

**VISUAL ODOMETRY ASSISTED QUADCOPTER FOR  
AUTOMATED STOCK COUNTING**

**A PROJECT REPORT**

*Submitted by*

**KARNIK RAM**

**HARISH S.**

**APEKSHA AVINASH SOMALINGA**

*in partial fulfillment for the award of the degree*

*of*

**BACHELOR OF ENGINEERING**

*in*

**ELECTRONICS AND COMMUNICATION ENGINEERING**

**SSN COLLEGE OF ENGINEERING**

**ANNA UNIVERSITY : CHENNAI 600 025**

**APRIL 2017**





# ACKNOWLEDGEMENTS

We would first like to thank our guide **Ms. S. HANIS**, Assistant Professor, Department of ECE, and our co-guide **Dr. G. SATHEESH KUMAR**, Associate Professor, Department of Mechanical Engineering, for guiding and supporting us through our project, as well as for their valuable inputs, advice and suggestions.

Our sincere thanks to **Dr. S. RADHA**, Professor and Head of the Department of ECE, and our project coordinator **Dr. P. VIJAYALAKSHMI**, Professor, Department of ECE, for providing us with the freedom and facilities to pursue a project of our choice.

We would like to show our deepest respect and appreciation to the founder **Dr. SHIV NADAR**, Chairman, SSN Institutions and also our principal **Dr. S. SALIVAHANAN**, for their support.

We would also like to give a special mention to Rahul, Sankaran and Siva for their support and help.

Last but not the least, we would like to extend our heartfelt thanks to all the teaching and non-teaching staffs of our department as well as the college, without whose help we would not have been able to carry out our work.

**Karnik Ram**

**Harish S.**

**Apeksha Avinash Somalinga**

# ABSTRACT

There is a growing interest in the design of aerial vehicles with advanced autonomous capabilities and many have already found their way into military and civil applications. However, while the focus has been on the development of these outdoor vehicles, little attention has been given to Unmanned Aerial Vehicles (UAVs) in indoor environments. Such aerial robots would be valuable for indoor surveillance, bomb disposal, personal assistive devices, or stock counting, which is the focus of our work. We present a navigation system for an aerial robot that uses an optic flow-based vision system for its localization, making it suitable for cluttered urban and GPS-denied indoor environments. Stock counting (or inventory counting) is implemented by the placement and detection of ArUco Markers on individual packages. The system is implemented on a quadrotor with all the computations performed on-board, and its experimental results are presented.

## **TABLE OF CONTENTS**

<b>CHAPTER NO.</b>	<b>TITLE</b>	<b>PAGE NO.</b>
	<b>ABSTRACT</b>	<b>iv</b>
	<b>LIST OF TABLES</b>	<b>vii</b>
	<b>LIST OF FIGURES</b>	<b>viii</b>
	<b>LIST OF SYMBOLS</b>	<b>x</b>
<b>1.</b>	<b>INTRODUCTION</b>	<b>1</b>
	1.1 OVERVIEW . . . . .	1
	1.2 MOTIVATION AND OBJECTIVE . . . . .	1
	1.3 LITERATURE SURVEY . . . . .	3
	1.4 ORGANIZATION OF THE REPORT . . . . .	8
<b>2.</b>	<b>HARDWARE ASSEMBLY OF QUADCOPTER</b>	<b>9</b>
	2.1 INTRODUCTION . . . . .	9
	2.2 HARDWARE . . . . .	10
	2.2.1 Frame . . . . .	11
	2.2.2 Motors . . . . .	11
	2.2.3 Electronic Speed Controller (ESC) . . . . .	14
	2.2.4 Propellers . . . . .	15
	2.2.5 Battery . . . . .	16
	2.2.6 Flight Controller . . . . .	18
	2.2.7 Radio Controller (RC) . . . . .	21
	2.2.8 Sensors . . . . .	22

<b>3.</b>	<b>VISION SYSTEM AND MARKER DETECTION</b>	<b>25</b>
3.1	INTRODUCTION . . . . .	25
3.2	VISION SYSTEM . . . . .	25
3.2.1	Introduction . . . . .	25
3.2.2	Feature Extraction . . . . .	27
3.2.3	Feature Tracking . . . . .	28
3.2.4	Translation to Real World Displace- ments . . . . .	32
3.3	MARKER DETECTION . . . . .	35
3.3.1	Introduction . . . . .	35
3.3.2	ArUco Markers . . . . .	36
<b>4.</b>	<b>EXPERIMENTAL RESULTS</b>	<b>38</b>
4.1	INTRODUCTION . . . . .	38
4.2	QUADCOPTER ASSEMBLY . . . . .	38
4.3	FEATURE EXTRACTION AND TRACKING	41
4.4	OPTIC FLOW CALCULATION . . . . .	42
4.5	TRAJECTORY TRACING . . . . .	43
4.6	MARKER DETECTION APPLICATION . .	45
4.6.1	Introduction . . . . .	45
4.6.2	ArUco Marker Detection . . . . .	46
<b>5.</b>	<b>CONCLUSION AND FUTURE WORK</b>	<b>49</b>
5.1	CONCLUSION . . . . .	49
5.2	FUTURE WORK . . . . .	49
	<b>REFERENCES</b>	<b>51</b>

## LIST OF TABLES

<b>TABLE NO.</b>	<b>TITLE</b>	<b>PAGE NO.</b>
2.1	Sample Specifications of a Motor . . . . .	13
2.2	Specifications of Motor . . . . .	14
2.3	Specifications of Tattu 6000 mAh 4S Li-Po Battery Pack . . . . .	18
4.1	Position of selected feature across two consecutive image frames . . . . .	41
4.2	Dimensions of Rectangular Path . . . . .	44

## LIST OF FIGURES

FIGURE NO.	TITLE	PAGE NO.
2.1	Hardware Connection Diagram . . . . .	9
2.2	Pitch, Roll, and Yaw axes of a quadcopter . . . . .	10
2.3	Propeller Definitions . . . . .	16
2.4	Full Control Architecture . . . . .	20
2.5	Main port connections and LED indicators . . . . .	20
2.6	Side Port and Supply Connections . . . . .	20
3.1	Visual Odometry Process . . . . .	26
3.2	Levels of the Pyramid . . . . .	29
3.3	Translation to Real World Coordinates . . . . .	33
3.4	ArUco Marker: ID 100 . . . . .	36
4.1	Basic Components and Assembly . . . . .	39
4.2	Stages of Quadcopter Assembly . . . . .	39
4.3	Full Assembled Quadcopter . . . . .	40
4.4	During Outdoor Flight Tests . . . . .	40
4.6	Display of Optic Flow Vectors . . . . .	42
4.7	Visualizing pose . . . . .	42
4.8	Visualizing Trajectory . . . . .	43
4.9	Indoor Setup: View 1 . . . . .	44
4.10	Indoor Setup: View 2 . . . . .	45

4.11	Detection of ArUco Markers . . . . .	46
4.12	Outdoor Setup for Marker Detection . . . . .	47
4.13	Marker Frequency . . . . .	47
4.14	Indoor Marker Detection Setup . . . . .	48

# **LIST OF SYMBOLS, ABBREVIATIONS AND NOMENCLATURE**

<b>UAV</b>	Unmanned Aerial Vehicle
<b>GPS</b>	Global Position System
<b>VTOL</b>	Vertical Take-Off and Landing
<b>IMU</b>	Inertial Measurement Unit
<b>MAV</b>	Micro Aerial Vehicle
<b>DOF</b>	Degree of Freedom
<b>PD</b>	Proportional-Derivative
<b>PID</b>	Proportional-Integral-Derivative
<b>SLAM</b>	Simultaneous Localization and Mapping
<b>AR</b>	Augmented Reality
<b>BLDC</b>	Brushless Direct Current
<b>DC</b>	Direct Current
<b>RPM</b>	Rotations Per Minute
<b>ESC</b>	Electronic Speed Controller
<b>PWM</b>	Pulse Width Modulation
<b>Li-Po</b>	Lithium Polymer
<b>TX</b>	Transmitter
<b>RX</b>	Receiver
<b>RC</b>	Radio Controller
<b>LIDAR</b>	Light Detection and Ranging
<b>SIFT</b>	Scale Invariant Feature Tracking
<b>SURF</b>	Speeded Up Robust Features
<b>FOV</b>	Field of View

**ROS**      Robot Operating System

# **INTRODUCTION**

## **1.1 OVERVIEW**

Our final year undergraduate project is an automated stock counting quadcopter, that uses visual odometry for its navigation. Using a camera, the motion of the quadcopter is estimated by observing the movement of pixels (features) across a sequence of consecutive image frames. This movement is captured using a feature tracking Optic Flow based algorithm [8]. From these optic flow vectors, the pose of the quadcopter is estimated. This system is used to make the quadcopter autonomous. Predetermined waypoints are provided as input and using optic flow, the quadcopter navigates along the desired trajectory. This system is implemented on a quadrotor framework and all the image processing is performed on-board, eliminating the need of an external operator and off-board computation. This navigation system is then utilized, along with a marker detection module, in warehouse environments for inventory counting.

## **1.2 MOTIVATION AND OBJECTIVE**

The recent years have seen a surge in the development of Unmanned Aerial Vehicles (UAVs), and specifically the quadcopter, in many military, civilian, and commercial applications. Thanks to its simplicity, quadcopters are favoured by a wide range of users from hobbyists to industrialists. There has been extensive research on the design and operation of quadcopters in outdoor environments. With the advent of the Global Position System (GPS), navigation

was simple and effective, as long as the overhead obstructions were kept to a minimum. However, indoor environments pose a problem for the use of GPS. Alternative navigation systems were required to be developed and the development of UAVs for indoor applications are still in their nascent stages.

Visual odometry systems are well-suited for localization and subsequent navigation in these indoor and GPS-denied environments. The quadcopter can be manually controlled using the camera feed as the operator's eyes, or, the images can be fed as input to a controller for the purpose of correcting itself along a predefined path. Our objective is to use this system to make the quadcopter autonomous, without the need for a manual operator. The data from the camera is utilized in an optic flow based algorithm that is used to obtain the orientation and position of the UAV in any environment. Given the pose of the quadcopter at any instant, its navigation along a predefined set path can be governed by an on-board flight controller.

Inventory counting and stock retrieval in warehouses rely heavily on manual labour. However, this process can be automated with an effective reduction in manpower cost and process time. We thus propose using a quadcopter to detect individual stock in warehouses and maintain an inventory. A quadcopter is better suited than fixed ground machinery to access objects at higher levels. It also eliminates the need for a human operator, reducing labour costs. The total time taken to complete the counting process is also reduced significantly. Inventory counting is implemented by placing and detecting fiduciary markers on the sides of packages. For our system, we utilize ArUco Markers.

### 1.3 LITERATURE SURVEY

Unmanned Aerial Vehicles have been in use since the early 20th century. The origin of UAVs began in military warfare, as an alternative to manned aircrafts. UAVs were useful military weapons, serving as "flying bombs", eliminating the possibility of pilot casualties [2]. It has since then been quickly integrated into outdoor applications including surveillance, reconnaissance, mapping, border patrol, search and rescue, inspection, agricultural monitoring, and more recently, transport of goods. Through the years the design of both fixed-wing and rotary-wing UAVs has been repeatedly refined to bring it to its current state. Navigation in outdoor applications has been made easy with the development of the Global Positioning System (GPS). Video surveillance, inspection and monitoring, and transport of goods (to name a few applications) all use the unobstructed airspace above trees and tall buildings. This facilitates the use of GPS which utilizes satellite signals for triangulation.

While there has been extensive research on the development of outdoor applications, the advancement of UAVs indoors has been slow, limited by the unavailability of GPS inside closed roofs or remote locations. Localization and pose estimation has to be achieved by other means. Popular odometry solutions that arose were vision-based solutions since conventional odometry methods like accelerometers and gyroscopes were more prone to unpredictable errors. Feature points tracked between image frames captured by a camera gives an estimate of the relative motion of the UAV. Implementations of optic flow algorithms can be performed on-board the aircraft, providing a great deal of autonomy. The use of low resolution cameras also provides an inexpensive solution for navigation.

In recent years, various control algorithms have emerged based on optical flow. Bruno Herisse, Francois-Xavier Russotto, Tarek Hamel, and Robert Mahony [7] presented a nonlinear controller for hovering flight and touchdown control for a Vertical Take-off and Landing (VTOL) UAV using inertial optical flow. The VTOL vehicle is assumed to be a rigid body, equipped with a minimum sensor suite (camera and IMU), manoeuvring over a textured flat target plane. Two different tasks are considered in this paper: the first concerns the stability of hovering flight and the second one concerns regulation of automatic landing using the divergent optical flow as feedback information. A pyramidal implementation of the Lucas-Kanade algorithm [1] has been used for optical flow computation. Their prototype uses four different boards with microprocessors performing different functions. The paper outlines the control laws and provides a stability analysis. Experimental results on a quadrotor UAV demonstrate the performance of the proposed control strategy.

Simon Zingg, Davide Scaramuzza, Stephan Weiss, and Roland Siegwart demonstrated in [14] the safe navigation of a Micro Aerial Vehicle (MAV) through indoor corridors, and need for autonomy. In this paper, they presented an approach for wall collision avoidance using a depth map based on optical flow from on-board camera images. An omnidirectional fisheye camera is used as a primary sensor, while IMU data is needed for compensating rotational effects of the optical flow. The presented approach is designed for safely maneuvering a helicopter through an indoor corridor. Results based on real images taken in a corridor with textured walls are presented to back their proposed approach.

F. Kendoul, I. Fantoni, and K. Nonami presented an on-board vision based autopilot for aerial vehicles, designed using optic flow and a low-cost IMU.

They demonstrated successful automatic take-off and landing, hovering and trajectory tracking. A Gumstix microcontroller with 16MB memory was used to perform computations of the data received from a 10 DOF IMU and a small analog camera. A three Nested Kalman Filter framework is implemented to improve the efficiency of the optic flow computations. Importantly, this paper demonstrated that all the computations can be performed on-board, thereby avoiding significant transmission latencies.

In [13], XIAN Bin, LIU Yang, ZHANG Xu, CAO Meihui, and Wang Fu presented a PD controller which uses optical flow to obtain position and velocity feedback for the autonomous hovering flight control of a nano quadrotor UAV. The nano quadrotor UAV has a mass less than 100 g and is comparatively much smaller than the micro vehicles utilized in previous research. Due to the limited size and payload ability, a wireless camera is employed as the on-board visual device to obtain the position and translational velocity of the UAV. Experiment results are included to demonstrate the good control performance of the proposed design. This paper showed that optical flow does not require a great number of computations nor does it require artificial landmarks to be placed in the environment which may impose restrictions to the flying path of the quad, making it very suitable for unknown indoor environments. IMU data is fused with optical flow results to compensate for the roll and pitch motions of the quad.

A potential indoor application for a visually-guided quadcopter is person following or person tracking. T. Naseer, J. Sturm and D. Cremers [9] used an AscTec Pelican quadcopter, equipped with a depth camera that is warped into a virtual static camera for video stabilisation whose stream is then used for person tracking. A monocular camera is used for localisation of the system

based on the ARToolkitPlus library. Relative pose of the person is determined and waypoints for the quadcopter are generated for maintaining the relative pose. However, additional camera payloads have to be suitably stabilised using IMU and visual pose estimates in order to perform any processing on them.

In a paper presented in the Australasian Conference on Robotics and Automation in Melbourne in 2014, Reuben Strydom, Saul Thurrowgood, and Mandyam V. Srinivasan [12] developed a vision-system consisting of two fish-eye cameras. The images obtained were stitched together upon which optical flow was calculated. Control of the aircraft is demonstrated by prescribing its trajectory in terms of a sequence of waypoints. Two nested PID controls were used to monitor the roll, pitch, and throttle settings of the UAV. The proposed method does not require prior knowledge of the environment and the control algorithm is shown to work smoothly for a limited set of test flights at two specified distances from the ground. The positional error is mainly due to the PID controllers acting more reactively than predictively.

During the IEEE 4th International Congress on Image and Signal Processing (CISP) held in 2011, Zdzisaw Gosiewski, Jakub Cieluk, and Leszek Ambroziak [6] proposed a method to process images obtained from a single camera to estimate the position and depth of an obstacle present in the UAVs trajectory. Again, a pyramidal Lucas-Kanade algorithm implements optical flow. Computational complexity is reduced by selecting risk areas containing important vectors of optical flow. The proposed vision-based obstacle avoidance algorithm is implemented using simulation and hardware on a UAV. This method demonstrated that low-resolution images are sufficient for the implementation of the algorithm although higher resolution increases accuracy with a trade-off in speed of response.

Besides design and control refinement, efforts have been made to develop cost-effective implementations. Jakob Engel, Jurgen Strurm, and Daniel Cremers [3] developed a low cost quadrotor coupled with a ground-based laptop to navigate autonomously in GPS-denied environments using a monocular SLAM system, an extended Kalman filter for data fusion and state estimation and a PID controller to generate steering commands. The system is shown to be robust to significant communication delays and loss of tracking. Compared to visual odometry, visual SLAM maintains a global estimate of the robot path. Visual SLAM is much more precise, but computationally expensive and not robust.

With the advent of various vision-based odometry techniques, unmanned aerial vehicles are finding its way into a lot of indoor applications. One such application which requires more autonomy is stock counting in warehouses. Currently, quadcopters are manually employed to scan each package for the barcode, using an onboard barcode scanner, to perform stock counting. With a vision-based odometry platform such an operation can be made autonomous without the need for human intervention.

Though there are various marker systems that can be used for product detection, few have certain advantages over other. In the IEEE Computer Society Conference on Computer Vision and Pattern Recognition in 2005, Mark Fiala [4] presented a new fiducial marker system which can be used for Augmented Reality (AR), robot navigation, and general applications where the relative pose between a camera and object is required. The author created a planar pattern marker system with 2002 markers to improve upon the existing marker systems called ARtags. Each marker consists of square border of either polarity for easy detection and the information is digitally encoded within the border. ARTag has low false positive rate, low false negative rate and low inter-marker

confusion rate that can work under adverse lighting conditions irrespective of the orientation of the marker.

## **1.4 ORGANIZATION OF THE REPORT**

The organization of the report is as follows: Chapter 2 describes the hardware setup of the quadcopter. The algorithms implemented for the vision system and the marker detection module are outlined in Chapter 3. Chapter 4 details the experimental results obtained at various stages of implementation. Finally, Chapter 5 summarizes our observations and conclusion, along with potential applications and future work.

# HARDWARE ASSEMBLY OF QUADCOPTER

## 2.1 INTRODUCTION

The first task of the project was the assembly of the quadrotor framework upon which our navigation system was to be tested. This involved the selection of components according to compatibility and payload constraints. Fig. 2.1 shows the various connections involved in the process. This section gives a brief overview of the working of a quadcopter and each component that was used in the assembly.

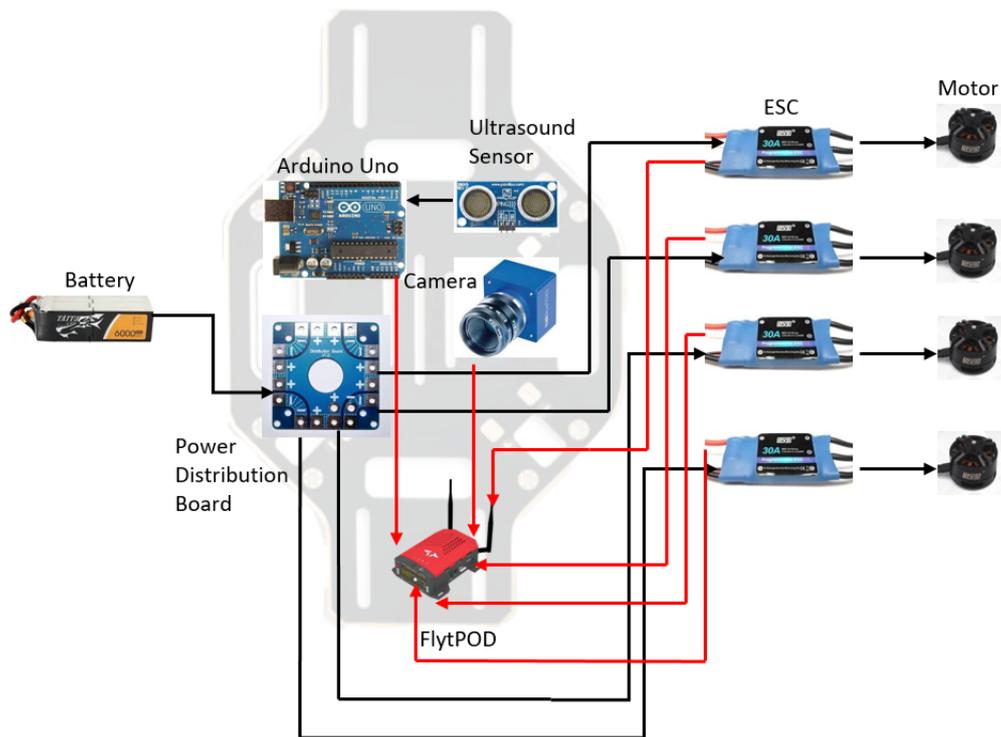


Figure 2.1: Hardware Connection Diagram

## 2.2 HARDWARE

A quadrotor was used as a framework to demonstrate our visual navigation system. As the name suggests, a quadrotor (or quadcopter) comprises of four propellers driven by motors, and situated at equal distances from the center of the body. The angular velocity of the motors decide the amount of thrust provided while the direction of rotation decides the movement of the quadrotor. For take-off, landing, and hovering, the diagonally opposite motors spin in the same direction. This allows for vertical take-off and landing (VTOL), which lends the quadrotor advantages while navigating through limited spaces. Additional thrust is provided by increasing the speed of particular motors. For example, to tilt the quadcopter in either the left or right directions, the speed of the motors on the right or left sides respectively are increased.

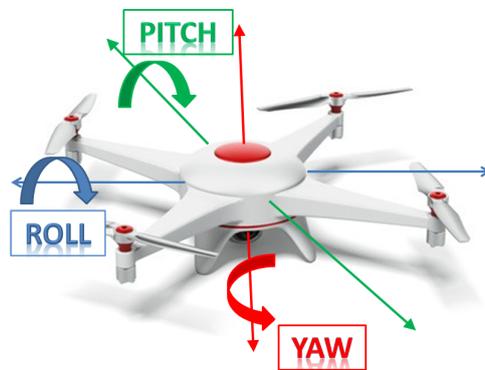


Figure 2.2: Pitch, Roll, and Yaw axes of a quadcopter <sup>1</sup>

The motion of a quadcopter is described in terms of three axes of rotation. These are the yaw, pitch, and roll axes. The yaw axis extends from the top to bottom of the quadcopter body; the pitch axis from the left to right; and the roll axis is perpendicular to the first two. Any movement of the quadcopter consists of a combination of rotations about these axes. These rotations are controlled

<sup>1</sup>Robert. "A Few Flight Related Terms." *No Runway* 28 Jun. 2014. Web. 20 Mar. 2017.

by the amount of thrust provided to each of the motors.

Building a quadcopter requires an understanding of each of its components. The components are chosen based on payload considerations, desired flight time, size and over-all cost of the quadcopter. The components that were selected based on these considerations for our framework are described below.

### 2.2.1 FRAME

The frame of a quadcopter must be robust, durable, lightweight, and must be able to support the desired payload. Payload includes weight of the quad's components as well as any additional load the quad has to carry, which will depend on the final application. The frame used for our system was the Hobbyking X650F Glass Fiber Quadcopter Frame. Made of glass fiber and aluminium, the Hobbyking frame is lightweight enough to allow for easy take-off, yet rigid enough to support the necessary components. The universal motor mounts are compatible with a wide range of motors, making it more flexible. The frame also sports extended landing gear providing the needed ground clearance for supporting the monocular camera which comprises our vision system. There is a dedicated camera attachment for easy mounting.

### 2.2.2 MOTORS

Brushless DC (BLDC) motors are the preferred choice over brushed DC motors because of their higher efficiency. Brushed DC motors contain permanent magnets as the stator and electromagnets as the rotor. For the rotor to rotate, the polarity of the electromagnet is changed by mechanical brushes. Al-

though the setup is simple, the brushes are prone to wear and tear easily and they limit the speed of the motor. Brushless DC motors do away with the use of brushes by switching the materials used for the stator and rotor. The permanent magnets then serve as the rotor while the electromagnets are the stator. The polarity of the electromagnets can now be controlled by a computer, making it more precise while reducing electric noise. Brushless motors offer more power and higher run times. Choice of the motors depends on the overall weight of the quadcopter. The motors should also be compatible with the frame used.

The important terminologies related to a BLDC motor are its RPM(kV rating), thrust, current rating and power based upon which the motor is selected. The KV rating on a BLDC motor is equal to the rotations per minute (RPM) per volt applied to the motor. So a BLDC motor with a KV rating of 1000 KV will spin at 1000 RPM when 1 volt is applied. If 12 volts are applied, it will spin at 12000 RPM.

Apart from the motor KV and thrust, the motor's power and efficiency are important specifications. A 70% efficient motor produces 70% power and 30% heat. A 90% efficient motor produces 90% power and 10% heat. With less efficient motors, not only a lot of power is wasted as well as flight time, but also a smaller thrust is achieved on full throttle. The most important fact is that because the motor runs so inefficiently, the response time suffers. It will take the motors a longer time to change the RPM and this influences the stability of the quadcopter, making it less stable. A sample specification of a motor is shown in Table 2.1

An important factor to consider is the thrust to weight ratio. The general rule is to select a motor that will provide a thrust that is 50% greater than the weight of the quadcopter. This will allow hovering at just over half throttle. However,

Specifications	Values
KV Rating	1300 rpm/v
Max. Power	190 W
Max. Thrust	920 g
Weight	53 g
Shaft Diameter	3.175 mm
Shaft Length	45 mm

Table 2.1: Sample Specifications of a Motor

this is only a general rule and it ultimately depends on the kind of flight desired; gentle or aggressive. To be able to perform complex and aggressive maneuvers, it is recommended to have a higher thrust to weight ratio. For a more gentle and steady flight as in the case of aerial photography, the weight is increased to allow hovering at around 70% throttle.

For our system, we desired a quadcopter that would be capable of performing complex maneuvers if necessary. If the thrust provided by the motors are too little, the quad will not respond well to our control, and even have difficulties taking off. If the thrust is too much, the quad might become too agile and hard to control. To this purpose, we used the following formula:

$$\text{Required Thrust per motor} = \frac{(\text{Weight} \times 2)}{4} \quad (2.1)$$

Thus if the all-up weight of the quadcopter is approximately 1kg, the total required thrust is 2 kg with each motor providing 500 g. Taking the above considerations into account, we used a BLDC motor with the specifications given in Table 2.2.

Specifications	Values
KV Rating	820 rpm/v
Max. Power	144 W
Max. Thrust	830 g
Weight	50 g
Shaft Diameter	28.5 mm

Table 2.2: Specifications of Motor

### 2.2.3 ELECTRONIC SPEED CONTROLLER (ESC)

Electronic Speed Controllers (ESC) serve as an interface between the motor and the receiver input. It receives the ppm (pulse position modulation) signal input from the user (or flight controller) and controls the speed of the spinning rotor by regulating the amount of power sent to it. ESCs have three sets of wires: one set receives power from the battery, one set plugs into the receiver and the last set powers the motors.

Control of the motors is achieved through Pulse Width Modulation (PWM) in which power supplied is switched on and off at a certain rate. The rate of switching controls the speed of the motors. For maximum throttle, power is always supplied whereas for half throttle there is a 50% duty cycle and so on. The overall efficiency is affected by the speed at which this switching takes place. The higher the rate, the more efficient the ESC as they lose less power as heat in the controller. Switching rates at around 3000 Hz are about optimum. Anywhere between 1000 Hz and 5000 Hz is acceptable.

ESCs are rated for a specific voltage and maximum current. The ESC chosen must be able to handle the voltage provided by the quad's battery pack. However, Lithium Polymer (Li-Po) batteries will also get permanently damaged if the voltage falls below 3 volts. Thus ESCs with a low voltage cutoff

should be chosen. This will cut power to the motor when the voltage drops. ESCs chosen should ideally have a maximum current rating slightly higher than that drawn by the motors at full throttle. A higher current rating increases both the cost and weight of the ESC. We don't want heavy ESCs adding to the dead-weight of the quad. Too much current may also damage the ESC quickly. Thus the choice of ESCs will be a result of careful considerations of the above mentioned factors.

#### 2.2.4 PROPELLERS

A quadcopter uses four propellers - two in the clockwise direction and two in the anti-clockwise direction. The choice of propeller depends on the overall weight of the quad and the compatibility with the motor and frame being used. Usually, the frame exercises certain limitations on the size of propellers that can be used. Propellers are characterized by their length and pitch. For example, 9"x4.7" propellers are 9 inches long and have a pitch of 4.7 inches. The size of a propeller is the maximum distance measured from one tip to the other. It is sometimes described in terms of diameter - the diameter of the circle formed when the propeller spins. Pitch is the distance travelled by one propeller rotation.

Propellers generate thrust by spinning and moving air. A larger diameter and pitch length would move more air, drawing more current from the motors and hence making it harder to maintain the RPM. A suitable propeller is one that finds a balance between the diameter and pitch length. A larger diameter pulls more current (and thus power) while a smaller pitch length propeller can spin faster. To perform stable maneuvers, a smaller propeller is generally preferred as it provides good speeds with less strain on the power system. It is also

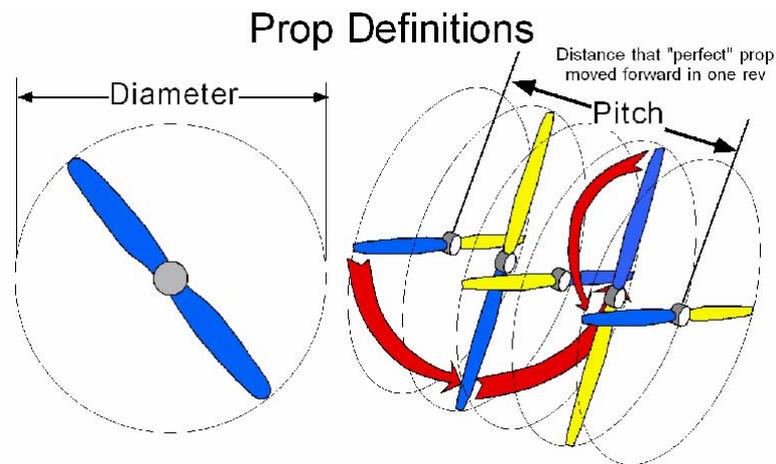


Figure 2.3: Propeller Definitions <sup>2</sup>

easier to start and stop the spinning of a smaller propeller because of its smaller inertia.

Like the frame, propellers can also be made of different materials like wood, carbon fiber and plastic. Each material has its uses. For example, carbon fiber propellers are preferred for their durability and noiseless spinning while plastic propellers generate a greater thrust. We have used glass fiber propellers with a prop length of 10 inches and a pitch of 4.5 inches in our system (10"x4.5"). The front two propellers are coloured differently from the back two to differentiate and obtain an understanding of the orientation of the quad while flying.

### 2.2.5 BATTERY

The main battery of the quadcopter supplies power to its motors (through the ESCs), the flight controller, radio controller, and sensors. Earlier designs required separate batteries for the receiver and motors due to the difference in voltage requirements, but this added to the dead-weight of the quadcopter which was not desired. However, with the inclusion of built-in Battery Elimi-

<sup>2</sup>"Basic Flight Aerodynamics", *D. L. Engineering*. 10 Feb. 2014. Web. 20 Mar. 2017.

nator Circuits in the ESCs, this is no longer a requirement and one main battery suffices.

The universal choice of battery for quadcopters is the Lithium-ion Polymer (Li-Po) battery. Li-Po batteries are chosen for their light weight, compactness and relatively higher discharge rates. There are three specifications to be considered while choosing the type of Li-Po battery. The first is voltage. A single Li-Po cell has a nominal voltage of 3.7 V (4.2 V at full charge). Connecting several cells in series causes the voltages to become additive. The number of cells added depends on the total voltage desired. Cell counts are denoted by the number of cells followed by an 'S'. A 4S Li-Po, therefore, is a battery of four 3.7 V cells providing an overall voltage of 14.8 V.

The second is the capacity of the battery, measured in milliamp-hours (mAh). The maximum discharge rate of the battery is denoted by 'C', and this serves as the third specification. A 20C pack can be discharged at a rate 20 times its capacity. The discharge rate together with the capacity is used to calculate the amount of current generated by the battery. Therefore, a 20C battery pack with a capacity of 4000 mAh will generate a current of  $20 \times 4000 \text{ mA} = 80,000 \text{ mA}$  or 80 A during discharge. Similar to ESCs, the general rule is to have a battery discharge rate higher than the combined current draw of the motors.

An additional specification is the number of cells connected in parallel. A parallel connection increases the capacity rather than the voltage and is denoted by the letter 'P'. Therefore, a 4S2P battery pack consists of 2 sets of 4-cell batteries connected in parallel. The battery pack used in our system is the Tattu 6000 mAh 4S Li-Po Battery Pack whose specifications are given in Table 2.3.

Battery capacity and battery life affect flight time. The general rule is to

Specifications	Values
Capacity	6000 mAh
Max. Continuous Discharge	25C
Burst Discharge	50C
Weight	595 g
Dimensions	165 × 65 × 53 mm
Voltage	14.8 V

Table 2.3: Specifications of Tattu 6000 mAh 4S Li-Po Battery Pack

fly the quad at 80% capacity. It is important to constantly monitor the battery voltage while flying because a drop in the voltage below a certain voltage may cause the quad to fall from the sky. Monitoring battery voltage helps take precautions against this by landing before the voltage drops to low levels.

*Calculation of maximum flying time:* A constant current draw of 20 A from a 2200 mAh Li-Po will get you:  $(2200mAh)/(1000)/20A = 0.11 \times 60 = 6.6$  minutes of flying.

## 2.2.6 FLIGHT CONTROLLER

The flight controller is the hardware that governs the flying of the quadcopter. It contains a microprocessor unit on which flight control algorithms can be flashed. The controller receives its input from sensors and user commands, and sends the corresponding control signals to the motors. Most flight controllers have both a hardware and software component, and the choice of controller depends on the kind of application. All controllers have basic sensors like gyroscopes and accelerometers built in, while some more advanced ones also have barometers and magnetometers. This combination of sensors is called the Inertial Measurement Unit (IMU), which exists within the con-

troller. The measurements from the IMU can be used to sense the orientation of the quad. Peripherals like GPS, sonar and ultrasound are all connected to the flight controller's input channels. In addition to these external sensors, the UAV's radio controller will also have a connection to convey the user's manual controls.

There exists an extensive variety of controllers on the market catering to a wide customer base. Commercial quadcopters use close-source controllers which doesn't allow the user to change any of the factory control settings. These have fixed purposes and can be used for applications like aerial photography and surveillance. However, many open-source controllers are also available, giving the user freedom to customise controls. While choosing a flight controller, one must ensure that it supports the software needed for the application. To provide more flexibility, open-source platforms are preferred but this involves a lot of trial and testing on the user's behalf.

For our system, we used FlytPOD by NavStik which is a flight computer with an in-built flight controller. Sporting the Samsung Exynos 5422 Octa-core processor with 2 GB RAM, a powerful Integrated GPU and 32 GB storage, FlytPOD has all the necessary capabilities to perform extensive on-board computations. It also contains a 3-axis accelerometer, 3-axis gyrometer, 3-axis magnetometer, barometer and a GPS for all orientation and navigation purposes. This, along with its support for external gimbals like cameras and LIDAR, FlytPOD serves as a complete package for all applications. Fig. 2.4 describes the communication between the sensors, FlytPOD and the commands given to it.

As the FlytPOD serves as the brain of the quadcopter, all peripherals like the camera and radio controller receiver are connected to it. The figures below detail the various port connections.

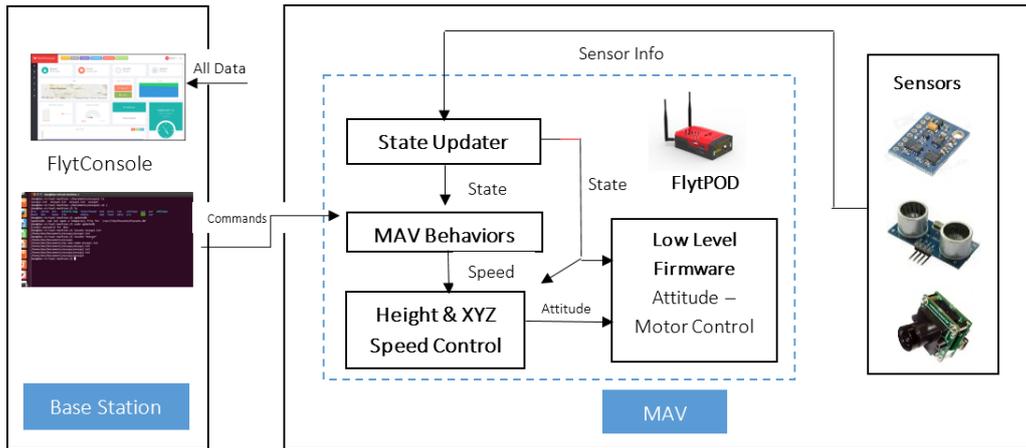


Figure 2.4: Full Control Architecture

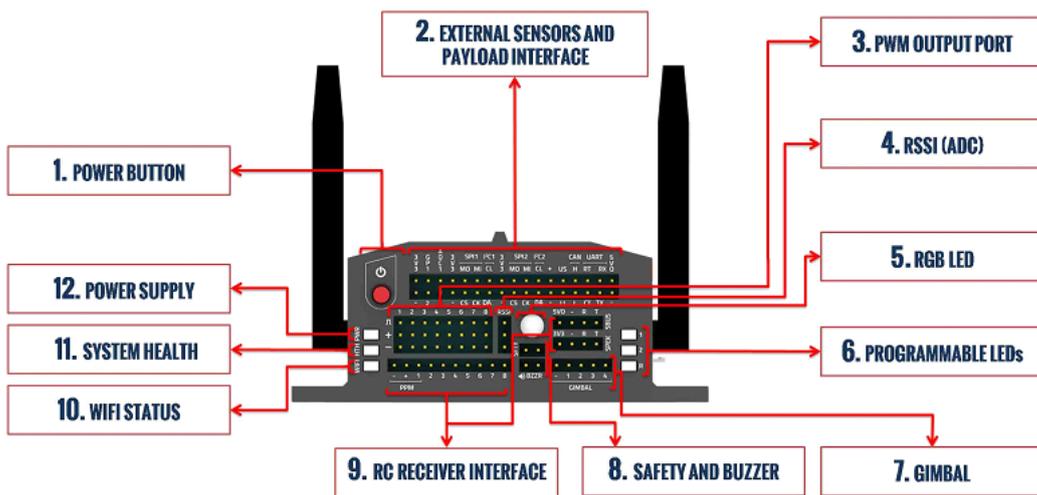
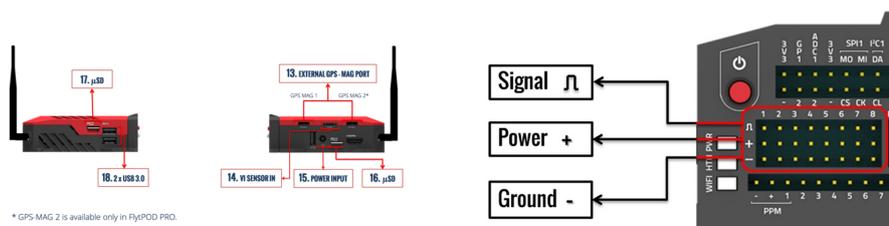


Figure 2.5: Main port connections and LED indicators



(a) Side Port Connections

(b) Supply Connections

Figure 2.6: Side Port and Supply Connections

### 2.2.7 RADIO CONTROLLER (RC)

Manual control of the quadcopter is achieved through radio control. The radio controller consists of a transmitter (TX) attached to the handset paired to a receiver (RX) present on-board the quadcopter. The signals are transmitted over radio frequencies.

The *number of channels* determines how many individual actions on the aircraft can be controlled. For example, channel 1 for throttle, channel 2 for yaw (rotating right and left), channel 3 for pitch (leaning forward and backward), channel 4 for roll (leaning left and right). Four channels is the minimum number of channels needed to control a quadcopter. If there are additional channels present, they are normally used as AUX channels for switches and potentiometers. These can be used to change flight modes or trigger certain functions and features on the quad.

The *channel number* however, refers to the radio frequency used for transmission. Earlier RCS utilized the frequency bands in the MHz range. In this frequency range, each band was numbered and given a legal designation which meant that no two pilots could fly using the same frequency as it would mess up the transmission system. Different countries also had different band designations. This problem was overcome with the advent of 2.4 GHz RCs, which implements the frequency hopping technology thereby reducing the risk of unwanted radio interference.

The transmitter handset is often referred to as the radio and contains control sticks, buttons, switches, rotating dials and sliding levers on its front and side surfaces. Each is used to control a particular function (for example, one control stick controls the amount of throttle to be provided to the quad), or set

a particular flying mode (this depends on the RC set used). GHz transmitters usually have a small plastic antenna attached to the top. These antennas are smaller than the ones used in MHz transmitters because GHz frequencies have a smaller wavelength and hence require shorter antennas. The transmitter may also contain an LCD screen to display information about the current settings of the quadcopter as well as battery information.

The receiver is placed on the quadcopter body and is attached to the flight controller as an input. The number of connection slots on a receiver depends on the number of channels i.e. a 5 channel RX will have 6 slots - one for each channel plus one for the battery pack connection. More complex receivers will have more slots. The GHz RX contains two smaller antennas and thus the problem of properly securing the long antennas of the MHz RX is absent.

The FlySky FS-i6 2.4 GHz 6CH AFHDS RC Transmitter With FS-iA6 Receiver was used in our quadcopter system. With an RF range of 2.40-2.48 GHz and 6 frequency channels, the FlySky TX-RX set provides reliable interference free transmission. Each FlySky transmitter has a unique ID, which is saved by the receiver when binding and therefore allows it to accept data only from the unique transmitter. This avoids picking another transmitter signal and dramatically increases interference immunity and safety.

## 2.2.8 SENSORS

Sensors sense factors or characteristics of the environment and feed it as input to the quadcopter's flight controller. Sensors like gyroscopes and accelerometers are necessary to obtain and control the orientation of the quadcopter, whether it is flown manually or autonomously. Aside from the sensors

that are embedded in the flight controller, external sensors can also be utilized. Ultrasonic sensors, cameras and LIDAR (Light Detection and Ranging) sensors can be used for applications that require navigation and mapping of the environment.

A gyroscope measures rotational acceleration while an accelerometer measures lateral acceleration. Together, they can sense any movement of the quadcopter along and about the three axes of rotation. Magnetometers measure the magnetic field around it, and acts like a compass in determining the quadcopter's orientation in space. A barometer is a pressure sensor and measures the pressure around the quadcopter. As pressure varies according to height from the ground, it tells the quadcopter how high it is flying, relative to the point from which it takes the first measurement. To reduce disparities and improve accuracy, readings from all the above sensors are combined in a process known as sensor fusion. Fusing data from various sensors provides a better understanding of the quadcopter's position and orientation.

Our flight controller also includes an inbuilt Global Positioning System, or GPS. GPS is used to extract the quadcopter's exact location on Earth (global coordinates). This works by sending several radio wave signals to the appropriate satellites orbiting the Earth. A minimum of three signals is required to obtain the two-dimensional location i.e., latitude and longitude. A minimum of four signals is required for the three-dimensional position, including altitude of the receiver. The time taken for the signals to be reflected from these satellites and received by the GPS receiver is measured and using this along with the receiver's time offset and the positions of the satellite, the quadcopter's coordinates are triangulated.

In addition to the above sensors, we have used a monocular camera for

implementing the optic flow based navigation system. The images captured from the camera are sent to the controller to calculate optic flow. The optic flow vectors are then fed to the state estimator to calculate real-world coordinates of the quadcopter's position which act as inputs for navigation. Sony's PS3 Eye Camera was used for this purpose. The camera captures frames of size 640x480 pixels at 60 frames/second and 320x240 pixels at 120 frames/second.

# **VISION SYSTEM AND MARKER DETECTION**

## **3.1 INTRODUCTION**

The software module of our project was broadly divided into two tasks. The first involved the implementation of the visual navigation system to achieve autonomy. The second part was focused on the implementation of the marker detection module for our proposed warehouse application. This chapter details the algorithms that were employed for the development of the vision-based navigation system. The usage of such a system in a warehouse environment is then outlined.

## **3.2 VISION SYSTEM**

### **3.2.1 INTRODUCTION**

The vision system consists of a monocular camera attached to the base of the quadcopter body. This arrangement allowed an unobstructed view of the environment below, and also matched the cameras center with the quadcopters. The camera input was fed into the flight controller whose on-board processor executed the optic flow algorithm. The algorithm calculates the median optic flow vectors of consecutive images and is used to estimate movement. These vectors are fed into the state estimator which converts them into real world coordinates describing the pose of the quadcopter.

The task of navigation was further divided into three subtasks. The first was feature extraction. Optic flow needs reliable features it can track across image sequences. The second task was the actual tracking of these features to compute the optic flow field vector. The last involved the translation of the optic flow vectors into the real world displacements that are fed into the controller as input. Such a vision system is computationally inexpensive (and hence can be performed on-board) and real-time (there is no need for the transmission of images). Fig. 3.1 describes the steps involved in the visual odometry process.

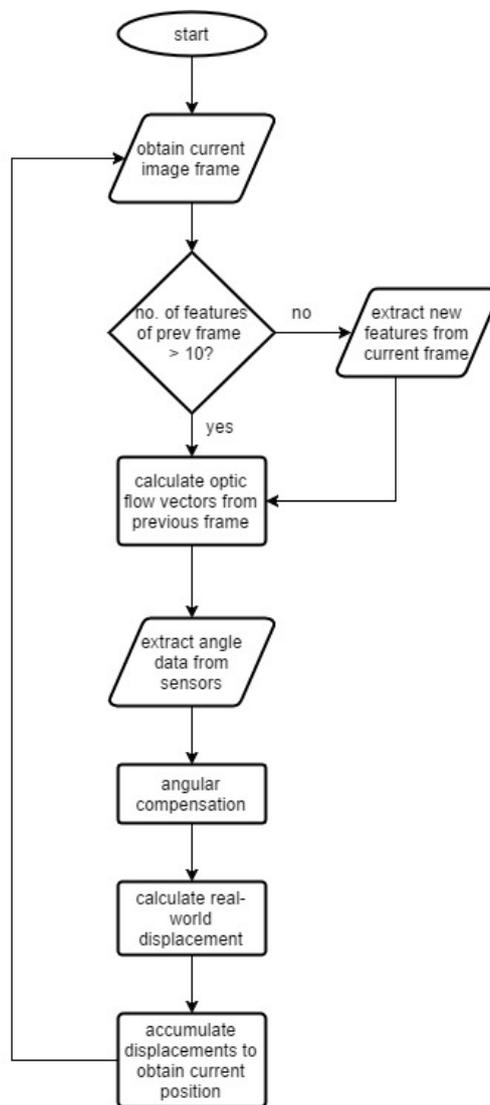


Figure 3.1: Visual Odometry Process

### 3.2.2 FEATURE EXTRACTION

Feature extraction is the process of extracting features or certain points in an image that are used for identification. A requirement for what classifies as a feature is its robustness, scale and rotation invariance. Image sequences can vary by scale and rotation, and hence the system should be able to identify the same features in consecutive images regardless of such changes.

The algorithm we used for the extraction of features is the Shi - Tomasi Good Features to Track algorithm [11]. The features extracted by this algorithm are rotationally invariant but do not exhibit scale invariance unlike SIFT or SURF. However, it is optimal for tracking and works better than Harris detection under affine transformations. Importantly, the process is computationally inexpensive and hence can be effectively implemented on-board the quadcopter, eliminating the need for communicating with an off-site processor that could produce transmission lag.

The algorithm is briefly described below.

1. Obtain the gradient image by applying derivatives at each pixel point
2. Compute the error function and second moment matrix for each pixel from the gradient image
3. Extract the eigen values from the second moment matrix and the following conditions are checked for to obtain 'cornerness':
  - The eigen values must be large and similar in magnitude. Small values correspond to flat regions, different magnitudes correspond to edges

- $\min(\lambda_1, \lambda_2) > \lambda$ , where  $\lambda$  is a predefined threshold
4. These corners are then stored into a vector that is passed onto the Pyramidal Lucas-Kanade algorithm for flow estimation.

### 3.2.3 FEATURE TRACKING

We employed optic flow in OpenCV for the purpose of tracking feature points between consecutive image frames. Optic flow is the apparent motion of brightness patterns in a series of images. It is assumed that the intensity of moving pixels remains constant during motion (either of the camera itself or an object in the environment). Optic flow algorithms are used to then track these points (features) across images. Optic flow techniques can be classified into four groups of which the differential Lucas-Kanade [8] technique is widely used.

The main objective of the algorithm is to track the features that were extracted in the previous task across a sequence of image frames. The extracted features are rotation invariant and hence will hold good through both lateral and rotational motions of the quadcopter. Optic flow searches for the minimum vector of difference between corresponding feature points in consecutive frames. In simple words, it searches for the distance by which a particular feature point has been displaced between consecutive frames. The median of these displacements of all extracted feature points in a window is defined to be the optic flow vector. This gives a measure of the displacement of the quadcopter as a whole.

The selection of the size of the tracking window, also called the integration window, involves a trade-off between accuracy and robustness. A smaller window is preferable to achieve higher sub-pixel accuracy so that the details are not

smoothed out. However, to handle larger motions, a larger window is preferable. This difficulty can be overcome by using a pyramidal implementation of the classical Lucas-Kanade algorithm. This is a coarse-fine strategy where the frames are divided into levels of different resolutions. The algorithm is iteratively applied at each level, beginning with the coarsest resolution down to the finest and most detailed. At each level, an estimate of the optic flow vector is calculated and translated to the lower level. The final result is the required optic flow vector.

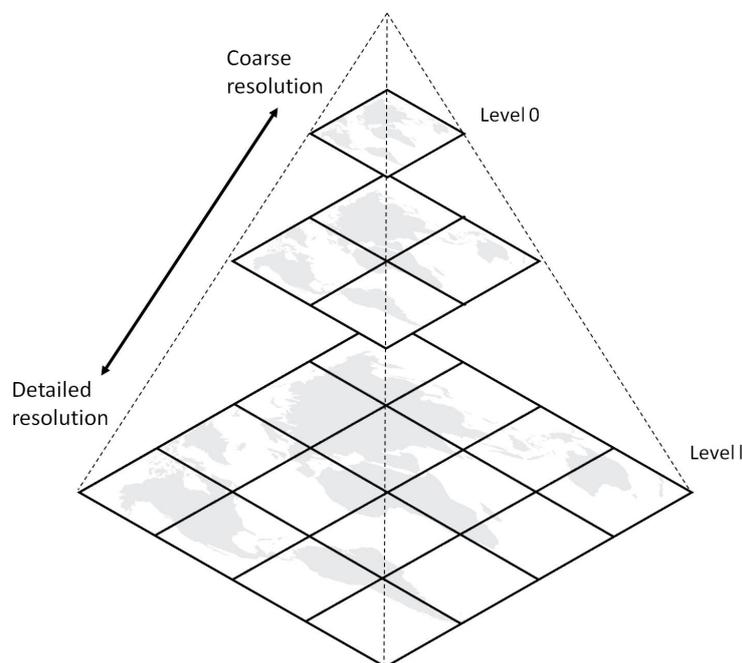


Figure 3.2: Levels of the Pyramid <sup>1</sup>

A brief description of the pyramidal Lucas-Kanade is as follows:

1. Consider  $I(\mathbf{x}) = I(x, y)$  and  $J(\mathbf{x}) = J(x, y)$ , representing two consecutive frames of images with  $I(\mathbf{x})$  as the first and  $J(\mathbf{x})$  the second.
2. Considering an image point  $u = [u_x \ u_y]^T$  on  $I(\mathbf{x})$ , the goal is to find the

<sup>1</sup>Garca R., de Castro JP., Verd E., Verd M. and Regueras L. (2012). "Web Map Tile Services for Spatial Data Infrastructures: Management and Optimization", Cartography - A Tool for Spatial Analysis, Dr. Carlos Bateira (Ed.), InTech.

point  $\mathbf{v} = \mathbf{u} + \mathbf{d} = [u_x + d_x \quad u_y + d_y]^T$  on second image  $J(\mathbf{x})$ , such that  $I(\mathbf{u})$  and  $J(\mathbf{v})$  correspond.

3. Image velocity  $\mathbf{d}$  is defined as the vector that minimizes the following equation:

$$\epsilon(\mathbf{d}) = \epsilon(d_x, d_y) = \sum_{x=u_x-w_x}^{u_x+w_x} \sum_{y=u_y-w_y}^{u_y+w_y} (I(x, y) - J(x + d_x, y + d_y))^2 \quad (3.1)$$

4. Since the pyramidal implementation is a coarse-fine strategy, before delving into the actual calculation of the optic flow vector, the image pyramid is first built. Using recursion, each level  $L$  of the pyramid is built using the equation given below:

$$\begin{aligned} I^L(x, y) = & \frac{1}{4} I^{L-1}(2x, 2y) + \\ & \frac{1}{8} (I^{L-1}(2x - 1, 2y) + I^{L-1}(2x + 1, 2y) + \\ & I^{L-1}(2x, 2y - 1) + I^{L-1}(2x, 2y + 1)) + \\ & \frac{1}{16} (I^{L-1}(2x - 1, 2y - 1) + I^{L-1}(2x + 1, 2y + 1) + \\ & I^{L-1}(2x + 1, 2y - 1) + I^{L-1}(2x - 1, 2y + 1)) \end{aligned} \quad (3.2)$$

The height of the pyramid  $L_m$ , is picked heuristically and the typical values are 2,3,4. Considering the size of the original image being captured, it usually doesn't make much sense to go beyond that.

Optic flow is first calculated at the deepest level of the pyramid (level  $L_m$  or the coarsest image). Then, the result is propagated to the upper level  $L_{m-1}$  using an initial guess for the pixel displacement. The results are propagated through the pyramid levels until the original image (level 0 or finest image) is reached.

5. Thus the initial pyramidal guess is calculated:  $\mathbf{g}^{L_m} = [g_x^{L_m} \ g_y^{L_m}]^T = [0 \ 0]^T$ .
6. The residual optic flow vector  $d^L$  is propagated to the next level using the new initial guess vector:  $\mathbf{g}^{L-1} = 2(\mathbf{g}^L + \mathbf{d}^L)$ . The residual optic flow  $\mathbf{d}^L$  is computed using iterative Lucas-Kanade. The following steps are computed within each pyramid level  $L$ .
7. The images  $I(\mathbf{x}) = I(x, y)$  and  $J(\mathbf{x}) = J(x, y)$  are redefined for computation purposes:  $A(x, y) = I^L(x, y)$  and  $B(x, y) = J^L(x + g_x^L, y + g_y^L)$ .
8. Redefining the image position and displacement vectors as well:  $p = [p_x \ p_y]^T = \mathbf{u}^L$  and  $v = [v_x \ v_y]^T = \mathbf{d}^L$ .
9. The goal is now to find the displacement vector minimizing the following function:

$$\epsilon(\mathbf{v}) = \epsilon(v_x, v_y) = \sum_{x=p_x-w_x}^{p_x+w_x} \sum_{y=p_y-w_y}^{p_y+w_y} (A(x, y) - B(x + v_x, y + v_y))^2 \quad (3.3)$$

10. This is achieved by setting the derivative of 3.3 to 0.

$$\text{i.e., } \frac{\delta \epsilon(v)}{\delta v} = [0 \ 0] \quad \text{at } v = v_{opt} \quad (3.4)$$

11. Differentiating 3.3 and solving, the solution can be expressed in terms of  $G$  and  $b$ , given below:

$$G = \sum_{x=p_x-w_x}^{p_x+w_x} \sum_{y=p_y-w_y}^{p_y+w_y} \begin{Bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{Bmatrix} \quad (3.5)$$

$$b = \sum_{x=p_x-w_x}^{p_x+w_x} \sum_{y=p_y-w_y}^{p_y+w_y} \begin{Bmatrix} \delta I & I_x \\ \delta I & I_y \end{Bmatrix} \quad (3.6)$$

12. Finally, the optimum optical flow vector can be calculated as:

$$v_{opt} = G^{-1}b. \quad (3.7)$$

13. The final optical flow solution is available after propagation through all pyramidal levels  $\mathbf{d} = \mathbf{g}^0 + \mathbf{d}^0$ . The location of the point  $\mathbf{u}$  can now be tracked to the point  $\mathbf{v}$  given by  $\mathbf{v} = \mathbf{u} + \mathbf{d}$ .

The advantage of this algorithm is that each residual optical flow vector  $\mathbf{d}^L$  can be small while computing a much larger overall displacement  $\mathbf{d}$ .

### 3.2.4 TRANSLATION TO REAL WORLD DISPLACEMENTS

The optic flow data calculated from the Pyramidal implementation of the Lucas-Kanade algorithm is used to calculate the real world displacement of the quadcopter. This translation is achieved by using the altitude data obtained from the sonar sensor. Subsequent motion of the quadcopter will generate more optic flow vectors which will be constantly added to the previous position of the quadcopter to update to its current position. Angular compensation of the pitch, roll, and yaw angles of the quadcopter was also performed.

Across consecutive image frames, the overall optic flow vector calculated can be used to estimate the displacement of the quadcopter in the image plane. The first step is to translate these displacements from the image plane to the real world plane. Consider the position of the quadcopter at a height  $h$  above

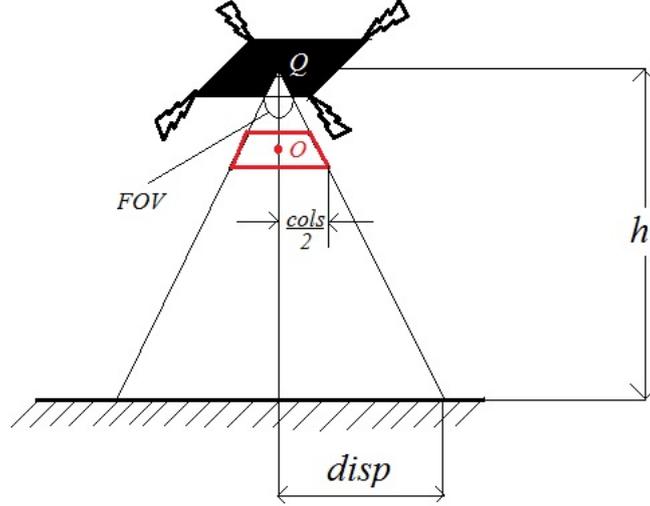


Figure 3.3: Translation to Real World Coordinates

the ground plane, as shown in the diagram. (see Fig. 3.3).

The image plane is indicated in red. Taking a point at the center of the quadcopter  $Q$  (where the camera is located) and drawing lines that graze the edges of the image plane gives us the field of view (FOV) of the camera. The normal from the quadcopter center  $Q$  passes through the image center  $O$ . If the total number of columns (of pixels) is given by  $cols$ , the center  $O$  is thus at a distance of  $cols/2$  from the edge of the image plane. Let the corresponding distance on the ground be  $disp$ . Using trigonometry, this distance is calculated in terms of the FOV and height  $h$ :

$$disp = h \times \tan(FOV/2) \quad (3.8)$$

If the quadcopter moves a distance equal to the  $cols/2$ , the corresponding distance is  $disp$ , according to figure 3.3. Let  $OF$  represent the optic flow vector across consecutive images. Thus the corresponding ground displacement can

be calculated using simple direct variation:

$$disp = \frac{2 \times h \times \tan(FOV/2) \times OF}{cols} \quad (3.9)$$

3.8 and 3.9 are general representations that can and are applied to both x and y directions. Thus the displacement in both x and y directions can be calculated. However, these equations hold true only if the quadcopter moves perfectly linearly in both directions. Practically, this is not possible because there will be a tilt in orientation for any movement that the quadcopter makes. This tilt (called pitch about the y axis and roll about the x axis) will also record optic flow, which does not contribute to linear motion and hence needs to be compensated for. This angular compensation is outlined below.

Consider the quadcopter titled at an angle  $\theta$  to the ground plane. For a roll angle equal to  $FOV/2$ , the center of the image plane  $O$  moves a distance of  $cols/2$ . Again by variation, the distance moved in the image plane corresponding to  $\theta$  is given by:

$$comp = \frac{\theta \times cols}{FOV} \quad (3.10)$$

This compensation is subtracted from the optic flow (both being in image plane coordinates), before the final distance is translated to real world displacements. Compensation is calculated for x and y directions as well. The final distance obtained after compensation is added to the previous position of the quadcopter to estimate the current position.

## **3.3 MARKER DETECTION**

### **3.3.1 INTRODUCTION**

Warehouses house a large number of commodities and for efficient running of a business, this stock must be tracked and well documented. Most companies employ stock and cycle counting methods for keeping a check on their inventory. While there are many methods that can be employed for cycle counting [10], all of them involve manual labour. Warehouses contain hundreds of packages, and manually sorting through them and documenting the count and location of each type is a tedious and time-consuming process. Thus, we propose a system to automate this process, cutting down on the manual labour as well as total stock counting duration. This poses a number of advantages including reduction of labour and operating costs.

Our solution involves using a quadcopter for scanning and recording details of the commodities. Using our optical flow based system for indoor navigation, a secondary camera attached to the side of the quadcopter is used to scan fiducial markers that are placed on each package. By providing the trajectory to traverse the length and different levels of each aisle in the warehouse, the quadcopter can quickly navigate autonomously and simultaneously scan the markers. Our navigation system is suitable for this application since the trajectory inside the warehouses are fixed and the path will not contain any obstacles to obstruct the movement of the quadcopter. Being able to easily access upper levels of the rack eliminates the need for ground machinery and the secondary manual operator required with it.

### 3.3.2 ARUCO MARKERS

For the purpose of inventory counting, we needed a simple and efficient method of detecting each box in the warehouse. Object detection was a potential but computationally intensive solution. The simpler way around the problem was to place fiduciary markers on the objects and detect those markers instead. These markers are generated to our convenience and placed in the position and orientation we desire. This reduces the complexity of detection, as opposed to correctly identifying and matching an unknown object in an environment. The markers we employed for this purpose are ArUco markers. [5]

ArUco markers contain black and white squares enclosed in a larger square. These squares correspond to binary digits or bits. A black square represents 1 while a white one represents 0. This inner binary matrix of squares is surrounded by a larger black border which allows for easier identification of the marker. The marker size determines the size of the inner matrix. For example, Fig. 3.4 has a marker size of  $10 \times 10$ , which means it contains 100 bits.

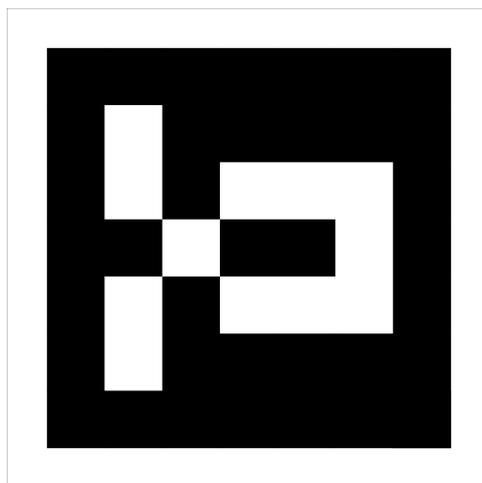


Figure 3.4: ArUco Marker: ID 100

The advantage of using ArUco markers is that they are rotation invariant,

i.e., they can be detected even when they are rotated in an image. Depending on the marker size, a dictionary of markers can be created. A dictionary is simply a list of all ArUco markers of the same marker size. The marker ID is not the decimal base conversion of the binary codification, however. This is due to the large number of bits in the inner matrix. Instead, the marker ID is the index number of the corresponding marker in the marker dictionary.

The marker dictionary that was employed contained markers whose IDs ranged from 0 to 1023. For our application, we assigned each product (package) with a particular marker ID. Thus, different packages could easily be identified and differentiated by their ArUco marker IDs.

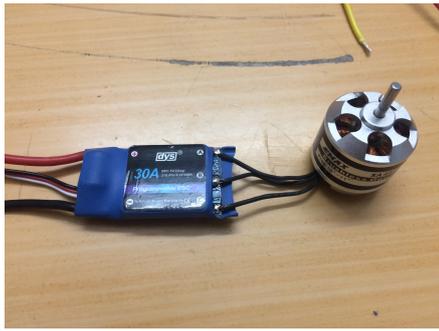
# EXPERIMENTAL RESULTS

## 4.1 INTRODUCTION

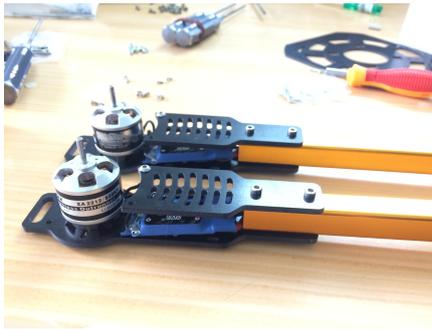
This chapter is dedicated to describing the results obtained over the various stages of project implementation. The assembly of the quadcopter is first shown. Images and data from testing of the vision-based algorithms are presented next. The translation of image displacements to real world displacements is visualized using the visualization tool RViz provided by ROS (Robot Operating System). Finally, the process of marker detection and display of the marker count for the application of inventory counting is presented.

## 4.2 QUADCOPTER ASSEMBLY

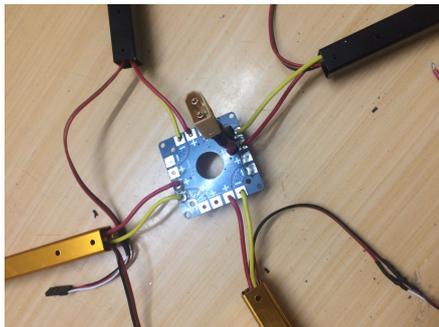
Fig. 4.1 shows the components used and their assembly for the quadrotor framework. Fig. 4.1a shows the Electronic Speed Controller used in our setup, connected to the brushless DC motor. Fig. 4.1b shows the motors bolted at the end of the quadcopter arms. Fig. 4.1c displays the distribution board that supplies the current to the ESCs which in turn send the appropriate level of current to the motors depending on the kind of motion to be performed. Fig. 4.1d shows how the previously mentioned components were connected.



(a) Electronic Speed Controller (ESC)



(b) Motors



(c) Distribution Board



(d) Connection of the motors, ESC and distribution board

Figure 4.1: Basic Components and Assembly

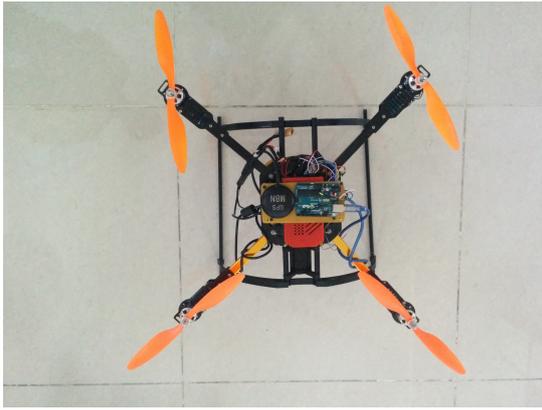


(a) With FlytPOD flight controller



(b) With propellers and landing gear

Figure 4.2: Stages of Quadcopter Assembly



(a) Top View



(b) Front View

Figure 4.3: Full Assembled Quadcopter

Fig. 4.2 shows the intermediate stages of assembly of the quadcopter (Fig. 4.2a is with the FlytPOD, Fig. 4.2b is with the propellers and landing gear). Fig. 4.3 shows the top and side views of the fully assembled quadcopter, with all necessary components for our application.



Figure 4.4: During Outdoor Flight Tests

Fig. 4.4 is a picture of the quadcopter during one of the manually controlled outdoor flight tests. Once the connections were secure and the quadcopter was stable when manually controlled, it was ready to be tested with the vision system on-board.

### 4.3 FEATURE EXTRACTION AND TRACKING

The vision system was simultaneously developed while the quadcopter framework was being assembled. However, testing was first performed off-board, before integration could take place. The feature tracking algorithm was first implemented on a PC before being loaded onto the quadcopter controller. The code was implemented using OpenCV. Figures 4.5a and 4.5b describe one of the initial tests performed using the PC front camera. The feature points are indicated by red points on the two images. Table 4.1 contains sample data describing the location of a single point (indicated in green) extracted from Frame 1 (on the left) and Frame 2 (on the right).



(a) Frame 1



(b) Frame 2

Frame	X-Coordinate	Y-Coordinate
1	244.281	210.084
2	228.078	130.497

Table 4.1: Position of selected feature across two consecutive image frames



## 4.5 TRAJECTORY TRACING

Using these optic flow vectors, we tested the trajectory of the quadcopter by manually moving it along a certain path. The distance travelled was visualized on the visualization application RViz provided by ROS. The rectangular path traced visualized is shown in Fig. 4.8.

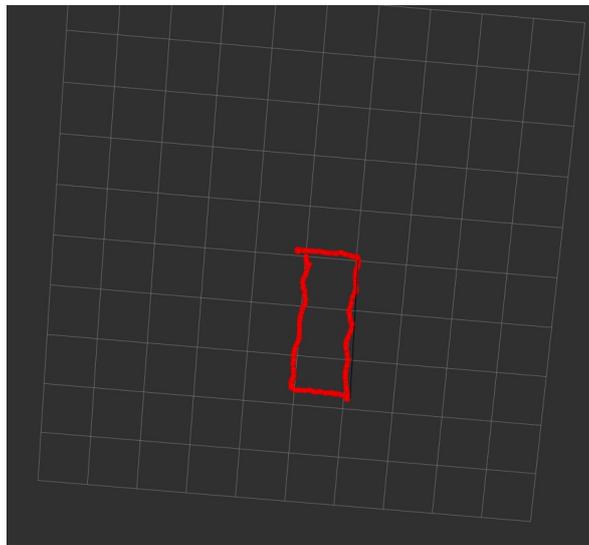


Figure 4.8: Visualizing Trajectory

Figures 4.9 and 4.10 show the indoor setup conditions under which the trajectory test was carried out and visualized. The length and breadth of the rectangular path as estimated by optic flow as well as the ground truths are tabulated and shown in Table 4.2. Outdoor testing was also carried out using the same setup but since the results achieved from the indoor setup were better, only those have been included. It is to be noted that setups with a textured ground surface achieved better results.

After several tests and refinements to the code, the data shown in Table 4.2 displays the results that were closest to the ground truth. Thus, once these visualized trajectory measurements were satisfactorily comparable to the ground



Figure 4.9: Indoor Setup: View 1

Measurement Obtained From	Length	Breadth
Ground Truth	2 m	0.8 m
Optic Flow Calculations	2.1 m	0.86 m

Table 4.2: Dimensions of Rectangular Path

truth (which was obtained by measuring distances along the ground with a measuring tape), the system was ready to be tested while flying the quadcopter. The first objective was to achieve hover control. The input coordinates fed into the system are the coordinates of the quadcopter during lift-off. By calculating the optic flow, the quadcopter can estimate the motion it needs to make in order to hold the current position. Testing the hover control (position hold) and trajectory tracing of the quadcopter while flying are the next steps that need to be carried out and constitute the immediate future work for this project.



Figure 4.10: Indoor Setup: View 2

## 4.6 MARKER DETECTION APPLICATION

### 4.6.1 INTRODUCTION

Our project focuses on stock counting or inventory counting in warehouses as an application of our navigation system. For this purpose, a secondary camera is attached to the side of the quadcopter. This camera is used to scan fiducial ArUco markers that are placed on warehouse packages. We first implemented the marker detection system using a PC that was attached to the monocular camera. Once the results were satisfactory, the program was transferred to the on-board computer and run simultaneously with the navigation system. The frequency of markers detected is collected and sent to a PC off-board to display

the required information.

#### 4.6.2 ARUCO MARKER DETECTION

Detection was first implemented on a PC using the secondary monocular camera. The boundary of the marker box and the marker ID are displayed on the input video feed and hence provides a visual representation of the detection process. The boundary of each marker is outlined in red and the marker ID is displayed in light blue text. Fig. 4.11 shows the marker detected in the frame, along with the details mentioned above. These figures are from the testing that was performed outdoors.

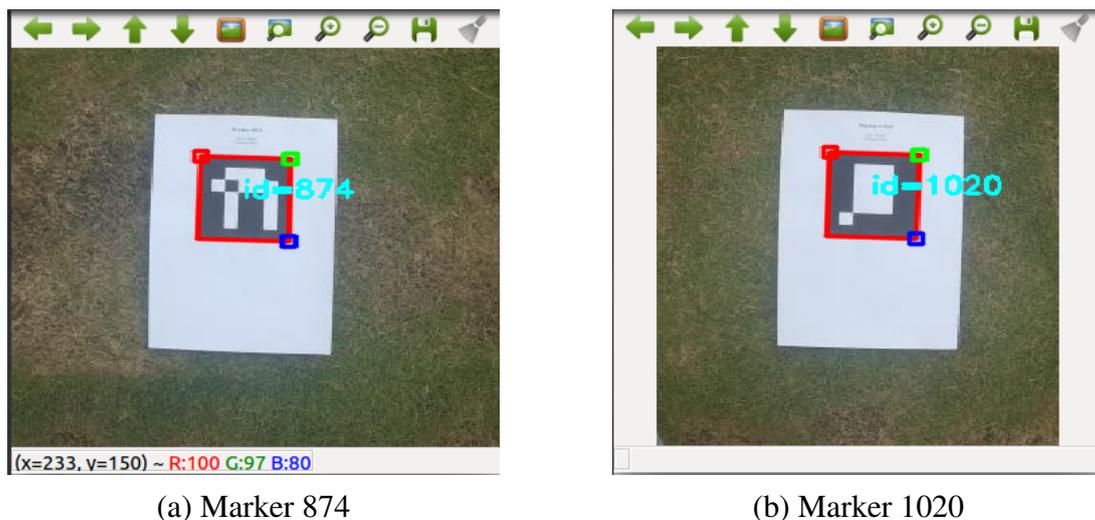


Figure 4.11: Detection of ArUco Markers

The outdoor setup for the detection of the ArUco markers is shown in Fig. 4.12. 6 ArUco markers were printed on individual sheets of paper and placed equidistant from each other on the ground. The quadcopter was then physically moved with the camera over the markers, starting from the top right marker and moving anticlockwise till the bottom right marker. The markers placed

had the following IDs: 201, 1020, 874, 777, 100, 1008. The frequency of each marker was then calculated and displayed on the console, as shown in Fig. 4.13.



Figure 4.12: Outdoor Setup for Marker Detection

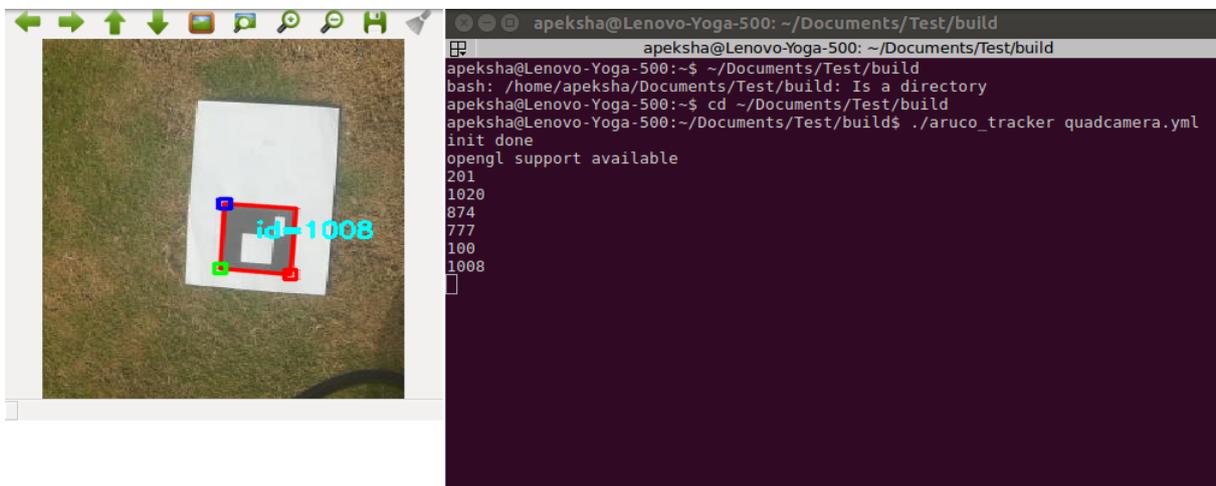


Figure 4.13: Marker Frequency

Finally, the marker detection module is to be executed while flying the quadcopter. This setup is shown in Fig. 4.14. Tests need to be carried out to verify the reliability of the marker detection module while flying.



Figure 4.14: Indoor Marker Detection Setup

# **CONCLUSION AND FUTURE WORK**

## **5.1 CONCLUSION**

A visual odometry system based on optic flow was developed and tested on a self assembled quadcopter. The system is to be used for the autonomous navigation of the quadcopter in GPS-denied indoor environments, such as warehouses. A pyramidal implementation of the Lucas-Kanade optic flow algorithm was employed for this purpose. Optic flow was calculated from consecutive image frames obtained from a monocular camera, attached to the bottom of the quadcopter, and these optic flow vectors were then utilized to estimate the pose of the quadcopter. With this information, along with a predetermined path fed as input in the form of way points, the quadcopter will follow the desired path. We aim to implement this system for inventory counting in a warehouse environment. An ArUco detection module was developed for this purpose that recognizes packages and keeps a count of them. The vision and marker detection modules were both individually tested with sufficient results. The final integration of all the modules for autonomous navigation is being tested and its results are being analyzed.

## **5.2 FUTURE WORK**

Autonomous navigation operations such as trajectory following and position hold are to be fully tested and recorded. The system is then to be integrated into a warehouse monitoring operation where a quadcopter navigates

along predetermined paths while the ArUco detection module detects and recognizes packages, and maintains an inventory check. To enhance the visual navigation system, obstacle avoidance and environment aware SLAM will be the next steps to implement. This would release the requirement of providing a predetermined path to the quadcopter for its navigation. Future work regarding the application is to integrate a gripper mechanism into the quadcopter design, expanding the application to stock retrieval as well. This would cut-down manual labour costs and operation times.

## REFERENCES

1. J. Y. Bouguet (2001). “Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm”. *Intel Corporation* 5.1-10, p. 4.
2. K. L. B. Cook (2007). “The Silent Force Multiplier: The History and Role of UAVs in Warfare”. *Aerospace Conference, 2007 IEEE*. IEEE, pp. 1–7.
3. J. Engel, J. Sturm, and D. Cremers (2012). “Camera-based navigation of a low-cost quadrocopter”. *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, pp. 2815–2821.
4. M. Fiala (2005). “ARTag, a fiducial marker system using digital techniques”. *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Vol. 2. IEEE, pp. 590–596.
5. S. Garrido-Jurado, R. Muñoz-Salinas, F. Madrid-Cuevas, and M. Marn-Jimnez (2014). “Automatic Generation and Detection of Highly Reliable Fiducial Markers Under Occlusion”. *Pattern Recognition* 47.6, pp. 2280–2292.
6. Z. Gosiewski, J. Ciesluk, and L. Ambroziak (2011). “Vision-based obstacle avoidance for unmanned aerial vehicles”. *Image and Signal Processing (CISP), 2011 4th International Congress on*. Vol. 4. IEEE, pp. 2020–2025.
7. B. Hriss, F.-X. Russotto, T. Hamel, and R. E. Mahony (2008). “Hovering flight and vertical landing control of a VTOL Unmanned Aerial Vehicle

- using optical flow”. *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*. IEEE, pp. 801–806.
8. B. D. Lucas and T. Kanade (1981). “An Iterative Image Registration Technique with an Application to Stereo Vision (IJCAI)”. *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI '81)*, pp. 674–679.
  9. T. Naseer, J. Sturm, and D. Cremers (2013). “FollowMe: Person following and gesture recognition with a quadrocopter”. *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE, pp. 624–630.
  10. M. D. Rossetti, T. Collins, and R. Kurgund (2001). “Inventory Cycle Counting—A Review”. *The proceedings of the 2001 Industrial Engineering Research Conference*. Vol. 1, pp. 457–463.
  11. J. Shi and C. Tomasi (1994). “Good Features to Track”. *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*. IEEE, pp. 593–600.
  12. R. Strydom, S. Thurrowgood, and M. V. Srinivasan (2014). “Visual Odometry: Autonomous Uav Navigation Using Optic Flow and Stereo”. *Proceedings of Australasian Conference on Robotics and Automation*.
  13. B. Xian, Y. Liu, X. Zhang, M. Cao, and F. Wang (2014). “Hovering control of a nano quadrotor unmanned aerial vehicle using optical flow”. *Proceedings of the 33rd Chinese Control Conference*. IEEE, pp. 8259–8264.
  14. S. Zingg, D. Scaramuzza, S. Weiss, and R. Siegwart (2010). “MAV navigation through indoor corridors using optical flow”. *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, pp. 3361–3368.