Abbas-PhD-Thesis.pdf

by

Submission date: 07-Jun-2022 01:00PM (UTC+0500)
Submission ID: 1852147105
File name: Abbas-PhD-Thesis.pdf (17.93M)
Word count: 54943
Character count: 283664

An Autonomous Water Canal Structural Profiling System using an Unmanned Aerial Vehicle (UAV)

A Thesis
Presented to
the Academic Faculty

by

Syed Muhammad Abbas

In Partial Fullfilment
of the Requirements for the Degree of
Doctor of Philosophy in
Electrical Engineering

Supervisor: Abubakr Muhammad (LUMS)

Syed Babar Ali School of Science and Engineering
Lahore University of Management Sciences
March 2022

To the most awaited one.

Abstract

The objective of this thesis is to build an autonomous aerial platform that is capable of traversing the length of water canals with minimum human interaction and to reconstruct the civil structures of the water canal for inspection purposes. The civil structures include canal bed, canal banks, and other engineered structures which plays an integral part in the efficient discharge of water flow. The water channels whether for irrigation or transportation require constant monitoring and maintenance for effective water discharge as they face consistent degradation threats from silt accumulation, floods, urbanization, changing landscapes, and unauthorized civil structures. Relevant authorities conduct regular periodic inspection surveys all around the world which include a random selection of canal length as an inspection site and manually collecting data using mostly handheld sensor systems. Later the collected data is analyzed to conclude the state of the canal at the inspected part. This method of manually traversing the length of the canal and collecting data using handheld devices is slow and prone to inaccuracy due to the subjective decisions made during the monitoring surveys and human errors. In addition, the large length of the canal networks makes this problem near to impossible as frequent monitoring surveys are required to keep the irrigation system efficiency. We have presented a novel approach for building an aerial autonomous system capable of traversing the length of water canals for inspection. First, in this thesis, we have presented a novel technique for the system integration of an aerial platform with the required state-of-the-art sensor systems. Hardware integration involves multiple sensors, such as GPS, IMU, monocular camera, and LiDARs. Calibration techniques for different sensors have been discussed in detail along with the data post-processing to increase the overall accuracy and efficiency of the proposed aerial sensor system. To increase the accuracy of an onboard GPS, a Kalman filter-based approach has been proposed as a post-processing step and showed the improvement by a detailed experiment. We have also addressed the noise associated with each axis of accelerometer and proposed Finite Impulse Response (FIR) based low-pass filter approach to remove noise. Moreover, this low-pass filter causes a delay in the real-time system due to data structuring the inputs in an array and by D Flip-Flops at the hardware level. This delay has been removed by the product of Z-transform of sensor input readings and z^{-i}. For an onboard monocular camera, Radial and Tangential distortions have also been identified and their solution is proposed by using the Levenberg-Marquart optimization approach. Second, an Extended Kalman Filter (EKF) based accurate state estimation approach for an aerial robot is presented along with customized sensor modeling for efficient and accurate data fusion. In addition to the estimation of state, 3D mapping is also presented based upon the generated trajectories to generate a 3D reconstruction of the canal structure using point clouds. Third, a novel technique to measure the accuracy of a localization algorithm in an outdoor environment is proposed using a state-of-the-art visual fiducial marker. We have analyzed that the error propagation of the selected visual fiducial marker is along two horizontal axes along with the effect of angular rotation about the vertical axis. We have

identified that the angular rotation of the camera (yaw angle) about its vertical axis is the primary source of error that decreases the precision to the point where the marker system is not potentially viable for sub-decimeter precise tasks. Other factors are the distance and viewing angle of the camera from the AprilTag. Based on these observations, three improvement steps have been proposed. One is the trigonometric correction of the yaw angle to point the camera towards the center of the tag. Second, the use of a custom-built yaw-axis gimbal, which tracks the center of the tag in real-time. Third, we have presented for the first time a pose-indexed probabilistic sensor error model of the AprilTag using a Gaussian Processes based regression of experimental data, validated by particle filter tracking. The fourth contribution of this thesis is a novel navigation technique to traverse the length of the water canal using a deep Convolutional Neural Network(CNN). Our Convolutional Neural Network is inspired by ResNet50. Given the uniqueness of our problem, we have generated our dataset for supervised learning and validation and later evaluated the proposed approach on a real canal. We have implemented our approach on a COTS micro-aerial vehicle. We have designed our system in such a way that it takes 200*ms* from perception to action thereby making the system real-time. The fifth contribution is a novel technique to classify the canal bed terrain for effective hydrological modeling. The last and sixth contribution is a practical method to estimate the volume of soil from a generated point cloud.

List of Contributions

Below is a list of the salient contributions resulted from this thesis. Where applicable, the contributions are linked in parenthesis to the publications list of the following section.

1. Design & Development and State Estimation of an Aerial Platform.
   (a) Design and development of an aerial robot with integrated state of the art sensor systems for canal structural monitoring **(Paper 2, Paper 5)**.
   (b) A customized approach to model the on-board sensor systems on an aerial platform for data fusion using an Extended Kalman Filter(EKF) **(Paper 2)**.
2. Ground-Truth Analysis of Localization using an External Marker in Outdoors.
   (a) Improvement in the accuracy and robustness of the most commonly used external fiducial marker *AprilTag* by proposing two correction techniques: Passive & Active **(Paper 1)**.
   (b) A probabilistic sensor error model of the AprilTag by using Gaussian Processes (GP) based regression of experimental data **(Paper 1)**.
3. Autonomous Navigation of an aerial platform over Water Canals.
   (a) A novel approach for the autonomous navigation of aerial robot over the length of water canal using deep Convolutional Neural Network (CNN) **(Papers 3)**.

(b) A Formal Method based aerial robot command controller for the navigation over water canal **(Paper 3)**.

4. Further Applications for the Efficient Canal Structural Monitoring.

   (a) A improve approach in terms of accuracy and response time for single camera based classification of terrain **(Paper 4)**.

   (b) A novel method of estimation of soil volume in a 3D pointcloud for the an excavation problem **(Paper 6)**.

Publications

Below is a list of publications that have resulted directly from the research described in this thesis. The chapters from the dissertation that include the material from each publication have also been given in the parenthesis.

Journal articles

Paper 1 (Chapters 4)

Syed Muhammad Abbas, Salman Aslam, Karsten Berns, Abubakr Muhammad, "Analysis and Improvements in AprilTag Based State Estimation", *Sensors*, vol. 19, no. 24, pp. 5480, Dec 2019.

Paper 2 (Chapter 3 & 5)

Syed Muhammad Abbas, Abubakr Muhammad, "Design & Development of an Aerial Vehicle for Structural Inspection of Water Channels", Submitted at Elsevier's Journal of Robotics and Autonomous Systems (RAS).

Conference proceedings

Paper 3 (Chapter 6)

Syed Muhammad Abbas, Hashim Ali, Abubakr Muhammad, "Autonomous Canal Following by a Micro-Aerial Vehicle Using Deep CNN", *6th IFAC Conference on Sensing, Control and Automation Technologies for Agriculture (AgriControl)*, Sydney, Australia, Dec 2019 .

Paper 4 (Chapter 7)

Syed Muhammad Abbas, Abubakr Muhammad, Syed Atif Mehdi, Karsten Berns, "Improvements in accuracy of single camera terrain classification", *16th IEEE International Conference on Advanced Robotics (ICAR)*, Montevideo, Uruguay, Nov 2013.

Paper 5 (Chapter 3 & 5)

Abubakr Muhammad, Ali Ahmed, Saad Hassan, **Syed Muhammad Abbas**, Talha Manzoor, Karsten Berns "Using unmanned aerial vehicle to inspect siltation in irrigation canals", *37th International Association for Hydro-Environment Engineering and Research (IAHR)*, Kuala Lumpur, Malaysia, Aug 2017.

Paper 6 (Chapter 8)

Hamza Anwar, **Syed Muhammad Abbas**, Abubakr Muhammad, Karsten Berns "Volumetric estimation of contained soil using 3d sensors", *3rd International Commercial Vehicle Technology Symposium, (CVT)*, Kaiserslautern, Germany, Mar 2014.

Acknowledgments

I would like to thank Almighty for giving me the opportunity, strength, health, and intellect to work on a challenging problem and complete the scope of this thesis.

Then I would like to thank my adviser Abubakr Muhammad for his unparalleled guidance and support that plays an integral part in the completion of this thesis. His guidance was crucial in solving critical thesis problems and he made available every hardware/sensor system required for the research. Other than academic guidance, I have learned a lot from him while working together to become a professional and most importantly a better human being.

I would also like to thank Prof. Karsten Berns for hosting me several times at his research lab in Kaiserslautern, Germany, and for his continued guidance in the research work. Afterward, I would like to thank all my friends and colleagues at LUMS. It might not be possible to list the names of all of them here but I would like to mention the names of Talha Manzoor and Muhammad Mudassir Khan. Being my colleagues and research fellows, we have had quality time together and they have helped and motivated me in tough times.

Then I would like to thank my elder brothers Syed Ali Abbas and Syed Raza Abbas for their support and motivation not only during my Ph.D. thesis time but from my childhood days up till now. Following this, I would like to thank my soulmate Syeda Alisah Fatima Jafri for her encouragement and backing during my thesis research span. I found her very supportive and patient. Even with short finances as a Ph.D. student and my long working

hours, she stood by my side firm and strong. Furthermore, I would like to extend my love to my daughters, Syeda Insiya Zainab and Syeda Nida Fatima as both of them came to this planet during my thesis time and filled our lives with joy and happiness.

Lastly, I would like to extend my gratitude to two persons, without whom I would not have achieved what I have done so far. First, is my maternal grandfather Syed Muhammad Naqi Haider who taught me from the very beginning of my schooling and laid the foundation which helped me to excel in my life. He took care of everything from my childhood to the university. I wish he would have been with us at this moment as without him it would have not been possible. Second, is my mother Syeda Zahira Naqi who is my father as well. She worked tirelessly her whole life and made everything available to me from tuition fees to living. I am in debt to her for the rest of my career and all my efforts in this direction can never countervail even a small part of her struggle.

Syed Muhammad Abbas,
LUMS, 2022.

Contents

List of Figures

List of Tables

Glossary

UAV	Unmanned Aerial Vehicle
GDP	Gross Domestic Product
2D	Two Dimensional
3D	Three Dimensional
VTOL	Verticle Takeoff and Landing
IMU	Inertial Measurement Unit
GPS	Global Positioning System
DOF	Degree of Freedom
SLAM	Simultaneous Localization and Mapping
EKF	Extended Kalman Filter
CNN	Convolutional Neural Network
MAV	Micro Air Vehicle
RPM	Revolutions Per Minute
GP	Gaussian Processes
MoCap	Motion Capture System
SYAC	Soft Yaw Axis Correction
RMSE	Root Mean Square Error

ROS	Robot Operating System
ResNet	Residual Network
ReLU	Rectified Linear Unit
FC	Fully Connected
SVM	Support Vector Machine
ICP	Iterative Closest Point
RANSAC	Random Sample Consensus
UTM	Universal Transverse Mercator

Part I

Background, Motivation and Problem Statement

Chapter 1

Introduction & Background

Water is considered to be the most valuable renewable resource for any country. It plays an important role in economic progress by improving GDP growth. Even in abundance, efficient policies and smart infrastructure is still required to optimally utilize this valuable resource and gain maximum benefits. Many countries around the globe depend heavily on their water reservoirs ranging from basic human needs to agriculture and even for sustainable economic development. Prominent river basins in the world are Amazon, Mississippi, Ganges, Indus, Nile, Yangtze, and Yenisei. Billions of people are dependent directly or indirectly on these huge river basins covering millions of Kilometers. An estimated 500 million people in Nepal, India, Tibet, and Bangladesh are served by the Ganges river basin alone [27]. Pakistan being an agricultural country is directly dependent on water for its social and economical development. Though Pakistan lies underneath the world's biggest glaciers outside polar regions and has many rivers unfortunately they are not sparse. Another challenge is the temporal non-uniformity of the water flow in these rivers around the year. The water flow is not uniform throughout the year and it is concentrated in mid-summers to Monsoon season. Indus river basin is the largest and the most prominent river basin in Pakistan serving as the backbone for all the water reservoirs but it is also not sparse. Therefore, this misfortune is converted into an opportunity by a

network of barrages, main canals, sub canals, minors, sub-minors, and outlets throughout the Indus river basin. Pakistan has the privilege of containing one of the largest contiguous canal irrigation systems under gravity flow shown in Figure1-1. The problem becomes more challenging as the approximate length of all the canal networks in the Indus basin is more than 50,000 Km. Figure 1-1 shows the major five rivers of Pakistan named Indus, Jehlum, Chenab, Ravi, and Sutlej which cover almost Northern, Eastern, and southern parts of Pakistan. However, there are other small rivers Swat, Kabul, Kurram, and Gomal which cover the north-western mountainous part of Pakistan. All the rivers fall into the Indus River and eventually, it goes into the Arabian Sea.

Pakistan has an arid to semi-arid climate, so the timely distribution of water to the fields via canals is an inevitable necessity. Table 1.1 shows the massive irrigation infrastructure present in Pakistan. Irrigated land contributes to 90% of the total agriculture production in

Table 1.1: Key facts for irrigation system in Pakistan.

No. of major Reservoirs:	3
No. of Barrages:	16
No. of Headworks:	2
No. of Inter link Canals:	12
No. of Canal Systems:	44
No. of Watercourses:	107,000
Length of Canals:	56,073 km
Length of Watercourses:	1.6 million km
Average Canal Water Diversions:	104.7 MAF
Groundwater Abstractions:	41.6 MAF
No. of Tubewells:	approx. 750,000
Irrigated Area:	36 million acres
Average Escapage to the Sea:	39.4 MAF

Pakistan and provides 54% of the labor force and 22% of the GDP [4].

Though this massive infrastructure fulfills the majority country's needs they are in

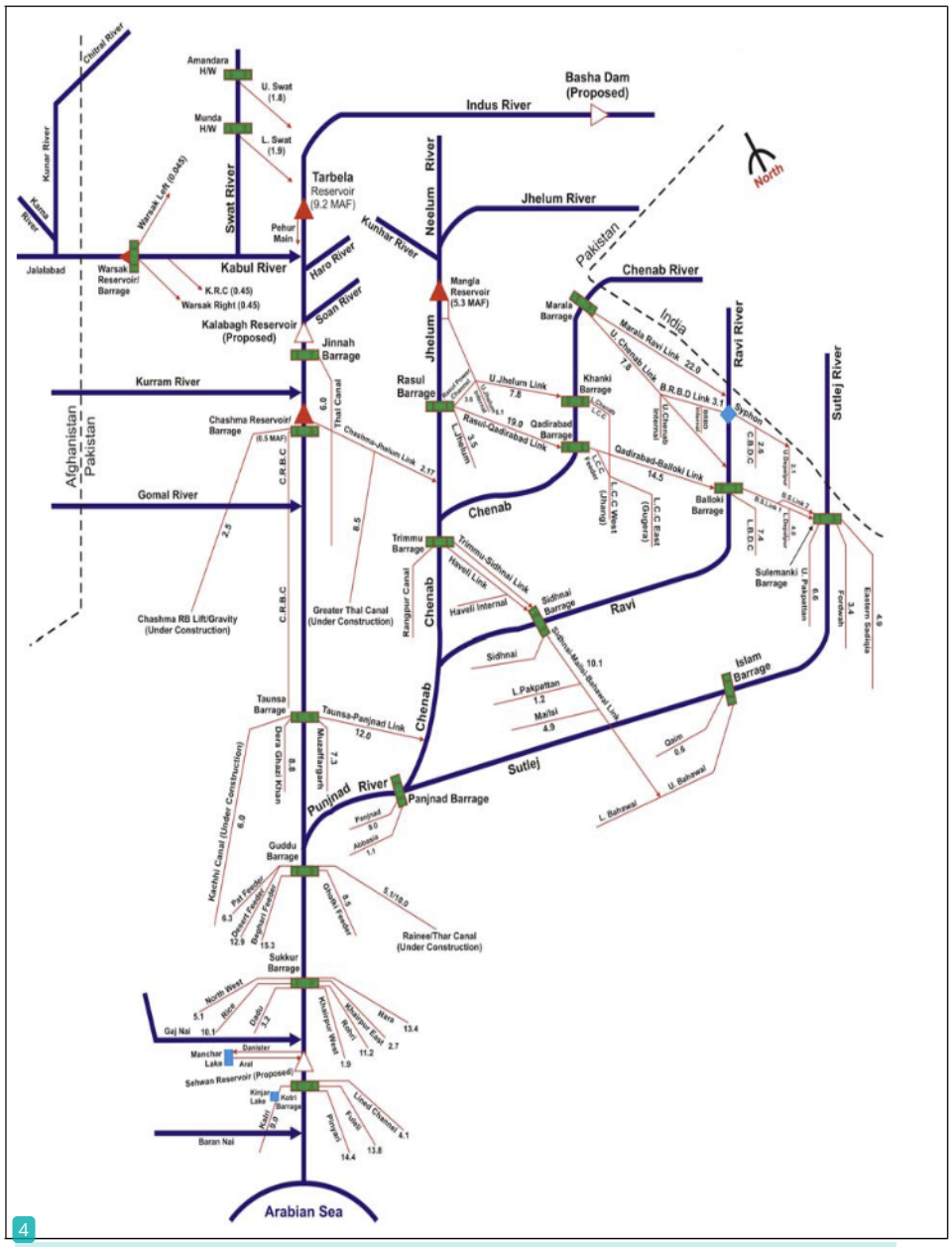


Figure 1-1: Irrigation infrastructure based on Indus River in Pakistan [190].

constant danger of degradation over time. This is a common problem for large irrigation systems around the world. Many factors contribute to the degradation of irrigation systems such as silt accumulation, floods, urbanization, changing landscapes and water-theft, etc. as shown in Figure 1-2.

Figure 1-2: *Top left*: The process of silt removal accumulated at the Lahore Grand Canal is underway. *Top middle*: Water channel bank breach at Faiz Ganj canal due to flooding in Punjab, Pakistan. *Top right*: Effect on Liyari river due to the urbanization at the suburbs of Karachi, Pakistan. *Bottom left*: Shows accumulation silt at one of the canals in Punjab, Pakistan. *Bottom middle*: A brick-lined canal is breached 60 feet wide at the Jhang Branch canal due to lack of maintenance. *Bottom right*: Shows a wearing structure of a canal head in Punjab, Pakistan.

One of the most common problems to the water reservoirs is the accumulation and settling of silt in Pakistan. This phenomenon is quite common in other parts of the world as well. Accumulation and sedimentation of silt on canal beds highly affect the water carrying and discharge capacities of the canals. Silt consists of small dust-like earthly particles hanging in water sizing in between clay and sand particles. There are several sources of silt. The most common one is the grinding of tiny rock fragments flowing in the water against each other, which results in the reduction of their size until they are silt-sized.

Another frequent danger is the erosion of canal banks due to the fast-flowing water either due to floods or heavy rains in the Monsoon season. It out-turn adversely on the

required quantity of water discharge at the downstream and results in substantial economic losses. Along with the canal bank erosion, urbanization also plays an important role in downgrading the flow of water in natural rivers and streams which causes either the loss of water discharge or pollution due to the domestic waste making it unhealthy for agriculture and domestic use. Another important factor is the change in the landscape of the river basins with time. This affects the connected canal systems and they have to be modified accordingly so that the one can achieve the required flow of water at the far areas. Similarly, water theft and unauthorized fall of waste discharge in the canal damage the infrastructure of the canal network as abnormal removal or addition of water along the length of the canal disturbs the flow of water and can create stress at different distribution heads and can even damage them.

To keep the irrigation network efficiently running, regular maintenance is required against all the factors discussed above. In case of silt accumulation, if the canals are not maintained and cleaned properly from time to time, the canals lose their water carrying capacity and may even start leaking at shallow points. So canals must be maintained and clean on regular basis. The irrigation department in Pakistan also conducts annual activity of canal cleaning called "Bhal Safai(De-silting)". In the de-silting operations conducted by the Government of Pakistan, all canals are altogether closed for approximately three weeks, once a year, and the de-silting process is commenced nationwide. The de-silting operation consists of three steps. One is the inspection of canal beds, in which canal beds are inspected through a walk-through to identify silted-up reaches. This is a manual technique based on many subjective decisions. Instead of calculating silt for every patch of the canal, readings are taken every 1000 feet and if the average silts of two consecutive inspected points are over or equal to 6 inches, the whole patch is marked for de-silting. The second step is the estimation of silt after identification. This process also includes major approximations because lack of structural data related to the canal. All canals are assumed trapezoidal of specific dimension and volume of the silt is estimated as shown in

Figure 1-3. The third part is the cleaning of the silt already identified and estimated. This is carried out in many ways including self-help, contracts (Public/Private), farm machinery, and through charged labor. The silt measuring activity is also shown in Figure 1-3.

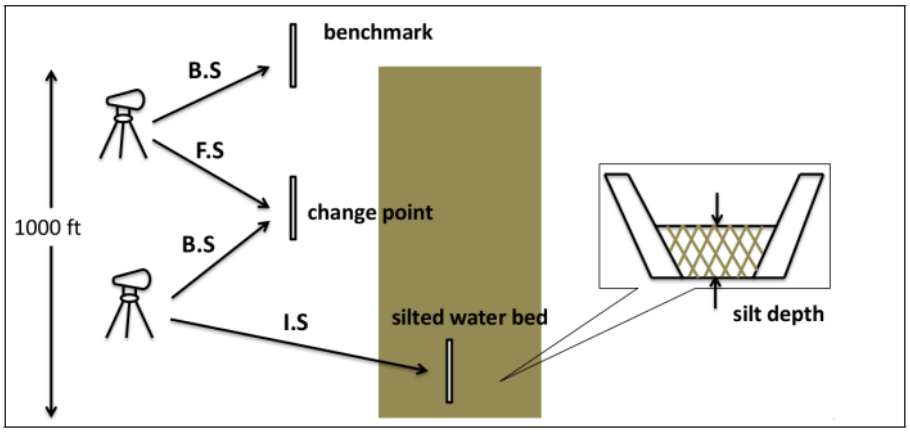


Figure 1-3: Leveling procedure for silt identification used in Pakistan

We can see that the above method of silt estimation is prone to human errors. It also includes a lot of approximation which either cause over or underestimation of silt. As mentioned earlier, the annual closure of canals is only for two weeks. So to identify and estimate the silt over thousands of kilometers in this short span is a challenge. Even if the identification is done, there is no mechanism for inspecting canals after cleaning operations to see whether the task has been done properly and the resulting shape of the canal allows smooth flow of water. Also instead of using a standard trapezoidal shape for all the canals to measure the silt, incorporating actual physical geometric dimensions of the canals can greatly increase the accuracy of the estimated volume of silt and can save a lot of valuable resources.

In addition, canal banks damaged due to flooding or erosion also need constant maintenance to keep the water flow unchanged. This requires constant inspection of the water canal for any damage over the entire length of the canals. Similarly, regular surveillance

is also required near the populated areas to access the effects of urbanization on the flow of the canals. This includes both the physical effects like blockage due to solid waste, the effect on the flow of water due to bridges and civil structures, and the pollution caused by the wastewater from sewers. Furthermore, the periodic inspection of canal infrastructure is also required for time-based degradation and changed due to human intervention for water theft. In addition, in many countries, the rivers and canal networks are also used for the transportation of both cargo and passengers. Regular inspection of these networks is more significant in such scenarios as regular traffic with massive mass damages the banks and other infrastructures along the length of the canal. In the Netherlands, the inland rivers and waterways contribute around 35% of the total national transportation services [26].

Hence it is inevitable to inspect the water canals regularly to keep the whole irrigation network effective and useful. As shown in Table 1.1, Pakistan has got a total length of the canal over 50 thousand kilometers. Manually inspection of the whole irrigation system is practically not possible especially if it is required at least once every year. Due to this very reason, the infrastructure of the Indus river basin is constantly degraded over time. Though the concerned departments are working hard to keep pace with the challenges they are facing but due to the massive volume of the total irrigation structure in Pakistan, it is near impossible to maintain or even inspect it on regular basis.

This leads us to the fact that there is a dire need for some automated/autonomous system capable of inspecting the water canals with the minimum involvement of the human operator. With the current advancement in Robotics, Computer Vision, and Artificial Intelligence, the required system can become a practical solution shortly. This reason motivates and forms the basis of the current research thesis. To keep the large irrigation systems effective, an autonomous robotics system in the need of the time capable of traversing the length of the irrigation system and identifies the structural changes with minimum human intervention so that timely maintenance work should be carried out by

the concerned authorities. Nonetheless, this kind of autonomous system would consist of large complexities as the working environment is quite uncertain and complicated. One of the most challenging complexity is the traversal of the water channels. The canals can either be traversed by using a surface platform or by an aerial platform. In the case of the surface platform, though it is quite a stable choice sometimes physical traversal of this platform becomes impossible due to blockages such as hanging tree branches, low altitude bridges, and narrow diversions. Consequently, an aerial platform is the most suitable one for the given task. Another benefit of an aerial platform is that with the water-filled canal, it is hard to completely inspect the structure of the canal including its banks, bottom, and other civil structures beneath the water. So ideally aerial platform should traverse the dried water canal for more comprehensive structural inspection.

1.1 Problem Statement

The inland waterways and rivers belonging to any river basin in the world require constant monitoring and maintenance of civil structures for effective water discharge so that they may serve their purpose. Especially man-build waterways face consistent degradation threats from silt accumulation, floods, urbanization, changing landscapes, and unauthorized civil structures. The civil structures include waterway bed, banks, and other civil constructions which plays an integral part in the efficient discharge of water flow. Currently, most of the methods being used for the monitoring and surveys of the water canals' civil structures involve manual data collection using random on-site surveys with handheld sensor devices along the length of canals. The prevalent current method of manually collecting the structural inspection data of water canals is slow and prone to inaccuracy due to the subjective decisions made during the monitoring surveys and human errors. In addition, the large length of the canal network makes this problem near to impossible as frequent structural monitoring surveys are required to keep the water channels network efficient and long-term. This very reason builds the

narrative of your research for this thesis. Our questions are: Can we build an autonomous system that is capable of traversing the length of water canals with minimum human interaction and 3D reconstructing the civil structures of the water canal for inspection purposes? The autonomous system should be capable of traversing the length of water canals for inspection in both dry and filled forms. What kind of system integration for a required inspection platform is required so that it may carry the necessary state-of-the-art sensor systems while operating? How accurate can we estimate the states of the traversing platform over the length of the canal for large distances? How can we measure the accuracy of the state estimation approach in large outdoor environments to estimate the confidence level? Can we construct a 3D accurate reconstruction of waterways civil structures by using the generated trajectory states? Is there a method to autonomously navigate the platform over the length of the water canal without any human input? Lastly, how can we process the collected data from the autonomous platform's survey to extract meaningful insights? If we answer the above arguments, then it will open a new era for the waterways structural inspection problem and can benefit many countries facing similar problems for the water canal networks.

In this thesis, we have proposed a novel approach for building such an autonomous aerial system by first identification of all the required system modules and then building the foundation modules which complement each other towards the completion of the required canal inspection autonomous system. The foundation modules for the required semi-autonomous system have been proposed in this thesis that include the system integration of an aerial platform with all the required sensor systems, a very precise state estimation of the aerial platform, a state of the art novel method for measuring the accuracy of localization approach using 2D markers in an outdoor environment, initial 3D mapping of the canal structures, a novel method of autonomous canal traversal using convolutional neural networks, a novel approach for the identification and classification of canal beds types and an initial approach for the estimation of silt using the point clouds.

In the next chapter, we shall overview the different aerial platforms according to their specifications and usability along with their dynamical modeling to find the best suitable aerial platform for our task. Then we shall examine the most advanced state estimation techniques in aerial robotics. After that, we shall survey the cutting-edge 3D mapping approaches that can be used for canal structural reconstruction in 3D. Following, we shall review the navigation techniques in aerial robotics along with the different approaches for trajectory tracking.

Chapter 2

Related Work

Over the past decade, aerial robots have become a prominent feature in the field of robotics. Their unique capability of performing challenging tasks like exploring non-traversal areas (rumbles, water canals, etc.) and quick reachability to the far-fetched areas has made it the obvious choice for many researchers. In addition, the low-cost development of these platforms has brought aerial robotics as the forefront leader in expanding the robotic horizons. Another reason for the recent popularity of this field is the availability of highly stabilized aerial platforms and high processing lightweight embedded boards. Full support of multi-tasking by these high-end embedded machines allows aerial robotics to push its limits and perform complex operations which were beyond the reach of the ground robots.

2.1 Aerial Platforms for Robotics

There are a large number of designs and characteristics of aerial platforms which are based upon their flying principle and propulsion mechanism. At first, aircraft or aerial platform is classified into two classes. One is 'heavier than air' and the other is 'lighter than air'. 'Heavier than air' are those aerial platforms that weigh more than the air. Hence,

to fly them, we need forces generation either an engine or some other mechanism to lift them in the air. Whereas, 'lighter than air' are those aerial platforms that are lighter than air, mostly in a circular shape, and need no extra engine to lift them in the air. 'Heavier than air' class is further classified into two sub-classes. One is the 'motorized' and the other one is the 'non-motorized' class. In the motorized sub-class, the engine is used to generate lifting force whereas, in the non-motorized sub-class, aircraft are designed in such a way that without the use of an engine it can generate lifting thrust normally by using external forces like gravity and wind. The Motorize sub-class is further divided into three categories. One is the 'Planes', which use fixed wings and engines to lift them in the air like aircraft. The second category is the 'rotor-crafts' which use a main single propeller along with the engine, to lift them in the air. The most common example of this category is helicopters. The third category is for the 'vertical takeoff and landing(VTOL)' aerial platforms. In this category, multiple propellers are used along with the engine or engines to lift the aircraft in the air like quad-rotors, hex-copter, octocopter, etc.

Similarly, the class 'lighter than air' is further divided into two sub-classes i.e. non-motorized and motorized. In the non-motorized sub-class, aircraft is not equipped with any kind of engine to either fly or navigate in the air. They depend on the wind gust to gain altitude and commonly do not navigate themselves. An example of this category is the balloons like weather balloons that fly freely in the atmosphere to collect the weather data. The other sub-class has an engine attached to it but that is only used to generate thrust for the navigation. Blimp is an example that is used for traveling. It is filled with helium gas which is lighter than air for gaining altitude and has external propellers for navigation in the air. Figure 2-1 shows the whole classification in a hierarchical form.

UAV is a general term used for these aerial platforms that stand for 'Unmanned Aerial Vehicle'. It intends a generic aerial platform that is designed in such a way to fly without an onboard pilot. There are some other names also associated with such aerial platforms which depend upon the type of tasks they are capable to perform. The most common one

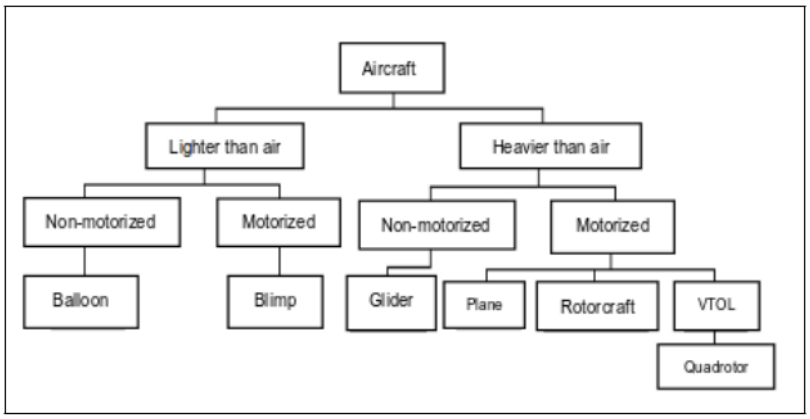


Figure 2-1: Aerial platforms classification hierarchy (Reproduced [85])

is UAVS. It means Unmanned Aerial Vehicle System which includes both unmanned aerial vehicle (UAV) and the ground control center often termed as GCC. The table 2.1 shows a few names of such aerial platforms along with the tasks they perform.

Table 2.1: Common names used for unmanned aerial vehicles. Here, general purpose operations means operations/tasks related to military, civil and agriculture.

Name of an Aerial Platform	**Tasks to Perform**
Remotely piloted vehicle (RPV)	General purpose operations
Drone	Use specifically in military operations
Remotley operated Aircraft (RPA)	General purpose operations
Unmanned combat air vehicle (UCAV)	Use in military for combat(air to ground)
Long Endurance UAV (LEUAV)	Use for surveillance over 1000s of kms
Medium Endurance UAV (MEUAV)	Use for surveillance over 100s of kms
Remote control helicopter (RCH)	General purpose operations
Micro aerial vehicle (MAV)	Portable, miniature and use for general tasks

Selecting the type of aerial vehicle that best suits the requirements is dependent on two factors. First is the geographical span of the task at hand and second is the amount of data

needed to process or complexities that need to handle in real-time during the flight. In our scenario, as we intend to traverse along the length of the water canal for profiling and mapping, hence we will be dealing with a large Geo-scene in multiple kilometers. Since we are also mapping each part of the canal structure for inspection so we will be dealing with a huge amount of data per scene as well. F Nex et. al. [153] has presented a thumb rule for selecting a method of collecting mapping information for the requirements as shown in Figure 2-2. In Figure 2-2, the horizontal axis shows the scale of the operation or scene and the vertical axis shows the amount of data processing required against each scenario. According to proposed rule [153], the best-suited air vehicle for our research would belong to a category of 'UAV'. Considering the 3D data collection is a slow process, fixed wings aircraft is not suitable for our use-case as they need to fly at a minimum speed to generate enough lift from the wings to stay airborne. As a result, VTOL capability is the reason we have chosen a multi-copter aerial platform for our problem.

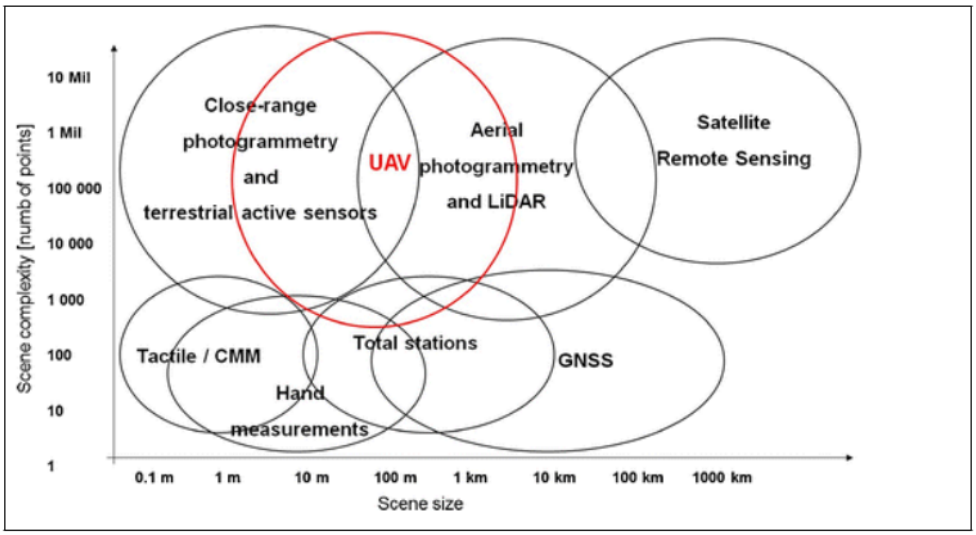


Figure 2-2: Identification of data processing required for an aerial platform based on the intended task (Reproduced [153]).

Since it is still an emerging field in robotics, there are not many commercial applications exits in the market yet but a lot of research is being done with high hopes of commercialization of these robotics systems shortly. Further in this chapter, first, we will elaborate on different techniques used to stabilize the multirotor. Then we will see state-of-the-art algorithms and techniques used by the researchers for an aerial platform for localization and state estimation. Afterward, we will review recent research regarding aerial 3D mapping, obstacle avoidance, navigation, terrain classification, and estimation of the volume of soil from a point cloud. As our goal is to build a novel aerial robotics system to fly over the length of a canal and follow the canal trajectory over large distances. It is a novel solution and has limited researchers have worked on it. However, we will examine the similar approaches which can be used under our scenario as well.

2.2 Multi-rotor Dynamic Modeling & Control

We have chosen multi-rotor as an aerial platform because of its steady stability and hovering capabilities. By nature, a multi-rotor is not a stable platform at all however an efficient control strategy does the trick for us. Initial work started on multi-rotors in the early 20^{th} century however practical work on the aerodynamics of rotor wings started as early as 40 years back [189] [165] [12]. At that time, the idea of multi-rotor was not realized, however main work was done for single and double blade rotorcraft because of its VTOL capability. Another reason was the inability to control all four propellers simultaneously in real-time because of a lack of bandwidth and processing power. In the 2000s [118], after the advancement of embedded systems and revolution of multi-core processing made this hovering machine a practical reality. Since then, a lot of research has been done on the dynamic model and control of the quadrotor. GM Hoffmann et.al. [94] has elaborated the dynamics of quadrotors at high speed and with high wind disturbances. They have modified the forces and moments modeling and incorporated disturbance that occurred by high-speed winds. Erginer et.al.[63] work on the stability of the quadrotor and have

presented a general framework that controls and moves quadrotor in $x-axis$ and $yaxis$ by changing roll and pitch angles while keeping the altitude fixed. P Pounds et.al.[163] have presented a dynamic model for a large quadrotor weighing over 4 KGs along with tuning the mechanical design after the analysis of flyer attitude dynamics with flapping.

Before elaborating different control strategies being used currently, it is important to establish a basic multi-rotor dynamics model. The VTOL aerial platforms with the

Figure 2-3: Different types of rotor-crafts.

configurations of four, six, and eight propellers are called quadrotor, hexacopter, and octocopter respectively as shown in the figure 2-3. For the dynamic model, we will consider a multi-rotor with four propellers as the basic flying dynamics of all the configurations are similar. As the name suggests, the quadrotor is lifted and propelled by the four propeller blades fixed to a rigid body in a pairwise manner as shown in Figure 2-4. Propeller pair (1,3) and (2,4) turn in opposite directions to balance the moments, one clockwise and the other anticlockwise. Another benefit of this pairwise configuration is that desired yaw angle can easily be achieved by changing the rotation speed of the motors in pairs. There are two major coordinate reference frames involved, one is the world frame W, and the other is fixed to the body B with aligned origin and center of mass as shown in the Figure 2-4.

Rotor 1 and 2 are at the positive side of x_B and y_B respectively whereas rotor 3 is on negative x_B and rotor 4 is on negative y_B. Z-X-Y Euler angles [183] are used for the

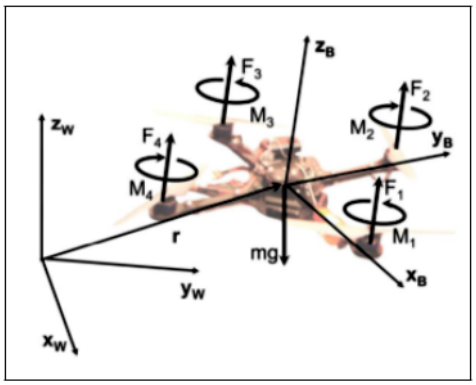


Figure 2-4: Coordinate systems and forces/moments of a quadrotor(Reproduced [142]).

rotation of multi-rotor from the body frame B to world frame W. The rotation matrix for transforming coordinates from body frame B to world frame W is given by

$$R_B^W = \begin{bmatrix} \cos\psi\cos\theta - \sin\phi\sin\psi\sin\theta & -\cos\phi\sin\psi & \cos\psi\sin\theta + cos\theta\sin\phi\sin\psi \\ \cos\theta\sin\psi + \cos\psi\sin\phi\sin\theta & \cos\phi\cos\psi & \sin\psi\sin\theta - \cos\psi\cos\theta\sin\phi \\ -\cos\phi\sin\theta & \sin\phi & \cos\phi\cos\theta \end{bmatrix}$$

where ϕ is the roll angle, θ is the roll angle and ψ is the yaw angle. The force from each of the rotor F_i is in z_B direction whereas the force due to gravity is directed to $-z_B$ direction. Acceleration of the center of mass can be given by

$$m\ddot{r} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + R_B^W \begin{bmatrix} 0 \\ 0 \\ \sum_{i=1}^{4} F_i \end{bmatrix}$$

where r is the position vector of the center of mass in W. Suppose p, q and r are the angular velocity component in frame B. These angular velocities are related to the derivative of roll, pitch, yaw as

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & -\cos\phi\sin\theta \\ 0 & 1 & \sin\phi \\ \sin\theta & 0 & \cos\phi\cos\theta \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}$$

However angular acceleration is given by

$$I \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} L(F_2 - F_4) \\ L(F_3 - F_1) \\ M_1 - M_2 + M_3 - M_4 \end{bmatrix} - \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times I \begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

where L is the distance from the center of the multi-rotor to the axis of rotation of rotor blades. M_i is the direction of rotation of ith rotor blade.

A quadrotor is an under-actuated system because we have to control six dimensions and we have only four inputs which is controlling the individual speed of each rotor. This makes quadrotor control a complex and challenging problem. A das et.al. [54] have used a two-tier approach by using input-output linearization to construct a non-linear controller for a particular quadrotor dynamical model. But their model fails in the presence of external disturbances like structural instability or wind etc. A solution to the atmospheric disturbances is given by Alexis et.al. [7]. They have presented a switching model predictive controller which forces the quadrotor to achieve precise control even in the presence of strong wind gusts. They have used an optical flow sensor along with IMU to get precise motion tracking. T Madani et.al. [130] have presented a new approach of back-stepping control. The idea behind this is to divide the whole system into three subsystems namely under-actuated, fully-actuated, and propeller. Back-stepping control stabilizes the whole system. Another interesting technique that has been used by R Xu et.al. [213] is a sliding control. To keep the quadrotor stable, they have also divided the whole system into under-actuated and fully-actuated subsystems. Both the subsystems are controlled separately using a sliding control method.

2.3 State Estimation

State estimation is a technique to determine the current position and pose of a robot. A good literature survey guild us to the latest state-of-the-art techniques similar to our problem. However, since aerial robotics is a new and emerging field so there is not much literature related to it but a lot of researchers are working in this field of robotics. One of them is [181], those have done a very similar task but for the rivers under the project name "Riverine". The project has focused on autonomously tracking the trajectories of the rivers and detecting the change in them over time. Since their task differs from ours as we are more interested in dry water channels for the silt and structural inspection. However, as far as the state estimation of the aerial platform is concerned the problem is almost identical. They have used multiple sensors to make the flying robot locally stable and globally consistent. For local consistency, they have used IMU, stereo cameras and for global consistency, they have relied on the GPS. In addition, they have proposed a self-supervised river detection algorithm. Since their objective is to fly the robot for a few kilometers so they devised a novel technique of scrolling incremental distance transform. Additionally, they have used the standard framework of state parameterization for the state estimation problem to fuse the data from the IMU, stereo vision, and GPS. Following, they have used sparse optimization technique to move the system to a state of overall low energy(2) using the Jacobian. They have presented a novel technique that combines different solutions and gives a nice river mapping system but experiments show that it is not accurate enough to compensate for the structural details we intend to measure especially in the long run. Moreover, the river detection technique does not work with the water canals, especially for the dried ones as they use the surface of the water to detect the flat plane for the banks' identification.

State estimation is one of the most challenging parts of aerial robotics. Especially for an under-actuated aerial robot such as a multirotor, state estimation plays an important role to

perform a given task autonomously in 6-DOF. Different techniques and sensors have been used for accurate state estimation by different researchers over the last decade. Some of the techniques use onboard sensors and some use external tracking systems which limits the flight in a deterministic lab environment. Muller et.al. [149] have used an external tracking system to track the state of the multirotor and have produced a required trajectory to bounce a ball back using a racket attach. The same tracking technique is used by [141] and [142]. They manually recorded the trajectory which enables the quadrotor to follow a certain known trajectory [112]. The use of a GPS can also help the multirotor explore the unstructured outdoor environment without prior knowledge [166]. Another important technique to improve the state estimation of a multirotor is to incorporate the visual feed with an IMU. Other than using an external tracking system for accurate localization, external makers can also be used and tracked by a monocular camera to get the exact position and pose of the flying robot by [59]. Amidi et.al. [11] have presented an odometer approach that first track and locks a moving target on the ground and then estimate the position and pose using a stereo camera along with the help of IMU readings. [156] has presented an overview related to how different perception techniques have increased the precision of aerial robots mainly in helicopters. Furthermore, most of the presented perception techniques are lately used by many researchers for their multi-rotors. P Corke et.al. [51] have used feature-based stereo imaging along with sonar to measure the height of the helicopter more accurately. [209] has also used the known external target to calculate the pose and position of the helicopter using an Extended Kalman Filter [124]. [37] has presented a visual approach to estimate the translation of a helicopter based on the views of a planer patch of a monocular camera.

Few of the researchers have treated the state estimation task as a non-linear optimization problem as well. Davison et.al. [55] and [64] have presented a recursive filtering sparse technique to track natural landmarks for the state estimation. However, [191] shows that the optimization approaches mostly outperform recursive filtering techniques which

further solidify the optimization approaches for its accuracy.

Parallel Tracking and Mapping [110] has been widely used for visual odometry [29] [206]. This technique relies on tracking and mapping hundred of features in real-time and estimating the 3D pose of the robotics platform along with generating map simultaneously at the same time using the Bundle Adjustment technique [111]. H Jin et.al. [101], [184] use a direct approach to simultaneously obtain the camera pose, correspondences among frames, and the scene structures. In a direct approach, instead of using features, pixel intensities are used directly to calculate the transformation between two consecutive frames. C Froster, M Pizzoli, and D Scaramuzza [74] propose a semi-direct visual state estimation approach that is robust and fast from any current state-of-the-art techniques. In this approach, processing extensive tasks like feature extraction and tracking are replaced by direct pixel-level operations which not even make it faster but also increase the accuracy to the sub-pixel level. A probabilistic mapping method is used to construct a 3D model for a pixel intensity point by explicitly modeling the outlier measurements as shown in Figure2-5. Experiments have been conducted on a micro aerial vehicle with limited processing power in a GPS-denied environment. The results have achieved an accuracy of 10 to 20 cms but with a limited flight time.

For real-time localization, [62] have used a monocular camera instead of stereo pair and used Extended Kalman Filter [124] for the fusion of data the IMU and the vision. They have not used GPS as they are not interested in global consistency. They have analyzed the system for up to milliseconds for enhanced usability for limited performance systems. For monocular SLAM, they have used the Parallel Tracking and Mapping algorithm [110] and generate visual odometry from it. However further investigation into the visual odometry, we have found Fast Semi-Direct Monocular Visual Odometry technique [74] way more accurate and faster than PTAM. In this approach, instead of detecting and tracking features, it uses pixel to pixel comparisons and tracking to increase the accuracy and processing time.

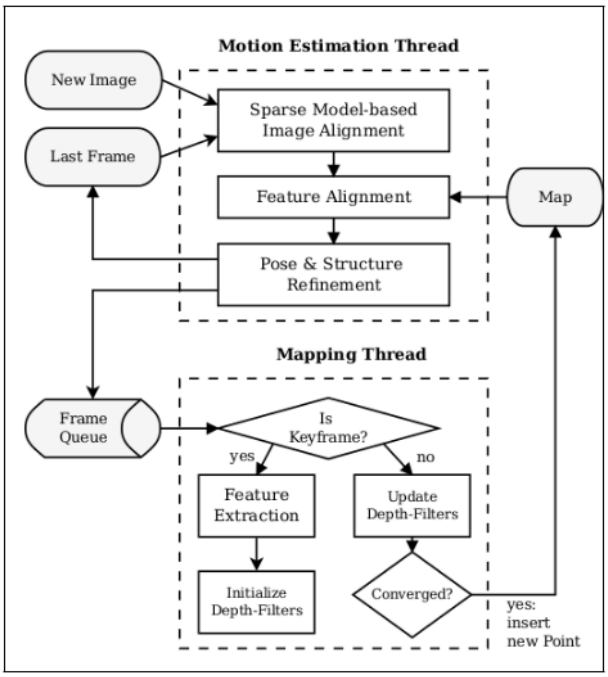


Figure 2-5: Semi-direct monocular tracking and mapping pipeline (Reproduced [74]).

2.4 Aerial 3D Mapping

3D mapping for aerial robots is also an emerging field in robotics. Though detailed 3D mapping is not in the scope of this thesis, we shall review the modern techniques to complete the overall understanding of the problem in the discussion. Researchers have used similar Simultaneous Localization and Mapping (SLAM) techniques for aerial mapping as the ones used for the ground robots. Nonetheless, the localization part differs in the case of aerial robotics as it has to deal with the 6-DOF instead of the commonly used 3-DOF for ground robots. Nuchter et.al [155] present a 6D SLAM which is based upon locally consistent 3D range scan and used ICP [217] to align them. They have used the cashed KD-tree data association technique to reduce the computational cost. D Borrmann

et.al [30] have proposed a 3D mapping approach that incorporates position and poses which is roll, pitch, and yaw, to create a globally consistent map and later used the Taylor Expansion and Cholesky decomposition to process huge 3D data in the real-time.

In aerial robots, [92] uses RGB-D image generated by the stereo camera along with the IMU readings to localize the mini quadrotor between two frames. After getting the pose and position, an occupancy grid is populated with the help of LiDAR scans to make a 3D map of the environment. [104] presents a novel method of mapping digital elevations from images of a stereo system taken from a low altitude. Extended Kalman Filter is used to estimate position and pose (6DOF) and 3D positions of features extracted from the images. [145] has presented a collaborative technique to build a common 3D map from the feeds of both ground and aerial robots. Merging of the map is done via a globally known common point that both robots deliberately visit during the course of their operation. Then iterative closest point algorithm is used to align both maps. Moreover, [151] constructs a 3D map from the digital imagery captured by the octocopter.

Another interesting work has been done by [61] which proposes to build large-scale 3D maps using direct monocular SLAM. As mentioned earlier in the previous section, direct approaches are those approaches that use image pixel intensities instead of extracted features from images. Extraction of features from an image is a time-consuming operation. Explicitly incorporation of detected scale-drift along with probabilistic noise depth estimation enables their approach to reconstruct a highly accurate large-scale environment map. The resulting map is semi-dense and in form of a point cloud. Overview of their complete algorithm can be seen in Figure 2-6

2.5 Navigation via Trajectory Tracking & Obstacle Avoidance

To the best of our knowledge, no one has previously done a water canal trajectory tracking via an aerial platform. However similar problems have been solved like road lane following which lies in the category of ground robots. However, lane following

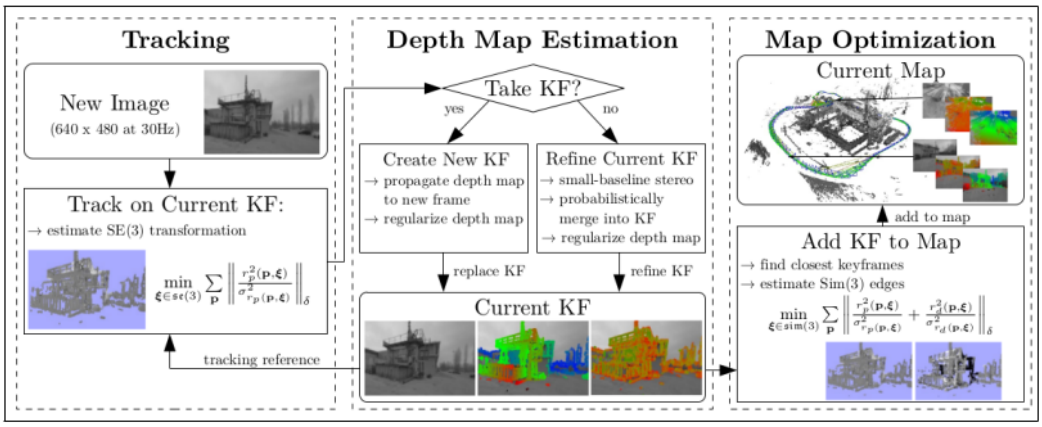


Figure 2-6: Large-scale 3D mapping approach which uses direct pixel intensities rather than extracted features for creating a 3D map (Reproduced [61]).

and tracking an unknown linear trajectory are well-studied phenomena in the field of ground robots. However, in aerial robotics, not much work has been done so far. In the case of ground robots, [6] has presented an approach to solving the trajectory and path following problem of an under-actuated robot by combining adaptive switching control with a non-linear Lyapunov based tracking control law. The tracked trajectory can be of any smooth shape parameterized by time. [95],[135],[136],[53] have used range sensor and visual feedback to detect and track pathway both roads and off-roads. [25] has used stereo camera pair to detect road and obstacles. For aerial robots, [175] provides an approach to visually lock a moving target and then track it using a predictive model in the line of sight. A solution to a similar problem was given by [75] as well. They have used small autonomous aircraft to follow a road using the visual and inertial feed. The system detects natural features of the scene and tracks the pathway by calculating yaw angle and displacement. [86] , [152] present methods to precisely track the trajectory generated. [176] has presented an electric pole tracking technique using visual servoing to monitor the health of the pole using an aerial platform.

Navigation and obstacle avoidance mainly depends upon the quality of the map

built in the first place. Normally maps are in the form of point clouds and processing millions of points is out of the question under a hard real-time system. To overcome this limitation, point clouds are first downsampled to minimum possible points using some approximation technique. Octomaps [210] are normally used for this purpose. Even after downsampling the point cloud, it is still a challenging task to traverse it and extract the required information. [186] presents an approach to traverse a 3-D known environment that avoids obstacles while keeping the altitude low. To update the 3D model of the environment while traversing, offline path information is fused with the online information coming from the visual feed.

B Call, R Beard, and C Taylor [38] have used a forward-looking onboard camera to detect features by Haris Corner Detector [87] and track them among the multiple successive frames to get the 3D location of salient features for the aerial robot. A 3D map has been created by using the tracked features and the features are hierarchically clustered [81] along with the traversing operation. Sliding mode control law [78] is used to avoid the obstacles. Z He et.al. [89] have presented a novel approach to solve the inherent uncertainty of the aerial platform and image noise which is introduced by the motion of the robot. A motion field information helps accurately estimate the ego-motion parameters which are used to develop the map and later have been used for obstacle avoidance. S Scherer et.al. [181] have used Discrete Transform Algorithm [140] to add, remove and modify the obstacles from the map. Since they are mapping a river over kilometers, so they have used a scrolling map approach to cater to the large map.

Part II

System Integration and Ground-truth Validation

Chapter 3

System Integration of an Unmanned Aerial System

In the previous part of the thesis, we have observed that though waterways serve as a backbone for global agriculture but are at a constant threat of deterioration due to bank erosion and silt accumulation over time. Hence, periodic inspections are necessary to estimate the structural damages caused over time as they significantly reduce water distribution efficiency. To address this problem, an aerial integrated sensor system has been proposed in this section of the thesis that enables a robotics platform to fly over the length of a water canal semi-autonomously and generate a 3D structural profile. Hardware integration involves multiple sensors, such as GPS, IMU, monocular camera, and LiDARs. Calibration techniques for different sensors have also been discussed in detail along with the data post-processing to increase the overall accuracy and efficiency of the proposed aerial sensor system.

3.1 Introduction, Background and Challenges in the Environment

As discussed earlier, Natural reservoirs and rivers are the main sources of irrigation for almost all agriculture-dependent economies in the world [103]. To utilize the maximum potential of these water resources, water is delivered to far-reach agricultural lands with the help of an extended waterways network. These waterways are designed in such a way that the water flows down freely under the effect of gravity for thousands of kilometers. Moreover, constructing waterways does not ensure the constant supply of water to far-reach lands as these structures are in a constant threat of degradation over time. Common threats include silt accumulation, bank erosion, fallen tree tusks, fallen garbage, and big rocks which flow down with the water during flooding [114]. Among these, the biggest threat is silt accumulation [21]. Silt is very small earthly sand-like particles called sediments that are produced from multiple factors like constant girding of rocks among water or mixed in water along the journey from upstream to downstream under the influence of gravity. When the velocity of water decreases, these sediments settle down on the bed of waterways and form silt accumulation. This drastically reduces the water carrying capacity of channels which not only results in insufficient water transportation but also damages the structure of the waterway like bank breaches [16]. Similarly, other factors like garbage, big rocks, and fallen tree tusks also hinder the smooth flow of water resulting in the inefficiency of the water canal network.

To avoid these structural threats, regular inspections of waterways are conducted periodically once every year. Large river basins such as Indus, Ganges, Nile, and Mississippi contain hundreds of thousands of kilometers of waterways. This makes manual physical structural inspection of these river basins nearly impossible. For instance, in the Indus basin alone, every year waterways are closed for only 20 days in January for inspection and repairs. Therefore, to properly inspect each water canal in the basin, an automated system is required as the manual inspection takes time and is practically unfeasible. As

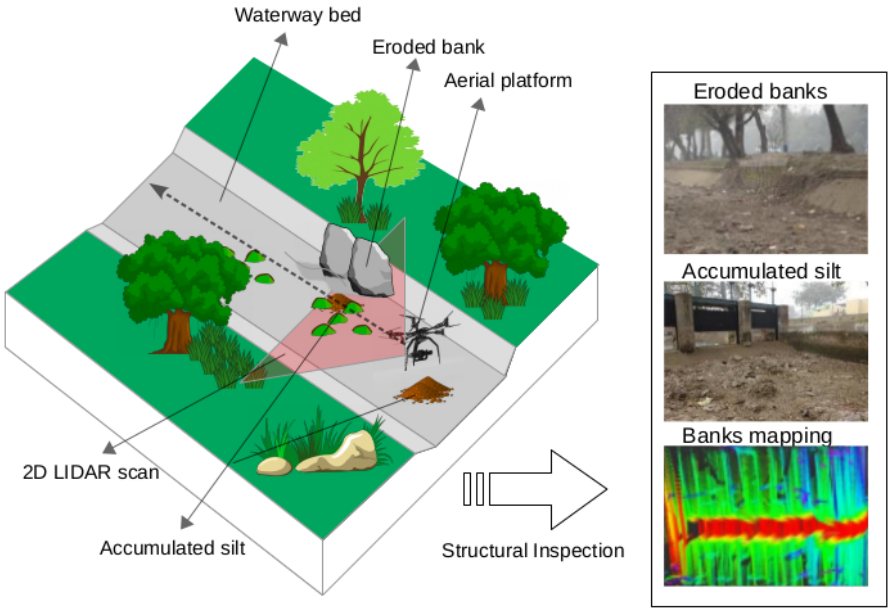


Figure 3-1: *Left* : An illustration showing an autonomous aerial platform (UAV) navigating the length of a waterway for structural inspection. *Right* : Shows structural deformation scenarios like eroded banks, accumulated silt which affect the efficiency of waterways.

Figure 3-1 shows, traversing waterways is quite a challenging task even when it is closed (dried) for inspection purposes. The use of the ground robot is out of the question as most of the canal bed is not traversable because of wet silt, garbage, rocks, bridges, wires, etc. Hence, a flying robot (UAV) can be used to traverse the canal length. Even for an aerial robot, there are a lot of challenges involved while canal traversals such as avoiding trees, bridges, electricity poles, wires, etc.

In this current chapter, we have proposed customization of an off-the-shelf multi-copter along with sensor integration and calibration, required for human-assisted semi-autonomous canal structural inspection. The proposed system has the capability of obstacle avoidance for both natural and man-made objects as far as the sensor systems are concerned. Moreover, navigation is relatively easy because of near-linear trajectories of water canals. In

the proposed system, we have equipped on-board multiple LiDARs, IMU, GPS, barometer, and monocular camera. The data from the mounted onboard sensors are sufficient to perform critical tasks such as detecting the current state of the aerial platform, following the trajectory of waterways, obstacles avoidance, and generating the 3D profile for structural inspection. Moreover, trajectory tracking, state estimation, and obstacle avoidance are real-time tasks. Hence, a high-end embedded processing unit has been retrofitted along with the $500GB$ of solid-state memory for data logging. All onboard sensor systems and an embedded processing board have a separate power supply that is large enough to support long flights.

3.2 Existing Research on State-of-the-Art Aerial Systems

Traces of aerial robotics goes back to the very start of the history of flights itself [67]. After the successful making of planes, aerial robotics came mainly to enhance aircraft designs for both military and civil purposes. Other than the fixed-wing planes, De [56] was the first person to give an idea of a multi-rotor flying machine. Since multi-rotor flying machines were inherently unstable, an efficient control system was required to keep them in the air. Afterward, a lot of work had been done on the control of multi-copters, however, rotor-crafts gained momentum in the early 2000s due to advancement in real-time embedded systems, their cost-effectiveness, and the production of accurate inertial navigation systems (INS).

In the early 2000s, due to the advancement in real-time embedded systems, rotor-crafts gained momentum. In 2002 Altug [8] has presented a novel approach of quad-rotor control using visual feedback. For the autonomous control, he used a ground-based visual camera system to detect the pose of the flying quad-rotor and proposed model-based feedback linearizing controller along with back-stepping inspired control law. Similarly, in 2005, Altug [9] has proposed an on-ground visual 6-DOF pose stabilization and tracking controller for a quadrotor. Waslander [205] has presented an Integral Sliding Mode and

reinforcement learning-based approach to control the non-linear disturbances caused by complex air inflow induced by four rotating rotors, especially for altitude hold near the ground.

After achieving robust stabilizing and control, momentum shifted towards flight autonomy. In 2007, Bouaballah [31] has worked on the modeling, design, and control for autonomous take-off and landing of a multirotor. Similarly, Zohar et.al. [220] have proposed an algorithm for multirotor attitude stabilization and control which they claimed to have improved vertical takeoff and landing (VTOL). Mellinger et.al. [143] have presented design, control, and autonomous planning for an off-the-shelve multirotor to claw an object while flying with the help of a motion capture system.

Later on, Mellinger et.al. [141] have proposed an algorithm for the generation of real-time optimal trajectories of quad-rotor in a tight indoor 3D space. Extending this, later they [198] have presented another trajectory generation and control approach for aggressive flight formation of multiple quad-rotors. Moreover, controlling these aerial machines with additional payload was another important and challenging research area. Pounds et.al. [164] has presented an analysis of a quad-rotor basic model with disturbances due to the payload on-board. They have shown experimentally that by using a simple PID control, we can easily avoid the flight disturbances caused by the payload. Similarly, Lee [117] and Mellinger [144] have proposed approaches to control multiple quad-rotors holding a common payload. Individual control design for each quad-rotor has been proposed such that it stabilizes the payload overall in a 3D work space along with the other drones.

One of the most challenging tasks is the state estimation of a flying drone. Lin et.al. [122] have presented an optimization-based state estimation approach that uses a monocular fisheye camera with low-cost IMU to estimate the accurate position of a micro-aerial-vehicle(MAV). For navigation, Papachristos [160] has presented an approach for autonomous exploration and path planning for an aerial inspection of terrain. Adhikari [5], Chen [43], Geng [82] have proposed approaches to track the trajectory of forest

trails using deep convolutional neural network. A similar approach has been used by Mansouri [134] for the trajectory tracking of underground mines using MAV. Moreover, Loquercio et.al. [126] have proposed an eight-layer residual convolutional neural network to fly drones autonomously around a city. The network took images as input and gave output in the form of collision probability and the steering angle for a drone. They have successfully flown a drone around an urban city. To conclude, a lot of research has been done regarding drones' state-estimation, trajectory tracking, mapping, and navigation in different applications.

No direct work has been done on aerial water canal structural inspection and mapping. Since the nature of the task is almost similar, we take inspiration from literature and develop the proposed system on it. Over the past few years, commercially off the shelf solution are available in the market for 3D mapping using drones [115],[52],[173],[45]. All of these systems are designed to be operated in an open and obstacle-free environment and have clear imagery with minimum occlusions. Our target environment of water channels is quite complex as discussed earlier as it contains a lot of occlusions, civil structures, and trees. To traverse the environment is complicated and requires a lot of different advanced sensor systems for accurate mapping and navigation.

3.3 Design Architecture of the Aerial System

109

The primary task of the proposed aerial system is to navigate autonomously along the length of a waterway and to generate 3D profile maps for structural inspection. For this purpose, the platform requires a variety of sensor systems on-board to fulfill this task. However, the integration of onboard multiple sensors also increases the payload of the flying machine. It even becomes more challenging to install high-density batteries for long flights with all the sensors working at full capacity. Figure 3-1 shows an illustration for a proposed operation using an aerial platform over a waterway. We can see that the nature of the task is inherently challenging but by lowering the forward velocity, precise 3D maps

can be created with minimum noise.

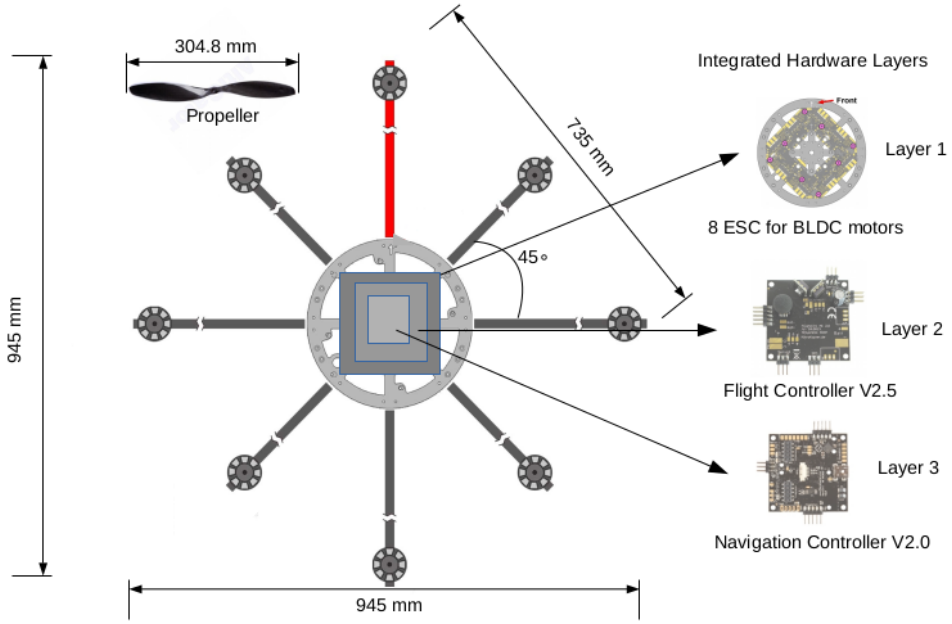


Figure 3-2: Selected off-the-shelf Mikrokopter OctoXL design along with the three layers of ESC, flight controller board, and navigation controller board.

The proposed aerial system uses MikroKopter *MK OctoXL* [84] as a flying platform as it can up-lift a maximum payload of 6 KG. This machine has eight propellers fixed on a frame at 45° to each other and has got three built-in layers of electronic speed controllers, a flight controller, and a navigation controller as shown in Figure 3-2. Layer 1 contains eight electronic speed controllers (ESC) for each propeller. Layer 2 consists of a flight controller board that acts as a motherboard and controls the flight. It also contains an air pressure sensor for better height estimation which can work up to 5000 meters above sea level. It also connects with the PPM-Sum signal receiver and is vibration resistant. Layer 3 contains a Navigation controller board which provides an interface to a compass and a GPS shield. It also gives autonomy functions such as position hold, waypoint following, and

Technical Specifications of ARF-MikroKopter OktoXL	
Drive	8 electric BL motors (14.8V)
Propeller size	$12''$ - pitch 4.5
Max motor speed	8200 rpm
Control	RC trans/receiver (ppm-sum)
Dimensions$(W \times L \times H)$	$735mm \times 735mm \times 360mm$
Weight(without battery/payload)	2050g
Maximum payload	3020g
Operational range	4 km

Table 3.1: Detailed technical specification for the selected aerial platform

home-coming(return to the starting point). The navigation controller and flight controller layer communicate with each other via an SPI protocol.

Table 3.1 shows a more detailed specification overview of the used octocopter.

3.4 On-board Customized Sensor Integration

Multiple sensors are installed on an aerial platform enabling it to perform aerial canal structural inspection and 3D environment sensing as shown in Figure 3-3. Mounted sensors include an Inertial Measurement Unit (IMU), two LiDARs for profiling and obstacle avoidance, Global Positioning System (GPS), and a monocular camera. Each sensor system is calibrated accordingly and data post-processing is done to make it more precise and accurate. To further elaborate on the calibration and data post-processing, we shall discuss each sensor system in detail below.

3.4.1 Global Positioning System (GPS)

For global positioning, the system has MK GNSS V4 which is also the recommended GPS shield for selected octocopter. It has a direct interface with the navigation controller board via a serial connection and has a built-in compass. Other than receiving signals from the American GPS satellites, the GPS device can also localize itself by using Russian

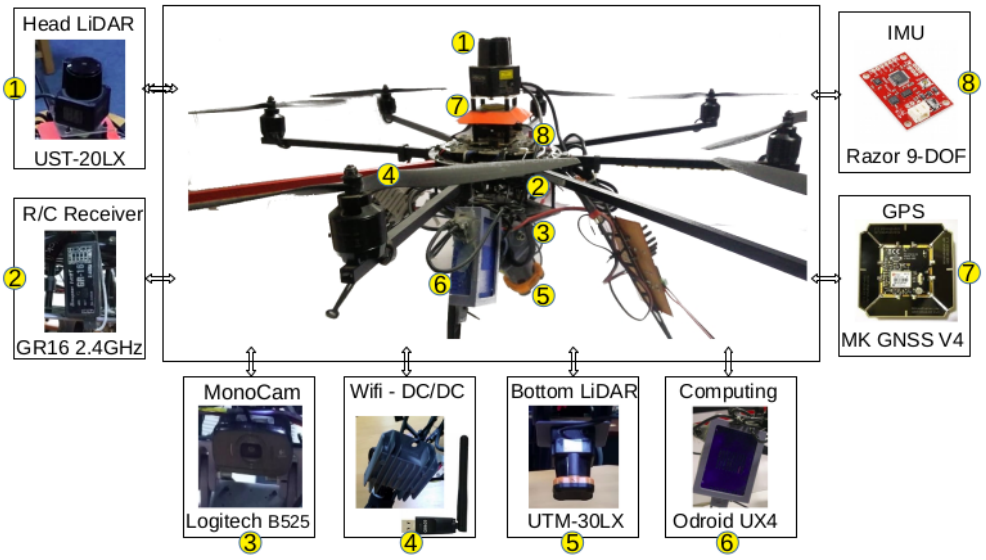


Figure 3-3: Detailed identification of each sensor system mounted on the multicopter for an aerial canal structural inspection.

GLONASS, European Galileo, or Chinese BeiDou satellites. This increases the efficiency and accuracy of GPS as a whole. Moreover, to further increase the accuracy of the MK GNSS V4 GPS shield, a Kalman Filter has been used as a post-processing step inspired by [208].

Suppose $\mathbf{x_t}$ be a state vector representing the GPS readings such that

$$\mathbf{x_t} = [x_{long}, x_{lat}, x_{alt}]^T,$$

where x_{long} represents longitude, x_{lat} represents latitude and x_{alt} represents altitude. Since altitude is independent of location, process model can be given as:

$$\mathbf{x_t} = \mathbf{x_{t-1}} + \kappa \mathbf{T} \delta_{t-1} + \xi_{t-1} \tag{3.1}$$

where $\mathbf{x_{t-1}}$ represents the previous state, κ is an auto-correlation parameter to reduce the randomness of the linear motion model, $\mathbf{T}$ is the rotational transition matrix which only depends upon mean turning yaw angle ψ as

$$\mathbf{T} = \begin{pmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

δ_{t-1} is the difference between two previous states x_{t-1} and x_{t-2} and ξ_t is a Multivariate Gaussian Distribution with zero mean and co-variance of

$$\Sigma = \begin{pmatrix} \sigma^2_{long} & \rho\sigma_{long}\sigma_{lat} & 0 \\ \rho\sigma_{long}\sigma_{lat} & \sigma^2_{lat} & 0 \\ 0 & 0 & \sigma^2_{alt} \end{pmatrix}.$$

Here σ^2_{long}, σ^2_{lat}, and σ^2_{alt} represents the variances in x_{long}, x_{lat}, and x_{alt} respectively. ρ is the co-relation coefficient. Noise in x_{alt} is independent of x_{long} and x_{lat}, therefore, co-relation against x_{alt} is zero.

Once we have the prediction of an unobserved state, now we relate the predicted state with the observed one through GPS sensor y_t. Before relating, an important step is to sync both the states at a common sampling rate. Although GPS is working at 5Hz, practically its data is not continuous. We may observe gaps in data receiving. The solution is to use the model of data regularization for irregular data in the state-space framework such that we assume that straight motion has occurred during gaps. Suppose l represents the location index which corresponds to a time instance between t and $t-1$ for which we have no GPS observation. We define m_l as a proportion of regular time interval as robot moves between x_t and x_{t-1} at which the l needs to be observed. Therefore, the measurement equation

becomes:

$$\mathbf{y_t^l} = m_l \mathbf{z_t} + (1 - m_l)\mathbf{z_{t-1}} + E_t \quad \forall \quad (0 < m_l < 1) \tag{3.2}$$

where $\mathbf{z_t} = \{z_{long}, z_{lat}, z_{alt}\}$ is the measurement current vector, $\mathbf{z_{t-1}}$ is the previous measurement vector, $\mathbf{y_t^l}$ shows the observation at index l between two observation z_t and z_{t-1} and E_t is a GPS noise modelled as Normal Random Distribution with zero mean and co-variance. All three observations are independent in nature, hence co-variance matrix E_t is diagonal.

3.4.2 Inertial Measurement Unit (IMU)

Location and altitude provided by GPS are not sufficient for precise localization. Moreover, the pose is also an essential part of state estimation. Both precise pose and location are required at high frequency to track the motion of an octocopter during the flight which also serves as a reference point for all onboard sensor readings. MEMS-based SparkFun Razor 9-DOF IMU has been used in the proposed system which has triple-axis digital-output gyroscope, 13-bit resolution triple-axis accelerometer, and triple-axis digital magnetometer. The accelerometer, gyroscope, and magnetometer are calibrated each time before flights for accurate readings.

During the flight, there are a lot of vibrations across the chassis of the octocopter due to the high RPM of the thrust motors. These vibrations produce data noise in all axis of a calibrated accelerometer which makes it significantly inaccurate. Even after using vibration-absorbing material around IMU, there is still noise due to the inherently noisy nature of MEMS accelerometers. Moreover, further noise reduction has been done using a moving average (Finite Impulse Response) filtering approach [42] applied on each axis of the accelerometer. Moving average filter takes N number of sensor readings and averages it to give one single smoothed output. The smoothness of the filter depends upon N, for a larger value, the output will be smoother but may lose meaningful rapid transitions in

accelerations. Therefore, the filter has a poor frequency response but good time-domain response. Suppose each axis of accelerometer is define by a vector $\mathbf{x_t} = \{a_x, a_y, a_z\}$ where a_x, a_y and a_z are the accelerations on x-axis, y-axis, and z-axis respectively. Since we have applied the filter in discrete-time hence moving average filter with an output of $y[t]$ and input of N readings is written as a difference equation

$$\begin{aligned} y[t] &= \frac{1}{N}\sum_{i=0}^{N-1}\mathbf{x}[t-i] \\ &= \underbrace{(\frac{1}{N}\mathbf{x}[t])}_{unit\ 1} + \underbrace{(\frac{1}{N}\mathbf{x}[t-1])}_{unit\ 2} + ... + \underbrace{(\frac{1}{N}\mathbf{x}[t-(N-1)])}_{unit\ N}. \end{aligned} \tag{3.3}$$

Equation 3.3 shows that at a given time instance t, the filter takes current and $N-1$ last sensor readings and calculates a single output $y[t]$ by simply calculating average. In practice, executing Equation 3.3 on a computing machine causes a delay at every computing unit step mainly because of two reasons: one is the data structuring of accelerometer inputs in an array and processing them, second is the delay caused by D Flip-Flops at a hardware implementation level. In a real-time system, these delays cause unnecessary complications. However, these delays have been improved by using a signal processing approach of multiplying z^{-i} with the Z-transform $X(z)$ of an input signal $x[t]$ at every unit delay. Moreover, input-output relationship of N-value moving average filter can be written as a transfer function of

$$H[z] = \frac{Y(z)}{X(z)} = \frac{1}{N}\sum_{i=0}^{N-1}\mathbf{z}^{-i}. \tag{3.4}$$

Figure 3-4 shows the frequency response of the proposed transfer function.

3.4.3 Monocular Camera

Visual perception is added in an aerial waterway inspection by using an onboard monocular camera. Other than environment visualization, the system has used a monocular

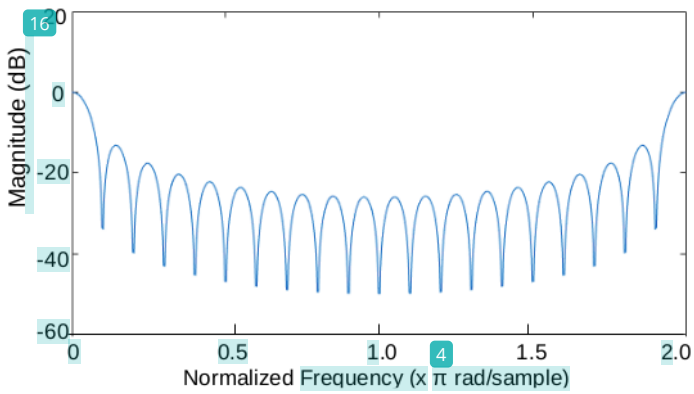


Figure 3-4: The frequency response of the proposed transfer function for N=20.

camera to help improve state estimation by using visual odometry. A high-end Logitech webcam with 720p support with 70° field of view lens has been used and it works at 30 FPS. A simple pinhole camera model has been utilized to relate a real-world point $P = (x_w, y_w, z_w)$ to a pixel location $\mathbf{x_i} = (x_p, y_p)$ as

$$\mathbf{x_i} = sCWP, \quad where \quad C = \begin{pmatrix} f_x & 0 & C_x \\ 0 & f_y & C_y \\ 0 & 0 & 1 \end{pmatrix}, W = [R \quad \mathbf{t}]. \tag{3.5}$$

Here, C is the camera matrix such that

$$x_p = f_x \left[\frac{x_w}{z_w}\right] + C_x \quad \& \quad y_p = f_y \left[\frac{y_w}{z_w}\right] + C_y \quad ,$$

where C_x and C_y represent the possible displacement of the center of the imaging screen from the optical axis, f_x and f_y are the two different focal lengths used because the image screen receptors also called as "focal plane array", are rectangular rather than square. Hence, the focal length is defined as a product of the total number of pixels represented by an image receptor in the x-axis 's_x' and the focal length of the lens 'f'. This can also be

written as $f_x = s_x f$ and $f_y = s_y f$ for y-axis. In Equation 3.5, s is the scale factor to indicate the validity of homography. R and $\mathbf{t}$ represent the rotation and translation respectively which relates the projected image plane inside the camera to the actual viewing image plane. Since the homogeneous coordinate system has been used, we can sum both rotation and translation into a single matrix W. These errors are normally due to a manufacturing fault and are quite common in low-cost cameras. Parameters, f_x, f_y, C_x, C_y are also called the "Intrinsic Parameters" of monocular camera.

Other than the intrinsic parameters, the lens also adds distortion into the projected image because of the uneven magnification of the image. It is mainly because of radial and tangential distortions. Radial distortions are due to the non-perfect spherical nature of the lens which causes magnification rings by magnifying all pixels evenly at the same distance from the image center. Tangential distortion occurs when a focal plane array of a camera is not exactly at 90° to the optical axis of the lens. It causes an image to wrap non-linearly. To have a perfect relation between the world-point and the pixel location, we need to solve both the distortions caused by the lens. In robotics, a common approach to solve the lens distortion errors is inspired by [76]. A similar approach has been used to remove the distortions. For Radial distortion, the first few terms of the Taylor series approximation have been used to exploit the varied nature of Radial distortion that is zero at the center of the image and increases as we go away from the center. Assuming first three steps of Taylor series with h_1, h_2, and h_2 as coefficients, we have approximated the effect of Radial distortion as

$$\hat{x_p} = x_p(1 + h_1 r^2 + h_2 r^4 + h_3 r^6), \tag{3.6}$$

$$\hat{y_p} = y_p(1 + h_1 r^2 + h_2 r^4 + h_3 r^6), \tag{3.7}$$

where $\hat{x_p}$ and $\hat{y_p}$ are the estimated pixel locations in the x-axis and y-axis respectively, x_p and y_p is the original pixel location and r is the radius from the center of the image where the radial distortion is zero. In addition, for removing tangential distortion effects, the "Plumb bob model" approach [35] has been used as

$$\hat{x_p} = x_p(2j_1y_p + j_2(r^2 + 2x_p^2)) \tag{3.8}$$

$$\hat{y_p} = y_p(2j_2x_p + j_1(r^2 + 2y_p^2)) \tag{3.9}$$

where r is the radius from the center of an image, j_1 and j_2 are the coefficients of tangential distortion, x_p and y_p represent original pixel location in a projected image, $\hat{x_p}$ and , $\hat{y_p}$ represent estimated pixel location after removing tangential distortion.

From Equations 3.5,3.6, 3.7, 3.8 and 3.9, we have 9 unknown parameters $(f_x, f_y, C_x, C_y, h_1, h_2, h_3, j_1, j_2)$ necessary to determine unknown transformations in pixel space. However, unknown transformation can be determined if both starting and ending states are known. Hence by using a deterministic object such as chessboard with at least 9 feature points, unknown parameters can be derived. However, [32] shows that maximum 8 unique parameters can be derived using a single planner calibration chessboard. Therefore, multiple images of calibration plane has been used either by moving the camera or by moving a calibration board. To solve these equation for unknown parameters with known start and end states, Levenberg-Marquart optimization approach [138] has been used, which is to minimize the re-projection error E between $(\hat{x_p}, \hat{y_p})$ and (x_p, y_p).

$$E = \frac{\sum^k (\hat{x_p} - x_p)^2 + (\hat{y_p} - y_p)^2}{k}, \tag{3.10}$$

where E is the re-propagation error and k represents total number of features in all input

images. Once we know the intrinsic and extrinsic parameters of the camera, we can rewrite the calibrated pixel location (x_{cali}, y_{cali}) against the projected pixel location (x_p, y_p) by using equations 3.6, 3.7, 3.8, 3.9 and 3.10 as,

$$x_{cali} = x_p(1 + h_1 r^2 + h_2 r^4 + h_3 r^6 + 2j_1 y_p + j_2(r^2 + 2x_p^2)), \tag{3.11}$$

$$y_{cali} = y_p(1 + h_1 r^2 + h_2 r^4 + h_3 r^6 + 2j_2 x_p + j_1(r^2 + 2y_p^2)). \tag{3.12}$$

3.4.4 2D Laser Scanners (LIDAR)

For environment sensing and profiling, two LiDARs have been used called a head-LiDAR and a bottom-LiDAR. Head-LiDAR has been mounted on the top of the drone parallel to the xy plane, while bottom-LiDAR has been mounted at an angle of 45° to the xy plane as shown in the Figure 3-3. The principle working of LiDAR is similar to SONAR or RADAR but instead of using sound or radio waves, it uses a laser (light waves). It calculates the time taken by a laser to hit an object and return to get the proximity reading. For bottom-LiDAR, Hokuyo UTM-30LX has been used which has a scan a field of 270° with the maximum range of 30 meters. It has 0.25° of angular resolution and it takes 25 ms to scan the whole plane. It is designed for aerial applications as its weight is barely 370 grams with decent accuracy of $\pm 50mm$. For head-LiDAR, Hokuyo UST-20LX has been used on the top of the octocopter for localization and obstacle avoidance. This LiDAR is mounted parallel to the horizontal plane of the octocopter. It has a maximum scan range of 20 meters and an angular resolution of 0.25°. For localization, co-related features have been extracted from consecutive laser scans and 2D transformation is calculated. Similarly, proximity returns are used to avoid obstacles during the navigation above the water canal.

3.4.5 On-board Embedded Computing

Since the aerial platform is subject to navigate autonomously over the length of the waterway, a powerful onboard computing resource has been installed which is capable of handling real-time tasks. Though environment mapping and waterway structural profiling can be performed offline, state estimation, obstacle avoidance, and navigation are done onboard. Large memory space of $500GB$ has also been installed to store all the data from sensors. Almost all sensor systems are serially interfaced with the onboard embedded computer except both the LiDARs which use Ethernet to transmit their data. To access the onboard computer, a direct 802.11 link has been established between the ground station and the aerial platform.

3.5 Experimentation & Sensor Data Visualization

Figure 3-5: Test flights of fully equipped human-piloted proposed octocopter over different water canals. $Top\ left$: Aerial robot is being prepared before a test flight which is being overwhelmed by the local farmers.

Once the whole aerial system has been developed, a lot of trial flights have been conducted over the real water canals. All flights have been assisted by a human operator and multiple waterways with significantly different surroundings have been chosen as shown in Figure 3-5. Data from all the sensor are stored on-board and is later visualized offline.

Figure 3-6 shows noise reduction at each accelerometer axis after applying the Finite Impulse Response filtering. This significantly increases the accuracy of estimates of pose and position for an onboard state estimation module.

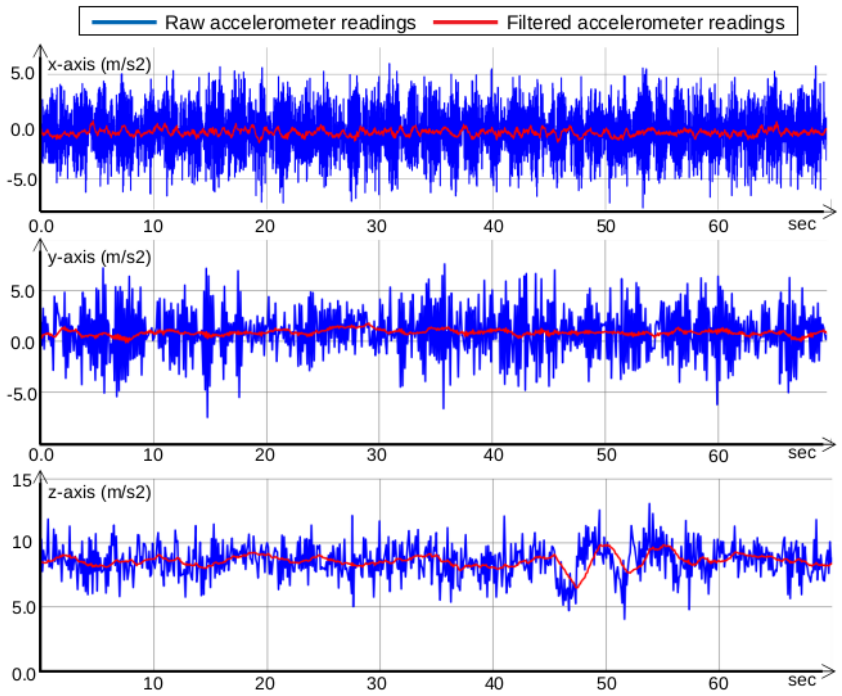


Figure 3-6: A comparison showing real-time noise reduction of accelerometer data on each axis by using Finite Impulse Response filtering approach during a flight over a water canal.

For a more comprehensive understanding and to visualize the output of LiDAR sensors,

Figure 3-7 shows a frame from the LiDAR data for both head and bottom laser scanners during a test flight.

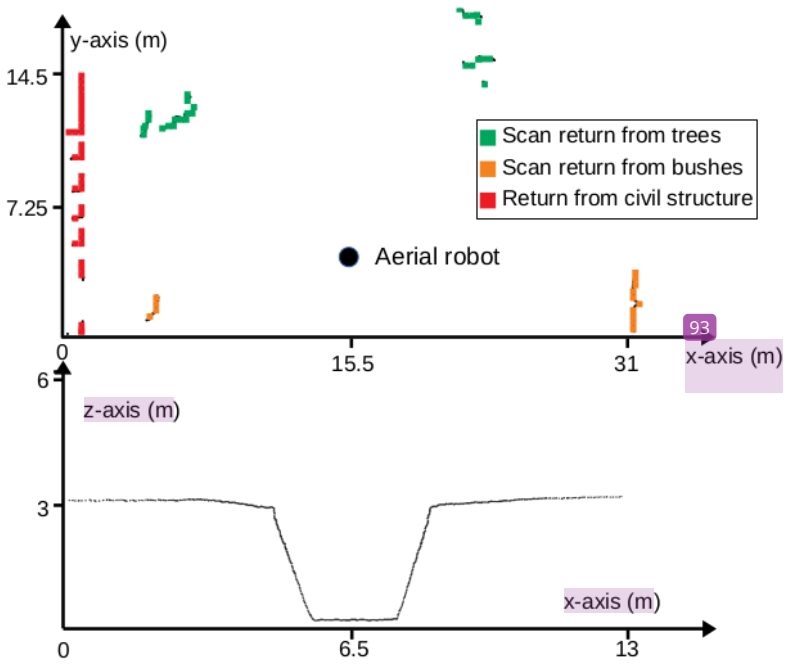


Figure 3-7: *Top* : Output of the *Head LiDAR* in a pointcloud form to identify obstacles around. *Bottom* : Output of the *Bottom LiDAR* showing the cross-section of an empty waterway in a pointcloud with both banks quite visible.

The data from the GPS, IMU, and the monocular camera has been further visualized and analyzed in the next section of this thesis.

In this chapter, we have proposed the design and development of an aerial platform equipped with multiple sensor systems required for the inspection of the water canal structure. In the next chapter, we present a novel technique to use an external fiducial marker for the ground-truth analysis and calibration of the estimation of the state of the proposed aerial drone. It serves as a backbone for the whole aerial system as every onboard sensor measurement depends upon it. Our proposed novel technique in the next chapter

not only increases the accuracy of the fiducial marker for ground-truth analysis but also makes it more robust in robotics applications, especially for our aerial problem.

Chapter 4

Analysis and Improvements in AprilTag Based State Estimation

In this chapter, a novel technique has been proposed to estimate the accuracy of the state estimation module using an external fiducial marker.

4.1 Introduction

Localization capability is the backbone of many robotic systems as it helps determine the state of the robot at a given time instance [119]. Many essential subsystems of an autonomous mobile system take localization as an input for developing maps or plan navigation strategies [195]. The nature of the application determines the level of localization accuracy required. Localization accuracy is commonly measured by comparing it with the ground truth at any given time instance. Therefore, the ground truth itself must be of superior accuracy to minimize the error in measuring localization accuracy. In robotics, there exist several ways of generating ground truth. A standard method for generating ground truth for localization is with the help of high precision motion caption cameras (MoCap) [146]. This system is considered to be the benchmark for many indoor localization

systems worldwide. MoCap setup contains multiple cameras calibrated at known positions, fuse the data to track a known marker at high accuracy. For indoor applications, another commonly used method is the use of fiducial or visual marker-based localization systems. This method is quite popular because of the readiness to use it. For many applications, fiducial markers' relative ease of use makes them the primary method for localization [69]. Besides being used for outdoor applications, GPS is the most popular source for ground truth verification [171]. Although it is globally consistent, nominal GPS systems do not provide adequate accuracy for tasks that demand sub-meter localization accuracies such as robot navigation, obstacle avoidance or structural inspection in confined environments. Some high-end GPS methodologies such as D-GPS and RKT-GPS have an accuracy of 0.1 meters or less but they are quite expensive and are hard to setup. In outdoor environments, the deployment of fiducial marker-based systems are also possible but they have limitations on operating distance and field-of-view.

AprilTag is one of the most commonly used fiducial markers that can be used both indoors and outdoors for ground truth generation in 6-DOF, but with limitations [157]. We have precisely identified these limitations and have explained the source of these limitations with statistical error models. The proposed research has established that both distance and orientation of viewing camera from the target tag effects accuracy. However, uncorrected orientation uncertainty is a more significant source of accuracy degradation. AprilTag's accuracy is maximum when the viewing camera is pointed towards the center of the tag. Moreover, in the current implementation of the AprilTag localization system, this source of error is left unaddressed. As a result, the system suffers from a loss of performance, which is rectifiable. The proposed research has filled this gap (only for 2D) via an empirical analysis of the AprilTag system. Furthermore, a data-driven probabilistic sensor model has also been proposed, which works both in indoor and outdoor environments.

In this chapter, we have proposed techniques to overcome this limitation and to increase the accuracy even for wider horizontal viewing angles. The proposed technique consists

of three approaches. One is a geometric *soft correction* to the displacement angle from the center of the tag. Second is a *active correction* to the angular displacement using a custom-built gimbal which detects the tag in real-time and physically keeps the camera viewing angle towards the center of the AprilTag horizontally. The third is a proposal of a probabilistic sensor error model of the AprilTag by using Gaussian Processes (GP) based regression of experimental data. The forward sensor model is directly usable in a standard Baye's filter for localization, mapping, SLAM or exploration algorithms [195]. We have used these approaches in combination and a detailed comparison is also presented to determine how different approaches have improved the overall precision. For example, in an ideal scenario, we have improved the accuracy from 4.4 cm to 0.8 cm in the x-axis and 2.56 cm to 0.54 cm in the y-axis. Moreover, other than the accuracy, improvement in the precision has also been achieved from 112 cm^2 to 0.29 cm^2 for the x-axis and 14 cm^2 to 0.60 cm^2 for the y-axis over a target distance of 70 cm. All the AprilTag measurements used in data error comparisons are raw and without modifications.

In Section II, an overview of the related work on visual marker systems has been discussed. This section describes different fiducial markers, their techniques and their applications. Section III illustrates the problem set up and the evaluation of AprilTag as a localization system. In this section, the implementation methodology of AprilTag is briefly discussed, then details regarding transformations required for trajectory generation is discussed. Then the error measurement setup is explained along with the method for taking measurements and lately identification of the AprilTag's shortcomings. Section IV discusses the reasons behind the identified shortcomings and proposes improvement techniques. Lately, a detailed comparison of all the proposed improvement approaches is presented. Afterward, a probabilistic sensor model for AprilTag has been proposed by using the Gaussian Processes (GP) regression along with the experimental verification of the proposed sensor model by implementing trajectory tracking using particle filter both in a laboratory setup and in an outdoor environment. Lastly, Section V concludes the whole

chapter with the identification of future work required.

4.2 Background and Existing Literature

A visual fiducial system uses 2D coded information embedded on a tag to give the position and the orientation of the marker to the camera. The 2D coded information also distinguishes between one marker from the other. Distinct fiducial systems are being used in robotics applications for pose estimation. All are best known for their use in augmented reality applications to support vision-based tracking [158]. Table 4.1 shows an overview of different commonly used fiducial markers in robotics application along with their key features.

Table 4.1: Commonly used different fiducial markers with key features.

Tag Names	Key Features
ARToolkit [107]	Use solid black outline for quick and robust detection.
Multi-ring Marker [48]	Use color rings instead of black marker for more robust detection.
TRIP [125]	Use a 2D circular mark for location identification.
ARTag [68]	Robustness marker detection against different lightening conditions.
ARToolKitPlus [200]	ARToolKit algorithm has been optimized for embedded devices.
Fourier-Tag [212]	Use robust tag encoding scheme using the phase spectrum of a 1-D signal (gray-scale).
RUNE-Tag [23]	Use perspective properties of circular dots for high accuracy and robustness.
CircularTag [60]	Use circular nature and non-linear optimization to further increase accuracy.
AprilTag [157]	Use stronger digital encoding, robust against different lighting conditions and occlusions.

In 2005, Fiala [68] has proposed a new visual fiducial marker called `ARTag`. To give robustness to marker detection against different lightening conditions, he has used digital coding theory to reduce the false positive detections by maximizing the Hamming distance

between different markers coding. Wagner et.al. [200] have proposed an improved version of `ARToolKit`, which is further optimized and can run on embedded hand-held devices.

Other then the squared or rectangular shape fiducials, circular fiducial are also used for 6-DoF localization. Xu et.al. [212] have presented a circular sinusoidal tag which stores data as the frequency spectrum of a synthetic image called `Fourier-Tag`. They have also presented a robust tag encoding scheme using the phase spectrum of a 1-D signal (gray-scale) to store digital content. The paper has not shown any accuracy or precision analysis for the proposed marker. Similarly, Bergamasco et.al. [23] have also presented a circular fiducial marker system in a form of sizable dots called `RUNE-Tag`. They have used the perspective properties of circular dots and have shown high accuracy and robustness. However, due to the concentrated nature of the dots in the proposed `RUNE-Tag`, maximum tag recognizable distance is reduced due to merging of dots for a low-resolution camera. Nonetheless, it has performed robustly against any occlusion up to 70 percent which makes it a suitable choice for short distance outdoor applications. Edwards et.al. [60] have also presented a circular fiducial marker for ground truth analysis. They have used the circular nature of the tag to optimize the error between the predicted tag's appearance and actual tag's appearance using a non-linear optimization routine which has helped them increase the accuracy manifolds.

In 2011, Olson [157] showed that AprilTag surpasses its predecessors in terms of detection rate, inter-coding Hamming distance, scale and angular-accuracy. Olson has also addressed the accuracy issues of AprilTag related to tag-detection percentage along with the camera distance from the tag. However, the results are not extensive enough to use them in creating a perfect sensor model for AprilTag and missed some necessary details which follow in the next sections. Nonetheless, because of more robustness and accuracy of AprilTag [157], many researchers have preferred it over any other visual fiducial markers so far. In 2017, Sagitov et al. [178] compared ARTag, AprilTag and CALTag for occlusions and showed that AprilTag is robust against small occlusions.

One of the advantages of AprilTag is the utilization as a low-cost localization solution in augmented reality and robotics applications. The setup requires only a monocular camera and a printed AprilTag on a paper. As a result, researchers prefer fiducial markers over other high-end localization systems. C Feng et al. [66] have used AprilTag as spatial indices for operations like navigation and inspection inside a building for engineering, construction and management related tasks. They have placed AprilTags on different parts of the building, which direct users with operation-specific information when seen through a mobile camera. Li et al. [120] combine the fiducial marker with inertial sensors to have an improved position and pose tracking of hand-held augmented reality system. They achieved an accuracy of 1.77 cm for the position and 4.15° for orientation estimation. Some researchers have used AprilTag as a landmark and track it in robotics applications. Wang et al. [202] and Wang [203] have proposed a vision-based vehicle tracking system in which an unmanned aerial vehicle (UAV) tracks a ground vehicle by using AprilTag attached to a ground vehicle. Ling et al. [123] have used AprilTag attached over a water vessel, to autonomously land an unmanned aerial vehicle (UAV) over it. Similarly, Zhang et al. [216] have used an identical approach to land the aerial vehicle over land using AprilTag. Later, a similar work regarding the autonomous landing of a quadrotor using AprilTag is done by Reference [100]. Tang et al. [193] have proposed an algorithm to fuse the data from multiple cameras and a 2D laser scanner. They have used an array of AprilTags as a target for calibration and employ a non-linear optimization technique to estimate a single camera intrinsic parameters out of multiple cameras and later fuse them with 2D laser scanner data to have an improved position and pose estimate.

Another advantage of AprilTag is for the accurate evaluation of individual localization systems or algorithms. Ramirez [168] has made a dataset for visual odometry and localization in which he used AprilTags as landmarks for accuracy evaluation. Parkison et al. [161] have used AprilTag to evaluate the position and pose of a micro aerial vehicle (MAV) for automated indoor RFID inventorying. Raina et al. [167] have used multiple AprilTags as a

ground truth evaluation system for 3D pose estimation in a cluttered environment. Maragh [137] has used AprilTag to control the position and angular velocity of a rotating body using PD control. She has also demonstrated the upper limit of the angular velocity of a moving object for the robust detection of AprilTag. Similarly, Zake et al. [214] have used AprilTag measurements as a ground-truth value to compare the output of the proposed pose-based visual serving technique for cable-driven robots. Florea et al. [73] have used AprilTags to localize a drone and other multiple waypoints as they have proposed a sensor fusion technique for localization by using numerical P systems. Researchers have also used AprilTags for modeling the dynamics of different physical systems. Britto et al. [34] have used fiducial marker AprilTag to estimate the position and orientation of an unmanned underwater vehicle (UUV), later used in the dynamic model of the underwater system. Fuchs et al. [77] have used AprilTag for the kinematic modeling and trajectory generation of a trailer attached to a truck using Kalman Filter [106]. Nissler et al. have used AprilTag for the robot to camera calibration to get the exact pose of each robot part in the camera frame of reference for precise operations. Mueggler et al. [147] have used multiple AprilTags to precisely estimated the position of an aerial vehicle in a swarm rescue operation. They have successfully demonstrated in a laboratory setup because localization from AprilTags has played an integral part in the completion of the task. Xie et al. [211] have used AprilTag to find the pose and extrinsic of multi-camera and multi-LiDAR system. They have shown that by using AprilTags in calibration process improves the overall robustness and accuracy of an autonomous driving platform. Later Nissler et al. [154] have used single and multiple AprilTags with a high-end camera to estimate the position and orientation of a manipulator in an industrial environment. They have shown that the use of AprilTag can help increase the precision tasks of a manipulator. Similarly, De et al. [57] have used AprilTag for the pose estimation in visual-inertial navigation of a real-time MAV application in an indoor environment.

The disadvantage of using AprilTag as a localization system may result in erroneous

localization due to multiple factors. These factors include configurations such as viewing angle, distance and camera rotation around its axis. Though AprilTag has been used in many applications ranging from virtual reality to tracking and localization, there are not many studies related to a systematic analysis of how the inaccuracy propagates over different distances and viewing angles. A similar study regarding the accurate evaluation of a similar fiducial marker (ARToolKit) has been conducted by Abawi et al. [1]. They have experimentally calculated the accuracy of ARToolKit, which is a similar fiducial marker as AprilTag but far less robust and accurate, as demonstrated by Reference [157]. They have given a conclusion that ARToolKit is accurate for short distances and for viewing angle between 40° and 80°. Furthermore, Wang et al. [201] have also proposed improvements in AprilTag but those improvements are limited to improving tag detection and lowering computational utilization and called it `Apriltag 2`. In 2017, Jin et al. [102] showed that the AprilTag pose output is inaccurate and noisy. They have proposed that by adding depth information along with the RGB information of the tag improves the overall pose accuracy. They have used an RGB-D camera to detect an AprilTag in an indoor setup. However, the proposed method fails outdoors as the RGBD camera does not work outdoors in direct sunlight. Zhenglong et al. [218] have used multiple AprilTags to estimate the pose of a flying quadrotor better. They have experimented in an indoor environment by laying multiple AprilTags on the floor and have flown a multirotor with a down-looking camera. A Kalman Filter with a constant velocity model has been used to estimate a more accurate pose by fusing poses from multiple AprilTags. It is shown that it has improved the overall pose estimation and matched it with a Motion Capture (MoCap). However, for a large outdoor environment, it is not possible to lay multiple AprilTags on the ground beneath a flying robot all the time for pose correction, so this makes the proposed approach not suitable for a large outdoor environment. In 2019, Kayhani et al. [108] have proposed that the raw AprilTag pose is not accurate enough for autonomous operations and has improved the accuracy of an indoor multi-copter by fusing pose data from multiple AprilTags with

the help of an Extended Kalman Filter.

Some researchers have evaluated the pose accuracy of the AprilTag in indoor environments and have improved it by using multiple tags along with data fusion techniques such as Kalman Filtering. Though they have improved the detected pose accuracy by fusing data from multiple tags, their proposed setup can only be possible in small indoor environments. For a large outdoor environment, it is still an open question. Moreover, AprilTags can be used for ground-truth analysis in many autonomous applications such as self-driving cars [162]. Hence, improving AprilTag accuracy to the point where it serves as a ground-truth solution, especially outdoors, is still an open challenge.

4.3 Problem Setup and System Evaluation

4.3.1 AprilTag Working Principle

As described in the earlier section, AprilTag also uses an embedded 2D-coded marker for tag detection and to differentiate it from the other tags. The visual marker tag can be of any size with a square dimension. The tag is printed on a white background with a black outline square. Inside the square is an embedded black bar-code. AprilTag [157] uses a unique detection algorithm for fast, robust detection and to minimize the effect of small occlusions. Figure 4-1 shows the algorithmic steps of AprilTag. In the first step, it computes the magnitude and direction of a gradient at every pixel in an image that contains the AprilTag. Afterward, these calculated gradients are grouped into clusters called components based on similar gradient attributes using a graph-based method. By using a weighted least square technique, a line is fitted on every component such that the direction of the gradients determines the direction of the fitted line. Moreover, gradient direction determines the direction of the line segments. Hence each line has a dark side on its left and a lighter side on its right. Furthermore, after identifying all lines, possible quad shapes are detected, as shown in step 3 of Figure 4-1. The quad shape with a valid code

scheme is extracted out. Also, a 6-DOF pose of the tag in the camera frame of reference is returned by using homography and intrinsic estimation over an extracted tag.

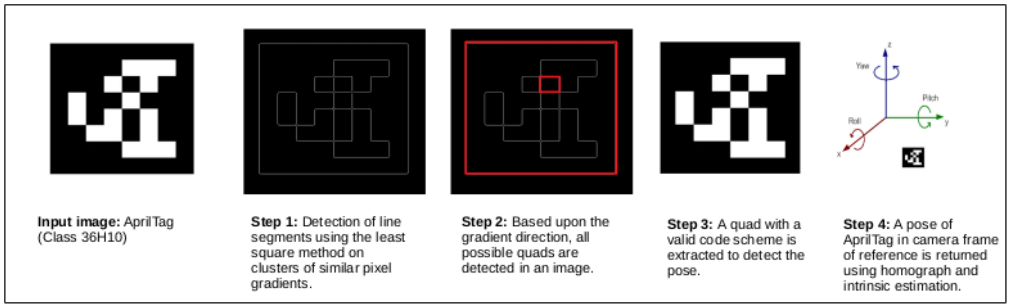


Figure 4-1: Figure shows four steps of AprilTag detection algorithm with an input image of AprilTag of class 36H10.

4.3.2 Trajectory Generation

In all robotics applications, odometry is key to every operation. Odometry includes all the positions and poses of a moving robot along its timestamp. As discussed in the literature survey, AprilTag is widely used for odometry generation of robots both in indoor and outdoor applications. It is illustrated in the previous section, AprilTag returns a single pose in 6-DOF relative to the camera frame of reference. Moreover, as the camera mounted on a robot changes its position along with the motion of a robot, it produces a series of posses from AprilTag at each time instance. Each pose shows the position of a robot along the moving robot trajectory at a particular time instance. Furthermore, to have a continuous trajectory we proposed a standard transformation technique between any two consecutive poses as shown in Figure 4-2. Suppose, we get a 6-DOF pose from an AprilTag in camera frame of reference, hence the camera attached frame, described in the Tag frame

of reference $\tau = (\bar{x}, \bar{y}, \bar{z})$ at instance i, is given by the homogeneous transformation:

$$T_i^\tau = \begin{bmatrix} R_i & d_i \\ 0 & 1 \end{bmatrix}_{4\times4}, \quad d_i = \begin{bmatrix} p_{i_x} & p_{i_y} & p_{i_z} \end{bmatrix}^T$$
$$R_i = \begin{bmatrix} c\phi c\theta & c\theta s\psi s\phi - c\psi s\theta & c\psi c\theta s\phi + s\psi s\theta \\ c\phi s\theta & c\psi c\theta + s\psi s\phi s\theta & -c\theta s\psi + c\psi s\phi s\theta \\ -s\phi & c\phi s\psi & c\psi c\phi \end{bmatrix}. \tag{4.1}$$

In Equation (4.1), the input angles are in camera frame of reference such as 'θ'is the rotation about z_i-axis and represents the roll motion of the camera, 'ϕ'is the rotation about y_i-axis and represents the yaw motion of the camera and 'ψ'is the rotation about x_i-axis and represents the pitch motion of the camera where $i = 0, 1, 2, ...n$. $p_{i_x}, p_{i_y}, p_{i_z}$ are the displacements in x-axis, y-axis and z-axis respectively in camera frame of reference. To have a trajectory in a single frame of reference, we need to find transformation T_i^{i+1} from point p_i to p_{i+1}.

$$T_i^{i+1} = T_\tau^{i+1} \times \left(T_\tau^i\right)^{-1}, \text{where } (T)^{-1} = \begin{bmatrix} R^T & -R^T d \\ 0 & 1 \end{bmatrix}. \tag{4.2}$$

In practice, AprilTag is detected at 10 HZ and the trajectory becomes almost continuous due to slow camera movement.

4.3.3 Error Measurements Setup

To analysis, the accuracy and precision of AprilTag, raw readings from AprilTag's native implementation have been compared with the readings of a high precision localization system called $Vicon\ MX - F40$, also known as "Motion Caption (MoCap)" [146]. MoCap consists of 16 high frame-rate cameras placed at the different known positions in an indoor environment. The system optically tracks a passive marker in 6-DOF with the sub-centimeter precision. It also has the capability of tracking multiple passive markers.

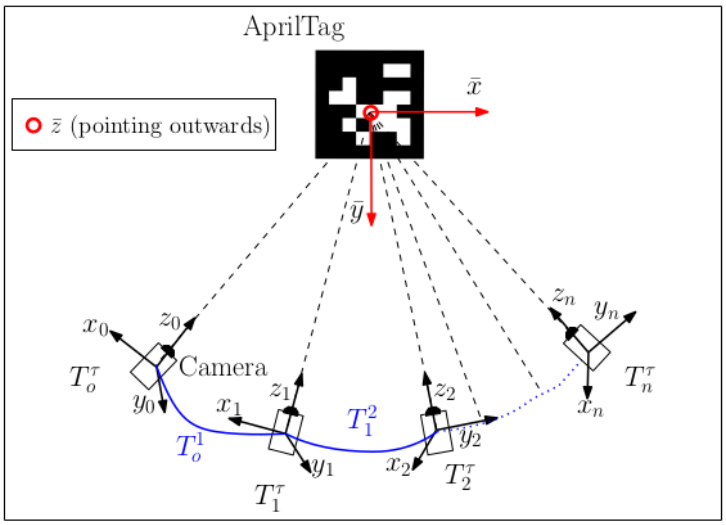


Figure 4-2: Trajectory using AprilTag detections. The trail of the transformation frame centers that constitute the trajectory is depicted in blue for various time instances. Here, p_{i_x},p_{i_y} and p_{i_z} of Equation 4.1 (although not shown in the figure) depict the position of AprilTag in x_i-axis, y_i-axis and z_i-axis in the respective camera frame of reference.

This system is considered as a benchmark for all indoor localization problems. Multiple monocular cameras which are calibrated at known positions, fuse the optical tracking data of a marker to track at good accuracy as shown in Figure 4-3.

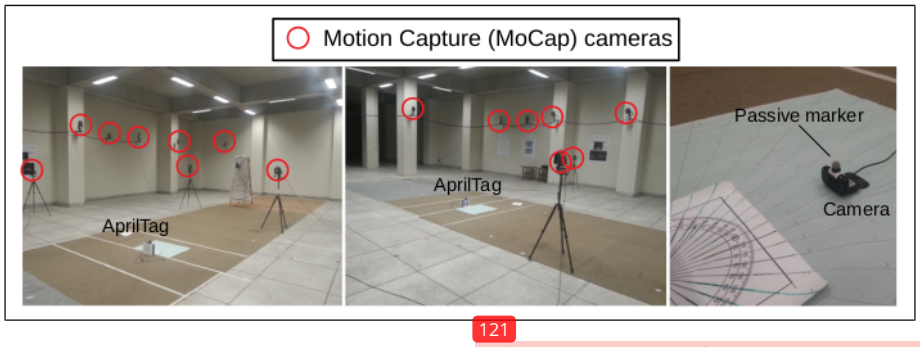


Figure 4-3: Motion Capture (MoCap) setup at Lahore University of Management Sciences Biomechanics lab in Lahore, Pakistan for AprilTag comparison.

AprilTag technology requires accuracy to the centimeter for the analysis, which is why we used a Motion Capture System (MoCap) for ground truth measurements. As described earlier, MoCap is an optical system that detects a passive marker; hence, a passive marker has been mounted above the camera to detect the position and orientation of the camera. For AprilTag localization, the measurement process has been made simple by making the origins of both AprilTag and MoCap frame of references aligned. Also, the camera is placed over a robotic platform that moves randomly around and the mounted camera has a constant motion of maximum 30° around its yaw axis to include maximum noise possible at a given nominal reference point. Table 4.2 shows the overall performance of the MoCap for estimating the robot's position on the ground. Column 1 'x_r'and column 2 'y_r'in Table 4.2 show the nominal positions of reference points from which the measurements have been taken. Column 3 'N'represents the total number of readings taken at a specific reference point, column 4 '$\mu_{\bar{x}}$'and column 5 '$\mu_{\bar{y}}$'shows the accuracy of the MoCap as the reported mean value in both x-axis and y-axis respectively. Lastly, column 6 '$\sigma^2_{\bar{x}}$'and column 7 '$\sigma^2_{\bar{y}}$'shows the precision of the MoCap in the form of variances reported in x-axis and y-axis respectively.

Moreover, Figure 4-4 shows the error plot for both $\bar{x}$-axis and $\bar{y}$-axis of MoCap. In Figure 4-4, the horizontal axis shows the $\bar{x}$ and $\bar{y}$ component of MoCap measurements in the left and right plot respectively and the vertical axis shows the accuracy after subtracting mean value from the measurements. Besides, the measurements are taken at different distances from the AprilTag in $\bar{y}$-axis, this information is coded in three colors such as red represents a distance of 30 cm, blue represents a distance of 50 cm and the green represents a distance of 70 cm. Plots in Figure 4-4 show that $\bar{x}$ component of MoCap measurements are more effected by viewing distance then $\bar{y}$ component. Further, as we move towards either the left or right side from the center of the tag, accuracy decreases.

Table 4.2: Measurement from Motion Capture (MoCap).

Nominal Reference Points		**Motion Capture (MoCap) Readings**				
x_r(cm)	y_r(cm)	N	$\mu_{\bar{x}}$(cm)	$\mu_{\bar{y}}$(cm)	$\sigma^2_{\bar{x}}(cm^2)$	$\sigma^2_{\bar{y}}(cm^2)$
0	30	217	0.7062	30.3437	0.003110	0.003120
6	30	195	6.9367	29.5193	0.004890	0.000590
-6	30	193	-3.9230	30.1210	0.045800	0.003080
0	50	204	2.9902	50.0462	0.09500	0.014700
15	50	202	17.7118	49.6042	0.009490	0.002650
-15	50	205	-12.0469	50.8171	0.000168	0.000242
0	70	217	3.2683	70.0810	0.002380	0.000267
20	70	199	23.3681	69.4768	0.000210	0.000197
-20	70	212	-16.8097	70.9505	0.037100	0.001090

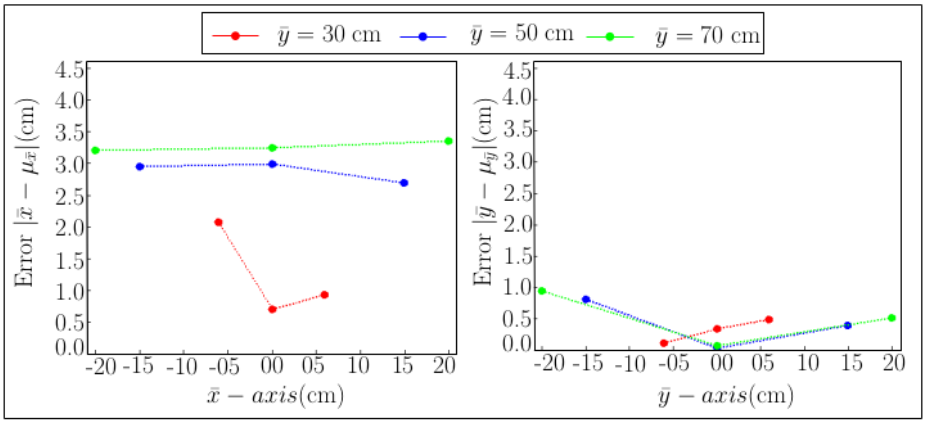


Figure 4-4: Accuracy plot for Motion Capture (MoCap).

For the theoretical point of reference in experiments, we have used an error measurement markings to get a rough estimation regarding the position of the camera from the AprilTag, as shown in Figure 4-5. The design of the measurement experiments is illustrated

in Figure 4-6, which shows how the readings have been taken to observe the actual inaccuracy caused by the various parameters such as distance and camera viewing angle. For the rest of the chapter, analysis measurements are in a plane only, namely x-axis x, z-axis z and yaw angle ϕ in the camera frame of reference. The output of the measurements is represented in a 3-DOF AprilTag frame of reference τ with variables $\bar{x}, \bar{y}$ and $\bar{\theta}$. In Figure 4-6, crosses represent the locations of the robot from which the readings are noted. This measurement technique is common for both the MoCap and the AprilTag data recordings for evaluation. These positions are obtained from the nominal reference points marked on an error measurement setup. The error measurement setup consists of a large paper sheet marked with angles and distances from the origin of AprilTag. At every nominal measurement point on error measurement setup, viewing yaw angle ϕ of the camera can be different. When the camera is pointing directly towards the center of the AprilTag, the yaw angle is $90°$. If the camera is pointed towards the right side of the center, the yaw angle is $90 + \phi$ and for left, the yaw angle is $90 - \phi$. We have taken measurements at 9 nominal points: three exactly in front of the tag and three on either side. The reason for selecting specific measurement points is to include maximum uncertainty in measurements for the viewing angles and the distances. Moreover, due to the limited field of view of the camera and the workspace environment, we keep the nominal points to 9 points. These points are uniformly covering each side and the face of the tag. Raw measurements at different angles and distances from AprilTag have been plotted and compared against the ground truth measured by MoCap.

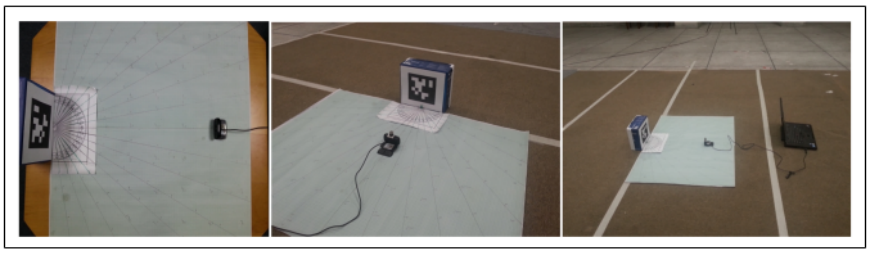


Figure 4-5: Photographs from different views of the AprilTag error measurement setup. *Left* : Shows the top-down view of the error measurement setup. *Middle* : Shows the placement of the camera in front of the AprilTag over error measurement setup. *Right* : Shows the side view of the measurement recording process.

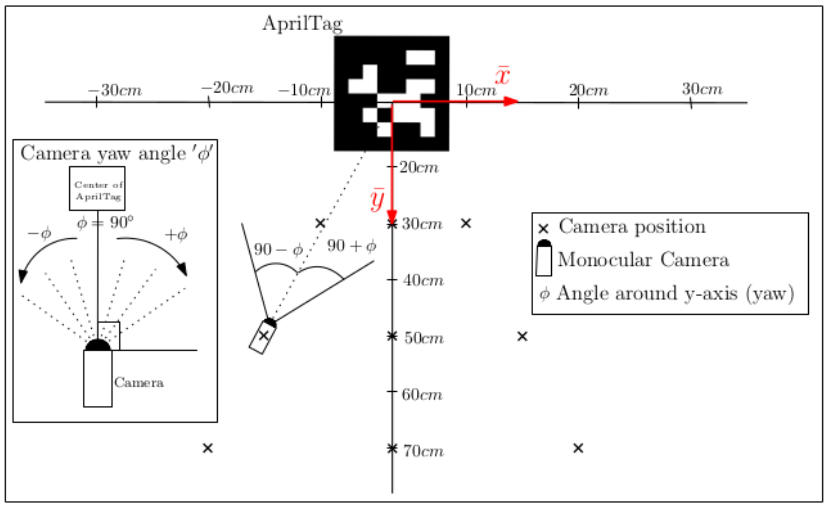


Figure 4-6: Error measurement setup showing measurement positions and yaw angles of the camera to AprilTag placed at the origin.

It is observed that the ideal scenario for AprilTag accuracy is when the camera is pointing towards the center of the tag or camera z-axis lies toward the center of AprilTag. Figure 4-7 shows the plots from raw measurements when the camera z-axis lies toward the center of AprilTag. The center of the tag is taken as $(\bar{x}, \bar{y}) = (0, 0)$ in AprilTag frame. These measurements are of the best accuracy that one can achieve from AprilTag and are used

later as a reference. Similarly, Figure 4-7 also shows blue readings for which the camera z-axis does not lie towards the center of AprilTag. It can be seen that blue readings incur large inaccuracy when the camera is wider.

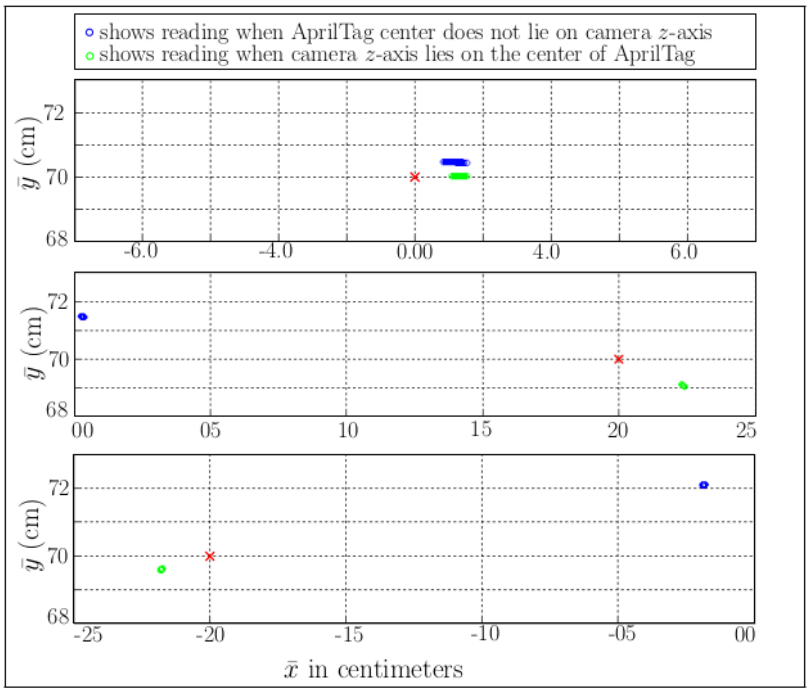


Figure 4-7: Multiple raw AprilTag readings plotted for ideal (green) and worst (blue) scenarios. Mean ground-truth (MoCap) readings are plotted as red crosses.

Table 4.3 summarizes the statistics of the measurements when the camera is pointing towards the center of the tag. For all readings, the yaw angle $\phi = 90^\circ$ means it is directed towards the center of AprilTag. First two columns 'x_r'and 'y_r'show the x-axis and y-axis of nominal reference points where we wanted to place the camera. Third column '$\bar{x}$'and forth columns '$\bar{y}$'show ground truth values on the desired reference points using MoCap. Fifth column 'N'represents total number of readings taken at that reference point (x_r, y_r). Sixth column '$\mu_{\bar{x}}$'and seventh column '$\mu_{\bar{y}}$'give the mean in $\bar{x}$-axis readings and $\bar{y}$-axis. The

eighth column '$\sigma_{\bar{x}}^2$'and ninth columns '$\sigma_{\bar{y}}^2$'show variances in $\bar{x}$-axis and $\bar{y}$-axis respectively. We get an average mean error of around 1.0 cm for $\bar{x}$ and around 0.40 cm error for $\bar{y}$ over a variable distance of $\pm$6 cm, $\pm$15 cm and $\pm$70 cm in $\bar{x}$-axis and 30 cm, 50 cm and 70 cm in $\bar{y}$-axis.

Table 4.3: Measurement stats with camera z-axis pointed towards the center of AprilTag.

Nominal Reference Points		**Ground Truth (MoCap)**			**AprilTag Readings**			
x_r(cm)	y_r(cm)	$\bar{x}$(cm)	$\bar{y}$(cm)	N	$\mu_{\bar{x}}$(cm)	$\mu_{\bar{y}}$(cm)	$\sigma_{\bar{x}}^2(cm^2)$	$\sigma_{\bar{y}}^2(cm^2)$
0	30	0.0166	30.020	110	-0.2420	30.1510	0.002000	0.000170
6	30	7.0161	30.010	115	7.0161	29.8573	0.000040	0.000040
-6	30	-5.9798	29.92	80	-5.9798	30.4293	0.000035	0.002080
0	50	0.102	49.960	107	0.7571	50.0819	0.000930	0.000002
15	50	14.952	50.69	113	16.9141	49.4206	0.000090	0.000030
-15	50	-14.98	49.90	103	-16.9185	49.4014	0.000090	0.000034
0	70	0.003	70.05	134	1.3080	70.0264	0.007390	0.000014
20	70	20.06	70.06	144	22.3574	69.0718	0.000310	0.000092
-20	70	-20.01	70.02	151	-21.7560	69.5979	0.000240	0.000063

Figure 4-8 shows the mean error plot for the Table 4.3 using nominal reference points. We can see that the error is minimum for both $\bar{x}$ and $\bar{y}$ exactly in front of AprilTag. As we move along the left or right side, the error increases. Another notable finding is that the error increases as we increase the camera distance from the AprilTag along $\bar{y}$-axis.

Instead of pointing the camera towards the center of AprilTag, if we fix the camera such that its z-axis never points towards the center of the AprilTag, one gets the worst readings in terms of accuracy no matter which side of the tag the camera is located. To empirically analyze this concept, an experiment is performed in which measurements are taken at fixed measurement points with the varying camera yaw angle ϕ ranging from 70° to 110°. Figure 4-9 shows the data plot of AprilTag with changing camera yaw angle 'ϕ'. Here, the spread of data around a measurement reference point is in a circular path distributed almost evenly on both sides. Other than the plot representation, Table 4.4 shows the statistics of

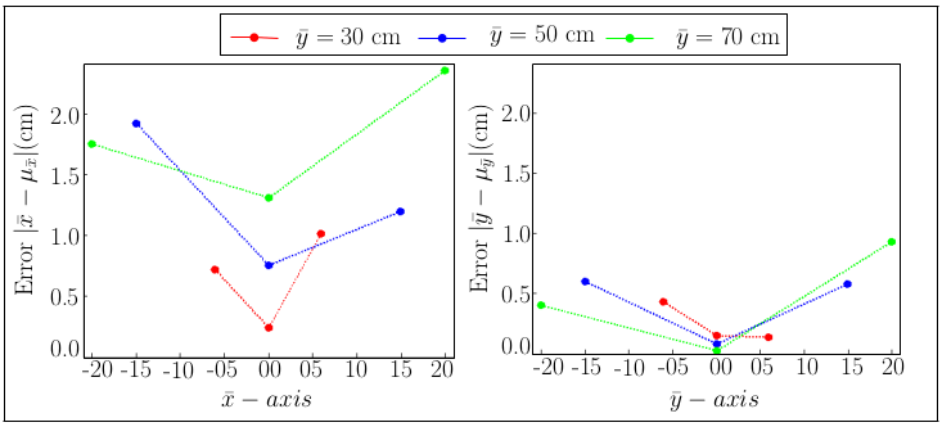


Figure 4-8: Error plot with camera's z-axis pointed towards the center of AprilTag. $Left$: Error plot for $\bar{x}$-axis. $Right$: Error plot for $\bar{y}$-axis.

the reported data in terms of mean and variance in both measurement axis. As shown in Table 4.4, the variance $(\sigma^2_{\bar{x}}, \sigma^2_{\bar{y}})$ and mean $(\mu_{\bar{x}}, \mu_{\bar{y}})$ values of both $\bar{x}$ and $\bar{y}$ have increased manifold especially when $\bar{x} = \pm 20$ cm and $\bar{y} = 70$ cm.

Table 4.4: Measurement stats with AprilTag center does not lie on the z-axis of the camera (changing camera yaw angle 'ϕ').

Nominal Reference Points		**Ground Truth (MoCap)**		**AprilTag Readings**				
x_r(cm)	y_r(cm)	$\bar{x}$(cm)	$\bar{y}$(cm)	N	$\mu_{\bar{x}}$(cm)	$\mu_{\bar{y}}$(cm)	$\sigma^2_{\bar{x}}(cm^2)$	$\sigma^2_{\bar{y}}(cm^2)$
0	30	0.7062	30.3437	152	-1.1162	30.1501	9.47	0.16
6	30	6.9367	29.5193	182	5.4598	29.4149	14.0	0.56
-6	30	-3.9230	30.1210	162	-6.4092	29.6787	14.0	1.07
0	50	2.9902	50.0462	133	-1.3508	49.1515	76.0	1.51
15	50	17.7118	49.6042	144	13.8889	48.8854	84.0	6.87
-15	50	-12.0469	50.8171	186	-15.5709	48.3046	57.0	7.43
0	70	3.2683	70.0810	163	-0.0285	68.3890	194.0	2.61
20	70	23.3681	69.4768	149	23.4214	67.4216	154.0	14.0
-20	70	-16.8097	70.9505	140	-24.4433	67.4337	112.0	14.0

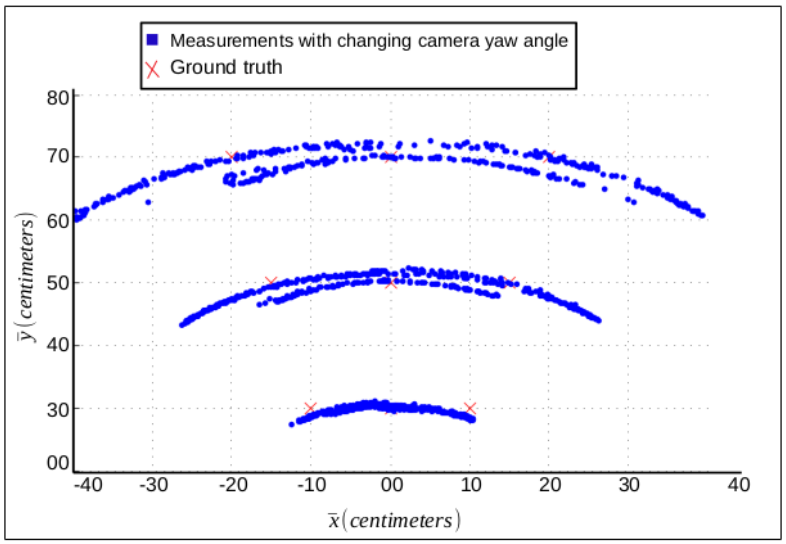


Figure 4-9: Plot for measurements with changing camera yaw angle 'ϕ'for $70° \leq \phi \leq 110°$.

Additionally, Table 4.4 shows the mean value of all the measurements taken at a particular reference point with changing camera yaw angle 'ϕ'. As Figure 4-9 shows, that the data spread is distributed almost evenly around the reference point along a circular path. In other words, it shows that for a particular reference point, as the camera yaw angle 'ϕ' changes, the reporting position also changes along the circular path of the distribution. Considering, the spread of data distribution is almost same on either side of the reference point, hence we get the mean values $(\mu_{\bar{x}}, \mu_{\bar{y}})$ relatively near to the measurement reference point itself. To further analyze the worst possible case of camera yaw angle 'ϕ', a similar experiment has been conducted with camera yaw angle 'ϕ' fixed to 110°. Table 4.5 shows the statistical analysis of the experiment with the camera yaw axis fixed at 'ϕ'=110°. Table 4.5 shows that the inaccuracy has increased in the mean values $(\mu_{\bar{x}}, \mu_{\bar{y}})$ especially in $\bar{x}$-axis. This is because the resulting measurements at 'ϕ=110°' lie at the farthest sides of the circular spread shown in Figure 4-9. Moreover, Figure 4-10 shows the error plot for Table 4.5 against the ground truth(MoCap). It shows that the error is minimum at $\bar{x} = 0$ but

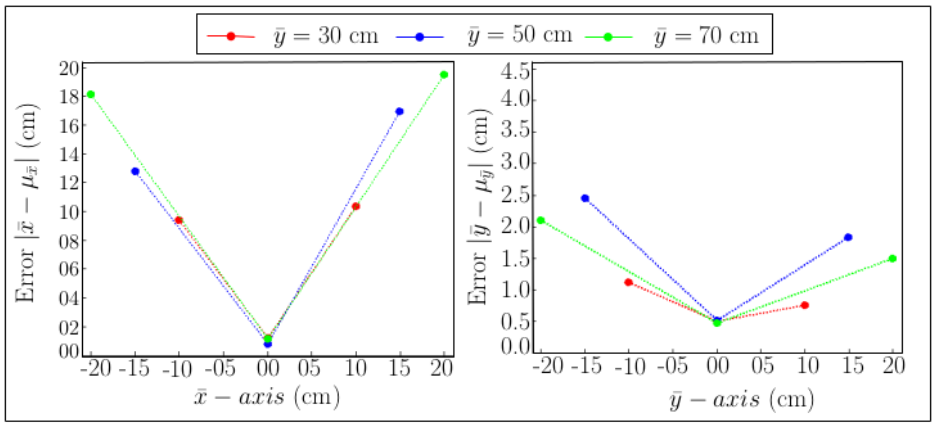


Figure 4-10: Error plot with camera yaw axis 'ϕ' fixed at 110°. *Left* : Error plot for $\bar{x}$-axis. *Right* : Error plot for $\bar{y}$-axis.

increases significantly as we move along the sides. For $\bar{y} = 70$ cm, the error is around 16 cm for $\bar{x} = \pm 20$ cm whereas at $\bar{x} = 0$ cm, the error is only around 2 cm.

Table 4.5: (Worst scenario) Measurement stats with fixed camera yaw angle 'ϕ' at 110°.

Nominal Reference Points			**Ground Truth (MoCap)**		**AprilTag Readings**				
x_r(cm)	y_r(cm)	ϕ(deg)	$\bar{x}$(cm)	$\bar{y}$(cm)	N	$\mu_{\bar{x}}$(cm)	$\mu_{\bar{y}}$(cm)	$\sigma^2_{\bar{x}}(cm^2)$	$\sigma^2_{\bar{y}}(cm^2)$
0	30	110	0.020	30.001	113	-1.2057	30.5027	0.000230	0.000003
10	30	110	10.07	30.01	154	-0.3694	30.7558	0.021100	0.003230
-10	30	110	-9.67	30.06	178	-0.5791	31.1269	0.000101	0.000021
0	50	110	0.100	50.071	147	-0.8202	50.5234	0.001350	0.000004
15	50	110	14.960	50.100	120	-1.9371	51.8464	0.000210	0.000046
-15	50	110	-14.91	49.97	117	-2.2290	52.4599	0.000165	0.000034
0	70	110	0.03	70.01	184	1.1841	70.4607	0.013600	0.000015
20	70	110	20.10	69.98	102	0.3113	71.4872	0.000750	0.000039
-20	70	110	-20.08	70.05	128	-1.8619	72.1167	0.000425	0.000018

Based on raw AprilTag measurements, the following shortcomings are identified in the current AprilTag implementation.

Distance from tag: It is observed that the accuracy decreases over distance as we move the camera away from the tag. As shown in Table 4.3 and Table 4.5, we can see the mean and variance for both $\bar{x}$ and $\bar{y}$ are increasing with increase in distance from the tag in z-axis.

Viewing Angle: From multiple experiments, it is understood that the accuracy also decreases as the camera position changes from front to sideways. In the ideal scenario (Table 4.3), though the camera is pointing toward the center of the AprilTag at all the points, the error is less for $\bar{x} = 0$ as compared to $\bar{x} \neq 0$. This error increases as we increase $\bar{x}$. Table 4.5 shows a similar pattern.

Yaw Angle of the Viewing Camera: Previous extensive experiments show that the main source of inaccuracy is the frame inconsistency caused due to motion and significantly reduces the performance. The reason is that AprilTag fiducial system is coded in such a way that the output frame of reference is dependent upon the yaw angle ϕ orientation of the camera attached to the moving body. As the orientation of the moving body changes, the output frame also changes, making it hard to have a consistent frame of reference. At any given point, in x and z camera coordinates, change in yaw angle ϕ causes the generation of a new origin hence making a new frame of reference for every yaw angle. The new origin is the intersecting point of the AprilTag face plane with a straight line 'z-axis' from the center of the camera. So the current distance is reported under newly formed origin. Though the resulting output is relatively accurate in its respective frame of reference, the overall accuracy of all the yaw angles ϕ combined against a constant frame of reference is inaccurate. Figure 4-9 shows the plot of AprilTag reporting at fixed measurement points with varying yaw angle ϕ ranging from 70° to 110°. Variance and mean readings of both $\bar{x}$ and $\bar{y}$ have also increased many folds as shown in Table 4.4 especially when $\bar{x} = \pm 20$ cm and $\bar{y} = 70$ cm.

4.4 Improvement Techniques

Based on the measurement analysis of the AprilTag system, the following improvement techniques have been proposed.

4.4.1 Passive Correction for Frame Consistency

As illustrated in Section 3.3, the key source of inaccuracy in AprilTag readings is the misalignment of the camera z-axis with the center of the tag. When the camera follows a certain trajectory, its orientation may change over time, which causes inconsistency between two consecutive frames. To solve this problem, we propose a passive-orientation correction. Also referred to as a "Soft Yaw Axis Correction (SYAC)" technique. In this technique, the geometry of the whole setup is modified in a way that the axis (z-axis) passing through the center of the camera always points towards the tag's origin that is, $(\bar{x}, \bar{y}) = (0, 0)$. Figure 4-11 shows the drawing for trigonometric correction. The solid triangle shows the original geometry without any correction in the camera frame of reference. The hypotenuse of a solid triangle z, which emerges from the camera center, should touch the center of tag. That ideal line is called $\bar{z}$, depicted as a dotted line in Figure 4-11. Moreover, ϕ is known, the angle $\acute{\omega}$ that aligns the dotted triangle hypotenuse with the center is calculated. By using simple trigonometry, $\bar{z}$ and $\acute{\omega}$ is calculated as:

$$\acute{\omega} = \phi - \tan^{-1}\left(\frac{z\sin\phi}{x + z\cos\phi}\right), \tag{4.3}$$

$$\bar{z} = \sqrt{((z\sin\phi))^2 + ((x + z\cos\phi))^2}. \tag{4.4}$$

Once $\bar{z}$ and $\acute{\omega}$ are known, $\bar{x}$ Equation (4.5), $\bar{y}$ Equation (4.6) and $\bar{\theta}$ Equation (4.7) are derived which eventually improves the accuracy.

$$\bar{x} = x + \acute{x} = x + z\cos\phi, \tag{4.5}$$

$$\bar{y} = z\sin\phi, \tag{4.6}$$

$$\bar{\theta} = \arctan\left(\frac{\acute{y}}{x+\acute{x}}\right) = \arctan\left(\frac{z\sin\phi}{x+z\cos\phi}\right). \tag{4.7}$$

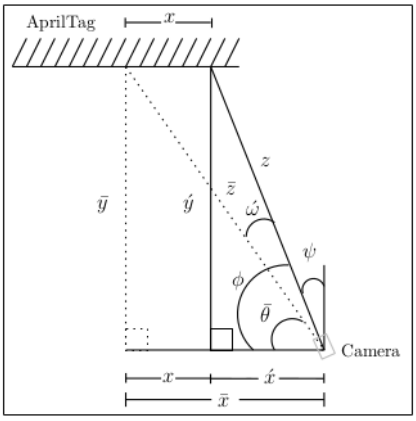


Figure 4-11: Geometrically aligning subsequent frames.

Figure 4-12 shows the data scatter plot after applying this passive correction technique. It can be seen that the spread of the transformed data is decreased and Table 4.6 shows decreased variance both in $\bar{x}$ and $\bar{y}$ axis. By zooming point $(x, z) = (0.20, 0.70)$, it can be observed that the original readings are displaced only after applying the correction, making them more closely to the reference point. At camera yaw angle of 110°, it is almost

40 cm off the true position in $\bar{x}$-axis and 2 cm in $\bar{y}$-axis. After applying the correction, the error in $\bar{x}$-axis is reduced to 5 cm and in $\bar{y}$-axis to 1 cm. Similarly, at yaw angle, 70° in $\bar{x}$-axis, the error is reduced from 26 cm to 4 cm.

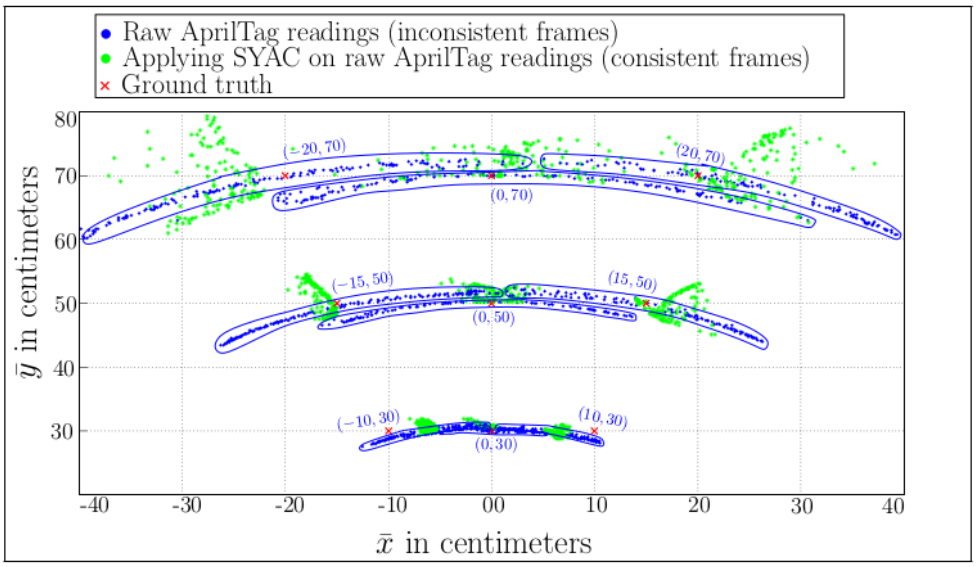


Figure 4-12: A comparison plot for AprilTag raw readings and improved SYAC (Soft Yaw Axis Correction) measurements with changing camera yaw angle 'ϕ'for $70° \leq \phi \leq 110°$. Blue circles show the clustering of the plotted data around a ground truth point.

Table 4.6: Table showing measurement stats after applying Soft Yaw Axis Correction (SYAC) on raw AprilTag data.

Nominal Reference Points		**Ground Truth (MoCap)**			**AprilTag Readings**			
x_r(cm)	y_r(cm)	$\bar{x}$(cm)	$\bar{y}$(cm)	N	$\mu_{\bar{x}}$(cm)	$\mu_{\bar{y}}$(cm)	$\sigma^2_{\bar{x}}(cm^2)$	$\sigma^2_{\bar{y}}(cm^2)$
0	30	0.7062	30.3437	113	-0.5813	30.6166	0.31	0.11
6	30	6.9367	29.5193	154	6.4232	29.7444	0.27	0.31
-6	30	-3.9230	30.1210	178	-6.2826	30.5416	0.20	0.44
0	50	2.9902	50.0462	147	0.1930	51.3185	3.06	0.91
15	50	17.7118	49.6042	120	17.3150	49.5959	3.32	3.37
-15	50	-12.0469	50.8171	117	-16.2733	50.1972	1.12	5.04
0	70	3.2683	70.0810	184	1.8551	71.8400	30.0	3.37
20	70	23.3681	69.4768	102	24.8826	70.5867	12.0	13.0
-20	70	-16.8097	70.9505	128	-26.4522	69.1570	10.0	19.0

To further extend the comparison, Figure 4-13 shows the improvement of AprilTag readings concerning the camera yaw angle 'ϕ.' The rotation of the camera around its yaw axis is limited to five sampling angles that is, 70°, 80°, 90°, 100° and 110°. The rate of rotation for yaw angle 'ϕ' is 10 degrees/sec. Hence, it takes the camera 5 seconds to sweep in one direction. Moreover, AprilTag is being detected at 11 Hz; hence, we have approximately 11 readings at an individual yaw angle ϕ during a single sweep. As Figure 4-13 shows that at each measurement angle 'ϕ,' our proposed Soft Yaw Axis Correction approach (SYAC) has significantly improved the accuracy of raw AprilTag. Red cross $(\bar{x}, \bar{y}) = (20, 70)$ shows the ground-truth value for the whole experiment. As we can see from the plot that the accuracy of AprilTag decreases as we increase the yaw axis angle 'ϕ' of the camera. The accuracy is worse when 'ϕ' is either 110° or 70°. As the camera yaw angle 'ϕ' approaches 90°, which implies the camera's z-axis points towards the center of the tag, accuracy increases. As a result, Figure 4-13 shows data at '$\phi = 90^\circ$' most accurate.

4.4.2 Active Correction with a Yaw Axis Gimbal

Another way to correct for misalignment of camera z-axis with the center of the tag is to track and correct it in real-time using a yaw axis gimbal actively. The custom-built hardware setup is proposed to achieve this, as shown in Figure 4-14. The tracking algorithm 1 consists of a Proportional-Integral-Derivative (PID) based action tracking controller working at 10Hz. The input to the algorithm 1 is a raw yaw angle of the camera 'ϕ'in the camera frame of reference reported by native AprilTag implementation. As discussed in Section 3.3, the yaw angle of the camera 'ϕ'depends upon the alignment of the camera z-axis with the center of AprilTag. If the camera z-axis lies on the center of the AprilTag, 'ϕ'is equal to zero. As the face of the camera moves away from the center of the AprilTag, 'ϕ'value changes and introduces inaccuracy. Moreover, the goal of the Active Correction with a Yaw Axis Gimbal (ACYG) is to keep the z-axis of the camera aligned towards the center of the tag by keeping angle 'ϕ'equals to zero. Significant improvement in the tag's precision has been

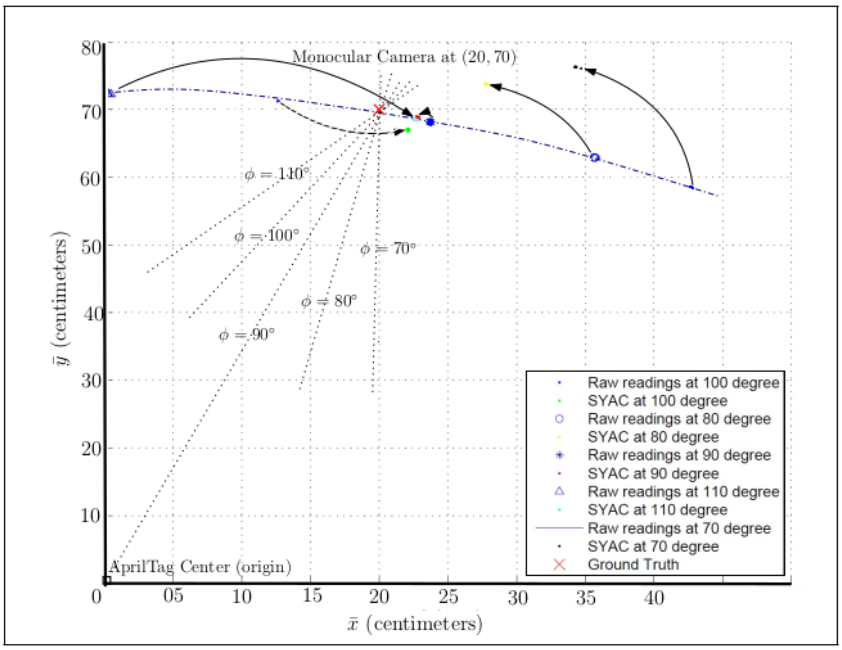


Figure 4-13: An angle-wise comparison plot for AprilTag raw readings and improved SYAC measurements with changing camera yaw angle 'ϕ'for $70° \leq \phi \leq 110°$. Plot shows that our proposed technique has significantly improved AprilTag raw measurements.

observed with almost similar accuracy by using this technique. One of the problems with the Passive Correction for Frame Consistency technique is that it does not align the camera center accurately towards the center of the tag if the yaw angle ϕ is too big. Therefore, active compensation in combination with a passive correction ensures that the camera yaw angle does not become too big. Figure 4-15 shows the effect of one-axis tracking gimbal in the combination of passive frame consistent Correction and without passive frame consistent Correction. Data scatter plots show the accuracy has increased significantly, especially in combination with SYAC. Table 4.7 and Table 4.8 summarize the variances and mean values while using yaw axis gimbal with raw AprilTag and with SYAC correction, respectively.

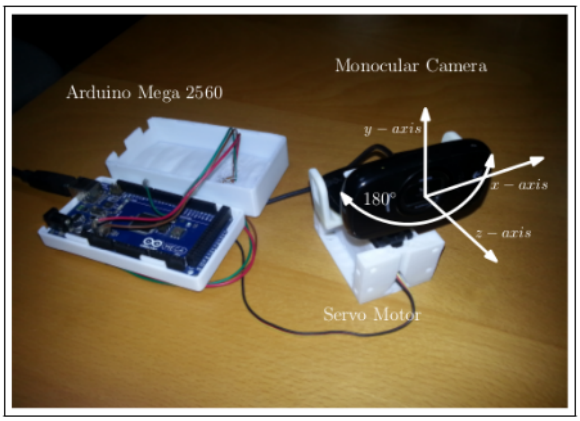


Figure 4-14: Yaw-axis gimbal hardware setup developed by the authors. A monocular camera has been mounted on a Dynamixal stepper motor, which is controlled by an Arduino Mega 2560 controller. The controller is used as a slave ROS process in localization application. Housing is in a 3D printed retrofit.

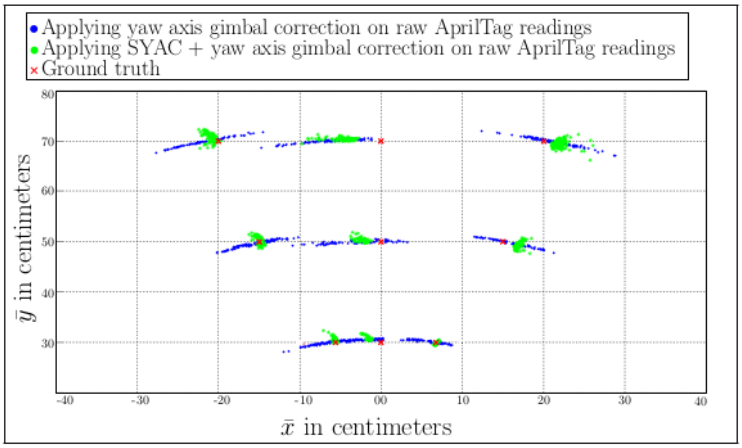


Figure 4-15: Data scatter plot for geometrically consistent (SYAC) and non consistent frames(raw AprilTag) with custom-built yaw axis gimbal.

Figure 4-15 shows that though active correction has improved the overall accuracy. If this technique is applied with a combination of passive correction (SYAC), the resultant

readings are more accurate. The reason behind this is that both the correction methods have their limitations. In Soft correction (SYAC), sometimes the camera z-axis fails to align with the tag center if the measured yaw angle 'ϕ' of the viewing camera is too large. Moreover, in Active correction, the tag is being detected at 11 Hz and the active correction is being done at 8 Hz to due system limitations. Hence, this results in the incursion of inaccuracy in the active correction system. Henceforward, by using both correction techniques in combination with each other improves the overall accuracy manifolds.

Table 4.7: Use of yaw axis gimbal on raw AprilTag system.

Nominal Reference Points		**Ground Truth (MoCap)**		**AprilTag Readings**				
x_r(cm)	y_r(cm)	$\bar{x}$(cm)	$\bar{y}$(cm)	N	$\mu_{\bar{x}}$(cm)	$\mu_{\bar{y}}$(cm)	$\sigma^2_{\bar{x}}(cm^2)$	$\sigma^2_{\bar{y}}(cm^2)$
0	30	0.642	31.006	156	-2.7569	30.3124	3.39	0.07
6	30	6.71	29.820	144	5.8448	30.0716	1.65	0.08
-6	30	-5.89	29.851	126	-6.6391	29.8201	2.99	0.19
0	50	2.017	50.02	124	-2.9412	49.8801	6.92	0.08
15	50	16.90	51.13	115	15.9652	49.6746	4.71	0.49
-15	50	-14.42	49.63	144	-15.6127	49.4842	4.16	0.49
0	70	-2.10	68.90	149	-5.5017	70.0076	7.35	0.11
20	70	22.70	71.16	152	21.0146	69.8425	5.84	0.58
-20	70	-21.10	71.23	138	-21.7858	69.7444	5.92	0.56

Table 4.8: Use of yaw axis gimbal with consistent frames (SYAC).

Nominal Reference Points		**Ground Truth (MoCap)**		**AprilTag Readings**				
x_r(cm)	y_r(cm)	$\bar{x}$(cm)	$\bar{y}$(cm)	N	$\mu_{\bar{x}}$(cm)	$\mu_{\bar{y}}$(cm)	$\sigma^2_{\bar{x}}(cm^2)$	$\sigma^2_{\bar{y}}(cm^2)$
0	30	0.642	31.006	156	-1.4932	30.6753	0.06	0.08
6	30	6.71	29.820	144	6.7595	29.7747	0.01	0.03
-6	30	-5.89	29.851	126	-5.5643	30.4668	0.05	0.17
0	50	2.017	50.02	124	-2.1262	50.1620	0.19	0.13
15	50	16.90	51.13	115	16.8541	49.2115	0.19	0.43
-15	50	-14.42	49.63	144	-14.8491	50.1751	0.10	0.42
0	70	-2.10	68.90	149	-4.3831	70.2868	1.63	0.05
20	70	22.70	71.16	152	21.9349	69.3702	0.71	0.43
-20	70	-21.10	71.23	138	-20.8403	70.5489	0.29	0.60

Algorithm 1 Active Camera Tracking of AprilTag Center

Input: camera yaw angle 'ϕ'

Output: servo angle 'Γ'

$K_P \leftarrow$ Propotional gain depending upon z-axis value.
K_I Integral gain depending upon z-axis value.
$K_D \leftarrow$ Propotional gain depending upon z-axis value.
$\epsilon \leftarrow$ Initialize error with zero
$\mathbb{T} \leftarrow 0.05$ ▷ Servo stopping threshold.
$\alpha \leftarrow 0.008$ ▷ Smoothing factor.
$\epsilon \leftarrow \phi$
if $\epsilon > \mathbb{T}$ **then**
Integral ← Integral + ϵ
else Integral ← 0.00
end if
P ← $\epsilon \times K_P$
I ← Integral $\times K_I$
D ← (LastYawAngle − CurrentYawAngle) $\times K_D$
Drive ← P + I + D
Drive ← Drive × α
if Drive > 90 **then** ▷ To keep camera facing AprilTag
    Drive ← 90
else Drive < -90
    Drive ← -90
end if
Γ ← CurrentServoAngle + Drive
LastYawAngle ← CurrentYawAngle
CurrentServoAngle ← Γ
return Γ

4.4.3 Comparative Results

Following the extensive experimentation and dataset collection, comparative results have been deducted to show the improvement more comprehensively. Table 4.9 shows the error comparison of different approaches against AprilTag. Here error is represented as the difference between the reported mean value and the ground truth both in $\bar{x}$ and $\bar{y}$ axis. Columns 1 and 2 show the ground truth (MoCap) values of $\bar{x}$ and $\bar{y}$, respectively, against which the standard error is compared. Columns 3 and 4, which are labeled as "Raw AprilTag readings (camera pointing towards tag's center)," show the error for the AprilTag system when the camera is always pointed towards the center of tag. Earlier experiments have shown that this is the maximum possible accuracy one can achieve using AprilTag. Moreover, columns 5 and 6 "Raw AprilTag readings (camera pointing away from tag's center)" shows the raw data from AprilTag when the camera z-axis is not aligned with the center of the tag resulting in inconsistent frames induced by camera motion. Columns 7 and 8, which are labeled as "Applying Soft Yaw Angle Correction (SYAC) on raw AprilTag readings," shows mean error after applying the proposed approach of Soft Yaw Angle Correction (SYAC) to make the inconsistent frames consistent. Similarly, columns 9 and 10 labeled as "Applying Active Correction with Yaw Axis Gimbal on raw AprilTag readings" show error after using custom build yaw axis gimbal on raw AprilTag system. Lastly, columns 11 and 12, which are labeled as "Applying (SYAC + Active Yaw Axis Gimbal correction) on raw AprilTag readings," show an error when both proposed approaches of soft and active yaw axis correction are applied in combination.

Additionally along with the accuracy, precision of the AprilTag system has also been increased manifolds by our proposed approaches as shown in Figure 4-16 and Figure 4-17. Figure 4-16 shows the resulting precision of different approaches in *cm* for a nominal reference point of $(\bar{x}, \bar{y}) = (0, 70)$. We can see that the spread of $\bar{x}$ for raw AprilTag data with camera's z-axis not pointing towards tag's center is around 13.9 cm while after applying Soft Yaw Angle Correction (SYAC) plus Active Yaw Axis Gimbal Correction,

it is decreased to 1.27 cm. Moreover for $\bar{y}$, the spread is decreased from 1.61 cm to 0.22 cm. In addition, Figure 4-17 shows the similar analysis for nominal reference point of $(\bar{x}, \bar{y}) = (20, 70)$. Here after applying Soft Yaw Axis Correction (SYAC) and Active Yaw Axis Gimbal Correction on AprilTag, the precision has improved manifolds and the data spread for $\bar{x}$ and $\bar{y}$ is decrease from 12.04 cm to 0.84 cm and 3.74 cm to 0.65 cm respectively. Nonetheless, Motion Capture (MoCap) spread has also been illustrated for both the nominal reference points for ground-truth analysis.

As mentioned earlier, the objective is to reduce the measurement error close to the ground truth. Figure 4-18 shows the statistical analysis of accuracy by plotting Mean Root Square Error (RMSE) achieved by the proposed approaches against the raw AprilTag. It shows that our proposed approaches have significantly reduced the RMSE as compared to bare AprilTag results. Moreover, this error is further reduced when both the proposed approaches are combined. The resulting error is significantly close to ground-truth and the ideal scenario when the camera is pointing towards the center of AprilTag hence achieving our objective.

Table 4.9: Comparison of AprilTag against various proposed approaches.

Ground-Truth (cm)		**Error in mean**(μ) **using different approaches for AprilTag.** (cm)									
Motion Capture System (MoCap))		Raw AprilTag readings (camera pointing towards tag's center)		Raw AprilTag readings (camera pointing away from tag's center)		Applying Soft Yaw Angle Correction (SYAC) on raw AprilTag readings		Applying Active Correction with Yaw Axis Gimbal on raw AprilTag readings		Applying (SYAC + Active Yaw Axis Gimbal correction) on raw AprilTag readings	
$\bar{x}$	$\bar{y}$	$\lvert\bar{x}-\mu_{\bar{x}}\rvert$	$\lvert\bar{y}-\mu_{\bar{y}}\rvert$	$\lvert\bar{x}-\mu_{\bar{x}}\rvert$	$\lvert\bar{y}-\mu_{\bar{y}}\rvert$	$\lvert\bar{x}-\mu_{\bar{x}}\rvert$	$\lvert\bar{y}-\mu_{\bar{y}}\rvert$	$\lvert\bar{x}-\mu_{\bar{x}}\rvert$	$\lvert\bar{y}-\mu_{\bar{y}}\rvert$	$\lvert\bar{x}-\mu_{\bar{x}}\rvert$	$\lvert\bar{y}-\mu_{\bar{y}}\rvert$
0.642	31.00	0.884	0.855	1.758	0.855	1.223	0.389	3.3989	0.693	2.135	0.330
6.71	29.82	0.3061	0.037	1.250	0.405	0.286	0.075	0.8652	0.251	0.049	0.045
-5.89	29.85	0.0898	0.578	0.519	0.172	0.392	0.690	0.7491	0.030	0.325	0.615
2.017	50.02	1.2599	0.0618	3.367	0.868	1.824	1.298	4.958	0.139	4.143	0.141
16.90	51.13	0.0141	1.709	3.011	2.244	0.415	1.534	0.934	1.455	0.045	1.918
-14.42	49.63	2.4985	0.228	1.150	1.325	1.853	0.567	1.192	0.145	0.429	0.545
-2.10	68.90	3.408	1.126	2.071	0.511	3.955	2.940	3.401	1.107	2.283	1.386
22.70	71.16	0.3426	2.088	0.721	3.738	2.182	0.573	1.685	1.317	0.765	1.789
-21.10	71.23	0.6559	0.432	3.343	2.596	5.352	0.873	0.685	0.285	0.259	0.518

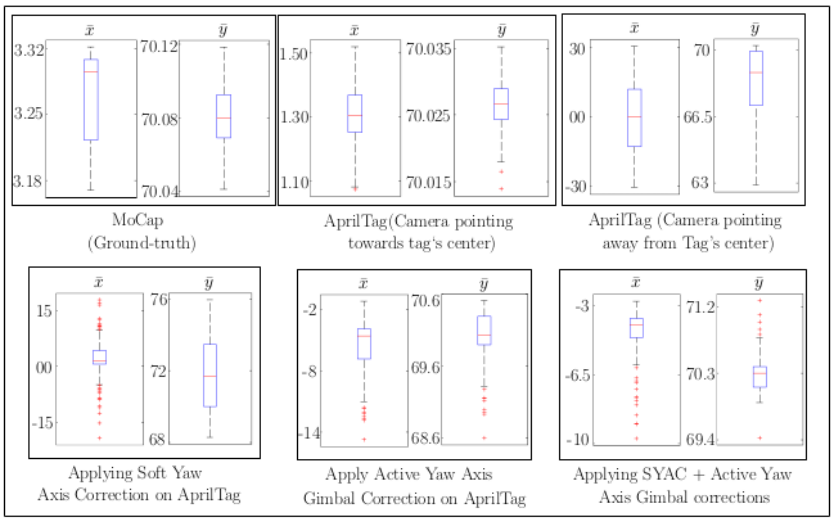


Figure 4-16: Comparison of resulting data spread (precision) from different approaches against the ground truth (Mocap) at nominal reference point straight in front of AprilTag i.e., $(\bar{x}, \bar{y}) = (0, 70)$.

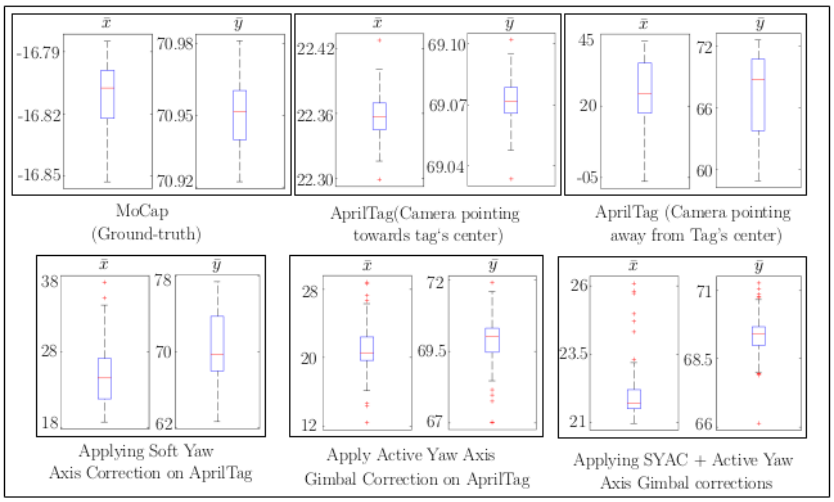


Figure 4-17: With an oblique viewing angle i.e., $(\bar{x}, \bar{y}) = (20, 70)$, a comparison of resulting data spread (precision) from different approaches against the ground truth (Mocap).

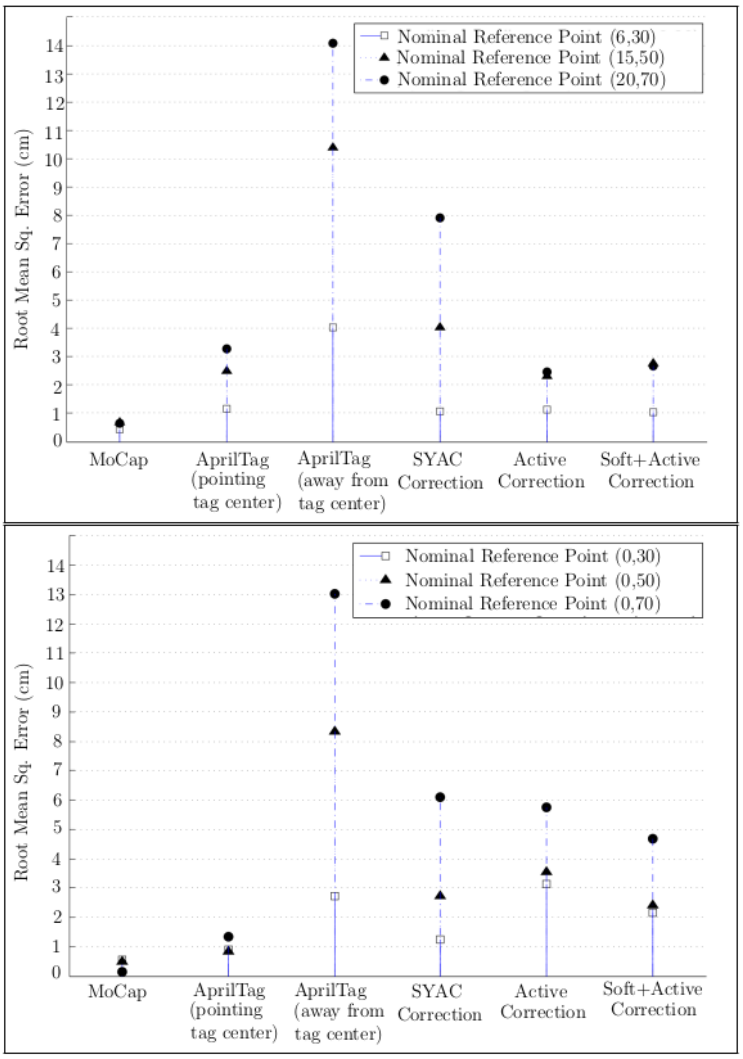


Figure 4-18: Root Mean Square Error (RMSE) comparison of raw AprilTag against proposed approaches and MoCap.

Table 4.10: Average execution time for different approaches.

Approaches	Average Execution Time Per Input Image
Raw AprilTag implementation	90 ms
Passive yaw axis correction (SYAC)	130 ms
Active Correction with yaw axis gimbal	125 ms
(Passive yaw axis + Active gimbal) correction	255 ms

Furthermore, results from Table 4.9, Figure 4-16 and 4-17 have shown that we can achieve significant improvements in the accuracy and precision of the AprilTag by the slight trade-off with execution time. Table 4.10 shows an average execution time for a single input frame for different approaches. Nevertheless, the raw implementation of AprilTag has the quickest execution time as compared to the proposed approaches but the difference is not significant. As illustrated by Table 4.10, a combination of both proposed methods (Passive yaw axis correction + Active correction with yaw axis gimbal) can achieve a maximum operating frequency of approx. 4 Hz, which is acceptable for most of the robotics applications.

4.4.4 Probabilistic Sensor Model for AprilTag

The third contribution of this chapter is the development of a forward probabilistic sensor model $p(Y|X)$ for the AprilTag. The model is based on the collected measurement data and works for all the locations. These locations include both direct and indirect measurement points. Hence, it makes the empirical analysis of the current work applicable to a probabilistic decision-theoretic framework. With reference to Figures 4-2 and 4-11, the true state X of the robot is given by the tuple $X = [\bar{x}, \bar{y}, \bar{\theta}]^T$. The measurement vector Y is also a triplet $Y = [z, x, \phi]^T$. The measurements are assumed to be a nonlinear transformation of the true state, corrupted by some additive sensor noise, $Y = F(X) + \varepsilon$.

Explicitly, these relations can be written as:

$$z = \sqrt{(\bar{z}\cos\bar{\theta} - x)^2 + (z\sin\phi)^2} + \varepsilon_z, \tag{4.8}$$

$$x = \bar{z}\cos\bar{\theta} - z\cos\phi + \varepsilon_x, \tag{4.9}$$

$$\phi = \arctan\left(\frac{\bar{z}\sin\bar{\theta}}{\bar{z}\cos\bar{\theta} - x}\right) + \varepsilon_\phi. \tag{4.10}$$

We are interested in finding the joint probability distribution of the measurement vector given the true states $p\left(z, x, \phi|\bar{x}, \bar{y}, \bar{\theta}\right)$. In order to find the above mentioned probability, Bayes' theorem is applied to obtain:

$$p\left(\bar{x}, \bar{y}, \bar{\theta}|z, x, \phi\right) = \frac{1}{J}\, p\left(z, x, \phi|\bar{x}, \bar{y}, \bar{\theta}\right) p(x, z, \phi), \tag{4.11}$$

where J is a constant that can be factored out. Since we have no prior distribution $p(x, z, \phi)$, one can use a Maximum Likelihood Estimator for a uniform prior, that is, a simple inversion of the model to deduce the states from the measurements by using Equations (4.5), (4.6) and (4.7).

Hence, if we have a model $p\left(\bar{x}, \bar{y}, \bar{\theta}|z, x, \phi\right)$ for *all* states, we can use it to localize at even points where we do not have measurement data. We achieve this using a Gaussian Process (GP) based regression method [170] as follows. First we make a simplifying assumption that Y are not mutually correlated. While this may not be factually true, we find below that this is sufficient for using the Tag in practice. (The extension of the framework to correlated sensor measurements is a work in progress.) Therefore, we focus on either of the measurement variable in Y as scalar nonlinear transformations $f(X)$. These are precisely

the individual measurement equations given above. Using the notation introduced in Reference [170], we are interested in finding the distribution $p(f_*^{test}|X, X_*, Y_*)$, where f_*^{test} is a stochastic process for which $\bar{x}, \bar{y}$ and $\bar{\theta}$ has a joint Gaussian distribution, $X = (\bar{x}, \bar{y}, \bar{\theta})$) is the unknown test point where the distribution has to be calculated, X_* are the ground truth points for training data $X_* \in \{(\bar{x}_i, \bar{y}_i, \bar{\theta}_i)\}_{i=1}^{N}$ and $Y_* \in \{(\bar{x}_i, \bar{y}_i, \bar{\theta}_i)\}_{j=1}^{M}$ are the data collected in experiments as output of AprilTag at training points X_*.

In GP regression, we have to define a covariance function (or Kernel function) whose parameters (the so-called hyper-parameters) have to be tuned to best explain the data at hand. We have chosen a squared exponential covariance function, which is widely used because of its smoothness and differentiability:

$$k(x_a, x_b) = \alpha \exp^{\frac{-|x_a - x_b|^2}{2\beta}} \tag{4.12}$$

where α and β are the hyper-parameters of the kernel function.

The GP regression methodology makes the assumption that the training output Y_* and test output f_*^{test} have a joint Gaussian distribution.(Once again, this is a simplifying assumption that may be invalid in practice but works in practice.)

$$\begin{bmatrix} Y_* \\ f_*^{test} \end{bmatrix} \backsim \mathcal{N}\left(0, \begin{bmatrix} K(X_*, X_*) & K(X_*, X) \\ K(X, X_*) & K(X, X) \end{bmatrix}\right) \tag{4.13}$$

where $K(X_*, X_*)$ is a N $\times$ N matrix defined by covariance function (kernel) evaluated at every training point X_* against each training point X_*. $K(X, X)$ is an $M \times M$ matrix defined by kernel evaluated at every test point X against each test point X. $K(X_*, X)$ is an $N \times M$ matrix formed by the kernel evaluated at every training point X_* against each test point X. Further details can be seen in a standard references on GP (e.g., Reference [170]).

Here, we are only interested in incorporating the knowledge provided by the training data X_* regarding distribution functions other than drawing random functions from

prior knowledge. So we will restrict the joint prior distribution to contain only those functions which agrees with the observed data points Y_* to get the posterior distribution over functions. In other words, we will reject all those functions generated from prior that disagrees with the observations. In probabilistic terms, this can be achieved by marginalizing the observations over the joint distribution to get the predicted distribution as $p(f_*^{test}|X, X_*, Y_*) \backsim \mathcal{N}(\mu, \Sigma)$, where

$$\begin{aligned} \mu &= K(X, X_*)(K(X_*, X_*) + \sigma_A^2 I)^{-1} Y_*, \\ \Sigma &= K(X, X) - K(X, X_*)(K(X_*, X_*) + \sigma_A^2 I)^{-1} K(X_*, X)), \end{aligned} \tag{4.14}$$

where σ_A^2 is the noise variance for the particular AprilTag measurement variable under consideration. The process is repeated for all three measurement variables to regress the distribution for all measurement-state pairs. The results of the regression have been summarized in Table 4.11.

Table 4.11: GP predicted distributions at unseen points against $\alpha = 0.01, \beta = 20000$

Unknown Points	**Predictive Distribution**		**Experimental Distribution**	
$(\bar{x}, \bar{y}, \bar{\theta}(cm, deg))$	$\mu_{f_*^{test}}(cm)$	$\sigma^2_{f_*^{test}}(cm^2)$	$\mu_{x_*}(cm, deg)$	$\sigma^2_{x_*}(cm^2)$
(0,30,90)	(0.4,30.9,95.3)	$(3.9 \times 10^{-8}, 2.2 \times 10^{-6}, 2.6 \times 10^{-6})$	(0.1,30.6,89.77)	$(6.1 \times 10^{-4}, 1.4 \times 10^{-4}, 1.4 \times 10^{-8})$
(10,30,100)	(11.9,30.9,92.2)	$(1.3 \times 10^{-5}, 2.2 \times 10^{-6}, 8.6 \times 10^{-6})$	(10.4,29.5,70.57)	$(3.7 \times 10^{-5}, 4.7 \times 10^{-5}, 4.7 \times 10^{-9})$
(0,50,80)	(0.4,49.9,88.56)	$(3.9 \times 10^{-8}, 3.0 \times 10^{-6}, 2.2 \times 10^{-6})$	(1.0,51.8,88.84)	$(7.5 \times 10^{-4}, 1.0 \times 10^{-5}, 1.0 \times 10^{-9})$
(-15,50,100)	(-16.01,49.9,92.2)	$(3.3 \times 10^{-5}, 3.0 \times 10^{-6}, 8.6 \times 10^{-6})$	(-18.8,52.7,109.6)	$(1.1 \times 10^{-4}, 1.2 \times 10^{-5}, 1.2 \times 10^{-9})$
(20,70,100)	(20.7,70.01,92.2)	$(1.9 \times 10^{-4}, 4.5 \times 10^{-6}, 8.6 \times 10^{-6})$	(22.1,67.0,72.1)	$(1.1 \times 10^{-4}, 1.1 \times 10^{-4}, 1.1 \times 10^{-8})$
(-20,70,100)	(-21.3,70.01,92.2)	$(4.9 \times 10^{-5}, 4.5 \times 10^{-6}, 8.6 \times 10^{-6})$	(-26.1,78.0,109.8)	$(1.8 \times 10^{-4}, 2.4 \times 10^{-5}, 2.4 \times 10^{-9})$

Experimental Verification of Sensor Model

To verify the validity of our proposed AprilTag sensor model, we have used our sensor model in various settings to estimate the state of a robot. We assume a standard odometry model in which robot can rotate around its axis and can move forward [50]. We have performed both indoor and outdoor experiments to validate our proposed sensor model.

For indoor experiment, at any time step t state vector X_t is given by $X_t = [\bar{x}, \bar{y}, \bar{\theta}]^T$, where $\bar{x}$ shows the movement of robot in x-axis, $\bar{y}$ show is the movement in y-axis and $\bar{\theta}$ shows the rotation around robot's own axis. Our goal is to find $p(X_t|X_{t-1}, u_t, z_t)$ where X_{t-1} is the robot state in previous time stamp, u_t is the current input command and z_t is the current sensor measurement.

We have used Monte Carlo simulation technique [195] to estimate the position and pose of the robot since it does not require any prior knowledge for data distribution. In this method, k number of particles are randomly generated around an initial starting point X_i with certain initial uncertainty based upon system

$$X_p \sim \mathcal{N}\left(\begin{bmatrix} x_i \\ y_i \\ \theta_i \end{bmatrix}, \begin{bmatrix} \sigma_{xx}^2 & 0 & 0 \\ 0 & \sigma_{yy}^2 & 0 \\ 0 & 0 & \sigma_{\theta\theta}^2 \end{bmatrix}\right), \tag{4.15}$$

where X_p are the randomly generated particles and $p \in \{1, ..., k\}$, x_i is the initial value for x-axis, y_i is the initial value for y-axis, θ_i us the initial angle and $\sigma_x^2, \sigma_y^2, \sigma_\theta^2$ are the initial variance in x-axis, y-axis and θ respectively. Then, each particle is propagated forward based upon the motion model assumed

$$X_t = f_t(X_{t-1}) + n = f_t(x_{t-1}, y_{t-1}, \theta_{t-1}) + n, \tag{4.16}$$

where f_t is a function representing motion model of the system and n is the Gaussian noise. Then observation model is applied on each propagated particle to get observation

measurements as $\hat{z}_t$. Then these observation measurements are weighted against the measurement data from the sensor z_t. Each particle is assigned a probabilistic weight based upon how close it is to the measurement after applying observation model as

$$P^p_{weight} = \frac{1}{\sqrt{(2\pi)^3 \det\{R\}}} * exp\left(-\frac{1}{2}(z_t - \hat{z}_t)R^{-1}(z_t - \hat{z}_t)^T\right), R = \begin{bmatrix} r_x & 0 & 0 \\ 0 & r_y & 0 \\ 0 & 0 & r_\theta, \end{bmatrix} \quad (4.17)$$

where $p \in \{1, ..., k\}$ represents number of particles, R is a 3x3 co-variance matrix. Then the assigned probability weights are normalized such that their sum is equal to 1 as

$$P_{CDF} = \frac{P^p_{weight}}{\sum_{n=0}^{k} P^n_{weight}}, \quad (4.18)$$

where P^p_{CDF} is the cumulative distribution of the probability density of weighted vector P^p_{weight}. Then weighted particles are re-sampled for the next step by uniformally sampling from the cumulative distribution as shown in equation 4.19. Since the particles are being selected by statistical probabilities, so on average, particles with the greater weights are being selected.

$$X_p = P^{-1}_{CDF}(h), \text{ where } h \sim \mathcal{U}(0,1). \quad (4.19)$$

After getting new particles, the whole process is repeated for m_o times where m_o is the total number of tag observations observed in an experiment. At every step, the average of all the particles is considered to be the true position of the robot. This algorithm relies on the survival of the fittest philosophy. Those particle which are close to the sensor measurement are weighted higher then others give them the chance to be selected again for the next round.

In our experiment, an incremental motion model has been used for the propagation of

particles from one configuration c_i to another configuration c_f. Three parameters $\delta\theta_i$, δd and $\delta\theta_f$ have been used to encode the complete motion from one configuration to another. Input command u_{θ_i} maps as rotation $\delta\theta_i$ of robot at initial configuration c_i such that it faces final configuration c_f. u_d maps as the straight forward motion δd from initial configuration c_i to final configuration c_f and u_{θ_f} maps the final rotation $\delta\theta_f$ at destination point for final pose angle. See Figure 4-19 which shows each parameter in detail.

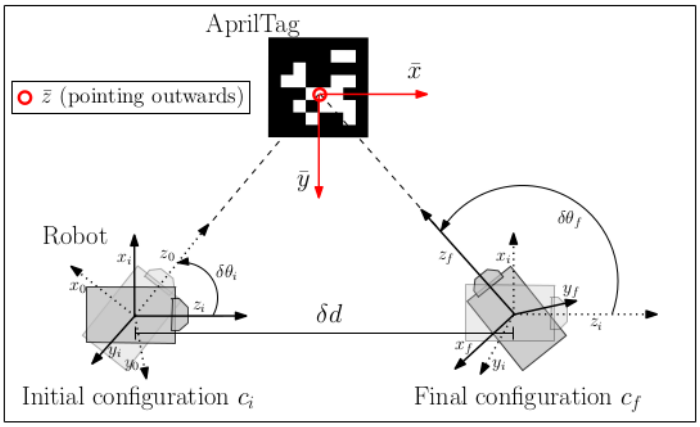


Figure 4-19: Incremental motion model used between two configuration points c_i and c_f, encoded by three parameters $\delta\theta_i$, δd and $\delta\theta_f$ for Monte Carlo simulation.

For proposed sensor model verification using Monte Carlo simulation, we used $10,000$ particles initially generated at a known starting point $X_i = [x_i, y_i, \theta_i]$ with initial variance of n_i. As it is assumed earlier, our robot can only move forward and can rotate around its own axis, so motion model for each particle can be given by

$$X_t = \begin{bmatrix} x_{t-1} + d_{for}\cos(\theta_{t-1} + \delta\theta_i) + n_x \\ y_{t-1} + d_{for}\sin(\theta_{t-1} + \delta\theta_i) + n_y \\ \theta_{t-1} + \delta\theta_i + \delta\theta_f + n_\theta \end{bmatrix}, \tag{4.20}$$

where d_{for} is the forward distance as a result input command u_{for}, $\delta\theta_i$ is the angle of rotation at initial position as a result of input command u_{θ_i}, $\delta\theta_f$ is the rotation angle at final configuration point as a result of input command u_{θ_f}. n_x is the Gaussian noise in x-axis, n_y is the Gaussian noise in y-axis and n_θ is the Gaussian noise in θ.

At any time t, measurement vector is given by $z_t = [x_{tag}, y_{tag}, \theta_{tag}]^T$. In this experiment, we have assumed that x, y and θ are independent in nature. So for observation model, the proposed AprilTag sensor model as shown in Equation 4.14 has been used.

Figure 4-20 shows the trajectory generated by applying the particle filter empowered with our proposed AprilTag sensor model in comparison with the ground truth generated by MoCap. AprilTag's center is placed at $(\bar{x}, \bar{y}, \bar{\theta}) = (0, 0, 90)$ over a calibrated setup and robot is moved in front of AprilTag in a rectangular shape. The rectangular shape is selected to have a better visualization of trajectory data and to see the loop closure. Figure 4-20 shows that the trajectory generated by the particle filter (red) is very close to the ground truth trajectory (green). The experiment shows that the particles converge very quickly because of the high precision of the system achieved by applying proposed techniques.

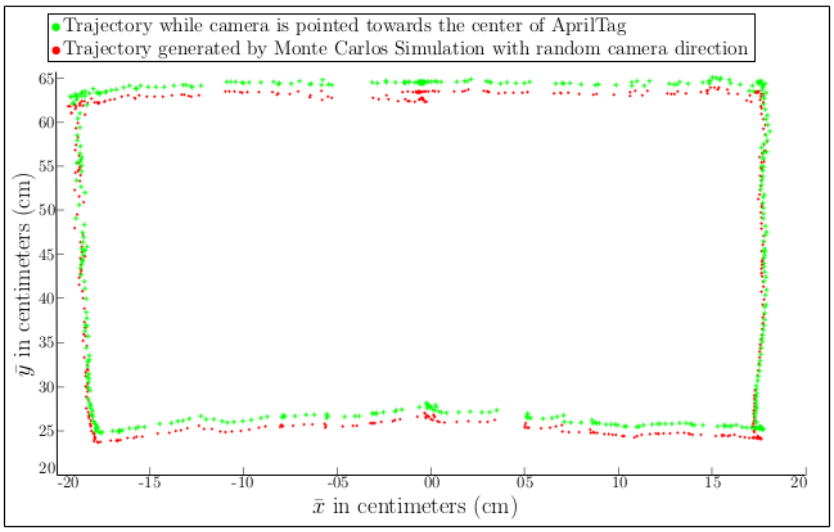
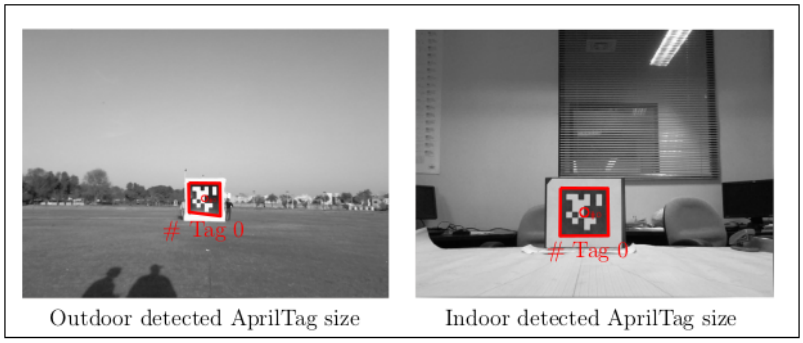


Figure 4-20: Trajectory comparison between MoCap and trajectory generated by Monte Carlo simulation using our proposed AprilTag sensor model.

To further investigate the performance of our proposed sensor model, a similar experiment in a large outdoor environment has been performed. For this purpose, a larger AprilTag of size 305 × 305 cm fixed on the ground has been used, as shown in Figure 4-21. In this experiment, the robot has moved along an irregular path from the left side of the AprilTag to the right side as far as the tag is visually detectable and then back to the left side towards starting position, as shown in Figure 4-22. To show the significance of each proposed improvement, we have divided the experiment into two phases. In phase one, active tracking of AprilTag is not activated and only passive correction is done using the sensor model (red path). In phase two, active tag tracking is also activated along with the passive correction (blue path), as shown in Figure 4-22. Considering the experiment is in an outdoor environment, therefore ground truth trajectory can not be generated. Hence for ground truth verification, we have manually marked three validation points in meters that is, $A(\bar{x},\bar{y})$=(0,40), $B(\bar{x},\bar{y})$=(26,30) and $C(\bar{x},\bar{y})$=(-19,20) before the experiment and have

deliberately passed through them. Figure 4-22 shows that the trajectory passes through the validation points.



Outdoor detected AprilTag size Indoor detected AprilTag size

Figure 4-21: *Left*: Camera view of detected AprilTag (red polygon) in outdoors. *Right*: Camera view of detected AprilTag (red polygon) in indoors. Both images show detection polygons along with the detected tag IDs based on implanted code.

Moreover previously proposed pose-indexed probabilistic sensor model in Equation 4.14 is regressed over an indoor small scale experimental data. The training points are at the maximum of 1 meter from the AprilTag. Therefore, the model trained by using Gaussian Processes(GP) 4.14 is only valid for sub-meter trajectories. To make it workable in long distances, we have proposed a general sensor model with some scale factor d, where d is the distance of the camera from the tag along the $\bar{y}$-axis. To calculate the scale factor d, we have used the equality as shown in Equation (4.21).

$$d = \frac{f \times h_r \times I_p}{h_p \times S_s \times c} \tag{4.21}$$

where f is focal length of camera in mm, h_r is the real height of the AprilTag in mm, I_p is the height of image sensor in pixels, h_p is the AprilTag height in pixels and S_s is the image sensor's height in mm. c is a constant to change the unit scale. Since for outdoor experiment, wehave used meters as our unit of choice for distances, so we have used

$c = 1000$. After evaluating scale factor d, our general sensor model would become:

$$
\begin{aligned}
\mu_G &= K_G(X, X_*^G)(K_G(X_*^G, X_*^G) + \sigma_A^2 I)^{-1} Y_*^G, \\
\Sigma_G &= K_G(X, X) - K(X, X_*^G)(K_G(X_*^G, X_*^G) + \sigma_{GA}^2 I)^{-1} K_G(X_*^G, X)), \\
K_G &= k_G(x_a, x_b) = d^2 \times \alpha \exp\left(\frac{-|x_a - x_b|^2}{2 \times d^2 \times \beta}\right), \\
X_*^G &= X_* \times d,\ Y_*^G = Y_* \times d\,,\ \sigma_{GA}^2 = \sigma_A^2 \times d^2.
\end{aligned}
\tag{4.22}
$$

Here μ_G and σ_G is the mean value and variance for test point X using generalized sensor model respectively. K_G is the generalized kernel, X_*^G is the generalized training point and Y_*^G is the generalized observed value. We have empirically tested and verified experimentally that the generalized sensor model gives almost same result at certain distance 'd'.

Figure 4-22 shows the trajectory generated by our generalized sensor model in an outdoor environment by using Monte Carlo Simulation. Figure 4-23 shows the axis-wise plot of raw AprilTag data (red) and the particle filter output (blue). It shows the filter is filtering the noise and improving the overall performance.

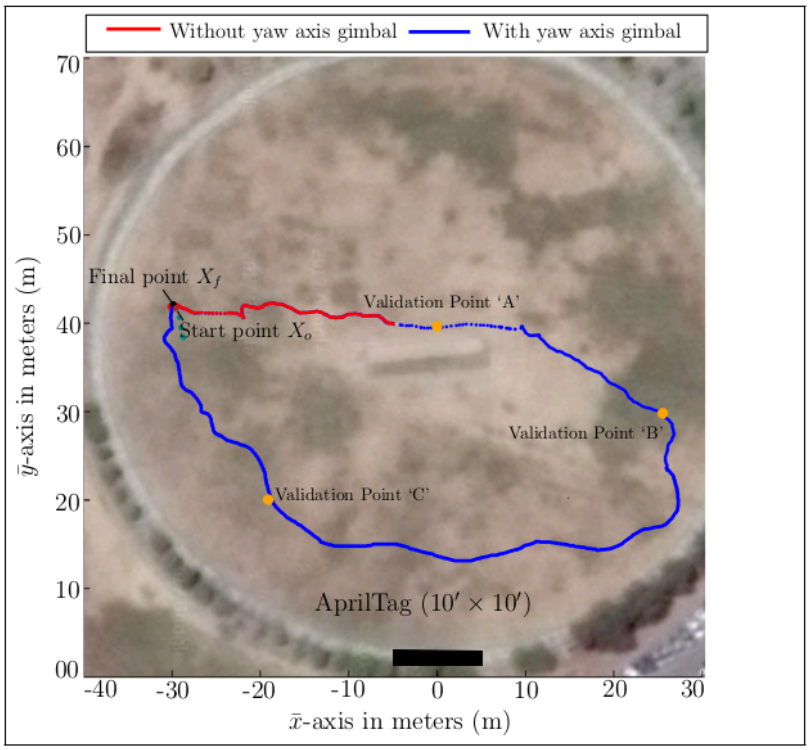


Figure 4-22: Trajectory generated using Monte Carlo Simulation in an outdoor environment.

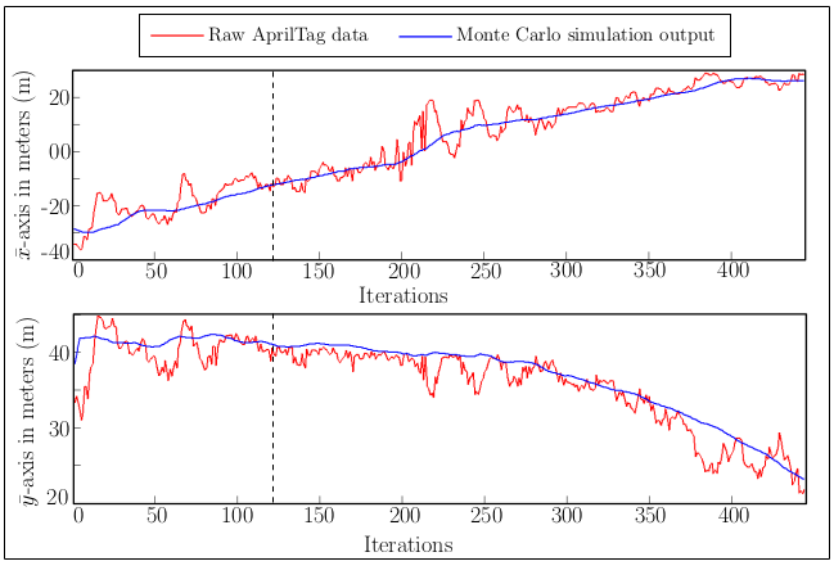


Figure 4-23: Comparison of raw AprilTag data and proposed generalize sensor model based particle filter output along $\bar{x}$-axis and $\bar{y}$-axis. Dotted line shows the initialization of yaw axis gimbal for active correction.

Part III

State Estimation, 3D Environment Mapping & Autonomous Navigation

Chapter 5

State Estimation & Mapping for the Aerial Structural Inspection of the Water Canals

In the previous section, we have proposed the system integration of an aerial platform equipped with all the necessary sensor systems required to traverse and reconstruct the structural survey of the water canal. Lately, we proposed a novel technique to improve the truthfulness and robustness of an external marker AprilTag for the ground-truth and accuracy analysis of the localization system of the aerial robot. Henceforward in the current chapter, we propose our state estimation technique and its accuracy with the help of experimental results.

5.1 Introduction & State-of-the-Art State Estimation Systems

In this chapter, a novel state estimation of an aerial system has been proposed for the 3D structural inspection of water canals. The proposed state estimation calculates the three-dimensional position and pose of the aerial robot and generates a precise trajectory

of the flying robot over the course of its operation. The precise trajectory in later used to reconstruct the environment with the help of LiDAR for the structural inspection which includes a 3D mapping of the bed of the canal along with the both left and right banks. As discussed in the previous section, the robot is equipped with the state of the art sensors and has the capability of making highly accurate 3D models of cross-sections of canals for accurate silt volume estimation and structural damages.

State estimation is one of the most challenging parts of aerial robotics. Especially for an under-actuated aerial robot-like multirotor, state estimation plays an important role to perform tasks autonomously in 6DOF. Different techniques and sensors have been used for accurate state estimation. Some of the techniques use onboard sensors and some use external tracking systems which limits the flight in a deterministic lab environment. Muller et.al. [149] used an external tracking system to track the state of the quadrotor and produced a required trajectory to bounce a ball back using a racket attach. Same tracking technique is used by [141],[142]. Manually previously recorded trajectory can also enable quadrotor to follow a certainly known trajectory [112]. The use of GPS can also help the multirotor explore the unstructured outdoor environment without prior knowledge [166].

Another important technique to improve the state estimation of multirotor is to incorporate visual feed with IMU. Other than an external tracking system, external makers can also be used and tracked by a monocular camera to get the exact position and pose of the flying robot [59]. Amidi et.al. [11] presented an odometer approach that first track and locks a moving target on the ground and then estimate the position and pose using a stereo camera along with IMU readings. [156] presented an overview related to how different perception techniques have increased the precision of aerial robots mainly in helicopters. However, most of them are lately used by researchers for quadrotors. P Corke et.al. [51] used feature-based stereo imaging along with sonar to measure the height of the helicopter more accurately. [209] also used the known external target to calculate the pose and position of a helicopter using an Extended Kalman filter [124]. [37] presents a

visual approach to estimate translation of helicopter-based upon views of a planer patch of a monocular camera.

Most of the researchers have treated state estimation as a non-linear optimization problem. Davison et.al. [55] and [64] have presented a recursive filtering sparse technique to track natural landmarks for state estimation. However, [191] shows that optimization approaches outperform recursive filtering techniques mostly.

Parallel Tracking and Mapping [110] has been widely used for visual odometry [29] [206]. This technique relies on tracking and mapping hundred of features in real-time and estimating position and mapping simultaneously at the same time using Bandle Adjustment [111]. H Jin et.al. [101], [184] use a direct approach to simultaneously obtain the camera pose, correspondences among frames, and the scene structures. In the direct approach, instead of using features, pixel intensities are used directly to calculate the transformation between two consecutive frames. [62] have used PTAM from a monocular camera along with an IMU and used Extended Kalman Filter for data fusion. However, C Froster et.al. [74] propose a semi-direct visual state estimation approach that is robust and fast from previous techniques. In this approach, processing extensive tasks like feature extraction and tracking are replaced by direct pixel-level operations which not even make it faster but also increase the accuracy to sub-pixel level.

Along with state estimation, a lot of work has also been done in 3D mapping for aerial robots. Simultaneous localization and mapping (SLAM) techniques used for aerial mapping are similar to that of ground robots techniques however only the localization part differs as it deals with 6-DOF instead of 3-DOF. Initially, Nuchter et.al [155] present a 6D SLAM which is based upon locally consistent 3D range scan and used ICP [217] to align them. D Borrmann et.al [30] proposed a 3D mapping approach that incorporates position and pose(roll, pitch, and yaw) to create a globally consistent map and later used Taylor Expansion and Cholesky decomposition to process huge 3D data in real-time.

In aerial robots, [92] have used RGB-D images along with IMU to localize mini mul-

tirotor between two frames and have generated occupancy grid-based 3D map. [145] present a collaborative technique to build a common 3D map from the feeds of both ground and aerial robots. Merging of the map is done via a globally known common point that both robots deliberately visit. Then iterative closest point algorithm is used to align both maps. [151] constructs a 3D map from digital imagery captured by octocopter. Another interesting work has been done by [61] which proposes to build large-scale 3D maps using image pixel intensities instead of extraction of features.

5.2 State Estimation

State Estimation lies at the heart of the water canal structural mapping problem. Considering all the laser scans are being taken from an onboard LiDAR, hence precise pose is required to have an accurate canal structural map and precisely co-relate the current scan with the previous scans. To achieve the required pose precision, the system has fused data from different sensor systems using an Extended Kalman Filter [124]. Fused sensor systems include Inertial Measurement Unit(IMU), Global Positioning System(GPS), Monocular camera, Barometer, and Head-LiDAR. Table 5.1 shows the naming convention used in this paper for each sensor system frame of reference integrated on the multi-copter.

Frame of reference against each integrated sensor system			
Sensor Name	**Reference Frame**	**Measurement Vector(z)**	**Observation Function(O)**
IMU	γ	$\mathbf{z}_{\gamma,t}$	$O_{\gamma}(\mathbf{x_t})$
Barometer	α	$\mathbf{z}_{\alpha,t}$	$O_{\alpha}(\mathbf{x_t})$
Camera	λ	$\mathbf{z}_{\lambda,t}$	$O_{\lambda}(\mathbf{x_t})$
Head-LiDAR	ω	$\mathbf{z}_{\omega,t}$	$O_{\omega}(\mathbf{x_t})$
GPS	ν	$\mathbf{z}_{\nu,t}$	$O_{\nu}(\mathbf{x_t})$

Table 5.1: The frame of reference guide of each sensor system that is integrated on the multi-copter for data fusion along with the corresponding Measurement vector and Observation function.

Our proposed state vector $\mathbf{x_t}$ has 15 state variables as

$$\mathbf{x_t} = [x_t, y_t, z_t, \dot{x}_t, \dot{y}_t, \dot{z}_t, \phi_t, \theta_t, \psi_t, \dot{\phi}, \dot{\theta}, \dot{\psi}, \ddot{x}_t, \ddot{y}_t, \ddot{z}_t]^T,$$

where x_t, y_t and z_t refers as the positions in meters along the x-axis, y-axis and z-axis of the map frame of reference 'M'respectively, $(\dot{x}_t, \dot{y}_t, \dot{z}_t)$ as their linear velocities in m/s and $(\ddot{x}_t, \ddot{y}_t, \ddot{z}_t)$ as their linear accelerations in m/s^2 all in the map frame of reference. Moreover, ϕ_t,θ_t,ψ_t are the rotation angles in *degree* around y-axis, x-axis and z-axis respectively of the same map frame of reference and $(\dot{\phi}, \dot{\theta}, \dot{\psi})$ refers as their angular velocities in $degree/s$.

To track the transformations between poses of an octocopter at a high frequency, the proposed system has used '9-DOF Sharpfun Razor' as an inertial measurement unit(IMU). It has a 3-axis accelerometer, 3-axis magnetometer, and 3-axis gyroscope. It works on a maximum frequency of $200Hz$. The IMU is attached to the body of the multi-rotor exactly at the point of the center of the mass of the multi-rotor. In addition the frame of reference of multi-rotor 'O'and IMU 'γ 'are aligned. The measurement vector $\mathbf{z}_{\gamma,t}$ for IMU is given by:

$$\mathbf{z}_{\gamma,t} := [\dot{x_{\gamma,t}}, \dot{y_{\gamma,t}}, \dot{z_{\gamma,t}}, \phi_{\gamma,t}, \theta_{\gamma,t}, \psi_{\gamma,t}, \dot{\phi_{\gamma,t}}, \dot{\theta_{\gamma,t}}, \dot{\psi_{\gamma,t}}, \ddot{x_{\gamma,t}}, \ddot{y_{\gamma,t}}, \ddot{z_{\gamma,t}}]^T,$$

where $\dot{x_\gamma}$, $\dot{y_\gamma}$ are the horizontal velocities in the multi-copter co-ordinate frame of reference, $\dot{z_\gamma}$ is the vertical velocity in the multi-copter co-ordinate frame,$(\phi_\gamma, \theta_\gamma, \psi_\gamma)$ and $(\dot{\phi_\gamma}, \dot{\theta_\gamma}, \dot{\psi_\gamma})$ are the roll, pitch and yaw angles and their respective angular velocities in the multi-copter frame of reference, $(\ddot{x_\gamma}, \ddot{y_\gamma})$ are horizontal linear accelerations along x-axis and y-axis of the octocopter and $\ddot{z_\gamma}$ is the vertical acceleration in the multi-copter frame of reference. A constant time interval of $10ms$ is used to generate linear velocities from the IMU raw data

output. Hence its sensor model $O_\gamma(x_t)$ can be written as:

$$O_\gamma(\mathbf{x_t}) := \begin{bmatrix} \dot{x}_t \cos\psi_t - \dot{y}_t \sin\psi_t \\ \dot{x}_t \sin\psi_t + \dot{y}_t \cos\psi_t \\ \dot{z}_t \\ \arctan\left(\frac{\ddot{y}_t}{\sqrt{\ddot{x}_t^2 + \ddot{z}_t^2}}\right) \\ \arctan\left(\frac{\ddot{x}_t}{\sqrt{\ddot{y}_t^2 + \ddot{z}_t^2}}\right) \\ \arctan\left(\frac{-y_t \cos\phi_t + z_t \sin\phi_t}{x_t \cos\theta_t + y_t \sin\theta_t \sin\phi_t + z_t \cos\phi_t \sin\theta_t}\right) \\ \dot{\phi}_t \\ \dot{\theta}_t \\ \dot{\psi}_t \\ \ddot{x}_t \\ \ddot{y}_t \\ \ddot{z}_t \end{bmatrix}.$$

Here velocities along x-axis and y-axis are transformed from multi-copter co-ordinate frame system 'O'to the map co-ordinate system 'M', $\dot{z}_t$ represents difference in altitude taking it as a direct observation. $\ddot{x}_t$, $\ddot{y}_t$ and $\ddot{z}_t$ are the accelerometer readings alone x, y and z-axis. Angular velocities ($\dot{\phi}_t$,$\dot{\theta}$,$\dot{\psi}_t$) and linear accelerations ($\ddot{x}_t$, $\ddot{y}_t$, $\ddot{x}_t$) have not been transformed from multi-copter co-ordinate frame system to map co-ordinate frame system as they are considered as direct observations with the similar affect.

To further enhance the precision of the altitude of the aerial robot, barometer readings have been fused with other altitude observations. Our system has used a barometer to detect the change in altitude. Therefore, its observation model $O_\alpha(\mathbf{x_t})$ and measurement vector $\mathbf{z}_\alpha$ can be written as:

$$O_\alpha(\mathbf{x_t}) = \begin{bmatrix} z_t \end{bmatrix} \quad = \quad \mathbf{z}_{\alpha,t},$$

As the altitude, z_t has been used as a direct observation thus no transformation is needed.

Although IMU gives precise pose tracking at a high frequency, it starts drifting due to the inherent inaccurate nature of IMU pose and position calculations. Hence it is not a reliable sensor system for long flight operations as we intend to perform over a water canal. To remunerate this shortcoming, our system also uses a monocular camera to generate visual odometry as discussed in the previous section. To fuse the visual odometry pose, measurement vector for visual odometry $\mathbf{z}_{\lambda,t}$ is:

$$\mathbf{z}_{\lambda,t} = [x_{\lambda,t}, y_{\lambda,t}, z_{\lambda,t}, \phi_{\lambda,t}, \theta_{\lambda,t}, \psi_{\lambda,t}]^T,$$

where $(x_\lambda, y_\lambda, z_\lambda)$ and $(\phi_\lambda, \theta_\lambda, \psi_\lambda)$ are the position and orientation of multi-copter in camera frame of reference 'λ' respectively. Hence, observation model $O_\lambda(\mathbf{x_t})$ can be written as:

$$O_\lambda(\mathbf{x_t}) = \left[T^\lambda_{O,t} T^O_{M,t} \mathbf{c_t}\right], \quad \mathbf{c_t} = [x_t, y_t, z_t, \phi_t, \theta_t, \psi_t]^T \subset \mathbf{x_t}$$

where $T^O_{M,t}$ is the homogeneous pose transformation from the map co- 'M'to the multi-copter co-ordinates frame of reference 'O'and $T^\lambda_{O,t}$ is the pose transformation from the multi-copter co-ordinates 'O'to the camera co-ordinates 'λ '.

Horizontal positioning and heading accuracy are further enhanced by fusing 3-DOF odometry calculated with the help of the Head-LiDAR scans using range flow-based approach [98]. To calculate the odometry, the system has used consecutive scans of the head-mounted 2D LiDAR and are matched together to get the instantaneous transformation between the two consecutive poses. Therefore, observation model $O_\omega(\mathbf{x_t})$ and measurement vector $\mathbf{z}_{\omega,t}$ can be written as:

$$O_\omega(\mathbf{x_t}) = \left[A^\omega_{O,t} A^O_{M,t} \mathbf{m_t}\right], \quad \mathbf{m_t} = [x_t, y_t, \psi_t]^T \subset \mathbf{x_t}$$

$$\mathbf{z}_{\omega,t} = [x_{\omega,t}, y_{\omega,t}, \psi_{\omega,t}]^T \quad where$$

$$A^{O}_{M,t} = \begin{bmatrix} c\psi_t & -s\psi_t & d_{x,t} \\ s\psi_t & c\psi_t & d_{y,t} \\ 0 & 0 & 1 \end{bmatrix}, \quad A^{\omega}_{O,t} = \begin{bmatrix} c\psi_{o,t} & -s\psi_{o,t} & e_{x,t} \\ s\psi_{o,t} & c\psi_{o,t} & e_{y,t} \\ 0 & 0 & 1 \end{bmatrix}.$$

Here, $A^{O}_{M,t}$ and $A^{\omega}_{O,t}$ are the transformation matrices to transform the vector $\mathbf{x_t}$ from map coordinate frame to multi-copter frame of reference and then from multi-copter frame of reference to Head-LiDAR frame of reference 'ω'. (d_x, d_y) is the horizontal displacement between map frame and octocopter frame, (e_x, e_y) is the horizontal displacement between the multi-copter frame and the Head-LiDAR frame of reference. c represents cos and s represents sin angles and $\psi_{o,t}$ represents the yaw angle in the multi-copter frame of reference. In measurement vector $Z_{\omega,t}$, $(x_{\omega,t}, y_{\omega,t})$ and $\psi_{\omega,t}$ represents the horizontal position and yaw angle in the Head-LiDAR frame of reference.

Additionally, GPS data has also been fused in the Extended Kalman filter for global consistency. As GPS reports the global position in longitude and latitude with altitude, therefore its observation model $O_v(\mathbf{x_t})$ and measurement vector $\mathbf{z}_{v,t}$ can be written as:

$$O_v(\mathbf{x_t}) = \begin{bmatrix} B^{v}_{O,t} B^{O}_{M,t} \mathbf{g_t} \\ z_t \end{bmatrix}, \quad \mathbf{g_t} = [x_t, y_t]^T \subset \mathbf{x_t}$$

$$\mathbf{z}_{v,t} = [x_{v,t}, y_{v,t}, (h_{v,t} - h_{v,t-1})]^T \quad where$$

$$B^{O}_{M,t} = \begin{bmatrix} I & \mathbf{a_t} \\ 0 & 1 \end{bmatrix}_{3\times3}, \mathbf{a_t} = [d_{x,t}, d_{y,t}]^T,$$

$$B^{v}_{O,t} = \begin{bmatrix} I & \mathbf{a}_\circ \\ 0 & 1 \end{bmatrix}_{3\times3}, \mathbf{a}_\circ = [a_{x,t}, a_{y,t}]^T.$$

Similarly, $B^{O}_{M,t}$ and $B^{v}_{O,t}$ are the transformation matrices from the map frame of reference to multi-copter frame and from multi-copter frame to GPS frame of reference 'v'respectively. Vector $\mathbf{a}_\circ$ represents the horizontal displacement between the multi-copter frame and the

GPS frame of reference 'ν'. In measurement vector $\mathbf{z}_{\nu,t}$, $(x_{\nu,t}, y_{\nu,t})$ represents horizontal position in GPS frame of reference and $(h_{\nu,t} - h_{\nu,t-1})$ is the change in altitude in GPS local co-ordinates.

Moreover, multi-copter poses complex dynamics which is not trivial to model completely. Therefore for process model, the proposed system has used multi-copter approximated dynamical model using Newton-Euler equations [129] based upon control input $\boldsymbol{\mu}_t = \{\dot{\mu_{z,t}}, \mu_{\phi,t}, \mu_{\theta,t}, \dot{\mu_{\psi,t}}\}$ and series of tuning parameters κ [62]. In $\boldsymbol{\mu}_t$, $\dot{\mu_{z,t}}$ represents the altitude velocity command, $\mu_{\phi,t}$ is the roll angle 'ϕ'input angle command, $\mu_{\theta,t}$ is the pitch angle 'θ'and $\dot{\mu_{\psi,t}}$ shows the angular velocity command for yaw angle 'ψ'. Whereas, $\kappa_1, \kappa_2, ...$ are the proportionality coefficients that are turned after rigorous experimentation and they change with different payload configurations as it depends upon the mass of the multi-rotor.

For our approximated multi-copter dynamic model, we assume the thrust generated by each propeller is constant. State transition equations for a x_t, y_t and z_t are linear and are given by:

$$x_{t+1} \approx x_t + \delta_t(\dot{x_t}), \quad y_{t+1} \approx y_t + \delta_t(\dot{y_t}), \quad z_{t+1} \approx z_t + \delta_t(\dot{z_t}). \tag{5.1}$$

Practically, we have used $\delta_t = 100ms$ as the sampling frequency for our state estimation fusion filter. During the flight, the forces applied on the multi-rotor can be calculated using Newton's second law of motion as:

$$F_M = m\ddot{\zeta} = F_W + F_T + F_D \tag{5.2}$$

where ζ represents the positional states (x_t, y_t, z_t) of the multi-rotor in map frame of reference 'M', F_M is the force applied on the multi-rotor in map frame of reference, F_W is the weight of the multi-rotor due to gravity, F_T is the thrust force generated by the propellers and F_D is the opposing drag force which is proportional to the horizontal and vertical

velocities. The force F_W due to the weight of the multi-rotor as being given as:

$$F_W = \begin{bmatrix} 0 & 0 & -mg \end{bmatrix}^T, \tag{5.3}$$

where m is the mass of the multi-rotor, g is the gravity pointing downwards the ground. The force F_T due to the thrust of the motors acts on the $z-axis$ only and can be expressed as:

$$F_T = \begin{bmatrix} c\psi_t c\theta_t & c\psi_t s\theta_t c\phi_t - s\psi_t c\phi_t & c\psi_t s\theta_t c\phi_t + s\psi_t s\phi_t \\ s\psi_t c\theta_t & s\psi_t s\theta_t s\phi_t + c\psi_t c\phi_t & s\psi_t s\theta_t c\phi_t - c\psi_t s\theta_t \\ -s\theta_t & c\theta_t s\phi_t & c\theta_t c\phi_t \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ j\Sigma_{i=1}^{8}\omega_i^2 \end{bmatrix}$$

$$F_T = j\Sigma_{i=1}^{8}\omega_i^2 \begin{bmatrix} c\psi_t s\theta_t c\phi_t + s\psi_t s\phi_t \\ s\psi_t s\theta_t c\phi_t - c\psi_t s\theta_t \\ c\theta_t c\phi_t \end{bmatrix} = \kappa_1 \begin{bmatrix} c\psi_t s\theta_t c\phi_t + s\psi_t s\phi_t \\ s\psi_t s\theta_t c\phi_t - c\psi_t s\theta_t \\ c\theta_t c\phi_t \end{bmatrix} \tag{5.4}$$

where j is the thrust constant, ω_i is the angular velocity of the i^{th} propeller, and c and s are cosine and sin respectively. For simplicity of the dynamic model, we have assumed κ_1 as a proportionality coefficient approximating the net effect of the overall thrust. It is calculated and tuned by a considerable number of experiments. Similarly, the drag force F_D can also be expressed as:

$$F_D = B\dot{\zeta}_t = \begin{bmatrix} -B_x & 0 & 0 \\ 0 & -B_y & 0 \\ 0 & 0 & -B_z \end{bmatrix} \begin{bmatrix} \dot{x}_t \\ \dot{y}_t \\ \dot{z}_t \end{bmatrix} = - \begin{bmatrix} B_x\dot{x}_t \\ B_y\dot{y}_t \\ B_z\dot{z}_t \end{bmatrix} = -\kappa_2 \begin{bmatrix} \dot{x}_t \\ \dot{y}_t \\ \dot{z}_t \end{bmatrix}, \tag{5.5}$$

where B is the matrix containing drag coefficients related to translational motion, B_x,B_y and B_z are the x-axis, y-axis, and z-axis components of the translational drag coefficients respectively and κ_2 is the again the proportionality coefficient approximating the opposing

drag force. By using Equations 5.3, 5.4 and 5.5, Equation 5.2 can further be expanded as:

$$\begin{bmatrix} \ddot{x}_t \\ \ddot{y}_t \\ \ddot{z}_t \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + \kappa_1 \begin{bmatrix} c\psi_t s\theta_t c\phi_t + s\psi_t s\phi_t \\ s\psi_t s\theta_t c\phi_t - c\psi_t s\theta_t \\ c\theta_t c\phi_t \end{bmatrix} + -\kappa_2 \begin{bmatrix} \dot{x}_t \\ \dot{y}_t \\ \dot{z}_t \end{bmatrix} \tag{5.6}$$

Consequently, the translational motion of the multi-rotor can be written as:

$$\ddot{x}_t = \kappa_1(c\psi_t s\theta_t c\phi_t + s\psi_t s\phi_t) - \kappa_2(\dot{x}_t), \tag{5.7}$$

$$\ddot{y}_t = \kappa_1(s\psi_t s\theta_t c\phi_t - c\psi_t s\theta_t) - \kappa_2(\dot{y}_t) \tag{5.8}$$

and

$$\ddot{z}_t = \kappa_1(c\theta_t c\phi_t) - \kappa_2(\dot{z}_t) - g. \tag{5.9}$$

Here, $\kappa_1(c\theta_t c\phi_t)$ is dependent upon the command input $\dot{\mu_{z,t}}$, so Equation 5.9 can also be approximated as a function of both the vertical acceleration and the input vertical command velocity by replacing the thrust velocity with the command input velocity for z_t,

$$\ddot{z}(\boldsymbol{x_t}, \boldsymbol{\mu_t}) = \kappa_3(\dot{\mu_{z,t}}) - \kappa_4(\dot{z}_t). \tag{5.10}$$

Here, κ_3 and κ_4 are the proportional coefficients again turned manually for the vertical acceleration $\ddot{z}(\boldsymbol{x_t}, \boldsymbol{\mu_t})$. For our proposed system, the state transition equation for $\dot{x}_t$ becomes,

$$\dot{x_{t+1}} \approx \dot{x}_t + \delta_t(\ddot{x}_t), \tag{5.11}$$

Similarly, state transition equations for $\dot{y_t}$ and $\dot{z_t}$ can also be expressed as:

$$\dot{y_{t+1}} \approx \dot{y_t} + \delta_t(\ddot{y_t}), \tag{5.12}$$

$$\dot{z_{t+1}} \approx \dot{z_t} + \delta_t(\ddot{z_t}). \tag{5.13}$$

Moreover, ϕ_t and θ_t are also dependent upon command inputs $\mu_{\phi,t}$ and $\mu_{\theta,t}$ and their angular velocities can be approximated by the difference of command input and the current angle as:

$$\phi_{t+1} \approx \phi_t + \delta_t(\kappa_5\mu_{\phi,t} - \kappa_6\phi_t) \tag{5.14}$$

$$\theta_{t+1} \approx \theta_t + \delta_t(\kappa_7\mu_{\theta,t} - \kappa_8\theta_t), \tag{5.15}$$

where κ_5 to κ_8 is the approximation proportional coefficients calculated empirically. Similarly, Yaw angle ψ_t with tuning coefficient κ_9 can be expressed as:

$$\psi_{t+1} = \psi_t + \delta_t(\dot{\psi_t}). \tag{5.16}$$

The angular velocities $\dot{\phi_t}$ and $\dot{\theta_t}$ are approximated by using the difference of angular velocities between the last two consecutive instances. Thus transition equations for $\dot{\phi_t}$ and $\dot{\theta_t}$ can be written as:

$$\dot{\phi_{t+1}} \approx \dot{\phi_t} + \delta_t(\ddot{\phi_t}), \quad where \quad \ddot{\phi_t} \approx \dot{\phi_t} - \dot{\phi_{t-1}} \tag{5.17}$$

and

$$\dot{\theta_{t+1}} \approx \dot{\theta_t} + \delta_t(\ddot{\theta_t}), \quad where \quad \ddot{\theta_t} \approx \dot{\theta_t} - \dot{\theta_{t-1}}. \tag{5.18}$$

Similarly, as yaw angular velocity $\dot{\psi}_t$ is dependent on command input, therefore its state transition can be expressed as:

$$\dot{\psi_{t+1}} \approx \dot{\psi_t} + \delta_t(\kappa_9 \mu_{\dot{\psi},t} - \kappa_{10}\dot{\psi_t}) \tag{5.19}$$

Where κ_9 and κ_{10} are the proportional coefficients for the angular velocity of yaw angle 'ψ'. Similarly by using Equations 5.7, 5.8 and 5.10, state transition equation for $\ddot{x}_t$, $\ddot{y}_t$ and $\ddot{z}_t$ can be given by:

$$\ddot{x_{t+1}} \approx \ddot{x}_t + (\ddot{x}_t - \ddot{x_{t-1}}) \tag{5.20}$$

$$\ddot{y_{t+1}} \approx \ddot{y}_t + (\ddot{y}_t - \ddot{y_{t-1}}) \tag{5.21}$$

$$\ddot{z_{t+1}} \approx \ddot{z}_t + (\ddot{z}_t - \ddot{z_{t-1}}) \tag{5.22}$$

5.3 Field Trial and Results

A lot of field trials have been conducted to determine the accuracy of the proposed EKF based state estimation technique. The trajectory generated has been tested against the different combinations of sensor systems to visualize the accuracy resulting in the fusion of the data.

5.3.1 Trajectory Generation

This approximation works well in practice due to high operational frequency. Normally, we have a very short time interval among x_t and x_{t-1} as GPS is working at 5Hz. Figure 5-1 shows an experiment result with improved GPS accuracy using the proposed post-processing filter. To check the loop closing performance along with accuracy, the experiment is done over a round track of 435 meter circumference. Figure 5-1 shows the accuracy of GPS is improved from multiple meters to sub-meter.

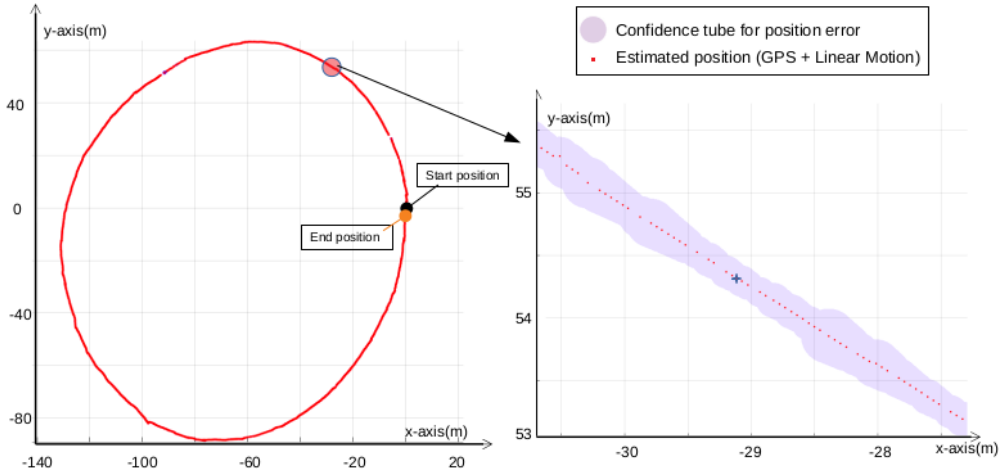


Figure 5-1: *Left* : Plot showing 435m long round trajectory as an output of Kalman Filter by fusing linear process model with GPS readings. *Right* : Shows the zoomed trajectory of the red shaded area with a confidence tube around the predicted trajectory positions.

An Extended Kalman Filter [124] is used with the proposed process model and sensor system models to generate the flight trajectory. Figure 5-2 shows a comparison plot between the trajectory generated by the customized EKF and the trajectory generated by using raw GPS readings. Due to a lack of accuracy in altitude readings of GPS, the comparison trajectory is in the horizontal plane only.

Similarly, Figure 5-3 shows the comparison between *EKF* fused trajectory and a trajec-

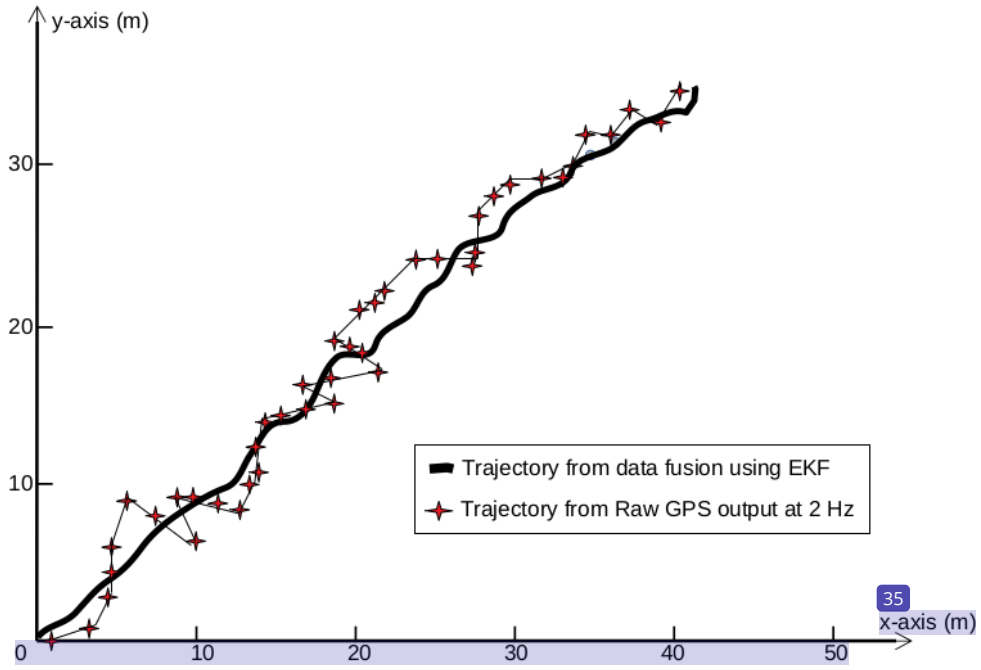


Figure 5-2: Comparison among 2D trajectories generated by EKF and raw GPS readings. Landmarks show the raw reading from GPS at a given instance.

tory generated by the *HeadLiDAR* using a range flow-based approach [98] as discussed earlier. As shown in Figure 5-3, due to the abrupt maneuvers of the octocopter during flight, LiDAR trajectory loses track of features and has to reinitialize occasionally. However, our fusion approach is consistent and robust throughout the trial. Figure 5-4 shows a 3D trajectory comparison with a $z - axis$ showing the altitude of the octocopter. The black line shows the trajectory generated by fusing GPS, IMU, visual odometry and LiDAR odometry with the help of the proposed EKF and the green line shows the trajectory generated by visual odometry using LSD-SLAM [61]. Since the LSD technique uses pixel intensities to track the motion, hence the abrupt motion of the octocopter fails the tracking due to a lack of tracking robustness. In Figure 5-4, the visual trajectory is not continuous and is

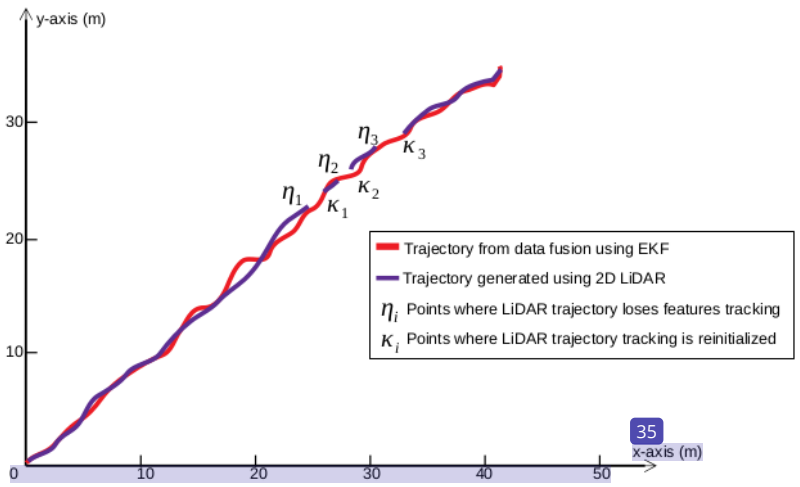


Figure 5-3: Comparison plot for 2D trajectories generated by data fusion using proposed EKF approach against the only 2D-LiDAR based trajectory. Due to the abrupt motion of the multirotor during flight, LiDAR trajectory is not continuous due to loss of features tracking at point ν_i where $i = 1, 2, 3$. It is re-initiated at points κ_i where $i = 1, 2, 3$.

reinitialized occasionally during the flight operation. A long experiment over the length of the water channel has also been conducted to validate the robustness of the proposed approach. For this purpose, a canal stretch is selected on a branch canal near Lahore in Pakistan. The proposed integrated system is mounted over a floating platform with a similar configuration as mounted on the multi-rotor with the height of 2 meters above the surface of the canal water. Considering the aerial platform is quite prone to an accident because of its inherent instability, for the long stretch we selected a floating platform to minimize the risk of the accident against the expensive sensor systems. The result long endurance test over a water channel is shown in Figure 5-5. As represented in Figure 5-5, we have closed the loop with the floating platform by returning to the start point as illustrated by a red star. The total length of the generated trajectory is approximately 3.65 KM. As the water in the canal is flowing at a relatively high velocity, so the trajectory generated is in an irregular line mainly because of forces applied on the floating platform

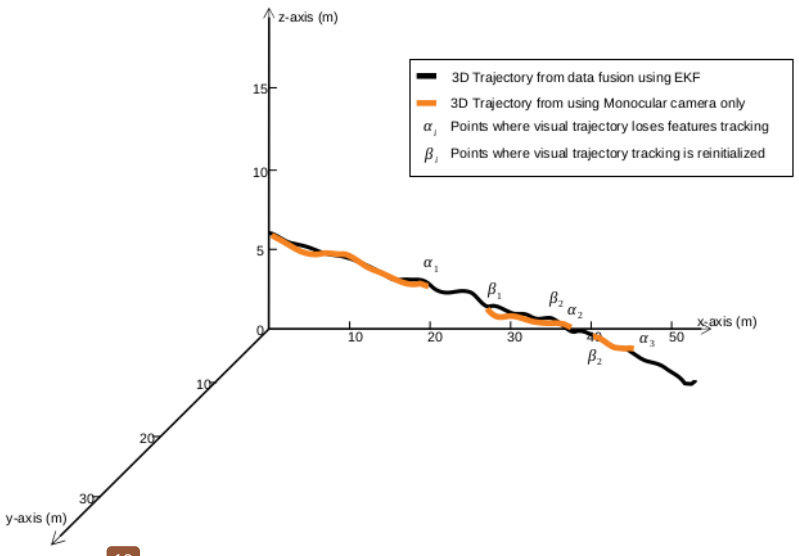


Figure 5-4: A 3D comparison of the proposed EKF approach and the visual trajectory generated by the LSD-SLAM approach. $z-axis$ shows the altitude of the multirotor while $x-axis$ and $y-axis$ represent the horizontal plane. α_1,α_2,α_3 show the loss of visual tracking features due to abrupt aerial maneuvers whereas β_1, β_1 show the points of re-initialization for the LSD-SLAM approach.

from the bouncing currents from the canal bank.

5.3.2 Initial 3D Mapping

Based upon the localization data from trajectories generated at an every time instance, we have generated 3D maps by stitching the sensor data together. The source of stitched data is either from a visual sensor (monocular camera) or a 2D laser scan from the $BottomLiDAR$. Point clouds have been used to visualize the reconstructed environment in 3D after stitching the raw data from the sensor systems. Figure 5-7 shows the zoomed section of a dry waterway from inside showing the bed and the depth of a dry water channel.

Similarly, Figure 5-6 shows the comparison of a particular reconstructed dry water

Figure 5-5: *Top left* : Image showing a globe map of Asia with a red box representing the location of the water channel experiment. *Top middle* : An image showing the zoomed satellite image of the red box in *Top left*. The is a red box represents the closer location of the experiment. *Top right* : A Satellite image representing a further zoom against the red box of *Top middle*. *Bottom* : A satellite view of the part of the branch water channel near Lahore in Pakistan. The yellow line shows the trajectory generated by our proposed method.

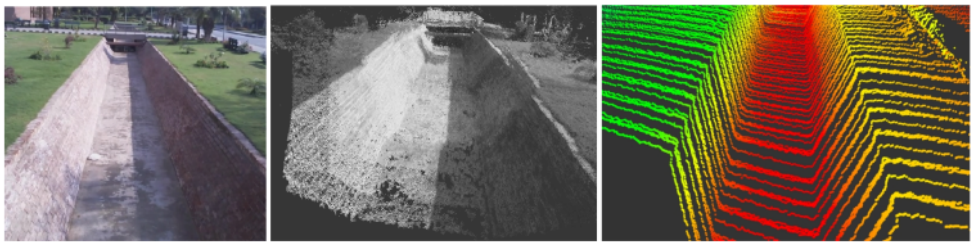


Figure 5-6: *Left* : An RGB image representing a section of a real water canal during a test flight. *Middle* : Shows a 3D point cloud of the same waterway section, generated using Visual trajectory only. *Right* : Shows a detailed map in point cloud generated by *BottomLiDAR* using a fused trajectory by our proposed EKF approach.

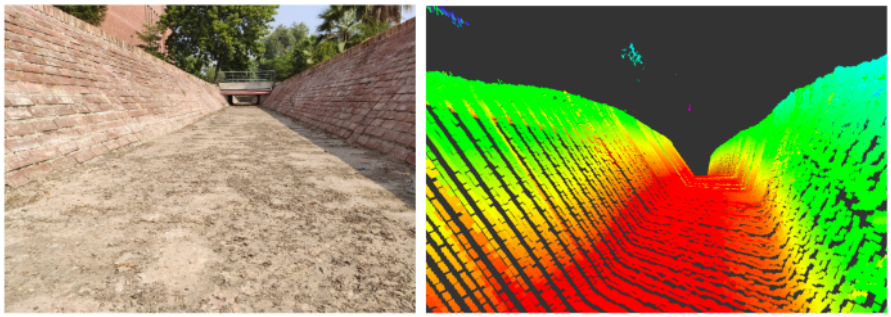


Figure 5-7: *Left* : Inner cross-sectional view of a lined dry water canal. *Right* : Output of the map generated by *Bottom LiDAR* using the trajectory generated by our proposed fusion approach. It shows the cross-section of an empty waterway in a 3D point cloud.

canal section against the real scenario. The channel section is approximately around 50 meters. The middle image shows the reconstruction of the canal structure using visual features only. Moreover, the reconstructed structure using a visual sensor is not much accurate as there is some misalignment in the stitching of the point cloud. Whereas in comparison to this, the reconstructed similar water channel structure using an EKF based fused trajectory and *BottomLiDAR* is more precise and accurate.

Other than the 3D reconstruction of the beds and the banks of the water canal, other physical structures are also of great importance for the irrigation network. These physical

Figure 5-8: *Left* : Image showing a bridge and an associated pipe structure over a waterway. *Right* : Shows the reconstruction of a bridge and pipe structure in a 3D point cloud for inspection.

structures include bridges, pipelines, and other utility infrastructures. Damages to these infrastructures affect the utility benefits associated with these irrigation networks as a whole. The proposed integrated aerial sensor system has the embedded capability to map these supporting infrastructures other than mapping the water channel core structures. Figure 5-8 shows the reconstruction of utility structures such as waterway crossing electricity pipes and a pedestrian bridge.

In the current chapter, we have demonstrated the proposed localization approach along with its technical details adapted to fuse the data in an Extended Kalman Filter. Later, we have reconstructed the water canal environment by using the trajectory generated by our proposed state estimation approach. The next logical step is the autonomous traversing through the length of the canal to reconstruct the canal infrastructure with minimum human interaction. In the next chapter, we have proposed a novel technique to traverse the water canal along its trajectory autonomously.

Chapter 6

Deep CNN based Real-Time Autonomous Water Canal Trajectory Tracking using an Aerial Robot

The structural and hydraulic deterioration of the irrigation networks contributes majorly towards a reduction in the irrigation efficiency of the system. A continuous structural inspection is, therefore, a critical requirement for operations and maintenance. Currently, O&M relies mostly on manual methods resulting in large knowledge gaps and performance loss. Common tasks include sensing or sampling at high-spatial and low-temporal frequency such as seasonal detection of silt deposits and deterioration of banks during flooding; and those that require low-frequency Spatio-temporal sensing & samplings such as monitoring of pollutants and debris, and the prevention of water theft at critical times and locations. Therefore, there is a great potential for economic and social impact on agricultural productivity in such river basins, if large-scale inspection of the waterways can be automated using advanced sensing and manipulation systems carried by mobile agents ([148]).

Due to the linear asymmetric geometry of the water channels, one common requirement would be the ability of an agent to traverse the length of the channel at great distances (on the order of several kilometers) with minimal human assistance. The agent will generally need to traverse the waterways laterally, center its medial position and avoid obstacles. Challenges are also peculiar to whether the agent's mobility is restricted to air, land, or water. In robotics, such problems are related to trajectory tracking and motion planning, relying on feedback provided by sensors. In GPS-based tracking systems, the precision of trajectory tracking depends upon the accuracy of GPS. In the case of waterways, there are occasional occlusions such as trees and bridges which block GPS signals. Monocular or stereo camera-based visual servoing techniques can overcome this limitation but they too suffer from drawbacks such as high computational loads and scene variations due to weather and vegetation.

In the previous works by our research group, we have tackled the traversal problem for canals by a Micro-Aerial Vehicle (MAV) using feature-based computer vision techniques using visual and proximity sensors ([14], [148]). We modified an image processing-based road detection algorithm which has four stages. First, is the generation of dominant feature orientation at every image pixel using a Gabor filter bank. The second is the allocation of confidence scores to all orientation features. Third, an adaptive voting scheme for the detection of the vanishing point in an image; and finally, the detection of canal edges based upon cues given by the vanishing point. Figure 6-1(top row) reproduces some results of canal edge detection using the vanishing point approach. Although the results were promising, the algorithm took an unacceptably long time to process each frame. We investigated another method in which we projected 2D LiDAR beams across the cross-section of the canal and generated its point-cloud representation. From this, the center of the waterway was easily calculated. See Figure 6-1(bottom row). The main disadvantage of this technique was the requirement to equip the aerial vehicle with an expensive sensor and the inability of the algorithm to cope with large deformations of the canal banks due

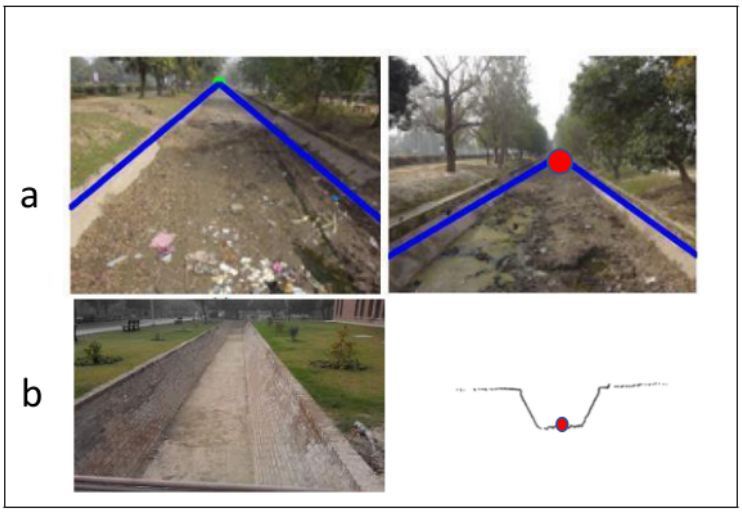


Figure 6-1: Results of earlier research ([148]). (top row) Waterway edge detection for trajectory tracking using vanishing point approach. *Green* and *red* dots show the detected vanishing point and the edges depicted by the *blue* lines. (bottom row) The point-cloud representation of the water canal cross-section using a 2D LIDAR scan. *Red* dot is the computing center.

to erosion or silt accumulation. In this chapter, we have approached this problem using supervised machine learning to develop a technique that is real-time, robust, and does not require complex sensor systems.

6.1 Current Cutting-edge Techniques and Methods

GPS is the most popular sensor for the navigation of Unmanned Aerial Vehicles (UAV) in an outdoor obstacle-free environment such as open agricultural fields. [172] report the use of an open-source autopilot system for this purpose. [179] have used a Kalman filter to fuse the onboard inertial data from a MAV with GPS to calculate the current state. For GPS-denied environments, computer vision-based data fusion techniques have been used to estimate the accurate position and orientation of a MAV. In 2005, [131] have presented a

visual navigation system for a fixed-wing UAV that extracts the color information from an image to classify terrain cover and generate navigation commands. Data from IMU and laser scanner have been fused by [36] for state estimation. [41] have used a laser scanner and monocular vision to generate a 3D environment map and have navigated along the length of the river without a prior map with obstacle avoidance. [33] have reported data fusion techniques in commercially available AR-Drone. [18] have used an RGB-D camera to generate a 3D map of the environment and combine it with visual odometry to navigate quadrotor through a complex cluttered environment. [180] have used different types of MAVs for a practical demonstration of different navigation techniques mainly relying on visual navigation and localization. [177] have presented a novel technique for autonomous MAV navigation in GPS denied environment as they minimize the cost function obtained from visual state estimation and drift-free altitude measurement. [169] have used a monocular camera for metric 3D reconstruction of the environment and have later used that to generate navigation commands to the UAV. In 2015, [80] have used a monocular camera image and extracted the contours of the environment to navigate the MAV through the hallway indoors. [128] have presented a system for autonomously following a target using MAV. [97] have applied an incremental motion model on IMU data to estimate the 3D pose of the drone. Then they have used PID to generate control commands to navigate the drone through the path generated by a generic Simulated Annealing (SA) optimization algorithm. [10] have proposed another novel approach to generate collision-free waypoints by only using a monocular camera. [65] have used an onboard monocular camera to generate a dense environment 3D map for motion planning. In 2017, [139] have used edge distribution to determine optical flow and depth map through stereo disparity. Then they have used the depth map to scale the optical flow and get velocity. Both the depth map and velocity are used to navigate the UAV.

Contemporary techniques of navigating UAVs have started to deploy machine learning-based techniques such as a deep convolutional neural network. These techniques are

broadly categorized as either based on supervised learning or reinforcement learning (RL). Using supervised learning, [174] have proposed a novel approach DAgger Algorithm, to navigation MAV through the cluttered forest environment at a decent velocity of $1.5m/s$. They have used the human expert pilot's demonstration to imitate the control policies to traverse the tree's clattered environment. [187] have also used supervised learning to imitate navigation policies for forest trail traversal. Moreover, they have used another deep neural network (DNN) along with visual odometry to avoid low-level obstacles. [105] have proposed a complex policy learning algorithm for MAVs, which imitates Model Predictive Control (MPC). [83] have also used the deep neural network (DNN) to imitate navigation policies through a forest trail. [79] have proposed a data-driven approach where a DNN is trained by crashing the drone and learning the incident as a false trajectory. [121] have presented deep reinforcement learning to learn generalized policies of more than 20 simulated physics tasks like car driving, catapult swing-up, and legged locomotion. For MAV navigation, a deep reinforced learning model is proposed by [219] to find an indoor target object in a minimum number of steps. It takes the input image from a monocular camera and a target object image and then tries to construct optimized navigation policies.

6.2 Methodology

In this part, we shall discuss details regarding aerial hardware, system architecture, datasets and construct the architecture of a deep convolutional neural network.

6.2.1 Hardware

To evaluate our proposed approach, we have used two commercially off-the-shelve drones. One is called 'AR-Drone' and the other is MK-Octocopter (details in the previous chapter 3). These are lightweight, easily available, and very stable aerial platforms with many sensors on-board. Our primary drone for the research and development of this problem is 'AR-Drone'. It is equipped with IMU, sonar for height measurement, and two

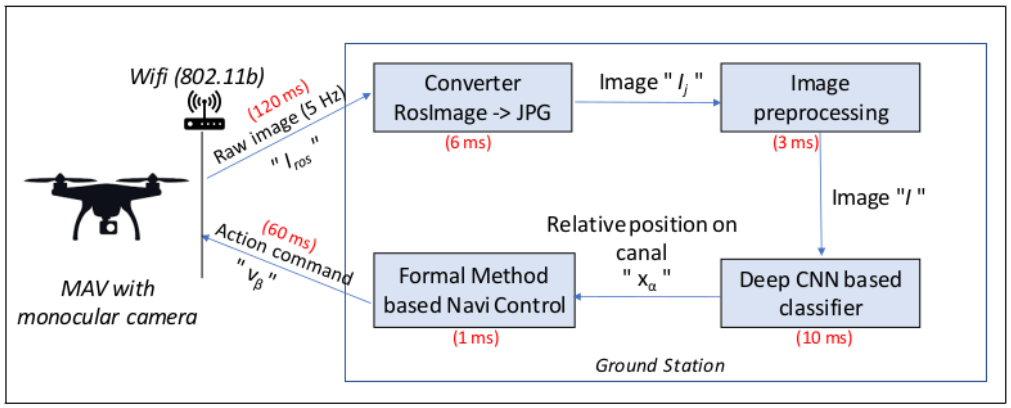


Figure 6-2: The system architecture of the proposed autonomous aerial navigation system. The time required to execute each component is given in red. The total loop closure time from the raw image to velocity command is 200ms.

cameras one in the front with 720p resolution and one in a downward position for motion estimation. However, it has insufficient on-board computational capacity to run DNN based classification algorithms. To overcome this, we use external computational hardware with a Core i5 processor and 4GB RAM linked via WiFi.

6.2.2 System Architecture

Figure 6-2 shows the overall system architecture. We use the Robot Operating System (ROS) as a software framework to interface ground station with MAV via WiFi. Image I_{ros} is captured from an on-board front camera of 640 $\times$ 420 pixels and is transmitted to a base station wirelessly via ROS message encoding. Encoded ROS image message I_{ros} is converted into standard JPG encoding I_j. After changing the image encoding, the image size is reduced to 160 $\times$ 90 pixels with color channels arranged in red, green, and blue (RGB) to generate I. Image I becomes an input to a deep convolutional neural network that identifies the current state x_α of MAV relative to the cross-section of the waterway. Based upon current position x_α, action command velocity v_β is calculated using formal

methods to navigate the MAV along the trajectory of the waterway. Action commands v_β are transmitted back to MAV via a Wifi connection to complete the feedback loop.

6.2.3 Data Sets

The nature of our task prohibits us to collect extensive datasets using other methods. Therefore, we have used a manually controlled MAV to collect the images for the training of our deep neural network. Figure 6-3 shows the relative positions of MAV to collect the dataset above the cross-section of a waterway. The chosen three state positions can be represented as X_α where $\alpha \in \{L, C, R\}$. State position X_R is chosen as the extreme right of the waterway bed, X_L is the extreme left, and X_C represents the center of the waterway.

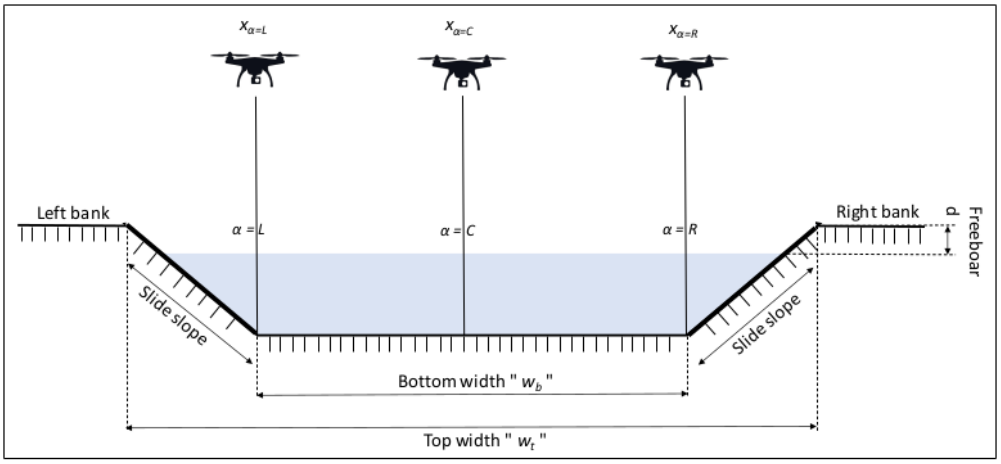


Figure 6-3: Cross-sectional view of the water canal and state positions of MAV relative to the canal for the dataset generation.

At each of the three positions above the waterway, a MAV has been flown for approximately $120m$ at a relatively slow speed, completing the mission in 30 minutes. Images of 640×480 pixel resolution have been collected at 3 frames per second (FPS). Approximately 9500 images have been collected for each of the three positions. Since it is difficult to keep the MAV flying at a desired reference state above the waterway, manual data annotation

has been done to remove undesired images from the dataset. Following the process of annotation, the dataset is finalized with 29,282 images. The dataset has been split into two disjoint sets: 26282 images for the training and 3000 images for testing.

It is important to clarify that this manual method of data collection and annotation is not scalable to generate a representative data-set covering all situations arising in thousands of kilometers of channels. However, the purpose here is to demonstrate a general methodology, if and when such a data set becomes available by some other method. We are looking into the use of realistic simulation engines to generate data and other techniques to address this problem in our future work.

6.2.4 Deep Residual Convolutional Neural Network

Figure 6-4 shows the detailed layers of our deep residual convolutional neural network. This deep CNN is inspired by ResNet-50 ([88]). The initial input to our CNN is an image of size $(I_h, I_w, I_c) = (160, 90, 3)$ pixels with 3 color channels in order of red, green, and blue (RGB). Then zero-padding of 3 is applied to both sides of the image both vertically and horizontally. Next, convolution is applied with a filter size of $(f_H, f_W, f_n) = (7, 7, 3)$, stride $s = 2$, batch size $m = 16$ and number of filters $n^f = 64$. After applying convolution, it gives us the output of $80 \times 45 \times 64$. During convolution, bias b^i is added after each complete convolution operation followed by non-linear activation function ReLU ([116]). Since we apply the convolution operation in batches, we need to batch normalize keep the mean activation at zero and standard deviation of 1. After applying batch normalization, the dimensions remain the same and the output is called activation at level one, denoted by $a^{[1]}$. Formally we can write this process as:

$$\begin{aligned} z^{[i]} &= w^{[i]} \circledast a^{[i-1]} + b^{[i]}, \\ a^{[i]} &= g(z^{[i]}), \end{aligned}$$

where $i = 0, 1, 2, ...n$, $b^{[i]}$ is the bias added after current convolution operation, $w^{[i]}$ represents the weights (filters) and $a^{[i-1]}$ and $a^{[i]}$ represents the activation layers of previous and current iterations respectively. g is the activation function $ReLu$ ([116]).

After obtaining activation $a^{[1]}$, layer 2 of the deep neural network starts with max pooling ([116]) using a filter size of $f_{max}(3,3)$ and $s = 2$ resulting in dimensions $39 \times 22 \times 64$. Again, convolution is applied at layer 2 with filter size of $(f_H, f_W, f_n) = (3,3,64)$ and $s = 1$. In order to diversify the detected features, 64 filters are again applied at this layer. Since we used zero padding of 2, the overall dimensions remains same after convolution. After convolution with each filter f, a bias $b^{[2]}$ is added and a non-liner activation function ReLu g is applied with batch normalization operation to get the activation $a^{[2]}$ for layer 2.

Layer 3 is slightly different from layers 1 and 2. In this layer, we do not apply max pooling operation. Instead, first convolution operation is applied with a filter size of $(f_H, f_W, f_n) = (3,3,64)$, with $n^f = 64$, $s = 1$ and padding. Bias $b^{[3]}$ is added after convolution against each filter and ReLu is applied to get the activation $a^{[3]}$. Once again, convolution is applied with the same filter, stride and zero padding but this time for $f^n = 256$.

An addition operation is applied to add the output of two activation layers i.e. $a^{[3]}$ and $a^{[r1]}$ as follows. Activation output $a^{[r1]}$ is generated via a shortcut patch in the deep neural network by taking the activation $a^{[1]}$ of layer 1 as an input to a convolutional operation with number of filter n^f equal 256, filter size of 3, stride of 1 and zero padding of 2. After applying ReLu and batch normalization we get the activation $a^{[r1]}$ of dimensions $(39 \times 22 \times 256)$. Since both $a^{[3]}$ and $a^{[r1]}$ are of same dimensions, we can apply the addition operation to get the activation $a^{[4]}$ for layer 4. This shortcut embeds the residual network properties in our network. The fundamental benefit of ResNet is that it keeps the loss function maximum even after adding multiple layers which may minimize the loss function. As a result, our network remains optimal.

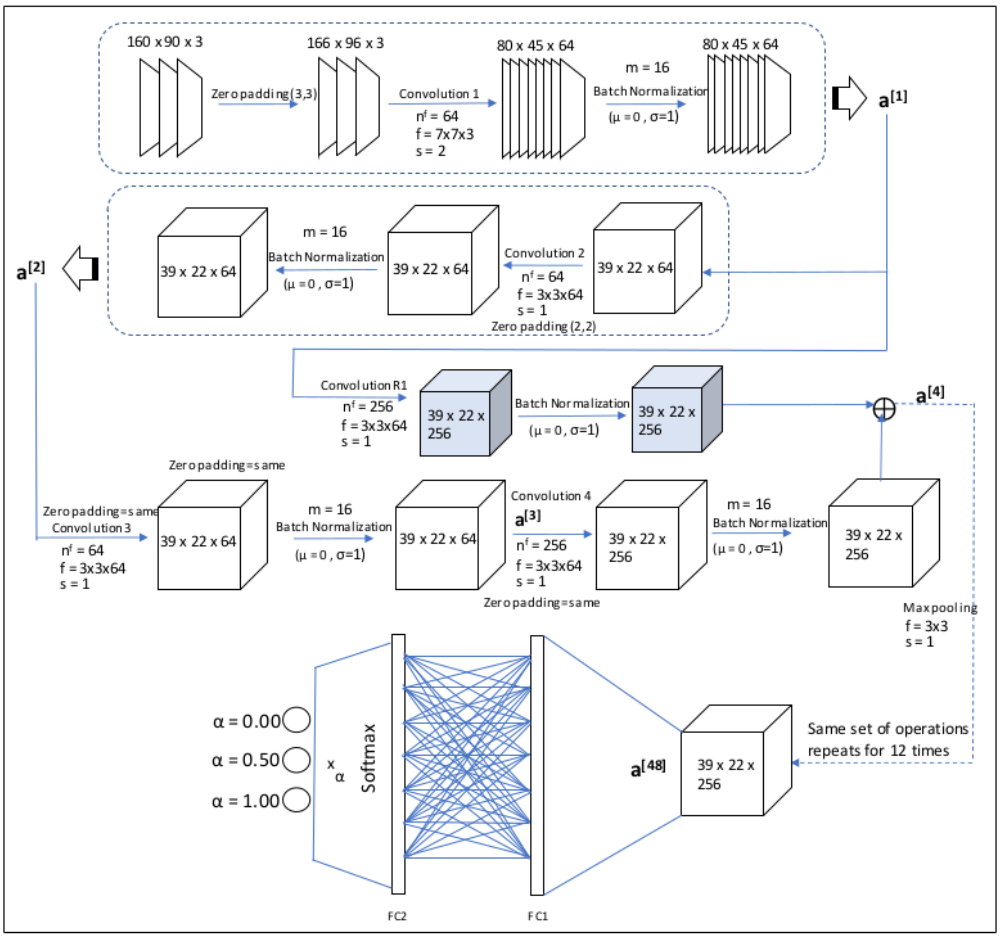


Figure 6-4: Detailed operations and configurations for the construction of Residual 50 layers of a convolution neural network.

In our proposed deep convolutional neural network, the above-mentioned set of operations is repeated 12 times to complete 48 layers. The shortcut connections are also repeated along with the other layers to make it a complete residual network. After generating the activation $a^{[48]}$ and batch normalizing it, the dimension becomes $(39 \times 22 \times 256)$. Following this are the fully connected layers $FC1$ and $FC2$, which have $219,648$ fully connected nodes.

Finally, by applying Softmax ([28]), our network classifies each class with a probability.

6.2.5 MAV controller design

We have used formal methods to define a policy to generate action commands based on the MAV's position estimated by CNN. One can imagine the three-position states as analogous to three lanes laid over the waterway i.e. left, center and right. To keep the MAV over the waterway, we desire to keep it in the central lane. Its motion capabilities can be modelled as a transition system

$$\mathcal{T} = (\mathcal{S}, s_{init}, \mathcal{R}, AP, \mathcal{L})$$

where $\mathcal{S}$ is the state space, s_{init} is the initial state, $\mathcal{R}$ is the transition relation, AP is the set of atomic propositions and $\mathcal{L}$ is a labeling function which maps states of the transition system to the power set of AP ([197], [19]). Each state of the transition system is composed of two components:

1. the lane X_α, which MAV occupies over the width of the waterway;
2. the action V_β, which MAV has taken to reach the current lane.

Therefore we have a total of 9 states described as:

$$\mathcal{S} = \{(X_\alpha, V_\beta)\,|\,\alpha \in \{L, C, R\}\,,\ \beta \in \{L, F, R\}\}$$

where $\alpha = \{L, C, R\}$ denotes the left, center and right lanes and $\beta = \{L, F, R\}$ denote velocity commands for left, forward and right respectively. Some states are not reachable e.g. the MAV cannot reach the left lane after executing a velocity command towards the right. Therefore, we eliminate two states, (X_L, V_R), (X_R, V_L) from our transition system. Figure 6-5 shows how state-to-action command mapping would be performed based upon a defined set of rules.

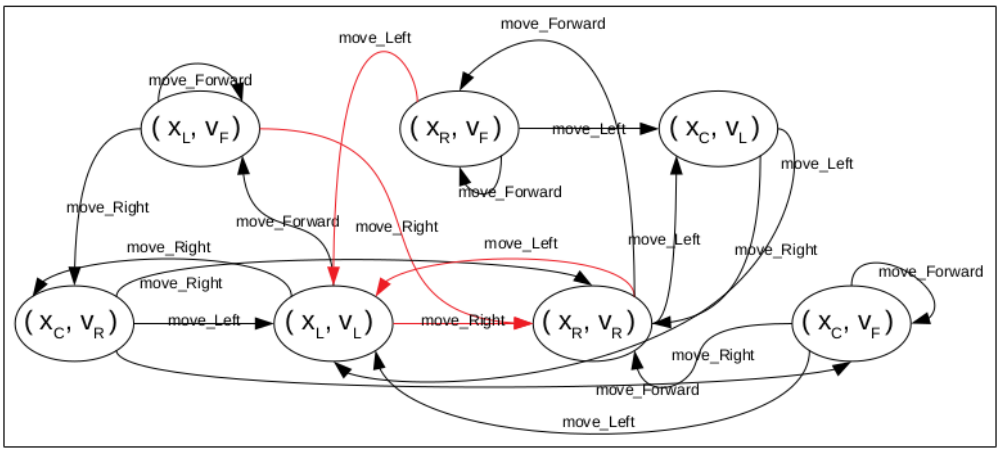


Figure 6-5: Transition Graph shows all sets of actions v_β against all the states x_α. The Black link shows the possible connection and the red shows impossible connections.

Our set of atomic propositions are defined as,

$$AP = [lane\_x, move\_y]$$

where, $x \in \{Left, Center, Right\}$ and $y \in \{Left, Forward, Right\}$. Since our MAV can not directly move from left lane to right lane without entering into the center lane and vice versa so this rule can formally be defined in Linear Temporal Logic as follows:

- MAV can not go from left lane to right lane by executing action command towards right.

$$\phi_1 = G(\neg(lane\_Left \wedge X(lane\_Right)))$$

- Similarly, it can not go from right lane to left lane by executing action command towards left.

$$\phi_2 = G(\neg(lane\_Right \wedge X(lane\_Left)))$$

The commands synthesized by this controller are passed on to the low-level control unit for trajectory generation and reference tracking.

6.3 Experimental Results

To evaluate the real-time performance of our proposed waterway tracking approach, we flew the MAV over the length of the canal with command velocities generated by our proposed system.

6.3.1 Outdoor Experiments on Real Waterways

One of the biggest challenges of an outdoor environment is the uncertainty in flight dynamics due to external factors such as wind. Internally, decrements in the battery's state of charge can also destabilize the flight. To address these challenges, a very robust navigation system is required to follow the trajectory of a waterway. For outdoor experiments, we assume a fixed initial position X_start with a similar height as the training dataset. Since we do not have a recovery mechanism in our system, the MAV must start at one of the three training positions i.e. left side, center, or right side of the waterway. We pick this to be the center position. Once the MAV starts to hover over the center position of the canal, autonomous navigation control is turned on. Once the autonomous process starts, MAV starts using a deep CNN-based network for classifying the current state x_α of MAV, based on which the next velocity command $\vec{v_\beta}$ is generated. Figure 6-6 shows the outcome of outdoor experiments. Each figure shows a frame on which our proposed approach predicts the current position over the waterway bed which translates into command velocity shown by green arrows. Inside the images, the probability graph is shown to depict the output of the proposed CNN against which velocity command is chosen.

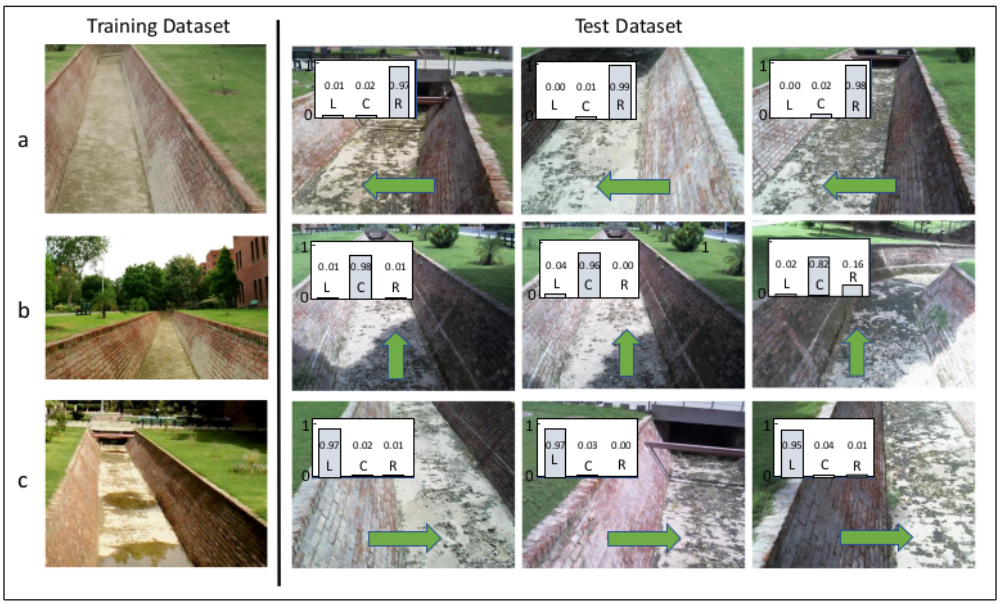


Figure 6-6: Results showing probabilities of winning class against selected frames from an outdoor experiment, tracking a waterway. (a) Shows the results of frames for which x_R is classified. (b) Shows the results of frames classified as center position x_C (c) Shows the frames where MAV is at the left side of waterway bed x_L.

6.3.2 Result Analysis

Figure 6-7 shows the classification performance of our proposed system against the test dataset for which the true values are known. The results show that the proposed system is quite accurate in term of classifications.

Table 6.1 shows the comparison of ResNet50 with other state-of-the-art convolutional neural networks trained on our waterways dataset. VGG16 ([185]) and VGG19 ([185]), both network models fail to train the dataset as training and testing accuracy remains below 35 percent. Training and Testing losses of both the networks are very large as the network model fails to minimize the loss function. Xception ([49]) and MobileNet ([96]) networks train well on the dataset with good accuracy of 99 percent for both training and testing.

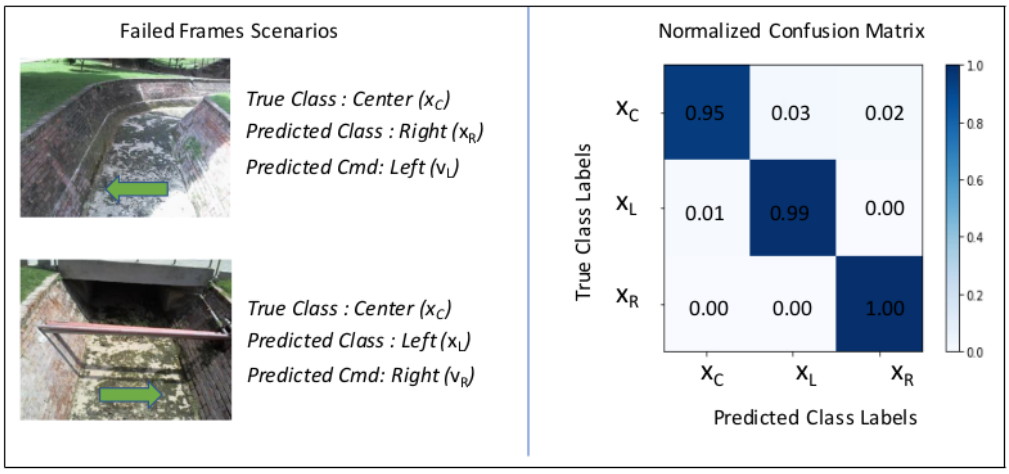


Figure 6-7: Confusion matrix along with failed scenarios.

Their training and validation losses are also small enough to show that both models learn nicely to keep the loss function minimized. Best results are from ResNet50 and InceptionV3 ([192]) as both network model achieve 100 percent accuracy in training and validation. Losses of both the networks are even smaller than Xception and MobileNet, which shows the perfect minimization of the loss function. But the classification time for a single image frame using ResNet50 is 8*ms* faster than InceptionV3, which makes it the best choice for our proposed system.

Table 6.1: A comparison of used Covolutional Neural Network(CNN) approach with the other state of the art CNNs

Comparison with other state of the art CNNs					
Model	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss	Execution time per frame (ms)
VGG16	0.35	0.34	10.5	10.8	8
VGG19	0.30	0.335	11.5	10.72	12
Xception	0.99	1.00	0.0176	0.0003	11
ResNet50	1.00	1.00	0.000016	0.00007	10
MobileNet	0.9984	1.00	0.0051	0.0054	7
InceptionV3	1.00	1.00	0.00002	0.00008	18

Part IV

Further Applications for Efficient Canal Structural Monitoring

Chapter 7

Single Camera Terrain Classification for Water Canals

7.1 Introduction

In this section of the thesis, we have illustrated two additional applications which can further enhance the water canal structural monitoring. The addition applications described can be among the many other applications which can be developed using the already proposed aerial autonomous canal monitoring system. One of the explored applications proposed in this chapter is the identification and classification of the terrain of the dried water canal. The classification helps us to determine the exact nature of the canal bed terrain which affects directly the flow and discharge of water from that part of the canal. In this chapter, we have identified different types of terrains along the length of the water canal which can lately be fused with the trajectory of the aerial robot to localize the identified terrains. Furthermore, hydrologists can use the classified terrain details (both type and location) to exactly estimate the flow of water discharge at that part of the canal. To calculate the open channel discharge flow of the waterways, hydrologists commonly use

Manning Equation [15] as below:

$$Q = VA = (1.00/n)\, AR^{2/3}\sqrt{S} \tag{7.1}$$

where Q is the discharge rate in ft^3/s, V is the velocity in ft/s, A is the area of flow in ft^2, n is the Manning's roughness coefficient, S is the water canal slope in ft and R is the hydraulic radius in ft. In Equation 7.1, every parameter can be determined accurately except the n which represents the roughness co-efficient of the canal bed. In other words, the flow rate of the water canal effects greatly by the change in the roughness of the bed along its length. Commonly, researchers use a constant approximation of roughness to calculate the discharge rate of the channel at a given location. However, this approximation results in large errors as the initial flow rate error get accumulated in further readings downstream as well. If we determine the type of terrain in the canals, we can significantly improve the accuracy of flow rate estimations.

Terrain classification is done using different methods such as by using color feature information, texture features of the terrain, and using depth information. In this chapter, we aim to do terrain classification using a single camera, in line with our overall goal of developing a very cost-effective outdoor robot for off-road applications [3]. Considering we have a limited processing power on-board our aerial platform, we prefer to use the computer vision technique instead of using a deep Convolution Neural Network(CNN) for the classification of the canal bed terrain.

In this chapter, we have proposed an approach for terrain classification which uses the best of both worlds i.e. both the color features and the texture features to decide the winning class for identification. Our idea behind this is some images are very rich in colors and it is wise to classify them for the color features as shown in Figure 7-1. Similarly, some images are high in texture so they should be classified with texture-based features. Moreover, the same terrain with different lighting conditions and viewing angles of the camera can make it vary in color features or texture features. So instead of dealing with all

the images with the same approach, we have developed a system that suitably combines the classification of both texture and color-based approaches. Experiments have shown an overall increase in accuracy by combining the two classification approaches.

Figure 7-1: Sample images: *Left* An image rich in color. *Right* An image contains high number of features.

In the next part of this chapter, we have presented the current state-of-the-art approaches for terrain classification.

7.2 Related Work

As outdoor robotics is evolving, more and more challenging tasks are being assigned to robots [24]. One of the most challenging tasks in an outdoor robotics application is to classify the terrain on which the robot is moving. An autonomous outdoor ground robot needs to know each feature around it to execute a complex task. By using only onboard sensors, we cannot identify the environment fully. For indoor environments, normally we have smooth planar floors and walls. But in an outdoor environment, we can have different types of terrains built both by nature and humans. Intelligent selection of a traversable terrain during the execution of a particular task guarantees success. Selecting a terrain can be helpful for a robot in avoiding a fail-state or even total loss. Sometimes the nature of the task itself demands the classification of terrain e.g. autonomous driving, autonomous land

mines sweeping and planetary explorations, etc. In these applications, the classification plays a direct role in a successful completion.

Terrain classification started in the early 70s for satellite imagery mainly for military purposes. Later it has been used in civilian applications such as estimating total cultivated land, water resources, flood damage, etc. by using satellite imagery. As robotics evolved from simple predefined tasks to complex uncertain operations, many researchers have investigated the problem of terrain classification, mainly for wheeled autonomous robots. This domain gained huge momentum after the Mars Exploration mission and DARPA Grand Challenge in 2004 [196]. LIDARS and depth cameras are generally used to detect the obstacles in an outdoor/indoor environment for autonomous navigation. But classifying the cover of the terrain on which the robot is traversing is not possible with depth information only.

J.F. Lalonde et al. [113] presented a terrain classification approach using 3D LIDAR. They segmented the laser scanner data using local 3-D point statistics into three different classes: clutter to capture grass and tree canopy, linear to capture thin objects like wires or tree branches, and finally surface to capture solid objects like ground terrain surface, rocks, or tree trunks. Though they manage to classify into three different classes still each class has its complexity to be dealt with. And without visual information it is impossible.

Most of the research under this domain concerns sensor fusion between stereo cameras [47], LIDARS, and other range sensors to classify type and slope of the terrain [132]. Initially, terrain classification was limited to certain easily differentiated classes such as sky, green (for grass and trees), and ground (for off-road and roads). Subsequently, it required more complexity such as the ability to distinguish between visually similar terrains like mud, gravel, and sand. In the past decade, a lot of work has been done on color-based classification. Lately, the focus has shifted to scale-invariant feature recognition and classification because of the advancement in computer vision. With efficient algorithms like SIFT [127] and SURF [20]; and super-fast hardware, we can easily extract different

scale-invariant features from different images of varying lighting conditions, size, and viewing angle in real-time.

Jansen et al. [99] presented a color-based terrain classification method assuming that the same environment has similar color-space pixel distribution. The paper has used a mixture of Gaussian models. On top of it, a maximum likelihood classifier is used.

Manduchi et al. [133] and Bellutta et al [22] also have used color features to classify the terrain. They classify green vegetation, dry vegetation, and rock. A maximum likelihood classifier has been used based on a mixture of Gaussian models. To minimize the environmental illumination, they have used images of a large variety of terrains in different environmental conditions to train the classifier. Their results show that color features can classify the terrain cover very efficiently.

Castano et al. [40] have used texture-based features to classify an image. They have carried out a performance analysis of texture-based terrain classification methods using different real-world images relevant to off-road autonomous navigation. Features have been extracted by Gabor filter banks and then two different classifiers have been trained on the features extracted for classification. The first classifier models feature distributions for each texture class using a mixture of Gaussians with a maximum likelihood classifier. The second classifier represents local statistics using marginal histograms of the features over a region centered on the pixel to be classified. The results they showed are promising but need a significant improvement in accuracy and processing time for today's robotics applications.

Manduchi et al. [133] presented a method of terrain classification which uses both color and texture features of the image. They fused the color and texture features with a Cartesian product. It results in the creation of a large number of classes which is later reduced by a class reduction method. Their method improves the overall quality of color-based supervised classification by fusing unsupervised texture-based segmentation with it. In 2003, Shi and Manduchi [182] did a study regarding the statistical fusion of color-

based and texture-based features to get better accuracy. They attempted to assess the trade-off between these two factors through experimental tests on two image data sets using color and texture features. Their results suggest that the difference between the correct classification rates using Bayes fusion and using the joint feature distribution is a function of the conditional dependence of the features (measured in terms of mutual information). However: for a small training dataset size, Bayes fusion performs almost as well as joint distribution. Hence due to better and more accurate results, texture features for classification is the most popular approach in general outdoor robotics.

In 2018, Zhang et al. [215] have used a 3D Convolutional Neural Network (CNN) to classify the polarimetric SAR data for the identification of terrains. In 2022, Wang et al. [204] have used CNN and SVM to present a hybrid terrain classification technique for the mobile robot. They have used RGB images of six different kinds of terrains to train both the CNN and SVM and by fusing the classification of both techniques, they have achieved high accuracy.

7.3 Methodology

We have used a supervised learning framework and the sequence of operations can be described as

$$\underbrace{I \in \mathrm{R}^{m \times n}}_{\text{Image}} \rightarrow \underbrace{v \in \mathrm{R}^{m}}_{\text{Vector description of I}} \rightarrow \underbrace{f(v) = \{C_i, T_i\}}_{\text{Classifier function}},$$

where image I is processed to get a vector v which describes the important information relative to classification in an image. Then this vector becomes the input to a classifier function $f()$ which gives us two results: the class index C_i and the confidence level T_i with which the class is selected. The framework consists of two steps.

- A model is learned using labeled training data to train a classifier for an off-line environment. So the trained model is highly dependent upon the training data. We have used five different classes of data to train our classifier model.
- The model is tested using an unseen test to assess the accuracy of the trained model. The new unseen data should also be labeled to get accurate statistics.

After training the classifier, the problem of over-fitting is avoided by using different test datasets containing similar types of terrains.

Refer to Figure 7-2, we use two approaches i.e. texture-based and color-based, in parallel for an input image. After running both approaches in parallel, we calculate the confidence level with which each approach classifies the terrain. Once we have the confidence level of each approach in the form of probabilities, we have applied Geometric Average Rule (GAR) [194] to combine both texture-based and color-based classifiers. After combining the classifiers the class with the greatest confidence level is declared as a winner or the classified class.

In our scenario, the probability of having multiple terrains in an image is low as our camera is pointed towards the ground. But still, during terrain traversal, some images might contain multiple terrains. For multiple terrains in an image, the terrain with the highest confidence level shall be labeled as the identified terrain type. Detail for each of the steps is as follows:

7.3.1 Texture Features Based Classification

A texture feature is an interest point on an image which is described as a key point and a description. The key point contains information regarding the orientation, pixel coordinates, etc. The descriptor is a real value that uniquely illustrates the local neighborhood to the key point. A lot of work has been done in texture-based classification. We have found that terrain classification using a visual bag of words with some classifier, is the

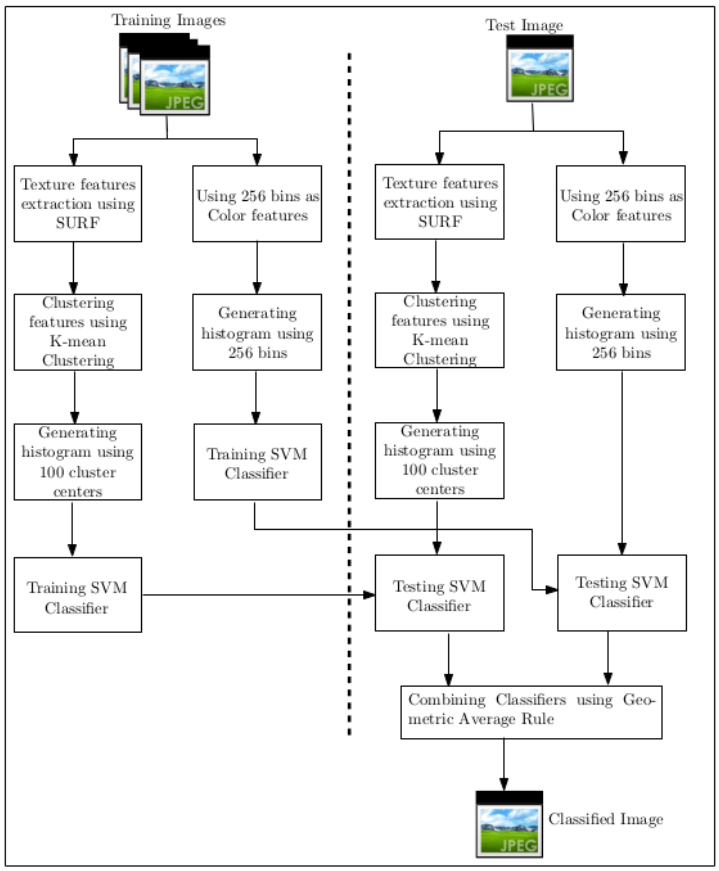


Figure 7-2: Our novel approach for the classification of the terrain for an input image.

state-of-the-art approach. Thus, our approach is inspired by [71] for the classification of an image using the features present in it.

For texture feature generation, there are two mostly used algorithms [44][150][199][159], Scale Invariant Feature Transform (SIFT) [127] and Speeded-up Robust Features SURF [20]. SURF is a better choice as it is faster and slightly more accurate [20]. Therefore, we have used SURF for the feature extraction from an image based on the texture. SURF relies on the FAST-Hessian detector [20] for detecting key points like corners, blobs, and T-junctions at unique locations in the image. Afterward, it uses the Haar wavelet response in the horizontal and vertical directions to compute a real value that uniquely describes the neighborhood around each key point. Figure 7-3 shows texture features being extracted after applying the SURF-based extractor.

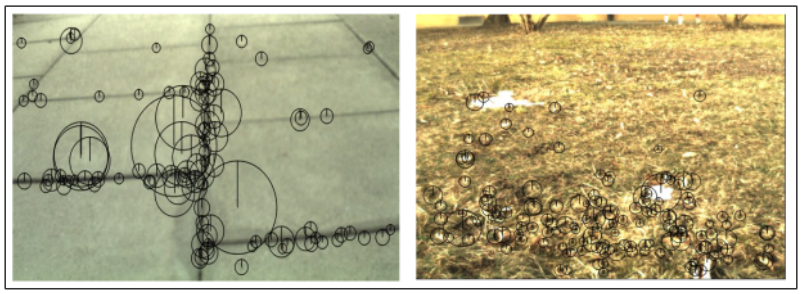


Figure 7-3: $Left$: An image showing the extracted features against an input image of the large pavement Tiles. $Right$: An image showing the extracted features from an input image of grass.

In SURF, the number of extracted interest points (keypoints and corresponding feature vectors) is directly related to the threshold value. If this value is too small then a lot of key points shall be extracted, thereby overloading the system. Similarly, a large threshold value will result in an unacceptably low number of key points. So, the number of extracted key points should lie somewhere in between the two extremes for the best results. (See Figure 7-4). To keep the number of extracted interest points in an acceptable range, we have used a Gradient Descent Threshold Adjuster [70].

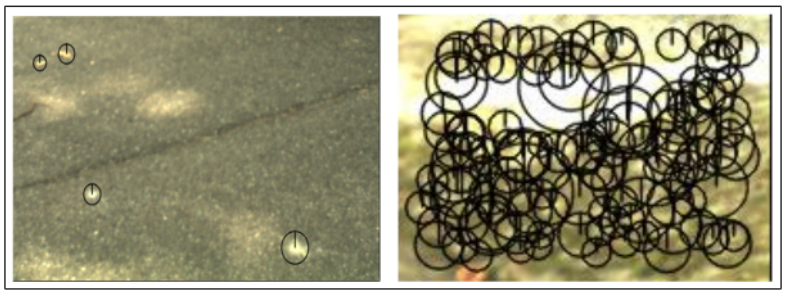


Figure 7-4: High variation in the extraction of interest points count using SURF with the same Hessian threshold.

Once the texture features have been extracted, we need to make the vocabulary from all the extracted features to build a histogram. For this purpose, an iterative K-Mean Clustering (Lloyd's algorithm/K-mean++) is used for 15 iterations [17]. Once the texture features have been extracted, each image can be represented by a word of frequency vector q. Every center of the cluster becomes the vocabulary word. This process approximates each descriptor to the nearest vocabulary word concerning Euclidean distance and then counts them.

The histogram thus created becomes the input to a classifier. We use a Support Vector Machine (SVM) for the classification. The goal of SVM is to find hyperplanes to give us the maximum separation between the classes for the training data. Once the SVM classifier is trained, the intersections are used to define the boundaries of the classes in the classifier. We used OpenCV $CvSVM$ [207] with the configuration of C SVC, POLY, 2, 0.5 for type, kernel, degree and gamma respectively.

7.3.2 Color Features Based Classification

After training for the texture-based classification, we train the system for colors with the same training dataset. Color-based classification is fairly straightforward. We have simply used the different colors for the vocabulary and can create the color histogram

from it. For our system, we have considered RGB color space and have used 256 bins for each color channel to create three separate histograms against each image as shown in Figure 7-5. Once the histograms based on color channels are created, we have used

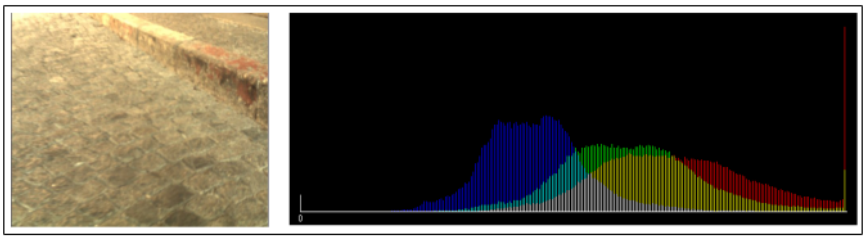


Figure 7-5: *Left* : An input raw image. *Right* : A Color histogram is generated using the RGB channels.

another SVM classifier with the same configuration as mentioned above to classify the input image again. We have used the same classifier because it gives us a nice comparison between the accuracy of the two approaches. Our experiments have shown that overall the texture-based classification is overall more accurate than the color-based classification.

7.3.3 Combining Classifications

After comparing the results of both approaches, we have noticed that some images that are rich in color and low in texture are wrongly classified by the texture-based approach and vice versa. So we devised an approach that combines both classification approaches (Color-based and Texture-based), by removing the deficiency of one approach with the help of the other. After combining, the class with more confidence level would have a high probability of correctness. For combining descriptions, we used the Geometric Average Rule [194].

7.4 Accuracy & Robustness Validation

7.4.1 DataSet

For our validation of the accuracy of our proposed approach, we have used sample off-line experiments and have used the dataset [109] with 500 images. The images are taken from a simple webcam. All the images are 480 pixels by 680 pixels. The images are of five different classes namely *asphalt*, *bigtiles*, *smalltiles*, *grass* and *gravel*. The dataset is

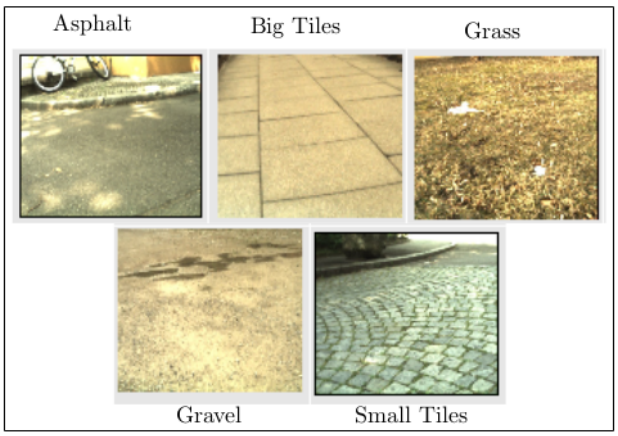


Figure 7-6: Five classes of the dataset have been used for verification of our proposed approach.

deliberately designed to be very challenging as it contains a lot of variation in terms of light intensities, viewing angle, and appearance. The goal was to include all the uncertainties that can occur while the robot is driving outdoors. For instance, *bigtiles* and *smalltiles* classes are very similar in nature and appearance. In *asphalt*, images were taken at different daylighting conditions and shadows along with the different marks or lines painted on it e.g. road signs. *Grass* class contains different grass textures with weeds and bushes. See Figure 7-7 for some samples. For the training dataset, we have used 211 images and for the testing dataset, we have used 289 images.

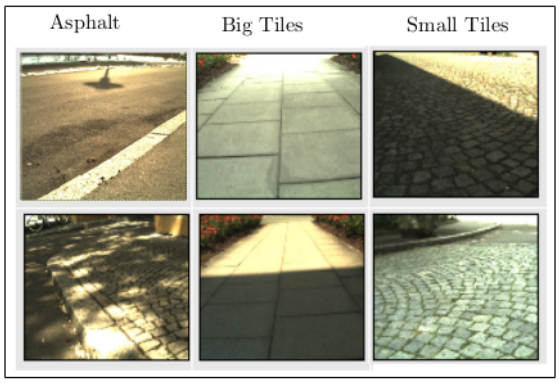


Figure 7-7: Sample images of Asphalt, Big Tiles, and Small Tiles under different lighting conditions with shadows to increase the classification complexity.

7.4.2 Validation Results

After training the system with the help of the training dataset, we tested our algorithm against the test dataset. Our results have proved that our combined classifier approach is quite efficient both in terms of classification speed and accuracy. Result details are shown in the Table 7.1. In Table 7.1, the first column shows the type of the classes, the second

Table 7.1: Result analysis of our improved approach.

Class	Total images	True positive	Accuracy%	Average time
Asphalt	55	51	97.23	20 msec
Big Tiles	40	38	99.30	19 msec
Grass	32	31	99.30	20 msec
Gravel	28	24	97.92	20 msec
Small Tiles	134	133	97.92	17 msec
Overall	289	277	98.33	19 msec

column shows the total number of images for that particular class in the test dataset. The third column shows true positive that is the number of images truly classified. And the

fourth column shows the accuracy of the algorithms for that particular class which is being calculated by:

$$\text{Accuracy} = \frac{\text{True positives + True negatives}}{\text{Total number of images}}.$$

The last column shows the average processing time for an image. Note that this time does not include the training time of the classifier. We ran these experiments on a core 2 processor with 2 GB of RAM. To evaluate the results without any bias, we also created a

Table 7.2: Confusion Matrix of our Improved Approach

		Predicted Class				
		Asphalt	BigTiles	Grass	Gravel	SmallTiles
ActualClass	Asphalt	51	0	0	2	2
	BigTiles	1	38	0	0	1
	Grass	1	0	31	0	0
	Gravel	2	0	0	24	2
	SmallTiles	0	0	1	0	133

confusion matrix based upon the classified images from our improved approach, see Table 7.2. The diagonal of the table shows all the correct predictions, other than the diagonal, all are errors in classification, so it is quite easy to check the table for errors, as they will be represented by any non-zero values outside the diagonal. To further illustrate the results, the confusion matrix is represented in a form of true/false positive/negative table in Figure 7-8.

7.4.3 Comparison with each Individual Approach

To further prove the results, we have compared our algorithm with stand-alone texture-based and color-based classifications. The same dataset has been used to keep the challenges constant. For the texture-based classification, we have used the algorithm [71]. The results are shown in Table 7.3. Results show that our algorithm outperformed the state-of-the-art texture features-based approach in all classes. This is because the images which

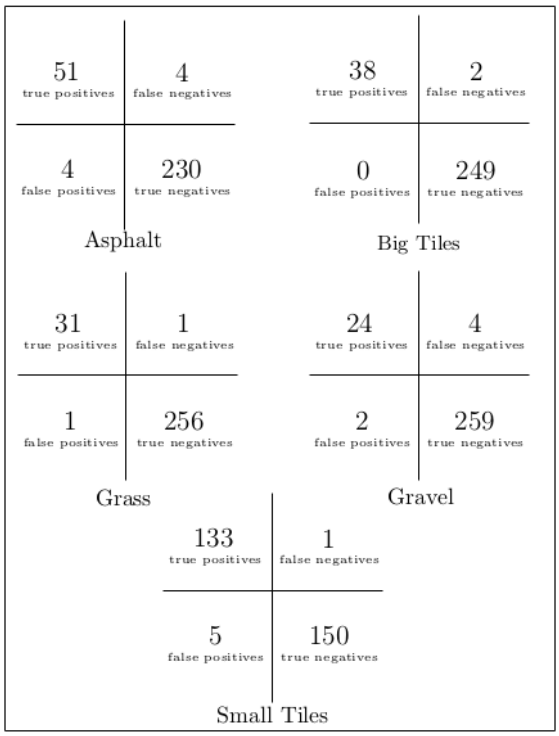


Figure 7-8: Confusion matrix for combining classifiers approach.

are rich in colors should include more weight from the color channels while classifying. Color-rich images are those images that can better be described for the 256 bins of the RGB channels. Whereas texture-rich images have more corners, T-junctions and blobs can be described better by the texture-based feature points rather than color intensities.

For the color-based classification, we have used the standard approach of using three color histograms of 256 bins for the RGB color space. The results are shown in Table 7.4. Results show that the color-based approach is better than the above-mentioned texture-based approach as far as the grass and gravel classes are concerned because both of them are rich in colors. But our combining classifier approach is far better in results as compared to color based approach as well. The noticeable point is that the color-based approach took

Table 7.3: Result analysis of Texture Based Approach.

Class	Total images	True +	Accuracy%	average time
Asphalt	55	50	93.73	16 msec
Big Tiles	40	38	96.19	15 msec
Grass	32	27	90.65	17 msec
Gravel	28	20	82.54	16 msec
Small Tiles	134	131	97.42	17 msec
Overall	289	266	92.10	17 msec

Table 7.4: Result analysis of Color Based Approach

Class	Total images	True +	Accuracy%	Average time
Asphalt	55	48	90.45	5 msec
Big Tiles	40	32	85.06	5 msec
Grass	32	32	100.00	4 msec
Gravel	28	22	89.79	6 msec
Small Tiles	134	129	89.14	6 msec
Overall	289	263	90.89	5 msec

very less time as compared to the texture-based approach.

7.5 Adapting towards Classification of Dry Water Canals

Algorithms related to computer vision are normally very time-consuming due to high processing requirements. Applying them to aerial robots which have limited resources is quite a challenging task. We have implemented the proposed algorithm in such a way that the classification response time should be predictably quick. We have tested our proposed technique on our aerial platform with the classification algorithm running on the onboard processing unit in real-time. We have used 'Odroid UX4' as our embedded processing unit attached with the 'Logitech C500' webcam serially. The drone has been flown over

the length of the canal manually with the help of a human pilot. We have done the canal bed classification experiments on different canals with different sized to include as much variation as possible. The video feed was processed every second by printing the class name on each acquired image.

Earlier, we have trained our classifier with a dataset containing urban environment terrains that are different from the ones we have for the dried water canal beds. Therefore, we collected a new dataset for training and testing purposes. As earlier, we have included a lot of class variation in the dataset and even among the images of the same classes as well. We have divided our dried canal bed dataset into six classes namely *bigstones*, *clearbed*, *sand*, *smallstones*, *vegetation* and *water*(*stationary*) as shown in the Figure 7-9.

Figure 7-9: Six classes of the dataset are used for the adaptation of the proposed terrain classification technique for the dried water canal bed. *Top left*: Big stones. *Top middle*: Clear bed. *Top right*: Sand. *Bottom left*: Small stones. *Bottom middle*: Vegetation. *Bottom right*: Water(stationary).

Furthermore, a Confusion Matrix for the canal bed dataset also validated the accuracy and robustness of our proposed approach for the identification of dried canal bed as shown in Table 7.5. The diagonal values of the confusion matrix show our proposed approach has trained well on the training dataset and classified the dried water bed terrain with high accuracy.

Table 7.5: Confusion Matrix against our proposed approach with Canal Bed Dataset

		Predicted Class					
		BigStones	ClearBed	Sand	SmallStones	Vegetation	Water
Actual Class	BigStones	458	0	3	21	6	0
	ClearBed	0	510	16	0	2	5
	Sand	1	12	370	0	6	3
	SmallStones	11	0	0	455	8	0
	Vegetation	6	1	11	3	505	14
	Water	0	0	3	0	18	510

Furthermore, Figure 7-10 shows some sample images of the truly classified output images from our aerial platform over a dried water channel. Similarly, the wrongly classified images are also shown in Figure 7-11. These classifications are performed in real-time with the help of an onboard processing unit in the drone.

Figure 7-10: Images showing the real-time classification result of dried water canal bed terrains.

Figure 7-11: Sample images showing the false positive and false negative results from real-time terrain classification experiments on the dried water canals.

In the next chapter, we have discussed a soil volume estimation approach that can be adapted to be another possible water canal structural monitoring application for silt volume estimation. It is performed over-the-point cloud data gathered during the structural inspection of water canals to estimate the approximate volume of the silt accumulated to some extent.

Chapter 8

Soil/Silt Estimation using 3D Scan data

In this chapter, we have presented a technique to measure the quantity of silt accumulated at any given part of the water canal. Our objective here is to demonstrate the utility of our proposed aerial robotics platform for extended applications to further assist the structural monitoring problem for the water channels. The current application illustrated in this chapter can play an important part in the monitoring and maintenance of water canals. As discussed previously, silt accumulation decreases the water carrying capacity of the water channels and can even result in leakage from the banks of the canal. The proposed integrated aerial platform in this thesis earlier can scan the canal environment using a Laser scan mounted at the bottom of the drone and helps reconstruct the environment in a form of a 3D point cloud. This point cloud can be used to estimate the accumulated silt present at any particular part of the canal scene.

Earlier, we have proposed a similar technique to estimate the quantity of soil present in the environment after reconstructing it in a form of a point cloud [13]. The overall goal of the presented system [13] is to estimate the amount of soil in cm^3 in a single scoop of a bucket excavator while excavating the soil. We have used a stereo camera to reconstruct

the environment in a form of a 3D point cloud after the excavator has scooped the soil, and then presented a novel approach to locate the excavator bucket in the 3D point cloud. After identification of the bucket, the volume of the soil is calculated inside the bucket against a reference empty bucket using the point cloud subtraction technique. In the next part of this chapter, we have described in detail the working principle of the soil estimation technique for the bucket excavator problem. Later, we shall discuss how the proposed technique can be adapted for the estimation of accumulated silt in a water canal using the 3D point cloud reconstruction.

8.1 Literature Survey

For the estimation of soil from a point cloud, research work on generic quantitative estimation is found to be sporadic, but concerning this, there is considerable research on autonomous loading and excavation operations. Cannon [39], in his thesis on earth-moving with an autonomous excavator, talks about perception enhancements in a dig execution. The focus of his work is on planning an optimal dig of the excavator to achieve the minimum cost – where the cost function is determined by volume scooped, energy used, and time spent for the single dig. His methodology is to store the shape of terrain before digging and, based on the bucket's trajectory of scooping, continuously integrate the swept volume. Similar to our problem, the digging and perception are decoupled problems for him. But, his work does not incorporate the alignment issues, nor is it expendable to all sorts of scenarios.

Dunbabin [58] also addresses the similar problem of determining dipper fullness in rope shovel excavation. He has used two approaches for it: one to use laser-based scanning, and the other to use motor power signals at the dipper – measuring the quantity of excavated soil. His laser-based sensing determines the maximum height of soil in a bucket to calculate the volume and ignores the precise soil profile of the container compromising accuracy and portability. A study on laser scanning for excavation measurement [46] has talked about

the registration (alignment) problem in excavations which later leads to a discussion of the volume estimation problem. It has solved this estimation problem by using the depth difference in the terrain after excavation but their problem of getting the ground truth i.e. the actual excavated soil volume remained unsolved. Other works by Stentz [188] and Hemami [[91], [90]] have devised methods to plan dig trajectories that maximize excavated soil volume. However, consideration of the accuracy in achieving the volume requirement in this planned scooping has not been properly dealt with, and they have not focused on the precision of volume measurements.

8.2 Soil Estimation for the Excavation Problem

The methodology of the proposed approach contains several steps to compute the volume of soil inside the bucket of an excavator as shown in Figure 8-1. The initial step is

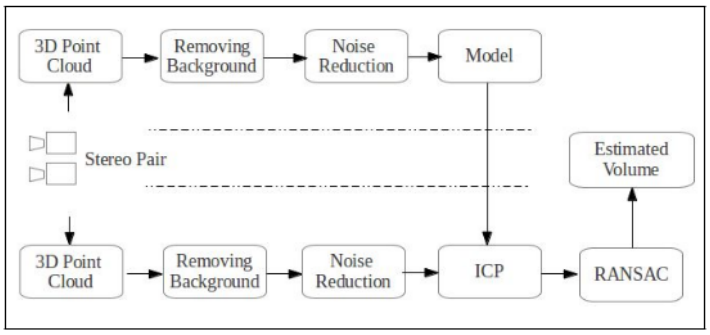


Figure 8-1: System Architecture for the estimation of soil inside the bucket of an excavator by using 3D point cloud.

the sensing part of the system and we have used a stereo camera for the 3D sensing of the environment. We have used custom build stereo cameras with a stereo baseline of $10cm$. Both the cameras are calibrated using a set of chessboard images to find both intrinsic and extrinsic parameters of the cameras for the calibration process. Once calibrated, as a next step, a Semi-Global Block matching technique [93] is used with the block window

size of 9×9 pixels to get the disparity image. A disparity image shows the distance of each pixel in the image from the camera frame of reference in a form of pixel brightness intensities. Next, we reconstruct all the disparity pixel points into a 3D point cloud form with the help of the disparity image and the camera intrinsic parameters. After we obtain the environment's point cloud, we subtract the unnecessary background (the region of space more than a maximum expected distance of the bucket container) and then apply a statistical filter to remove the outliers from the 3D cloud. Figure 8-2 shows the intermediate results from a camera image to the disparity image and then to the generation of the 3D point cloud of the bucket of an excavator.

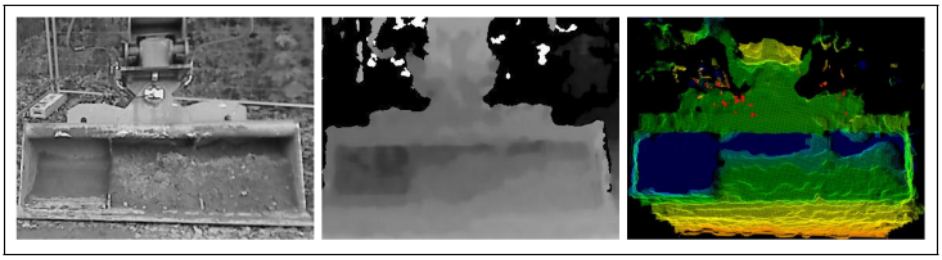


Figure 8-2: *Left*: An image showing a two-third filled excavator bucket. *Middle*: A disparity image generated using a custom stereo camera. *Right*: 3D point cloud is generated using the disparity image.

In the next step, we have modeled the bucket of the excavator as a container that contains soil inside it to measure. There are plenty of ways to accomplish this, including a CAD drawing available from the manufacturer of the container, or a database of standard containers in varying shapes and sizes. But to make our process independent of such initial data we make our model of the container. The container model is generated only once. We employ the aforementioned 3D sensing technique on a stereo image pair in which our container is empty and completely visible. This will serve as a reference cloud when comparing it with the current cloud for estimating the change in the volume contained. Once we have the container model, we manually mark the four corners of the container

as shown in Figure 8-3. This way we divide our model into two point clouds: the outside of the container (used in alignment) and the other, comprising the inside of the model for volume measurement. Manual marking is acceptable as the model is only created once for a particular container. Once we have the empty container model in the form of two

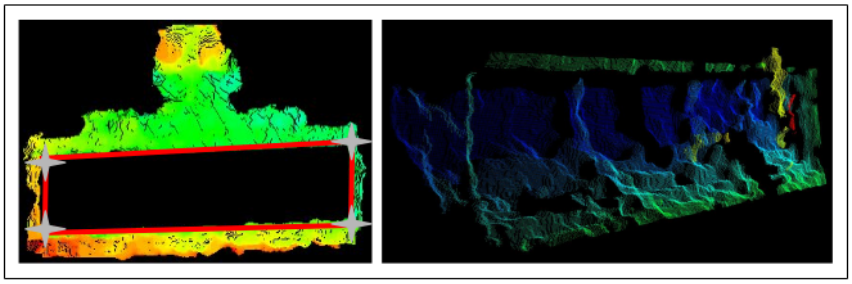


Figure 8-3: *Left*: Outer part of the model for alignment and gray marks to manually choose the container boundary. *Right*: Inside of the container area is separated from the rest of the model for the volume estimation.

complementary point clouds, we are ready to take the new scan of the filled container. The newly acquired point cloud will most likely not have the container at the same position as the model, due to bucket motion during excavation. Thus, we need to align the new point cloud with the outer part of the container model for the exact measurements. The alignment is achieved by using the Iterative Closest Point(ICP) matching technique [217]. The current scan is taken as the target and the container model which shows the outside part of the soil bucket as the source with which target points are to be aligned. Maximum iterations of 200 are used for ICP and 0.05 meters is the maximum correspondence distance. RANSAC outlier rejection threshold was set to 1 and transformation epsilon to $10 - 8$. It gives us a 3D transformation matrix existing between the two clouds by which we transform the current cloud. After applying the transformation on every point in a current point cloud, the resulting points are exactly aligned to the model as shown in Figure 8-4.

Finding the volume is the final step of this process. In this, we compartmentalize both the model and the current clouds into two 3D grids respectively. The grid size is manually

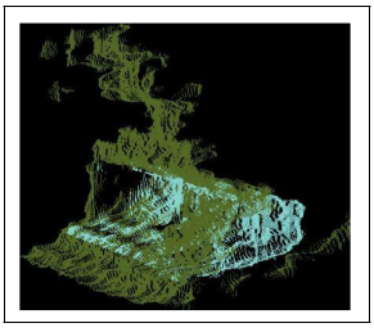


Figure 8-4: *Blue*: A container model showing the empty container. *Green*: Current point cloud aligned with the model point cloud after using ICP.

chosen based on a variety of parameters – density of the point cloud, the size and capacity of the container box, the granularity of the material (soil), the required precision, and the required accuracy of measurement. Since our current and container model point clouds are both aligned we can directly compare the points in their grid elements. Inside every raster, we fit a plane in each of the two corresponding point cloud boxes. The plane coefficients are approximated by running RANSAC (RANdom SAmple Consensus) over the data [72]. With this, we get the two plane equations.

For the missing cells i.e. holes with not enough points for the plane fitting, we fill the region of space by assigning it the average plane coefficients of the nearest neighboring rasters. Next, we find the two median points of the two planes. While finding the height of soil in a grid cell, we use the median point of the plane instead of using the mean point to reduce outlier weightage. Euclidean distance between these two points is computed – this is the height of soil for a particular cell. The height along with the knowledge of the grid size in units of length is used to calculate the volume enclosed by the raster. This process of numerical integration is repeated over the entire grid to get the total aggregated volume of the contained soil.

Although we tested our soil estimation technique in the simulation environment first for verification, we do not discuss them here in this thesis. All the simulation experiments are

described here [13]. After getting encouraging results in simulation, we have conducted experiments on a real bucket excavator. We have filled the excavator bucket with a measured quantity of soil using a smaller handy bucket of a known capacity – this is our source of the volume ground truth for comparison. We have conducted experiments at different stages during our filling of 305 liters of soil. Our sensing system has shown great results for generating the dense point cloud despite it being our custom-made stereo pair. The results are shown in Table 8.1.

Table 8.1: Real world experiment results on a Bucket Excavator for the soil estimation.

Quantity of Soil in Bucket	Actual Volume ($10^3\ cm^3$)	Estimated Volume ($10^3\ cm^3$)	Error in Estimated Volume (%)
Full	305.4	307.2	0.57
Two-Third	215.8	220.7	2.23
One-Third	146.6	169.1	15.4
Empty	0.0	27.1	-

8.3 Adapting Soil Estimation Technique for Silt Estimation

The proposed method of estimating the quantity of soil inside the bucket of an excavator can be adapted for the determination of accumulated silt inside the dried water canal. The objective of both operations are quite similar hence we can use a significant part of the already developed system for the estimation of silt quantity. An aerial robot which is being presented in Chapter 3 of this thesis, is equipped with an integrated LiDAR sensor system at the bottom called the '*Bottom LiDAR*'. A point cloud can be generated against each scan of the LiDAR and can replace the need for the stereo camera system used in the soil estimation approach earlier. Moreover, LiDAR is more accurate and robust in comparison to the stereo camera system. Different lighting conditions or the viewing angle can negatively affect the result performance which eventually results in poor disparity.

Furthermore, the point cloud generated from the poor disparity results in an incomplete point cloud.

For the estimation of silt, we can use the point cloud generated by using the onboard LiDAR. In the next step, we need to compare the generated point cloud against a reference point cloud to calculate the difference between both point clouds over time. That difference can be the accumulated silt of other structural changes. Before comparing two point clouds, we need to make sure that both point clouds are exactly aligned to each other. This alignment can be done in two steps. In the first step, global alignment is done by using the trajectory generated by the state estimation module presented in Chapter 5. We shall transform the trajectory generated into the 'UTM frame of reference' for a constant global reference. The UTM-based trajectories are compared to calculate the relative transformation between the reference point cloud and the current point cloud. It will give us the transformation matrix for the global consistency but this may not reflect the precise alignment details. This can be removed in the second step by using the natural landmarks either some tree, structure, or the bank edges of the water canal. This represents the same step as we have used in the excavator problem for the identification of the bucket by aligning the container model with the current point cloud using ICP.

After applying the above-mentioned global and local alignment techniques, we shall get two closely aligned reference and current point clouds. Again, by using the same point cloud comparison technique used in the excavation problem, we can calculate the difference between both point clouds in the form of volume in cm^3. The calculated volume can be the accumulated silt or some other object present there which may be subject to either removal or further inspection.

Chapter 9

Epilogue

The chapter serves as an overview of the research work presented in this thesis. The first part starts with a review of the thesis research work by visiting each module presented earlier and how it help for the development of an aerial robotic system for a canal structural inspection problem. In the second part, we put forward the limitation and shortcomings of the proposed modules and how they may affect the overall performance of the aerial inspection system. Lastly, we look towards the future avenues that might get addressed in the future research to further complete the proposed system in terms of both operation and practicality.

9.1 Modular Approach towards Building an Autonomous Canal Inspection System

Water is considered to be the most valuable renewable resource for any country in the world. It plays an important role in economic progress by improving GDP growth. Even in abundance, efficient policies and smart infrastructure is still required to optimally utilize this resource and gain maximum benefits. A lot of countries that rely on agriculture are directly dependent on water for their social and economical development. Indus basin has

the privilege of containing one of the largest contiguous canal irrigation systems under gravity flow. It has an arid to semi-arid climate, so the timely distribution of water to the fields via water canals is an inevitable necessity.

A common problem with these water reservoirs is the constant degradation of the infrastructure due to some inevitable factors. The factors are seasonal floods, accumulation, and settling of silt, erosion of water canal banks, the effect of urbanization and wastewater, etc. Accumulation and sedimentation of silt on canal beds highly affect the water carrying and discharge capacities of canals. Also, the fast-flowing water causes canal banks to erode and damage. If canals are not maintained and cleaned properly from time to time, the canal loses its water carrying capacity and may also start leaking. So canal must be maintained and clean on regular basis to keep its structural integrity.

Practically structural inspection in the Indus basin is done once every year, simultaneously closing all the water canals for three weeks. Here, the inspection is done through a walk-through process to identify damages or accumulation of silt. This is a manual technique based on many subjective decisions. As this method is prone to human errors and includes a lot of approximation which either cause over or underestimation of the damage. Furthermore, to identify and estimate the structural damage over thousands of kilometers in this short span of three weeks is near to impossible. Consequently, an automated system for the inspection of canal structure is the need of time and forms the motivation of this thesis.

A complete automated canal inspection system is quite a complex problem that requires many cutting-edge technologies to work together to achieve the goal. In this thesis, we have divided the whole structural inspection problem into different modules and have presented novel approaches to solve the most important of them. First, we have chosen an aerial drone to be the most suitable platform for the traversal of the water canal. In addition, we have selected the annual closure period for our inspection so that we may have a complete picture of the canal structure including the bed and the banks. In this thesis, we

have developed and proposed a working module against each of the problem statements discussed in Chapter 1. In the current chapter, Section 9.1.1 shows the system integration and data post-processing of the proposed system that is capable of traversing the length of the canal with minimum human interaction. It further describes the integration of state-of-the-art sensor systems that are sufficient for the proposed aerial platform to reconstruct the civil structures of a water canal in a 3D point cloud. In Section 9.1.3, a customized state estimation approach is proposed to accurately determine the pose of the aerial robot in 6DOF using Extended Kalman Filter (EKF). Furthermore, in Section 9.1.2, we have proposed a novel technique to validate the accuracy of the proposed state estimation module in a large outdoor environment by using the commonly used visual fiducial marker. Likewise, Section 9.1.4 discusses the proposed autonomous aerial navigation technique to traverse the length of the water canal using a Convolutional Neural Network (CNN). In the last Section (9.1.5) of the current chapter, we have utilized the developed aerial system and presented a novel technique for the classification of the water canal bed for accurate water discharge estimation and the estimation of the quantity of the soil accumulated on canal's bed using generated 3D point clouds.

9.1.1 Development & System Integration of an Aerial Robot

Initially, we presented an innovative approach for the development of an aerial platform including the integration of sensor systems over it. Integrated sensor systems include GPS, IMU, monocular camera, bottom LiDAR, and top LiDAR. Calibration and data post-processing approaches have been presented to optimally utilize these sensors for waterway inspection. To increase the accuracy of an onboard GPS, a Kalman filter-based approach has been proposed as a post-processing step and showed the improvement by a detailed experiment. We have also addressed the noise associated with each axis of the accelerometer and proposed a Finite Impulse Response (FIR) based low-pass filter approach to remove noise. Moreover, this low-pass filter causes a delay in the real-time

system due to data structuring the inputs in an array and by D Flip-Flops at the hardware level. This delay has been removed by the product of Z-transform of sensor input readings and z^{-i}.

For an onboard monocular camera, Radial and Tangential distortions have also been identified and their solution is proposed by using the Levenberg-Marquart optimization approach. In the end, multiple test flights of the proposed aerial system have been conducted over real water canals along with data logging and later data visualization from the onboard integrated LiDARs are also shown. Moving ahead, we have proposed an integrated aerial robot for the required autonomous operation. However, the most important module for the success of the whole water canal inspection system is the accuracy of the estimation of the state of the aerial platform during the operation. Since on-board range sensors are taking their readings from the aerial platform to reconstruct the canal structures, hence knowing the precise location and pose of the flying robot is inevitable as everything depends upon it.

9.1.2 Ground-truth Validation for State Estimation

One of the most important modules is the estimation of the state of the drone during operation. Hence, we have used fiducial markers for the ground-truth analysis of the localization module as they are the most suitable ones in an outdoor environment. Fiducial markers are a low-cost solution for getting accurate ground truth measurements in applications, especially in robotics. Among all state-of-the-art fiducial markers, AprilTag is the most commonly used fiducial marker by researchers. Fast and sturdy tag detection techniques, stronger digital coding for an embedded marker, robustness against different lighting conditions, lens distortion, and small occlusion are the main features that make AprilTag unique from other fiducial markers. However, researchers have experienced that AprilTag lacks the required precision and accuracy for delicate tasks. Hence, researchers have used a different combination of sensors along the AprilTag to improve its accuracy. In

the chapter 4, we have empirically analyzed AprilTag with the identification of shortcomings causing inaccuracies. With the help of extensive experiments and analysis, we have analytically identified that the primary source of error is the yaw angle variation of the viewing camera, which has not been compensated in the current AprilTag implementation. Other inaccuracy sources include the distance and viewing angle of the camera to tag.

Besides, based upon the identified shortcomings, three improvement approaches have been proposed to further improve the accuracy and precision of AprilTag with a slight execution time trade-off. The first proposed approach includes passive correction of camera yaw angle by using trigonometric corrections to point the camera towards the center of AprilTag. The second approach uses a custom build hardware-based tracking gimbal to align the face of the camera towards the center of the tag. Lastly, we have demonstrated how to use the experimental data to build a probabilistic model of the AprilTag sensor using Gaussian Processes (GP) Regression that can be reused as a forward sensor model in many localization-based applications. Also, we have demonstrated that the accuracy and precision of AprilTag increase manifolds if we use the proposed approaches in combination with each other. Comparative results with the Motion Capture (MoCap) system have been shown to best show the improvement proposed. The suggested enhancement approaches can be used in multiple applications, including robotics and virtual reality (VR). We have experimentally tested the proposed approaches in both indoor and outdoor environments to show the completeness of the proposed probabilistic sensor model.

9.1.3 Estimating the State of the Aerial Robot

Later in Chapter 5, we have presented our approach for the estimation of the state of a flying drone with the help of onboard sensors in real-time. The state of the drone includes the position and pose angles in 6-DOF. To precisely restructure the canal environment, a high degree of accuracy is achieved for the localization module by using customized sensor models for each onboard sensor and fusing them by using an Extended Kalman

Filter(EKF)[124]. The Sensor models and the measurement vector of each sensor system are described in detail along with the required transformation needed to align the data from the multiple sensors system against a common frame of reference. The fused sensor systems include IMU, visual odometry (generated using a monocular camera), Head-LiDAR for horizontal positioning and heading accuracy, and GPS for global consistency. Moreover, we also presented a dynamical model of the octocopter which is based upon a control input vector μ_t and on a series of proportionality coefficient parameters (κ). The thrust generated by each propeller is assumed constant and the tuning parameter are determined by rigorous experimentation and can differ with the change in the payload of the drone. Following in the chapter, we compared the trajectory generated by the proposed EKF approach with the trajectories generated by the individual sensors or by other state estimation techniques such as LSD-SLAM [61]. The comparisons show that our proposed approach is more accurate and robust than either of the individual sensor system or against the state-of-the-art vision-based localization approach.

At the end of Chapter 5, 3D reconstruction of the canal structures is presented using the same trajectories generated earlier. The reconstructed structures are in a form of a 3D point cloud generated by the use of the bottom-LiDAR(2D). Through the LiDAR scans the environment in a 2D plane but the lateral movement of the drone along the length of the water canal and stitching the consecutive scan together based upon the state estimation module, builds a 3D representation of the canal scene.

9.1.4 Autonomous Aerial Robot Navigation over Water Canal

Afterward, in Chapter 6, we present an autonomous aerial traversal system for waterways using ResNet50 inspired deep convolutional neural network (CNN). The network is designed in such a way that it takes an RGB image as an input and predicts the position of a micro aerial vehicle over a canal. Due to the unique nature of the task, there is no pre-existing dataset that consists of images of the irrigation canal taken along its banks.

Therefore, we created the required first dataset with nearly thirty thousand canal images to train and validate our proposed system. Creating datasets was itself a challenging task as a manual traversal of canals is not possible due to limited mobility options. It can only be collected by manually flying an aerial platform with an onboard camera. Annotating various classes from the collected data required a lot of effort. This was due to uncertain conditions during flight time generated by wind, tree obstructions, electricity wires, and internal drifts of the flight controller. Trained on this dataset, our CNN labels the current position of a micro aerial vehicle in one of three ways: either it is flying near the left bank or above the center of the canal or near the right bank. Based upon the predicted MAV position, a formal method-based command controller has been synthesized which ensures that the platform stays over the canal and follows its trajectory. The system has been tested successfully on a real canal with real-time response. Moreover, the proposed ResNet50 inspired network gives superior performance over other state-of-the-art CNNs trained on a similar canal dataset.

9.1.5 Further Application for Canal Structural Inspection

The last section of the thesis presents two applications that can be developed using the proposed integrated aerial robot to further assist the structural inspection of the water channels. First, we proposed a novel approach for the classification of the canal bed to identify its terrain. This helps the hydrologist to come up with a more accurate water flow model as the terrain beneath the water is known. A novel terrain classification system has been reported in chapter 7 that combines texture-based learning with color-based learning for performance enhancement. For a fair comparison, the system is trained offline on an image database and then compared against the two approaches, one existing texture feature-based classification, the other our proposed color-based classification. It is also shown that the algorithm performs in real-time with little or no overheads. An appreciable performance boost has been observed in overall accuracies of all classes where

the predecessor systems fall short. The classification combination approach supports the hypothesis that natural images are sometimes rich in color and sometimes in textures but not always in both.

Second, we proposed a novel technique to estimate the quantity of accumulated silt in a water canal. We have adapter an earlier technique of soil estimation in an excavation problem and have proposed the adaptation to measure the accumulated silt in a canal environment. We have presented a novel method of estimating the volume of contained soil using a 3D point cloud. A major advantage of this approach is the use of a reference 3D point cloud as a metric to measure the quantity of soil by aligning currently scanned and a model point cloud generated earlier using either localization or natural marker. Once closely aligned, the change in the currently scanned point cloud is calculated by comparing it with the reference point cloud.

9.2 Limitations and Future Work

The flight time of the proposed aerial platform is limited to a few kilometers due to the shortage of space and restrictions in the payload carrying capability. This has to be increased to multiple hundreds of kilometers as the water canals in any irrigation network are spread over a large landscape. Another limitation with the aerial platform is the lack of auto-pilot features like home-returning, GPS path following, and fail-safe state maneuvers. The integration of the above-mentioned autopilot features can improve the overall safety and usability of the system. Furthermore, we have considered certain flight parameter constant for all the proposed modules of the aerial autonomous system. The primary objective of these assumptions is to minimize the uncertainty of the environment which can affect the overall accuracy of the proposed system. The details of the assumed flight parameters are shown in Table 9.1. For the whole thesis, we have assumed these assumptions for the proposed aerial system. The flying altitude of the aerial platform is between 15m to 20m from the bed of a water canal in all scenarios. All the data sets

generated for the training of the presented navigation and terrain classification modules are captured from the same altitude. Changing the altitude of the aerial system can affect the accuracy of the proposed modules. Similarly, the autonomous aerial system is flown at a constant speed of 3 m/s over the length of the water canal for the detailed collection of the laser scans. This helps us to generate a highly detailed and accurate 3D reconstruction of the structures of a water canal for inspection purposes. In addition, to reduce the complexity of the navigation module, all the test flights are performed with a maximum wind speed of 20 km/h. High wind introduces destabilization in the flight of the aerial platform which adds additional complexity to the navigation module.

Technical Flight Parameters for the Proposed Aerial System	
Flight Altitude	from 15m to 20m above a canal bed
Aerial Platform Ground Speed	3 m/s
Maximum Wind Speed	20 km/h
Weather Condition	Clear or cloudy skies (no rain)
Operating Temperature	from 10° C to 35° C
Maximum Humidity	70%

Table 9.1: Constant technical flight parameters for the proposed aerial platform.

In the ground-truth validation of state estimation, we have only analyzed horizontal, vertical and yaw axis accuracy reported by the AprilTag, which is sufficient for localization applications dealing with the two-dimensional plane. However, for more complex operations in 6-DOF environments like aerial robotics, other axes such as height, roll, and pitch axis are also principal. Further work needs to be done in that direction as well to validate the height and the pose angles as well. Additionally, for hard real-time applications, we believe that the proposed custom build yaw-axis gimbal for active correction of camera yaw angle does not move fast enough to match the hard time constraints. The proposed approach can be further improved by using FPGA-based implementation for a quick response. Furthermore, a servo motor can also be improved to increase the speed of tracking. Also, the theoretical framework for GP regression in chapter 4 makes some

assumptions and simplifications that need further investigation. Another possible open direction of future work might be the inclusion of multiple sensors tag in the probabilistic sensor model to enhance the performance further. Furthermore, fusing the data of the Inertial Measurement Unit (IMU) while tracking AprilTag can enhance performance. It may improve the robustness of the robot-generated trajectory by filling the gaps when AprilTag is not detected. Nonetheless, there exist multiple directions for extension of this work, which we have attempted to make accessible to the robotics community for reuse in their research [2] and lay the exposition open to critical examination and investigation of the community.

Bibliography

[1] Daniel F Abawi, Joachim Bienwald, and Ralf Dorner. Accuracy in optical tracking with fiducial markers: an accuracy function for artoolkit. In *Third IEEE and ACM International symposium on mixed and augmented reality*, pages 260–261. IEEE, 2004.

[2] Syed M. Abbas. Online AprilTag Dataset. `http://cyphynets.lums.edu.pk/index.php/Apriltag`, 2019. Accessed: 2019-12-20.

[3] Syed Muhammad Abbas and Abubakr Muhammad. Outdoor rgb-d slam performance in slow mine detection. In *Robotics; Proceedings of ROBOTIK 2012; 7th German Conference on*, pages 1–6. VDE, 2012.

[4] Gul-I-Rayna Abubakr M. Silt removal from irrigation canals in punjab. *LUMS internal report*, 2013.

[5] Shyam Adhikari, Changju Yang, Krzysztof Slot, and Hyongsuk Kim. Accurate natural trail detection using a combination of a deep neural network and dynamic programming. *Sensors*, 18(1):178, 2018.

[6] A Pedro Aguiar and Joao P Hespanha. Trajectory-tracking and path-following of underactuated autonomous vehicles with parametric modeling uncertainty. *Automatic Control, IEEE Transactions on*, 52(8):1362–1379, 2007.

[7] K Alexis, G Nikolakopoulos, and A Tzes. Model predictive quadrotor control: attitude, altitude and position experimental studies. *Control Theory & Applications, IET*, 6(12):1812–1827, 2012.

[8] Erdinc Altug, James P Ostrowski, and Robert Mahony. Control of a quadrotor helicopter using visual feedback. In *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, volume 1, pages 72–77. IEEE, 2002.

[9] Erdinç Altuğ, James P Ostrowski, and Camillo J Taylor. Control of a quadrotor helicopter using dual camera visual feedback. *The International Journal of Robotics Research*, 24(5):329–341, 2005.

[10] H Alvarez, Lina Maria Paz, Jürgen Sturm, and Daniel Cremers. Collision avoidance for quadrotors with a monocular camera. In *Experimental Robotics*, pages 195–209. Springer, 2016.

[11] Omead Amidi, Takeo Kanade, and Keisuke Fujita. A visual odometer for autonomous helicopter flight. *Intelligent Autonomous Systems, IOS Press, Amsterdam*, pages 123–130, 1998.

[12] Seth B Anderson. Historical overview of v/stol aircraft technology. 1981.

[13] Hamza Anwar, Syed Muhammad Abbas, Abubakr Muhammad, and Karsten Berns. Volumetric estimation of contained soil using 3d sensors. In *Intl. Commercial Vehicle Technology Symposium*, pages 11–13, 2014.

[14] Hamza Anwar, Abubakr Muhammad, and Karsten Berns. A theoretical framework for aerial inspection of siltation in waterways. In *IEEE IRSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[15] George J Arcement and Verne R Schneider. Guide for selecting manning's roughness coefficients for natural channels and flood plains. 1989.

[16] M Arshad, N Ahmad, M Usman, A Shabbir, et al. Comparison of water losses between unlined and lined watercourse in indus basin of pakistan. *Pak. J. Agri. Sci*, 46(4):2076–0906, 2009.

[17] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

[18] Abraham Bachrach, Samuel Prentice, Ruijie He, Peter Henry, Albert S Huang, Michael Krainin, Daniel Maturana, Dieter Fox, and Nicholas Roy. Estimation, planning, and mapping for autonomous flight using an rgb-d camera in gps-denied environments. *The International Journal of Robotics Research*, 31(11):1320–1343, 2012.

[19] Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT press, 2008.

[20] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.

[21] Gilles Belaud and Jean-Pierre Baume. Maintaining equity in surface irrigation network affected by silt deposition. *Journal of irrigation and drainage engineering*, 128(5):316–325, 2002.

[22] Paolo Bellutta, Roberto Manduchi, Larry Matthies, Ken Owens, and Art Rankin. Terrain perception for demo iii. In *Intelligent Vehicles Symposium, 2000. IV 2000. Proceedings of the IEEE*, pages 326–331. IEEE, 2000.

[23] Filippo Bergamasco, Andrea Albarelli, Emanuele Rodola, and Andrea Torsello. Rune-tag: A high accuracy fiducial marker with strong occlusion resilience. In *CVPR 2011*, pages 113–120. IEEE, 2011.

[24] Karsten Berns, Klaus-Dieter Kuhnert, and Christopher Armbrust. Off-road robotics—an overview. *KI-Künstliche Intelligenz*, 25(2):109–116, 2011.

[25] Massimo Bertozzi and Alberto Broggi. Gold: A parallel real-time stereo vision system for generic obstacle and lane detection. *Image Processing, IEEE Transactions on*, 7(1):62–81, 1998.

[26] Antoine Beyer. Inland waterways, transport corridors and urban waterfronts. International Transport Forum Discussion Paper, 2018.

[27] Luna Bharati, Bharat R Sharma, and Vladimir Smakhtin. *Ganges River Basin*. Taylor & Francis, 2016.

[28] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[29] Michael Blosch, Stephan Weiss, Davide Scaramuzza, and Roland Siegwart. Vision based mav navigation in unknown and unstructured environments. In *Robotics and automation (ICRA), 2010 IEEE international conference on*, pages 21–28. IEEE, 2010.

[30] Dorit Borrmann, Jan Elseberg, Kai Lingemann, Andreas Nüchter, and Joachim Hertzberg. Globally consistent 3d mapping with scan matching. *Robotics and Autonomous Systems*, 56(2):130–142, 2008.

[31] Samir Bouabdallah. Design and control of quadrotors with application to autonomous flying. Technical report, Epfl, 2007.

[32] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. " O'Reilly Media, Inc.", 2008.

[33] Pierre-Jean Bristeau, François Callou, David Vissière, and Nicolas Petit. The navigation and control technology inside the ar. drone micro uav. *IFAC Proceedings Volumes*, 44(1):1477–1484, 2011.

[34] João Britto, Diego Cesar, Rafael Saback, Sascha Arnold, Christopher Gaudig, and Jan Albiez. Model identification of an unmanned underwater vehicle via an adaptive technique and artificial fiducial markers. In *OCEANS 2015-MTS/IEEE Washington*, pages 1–6. IEEE, 2015.

[35] Duane C Brown. Decentering distortion of lenses. *Photogrammetric Engineering and Remote Sensing*, 1966.

[36] Adam Bry, Abraham Bachrach, and Nicholas Roy. State estimation for aggressive flight in gps-denied environments using onboard sensing. In *2012 IEEE International Conference on Robotics and Automation*, pages 1–8. IEEE, 2012.

[37] Fernando Caballero, Luis Merino, Joaquín Ferruz, and Aníbal Ollero. A visual odometer without 3d reconstruction for aerial vehicles. applications to building inspection. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 4673–4678. IEEE, 2005.

[38] Brandon Call, Randy Beard, Clark Taylor, and Blake Barber. Obstacle avoidance for unmanned air vehicles using image feature tracking. In *AIAA Guidance, Navigation, and Control Conference*, pages 3406–3414, 2006.

[39] Howard N Cannon. *Extended earthmoving with an autonomous excavator*. PhD thesis, Carnegie Mellon University, 1999.

[40] Rebecca Castano, Roberto Manduchi, and Justin Fox. Classification experiments on real-world texture. 2001.

[41] Andrew Chambers, Supreeth Achar, Stephen Nuske, Jörn Rehder, Bernd Kitt, Lyle Chamberlain, Justin Haines, Sebastian Scherer, and Sanjiv Singh. Perception for a river mapping robot. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 227–234. IEEE, 2011.

[42] HC Chen and Szi-Wen Chen. A moving average based filtering system with its application to real-time qrs detection. In *Computers in Cardiology, 2003*, pages 585–588. IEEE, 2003.

[43] Lyujie Chen, Wufan Wang, and Jihong Zhu. Learning transferable uav for forest visual perception. *arXiv preprint arXiv:1806.03626*, 2018.

[44] Wei-Chao Chen, Yingen Xiong, Jiang Gao, Natasha Gelfand, and Radek Grzeszczuk. Efficient extraction of robust image features on mobile devices. In *Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 1–2. IEEE Computer Society, 2007.

[45] Min-Lung Cheng, Masashi Matsuoka, Wen Liu, and Fumio Yamazaki. Near-real-time gradually expanding 3d land surface reconstruction in disaster areas by sequential drone imagery. *Automation in Construction*, 135:104105, 2022.

[46] Geraldine S Cheok, Robert R Lipman, Christoph Witzgall, Javier Bernal, and William C Stone. Field demonstration of laser scanning for excavation measurement. In *Proceedings of the 17th International Symposium on Automation in Robotics in Construction, Taipei, Taiwan*, 2000.

[47] Annett Chilian and Heiko Hirschmuller. Stereo camera based navigation of mobile robots on rough terrain. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 4571–4576. IEEE, 2009.

[48] Youngkwan Cho, Jongweon Lee, and Ulrich Neumann. A multi-ring color fiducial system and an intensity-invariant detection method for scalable fiducial-tracking augmented reality. In *In IWAR*. Citeseer, 1998.

[49] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

[50] Howie M Choset, Kevin M Lynch, Seth Hutchinson, George Kantor, Wolfram Burgard, Lydia Kavraki, Sebastian Thrun, and Ronald C Arkin. *Principles of robot motion: theory, algorithms, and implementation*. MIT press, 2005.

[51] Peter Corke, Pavan Sikka, and Jonathan Roberts. Height estimation for an autonomous helicopter. In *Experimental Robotics VII*, pages 101–110. Springer, 2001.

[52] Dayana Bastos Costa and Alexandre TC Mendes. Lessons learned from unmanned aerial system-based 3d mapping experiments. In *52nd ASC Annual International Conference Proceedings*, 2016.

[53] Hendrik Dahlkamp, Adrian Kaehler, David Stavens, Sebastian Thrun, and Gary R Bradski. Self-supervised monocular road detection in desert terrain. In *Robotics: science and systems*. Philadelphia, 2006.

[54] A Das, K Subbarao, and F Lewis. Dynamic inversion with zero-dynamics stabilisation for quadrotor control. *IET control theory & applications*, 3(3):303–314, 2009.

[55] Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. Monoslam: Real-time single camera slam. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(6):1052–1067, 2007.

[56] Bothezat George De. Helicopter, March 4 1930. US Patent 1,749,471.

[57] Joao Paulo de Almeida Barbosa, Stiven Schwanz Dias, and Davi Antônio dos Santos. A visual-inertial navigation system using apriltag for real-time mav applications. In *2018 25th International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*, pages 1–7. IEEE, 2018.

[58] Matthew Dunbabin and Peter Corke. Autonomous excavation using a rope shovel. *Journal of Field Robotics*, 23(6-7):379–394, 2006.

[59] Daniel Eberli, Davide Scaramuzza, Stephan Weiss, and Roland Siegwart. Vision based position control for mavs using one single circular landmark. *Journal of Intelligent & Robotic Systems*, 61(1-4):495–512, 2011.

[60] Matthew J Edwards, Michael P Hayes, and Richard D Green. High-accuracy fiducial markers for ground truth. In *2016 International Conference on Image and Vision Computing New Zealand (IVCNZ)*, pages 1–6. IEEE, 2016.

[61] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *Computer Vision–ECCV 2014*, pages 834–849. Springer, 2014.

[62] Jakob Engel, Jürgen Sturm, and Daniel Cremers. Accurate figure flying with a quadrocopter using onboard visual and inertial sensing. *IMU*, 320:240, 2012.

[63] Bora Erginer and Erdinc Altug. Modeling and pd control of a quadrotor vtol vehicle. In *Intelligent Vehicles Symposium, 2007 IEEE*, pages 894–899. IEEE, 2007.

[64] Ryan M Eustice, Hanumant Singh, and John J Leonard. Exactly sparse delayed-state filters. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 2417–2424. IEEE, 2005.

[65] Matthias Faessler, Flavio Fontana, Christian Forster, Elias Mueggler, Matia Pizzoli, and Davide Scaramuzza. Autonomous, vision-based flight and live dense 3d mapping with a quadrotor micro aerial vehicle. *Journal of Field Robotics*, 33(4):431–450, 2016.

[66] Chen Feng and Vineet R Kamat. Augmented reality markers as spatial indices for indoor mobile aecfm applications. In *Proceedings of 12th international conference on construction applications of virtual reality (CONVR 2012)*, pages 235–24, 2012.

[67] Eric Feron and Eric N Johnson. Aerial robotics. *Springer Handbook of Robotics*, pages 1009–1029, 2008.

[68] Mark Fiala. Artag, a fiducial marker system using digital techniques. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 590–596. IEEE, 2005.

[69] Mark Fiala. Comparing artag and artoolkit plus fiducial marker systems. In *IEEE International Workshop on Haptic Audio Visual Environments and their Applications*, pages 6–pp. IEEE, 2005.

[70] Paul Filitchkin. *Visual terrain classification for legged robots*. PhD thesis, University of California, Santa Barbara, 2011.

[71] Paul Filitchkin and Katie Byl. Feature-based terrain classification for littledog. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 1387–1392. IEEE, 2012.

[72] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[73] Andrei George Florea and Catalin Buiu. Sensor fusion for autonomous drone waypoint navigation using ros and numerical p systems: A critical analysis of its advantages and limitations. In *2019 22nd International Conference on Control Systems and Computer Science (CSCS)*, pages 112–117. IEEE, 2019.

[74] Christian Forster, Matia Pizzoli, and Davide Scaramuzza. Svo: Fast semi-direct monocular visual odometry. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 15–22. IEEE, 2014.

[75] Eric Frew, Tim McGee, ZuWhan Kim, Xiao Xiao, Stephen Jackson, Michael Morimoto, Sivakumar Rathinam, Jose Padial, and Raja Sengupta. Vision-based road-following using a small autonomous aircraft. In *Aerospace Conference, 2004. Proceedings. 2004 IEEE*, volume 5, pages 3006–3015. IEEE, 2004.

[76] John G Fryer and Duane C Brown. Lens distortion for close-range photogrammetry. *Photogrammetric engineering and remote sensing*, 52(1):51–58, 1986.

[77] C Fuchs, F Neuhaus, and D Paulus. 3d pose estimation for articulated vehicles using kalman-filter based tracking. *Pattern Recognition and Image Analysis*, 26(1):109–113, 2016.

[78] Katsuhisa Furuta. Sliding mode control of a discrete system. *Systems & Control Letters*, 14(2):145–152, 1990.

[79] Dhiraj Gandhi, Lerrel Pinto, and Abhinav Gupta. Learning to fly by crashing. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3948–3955. IEEE, 2017.

[80] Adriano Garcia, Edward Mattison, and Kanad Ghose. High-speed vision-based autonomous indoor navigation of a quadcopter. In *2015 international conference on unmanned aircraft systems (ICUAS)*, pages 338–347. IEEE, 2015.

[81] Olivier Gascuel and Andy McKenzie. Performance analysis of hierarchical clustering algorithms. *Journal of classification*, 21(1):3–18, 2004.

[82] Mingyang Geng, Shuqi Liu, and Zhaoxia Wu. Sensor fusion-based cooperative trail following for autonomous multi-robot system. *Sensors*, 19(4):823, 2019.

[83] Alessandro Giusti, Jérôme Guzzi, Dan C Cireşan, Fang-Lin He, Juan P Rodríguez, Flavio Fontana, Matthias Faessler, Christian Forster, Jürgen Schmidhuber, Gianni Di Caro, et al. A machine learning approach to visual perception of forest trails for mobile robots. *IEEE Robotics and Automation Letters*, 1(2):661–667, 2015.

[84] HiSystems GmbH. MikroKopter basic mk octo v3, 2008.

[85] Shweta Gupte, Paul Infant Teenu Mohandas, and James M Conrad. A survey of quadrotor unmanned aerial vehicles. In *Southeastcon, 2012 Proceedings of IEEE*, pages 1–6. IEEE, 2012.

[86] Kale Harbick, James F Montgomery, and G Sukhatme. Planar spline trajectory following for an autonomous helicopter. In *Computational Intelligence in Robotics and Automation, 2001. Proceedings 2001 IEEE International Symposium on*, pages 408–413. IEEE, 2001.

[87] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, page 50. Manchester, UK, 1988.

[88] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[89] Zhihai He, Ram Venkataraman Iyer, and Phillip R Chandler. Vision-based uav flight control and obstacle avoidance. In *American Control Conference, 2006*, pages 5–pp. IEEE, 2006.

[90] Ahmad Hemami and Ferri Hassani. An overview of autonomous loading of bulk material. In *26th International Symposium on Automation and Robotics in Construction*, pages 405–411, 2009.

[91] H Hemami. Study of bucket trajectory in automatic scooping with lhd loaders. *Transactions Institution Mining, Metallurgy*, 1992.

[92] Lionel Heng, Gim Hee Lee, Friedrich Fraundorfer, and Marc Pollefeys. Real-time photo-realistic 3d mapping for micro aerial vehicles. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 4012–4019. IEEE, 2011.

[93] Heiko Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on pattern analysis and machine intelligence*, 30(2):328–341, 2007.

[94] Gabriel M Hoffmann, Haomiao Huang, Steven L Waslander, and Claire J Tomlin. Quadrotor helicopter flight dynamics and control: Theory and experiment. In *Proc. of the AIAA Guidance, Navigation, and Control Conference*, volume 2, 2007.

[95] Tsai Hong Hong, Christopher Rasmussen, Tommy Chang, and Michael Shneier. Road detection and tracking for autonomous mobile robots. In *AeroSense 2002*, pages 311–319. International Society for Optics and Photonics, 2002.

[96] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[97] Ahmed Hussein, Abdulla Al-Kaff, Arturo de la Escalera, and Jose Maria Armingol. Autonomous indoor navigation of low-cost quadcopters. In *2015 IEEE international conference on service operations and logistics, and informatics (SOLI)*, pages 133–138. IEEE, 2015.

[98] Mariano Jaimez, Javier G Monroy, and Javier Gonzalez-Jimenez. Planar odometry from a radial laser scanner. a range flow-based approach. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4479–4485. IEEE, 2016.

[99] Paul Jansen, Wannes van der Mark, Johan C van den Heuvel, and Frans CA Groen. Colour based off-road environment and terrain type classification. In *Intelligent Transportation Systems, 2005. Proceedings. 2005 IEEE*, pages 216–221. IEEE, 2005.

[100] Hu Jiaxin, Guo Yanning, Feng Zhen, and Guo Yuqing. Vision-based autonomous landing of unmanned aerial vehicles. In *2017 Chinese Automation Congress (CAC)*, pages 3464–3469. IEEE, 2017.

[101] Hailin Jin, Paolo Favaro, and Stefano Soatto. A semi-direct approach to structure from motion. *The Visual Computer*, 19(6):377–394, 2003.

[102] Pengju Jin, Pyry Matikainen, and Siddhartha S Srinivasa. Sensor fusion for fiducial tags: Highly robust pose estimation from single frame rgbd. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5770–5776. IEEE, 2017.

[103] Apoorva Joshi and Layak Ali. A detailed survey on auto irrigation system. In *2017 Conference on Emerging Devices and Smart Systems (ICEDSS)*, pages 90–95. IEEE, 2017.

[104] Il-Kyun Jung and Simon Lacroix. High resolution terrain mapping using low attitude aerial stereo imagery. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 946–951. IEEE, 2003.

[105] Gregory Kahn, Tianhao Zhang, Sergey Levine, and Pieter Abbeel. Plato: Policy learning using adaptive trajectory optimization. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3342–3349. IEEE, 2017.

[106] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. 1960.

[107] Hirokazu Kato and Mark Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Proceedings 2nd IEEE and ACM International Workshop on Augmented Reality (IWAR'99)*, pages 85–94. IEEE, 1999.

[108] Navid Kayhani, Adam Heins, Wenda Zhao, Mohammad Nahangi, Brenda McCabe, and Angela P Schoelligb. Improved tag-based indoor localization of uavs using extended kalman filter. In *Proceedings of the ISARC. International Symposium on Automation and Robotics in Construction, Banff, AB, Canada*, pages 21–24, 2019.

[109] Yasir Niaz Khan, Philippe Komma, and Andreas Zell. High resolution visual terrain classification for outdoor robots. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 1014–1021. IEEE, 2011.

[110] Georg Klein and David Murray. Parallel tracking and mapping for small ar workspaces. In *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, pages 225–234. IEEE, 2007.

[111] Kurt Konolige and Motilal Agrawal. Frameslam: From bundle adjustment to real-time visual mapping. *Robotics, IEEE Transactions on*, 24(5):1066–1077, 2008.

[112] Tomáš Krajník, Vojtěch Vonásek, Daniel Fišer, and Jan Faigl. Ar-drone as a platform for robotic research and education. In *Research and Education in Robotics-EUROBOT 2011*, pages 172–186. Springer, 2011.

[113] Jean-François Lalonde, Nicolas Vandapel, Daniel F Huber, and Martial Hebert. Natural terrain classification using three-dimensional ladar data for ground robot mobility. *Journal of field robotics*, 23(10):839–861, 2006.

[114] Philip Lawrence and Edmund Atkinson. Deposition of fine sediments in irrigation canals. *Irrigation and drainage systems*, 12(4):371–385, 1998.

[115] CANH Le VAN, Cao XUAN CUONG, QUOC Nguyen, Tran TRUNG ANH, BUI Xuan-Nam, et al. Experimental investigation on the performance of dji phantom 4 rtk in the ppk mode for 3d mapping open-pit mines. *Inżynieria Mineralna*, 1(2):65–74, 2020.

[116] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[117] Taeyoung Lee, Koushil Sreenath, and Vijay Kumar. Geometric control of cooperating multiple quadrotor uavs with a suspended payload. In *52nd IEEE conference on decision and control*, pages 5510–5515. IEEE, 2013.

[118] J Gordon Leishman. *Principles of Helicopter Aerodynamics with CD Extra*. Cambridge university press, 2006.

[119] J Leonard. Mobile robot lacalization by tracking geometric beacons. *IEEE Trans. Robot. Autom.*, 7(3):89–97, 1991.

[120] Juan Li, Maarten Slembrouck, Francis Deboeverie, Ana M Bernardos, Juan A Besada, Peter Veelaert, Hamid Aghajan, Wilfried Philips, and José R Casar. A hybrid pose tracking approach for handheld augmented reality. In *Proceedings of the 9th International Conference on Distributed Smart Cameras*, pages 7–12, 2015.

[121] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[122] Yi Lin, Fei Gao, Tong Qin, Wenliang Gao, Tianbo Liu, William Wu, Zhenfei Yang, and Shaojie Shen. Autonomous aerial navigation using monocular visual-inertial fusion. *Journal of Field Robotics*, 35(1):23–51, 2018.

[123] Kevin Ling, Derek Chow, Arun Das, and Steven L Waslander. Autonomous maritime landings for low-cost vtol aerial vehicles. In *2014 Canadian Conference on Computer and Robot Vision*, pages 32–39. IEEE, 2014.

[124] Lennart Ljung. Asymptotic behavior of the extended kalman filter as a parameter estimator for linear systems. *Automatic Control, IEEE Transactions on*, 24(1):36–50, 1979.

[125] Diego Lo, Paulo RS Mendonça, Andy Hopper, et al. Trip: A low-cost vision-based location system for ubiquitous computing. *Personal and Ubiquitous Computing*, 6(3):206–219, 2002.

[126] Antonio Loquercio, Ana I Maqueda, Carlos R Del-Blanco, and Davide Scaramuzza. Dronet: Learning to fly by driving. *IEEE Robotics and Automation Letters*, 3(2):1088–1095, 2018.

[127] David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. IEEE, 1999.

[128] Jacobo Jiménez Lugo and Andreas Zell. Framework for autonomous on-board navigation with the ar. drone. *Journal of Intelligent & Robotic Systems*, 73(1):401–412, 2014.

[129] Teppo Luukkonen. Modelling and control of quadcopter. *Independent research project in applied mathematics, Espoo*, 22:22, 2011.

[130] Tarek Madani and Abdelaziz Benallegue. Backstepping control for a quadrotor helicopter. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 3255–3260. IEEE, 2006.

[131] Babak Majidi and Alireza Bab-Hadiashar. Real time aerial natural image interpretation for autonomous ranger drone navigation. In *Digital Image Computing: Techniques and Applications (DICTA'05)*, pages 65–65. IEEE, 2005.

[132] Roberto Manduchi. Bayesian fusion of color and texture segmentations. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 2, pages 956–962. IEEE, 1999.

[133] Roberto Manduchi, Andres Castano, Ashit Talukder, and Larry Matthies. Obstacle detection and terrain classification for autonomous off-road navigation. *Autonomous robots*, 18(1):81–102, 2005.

[134] Sina Sharif Mansouri, Christoforos Kanellakis, George Georgoulas, and George Nikolakopoulos. Towards mav navigation in underground mine using deep learning. In *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 880–885. IEEE, 2018.

[135] Michael Manz, Michael Himmelsbach, Thorsten Luettel, and Hans-Joachim Wuensche. Fusing lidar and vision for autonomous dirt road following. In *Autonome Mobile Systeme 2009*, pages 17–24. Springer, 2009.

[136] Michael Manz, Felix von Hundelshausen, and H-J Wuensche. A hybrid estimation approach for autonomous dirt road following using multiple clothoid segments. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 2410–2415. IEEE, 2010.

[137] Janille Maria Maragh. *Dynamic tracking with AprilTags for robotic education*. PhD thesis, Massachusetts Institute of Technology, 2013.

[138] Donald W Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the society for Industrial and Applied Mathematics*, 11(2):431–441, 1963.

[139] Kimberly McGuire, Guido De Croon, Christophe De Wagter, Karl Tuyls, and Hilbert Kappen. Efficient optical flow and stereo vision for velocity estimation and obstacle avoidance on an autonomous pocket drone. *IEEE Robotics and Automation Letters*, 2(2):1070–1076, 2017.

[140] Arnold Meijster, Jos BTM Roerdink, and Wim H Hesselink. A general algorithm for computing distance transforms in linear time. In *Mathematical Morphology and its applications to image and signal processing*, pages 331–340. Springer, 2000.

[141] Daniel Mellinger and Vijay Kumar. Minimum snap trajectory generation and control for quadrotors. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2520–2525. IEEE, 2011.

[142] Daniel Mellinger, Nathan Michael, and Vijay Kumar. Trajectory generation and control for precise aggressive maneuvers with quadrotors. *The International Journal of Robotics Research*, page 0278364911434236, 2012.

[143] Daniel Mellinger, Michael Shomin, and Vijay Kumar. Control of quadrotors for robust perching and landing. In *Proceedings of the International Powered Lift Conference*, pages 205–225, 2010.

[144] Daniel Mellinger, Michael Shomin, Nathan Michael, and Vijay Kumar. Cooperative grasping and transport using multiple quadrotors. In *Distributed autonomous robotic systems*, pages 545–558. Springer, 2013.

[145] Nathan Michael, Shaojie Shen, Kartik Mohta, Yash Mulgaonkar, Vijay Kumar, Keiji Nagatani, Yoshito Okada, Seiga Kiribayashi, Kazuki Otake, Kazuya Yoshida, et al. Collaborative mapping of an earthquake-damaged building via ground and aerial robots. *Journal of Field Robotics*, 29(5):832–841, 2012.

[146] Thomas B Moeslund, Adrian Hilton, and Volker Krüger. A survey of advances in vision-based human motion capture and analysis. *Computer vision and image understanding*, 104(2-3):90–126, 2006.

[147] Elias Mueggler, Matthias Faessler, Flavio Fontana, and Davide Scaramuzza. Aerial-guided navigation of a ground robot among movable obstacles. In *2014 IEEE International Symposium on Safety, Security, and Rescue Robotics (2014)*, pages 1–8. IEEE, 2014.

[148] Abubakr Muhammad, Ali Ahmad, Saad Hassan, Syed M. Abbas, Talha Manzoor, and Karsten Berns. Using unmanned aerial vehicles to inspect siltation in irrigation canals.

[149] Mark Muller, Sergei Lupashin, and Raffaello D'Andrea. Quadrocopter ball juggling. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 5113–5120. IEEE, 2011.

[150] Ana Cris Murillo, José Jesús Guerrero, and C Sagues. Surf features for efficient robot localization with omnidirectional images. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 3901–3907. IEEE, 2007.

[151] Frank Neitzel and J Klonowski. Mobile 3d mapping with a low-cost uav system. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci*, 38:1–6, 2011.

[152] Derek R Nelson, D Blake Barber, Timothy W McLain, and Randal W Beard. Vector field path following for miniature air vehicles. *Robotics, IEEE Transactions on*, 23(3):519–529, 2007.

[153] Francesco Nex and Fabio Remondino. Uav for 3d mapping applications: a review. *Applied Geomatics*, 6(1):1–15, 2014.

[154] Christian Nissler and Zoltan-Csaba Marton. Robot-to-camera calibration: a generic approach using 6d detections. In *2017 First IEEE International Conference on Robotic Computing (IRC)*, pages 299–302. IEEE, 2017.

[155] Andreas Nüchter, Kai Lingemann, Joachim Hertzberg, and Hartmut Surmann. 6d slam—3d mapping outdoor environments. *Journal of Field Robotics*, 24(8-9):699–722, 2007.

[156] Aníbal Ollero and Luís Merino. Control and perception techniques for aerial robotics. *Annual reviews in Control*, 28(2):167–178, 2004.

[157] Edwin Olson. Apriltag: A robust and flexible visual fiducial system. In *2011 IEEE International Conference on Robotics and Automation*, pages 3400–3407. IEEE, 2011.

[158] Charles B Owen, Fan Xiao, and Paul Middlin. What is the best fiducial? In *The First IEEE International Workshop Agumented Reality Toolkit,*, pages 8–pp. IEEE, 2002.

[159] Yanwei Pang, Wei Li, Yuan Yuan, and Jing Pan. Fully affine invariant surf for image matching. *Neurocomputing*, 85:6–10, 2012.

[160] Christos Papachristos, Mina Kamel, Marija Popović, Shehryar Khattak, Andreas Bircher, Helen Oleynikova, Tung Dang, Frank Mascarich, Kostas Alexis, and Roland Siegwart. Autonomous exploration and inspection path planning for aerial robots using the robot operating system. In *Robot Operating System (ROS)*, pages 67–111. Springer, 2019.

[161] Steven A Parkison, Eric T Psota, and Lance C Pérez. Automated indoor rfid inventorying using a self-guided micro-aerial vehicle. In *IEEE International Conference on Electro/Information Technology*, pages 335–340. IEEE, 2014.

[162] Jeff Plungis. Driving into the future. *Consumer Reports*, 82(4):10–16, 2017.

[163] Paul Pounds, Robert Mahony, and Peter Corke. Modelling and control of a large quadrotor robot. *Control Engineering Practice*, 18(7):691–699, 2010.

[164] Paul EI Pounds, Daniel R Bersak, and Aaron M Dollar. Stability of small-scale uav helicopters and quadrotors with added payload mass under pid control. *Autonomous Robots*, 33(1-2):129–142, 2012.

[165] Raymond W Prouty. *Helicopter performance, stability, and control*. 1995.

[166] Tim Puls, Markus Kemper, R Kuke, and Andreas Hein. Gps-based position control and waypoint navigation system for quadrocopters. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 3374–3379. IEEE, 2009.

[167] Siddharth Raina, Huan-Yang Chang, Sambuddha Sarkar, Man-Ning Chen, and Yiqing Cai. An integrated system for 3d pose estimation in cluttered environments.

[168] Ernesto Alejandro Ramirez. *An experimental study of mobile device localization*. PhD thesis, Massachusetts Institute of Technology, 2015.

[169] Benjamin Ranft, Jean-Luc Dugelay, and Ludovic Apvrille. 3d perception for autonomous navigation of a low-cost mav using minimal landmarks. In *Proceedings of the International Micro Air Vehicle Conference and Flight Competition (IMAV2013), Toulouse, France*, volume 1720, 2013.

[170] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*, pages 63–71. Springer, 2003.

[171] Giulio Reina, Andres Vargas, Keiji Nagatani, and Kazuya Yoshida. Adaptive kalman filtering for gps-based mobile robot localization. In *2007 IEEE International Workshop on Safety, Security and Rescue Robotics*, pages 1–6. IEEE, 2007.

[172] Bart Remes, Dino Hensen, Freek Van Tienen, Christophe De Wagter, Erik Van der Horst, and GCHE De Croon. Paparazzi: how to make a swarm of parrot ar drones fly autonomously based on gps. In *International Micro Air Vehicle Conference and Flight Competition (IMAV2013)*, pages 17–20, 2013.

[173] Riccardo Rocca. Low cost 3d mapping using a commercial drone/uav: Application in structural geology. In *International conference and exhibition, Barcelona, Spain*, pages 3–6, 2016.

[174] Stéphane Ross, Narek Melik-Barkhudarov, Kumar Shaurya Shankar, Andreas Wendel, Debadeepta Dey, J Andrew Bagnell, and Martial Hebert. Learning monocular reactive uav control in cluttered natural environments. In *2013 IEEE international conference on robotics and automation*, pages 1765–1772. IEEE, 2013.

[175] Rolf Rysdyk. Uav path following for constant line-of-sight. In *2th AIAA Unmanned Unlimited. Conf. and Workshop and Exhibit, San Diego, CA*, 2003.

[176] Inkyu Sa and Peter Corke. 100hz onboard vision for quadrotor state estimation. In *Australasian Conference on Robotics and Automation*, 2012.

[177] Inkyu Sa, Hu He, Van Huynh, and Peter Corke. Monocular vision based autonomous navigation for a cost-effective mav in gps-denied environments. In *2013 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 1355–1360. IEEE, 2013.

[178] Artur Sagitov, Ksenia Shabalina, Roman Lavrenov, and Evgeni Magid. Comparing fiducial marker systems in the presence of occlusion. In *2017 International Conference on Mechanical, System and Control Engineering (ICMSC)*, pages 377–382. IEEE, 2017.

[179] Lucas Vago Santana, Alexandre Santos Brandao, and Mario Sarcinelli-Filho. Outdoor waypoint navigation with the ar. drone quadrotor. In *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 303–311. IEEE, 2015.

[180] Martin Saska, Tomáš Krajník, Jan Faigl, Vojtěch Vonásek, and Libor Přeučil. Low cost mav platform ar-drone in experimental verifications of methods for vision based autonomous navigation. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4808–4809. IEEE, 2012.

[181] Sebastian Scherer, Joern Rehder, Supreeth Achar, Hugh Cover, Andrew Chambers, Stephen Nuske, and Sanjiv Singh. River mapping from a flying robot: state estimation, river detection, and obstacle mapping. *Autonomous Robots*, 33(1-2):189–214, 2012.

[182] X Shi and R Manduchi. A study on bayes feature fusion for image classification. In *Computer Vision and Pattern Recognition Workshop, 2003. CVPRW'03. Conference on*, volume 8, pages 95–95. IEEE, 2003.

[183] Ken Shoemake. Euler angle conversion. *Graphics gems IV*, pages 222–229, 1994.

[184] Geraldo Silveira, Ezio Malis, and Patrick Rives. An efficient direct approach to visual slam. *Robotics, IEEE Transactions on*, 24(5):969–979, 2008.

[185] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[186] Bruno Sinopoli, Mario Micheli, Gianluca Donato, and Tak-John Koo. Vision based navigation for an unmanned aerial vehicle. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 2, pages 1757–1764. IEEE, 2001.

[187] Nikolai Smolyanskiy, Alexey Kamenev, Jeffrey Smith, and Stan Birchfield. Toward low-flying autonomous mav trail navigation using deep neural networks for environmental awareness. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4241–4247. IEEE, 2017.

[188] Anthony Stentz, John Bares, Sanjiv Singh, and Patrick Rowe. A robotic excavator for autonomous truck loading. *Autonomous Robots*, 7(2):175–186, 1999.

[189] Wieslaw Zenon Stepniewski. *Rotary-wing aerodynamics*. Courier Corporation, 1984.

[190] JP Stewart, GM Podger, MD Ahmad, MA Shah, H Bodla, Z Khero, and MKI Rana. Indus river system model (irsm)–a planning tool to explore water management options in pakistan: model conceptualisation, configuration and calibration. Technical report, August 2018. Technical report. South Asia Sustainable Development Investment . . . , 2018.

[191] Hauke Strasdat, JMM Montiel, and Andrew J Davison. Real-time monocular slam: Why filter? In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 2657–2664. IEEE, 2010.

[192] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[193] Dengqing Tang, Tianjiang Hu, Lincheng Shen, Zhaowei Ma, and Congyu Pan. April-tag array-aided extrinsic calibration of camera–laser multi-sensor system. *Robotics and biomimetics*, 3(1):1–9, 2016.

[194] Sergios Theodoridis, Aggelos Pikrakis, Konstantinos Koutroumbas, and Dionisis Cavouras. *Introduction to Pattern Recognition: A Matlab Approach: A Matlab Approach*. Access Online via Elsevier, 2010.

[195] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. Probabilistic robotics. 2005. *Massachusetts Institute of Technology, USA*, 2005.

[196] Sebastian Thrun, Mike Montemerlo, Hendrik Dahlkamp, David Stavens, Andrei Aron, James Diebel, Philip Fong, John Gale, Morgan Halpenny, Gabriel Hoffmann, et al. Stanley: The robot that won the darpa grand challenge. In *The 2005 DARPA Grand Challenge*, pages 1–43. Springer, 2007.

[197] Jana Tumova, Gavin C Hall, Sertac Karaman, Emilio Frazzoli, and Daniela Rus. Least-violating control strategy synthesis with safety rules. In *Proceedings of the 16th international conference on Hybrid systems: computation and control*, pages 1–10, 2013.

[198] Matthew Turpin, Nathan Michael, and Vijay Kumar. Trajectory design and control for aggressive formation flight with quadrotors. *Autonomous Robots*, 33(1-2):143–156, 2012.

[199] Tinne Tuytelaars and Krystian Mikolajczyk. Local invariant feature detectors: a survey. *Foundations and Trends® in Computer Graphics and Vision*, 3(3):177–280, 2008.

[200] Daniel Wagner and Dieter Schmalstieg. Artoolkitplus for pose tracking on mobile devices. 2007.

[201] John Wang and Edwin Olson. Apriltag 2: Efficient and robust fiducial detection. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4193–4198. IEEE, 2016.

[202] Ju Wang, Chad Sadler, Cesar Flores Montoya, and Jonathan CL Liu. Optimizing ground vehicle tracking using unmanned aerial vehicle and embedded apriltag design. In *2016 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 739–744. IEEE, 2016.

[203] Kangli Wang, Swee King Phang, Yijie Ke, Xudong Chen, Kehong Gong, and Ben M Chen. Vision-aided tracking of a moving ground vehicle with a hybrid uav. In *2017 13th IEEE International Conference on Control & Automation (ICCA)*, pages 28–33. IEEE, 2017.

[204] Wanli Wang, Botao Zhang, Kaiqi Wu, Sergey A Chepinskiy, Anton A Zhilenkov, Sergei Chernyi, and Aleksandr Yu Krasnov. A visual terrain classification method for mobile robots' navigation based on convolutional neural network and support vector machine. *Transactions of the Institute of Measurement and Control*, 44(4):744–753, 2022.

[205] Steven Lake Waslander, Gabriel M Hoffmann, Jung Soon Jang, and Claire J Tomlin. Multi-agent quadrotor testbed control design: Integral sliding mode vs. reinforcement learning. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3712–3717. IEEE, 2005.

[206] Stephan Weiss, Markus W Achtelik, Simon Lynen, Michael C Achtelik, Laurent Kneip, Margarita Chli, and Roland Siegwart. Monocular vision for long-term micro aerial vehicle state estimation: A compendium. *Journal of Field Robotics*, 30(5):803–831, 2013.

[207] Willow-Garage. Open source computer vision library. *URL http://opencv. willowgarage. com. Retrieved May*, 2012.

[208] A.J. Winship, Salvador Jorgensen, Scott Shaffer, Ian Jonsen, Patrick Robinson, Daniel Costa, and Barbara Block. State-space framework for estimating measurement error from double-tagging telemetry experiments. *Methods in Ecology and Evolution*, 3, 01 2011.

[209] Allen D Wu, Eric N Johnson, and Alison A Proctor. Vision-aided inertial navigation for flight control. *Journal of Aerospace Computing, Information, and Communication*, 2(9):348–360, 2005.

[210] Kai M Wurm, Armin Hornung, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. Octomap: A probabilistic, flexible, and compact 3d map representation for robotic systems. In *Proc. of the ICRA 2010 workshop on best practice in 3D perception and modeling for mobile manipulation*, volume 2, 2010.

[211] Yuanfan Xie, Rui Shao, Popo Guli, Bo Li, and Liang Wang. Infrastructure based calibration of a multi-camera and multi-lidar system using apriltags. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 605–610. IEEE, 2018.

[212] Anqi Xu and Gregory Dudek. Fourier tag: A smoothly degradable fiducial marker system with configurable payload capacity. In *2011 Canadian Conference on Computer and Robot Vision*, pages 40–47. IEEE, 2011.

[213] Rong Xu and Umit Ozguner. Sliding mode control of a quadrotor helicopter. In *Decision and Control, 2006 45th IEEE Conference on*, pages 4957–4962. IEEE, 2006.

[214] Zane Zake, Stéphane Caro, Adolfo Suarez Roos, François Chaumette, and Nicolò Pedemonte. Stability analysis of pose-based visual servoing control of cable-driven parallel robots. In *International Conference on Cable-Driven Parallel Robots*, pages 73–84. Springer, 2019.

[215] Lamei Zhang, Zexi Chen, Bin Zou, and Ye Gao. Polarimetric sar terrain classification using 3d convolutional neural network. In *IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium*, pages 4551–4554. IEEE, 2018.

[216] Yongwei Zhang, Yangguang Yu, Shengde Jia, and Xiangke Wang. Autonomous landing on ground target of uav by using image-based visual servo control. In *2017 36th Chinese Control Conference (CCC)*, pages 11204–11209. IEEE, 2017.

[217] Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces. *International journal of computer vision*, 13(2):119–152, 1994.

[218] Guo Zhenglong, Fu Qiang, and Quan Quan. Pose estimation for multicopters based on monocular vision and apriltag. In *2018 37th Chinese Control Conference (CCC)*, pages 4717–4722. IEEE, 2018.

[219] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3357–3364. IEEE, 2017.

[220] Ilan Zohar, Amit Ailon, and Hugo Guterman. An automatic stabilization system for quadrotors with applications to vertical take-off and landing. In *AUVSI-IEEE Joint Conference, Israel*, 2010.

Vita

Syed Muhammad Abbas was born on 16th October 1984 in Lahore. He completed his schooling up till matriculation from the Divisional Public School Model Town Lahore in 2000. Then he did his intermediate from Government College Lahore (now Government College University Lahore) in 2002. After that, he graduated with a BS in Computer Science from the National University of Computer and Emerging Sciences (FAST-Lahore). After graduation, he started building software solutions for the industry and establish a software house that continues its operations till the year 2010. In 2013, he completed his MS in Computer Science from Lahore University of Management Sciences (LUMS) and enrolled as a Ph.D. student with the Electrical Engineering Department in the university. During his Ph.D., he spent time as a visiting researcher at Robotics Research Lab (RRLab) at the Technical University of Kaiserslautern from the year 2013 till the year 2019. His research areas are localization, 3D mapping, and autonomous navigation. Recently in 2021, he has founded a robotics startup company that is currently serving both national and international clients.

Abbas-PhD-Thesis.pdf

ORIGINALITY REPORT

7%	4%	2%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to University of Hong Kong Student Paper	1%
2	Submitted to Higher Education Commission Pakistan Student Paper	1%
3	www.researchgate.net Internet Source	<1%
4	mafiadoc.com Internet Source	<1%
5	ifac.papercept.net Internet Source	<1%
6	web.wpi.edu Internet Source	<1%
7	tel.archives-ouvertes.fr Internet Source	<1%
8	livrepository.liverpool.ac.uk Internet Source	<1%
9	Submitted to Imperial College of Science, Technology and Medicine	<1%

	Student Paper	
10	e-archivo.uc3m.es Internet Source	<1 %
11	uwspace.uwaterloo.ca Internet Source	<1 %
12	"Intelligent Autonomous Systems 13", Springer Science and Business Media LLC, 2016 Publication	<1 %
13	Syed Muhammad Abbas, Abubakr Muhammad, Syed Atif Mehdi, Karsten Berns. "Improvements in accuracy of single camera terrain classification", 2013 16th International Conference on Advanced Robotics (ICAR), 2013 Publication	<1 %
14	agrosy.cs.uni-kl.de Internet Source	<1 %
15	210.72.131.170 Internet Source	<1 %
16	drum.lib.umd.edu Internet Source	<1 %
17	Springer Tracts in Advanced Robotics, 2015. Publication	<1 %
18	Submitted to University of Sheffield Student Paper	<1 %

19	dr.ntu.edu.sg Internet Source	<1 %
20	frc.ri.cmu.edu Internet Source	<1 %
21	Sebastian Scherer, Joern Rehder, Supreeth Achar, Hugh Cover, Andrew Chambers, Stephen Nuske, Sanjiv Singh. "River mapping from a flying robot: state estimation, river detection, and obstacle mapping", Autonomous Robots, 2012 Publication	<1 %
22	Submitted to National University of Singapore Student Paper	<1 %
23	Springer Tracts in Advanced Robotics, 2014. Publication	<1 %
24	eprints.qut.edu.au Internet Source	<1 %
25	Submitted to University of Central Florida Student Paper	<1 %
26	arxiv.org Internet Source	<1 %
27	rpg.ifi.uzh.ch Internet Source	<1 %
28	esajournals.onlinelibrary.wiley.com Internet Source	<1 %

29	www.scribd.com Internet Source	<1 %
30	idus.us.es Internet Source	<1 %
31	www.dtic.mil Internet Source	<1 %
32	Pfeiffer, David(Eisert, Peter, Franke, Uwe and Reulke, Ralf). "The Stixel World", Mathematisch-Naturwissenschaftliche Fakultät II, 2012. Publication	<1 %
33	hdl.handle.net Internet Source	<1 %
34	www.diplomarbeiten24.de Internet Source	<1 %
35	nuxeo.edel.univ-poitiers.fr Internet Source	<1 %
36	opus.lib.uts.edu.au Internet Source	<1 %
37	ansatte.hin.no Internet Source	<1 %
38	apps.dtic.mil Internet Source	<1 %
39	hal.archives-ouvertes.fr Internet Source	<1 %

40	link.springer.com Internet Source	<1 %
41	oa.upm.es Internet Source	<1 %
42	web.mit.edu Internet Source	<1 %
43	www.seas.upenn.edu Internet Source	<1 %
44	www.zora.uzh.ch Internet Source	<1 %
45	Submitted to KDU College Sdn Bhd Student Paper	<1 %
46	R. Sengupta. "Vision-based road-following using a small autonomous aircraft", 2004 IEEE Aerospace Conference Proceedings (IEEE Cat No 04TH8720) AERO-04, 2004 Publication	<1 %
47	archive.org Internet Source	<1 %
48	biblio.ugent.be Internet Source	<1 %
49	d-nb.info Internet Source	<1 %
50	dokumen.pub Internet Source	<1 %

51	dspace.jaist.ac.jp Internet Source	<1 %
52	pure.port.ac.uk Internet Source	<1 %
53	repository.tudelft.nl Internet Source	<1 %
54	www.es.ewi.tudelft.nl Internet Source	<1 %
55	www.iros2011.org Internet Source	<1 %
56	"Advances in Mechanism and Machine Science", Springer Science and Business Media LLC, 2019 Publication	<1 %
57	Submitted to Wright State University Student Paper	<1 %
58	dar.aucegypt.edu Internet Source	<1 %
59	Abdelaziz Benallegue. "Backstepping control with exact 2-sliding mode estimation for a quadrotor unmanned aerial vehicle", 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, 10/2007 Publication	<1 %

60	Submitted to International Islamic University Malaysia Student Paper	<1%
61	Submitted to University of Newcastle Student Paper	<1%
62	www.docstoc.com Internet Source	<1%
63	"Machine Vision and Navigation", Springer Science and Business Media LLC, 2020 Publication	<1%
64	Alexis, K.. "Switching model predictive attitude control for a quadrotor helicopter subject to atmospheric disturbances", Control Engineering Practice, 201110 Publication	<1%
65	Submitted to King Fahd University for Petroleum and Minerals Student Paper	<1%
66	Springer Tracts in Advanced Robotics, 2016. Publication	<1%
67	deepblue.lib.umich.edu Internet Source	<1%
68	docs.lib.purdue.edu Internet Source	<1%
69	dspace.library.uvic.ca:8080 Internet Source	<1%

70	publications.theseus.fi Internet Source	<1 %
71	scholar.afit.edu Internet Source	<1 %
72	www.lirneasia.net Internet Source	<1 %
73	Submitted to Associatie K.U.Leuven Student Paper	<1 %
74	encore.lib.gla.ac.uk Internet Source	<1 %
75	eprints-phd.biblio.unitn.it Internet Source	<1 %
76	ww3.comsats.edu.pk Internet Source	<1 %
77	www.cs.utexas.edu Internet Source	<1 %
78	www.dgk.badw.de Internet Source	<1 %
79	www.dsl.mei.titech.ac.jp Internet Source	<1 %
80	Ting Shu, Jin He, Trieu-Kien Truong. "Particle-Velocity Coarray Augmentation For Direction Finding with Acoustic Vector Sensors",	<1 %

	Institute of Electrical and Electronics Engineers (IEEE), 2022 Publication	
81	cdn.intechopen.com Internet Source	<1 %
82	cogx.eu Internet Source	<1 %
83	digital.lib.washington.edu Internet Source	<1 %
84	os.zhdk.cloud.switch.ch Internet Source	<1 %
85	tgs.freshpatents.com Internet Source	<1 %
86	th.wukihow.com Internet Source	<1 %
87	tobias-lib.uni-tuebingen.de Internet Source	<1 %
88	tuprints.ulb.tu-darmstadt.de Internet Source	<1 %
89	uknowledge.uky.edu Internet Source	<1 %
90	www.duo.uio.no Internet Source	<1 %
91	www.research-collection.ethz.ch Internet Source	<1 %

92	www.science.uva.nl Internet Source	<1 %
93	Publication	<1 %
94	"Book of abstracts", 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2012 Publication	<1 %
95	"Computer Analysis of Images and Patterns", Springer Science and Business Media LLC, 2015 Publication	<1 %
96	Denuelle, Aymeric, Saul Thurrowgood, Reuben Strydom, Farid Kendoul, and Mandyam V. Srinivasan. "Biologically-inspired visual stabilization of a rotorcraft UAV in unknown outdoor environments", 2015 International Conference on Unmanned Aircraft Systems (ICUAS), 2015. Publication	<1 %
97	Jakob Engel, Jurgen Sturm, Daniel Cremers. "Camera-based navigation of a low-cost quadrocopter", 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2012 Publication	<1 %

98	Ji Zhang, Sanjiv Singh. "Visual-lidar odometry and mapping: low-drift, robust, and fast", 2015 IEEE International Conference on Robotics and Automation (ICRA), 2015 Publication	<1 %
99	Lecture Notes in Computer Science, 2014. Publication	<1 %
100	Lecture Notes in Computer Science, 2015. Publication	<1 %
101	Yuheng Yan, Yiqiu Liang, Zihan Zhou, Bin Jiang, Jian Xiao. "FastQR: Fast Pose Estimation of Objects Based on Multiple QR Codes and Monocular Vision in Mobile Embedded Devices", Wireless Communications and Mobile Computing, 2021 Publication	<1 %
102	backend.orbit.dtu.dk Internet Source	<1 %
103	bitsavers.trailing-edge.com Internet Source	<1 %
104	elib.suub.uni-bremen.de Internet Source	<1 %
105	gaowenliang.github.io Internet Source	<1 %
106	hal.laas.fr Internet Source	<1 %

107	orbilu.uni.lu Internet Source	<1 %
108	papers.neurips.cc Internet Source	<1 %
109	propertibazar.com Internet Source	<1 %
110	syafii.staff.uns.ac.id Internet Source	<1 %
111	www.dis.uniroma1.it Internet Source	<1 %
112	www.ensta-bretagne.fr Internet Source	<1 %
113	www.nada.kth.se Internet Source	<1 %
114	"Informatics in Control, Automation and Robotics", Springer Science and Business Media LLC, 2020 Publication	<1 %
115	Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, Chrisina Jayne. "Imitation Learning", ACM Computing Surveys, 2017 Publication	<1 %
116	Alexander Schaub. "Robust Perception from Optical Sensors for Reactive Behaviors in	<1 %

Autonomous Robotic Vehicles", Springer Science and Business Media LLC, 2018
Publication

---

117 Alison Proctor, Eric Johnson. "Vision-Only Aircraft Flight Control Methods and Test Results", AIAA Guidance, Navigation, and Control Conference and Exhibit, 2004 <1 %
Publication

---

118 Magree, Daniel P., Gerald J.J. van Dalen, Stephen Haviland, and Eric N. Johnson. "Light-weight quadrotor with on-board absolute vision-aided navigation", 2015 International Conference on Unmanned Aircraft Systems (ICUAS), 2015. <1 %
Publication

---

119 Rui Fan, Hengli Wang, Peide Cai, Ming Liu. "SNE-RoadSeg: Incorporating Surface Normal Information into Semantic Segmentation for Accurate Freespace Detection", Institute of Electrical and Electronics Engineers (IEEE), 2020 <1 %
Publication

---

120 Tianguang Zhang, Wei Li, Markus Achtelik, Kolja Kuhnlenz, Martin Buss. "Multi-sensory motion estimation and control of a mini-quadrotor in an air-ground multi-robot system", 2009 IEEE International Conference on Robotics and Biomimetics (ROBIO), 2009 <1 %
Publication

121	Yoshihide Wada, Adriano Vinca, Simon Parkinson, Barbara A. Willaarts et al. "Co-designing Indus Water-Energy-Land Futures", One Earth, 2019 Publication	<1 %
122	addi.ehu.es Internet Source	<1 %
123	amsdottorato.unibo.it Internet Source	<1 %
124	conservancy.umn.edu Internet Source	<1 %
125	docplayer.net Internet Source	<1 %
126	dspace.library.uvic.ca Internet Source	<1 %
127	ecommons.usask.ca Internet Source	<1 %
128	etheses.whiterose.ac.uk Internet Source	<1 %
129	keep.lib.asu.edu Internet Source	<1 %
130	manualzz.com Internet Source	<1 %
131	mediatum.ub.tum.de Internet Source	

		<1 %
132	ourspace.uregina.ca Internet Source	<1 %
133	paduaresearch.cab.unipd.it Internet Source	<1 %
134	porto.polito.it Internet Source	<1 %
135	pure.tue.nl Internet Source	<1 %
136	repository.ntu.edu.sg Internet Source	<1 %
137	ro.uow.edu.au Internet Source	<1 %
138	scientiairanica.sharif.edu Internet Source	<1 %
139	sethares.engr.wisc.edu Internet Source	<1 %
140	soar-ir.repo.nii.ac.jp Internet Source	<1 %
141	tu-dresden.de Internet Source	<1 %
142	tutorsonspot.com Internet Source	<1 %

143	viejournal.springeropen.com Internet Source	<1 %
144	wrap.warwick.ac.uk Internet Source	<1 %
145	www.aero.psu.edu Internet Source	<1 %
146	www.amrita.edu Internet Source	<1 %
147	www.bmvc2021-virtualconference.com Internet Source	<1 %
148	www.cl.cam.ac.uk Internet Source	<1 %
149	www.frontiersin.org Internet Source	<1 %
150	www.tradr-project.eu Internet Source	<1 %
151	"Intelligent Autonomous Systems 14", Springer Science and Business Media LLC, 2017 Publication	<1 %
152	Suraj Bijjahalli, Roberto Sabatini, Alessandro Gardi. "Advances in intelligent and autonomous navigation systems for small UAS", Progress in Aerospace Sciences, 2020 Publication	<1 %

153 Martinez-Gomez, Jesus, Antonio Fernandez-Caballero, Ismael Garcia-Varea, Luis Rodriguez, and Cristina Romero-Gonzalez. "A Taxonomy of Vision Systems for Ground Mobile Robots", International Journal of Advanced Robotic Systems, 2014.
Publication

<1%

154 Mellinger, D., N. Michael, and V. Kumar. "Trajectory generation and control for precise aggressive maneuvers with quadrotors", The International Journal of Robotics Research, 2012.
Publication

<1%

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		