Bachelor of Science in Electrical Engineering
March 2023

Platform Tilt Detection

For Drone Landing

Hari Veera Mani Kumar Vallu
Meher Vijay Anupoju
Prudhvi Ponnada

School of Engineering, Blekinge Institute of Technology, 371 79 Karlskrona, Sweden

Contact Information:
Author(s):
Hari Veera Mani Kumar Vallu
E-mail: hava22@student.bth.se
Meher Vijay Anupoju
E-mail: mean22@student.bth.se
Prudhvi Ponnada
E-mail: prpo22@student.bth.se

University advisor:
Irina Gertsovich
Department of Mathematics and Natural Sciences

Dept. of Mathematics and Natural Sciences	Internet	:	www.bth.se
Blekinge Institute of Technology	Phone	:	+46 455 38 50 00
SE–371 79 Karlskrona, Sweden	Fax	:	+46 455 38 50 57

Abstract

This report describes a project aimed at determining the angle of the landing platform to the drones using a tilted platform, IMU, and image processing algorithms. The goal of this project is to determine the safe landing of drones and drone landings by optimizing the landing location via platform tilt adjustment and angle computation. A robotic arm was used to tilt the rectangular platform, and a web camera was used to get images from the top view. Image processing methods built-in MATLAB allow for approximate landing angle determination and angle determination is also done with IMU. The findings confirm the system's efficiency, with reasonable angle estimates and successful drone landings.

Keywords: Drone, MATLAB, platform, robotic arm, webcam.

Acknowledgments

We would like to express our profound gratitude to our supervisor, Irina Gertsovich, who guided us through every stage of this thesis. Her encouragement and guidance helped us complete this project.

We would like to express our special thanks to Peter Blaschke, who helped in building our prototype. We personally thank our parents and everyone who inspired us and offered suggestions to improve the quality of this project.

Hari Veera Mani Kumar Vallu
Meher Vijay Anupoju
Prudhvi Ponnada

Contents

List of Figures

List of Tables

List of Acronyms

Chapter 1

Introduction

1.1 Introduction

Unmanned Aerial Vehicles (UAVs) and drones have grown in popularity and are being used in a variety of applications such as surveillance, photography, and delivery services. However, guaranteeing dependable and precise drone landings remains a major challenge. Traditional landing pads frequently lack adaptation to rough terrains and are unable to modify landing angles, resulting in unstable potential damage to the drone [10].

To address these issues, this thesis proposes a unique solution to drone landings that makes use of a rectangular platform with a tilting mechanism and image processing tools. We intend to optimize the landing location for drones by tilting the platform at precise angles and leveraging image processing techniques, enabling safer and more controlled descents.

1.1.1 Background

The background establishes a structure for the thesis and emphasizes the significance of the selected approach. The goal of this project is to create a drone landing system that makes use of a slanted platform and image processing. The analysis of the literature highlights the limits of existing landing pads and the necessity for adjustable and accurate drone landing technologies. The background part also discusses previous research efforts and issues connected to drone landing systems.

1.1.2 Scope of the Thesis

The scope of this project includes a rectangular platform with a tilting mechanism, integrating Arduino microcontrollers (Arduino Nano 33 IoT and Arduino Motor Shield Rev 3) to control the platform tilt, and incorporating image processing algorithms to calculate the platform tilt angle. The thesis also includes the hardware and software implementation components, as well as an analysis of the data acquired.

Chapter 2

Related Work

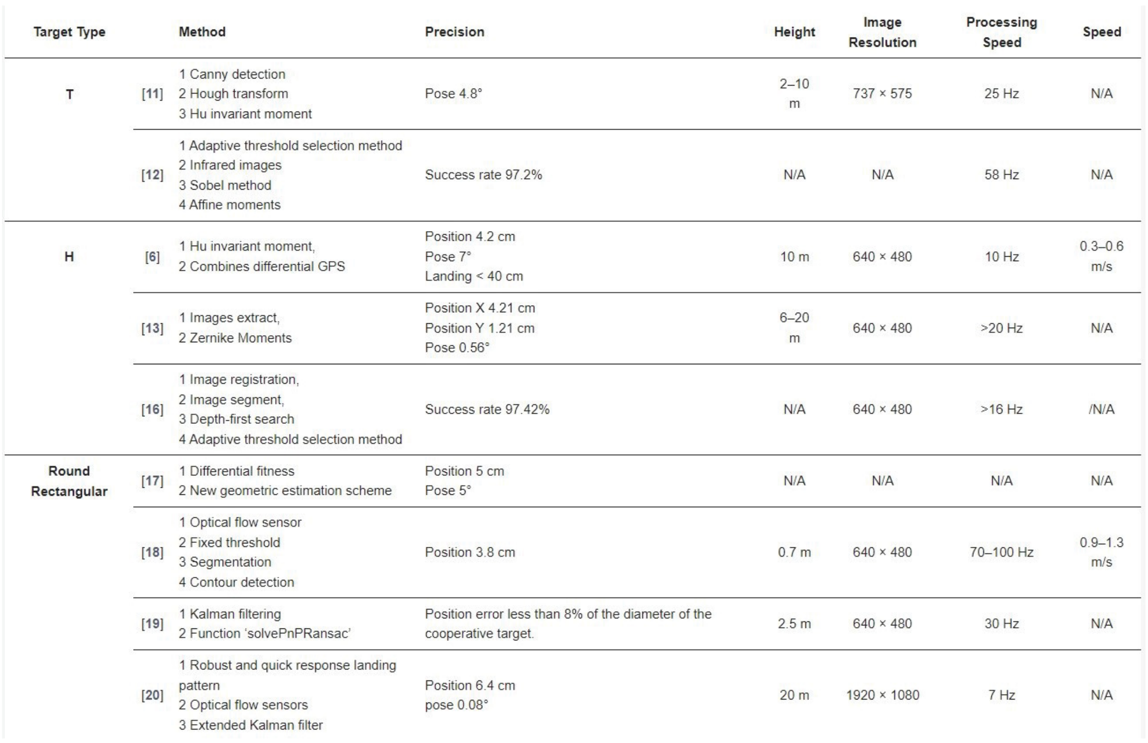
2.1 Background Knowledge

Autonomous landing for unmanned aerial vehicles (UAVs) requires the use of cooperative targets, such as artificial markers with specific geometric patterns, to guide the aircraft's position and attitude. However, in scenarios where cooperative targets are not feasible, such as post-disaster environment, UAV requires greater autonomy in data analysis and control in order to land successfully. Various feature-based methods that rely on artificial markers have been developed, but they face difficulties in accurately estimating the pose of the aircraft, especially when there are larger pitch and attitude changes or image noise [1].

Different algorithms of UAV landings with the use of artificially designed and natural landing markers have been discussed by Xin et alt. Cooperative target designs as shown from Figure 2.1 and References [11–20] are based on image processing algorithms. The design and associated algorithm use techniques such as geometric estimation and Kalman filtering for stable landing. Compared to other designs and algorithms, the algorithms using landing markers based on the circular and rectangular shapes have better computational efficiency and adaptability. Additionally, this design has allows for reliable detection even in challenging environments [1].

In our project of estimating the angle of the landing platform, the Kalman filter and the extended Kalman filter are recommended state estimation techniques. These filters are well-suited for linear systems and are computationally efficient, widely used in practical applications, and provide accurate minimum variance estimates. However, they require reasonable initial conditions and a known system model to work effectively. It's important to consider other factors like computation speed, accuracy, and robustness when selecting the appropriate technique for our specific requirements. It's also worth noting that while the unscented Kalman filter and particle filter may be better suited for highly nonlinear systems, they may not be necessary for our relatively simple and linear system. Ultimately, the selection of a state estimation technique should be based on the specific problem and system requirements, as well as the hardware and sensors available [21].

In our project, we use the webcam that we maintain on the platform's top to take images and determine an angle from them. We learned from this survey that they had two processor board-connected industrial machine vision cameras. IDS UI-1241LEC-HQ and a 12 mm lens are used in conjunction with the front camera,



Target Type		Method	Precision	Height	Image Resolution	Processing Speed	Speed
T	[11]	1 Canny detection 2 Hough transform 3 Hu invariant moment	Pose 4.8°	2–10 m	737 × 575	25 Hz	N/A
	[12]	1 Adaptive threshold selection method 2 Infrared images 3 Sobel method 4 Affine moments	Success rate 97.2%	N/A	N/A	58 Hz	N/A
H	[6]	1 Hu invariant moment, 2 Combines differential GPS	Position 4.2 cm Pose 7° Landing < 40 cm	10 m	640 × 480	10 Hz	0.3–0.6 m/s
	[13]	1 Images extract, 2 Zernike Moments	Position X 4.21 cm Position Y 1.21 cm Pose 0.56°	6–20 m	640 × 480	>20 Hz	N/A
	[16]	1 Image registration, 2 Image segment, 3 Depth-first search 4 Adaptive threshold selection method	Success rate 97.42%	N/A	640 × 480	>16 Hz	/N/A
Round Rectangular	[17]	1 Differential fitness 2 New geometric estimation scheme	Position 5 cm Pose 5°	N/A	N/A	N/A	N/A
	[18]	1 Optical flow sensor 2 Fixed threshold 3 Segmentation 4 Contour detection	Position 3.8 cm	0.7 m	640 × 480	70–100 Hz	0.9–1.3 m/s
	[19]	1 Kalman filtering 2 Function 'solvePnPRansac'	Position error less than 8% of the diameter of the cooperative target.	2.5 m	640 × 480	30 Hz	N/A
	[20]	1 Robust and quick response landing pattern 2 Optical flow sensors 3 Extended Kalman filter	Position 6.4 cm pose 0.08°	20 m	1920 × 1080	7 Hz	N/A

Figure 2.1: Comparison of different autonomous landing solutions in static scenes [1].

while IDS UI-1221LE-M-GL and an 8 mm lens are used in conjunction with the bottom camera. Industrial cameras feature a global shutter, it is a mechanism in the camera that captures the entire image simultaneously. It is particularly useful for applications involving fast-moving objects or dynamic scenes and it eliminates the shaking movement of an object and allows the accurate analysis and measurement of motion. Only consumer cameras with a rolling shutter were used in the survey; they are unsuitable for motion-sensitive image processing applications. The front camera is attached to a servo-controlled gimbal for better stabilization during image acquisition. We considered utilizing a webcam that is appropriate for our objective and keeping track of our progress using the webcam to estimate the platform angle [22].

In our project, we want to tilt a rectangular landing platform for drone landings using a robotic arm. The robotic arm is controlled by an Arduino microcontroller, which coordinates the leg motions and gathers sensor data. The robotic arm we used has three servo motors of the D645MW model and we want to use two of them: one for the pitch rotation of the platform and the other for its roll rotation. The research presented here illustrates how a hexapod robot's legs were controlled using servo motors and some other features. The movement of the robot will adjust the legs according to the land, even if it has unstable conditions like rough terrain. From an Arduino Uno Rev 3 that is mounted with an Arduino motor driver, we are providing an external power source. When compared to the robot employed in this study, the robotic arm is the simpler one, than controlling the robot on rough terrain. As a result, we are using servo motors which are attached to the arm to control the platform tilt [23].

Several papers have been researched based on the criteria of our project. Our main objective is to use the webcam and IMU to determine the platform's inclination angle and the arm to control tilt. However, we chose the Arduino Nano 33 IoT because it contains an integrated IMU device that has a 3-axis accelerometer and 3-axis gyroscope sensors. To estimate the platform tilt, the IMU is used, which provides the desired angle for our project. Additionally, the drone may have external inclinometers, laser range finders, and GPS modules added to it to increase its functionality. So, utilizing Arduino Nano 33 IoT on the platform is an efficient way to estimate tilt, even if many research papers have used IMU on the drone when it's landed to estimate angle. The Arduino Nano 33 IoT comes with an integrated IMU sensor that can offer precise information for determining the tilt angle of the platform, which is appropriate for our project's purposes [1].

2.2 Research on Existing Technologies

The authors of a research paper propose an image-based visual servoing approach to guide a VTOL UAV to safely land on a moving platform. This involves using computer vision techniques to estimate the UAV's position and orientation with respect to the platform, followed by a control algorithm to adjust its motion and guide it to a safe landing. The effectiveness of the approach is demonstrated through experiments. Although the approach requires visual markers on the landing pad, it provides valuable insights into the development of a control algorithm for UAV landing operations, which could improve safety and reliability [24].

Our project is exploring a different approach to landing UAVs on moving platforms by incorporating a tilting mechanism using a robotic arm to improve accuracy and stability during landing. The research paper we are referencing does not include this mechanism. While it is unclear whether the authors considered an adaptive controller, the sliding mode controller may have been sufficient for their experimental setup. Our project builds upon the ideas presented in the research paper where our own image processing algorithm uses the vector projection idea described in the paper [25].

The laser-based guidance is one of the earlier technology that enables the precise landing of a quadrotor UAV on an inclined surface, addressing challenges in sensing and control. A laser range finder measures the distance to the surface, and a control algorithm calculates thrust and orientation for landing on desired coordinates. The method has limitations, such as laser sensor sensitivity and the need for an accurate UAV dynamics model. This research paper suggests vision-based systems as potential solutions. However, this method assumes a flat and smooth surface and is sensitive to ambient light, which can cause inaccuracies in distance measurement. This laser-based technology does not discuss the use of a robotic arm to adjust the landing surface for inclined landings, which may be necessary for some other methods [26].

By taking into account all of the issues and difficulties in existing technologies, we are designing a prototype capable of providing more accurate readings by considering the required parameters. Here both the image processing through webcam and Arduino Nano 33 IoT will calculate the platform angle. We write code for the webcam and Arduino Nano 33 IoT both in MATLAB. The MATLAB gives the output for the IMU and webcam and we will also combine both angles, we are giving what is the platform tilt angle for the drone to be landed on the platform.

Chapter 3

Method

In this chapter, we give specifics regarding our problem statement, as well as the objectives we aim to achieve and our main contributions.

3.1 Problem Statement

Drones are widely used for a variety of purposes such as surveillance, delivery, search and rescue. However, accurate and secure drone landing remains a considerable barrier. We are working on this project to improve drone landing safety by utilizing a tilting platform mechanism. Our aim is to land the drone safely using the platform angle calculation through the webcam which is placed on the top with the help of the stand and the IMU placed on the platform. Existing methods, such as manual landing, landing pads, and GPS-based landing, are sometimes restricted in their capacity to offer accurate landing positions, particularly when external elements like wind, obstructions, and uneven terrain are present. Furthermore, these solutions might be expensive, and impractical. We can also use the drone's camera to determine the angle, however, when the weather conditions are taken into account, the drone's camera is insufficient.

The research aim of this project is to control a platform equipped with a robotic arm that can provide drones with landing spots at a specific tilt angle. We are using a webcam and Arduino Nano 33 IoT to calculate the angles of the platform and gives the angle for the drone to land. With methodology, this project has the potential to make an impact on the field of drone technology and address a real-world problem that is relevant to many industries.

3.2 Research Questions

1. How can the angle detected by using the IMU placed on the platform as well as the angle detected by using the image from the drone be combined to make a decision regarding the safety of drone landing?

2. What are the optimal range and viewing angle for the webcam to detect the angle of the drone during landing?

3. How accurate and reliable is the landing platform angle detection system using a webcam with image processing and Arduino Nano 33 IoT?

4. How can an image be designed and placed on the platform to accurately detect tilt using a webcam?

3.3 Objectives

The primary objective of this project is a platform that can provide a safe landing position for drones using a robotic arm. The platform will be in rectangular shape and a webcam will be placed above the platform with the help of the stand to calculate the angle. To provide feedback on the landing position for the drone, we will use Arduino Nano 33 IoT to calculate the angle. To power the robotic arm, we will give an external power supply from an Arduino motor driver which is mounted on Arduino Uno Rev 3. We will also use MATLAB to develop code for the webcam and Arduino Nano 33 IoT and generate a plot with information on the angle for drone landing and we will determine the determined angle with Arduino 33 IoT. In addition, we will analyze the platform's limits with the robotic arm and image processing capabilities.

3.4 Main Contribution

The main contributions of this project are summarized as follows:

1. Adoption of the cross as a landing marker.
2. Evaluation of the cross marker's suitability for platform tilt detection using image processing algorithms.
3. Introduction of two methods for estimating the tilt angle: one based on image processing algorithms and another based on IMU data.

Chapter 4

System Design and Implementation

This chapter describes the answer to the problem statement as well as how we developed the system and its hardware and software implementations.

4.1 Modeling

In this section, we will implement the design of our prototype, possible technologies, and algorithms. The development of a working prototype has been done through the following methods. By performing this, we aim to provide an improved understanding of our prototype.

4.1.1 Prototype of Our Project

In our project, we use two main components to detect the platform tilt. The webcam represents a drone's camera that is hovering over the center of the platform. We built a wooden stick holder for holding a webcam. The webcam is fixed on the wooden stick holder. From the top view, the webcam is used to detect the platform angle using image processing techniques and we are using a platform stand for mounting the arm on it for controlling the platform in a specified direction.

Figure 4.1 shows the prototype of our project. In our prototype, we use two main parts for platform movement which has wheels and another one is for holding the webcam stand but it does not have any wheels. The height of the platform prototype to the webcam is 70 cm. The height of the stand that holds the webcam at the center of the platform is 152 cm. The webcam holding bar length is 58.5 cm and the webcam holding position at the length of 38.5 cm to the holding bar. We are assuming the web camera plays the role of the hovered drone's camera, directed downward, and taking into account of the field view of the webcam, we decided to place it in a position, where the webcam is directed downward so that can view the platform clearly.

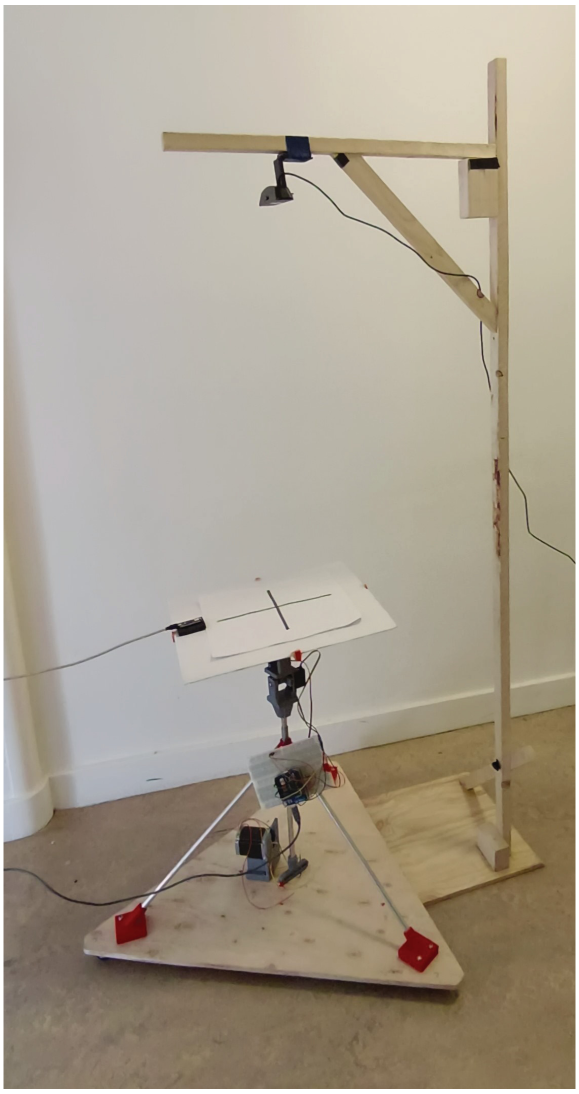


Figure 4.1: Prototype of our project.

4.1.2 Detection of Platform Tilt Using Arduino Nano 33 IoT

The Arduino Nano 33 IoT has a built-in IMU, that is known to be LSM6DS3. The LSM6DS3 combines the 3-axis accelerometer and 3-axis gyroscope angles that are used to detect the platform tilt angles in pitch (X-axis) and roll (Y-axis). The Arduino Nano 33 IoT is placed on top of the platform near the edge. When the platform is controlled using the arm in 2 directions accordingly then the Arduino Nano 33 IoT rotates along with the platform and gives angles of pitch and roll. MATLAB communicates with Arduino Nano 33 IoT using serial communication with the support of a USB cable port (COM). From the code, the Arduino Nano 33 IoT establishes the communication with in-built LSM6DS3 using the I2C (Inter-Integrated Circuit) protocol. Hence estimating the angles, we plot the graph of measured angles of pitch and roll for a time in MATLAB. Figure 4.2 represents the detection of platform tilt using Arduino Nano 33 IoT.

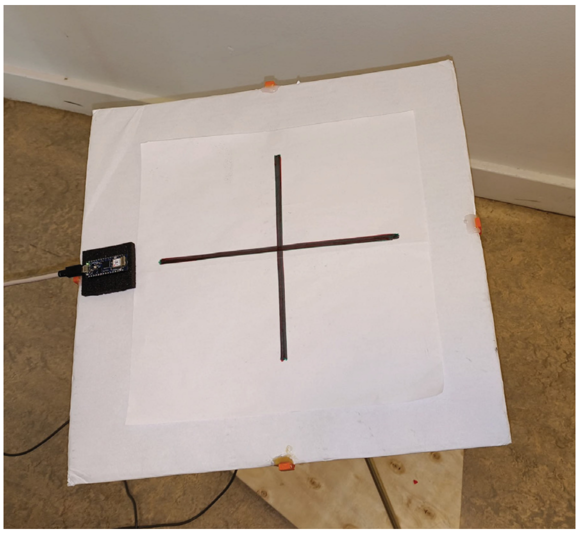


Figure 4.2: Detection of platform tilt using Arduino Nano 33 IoT.

The equations 4.5 and 4.6 were used in our MATLAB code for the calculation of angles in Arduino Nano 33 IoT. These two equations consist of estimating angles on the platform, signals coming from the accelerometer, gyroscope, and complementary angles.

4.1.3 Axis Alignment Techniques for Platform Rotation

These are the following methods used for the axis rotation on the platform are:

1. Implementation of the axis of rotation in two directions
2. Alignment of IMU axis in Arduino Nano 33 IoT
3. Alignment of the Arduino Nano 33 IoT axis on the platform

1. **Implementation of the axis of rotation in two directions**

   In our project, we are representing the directions of the x-axis as pitch and the y-axis as roll. Here,

   Pitch: It is the vertical rotation of objects from the front side view that is known to be the x-axis.

   Roll: It is the horizontal rotation of objects from the front side view and is known to be the y-axis.

   The diagrammatic representation of the three-dimensional axes of rotation is shown in Figure 4.3.

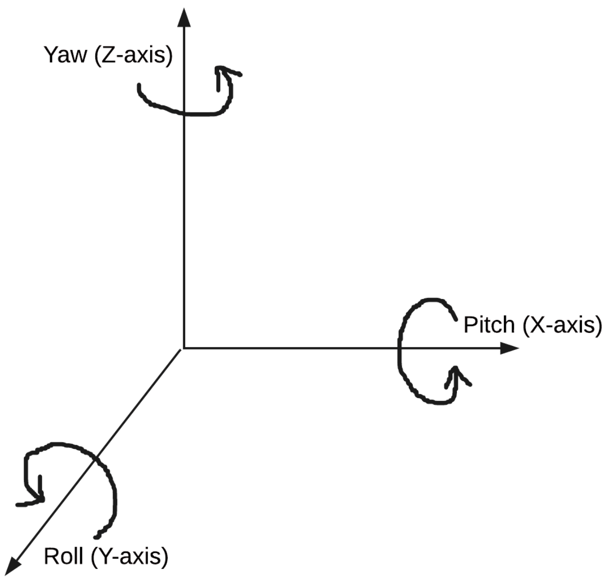


Figure 4.3: Three-dimensional axes of rotation.

2. **Alignment of IMU axis in Arduino Nano 33 IoT**

   The LSM6DS3 is the 6-axis of rotation that has a 3-axis accelerometer and 3-axis gyroscope sensors. The x-axis is pitch and the y-axis is roll. The alignment of the LSM6DS3 axis respective to the Arduino Nano 33 IoT is shown in Figure 4.4:

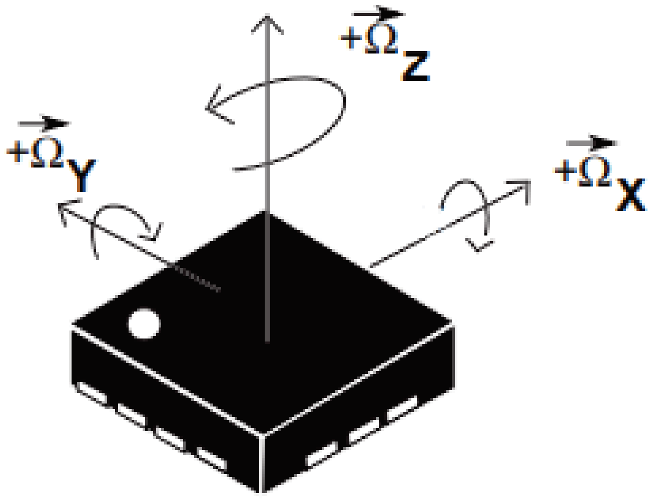


Figure 4.4: Axis of rotation in LSM6DSM with Arduino Nano 33 IoT [2].

3. **Alignment of the Arduino Nano 33 IoT axis on the platform**

   The Arduino Nano 33 IoT is mounted on the platform. The angles of the IMU of that axis in X and Y are rotated along the platform when the arm is turned in two directions along the X and Y axes. Figure 4.5 represents the rotation of axes in IMU and arm.

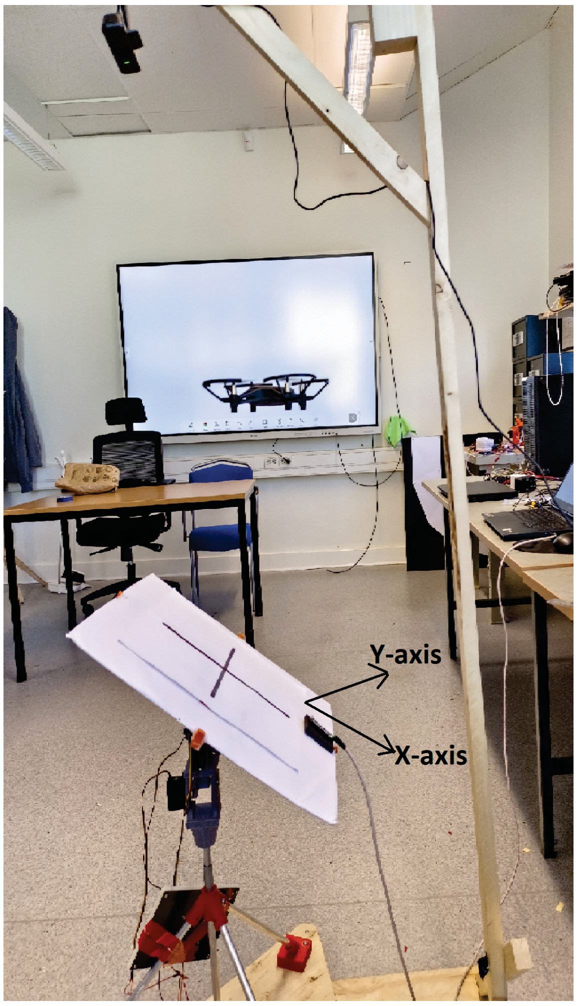


Figure 4.5: Alignment of Arduino Nano 33 IoT on the platform.

4.1.4 Performance of IMU

In this method, we are going to implement the angle orientation of the platform using IMU in the following ways that have known to be:

1. Estimation of the tilt angle using the signals from the accelerometer.
2. Estimation of the tilt angle using the signals from the gyroscope.
3. Complementary filter method.

1. **Estimation of the tilt angle using the signals from the accelerometer**

   The accelerometer we are using from the LSM6DS3 device is known as the 3-axis accelerometer sensor. It measures the acceleration along a mutually perpendicular axis, typically X, Y, and Z axis. Acceleration refers to any change in velocity including changes in speed or direction. In our project, we are measuring the angle in two axes known as pitch and roll. For measuring two axes, use the three-axis orientation to get the desired angles of an IMU. Equations 4.1 and 4.2 form by using gravity (g=9.8m/s2) and trigonometry to use accelerometer readings and to know the angle of the inclination. It represents the measurements of acceleration in three directions. The formulas used to calculate the acceleration in three directions are:

$$AccelRoll = \arctan\left(\frac{A_y}{\sqrt{{A_x}^2 + {A_z}^2}}\right) \tag{4.1}$$

and

$$AccelPitch = \arctan\left(\frac{A_x}{\sqrt{{A_y}^2 + {A_z}^2}}\right). \tag{4.2}$$

Here :

(a) Ax represents the acceleration along the direction of the X-axis which is a forward-backward motion.

(b) Ay represents the acceleration along the direction of the Y-axis which is side-to-side motion.

(c) Az represents the acceleration along the direction of the Z-axis which is left-right motion.

2. **Estimation of the tilt angle using the signals from the gyroscope**

   The gyroscope we are using from the LSM6DS3 device is known as a 3-axis gyroscope sensor. It measures the angular velocity or the rate of rotation around three perpendicular axes in a unit of time. By setting the elapsed time, can adjust the time interval between the samples. Equations 4.3 and 4.4 represent the measurement of a gyroscope in three directions.

$$GyroPitch = GyroPitch + Gx * elapsedtime \tag{4.3}$$

and

$$GyroRoll = GyroRoll + Gy * elapsedtime. \tag{4.4}$$

   Here Gy and Gx are angular velocities in X and Y directions.

3. **Complementary filter method**

   The complementary function is used to combine data using sensor fusion. In this method, equations 4.5 and 4.6 represent combined angles of signals from the accelerometer and gyroscope. Setting the high pass filter for the gyroscope and a low pass filter for the accelerometer ensure reduction of noise in signal waveform. The accelerometer gives a good indicator of orientation in static conditions and the gyroscope provides a good indicator of tilt in dynamic conditions. The formula for complementary function by combining the two filters using alpha and beta are:

$$Pitch = \alpha * GyroPitch + \beta * AccelPitch \tag{4.5}$$

and

$$Roll = \alpha * GyroRoll + \beta * AccelRoll. \tag{4.6}$$

The common value of alpha = 0.98 and beta=0.02. Hence adjusting the values of alpha and beta will make the filter respond quickly to changes in noise and reduce the delay in the pitch and roll angles.

4.1.5 Monitoring of Platform Arm

We placed an Arduino Nano 33 IoT on top of the platform. The components that are used for moving of the arm are D645MW digital high torque servo motors. These servo motors are powered up externally and controlled by Arduino Uno Rev 3 and Arduino Motor Shield Rev 3. Three pins of the servo motor are first linked to the Arduino Motor Shield Rev 3 placed on the Arduino Uno Rev 3. These servo motors are mounted onto the arm for control of the platform. Here, we are using two servo motors which are used to position the platform in an inclined way. The arm rotates the platform in two axes known as pitch and roll. Hence by controlling the arm, the platform is inclined to its axes. We detect the inclined platform using Arduino Nano 33 IoT and a webcam. Figure 4.6 shows the platform tilt using a robotic arm.

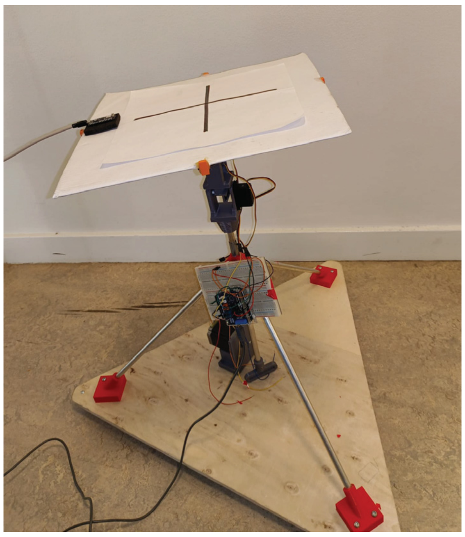


Figure 4.6: Platform tilt using robotic arm.

4.1.6 Image processing

Image processing is the field that focuses on using computer algorithms to analyze and manipulate digital images. In this case, image processing is being used to determine the platform's angle in one direction, either in roll or pitch. We will be capturing the image of the landing platform at instance to exhibit the image processing algorithm and this image is shown in Figure 4.10. The landing platform has a cross symbol on it as a landing marker. We are using this cross to detect the angle in the direction of roll and pitch by utilizing the block blob analysis. The block blob analysis will block one of the lines in the cross on the landing platform in such a way that we fetch lines from the Figure 4.10, this method will be discussed in more detail in the section of *Step-wise approach of image processing.* This image shown in Figure 4.10 undergoes a series of processing steps, starting with block blob analysis. The block blob analysis is a technique used to reduce the noise and unwanted object to be detected in the image and after this process, the image is converted into a gray-scale image. Next, we apply edge detection functions in MATLAB to generate a binary representation with noise reduction. Utilizing the Hough transform, we detect lines within the image and calculate the detected line length, enabling analysis of the detected lines. This methodology forms a crucial part of our thesis in assessing platform inclination angle. The image processing flow is shown in Figure 4.7.

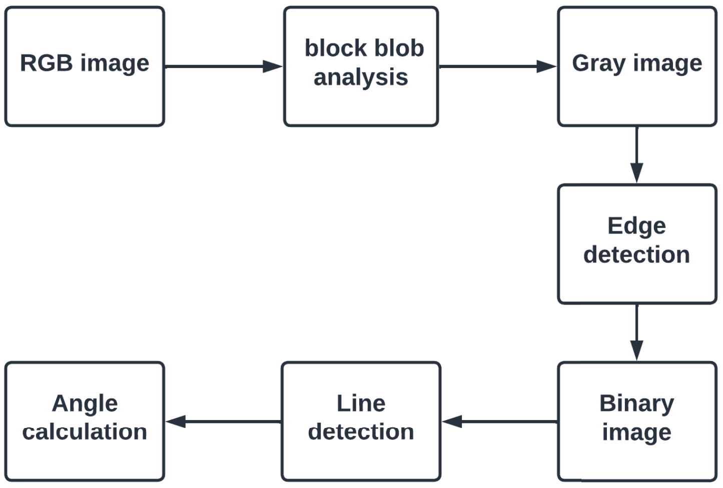


Figure 4.7: Block diagram represents the image processing flow.

For the image processing algorithm to determine the angle of the platform, perspective distortion is used. This distortion arises due to the reduction in line length caused by the perspective view. Our focus is on detecting the apparent line length on the platform, which we utilize for angle calculations in our analysis. Understanding and compensating for this distortion is crucial for accurate measurements and calculations in our research. Here is Figure 4.8 that represents this phenomenon in a geometrical way whereas the H.P and V.P represent the horizontal plane and vertical plane respectively [3].

As one can see in the placement of the webcam for our prototype, we are investigating a scenario in which the drone approaches the platform from above so that the platform's center and the drone's landing route coincide. The length of the line drawn on the platform becomes shorter from the top perspective if the platform is inclined in one direction. The ratio of the length observed to the actual length is calculated and fed into the inverse cosine function to determine the angle of the platform in radians. This angle is then converted to degrees (°) for the final output.

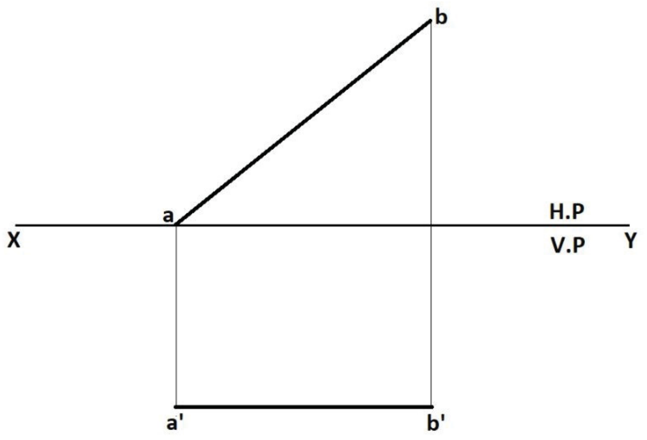


Figure 4.8: Geometrical representation of perspective distortion [3].

This angle calculation is shown in equation 4.7.

$$\Theta = \arccos\left(\frac{a'b'}{ab}\right) \cdot \frac{180}{\pi} degrees. \tag{4.7}$$

Here $a'b'$ and ab are the adjacent side and the hypotenuse of a triangle, that construct the angle Θ as shown in Figure 4.9.

Why *acos* trigonometric function for angle detection?

In our project, we utilize the cosine function as a fundamental component for angle detection. When viewing the platform from above, the length of a line drawn on an inclined platform appears shorter due to the viewing perspective. By leveraging the cosine function, we can relate the observed length of line in the top view to the original length of drawn line as shown in Figure 4.9 [27]. This is based on the property of the cosine function, which describes the ratio of an adjacent side to the hypotenuse in a right triangle. By calculating the ratio of observed length to the original length, we effectively obtain the cosine of the angle. We can then use the inverse cosine function to determine the angle in radians, which can be converted to degrees for ease of interpretation. Thus, the inverse cosine function plays a crucial role in our approach to accurately estimate the angle of the platform. Although, we will get only a positive angle irrespective of the negative and positive direction of the pitch and roll angle because the cosine trigonometric function will not produce the negative direction angles.

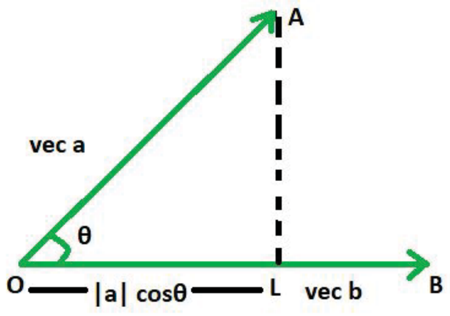


Figure 4.9: vector application of the cosine function.

Step-wise approach of image processing

1. **RGB image**

   The RGB image represents the original image captured by the webcam, preserving the full-color information. This image serves as a reference for comparing the processed versions and evaluating the effectiveness of the applied techniques. The RGB image can provide insights into color variations, texture, and overall visual appearance of the scene. Figure 4.10 represents the RGB image then it is applied with block blob analysis.

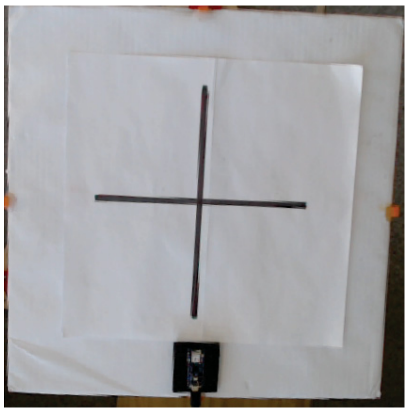


Figure 4.10: Image of the landing platform with the cross as a landing marker.

2. **Block blob image**

   The block blob image was generated by applying a block-based segmentation algorithm to the original image. This technique divided the image into smaller blocks and representative values to each block based on its content. The resulting image consists of blobs or regions with similar characteristics, allowing for a more localized analysis of the image. This segmentation can be useful for identifying distinct objects or patterns within the image. Figure 4.11 represents the block blob image. Here, we used two blocks, one for blocking the IMU in the image and another for the line that should not be allowed to be detected because it will produce the incorrect angles in the output and this block color also depends on the lighting conditions in the testing environment [28].

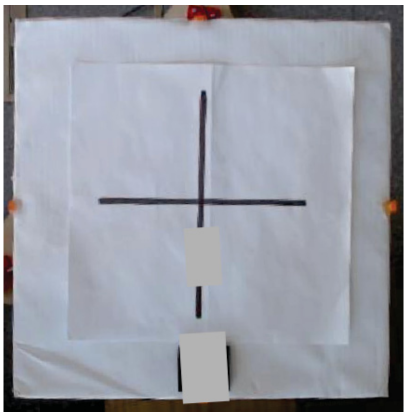


Figure 4.11: Image of a Block Blob.

3. **Grayscale image**

   The grayscale image was obtained by converting the original RGB image into a grayscale representation. Grayscale images only contain intensity information, removing color information from the image. This conversion simplifies the image and reduces computational complexity for certain types of analysis, such as edge detection or texture analysis. The grayscale image can reveal variations in brightness and contrast, making it suitable for specific image processing operations. This blocked image is preprocessed by converting it to grayscale, applying median filtering, and detecting edges. Figure 4.12 represents the grayscale image [29].

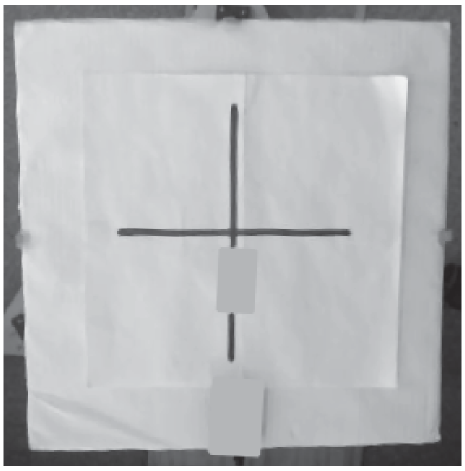


Figure 4.12: Grayscale version of the image in Figure 4.10.

4. **Binary image**

   The binary image was obtained by applying thresholding techniques to the original image and it is preprocessed with edge detection using the special in-built methods in MATLAB. Thresholding is a commonly used technique in image processing that converts a grayscale or color image into a binary image by using a threshold value. In our case, we selected a threshold value that separates the foreground objects from the background. We perform morphological operations to fill holes, remove border objects, and refine edge detection using functions named *imfill*, *imclearborder* [30] [31], dilation and erosion (*imdilate*, *imerode*) with specific structuring elements (*strel*) to refine the binary image [32] [33] [34]. The resulting binary image highlights the region of interest, which can be used for line detection. Figure 4.13 represents a binary image where there is a discontinued line that block blob analysis [35].

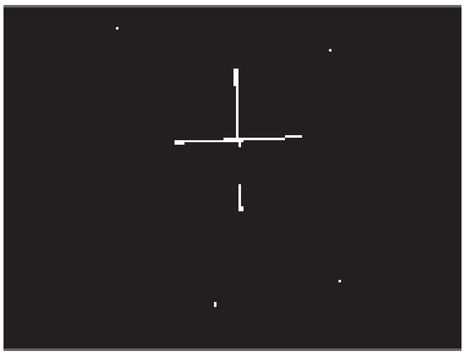


Figure 4.13: Binary image version of the image in Figure 4.12.

5. **Line detected image**

   There are so many methods to detect the lines. In our case, we apply the Hough transform to detect lines in the preprocessed image [36]. It identifies peaks in the Hough transform accumulator matrix. By joining the peaks, we are able to extract lines from the Hough transform result then we visualize the original image with the detected lines and their endpoints. The line-detected algorithm was used to extract linear structures or edges in the image. Figure 4.14 represents a line detected image. The lengths of the upper half and lower half of the line are determined and when it is inclined, we will determine the lengths again and we will calculate which one is smaller, then the direction of the angle assigned.

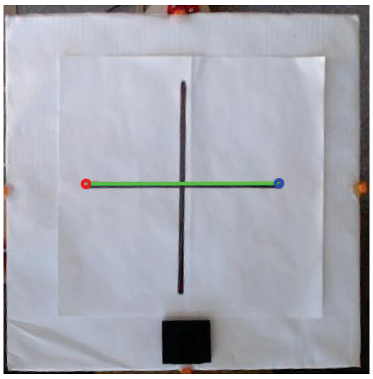


Figure 4.14: Image of line detected.

The block diagram shown in Figure 4.15 illustrates the integration of components using Arduino and a webcam. This block diagram depicts the working of each component to estimate the angle of the platform.

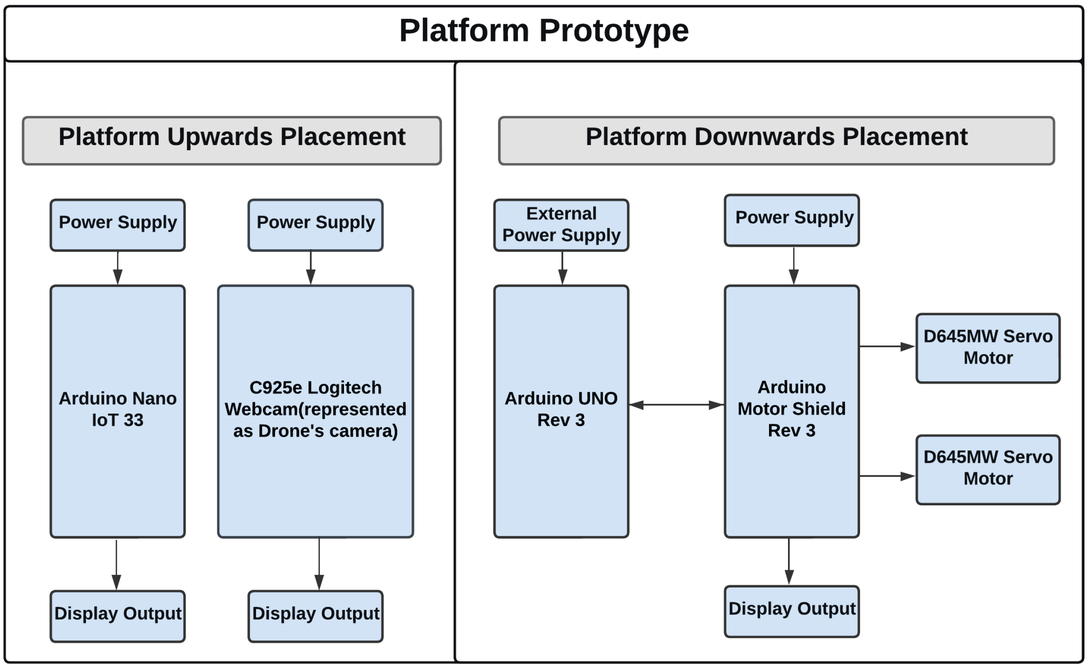


Figure 4.15: Block Diagram for whole prototype.

4.2 Implementation

To achieve the stated objectives, the implementation of the solution to the given problem consists of two parts :

1. Hardware Implementation
2. Software Implementation

4.2.1 Hardware Implementation

The required objectives and functionality can be achieved by connecting various hardware components. The hardware implementation requires the following components. Figure 4.16 represents two servo motors that are directly attached to an Arduino Motor Shield Rev 3 which is mounted on Arduino Uno Rev 3, and the IMU is directly connected to a computer. Two servo motors are used to control the arm; the top motor's shaft moves used for pitch direction, while the bottom motor moves used for roll direction. And to estimate the angle, we set the platform at 0 degrees horizontally, when the upper and lower controls are set at 3 degrees and 5 degrees, respectively. The Arduino Nano 33 IoT has built-in IMU known as LSM6DS3, which is used to measure the platform's angle and is connected to the computer through a USB port connection, and the webcam's USB port connection is also immediately connected to another computer for testing and validation.

1. Arduino Nano 33 IoT,
2. Arduino Motor Shield Rev3,
3. Arduino Uno WI-FI Rev 3,
4. Webcam,
5. Robotic arm,
6. Servo motor,
7. Connecting wires,
8. DC adapter cable.

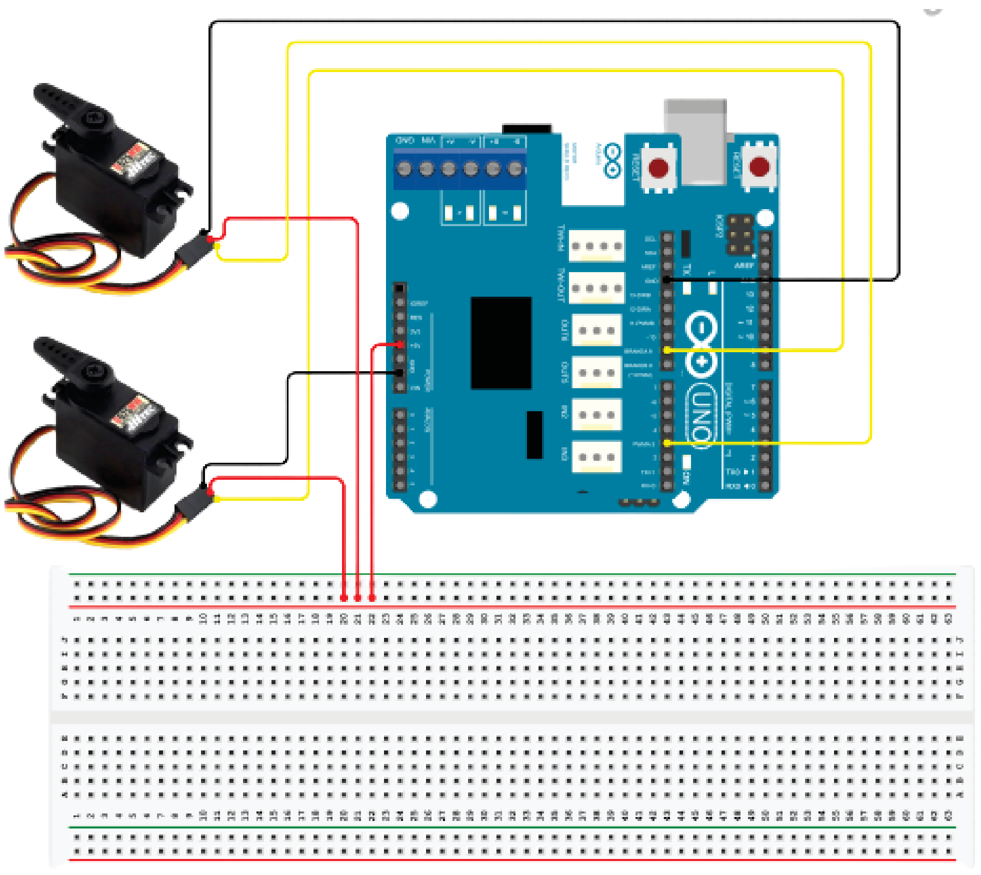


Figure 4.16: Circuit diagram of servo motors connected to Arduino Motor Shield Rev 3 which is mounted on Arduino Uno Rev 3.

4.2.1.1 Arduino Nano 33 IoT

The Arduino Nano 33 IoT is a suitable point of entry for enhancing current devices to become part of the IoT and construct pico-network applications. The main CPU on the board is a low-power Arm Cortex-M0 32-bit SAMD21. The WI-FI and Bluetooth communication is handled by a u-blox module, the NINA-W10, which is a low-power chipset that operates at 2.4 GHz. Furthermore, the Microchip's ECC608 crypto chip ensures secure communication. There is a 6-axis IMU, which makes this board ideal for simple vibration alarm systems, pedometers, the relative positioning of robots, and so on [4]. The Arduino nano specifications were given in Table 4.1

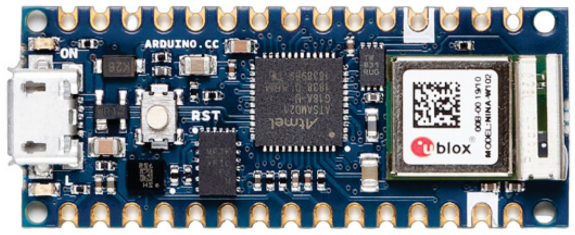


Figure 4.17: Image of Arduino Nano 33 IoT [4].

Table 4.1: Arduino Nano specifications [4].

Technical Specifications	Description
Microcontroller	SAMD21 Cortex-M0+ 32bit low power ARM MCU
Radio module	u-blox NINA-W102
Operating Voltage	3.3 V
Input Voltage (limit)	21 V
DC Current per I/O Pin	7 mA
Clock Speed	48 MHz
CPU Flash Memory	256 KB
SRAM	32 KB
Digital Input / Output Pins	14
PWM Pins	11 (2, 3, 5, 6, 9, 10, 11, 12, 16 / A2, 17 / A3, 19 / A5)
SPI	1
I2C	1
Analog Input Pins	8 (ADC 8/10/12 bit)
Analog Output Pins	1 (DAC 10 bit)
External Interrupts	All digital pins
IMU	LSM6DS3
Length	45 mm
Width	18 mm
Weight	5 g (with headers)

4.2.1.2 Arduino Motor Shield Rev3

The Arduino motor shield allows driving DC and stepper motors with Arduino. The Arduino motor shield is based on the L298, a dual full-bridge driver designed to drive inductive loads. It allows controlling the speed and direction of two DC motors separately using an Arduino board. Only an external power supply may be used to power the Arduino motor shield. External power can be supplied via an AC to DC adapter or a battery. It is recommend utilizing an external power source with a voltage between 7 and 12V to minimize any harm to the Arduino board on which the shield is installed. The motor shield PCB has a maximum length and width of 2.7 and 2.1 inches, respectively [5]. The pin functions were given in Table 4.2 and technical specifications given in Table 4.3

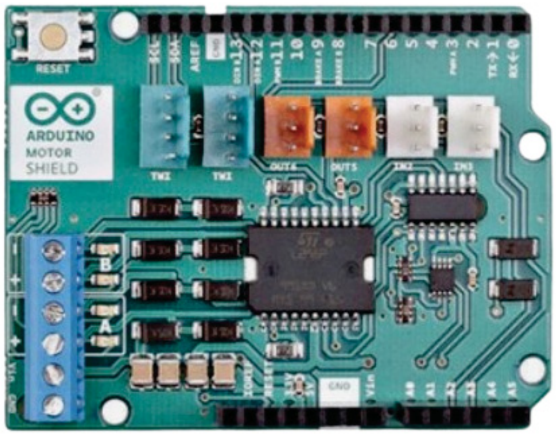


Figure 4.18: Arduino Motor Shield Rev3 [5].

Table 4.2: Arduino Motor Shield Rev3 pin functions [5].

Function	**pins per Ch. A**	**pins per Ch. B**
Direction	D12	D13
PWM	D3	D11
Brake	D9	D8
Current Sensing	A0	A1

Table 4.3: Technical specifications [5].

Technical specifications	**Deescription**
Operating Voltage	(5-12) V
Motor controller	L298P, Drives 2 DC motors
Max current	2A per channel or 4A max
Current sensing	1.65 V/A

4.2.1.3 Arduino Uno Rev 3

We are using Arduino Uno Rev 3, a microcontroller based on ATmega328P. This board includes 14 digital input and output pins, a USB port, a power connector, and an ICSP header. An external power supply ranging from 6 to 20V can power the board. The ATmega328 has 32 KB. There is additionally 2 KB of SRAM and 1 KB of EEPROM [6]. The technical specifications are given in Table 4.4.

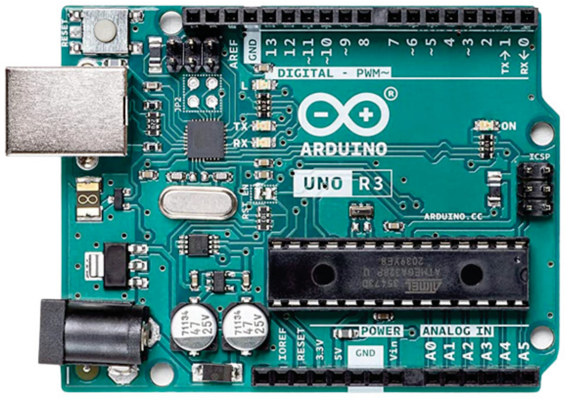


Figure 4.19: Image of Arduino Uno Rev3 [6].

Table 4.4: Arduino Uno Rev3 specifications [6].

Technical Specifications	**Description**
Microcontroller	ATmega328P
Operating Voltage	5 V
Input Voltage (recommended)	(7-12) V
Input Voltage (limit)	(6-20) V
Digital I/O Pins	14
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB
SRAM	2 KB
EEPROM	1 KB
Clock Speed	16 MHz
Length	68.6 mm
Width	53.4 mm
Weight	25 g

4.2.1.4 Logitech C925e (webcamera)

The C925e has a 78° diagonal field of vision, HD autofocus, and auto light correction, which automatically improves visual quality; two omnidirectional microphones catch sounds clearly up to one meter away [7]. Our project uses it to calculate the angle using image processing. The camera specifications are given in Table 4.5

Figure 4.20: Image of the webcamera [7].

Table 4.5: Webcamera specifications [7].

Technical Specifications	**Description**
Video	Supports 1080p (Full HD) @ 30 fps that makes a vivid impression every time with true-to-live video calls.
	78° diagonal fixed field of view (dFOV)
	1.2x digital zoom (Full HD) available
	Built-in HD autofocus ensures you're seen clearly throughout your video calls.
	Supports UVC H.264 encoding technology for a smoother video stream in applications.
Audio	Dual omni-directional mics, optimized to capture audio clearly from up to one meter away.
Connectivity	Connectivity Easily connects via USB-A; cable length of 6 ft (1.83m)
Piracy shutter	Privacy Shutter Open or close the built-in lens cover by sliding the tab on top of the webcam
Dimensions	Height x Width x Depth : (73*126*45) mm
Weight	170g

4.2.2 Robotic Arm

The robotic arm serves a crucial part in our project by tilting the landing platform to create a suitable landing position for the drone on the ship. By adjusting the angle of the platform, the robotic arm helps compensate for the ship's motion and provides a stable landing position. Each joint of the robotic arm is controlled by a servo motor. A total of 3 servo motors will be there but we are using 2 servo motors one is for yaw rotation and another motor is for pitch rotation of the robotic arm. The servo motor in the robotic arm is controlled using the Arduino Uno Rev 3 and Arduino Motor Shield Rev 3. The Arduino Uno Rev 3 receives commands and generates the necessary signals for the desired angles for the servo motor. The Arduino Motor Shield Rev 3 receives these commands and generates the necessary signals to control the speed and direction of the servo motors.

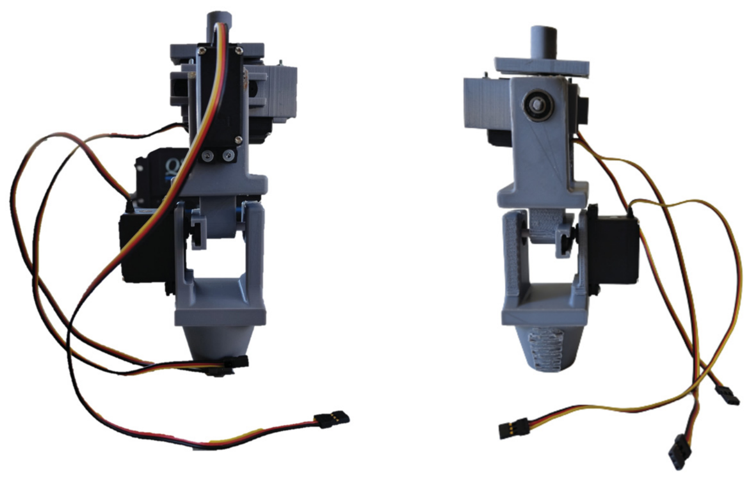


Figure 4.21: Image of robotic arm with two servo motors.

4.2.2.1 Servo Motor D645MW

The D645MW is a metal-geared, high-torque servo motor designed for radio-controlled automobiles, airplanes, and other applications. The quick, powerful motor runs on a wide 4.8 to 8.4 voltage range, making it compatible with any common battery chemistry without the need for adapters or regulators. It has a durable metal gearbox that delivers constant performance and precise control of the operation. Hence D645MW is high-performance, and long-lasting which is suitable for a wide range of applications [9]. The performance and servo specifications are given in Table 4.6 and Table 4.7.

Figure 4.22: Image of servo motor [8].

Table 4.6: Performance specifications [9].

Specifications	**Description**
Operating Voltage Range (Volts DC)	4.8V ∼6.0V ∼7.4V
Speed (Second @ 60°)	0.28 ∼0.20 ∼0.17
Maximum Torque Range oz. / in.	115 ∼158 ∼180
Maximum Torque Range kg. / cm.	8.3 ∼11.3 ∼12.9
Current Draw at Idle	30 mA
No Load Operating Current Draw	500 mA
Stall Current Draw	2,650 mA
Dead Band Width	2 µs

Table 4.7: Servo specifications [8].

Specifications	**Description**
Servo	Hitec D645MW
Manufacturer	Hitec
Applications	Heli/Airplanes 7.4 V (2S)
Type	Digital High Torque
Torque 6 V	158 oz/in (11.3 kg/cm)
Torque 7.4 V	180 oz/in (12.9 kg/cm)
Speed 6 V	0.20 sec/60 degrees
Speed 7.4 V	0.17 sec/60 degrees
Dimensions	(40.5 x 38 x 20) mm
Weight	60 g
Bearings	Dual Ball Bearing
Gear Type	Metal

4.2.3 Software Implementation

The implementation of the software requires to connect the various microcontrollers and webcam to get the desired angle. For the establishment of the communication between the microcontrollers, and the webcam the software we are using is MATLAB. MATLAB plays a significant role in our project, and here are some of the ways it helps us like image manipulation, algorithm creation, data visualization, and Arduino integration. There are two parts of the software implementation design patterns and the development tools utilized for the project for a required solution. Figure 4.23 shows the process in sequential order. In this flowchart, the main process is to estimate the IMU angle, we control the platform using the arm, and image processing techniques of the webcam used to detect the platform angle.

4.2.3.1 MATLAB programming of Arduino microcontrollers

After connecting all components, the power supply is given to the Arduino Nano 33 IoT, Arduino Motor Shield Rev 3 by computer. The MATLAB code is basically in C/C++, Fortan languages are written in MATLAB software which is a proprietary tool.

1. **For Arduino Nano 33 IoT**

   When the platform is inclined by controlling the arm using a servo motor, the Arduino Nano 33 IoT estimates the angles in pitch and roll at particular inclinations measurements of the platform.

2. **For Arduino Uno Rev 3**

   The servo motor is attached to the arm, which is used to control the platform movement in the directions of pitch and roll angles it was operated with an Arduino Motor Shield Rev 3. By controlling the arm here we estimate the inclined platform of its angle using Arduino Nano 33 IoT and webcam.

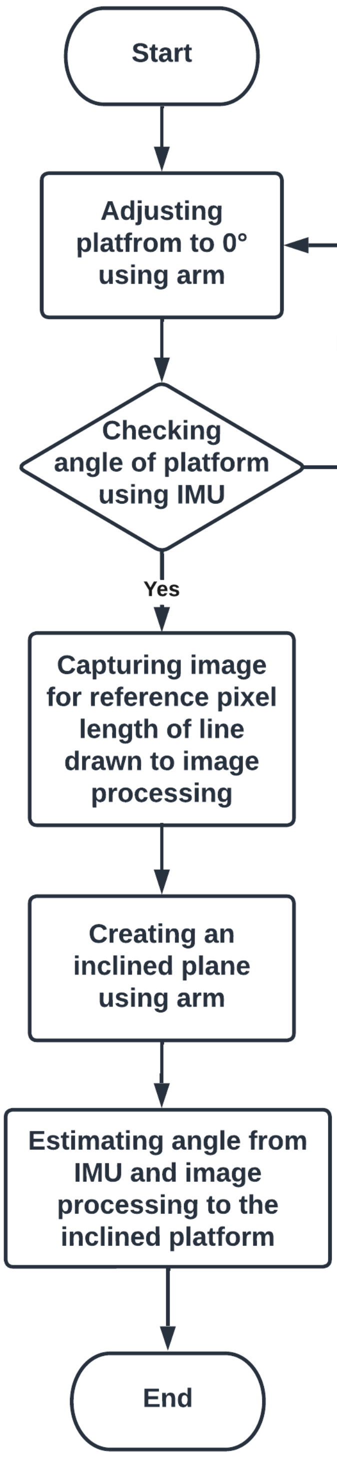

Start
Adjusting platfrom to 0° using arm
NO
Checking angle of platform using IMU
Yes
Capturing image for reference pixel length of line drawn to image processing
Creating an inclined plane using arm
Estimating angle from IMU and image processing to the inclined platform
End

Figure 4.23: Flowchart for the whole prototype.

4.2.3.2 MATLAB code for image processing

In a particular instance we are taking pictures of the platform from the top view of the prototype at that instance we are compiling the image processing code for that image to calculate the angle made by the platform in one direction either in pitch or in roll.

Chapter 5

Results and Analysis

5.1 Results

When the pitch and roll of the arm that is the angle made by the y-axis and x-axis respectively are tilted at ± 10° in either pitch or roll at a time, these are the results of the angles calculated from the IMU, reference device, and image processing. For the reference device, we used mobile phone of model "Nothing Phone 1" to compare angles estimated by the IMU and image processing because it is flexible to use and provides almost accurate output in most cases [37]. We used an application to determine orientation that is called "My device", which has given access to many sensors in our mobile. When the robotic arm is set to 90° then the platform looks parallel to the ground from the top view. For getting results with image processing from the top view we have set the platform with 80° which is -10° and 100° is +10° in image processing. In these cases, we primarily focused on adjusting the platform in the pitch direction.

For +10° made in pitch by the arm the result is shown in Table 5.1.

Table 5.1: Comparison of angles at +10° in pitch.

Platform setup	**IMU**	**Mobile**	**Image Processing**
Roll	0.41°	0.72°	0°
Pitch	13.41°	13.02°	13.04°

For -10° made in pitch by the arm the result is shown in Table 5.2.

Table 5.2: Comparison of angles at -10° in pitch.

Platform setup	**IMU**	**Mobile**	**Image Processing**
Roll	-2.41°	1.79°	0°
Pitch	-15.92°	-16.15°	16.17°

Similarly, angles are made in roll by the arm of the platform. In these cases, we moved the platform in the roll and negligible change in pitch.

For +10° made in roll by the arm the result is shown in Table 5.3. These are the results of the angles calculated from the IMU, mobile, and image processing that are shown in Table 5.3 5.4.

Table 5.3: Comparison of angles at +10° in the roll.

Platform setup	**IMU**	**Mobile**	**Image Processing**
Roll	17.9°	17.5°	18.1°
Pitch	-0.9°	-1.0°	0°

For -10° made in roll by the arm the result is shown in Table 5.4.

Table 5.4: Comparison of angles at -10° in the roll.

Platform setup	**IMU**	**Mobile**	**Image Processing**
Roll	-11.7°	-10.1°	12.5°
Pitch	-0.4°	-0.8°	0°

Table 5.5: Output angles of an IMU.

IMU output angles		
Arm angle condition	**Pitch**	**Roll**
When pitch is in +10°	+15.00°	-2.38°
When pitch is in -10°	-13.59°	0.43°
When roll is in +10°	-0.65°	17.16°
When roll is in -10°	0.84°	-11.63°

Table 5.5 represent the output angles of the IMU respective to the control of the arm in pitch and roll conditions.

Here we take the mobile angle as the reference angle, Table 5.6 represents the Reference angles used to estimate the angle with respect to the control of the arm in pitch and roll conditions.

Table 5.6: Reference output angles.

Reference output angles		
Arm angle condition	**Pitch**	**Roll**
When pitch is in +10°	13.36°	0.48°
When pitch is in -10°	-16.89°	-1.37°
When roll is in +10°	-2.81°	17.53°
When roll is in -10°	-4.93°	-10.43°

5.2 Analysis

In this section we present the analysis of arm performance and comparison of angles at 0° and analysis of image processing techniques applied to various image captures using a webcam.

5.2.1 Analysis of Image Processing

Considering the camera angle, it is essential to note that the orientation of the webcam can introduce variations. To ensure accurate angle detection through image processing, it is crucial that the plane of the image sensor in capturing the platform is parallel to the plane of the platform itself. Any misalignment between these planes may result in uncertainties in the angle measurements obtained. We conducted experiments to detect angles using image processing when the axis of the webcam capture that is the lens of the webcam and the centroid of the platform are aligned. In this process, a weight is attached to a thread and suspended from the lens of the webcam. The weight is adjusted until it reaches a state of equilibrium. Once the weight is stable, the centroid of the platform is aligned with the direction indicated by the weight. It is crucial to ensure that these alignments are accurate in order to obtain precise results for angle detection in image processing.

5.2.2 Performance of an Arm in Our Project

The robotic arm's ability to precisely maneuver and perform tasks with accuracy. The arm can move the platform in two directions. The platform tilt was positioned by the arm equipped with two servo motors. The primary key performance of the arm is load capacity, it can be capable of bearing the amount of platform's weight by giving the external power to the motor. By examining Figure 5.1, we can observe the arm's movement and evaluate its overall load and responsiveness.

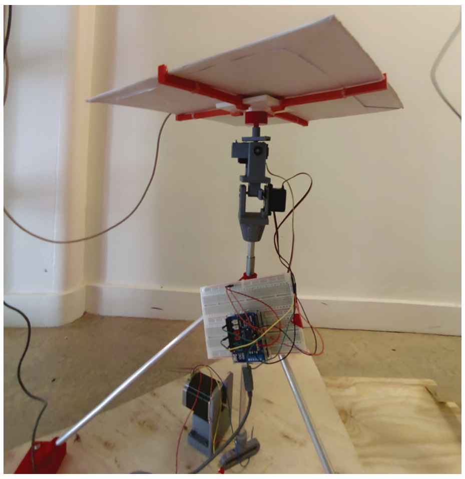


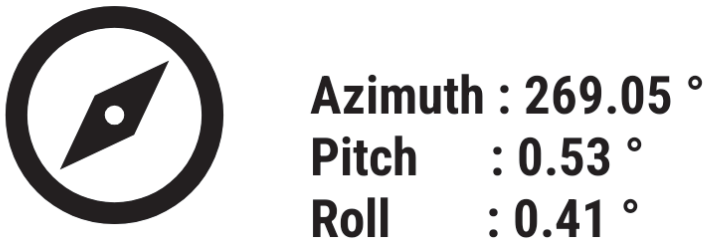
Figure 5.1: Performance of arm.

The output of an IMU is when the arm maintains the platform at 0° in both directions. Here is Table 5.7 output of an IMU.

Table 5.7: Output of IMU at 0° in pitch and roll

IMU output angle	
Roll angle	0.72561°
Pitch angle	-0.77129°

The reference output angle shows when the arm maintains the platform at 0° in both directions. Here is the output of the mobile showing this Figure 5.2.

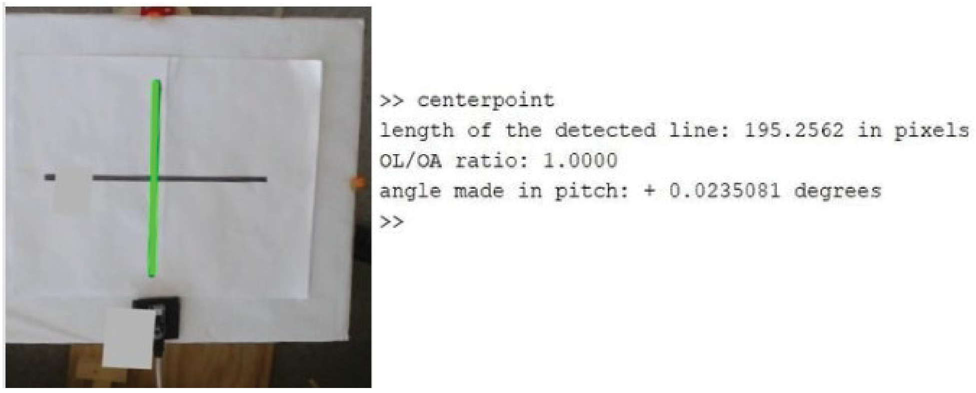


```
Azimuth : 269.05 °
Pitch     : 0.53 °
Roll      : 0.41 °
```

Figure 5.2: Angle of the pitch and roll when the angle of the arm is 0°.

When the IMU output is 0° in roll and pitch which is set for the formation of platform angle in an accurate way, we are using block blob analysis and were able to detect only one line and calculate the length of the individual line in one direction. Similarly, the length of the other line is also calculated using the blob technique to mask out the artifact. We are considering the length as a reference to the original length from the top view for the calculation of the angle.

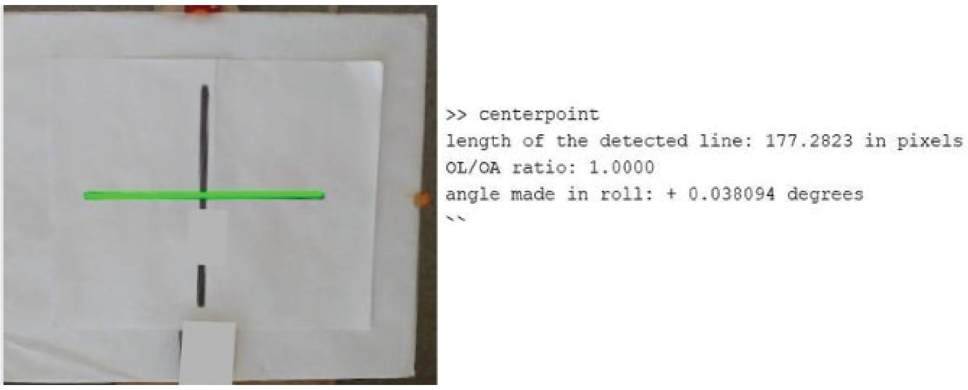
Figure 5.3 of image processing where the pitch angle made by the platform is 0° from the top view by using block blob analysis. We avoid the detection of another line in one direction for determining the angle in specified directions.



```
>> centerpoint
length of the detected line: 195.2562 in pixels
OL/OA ratio: 1.0000
angle made in pitch: + 0.0235081 degrees
>>
```

Figure 5.3: Visualizing line detection and pitch angle.

Figure 5.4 shows the output of image processing where the roll angle made by the platform is 0° from the top view by using block blob analysis. We avoid the detection of another line for determining the angle in another direction.



```
>> centerpoint
length of the detected line: 177.2823 in pixels
OL/OA ratio: 1.0000
angle made in roll: + 0.038094 degrees
``
```

Figure 5.4: Visualizing line detection and roll angle.

When the arm is holding the platform at 0° from the top view. Table 5.8 gives details about the comparison of angles between IMU, mobile, and image processing.

Table 5.8: Comparison of angles at 0° in the Pitch and roll.

Robotic Arm Conditions	**IMU**	**Mobile**	**Image Processing**
Roll	0.72°	0.41°	0°
Pitch	-0.77°	0.53°	0°

At which angle does the drone start sliding on the platform?

In our project we employed the RYZE Tello EDU drone, a versatile, and educational unmanned aerial vehicle, to conduct our research and experimentation. We conducted the test for drone landing with various angles of the platform with the help of the arm. We checked where the drone started sliding when it is placed on the platform and changed the inclination of the platform using the arm. The result of this test is ±32° with respect to the ground in both roll and pitch. For this kind of drone, the legs of the drone are made up of rubber. So, drone sliding started at the angle that is 32°.

Here we capture an image of an aerial perspective of the drone as it navigates through the air as shown in Figure 5.5. The image showcases the drone's flight path directly toward the platform. This angle corresponds to the moment when the drone reached its peak altitude while maintaining a straight trajectory toward the landing platform.

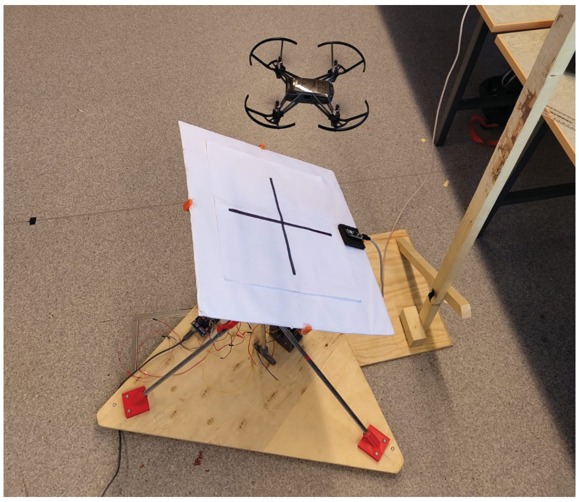


Figure 5.5: Drone approaching the platform.

Here we capture an image of the successful landing of the drone on the designed platform as shown in Figure 5.6. From our analysis, we determined that the minimum angle achieved during the landing process was 32° with respect to the drone. This angle signifies the moment when the drone gradually approached the landing platform, ensuring a safe and controlled landing.

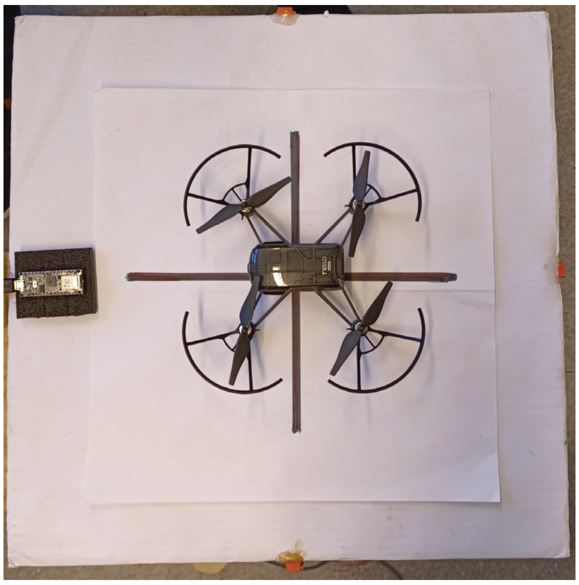


Figure 5.6: Drone landing on the platform.

The results indicate that the drone exhibited a range of angles during the landing process, within a minimum angle of 32° respective to the drone. The analysis of these angles provides valuable insights into the drone's maneuverability and its ability to approach the landing platform at varying degrees of inclination. Moreover, when the drone gets close to the platform for a safe landing, take into consideration the average samples of IMU final output angles.

Table 5.9 represents the relative error estimation of angle using IMU, Image processing, and reference angle with respect to the platform.

Table 5.9: Relative error percentage of the platform.

Position of arm		**Relative error**	**Reference angle**		**IMU angle**		**Image processing angle**	
			Roll	**Pitch**	**Roll**	**Pitch**	**Roll**	**Pitch**
Set pitch angle	-10°	67 %	1.79°	-16.15°	-2.4°	-15.9°	0°	16.7°
	+10°	34%	0.7°	13.02°	0.4°	-13.4°	0°	13.084°
Set Roll angle	-10°	25%	-10°	-4.08°	-11.7°	-0.4°	12.5°	0°
	+10°	81%	17.5°	-1.0°	+17.9°	-0.9°	18.3°	0°
Set tilt	0°	5.3%	0.41°	0.53°	0.17°	0.48°	0°	0°

Chapter 6

Discussion

In this chapter, we analyze and interpret the findings of this study and their relation to the research question and objectives. This chapter examines the significance of the results, compare them with previous studies and discuss their implications.

1. **Introduction to the discussion chapter**

   The discussion chapter aims to analyze and interpret the findings of the project. This chapter explores the significance of the results, compare them with relevant literature, and discusses the applications and limitations of the developed platform for drone landing. The primary goal of this chapter is to analyze and restate our research questions and hypothesis to provide a clear focus for the discussion. Furthermore, we critically examine our project findings. This allows us to identify gaps additionally, we address the limitations encouraged during the course of our project.

2. **Research Questions / Hypotheses**

   Research Question 1

   How can the angle detected by using the IMU as well as the angle detected by using a webcam be combined to make a decision regarding the safety of drone landing?

   Answer: Combining the angle detected by the IMU and the angle detected using the image from the drone can provide a comprehensive assessment of the safety of drone landing. To ensure a safe landing of the drone on the platform, the average of the final output samples from both the IMU and webcam angles is calculated, with the condition that it falls within a permissible range of 32 degrees.

   Additionally, when considering the webcam and the weights of the Arduino Nano 33 IoT angles in the final output samples. The weights refers to the output values of IMU angles that is higher or lower of it's acceptable range. If the weights are higher, the average output samples with greater weight will be disregarded, and only the remaining sample weights will be considered. Furthermore, we provided possible case suggestions in this discussion chapter.

Research Question 2

What are the optimal range and viewing angles for the webcam to detect the angle of the drone during landing?

Answer: The optimal range and viewing angle for the webcam to detect the angle of the drone during landing depend on several factors, including the specific camera specifications, the size of the landing platform, and the desired level of accuracy in angle detection. However here are the general considerations:

Range: The distance between the webcam and the landing platform. It should be determined based on the field of view and resolution capabilities of the camera. The webcam should be positioned at a distance that allows for clear and detailed imaging of the landing platform. Image processing techniques that are used to identify the platform tilt can be used when the distance between the drone's camera and the platform allows to acquire an complete image of the landing marker on the platform. Once it reaches a certain point it might be able to perfom the line recognition and estimation of the platform's angle.

Field of view: The field of view of the webcam determines the area that can be captured by the camera. A wider field of view allows for a larger coverage area.

Lighting conditions: Lighting conditions plays a crucial role in the performance of the webcam. Sufficient lighting is essential to ensure clear and well-defined images of the landing platform. Make sure that the lighting conditions will not produce any shadows on the platform.

Research Question 3

How accurate and reliable is the drone landing angle detection system using a webcam through image processing and Arduino Nano 33 IoT?

Answer:

Several elements will influence the accuracy and reliability of the drone landing angle detection system employing an image processing camera and an Arduino Nano 33 IoT. Here are some factors that might impact system performance.

Image processing algorithms:

The accuracy of the angle detection system is greatly dependent on the image processing methods utilized.

Webcam quality:

The quality and resolution of the webcam utilized for picture capture are important factors in angle detection accuracy. In our project, we are employing a camera with a 1080p HD resolution and it has autofocusing features which enable us to capture clear images.

Calibration process:

The system's calibration is critical for accurate angle detection.

Calibration should take lens distortions, camera position, and platform orientation into consideration. Throughout our project, we have fixed the webcam and IMU positions. We took an angle by remaining in a stationary position. We compare the two angles after obtaining both.

Arduino Nano 33 IoT:

The dynamic range measurement of acceleration, gyroscope angular velocities, and noise from the signal are used in the calibration of the Arduino Nano 33 IoT to estimate platform tilt.

Environmental conditions:

External factors like lighting conditions, background clutter, and drone movement can all have an impact on the system's dependability. A regulated atmosphere and enough illumination can increase system performance.

Research Question 4

How can an image be designed and placed on the platform to accurately detect tilt using a webcam?

Answer:

Using a webcam here are the possibilities that we have taken into our consideration.

1. Contrast patterns:

Design an image with clear and distinct contrast or patterns that can be easily identified by image processing. High contrast between different regions or identifiable patterns can aid in accurate tilt detection.

2. Orientation markers:

We have used the cross marker which is placed on the platform, so the webcam can capture clearly include orientation markers or reference points in the image that indicates the desired alignment or orientation of the platform. These markers can provide a reference for angle calculation and help to determine the tilt accurately.

3. Image size and resolution:

The size and resolution of the image relative to the field of view of the webcam. Ensure that the image is large enough and has sufficient detail to allow precise angle detection. Higher-resolution images can provide more accurate measurements but may increase computational requirements.

4. Image placement alignment:

We have put the webcam on the central location on the platform where the image can be captured by the webcam which is discussed in the previous Chapter 5. The center of the cross which was used as a landing marker was aligned with the center of the platform and the center of the webcam. The placement should take into account the camera field of view and the desired range of tilt angle.

5. Image processing techniques:

   Appropriate image processing techniques, such as edge detection, feature extraction, or template matching to analyze the captured image. These techniques can identify the markers or patterns in the image and extract the necessary information for angle detection.

3. **Project overview**

   The project involves the development of a rectangular platform for drone landing. The platform incorporates a robotic arm at the bottom to tilt the platform at a specific angle. An Arduino Uno Rev 3 and an Arduino motor driver are used to provide the power supply and control of the robotic arm. Additionally, an Arduino Nano 33 Iot is placed on the platform to estimate the angle of the platform through MATLAB. Using image processing through the webcam, we could calculate the angle through image processing. The code for the webcam and Arduino Nano 33 IoT have been written in MATLAB, that provides a final output and allows to make a decision regarding the drone landing.

4. **Comparison with existing literature**

   Our project has a different objective compared to Jacqueline M. Moore's research on autonomous helicopter landings on sloped terrain. Our project differs from Jacqueline M. Moore's research in that our focus is solely on integrating IMU and image processing components. While Moore's research covers sensors, control algorithms, and feedback mechanisms for autonomous helicopter landings, our research is centered around a smaller-scale platform and places greater emphasis on detecting tilt as a crucial aspect of our analysis. Although the two projects have the same goal of dealing with landings on irregular ground, their methods, systems, and uses are significantly distinct.

   On Comparison with the "A Vision-Based Onboard Approach for Landing and Position Control of an Autonomous Multirotor UAV in GPS-Denied Environments," referenced research, our project involves less complex vision-based techniques for guidance during flight and landing. This involves leveraging visual information from onboard cameras or image sensors, rather than a more sophisticated drone with multiple rotors. Additionally, our project does not specify the type of UAV or drone being utilized, indicating a potentially simpler setup. By acknowledging these differences, it is evident that the scope and complexity of our project are smaller than that of the referenced research.

   The research paper "Control of Quadrotors for Tracking and Landing on a Mobile Platform" discusses methods to track and land on a moving platform using sliding mode and adaptive controllers, while considering the ground effect. The paper describes calculating the pitch, roll, and yaw angles to track the mobile platform, and using an adaptive sliding mode controller to safely land on an inclined platform. In our case, we use a controlled platform attached to an arm with two servo motors to adjust the pitch and roll angles. We are measuring the angles using an Arduino Nano 33 IoT with an inbuilt IMU and processing measurement with complementary filters. Our approach improves the drone landing methods in terms of controlling the platform in unstable conditions and saves time with image processing techniques for tracking.

According to "Laser-based Guidance of a Quadrotor UAV for Precise Landing on an Inclined Surface" the research effort focuses on the precise landing measurement and control of UAVs on flat and inclined surfaces. Compared with our project, the paper is mainly based on the landing of quadrotor drones. A quadrotor UAV performs a flyover and uses lasers and a CMOS camera to detect the projections of the lasers on the ground plane. For this, they determine the altitude of landing and attitude tracking of the quadrotor that matches the slope of the landing platform. And considering the laser modules the altitude would be determined and using a CMOS camera the bright points reflected on the landing surface are detected. In comparison to our project, the paper is mainly based on the landing of a quadrotor UAV is unknown, it may be the inclined surface or flat surface and control of the UAV in the form of the laser modules and CMOS camera. So in our scenario, we consider the platform to be controlled by an arm with two D645MW servo motors. The arm is used to control the platform in pitch and roll directions for safe landing of the drone. Finally, we aim to control the platform in a specified direction to ensure that the landing drone is safely on the platform.

5. **Possible cases for safety drone landing on the platform in real-life applications**

   Case-1: When the drone is landing on an inclined condition, here we consider the case when the data obtained from both the camera in the drone and the installed IMU on the platform are similar. In this case, we are making decision regarding the drone landing based on estimation of the angles of the platform using IMU on the platform and using a drone camera. Here we take one data sample (representing the angle) from IMU, and one data sample (the angle) from image processing technique and find the average of these two samples in order to get the estimated angle of the platform tilt and make a decision to land on the platform. The problem with this approach is if there is a significant outlier sample then the decision will be affected by this outlier. Hence excluding the outliner sample, the remaining samples are considered to be a combination of both drone cameras and IMU angles that can specify the appropriate angle for the safe landing of the drone on the platform. Next case discuss the technique where the tilt is estimated using several samples instead of a single sample.

   Case-2: If the samples of both drone and IMU have some outlier samples, consider averages of the samples from IMU and image processing. Here we take the average of several samples from IMU and another average from several samples from image processing and combining those two averages we get a final output sample. The problem here could be that if the two averages are very different from the acceptable range of landing or from each other, then the data either from the image processing algorithm of from IMU can be considered as unreliable. Hence, the decision regarding the landing is difficult to make and the drone might need to search for another platform. If the two averages are similar then we get the desired angle for safe landing of the drone on the platform.

Case-3: If the IMU or the drone's camera gets damaged due to environmental circumstances and functioning of the remaining one is in good condition. In this case, to reduce the uncertainty in the data and to increase the reliability of the system, we are adding one extra IMU-2 used to safely land the drone on the platform. If the IMU-1 gets damaged or not working properly, here we consider the estimation of angle using another IMU-2 and the drone's camera and taking the average of it's both samples to get an appropriate angle, that makes to ensure that the drone landed on the platform safely. Similarly, if the drone's camera is damaged, we can use both IMU-1 and IMU-2 to estimate the platform's angle. Again, taking the average of the angle measurements from both IMUs allows us to ensure a secure landing on the platform. This will also be used to verify if the user acknowledges one of them is damaged by comparing the outputs with IMU-2 outputs generated.

6. **Discussion of major findings**

   Tilted landing platform

   The results demonstrate the successful integration of the robotic arm and Arduino components to achieve a tilted platform. The platform's adjustable angle allows for control of the drone's landing position, enhancing landing accuracy and stability. The findings suggest that a tilting landing platform can provide a flexible and adaptable solution for drone operations in various environments.

   Image processing and angle calculation

   The implementation of image processing techniques using a webcam, the Arduino Nano 33 IoT estimates the platform angle. The MATLAB code analyzes the captured images, calculates the angle, and provides the necessary information for the drone to land at the optimal position. This finding highlights the potential of integrating image processing technologies in drone landing systems.

   Why we are using the cross to detect the angle?

   To calculate the angle, we are using the apparent length of the line detected and its original length of it instead of a single line, we are using the cross to determine the angle in both directions such that there will be no need to rotate the platform whenever to determine the angle in another direction.

7. **Interpretation and analysis**

   The successful development of the tilted landing platform and the utilization of image processing techniques offer several advantages. The adjustable angle capability allows for customized landing based on specific requirements or environmental conditions. The angle calculation enhances the precision and efficiency of drone landing, minimizing the risk of collisions and ensuring safe operations.

8. **Addressing limitations**

   This project has certain limitations that should be acknowledged. Firstly the accuracy and reliability of the image processing technique may be influenced by lighting conditions, shadows, image quality, the drone's altitude and other factors.

   Another constraint is that we utilized a webcam that stayed at a constant altitude, whereas a real drone does not maintain an exact height over the platform. Consequently, the distance between the platform and the drone will fluctuate.

   Another limitation is the image sensor is parallel to the platform. However, in real-life scenarios, this alignment is often not feasible. Our proposed solution necessitates aligning the image sensor in the drone camera with the center of the cross in the image to detect the angle of the landing platform. However, with an actual drone, a misalignment is anticipated and must be compensated for.

Chapter 7

Conclusions and Future Work

Conclusion

Finally, employing a tilting platform and image processing algorithms, this study successfully constructed a drone landing system. The system exhibits the determination of the angle of the platform from IMU and image processing. The platform can be tilted using robotic arm that is implemented in MATLAB, the image processing algorithms of the top-view captured images proved to be effective in attaining platform angle estimates and aiding successful drone landings.

The project aimed to improve landing systems by utilizing cooperative platform data, resulting in enhanced adaptability, control, and safety of drone operations, particularly in inclined terrains.

Future work

Further improvements could focus on optimizing the image processing algorithms and considering alternative sensors or technologies for angle calculation. Additionally, the scalability and compatibility of the developed platform with different types of drones should be further explored. The developed tilted landing platform has various potential applications in drone delivery systems, aerial surveillance, and search and rescue operations.

Further research could explore the integration of advanced navigation systems, obstacle detection, and autonomous landing capabilities. The findings of this project can serve as a foundation for further developments in drone landing technology. A problem to solve in the future would be a fusion of the information from the IMU sensor and the results of image processing technique to automate the decision regarding the possibility to land.

By incorporating another IMU within the system, we can potentially enhance the system's output to provide more reliable results, even if one of the system's applications, such as image processing or IMU becomes impaired or damaged. This specific situation is addressed in the chapter 6.

References

[1] L. Xin, Z. Tang, W. Gai, and H. Liu, "Vision-Based Autonomous Landing for the UAV: A Review," *Aerospace*, vol. 9, no. 11, p. 634, Nov. 2022, accessed: 2023-04-23. [Online]. Available: https://www.mdpi.com/2226-4310/9/11/634

[2] Accessed: 2023-06-01. [Online]. Available: https://content.arduino.cc/assets/st_imu_lsm6ds3_datasheet.pdf

[3] "PROJECTIONS OF LINES:," Oct. 2017, accessed: 2023-05-16. [Online]. Available: https://ktuengineeringgraphics.wordpress.com/projections-of-lines/

[4] "Arduino Nano 33 IoT," accessed: 2023-04-29. [Online]. Available: https://store.arduino.cc/products/arduino-nano-33-iot

[5] "Arduino Motor Shield Rev3," accessed: 2023-04-29. [Online]. Available: https://store.arduino.cc/products/arduino-motor-shield-rev3

[6] "Arduino Uno Rev3," accessed: 2023-04-29. [Online]. Available: https://store-usa.arduino.cc/products/arduino-uno-rev3

[7] "c925e-datasheet.pdf," accessed: 2023-04-29. [Online]. Available: https://www.logitech.com/content/dam/logitech/en_eu/video-collaboration/pdf/c925e-datasheet.pdf

[8] "Hitec D645MW D-Series High Torque 7.4V Digital Servo," accessed: 2023-05-07. [Online]. Available: http://www.espritmodel.com/hitec-d645mw-d-series-high-torque-7-4v-digital-servo.aspx

[9] "D645MW 32-Bit, High Torque, Metal Gear Servo | HITEC RCD USA," accessed: 2023-04-29. [Online]. Available: https://hitecrcd.com/products/servos/sport-servos/digital-sport-servos/d645mw/product

[10] M. Y. Arafat, M. M. Alam, and S. Moh, "Vision-Based Navigation Techniques for Unmanned Aerial Vehicles: Review and Challenges," *Drones*, vol. 7, no. 2, p. 89, Feb. 2023, accessed: 2023-05-31. [Online]. Available: https://www.mdpi.com/2504-446X/7/2/89

[11] A. C. Tsai, P. W. Gibbens, and R. H. Stone, "Terminal Phase Vision-Based Target Recognition and 3D Pose Estimation for a Tail-Sitter, Vertical Takeoff and Landing Unmanned Air Vehicle," in *Advances in Image and Video Technology*, ser. Lecture Notes in Computer Science, L.-W. Chang and W.-N. Lie, Eds. Berlin, Heidelberg: Springer, 2006, pp. 672–681.

[12] G. Xu, Y. Zhang, S. Ji, Y. Cheng, and Y. Tian, "Research on computer vision-based for UAV autonomous landing on a ship," *Pattern Recognition Letters*,

vol. 30, no. 6, pp. 600–605, Apr. 2009, accessed: 2023-06-01. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167865509000051

[13] Y. Fan, S. Haiqing, and W. Hong, "A Vision-Based Algorithm for Landing Unmanned Aerial Vehicles," in *2008 International Conference on Computer Science and Software Engineering*, vol. 1, Dec. 2008, pp. 993–996.

[14] M.-K. Hu, "Visual pattern recognition by moment invariants," *IRE Transactions on Information Theory*, vol. 8, no. 2, pp. 179–187, Feb. 1962, conference Name: IRE Transactions on Information Theory.

[15] A. Yol, B. Delabarre, A. Dame, J.- Dartois, and E. Marchand, "Vision-based absolute localization for unmanned aerial vehicles," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sep. 2014, pp. 3429–3434, iSSN: 2153-0866.

[16] Z. Fucen, S. Haiqing, and W. Hong, "The object recognition and adaptive threshold selection in the vision system for landing an Unmanned Aerial Vehicle," in *2009 International Conference on Information and Automation*, Jun. 2009, pp. 117–122.

[17] O. Shakernia, Y. Ma, T. Koo, J. Hespanha, and S. Sastry, "Vision guided landing of an unmanned air vehicle," in *Proceedings of the 38th IEEE Conference on Decision and Control (Cat. No.99CH36304)*, vol. 4, Dec. 1999, pp. 4143–4148 vol.4, iSSN: 0191-2216.

[18] S. Lange, N. Sunderhauf, and P. Protzel, "A vision based onboard approach for landing and position control of an autonomous multirotor UAV in GPS-denied environments," in *2009 International Conference on Advanced Robotics*, Jun. 2009, pp. 1–6.

[19] A. Benini, M. J. Rutherford, and K. P. Valavanis, "Real-time, GPU-based pose estimation of a UAV for autonomous takeoff and landing," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 3463–3470.

[20] H. Yuan, C. Xiao, S. Xiu, W. Zhan, Z. Ye, F. Zhang, C. Zhou, Y. Wen, and Q. Li, "A Hierarchical Vision-Based UAV Localization for an Open Landing," *Electronics*, vol. 7, no. 5, p. 68, May 2018, accessed: 21-5-2023. [Online]. Available: https://www.mdpi.com/2079-9292/7/5/68

[21] D. Simon, *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. John Wiley & Sons, 2006.

[22] J. van de Loosdrecht, K. Dijkstra, J. Postma, W. Keuning, and D. Bruin, "Twirre: Architecture for autonomous mini-uavs using interchangeable commodity components," in *IMAV 2014: International Micro Air Vehicle Conference and Competition*, 2014.

[23] S. Cherlet and P. Maelegheer, "Developing an autonomous hexapod robot for environmental exploration," *Faculty of Engineering and Architecture, Universiteit Gent, Master's Thesis*, 2015.

[24] D. Lee, T. Ryan, and H. J. Kim, "Autonomous landing of a vtol uav on a moving platform using image-based visual servoing," in *2012 IEEE international conference on robotics and automation*. IEEE, 2012, pp. 971–976.

[25] H. Lee, S. Jung, and D. H. Shim, "Vision-based uav landing on the moving vehicle," in *2016 International conference on unmanned aircraft systems (ICUAS)*. IEEE, 2016, pp. 1–7.

[26] J. Dougherty, D. Lee, and T. Lee, "Laser-based guidance of a quadrotor uav for precise landing on an inclined surface," in *2014 American Control Conference*. IEEE, 2014, pp. 1210–1215.

[27] "Vector Projection Formula," Feb. 2022, accessed: 2023-05-21. [Online]. Available: https://www.geeksforgeeks.org/vector-projection-formula/

[28] "Blob Analysis | ViSCO Technologies Corporation," accessed: 2023-06-01. [Online]. Available: https://www.visco-tech.com/english/technical/direction-presence/blob/

[29] T. Kumar and K. Verma, "A Theory Based on Conversion of RGB image to Gray image," *International Journal of Computer Applications*, vol. 7, no. 2, pp. 5–12, Sep. 2010, accessed: 2023-06-01. [Online]. Available: http://www.ijcaonline.org/volume7/number2/pxc3871493.pdf

[30] "How to use imfill() function to fill the interior of the boundary surface using Matlab? - MATLAB Answers - MATLAB Central," accessed: 2023-05-23. [Online]. Available: https://se.mathworks.com/matlabcentral/answers/769527-how-to-use-imfill-function-to-fill-the-interior-of-the-boundary-surface-using-matlab?requestedDomain=

[31] R. Bhat and B. Mehandia, "Recognition of vehicle number plate using matlab," *International journal of innovative research in electrical, electronics, instrumentation and control engineering*, vol. 2, no. 8, pp. 1899–1903, 2014.

[32] "Dilate an Image to Enlarge a Shape - MATLAB & Simulink - MathWorks Nordic," accessed: 2023-05-23. [Online]. Available: https://se.mathworks.com/help/images/dilate-an-image.html

[33] "imerode (Image Processing Toolbox User's Guide)," accessed: 2023-05-23. [Online]. Available: http://matlab.izmiran.ru/help/toolbox/images/imerode.html

[34] "Structuring Elements - MATLAB & Simulink - MathWorks Nordic," accessed: 2023-05-23. [Online]. Available: https://se.mathworks.com/help/images/structuring-elements.html

[35] P. K. Sahoo, S. Soltani, and A. K. Wong, "A survey of thresholding techniques," *Computer vision, graphics, and image processing*, vol. 41, no. 2, pp. 233–260, 1988.

[36] "Hough transform," May 2023, accessed: 2023-05-23. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Hough_transform&oldid=1153215890

[37] "Nothing Phone (1)," accessed: 2023-06-03. [Online]. Available: https://in.nothing.tech/pages/phone-1