AUTONOMOUS NAVIGATION OF AN UNMANNED AERIAL VEHICLE USING INFRARED COMPUTER VISION

by

Ephraim Nowak

B.Sc., The University of British Columbia, 2015

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

in

THE COLLEGE OF GRADUATE STUDIES

(Electrical Engineering)

THE UNIVERSITY OF BRITISH COLUMBIA

(Okanagan)

April 2018

The following individuals certify that they have read, and recommend to the College of Graduate Studies for acceptance, a thesis/dissertation entitled:

Autonomous Navigation of an Unmanned Aerial Vehicle Using Infrared Computer Vision

submitted by Ephraim Nowak in partial fulfillment of the requirements of the degree of Master of Applied Science

Dr. Homayoun Najjaran, School of Engineering

Supervisor

Dr. Ayman Elnaggar, School of Engineering

Supervisory Committee Member

Dr. Dwayne Tannant, School of Engineering

Supervisory Committee Member

Dr. Rudolf Seethaler, School of Engineering

University Examiner

Abstract

The goal of this thesis is to develop infrared (IR) vision-based navigation methods for small unmanned aerial vehicles (UAVs). As small unmanned aerial systems (sUAS) see increased use, this technology promises to benefit many civilian applications including agriculture, fire suppression, search and rescue, as well as military operations. Small UAVs have proliferated the market, ranging from pocket-sized photography drones controlled using a smartphone, to larger systems used in remote surveying and disaster response applications. UAVs share many commonalities with ubiquitous smartphones, including a powerful on-board processor or Central Processing Unit (CPU), gyroscope, compass, barometer, GPS, and electro-optical colour camera. Incorporating vision-based navigation methods in a UAV autopilot makes use of the on-board camera and processor (often standard features on drones) for precise localization and navigation, in environments where other sensors may fail or prove inaccurate.

This work presents two types of high-level altitude and position control systems based on computer vision. Specifically, both navigation systems use IR technology; the first method uses an active IR beacon and computer vision methods to coordinate autonomous localization and landing of a small quadrotor UAV on a stationary platform. The autonomous IR-based landing method was tested using the AR.Drone quadcopter and a computer vision and control script developed in Python. The second method uses passive short-wave infrared (SWIR) and the Normalized Difference Vegetation Index (NDVI), used in remote sensing applications, to coordinate navigation of a small UAV in a GPS-denied indoor greenhouse environment. This method was integrated with a NAVIO2 autopilot to determine the UAV's position and orientation in a greenhouse row. The real-time image processing pipeline includes machine vision principles such as binarization, thresholding, edge detection, and line detection. Both methods presented in this thesis use an eye-in-hand approach to the image-based visual servoing problem to navigate and land the UAV.

Lay Summary

The goal of this thesis is to develop new vision-based autopilots for quadcopter drones in civilian applications. This includes automating the landing process using an on-board infrared (IR) camera and vision computer. Two systems are proposed and tested: the first system relies on IR beacons to automatically guide a drone to a safe landing site. The second method uses the passive IR signature of photosynthesizing vegetation (specifically, the geometry of a row of plants in a greenhouse) to compute the vehicle's position and orientation. These parameters are subsequently used to guide the drone along the row for agricultural purposes. These methods are desirable in an indoor environment where a GPS signal is likely not available.

Preface

This research was done under the supervision of Dr. Homayoun Najjaran in the Advanced Control and Intelligent Systems laboratory at the School of Engineering of the University of British Columbia. The work presented in this thesis was done over the course of my Master of Applied Science degree.

Chapter 3 includes work done during an undergraduate research project with Kashish Gupta. A version of Chapter 3 has been published [1]. I was responsible for designing and theorizing the idea, assembling the hardware, conducting the experiments, analyzing the results, and preparing the manuscript. Vision software was written by Kashish Gupta, who also assisted in initial experimental tests.

Chapter 4 is based on work conducted during a research internship in the Robotics and Control Laboratory at Ben-Gurion University of the Negev under the supervision of Dr. Shai Arogeti. I was responsible for adapting an existing computer vision algorithm [2], implementing the new algorithm, constructing the hardware for the test platform, conducting the experiments, formulating the results, and preparing the findings for publication. The original corridor navigation algorithm was published [2], and adapted to the vegetation navigation problem.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

Abbreviation	**Definition**
API	Application programming interface
APM	ArduPilot Mega (autopilot)
COTS	Commercial off-the-shelf
CPU	Central Processing Unit
DOF	Degree of Freedom
EO	Electro Optical
FPS	Frames per Second
GIS	Geographic Information System
GPIO	General Purpose Input / Output
GPS	Global Positioning System
HD	High Definition
HSV	Hue Saturation Value (referring to colour system)
IBVS	Image Based Visual Servoing
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
IR	Infrared
LED	Light Emitting Diode
LiPo	Lithium Polymer (batteries)
LWIR	Long Wave Infrared
NDVI	Normalized Difference Vegetation Index
NIR	Near Infrared
NoIR	No Infrared (camera)
RGB	Red Green Blue (referring to colour channels)
ROI	Region of Interest
SAR	Search and Rescue
SBC	Single-board computer
SLAM	Simultaneous Localization and Mapping
sUAS	small Unmanned Aerial System

SWIR	Short Wave Infrared
UAS	Unmanned Aerial System
UAV	Unmanned Aerial Vehicle
VTOL	Vertical Takeoff and Landing

Acknowledgements

I would like to express my sincere gratitude to everyone who supported me over the course of my graduate studies. Firstly, I would like to thank my supervisor Dr. Homayoun Najjaran for his guidance, motivation, and support during our work together over the last four years. His facilitating nature encouraged me to pursue learning opportunities and experiences above and beyond my degree requirements, which significantly enhanced my studies and personal development. His collaborative nature allowed me to learn from, and contribute to many projects in the lab and in industry, strengthening my interest in the field of engineering. I would like to thank my committee members Dr. Ayman Elnaggar and Dr. Dwayne Tannant for their encouragement and valuable input throughout this work.

Special thanks go to Kashish Gupta for his on-going encouragement and support, as well as his collaboration while developing more efficient methods for testing and implementing embedded image processing on-board a UAV. I would like to thank Dr. Shai Arogeti of the Robotics and Control Laboratory at Ben-Gurion University of the Negev for supervising me during my NSERC-MSFSS research internship in Israel. The skills and experience I learned during this time are invaluable and opened many new avenues for me. I am very grateful to Dr. James Yu for his support while applying for the NSERC-CGS scholarship.

I truly enjoyed working with my fellow ACIS Laboratory students, including Bara Emran and Nasser Alqahtani. I would also like to thank my undergraduate assistants Jessica Rempel and Kurt Yesilcimen for their help on various projects. Finally, I would like to thank my family members, who have been my greatest supporters not only during my graduate degree, but throughout my entire life. Without the support from my parents, I would not have been able to complete this work. I would also like to thank my brother, Raphael Nowak, for his scientific advice and literary assistance during the writing of this thesis. Their guidance, motivation and love will continue to shape my life and career.

Most of all, my Father in Heaven has blessed and supported me in all aspects of my life; my thesis research is no exception.

Dedication

This work is dedicated to יהוה *(YHWH), the G-d of Israel, the maker and sustainer of the universe, who was, and is, and is to come. All the glory be to Yeshua HaMashiach for his salvation!*

Chapter 1: Introduction

An unmanned aerial vehicle (UAV) is defined by Transport Canada as a power-driven aircraft, other than a model aircraft, that is designed to fly without a human operator onboard [3].

UAVs have proven their capabilities in both military and civilian applications, and Canada has shown itself to be a world leader in the development of UAV technology and policies surrounding their use [4]. Small UAVs have proliferated the market, ranging from pocket-sized photography drones controlled using a smartphone, to larger systems used in remote surveying and disaster response applications [5]–[7]. UAVs share many commonalities with ubiquitous smartphones, including a powerful on-board processor or central processing unit (CPU), gyroscope, compass, barometer, GPS, and electro-optical colour camera.

The Federal Aviation Association (FAA) refers to UAVs within the larger context of an Unmanned Aircraft System (UAS). An unmanned aircraft system is an unmanned aircraft and the equipment necessary for the safe and efficient operation of that aircraft. An unmanned aircraft is a component of a UAS. It is defined by statute as an aircraft that is operated without the possibility of direct human intervention from within or on the aircraft [8].

As small Unmanned Aerial Systems (sUAS) see increased use, they promise to benefit many civilian applications including agriculture, fire suppression, search and rescue, as well as military operations. UAVs are often used in applications too dangerous for human operators, including search & rescue and wildfire deployments. Some of the civilian uses of UAVs include:

- Remote Sensing: surveying & crop health [9]–[11]
- Search & Rescue [5], [12]–[15]
- Videography & Photography
- Environmental Monitoring: pipeline inspection & wildfire monitoring [16], [17]

Despite their increased use, one of the major barriers to wider deployment of UAVs is the lack of situational awareness associated with current autopilot systems. This limitation places a line-of-sight restriction on UAV pilots, thereby constraining the operational usefulness of UAVs in many industrial applications such as the oil & gas industry, fighting of forest fires, and search & rescue activities. This thesis presents methods for increasing the autonomy of UAVs through smarter on-board processing and decision-making.

The goal of this thesis is to develop infrared (IR) vision-based navigation methods for quadrotor Unmanned Aerial Vehicles (UAVs). Once navigation and landing using active IR beacons was successfully demonstrated, the more difficult challenge of indoor navigation using passive IR features was approached. This was done in an indoor GPS-denied environment (a greenhouse) within the context of a larger project with the goal of evaluating the use of UAVs to pollinate tomato plants using the Normalized Difference Vegetation Index (NDVI) index of photosynthesizing plants.

The most popular UAVs can be categorized as either fixed-wing aircraft or rotary wing aircraft [18]. Fixed-wing UAVs are typically larger in size, can carry a heavier payload, and have longer flight times. However, they usually require a runway for takeoff and landing, or a combination of a catapult and landing net if a runway is not available.

Rotary-wing UAVs include helicopters and multicopters, and are quite popular due to their vertical takeoff and landing (VTOL) capabilities. Helicopter UAVs function similarly to a conventional manned helicopter, with a main rotor and an anti-torque tail rotor [19], [20]. Multicopters have some advantages over the conventional helicopter, including mechanical simplicity and increased maneuverability. Most common multirotors have four propeller blades, and are called quadcopters. Other common configurations have six blades (hexacopter) or eight blades (octacopter). Many industries and researchers use quadrotor UAVs, as they are lightweight systems with well-known system dynamics. Such systems have a typical flight time between 15 and 20 minutes, and can carry a small payload of a few kilograms.

Finally, some specialty UAVs include lighter-than-air UAVs such as blimps, flexible wing UAVs, and ornithopters which mimic the flight behaviour of insects or birds.

1.1 Objectives

The two methods presented in this thesis include:

- Autonomous landing of a quadrotor UAV on a stationary platform in an indoor environment using computer vision and active IR beacons
- Autonomous navigation of a quadrotor UAV in an indoor GPS-denied environment using passive IR feature detection and geometry

The combination of these two developments results in an sUAS capable of performing a completely autonomous mission including takeoff, navigation, and landing in an indoor GPS-denied environment using an on-board IR camera and vision-processing computer. The entire flight can be accomplished using standard hardware available on-board the UAV, requiring no separate ground station or operator.

Additionally, this system has the advantage of using the application-specific imager (IR camera) for navigation purposes, instead of simply considering it to be a passive payload which records data for later analysis.

1.2 Motivations

Small multirotor unmanned aerial vehicles (UAVs) have become extremely popular in recreational and industrial applications in recent years. Due to their affordability and portability, UAVs have replaced their manned counterparts in many industries. Their commercial uses include mine surveying and pipeline inspections, as well as applications in law enforcement, search & rescue operations, and wildfire monitoring.

The most complicated and delicate phase of a UAV flight is the landing phase [21]. This is further compounded in GPS-denied environments or vision-compromised situations encountered during nighttime operations or in smoky conditions. Although many industries require UAVs to be operated specifically during such sub-optimal conditions, these environments present a challenge to common autopilot systems. Therefore, the focus of recent research has been on the development of autonomous colour and IR vision-based landing systems [22]. Such systems can locate landing zones without requiring GPS waypoints, and can do so during day- or nighttime. In this thesis, a review of the two most common techniques for IR vision-based autopilot design is provided. The first method uses active IR beacons operating on shortwave frequencies, while the second method uses passive longwave IR signatures for navigation.

The lack of situational awareness inherent in most commercial autopilot systems places a line-of-sight restriction on UAV pilots, thereby constraining the operational usefulness of UAVs in civilian applications. This research project seeks to incorporate biologically-inspired sensing into UAVs to augment their awareness of the surrounding environment. Computer vision modalities including colour vision and IR detection are incorporated into existing UAV autopilot systems, allowing the vehicle to make self-directed navigational adjustments in real-time, based on environmental stimuli. This will allow for an increased level of autonomy in applications requiring the UAV to venture beyond the operator's line of sight [23].

This project develops a UAV autopilot system with heightened awareness of its surroundings using vision-based sensing methods. This research takes a giant leap towards solving the sense-and-avoid limitation of current UAVs, and promises to open countless new applications of UAVs; applications which will contribute to saving lives and assets of Canadians, while improving the overall quality of life in Canada.

This research breaks new ground in a cutting-edge industry; an industry with projected sales of $90 billion over the next decade [24], and far-reaching effects that will touch many aspects of modern life.

1.3 Problem Statement & Proposed Solution

This thesis seeks to solve the problem of autonomous UAV navigation and landing in an indoor GPS-denied environment using IR-based computer vision.

A computer vision system for UAV navigation and landing using an IR camera and on-board vision processing computer is presented. The problem is divided into two subproblems. The first is autonomous landing on a stationary platform using active IR LED markers on the ground. The second is autonomous navigation within a greenhouse row bordered by vegetation using passive (reflected) IR radiation.

The proposed solution for the autonomous landing system is presented in Chapter 3. The ground station consists of a landing platform and high-powered IR radiator. A camera capable of detecting IR signals is mounted in a down-facing configuration on the UAV. The camera is processed using an on-board single-board computer (SBC), and high-level position control commands are sent over a WiFi network to the drone's autopilot. Once positioned directly over the landing zone, the UAV's altitude is gradually decreased until a safe landing can be achieved.

The proposed solution for the autonomous greenhouse navigation system is presented in Chapter 4. The system is composed of a quadrotor UAV with a front-facing camera sensitive to IR light. The normalized difference vegetation index (NDVI) is used to selectively detect vegetation. Once detected, the geometry of the plants in the greenhouse row is used to determine the UAV's position and orientation within the row. This allows high-level angular rate (yaw, pitch, and roll) control commands to be sent to the UAV autopilot using the on-board localhost network.

The combination of these two systems presents a standalone solution for autonomous IR-based navigation and landing in an indoor GPS-denied environment. All image processing is done on-board using lightweight, low-cost hardware. Instead of relying on additional sensors, such as sonar or LiDAR, the proposed system takes advantage of the on-board camera found almost ubiquitously on UAVs. The use of on-board processing removes the need for a wireless communication link between the ground station (landing zone) and the UAV. The vehicle can simply be placed in an environment and will perform the required processing on-board without any special room setup. This system can act in tandem with, or augment, existing navigation methods using GPS, INS, and barometers. For example, INS sensors and barometers are used to provide high-frequency data to the low-level stability autopilot controller, while the proposed vison system provides low-frequency position commands to the vehicle.

1.4 Related Work

A brief outline on existing methods for UAV navigation and landing is provided below. To match the objectives of this thesis, the related work is presented in two parts: active IR and passive IR.

1.4.1 Autonomous Navigation and Landing Using Active IR

Vision-based landing techniques for UAVs have been studied in indoor and outdoor environments, but the fundamental solutions remain the same for both environments [25], [26]. Oftentimes indoor test setups are small-scale replicas of full-scale field setups [27]. A common method of localization and navigation in autonomous vehicles is the use of shortwave IR beacons [28]. In such a system, the landing zone is outfitted with a single IR beacon, or multiple beacons arranged in a predetermined pattern [29], which are located by a camera on-board the UAV. A vision-processing computer is used to analyze the images during flight, and guides the vehicle to a safe landing. Wenzel et al. [30] demonstrated an indoor IR-based landing system using a Wii remote camera. In their experiment, a microcontroller on the UAV was used to detect a pattern of IR LEDs on the ground. They achieved a fast system response time by using a hardware-based IR tracker, while employing colour vision as a secondary verification for the existence of the landing zone. Their system worked well on close range, but if the distance between the UAV and target became too large, the camera was not able to detect the point-source LEDs required by their classification algorithm. Gui et al. [31] tested a similar IR vision-based landing system for fixed-wing UAVs in an outdoor environment. Their system used a series of IR lamps arranged on a runway to estimate the UAV's position, and guide the vehicle to a landing. Their system also used a hardware DSP processor for the IR camera to achieve fast response time, and the centers of the IR lamps were detected using blob detection. Their successful field trials demonstrated the benefits of UAV-board hardware processing. Gui et al. [31] and Xu et al. [32] both used a specific arrangement of IR targets to positively identify the landing zone. For example, the use of an active cooperative "T"-shaped target to estimate UAV pose is described [32]. An active target was used to counter the effects of image blurring due to bad weather, as this resulted in more consistent image features. They also demonstrated the application of their system by autonomously landing on a target ship [27]. Alternatively, the UAV airframe itself can be equipped with infrared emitters to be detected by a ground-station camera, as demonstrated by Lugo [33]. These studies demonstrate the use of hardware-based image processors for fast response time, and the use of shortwave IR beacons to locate a landing zone and estimate relative UAV pose. In some cases, however, it is not feasible to use active IR beacons, so an object's passive longwave signature is used.

1.4.2 Autonomous Navigation and Landing Using Passive IR

The use of passive IR vision to identify targets has been studied by other researchers [34]. In this method, thermal signatures are detected and classified, similar to feature classification using colour vision. Yang et al. [35] describes the use of a monocular vision system to take off, hover, and land using a micro aerial vehicle. Their system used real-time landing pad recognition to determine its location in an indoor environment. Some outdoor landing systems have also been developed for both fixed-wing and rotary-wing UAVs. Kummer et al. [36], [37] developed a vision-based landing system to guide fixed-wing UAVs to a safe landing on an inflatable airbag. As with all colour-based methods, this approach deteriorates in low light or vision-compromised environments. The same image feature classification strategies can, however, also be applied to longwave thermal imaging. Kim et al. [38] demonstrated a sensor-fusion algorithm which combines colour vision and longwave IR imaging in a firefighting robot. In the same way, heat signatures can also be used to locate other sources, such as outlines of ships. Yakimenko et al. [39] demonstrate that a ship's thermal signature can easily be detected on the open ocean; the smokestack is especially visible to a UAV at long distances. An inverted approach was taken by Kong et al. [40]. They placed an IR stereo vision system on the ground and used the long-wave IR heat signature radiated by the UAV for localization. The stereo IR vision system was mounted on an actuated pan/tilt head, allowing it to track the movement of the UAV and guide it towards a simulated aircraft carrier deck. This method seems attractive, as placing the camera on the ground, rather than on the airframe, reduces UAV payload weight and allows for increased computational power provided by off-board computers. This separation, however, also severely limits the system, as a communications link is mandatory between the ground station and the UAV. Regardless of experimental setup, each of these studies [38]–[40] demonstrates the successful application of image feature classification algorithms to longwave IR images to detect and land on a target. Another method of passive IR-based navigation relies on short-wave IR and the Normalized Difference Vegetation Index (NDVI). The NDVI is a graphical indicator used to detect vegetation, and is often used in remote sensing applications to monitor plant health. It can also be used is geophysical surveys to detect anomalies [10]. The NDVI theory is detailed in Chapter 4, and its usefulness for navigation purposes is evaluated.

A review of the two most common IR vision-based UAV landing methods was provided. The first method, using active shortwave IR beacons, is especially practical when a specific landing zone is known and marked. The use of small, lightweight hardware-based processing methods allows for fast system response times ideal for close-range autonomous landing. The second method, using an object's passive IR signature, is better suited to situations where the landing zone location may change over time, or when it is impractical to set up IR beacons. With the advent of smaller single-board computers, a combination of these two methods may be feasible in the future; longwave IR could be used to detect the landing target from a distance, while shortwave IR would aid in the final landing approach.

1.5 Thesis Outline

Chapter 2 provides background information and an overview of concepts which are necessary to understanding the methods described in this thesis. This chapter offers an introduction to computer vision and image processing, and briefly describes the quadrotor UAV and related control concepts. Chapter 3 provides details on the implementation and testing of the developed active IR-based autonomous landing system. Chapter 4 extends the autonomous IR-based navigation problem to the application-specific challenge of navigating in a greenhouse using passive IR. Both chapters outline experimental design and results. Chapter 5 concludes the work by highlighting the contributions made, and suggesting further improvements and continued research.

Chapter 2: Background

This chapter presents a brief description of concepts and methods used provide the reader with a better understanding. It presents an introduction to basic terms and concepts of image processing and computer vision, as well as an introduction to control concepts related to the quadrotor UAV. To start, a history of the field of photography and an overview of the development of aerial photography is given. Next, the topics of colour theory and colour models are introduced. This is followed by an introduction to computer vision and various image processing techniques employed in the computer vision pipeline. This chapter concludes with a discussion on the quadrotor UAV.

2.1 History of Photography

A thesis on the topic of computer vision would seem incomplete without a brief history of photography. Concepts from the field of photography are vital to this work, as the phrase "garbage in = garbage out" could not be truer when it comes to computer vision. No amount of image processing can compensate for a poor input image.

The term "photography" comes from two Greek words "photo" and "graphy" which means "writing with light". Although the word "photography" was used as early as 1932 in various circles, it was Herschel's 1839 paper, "On the Art of Photography; or the Application of the Chemical Rays of Light to the Purpose of Pictorial Presentation" [41] that finally gave photography a common nomenclature [42], [43].

The pinhole camera concept, also known as the camera obscura, was first written about by the Chinese in 470 BCE. It was used during the 16th century by artists including Michelangelo and Leonardo da Vinci to help them draw pictures by tracing. In the 1700s, the camera obscura was made portable by placing it in box with a pinhole on one side and a glass screen on the other; light was projected onto the glass screen. In 1826, a French scientist, Joseph Nicéphore Niépce put a plate coated with bitumen in a camera obscura for eight hours. This became the earliest known photograph still in existence today (Fig. 2.1). Niépce shared his findings with Louis Jacques Mande Daguerre, a Parisian artist. The two became partners three years later. In 1835, two years after Niépce died, Daguerre discovered that silver iodide was much more sensitive to light than Niepce's bitumen. The process, which Daguerre named the daguerreotype, was announced to the world on January 7, 1839, and was the predecessor to modern-day cameras. Multiple improvements to the recording medium occurred over time, with the development of silver chloride negatives, panchromatic film, and 35mm film. The concept for a digital camera was first introduced in 1969, with a first recorded attempt at building a digital camera occurring in 1975. The camera weighed eight pounds and recorded black and white images to a cassette tape. By 1984, Canon® had created the world's first commercial digital camera. This was quickly followed by the invention of the world's first megapixel sensor by Kodak® in 1986. Throughout the 1990s, the popularity of digital cameras rose dramatically, making digital cameras the most prevalent type of camera in the early 2000s. With the proliferation of cameras in modern smartphones, the on-going successes of photo-sharing websites such as Flickr™, and Instagram® demonstrate the popularity of photography in society to this day.

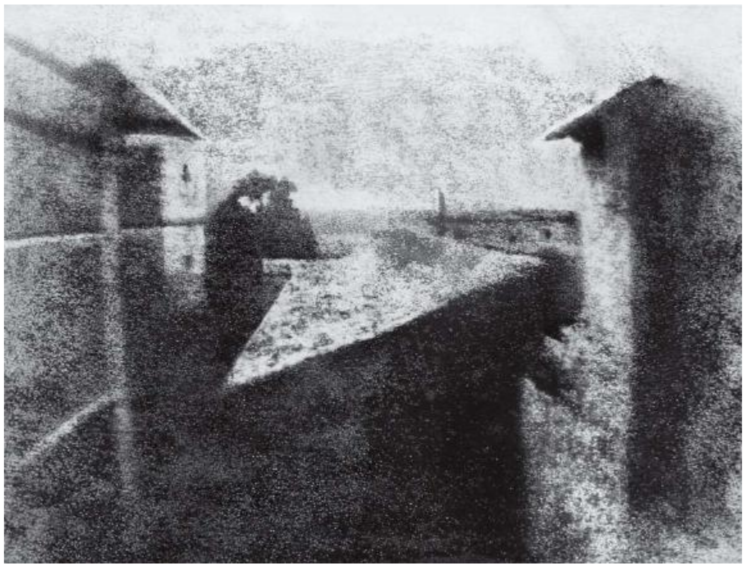


Figure 2.1 Niépce's photo (© Joseph Nicéphore Niépce, by permission)

2.2 History of Aerial Photography

Once a reliable technique for taking pictures was established during the 1800s, the concept of aerial photography was not far behind; however; the challenge came in finding an adequate aerial platform for taking aerial photographs. The only platforms available at the time were balloons and kites.

In 1858, Gaspard Felix Tournachon, also known as Nadar, captured the first recorded aerial photograph from a balloon tethered over the Bièvre Valley in France, though records of his initial work were apparently lost. A representation of his efforts was preserved in a drawing prepared by Honoré Daumier, for the May 25, 1862 issue of Le Boulevard (Fig. 2.2). In 1859, Nadar contacted the French military with a proposal to capture reconnaissance photos for the French Army's campaign in Italy. Maps for ground teams would then be produced from these aerial photographs. Nadar continued his efforts in aerial photography, and in 1868, he used a tethered balloon to take high-elevation photographs over the city of Paris.

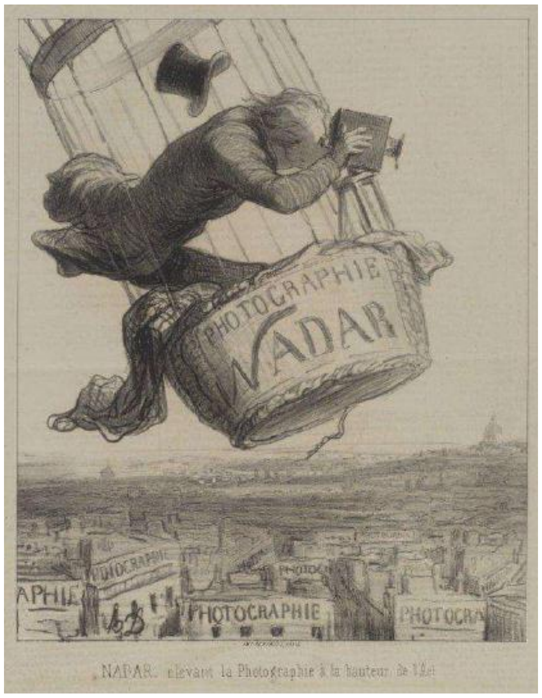


Figure 2.2 Nadar "elevating photography to the condition of art" caricature by Honoré Daumier (© Le Boulevard 25th May, 1862, by permission)

Another prominent figure in the history of aerial photography was James Wallace Black. On October 13, 1860, Black and a colleague ascended to an altitude of 365 metres (1200 feet) in a tethered balloon and photographed portions of the city of Boston. Black made eight exposures, though only one resulted in a reasonable picture; this is the oldest known conserved aerial photograph. Black experienced some difficulties in capturing photos from the balloon, which, although it was secured to the ground, was constantly moving due to winds. This movement made it difficult to obtain a good exposure, especially since photographic materials used at the time were rather slow. He used wet plates, and these had to be prepared in the balloon before each exposure. Later, the duo went up again with the idea of recording the surrounding countryside. However, problems quickly arose. As they ascended, the hydrogen expanded, and opened the neck of the balloon. This hydrogen gas flowed down on their equipment and rendered the photographic plates useless. Further adding to this ordeal, the balloon ended up landing in some bushes in Marshfield, Massachusetts, about fifty kilometres away from their original location. Though these early attempts were faced with several challenges, they helped pave the way for future and more stable aerial photography platforms to be considered.

In 1903, Julius Neubronner, designed and patented a breast-mounted aerial camera for carrier pigeons (Fig. 2.3). The camera weighed only 70 grams, and was equipped with a mechanism to capture automatic exposures at 30-second intervals. Although faster than balloons, the pigeons were not always reliable in following the desired flight path, and the wings from the birds frequently obscured the images. The 1909, the Dresden International Photographic Exhibition featured many pigeon photos, indicating an increase in popularity of aerial photography - especially for picture postcards. Pigeons continued to be used to capture photos for military surveillance during the early 20th century.

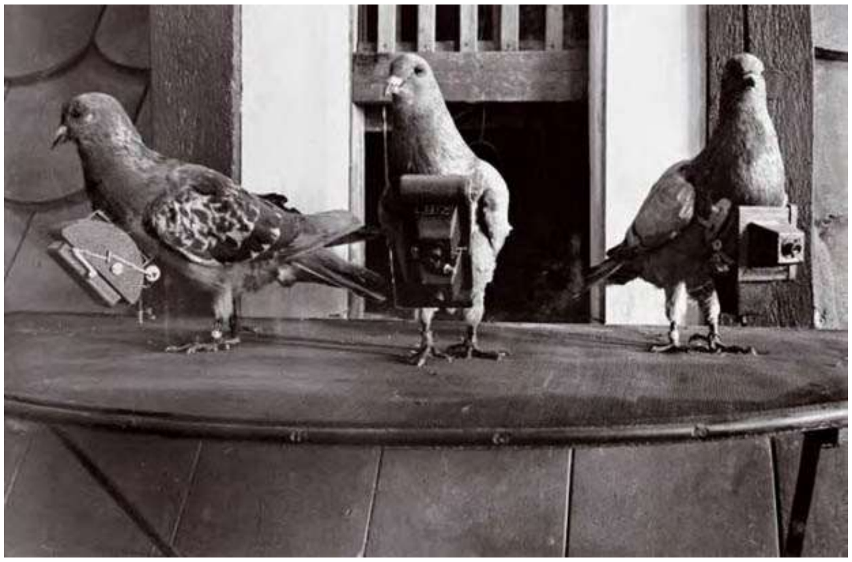


Figure 2.3 Carrier pigeons (© Deutsches Museum, München, by permission)

At the same time as Julius Neubronner patented his aerial camera for carrier pigeons, Alfred Maul, a German industrialist and engineer, developed a rocket system, known as the Maul Camera Rocket, to take aerial photographs. Maul was inspired by the work of Ludwig Rahrmann, who patented a method of attaching a camera to a large caliber artillery gun or rocket in 1891. Maul's camera was located in the nose section of the rocket with the lens looking through a hole in the nose cone. Wing stabilizers were attached to the base of the rocket to prevent the rocket from rotating and changing direction. When the flight reached its maximum altitude, a time-fuse opened the lens' shutter to capture a photo and deployed parachutes for gradual descent to the ground. The Maul Camera Rocket was successful in taking high-altitude photographs, including one famous photo of a North German landscape from over half a kilometre in altitude.

The military became especially interested in the practical uses of such equipment, and secret tests were conducted in August 1906 for military observers at a German firing range. Following the tests, Maul went on to develop a gyroscopic-stabilized plate camera system in 1907. This gyroscopic system, like modern-day camera gimbals, ensured a stable flight and thus produced sharper images. The upgraded rocket camera system was tested by the Austrian

Army for reconnaissance in the Turkish-Bulgarian War in 1912 and 1913; however, work on the system was discontinued after 1913 due to rapid advances in manned airplanes which could be used effectively for taking aerial photographs.

While it took some time for militaries to realize the full value of photography from airplanes, World War I served as a major contributor to advancing the field of aerial photography. Roughly sketched maps and exaggerated or misinterpreted observations were almost exclusively replaced with photographs by 1915. The English were the first to record enemy positions using cameras (Fig. 2.4), and found this method much easier and more accurate than sketching and observing. As a result, the aerial observer became the aerial photographer. In many cases, accurate maps, such as those used in the Battle of Neuve-Chapelle in 1915, were produced based on aerial photographs.

Soon, all nations involved in the war were using aerial photography. By the end of the war, both the Germans and the British were recording the entire front at least twice a day, providing up-to-date records of enemy trench

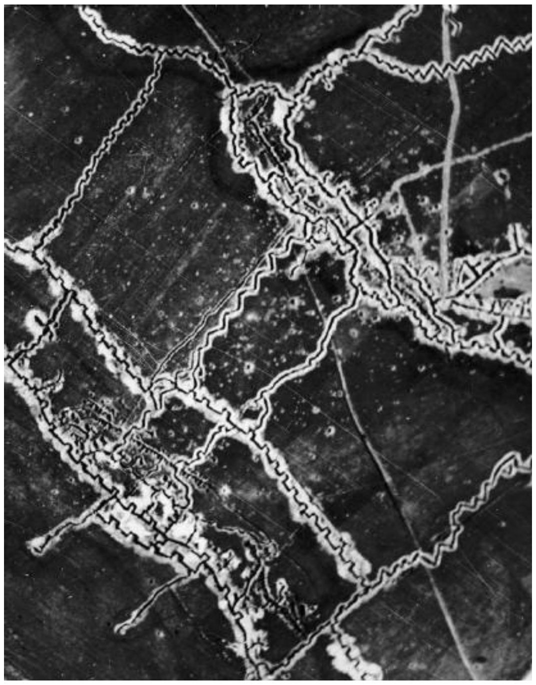


Figure 2.4 Vertical image representing trench locations (© No. 4 Squadron RFC, by permission)

construction. To put the utility of military aerial photography into perspective, it is estimated that the British reconnaissance planes took one-half million photographs during the war. On the German side, calculations showed that if all its aerial photographs were arranged side by side, there would be enough to cover the country six times! Not only did the sheer quantity of aerial photographs dramatically escalate, but the war fostered major improvements in the quality of cameras. Some photographs taken at 4,500 metre altitude could be enlarged to reveal features as small as footprints.

Most aerial photographs taken until this time were taken at an angle other than 90 degrees with respect to the earth's surface. Therefore, they are called oblique images. Photographs taken from a low angle are referred to as low oblique, while those taken from a high angle are called high (or steep) oblique. With the increased use of aerial cameras as observation and surveying tools during the war, increased emphasis was placed on developing specialized cameras for aerial photography. These cameras typically have precisely calibrated and documented properties, allowing them to be used to accurately measure targets on the ground.

Toward the end of the war, Sherman M. Fairchild developed a camera with the shutter located inside the lens which significantly reduced distortion problems caused by aircraft vibration. Fairchild also invented an intervalometer that could be used to capture photos at a desired interval. These developments made the Fairchild camera the best aerial camera system available. In fact, the Fairchild camera was so successful, that it remained the desired camera system for aerial photography for the next fifty years. Eventually, Fairchild entered the aviation industry to influence the way aircraft were designed to provide a solid aerial platform. For instance, open-cockpit biplanes were replaced with enclosed and heated cabins to protect the camera equipment. Fairchild's camera designs and commitment to aerial photography was instrumental in bringing the field to full maturity. Fairchild cameras were eventually carried on the NASA Apollo 15, 16, and 17 lunar missions in 1971 and 1972, and acquired high resolution lunar surface photos from the orbiting command module.

Aerial photography continued to evolve during the 1950s from work started during World War II and the Korean War. Colour-IR was used to distinguish between different vegetation types, and for detecting diseased and damaged vegetation. Multispectral imagery, which involves taking several images at different portions of the electromagnetic spectrum, was also being tested for different applications.

The rise of the cold war, with tensions between the United States and the Soviet Union during the 1950s, brought a heightened interest to aerial photography. The United States needed to know how many weapons and military bases the Soviet Union was operating and where they were located. Conventional military planes could not fly over the Soviet Union without being detected and shot down, and satellite technology had not yet fully matured. In the mid-1950s, Clarence Johnson from Lockheed's "Skunk Works" in Burbank, California, built the U-2 “spy plane” for the CIA. The U-2's cameras were used for five years, taking photos from over 70,000 feet of Intercontinental Ballistic Missile (ICBM) testing sites and air bases within the Soviet Union. The U-2’s usability was reduced with advances in satellite technology, though the U-2 continues to be used throughout the world for a variety of purposes.

As satellites with high resolution cameras and various multispectral sensors became more common, the term remote sensing was established to describe the process of investigating conditions on the Earth's surface using remotely collected data. Formally, the definition of remote sensing is: “the art and science of observing and measuring items on the Earth’s surface from a distance”. This includes aerial photography. The term “remote imaging” was first introduced by Evelyn Pruitt of the U.S. Office of Naval Research in 1960.

The field of aerial photography has come a long way since Nadar took the first aerial photograph in 1858. It is now commonplace for thousands of satellites and observation platforms, equipped with microcomputers and sensors, to be gathering data of the Earth’s surface from afar. In the last decade, the proliferation of affordable UAV’s has continued to evolve the field of remote sensing, making it possible to readily acquire aerial imagery for a myriad of research and commercial applications.

2.3 Colour Theory

This section provides some background information on colour theory as it relates to the electromagnetic spectrum. An explanation of common colour models and colour spaces is also provided.

2.3.1 Electromagnetic spectrum

In support of a discussion on the applications of electro-optical sensors and computer vision, a brief overview on the physical properties of light is required. Electromagnetic radiation, comprising the electromagnetic spectrum, describes one way in which energy propagates through a medium. The electromagnetic spectrum is determined based on the frequencies of light and their corresponding wavelengths. Higher energy waves, such as X-ray waves, propagate with shorter wavelengths and higher frequencies, while lower energy waves, such as IR waves, propagate with longer wavelengths and lower frequencies. The relationship between wave speed (c), wavelength (λ), and frequency (f) is provided by the following equation:

$$c = f \times \lambda$$

Similarly, the relationship between energy and frequency is provided by the following equation, theorized by Max Planck around 1900 and published in his book "The theory of heat radiation" [44]. (Note that the h indicates Planck's Constant of 6.6×10^{-34} J/s):

$$E = h \times f$$

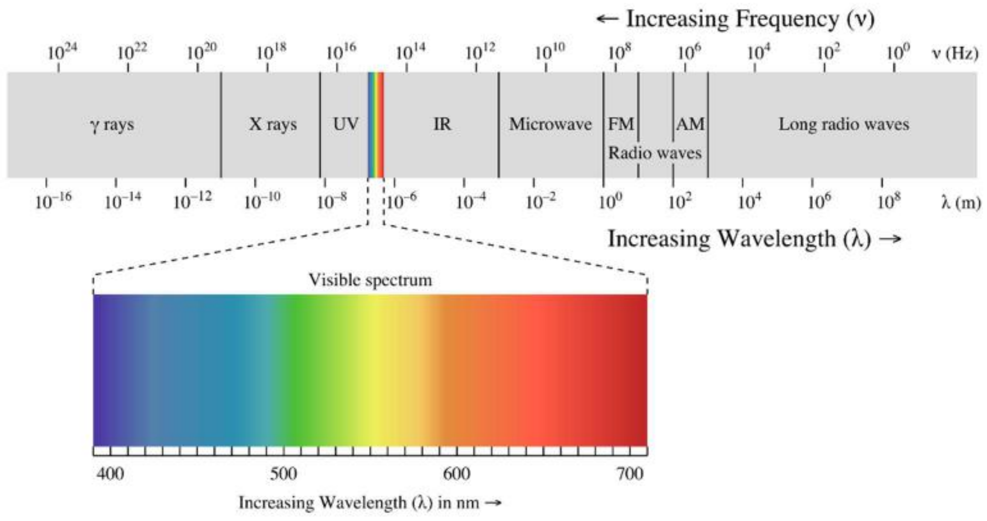

← Increasing Frequency (ν)
10^{24} 10^{22} 10^{20} 10^{18} 10^{16} 10^{14} 10^{12} 10^{10} 10^{8} 10^{6} 10^{4} 10^{2} 10^{0} ν (Hz)
γ rays
X rays
UV
IR
Microwave
FM
AM
Radio waves
Long radio waves
10^{-16} 10^{-14} 10^{-12} 10^{-10} 10^{-8} 10^{-6} 10^{-4} 10^{-2} 10^{0} 10^{2} 10^{4} 10^{6} 10^{8} λ (m)
Increasing Wavelength (λ) →
Visible spectrum
400
500
600
700
Increasing Wavelength (λ) in nm →

Figure 2.5 Diagram of the electromagnetic spectrum (© 2018 Philip Ronan, by permission)

2.3.2 Visible Light

The visible spectrum of light, perceivable by the human eye, ranges between 400 – 700 nm. The brain interprets these wavelengths as different colours. As illustrated in Fig. 2.5, the visible spectrum comprises only a relatively small section of the electromagnetic spectrum. Colour cameras used in many computer vision applications typically function in this band of the electromagnetic spectrum; many of the frequencies employed in this study fall beyond the visible range.

2.3.3 Near Infrared

The near IR (NIR) band is located most closely to the red portion of the visible light spectrum [45]. It is also commonly referred to as the shortwave IR (SWIR) band. It's wavelength ranges from 0.75 - 1.4 µm. This corresponds to a frequency of 214 - 400 THz. The IR-based methods described in this thesis use the NIR frequency band.

2.3.4 Long-wave Infrared

Long-wave IR (LWIR) is known as the thermal imaging region of the electromagnetic spectrum. LWIR has a wavelength of 8 to 50 μm, which corresponds to a frequency of 2,237 THz. Sensors operating in this region can obtain a completely passive image of objects by using a temperature gradient [46]. Therefore, it is also known as the thermal IR region. Although the author conducted some preliminary studies with LWIR cameras, such as the FLIR Lepton (Fig. 2.6), these will not be discussed in this work. Nevertheless, this frequency band is well-known and often confused with the NIR band described in this work.

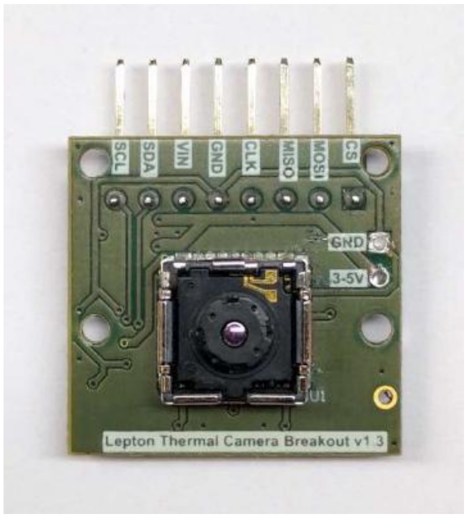


Figure 2.6 FLIR Lepton

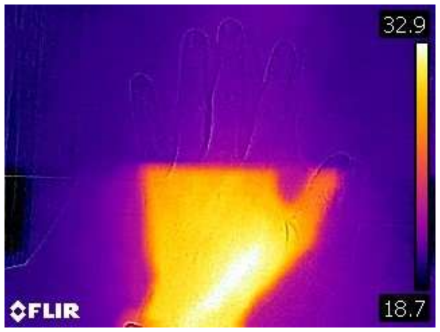


Figure 2.7 Thermal image

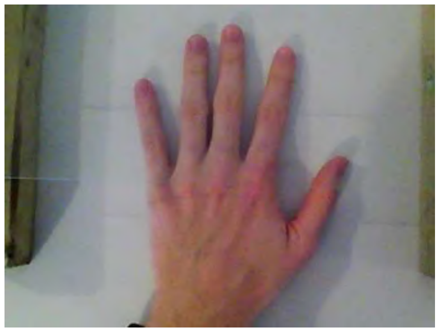


Figure 2.8 Colour image

Note that using LWIR, glass appears opaque (Fig. 2.7), the opposite of a regular colour camera (Fig. 2.8).

2.3.5 Colour models

Colour models describe a system for creating a range of colours by using only a small number of primary colours. The RGB (Red, Green, and Blue) colour model is the most common colour model, and is used for displaying electronic images through projections, television screens, and computer monitors.

The RGB colour model is an additive model, meaning that colour mixing begins with black and ends with white. The mixing of RGB in equal proportions produces white light. Cyan, magenta, and yellow are subtractive colours; the CMYK colour model uses these colours, producing black when CMY are mixed in equal proportions. The "K" in CMYK denotes the "Key" colour, which most often is black, and is important in printing applications.

2.3.5.1 RGB

RGB is an additive colour model, in which red, green, and blue light are added together to reproduce a broad array of colours. The red, green, and blue colour beams are called components, and are separated into three corresponding image channels. Zero intensity for each component gives the darkest colour (no light, considered to be black), and full intensity of each results in a white. Modern electronic devices, such as CRT, LCD, and OLED displays use the RGB colour model to represent colours. Photographic digital cameras that use a CMOS or CCD image sensor often operate with some variation of the RGB model. A colour in the RGB colour model is described by indicating how much of each of the red, green, and blue is included. The colour is expressed as an RGB triplet (r, g, b), each component of which can vary from zero to a defined maximum value. If all colour components are zero, the resulting output is black; if all components are at the maximum value, the result is the brightest representable white.

2.3.5.2 HSL and HSV

Two alternative colour models to RGB were developed in the 1970s by computer graphics researchers. These new models are the HSL (Hue, Saturation, Lightness) and HSV (Hue, Saturation, Value) representations, which mimic human colour perception. In these models, colours are arranged in a radial slice around a central axis of neutral colours. Each model has its specific advantages, with HSV (Fig. 2.9) being more accurate at depicting colour mixing, whereas the HSL model resembles perceptual colour models. Both are cylindrical geometries, with hue (their angular dimension) starting at the red primary at 0°, passing through the green primary at 120° and the blue primary at 240°, and then wrapping back to red at 360°. The central vertical axis comprises the neutral, achromatic, or gray colours, ranging from black (value 0) at the bottom, to white (value 1), at the top of each geometry.

HSL, HSV and other related models are often used in computer vision applications due to their separation of colour components from intensity. This provides an advantage to the RGB model, because the Red, Green, and Blue components of an object’s colour in a digital image are all correlated with the amount of light hitting the object, and therefore with each other. Therefore, image descriptions in terms of those components makes object discrimination difficult. In situations where lighting or shadows may cause intensity changes, description in terms of HSV or HSL is often more relevant and improves robustness in image analysis for feature detection or image segmentation.

Since computer vision algorithms for colour images are generally straightforward extensions to algorithms designed for grayscale images, it is important that the features of interest can be distinguished in the colour dimensions used. As the field of computer graphics gained popularity in the late 1970s, transformations such as HSV were developed as a compromise between effectiveness for segmentation and computational complexity. These transformations are similar in approach and intent to neural processing used by human colour vision: roughly separating hue, saturation, and lightness is effective for object detection. These models have continued to see wide use in recent years, as their performance compares favorably with more complex models, while maintaining computational simplicity.

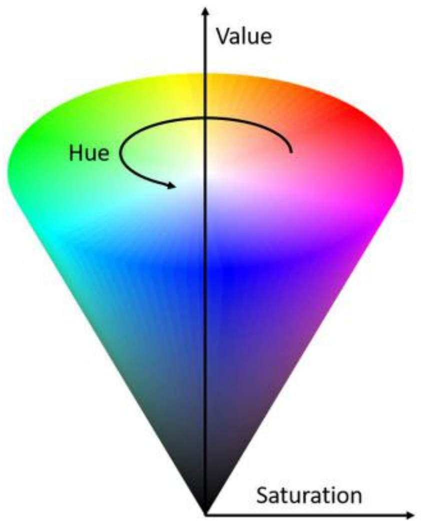


Figure 2.9 Conical representation of the HSV colour model

2.3.6 Colour Space

A colour space is used to describe the physical representation of a colour model. It is the set of colours which can be displayed or reproduced in a medium. For example, sRGB is a particular set of red, green and blue intensities, which defines the colours that can be reproduced by mixing those ranges of red, green and blue. In other words, a colour space is used to map real colours to the discrete colour values obtained from a colour model. Therefore, it is important to select a specific colour model before discussing colour spaces.

2.4 Introduction to Computer Vision

In this section, a brief introduction and background to the field of computer vision is provided. From a broad perspective, the discipline of computer vision can be divided into two categories:

- Biological: computational models of the human visual system are developed for research purposes. These studies provide insight into psychology and biological sciences, and medical fields such as ophthalmology.

- Engineering: machine vision is being integrated with robotic technology to assist in performing autonomous tasks that would otherwise only be possible using the human visual system.

These two fields of study are closely related; a good understanding of the biological vision system is necessary to mimic such a system using specialized sensors and computers. The human vision system provides inspiration to engineers designing computer vision systems. Conversely, a strong understanding of computer vision algorithms offers insights into how the human visual system works. This thesis presents computer vision from the engineering point of view.

It is commonly accepted that the father of computer vision is Dr. Lawrence Roberts from MIT, who discussed the possibilities of extracting 3D geometrical information from 2D perspective views of blocks in 1963 [47]. Many researchers followed his work and studied computer vision in the context of similar 2D blocks, though it was eventually realized that it was necessary to tackle images from the real world. As a result, research expanded for computers to delineate image data in the form of low-level processing. Unlike high-level processing, which was developed much later, low-level processing enabled computers to perform basic tasks such as edge detection and colour differentiation.

David Marr, circa 1978 at MIT, further contributed to the computer vision framework by taking a bottom-up approach to scene understanding. The bottom-up approach uses individual stimuli or detections, such as generic features and lines to explain a larger, more complex, feature. For this method, low-level image processing algorithms are applied to 2D images to obtain a primary sketch of edge segments. Finally, high-level techniques are used to produce a virtual 3D model of the real-world object.

Despite Marr's significant contribution to the field of computer vision, several researchers have identified limitations to his approach, and advocate for a top-down approach. The top-down approach receives the entire stimulus of an object, as opposed to basic features. In addition, Marr's program is extremely difficult to carry out, and it is usually not necessary to obtain complete 3D object models. For example, autonomous vehicles employing computer vision for navigation assistance, may only need to distinguish whether an object is moving away from or towards the vehicle, but not the exact 3D composition and motion of the object.

Use of computer vision has become increasingly goal-driven as a broad spectrum of applications and capabilities have emerged. As shown in history, it is important to note that the broad spectrum of potential applications has resulted in the field of computer vision to merge with other closely related fields. For instance, the field of image processing, where raw images must be processed before further analysis by applying various filters, colour-correction measures, and enhancing layers. Likewise, photogrammetry, or the science used to obtain quantitative information from photographs requires special camera calibration techniques.

Computer vision is a notoriously difficult field, mainly because the human visual and nervous system is so well designed for tasks such as facial recognition; computer vision systems suffer by comparison. Humans can recognize faces under variations in illumination, perspective, emotional expression, along with no apparent limit on the number of faces we can store in our brain for future recall. In short, it seems unlikely that an autonomous system will ever be constructed with such remarkable performance as the human body. Two major challenges in computer vision which have yet to be solved include:

- Representing the enormous amount of human knowledge with a computer using a method through which retrieval is easy.
- Using hardware and software to conduct complex computations, such as facial recognition, in real-time.

2.5 Navigation

Navigation is a field of study that focuses on the process of monitoring and controlling the movement of a craft or vehicle from one place to another [48].

The field of navigation includes four general categories [49]:

- land navigation
- marine navigation
- aeronautic navigation
- space navigation

This paper focusses on aeronautic navigation.

During landing, most large UAVs are controlled from the ground by experienced pilots. This method, however, presents several challenges including a lack of situational awareness for the pilots. There is a lack of realism as they cannot hear the engine, feel any vibrations, acceleration, or motion [21]. This lack of realism often results in a computer game-like feeling, and as a result many UAVs are lost due to crashes during landing. Automatic landing systems offer increased accuracy in measuring aircraft position compared to human operators. They also provide other benefits such as around-the-clock operation, no matter the weather conditions. The landing phase of a UAV flight is currently the most critical element. Existing methods land a fixed-wing UAV onto a net or on a runway, while VTOL aircraft can land on a helicopter pad. Most of these vision-based landing systems rely on a clear line of sight to the landing zone. In vision compromised or GPS denied environments, an alternative solution is required. This alternative solution must also work without a communication link between the UAV and the landing zone. The methods presented in this paper have therefore eliminated the need for a wireless communication link.

2.6 Computer Vision Pipeline

This section of the thesis provides a background to the various stages involved in the computer vision pipeline. Establishing an understanding of these topics and their relation to computer vision applications is critical to designing a computer vision image processing pipeline. All following concepts are applied in the implementations described in Chapter 3 and Chapter 4.

2.6.1 **Image Acquisition / Capture**

Image acquisition is the first step in any computer vision pipeline, and refers to the process of acquiring an image from the camera's imaging sensor. An imaging sensor detects and conveys information that is used to construct an image by converting the variable attenuation of light waves into small and specific levels of current. The waves can be visible light or other forms of electromagnetic radiation - such as ultraviolet or IR light. Image sensors are used in electronic imaging devices including digital cameras, medical imaging equipment, night vision apparatus, thermal imaging devices, and others.

An understanding of manual camera settings and photography is essential to the field of computer vision and image processing. Here follows a brief introduction to exposure, aperture, shutter speed, ISO and white balance.

2.6.1.1 Exposure Triangle

The exposure triangle (Fig. 2.10) is a method of visualizing the three key variables which determine the exposure of a photograph: aperture, shutter speed, and ISO. Each of these aspects relates to light and how it enters and interacts with the camera, and are further discussed below. A change in any one of the three variables will influence the others.

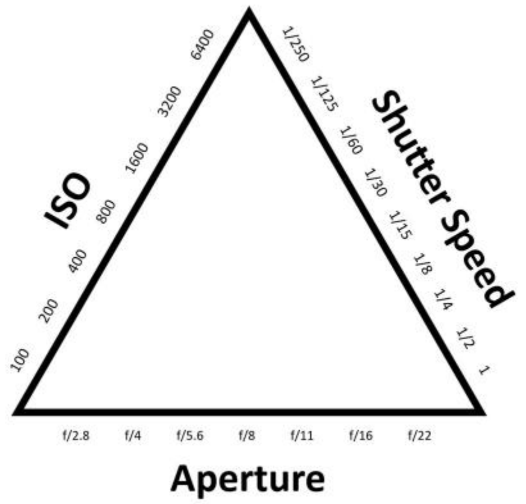


Figure 2.10 The exposure triangle

2.6.1.2 Aperture

A camera's aperture setting controls the amount of light entering the camera lens, and therefore hitting the sensor. A camera's aperture size, or diameter, is controlled by the diaphragm, also known as the iris. The diaphragm attenuates or restricts all light except for that which passes through. In photography, aperture values are expressed as f-stop values. A large f-stop value indicates a small aperture size, limiting light exposure to the image sensor, while a camera's smallest f-stop value indicates a wide-open aperture, and permits much light to reach the image sensor.

2.6.1.3 Shutter Speed

Shutter speed dictates the length of time that the shutter on a camera is open. It is measured in seconds or fractions of seconds. A quick shutter speed exposes the imaging sensor to light for a short amount of time, whereas a long shutter speed exposes the imaging sensor to light for a long amount of time. A long shutter speed will result in blurred images if the image sensor experiences motion, whereas a short shutter speed captures an almost instantaneous image - thereby reducing any blurred images even when experiencing motion. For example, helicopter rotor blades will appear as a blurry circle at a shutter speed of 1/60, but will appear solid at a shutter speed of 1/500.

2.6.1.4 ISO

ISO is an acronym for the International Standards Organization, and is a standardized industry scale for measuring sensitivity to light, in addition to other analytical measures. In digital cameras, the ISO pertains to the sensitivity of a digital image sensor. ISO is measured in integer numbers, with a low number (e.g. ISO 50) being least sensitive to light, and the highest number being most sensitive to light (e.g. ISO 6400). Doubling the ISO value doubles the amount of light reaching the sensor. Since ISO can be thought of as digital gain, a large ISO value introduces digital noise in an image, therefore it is preferable to shoot at the lowest possible ISO value.

ISO numbers usually start from 100-200 (base ISO) and increase by a multiple of two. For instance, the ISO sequence is: 100, 200, 400, 800, 1600, 3200, 6400, where each step between the ISO numbers effectively doubles the sensitivity of the sensor.

2.6.1.5 White Balance

Another crucial camera setting to understand within the context of computer vision applications is white balance. White balance settings can be used to counteract changes in environmental lighting conditions which affect the colour of an object. For example, a white sheet of paper will appear to be a different colour when viewed in sunlight, under overcast skies or indoors under incandescent or fluorescent light. The white balance setting on digital cameras can be used to compensate for this effect.

Cameras are often equipped with an auto white balance (AWB) mode, which reads the scene's colour temperature and chooses a pre-programmed colour adjustment. The colour temperature is measured based on the hue and intensity of various light sources, and is measured in degrees Kelvin. This method benefits most photography applications; however, being unaware of this setting can cause problems for computer vision applications.

A computer vision application installed in a permanent location with minimal changes in lighting must have the AWB mode disabled. Otherwise, the white balance setting may change sporadically throughout the device's deployment, creating seemingly unknown colour differences from frame to frame. These colour differences can cause problems with thresholding applications, resulting in the same object being recognized as being within the threshold in one frame, and outside the threshold in another. Thresholding is typically affected by lighting conditions, and various algorithms to counteract these problems are outlined in the following section. Nevertheless, capturing quality images is the key to any further stage of the image processing pipeline.

2.6.2 Image pre-processing

After acquisition, image pre-processing is the next step in the computer vision pipeline. It is an important step, which prepares the image to be fed into the computer vision pipeline. The goal of pre-processing is to clean up the image, preparing it to produce dramatic positive effects downstream in the vision pipeline. At its most basic level, the pre-processing stage can include cropping, resizing, and selecting regions of interest in the image. More advanced pre-processing techniques include denoising, colour enhancement, dynamic range expansion, artifact removal, and image stabilization.

2.6.3 Thresholding (Simple Segmentation)

A primary objective of image processing is the demarcation of objects in digital images. This is done using the process of image segmentation [50]. In its most basic form, segmentation separates light and dark regions in the image, thereby identifying dark objects on a light background or vice versa. An idealization of the algorithm proposes that the image is split into regions, with each region having a high level of uniformity in some parameter such as brightness, colour, texture or even motion.

The simplest image segmentation method is known as thresholding. This method is often used to extract foreground, background, or object features in an image by segmenting at specific intensity levels. Thresholding is a difficult process, and therefore often requires additional pre-processing methods to be successful. A variety of thresholding methods have also been developed, including global to locally adaptive thresholding. When employed in a computer vision pipeline, thresholding is usually highly tuned and application-specific.

Although the concept of segmentation is often reasonably accurate, it is an invention of the human mind, generalized from certain simple cases. The human eye can understand scenes at a glance, which provides context to perceiving objects in their known form. Computer vision systems lack this ability, which makes segmentation one of the central and most difficult practical problems of machine vision.

The major problem for the thresholding algorithm is nonuniform illumination, so special attention must be given to ensure environmental lighting conditions remain constant and application-specific light is uniformly placed. Many images do not respond well to global thresholding involving simple methods, and local methods are often required. These use local pixel structure and statistical relationships to create effective thresholds.

A threshold can take several forms:

- Floor: the lowest pixel intensity allowed
- Ceiling: the highest pixel intensity allowed
- Ramp: shape of the pixel ramp between floor and ceiling, such as linear or logarithmic
- Point: may be a binary threshold point with no floor, ceiling, or ramp

The `threshold` function of the OpenCV library was used to perform thresholding operations for the experiments outlined in this thesis.

2.6.4 Binarization

Binarization is the process of converting a pixel image with many different bit values to a binary image. After application of a threshold, each pixel value becomes a binary value of "*0*" or "*1*". It is the simplest form of thresholding or quantization, and is essentially a quantization function with $n = 2$. It is also the basis of segmentation.

Binarization was accomplished in this experimental research using the `threshold` function of the OpenCV library.

2.6.5 Masking

Masking is the process by which a binary image is multiplied with an original image. In many cases, the binary image is the result of a thresholding operation. This results in a colourized image with only the *"true"* values remaining. This is often used for visualization purposes.

The work presented in this thesis uses masking functions included in the OpenCV and NumPy libraries. OpenCV includes a bitwise `AND` function which can be used for masking operations. The NumPy package for scientific computing using Python provides arithmetic operations for matrices, and provides the same result.

2.6.6 **Segmentation**

Segmentation partitions an image into regions with distinct properties, such as a group of pixels with similar attributes. Segmentation is often employed to transform a greyscale or colour image into multiple new images to perform low-level image processing before proceeding to high-level image description in terms of colour, features, objects, and scenes. While image segmentation is an important procedure, creating an accurate partition of an image is quite difficult to achieve.

Segmentation techniques can be either contextual or non-contextual. As the name implies, the contextual technique accounts for various spatial relationships between features in the image. Conversely, non-contextual segmentation takes no account of spatial relationships between features in an image, and instead groups the pixels together based on a global attribute such as colour. This thesis research focuses on non-contextual thresholding, more specifically, colour thresholding.

Segmentation on colour images is more accurate when compared to greyscale images, because more information is available at the pixel level. Colour models, such as HSV and HSL, have been designed to exclude redundancy, can determine actual object or background colours irrespectively of illumination, and obtain more stable segmentation. Segmentation of colour images involves a partitioning of the colour space, as achieved when the segmentation is based on a reference colour $(R0, G0, B0)$ and thresholding using Cartesian distances to it from every surrounding pixel colour $f(x,y) = (R(x,y), G(x,y), B(x,y))$:

$$g(x,y) = \begin{cases} 1 \; if \; d(x,y) \leq d_{max} \\ 0 \; if \; d(x,y) > d_{max} \end{cases};$$

$$d(x,y) = \sqrt{(R(x,y) - R_0)^2 + (G(x,y) - G_0)^2 + (B(x,y) - B_0)^2}$$

where $g(x,y)$ is the binary region map after thresholding. The thresholding defines a spatial zone which is centered on the reference colour. Pixels inside this zone belong to the region, and are denoted with a "*1*", while all other outlying pixels are denoted with a "*0*".

2.6.7 Feature Detection

A feature in image processing refers to a point of interest useful for image description. Feature detection is performed using various methods which compute abstractions of image information and make decisions whether there is an image feature of a given type at that point or not. The resulting features will be subsets of the larger image domain, often in the form of isolated points, curves, or outlines. Common types of image features include edges, corners, blobs (regions of interest points) and ridges.

The research presented in this thesis uses edge detection and blob detection.

2.6.7.1 Canny Edge Detection

The goal of an edge detector is to enhance the connected gradients in an image, which may take the form of an edge, contour, line, or some connected set of edges. Many edge detectors are simply implemented as kernel operations, or convolutions, and an overview of the Canny method [51] is provided here. Canny Edge Detection takes on a multi-stage approach:

1. Filter: Perform a Gaussian blur to filter out noise in the image. A variety of convolution kernels (7×7, 5×5) can be used, depending on the level of low-pass filtering required.
2. Find Intensity Gradient: Apply two directional Sobel filters and find gradient strength and direction.
    a. Convolution masks (Sobel filters) applied in x- and y-axis:

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix}$$

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix}$$

b. Determine gradient strength and direction:

$$G = \sqrt{G_x^2 + G_y^2}$$

$$\theta = \tan^{-1}\left(\frac{G_y}{G_x}\right)$$

3. Apply non-maximal value suppression. This removes non-edge pixels, resulting in candidate edges remaining as thin lines.
4. Perform hysteresis thresholding along the gradient using Canny with an upper and lower threshold. This eliminates edge aliasing and outlier artifacts and results in more distinct connected edges.

```
a. If pixel gradient value > upper threshold then
       accept pixel as edge;
b. If pixel gradient value < lower threshold then
       reject pixel;
c. If (lower threshold < pixel gradient value > upper threshold)
   AND (connected to another pixel > upper threshold) then
       accept pixel as edge;
```

The OpenCV library was used to provide Canny Edge Detection, which is implemented in the `Canny` function. The function searches for edges in the input image (`image`) and marks them in the output map (`edges`) using the Canny algorithm.

2.6.7.2 Hough Line Transform

The Hough Line Transform is used to detect lines in an image [52]. It is a popular technique capable of detecting any shape representable in a mathematical form. It is especially useful since it can detect shapes even if they are broken or slightly distorted. The algorithm for detecting a line is presented here.

A line (y) in the image space can be represented using two variables:

a. In the Cartesian coordinate system, using parameters (m, b):

$$y = mx + b$$

b. In the Polar coordinate system (Fig. 2.11), using parameters (r, θ):

$$r = x \cos\theta + y \sin\theta$$

where r is the perpendicular distance from the origin to the line, and θ is the angle between the perpendicular line and horizontal axis measured in counter-clockwise degrees.

For Hough Transforms, lines are expressed in the Polar coordinate system (Fig. 2.12), therefore the line equation is written as:

$$y = \left(-\frac{\cos\theta}{\sin\theta}\right) x + \left(\frac{r}{\sin\theta}\right)$$

$$r = x \cos\theta + y \sin\theta$$

A line located below the origin will have a positive r-value and $\theta < 180°$. A line located above the origin will have a negative r-value and $\theta < 180°$. Vertical lines have a θ-value of 0° and horizontal lines have a θ-value of 90°.

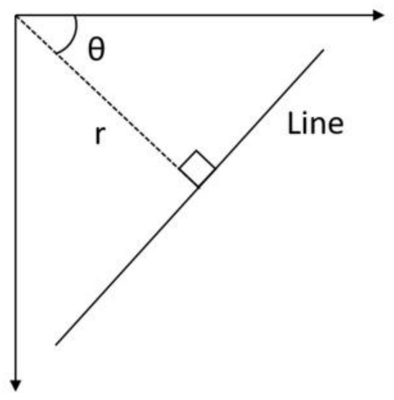


Figure 2.11 Hough Line in polar coordinate system

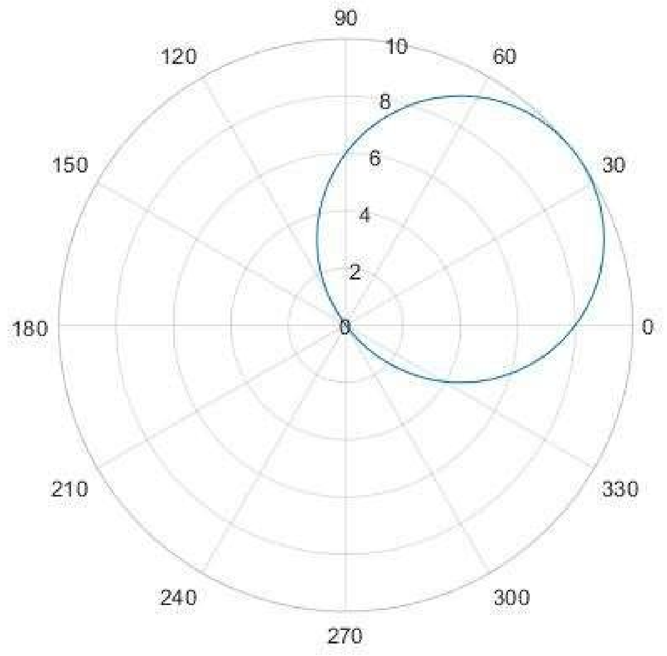


Figure 2.12 Polar plot with x₀ = 8 and y₀ = 6

For any point (x_0, y_0), the family of lines which intersect that point can be defined as:

$$r_\theta = x_0 \cdot \cos\theta + y_0 \cdot \sin\theta$$

Each pair (r_0, θ) represents a line passing (x_o, y_0). If for any given (x_0, y_0) the family of lines passing through it is plotted; the result is a sinusoid. Only points where $r > 0$ and $0 < \theta < 2\pi$ are considered. The same operation can be repeated for all points in an image. If the curves of two different points intersect in the plane $\theta - r$, both points belong to the same line.

In Fig. 2.13, the three plots can be seen intersecting at a single point $(0.925, 9.6)$, these coordinates are the parameters (θ, r), and represent the line on which (x_0, y_0), (x_1, y_1), and (x_2, y_2) lie.

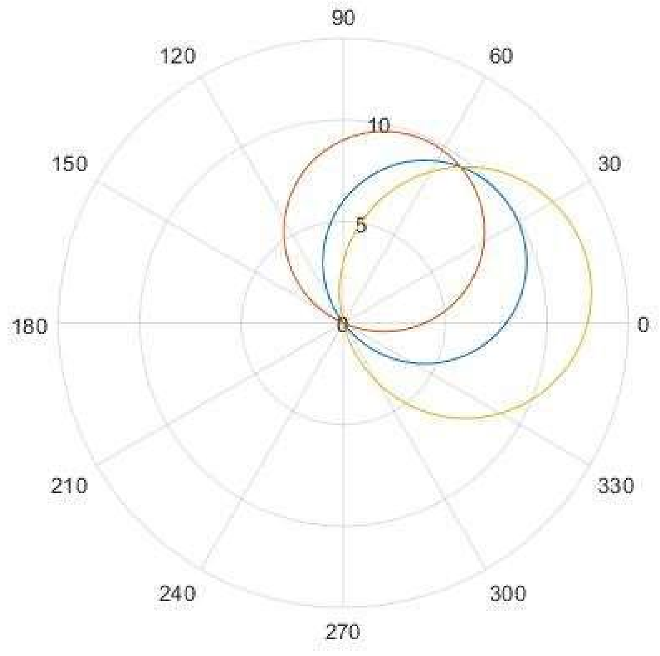


Figure 2.13 Polar plot with $x_1 = 4$, $y_1 = 9$, $x_2 = 12$, $y_2 = 3$. The plots intersect at (0.925, 0.6)

The OpenCV library was used to provide the Hough Transform required in this project. The Standard Hough Transform behaves similarly to the algorithm explained above. It returns a result of a vector of couples (θ, r_θ). In OpenCV it is implemented with the function `HoughLines`. The Probabilistic Hough Line Transform is a more efficient implementation of the Hough Line Transform, and provides an output of the extremes of the detected lines $(x_0, y_0, x_1 y_1)$. In OpenCV it is implemented with the function `HoughLinesP`.

2.7 Aerial Computer Vision Concepts

The following sections outline some computer vision concepts specific to aerial imaging applications. These include an introduction to pixel and image coordinate systems and image- to world-space mapping using external and internal calibration matrices.

2.7.1 Coordinate Systems

An understanding of image projection and coordinate frames is critical to computer vision projects when working with 3D scenes and image planes. Consider the three coordinate frames:

- World Coordinate Frame ($\vec{X}_w$): These 3D coordinates are fixed in the world.
- Camera Coordinate Frame ($\vec{X}_C$): These 3D coordinates are fixed in the camera. The origin of the camera coordinates is at the center of projection of the camera (say at $\vec{d}_w$ in world coords). The z-axis is taken to be the optical axis of the camera (with points in front of the camera in the positive z direction).
- Image Coordinate Frame ($\vec{p}$): The image coordinates are written as a 3-vector, $\vec{p} = (p_1, p_2, 1)^T$, with p_1 and p_2 the pixel coordinates of the image point. Here, the origin is in the top-left corner of the image. The first image coordinate p_1 increases to the right, and p_2 increases downwards.

Many computer vision applications use the camera coordinate system, with the coordinates $(0, 0)$ centered in the middle of the x- and y-axes of the image. The OpenCV library, however uses the image coordinate system, placing $(0, 0)$ at the top-left corner of the image. Therefore, a brief explanation of these two coordinate systems is provided in Fig. 2.14. The image coordinate system has the benefit of only providing positive coordinate values.

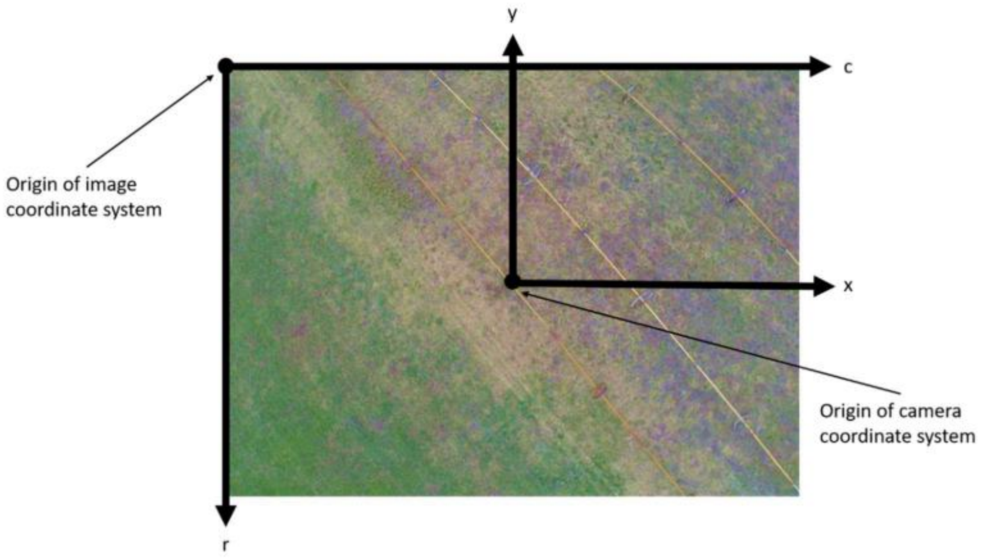

y
c
Origin of image coordinate system
x
Origin of camera coordinate system
r

Figure 2.14 Camera coordinate system and image coordinate system

Next, the transforms from world coordinates to camera coordinates and then to image coordinates (Fig. 2.15) are introduced.

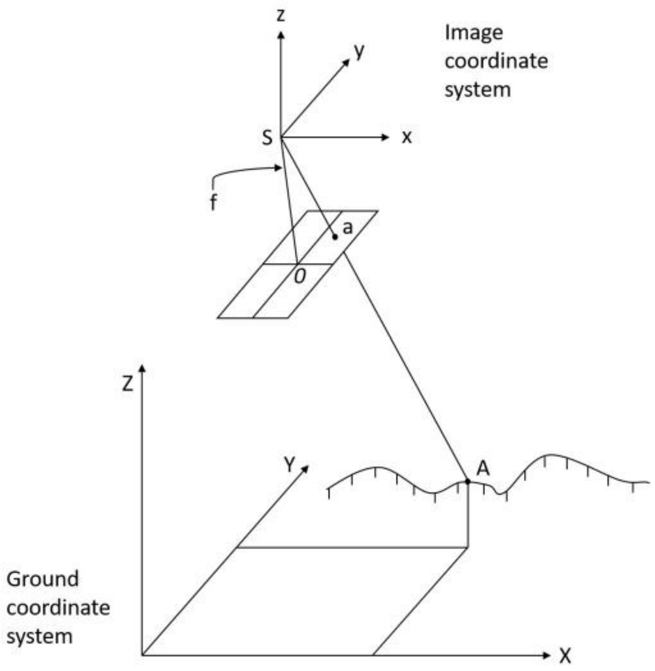


Figure 2.15 World-space to image-space mapping diagram

2.7.2 Extrinsic Calibration Matrix

The extrinsic calibration parameters specify the transformation from world to camera coordinates, which is a standard 3D coordinate transformation,

$$\vec{X}_c = M_{ex}\left[\vec{X}_w^T, 1\right]^T. \tag{1}$$

The extrinsic calibration matrix M_{ex} is a 3×4 matrix of the form

$$M_{ex} = \begin{pmatrix} R & -R\vec{d}_w \end{pmatrix}, \tag{2}$$

where R is a 3×3 rotation matrix and $\vec{d}_w$ is the location, in world coordinates, of the center of projection of the camera. The inverse of this mapping is

$$\vec{X}_w = R^T\vec{X}_C + \vec{d}_w. \tag{3}$$

The perspective transformation can be applied to the 3D point $\vec{X}_c$ (i.e., in the camera's coordinates),

$$\vec{x}_c = \frac{f}{X_{3,c}}\vec{X}_c = \begin{pmatrix} x_{1,c} \\ x_{2,c} \\ f \end{pmatrix}. \tag{4}$$

Values used in these equations are measured in distance (e.g. metres), not pixels, and *f* is the camera's focal length.

2.7.3 Intrinsic Calibration Matrix

The intrinsic calibration matrix, M_{in}, transforms the 3D image position $\vec{x}_c$ (measured in metres, for example) to pixel coordinates,

$$\vec{p} = \frac{1}{f}M_{in}\vec{x}_c, \tag{5}$$

where M_{in} is a 3×3 matrix. The factor of $1/f$ here is conventional.

For example, a camera with rectangular pixels of size $1/s_x$ by $1/s_y$, with focal length *f*, and piercing point (o_x, o_y) (i.e., the intersection of the optical axis with the image plane provided in pixel coordinates) has the intrinsic calibration matrix

$$M_{in} = \begin{pmatrix} fs_x & 0 & o_x \\ 0 & fs_y & o_y \\ 0 & 0 & 1 \end{pmatrix}. \tag{6}$$

Note that, for a 3D point $\vec{x}_c$ on the image plane, the third coordinate of the pixel coordinate vector $\vec{p}$ is $p_3 = 1$.

Equations (1), (4) and (5) define the transformation from the world coordinates of a 3D point, $\vec{X}_w$, to the pixel coordinates of the image of that point, $\vec{p}$. The transformation is nonlinear, due to the scaling by $X_{3,c}$ in equation (4).

2.7.4 Pinhole Camera Concept

In this work, the Raspberry Pi camera mounted aboard the UAV is modelled using the pinhole camera model with the camera's optical center located at the origin of B (the body reference frame). The inertial and body reference frames are used when explaining camera mounting configurations (Fig. 2.16).

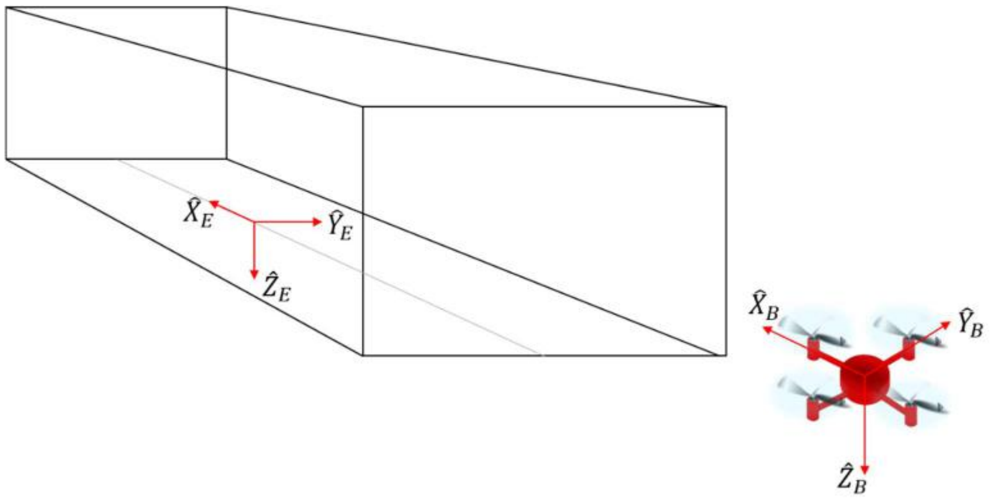


Figure 2.16 Inertial (earth) and body frames of reference

For Chapter 3, the camera is placed in a nadir (downwards-facing) position, such that the optical axis coincides with $\hat{Z}_B$. The image plane of the camera is defined such that $\hat{X}_C$ (image plane horizontal axis) coincides with $\hat{Y}_B$, and $\hat{Y}_C$ (image plane vertical axis) coincides with $\hat{X}_B$. According to the pinhole camera model, the image of any point $[X_B\ Y_B\ Z_B]^T$ expressed in B is:

$$x_i = f\frac{Y_B}{Z_B}$$

$$y_i = f\frac{X_B}{Z_B}$$

where f is the focal length and x_i and y_i are the image point coordinates.

For Chapter 4, the camera is placed in a forward-facing position, such that the optical axis coincides with $\hat{X}_B$. The image plane of the camera is defined such that $\hat{X}_C$ (image plane horizontal axis) coincides with $\hat{Y}_B$, and $\hat{Y}_C$ (image plane vertical axis) coincides with $-\hat{Z}_B$. According to the pinhole camera model, the image of any point $[X_B\ Y_B\ Z_B]^T$ expressed in B is:

$$x_i = f\frac{Y_B}{X_B}$$

$$y_i = f\frac{-Z_B}{X_B}$$

where f is the focal length and x_i and y_i are the image point coordinates.

2.8 Quadrotor UAV

A quadrotor is a rotary-wing UAV, which results in numerous advantages over fixed-wing UAVs. Some of these include Vertical Takeoff and Landing (VTOL) capability, the ability to hover and make slow precise movements. Four rotors also increase the vehicle's payload, while enhancing maneuverability when in a crowded environment or landing in a small area.

2.8.1 Quadrotor Description and Modeling

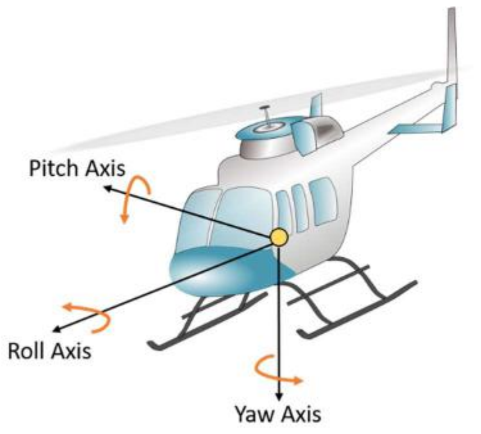


Figure 2.17 Yaw, pitch, and roll angles

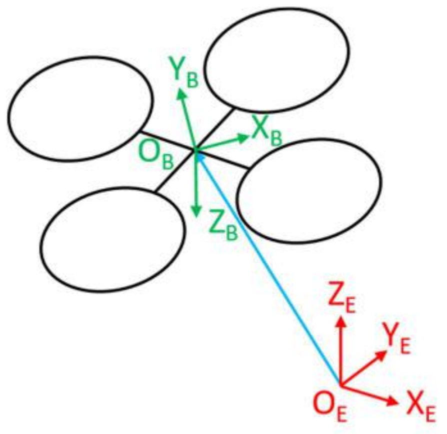


Figure 2.18 Inertial and Body reference frames

Two coordinate frames are used to explain the equations of motion for the quadrotor: the Earth (E) inertial reference frame and the Body (B) fixed reference frame [53]. The angular position or attitude of the quadrotor is defined by the orientation of the B-frame with respect to the E-frame whereas position is defined on the E-frame. The following assumptions are made:

- The origin of the B-frame is located at the center of mass of the vehicle
- the body is rigid
- the axes of the B- frame coincide with the body principal axes of inertia (Fig. 2.18)

The six degrees of freedom of the rigid body are defined as $q = [x, y, z, \phi, \theta, \psi]^T$, where the triplet (x, y, z) represents the position of the center of mass in the E-frame and the "roll-pitch-yaw" (ϕ, θ, ψ) set of Euler angles is the representation of the orientation of the quadrotor in the same reference frame (Fig. 2.17). The dynamical model is derived using the Lagrangian approach:

$$\ddot{x} = U_1 \frac{(\cos\psi \cos\phi \sin\theta + \sin\psi \sin\phi)}{m}$$

$$\ddot{y} = U_1 \frac{(\sin\psi \cos\phi \sin\theta - \sin\phi \cos\psi)}{m}$$

$$\ddot{z} = U_1 \frac{(\cos\theta \cos\phi)}{m} - g$$

$$\dot{p} = \frac{(I_y - I_z)}{I_x} qr + \frac{1}{I_x} U_2 - \frac{J_m}{I_x} q\omega_R$$

$$\dot{q} = \frac{(I_z - I_x)}{I_y} pr + \frac{1}{I_y} U_3 + \frac{J_m}{I_y} p\omega_R$$

$$\dot{r} = \frac{(I_x - I_y)}{I_z} pq + \frac{1}{I_z} U_4$$

$$\dot{\phi} = p + \sin(\phi)\tan(\theta)q + \cos(\phi)\tan(\theta)r$$

$$\dot{\theta} = \cos(\phi)\, q - \sin(\phi)\, r$$

$$\dot{\psi} = \frac{\sin(\phi)}{\cos(\theta)}q + \frac{\cos(\phi)}{\cos(\theta)}r$$

where p, q and r are the attitude velocities in the B-frame, m is the vehicle mass, I_x, I_y and I_z are the body principal moments of inertia, J_m is the motor inertia and ω_R reads $\omega_R = -\omega_1 - \omega_3 + \omega_2 + \omega_4$. As is common practice in the quadrotor literature, the input vector (U_1, U_2, U_3, U_4) has been defined so that the state rates are linear in the control variables. More precisely, the U_is are defined in terms of the thrust and torque generated by each rotor (L_i and T_{Li}, $i = 1, \dots, 4$ respectively) as:

$$U_1 = \sum_{i=1}^{4} L_i,$$

$$U_2 = l(L_4 - L_2),$$

$$U_3 = l(L_3 - L_1),$$

$$U_3 = -T_{L1} + T_{L2} - T_{L3} + T_{L4},$$

where l is the distance from the center of gravity of the vehicle to the center of each of the rotors [54].

2.8.2 Control Schemes

A minimum of two controllers are typically involved in a UAV autopilot system [55]. A low-level controller maintains stability and attitude control with a high update frequency. A programmable high-level controller is used to interact with the vehicle’s attitude and position.

2.8.2.1 Low-level Controller

The default low-level controllers provided with UAV autopilots were used in this project. These controllers are responsible for maintaining flight stability through rapid sensor readings and high-frequency update cycles.

Specifically, these controllers collect sensor data, provide attitude, pose, and height stabilization, and manage data transition and reception.

2.8.2.2 High-level Visual-based Control

The computer vision systems developed in this work provide high-level control commands to the UAV. These position control commands resemble input provided by a human operator using a remote control, and correspond to yaw, pitch, and roll rates.

Chapter 3: IR-based Autonomous Landing

3.1 Introduction

In this chapter, the design, development, and validation of an IR-based landing system for a quadrotor UAV is presented. The system was developed for a downward-facing camera mounted on a small quadrotor. The ground station was marked using an IR radiator. The computer vision algorithm was designed to orient the UAV above the IR beacon before executing the landing procedure. Experimental tests were conducted indoors, but range tests demonstrate the system's feasibility at longer ranges such as those experienced outdoors. The hardware required for this system is separated into two sections; the on-board image processing on the UAV and the active IR ground station.

3.2 Problem Statement

Precise landing of multirotor unmanned aerial vehicles (UAVs) in confined, GPS-denied and vision-compromised environments presents a challenge to common autopilot systems. In this work, an autonomous IR landing system using a ground-based IR radiator, UAV-mounted IR camera, and image processing computer is outlined. Previous work has focused on UAV-mounted IR sources for UAV localization, or systems using multiple distributed ground-based IR sources to estimate UAV pose [22]. This experiment considers the use of a single ground-based IR radiator to determine the UAV's relative location in three- dimensional space. The outcome of this research significantly simplifies the landing zone setup by requiring only a single IR source, and increases operational flexibility, as the vision-based system adapts to changes in landing zone position. The usefulness of the system is especially demonstrated in vision-compromised applications such as nighttime operations, or in smoky environments observed during forest fires. A high-power IR radiator for future research in the field of outdoor autonomous point-to-point navigation between IR sources where GPS is unavailable is also evaluated.

3.3 Related Work

Small multirotor unmanned aerial vehicles (UAVs) have become extremely popular in recreational and industrial applications in recent years. Due to their affordability and portability, UAVs have replaced their manned counterparts in many industries. Their commercial uses include mine surveying and pipeline inspections, as well as applications in law enforcement, search & rescue operations, and wildfire monitoring.

The most complicated and delicate phase of a UAV flight is the landing phase [21]. This is further compounded in GPS-denied environments or vision-compromised situations such as those encountered during nighttime operations or those in smoky conditions. Many industries require UAVs to operate during such sub-optimal conditions, therefore this critical phase of flight must be simplified to ensure reliable and uninterrupted operation.

This chapter is primarily concerned with developing an autonomous IR vision-based landing system using the concepts of visual servoing for exactly such conditions. Using a single, ground-based, IR beacon, the UAV was able to estimate its relative position in three-dimensional space in a dark environment. IR light was selected for the beacon, as in some applications it is critical for the landing zone to be invisible to the human eye.

To generate the IR source, A high-power IR radiator developed by Sennheiser for use in commercial audio reinforcement applications was tested. The IR signal transmitted by this beacon can be modulated, which may prove useful in associating a specific signature to the IR target, ruling out potential false positives from other IR sources, as validated by Conte and Trancossi [56].

The IR source was detected and processed on board the UAV using an IR camera and a Raspberry Pi single-board computer. The coordinates of the IR beacon were obtained from each frame, and high-level control commands were sent to the UAV autopilot to center the vehicle above the target. UAV height was estimated based on the image area occupied by the beacon, and once centered, the UAV's height was decreased incrementally until the vehicle had landed safely. Fig. 3.1 shows the hardware used for the experiments.

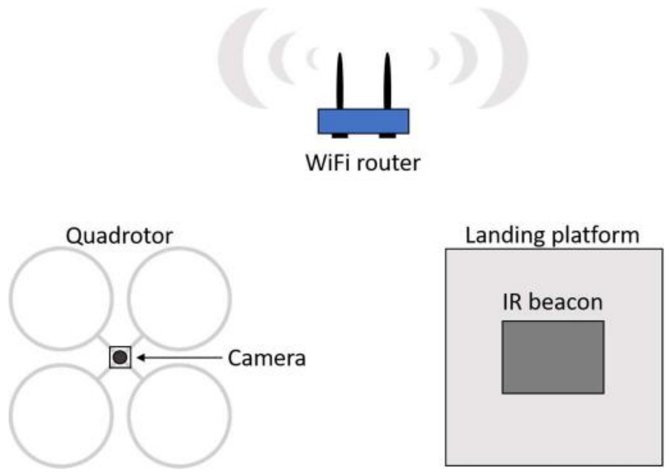


Figure 3.1 System diagram

Various vision-based landing techniques for UAVs have been studied, including autonomous indoor and outdoor solutions [25]. Yang *et al.* [35] used a monocular vision system to take off, hover, and land using a micro aerial vehicle. This system used real-time landing pad recognition to determine its location in an indoor environment. Since it used colour vision, the system required a well- and consistently-lit area, making it unsuitable for use at night or in environments with varying lighting conditions. Wenzel *et al.* [30] demonstrated an IR-based indoor landing system based around a Wii remote camera. They used a microcontroller on the UAV to detect a pattern of IR LEDs on the ground. Their system worked well on short range, but was extremely limited by distance. If the UAV was too far from the target, the camera could not detect the point-source LEDs required by their algorithm. The outlined approach solves this problem by using a single IR source with a wide dispersion pattern, and proposes the possibility of replacing pattern recognition with a specifically modulated signal. Like the work presented by Wenzel *et al.*, this work can also be applied to a stationary or slowly moving target.

Some outdoor landing systems have also been developed for both fixed-wing and rotary-wing UAVs. Kummer *et al.* [36] developed a vision-based landing system to guide fixed-wing UAVs to a safe landing on an inflatable airbag. As with all colour-based methods, this approach fails in low light or vision-compromised environments. Gui *et al.* [31] tested an IR vision-based landing system for fixed- wing UAVs. Their system used a series of IR lamps arranged on the runway to estimate the UAV position and guide the vehicle to a landing. Although successful, their method required precise setup of multiple lamps, thereby making quick field deployment difficult. An inverted approach was taken by Kong *et al.* [40], by placing an IR stereo vision system on the ground and using the long-wave IR signature of the UAV for localization. The stereo IR vision system was mounted on an actuated pan/tilt head, allowing it to track the movement of the UAV and guide it towards a simulated aircraft carrier deck. This method seems attractive, as placing the camera on the ground rather than on the airframe reduces UAV payload weight and allows for increased computational power provided by off-board computers. This separation, however, also severely limits the system, as a communications link is mandatory between the ground station and the UAV.

The presented system builds on the successful application of IR in the literature to create a robust landing system for GPS-denied, nighttime and vision-impaired environments. By placing the camera on board the UAV the need for a communications link between the air and ground stations is removed, while still maintaining specificity through the possibility of modulating the IR signal. The complexity of the landing zone setup is also reduced by using a single IR source, which is bright enough to be detected from over 30 meters away, and large enough to be used for height estimation.

3.4 System Description

A large part of developing computer vision systems for small UAVs is the challenge of system integration [57]. Developing a streamlined workflow when working with multiple embedded devices which need to communicate can be a challenge at the best of times. The process is often plagued with developing workaround after workaround for problems which arise. The system described here is the result of much research and experimentation to find the best hardware and software tools to assist in rapid prototyping of indoor computer vision applications.

3.5 Hardware

In this section, details on the choice of hardware for the experiments and filtering techniques for the IR signal are presented.

3.5.1 Quadrotor UAV

The Parrot AR.Drone 2.0 was selected as the platform of choice for these tests, as it is safe for indoor flights when using the external frame (Fig. 3.2). It can lift the required payload (120 g) consisting of a single-board computer (SBC), camera, and 1800 mAh portable USB powerbank. This AR.Drone 2.0 is a typical X-configuration quadcopter with an external foam hull. Some of its specification are provided in Table 3.1.

Table 3.1 Parrot AR.Drone 2.0 specifications

Camera	720p 30fps HD camera
Lens	Wide-angle lens: 92° diagonal
Connection	Wi-Fi
Weight	With internal frame: 380 g
	With external frame: 420 g

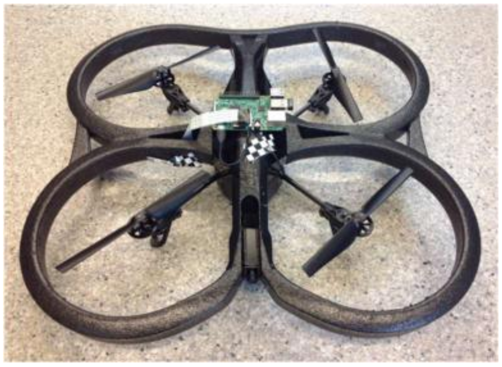


Figure 3.2 Parrot AR.Drone 2.0 UAV and onboard Raspberry Pi vision computer

3.5.1.1 Battery Life:

Multiple tests were conducted using the standard 1,000 mAh battery which ships with the AR.Drone. This proved insufficient for experimental use, as flight time was severely limited at under 10 minutes. The AR.Drone 2.0 Power Edition was purchased, which included two 1,500 mAh batteries. These proved to be more useful, as flight time was increased to approximately 15 minutes per battery. With the increase in battery life, it became possible to power the Raspberry Pi from the on-board USB power port, thereby shedding the extra weight of the USB power bank.

3.5.2 Vision Computer

The payload consisted of a Raspberry Pi 2 Model B SBC (Fig. 3.3) and a Raspberry Pi NoIR camera. The Raspberry Pi 2 Model B is the second-generation Raspberry Pi, and includes improved features such as a 900MHz quad-core ARM Cortex-A7 CPU and 1GB RAM. It also features 4 USB ports, 40 GPIO pins, a full-size HDMI port, Ethernet port, Camera interface (CSI), and VideoCore IV 3D graphics core.

Power for the Raspberry Pi and camera was provided by the built-in USB port on the AR.Drone 2.0. Since the AR.Drone broadcasts its own 2.4 GHz

Figure 3.3 Raspberry Pi 2 Model B

wireless network, a Wi-Fi dongle was used to communicate between the vision computer and the UAV. Connecting a remote computer to the AR.Drone's network allowed Secure Shell (SSH) access to the Rapsberry Pi to complete remote programming or command execution tasks. This allowed for faster turnaround times when making minor code adjustments during testing.

3.5.3 Camera

The Raspberry Pi NoIR camera (V1) was selected to be used in this experiment. Table 3.2 outlines some of the camera-specific features:

Table 3.2 NoIR v1 specifications (source: Raspberry Pi Foundation)

NoIR Camera Module v1	
Size	25 × 24 × 9 mm
Weight	3 g
Still resolution	5 Megapixels
Video modes	1080p30, 720p60 and 640 × 480p60/90
Sensor	OmniVision OV5647
Sensor resolution	2592 × 1944 pixels
Sensor image area	3.76 × 2.74 mm
Pixel size	1.4 µm × 1.4 µm
Optical size	1/4"
Fixed focus	1 m to infinity
Focal length	3.60 mm +/- 0.01
Horizontal field of view	53.50 +/- 0.13 degrees
Vertical field of view	41.41 +/- 0.11 degrees
Focal ratio (F-Stop)	2.9

The NoIR camera features the same specifications as the regular Raspberry Pi camera module, except that it does not employ an IR filter (the abbreviation “NoIR” stands for “No Infrared,” referring to the fact that the camera ships without an IR filter). This allows IR light to be detected by the imaging sensor. The Picamera Python library was used to access and control the camera.

The NoIR camera was mounted in a nadir-pointing configuration in a central location on the AR.drone airframe. According to the pinhole camera model, the image of any point $[X_B\ Y_B\ Z_B]^T$ expressed in B is:

$$x_i = f\frac{Y_B}{Z_B}$$

$$y_i = f\frac{X_B}{Z_B}$$

where f is the focal length and x_i and y_i are the image point coordinates.

3.5.4 IR Beacon

A Sennheiser SZI 1029 IR radiator (Fig. 3.4) was tested for suitability as a beacon for UAV landing applications. The SZI 1029 is a 5-watt high-power IR radiator designed to transmit audio signals for hearing assistance systems in commercial sound reinforcement applications. In its intended purpose, it is

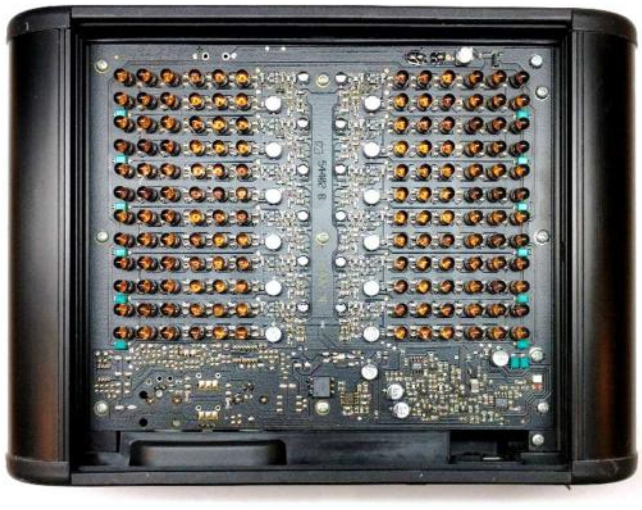


Figure 3.4 Sennheiser SZI 1029 high-power IR radiator with front cover removed

typically mounted on the interior wall of a building and connected to a modulator, which converts audio signals to be transmitted over IR. These IR signals are then picked up by assistive hearing devices. The SZI 1029 uses 144 GaAlA (Gallium Aluminum Arsenide) LEDs transmitting at the 880 nanometer wavelength and can cover areas of up to 800 square meters with IR light [58].

Table 3.3 Sennheiser SZI1029 specifications

Sennheiser SZI 1029 IR Radiator	
Technical Data	
IR Diodes	144 GaAlAs
Average Radiating Power	5 W
Operating Voltage	85 – 260 V, 50 – 60 Hz
Current consumption (operation)	350 mA (120 V: 610 mA)
Current consumption (stand-by)	50 mA
RF input	BNC socket
Input impedance	approx. 5 kΩ
Max. coverage area	approx. 800 m^2
Threshold voltage for automatic on/off function	50 mV (RF signal)
RF output	BNC socket
Weight	approx. 2.1 kg
Dimensions	250 mm (L) × 80 mm (W) × 180 mm (H)

The radiator requires a sub-carrier frequency in the range of 30 kHz - 6 MHz to trigger the automatic on/off function of the LEDs. A 1 MHz sine wave at 1 volt (peak-to-peak) was provided via the BNC input socket. The SZI 1029 used for these tests had been decommissioned due to the failure of 24 LEDs in one bank (they are arranged in 2 banks of 72 LEDs), therefore only a single bank of LEDs was used for these experiments. Even with the unit operating at only half capacity, the results look promising for long range outdoor applications.

Fig. 3.5(1) shows one bank of LEDs as seen by the IR camera at a distance of 40 centimeters; the individual LEDs are clearly visible. Fig. 3.5(2) shows the same bank of LEDs at a distance of 1 meter. Fig. 3.5(3) shows the bank of LEDs at a distance of 30 meters. The radiator is the bright object in the center of the image; the surrounding reflections are due to this test being conducted in an indoor hallway. Even at a 30-meter separation, the IR blob detection algorithm successfully located and isolated the beacon's coordinates.

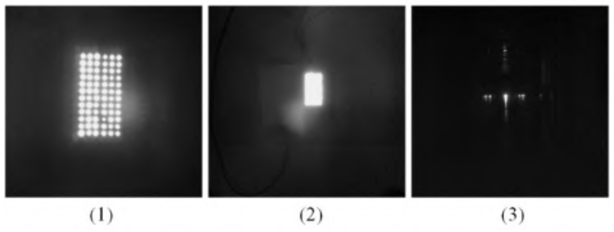


Figure 3.5 Images of the IR radiator at 0.4, 1, and 30 meter distances

3.5.5 Camera and Filtering

A combination of hardware and software filters were used to modify the IR signal before extracting the coordinates of the beacon from each frame. Hardware filters were applied to the IR beacon and camera. Further software filters were applied to the captured images.

The IR source was provided by the Sennheiser SZI 1029 radiator. Fig. 3.6(1) shows the IR radiator at a distance of 40 centimeters. Half of the radiator was covered with an opaque polymer sheet, resulting in 72 LEDs being considered active. At close range, each individual LED is visible as a distinct point source of light, which caused the blob detection algorithm to detect each LED as a separate feature. This was not desirable, so a translucent diffuser was applied to the radiator. The result of this is shown in Fig. 3.6(2). Next in the signal chain (at the camera end) was a visible light filter. Since the Raspberry Pi NoIR camera detects both visible and short-wave IR frequencies, a hardware visible light filter was installed in front of the camera to filter out visible light, only allowing IR wavelengths to pass to the sensor. The Raspberry Pi NoIR camera was used to capture the images. Fig. 3.6(3) shows the regular Raspberry Pi NoIR camera image with both visible and IR light before application of the visible light filter. Fig. 3.6(4) shows the result after addition of the filter in front of the camera lens. Now only the IR beacon is visible in the center of the image.

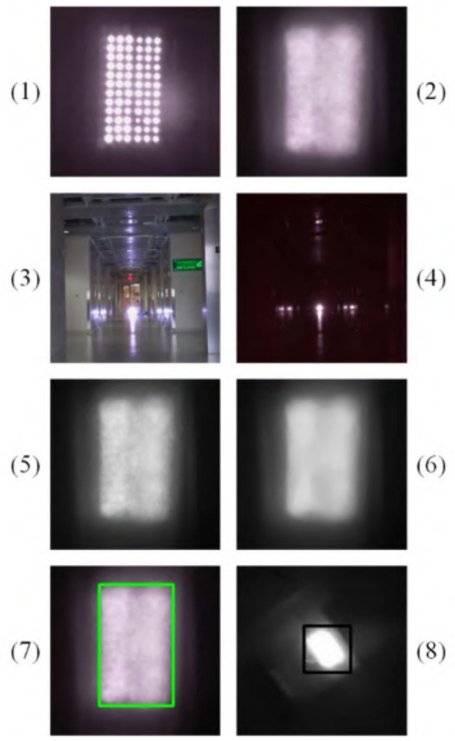


Figure 3.6 Results of hardware and software filters

Next, two software processing steps were applied; the OpenCV libraries were used for the software-based image processing. Fig. 3.6(5) shows the effect of converting the image to grayscale, and Fig. 3.6(6) shows the same image after application of a 17×17 pixel median blur filter to remove any background scatter. Finally, Fig. 3.6(7) shows a sample bounding box around the region of interest. Fig. 3.6(8), taken from a flight test, shows the bounding box used to determine the *x* and *y* coordinates of the beacon. These were obtained from the center of the box.

3.6 Software (Servoing Algorithm)

The visual servoing algorithm sent image error values to a proportional controller, which altered the duty cycle, or rate, at which high-level control commands were sent. Multiple image regions were defined, which specified which control commands were given priority depending on the location of the beacon in the camera's field of view. This was required, as the UAV only accepts high level commands from the control computer. By defining multiple regions, the priority level and duty cycle of each command could be altered to achieve the most stable response. Fig. 3.7 outlines the terminology for various image regions. The Image Area consists of the entire frame captured by the camera. A capture resolution of 320×240 pixels was used to maintain a high frame rate. A camera coordinate frame was overlaid on the image, with the coordinates $(0,0)$ being at the center of the image. The Partially Stable region was defined as a 70×70 pixel region located at the center of the image. If the beacon was located anywhere in the image region outside of the Partially Stable region, commands were sent at a high duty cycle to minimize Err_x and Err_y as seen in Fig. 3.9. If the beacon moved out of the Image Region during this phase, the last command was inverted until the beacon was detected again. If the beacon was still not relocated, the UAV increased altitude until the beacon was once again within the field of view. Err_x was given priority, and was minimized first. Once an appropriate value was obtained, Err_y was minimized. Once inside the Partially Stable region, commands were sent at a reduced duty cycle (50%) to bring the beacon into the Stable or Second Landing regions. The Second Landing region is defined as an in-between zone where the beacon borders the Stable and Partially Stable regions. The UAV was allowed to decrease altitude and land in this region, however higher priority was still given towards moving the vehicle into the Stable zone. Once in the stable region, the beacon location is maintained at $(0, 0)$, altitude is decreased at the highest priority, the command duty cycle is reduced to 33%, and a land command is sent if the beacon occupies 4% of the total image size.

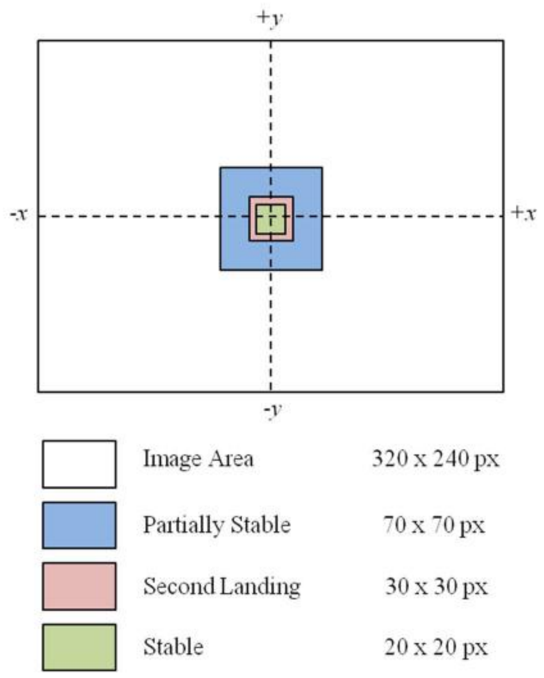


Figure 3.7 Description of regions and terms used in the servoing algorithm

3.7 Experimental Methodology

The experiments were conducted indoors using a stationary IR beacon (Fig. 3.8). The beacon was placed on the ground, facing upwards. A transparent plastic sheet was placed above the beacon to provide a flat surface for the landing platform. The UAV was positioned approximately 2 metres back, and 1 metre to the left of the target. This ensured that once the beacon was detected, Err_x and Err_y would be greater than zero. Once positioned, the UAV was commanded to take off and follow a forward trajectory until the beacon was detected. Once the beacon was in frame, the visual servoing algorithm began sending high level control commands to the UAV. These commands included:

1) *pitch forward if $Err_y > 0$*
2) *pitch backward if $Err_y < 0$*
3) *roll right if $Err_x > 0$*
4) *roll left if $Err_x < 0$*
5) *decrease altitude if area of target < 4% of image area*

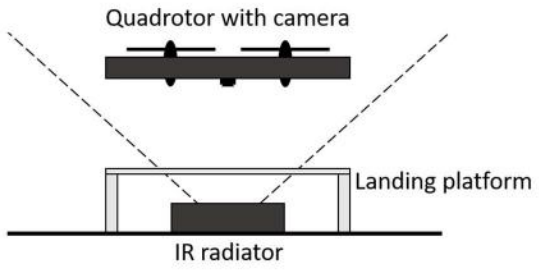


Figure 3.8 Chapter 3 experimental setup

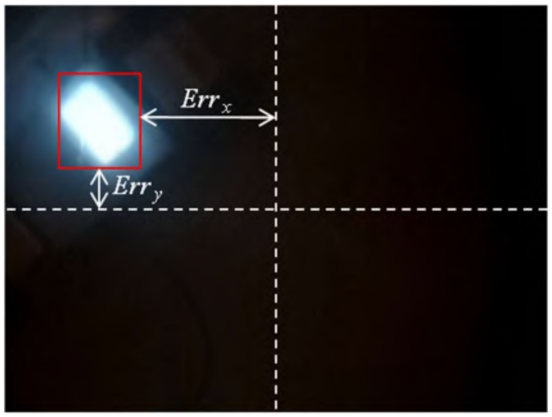


Figure 3.9 Error determination for visual servoing algorithm

3.8 Results and Discussion

Fig. 3.10, 3.11, and 3.12 show the results of a typical flight and landing. Fig. 3.10 shows Err_x and Fig. 3.11 shows Err_y. The image error in X is 0 for the first 3.8 seconds, as the UAV was moving forward and searching for the beacon. At 3.8 seconds, the beacon was detected. The rapid spike in Err_y is explained by the detection of the beacon at the top of the image frame, which was quickly negated by the UAV's forward momentum. The servoing algorithm worked to minimize Err_x and Err_y between 3.8 and 10.6 seconds into the flight. At that point, the errors were sufficiently small to begin a descent as seen in Fig. 3.12. As the UAV displaced during this phase, the algorithm re-stabilized the vehicle from 12.4 to 20.2 seconds, at which point a land command was sent.

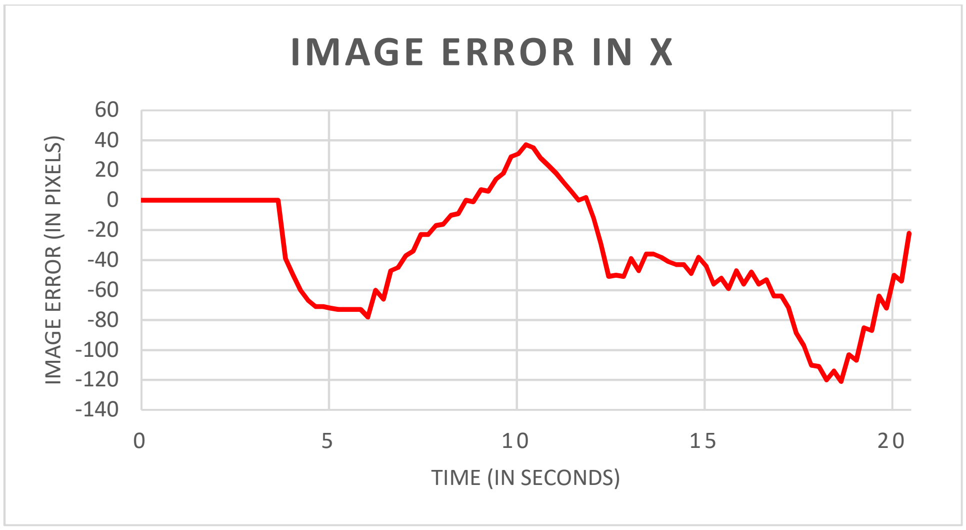


Figure 3.10 Image error (in pixels) in the X direction

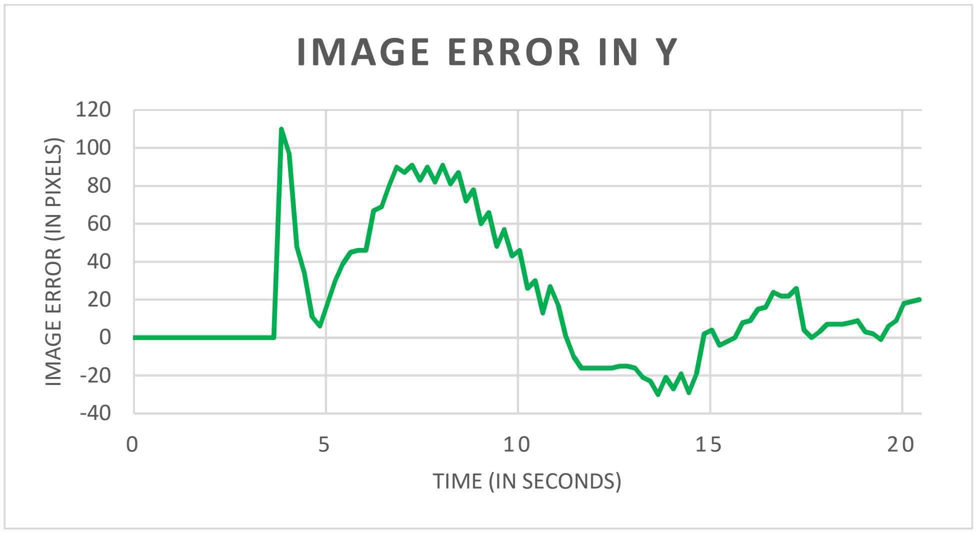


Figure 3.11 Image error (in pixels) in the X direction

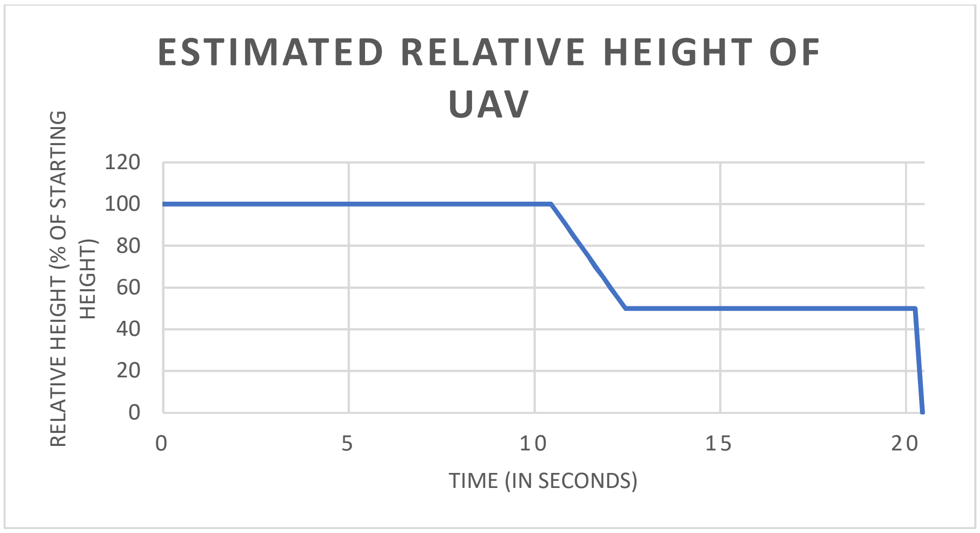


Figure 3.12 Estimated relative height of UAV in % of starting height

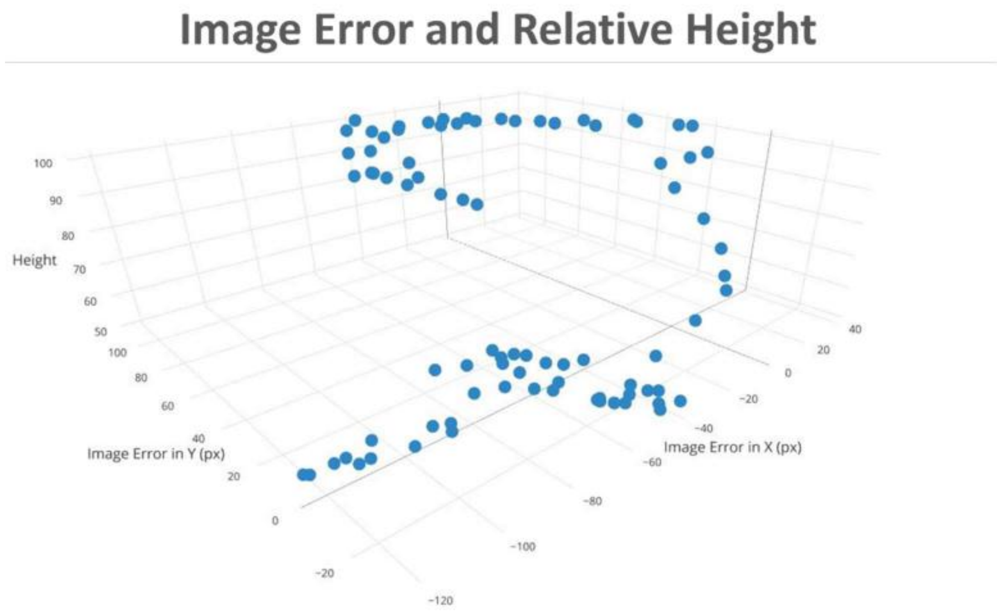


Figure 3.13 Image error in X, Y, and relative height of the UAV over time

Fig. 3.13 shows the image error in x, y, and relative height of the UAV plotted in a 3D graph over time; this re-creates the UAV's 3D flight path during the experiment. The decreasing spiral pattern results from slight overshoot in the x- and y-axes as the system minimizes error and decreases altitude. The defined regions determine which commands are given priority.

Success of the visual servoing algorithm was defined as the UAV’s x- and y- position converging on the beacon with a stable control response, and a successful landing on the platform. The visual servoing algorithm was successful in minimizing overshoot and in maintaining the beacon within the image frame. Even at a close 1-meter altitude above the beacon, the system remained stable due to the use of varying duty cycles for sending control commands to the UAV. Approximately 20 consecutive tests were conducted, and the UAV consistently landed on the 40 × 40 cm platform, which was just slightly larger than the vehicle itself.

In this chapter, an infrared landing system for multirotor UAVs in GPS-denied and vision-compromised environments was presented. A new type of IR beacon with possibilities for uniquely modulated carriers was evaluated for use in indoor and outdoor applications. This beacon was successfully detected and used by the vision-based servoing algorithm to land the UAV during indoor experiments. The application of this easily deployable system to outdoor environments and slow-moving targets is being further evaluated.

Chapter 4: NDVI Navigation

4.1 Introduction

In this chapter, the design, development, and validation of an indoor navigation system for a quadrotor UAV based on IR computer vision is presented. A small quadrotor UAV is outfitted with an IR camera and vision computer, and on-board software calculates the Normalized Difference Vegetation Index (NDVI) to differentiate between vegetation and non-vegetation in real time, allowing the geometrical characteristics of a row of plants in a greenhouse to be used for navigation.

4.2 Problem Statement

The research presented in Chapter 4 aims to solve the overarching problem of pollination of plants in indoor plantations, such as greenhouses. Pollination in greenhouses is currently a manual labour-intensive process done by humans. The alternative option uses Bombus bees (bumblebees), however the bees have several limitations, including sensitivity to pesticides and heat. The development of a plant-specific indoor vision-based navigation algorithm for UAVs was a pre-requisite to developing a robotic solution for pollination of plants in indoor environments, such as greenhouses. To accomplish such a task, indoor vision-based navigation algorithms for Unmanned Aerial Vehicle (UAV) systems, as well as novel control algorithms for indoor precision-agriculture tasks must first be developed.

The UAV must detect rows of vegetation, and use the rows of plants to navigate without a human operator. The first step in achieving this goal was to detect the difference between vegetation and non-vegetation. This was done using a short-wave IR camera and NDVI, which stands for Normalized Difference Vegetation Index, and can be used to analyze whether an image contains live green vegetation or not. Shortwave IR cameras are very similar to regular colour cameras, with the exception that the red colour band is replaced with IR. After filtering images obtained using this camera, an NDVI image can be obtained. In a greenhouse, this can be used to selectively differentiate between plants and non-vegetation, such as the ground and structural components. The cost of an NIR-modified camera is comparable to a regular colour camera [59], but provides the added benefit of providing additional information on plant health; they are the standard in agricultural imaging. NIR cameras also eliminate the potential for false positives arising when a regular colour camera thresholds for green, as not necessarily everything green in the image is plant matter. A NIR camera processed using the NDVI guarantees detection of photosynthesizing plant matter and rejection of non-photosynthetic material.

4.3 Theory

The approach to solving this problem is two-fold: The first challenge was to use the IR camera to determine the normalized difference vegetation index (NDVI). Once an accurate method for determining the NDVI was achieved, the second step was to use the geometric properties of the image to control the UAV. Here an introduction to the NDVI concept and the control theory required to navigate a UAV using the geometric properties of a greenhouse row is presented.

4.3.1 Normalized Difference Vegetation Index (NDVI)

The normalized difference vegetation index (NDVI) is a simple graphical indicator developed to analyze remote sensing measurements to determine if the observed target contains live vegetation or not. Early observation satellites acquired imagery in the visible and NIR bands, allowing researchers to exploit these strong differences in plant reflectance values to determine their spatial distribution. Although the NDVI is typically associated with satellite imagery, it is just as applicable in oblique aerial or terrestrial photography.

The NDVI relies on differences between reflected and absorbed wavelengths of light by healthy photosynthesizing vegetation compared to unhealthy or sparse vegetation. Healthy green vegetation absorbs solar radiation in the photosynthetically active radiation (PAR) spectral region in order to provide the process of photosynthesis with its required energy. Leaf cells re-emit solar radiation in the near-infrared (NIR), because the photon energy at wavelengths longer than approximately 700 nanometers is not large enough to synthesize organic molecules; stronger absorption at these values would cause the plant to overheat. Therefore, live green vegetation appears relatively dark in the PAR region, and relatively bright in the NIR region. In comparison, clouds and snow typically have a much brighter red colour response and are darker in the NIR frequency band.

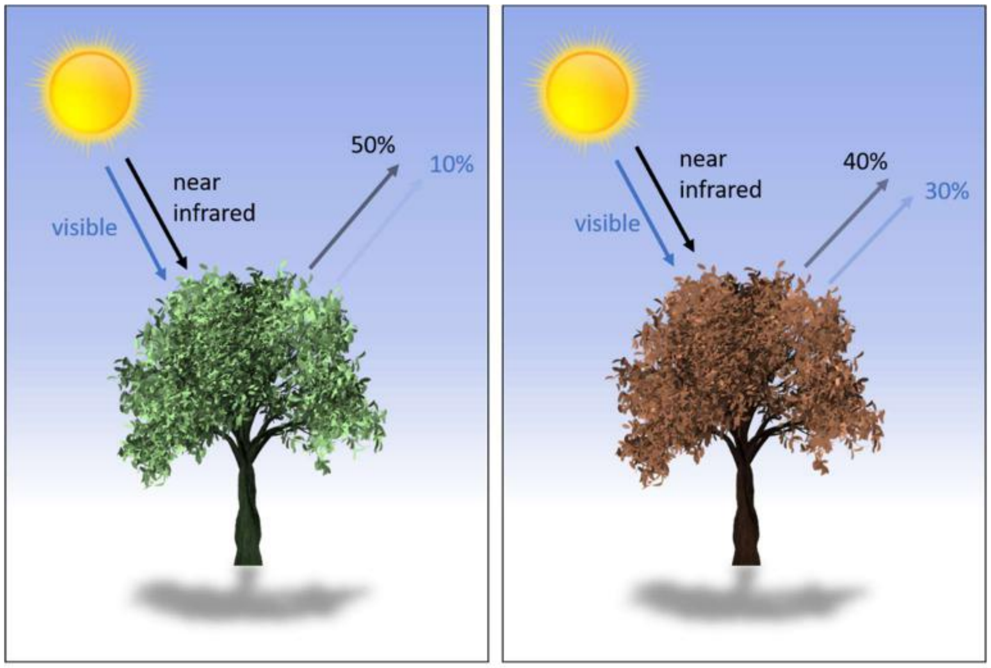
Chlorophyll, the pigment in plant leaves, strongly absorbs visible light between 0.4 and 0.7 μm wavelengths for photosynthesis. In contrast, the cell structure of the plant's leaves strongly reflects NIR light from 0.7 to 1.1 μm. Plants with more leaves have a greater effect on the reflected visible light and NIR values.

The NDVI is calculated from these individual measurements as follows:

$$NDVI = \frac{(NIR - VIS)}{(NIR + VIS)}$$

where VIS and NIR stand for the spectral reflectance measurements acquired in the Reg, Green, or Blue (visible) and NIR regions, respectively. The spectral reflectance values are themselves a ratio of the reflected vs incoming radiation in each spectral band; this results in values between 0.0 and 1.0. The NDVI therefore varies between -1.0 and +1.0. Although the NDVI is functionally related to the simple IR/visible ratio, it is not linearly equivalent. The NDVI has several advantages over the simple IR/visible ratio, including possible linearity with vegetative properties such as biomass, as well as a limited range. The simple ratio on the other hand is always positive, but has an infinite range. It is also important to note that the VIS term in the NDVI numerator creates negative values by scaling the result. Therefore, the NDVI is functionally and linearly equivalent to the ratio NIR / (NIR+VIS), which ranges from 0 to 1 and is thus never negative nor limitless in range. It is important to remember that the NDVI algebraic formula is a transformation of a spectral ratio (NIR/VIS), and it has no functional relationship to a spectral difference (NIR-VIS).

In general, a large amount of reflected radiation in NIR wavelengths compared to visible wavelengths correlates to dense vegetation. In satellite imagery, this corresponds to forested areas. The NDVI can also be used to determine photosynthetic activity, capacity, and energy absorption.



$$\frac{(0.50 - 0.10)}{(0.50 + 0.10)} = 0.67 \qquad \frac{(0.40 - 0.30)}{(0.40 + 0.30)} = 0.14$$

Figure 4.1 NDVI differences between healthy and non-healthy vegetation

NDVI is calculated from the visible and NIR light reflected by vegetation. Fig. 4.1 demonstrates that healthy vegetation (left) absorbs most of the visible light, and reflects a large portion of NIR light. Unhealthy or sparse vegetation (right) reflects more visible light and less NIR light. The numbers used in the figure represent actual values, but real vegetation values are much more varied.

4.3.2 Geometry

It is assumed that roll and pitch angles can be measured by the onboard inertial measurement unit (IMU). Conversely, since GPS and compass sensors are ineffective in a greenhouse environment, the UAV's position and heading are estimated from image features. In particular, the two ground lines of the greenhouse row are utilized. These lines are formed by the intersection of the vegetation "walls" and the ground; namely, by the intersection of vegetation and non-vegetation planes. The two ground lines are defined as:

$$Lf_r = [t \quad \mathrm{W} \quad 0]^T, \qquad Lf_l = [t \quad -\mathrm{W} \quad 0]^T$$

where $t \in [0 \quad L]$ is the row longitudinal parameter, L is the row length and W is the row width, which is assumed to be known.

The UAV state is defined by six variables, the vector $[x \quad y \quad z]^T$ is the UAV position, and its angular state is represented by the three Euler angles, Roll ϕ, Pitch θ and Yaw ψ . The camera is attached to the UAV, and for simplicity, its state is assumed identical to the state of the UAV.

Projecting the two (parallel) ground lines to the onboard camera image plane, two inclined lines are obtained in the image (Fig. 4.2). The intersection of the two lines is defined as the vanishing-point, and its image coordinates are,

$$Vp_x = f(\sin\phi\tan\theta - \cos\phi\sec\theta\tan\psi)$$
$$Vp_y = f(-\cos\phi\tan\theta - \sin\phi\sec\theta\tan\psi)$$

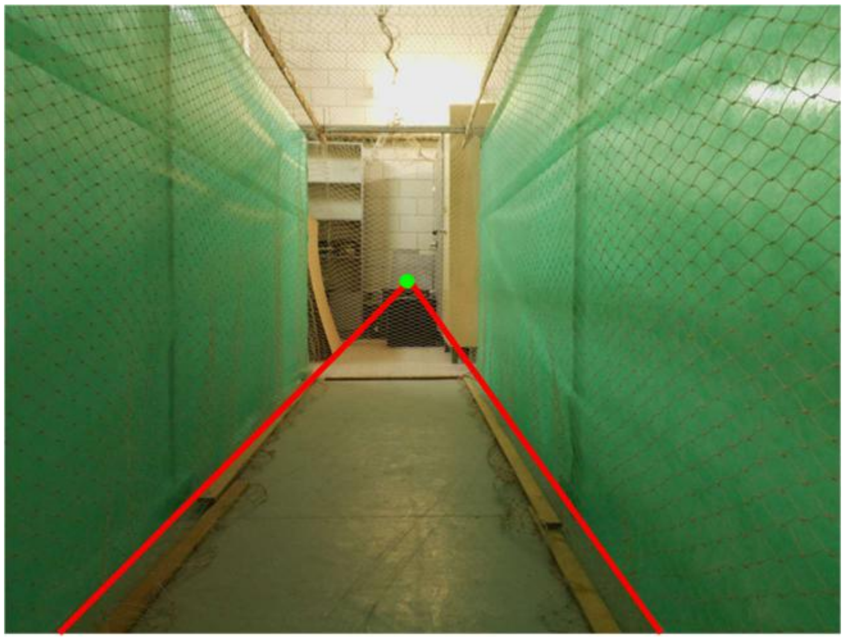


Figure 4.2 Theory image with floor lines (red) and vanishing point (green)

An important conclusion is that the vanishing point is insensitive to the UAV position (x, y, z), and highly sensitive to its heading. Assuming a negligible roll angle, then ψ can be estimated from Vp_x by

$$\tan\psi = -Vp_x \cos\theta / f$$

Using this measurement, ψ can be controlled and kept close to zero. Now calculating the slopes of the two image lines, for a negligible ϕ and ψ, one has,

$$a_r = \frac{-z}{(y - W)\cos\theta}, \quad a_l = \frac{-z}{(y + W)\cos\theta}$$

Note that the pitch (θ) is not assumed negligible, as it is used for longitudinal motion. Solving for the two position coordinates gives:

$$y = W\left(\frac{a_r + a_l}{a_r - a_l}\right), \quad z = -2W\left(\frac{a_l a_r}{a_r - a_l}\right)\cos\theta$$

It is interesting to note that the two slopes are insensitive to the longitudinal coordinate x. A possible method of estimating the longitudinal position of the UAV is by measuring the row width in the image at a given longitudinal point. Calculating the image row-width at the end of the row (where $x = L$) gives:

$$d = \frac{2Wf_c}{(L - x)\cos\theta + z\sin\theta} \Rightarrow$$

$$x = \frac{d(L\cos\theta + z\sin\theta) - 2Wf}{d\cos\theta}$$

Using this concept, the coordinates $[x \quad y \quad z \quad \psi]^T$ of the UAV can be estimated from image features, which replaces the need for GPS and compass sensors (the two primary sensors typically used for outdoor navigation). A rear facing camera may also be added to the UAV to improve line slope estimation, especially when the UAV is close to one of the two row ends.

$$ar\_\theta = \frac{z}{\cos\theta\,(W - Y)}$$

$$al\_\theta = \frac{-z}{\cos\theta\,(W + Y)}$$

$$d_\theta^2 = \frac{4W^2 {f_c}^2}{(L\cos\theta - X\cos\theta + Z\sin\theta)^2}$$

$$d\_\theta = \frac{2Wf_c}{L\cos\theta - X\cos\theta + Z\sin\theta}$$

4.4 System Description

Various components must be integrated to achieve the goal of an autonomous vision-based UAV, navigating using the geometry of plants in a greenhouse row. The on-board IR camera captures a forward-facing view from the UAV's perspective. This image is then analyzed by a multi-processor single-board-computer. The computer-vision pipeline must be optimized to achieve maximum performance using limited on-board processing power. The geometry of the image is extracted, and corresponding control commands, resembling those of a human operator, are issued to the vehicle's autopilot.

The system's hardware consists of three main components:

- Quadrotor UAV
- Flight controller autopilot
- Camera and vision computer

4.5 Hardware

4.5.1 Quadrotor UAV

The Spedix S250Q quadcopter from HobbyKing.com was used as the platform of choice (Fig. 4.3). It is a high quality, lightweight frame constructed of carbon fiber and high strength ABS resin. The frame includes two camera mounts on the front and rear, which perfectly accommodate the Raspberry Pi camera. The frame houses all components for a computer vision system in a compact sustainable package, and is ideally suited for an indoor or outdoor computer vision testbed.

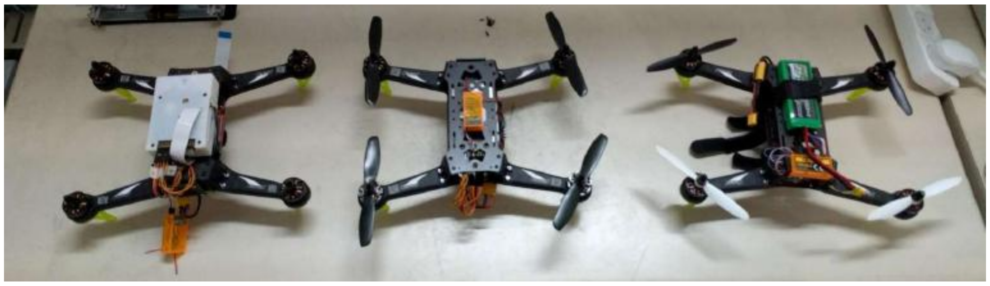


Figure 4.3 Small quadcopters fitted with an onboard camera and image-processing computer

Table 4.1 Specifications of the Spedix S250Q quadcopter frame

Spedix S250Q	
Dimensions	260 × 195 × 90 mm
Frame weight	143 g

4.5.1.1 Flight Time

Using the MultiStar Racer Series 3S 1400 mAh LiPo battery from HobbyKing.com, resulted in approximately 15-minute flight time. The Spedix Power Distribution Board was used to connect four ESCs; it outputs 5 VDC at 3 A, and 12 VDC at 3 A.

4.5.2 Vision Computer

The vision computer used was a Raspberry Pi 3 Model B (Fig. 4.4). This is the third-generation Raspberry Pi, and includes various new features such as a built-in WLAN chip. This computer is ideally suited for on-board image processing, as it is small, lightweight, interfaces directly with the Raspberry Pi Camera, has a quad-core processor, and connects to the NAVIO2 autopilot hat. Some of its key features include:

- Quad Core 1.2GHz Broadcom BCM2837 64bit CPU
- 1GB RAM
- BCM43438 wireless LAN and Bluetooth Low Energy (BLE) on board

- 4 USB 2 ports
- Full size HDMI
- CSI camera port for connecting a Raspberry Pi camera
- DSI display port for connecting a Raspberry Pi touchscreen display

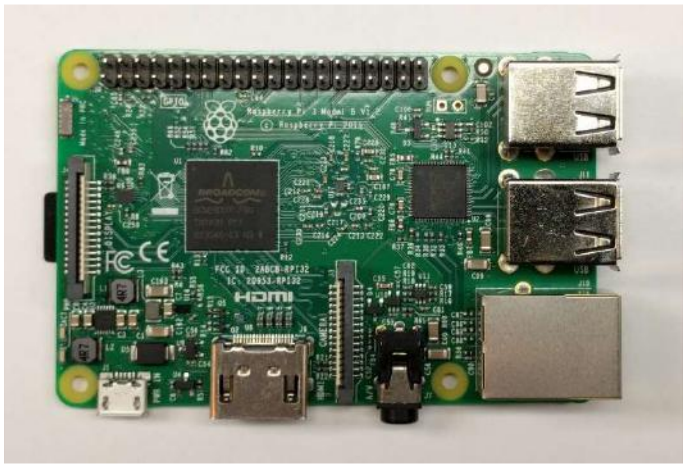


Figure 4.4 Raspberry Pi 3 Model B

4.5.3 Camera

The camera used for the on-board vision system is the NoIR Camera V2 developed by the Raspberry Pi Foundation. This camera offers various improvements over the version one camera used in the experiments in Chapter 3. To process images using the NDVI, the camera was outfitted with a Cinegel$^{®}$ R2007-Storaro Blue filter. This results in an NGB image (Near IR replaces Red channel), which allows the camera to detect IR and not blue wavelengths, resulting in bright areas in a picture representing areas of increased photosynthesis. This is explained in more detail in the “Software” section of this chapter.

Table 4.2 Raspberry Pi NoIR camera specifications

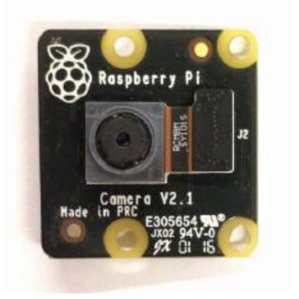
	
Sensor type	Sony IMX219PQ Colour CMOS 8-megapixel
Sensor size	3.674 × 2.760 mm (1/4" format)
Pixel count	3280 × 2464 (active pixels) 3296 × 2512 (total pixels)
Pixel size	1.12 × 1.12 um
Lens	f=3.04 mm, f/2.0
Angle of View	62.2 × 48.8 degrees
Board size	25 × 23.86 × 9mm
Mounting Holes	4 × D =2.20 mm on 12.5 × 21.0 mm centers

Table 4.3 Raspberry Pi NoIR camera video modes

Video Modes		
	Resolution	**FPS**
1	1080p30 cropped (680 pixels off left/right, 692 pixels off top/bottom	Up to 30 fps
2	3240x2464 full 4:3	Up to 15 fps
3	3240x2464 Full 4:3	up to 15fps (identical to 2)
4	1640x1232 binned 4:3	up 40fps
5	1640x922 2x2 binned 16:9 (310 px crop T/B before binning)	up to 40fps
6	720P bin+crop (360 px L/R, 512 px T/B before binning)	40..90fps (OC: 120fps)
7	VGA bin+crop (1000 px L/R, 752 px T/B before binning)	40..90fps (OC: 120fps)

4.5.4 **Filter**

To process images using the NDVI, the camera was outfitted with a Cinegel® R2007-Storaro Blue filter. This allows the camera to detect IR and not blue wavelengths, resulting in bright areas in a picture representing areas of increased photosynthesis. This will be explained in more detail in the explanation of the NDVI algorithm.

4.5.5 **Flight Controller Autopilot**

The UAV was outfitted with the NAVIO2 autopilot hat for the Raspberry Pi computer (Fig. 4.5). This small hat serves as an I/O board, harnessing the power of the Raspberry Pi, using it as a drone controller. The board features dual IMUs: accelerometers, gyroscopes and magnetometers for orientation and motion sensing and a high-resolution barometer which senses altitude with 10-centimeter resolution.

The module also features an RC I/O co-processor, capable of accepting PPM/SBUS input and providing 14 PWM output channels for motors and servos. Additional extension ports include exposed ARC, I2C and UART interfaces. A triple redundant power supply guards against system failures. A pre-configured Raspbian OS image comes with ArduPilot and DroneKit pre-installed, was used for this project.

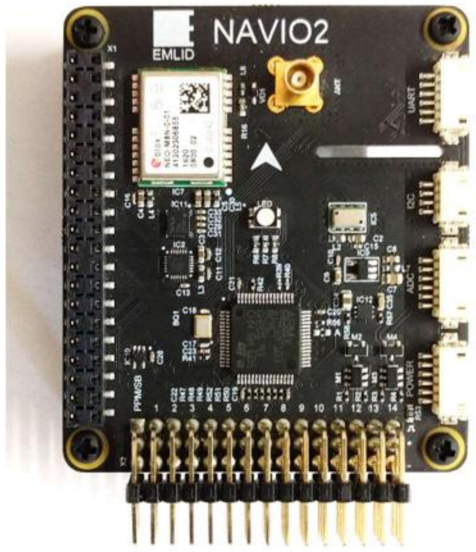


Figure 4.5 NAVIO2 autopilot hat for Raspberry Pi

Some of the NAVIO2's specifications are provided in Table 4.4.

Table 4.4 NAVIO2 specifications (source: emlid)

Mechanical	
Size:	55 × 65 mm
Weight:	23 g
Operating tº:	-40…+85ºC
Electrical	
Supply voltage:	4.75-5.25V
Power supply:	Power module, servo rail, USB
Average current consumption:	<150mA

Ports	
	Power module
	UART
	I2C
	ADC
	12 PWM servo outputs
	PPM/S.Bus input
ICs	
	MPU9250 9DOF IMU
	LSM9DS1 9DOF IMU
	MS5611 barometer
	U-blox M8N GLONASS/GPS/Beidou (antenna connector type MCX)
	Cortex-M3 RC I/O co-processor
	RGB LED

4.6 Software

The software, or image processing pipeline (Fig. 4.6), required to implement the solution was written in Python and can be divided into three distinct stages:

1. Capture
2. Process
3. Control

The capture stage of the image processing pipeline refers to the image acquisition using the Raspberry Pi camera. The pre-processing stage begins with the creation of regions of interest and the application of a box filter. If the image was captured using a colour camera (in the lab setup), the image was converted from the Blue, Green Red (BGR) colourspace to a Hue, Saturation, value (HSV) colourspace before going through a thresholding and masking process to extract the green vegetation. If the image was captured using an IR camera (in the greenhouse environment), it was processed using the previously-described NDVI algorithm. After detecting green vegetation in the image, the multi-stage Canny edge detection algorithm was used to obtain edges in the image.

Finally, a Hough Line Transform function was used to obtain a list of line segments indicative of the Left Floor Line (Lf_l) and Right Floor Line (Lf_r) lines. The resulting line segments are further processed to obtain the x and y coordinates of the vanishing point (Vp_x and Vp_y), the intersection of the left and right floor lines. After performing the required geometric calculations required for the UAV navigation system, control signals were issued to the UAV's autopilot.

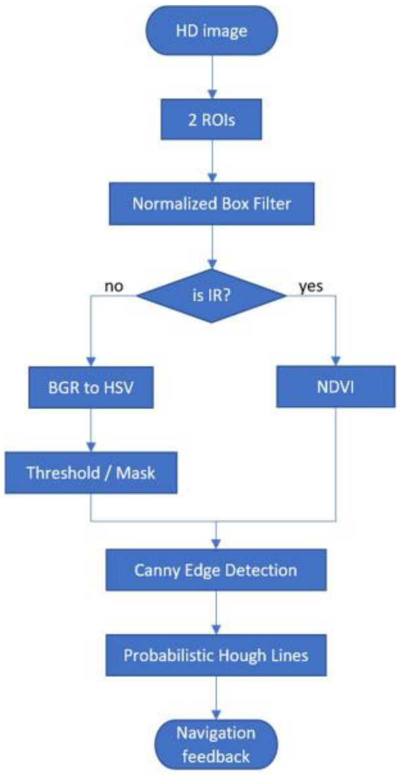


Figure 4.6 Image processing pipeline

4.6.1 Capture

The capture stage of the image processing pipeline refers to image acquisition using the Raspberry Pi camera. It also includes basic pre-processing. The `Picamera` library was used to control the camera's settings. Manual camera settings as described in Table 4.5 were used to ensure uniformity across all frames.

Table 4.5 Manual camera capture settings

Setting	Value
Resolution	HD
FPS	30
AWB mode	OFF
ISO	200
Exposure Mode	OFF
Shutter Speed	30,000

4.6.1.1 imutils Library

Various computer vision functions contained in the imutils library developed by Adrian Rosebrock (Ph.D) of PyImageSearch.com were used to improve and monitor capture performance on the Raspberry Pi. The `PiVideoStream` and `FPS` functions were used to improve capture performance and assisted with code efficiency estimation. The library also contains a series of convenience functions to make basic image processing functions such as translation, rotation, resizing, skeletonization, and displaying `Matplotlib` images easier with OpenCV and both Python 2.7 and Python 3.

4.6.2 Pre-Processing

Various steps were taken to pre-process captured images to obtain values necessary for UAV navigation. A high definition (HD) image was captured at approximately 20 frames per second (FPS). The pre-processing stage of the image processing pipeline included extracting two regions of interest (ROIs). A normalized box filter was applied to the ROI to remove any noise artifacts.

4.6.2.1 ROIs

Each captured frame was divided into four quadrants (Fig. 4.7, 4.8), and two regions of interest (ROIs) were extracted from each frame. The two ROIs were the bottom left and bottom right quadrants of the frame.

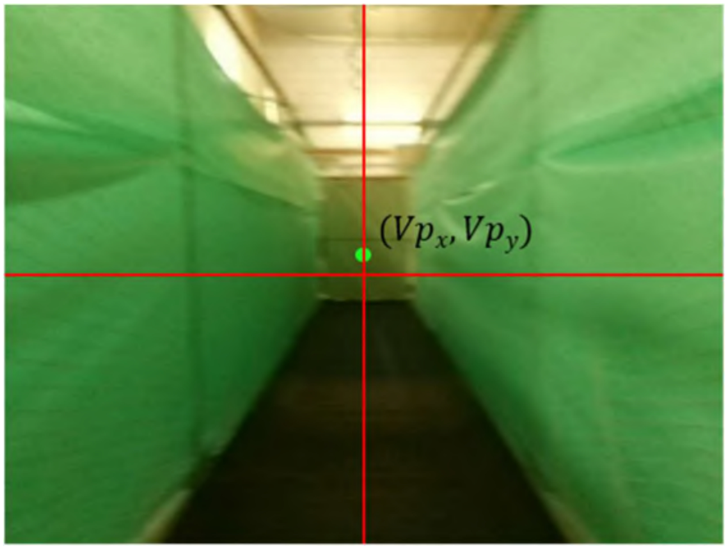


Figure 4.7 A captured frame divided into four ROIs and vanishing point

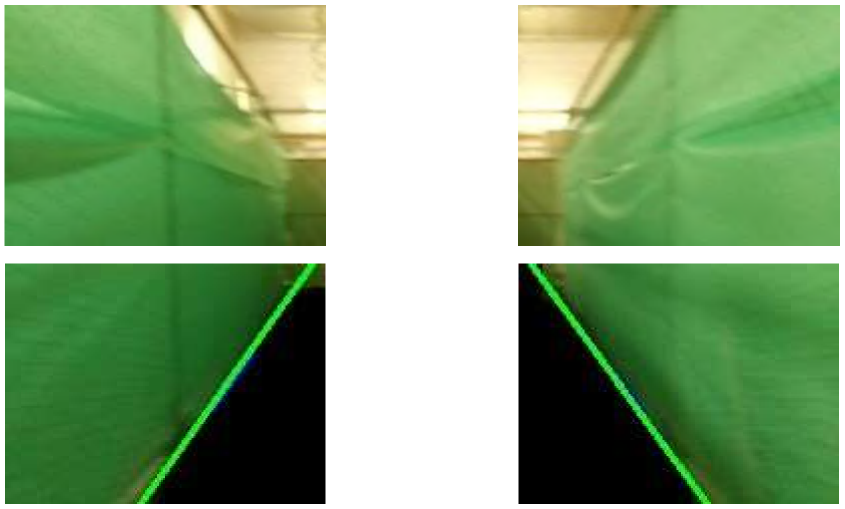


Figure 4.8 Four ROIs (bottom two ROIs were used in further image processing stages)

4.6.2.2 Normalized Box Filter

After selecting the regions of interest, a normalized box filter was applied to each ROI (Fig. 4.9). The role of a filter is too smooth an input image. The normalized box filter is the simplest filter used in image processing and is often used to remove artifacts; it is also commonly known as a blur filter. A bilateral filter was also evaluated for noise-removal, but ended up being too processor-intensive therefore the simpler normalized box filter was selected for the final algorithm.

$$K = \frac{1}{K_{width} \cdot K_{heigh}} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & 1 & 1 & \cdots & 1 \\ \cdot & \cdot & \cdot & \cdots & 1 \\ \cdot & \cdot & \cdot & \cdots & 1 \\ 1 & 1 & 1 & \cdots & 1 \end{bmatrix}$$

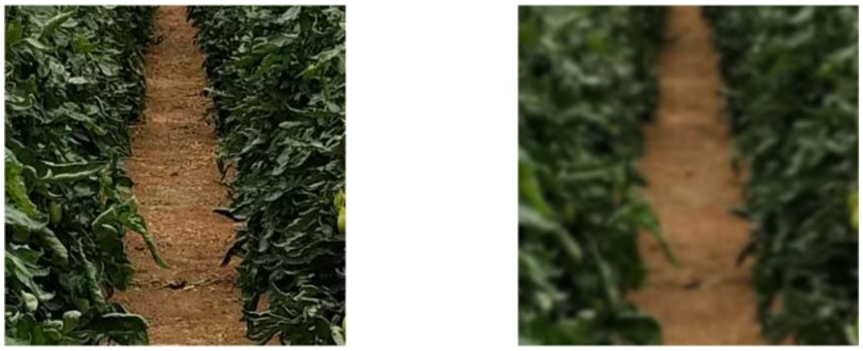


Figure 4.9 Before (left) and after application of a 10×10 normalized box filter (right)

4.6.3 Process

This section outlines the “Processing” stage, which is the main computer vision component of the project. As seen in Fig. 4.10, the captured camera image is split into its respective channels (Blue, Green, and Red / IR) before the NDVI being performed. It is important to note that the Red channel in the image corresponds to IR, due to the application of an optical bandgap filter. The NDVI formula uses the Red channel for IR and the Blue channel for visible light. The pipeline continued with a conversion to grayscale and application of a colourmap. A threshold was then applied to mask for vegetation. The resulting image was then supplied to the Canny Edge Detector and Hough Line transform for extraction of the relevant geometric properties.

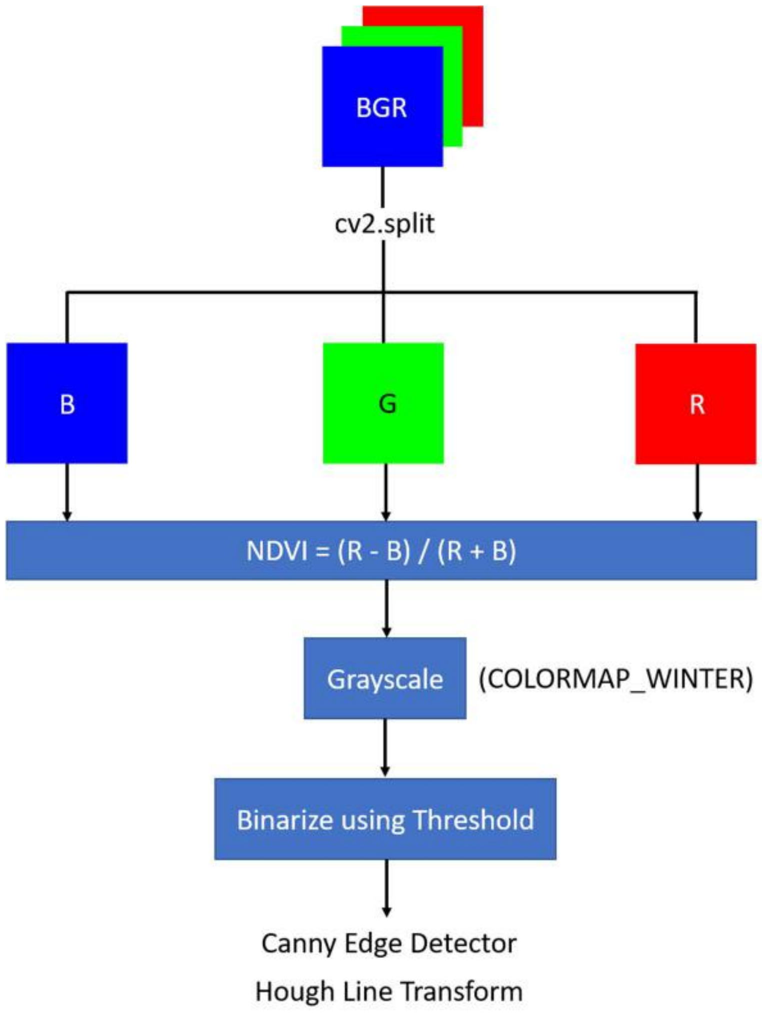


Figure 4.10 Processing stage (BGR to Hough Lines)

4.6.3.1 NDVI

Photosynthesis describes the process by which chlorophyll absorbs light, and uses this energy to drive a charge separation which ultimately generates oxygen and carbohydrate. Fig. 4.11 shows the absorption spectra of two types of chlorophyll. The NDVI algorithm relies on the fact that both Chlorophyll A and B absorb blue and red light, but not green or IR.

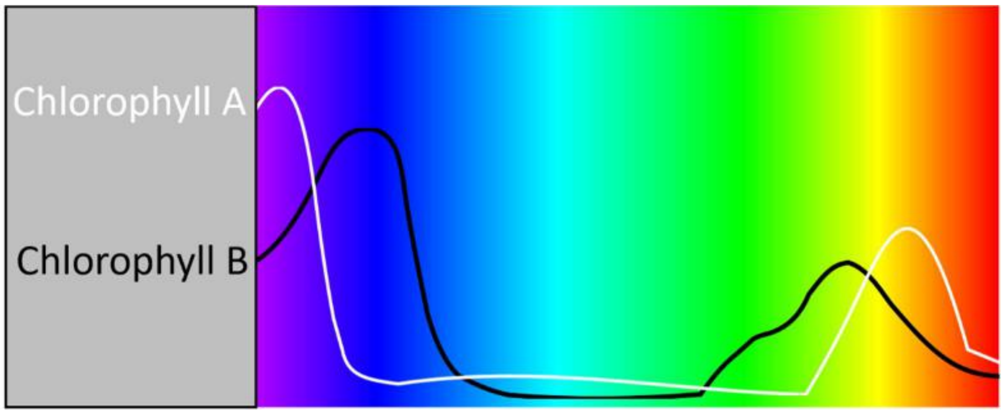


Figure 4.11 Absorption spectra of Chlorophyll A and B

Fig. 4.11 also explains why trees are green: because green is the only colour remaining once chlorophyll has absorbed the long wavelength (red) and short wavelength (blue) light. Greenness of a plant is a measure of how much photosynthesis is occurring, but an even better method is to measure IR and not blue. A filter is required to achieve this. Bright areas in a filtered image indicate increased photosynthetic activity.

This form of remote sensing to determine photosynthesis levels and vegetation health has a long-standing track record. The Landsat satellites, for example, capture images of the Earth's surface across a very broad spectrum. PublicLab has done much research as part of the Infragram project to develop methods for modifying COTS components to apply the same concepts using low-cost components. An alternative to expensive optical bandgap filters for the NDVI analysis is the Cinegel® R2007-Storaro Blue filter. Designed as a "gel" mainly used in the theatrical industry to colour spotlight beams, this low-cost material filters the electromagnetic spectrum in wavelengths suited for the NDVI application. Fig. 4.12 illustrates the transmission diagram of this filter across the visible light spectrum. The filter acts as a bandgap filter in the visible light range, while allowing infrared and wavelengths closer to the ultraviolet frequencies to pass.

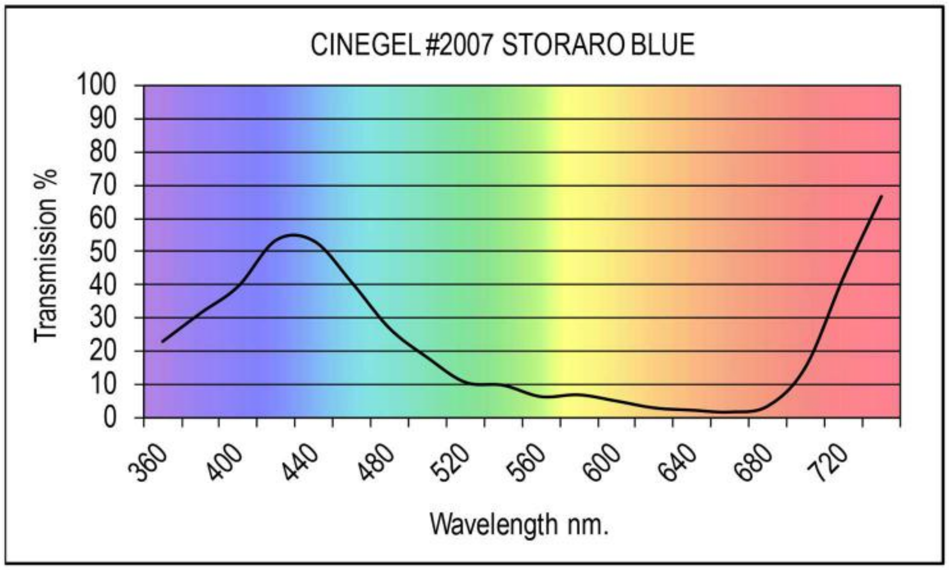


Figure 4.12 Transmission diagram of the Cinegel® R2007-Storaro Blue filter (© Rosco Laboratories 2018, by permission)

4.6.3.2 BGR to HSV

Once the NDVI algorithms had been implemented and tested, a second option was added to use the colour camera for indoor field tests. The BGR to HSV function was used to pre-process colour images to give similar results as NDVI images from the IR camera.

4.6.3.3 Threshold / Mask

Thresholding, also sometimes referred to as binarization, looks at pixel colour values, and if they are greater than a certain threshold value they are assigned one value, whereas if they are less than another value, they are assigned a different colour. The input is typically a greyscale image and the output is a binary coloured image. The binary image can be used to mask the original image: for example, every pixel with a value of 0 becomes the mask (Fig. 4.13).

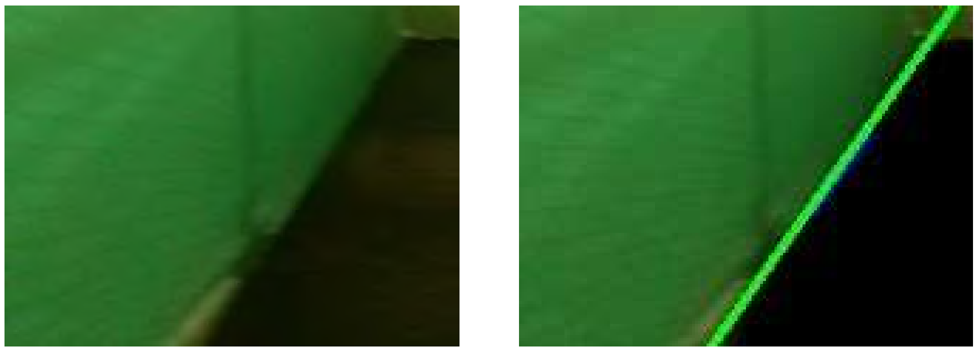


Figure 4.13 Before (left) and after thresholding for green and masking (right)

4.6.3.4 Canny Edge Detection

Canny edge detection was used to detect line segments making up the left floor line (Lf_l), right floor line (Lf_r), and vertical lines at the end of the corridor. Next, these disconnected edges (line segments) were passed through the Probabilistic Hough Lines function.

4.6.3.5 Probabilistic Hough Lines

The Probabilistic Hough Line function `HoughLinesP` was used to sort the located line segments into their respective categories, based on slope:

- left floor line (Lf_l)
- right floor line (Lf_r)
- left corridor end (vertical line)
- right corridor end (vertical line)

Once lines had been sorted into these categories, an average slope value was calculated to obtain best fit lines for Lf_l and Lf_r. These best-fit lines were then combined with the vanishing point coordinates to generate a geometric scene description of the UAV's environment.

4.6.4 Control

This section explains the software processes used to convert the geometric properties extracted from the computer vision pipeline to estimate the

UAV's pose within the greenhouse corridor, and generate respective control commands to the autopilot.

4.6.4.1 Geometry

As described in the "Theory" section of this chapter, a geometric scene description was developed to represent the UAV's pose using geometric properties extracted from the image frame (Fig. 4.14). These geometric properties allow the algorithm to determine the UAV's position and orientation within the corridor. Fused with IMU roll, pitch, and yaw information, this is used to generate corresponding control commands to be sent to the autopilot as control commands.

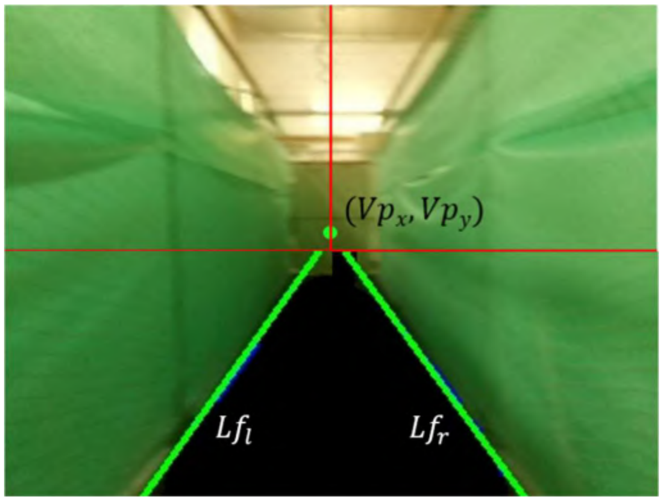


Figure 4.14 Geometric Scene Description

4.6.4.2 PID Controller

The ivPID library developed by IVMECH Mekatronik & Inovasyon was used to implement PID control in the project. This library is a simple implementation of a Proportional-Integral-Derivative (PID) Controller (Fig. 4.15) written in the Python Programming Language.

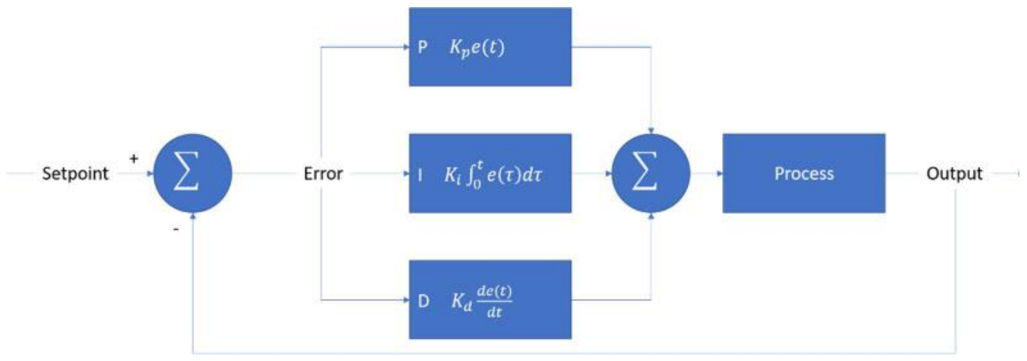


Figure 4.15 PID controller

4.6.4.3 DroneKit

The DroneKit-Python library was selected to send high-level control commands to the UAV autopilot. This library allows python applications to run on an onboard companion computer and communicate with the ArduPilot flight controller using a low-latency link. This allows the Raspberry Pi used for the NAVIO2 to serve the dual purpose of also performing image-processing and control commands, thereby making it the ultimate multi-purpose onboard processor. The DroneKit library is ideal for applications designed to add greater intelligence to vehicle behaviour, and for performing tasks that are computationally intensive or time-sensitive (for example, computer vision, path planning, or 3D modelling).

The DroneKit API communicates with the connected UAV's autopilot using the MAVLink protocol. It also provides programmatic access to the vehicle's telemetry, state and parameter information, and enables mission management and direct control over movement and operations. The API provides classes and methods to:

- Connect to a vehicle from a script
- Get and set vehicle state, telemetry, and parameter information
- Receive asynchronous notification of state changes
- Send arbitrary custom messages to control UAV movement and other hardware (GUIDED mode)
- Override RC channel settings

A combination of these features was used to read sensor and vehicle state information, issue vehicle control commands, and override RC channel values in this chapter.

4.6.5 Optimization

Various optimization techniques were evaluated to achieve a computer vision pipeline frame rate of 10 FPS at VGA resolution (Fig. 4.16).

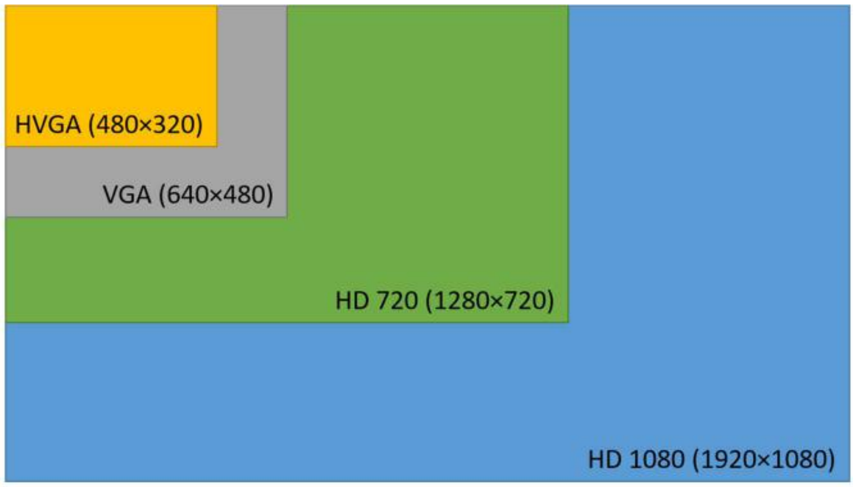


Figure 4.16 Common video resolutions used in this implementation

The first step to optimizing the computer vision pipeline involves capturing frames from a video stream instead of capturing single still images; this allows an increase in framerate. The `camera.capture` function was used to increase the complete line detection framerate (without `imshow`) from 3 FPS to 10 FPS at 320 × 480 resolution (HVGA). However, only 30% of the processor was used at this point.

It is important to remember that using `imshow` to display each captured frame significantly increases processor load and decreases processing time. Therefore, it is important to limit the use of `imshow` to design and troubleshooting uses only. Never use `imshow` on a real-time deployment. Most often, experimental deployments will run headless (without a monitor connected) during testing anyways. Using try-except statements automatically disables the `imshow` statements when no monitor is detected on the Raspberry Pi's HDMI output.

An analysis was performed to determine which stages in the image processing pipeline contributed to the processing time. If was discovered that the NDVI and Hough Line step (processing) takes ~.25s. Therefore, the fastest single-threaded processing is 4 FPS. The processing times for each stage are presented in Table 4.6 and illustrated with their contributions to overall performance in Fig. 4.17.

Table 4.6 Average processing times

Processing step	Processing time (s)
NDVI	0.1
Hough Lines	0.12
Canny Edge Detection	0.05

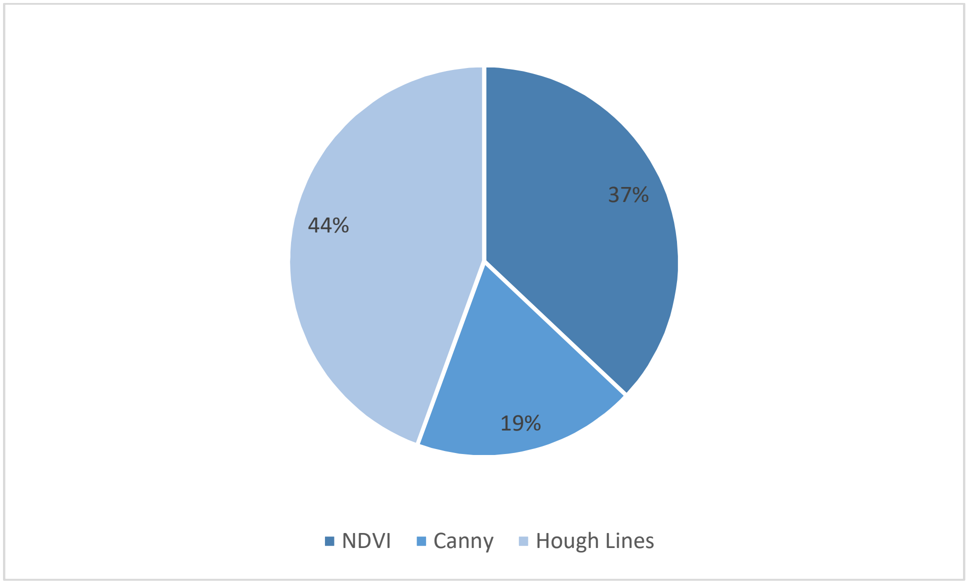


Figure 4.17 Performance time of major steps in the image processing timeline

These performance measurements were taken at VGA resolution (640 × 480 pixels). With this information in hand, it was possible to investigate further methods for improving overall performance in the image processing pipeline. This was done using Multiprocessing to make use of all four cores on the Raspberry Pi for image processing, and implementing Multithreading for the various components of the application. This problem can easily be split into three distinct sections for multithreading: capture, process, and control.

4.6.5.1 Multiprocessing

The next step was to implement multiprocessing to make use of all four processor cores on the Raspberry Pi. This was achieved using `PiVideoStream` and `FPS` from the `imutils` library, and using `Process` and `Queue` from the `multiprocessing` library. Some pros and cons of multiprocessing are outlined in Table 4.7.

Table 4.7 Pros and cons of Python multiprocessing

Pros	Cons
Separate memory space	More complicated instructions per clock cycle
Code is usually straightforward	Larger memory footprint
Takes advantage of multiple CPUs & cores	
Avoids Global Interpreter Lock for cPython	
Eliminates most needs for synchronization primitives	
Child processes are interruptible	
Python multiprocessing module includes an interface for abstraction	

The Python code was adopted to incorporate the Raspberry Pi's four CPU cores. This resulted in an improved average framerate of 8.8 FPS. Despite the increased framerate, it was discovered that the input and output queues required for multiprocessing frames still contributed to visible lag in the `imshow` display. Queuing images for too long could also cause issues for real-time control of the UAV. This was addressed by decreasing queue sizes.

4.6.5.2 Multithreading

Multithreading was used to further enhance and streamline the entire application. Some pros and cons of multithreading are outlined in Table 4.8.

Table 4.8 Pros and cons of Python multithreading

Pros	Cons
Low memory footprint	cPython is subject to Global Interpreter Lock
Shared memory	Not interruptible
C extension modules which properly release the Global Interpreter Lock will run in parallel	Manual use of synchronization primitives become a necessity If not following a command queue/message pump model (using the Queue module).
Great option for I/O-bound applications	Code is usually harder to understand - the potential for race conditions increases dramatically

The multi-tiered software architecture separated into capture, process, and control steps lent itself well to a multi-threaded approach. Three separate threads were created: one to capture images, a second to process the captured frames, and the third to output control commands to the UAV autopilot.

Table 4.9 Capture, process, control workflow used to create the multithreaded application

Capture	Captures frames from the camera stream
Process	Processes each frame using the computer vision pipeline
Control	Sends high-level control commands to the position controller

It was important to remember while optimizing this software, that the NAVIO2 autopilot also uses the Raspberry Pi's processor for mission-critical low-level autopilot control of the vehicle. It was imperative to allocate sufficient resources to the NAVIO autopilot while streamlining the computer vision software, to avoid locking up the processor, resulting in a loss of control over the vehicle.

4.6.6 Logging

Data logging functionality was added to the python application. Every experimental test automatically saved each original frame captured from the camera, as well as the processed left and right ROIs. The logger generates a CSV file containing a timestamp, distance measurement, the PID output value, and the yaw command issued to the autopilot. A sample output from the data logger is provided below:

Table 4.10 Sample output from data logger

Time:	12:35.2
Distance:	-17.8736
PID output:	35.74712
Yaw Command:	1545.747

4.7 Experimental Methodology

4.7.1 NDVI - Outdoor and Greenhouse Tests:

Tests of the NDVI image processing algorithm were conducted on videos recorded on the grounds of the Marcus Family Campus of BGU. Once the algorithm was fine-tuned, additional sample videos were collected inside a commercial tomato greenhouse. These videos were later used during simulations to develop and test the UAV control algorithm. Sample images obtained in a commercial tomato greenhouse are shown below (Fig. 4.18, 4.19).

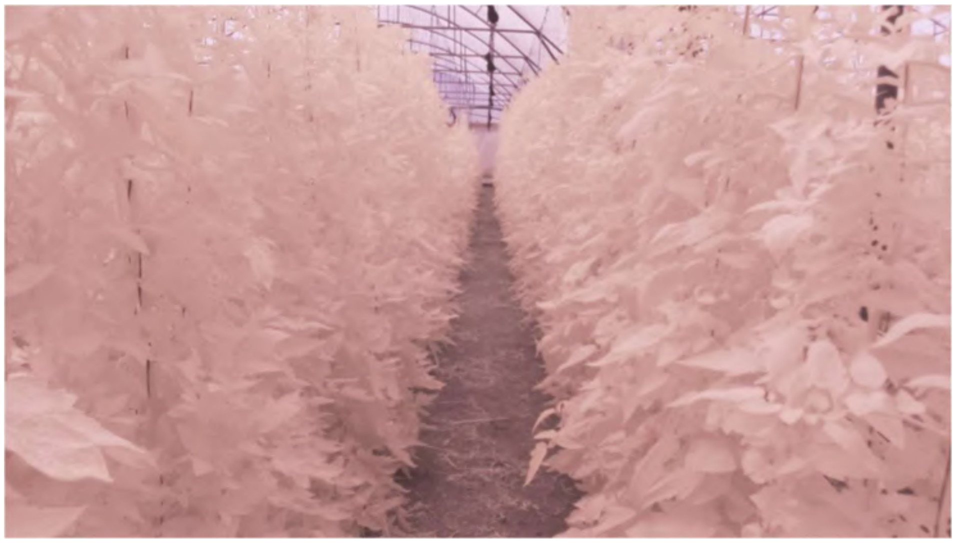


Figure 4.18 Standard IR image

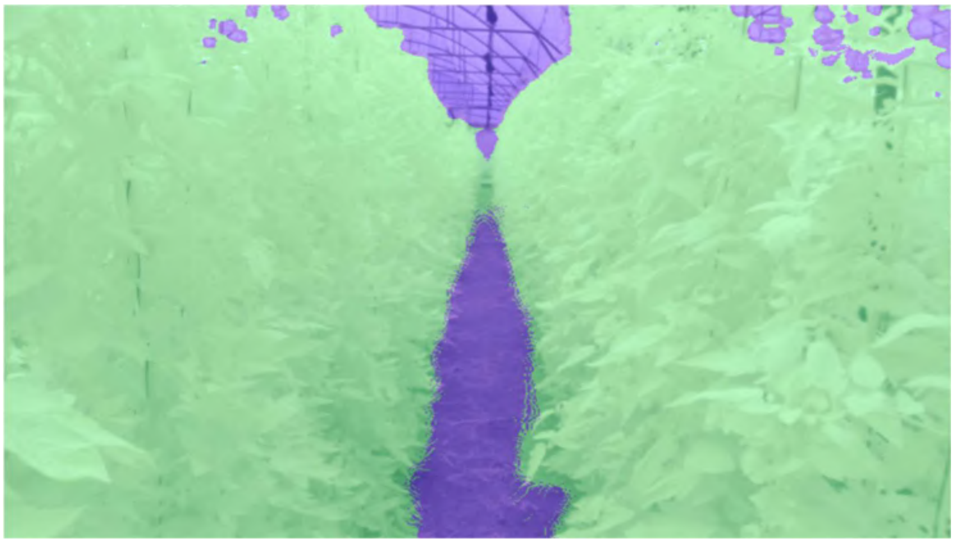


Figure 4.19 NDVI processed image (green is vegetation, blue is non-vegetation) overlaid on original image

4.7.2 Simulation

Once the NDVI algorithm had been developed, work began on simulating the geometrical properties required to navigate within a greenhouse row.

MATLAB and Simulink were used to model the geometry (Fig. 4.20) and controller (Fig. 4.21).

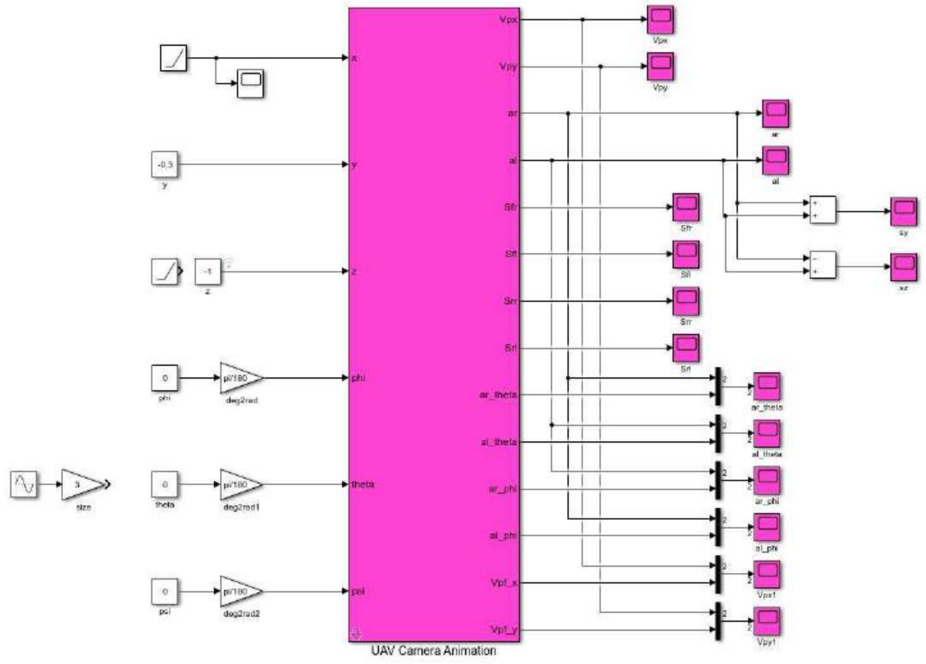


Figure 4.20 UAV controller

Fig. 4.20 shows the UAV camera animation used to simulate the greenhouse row in preparation for development of the computer vision navigation system. Input parameters include the UAV's pose parameters *x, y, z, φ, θ, ψ*. The UAV Camera Animation block is expanded in more detail in Fig. 4.21. Parameters include sample time (framerate), camera focal length, sensor width, sensor height, UAV width, UAV length, row width, row height, and row length. Vision-algorithm outputs include Vp_x (vanishing point x-coordinate), Vp_y (vanishing point y-coordinate), a_r (slope of right line), and a_l (slope of left line). These values generated outputs for the vision-based navigation algorithm including *ar_theta* (yaw of UAV based on right line), *al_theta* (yaw of UAV based on slope of left line), *ar_phi* (roll of UAV based on slope of right line), and *al_phi* (roll of UAV based on left line).

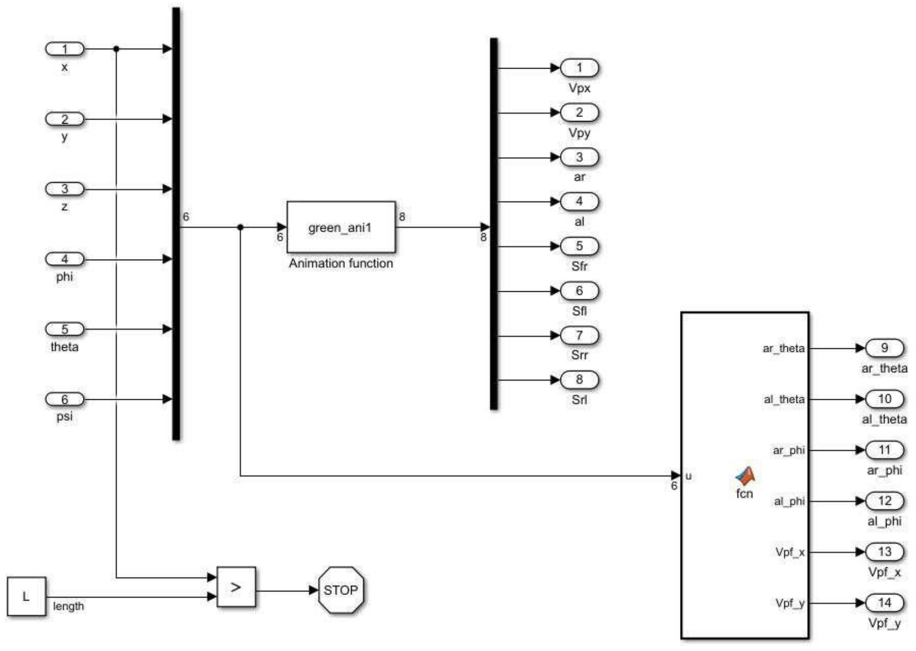


Figure 4.21 Geometry simulation

Fig. 4.22 shows the UAV positioned at the end of a greenhouse row at the start of the animation. As the animation progressed, sample images (Fig. 4.23-4.34) were generated to represent the camera view.

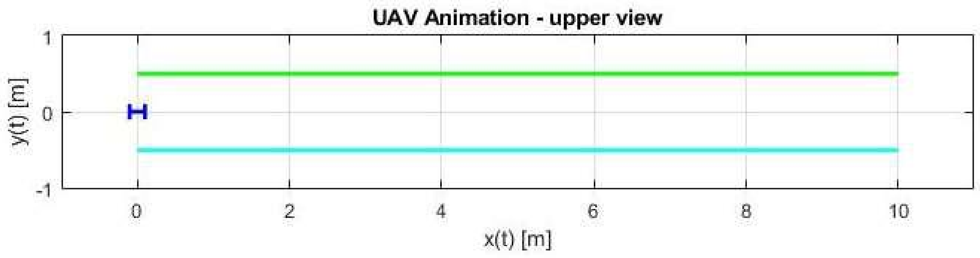


Figure 4.22 UAV simulated in greenhouse corridor

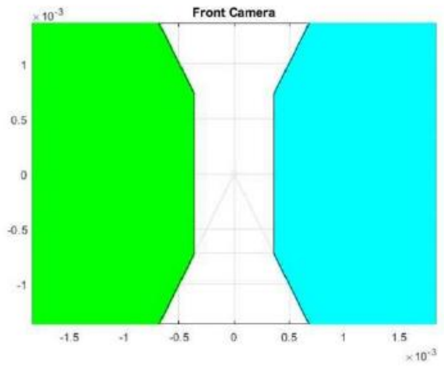


Figure 4.23 x = 0

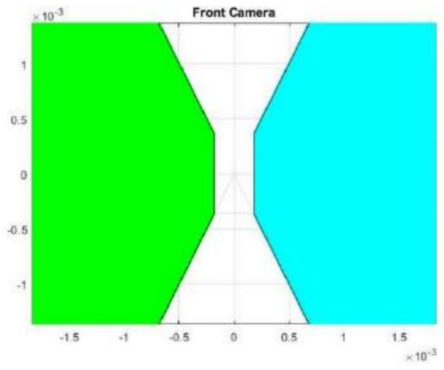


Figure 4.24 x = 5

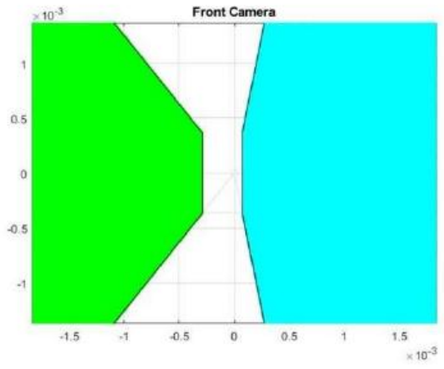


Figure 4.25 y = 0.3

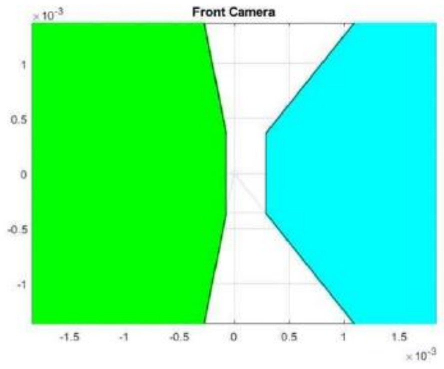


Figure 4.26 y = -0.3

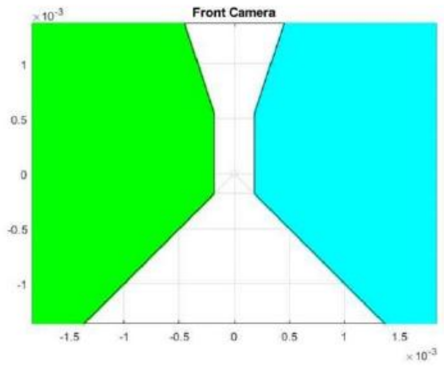


Figure 4.27 z = -0.5

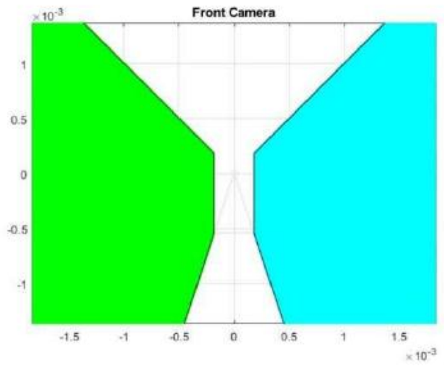


Figure 4.28 z = -1.5

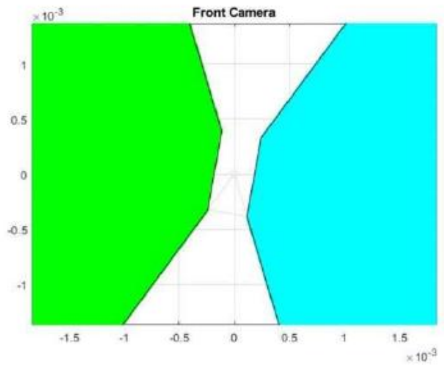


Figure 4.29 Roll (φ=-10)

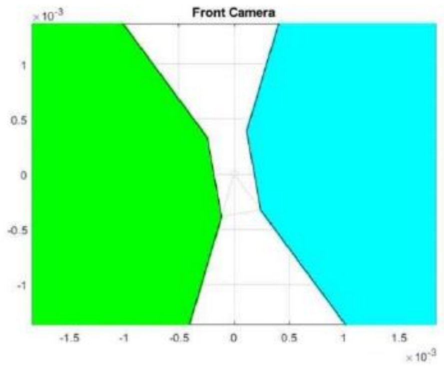


Figure 4.30 Roll (φ=10)

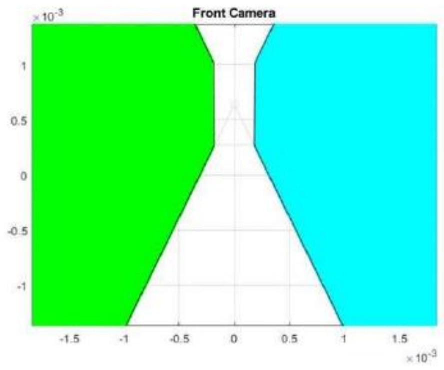


Figure 4.31 Pitch (θ=-10)

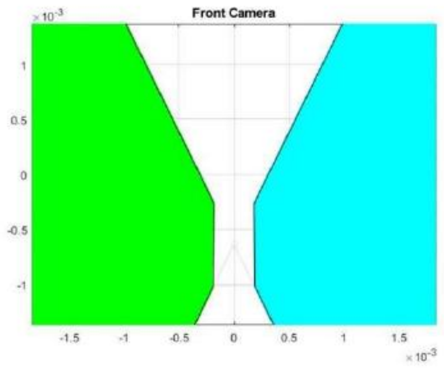


Figure 4.32 Pitch (θ=10)

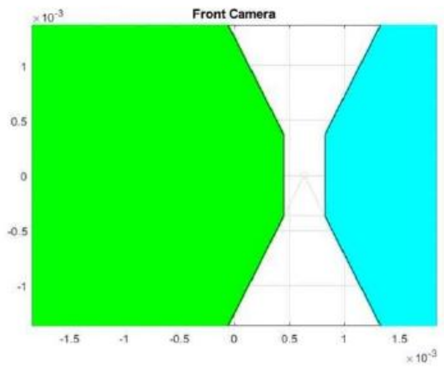


Figure 4.33 Yaw (ψ=-10)

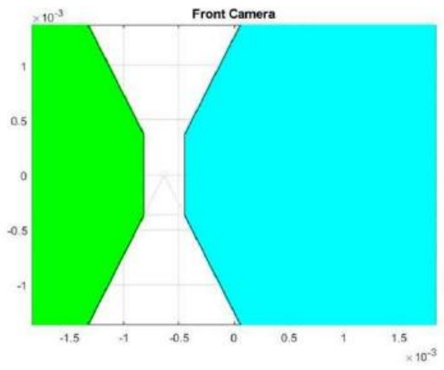


Figure 4.34 Yaw (ψ=10)

4.7.3 Construction of Indoor Test Arena:

After simulating the NDVI algorithm, an indoor lab setup was required to develop and test the control system. An aluminum frame was constructed from modular T-slotted aluminum extrusion profiles as the basis for a 7.2 m long corridor to perform lab tests. The corridor was 1.1 m in width and 1.5 m in height. The inside of the corridor was fitted with netting to provide a safe indoor test environment. Green plants (Fig. 4.35) were simulated by applying green fabric to the sides of the corridor (Fig. 4.36). The floor was lined with matte black paper, as the polished floor reflected the green fabric, causing difficulties for the thresholding and detection algorithm. This issue would not occur in an actual greenhouse environment, as the tests of the NDVI algorithm have demonstrated.

Table 4.11 Dimensions of the simulated indoor experimental greenhouse row

Dimensions (m)	
Length	7.2
Width	1.1
Height	1.5

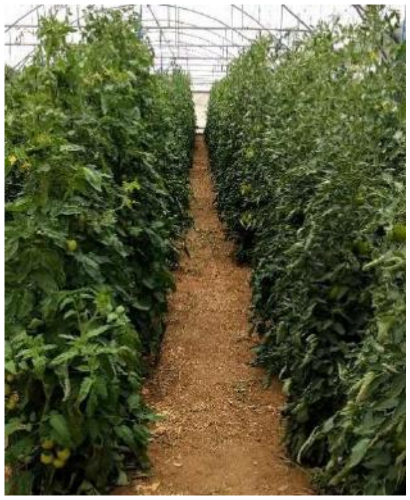


Figure 4.35 Row of tomato plants in greenhouse

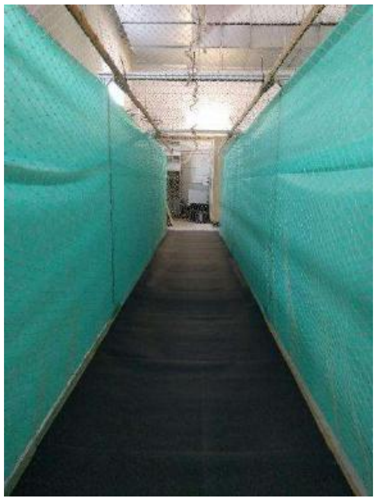


Figure 4.36 Simulated greenhouse row in the experimental lab setup

4.7.4 Calibration of Autopilot Sensors:

The quadrotor was re-calibrated on a level surface and sensor calibration was double-checked using the BevelBox Digital Angle Protractor. The vehicle's state was read using DroneKit. vehicle.attitude returns roll and pitch in float values ranging from -1.5 to +1.5. For example, an angle of 45° returns a value of 0.75 in DroneKit. These scaled attitude values were compared with the angles given by the BevelBox. They were found to be within accurate to 0.1 degree, the accuracy of the BevelBox gauge.

4.7.5 Zero-yaw Test Rig

To examine the efficacy of the computer vision algorithm for navigation purposes, a one degree-of-freedom (DOF) testbed was constructed, which allowed the vehicle to rotate freely about its y-axis (yaw). The test stand consisted of a vertical post with a ball-bearing attached at the top. The UAV's center of mass was attached to the ball-bearing, thereby allowing the UAV to rotate freely about the z-axis (yaw) with minimal friction. Fig. 4.37 shows the UAV mounted on the test stand, approximately one meter off the ground, at the end of the simulation greenhouse row, and centered laterally within the row. The cable visible in the picture was used to connect a monitor for debugging purposes; it was disconnected while experimental tests were conducted.

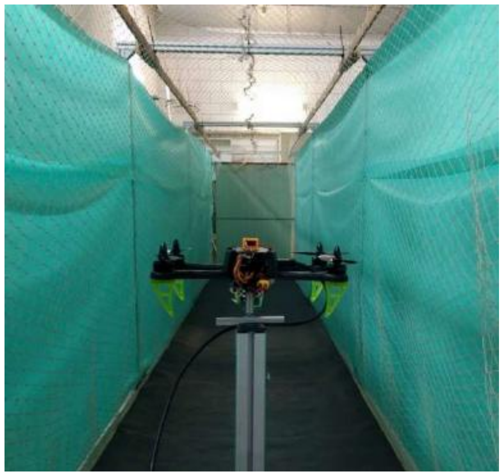


Figure 4.37 UAV mounted atop the 1-DOF test setup in the lab simulation environment

4.7.6 PD Controller

A proportional controller was implemented to test the closed-loop image-based navigation system. The ivPID library provided my ivmech was used for this purpose. Initially, a simple Proportional controller was evaluated; the results were not satisfactory due to large overshoots, so a Derivative term was added. This resulted in fast action with minimum overshoot. Since a PID library was used to implement this control system, the additional Integral term can be added in future, if desired.

The UAV was mounted on the testbed inside the lab greenhouse corridor, and could rotate freely. Initial offsets of varying acute angles were provided, and the controller was used to minimize the error (yaw).

4.8 Results and Discussion

Success of the image-based algorithm was defined to have been reached if the system recovered from an acute angle disturbance, and minimized the vehicle's yaw angle with a stable response and minimal overshoot. The results in Fig. 4.38 demonstrate the efficacy of the image-based visual servoing algorithm for one DOF in the indoor test setup. The graph shows the minimization of a 93-pixel image error, corresponding to an angle of 9°, in 1.4 seconds with minimal overshoot; a successful implementation of the control system to minimize yaw. This is a representative graph of a typical test result; overall, ten such tests were conducted in rapid succession, all of which managed to zero the yaw angle with a similar response. During these tests, results were collected for 60 seconds, and multiple disturbances were introduced manually, by misaligning the UAV, and allowing the vision-based control system to correct the disturbance.

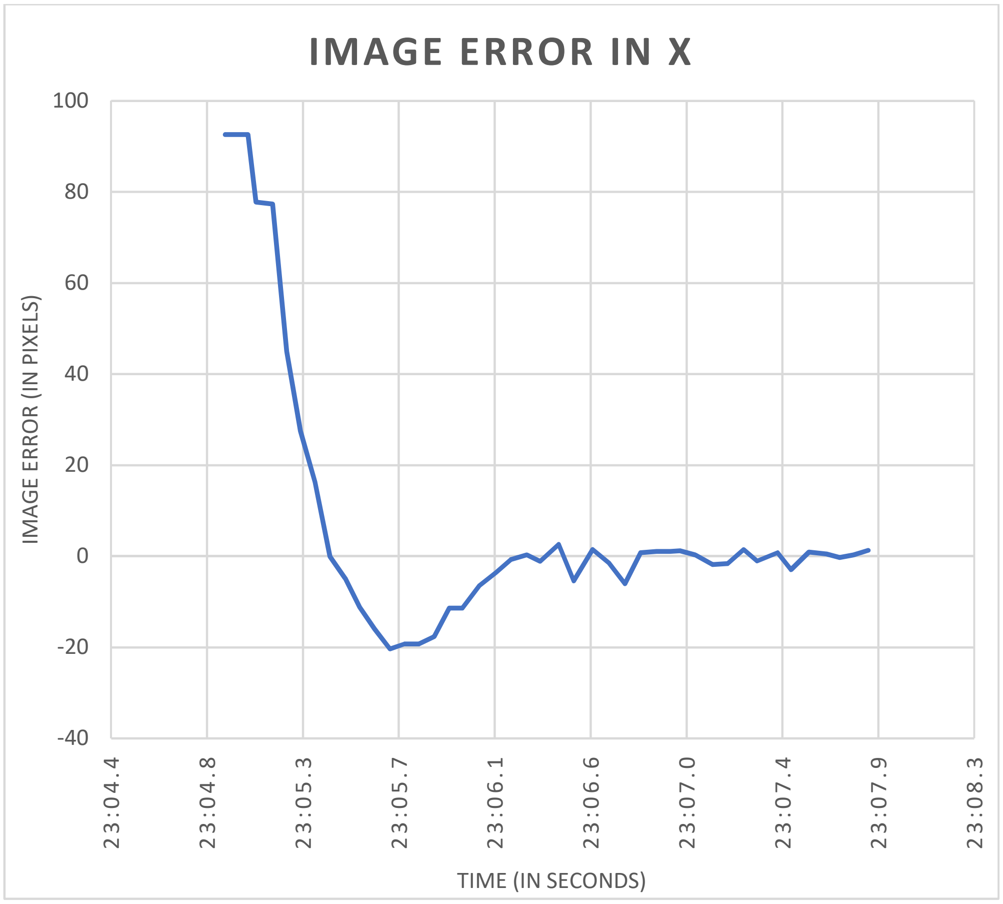


Figure 4.38 Results of image-based servoing controller minimizing yaw

The minimization of the vehicle's yaw angle within the corridor using computer vision demonstrates the possibility of further expansion to 6 DOF. The developed image processing pipeline successfully differentiates between vegetation and non-vegetation, and image features were successfully and reliably extracted to aid in navigation within the greenhouse row. A closed-loop controller was successfully tested for one degree-of-freedom in a laboratory setting.

Not only can these results be applied to plants in a greenhouse, but the robust algorithm presented and tested promises functionality for any type of vegetation planted in parallel rows. The presented system can be applied to most standard greenhouse setting, and with minimal adjustments could also be used in outdoor agriculture applications. The structure of vertically growing plants, for example grapevines in outdoor viticulture, closely resemble the experimental setup tested here. It would not be too unrealistic, therefore, to apply these concepts to aerial platforms in these fields. A small micro aerial vehicle using IR cameras, for example can be used to investigate plant health on a micro-scale in such an application. The results obviously extend far beyond this application, and present promise for future implementations and iterations.

Chapter 5: Concluding Chapter

5.1 Conclusions

In this thesis, the challenge of vision-based landing and navigation of quadrotor UAVs in an indoor environment was presented. The work provides an outline to computer vision methods and the quadrotor vehicle. The growing UAV industry, as well as the challenges associated with manual UAV landings are explained. The industry's move towards auto-landing, and the challenges associated with such systems is established. Research on active and passive landing methods for fixed-wing and quadrotor UAVs are presented. Many methods still rely on a communication link between the ground station and the vehicle; the presented system attempts to remove the necessity for this communication link to create a system requiring minimal setup and infrastructure. The proposed system only requires a simple IR beacon as a landing pad marker, and uses passive IR for navigation purposes in the GPS-denied environment.

Chapter 3 presented an IR landing system for multirotor UAVs in GPS-denied and vision-compromised environments. A new type of IR beacon with possibilities for uniquely modulated carriers was evaluated for use in indoor and outdoor applications. This beacon was successfully detected and used by the vision-based servoing algorithm to land the UAV during indoor experiments. The application of this easily deployable system to outdoor environments and slow-moving targets is being further evaluated.

Chapter 4 demonstrated the feasibility of using the NDVI and geometric properties of a row of plants for indoor navigation. Not only can these results be applied to plants in a greenhouse, but the robust algorithm presented and tested promises functionality for any type of vegetation planted in parallel rows. The presented system can be applied to most standard greenhouse settings, and with minimal adjustments could also be used in outdoor agriculture applications. The structure of vertically growing plants, for example grapevines in outdoor viticulture, closely resemble the experimental setup tested here. It would not be too unrealistic, therefore, to apply these concepts to aerial platforms in these fields. The use of infrared cameras and the NDVI for navigation can be extended to investigate plant health. This dual use could be critical for applications restricted to small-scale drones. The results also show the potential use of modern cost-effective equipment for precision agriculture.

5.2 Strengths and Limitations

To investigate the applicability of any system to a certain field, it is important to recognize its strengths and limitations. The vision-based landing and navigation methods presented in this thesis have very specific applications and therefore strict boundary conditions, however, the computer vision methods used to implement them remain universal. Chapter 3 presented an IR vision-based landing system for a quadrotor UAV in an indoor GPS-denied environment. The system was shown to be well-suited for a stationary landing platform, but the presented literature also suggests the same methodology can be applied to a slow-moving target. The UAV would have to use alternate navigation methods to travel to the general landing area before the IR camera can detect the beacon and take control of the aircraft. The use of an IR camera makes the system an ideal choice in environments where visibility is restricted by smoke or fog. The presented system uses a heuristic method of landing, whereby UAV orientation is disregarded, thereby simplifying the ground station setup. If UAV orientation is considered important, and more complicated ground station consisting of multiple beacons would be required. Overall, the landing system is designed for ease-of-use and rapid deployment, in an environment where a wireless communication link between ground station and UAV is not desirable.

Chapter 4 presented a vision-based navigation method based on IR computer vision and the NDVI method for navigation in agricultural applications. The computer vision pipeline was proven to be capable to differentiating between vegetation and non-vegetation. The demonstrated system was experimentally proven in one DOF. The experimental tests, however, were conducted in a controlled environment, and a variety of edge cases will need to be addressed. The current implementation required both floor lines to be visible in the image, to be successful. These edge cases will need to be addressed in a more robust implementation. NDVI was selected as the method of choice, as it is a popular payload on agricultural UAVs, and because of its robustness in detecting vegetation. A system using both NDVI and colour vision could be implemented for more widespread applicability; the IR and colour camera could be fused using methods such as the Dempster-Shafer Theory to increase accuracy and reliability in detecting vegetation. Overall, the demonstrated system was designed to be a standalone system, which uses the UAV's standard payload for navigation purposes, without requiring any external setup or additional expenses.

5.3 Future Work

Chapter 3 presents a system for automated landing of UAVs, but various enhancements are possible, and recommended. The first would be an enhanced "handshake" between the IR source and the vehicle. The radiator selected for the work has the added benefit of incorporating a modulation input. This allows the carrier signal to send a specific modulated "handshake" message to the UAV. This can prevent false detections from other IR sources, and effectively established a communication link between the landing pad and the UAV without requiring any additional setup or wireless telemetry modem. This would eliminate the possibility of false alarms.

The groundwork completed in Chapter 4 proves the feasibility of using small drones to pollinate tomato plants in a greenhouse setting using vision-based navigation. Future work requires the one DOF lab test rig to be expanded to include multiple degrees of freedom. Once all DOFs have been achieved, the UAV can be taken to a greenhouse for a field test. The vison-based algorithm may require some additional pre-processing and filtering steps to compensate for noisy data associated with a real-world setting. However, the precursors to this have already been tested, and the algorithm's robustness has been proven. Further experiments and results are also being compiled for publication.

Further work on the project also seeks to incorporate dual Raspberry Pi cameras into the navigation algorithm. One camera would be facing forward (as demonstrated in this paper), while the other faced backwards. This would allow the UAV greater accuracy in position measurements, and specifically in estimation of the longitudinal position within the greenhouse row. The feasibility of this concept has already been tested as part of this project in simulation. Hardware to accommodate dual cameras on-board the UAV was also tested, and the IVPort Dual Raspberry Pi Camera Module Multiplexer developed by Ivmech was successfully tested. Multiplexing the camera feed will reduce the framerate by half, as well as minimal time required to switch the input using a GPIO pin. Using the described computer vision pipeline, this would result in an approximately 5 PFS for the computer vision algorithm, which is quite slow, but likely sufficient for a high-level position controller at low flight speeds. Additional optimization techniques, as well as constant improvements in computing hardware will likely make this a reality very soon.

In conclusion, the work presented in this thesis demonstrates that the field of computer vision in unmanned vehicle applications promises to be an exciting field. The miniaturization and cost reduction experienced by the electronics sector over recent years has made many new technologies possible which would have been out of reach for most commercial and consumer applications. This research, as well as future work on the horizon, demonstrates that smart electronics are not far into the distant future, and continual advances will bring many positive changes to the industry.

References

[1] E. Nowak, K. Gupta, and H. Najjaran, “Development of a plug-and-play infrared landing system for multirotor unmanned aerial vehicles,” In Proc. 14th Conf. on Comput. and Robot Vision, 2017. doi: 10.1109/CRV.2017.23

[2] H. Efraim, S. Arogeti, A. Shapiro, and G. Weiss, “Vision based output feedback control of micro aerial vehicles in indoor environments,” *J Intell Robot Syst*. vol. 87, no. 1, Jul., pp. 169-186, 2017. doi: 10.1007/s10846-017-0510-0

[3] “Advisory Circular (AC) No. 600-004 - Transport Canada.” [Online]. Available: https://www.tc.gc.ca/eng/civilaviation/opssvs/ac-600-004-2136.html#s1_2. [Accessed: 15-Jan-2018].

[4] “Leading UAV regulation - Vanguard Magazine.” [Online]. Available: http://www.vanguardcanada.com/2014/11/05/leading-uav-regulation/. [Accessed: 05-Dec-2015].

[5] V. Baiocchi, D. Dominici, M. V. Milone, and M. Mormile, “Development of a software to optimize and plan the acquisitions from UAV and a first application in a post- seismic environment,” *European J of Remote Sensing* vol. 47, Apr., pp. 477–496, 2014. doi: 10.5721/EuJRS20144727

[6] S. Bogatov, N. Mazny, A. Pugachev, S. Tkachenko, and A.Shvedov, “Emergency radiation survey device onboard the UAV,” *Int Archives of the Photogrammetry, Remote Sensing and Spatial Inf Sci* vol. XL-1/W2,

Sept., pp. 51–53, 2013. doi: 10.5194/isprsarchives-XL-1-W2-51-2013

[7] “Drones in humanitarian action” Swiss Foundation for Mine Action (FSD) [Online]. Available: http://drones.fsd.ch/wp-content/uploads/2016/11/Drones-in-Humanitarian-Action.pdf [Accessed: 05-Dec-2015].

[8] “Unmanned aircraft systems (uas) frequently asked questions.” [Online]. Available: https://www.faa.gov/uas/faqs/. [Accessed: 15-Jan-2018].

[9] G. J. Verhoeven, “Near-infrared aerial crop mark archaeology: from its historical use to current digital implementations,” *J Archaeol Method Theory*. vol. 19, Mar., pp. 132–160, 2012. doi: 10.1007/s10816-011-9104-5

[10] A. Y. Lin, A. Novo, S. Har-noy, N. D. Ricklin, and K. Stamatiou, “Combining geoeye-1 satellite remote sensing, UAV aerial imaging, and geophysical surveys in anomaly detection applied to archaeology,” *IEEE J. of Sel. Topics in Appl. Earth Obs. and Remote Sens.* vol. 4, no. 4, Dec., pp. 870–876, 2011. doi: 10.1109/JSTARS.2011.2143696

[11] S. Lee and Y. Choi, “Reviews of unmanned aerial vehicle (drone) technology trends and its applications in the mining industry,” *Geosystem Eng.*, vol. 19, no. 4, Nov., pp. 187–204, 2016. doi: 10.1080/12269328.2016.1162115

[12] L. Lin, M. Roscheck, M. a Goodrich, and B. S. Morse, “Supporting wilderness search and rescue with integrated intelligence: autonomy and information at the right time and the right place,” In Proc. Twenty-Fourth

AAAI Conf. Artif. Intell., 2010, pp. 1542–1547.

[13] V. B. Hammerseth, “Autonomous unmanned aerial vehicle in search and rescue,” M.S. thesis, Norwegian University of Science and Technology, Trondheim, Norway, 2013.

[14] M.G. Wing, J. Burnett, J. Brungardt, D. Dobler, V. Cordell, and J. Sessions, "Search and rescue operations with an unmanned helicopter," *Int. J. of Remote Sensing Appl.* vol 6, no. 65. Jan., 2016. doi: 10.14355/ijrsa.2016.06.007

[15] M. A. Goodrich, J. L. Cooper, J. A. Adams, C. Humphrey, R. Zeeman and B. G. Buss, "Using a mini-UAV to support wilderness search and rescue: practices for human-robot teaming," *2007 IEEE Int. Workshop on Safety, Security and Rescue Robotics*, Rome, 2007, pp. 1-6. doi: 10.1109/SSRR.2007.4381284

[16] F. Esposito, G. Rufino, and A. Moccia, “Real-time detection of fire hotspots from mini-UAV based, thermal infrared / VIS-NIR hyperspectral image data,” In Proc. AIAA Infotech@Aerospace Conference, April, pp. 1–18, 2009. doi: 10.2514/6.2009-1980

[17] V. G. Ambrosia, S. S. Wegener, D. V. Sullivan, S. W. Buechel, S. E. Dunagan, J. A. Brass, J. Stoneburner, and S. M. Schoenung, “Demonstrating UAV-acquired real-time thermal data over fires,” *Photogramm. Eng. Remote Sens.*, vol. 69, no. 4, Apr., pp. 391–402, 2003. doi: 10.14358/PERS.69.4.391

[18] N. V Hoffer, C. Coopmans, A. M. Jensen, and Y. Chen, “Small low-cost

unmanned aerial vehicle system identification: a survey and categorization," In Proc. 2013 Int. Conf. on Unmanned Aircraft Syst.*,* 2013, pp. 897–904.

[19] M. Laiacker, M. Schwarzbach and K. Kondak, "Automatic aerial retrieval of a mobile robot using optical target tracking and localization," *2015 IEEE Aerospace Conference*, 2015, pp. 1-7. doi: 10.1109/AERO.2015.7118992

[20] C. S. Sharp, O. Shakernia and S. S. Sastry, "A vision system for landing an unmanned aerial vehicle," In Proc. IEEE International Conference on Robotics and Automation, 2001, vol.2., pp. 1720-1727. doi: 10.1109/ROBOT.2001.932859

[21] R. Whitehouse, "Autolanding UAVs: vision for the future," Romsey, UK: Roke Manor Research Ltd., 00371, 2010. [Online] Available: http://www.roke.co.uk/resources/white-papers/0371-Autoland-WP.pdf [Accessed: 05-Dec-2015].

[22] P. Haridasan and A. James, "Vision based algorithm for automatic landing system of unmanned aerial vehicles: a review," *Int. J. of Innovative Sci., Eng. & Tech.* vol. 2, no. 4, pp. 1126–1129, 2015.

[23] J. P. How, C. Fraser, K. C. Kulling, L. F. Bertuccelli, O. Toupet, L. Brunet, A. Bachrach, and N. Roy, "Increasing autonomy of UAVs," *IEEE Robot. Autom. Mag.*, vol. 16, no. 2, pp. 43–51, 2009.

[24] D. Jenkins and B. Vasigh, "The economic impact of unmanned aircraft systems integration in the United States," Association for Unmanned

Vehicle Systems Intl. March, pp. 1–40, 2013.

[25] A. Gautam, P. B. Sujit, and S. Saripalli, “A survey of autonomous landing techniques for UAVs,” In Proc. 2014 Int. Conf. Unmanned Aircr. Syst., pp. 1210–1218, 2014. doi: 10.1109/ICUAS.2014.6842377

[26] W. Kong, D. Zhou, D. Zhang and J. Zhang, "Vision-based autonomous landing system for unmanned aerial vehicle: a survey," In Proc. 2014 International Conference on Multisensor Fusion and Information Integration for Intelligent Systems (MFI), Beijing, 2014, pp. 1-8. doi: 10.1109/MFI.2014.6997750

[27] G. Xu, Y. Zhang, S. Ji, Y. Cheng, and Y. Tian, “Research on computer vision-based for UAV autonomous landing on a ship,” *Pattern Recognit. Lett.*, vol. 30, no. 6, pp. 600–605, 2009. doi: 10.1016/j.patrec.2008.12.011

[28] C. D. McGillem and T. S. Rappaport, "A beacon navigation method for autonomous vehicles," in *IEEE Transactions on Vehicular Technology*, vol. 38, no. 3, pp. 132-139, Aug 1989. doi: 10.1109/25.45466

[29] P. Ghyzel, “Vision-based navigation for autonomous landing of unmanned aerial vehicles,” M. Sc. thesis, Naval Postgraduate School, United States, 2000.

[30] K. E. Wenzel, P. Rosset, and A. Zell, “Low-cost visual tracking of a landing place and hovering flight control with a microcontroller,” *J. Intell. Robot. Syst.*, vol. 57, no. 1–4, pp. 297–311, 2010. doi: 10.1007/s10846-009-9355-5

[31] Y. Gui, P. Guo, H. Zhang, Z. Lei, X. Zhou, J. Du, and Q. Yu, "Airborne vision-based navigation method for UAV accuracy landing using infrared lamps," *J. Intell. Robot. Syst. Theory Appl.*, vol. 72, no. 2, pp. 197–218, 2013. doi: 10.1007/s10846-013-9819-5

[32] G. Xu, X. Qi, Q. Zeng, Y. Tian, R. Guo, and B. Wang, "Use of land's cooperative object to estimate UAV's pose for autonomous landing," *Chinese J. Aeronaut.*, vol. 26, no. 6, pp. 1498–1505, 2013. doi: 10.1016/j.cja.2013.07.049

[33] J. Lugo, "Autonomous landing of a quadrotor UAV using vision and infrared markers for pose estimation," M. S. Thesis. Robotics Research Institute, Technical University Dortmund. 2011.

[34] F. S. Leira, "Infrared object detection & tracking in UAVs," M.S. Thesis. Norwegian University of Science and Technology, Trondheim, Norway, 2013.

[35] S. Yang, S. a. Scherer, and A. Zell, "An onboard monocular vision system for autonomous takeoff, hovering and landing of a micro aerial vehicle," *J. Intell. Robot. Syst. Theory Appl.*, vol. 69, no. 1, pp. 499–515, 2013. doi: 10.1007/s10846-012-9749-7

[36] N. Kummer, H. Firouzi, D. Jacobs, and H. Najjaran, "Autonomous UAV landing via eye in hand visual servoing," *Unmanned Syst. Canada*, pp. 2–7, 2011.

[37] N. Kummer, C. Jee, J. Garbowski, and E. Nowak, "Design and development of the hardware for a vision-based UAV autopilot," In

Proc. Can. Soc. Mech. Eng. Int. Congr. 2014, pp. 1–6, 2014.

[38] J. Kim, J. W. Starr, and B. Y. Lattimer, "Firefighting robot stereo infrared vision and radar sensor fusion for imaging through smoke," *Fire Technol.*, vol. 51, no. 4, pp. 823–845, 2015. doi: 10.1007/s10694-014-0413-6

[39] O. A. Yakimenko, I. I. Kaminer, W. J. Lentz, and P. A. Ghyzel, "Unmanned aircraft navigation for shipboard landing using infrared vision," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 38, no. 4, pp. 1181–1200, 2002. doi: 10.1109/TAES.2002.1145742

[40] W. Kong, D. Zhang, X. Wang, Z. Xian, and J. Zhang, "Autonomous landing of an UAV with a ground-based actuated infrared stereo vision system," *IEEE Int. Conf. Intell. Robot. Syst.*, no. 1, pp. 2963–2970, 2013. doi: 10.1109/IROS.2013.6696776

[41] J. Herschel, "Note on the art of photography, or the application of the chemical rays of light to the purposes of pictorial representation," *Abstr. Pap. Print. Philos. Trans. R. Soc. London*, vol. 4, pp. 131–133, 1837. doi: 10.1098/rspl.1837.0061

[42] L. Schaaf, "Sir John Herschel's 1839 Royal Society Paper on Photography," *Hist. Photogr.*, vol. 3, no. 1, pp. 47–60, Jan. 1979. doi: 10.1080/03087298.1979.10441071

[43] J. Herschel, "Note on the art of photography, or the application of the chemical rays of light to the purposes of pictorial representation," *Abstr. Pap. Print. Philos. Trans. R. Soc. London*, vol. 4, pp. 131–133, 1837.

doi: 10.1098/rspl.1837.0061.

[44] M. Planck, “The Theory of Heat,” *Nature*, vol. 105, no. 2634, pp. 228–228, 1920.

[45] “NIR Spectroscopy: A guide to near-infrared spectroscopic analysis of industrial manufacturing processes,” Metrohm NIR Systems, February 2013.

[46] "More Precision: Basics of non contact temperature measurement," Micro-epsilon, Ortenburg Germany.

[47] Roberts, L., "Machine perception of three-dimensional solids." Ph. D. thesis, Massachusetts Institute of Technology, 1963. [Online], Available: http://hdl.handle.net/1721.1/11589 [Accessed Jan 15, 2018].

[48] “Basic Land Navigation,” National Wildfire Coordinating Group, June, 2016.

[49] B. Hofmann-Wellenhof, K. Legat, and M. Wieser, Augmentation systems: satellite-based navigation. *Navigation,* pp. 191–213. Springer, Vienna. doi: 10.1007/978-3-7091-6078-7_9

[50] S. Sural, Gang Qian and S. Pramanik, "Segmentation and histogram generation using the HSV color space for image retrieval," In Proc. *International Conference on Image Processing*, 2002, vol. 2, pp. 589-592. doi: 10.1109/ICIP.2002.1040019

[51] J. Canny, “A computational approach to edge detection,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 8, no. 6, pp. 679–698, Nov. 1986. doi: 10.1109/TPAMI.1986.4767851

[52] F. Maski, “Line detection by hough transformation,” April 2009. [Online] Available: http://web.ipac.caltech.edu/staff/fmasci/home/astro_refs/HoughTrans_lines_09.pdf [Accessed: 15-Jan-2018].

[53] O. Cetin, S. Kurnaz, and O. Kaynak, “Fuzzy logic based approach to design of autonomous landing system for unmanned aerial vehicles,” *J. Intell. Robot. Syst.*, vol. 61, no. 1–4, pp. 239–250, 2011. doi: 10.1007/s10846-010-9508-6

[54] F. Riccardi, P. Milano, and P. Milano, “Control of variable-pitch quadrotors,” In Proc. IFAC Proceedings Volumes, vol. 46, Issue 19, 2013, pp. 206-211, doi: 10.3182/20130902-5-DE-2040.00143.

[55] L. Wills *et al.*, "An open software infrastructure for reconfigurable control systems," In Proc. *2000 American Control Conference. ACC (IEEE Cat. No.00CH36334)*, Chicago, IL, 2000, pp. 2799-2803 vol.4. doi: 10.1109/ACC.2000.878721

[56] M. Conte, and M. Trancossi, "Infrared piloted autonomous landing: system design and experimental evaluation," SAE Technical Paper. 2014-01-2196. doi: 10.4271/2014-01-2196.

[57] I. Mellado-Bataller, P.Campoy, M. Olivares-Mendez, and L. Mejias, "Rapid prototyping framework for visual control of autonomous micro aerial vehicles. *Intelligent Autonomous Syst.* AISC, vol. 193 pp. 487-499. doi: 10.1007/978-3-642-33926-4_45.

[58] “SZI 1029," Sennheiser IR Audio Transmission Technology |

Modulators / Radiators, [Online] Available: https://en-us.sennheiser.com/global-downloads/file/1060/SZI_1029_GB.pdf [Accessed: 15-Jan-2018].

[59] S. T. Aden, J. P. Bialas, Z. Champion, E. Levin, and J. L. McCarty, “Low cost infrared and near infrared sensors for UAVs,” *ISPRS - Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.*, vol. XL-1, no. November, pp. 1–7, 2014. doi: 10.5194/isprsarchives-XL-1-1-2014