A. AL-QASSAB

LOCALIZATION AND PATH PLANNING FOR MULTIPLE AGENTS USING OPTI-TRACK CAMERAS

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
ATILIM UNIVERSITY

AYMAN AL-QASSAB

A MASTER OF SCIENCE THESIS
IN
THE DEPARTMENT OF MECHATRONICS ENGINEERING

SEPTEMBER 2021

ATILIM UNIVERSITY 2021

LOCALIZATION AND PATH PLANNING FOR MULTIPLE AGENTS USING OPTI-TRACK CAMERAS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
ATILIM UNIVERSITY

BY

AYMAN AL-QASSAB

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
THE DEPARTMENT OF MECHATRONICS ENGINEERING

SEPTEMBER 2021

Approval of the Graduate School of Natural and Applied Sciences, Atilim University.

_____

Prof. Dr. Ender KESKİNKILIÇ
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of **Master of Science Engineering in Mechatronics Atilim University**.

_____

Prof. Dr. Hulusi Bülent ERTAN
Head of Department

This is to certify that we have read the thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

_____	_____
Asst. Prof. Dr. Mohammad Hassan GOL MOHAMMADZADEH Co-Supervisor	Asst. Prof. Dr. Muhammad Umer KHAN Supervisor

Asst. Prof. Dr. Amir NOBAHAR SADEGHI NAM
Department of Mechatronics Engg., Atilim University_____

Asst. Prof. Dr. Masoud LATIFI NAVID
Department of Mechatronics Engg., THK University _____

Asst. Prof. Dr. Muhammad Umer KHAN
Department of Mechatronics Engg., Atilim University_____

Date: 20/9/2021

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: Ayman, AL-QASSAB

Signature:

ABSTRACT

LOCALIZATION AND PATH PLANNING FOR MULTIPLE AGENTS USING OPTI-TRACK CAMERAS

AL-QASSAB, Ayman
M.S., Department of Mechatronics Engineering
Supervisor: Asst. Prof. Dr. M. Umer Khan
Co-Supervisor: Asst. Prof. Dr. M. Hassan GOL MOHAMMADZADEH

September 2021, 57 pages

The process of digitally detecting and recording the movements of objects or living beings in a physical space is performed using motion capture (MoCap) system, such as Opti-Track system. In this study, the purpose of the motion capture system is to continuously determine the poses of both quadrotor and mobile platform. The location information is utilised by the navigation system to guide the quadrotor towards the moving mobile platform and to land safely. To achieve the desired results with good accuracy, a Kalman filter is used to track the mobile platform. Moreover, another Kalman filter is employed to predict the future location of the mobile platform.

A model predictive controller (MPC) is utilized to guide the quadrotor toward the predicted location. The model predictive control helps the quadrotor to follow its desired path. This study proposed a navigation system that utilizes the motion capture system’s information and Kalman filter to predict the future location of a moving mobile platform and navigate the quadrotor to the mobile platform. The navigation system controls the quadrotor’s take off, cruising trajectory, and landing on the mobile platform, autonomously. Several experiments are conducted to verify the proposed navigation system's performance and reliability. The results of the

experiments demonstrate that the proposed navigation system has proved itself effective in achieving the desired results.

Keywords: Motion capture system, Opti-Track cameras, Kalman filter, Model predictive control, autonomous landing.

ÖZ

OPTI-TRACK KAMERALARI KULLANARAK BİRDEN FAZLA TEMSİLCİ İÇİN YERELLEŞTİRME VE YOL PLANLAMASI

AL-QASSAB, Ayman

Department of Mechatronics Engineering

Danışman: Dr. Ogr. Uy M. Umer Khan

Danışman: Dr. Ogr. Uy M. Hassan GOL MOHAMMADZADEH

Eylül 2021, 57 Sayfa

Fiziksel bir mekandaki nesnelerin veya canlıların hareketlerinin dijital olarak algılanması ve kaydedilmesi işlemi, Opti-Track sistemi gibi hareket yakalama (MoCap) sistemi kullanılarak gerçekleştirilir. Bu çalışmada hareket yakalama sisteminin amacı, hem quadrotor hem de mobil platformun pozlarını sürekli olarak belirlemektir. Konum bilgisi navigasyon sistemi tarafından quadrotor'u hareketli mobil platforma yönlendirmek ve güvenli bir şekilde inmek için kullanılır. İstenen sonuçları iyi bir doğrulukla elde etmek için mobil platformu izlemek için bir Kalman filtresi kullanılır. Ayrıca, mobil platformun gelecekteki konumunu tahmin etmek için başka bir Kalman filtresi kullanılmıştır.

Quadrotoru tahmin edilen konuma yönlendirmek için bir model öngörücü kontrolör (MPC) kullanılır. Model öngörücü kontrolü, quadrotor'un istenen yolu izlemesine yardımcı olur. Bu çalışma, hareketli bir mobil platformun gelecekteki konumunu tahmin etmek ve quadrotor'u mobil platforma yönlendirmek için hareket yakalama sisteminin bilgisini ve Kalman filtresini kullanan bir navigasyon sistemi önerdi. Navigasyon sistemi, quadrotor'un kalkışını, seyir yörüngesini ve mobil platforma

inişini otonom olarak kontrol eder. Önerilen navigasyon sisteminin performansını ve güvenilirliğini doğrulamak için çeşitli deneyler yapılmıştır. Deneylerin sonuçları, önerilen navigasyon sisteminin istenen sonuçlara ulaşmada etkili olduğunu kanıtladığını göstermektedir.

Anahtar Kelimeler: Hareket yakalama sistemi, Opti-Track kameralar, Kalman filtresi, Model öngörücü kontrol, otonom iniş.

To my family, and friends

ACKNOWLEDGMENTS

I would like to express deepest gratitude to my mentor and supervisor, Asst. Prof. Dr. Muhammad Umer Khan, for his advice, constant support, endless patience and encouragement. And also my co supervisor Asst. Prof. Dr. Mohammad Hassan GOL MOHAMMADZADEH for his advice, support, and patience.

The financial support of The Scientific and Technical Research Council of Turkey (TUBITAK) is also acknowledged.

Finally, I would like to thank my parents support in the last three years.

TABLE OF CONTENTS

LIST OF TABLES

TABLE OF FIGURES

LIST OF ABBREVIATIONS

MoCap	Motion capture
MPC	Model Predictive Control
UAV	Unmanned Aerial Vehicle
VTOL	Vertical Take-Off and Landing
GPS	Global Positioning System
IMU	Inertial Measurement Unit
RTS	Radio Technology Somfy
PID	Proportional, Integral, Derivative
KF	Kalman Filter
QP	Quadratic Problem
PVA	Position, Velocity, Acceleration
TCP	Transmission Control Protocol
API	Application Programming Interface
DOF	Degrees Of Freedom
FPS	Frames Per Second
SDK	Software Development Kit
RSSI	Received Signal Strength Indicator

CHAPTER 1

INTRODUCTION

Quadrotors are a type of Unmanned Aerial Vehicle (UAV) that have gained popularity among the researchers and hobbyists due to the unique and significant abilities of vertical takeoff and landing (VTOL), side slip, back-ward flight, and hovering. Thanks to their maneuverability and relatively simple structure, these vehicles have become one of the most popular UAVs. In addition to maneuverability, being capable of VTOL and stationary hovering, their considerable payload capacity and reasonable flight times make them ideal flight platforms for many indoor and outdoor applications including, but not limited to: aerial surveillance, agricultural irrigation, pipeline inspection, object transporting, and military applications. Many of these applications require tracking of or landing on an object moving on the ground.

Autonomous landing is an essential stage in a fully autonomous system for quadrotor, especially landing on a moving mobile platform. Its importance could be found in rapid deployments and recovery of the fleet, extended operational range, and a mobile recharging station. Over the years, many researchers have been exploring the problem of autonomous landing of UAVs on mobile platform. In the present study, a complete navigation and control system is powered by the Opti-Track cameras, filters and analyzers, and a dedicated control system for the quadrotor (Tello EDU).

The starting point of the research is to utilize the Opti-Track cameras to locate the drone and the mobile platform in a 3D space, and in a live stream mode. To investigate this matter, captured location data were analyzed and processed through a Kalman filter to predict the future location of the mobile platform. This process helps the drone to approach the predicted mobile platform's position to land on it. A Kalman filter is a set of mathematical equations that provides efficient computational data observed over time by minimizing the effect of noise and other inaccuracies.

This filter is very powerful in several aspects as it supports estimations of past, present, and future states even when the precise nature of the modeled system is unknown. It has been widely applied in the fields of orbit calculation, target tracking, and navigation. Common examples are the calculations of spacecraft orbit, tracking of maneuvering target, and positioning using Global Positioning System (GPS).

1.1 Motivation

The motivation behind this study is to test the capabilities of the motion capture systems to achieve an autonomous landing of the quadrotor on a mobile platform. The quadrotor has many applications in the real world, such as transporting packages, surveying, fast deployment and recovery. Autonomous landing is considered one of the most challenging tasks of any fully autonomous quadrotor. The autonomous landing on a moving platform is preferable over the manual landing as it ensures more precision with minimum consumption of time and energy.

1.2 Problem statement

The autonomous control is highly desirable for the landing operation as to reduce the overall time, land on a difficult terrain and on a moving platform. The success of this operation depends upon the accurate position and orientation information of the quadrotor and the moving vehicle. Therefore, the need for tracking the moving platform in the dynamic environment and predicting its future position and velocity become a necessity in the autonomous landing for the quadrotor.

1.3 Contributions

The focus of this study is to design a complete quadrotor navigation system for autonomous estimation, tracking, and landing on a moving mobile platform. This is achieved by utilizing the motion capture system (Opti-Track). The contributions of this work are stated as follows:

1. Develop a linear Kalman filter model for a mobile platform to track both its state and position, also predict the future position;
2. Design the Model Predictive Control to navigate the quadrotor to the desired target;

3. Build a navigation system to coordinate data and commands between the hardware components.

1.4 Organization of the Thesis

The present thesis consists of seven chapters: Introduction, Literature review, Concepts, Implementatıon, Experiments and Results, Discussion and Conclusion.

Chapter 1 (Introduction): information about the quadrotor, motion capture system, Kalman filter, and the problem statement is defined.

Chapter 2 (Literature Review): a general introduction to quadrotor, Kalman filter, model predictive controller (MPC), Opti-Track system, and their related works are discussed.

Chapter 3 (Mathematical Preliminaries): mathematical concept for the Kalman filter, model predictive controller, and the coordinate transformation between both the motion capture system and the quadrotor is discussed.

Chapter 4 (Methodology): information about the system software, model for Kalman filter, model for the model predictive controller (MPC), and comparison between MPC model and PID controller is presented.

Chapter 5 (Hardware Setup): information about the quadrotor Tello EDU, the Opti-Track system, the mobile platform is presented along with the communication links and data transition for all of the system components.

Chapter 6 (Experiments and Results): the laboratory experiments, the setup of each test, and the results are explained in detail.

Chapter 7 (Discussion and Conclusion): contributions and the findings of the obtained results along with the concluding statement and future recommendations are presented.

CHAPTER 2

LITERATURE REVIEW

2.1 Quadrotor Autonomous Landing

The quadrotor is a distinctive type of Unmanned Aerial Vehicle (UAV) which possesses the ability to perform vertical take-off and landing (VTOL). Because of its unique dynamic nature, the quadrotor possesses flexibility and manoeuvrability advantage. These characteristics gave it prominence over other contemporaries; therefore, it is extensively utilized in universities for research problems, such as flight control, guidance, and autonomous landing. The extraordinary capabilities and astounding features of the quadrotor made it desirable for outdoor applications as well e.g., environment monitoring, aerial photography, power line inspection tour, aero delivery as well as searching and rescue, and other operations.

In the past decade, many researchers focused on the development of autonomous control of the quadrotor as it introduces the ability to accurately and efficiently perform tasks that would be of high risk for a human pilot to perform. Among the tasks associated with the autonomous control of the quadrotor, take-off and landing are considered the most critical as the quadrotor has to time its descent to make a proper land. The difficulty of achieving the autonomous landing is dependent upon true localization of the quadrotor and the landing platform, and state estimation of the quadrotor and the landing platform.

Localization of the landing platform, as well as the quadrotor, is done by using one or more of these methods, such as global position system (GPS), inertial measurement unit (IMU), and computer vision. In [1], the researchers combine the use of GPS and on-board camera to move the quadrotor near the moving vehicle so the on-board camera can detect it and start the tracking and landing process. To test the propose system, the researchers conducted different test scenarios. The test scenarios setup include static or slowly moving pattern and within laboratory conditions, without the presence of wind, with stable light conditions and simulated

GPS location. The results show the quadrotor was able to successfully land on the moving vehicle. However, the simulated GPS assume to be working ideal conditions without 1 cm accuracy. Researchers in [2] used a custom landing pad with different patterns to be detected by the on-board camera from different distances to perform the landing, the results show the vision based distance estimation algorithm is capable to detect the landing platform from different heights and distances and estimate the distance to the landing platform. An experiment is conducted in a lab environment with stationary landing platform. Results show the quadrotor successfully landed on the platform smoothly. Moreover, the researchers in [3] employed a combination of cameras and laser range finders to perform an autonomous flight and navigation of the quadrotor. A limited test scenario is conducted by the researcher. The experiment is conducted in controlled lab environment to improve laser finder range and accuracy. Also a especial landing pad materials are used to eliminate laser finder noise. The results shows the using laser range finder for quadrotor is considered practical only in limited scenarios.

Another method that has been developed in the past years to localize the quadrotor and landing platform is known as a motion capture system (MoCap). The advantage that can be found in using a motion capture system is its higher position accuracy and faster real-time data transmission.

Different technologies and techniques have been developed to capture motion resulting in different approaches, such as optical-passive, optical-active, and video. The most commonly used motion capture system is VICON [4] and Opti-Track [5]. In [6], the researchers utilized a motion capture system to locate the quadrotor and the mobile platform, as well as acquire their orientation. Motion captured data was by the researchers guidance system to guide the quadrotor to the moving mobile platform and land safely. Multiple test scenarios were conducted in lab environment without any disturbance. Results show that the quadrotor was able to land on the moving platform in reasonable time assuming slowly moving platform. The authors of [7] also used a motion capture system and proposed a control law using an open-source autopilot controller to achieve autonomous flight of an indoor quadrotor. The open source quadrotor software uses PID controller to navigate the quadrotor to the landing platform. A simple test scenario was conducted in lab environments to land

the quadrotor on a slowly moving landing platform. The results show the quadrotor successfully landing on the moving platform. The researchers used the data from the motion capture system (VICON) to design a framework using Simulink to monitor a moving ground vehicle and navigate the quadrotor [8]. Several test scenarios were conducted including square moving ground vehicle. The results show the proposed navigation framework was successful in guiding the quadrotor to follow the moving ground vehicle without performing landing procedures. The researchers in [9] suggested an event-based controller, the RTS controller parameters were established using a PID scheme, and an event-based corrector was also included. To locate the quadrotor, the motion capture (VICON) technology was employed. The experimental result shows the advantage of an event-driven scheme in increasing the real time control and high reduction of the computing. As well as, communication resources utilization provided a dynamic model with states and observation equations and then utilized a Kalman filter to track an object using an accurate estimate adjusted by the measurements.

Quadrotors come in different types and capabilities. The quadrotors that are used for autonomous landing quadrotor research are limited to a couple of types. Some researchers use custom-made quadrotors like in [2-7]. In such a case, the researcher selects quadrotor components and sensors to fit his/her research area and to have full control over the quadrotor. Another group of researchers use off-the-shelf expensive high-end quadrotors that offer full control of the quadrotor and advanced sensors, like high accuracy Global Navigation Satellite System (GNSS) chip and Inertial Measurement Unit (IMU). In this study, a different kind of quadrotor is used. To the best of our knowledge, this is the first study to explore using budget orient, off-the-shelf low-end, quadrotor for Autonomous landing. Unlike high-end quadrotors, Low-end quadrotors have limited capabilities and controllability. Also, most researchers use quadrotor built-in sensors with external advanced algorithms to achieve best results. For low-end quadrotors using built-in sensors is not possible. Thus, in this study build-in algorithms are used and complemented with the research algorithm to achieve best results.

2.2 Kalman Filter (KF)

Kalman filter, originally proposed by R. E. Kalman in 1960 [10], is a linear optimal status estimation method that is known as one of the most famous Bayesian filter. It is a set of mathematical equations to provide an efficient computation of data observed over time, which contains noise and other inaccuracies [11]. Kalman filter has been widely applied in the fields of target tracking and navigation.

In this study, the Kalman filter is utilized to improve the localization of the objects in the quadrotor navigation. The researchers in [12] provided a comprehensive framework is proposed to deal with quadrotor trajectory tracking and positioning in indoor situations. The system is built on two major structures: a Kalman Filter (KF) to track the vehicle's 3D position and a nonlinear controller to guide it through its flying missions. Several experiments are conducted to test the performance of the quadrotor to track moving objects in different moving path. The experimental results demonstrated the effectiveness of the proposed framework. In [13], the researchers used a Kalman filter integrated with an optical flow method to measure the velocity of the mobile platform using a downward-looking camera. The results show that the differential optical flow technique has the benefit of being highly accurate in measuring velocity. Its inability to deal with massive picture displacements, on the other hand, limits its use. Therefor kalman filter is used to improve the velocity measurements of optical flow. Other researchers used a combination of machine learning and a Kalman filter to estimate the instantaneous positions of a moving target. This approach may be used to acquire target accelerations as well as precise location estimates. For example in [14] use wireless sensor network and collect received signal strength indicators (RSSI) of the target. Then machine learning algorithms are used on the collected database to compute a model that estimates the position of the target using only RSSI information. In addition Kalman filter is used to provide better estimation. Different test scenarios are conducted and the results were comparable to other methods. One drawback is that the tests are conducted in wirelessly isolated area.

2.3 Model Predictive Control (MPC)

Model predictive control (MPC) refers to a class of computer control algorithms that utilize an explicit process model to predict the future response of a plant. At each

interval, MPC attempts to optimize the future plant behavior by computing a sequence of future manipulated variable adjustments [15].

To improve the quadrotor's ability to achieve autonomous landing, several types of the controller have been implemented, such as PID, MPC, and many others. The model predictive control shows more capability and flexibility in controlling the quadrotor than the PID controller, as demonstrated in [16]. The simulation results show that MPC can improve the performance of the quadrotor in landing operation. In [17], the researchers used a method of combining model predictive control with feedback linearization. It is impossible to precisely linearize the dynamics of the quadrotor since it is an under-actuated nonlinear system. As a result, researchers only linearized the translational motions and then employed a linear optimum control to regulate it followed by the MPC to cope with disturbances. The simulation results were used to show the effectiveness of the proposed method. The researchers in [18] designed a model predictive controller to fully control a quadrotor to track and land on a moving ground vehicle. The motion capture system (Opti-track) was used to localize the ground vehicle and the quadrotor and also provided the orientation. The experimental results showed the successful operations of autonomous control and autonomous landing of the quadrotor on a moving ground vehicle**.**

CHAPTER 3

MATHEMATICAL PRELIMINARIES

3.1 Kalman Filter (KF) mathematical concept

After discussing the importance of KF in the previous chapter, the mathematical model of KF is explained in detail. The state equation is a linear combination of process noise (w_k), inputs (u_{k-1}), and prior state (x_{k-1}). The observation equation is also a linear combination, but of states (x_k) and measurement noise (v_k).

The state equation is defined as

$$x_k = Ax_{k-1} + Bu_{k-1} + w_k \tag{3.1}$$

The observation equation to determine the output z_k is written as

$$z_k = Hx_k + v_k \tag{3.2}$$

The state and observation vectors are represented as $x_k$and z_k; the state transition matrix is represented by A; the observation matrix is represented by H; and the input matrix is represented using B. The process (w_k) and measurement v_k noise vectors in Eq. (3.3) and (3.4) are assumed to be uncorrelated and zero-mean Gaussian white noise vector:

$$E(w) = 0, cov(w) = E(ww^T) = Q \tag{3.3}$$

$$E(v) = 0, cov(v) = E(vv^T) = R, E(wv^T) = 0 \tag{3.4}$$

The states obtained during estimation at instances $k-1$ and k are called the priori state estimation $\hat{x}_k^-$ and posterior state estimation $\hat{x}_k$, respectively. Based on these priori and posterior states, their estimation errors are calculated in Eq. (3.5) and (3.6) as:

$$e_k^- = x_k - \hat{x}_k^- \tag{3.5}$$

$$e_k = x_k - \hat{x}_k \tag{3.6}$$

The covariance of the priori and posterior states is calculated as:

$$P_k^- = E[e_k^- e_k^{-T}] \tag{3.7}$$

$$P_k = E[e_k e_k^T] \tag{3.8}$$

To derive the KF mathematical model, state estimation $\hat{x}_k$ is to be determined first. The state estimation introduces the linear relation between the priori estimation and the weighted difference of true measurements and measured forecast values. The prediction mathematical model in KF is defined as:

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1} \tag{3.9}$$

$$P_t^- = AP_{k-1}A^T + Q \tag{3.10}$$

The update equation model used in KF is obtained as:

$$K_k = \frac{P_k^- H^T}{(HP_k^- H^T + R)} \tag{3.11}$$

$$\hat{x}_{k-1} = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-) \tag{3.12}$$

$$P_k = (I - K_k H)P_k^- \tag{3.13}$$

The Kalman gain matrix, optimum state filter vector, process covariance matrix, and identity matrix are represented by K_k, x_k, P_k and I, respectively [11].

3.1.1 Tracking Process

After defining the predict and correct models for the system states in the previous section, the initial conditions and states are defined. This information helps the KF to initiate the prediction of the actual state in the first stage, followed by the estimation correction in the second stage. In this study, KF is employed to track the movements of the mobile platform during the process. The position and other necessary information are directly obtained through the Opti track camera. The overall process of KF is depicted in Figure 3.1.

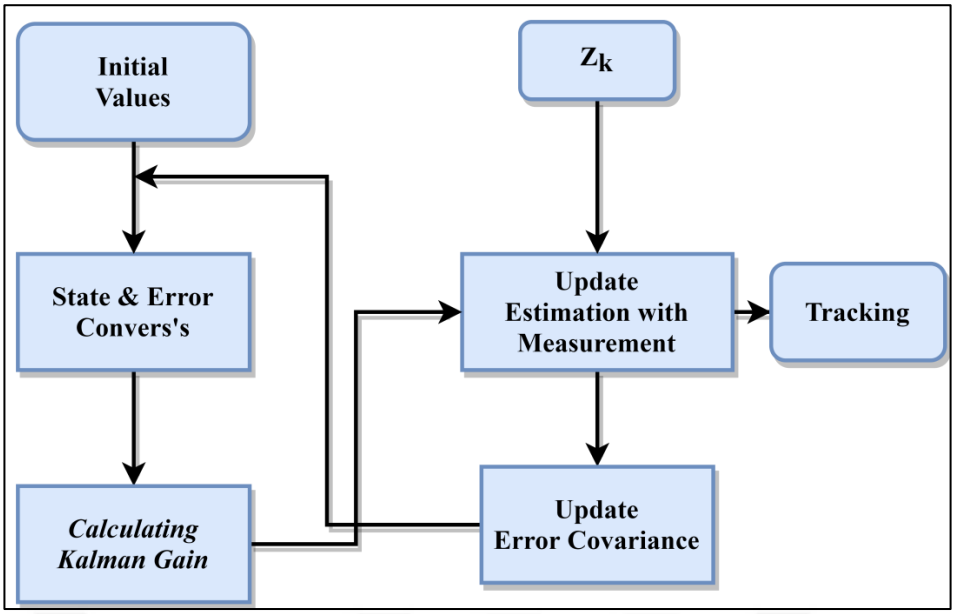


Figure 3.1 Overview of Kalman filter working, z_k represents the updated information

3.1.2 Prediction

To use the Kalman filter for prediction, two important assumptions are made. First, the tracked system will not experience any disturbance for the estimated time. Secondly, the tracked system will have constant velocity and rotation rate for the predicted period. To initialize the predication Kalman filter, Tracking Kalman filter, explained in previous section, current states (k), x_k, A, B, Q, P_k and K_K are used. Since there are no measurements to correct the estimated states; therefore, based upon the assumption that z_k is similar to the last estimated state $\hat{x}_{k-1}$ results in $\hat{x}_{k-1} = \hat{x}_k^-$. The $\hat{x}_{k-1}$ represent the future predicted state of the system.

3.2 Model Predictive Control

The Model Predictive Control (MPC) is unique as it uses a future prediction to calculate the input. To reduce the error between the output of the plant and the target reference output, the MPC controller uses an optimizer to optimize the output. The prediction strategy adapted here suggests the usage of the plant model by the MPC controller to simulate the quadrotor path in the next P time steps, where P the prediction horizon is. The prediction horizon represents the time, and the MPC controller looks ahead of time to make the prediction. The MPC is tested with different future scenarios in a systematic way. The optimizer is responsible to determine the best possible scenario using which minimum error between the reference and the predicted trajectory is achieved. The optimizer tries to minimize

the cost function which is defined as the error. Therefore, the predicted trajectory with the minimum cost function is considered as the optimal solution. Figure 3.2 shows the block diagram of the complete system with the MPC.

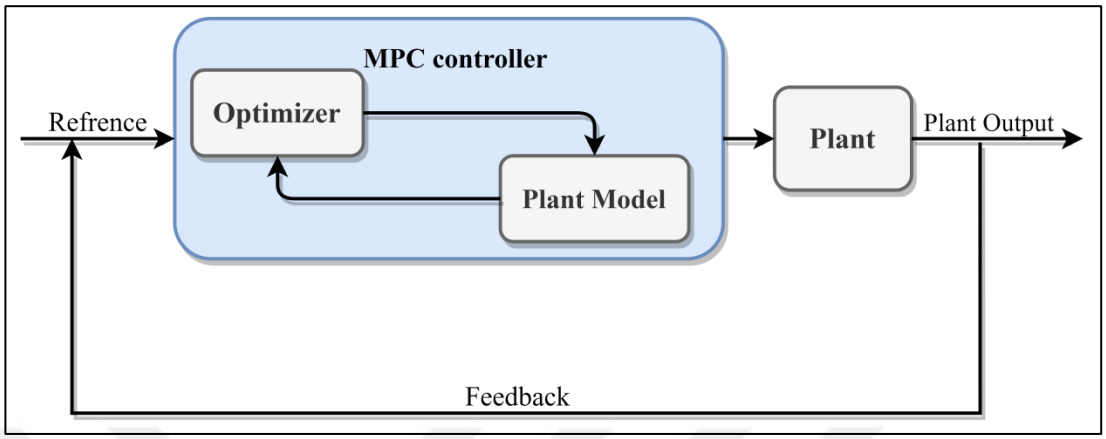


Figure 3.2 Model predictive control block diagram [19]

The control optimization problem is a Quadratic Problem (QP) which MPC tries to solve at each control interval. The optimization parameters are summarized as follows:

Cost Function is the objective function of the optimization problem; the minimization of the cost function is also used to evaluate the controller performance.

Constraints are either defined as soft or hard. These constraints must satisfy the system conditions, such as the physical bound.

The MPC controller is responsible for determining the desired control inputs to reduce the error between the output of the plant and the desired reference. This process is performed by applying different scenarios and minimizing the cost function of the optimization problem.

Decision is the process where the optimization algorithm modifies the manipulated variables to minimize the cost function and to satisfy the constraints.

The manipulated variable is computed by solving the quadratic problem using a custom QP solver. Figure 3.3 shows the control algorithm of the Model Predictive Controller.

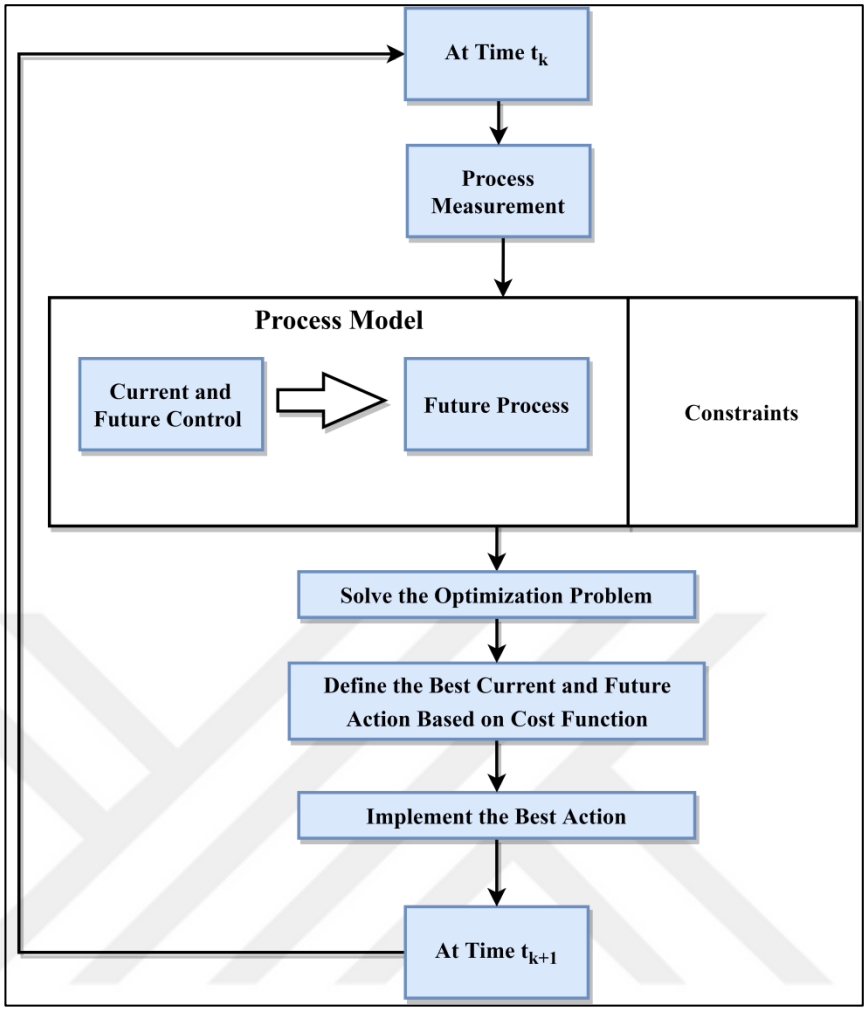


Figure 3.3 Model predictive control algorithm [20]

3.2.1 Design Parameters

Designing the MPC should take into consideration all the required constraints e.g., steering angle limits. Figure 3.4 depicts the functioning of MPC. Following terminologies are used in its implementation: k is the current sampling step, Ts is the control time step, P is the prediction horizon, and M is the control horizon. The prediction horizon represents the capability of MPC to look ahead of time to make the prediction. The control horizon is used to define the number of the possible control actions to time step $k + P$. The design parameters of the MPC directly affect the performance and the computational complexity of the optimization problem. The design parameters must be chosen carefully to attain the balance between the computational load and the performance.

Sample time (T_S) determines the execution rate of the control algorithm. If T_S is very big, the controller might not be able to respond in time to the disturbance, thus it deteriorates the performance. However, if T_s is very small, the controller's response will be faster, but it may cause a significant increase in the computational load.

Prediction horizon (P) parameters choice depends on the system dynamic and should be chosen to covers are the system motions.

Control Horizon (M) choice of this parameter depends on the control horizon. It should be much smaller than the control horizon value. . A small value of M provides stability while in contrast, large values reduce the robustness [19].

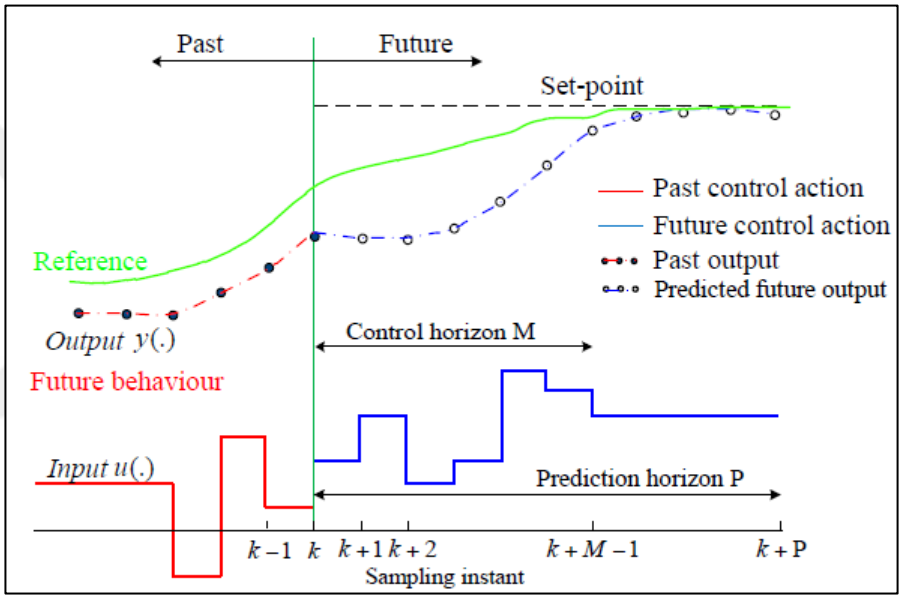


Figure 3.4 Model predictive control schema for the main concepts [19]

3.3 Coordinate transformation

In this study, two coordinate systems are defined; one for the quadrotor (body coordinate) and the other one for the motion capture system (global coordinate). The position, orientation, and all other calculations are done regarding the global coordinate system. A transformation matrix is required to represent the information from one reference system to another. The first step is to define the coordinates for both the quadrotor and the motion capture system. Quadrotor coordinate frame (denoted as B in Figure 3.5), is positioned at the quadrotor center of mass. The X_B points in the direction along with the nose of the quadrotor; Y_B points to the right of the quadrotor; and Z_B is pointing downward:

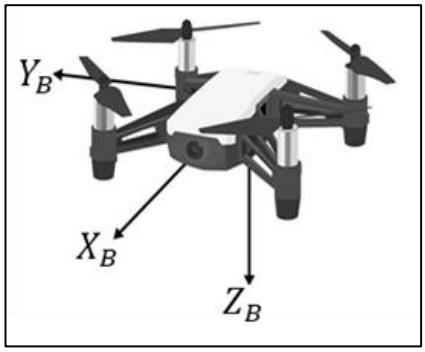


Figure 3.5 Quadrotor (Tello EDU) coordinate frame

Motion capture coordinate system (denoted as *W*) is defined as a fixed right-hand referential with X_W pointing towards north, Z_W pointing towards East, and Y_W pointing straight upwards, as shown in Figure 3.6. The pitch, yaw, and roll axis are defined about the X_W, Y_W, and Z_W, respectively.

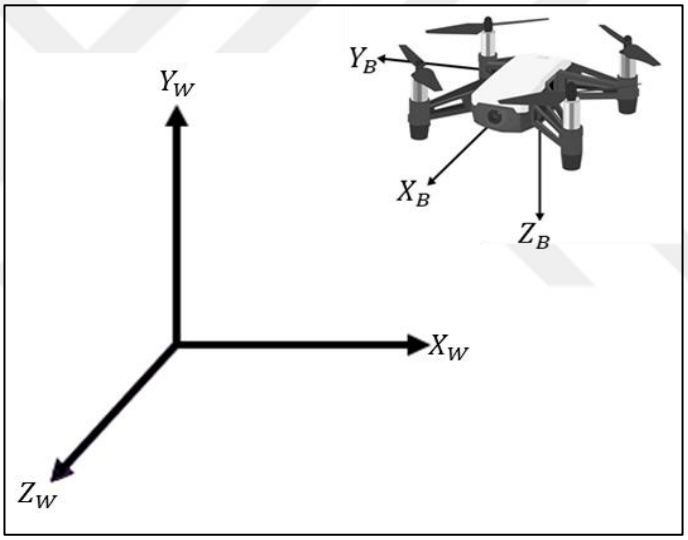


Figure 3.6 Motion capture system and quadrotor (Tello EDU) coordinate frame

A transformation matrix is employed to convert the motion capture coordinates system to the quadrotor coordinate system. This process can be split into two steps; the first is by applying the transformation matrix to change the quadrotor coordinates to the motion capture coordinates. The second is by performing the inversion operation on the results obtained in first step. The importance of the second step is to revert the operation performed in the first step. The rotation matrices needed to perform the transformation is defined as follows:

$$R_X = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\Phi & -sin\Phi & 0 \\ 0 & sin\Phi & cos\Phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.14}$$

$$R_y = \begin{bmatrix} cos\theta & 0 & sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -sin\theta & 0 & cos\Phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.15}$$

$$R_X = \begin{bmatrix} cos\varphi & -sin\varphi & 0 & 0 \\ sin\varphi & cos\varphi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.16}$$

The resulting rotation matrix is obtained as:

$$R = [R_X] \times [R_Y] \times [R_Z] \tag{3.17}$$

The overall transformation matrix is represented as:

$$M = \begin{bmatrix} & & & T_x \\ & R & & T_y \\ & & & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.18}$$

$T_x, T_y,$ and T_z represent the position of the quadrotor in the motion capture system. Once the world matrix is converted, multiplying it by the coordinates of a world-space point will give a point in the local space of the quadrotor [20].

CHAPTER 4

METHODOLOGY

4.1 System Design

The system involved in this study consists of several parts: a quadrotor, a mobile platform, a motion capture system (Opti-Track, NAT-NET SDK), a Kalman filter, a model predictive controller, and a System Navigation and Control system that runs on a laptop.

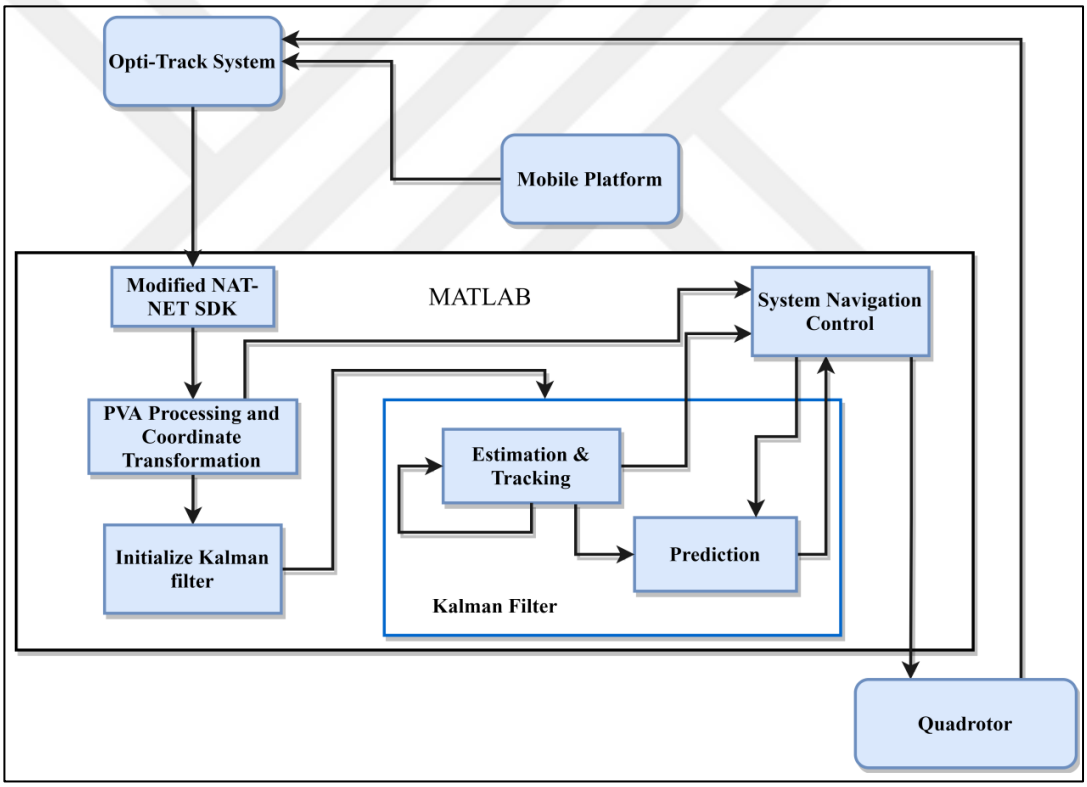


Figure 4.1 System design diagram

The prime task in the process is to setup the motion capture system and define the motion capture test area. This setup is done in the light of the recommendations provided by the manufacturers of Opti-Track cameras and the available space in the laboratory. Once the test area is defined, a software named as Motive initiates streaming data to the MATLAB, as shown in Figure 4.1. The data received by

MATLAB contains only the tracked objects' positions. Since Opti-Track only provides position information, a MATLAB function is written to calculate tracked objects velocities and accelerations. Another function is written for body-to-world coordinate transformation. Both functions are represented by PVA (Position, Velocity, Acceleration) Processing and coordination transformation block in figure (4.1). The data received from PVA is used to initiate the Kalman filter. One such filter is responsible for continuously tracking the mobile platform; whereas another KF is initiated after a defined time to predict the position of the mobile platform ahead of time. The predicted position is sent to the system navigation and control.

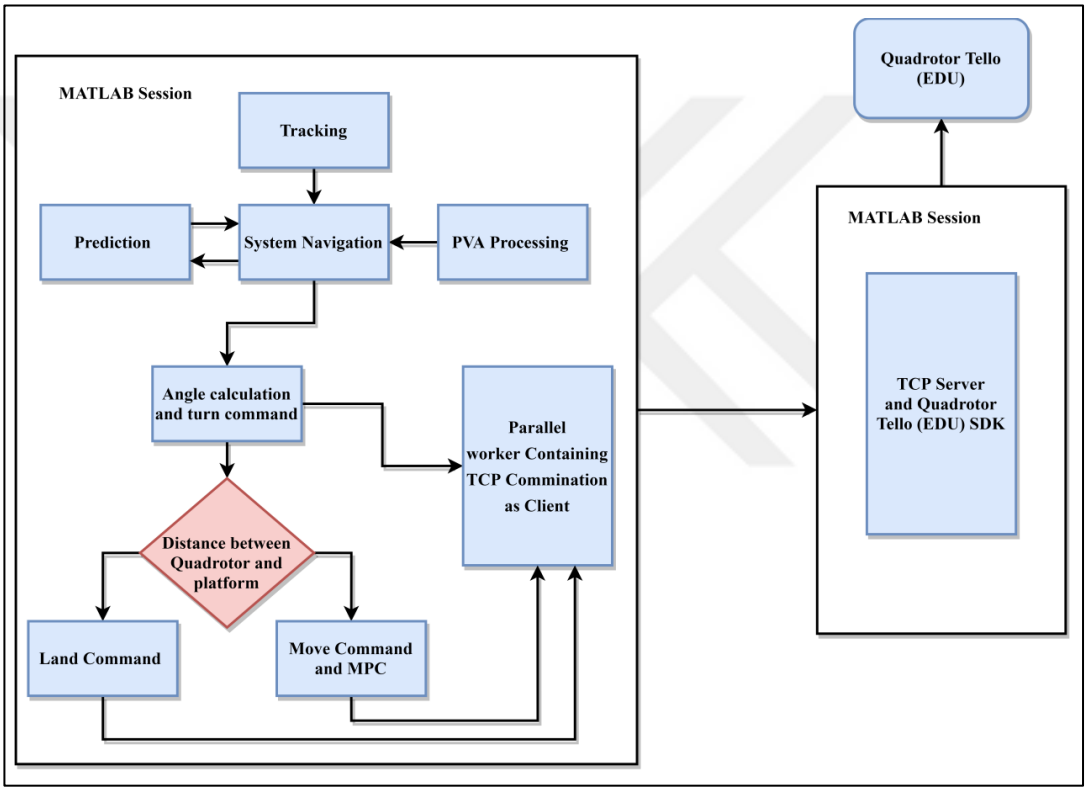


Figure 4.2 System design and navigation control

The predicted position is used to adjust the heading angle of the drone according to the mobile platform. The 'turn' command is sent to the sets of parallel workers running in another MATLAB session and wait until it receives 'finish' command from the quadrotor. Due to the inherent limitations of the Tello EDU quadrotor, the navigation commands are split into two iterations, move and wait. The communication between two MATLAB sessions is done using TCP protocol. The TCP-server will monitor and send commands to the quadrotor. The TCP-client will

receive the command from system navigation control. The TCP server/client will work as independent threads and their purpose is to ensure that the quadrotor will execute all the commands continuously. Two sets of errors in the navigation control are calculated continuously; the first set will monitor the error between the predicted position and the mobile platform's position and if the error exceeds the predetermined limits, a new prediction will start. The second set will calculate the distance between the drone and the mobile platform, and this information is used to pick the next possible command for the drone as move, turn, or land. In the move command, model predictive control will generate an angle which is then converted into x in y, as expected by the quadrotor's Application Programming Interface (API). If the drone drifts sideways, MPC will start to take measures to correct the drone's position.

4.2 Kalman Filter (KF) Model

The linear Kalman filter serves two purposes, tracking of the mobile platform and predicting the future location of the platform. The KF mathematical model for the mobile platform' tracking is defined as follows:

Assuming that the mobile platform can only move in a 2d plane, its motion can be completely defined using three variables: translation on the x-axis and y-axis, and rotation by an angle θ around the z-axis. To track the mobile platform, the pose information (x, y, θ) must be known at every instant of the specified time interval T. The movements of the mobile platform are defined as follows:

$$x_{k+1} = x_k + T_s v_{x(k)} + \frac{T_s^2}{2} a_{x(k)} \tag{19}$$

$$y_{k+1} = y_k + T_s v_{y(k)} + \frac{T_s^2}{2} a_{y(k)} \tag{4.2}$$

$$\theta_{k+1} = \theta_k + T_s \omega_{(k)} + \frac{T_s^2}{2} \alpha_{(k)} \tag{4.3}$$

$$v_{x(k+1)} = v_{x(k)} + T_s a_{x(k)} \tag{4.4}$$

$$v_{y(k+1)} = v_{y(k)} + T_s a_{y(k)} \tag{4.5}$$

$$\omega_{(k+1)} = \omega_{(k)} + T_s \alpha_{(k)} \tag{4.6}$$

The state-space model of the previous equations is obtained as follows:

$$X_{(k+1)} = \begin{bmatrix} x_{(k+1)} \\ y_{(k+1)} \\ \Theta_{(k+1)} \\ v_{x(k+1)} \\ v_{y(k+1)} \\ \omega_{(k+1)} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & T_s & 0 & 0 \\ 0 & 1 & 0 & 0 & T_s & 0 \\ 0 & 0 & 1 & 0 & 0 & T_s \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_k \\ y_k \\ \Theta_k \\ v_{x(k)} \\ v_{y(k)} \\ \omega_{(k)} \end{bmatrix} + \begin{bmatrix} \frac{T_s^2}{2} & 0 & 0 \\ 0 & \frac{T_s^2}{2} & 0 \\ 0 & 0 & \frac{T_s^2}{2} \\ T_s & 0 & 0 \\ 0 & T_s & 0 \\ 0 & 0 & T_s \end{bmatrix} \begin{bmatrix} a_{x(k)} \\ a_{y(k)} \\ \alpha_{(k)} \end{bmatrix} \quad (4.7)$$

where, T_s represent the time interval and k represents the time.

In compact form, Eq. (4.7) can be written as:

$$X_{k+1} = AX_k + Bu_k \quad (4.8)$$

A is defined as the state-transition matrix, B is the input matrix, and u_t represents the input (acceleration). X_{k+1} represents the state matrix terms, such as position, velocity, angle, and acceleration. To account for the uncertainty due to the inaccuracy of the model and/or the inputs (acceleration), a white-noise term is introduced to the model:

$$x_k = Ax_{k-1} + Bu_{k-1} + w_k \quad (4.9)$$

It is assumed that w is a Gaussian distribution noise with a mean 0 and a variance Q.

$$Q = BQ_bB \quad (4.10)$$

$$Q_b = \begin{bmatrix} \sigma_{a_x}^2 & 0 & 0 \\ 0 & \sigma_{a_y}^2 & 0 \\ 0 & 0 & \sigma_\alpha{}^2 \end{bmatrix} \quad (4.11)$$

The diagonal values of Q_b are chosen based upon the laboratory tests as 2. The off-diagonal elements of Q_b are zeros; this is due to the assumption that the motions on the x-axis and the y-axis, and the rotation around the z-axis are uncorrelated.

The measurement vector z_k is defined as:

$$z_k = Hx_k + v_k \quad (4.12)$$

Where, H is the output matrix that maps the state vector into the measurement domain and v_k is the measurement noise. The output matrix H is chosen as:

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{4.13}$$

The measurement noise v_k is introduced by means of measurements. The measurement noise is assumed to be Gaussian distributed with a mean of 0 and a variance R , where R also defines the covariance of the observation noise:

$$R = HH^T v_k \tag{4.14}$$

$$v_k = \begin{bmatrix} \sigma_{z_x}^2 & 0 & 0 \\ 0 & \sigma_{z_y}^2 & 0 \\ 0 & 0 & \sigma_{z_\theta}^2 \end{bmatrix} \tag{4.15}$$

Through trial and error, the diagonal values of v_k are chosen as 2.1. The Kalman filter equations for predicting and tracking the states are:

Predicting the state

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1} \tag{4.16}$$

Predicting the covariance matrix

$$P_k^- = AP_{k-1}A^T + Q \tag{4.17}$$

The Kalman gain is determined as:

$$K_k = \frac{P_k^- H^T}{(HP_k^- H^T + R)} \tag{4.18}$$

Correcting the state

$$\hat{x}_{k-1} = \hat{x}_k^- + K_t(z_k - H_k\hat{x}_k^-) \tag{4.19}$$

Estimating the covariance matrix

$$P_k = (1 - K_k H)P_k^- \tag{4.20}$$

At the beginning of the process, the Kalman filter must be provided a correct initial state and an initial covariance matrix.

For the prediction branch in the proposed navigation system, a Kalman filter is employed. The concept of using the Kalman filter for prediction is explained in the Section 3.1.2. Using the same sampling period T_S, the predicted state $\hat{x}^-(k-1)$ is updated after every T_S. The number of updates depends on the desired prediction time.

4.3 Model Predictive Control (MPC) Model

The basic kinematical model of a moving object in a two demotion plane at a content speed as shown below:

$$\dot{x} = V \cos a \tag{4.21}$$

$$\dot{y} = V \sin a \tag{4.22}$$

where, V represent the velocity, $\dot{x}$ is the longitude displacement in the x-axis, $\dot{y}$ is the lateral displacement in the y-axis, and a is the heading angle. Assume tracking a straight line. The linear approximation is made by considering the very small heading, $a \ll 1$. Based upon this assumption, the above model is linearized as:

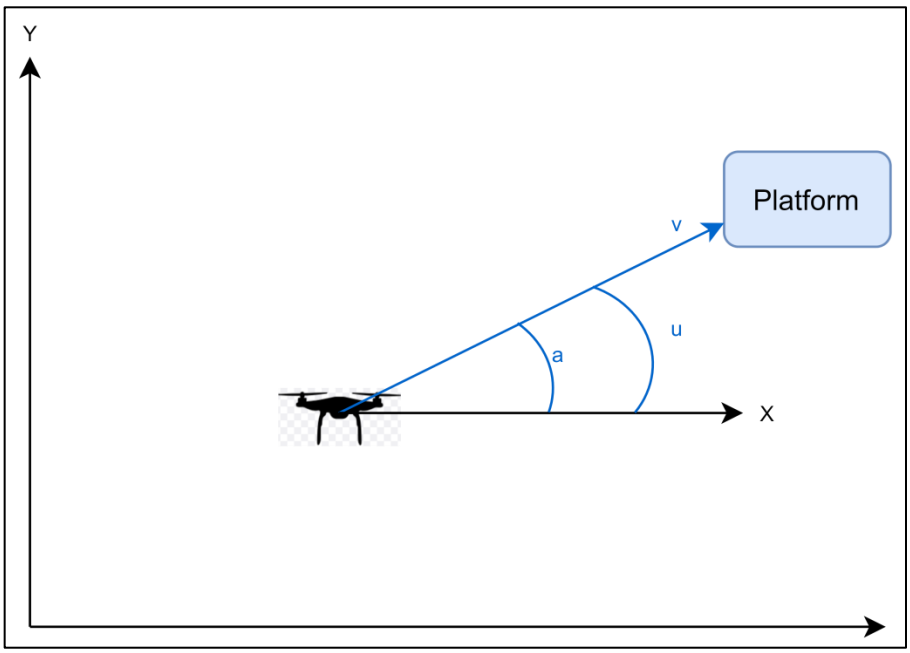


Figure 4.3 Steering diagram

$$\sin a \approx a \tag{4.23}$$

$$\cos a \approx 1 \tag{4.24}$$

$$\dot{a} = u \tag{4.25}$$

$$\dot{y} = Va \tag{4.26}$$

The sample time equation is written as:

$$a(t+1) = a(t) + u(t)T_s \tag{4.27}$$

$$y(t+1) = y(t) + a(t)VT_s + u(t) \times 0.5VT_S^2 \tag{4.28}$$

T_s represents the sampling time. Based upon the Eqs. (4.27) and (4.28), the state-space representation is written as:

$$X_{t+1} = Ax_t + Bu_t \tag{4.29}$$

where:

$$X(t) = \begin{bmatrix} a(t) \\ y(t) \end{bmatrix} \tag{4.30}$$

$$A = \begin{bmatrix} 1 & 0 \\ VT_s & 1 \end{bmatrix} \tag{4.31}$$

$$B = \begin{bmatrix} T_s \\ 0.5VT_s^2 \end{bmatrix} \tag{4.32}$$

A is the state matrix, B is the input matrix, u_t is the input (steering), and X_{t+1} is the state matrix.

The observation matrix is defined as:

$$y(t) = Cx(t) \tag{4.33}$$

$$C = [0 \quad 1] \tag{4.34}$$

Formation predictive model

$$Y = Gx + HU + Fu \tag{4.35}$$

MPC optimization problem

$$J = (Gx + HU + Fu)^T(Gx + HU + Fu) + rU^T D^T DU \rightarrow min \tag{4.36}$$

subject to: $|U| \leq U_0$

The solution for this optimization problem can be chosen as:

$$x(t) \rightarrow [MPC\ QP] \rightarrow U \rightarrow u(t+1) = U(1) \tag{4.37}$$

The predictive model, and the MPC optimization problem are solved using MPC designer toolbox provided by MATLAB. The state-space information is provided to the designer and is later used to tune the controller and set the soft constraints, hard constraints, prediction horizon, and control horizon. The reference input for the MPC is the error between the quadrotor's position and the mobile platform's position. The simulation results are obtained by assuming the quadrotor's position as 5 m on the x-axis and 5.5 m on y-axis, whereas the mobile platform's position is set at 20 m on the x-axis and 5 m in the y-axis. The MPC parameters for the simulation are configured as 2 prediction horizon sample points, 1 control horizon sample point, hard constraints are from 90 to -90 degrees, and the soft constraints are from 60 to -60 degrees.

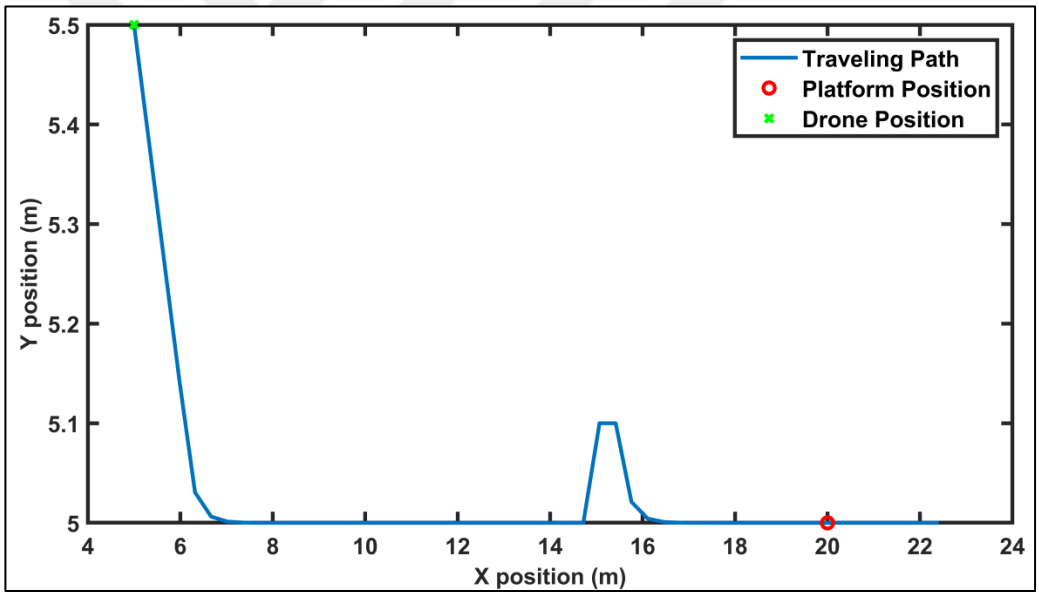


Figure 4.4 Traveling path of the quadrotor

From Figure 4.4, it is obvious that the MPC is providing the necessary angle for the quadrotor to approach the mobile platform with a 0.5 seconds sampling time. The quadrotor is drifted by 10 cm along the y-axis when it reaches 14.3 m in x-axis. The MPC response to correct the quadrotor position is lagged by 0.5 seconds due to the measurement time sampling period.

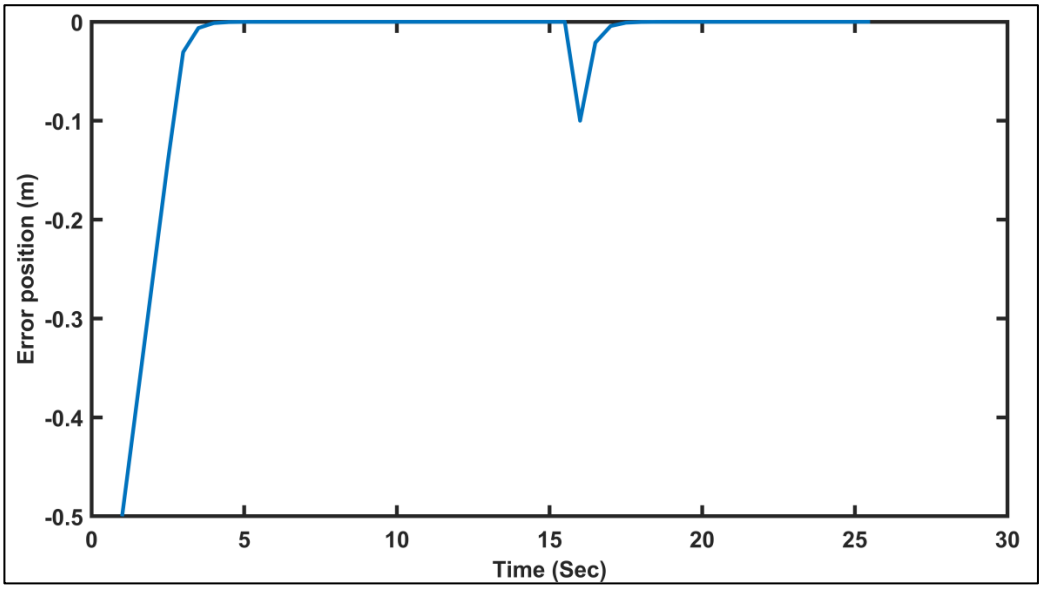


Figure 4.5 Error as a reference for the model predictive control

Figure 4.5 shows the position error between the quadrotor and the mobile platform. At the start of the simulation, the error is recorded as 50 cm. But, the controller exhibits its performance by reducing the error in a very short time. To further evaluate the performance, a disturbance of 10 cm is introduced at 15.5 sec.

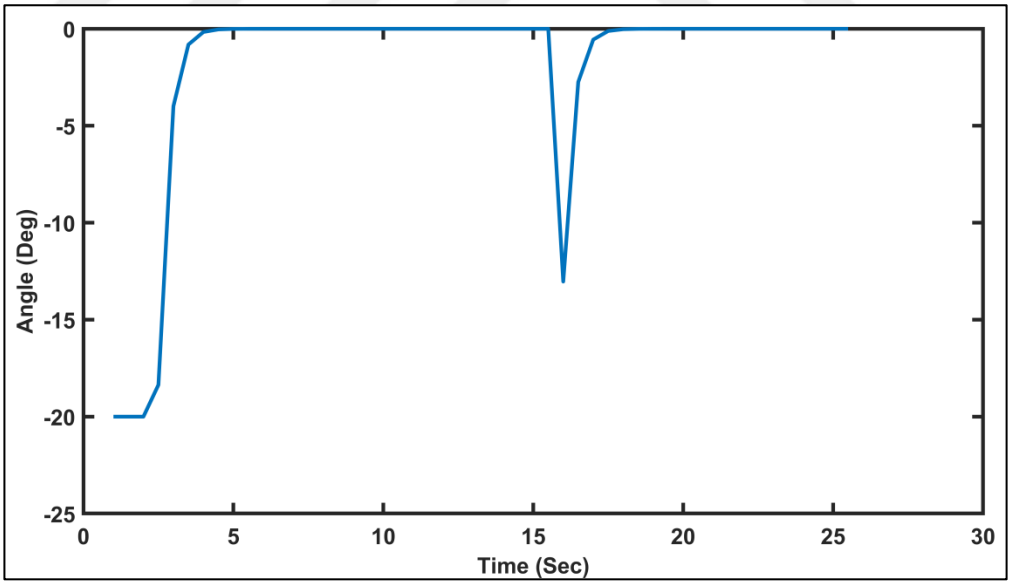


Figure 4.6 Angle value from the MPC

The figure 4.6 shows the generated angle from the MPC that is sent to the quadrotor.

4.4 Comparisons between a Model Predictive Control and PID linear Control.

A comprehensive comparison between PID an MPC for quadrotor control is discussed in details in [16]. To compare the performance of MPC with PID for quadrotor guidance, quadrotor control simulation is utilized. The MPC simulation parameters are kept the same as in Section 4.3. To test the controller's noise tolerance, a small white-noise is added to the measured reference error. The PID controller is designed by using the general equation as:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt} \tag{4.38}$$

K_p stands for the proportional gain, the integral gain is represented using K_i, and K_d stand for the derivative gain. The $e(\tau)$ and $u(t)$ are defined as the input error and system's response output, respectively [21]. The PID parameters are initially computed by using the MATLAB PID-tuner toolbox followed by the manual tuning to improve the results. The PID values are chosen as $K_p = 80.80$, $K_i = 0.3$, and $K_d = 0.484$.

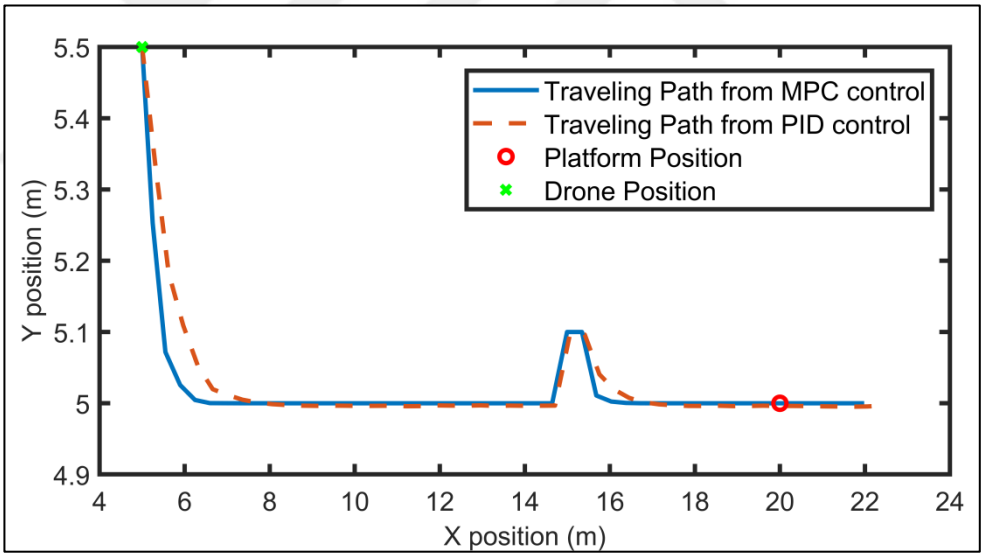


Figure 4.6 Traveling paths of quadrotor with MPC or PID controller

From figure 4.6, it can be observed that the traveling path of the quadrotor using MPC is more robust as compared to the PID-controlled quadrotor traveling path.

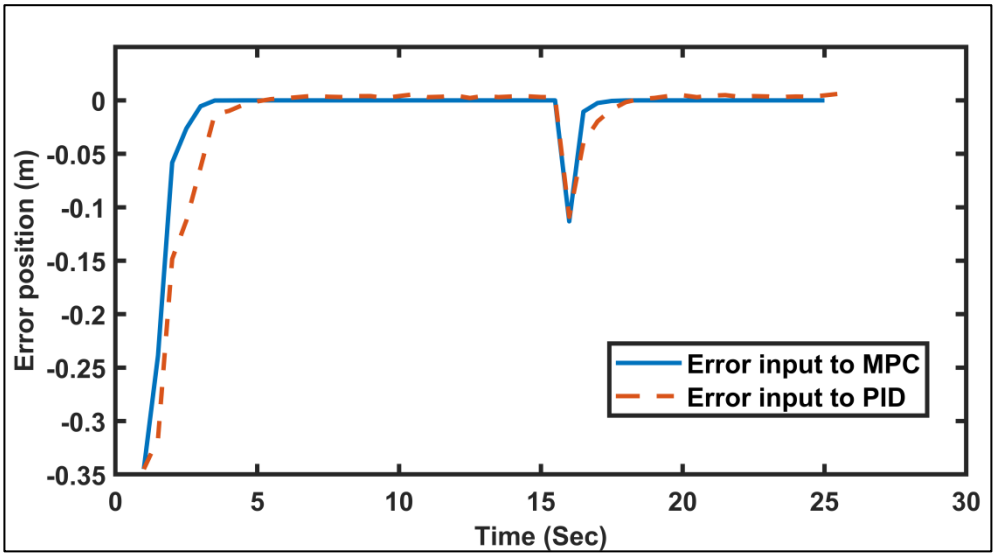


Figure 4.7 The input signal to the controller

Figure 4.7 shows the controllers' input signal defined as the error between the quadrotor's lateral position and the mobile platform's lateral position. The input error for MPC is much smoother than the PID controller.

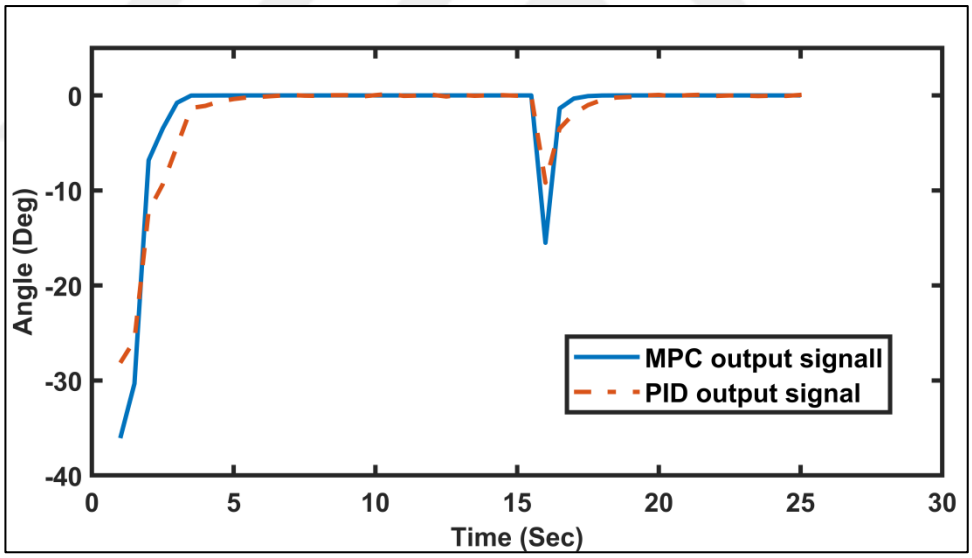


Figure 4.8 the generated angle value from MPC and PID controllers

Figure 4.8 shows the controller output as measured angle (degree). It is evident that the MPC can converge faster than the PID controller. In addition, the output angles obtained through MPC are smoother than the PID controller. In [16] further experiments were conducted to compare between PID and MPC. The experimental results show similar funding to this study. An important difference between both controllers is that the MPC has soft and hard constraints to better control the input

and output of the desired plant, and it uses optimization and Kalman filter to obtain the better performance.

CHAPTER 5

HARDWARE SETUP

5.1 Quadrotor

Tello EDU, a quadrotor chosen for the experimentation is, made by the Shenzhen Ryze Technology that incorporates the DJI flight control technology and Intel processor. It is a ready-to-fly quadrotor with advanced flight control, vision position system, and interacted API. It has its own SDK that works with third-party programs and consists of the basic commands, such as move, turn, tack-off, and land. The components of the quadrotor are mentioned in Table 5:

Table 5.1 Description of component of the Tello EDU [22]

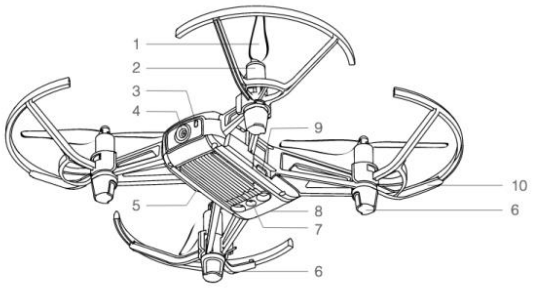
	1. Propellers 2. Engines 3. Drone Status Indicator 4. Camera 5. Power Button 6. Antennas 7. Vision Positioning system 8. Battery 9. Micro USB port 10. Propeller

Table 5.2 Description of the Tello EDU [22]

weight	87 g
Dimensions	98×92.5×41 mm
Propeller	3 inches
Integrated Functions	Telemetric sensor
	Barometer
	LED
	Vision System
	Wi-Fi 2.4 GHz 802.11n
	Real-time streaming 720p
Port	USB battery charging port
Operating temperature range	from 0º to 40º
Operating frequency range	from 2.4 to 2.4835 GHz
Transmitter (EIRP)	20 dBm (FCC)
	19 dBm (CE)
	19 dBm (SRRC)

Being a quadrotor for mainly indoor use, its operations are quite limited. A set of the sensor were placed on the quadrotor so it can be detected by the motion capture system, as shown in Figure 5.1.

Figure 5.1 Quadrotor and the sensor placement

The set of commands can be send in a sequence to the quadrotor, but after the execution of each command 'finish' command from the quadrotor would be expected. Thus, the quadrotor can only execute one command at a time, and if another command is initiated it will not be acknowledged by the quadrotor. This can be characterized as one of the quadrotor's inherent limitation.

Table 5.3 Description of the Tello EDU commands defined in MATLAB

Name	**Description**	**MATLAB code**
ryze	Call Tello SDK to MATLAB workspace	ryze (dronObject)
Take-off command	Initiate quadrotor's take-off	takeoff(dronObject)
Move command	Move the quadrotor in x-y-z direction at the same time	move(dronObject,[x y z],'Speed',0.8);
Turn command	Turn the quadrotor clockwise or counter-clockwise at any given degree	turn(dronObject,deg2rad(angle))
Land command	Start the landing process	land(dronObject)

Another limitation imposed by the quadrotor is related to its maneuvering; the displacement in any direction must be within the range of -5 m and 5 m. Additionally, at least one of the displacements along x- or y-axis must be greater than 0.2 m or less than -0.2 m; otherwise, the quadrotor will stop immediately. To control the quadrotor through MATLAB, a Wi-Fi communication link is established by typing ryze (dronObject) in the command window. The quadrotor must have a separate workspace as the Tello command will pause the program until it receives a response from the quadrotor (Tello). This unwanted pause will cause a delay in receiving data from the motion capture system and execution of the rest of the commands.

5.2 Mobile Platform

The mobile platform is a two-wheel car, whose motions are controlled by a motor controller. The mobile platform is programmed through Arudino to perform defined sequences without the desire of any feedback sensors and external communication. The mobile platform dimensions are 45 cm by 45 cm and a height of 15 cm. The sensors were placed at the corners of the mobile platform, as shown in Figure 5.2.

Figure 5.2 Mobile platform

5.3 Opti-Track System and Setup

Opti-Track is one of the biggest motion capture providers, offering high-performance tracking using motion capture software and high-speed tracking cameras.

Motion capture (MoCap) is the process of digitally tracking and recording the movements of objects or people in space. It involves measuring the position as well as the orientation of the objects or people in the physical space [23]. Depth sensitive cameras by projecting light towards an object can estimate depth based upon the time delay from light emission to backscattered light detection. Optical-passive technology is be used to triangulate the location of retro-reflective rigid bodies attached to the targeted subject. It is the most practical and commonly used method in the industry [24].

Opti-Track is a high-performance motion capture system that is based upon optical-passive technology. A single Opti-Track motion capture system may track the 6 degrees of freedom (DOF) pose of one or more objects in the workspace [24]. The objects meant for tracking are defined as rigid bodies. Each rigid body is a cluster of reflective markers arranged in a unique configuration. Figure 5.3 shows a quadrotor defined as a rigid body.

Figure 5.3 Quadrotor and marker placement

The camera model for the setup is Prime 13 (Figure 5.4) that offers high-speed tracking with excellent precision in a medium-sized capture volume with megapixel resolution, and 240 frames per second (FPS).

Figure 5.4 Opti-Track cameras (Prime 13)

Motive is the software provided by the same company to connect and sync the cameras with each other and is responsible for processing Opti-Track cameras. The obtained data contains global 3D positions, marker IDs, rotational data (concerning the object's local reference). Figure 5.5 shows the interface of the Motive software.

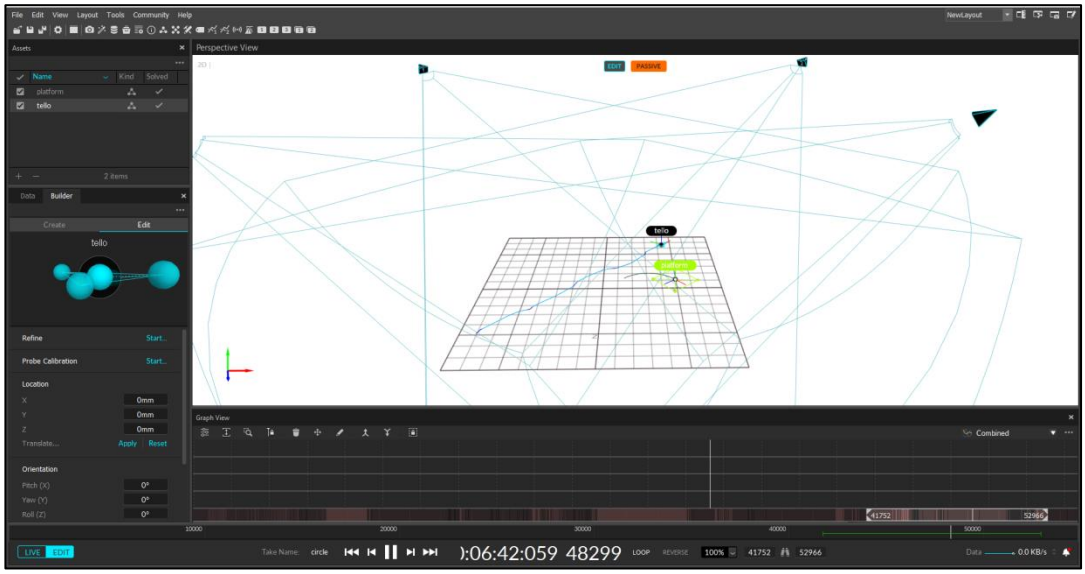


Figure 5.5 MOTIVE program interface

The setup area is defined according to the recommendation of motion capture system as 6.55m x 4.23m and a height of 2.5 m. The laboratory setup where cameras are placed is shown in Figure 5.6.

Figure 5.6 Laboratory and camera placements

Four cameras were used for the system and each camera was pointing at the center of the setup area. The calibration wand was used to sync the cameras and the calibration square was used to define the ground plan. All these calibrations are done using Motive. The resulting captured area of 3.21 m x 2.74 m and placement of cameras is depicted in Figure 5.7.

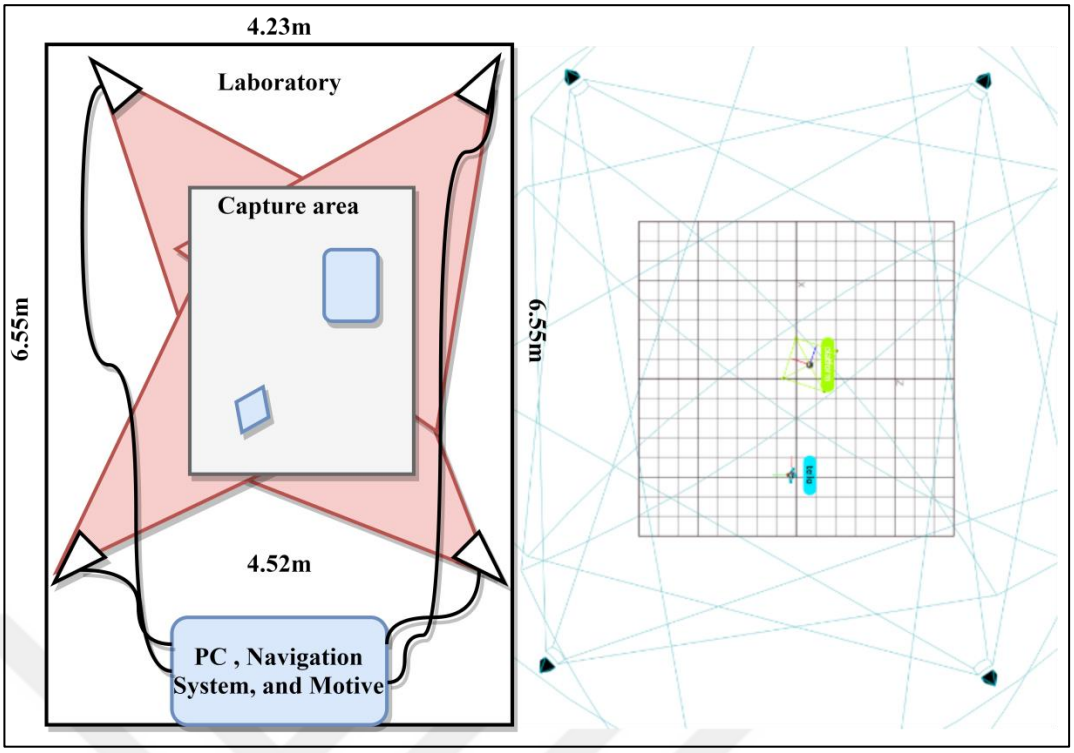


Figure 5.7 Laboratory and camera placements diagram

CHAPTER 6

EXPERIMENTAL RESULTS

6.1 Experimental setup

The setup for the experiments starts by performing the calibration for the motion captures system (Opti-Track) and placing the motion sensor markers on the quadrotor and the mobile platform. The markers are then defined inside Opti-Track software (MOTIVE). This software is then used to transmit the captured data to the local network PC. NAT-NET SDK is used by the MatLab to receive the Motive data stream. The second part is to set the Wi-Fi communication link between the MatLab and the quadrotor (Tello EDU). The command (ryze(dronObject)) is used in the MatLab workspace to establish a connect with Tello SDK. The mobile platform is set up by uploading the desired path to the open-source hardware (Arduino) using an IDE software. These experimental setup procedures are repeated for each experiment scenario to ensure consistency and reliable performance. Figure 6.1 shows the communication chain for the experimental setup.

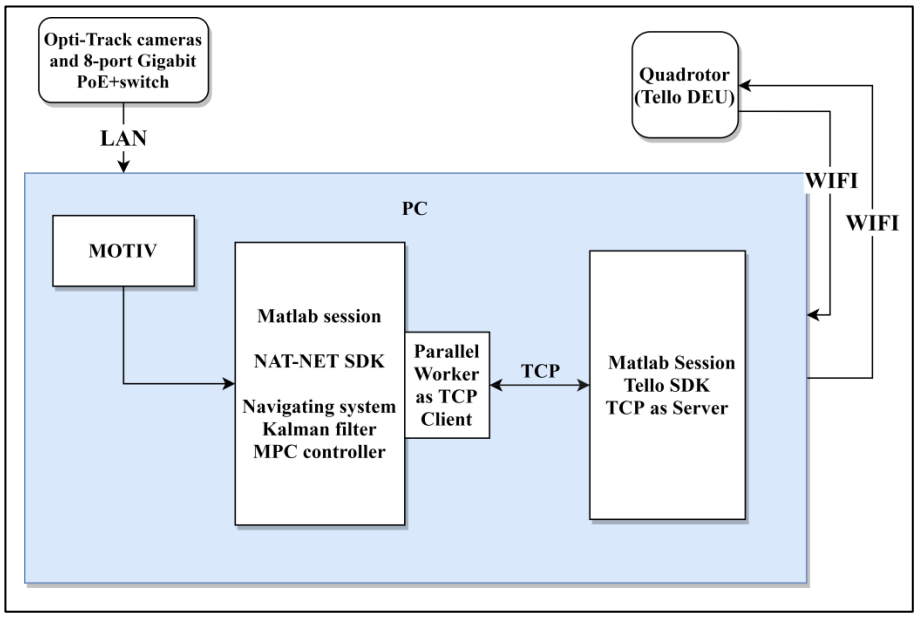


Figure 6.1 Communication chain

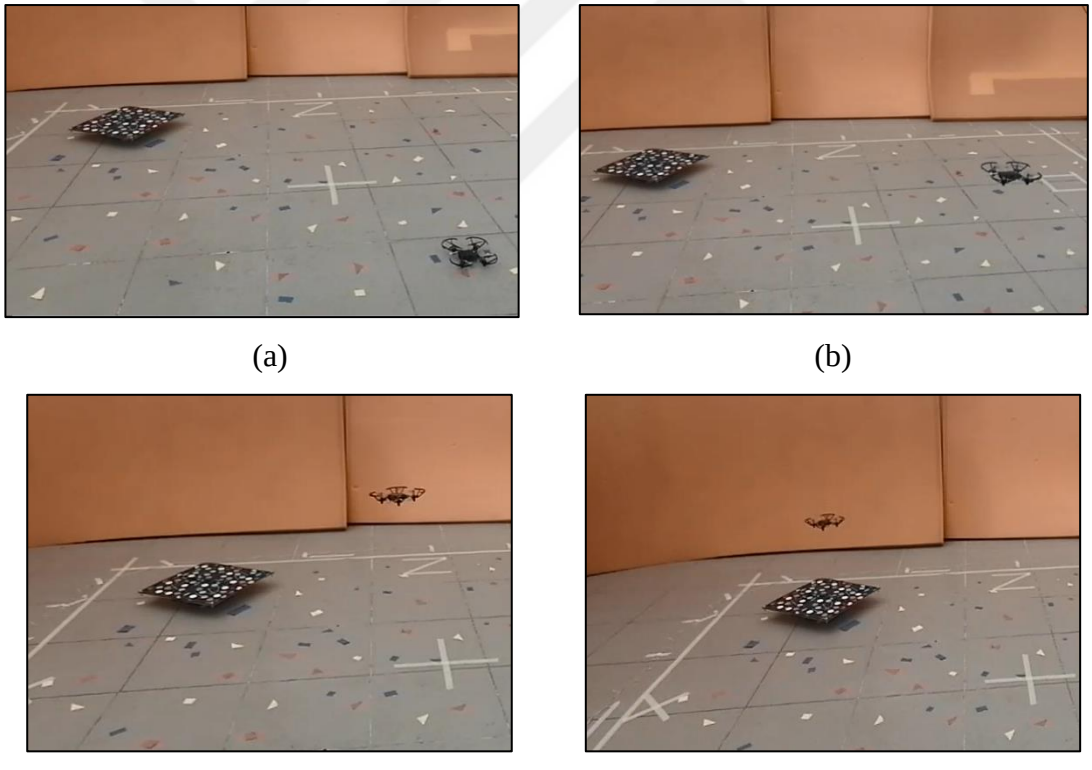
After completing the experimental setup procedure, the software program is initiated. The system flowchart is explained in detail in Section 4.1.

6.2 Experiments Scenarios

To test the effectiveness of the proposed navigation system, a series of four experiments were conducted. These tests were conducted in the lab test area, discussed in detail in Chapter 4.

6.2.1 Static Scenario

The mobile platform was considered stationary and placed at a distance of 2.14 m from the quadrotor. The quadrotor took-off and turned its face towards the mobile platform and then started to move 35 cm as a step until it reached the mobile platform and performed landing. Note that the distance is measured from the center point of the mobile platform to the center point of the quadrotor. Figure 6.2 (a)-(d) shows the complete sequence of events, from taking-off, flying trajectory, to landing.



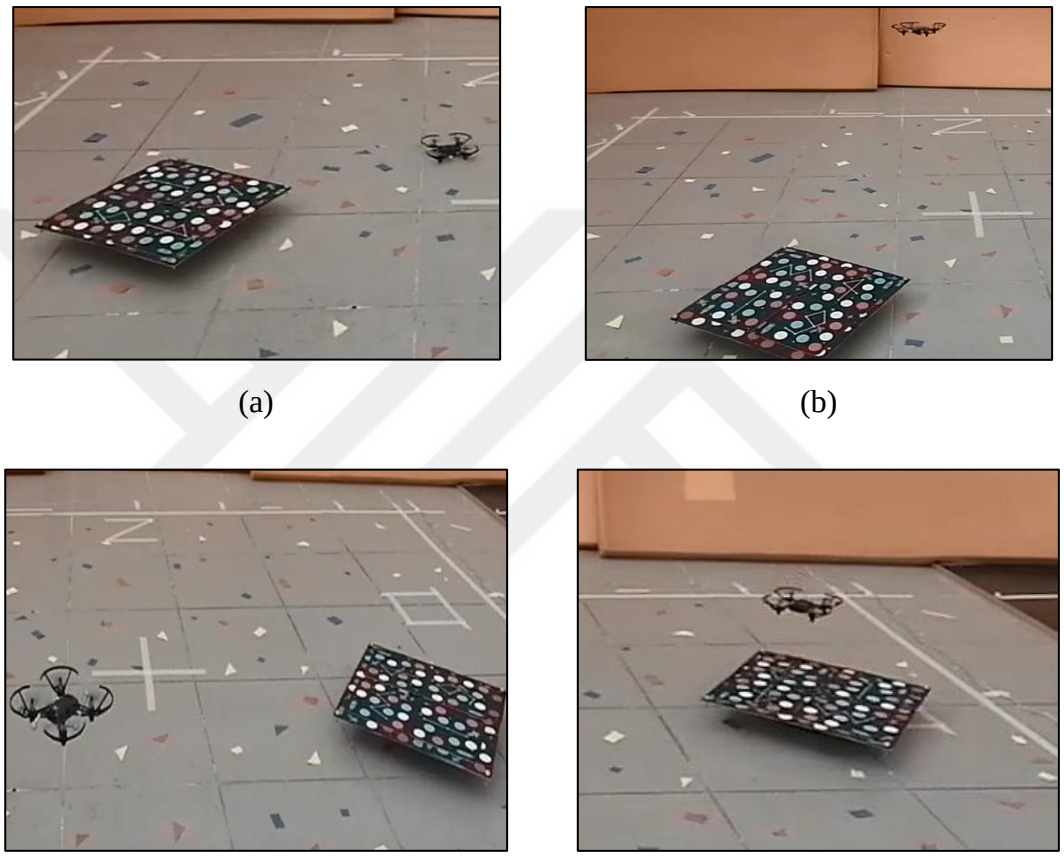
(a) (b)

(c) (d)

Figure 6.2 Sequences of events in the static scenario, (a) initial (b) take-off (c) flying trajectory (d) landing

6.2.2 Square Path Scenario

In this scenario, the mobile platform was placed at a distance of 71 cm from the quadrotor. After two seconds, the platform started to move in a square-shaped path. The quadrotor made several attempts to land on the platform. Finally, the quadrotor reached the mobile platform and performed landing on a moving platform. Figure 6.3 (a)-(d) shows the complete sequence of events, from taking-off, flying trajectory, to landing.



(a) (b)

(c) (d)

Figure 6.3 Sequences of events in the square-path scenario, (a) initial (b) take-off (c) flying trajectory (d) landing

6.2.3 Circle Path Scenario

The mobile platform was placed at a distance of 1.14 m from the quadrotor. The mobile platform started to move in a circle-shaped path with a 1-meter diameter. The quadrotor after take-off and stabilizing itself turned towards the predicted location of the mobile platform and moved in a step of 45 cm every 0.5 seconds. After several attempts, the quadrotor landed successfully on the moving mobile platform Figure

6.4 (a)-(d) shows the complete sequence of events, from taking-off, flying trajectory, to landing.

(a)

(b)

(c)

(d)

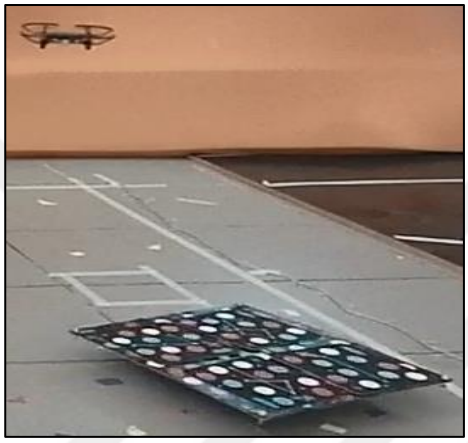
Figure 6.4 Sequences of events in the circle-path scenario, (a) initial (b) take-off (c) flying trajectory (d) landing

6.2.4 Random path Scenario

The mobile platform was set to move in a random-shaped path, and the quadrotor was placed at a random location in the captured area. The quadrotor took-off, stabilized itself, and waited for a few seconds to receive the predicted location from MatLab. The quadrotor turned, moved 35 cm, and landed successfully on the mobile platform in just one attempt. Figure 6.5 (a)-(d) shows the complete sequence of events, from taking-off, flying trajectory, to landing.

(a)

(b)



(c)

(d)

Figure 6.5 Sequences of events in the random-path scenario, (a) initial (b) take-off (c) flying trajectory (d) landing

6.3 Experiment Results

6.3.1 Static Scenario

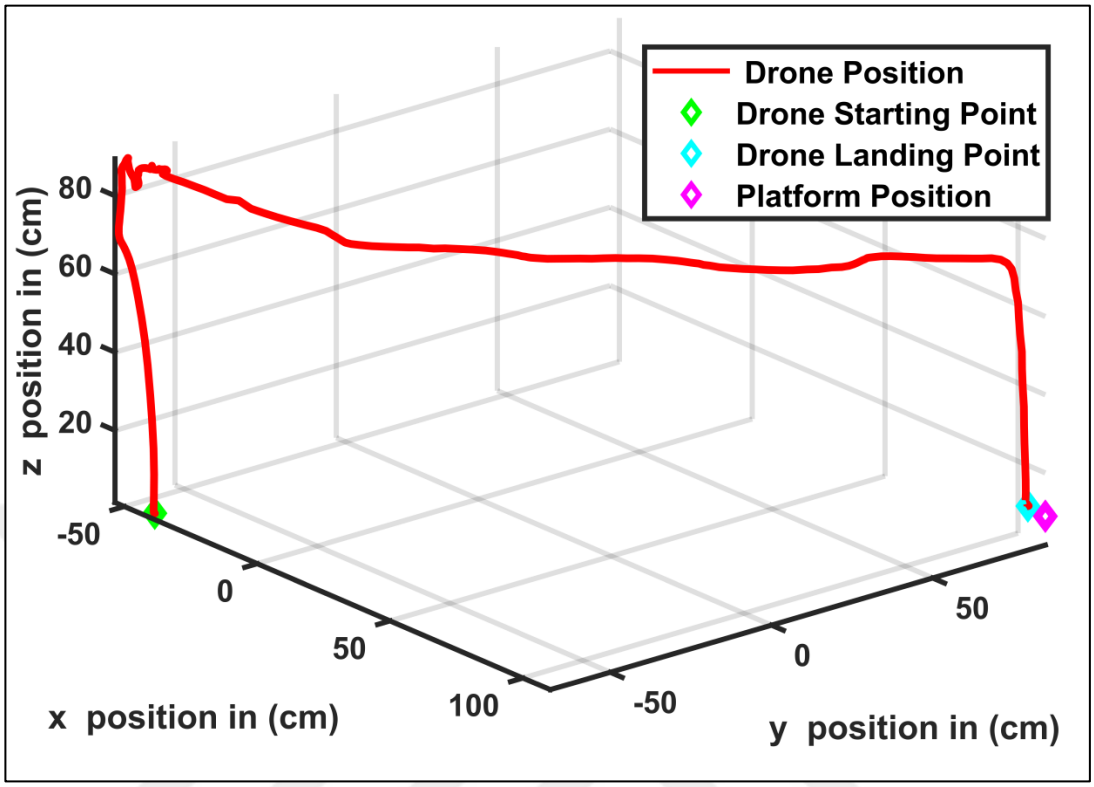


Figure 6.6 Quadrotor and mobile platform traveling path

Figure 6.6 shows the traveling path for the quadrotor and the mobile platform and their starting and ending points. The initial distance between them is defined as 2.14 m. Note that the 3-dimensional figure (Figure 6.6) does not start from the camera origin points of the x,y,z-axis, instead it was rotated to show the best traveling path. Since the platform was stationary, the quadrotor was able to land successfully in merely one attempt.

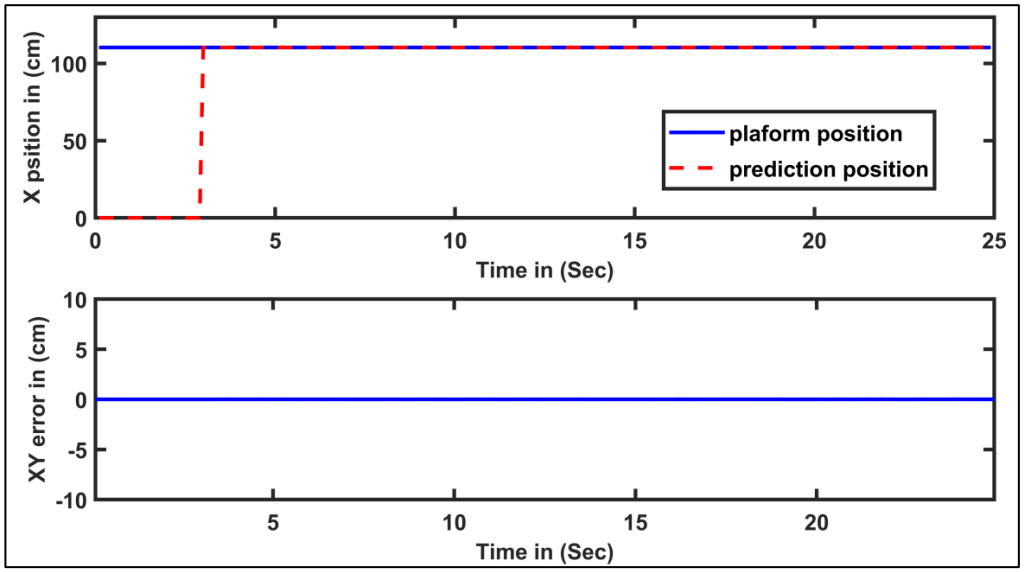


Figure 6.7 Prediction and the actual position of mobile platform in x-axis

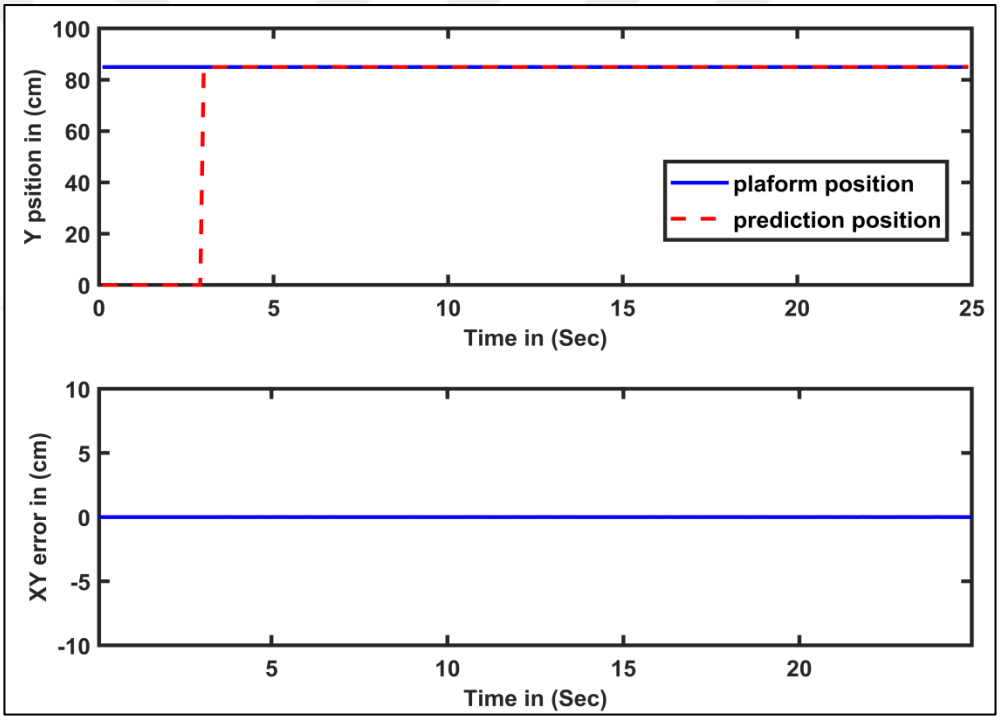


Figure 6.8 Prediction and the actual l position of mobile platform in y-axis

Figure 6.7 and 6.8 shows the predictions and actual position and the error between the X and Y-axis of both prediction and the actual position of the platform. It is clear from the figures that the platform's future prediction has zero error since the platform was stationary.

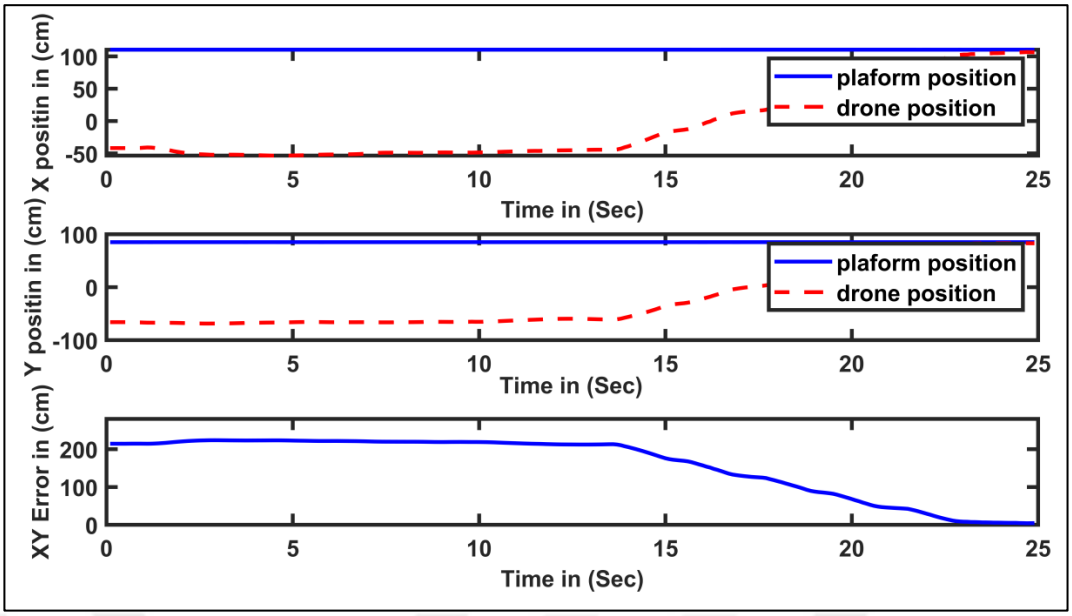


Figure 6.9 Quadrotor position and mobile platform position regarding the distance between them

Figure 6.9 shows the distance between the mobile platform and the quadrotor on the x and y-axis. The XY error represents the horizontal distance between the quadrotor and the platform. At t=14 sec , the quadrotor starts moving towards the platform and once the quadrotor reaches the platform it lands on it, as shown in Figure 6.10.

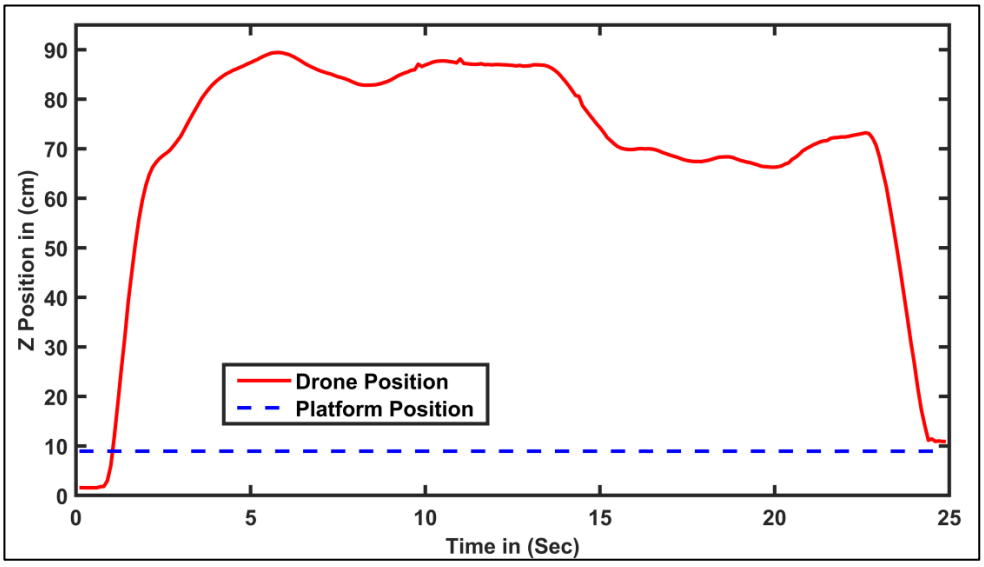


Figure 6.10 Quadrotor position and mobile platform position in z-axis

Figure 6.10 shows the change in heights for the mobile platform and the quadrotor during the experiment. The mobile platform height is measured as 15cm.

6.3.2 Square-Path Scenario

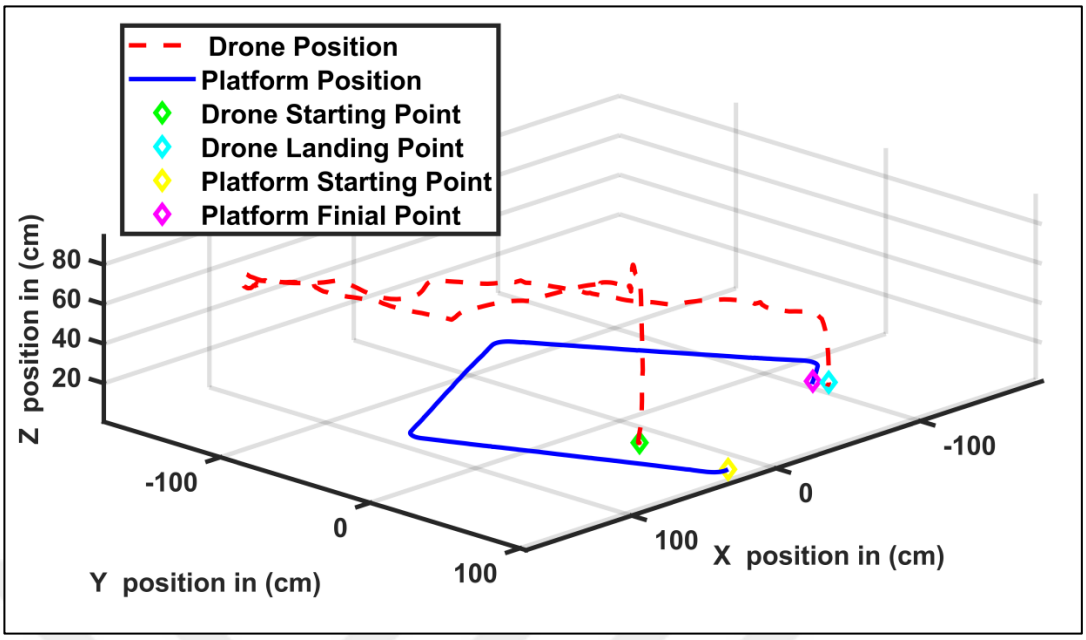


Figure 6.11 Quadrotor and mobile platform traveling path

Figure 6.11 shows the traveling path for the quadrotor and mobile platform and their starting and ending points. The initial distance between them is defined as 71 cm. The quadrotor takes off and waits for the first prediction. After every prediction, the quadrotor starts with turning towards the predicted target point and move in the 35 cm step. The platform was continuously moving in a square-shape during the experiment. After several attempts, the quadrotor successfully landed on the moving platform while it was moving.

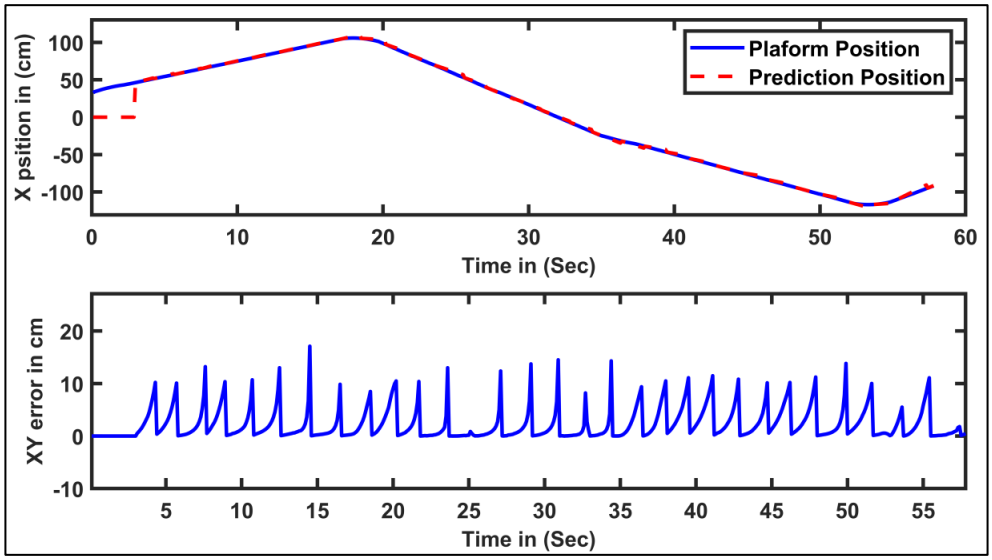


Figure 6.12 Prediction and the actual position of mobile platform in x-axis

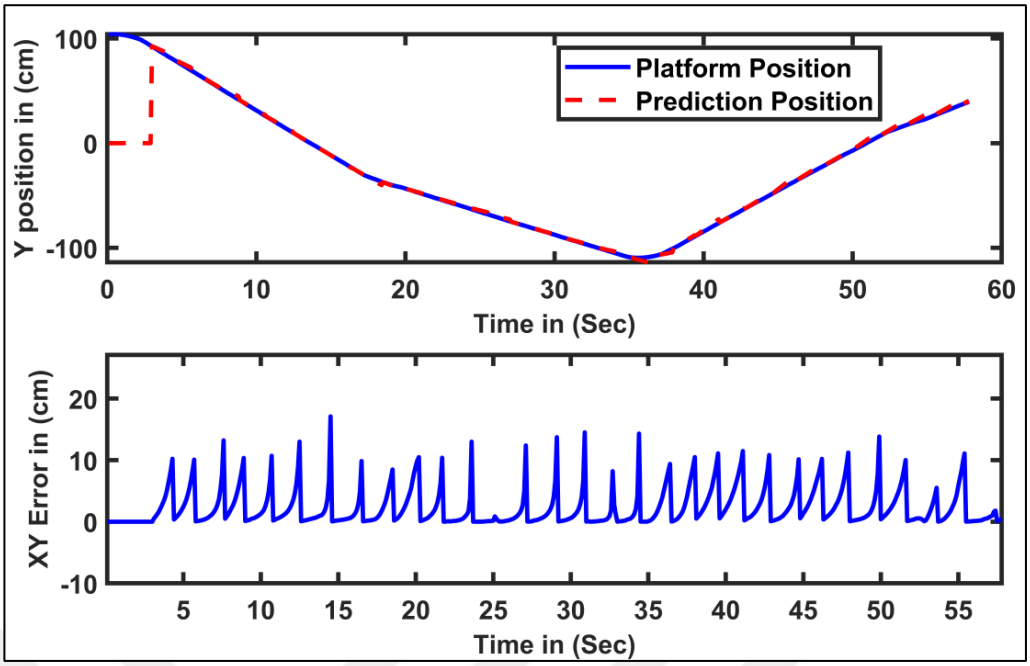


Figure 6.13 Prediction and the actual position of mobile platform in y-axis

Figure 6.12 and 6.13 show the actual and the predicted position and the error between them along X and Y-axis. It is clear from the figures that prediction of the platform future position has some errors since it is in dynamic state. The maximum predicted error is recorded as 10cm, which is considered acceptable.

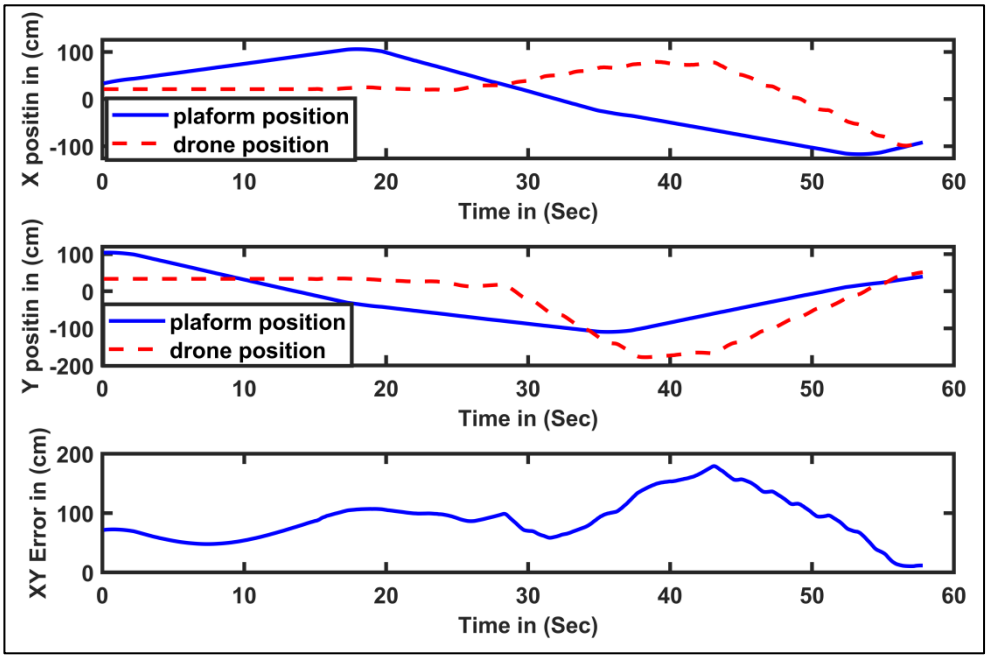


Figure 6.14 Quadrotor position and mobile Platform position regarding the distance between them

Figure 6.14 shows the distance between the mobile platform and the quadrotor on the x and y-axis. The XY error represents the horizontal distance between the quadrotor and the platform. At time t=38 sec, the distance travelled in x-axis is almost zero but the distance in y-axis is not as the quadrotor did not performed the landing yet. At t=34.5 sec, the error in y-axis is almost zero but the distance in x-axis is not as the quadrotor has not initiated the landing yet. At t= 56 sec, the error in both x-axis and y-axis was near zero and the quadrotor was able to land successfully.

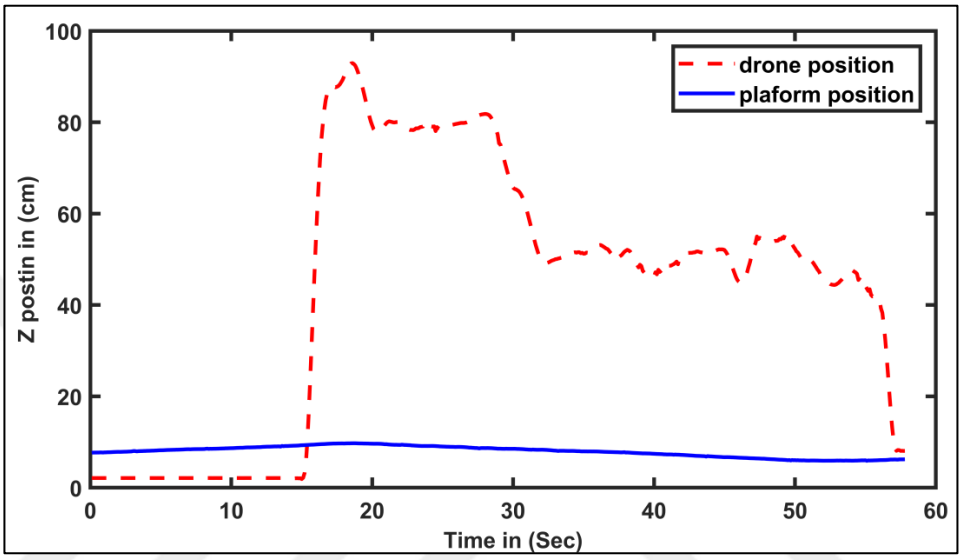


Figure 6.15 Quadrotor position and mobile Platform position in z-axis

Figure 6.15 shows the elevation of the quadrotor and the mobile platform. When the quadrotor takes-off, it attains the height of 90 cm. The quadrotor starts descending gradually until it reaches the target.

6.3.3 Circle-Path Scenario

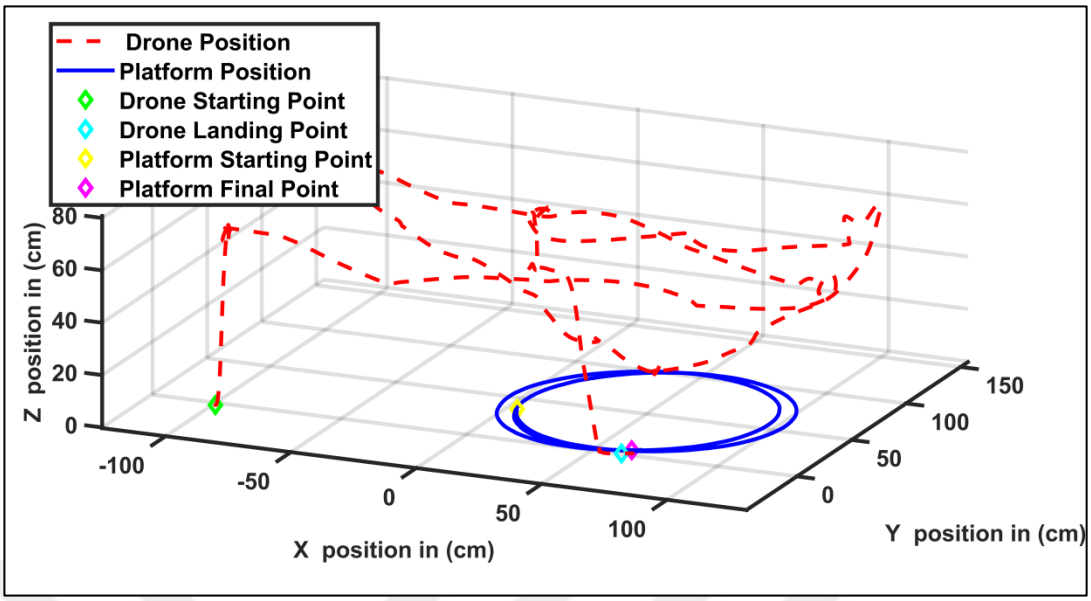


Figure 6.16 Quadrotor and mobile platform traveling path

Figure 6.16 shows the traveling path for the quadrotor and mobile platform along with the starting and ending points. The distance between them at the starting time is recorded as 1.15 m. The quadrotor takes-off and waits for the first prediction to be made. After every prediction, the quadrotor starts with turning toward the predicted target point and take a 35 cm step. The platform is continuously moving in a circle-shape during the experiment. After several attempts, the quadrotor manages to land on the moving platform. In this experiment, the quadrotor struggled to follow the commands of the navigation and control system. Due to the high rate of the command, the quadrotor suffered from technical issues and started skipping commands. This test scenario exposed the technical limitation of the Tello quadrotor.

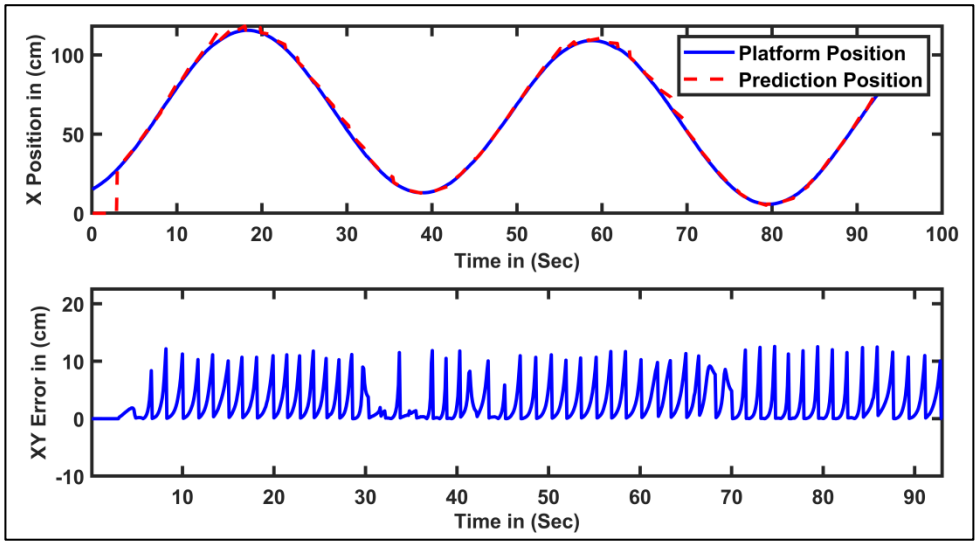


Figure 6.17 Predicted and the actual position of the mobile platform in x-axis

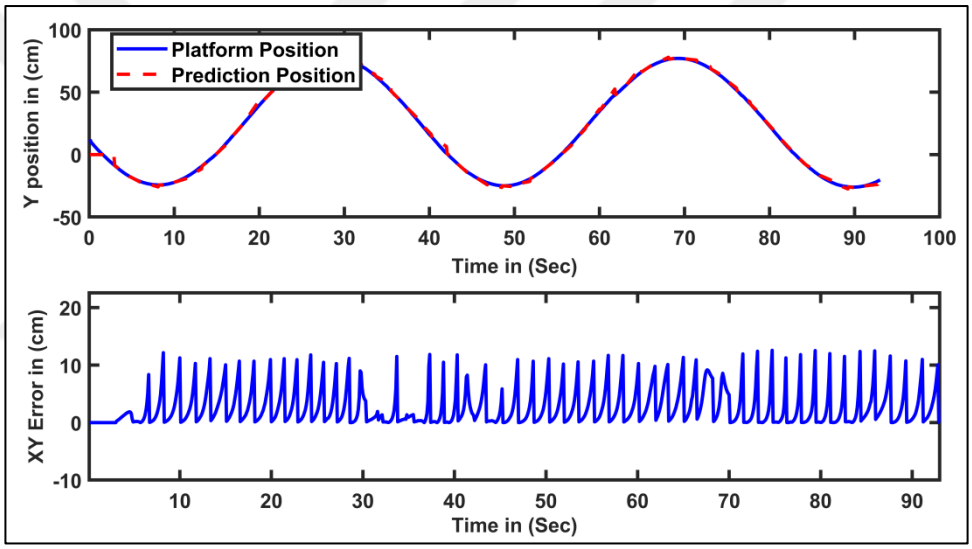


Figure 6.18 Predicted and the actual position of the mobile platform in y-axis

Figure 6.17 and 6.18 shows the predicted and the actual positions of the platform and the error between them along X and Y-axis. It can be observed from the results that prediction of the platform's future position has some errors, due to being dynamic. The maximum predicted error is recorded as 15cm, which is considered acceptable.

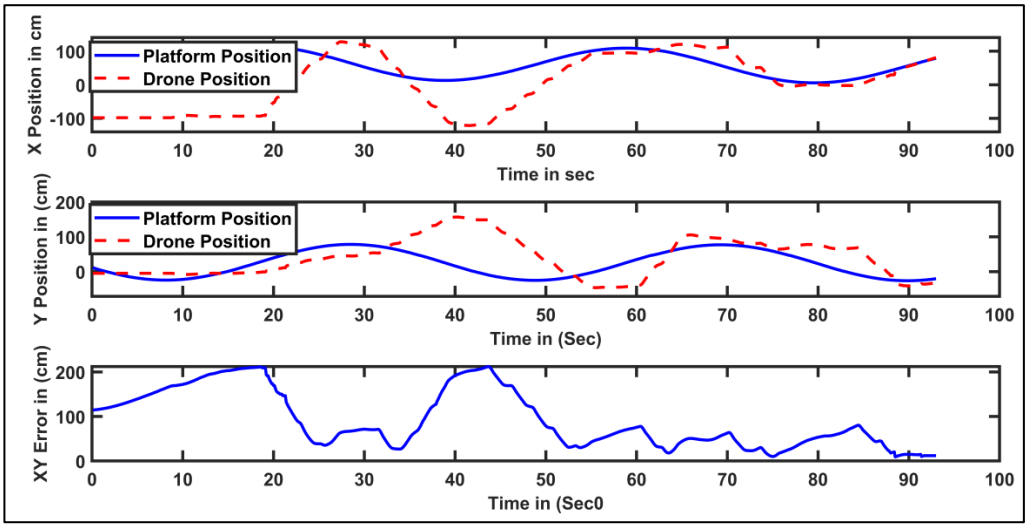


Figure 6.19 Quadrotor position and mobile platform position regarding the distance between them

Figure 6.19 shows the distance between the mobile platform and the quadrotor on the x and y-axis. The XY error represents the horizontal distance between the quadrotor and the platform. The distance between the platform and the quadrotor almost reaches zero at 25, 36, 61.5, and 74 sec in the x direction. However, the error was not the same along y axis at these instances. Similarly, the error almost reaches zero in the y-axis at 12, 32, 53, and 63 sec. Finally, at t=88 sec, the quadrotor distance from the platform is measured less than 10 cm, so the quadrotor can now land. As it can be noticed from figure 6.19, the quadrotor is very close to landing at t=35 sec. However, due to the technical issue with the Tello quadrotor, it stopped responding which causes it to drift away from the target.

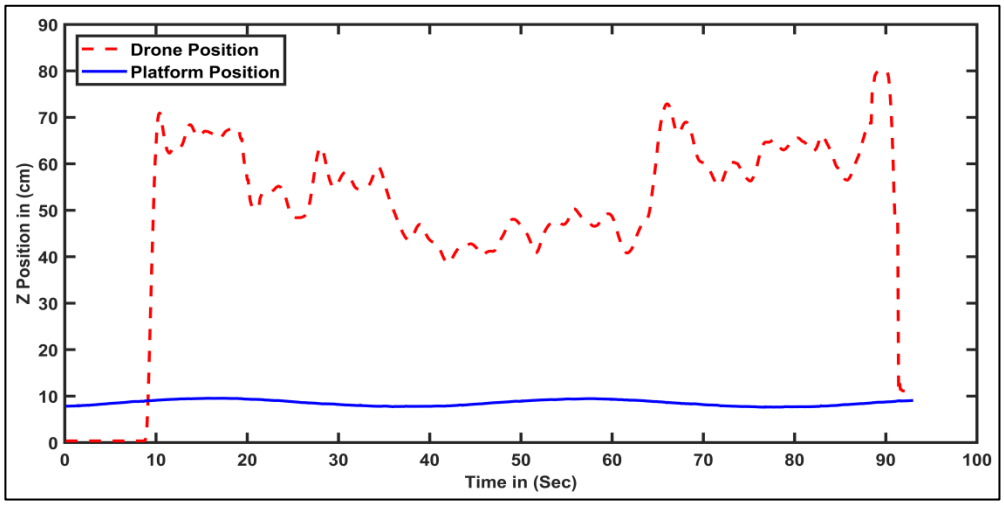


Figure 6.20 Quadrotor position and mobile platform position in z-axis

Figure 6.20 shows the elevation of the quadrotor and the mobile platform. The quadrotor takes-off and stabilizes itself and starts moving toward the mobile platform. However, due to a technical issue with the Tello quadrotor, the quadrotor starts ascending unexpectedly. After getting stabilized, the quadrotor resumes travel towards the target and initiate the landing at t=90 sec.

6.3.4 Random-Path Scenario

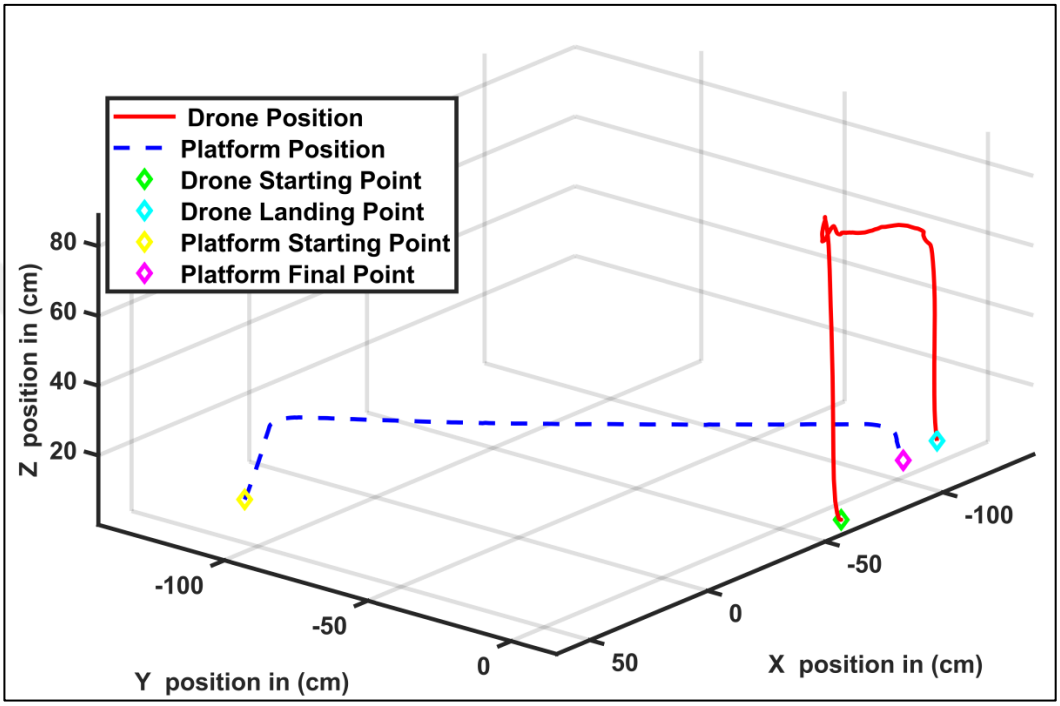


Figure 6.21 Quadrotor and mobile platform traveling path

Figure 6.21 shows the traveling path for the quadrotor and the mobile platform as well as the starting and ending points for both. The mobile platform is moving randomly in the captured area and the quadrotor is also placed at a random location. The quadrotor takes-off and turn almost 180 degrees and move 35 cm towards the target and then land on it.

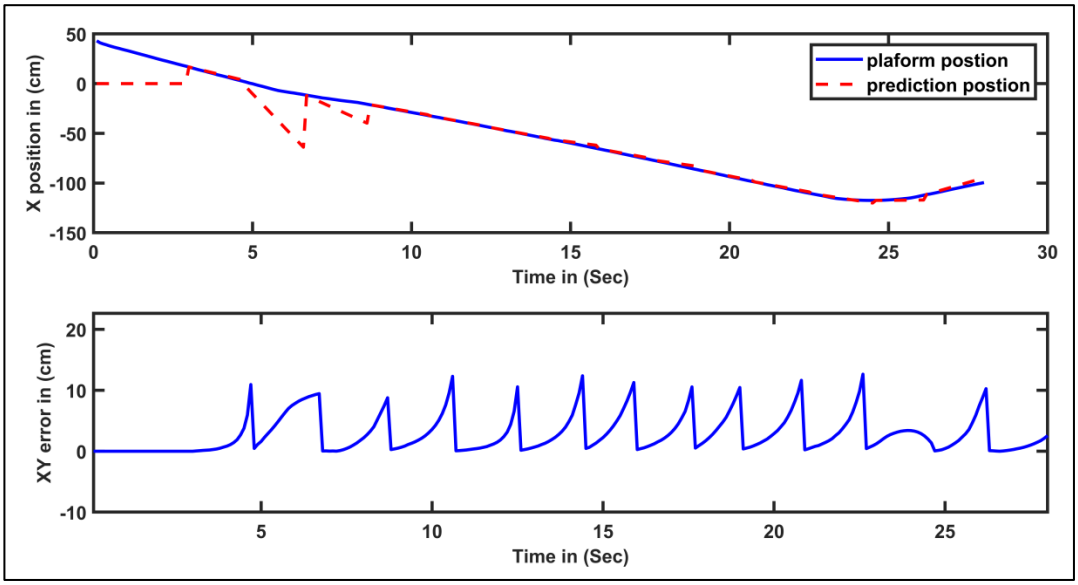


Figure 6.22 Predicted and the actual position of the mobile platform in x-axis

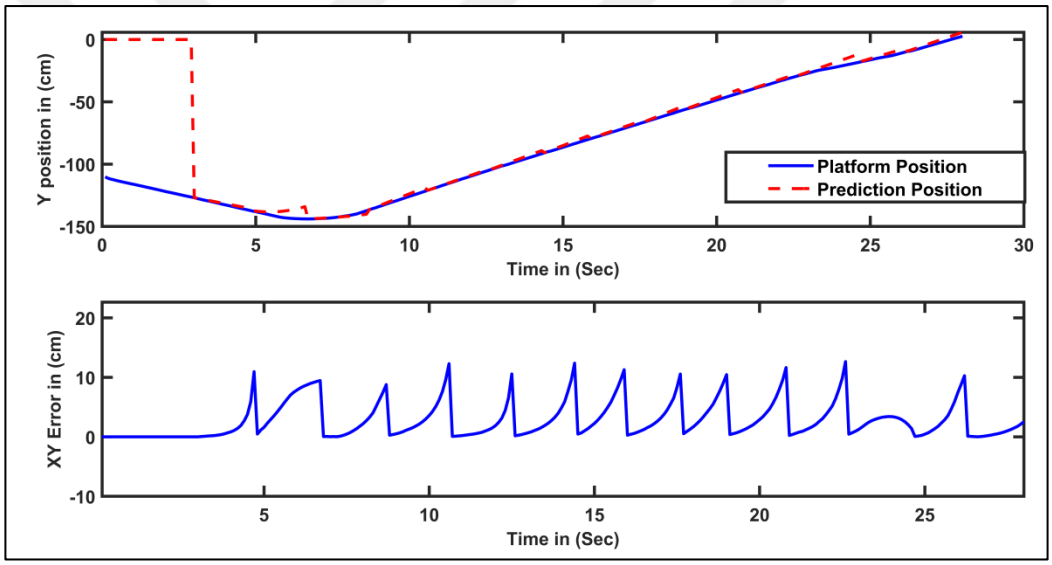


Figure 6.23 Predicted and the actual position of the mobile platform in y-axis

Figure 6.22 and 6.23 shows the predicted and the actual positions of the platform and the error between them. The high prediction error appeared due to the misinformation caused due to the occlusion of one motion sensor. As soon as it is re-detected, the system corrected itself to obtain true predictions.

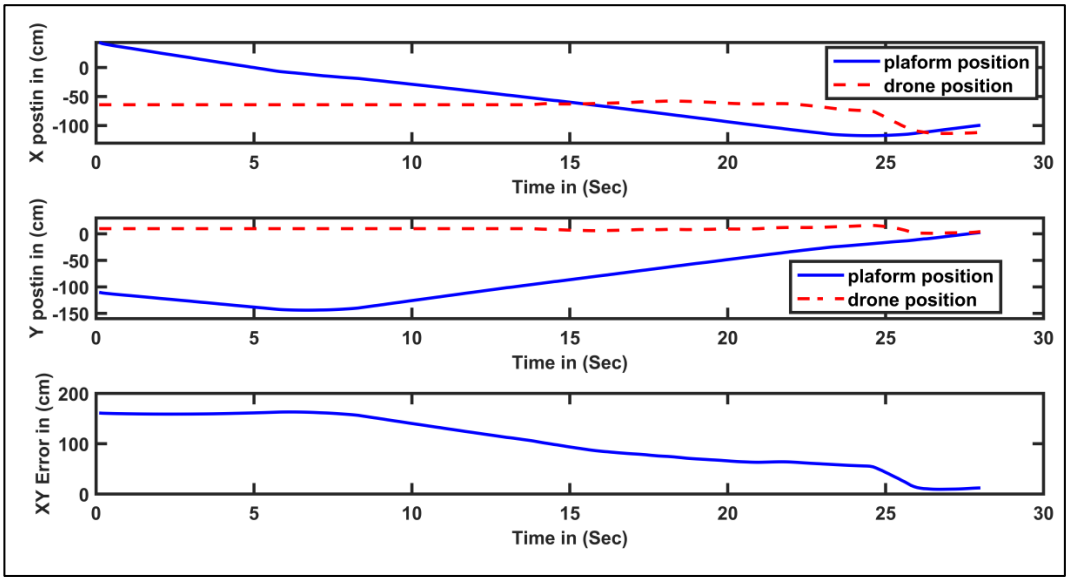


Figure 6.24 Quadrotor position and mobile platform position regarding the distance between them.

Figure 6.24 shows the distance between the mobile platform and the quadrotor in both x and y-axis and the combined error. At t=15 sec, the individual distance measured on the x-axis is almost zero but not along y-axis. Thus, the land operation is not executed. At t=26 sec, the combined error almost reaches zero, thus the quadrotor can land on the mobile platform.

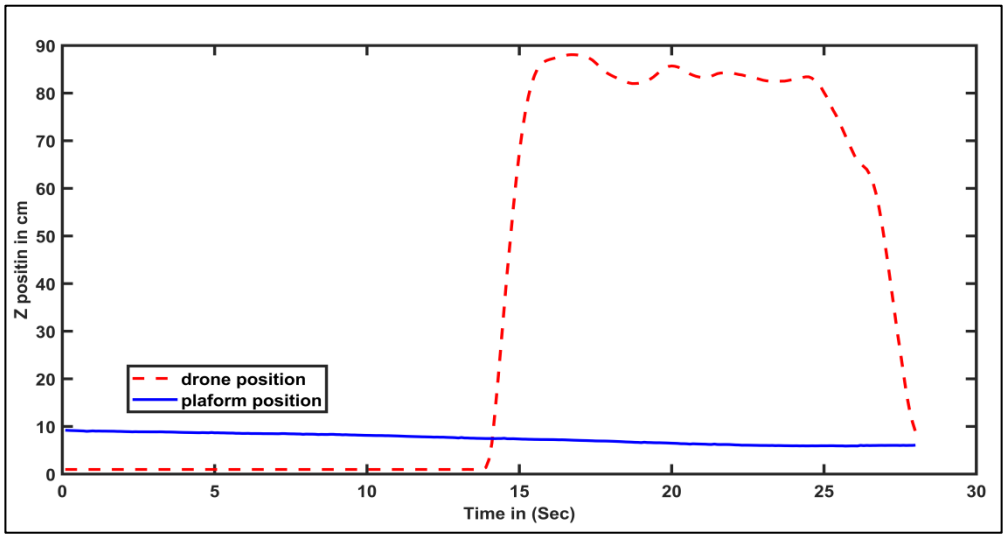


Figure 6.25 Quadrotor position and mobile platform position in z axis.

Figure 6.25 shows the elevation of the mobile platform and the quadrotor. The quadrotor take-off and started to descend at t=20 sec. As the platform was recorded within the allowed landing distance, the quadrotor initiated to land.

CHAPTER 7

CONCLUSION

In the present study, the quadrotor (Tello EDU), the Kalman filter, the model predictive controller, and the motion capture system (Opti-Track) have been used to design the system whose role is to achieve an autonomous landing of the quadrotor on a moving mobile platform. The main task of the autonomous take-off and landing that defines the success of the whole operation includes tracking, predicting, and landing.

The quadrotor involved in this study has many limitations, such as not being able to receive continuous commands and only accepts commands when not executing any command. To overcome this limitation, a TCP server/client program is developed. The TCP server runs on a separate MatLab session. The TCP server queue up the commands from the main navigation MatLab program that are needed to control the quadrotor. The TCP Server continuously monitors the quadrotor status and only sends a command when the quadrotor is available. This solution helps in improving the delay time caused by the quadrotor's SDK.

To provide a smooth and noise-free information to the control algorithm, a Kalman filter is employed. Kalman filter is used for tracking the mobile platform and estimate the velocity and acceleration of the mobile platform. To ensure the successful landing of the quadrotor, only the real-time data do no suffice. Using Kalman filter, the quadrotor can track the moving mobile platform very well. However, landing the quadrotor on the mobile platform took a longer time than desired. Hence, a second Kalman filter is used to predict the future locations of the mobile platform using the last known state. Kalman filter helped to accurately predict the future position of the mobile platform; consequently, the timing involved for landing is also reduced.

The second limitation of the quadrotor and the major obstacle in controlling the quadrotor is not having the accessibility to change the build-in designed code of the

quadrotor. Thus, the controller is designed following the constraints introduced by the system. Through simulation results, it is observed that the MPC is more effective for fast quadrotor dynamics than linear control methods, such as PID. Thus, MPC is chosen to control the quadrotor horizontal position and orientation.

Several experiments are conducted to test the designed navigation system. The experiments involve different movements of the mobile platform, such as square, circle, and random. The navigation system effectively tracks the mobile platform, predict its future location, and achieved successful landing in all the experiments.

7.1 Future work

The future work can focus on the improvement of the model predictive control by introducing different noise and disturbance models. Another potential area is to work upon the Kalman prediction by involving the estimating of mobile platform elevation.

REFERENCES

[1] M. Fu, K. Zhang, Y. Yi and C. Shi, “Autonomous landing of a quadrotor on an UGV,” in *2016 IEEE International Conference on Mechatronics and Automation*, 2016, pp. 988-993.

[2] X. Liu, S. Zhang, J. Tian and L. Liu, “An Onboard Vision-Based System for Autonomous Landing of a Low-Cost Quadrotor on a Novel Landing Pad,” *Sensors*, vol. 19, no. 21, p. 4703, 10 2019.

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

[4] Vicon, Apr 2020. [Online]. Available: https://www.vicon.com/about-us/what-is-motion-capture/.

[5] OptiTrack, “About OptiTrack,” 2021. [Online]. Available: https://optitrack.com/about/. [Accessed 1 Oct. 2021].

[6] G. Ducard and R. D'Andrea, “Autonomous quadrotor flight using a vision system and accommodating frames misalignment,” in *2009 IEEE International Symposium on Industrial Embedded*, 2009, pp. 261-264.

[7] F. Yu, G. Chen, N. Fan, Y. Song and L. Zhu, “Autonomous flight control law for an indoor UAV quadrotor,” in *2017 29th Chinese Control And Decision Conference (CCDC)*, 2017, pp. 6767-6771.

[8] A. Mashood, A. Dirir, M. Hussein, H. Noura and F. Awwad, “Quadrotor object tracking using real-time motion sensing,” in *2016 5th International Conference on Electronic Devices, Systems and Applications (ICEDSA)*, 2016, pp. 1-4.

[9] S. Durand, J. Dumon, N. Marchand and J. F. Guerrero-Castellanos, “Event-based control for embedded and networked system application to a mini quadrotor helicopter using motion capture,” in *2014 International Conference*

on Unmanned Aircraft Systems (ICUAS), 2014, pp. 1188-1195.

[10] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering*, vol. 82, pp. 35-45, 03 1960.

[11] Q. Li, R. Li, K. Ji and W. Dai, "Kalman Filter and Its Application," in *2015 8th International Conference on Intelligent Networks and Intelligent Systems (ICINIS)*, 2015, pp. 74-77.

[12] L. V. Santana, A. S. Brandão, M. Sarcinelli-Filho and R. Carelli, "A trajectory tracking and 3D positioning controller for the AR.Drone quadrotor," in *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2014, pp. 756-767.

[13] X. Song, L. D. Seneviratne and K. Althoefer, "A Kalman Filter-Integrated Optical Flow Method for Velocity Sensing of Mobile Robots," *IEEE/ASME Transactions on Mechatronics*, vol. 16, no. 3, pp. 551-563, June 2011.

[14] S. a. M.-C. F. a. H. P. a. F. J. a. S. H. {Mahfouz, "Target Tracking Using Machine Learning and Kalman Filter in Wireless Sensor Networks," *IEEE Sensors Journal*, vol. 14, no. 10, pp. 3715-3725, Journal 2014.

[15] S. J. Qin and T. A. Badgwell, "A survey of industrial model predictive control technology," *Control Engineering Practice*, vol. 11, no. 7, pp. 733-764, 2003.

[16] B. Chen, W.-L. Cao, H.-C. Zhang and L. Feng, "Model Predictive Control Research Based on Auto Landing System," in *2007 International Conference on Machine Learning and Cybernetics*, 2007, pp. 398-402.

[17] Y. Zhou and K. Takaba, "Optimal Landing Control of an Unmanned Aerial Vehicle via Partial Feedback Linearization," in *2019 International Conference on Advanced Mechatronic Systems (ICAMechS)*, 2019, pp. 218-223.

[18] J. A. Macés-Hernández, F. Defaÿ and C. Chauffaut, "Autonomous landing of an UAV on a moving platform using model predictive control," in *2017 11th Asian Control Conference (ASCC)*, 2017, 2298-2303.

[19] A. Reda, A. Bouzid and J. Vásárhelyi, “Model Predictive Control for Automated Vehicle Steering,” *Acta Polytechnica Hungarica*, vol. 17, no. 7, pp. 163-182, 09 2020.

[20] NaturalPoint Help Center, 07 2015. [Online]. Available: https://help.naturalpoint.com/kb/articles/transform-world-space-coordinates-to-local-rigid-body-coordinates. [Accessed 5 Oct. 2021].

[21] K. H. Ang, G. Chong and Y. Li, “PID control system analysis, design, and technology,” *IEEE Transactions on Control Systems Technology*, vol. 13, no. 4, pp. 559-576, 2005.

[22] Tello EDU, 2020. [Online]. Available: https://www.ryzerobotics.com/tello-edu/specs. [Accessed 5 Oct. 2021].

[23] P. Nogueira, “Motion capture fundamentals,” in *Doctoral symposium in informatics engineering*, 2011, pp. 303.

[24] J. S. Furtado, H. H. Liu, G. Lai, H. Lacheray and J. Desouza-Coelho, “Comparative Analysis of OptiTrack Motion Capture Systems,” in *Advances in Motion Sensing and Control for Robotic Applications*, 2019, pp. 15-31.