right, the kernel increased form 1, 5, and 9.

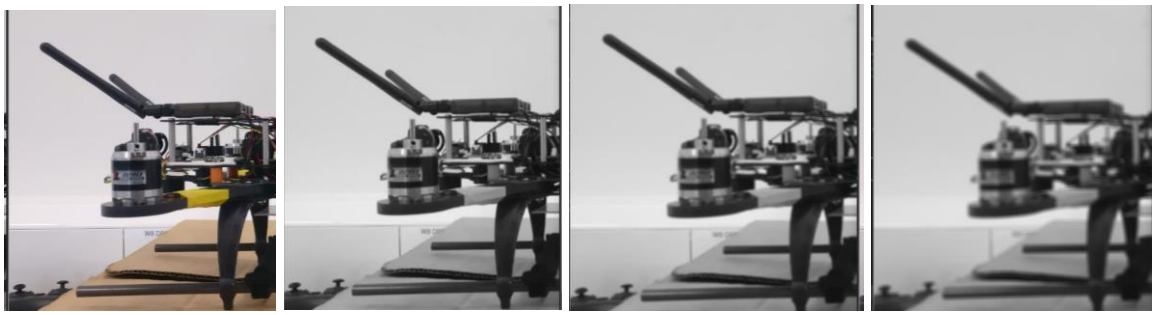
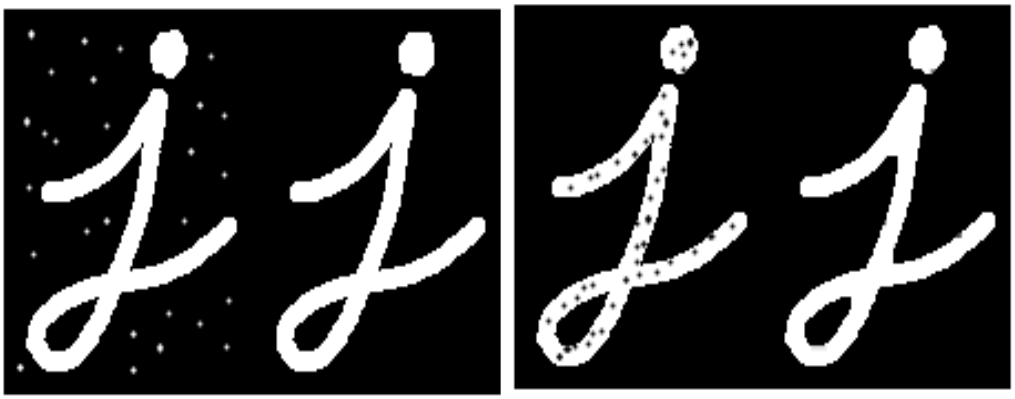


Figure 2.2: Gaussian blurring

2.1.4. Noise reduction using Morphological operation

It is possible to use Gaussian blur on colour image or binary image; however, the result is only to reduce the sharpness of object. Especially on binary images, the noises usually tend to expand. Morphological operation is another technique to manipulate images. The operation is similar to filtering which is multiplying the source image by a kernel. Two basic morphological operations are Erosion and Dilatation. Depending on the way the operation is performed, the process might be able to remove small blobs or enlarge small blobs so it can merge with other small blobs creating bigger blobs. For noise reduction, opening and closing are the most useful variations. Opening is the variation which is applying the erosion followed by the dilatation and closing is opposite of the opening. Figure 2.3 shows the effect of opening and closing on

separate problem. Opening is process to remove white dots in the frame while closing removing black dots in the frame.



a1 a2 b1 b2

Figure 2.3: visualizes the effect of opening and closing operation of morphological operation. (a1) original image, (a2) result image after opening, (b1) original image, (b2) result image after closing (OpenCV: Morphological Transformations. 2017).

2.2. General tracking methodology

Yilmaz et al. (2006) describes that Object representation, feature selection, object detection, and object tracking are steps to create visual tracking system. Object representation is the steps to determine how the object is represented after being detected for future analysis. There are multiple ways of representing an objects (target). Figure 2.4 illustrates different types of object representation that are commonly used. The simpler the representation of the desired object, the easier the algorithm to execute. However, depending on the type of the future analysis after detecting, some type of representation can give more information than the others. For example, for analysis of human movement, representing the target as skeleton (Figure 2.4.f) will give more information about specific body movements compared with representing the target as single point (Figure 2.4.a). However, single point representation is more efficient for tracking the position of the whole body (Yilmaz et al., 2006).

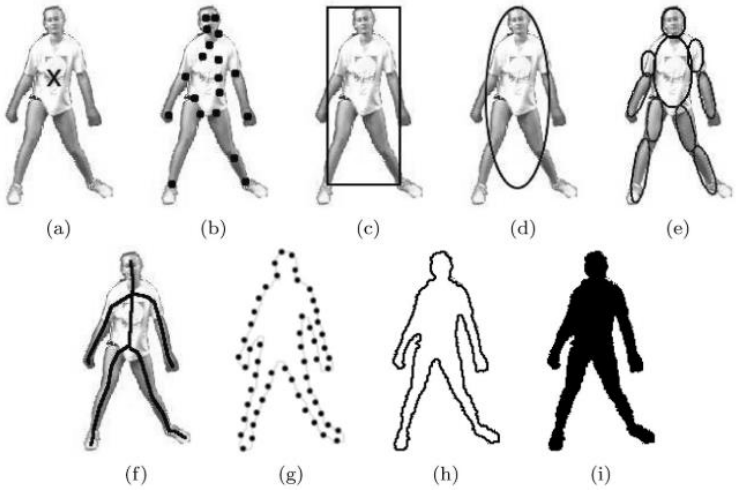


Figure 2.4: Representation of object. (a) Single point , (b) multiple points, (c) rectangular patch, (d) elliptical patch, (e) part-based multiple patches, (f) object skeleton, (g) complete object contour, (h) control points on object contour, (i) object silhouette (Yilmaz et al., 2006).

The next is feature selection. From the list of previous approaches in the introduction, many existing algorithms have been implemented for recognising objects automatically. Those algorithms vary depending on the object of interest (target) which can be the landing pad, the whole boat itself, or the UAV. Having a target that can be easily distinguished (recognised) from other objects or the complex environment is critical. Features of objects that can be extracted by cameras are colour, shape, and texture (Tian, 2013)

Object detection is the next stage where the objects on the frame are classified. This process can be done in two ways, manual and automatic. Manual object detection lets humans to do the detection during the process while automatic lets the algorithm do the detection based on information it is given. There are two types of information, fix information and adaptive information. Fix information is information generated from the first frame and never changes until the end of the tracking process while adaptive information generates temporal information which is calculated in every scene. A few methods of object detection are segmentation, background modelling, point detection and supervised learning (Yilmaz et al., 2006).

The last but not least is object tracking. The aim of object tracking is to establish correspondence between a detected object and a tracker. A tracker is a container that explains the object movement at any given time. This process can be done by having a container that records the motion of the detected object separately. Another way of doing object tracking is by estimating the motion of the object using information from previous frames. Besides relying fully on visual feedback, adding position sensor such as Global Positioning System (GPS),

inertial Measurement Unit (IMU), and Light Detection Ranging (LIDAR), can improve the accuracy of tracking (Yilmaz et al., 2006).

The benefits of having object tracking is when updating the motion of the object, it preserves the identity of the target from frame to frame. By understanding the motion of the target, the detection process can be narrowed around the estimated location. Hence, the detection process can be sped up due to the smaller search area. Besides, with a proper estimator, short events of occlusion on the target can be recovered. Moreover, certain estimators preserve the identity of the target; as a result, those estimators are capable of distinguishing the real target in the event where there are multiple objects that a have similar appearance to the target.

2.3. Existing Visual tracking for landing UAV

2.3.1. Visual tracking using Colour extraction

The paper written by Kim et al., (2014) showed an example of autonomous landing UAV using colour extraction. Their system used a smartphone as a downward looking camera as well as the processing unit for visual. Their system utilized colour as a feature to detect from the object. The landing pad was fully painted red to make it stand out from its environment. The algorithm proposed by Kim et al. (2014) implemented colour detection on RGB colour space. The algorithm has two parts: grouping and masking. By comparing pixels' value in one channel with its value in the other channel, Kim finished the first algorithm with three matrixes (I_{RR}, I_{GG}, I_{BB}). Each of these matrices contains the location of pixels with the highest intensity on each channel (RGB). The algorithm then continued with subtracting each matrix from I_{RR}. The pseudocode of the algorithm and the algorithm result is shown in the following Figure 2.5. The extra I_{Temp} is to reduce the chance of having yellowish red. After performing colour extraction, the object with the largest contour will be set as a target.

After being able to identify the landing pad, a single point was used to describe the whole landing pad that will be tracked by an Unscented Kalman Filter (UKF). This is used since the motion of the landing pad is nonlinear. On their test, they successfully identified and landed on a moving platform in an outdoor environment.

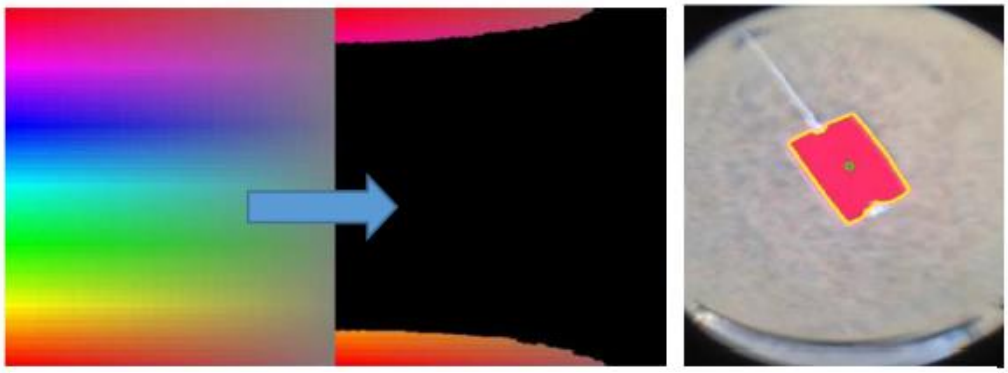
Algorithm 1.
Input: (I_1, I_2, I_3), where each I_k is a matrix of an 8-bit, 1 channel image frame.
Output: A filtered binary image frame matrix, I_{1b}

getBinaryImageofColor(I_1, I_2, I_3)
  For every pixel value $I_{k,ij}$ of i-th row and j-th column in the matrix I_k
    If $I_{1,ij} \geq K_I(I_{2,ij} + I_{3,ij}) + C_I$ then
    $I_{1b,ij} \leftarrow 1$
    Else
    $I_{1b,ij} \leftarrow 0$
    endIf
  endFor
return I_{1b}

Algorithm 2.
Input: (I_R, I_G, I_B), where I_R, I_G and I_B are split matrices of 8-bit, 1 channel image frames of red, blue and green color from the 3-channel RGB image.
Output: A filtered binary image frame matrix, I_{CF}

colorFilter(I_R, I_G, I_B)
  I_{RR} = getBinaryImageofColor(I_R, I_G, I_B)
  I_{GG} = getBinaryImageofColor(I_G, I_B, I_R)
  I_{BB} = getBinaryImageofColor(I_B, I_R, I_G)
  For every pixel value $I_{k,ij}$
    If $0.6I_{R,ij} < I_{G,ij}$ then
    $I_{Temp,ij} \leftarrow 1$
    else
    $I_{Temp,ij} \leftarrow 0$
    endIf
  $I_{CF,ij} \leftarrow I_{RR,ij} - I_{GG,ij} - I_{BB,ij} - I_{Temp,ij}$
  endFor
return I_{CF}

A. Algorithm

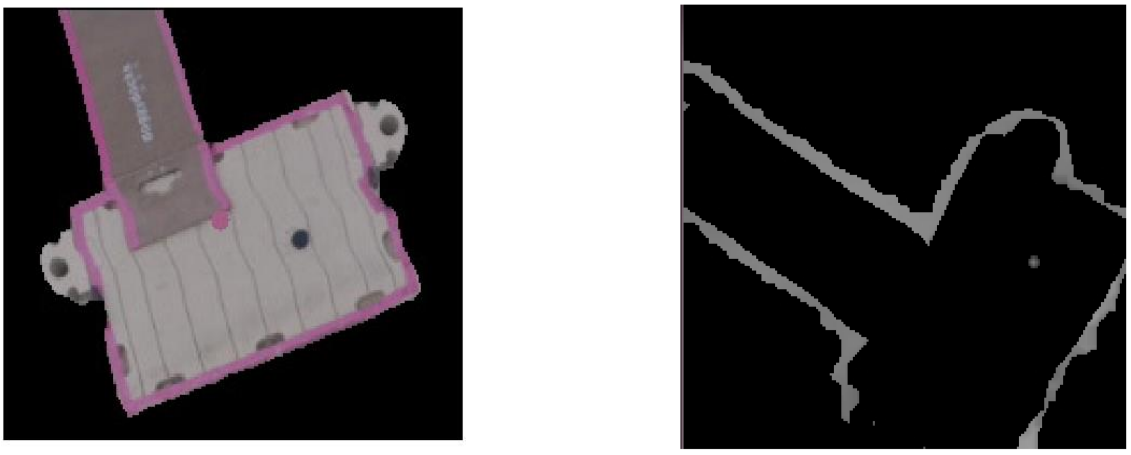


B. Test result

Figure 2.5: Colour Filter algorithm and test result of Kim purposed method (Kim et al., 2014)

A similar colour extraction method was also described by Weaver et al. (2013). The scenario that they tried to handle in this research was launching and recovering where the UAV deployed from a landing pad and flown following Global Positioning System (GPS) waypoints to pick up an object and then flown back to the landing pad. To achieve this problem, the UAV was required to do two different visual processing. The first was to identify the item to be picked-up using blobs detection and the second was to identify the landing pad using colour detection. For both processing, they used a quad core ARM ODROID-U2. For the landing pad detection, a colour extraction and detection is run on the acquired image to filter the border of the landing dock. The image was acquired using Logitech HD pro webcam C920, which was set to look downward. All the systems were located on the UAV for local processing. Once the border colour was successfully extracted, the system then calculated the nearest distance between the borders of the landing pad and a small dot in the middle of the landing pad to determine the

heading of the UAV. The illustration of the image after colour filtration is shown by the Figure 2.6. The final image obtained from the system is shown by the Figure 2.6.B. Furthermore, the standard Kalman filter was used by Weaver et al. (2013) to recover from short occlusion by brings the UAV to an expected area (Weaver et al., 2013).



(A) *(B)*

Figure 2.6: Weaver et al. colour extraction for detecting the landing platform (Weaver et al., 2013). (A) Border colour extracted and labelled, (B) Border edge extracted.

Having successful real world testing in outdoor environment is a good indication that the method can work under different light intensity. Comparing both colour extraction methods, extracting the whole landing pad is more robust than just the border since the colour to detect from the entire landing pad appears to be stronger than just the border. In addition, with a bigger target to detect, the higher the process of identifying can be done.

On the other hand, unless the environment is known, assuming the biggest contour is the landing pad is not a good approach since the probability of multiple object with similar characteristic is high. Small extra information such as dots or marks on the landing pad was a clever idea to distinguish the landing pad from the environment as well as to add extra information such as heading.

Moreover, since the test that was done by Kim et al. (2014) was on land, there is a possibility that the method may not be so robust against light reflection from the water. The water reflection might not only add glare to the lens but also shift the colour range of the landing pad. Hence, the detection will have more true negative detections than true positive detection. On the Weaver et al. (2013) test, although from the previous comparison it is known that detecting

just the border of landing pad is not so robust compared to the whole landing pad, they successfully identified and landed the UAV on the landing pad on their ASV on the water.

In conclusion, both methods applied all tracking features, which had been discussed above. The difference is on the way they extract the features and the tracking mechanism. However, both methods have some weaknesses in certain scenarios.

2.3.2. Visual tracking using IR beacon

In 2011, Wenzel et al. demonstrated autonomous landing of a UAV using an IR sensor from Nintendo WII remote control to detect the IR beacon in the landing pad. Their aim was to find the cheapest solution for autonomous landing. From the test, they were able to land a UAV onto the platform autonomously. However, their algorithm only works in indoor areas where everything was controlled (Wenzel et al., 2011). Figure 2.7 demonstrates the platform configuration used by Wenzel et al. (2011).

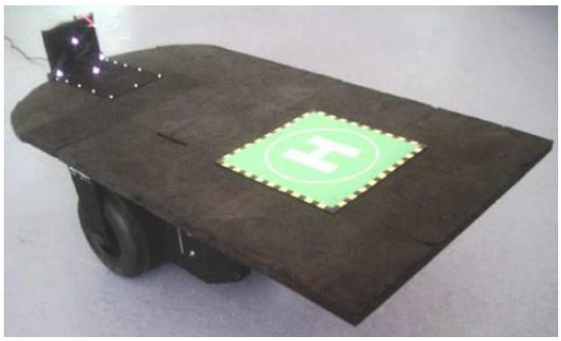


Figure 2.7: Landing system equipped with IR beacon (Wenzel et al., 2011).

Improvement showed by Gui et al. in 2013. They conducted similar research to land a fixed wing aircraft using infrared LED (IR LED) as beacons. Their vision system was located in the UAV for local processing and minimising delay. Figure 2.8.A shows the system diagram used by Gui et al. (2013). The camera was a visible light camera which was integrated with digital signal processing (DSP) with a fix 940nm optical filter located in front of the camera's lens to filter light interferences. After adding the optical filter, the camera is only capable of sensing 940nm wavelength region of colour spectrum. Figure 2.8.B and C displays the camera and the beacon size used for this experiment. With this configuration, the beacon was guaranteed to be visible even against the complicated background because most of the objects in the frame will appear to be grey to black except the beacons itself that appeared to be grey to white. To separate the target candidates from the background and noises, they applied thresholding using modified Otsu thresholding and filtered out all candidates that did not meet the required size.

After filtering the beacons, the algorithm continued with calculating the IR centre using the Negative Laplacian of Gaussian (NLOG) operator. After being able to find the IR blobs, the distance between the blobs were calculated for further verification. Next, the position and orientation (POSE) of the blobs were extracted to find the POSE of the UAV in the real world (Gui et al., 2013).

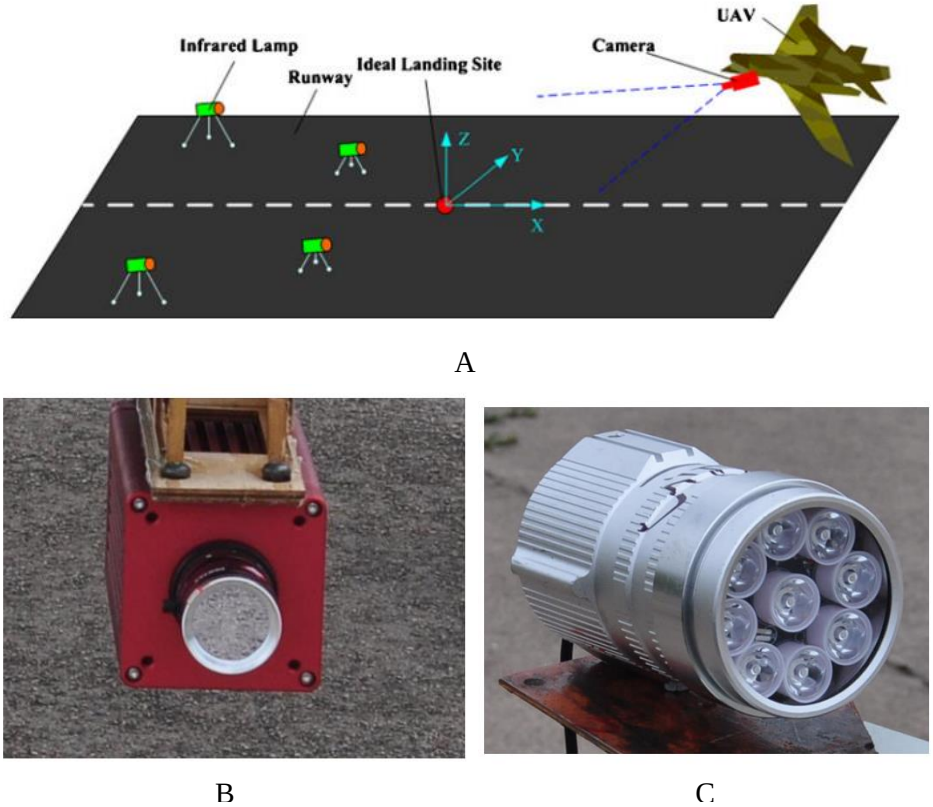


Figure 2.8: (A) illustration of the system configuration, (B) the DSP camera with 940nm filter, (C) the Beacon setup (Gui et al., 2013).

Since visible light is everywhere in the outdoor environment, detecting a more isolated light spectrum, which was higher or far from visible light spectrum, can decrease the chance of false positive detections without extra manipulation. In 2017, Khithov et al. applied the same method with higher IR wavelength. Their method was trying to detect Mid-Infrared Region (MIR) range wavelength (3-8um). With this configuration, a dedicated IR camera was required since light spectrum of interest was above the normal CMOS and CCD ability, which only around 1100nm (Pap et al., 2010). The testing showed that with the higher wavelength spectrum, the detection distance was improved and the clarity of the blobs was better since there were not many objects that emit the same wavelength. However, they never actually tested this method in the real world (Khithov et al., 2017).

A more compact IR based detection system was performed by Ding et al. in 2015. Their system uses a combination of Forward-Looking Infrared (FLIR), Inertial Navigation System, and Radio Altimeter (RA) to land aircrafts. However, this method is not applicable for our purposes since the equipment that is used on this method is beyond the budget for this project.

Using Infrared (IR) light as beacons (blobs) to give uniqueness to the platform has been proven as the most robust method against any problems that occurs when using colour detection. However, this is an expensive solution. The lack of the availability of the dedicated IR camera is the main reason (New Imaging Technologies and SENSOR UNLIMITED, 2017). In addition, modifying a normal camera to see IR wavelength is not an easy task, and the price of the IR bandpass lenses is expensive (THORLABS, 2017 and Precision Micro-Optics, 2016).

2.3.3. Pattern extraction

Placing a fiducial marker such as 2D pattern can also be used to identify the object. Saripalli et al. (2002) conducted a method that works in outdoor area. The research was about landing a UAV on a landing pad by extracting the "H" sign for visual positioning.

Using camera looking downward and a companion computer that attaches on to the UAV, The algorithm starts identifying by using thresholding and 7x7 median filtering on each input frame. From these stages, the landing pad was be able to be filtered out from the background. The resultant image was then tested by summing all connected pixels in the frame. All groups of pixels that the areas are lower than or bigger than the specified are removed. The remaining objects are the Region of Interest (ROI). To decide the actual pattern among the obtained ROI from the previous process, the moment of inertia of each obtained ROI are calculated and compared with a calibration value. The calibration value is a collection of a moment of inertia value of landing pad created prior the flight (Saripalli et al., 2002).

According to Saripalli et al. (2002) testing, the method can fully land the aircraft to the landing pad. Their method relies on two assumptions. Firstly, the pattern of the landing pad has to have a well-defined geometric shape. Secondly, the shape has to be in two dimensions. One weakness of this type of method is if parts of the pattern suddenly change such as a shadow covering the object, the algorithm will not be able to detect the shape. Thus, in the unknown environment like on the boat where shadows can come from any angles due to boat orientation and sun's position, this type of algorithm is not applicable. Unless, the algorithm is calibrated with a lot of images which covers all those possible scenarios.

One approach from Lange et al. (2009) can be used to solve problems above. Lange et al. (2009) use camera looking downward and companion computer on the quad to process the acquired images like the other. However, they proposed a method by using several centric white rings on the black background. Each rings is made with the same ratio between the inner and the outer sides; thus, the ring can be detected independently. As a result, if some of the rings were not visible, the others which visible will allow the algorithm to work. Moreover, since the gap between the rings is fixed, calculating the gap size between rings from different sides gives extra information of estimated position of the UAV to the pattern. Equal gap on every sides means the UAV and the pattern are perpendicular while unbalance means they are not. Besides, the altitude estimation can be acquired by simply calculating the radius of the rings, and increasing the number of the rings allows the detection process to be done at higher altitude. The detection process is illustrated in Figure 2.9. Because of the simplicity of the algorithm to recognise the shape of landing pad, a special tracking algorithm is not required (Lange et al., 2009).

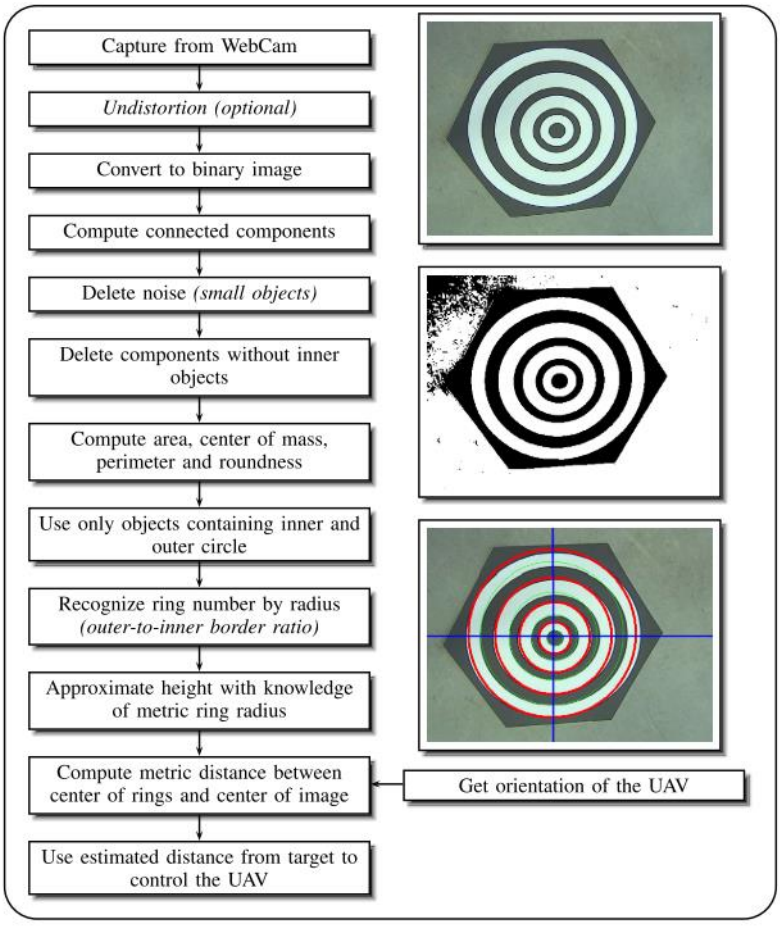


Figure 2.9: *Lange pattern and visual processing process (Lange et al., 2009).*

Recently, Lee et al. (2012) recommended a similar method that is using downward looking camera with fiducial marker such as AR tag located on the landing pad. The AR tag is better than the "H" sign and the collection of concentric circles since it can give UAV orientation directly from the extraction process. Using a pinhole model camera as shown Figure 2.10, the pose of the tag in the image plan can be transformed into global frame. Moreover, using Unscented Kalman filter they use this position information together with the reading from the sensor to estimate the motion of the quadrotor. To control the precise position of the Quadrotor they developed an algorithm called Image-based Visual Servoing (IBVS). For the control of the actual quadrotor movement, they use an Adaptive Sliding Mode Controller. In addition, they also equipped the surrounding testing room with VICON cameras for position estimation if the visual tracking should loses the target (Lee et al., 2012).

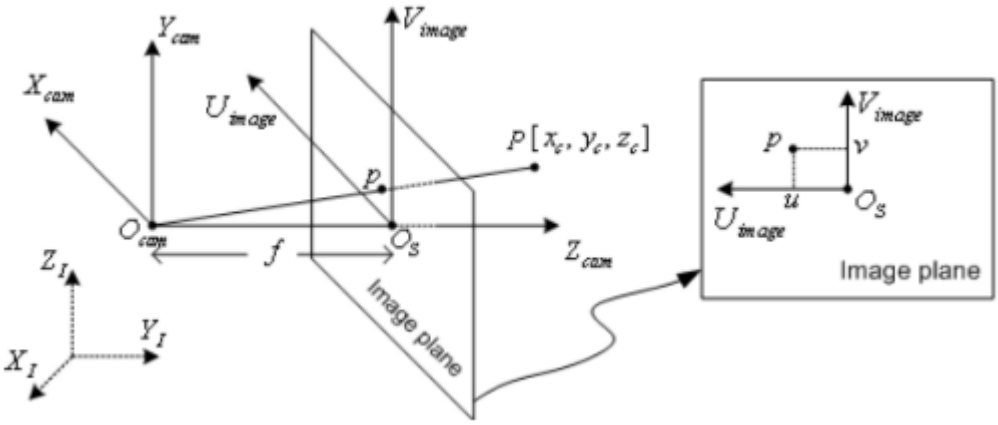


Figure 2.10: Pinhole model camera (Lee et al., 2012).

One drawback from using a single AR tag as the unique shape is, as the camera gets closer to the landing pad, the AR tag is getting bigger on the camera's FOV. As a result, the camera will not be able to detect the shape anymore unlike the colour detection and concentric circles, which have the feature inbuilt with the landing pad even the camera is very close. To solve this problem, Chandra et al. (2016) use multiple size of AR tags that are located inside the other. The AR tag was detected using Robot Operating System (ROS) build in library for AR tag detection. The library directly calculate 3D POSE of the AR tag and find the relative distance between the Quad on the camera and the AR tag location. Although the simulation showed the proof of the concept, the actual real world test has never been performed (Chandra et al., 2016).

Regarding the pattern detection method, the decision on the type of the pattern is critical. Complicated shapes may be robust and unique from the environment; however, to recognise

the shape many steps are required; hence, using more computational resources. Comparing the concentric circle from Lange et al. (2009) and AR tag from Chandra et al. (2016), detecting several circles is easier than multiple AR tag although the AR tags give can give extra information about the heading.

2.3.4. Background subtraction

As described on the tracking methodology above, there are two ways to provide information for object detection, fixed and adaptive information. Many object detection approaches above are working by directly extracting the features of objects and categorising them according to the region. The closest region (candidates) to the desired target then will be taken as the target. The disadvantages of fix-information is the algorithm is exposed to the same information during the process. Therefore, it is difficult for the algorithm to do its job when the environment is changing rapidly. To have a robust algorithm, object detection with adaptive type information is needed. However, compared with the object detection with fixed-information, object detection with adaptive information suffer from using more computational resources.

Pinto et al. (2014) uses background subtraction method to land a UAV in outdoor areas. The system that they used was a camera looking upward which was attached to the landing platform mounted to the back of their ASV. With this configuration, latency on the visual system can be minimised. Since the algorithm is running on the slightly powerful computer and the camera is closed to the computer. Besides having access to the entire computer resources, having camera looking upward also reduces the chance of having false positive detection since there are less objects in the sky except for birds and random noise (Pinto et al., 2014).

The image processing in their application was handled by OpenCV library. The background subtraction method was background subtraction with adaptive Gaussian mixture model that was developed by Zivkovic (2004 & 2006). There are several type of background subtraction implemented on OpenCV, but background subtraction with adaptive Gaussian mixture model has the better adaptability to the environment over the other method since the Gaussian distributor is automatically calculated and updated by the software according to the change of environment (OpenCV: Background Subtraction. 2017).

After applying the background subtraction, a bounding box was used to fit the entire quad to extract the centre point of the quad for Klaman filtered estimation. The Kalman filtered estimation then fused the point location with the sensors estimation to estimate the quad in the real world frame. After that, the heading of the quad is sent to the UAV and let another system

on the UAV worked out the best correction to centre itself to the camera. The controlled position is handle by PID controller running at 100Hz (Pinto et al., 2014).

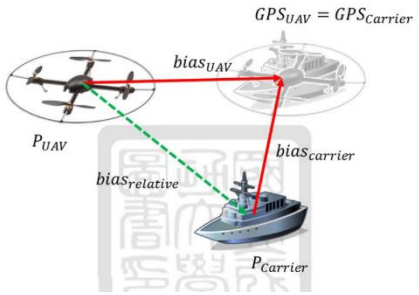
Yi-ChengLu (2017) designed a similar approach using fix thresholding with background subtraction to overcome the detection problem on critical environment such as overcast and clear sky. The background subtraction was applied to separate low dynamic objects and high dynamic objects. Background subtraction that they use was a frame differencing. Frame differencing method works by subtracting each input frame form a fix frame. After applying the background subtraction, the appropriate threshold value then calculated. By comparing the intensity pixels of the background against object itself. Difference of intensity can be found. This difference then applied to a fix thresholding for further process (Yi-ChengLu, 2017).

Their test showed that frequently, the method did not only pick up the target but also noises. To overcome this problem, image moment comparison was added to narrow down the possibility of false positive. After being able to detect the target through visual positioning, GPS correction is performed to allow the system to have dimension position prediction.

The correction is done using the following equation. It was derived from the shown diagram.

$$GPS_{UAV} = P_{UAV} + bias_{UAV}$$

$$GPS_{carrier} = P_{carrier} + bias_{carrier}$$



Where:

$$P_{visual} = P_{UAV} - P_{carrier}$$

Hence:

$$bias_{realtive} = GPS_{UAV} - GPS_{carrier} - P_{visual}$$

Figure 2.11: Equation for GPS correction using visual positioning (Yi-ChengLu, 2017).

In conclusion there are two possible method which can be used for this project. These are IR detection and background subtraction.

3. METHODOLOGY AND IMPLEMENTATION

This section outlines the specifications required by the visual detection method; as well as, deciding on the suitable approach for this project to be implemented based on quantitative assessment of two possible visual positioning methods found in the literatures, Background Subtraction and Infrared Detection. The qualitative assessment will reflect on how well both methods worked with required specifications. Moreover, it is also explained the implementation of the method to fulfil the task. The finally yet importantly, it is also describe the methodology of the visual positioning system.

3.1. Methodology

There were several challenges that had to be addressed when deciding a suitable visual based system for this project. Some of which were the ability of the method used to function under different weather conditions. Based on the speculation on the working environment, there were several key features that had to be considered when deciding the chosen method. Figure 3.1 lists all the required features.

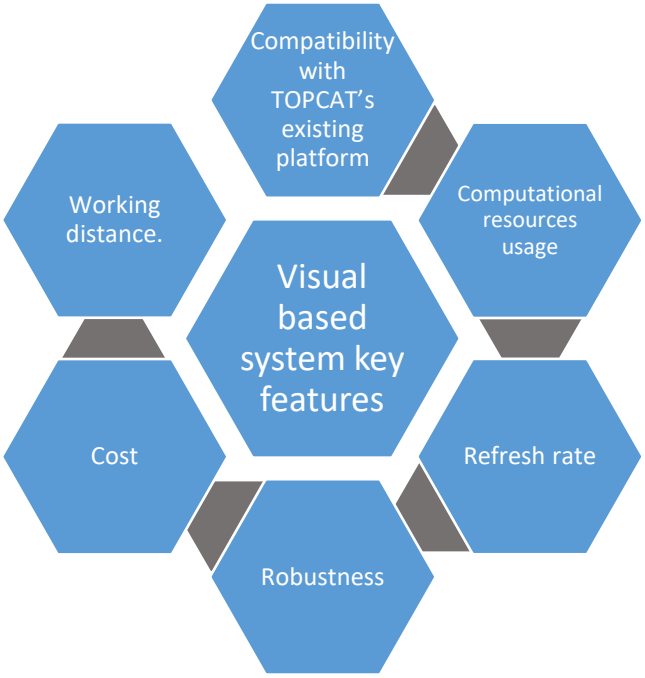


Figure 3.1: Required features of the project

As described in the introduction, TopCat's system is built on the ROS (Robot Operating System) environment (Sammut et al., 2016). All of the methods that had been researched and discussed in the literature review are compatible with the first required features above since most of the methods are already implemented in the OpenCV library, which can be used on the

ROS environment. Besides the OpenCV library, the ROS also has built in libraries that capable to do some image processing such as AR tag detection and Image transform geometry, which can be modified to suit the need.

In terms of the computational resources, there is no solid evidence as to which method was using the least computational resources; however, some of the IR detection methods above are tested and able to run on companion computer such as ODROID board and Raspberry PI. This indicated that those methods must have low usage of the computer resources since those companion computers generally have lower computational power compare to general computers with latest Central Processing Unit (CPU) and Graphical Processing Unit (GPU) products. For background subtraction, since most authors run it from either the ASV or the carrier vehicle, there was no reason to add separate companion computer since those vehicles was equipped with decent computer. Therefore, there is no prove whether the background subtraction detection can be run on companion computer or not.

Next, to get a smooth and precise tracking, a system with high refresh rate was essential. Number of processing required doing the tracking and the detection, optimisation of the algorithm, and latency determined the refresh rate of each methods. The less processing used by the algorithm do detection and tracking, the higher the frame rate was obtained. However, this point obviously had to be supported by compatible hardware such the camera used since camera usually come with fix number of frames it can captured per second (FPS). Furthermore, one of the reason most method above were using companion computer attached on the UAV was to avoid latency. Locating the camera unit and the processing unit on different location required wireless image streaming to complete the system. Streaming images through Wi-Fi and Telemetry Radio required high bandwidth. The better the quality of the image, the bigger the bandwidth was required. Wireless streaming usually causes bottleneck on the system even though the algorithm was optimised and supported by a decent computer system, especially for long-range transmission. Therefore, having a system that could integrate both camera and the computer in the same position was necessary to increase the refresh rate. Both the Background subtraction detections and the IR beacon detections method above used this configuration where the camera and the processing computer in the same vehicle.

Moreover, regarding the robustness of the algorithm, both methods were not affected by the light environment especially the IR detection. With the background subtraction, as long as the UAV stands out from the background and not stay too steady at one place, the method will

work. The down side of background subtraction is that high sensitive to vibration. However, this problem can be solve by adding a gimbal.

Nevertheless, with the right equipment, the IR detection was found to be better and faster compared with background subtraction since it could work without the complicated adaptive information to do the filtering. The key point of IR detection is to use an IR beacon that emits frequency is far from the sunlight IR and hance isolated. The infrared (IR) spectrum has a wide range.

According to ISO 20473, Infrared (IR) is divided into three regions, Near-Infrared (NIR) range from 0.78 to 3 μm, Mid-Infrared (MIR) range from 3 to 50 μm, and Far-Infrared (FIR) range from 50 – 1000 μm. The infrared from the sunlight lies in the NIR region. With this grouping, it can be concluded that having IR beacon which the emission is on MIR and FIR should be a good start.

However, the most common light sensors on consumer cameras, (CCD and CMOS), only have ability to detect electromagnetic spectrum from 0.380μm to 1.1μm which is in NIR region (Rau et al., 2017). This means to be able to detect IR beacon with emission higher than sunlight, a special camera sensor is required. As mention in the literature review, there are dedicated IR cameras on the market that can be used for detecting higher IR, but most of them are expensive ((New Imaging Technologies and SENSOR UNLIMITED, 2017).). In addition, the IR band pass filter, mentioned by Gui et al. (2013), which is used to isolate the beacon's wavelength from the other wavelengths in order to reduce the noise on the detection, is another expensive piece of equipment (THORLABS, 2017 and Precision Micro-Optics, 2016). The IR lock sensor is an optional inexpensive IR detection sensor that it is claimed can be used for autonomous landing of UAVs. This sensor is fully supported by PIXHAWK flight controller. See section 3.2 for more detail.

The last but not least was the working distance. Similar to the shape and the colour extraction, the detection range of using an IR beacon is very long because the distance is directly controlled by the size of the beacon or more precisely the amount of the power that the IR LEDs emit.

Background subtraction is affected by the size of the UAV; however, which make it limited. Furthermore, with the camera looking upward, background subtraction also suffers with on close range detection since the size of the object detected would be significantly enlarge, as the object got closer.

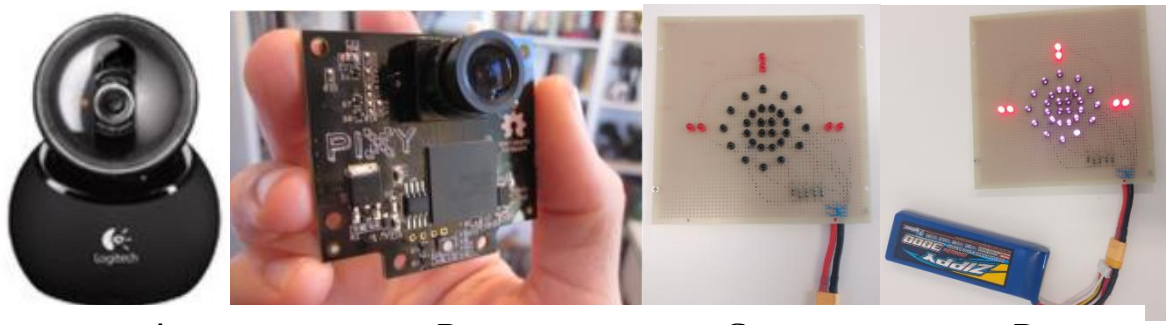
In conclusion, there is no such thing as perfect technique to over the whole requirements to create robust system. Therefore, combining both background subtraction and IR beacon detection was our approach. The background subtraction was intended on the long-range guidance using normal webcam while the IR beacon was planned for the short-range guidance using the IR lock sensor.

3.2. Hardware decision

The camera that was used in this project is the USB webcam Logitech QuickCam Orbit AF with a maximum resolution of 2MP (megapixel). The camera is a rolling shutter type with maximum video capture of 30FPS (Frame per Second). The camera has a diagonal field of view 75 degree (CNET. 2017).

The IR lock sensor is the CMUcam5 Pixy that has been modified to detect 940nm infrared. The CMUcam5 is equipped with a dual core processor NXP LPC4330 with 204MHz clock frequency. The image sensor is the OV9715 with ¼" sensor size. The maximum resolution is 1280x800 pixel. The typical power consumption is 140mA and will run with regulated 5V DC or 6V to 10V DC unregulated. It weights around 27grams. The IR lock sensor is connected to PIXHAWK via I^2C (CMUcam5 Pixy. 2017). The detail on configuration and the connection is described in navigation control (Keen, 2017). To be able to use the IR LOCK a rangefinder is required. The chosen rangefinder is LeddarOne rangefinder from LedarTech.

The beacon is constructed by 28 IR LEDs with the wavelength of each LED is 940nm. Each LED has forward voltage 1.6V and current consumption around 100mA. The LEDs are configured into three parallel connections where each branch contains eight LEDs in series with a 3.3Ω resistor. The total power consumption of the beacon is 4621mW.



A B C D

Figure 3.2: (A) QuickCam Orbit AF Logitech Webcam (CNET. 2017), (B) IR LOCK sensor (CMUcam5 Pixy. 2017), (C) IR beacon configuration, (D) IR beacon on

3.3. Preliminary Test of background subtraction

There are several preliminary test done to determine the best approach to use the background subtraction. The first test was finding the suitable parameter for the background subtraction. The background subtraction implemented by OpenCV is Background subtraction using Gaussian mixture model which was developed by Zivkovic, 2004 and Zhang et al., 2012. It has several parameters that can be changed to suit our need. Those are length of the history, threshold, and shadow detection. The length of history controls the number of history to compare. The threshold parameter is the Mahalanobis distance between a pixel and the background model. Shadow detection is a Boolean to activate the shadow detection. Besides there is also learning rate parameter. This parameter control how fast the input images is modelled (Zivkovic, 2004, Zhang et al., 2012)

The next stage is to use the parameter found in the previous test to determine the performance of the background subtraction directly to detect the quadrotor under different environmental conditions. However, in order to improve the accuracy of the background subtraction, several process were added on the top of the background subtraction. The algorithm process is shown in the Figure 3.3.

This algorithm was developed in the ROS environment. The frame were captured by "usb_cam" node from ROS library (usb_cam-ROS Wiki, 2017). After capturing, the frames are then processed in using background subtraction. The contour detection is used to label the detection result. The detection result produced by background subtraction is represented in one dimension matrix with intensity range from 0 to 255, where 255 shows the detection and zero is the background and somewhere between is the shadow detection if enabled. Due to the behaviour of the background subtraction, filter is added on the top of the morphological operation to filter out most of false positive detection and left true positive detection. The filter removes positive detection that contain pixel less than 100. In addition, the filter also removes all positive detection that the ratio of width and height is not in the range of 0.9 to 1.1. The actual ratio of the quad is one; this boundary is applied to compensate the yaw, pitch, and roll of the quad. Finally, drawing the result of filter to indicate the detection before outputting.

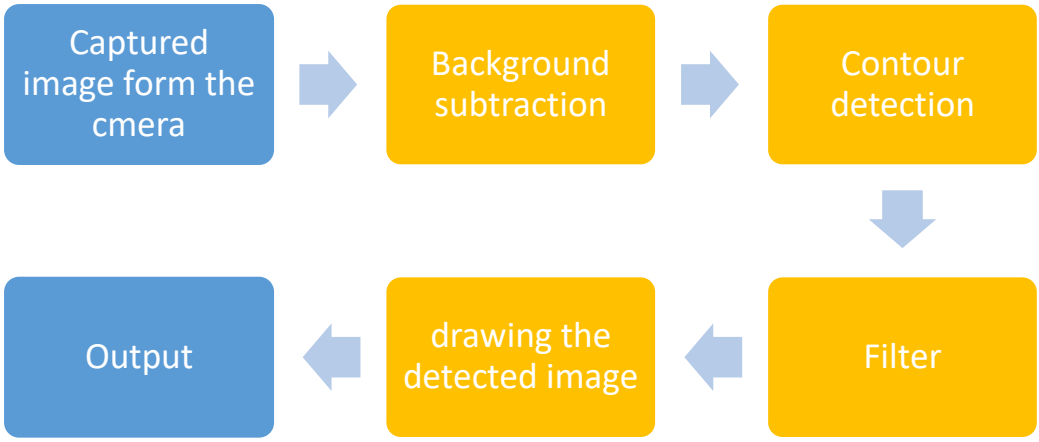


Figure 3.3*: Background subtraction algorithm develop during preliminary test*

3.4. Implementation

As discussed in the methodology section, background subtraction and IR detection are implemented for long guidance and short guidance. The previous background subtraction test results, listed on the result section, leaded us to combine the background subtraction with adaptive thresholding. This is because the combination of the background subtraction and the filter could not detect the quad while it was on loiter mode without introducing noise to the result or making the system more sensitive to the movement, which would be a problem when mounting the system to the dynamic platform on the ASV. In addition, the system could not detect the object in the event of occlusion such as sun glare.

Another solution is applying tracking system by fusing other sensor reading. Due to time constrain, implementing a tracking system is reserved for future development. Detection using shape extraction could be applied for this kind of task, but the working distance was limited since the further the drone, the lesser the drone shape detail shown in the camera. Detection using colour extraction was not being considered because any unique colours would be backlit by the sky, making them indistinguishable.

3.4.1. Visual detection algorithm overview

The algorithm is developed in the ROS environment. Developing the system in the ROS environment allows the single threaded C++, to be treated as multi-threaded. The following Figure 3.4 shows the nodes diagram of the visual guidance. The detection algorithm consists of three main nodes. The first node is backSub_filter. It process captured image using

background subtraction. The second node is adptTresh_filter. In parallel with the first node, it process the captured image using adaptive thresholding. The third node combines the result from the first and the second nodes to provide visual detection. The source image is captured by “usb_cam” node from ROS library and it is corrected by “image_proc” before visual processing (image_proc - ROS Wiki, 2017).

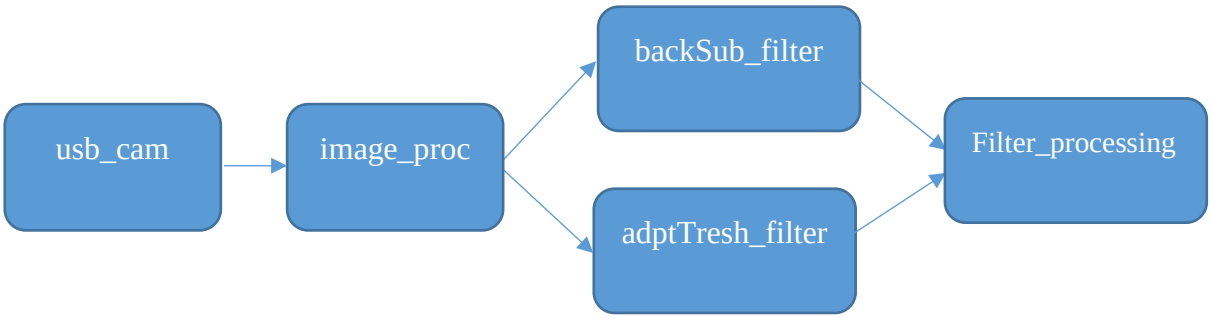


Figure 3.4*: Visual Guidance nodes arrangement*

Inside the backSub_filter node, the processing shows in Figure 3.5 is applied. This is almost the same process used on the preliminary test. The same process also happens inside the adapTresh_filter. The output of these two nodes are binary images with the intensity five for true and zero for false. Each detected object are represented as a circle where the radius of the circle is the width of the object.

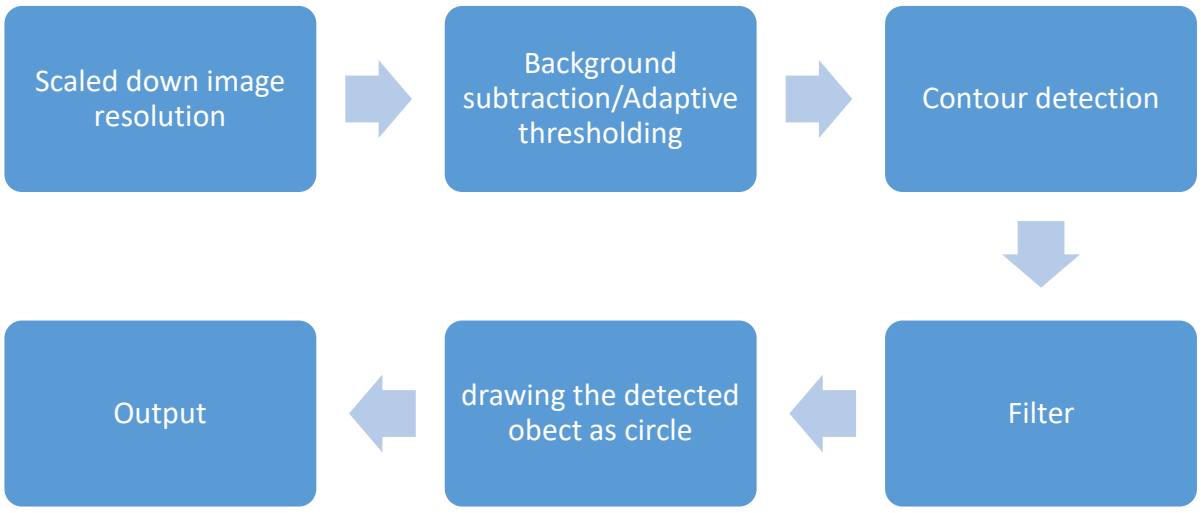


Figure 3.5: Processing inside backSub_filter node and adapTresh_filter node.

The Figure 3.6 illustrated the process inside the filter_processing node. The process starts from adding the frames produced by the background subtraction and the adaptive thresholding. The addition is used to reinforce the detection result from both nodes if both nodes detect objects at the same location while retaining all the detection results from both nodes. Keeping all the detection results from the inputs are critical because under certain circumstance one of the nodes may lose the detection. This is why addition is used rather than a Boolean operation such as “AND” since both Boolean operations remove all the true positive detections if it is only exist in one node. Next, the result is multiplied with the previous update frame to further reinforce the detection. The “add one” block on the addition result is to make sure no information is discarded when multiplying. Since this process is history based, the true positive pixel intensity will increase faster than the false positive pixel intensity after a few iteration. To further supress the false positive pixel, a fixed threshold is applied. This is to ensure only the highest intensity pixels taken for the prediction and visual positioning.

The feedback loop is used to reuse the history frame when determining the next coming frame. The scaling down and the adding one are used to make sure the history will never reach the maximum intensity value of 255 and the lowest intensity value of 0 when get multiplied with the incoming frames.

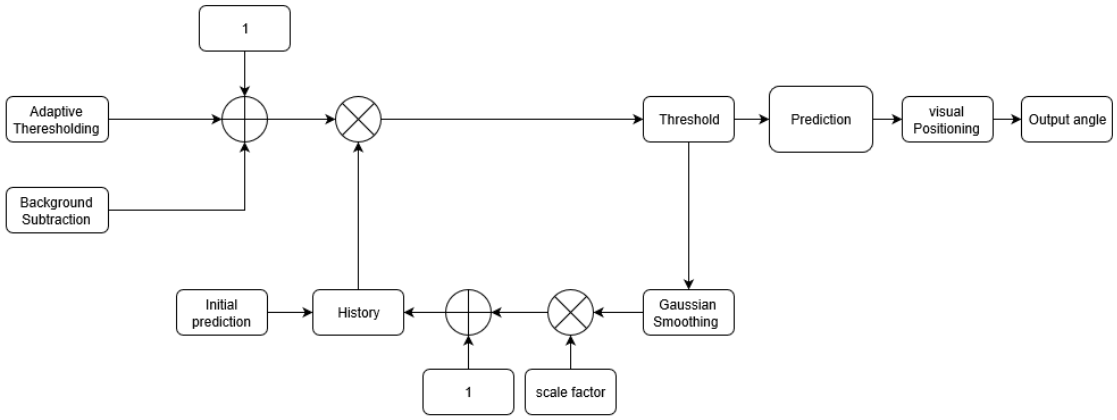


Figure 3.6: Flow chart diagram of filter processing node

Figure 3.7 illustrates the process using example figures. From the figure, it can be seen that timing is critical in this process. Processing frames form two nodes that is not coming from the same image source will never give us a detection because the drone is moving. Moreover, due to multi-threaded capability this condition is possible to happen since each node does not bond to the other. The ROS package called message_filter (message_filters - ROS Wiki. 2017) is used to make sure the filter processing always process the images that is coming from the same

source. In this project, approximately time synchroniser is intentionally used to give some uncertainty around the merging frame to make sure the history is counted. The effect of slightly different timing showed in the Figure 3.7 on the merge result. Every time the riming is off, the probability of getting the detection reduces and the uncertainty increases. Therefore, it opens the possibility for the history to merge with the uncertainty more which is not showed in this illustration.

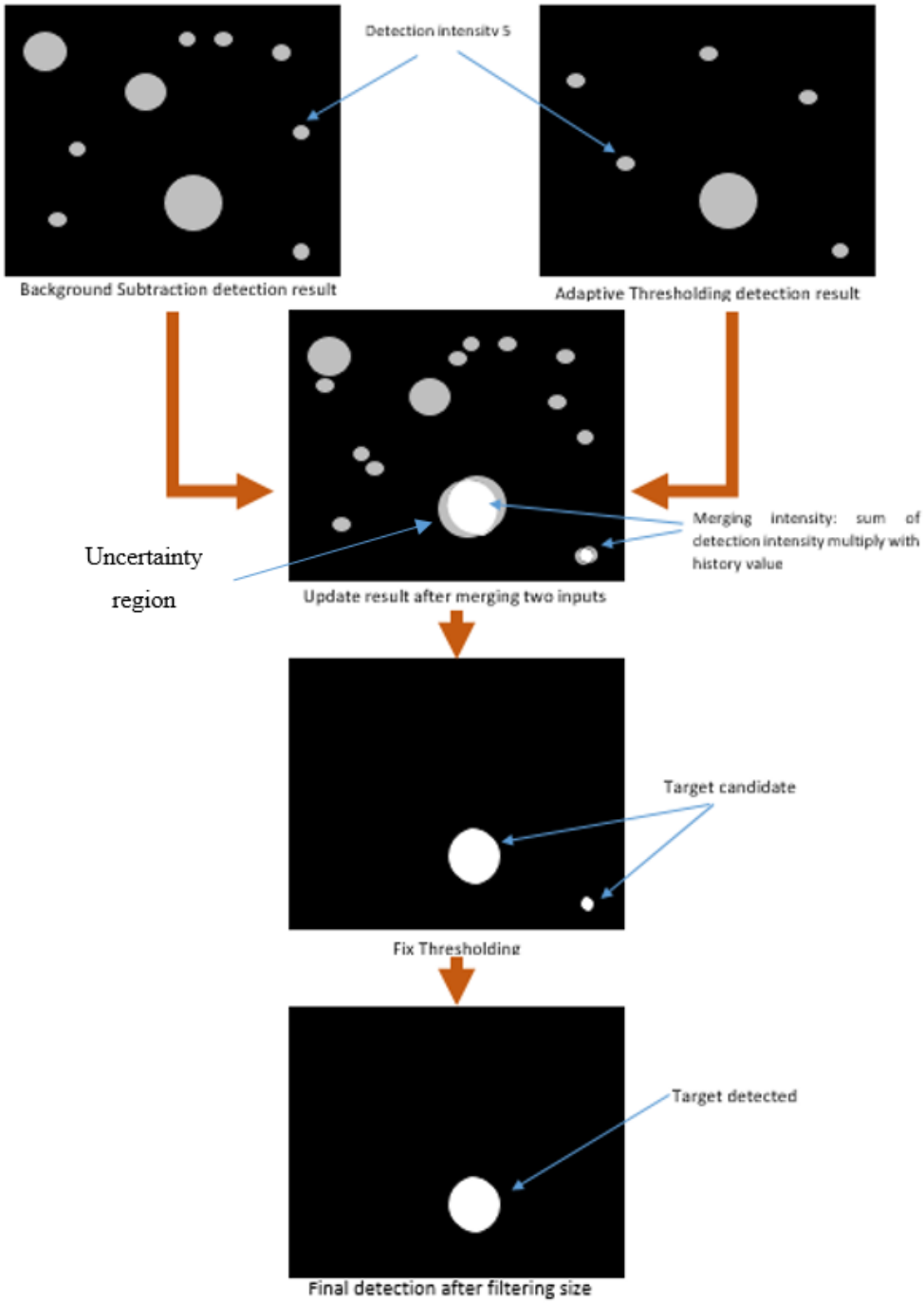


Figure 3.7: Illustration of the processing inside filter_processing node

3.4.2. Visual positioning system

To interact with the PIXHAWK flight controller for flight navigation, the flight navigation uses the ROS package called MAVROS. The MAVROS has implemented LANDING_TARGET functions form one MAVlink command to handle autonomous landing using visual guidance. This command takes the target angle offset position and the altitude of the quadrotor to guide the quadrotor to the landing pad autonomously. This function is aimed to be used in pairs with a camera looking downward from the quadrotor. However, since our approach is to use a camera looking upward from the landing pad, some transformation is required to transform the reading from the upward looking camera to downward looking camera. Figure 3.8 shows the difference between the system and the frame generated. For more detail of the function and the transformation, see the navigation controlled section of this project (Keen, 2017).

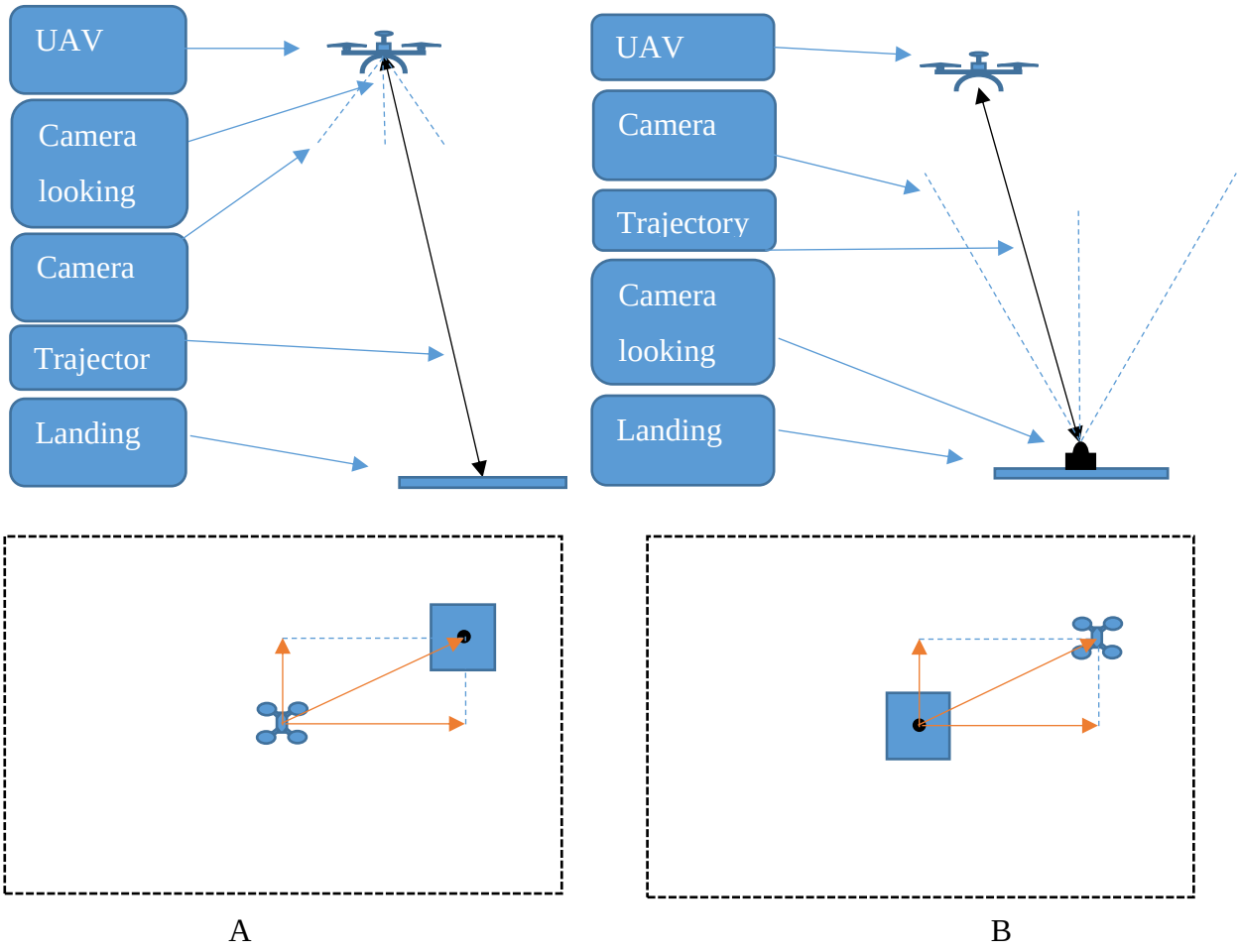


Figure 3.8: Difference in system overview and frame generated of using camera looking downward and upward. (A) Downward Looking Camera, (B) Upward Looking camera.

The angle-offset calculation is performed in the pixel coordinate frame and result in image coordinates. Figure 3.9 illustrates the relation between the two coordinate. The following

Figure 3.10 illustrates the modified calculation of the angle offset in the image coordinate of each axis using the similar triangle method that was developed previously by Squilter from Ardupilot (GitHub. 2017). “Target_V and Target_U” is the position of the UAV in the image frame in V and U axis while “VFOV and HFOV” is the image resolution. Moreover, α_y and α_x is the vertical and horizontal field of view (FOV) of camera and β_y and β_x is the offset angle in camera coordinates.

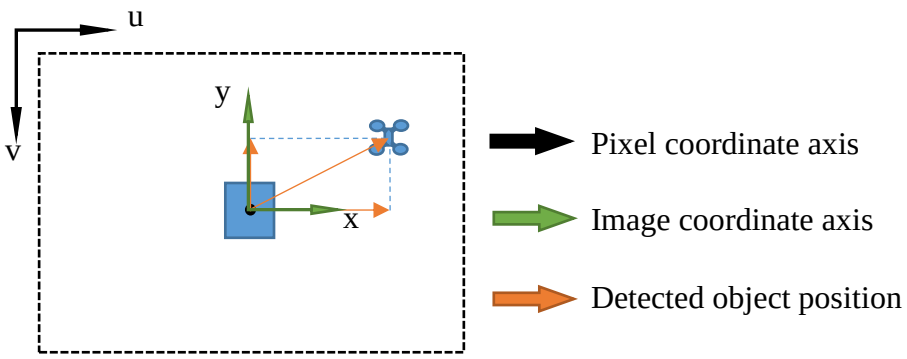
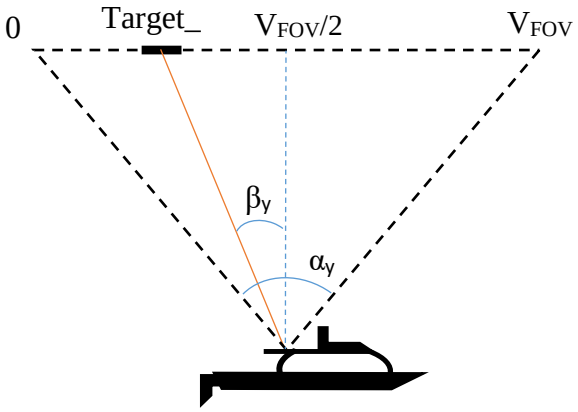


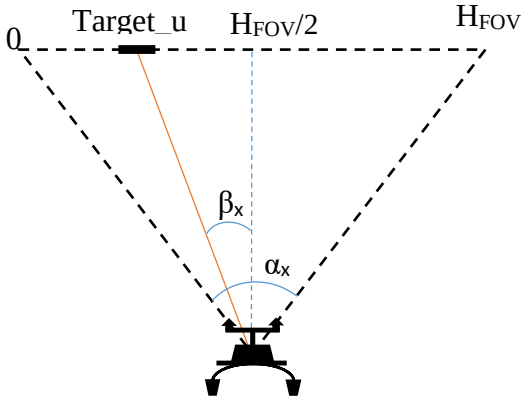
Figure 3.9: Coordinate *relation between image and pixel*



$$\frac{target\_v - {V_{FOV}}/{2}}{{V_{FOV}}/{2}} = \frac{\tan(\beta_y)}{\tan(\alpha_y/2)}$$

$$\tan(\beta_y) = \frac{\left(target\_v - {V_{FOV}}/{2}\right)\left(\tan\left(\frac{\alpha_y}{2}\right)\right)}{{V_{FOV}}/{2}}$$

$$\beta_y = \tan^{-1}\left(\frac{2\left(target\_v - {V_{FOV}}/{2}\right)\left(\tan\left(\frac{\alpha_y}{2}\right)\right)}{V_{FOV}}\right)$$



$$\frac{target\_u - {H_{FOV}}/{2}}{{H_{FOV}}/{2}} = \frac{\tan(\beta_x)}{\tan(\alpha_x/2)}$$

$$\tan(\beta_y) = \frac{\left(target\_u - {H_{FOV}}/{2}\right)\left(\tan\left(\frac{\alpha_x}{2}\right)\right)}{{H_{FOV}}/{2}}$$

$$\beta_y = \tan^{-1}\left(\frac{2\left(target\_u - {H_{FOV}}/{2}\right)\left(\tan\left(\frac{\alpha_x}{2}\right)\right)}{H_{FOV}}\right)$$

Figure 3.10: Angle offset calculation

4. RESULT AND DISSCUSSION

This section outlined the standalone tests for long-range and short-range detection in different environmental conditions as well as the integration of the detection algorithm with the overall software system. The tests begin with offline testing using ROSBAG files then, continue with real world tests. In addition, it also outlined any changing on the system that had been done to resolve some small error.

4.1. Preliminary test of Background subtraction

As shown in Figure 4.1, the tested quad could be detected against the complicated background using default parameters of the background subtraction from OpenCV. The resultant image was saved in a one dimensional matrix with an intensity range from 0 to 255, where 255 indicated the detection and zero represented the background.

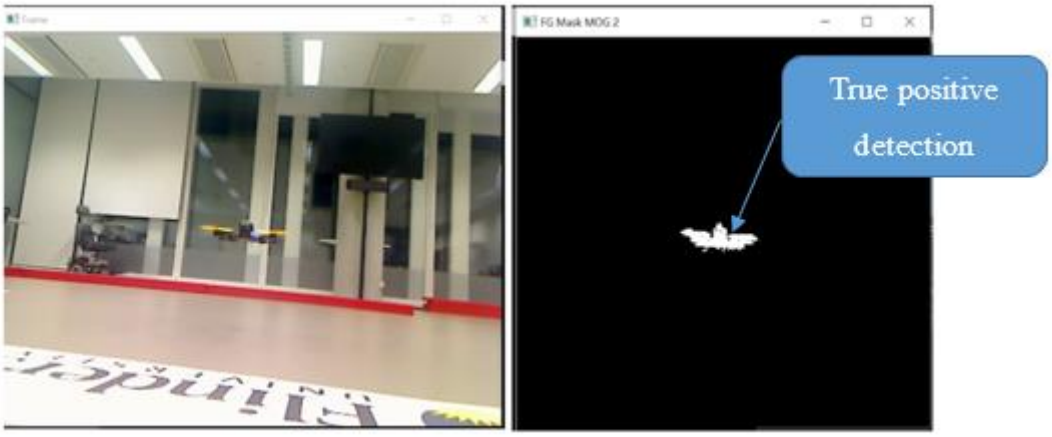


Figure 4.1: Background detection

Figure 4.2 shows the detection result when the threshold parameter was set to 0, 1, 5, 10, 16, 20, 50, and 1000. Excluding the case when the threshold equalled to zero, the detection result showed that when the threshold values were less than the default value of 16, the output showed more false positive detections compared with the detection result when the threshold values were larger than the default value. However, when the detection threshold much greater than the default value, more true negative detections were observed.

The zero case detection showed a different pattern due to the low boundary check from the implementation of the algorithm, which set the threshold to approximately the default value when it is set to zero or less. Therefore, the detection result was approximately identical to the detection result showed by the default threshold.

As explained in the OpenCV documentation (OpenCV: Image Thresholding. 2017), this was because the threshold value controlled how far the pixel in the model had to move before being classified as a foreground pixel. As a result, by adjusting the threshold value, the sensitivity of the background subtraction to movement can be adjusted.

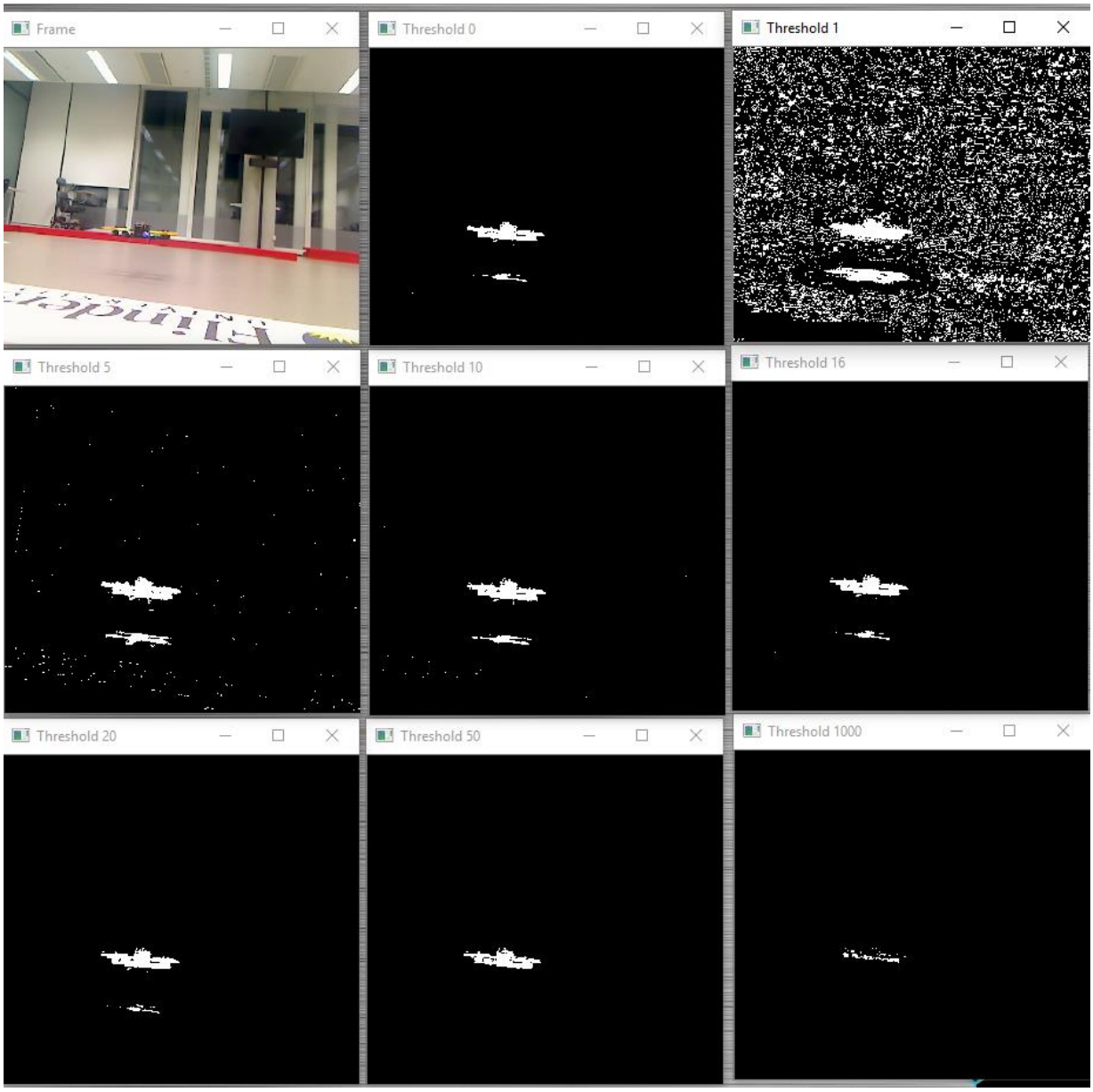


Figure 4.2: Background subtraction test with difference threshold values.

Next, the performance of the background subtraction was compared by changing the history and keeping the other parameter to default. The test values for the history were 0, 1, 50, 300, 500, 700, 1000, and1500. From the testing showed in Figure 4.3 below, there was no significant difference in the detection performance and computation usage when the history length was increased. Therefore, from the preliminary testing, the history parameter had negligible

difference, but would be re-evaluated in later tests. For time being, the parameter will be set to six.

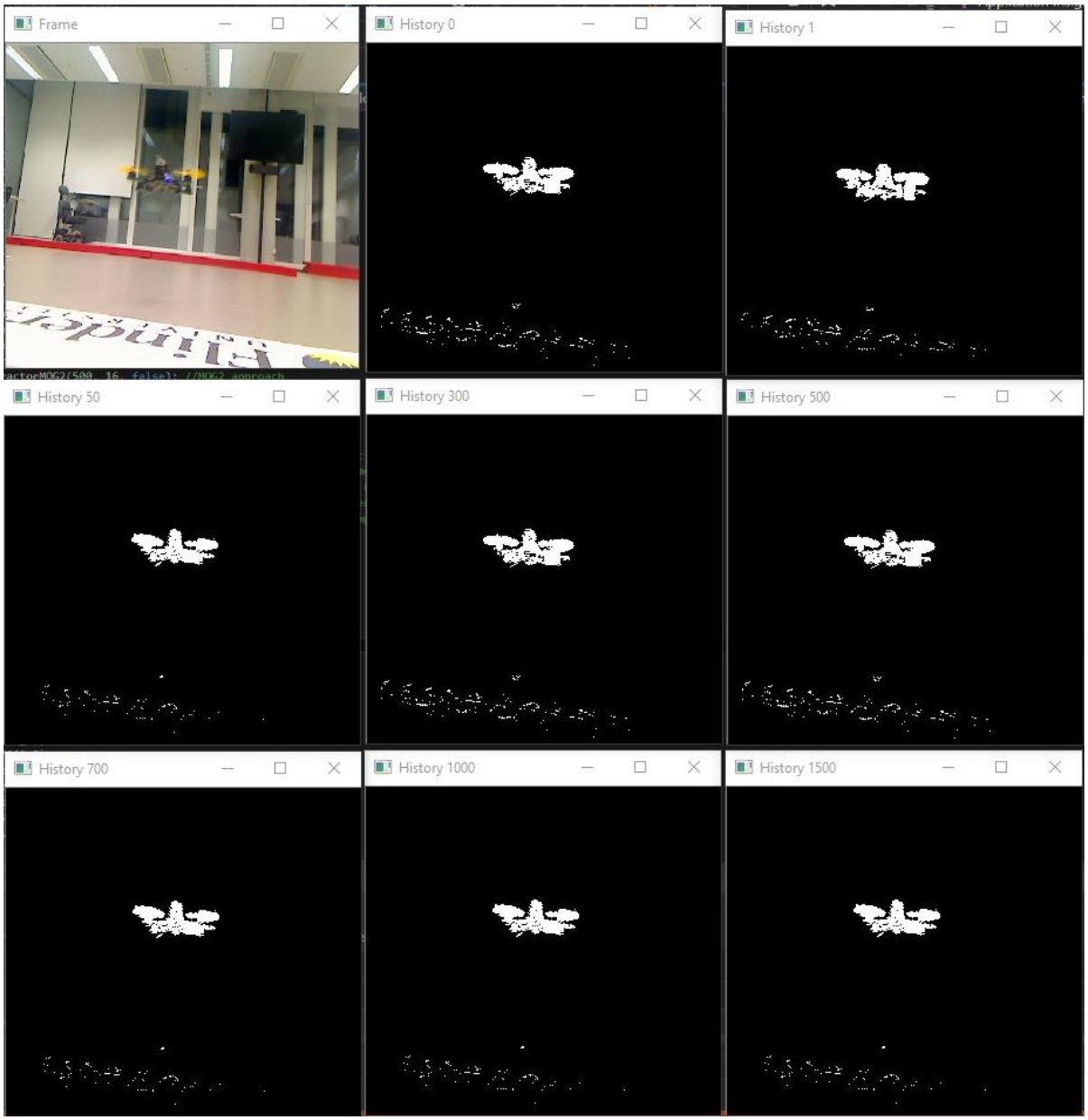


Figure 4.3: Background subtraction test result with different history values

On the other hand, significant difference was observed when the learning rate was increased. During testing, the learning rate was set to 0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, and 1.1. The results indicated the higher the learning rate, the faster the algorithm model the background. This was shown in Figure 4.4. These reading were taken by a webcam held by hand. It showed that the time taken by each learning rate value to settle the motion from hand was very different. A learning rate between 0.01 to 0.5 caused the small motion from the hand to settle faster than rates between 0.001 to 0.01. From this test, it can be concluded that the higher the learning rate,

the faster the motion required to be detected. In addition, a learning rate greater than or equal to one did not exist.

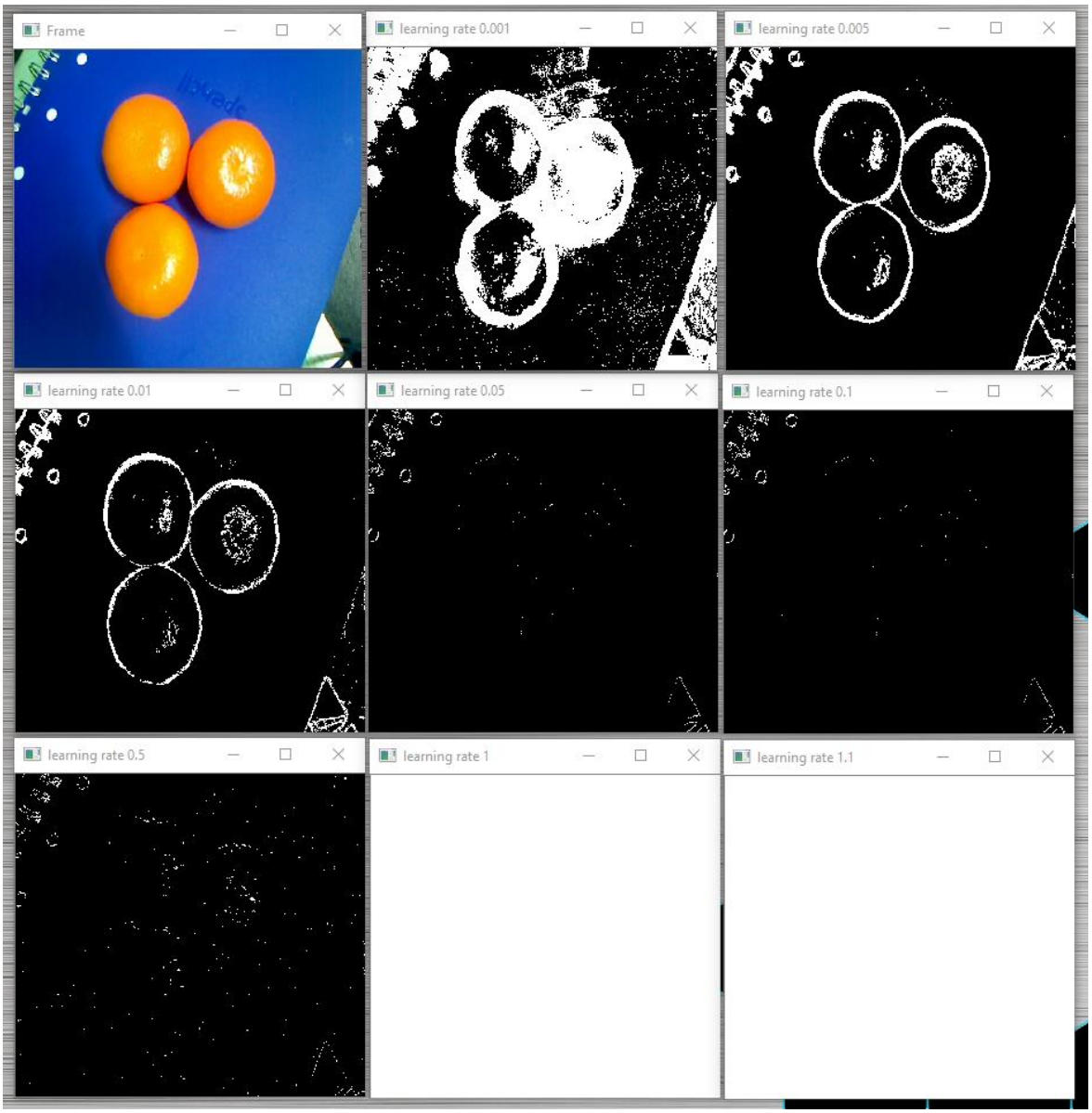


Figure 4.4: Background subtraction test result with different learning rate value

From the experiments before, the parameters that were critical for this project were the threshold and the learning rate. Due to wide range of operating condition, there was no exact values, which could be used for all conditions. The most suitable approach was adding an adaptive system, which would determine the most suitable values by assessing the environment condition. However, since changing the parameters primarily caused significant effects when the parameters were close to their extreme, set fix values which laid in the middle were deemed sufficient for our application. Therefore, the threshold value was set to 60 to ensure only the high dynamic object is detected. In addition, the learning rate was set to 0.05 to slow down the

learning process. In addition, the history was set to six to reduce the size of the history buffer, while the shadow was set to false to disable it.

4.2. Visual detection using background subtraction

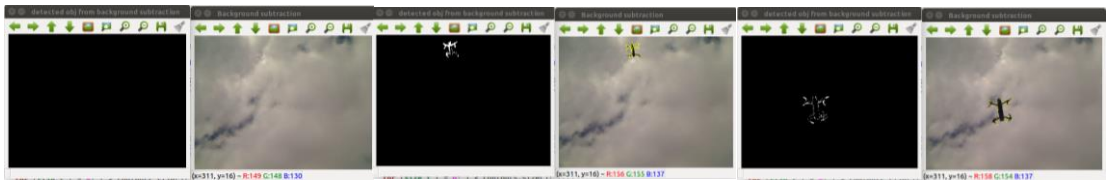
Figure 4.5 presents the offline test result using the parameters found in the previous test. For most of the condition, the algorithm perform very accurate. The detection was fast and clear even when the quad was moving faster and slowly alternately. The figures show three different condition: when there is not quad at all, when the quad was moving, and when the quad was loitering.

When the quad was on loiter mode after finishing the mission before deciding to land, the quad was surprisingly very steady when there was no wind. At this stage, the background subtraction frequently could not recognise the quad. To solve this issue, the learning rate was reduced to 0.01. As expected, lowering the learning rate introduced noises when groups of clouds with enough size passed through the FOV of camera. Adjusting the threshold value did not do any effect to the result. Adjusting the filter size to remove all object with size less than 500 seemed resolved the problem, but it decreased the working distance of the system since the quad needed to be close enough to fulfil the size requirement.

Besides unable to detect drone when it was on loiter mode, the background subtraction introduced noises when the camera did not stay still. Although this was not a problem right now due to the test platform was very steady, this would become a big challenge letter on since the ultimate goal of this project was to mount this setup onto Flinders's ASV, which would move and be unstable. The unstable platform could be compromised by mounting the camera on a gimbal letter on, but moving platform cannot be compromised with current setup unless the boat would to be forced to stop moving during the landing of the UAV.

Moreover, the detection also lack of ability to tell whether the quad was out of the field of view or it was occluded. At some condition, there were a case where the quad was in line with black area on the image such as dark sky. At this stage, the background subtraction was not be able to detect the quad even though visually human eyes could see the difference. This is because background subtraction uses some colour feature to model the background and if there is not enough difference between the background and the foreground, the background subtraction will see it as one object. Another form of occlusion was sun glares. Sun glare was hard to avoid since it was easily over contrast any objects in front of it especially if the object has smaller dimension.

The result from this experiment concluded that having background subtraction alone was not be enough to create a robust system. The problems that occurred from previous system were the system alone could not detect the quad while it was on loiter mode without introducing noise to the result and the system was too sensitive to movement, which would be a problem when mounting the system to the dynamic platform on the ASV. In addition, the system could not detect the object in event of occlusion such as sun glare.



A. Overcast condition

B. Fair day with cirrus clouds condition

C. Clear sky condition

D. Clear sky with little bit of sun glare condition

E. Direct sunlight condition

Figure 4.5: Background subtraction test using parameter found in preliminary test.

4.3. Visual detection using Background subtraction and adaptive thresholding

The test result of the combination of background subtraction and adaptive thresholding showed that by overlaying the detection result of both methods using addition and multiplication results in different intensity of detection. Representing the detected objects using circle is applied to increase the probability. Figure 4.6 illustrated the process on ideal environment. As can be seen, there were shadow effect on the merging result (Figure 4.6.C) after merging the result. This is happened due to a slight different on time stamp of the input frames. In addition, difference position of the quad on the history and the new input also increased the shadow region. However, the shadow effect then removed after applying fix thresholding. After fix thresholding, if there is only one candidate, the candidate will be set as the target; otherwise throw, the result and start new detection. The target was label with rectangular to help finding the centre point of the target for visual positioning. The other test result available on the Appendix B.

The following table 1 describes the accuracy of this approach against difference outdoor environment. Two things were be measured on each environment. The first was number of true positive detection against number of frame tested when the drone was in the frame, and the second was number of false positive against number of frame tested when the quad was not on the frame. The false positive detection when the drone was in the frame was not measured because the algorithm will always take the detection as true positive if there was only one detection left and ignore if there are more than one. Form the table, it can be concluded the method can give its best performance in cloudy days. The worst condition for this method to work is under direct sunlight where glares present.

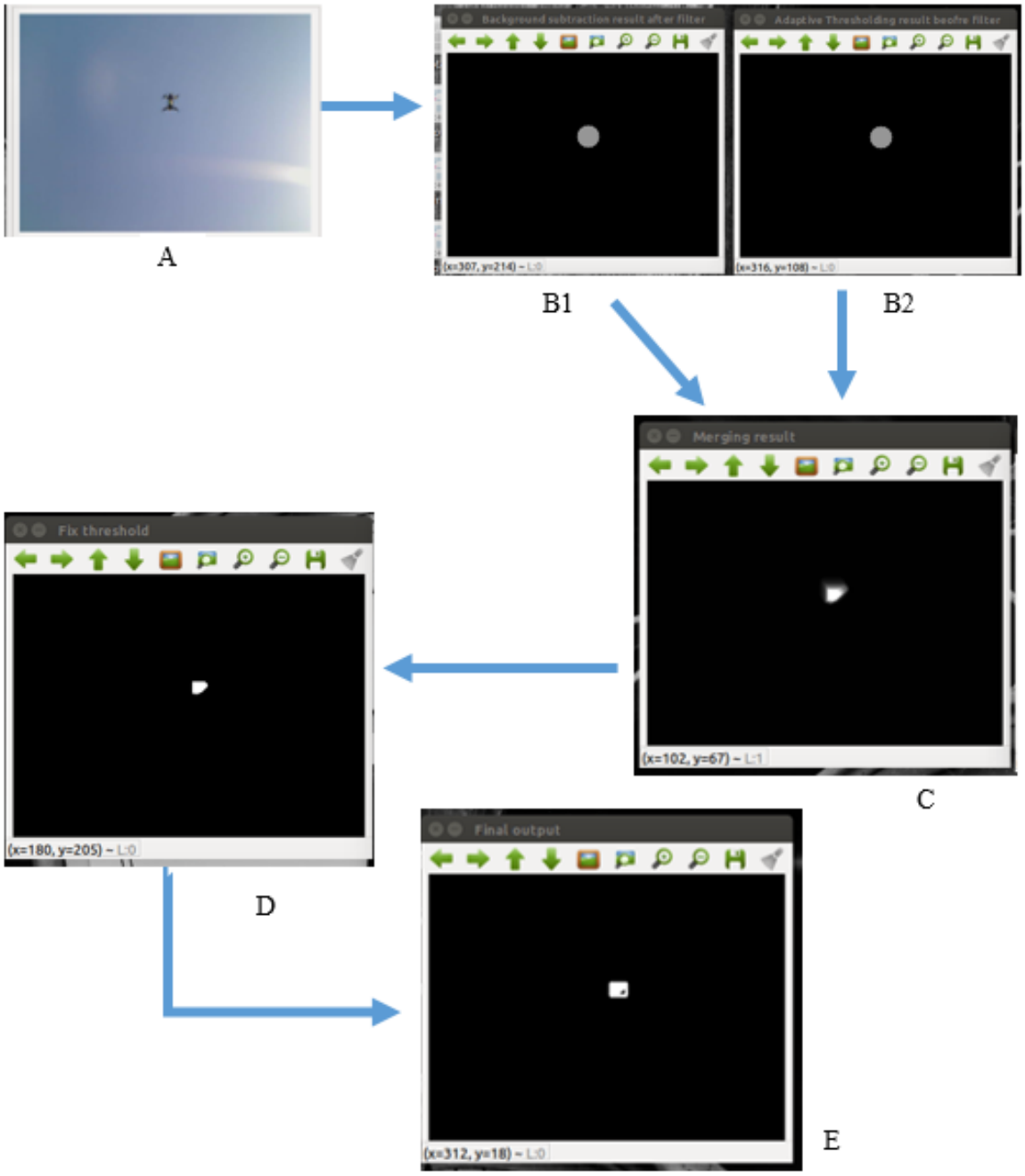


Figure 4.6: (A) Original frame, (B) Background Subtraction, (B2) Adaptive thresholding, (C) Merging, (D) Fix Threshold, (E) Final Detection/History.

Table 1. Observation of detection performance under different environments

Outdoor condition	Case drone was in frame			Case drone was not in frame		
	Number of frame tested	True positive	True negative	Number of frame tested	False Positive	False negative
Cloudy with little bit of blue sky	1138	1109	247	351	0	351
Cloudy less bright day	904	662	242	697	11	686
Clear sky bright day	403	160	243	815	0	815
Heavy Sun glare	1184	36	1158	104	65	39

4.4. IR beacon test

The following Figure 4.7 shows the test that was done using IR LOCK sensor and PixyMon software. PixyMon is a free software provided by PIXY (CMUcam5 Pixy – CMUcam, 2017). The software allows the user to observe live detection on the CMUcam5 pixy. The test verified that the pixy is ready to be mounted on the quad.

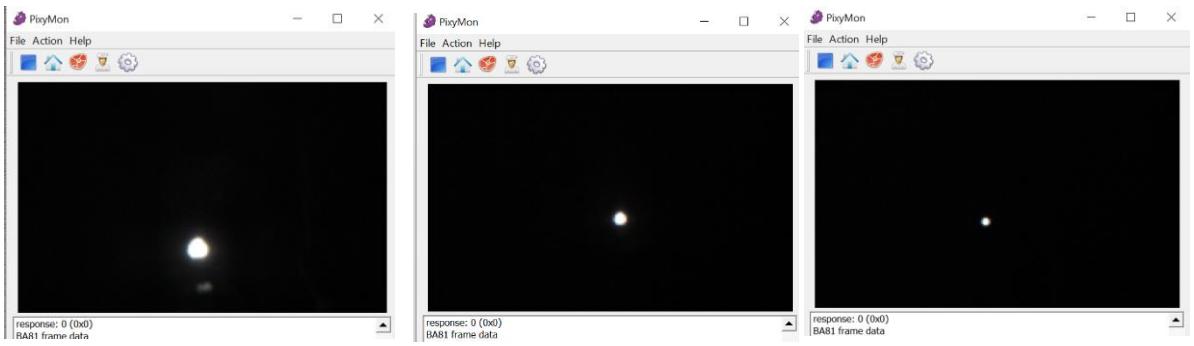


Figure 4.7: (A) Beacon at 1-meter distance, (B) Beacon at 5-meter distance, (C) Beacon at 7-meter distance

4.5. Autonomous landing result

The first integration test reveals that the IR lock does not communicate with flight controller. As a result, the system is force to land when the rangefinder sense altitude less than half a meter. The Figure 4.8 shows the flight-data from the flight controller. It shows two successful landing attempt of take-off and landing. The figure consists of three graphs. The first graph is throttle position. This by looking at the throttle position graph, it is known that there were to take-off and landing process on this graph. Due to the configuration of the remote control (R/C), the most minimum throttle is 1000.

The second graph is Precision Landing Target Angular Offset, which indicates the angular position of the drone with respect to the centre of camera's FOV during landing. The angular position is set by visual positioning system. The Green waveform represents Y-axis angle and red waveform represent the X-axis angle.

When the drone was above one meter, it is clearly shown in the graph that the visual positioning was able to correct the position of the quad while it is descending. However, when it reached altitude less than one-meter, the quad started to lose its position. The error mostly come from wind disturbance. At this particular fly test, the landing pad was raised about 0.3 meter to avoid ground effect. As a result, the quad is land before the rangefinder sense zero altitude. This is clearly shown in the graph where the quad is actually landing a little bit off form the centre of the camera.

As shown by orange circles, the first attempt was off few degrees on the positive Y-axis of the drone body while the second attempt was off few degrees on the negative X and positive Y-axis of drone body. In addition, there are also overshoot during the execution of the given angle correction when the drone was descending. The second attempt show better response compare to the previous one. One common condition that happened on both attempt is before initiating the land, the drone has to be in the camera FOV. The maximum camera offset in degree was about 50 degree and both of the waveforms clearly show that they are under 50 degree during the landing process.

The third graph plots the Rangefinder reading and target lock indicator. The target lock indicator is indicated by the blue waveform, and it is set by navigation system by reading the confidence level setting by visual positioning. History based system is added to compromise the effect of short occlusion since the probability of true negative given the target was in the FOV is high at certain light conditions. By having history-based system, when the new reading detects no target due to short occlusion, the history will output the old reading for a period of time. However, after several iteration, that output will not be valid anymore since there is no actual update. This confidence level is added to let the navigation system know that that output is no longer valid, so the amount of correction given to the quad should be minimised until the new update arrive. Obviously after several time, not receiving any update, the target lock will be set to zero. As can be seen, on this particular flight, the visual positioning was successfully directing the target to the desire area without losing the target.

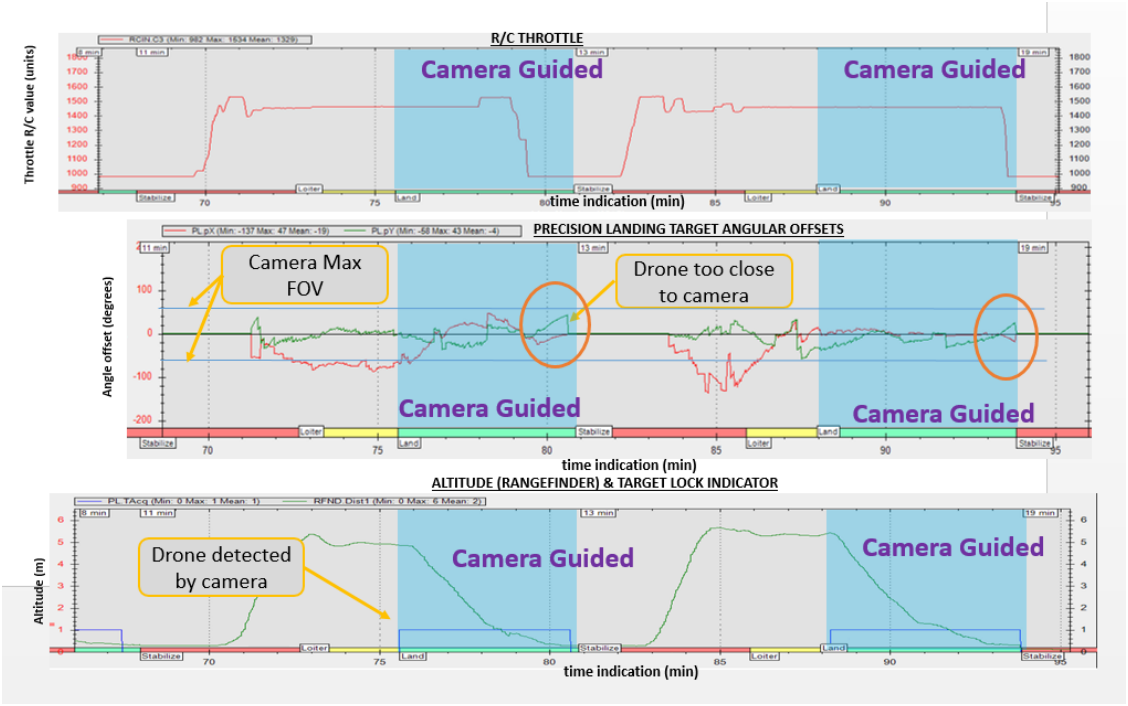


Figure 4.8: (Top) R/C throttle, (Middle) Precision Landing Target Angular Offset, (Bottom) Rangefinder and target lock indicator

Without the short-range guidance, the long range-guidance is still capable to guide the quad precisely. Form 30 attempt of landing, nine of them were less than one-meter form the camera, fifteen of it were less than two meter, and rest are combination of land beyond two matter, and aborted landing. This low rate of successful land is caused by ground affect at last half a meter. When the landing pad does not have ground affect compensation. The following Figure 4.9 shows the screen capture one of the successful landing. As can be seen, the quad was descending smoothly when it was above one meter, but when it reached the last one meter, the quad was struggle, but eventually it was able to land in the centre of the camera.

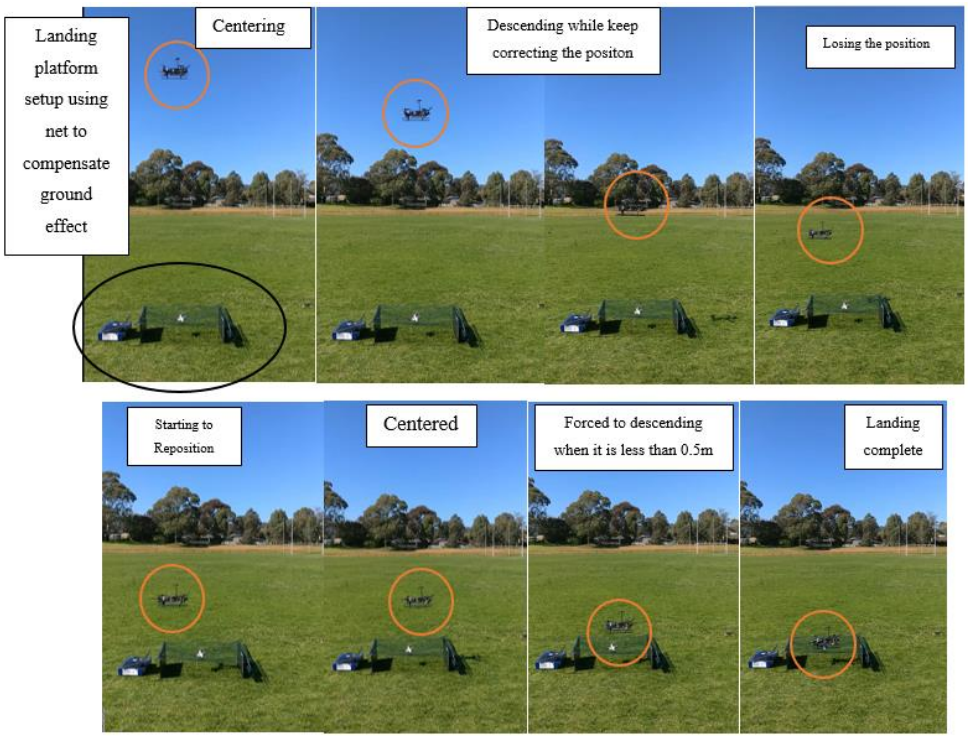


Figure 4.9: Screen capture of the landing process

4.6. Autonomous tracking

Another test that was done is precision loiter. The precision loiter test prove that the system can be use not only for landing but also for tracking the camera. The Figure 4.10 shows that when the camera was moved the quad was following.

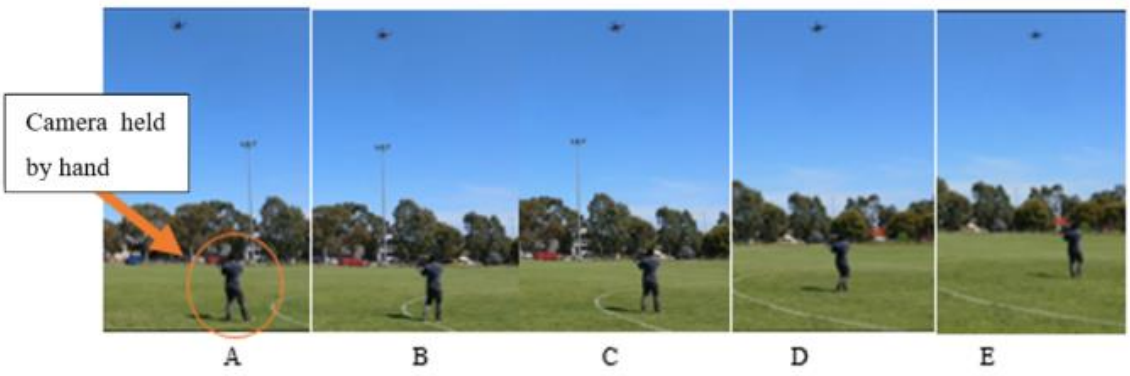


Figure 4.10: Precision loiter by tracking camera position.

5. DISCUSSION & ANALYSIS

The results section above showed several tests results. The first test's aim was to determine the suitable parameters for the background subtraction. The test was performed by observing the effect of varying each parameter while keeping the other at its default value. Also, the test was performed in the same environmental conditions, which was an indoor environment. The output of this test concluded that there is not fixed values that would work perfectly for all of the outdoor conditions. However, a set of values was decided to be used by sacrificing the performance of the method in certain conditions. The second test's purpose was to observe the performance of the background subtraction when doing object detection in different outdoor environments. The test concluded that using background subtraction only for detection was not enough.

To resolve the limitations of the background subtraction detection, the adaptive thresholding was added. The offline test result showed that the method combining these two algorithms is working. The detection was reasonably accurate with small weaknesses under certain conditions due to compromise made in the early stage. Several extra processes was added and tested in order to improve the performance. Firstly, to overcome over brightness problem, the blue channel of input frame was removed before the frames were processed. However, the effect was the other way around. Instead of improving the performance, the change reduced the sensitivity of the background subtraction because less colour was left in the model; hence, the more pixels looked the same. The second improvement was to adjust the contrast of the input image before being processed to overcome dim backgrounds. The first intention was to use local adjustment; however, due to the complexity of the process, this improvement will be added as future work for improvement, and instead global adjustment was used. Although this will not always be the case, due to auto exposure adjustment from the camera itself and high contrast on the drone due to backlit, the global contrast adjustment was enough to adjust some under exposure areas created by auto exposure and retain the detail of the quad. Obviously, this was going to be a problem for over exposure case.

During the development stage of the combination method, scale invariant and frame matching methods was considered. One of the advantages of using scale invariant was it was not affected by size of the object. However, the requirement to have pre-defined library which contain collection of all the possible angles and views of the quad lead the author to not use it because there were various possible working environments that needed to be covered, and to cover each

of those case, the library will need to be really large. The same reason was also applied for not using frame matching which has been commonly applied with SURF, ORB, and SHIFT algorithms.

Besides approaching this task from an image processing point of view, applying active tracking systems such as the Kalman filter and Particle filter were also considered. Limited time constrain prevented the author from doing further investigation in this area.

Furthermore, the overall performance of the autonomous landing was admirable although not as superior as expected. The long-range detection showed a significant correction on the positioning as seen in the results section. The system was able to centre the drone to the camera FOV. Another quick test, called autonomous tracking, had also been done to prove that the system was compatible with moving carrier. The system was quick enough to continuously correct the position of the drone when the camera was moving. This indicated that the system was partially ready to be attached to the final landing platform, which was to be placed on the rear of the TopCat.

The test results showed that from 30 attempts of landing, the quad had 9 successful landings; however, there were only a few cases where it was actually landed on the top of the camera. The most common problem of missing the landing was due to the absence of the short-range detection. Without the short detection, there was no correction provided for the last meter; therefore, whenever the drone was close to the camera, the drone would always loose its positioning. The individual test on the IR LOCK that had been done showed that the IR lock was able to detect the IR beacon easily. However, when the IR LOCK was connected to PIXHAWK as suggested on the IR LOCK manual, an error "BAD VISION POSITION" started to appear in the Mission Planner software. This error indicated the PIXHAWK was not able to access the IR LOCK sensor for precision landing. Further study on the flight controller response showed that the PL.px and PL.py indication parameters for the precision landing were not responding at all. Reconfiguring the parameters yielded no improvement. Increasing the voltage input from a regulated 5V, which is actually 4.5V in the connection, to unregulated 7VDC source, also did not create any change. One possible solution that has not been tested yet is swapping the LeddarOne rangefinder sensor to the SF10 rangefinder sensor, which is the sensor that is used by IR lock for their development. The earlier decision not to buy the SF10 was because it was overpriced for project's budget and LeddarOne was fully supported by PIXHAWK (Keen, 2017). This possible solution is reserved for future improvement. Altering

the IR beacon detection with a detection system using shape extraction could be a quick solution to overcome the gap on the short range. The complete approach is available in Appendix A. The method has been created, but not integrated yet to the overall system due to inability of the detection algorithm to filter the actual target from the noise.

Finally, besides developing the algorithm, there was also extra work that had been done in order to prepare the quad for the next stage, namely installing the First Person View (FPV) camera for aerial monitoring task. The setup is in Appendix C.

5.1. Recommendations for Future Research

First, testing the compatibility of the IR LOCK with SF10. This is the easiest solution for the short-range detection. Making the IR LOCK work will save a lot of time for the other projects.

Secondly, integrate a tracking system such as the Kalman filter or Particle filter by fusing the GPS sensors to improve the positioning system during short occlusion. From the discussion section above, using visual detection approach only for autonomous landing is not an ideal solution, because it is only makes the system reactive and not predictive. To get a fully robust system, a predictive system such as a tracking system is an essential component of the reactive system.

However, due to the inaccuracy of the on-board GPS, a GPS correction needs to be applied before applying sensor fusion. The GPS correction can be done by using RTK corrections (Keen, 2017) direct GPS correction used by Yi-ChengLu 2017.

In addition, because the current angle correction is a ready-to-use function from MAVLink, the correction given to the quad is fixed proportionally to the angle given. As a result, the correction sent in calm conditions and windy condition will be always be the same, which will result in an overshoot when the wind is in the same direction with the drone and vice versa. Adding a wind sensor would be a solution. By having a wind sensor, the angle correction could be tweaked to adapt to the wind conditions. As a result, more correction will be sent if the wind is in opposite direction to the drone's desired velocity and less correction will be sent if the wind is the same direction as the drone.

Lastly, to reduce the power consumption of the visual guidance, a python script can be developed to run this visual guidance only when the quad is ready to land and stop when the quad is performing a mission or has already landed.

6. CONCLUSION

In conclusion, the overall aim of this project is not been fully achieved yet, but the specific aim of this research has been achieved. The chosen methods, which are the combination of the background subtraction and the adaptive thresholding as the long-range detection and the IR detection as the short-range detection, have been implemented and tested. Several landing test results show that the long-range detection is capable of landing the quadrotor safely without the short-range detection. The long-range detection also allows the quadrotor to follow the position of the camera when the camera is moved.

Further investigation is required to find out why the IR LOCK sensor is not being recognised by the flight controller. On the other hand, improvement is still required in the long-range detection algorithm. The first is the system needs to be reactive and predictive to improve the robustness of the detection and tracking of the target's position. In addition, the system needs to have an independent controller or a system to correct the angle correction from the flight controller. Moreover, the system needs to have redundancy in case of failure.

A few finding from this project are that the background subtraction is a method that suitable for a project that involves a camera, attached to the landing pad, looking upward, while another detection method such as colour extraction, IR detection, and shape extraction is more applicable for project that involves a camera looking upward.

7. REFERENCES

CHANDRA, S., CHAPMAN, R., DIVERDI, R., PRESTON, V., WOO, J., BENNETT, A. & BARRETT, D. Protocol for autonomous landing of unmanned air vehicles on research vessels. OCEANS 2016 MTS/IEEE Monterey, 19-23 Sept. 2016 2016. 1-5.

Changing the contrast and brightness of an image! — OpenCV 2.4.13.4 documentation. 2017. Changing the contrast and brightness of an image! — OpenCV 2.4.13.4 documentation. [ONLINE] Available at: http://docs.opencv.org/2.4/doc/tutorials/core/basic_linear_transform/basic_linear_transform.html. [Accessed 16 October 2017].

CMUcam5 Pixy - CMUcam: Open Source Programmable Embedded Color Vision Sensors. 2017. Overview - CMUcam5 Pixy - CMUcam: Open Source Programmable Embedded Color Vision Sensors. [ONLINE] Available at: http://cmucam.org/projects/cmucam5. [Accessed 16 June 2017].

CNET. 2017. Logitech QuickCam Orbit AF Specs - CNET. [ONLINE] Available at: https://www.cnet.com/products/logitech-quickcam-orbit-af/specs/. [Accessed 20 June 2017].
DirkThomas. 2017. "ROS/Introduction" [Online]. Available: http://wiki.ros.org/ROS/Introduction. [Accessed 12 June 2017]

GAUTAM, A., SUJIT, P. B. & SARIPALLI, S. A survey of autonomous landing techniques for UAVs. 2014 International Conference on Unmanned Aircraft Systems (ICUAS), 27-30 May 2014 2014. 1210-1218.

GitHub. 2017. GitHub - squilter/target-land: land a 3DR Solo on a colorful target. [ONLINE] Available at: https://github.com/squilter/target-land. [Accessed 16 October 2017].

GUI, Y., GUO, P., ZHANG, H., LEI, Z., ZHOU, X., DU, J. & YU, Q. 2013. Airborne Vision-Based Navigation Method for UAV Accuracy Landing Using Infrared Lamps. Journal of Intelligent & Robotic Systems, 72, 197-218.

HDL-32E. 2017. HDL-32E. [ONLINE] Available at: http://velodynelidar.com/hdl-32e.html. [Accessed 16 October 2017].

image_proc - ROS Wiki. 2017. image_proc - ROS Wiki. [ONLINE] Available at: http://wiki.ros.org/image_proc. [Accessed 16 October 2017].

ISO 20473:2017

KHITHOV, V., PETROV, A., TISHCHENKO, I. & YAKOVLEV, K. 2017. Toward Autonomous UAV Landing Based on Infrared Beacons and Particle Filtering. In: KIM, J.-H., KARRAY, F., JO, J., SINCAK, P. & MYUNG, H. (eds.) Robot Intelligence Technology and Applications 4: Results from the 4th International Conference on Robot Intelligence Technology and Applications. Cham: Springer International Publishing.

KIM, J., JUNG, Y., LEE, D. & SHIM, D. H. Outdoor autonomous landing on a moving platform for quadrotors using an omnidirectional camera. 2014 International Conference on Unmanned Aircraft Systems (ICUAS), 27-30 May 2014 2014. 1243-1252.

KONG, W., ZHOU, D., ZHANG, D. & ZHANG, J. Vision-based autonomous landing system for unmanned aerial vehicle: A survey. 2014 International Conference on Multisensor Fusion and Information Integration for Intelligent Systems (MFI), 28-29 Sept. 2014 2014. 1-8.

LANGE, S., SUNDERHAUF, N. & PROTZEL, P. A vision based onboard approach for landing and position control of an autonomous multirotor UAV in GPS-denied environments. 2009 International Conference on Advanced Robotics, 22-26 June 2009 2009. 1-6.

LEE, D., RYAN, T. & KIM, H. J. Autonomous landing of a VTOL UAV on a moving platform using image-based visual servoing. 2012 IEEE International Conference on Robotics and Automation, 14-18 May 2012 2012. 971-976.

message_filters - ROS Wiki. 2017. message_filters - ROS Wiki. [ONLINE] Available at: http://wiki.ros.org/message_filters. [Accessed 16 October 2017].

NEW IMAGING TECHNOLOGIES, http://new-imaging-technologies.com/en/wdr-sensor-ingaas/swir-camera-hdr/page-products-sans-features/-7351665-7351691-widy-swir-320u-s.html

OpenCV 3.0.0-dev documentation. [ONLINE] Available at: http://docs.opencv.org/3.0-beta/doc/tutorials/imgproc/gausian_median_blur_bilateral_filter/gausian_median_blur_bilateral_filter.html. [Accessed 16 October 2017].

OpenCV: Background Subtraction. 2017. OpenCV: Background Subtraction. [ONLINE] Available at: http://docs.opencv.org/3.3.0/db/d5c/tutorial_py_bg_subtraction.html. [Accessed 16 October 2017].

OpenCV: cv::bgsegm::BackgroundSubtractorMOG Class Reference. 2017. OpenCV: cv::bgsegm::BackgroundSubtractorMOG Class Reference. [ONLINE] Available at: https://docs.opencv.org/3.2.0/d6/da7/classcv_1_1bgsegm_1_1BackgroundSubtractorMOG.html. [Accessed 16 October 2017].

OpenCV: Histograms - 2: Histogram Equalization. 2017. OpenCV: Histograms - 2: Histogram Equalization. [ONLINE] Available at: http://docs.opencv.org/3.1.0/d5/daf/tutorial_py_histogram_equalization.html. [Accessed 7 August 2017].

OpenCV: How to Use Background Subtraction Methods. 2017. OpenCV: How to Use Background Subtraction Methods. [ONLINE] Available at: https://docs.opencv.org/3.2.0/d1/dc5/tutorial_background_subtraction.html. [Accessed 16 October 2017].

OpenCV: Image Thresholding. 2017. OpenCV: Image Thresholding. [ONLINE] Available at: https://docs.opencv.org/trunk/d7/d4d/tutorial_py_thresholding.html. [Accessed 16 October 2017].

OpenCV: Morphological Transformations. 2017. OpenCV: Morphological Transformations. [ONLINE] Available at: http://docs.opencv.org/trunk/d9/d61/tutorial_py_morphological_ops.html. [Accessed 16 October 2017].

PINTO, E., MARQUES, F., MENDONÇA, R., LOURENÇO, A., SANTANA, P. & BARATA, J. An autonomous surface-aerial marsupial robotic team for riverine environmental monitoring: Benefiting from coordinated aerial, underwater, and surface level perception. 2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014), 5-10 Dec. 2014 2014. 443-450.

RUAG Group, 2017. "https://ruag.picturepark.com/Go/D4xnjmJf/V/7623/1"

RAU, S. H., ZHANG, Z., LEE, W. J. & DINI, D. A. 2017. Arc Flash Visible Light Intensity as Viewed from Human Eyes. IEEE Transactions on Industry Applications, PP, 1-1.

SAMMUT, K., WHEARE, J., LAMMAS, A., DONNELLY, B., CROUCH, T., WEBB, A., STEWART, J., KOSSATZ, M., HUTCHINSON, S. & GEYER, S. 2016. Development and Testing of the TopCat Autonomous Surface Vessel for the Maritime RobotX Challange 2016.

SARIPALLI, S., MONTGOMERY, J. F. & SUKHATME, G. S. Vision-based autonomous landing of an unmanned aerial vehicle. Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292), 2002 2002. 2799-2804.

SENSOR UNLIMITED. 2017 “http://www.sensorsinc.com/products/detail/micro-320csx-swir-camera”

SIERRA NEVADA CORPORATION, 2017a. "https://www.sncorp.com/media/1998/ucars-v2product-sheet.pdf"

SIERRA NEVADA CORPORATION, 2017b. "https://www.sncorp.com/media/2000/dual-thread-product-sheet2010.pdf"

Smoothing Images — OpenCV 3.0.0-dev documentation. 2017. Smoothing Images —

THORLABS. 2017, "https://www.thorlabs.com/newgrouppage9.cfm?objectgroup_id=5871"

TIAN, D. P. 2013. A Review on Image Feature Extraction and Representation Techniques. International Journal of Multimedia and Ubiquitous Engineering, 8, 11.

usb_cam - ROS Wiki. 2017. usb_cam - ROS Wiki. [ONLINE] Available at: http://wiki.ros.org/usb_cam. [Accessed 16 October 2017].

WEAVER, J. N., FRANK, D. Z., SCHWARTZ, E. M. & ARROYO, A. A. 2013. UAV PERFORMING AUTONOMOUS LANDING ON USV UTILIZING THE ROBOT OPERATING SYSTEM.

WENZEL, K. E., MASSELLI, A. & ZELL, A. 2011. Automatic Take Off, Tracking and Landing of a Miniature UAV on a Moving Carrier Vehicle. Journal of Intelligent & Robotic Systems, 61, 221-238.

YI-CHENGLU. 2017. The Guidance and Control of Variable-pitch Quadrotors - Autonomous Landing and Take-off on a Moving Vehicle. Ph.D, 國立成功大學 (National Cheng Kung University).

YILMAZ, A., JAVED, O. & SHAH, M. 2006. Object tracking: A survey. ACM Comput. Surv., 38, 13.

ZHANG, F., YANG, L. & ZHANG, G. Adaptive fast Gaussian background subtraction algorithm. Proceedings of 2012 2nd International Conference on Computer Science and Network Technology, 29-31 Dec. 2012 2012. 766-771.

ZIVKOVIC, Z. Improved adaptive Gaussian mixture model for background subtraction. Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004., 23-26 Aug. 2004 2004. 28-31 Vol.2.

APPENDIX A: Shape detection using concentric circle

The possible shape extraction that is use for short guidance is using collection of concentric white circles on a black background (Lange et al., 2009). The shape is design to be attached underneath the frame of the UAV. The algorithm is similar to the one that had been done before. The following figure 1 illustrated the shape design. An extra features is added in to the shape design to provide heading direction of the quad.

Figure 1. Shape designed for short guidance

APPENDIX B: background subtraction and adaptive thresholding other testing result

The following figures shows the system works under several test environments. Due to low intensity on the actual output of the background subtraction and adaptive thresholding, the results that are shown are modified intensity to make it visible for observation. Without change the actual process.

Case 1: Result on Clear sky with little bit of sun glare

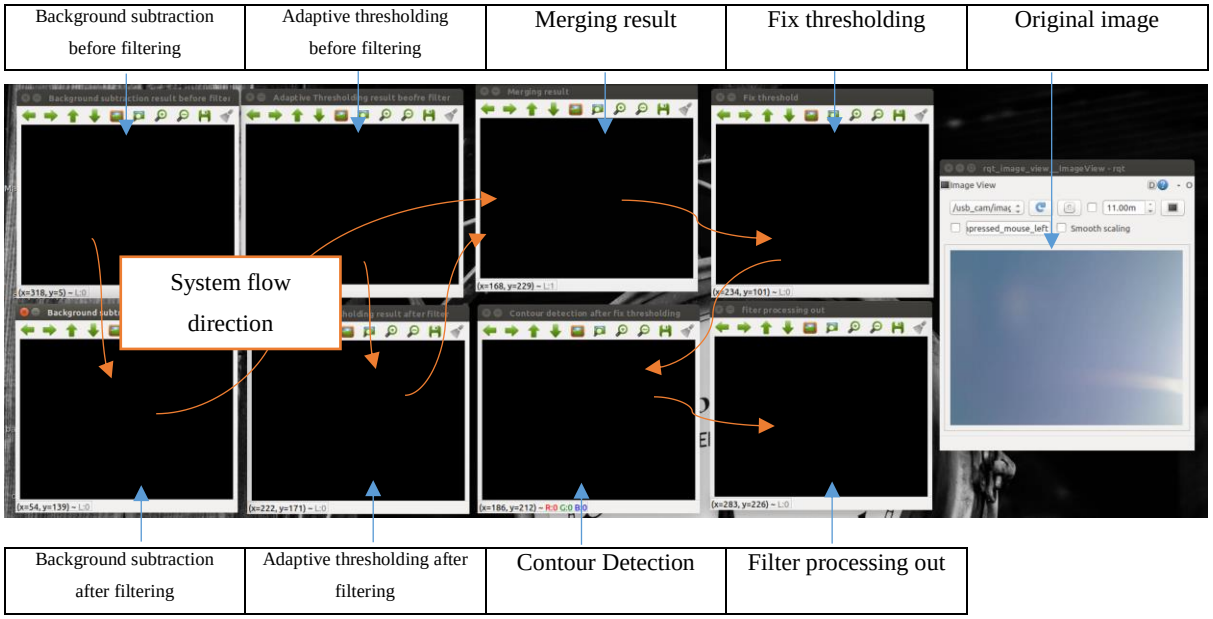


Figure 1. No positive detection

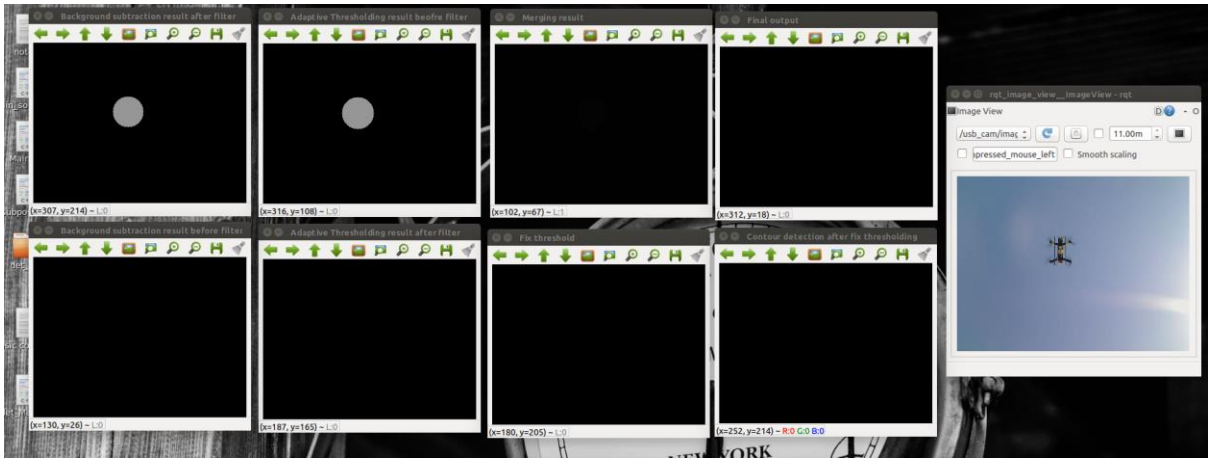


Figure 2. Target is found on each two nodes, but the intensity is not built up yet

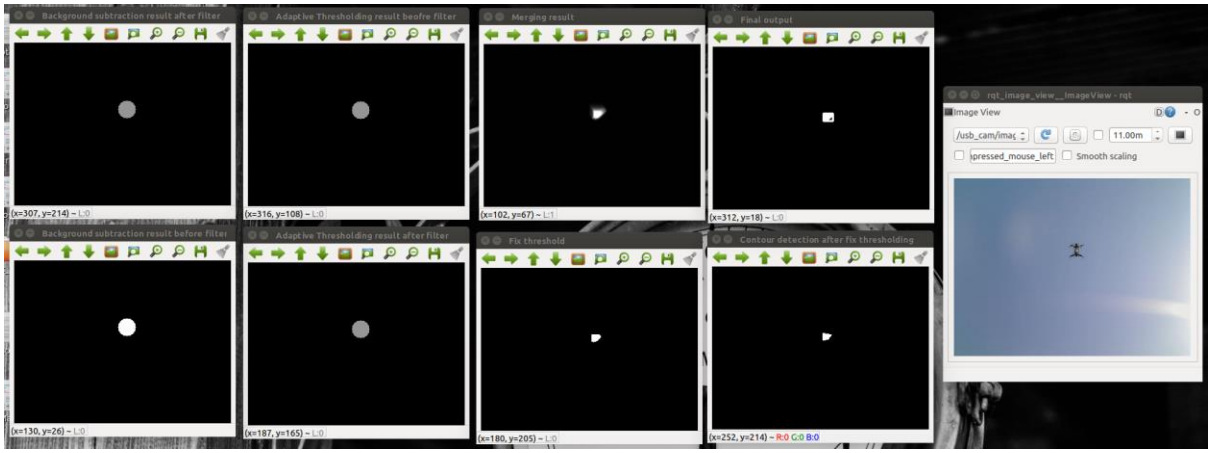


Figure 3. Target is found

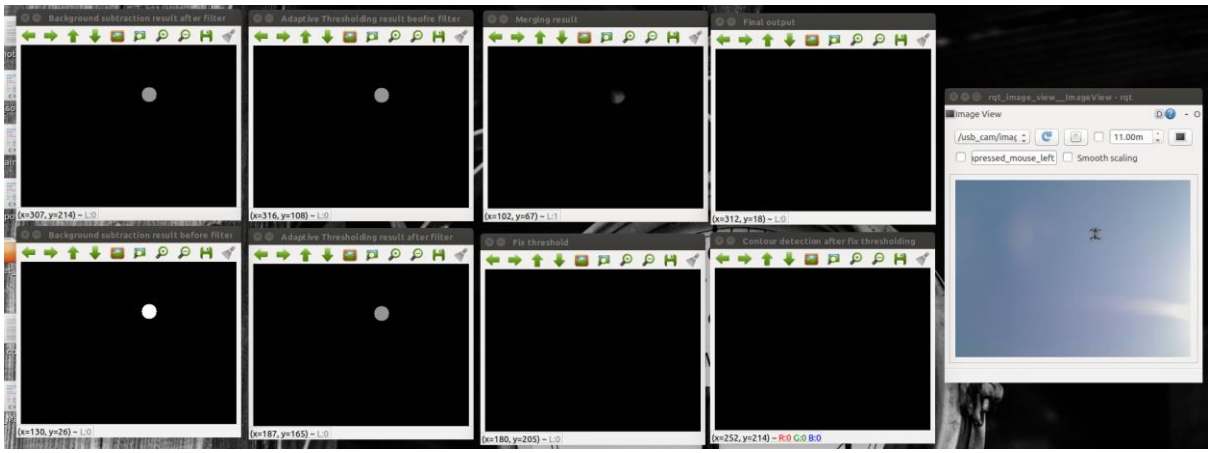


Figure 4. Target is found, but the intensity is too low to pass the fix thresholding

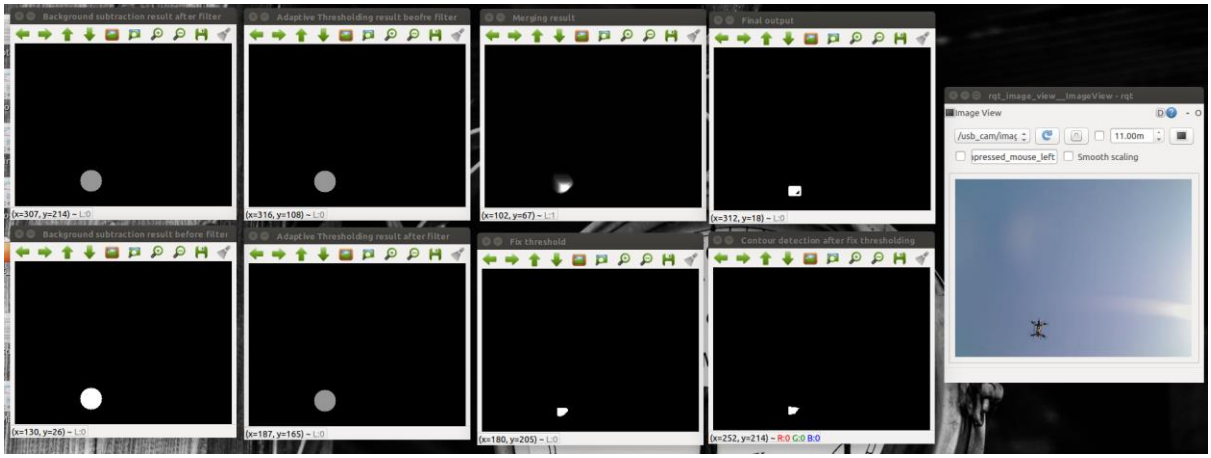


Figure 5. Target recovered

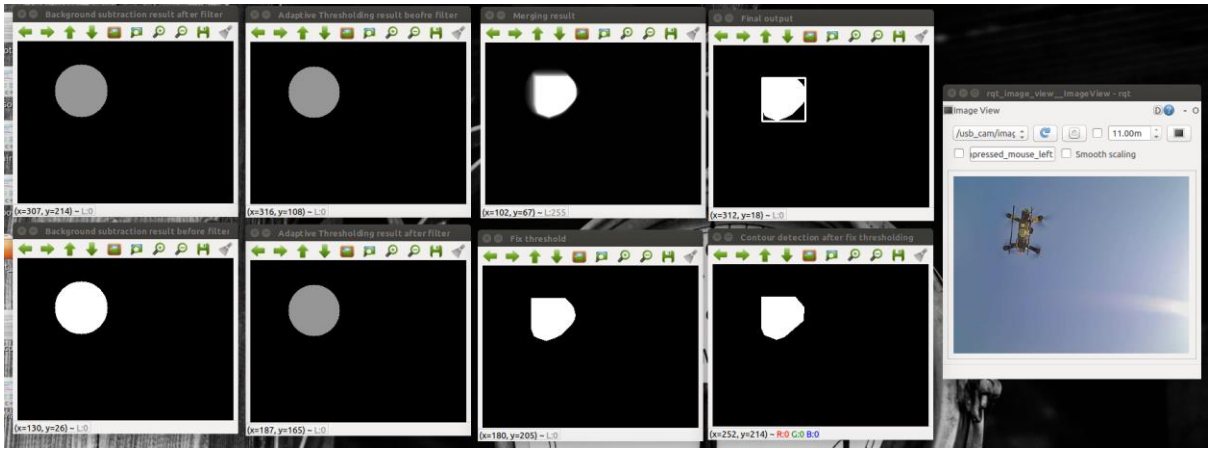


Figure 6. The target getting bigger as the drone start descending

Case 2: Cloudy with darker sky

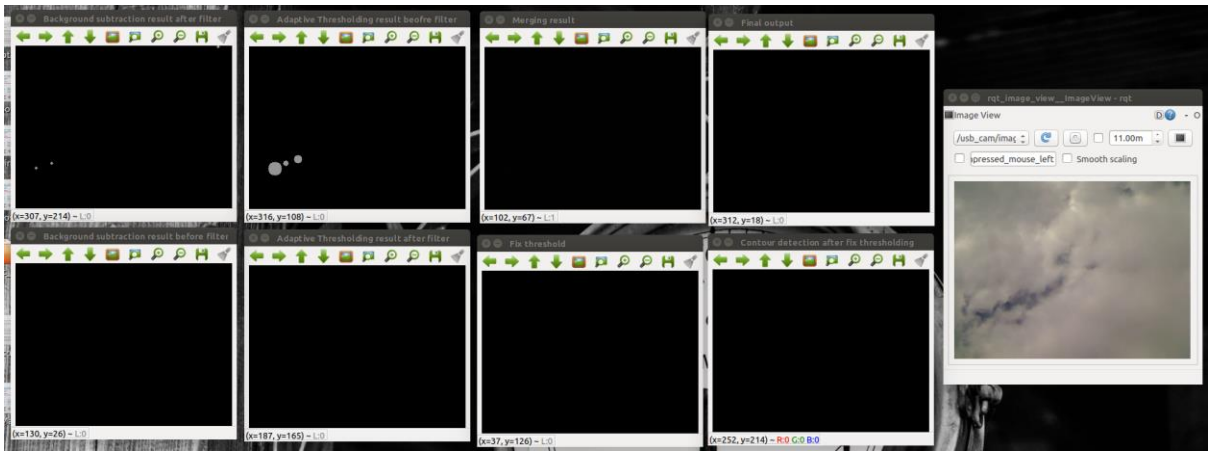


Figure 7. False positive detections are removed

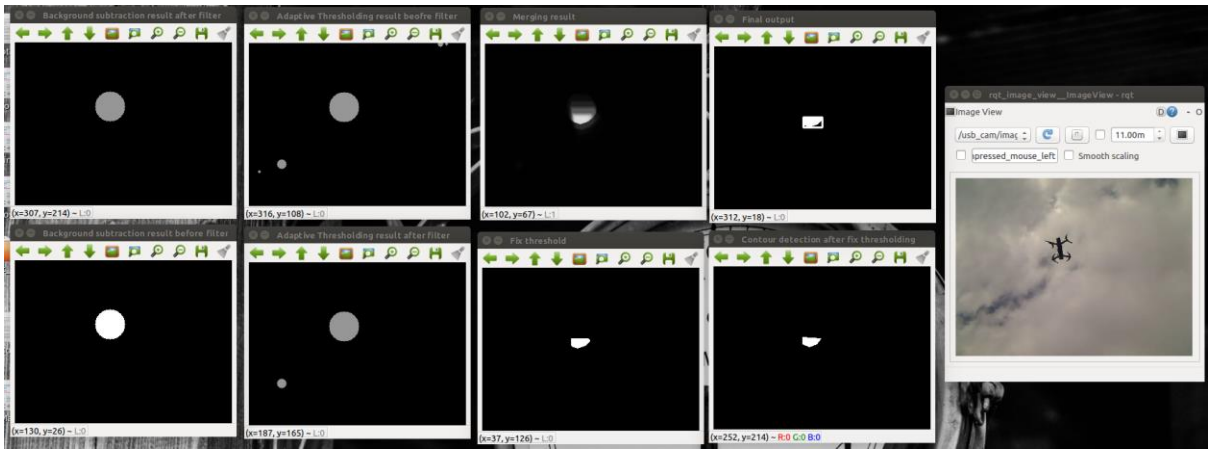


Figure 8. Target is found and noises are removed

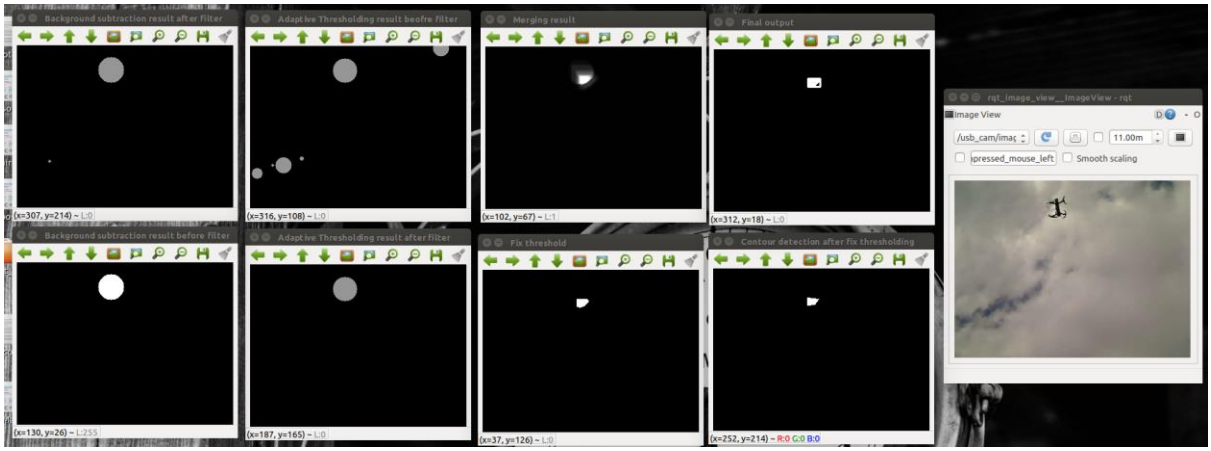


Figure 9. Target is found, noises are removed, and uncertainty increases

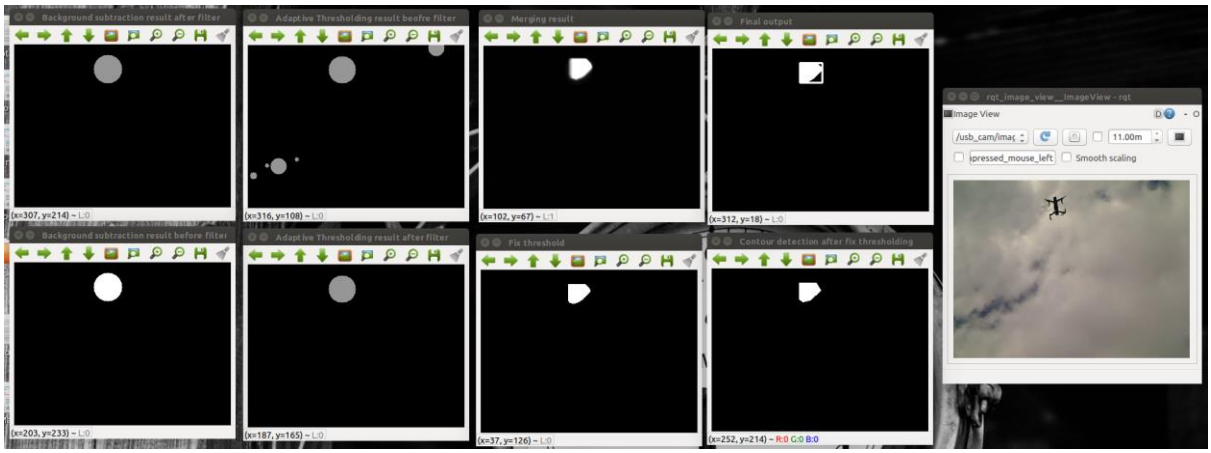


Figure 10. Target is found, noises are removed, and the uncertainty decreases

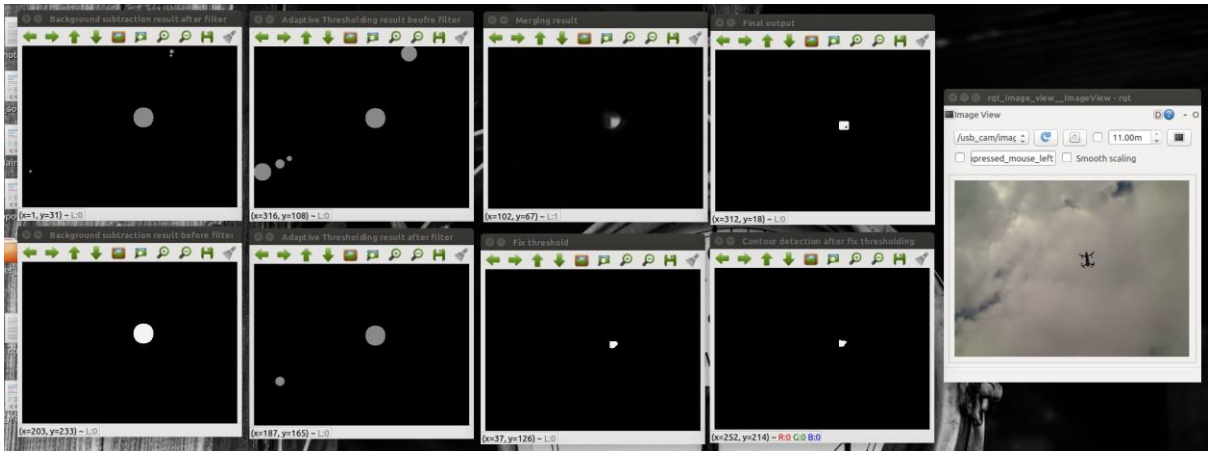


Figure 11. Target is found, noises are removed, and the uncertainty increases

Case 3: Cloudy with bright sky

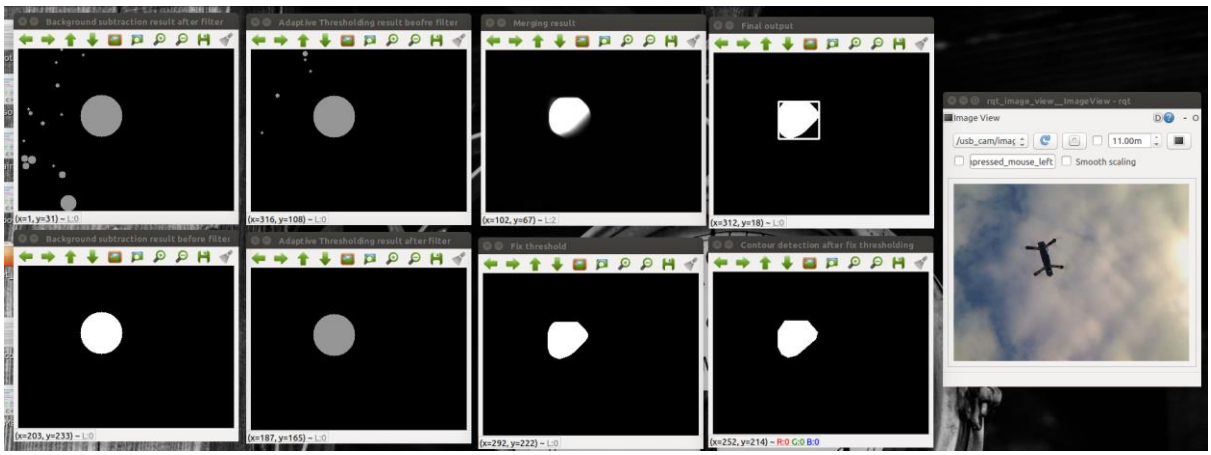


Figure 12. Target is found, noises are removed, and the uncertainty decreases.

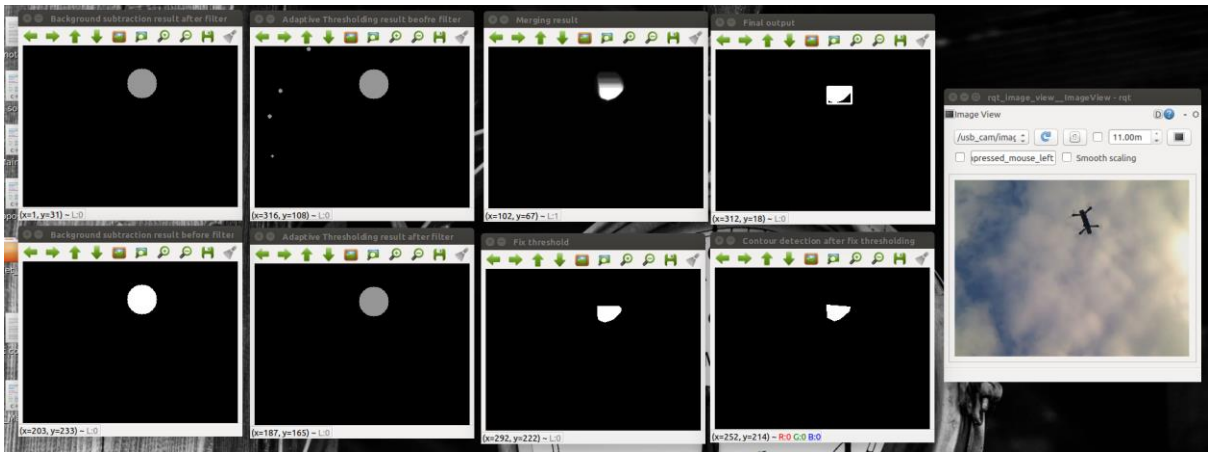


Figure 13. Target is found, noises are removed, and the uncertainty increases.

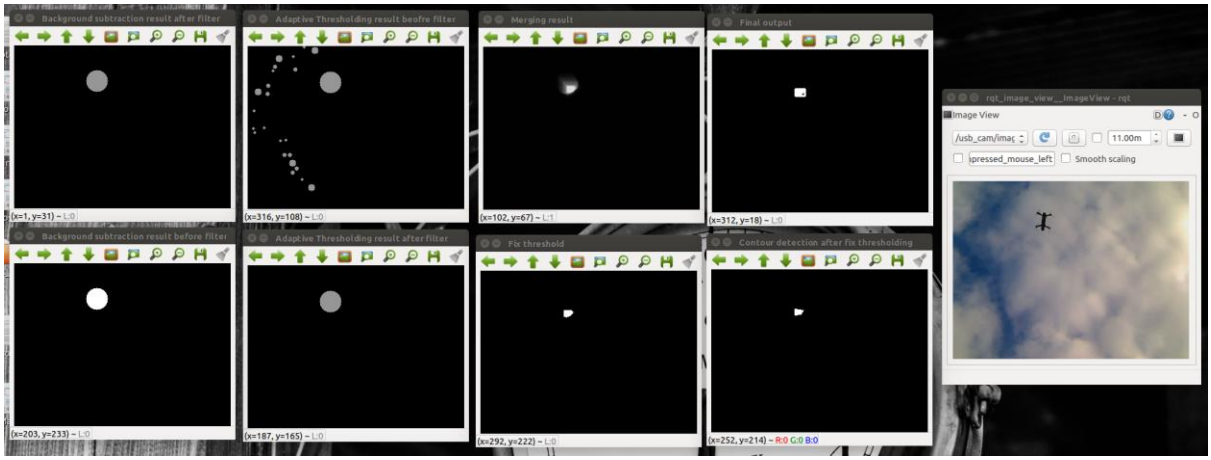


Figure 14. Target is found, noises are removed, and the uncertainty increases

APPENDIX C: First Person Video Setup

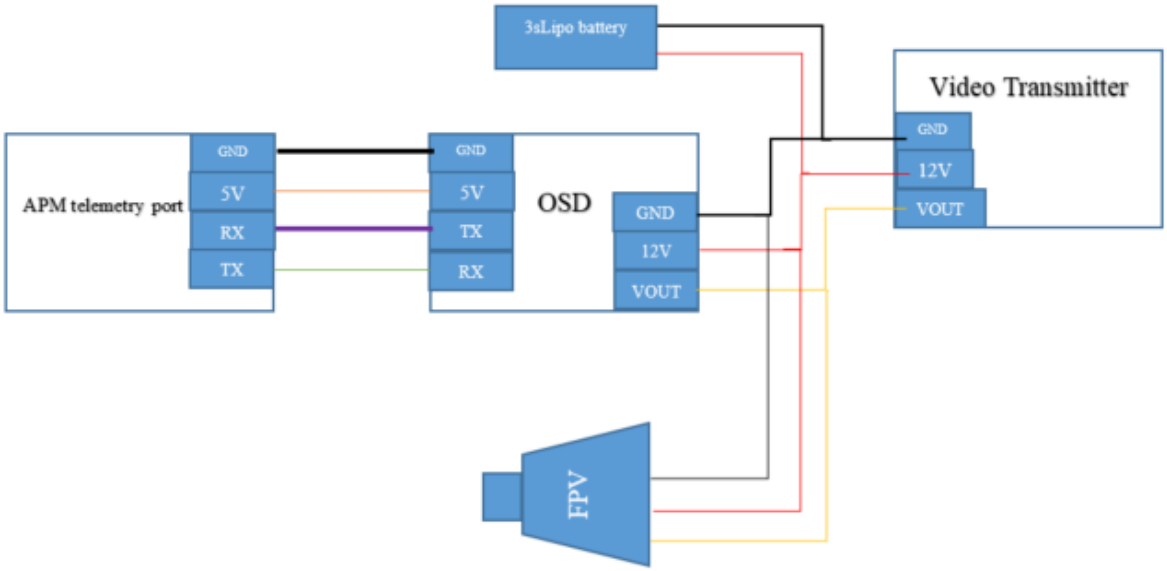
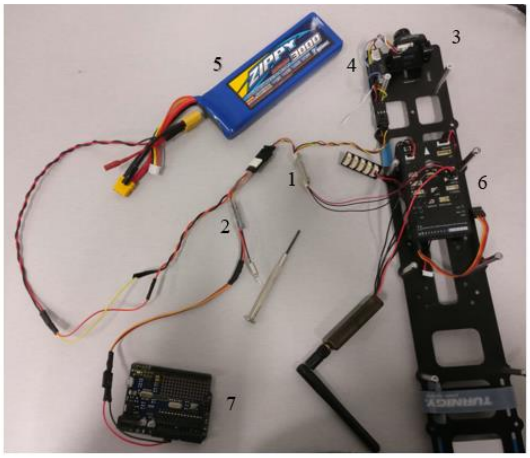


Figure 1. FPV wire diagram



1	OSD
2	5.8GHz transmitter
3	FPV camera
4	Voltage Filter
5	Battery
6	Flight controller
7	Temporary 5VDC

Figure 2. Wiring and hardware

Figure 4. FPV test Result

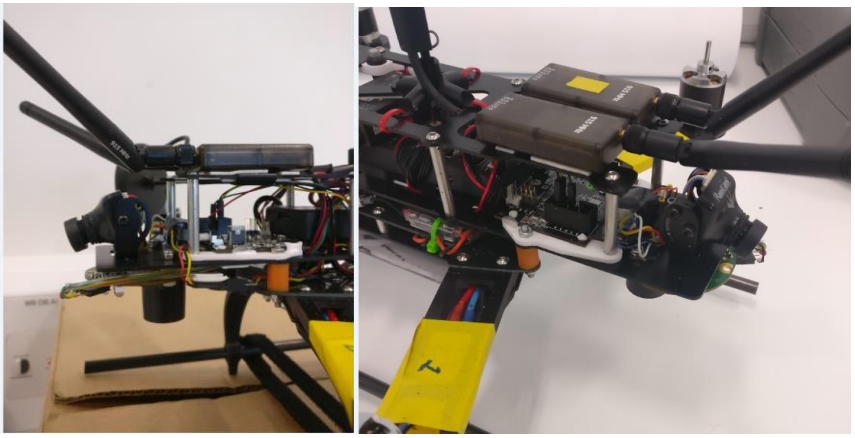


Figure 3. FPV final setup