ROYAL MILITARY COLLEGE OF CANADA
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

M.A.Sc. Thesis

Multi Model State Estimation and Control for Target Tracking UAVs

Sean Wolfe

Supervisors

Dr. Sidney Givigi
Dr. Camille-Alain Rabbath

January 31, 2020

Abstract

Wolfe, Sean. M.A.Sc. Royal Military College of Canada, November, 2019. *Cooperative Multi Model State Estimation and Control for Target Tracking UAVs*. Supervised by Dr. Sidney Givigi and Dr. Camille-Alain Rabbath.

In this thesis, the idea of using teams of Unmanned Aerial Vehicles (*UAV*s) to track a ground vehicle and exploiting the benefits of multiple UAVs is considered. First, a novel state estimation algorithm is developed using the standard Extended Kalman filter (*EKF*). This filter is distributed over a team of three UAVs, each adopting a different model for the target's motion. The Distributed EKF (*DEKF*) is then paired with the T Test model selection criteria, which decides which UAV has the best estimate at each iteration. The algorithm is validated in a real-time indoor flight environment and compared against the single model equivalents, as well as compared to other multi model variants. Furthermore, an analysis of different selection methods to use in times of occlusions is done. The second part of the thesis involves the design and testing of a multiple model Distributed Model Predictive Controller (*DMPC*) for tracking in formation flight. Using information from state estimation about which target model is performing best, the DMPC changes its target motion model accordingly. The MPC controller is first implemented for a single UAV, then tested in both a real-time simulation environment and in an indoor flight. This MPC is then expanded to the multi-UAV scenario, which is tested in the same real-time simulation environment.

Keywords: State Estimation, Extended Kalman Filter, Distributed, T Test, Multiple Models, Target Tracking, Unmanned Aerial Vehicle, Control, Model Predictive Control.

Resumé

Wolfe, Sean. M.Sc.A. Collège militaire royal du Canada, Novembre, 2019. *Estimation de l'état utilisant divers modèles et commande en coopération pour drones effectuant un suivi de cible*. Thèse dirigée par Sidney Givigi, Ph.D. et Camille-Alain Rabbath, Ph. D

La thèse considère utiliser une équipe de drones dans le but de suivre une cible mobile au sol. Tout d'abord, un nouvel algorithme est développé avec l'EKF standard comme système de base. Cet EKF est distribué parmi une équipe de trois drones, chacun utilisant un modèle différent pour prédire le mouvement de la cible. Ensuite, l'EKF distribué est combiné avec une méthode de sélection, le test T, qui décide quel drone a la meilleure estimation de la cible. L'algorithme au complet est valide expérimentalement contre l'algorithme équivalent avec un seul modèle dans un environnement intérieur et avec de véritables drones. De plus, une comparaison de performance avec différents algorithmes et plusieurs modèles est effectuée. Finalement, une analyse de différentes solutions en temps qu'aucun des drones détectent la cible est faite. La deuxième partie de la thèse consiste en un contrôle de l'équipe de drones pour suivre la cible d'une manière efficace. Pour accomplir cet objectif, un contrôleur MPC distribué utilisant plusieurs modèles pour le mouvement de la cible est développé et validé. Le contrôleur pour un seul drone est premièrement validé en simulation et dans des essais en vol intérieur. Ensuite, des contraintes de coopération sont ajoutées pour que les drones suivent la cible en formation, ce qui est validé en simulation.

Mots clés : Estimation, EKF, Distribution, Test T, Modèles, Suivi de Cible, Drones, Contrôle, MPC.

Table of Contents

List of Figures

List of Tables

Notation

- $a_{(\cdot)}$ - When $(\cdot)$ is x, y or z, a is acceleration. Otherwise a_{ij} represents entry ij of Matrix A
- $\mathbf{d}(k)$ - Gaussian noise present in prediction process
- c - normalization factor
- $d(k)$ - Gaussian noise present in measurement process
- $f(\cdot)$ - Prediction function
- f_d - Drag force acting on UAV
- f_g - Weight UAV
- f_i - Thrust from motor i
- f_M - Thrust force acting on UAV
- $\mathbf{f}_{tot}$ - Sum of forces acting on UAV
- $g(\cdot)$ - Function represent unmodeled dynamics
- $\mathfrak{g}$ - Cost function representing a two player zero sum game
- h - Horizon
- $h(\cdot)$ - The measurement function transforming the state vector into the measurement basis
- i_A - Anchor current

- i and j- When superscripted as ${}^{i}(\cdot)$, refers to UAV i and its neighbour UAV j. Note i is the UAV running the algrorithms and it has neighbours described by the set $\mathcal{J}$. The set of all UAVs is the set $\mathcal{Z}$. The size of $\mathcal{Z}$ is N, the total number of UAVs
- $j_{(\cdot)}$ - Moments of inertia in the direction of $(\cdot)$
- (k) - Any variable with (k) appended, such as $\mathbf{x}(k)$ represents the evaluation of the variable at time step (k)
- k_f - Final time of an experiment or simulation
- $\mathbf{l}$ - Vector of outputs used for system identification
- l_m - Distance between a rotor and the base frame of UAV
- m - Mass of the quadrotor
- $\mathbf{m}_d$ - Moment induced from drag on UAV
- m_e - Electromagnetic moment on UAV
- m_i - Moment of motor i
- m_m - Mechanical moment on UAV
- $\mathbf{m}_M$ - Total moment from thrust of UAVs
- $\mathbf{m}_{tot}$ - Total moment of UAV
- n - Size of residual window
- o - Number of states in the target model
- p - Number of states in UAV model
- $\mathbf{p}$ - Refers to a three dimensional position vector

- q - number of measurements made at time step k
- r - Distance or when specified radius of a circle
- $\mathbf{r}$ - Relative position. For example ${}^{ij}\mathbf{r}$ is the relative position of UAV j to UAV i
- - ${}^{ij}\mathbf{r}_s$ - Separation for UAV i to maintain with respect to UAV j
- r_a - Collision avoidance distance
- r_c - Communication maintenance distance
- s - Number of inputs in UAV model
- $s_{(\cdot)}$ and $c_{(\cdot)}$ - $\sin(\cdot)$ and $\cos(\cdot)$
- t - Continuous time
- $\mathbf{t}$ - Translation vector
- $tr(\cdot)$ - Trace of $(\cdot)$
- t_{stat} - Metric of how well proposed model is doing
- t_T - Thrust of a UAV motor
- $\mathbf{u}$ - Vector of inputs over horizon
- $\mathbf{u}(k)$ - Input vector at time step k
- u_f - Thrust input of UAV from first principles
- u_{min} and u_{max} - Maximum and minimum inputs allowable for stable UAV flight
- u_x - x input derived by MPC controller
- u_y - y input derived by MPC controller
- u_θ Roll input of UAV from first principles

- u_ϕ - Pitch input of UAV from first principles
- u_ψ - Yaw input of UAV from first principles
- $\sqcap$ - Overall propagated input vector
- $v_{(\cdot)}$ - Speed in the $(\cdot)$ direction. $\mathbf{v}$ is velocity.
- $\mathbf{v}_b$ - UAV velocity vector in body frame
- ${}^w\mathbf{v}_b$ - Wind velocity vector in the body frame
- $\mathbf{v}'$ - Velocity with obstacles
- $\mathbf{v}''$ - Velocity calculated from LGVF method
- $\mathbf{w}$ - Vector of unknowns to be identified through system identification
- x, y or z - Position in x, y and z, or x, y, z, axis depending on context
- $\mathbf{x}$ - State vector
- $\mathbf{x}_r(k)$ - Reference states at time step k
- ${}^ijx_s(k)$ - Distance separation term for UAV pair i, j
- $\S$ - Overall propagated state vector over the horizon
- $\mathbf{z}$ - Measurement vector
- $\mathcal{A}$ - Overall propagated A matrix
- $\mathcal{B}$ - Overall propagated B matrix
- C - Rotation matrix. For example, C_c^w is the rotation matrix going from the world frame to the camera frame

- $C_x^c(\theta)$ - Rotation matrix in the frame defined by c about the axis defined by x according to the angle defined by θ
- C_{axis} - Rotation matrix associated with a change of axis
- $D_{d,f}$ - Matrix of drag coefficients
- $D_{d,M}$ - Matrix of drag coefficients for torques
- E - Sum of all residuals
- $E[(\cdot)]$ - Expectation value of $(\cdot)$
- F - Linearised Prediction function matrix
- FIT - FIT percentage for model identification
- G - Matrix of inputs for system identification
- H - Linearised measurement matrix
- I - Identity matrix
- $\mathfrak{J}$ - Overall cost function
- K - Kalman gain
- K^+ - Kalman gain from combining a posteriori estimates in maximum likelihood method
- L - Moving horizon look back size
- N - Number of UAVs
- P - EKF overall covariance matrix

- Q - In the context of state estimation Q is the process noise covariance matrix. In the context of control, Q is a weighting matrix used in the objective function of an MPC controller
- Q_s - Weighting matrix for maintaining separation between neighboring UAVs
- R - Measurement noise matrix
- S - Projection matrix
- S_d - Standard deviation
- T - Transformation matrix. For example, T_w^b refers to the transformation necessary in going from world frame to the body frame
- T_s - Sample time
- $\alpha_{(\cdot)}$ - Angular acceleration in the $(\cdot)$
- β - Phase angle
- γ - Discount Factor
- $\gamma(\cdot)$ - Likelihood function
- δ - Velocity correction coefficient
- ϵ - Vector of residuals
- θ, ϕ and ψ - Roll, pitch and yaw angles
- κ - Ideal mean of residuals
- λ_{max} - Largest eigenvalue
- $\mu(k|k)$ - Model probability

- ρ - Model switching matrix with elements ρ_{ji}
- υ - Residual vector
- Υ - Innovation Covariance
- Ψ - Flux
- $\omega_{(\cdot)}$ - Angular velocity in the $(\cdot)$ direction
- ω^b - Angular velocity vector of UAV in body frame
- η - Number of measurements made at time step k
- ${}^{CV}(\cdot)$ - $(\cdot)$ is using the CV model. Similarly, ${}^{CA}(\cdot)$ is for the CA model and ${}^{CT}(\cdot)$ for the coordinated turn model.
- ${}_{g}(\cdot)$ - When superscripted, refers to the ground target.
- $(\cdot)^{-1}$ - The inverse of $(\cdot)$
- $(\cdot)^{+}$ and $(\cdot)^{-}$ - When superscripted, represents an a posteriori and a priori value, respectively, in context to the EKF.
- ${}_{c}(\cdot)$ - The subscript c when seen in front of $(\cdot)$ represents combined UAV and target information. For example, ${}_{c}\mathbf{x}$ is a state vector containing both UAV and target states
- ${}_{fg}(\cdot)$ - Final target value
- $(\cdot)_{x,y,z}$ - Represents the x, y or z component of $(\cdot)$
- $(\cdot)_p$ - The subscript or superscript p refers to the PTU unit. For example, θ_p refers to the roll angle of the PTU unit.
- $||\cdot||$ - Euclidean norm operator
- $\hat{(\cdot)}$ - An estimate of $(\cdot)$

- $\bar{(\cdot)}$ - Mean of $(\cdot)$
- $(\cdot)^\star$ - Optimal value of $(\cdot)$
- Boldface - Any symbol in boldface represents a vector
- Fractal Font - Represents a general cost function

Abbreviations

- CA - Constant Acceleration
- CIFER - Comprehensive Identification from Frequency Responses
- CKF - Curbature Kalman filter
- CT - Coordinated Turn
- CV - Constant Velocity
- DEKF - Distributed Extended Kalman Filter
- DMPC - Distributed Model Predictive Controller
- EKF - Extended Kalman Filter
- FIM - Fisher Information Matrix
- GCS - Ground Control Station
- HALE - High Altitude Long Endurance
- HLP - High Level Processor of AscTec Pelican
- IBM - Image-Based Navigation
- IFDS - Interfered Fluid Dynamical System
- IMU - Inertial Measurement Unit
- ISTAR - Intelligence, Surveillance, Target Acquisition and Reconnaissance

- JLF - Joint likelihood function
- KF - Kalman Filter
- LBMPC - Learning Based Model Predictive Control
- LALE - Low Altitude Long Endurance
- LASE - Low Altitude Short Endurance
- LLP - Low Level Processor of AscTec Pelican
- LOS - Line of Sight
- LTI - Linear Time-Invariant
- MALE - Medium Altitude Long Endurance
- MHE - Moving Horizon Estimation
- MMDEKF - Multiple Model Distributed Extended Kalman Filter
- MMDMPC - Multiple Model Distributed Model Predictive Controller
- M/NAV - Micro/Nano Air Vehicle
- MPC - Model Predictive Control
- PID - Proportional Integral Derivative
- PD - Proportional Derivative
- PTU - Pan-Tilt Unit
- RC - Remote Control
- RF - Radio Frequency
- RHC - Receding Horizon Control

- RMC - Royal Military College of Canada
- ROS - Robot Operating System
- RPY - Roll, Pitch and Yaw
- RSS - Received Signal Strength
- TRN - Terrain Relative Navigation
- UAV - Unmanned Aerial Vehicle
- VO - Visual Odometry
- VSMM - Variable Sturcture Multiple Model
- VTOL - Vertical Take Off and Landing

Chapter 1

Introduction

The world of Unmanned Aerial Vehicles (*UAV*s) is rapidly expanding with the presence of new technologies, software and cutting-edge research. Furthermore, the areas and domains in which UAVs are being deployed are broadening, spanning not only the military but also the civilian and consumer markets. UAVs were a $6 billion industry in 2016 with growth still projected in future years [1].

Current military applications can be divided into the military component in which they are employed [2]: in the navy, current uses of UAVs include fleet detection and shadowing, anti-submarine warfare, port protection and maritime surveillance, among many others; in the army, the primary applications are covert reconnaissance, target designation by laser and landmine detection; in the air force, there is target tracking, interception, anti-aircraft system's counter and airfield security. In the civilian and commercial areas, domains such as aerial photography, agriculture, search and rescue, fire services are just a few of the many industries in which UAVs are used [3]. As can be seen from these applications, most uses of UAVs involve dangerous environments where human lives are taken out of risk through their deployment (mostly military) or tasks that can be automated and are carried out by UAVs for efficiency reasons (commercial). These are two of the primary advantages UAVs offer and explain their increased presence over the past years [4].

Even more recent is the idea of using teams of individual UAVs to accomplish missions [5]. Militaries around the world are currently developing new swarming and cooperation strategies for a variety of applications and environments, as in the U.S. Army's plan to use teams of micro-drones in Intelligence, Surveillance, Target Acquisition and Reconnaissance (*ISTAR*) missions [6].

Of the aforementioned applications, of particular importance to the thesis is target tracking, the action of determining a target's, such as a person's or car's, position and following said target. Further investigation into the techniques and strategies currently employed for multi UAV target tracking leads to Kalman Filtering (*KF*) for state estimation and to Model Predictive Control (*MPC*) theory for control.

1.1 Thesis Statement

This thesis represents research done in a variety of fields. The main ones being robotics, state estimation and control. The thesis aims to show through the cooperative target tracking of a mobile ground target that using multiple models to represent target motion in a distributed way improves target estimation over conventional methods of estimation and distribution of information. Furthermore, the thesis aims to use this method of state estimation for the cooperative control of a team of UAVs, once again in a distributed manner, for tracking in formation flight with the goal of improving the UAVs collective response to an evasive target, while at the same time accounting for the possibility of occlusions.

1.2 Objectives

The main goal of the research aims to develop, implement and test novel algorithms in state estimation and control necessary for a team of UAVs to track an evasive ground target. As a result, three objectives can be made. The first objective is to configure a real time testing platform using UAVs for indoor and outdoor flights. The second objective is to implement a state estimation algorithm that combines information from different UAVs using multiple models to describe the target motion and is incorporated into a distributed extended Kalman filter. The final objective is to use the estimate and info generated from the state estimation algorithm for control. This control objective requires the UAVs to

follow the target while at the same time maintaining a formation and avoiding collisions with each other, through the use of a distributed model predictive controller.

1.3 Contributions

There are three main contributions presented here. The first pertains to the RMC lab and the field of robotics. It is the ground up implementation and configuration of the three AscTec Pelicans, and all the subsystems which make it a reliable testing ground for novel research algorithms.

The second contribution is to the field of state estimation, where a novel algorithm is presented in Chapter 4. The algorithm incorporates the EKF with the use of multiple models in a distributed network of UAVs, for the purpose of target tracking. It is shown through indoor tests that the algorithm improves the state estimation accuracy of the single model equivalents as well as other commonly used multi model approaches.

The third contribution is a novel multiple model MPC, where once again a distributed network of UAVs tracks a target. The use of multiple models for MPC in this context and for the tests done, shows that the algorithm does not significantly improve performance. However, more testing is needed to validate these results. This validation is carried out through both simulation and indoor flights.

1.4 Outline

The remainder of the thesis is divided as follows: relevant terminology, the area of research and general mathematical formulations are found in Chapter 2, the methods used are discussed in Chapter 3, the Multiple Model Distributed EKF (*MMDEKF*) is presented in Chapter 4, the Multiple Model Distributed MPC (*MMDMPC*) in Chapter 5, and a conclusion of the thesis is done in Chapter 6.

Chapter 2

Background

In this chapter, the general terminology and definitions employed in this research domain are identified and explained. In Section 2.1, the various types of UAVs will be explained. In Section 2.2, the different strategies and distributions used for cooperating UAVs are discussed. In Section 2.3, the general concepts of target tracking are expanded upon. In Section 2.4, the equations and theory behind the conventional EKF are presented. In Section 2.5, theory behind modelling and system identification are discussed. In Section 2.6, equations and theory of MPC are presented. Finally, in Section 2.7, the intended research area and scope of the thesis is specified.

2.1 UAV Classes

UAVs can be found in a variety of sizes and configurations. For the purpose of the thesis it is important to make the distinction between two main divisions of UAVs. That is, fixed wing UAVs and rotary wing UAVs. Depending on the UAV chosen, there are different flight characteristics available to the user. This thesis focuses on rotary wing UAVs, which are also known as Vertical Take Off and Landing (*VTOL*) UAVs [7]. By focusing on VTOL UAVs, it is assumed throughout the formulations presented here that the UAVs can hover and that the direction the UAV is facing can change without affecting its direction of travel. Therefore, although the algorithms presented here can be easily altered for the fixed-wing case, they should be considered for use with VTOL UAVs primarily.

2.2 Cooperative UAVs

The general idea behind cooperative UAVs is to have multiple UAVs share relevant information through a network in accomplishing one or more tasks [8]. For example, in a tracking scenario, UAVs can share estimate's of the target's position in order to track it more effectively. Furthermore, cooperative strategies incur less cost because many UAVs with inexpensive equipment that can be easily reproduced outperforms a single more expensive UAV with the best sensors [9]. Moreover, having multiple UAVs allows the mission to be completed regardless of if one of the UAVs malfunction or get taken out [10], due to the redundancies inherent in having multiple UAVs. Another advantage is improved efficiency in tasks where having multiple UAVs reduces the mission time, such as a search mission. Lastly, in any mission where there are multiple tasks that can be completed simultaneously, for example, tracking multiple targets, UAV cooperation becomes an asset [11].

The way information is distributed among the UAVs determines the type of cooperation they will have with each other. There are three main types of information distributions of UAVs: centralized, decentralized and distributed; as shown in Figure 2.1 [12]. Distributions are also commonly referred to in the literature as communication topologies. The first type of distribution is known as a centralized approach. This is where information flow is directed between one main UAV or a Ground Control Station (*GCS*) and the other UAVs. Consequently, the follower UAVs only obtain information about other followers through the leader or the GCS [13]. Advantages of this type of distribution are that it is easily manageable and is coherent. However these distributions are not really scalable or fault-tolerant. In the decentralized approach there is no leader or main hub through which information is passed through first. In fact, in completely decentralized network, UAVs do not share any information at all. A decentralized distribution is easily extensible and fault-tolerant. It can also be scaled better than a centralized approach, as it does not depend on the capacities of the leader [14]. For a distributed topology, as in Figure 2.1(c), agents can share information and measurements between one another. Other types of distribu-

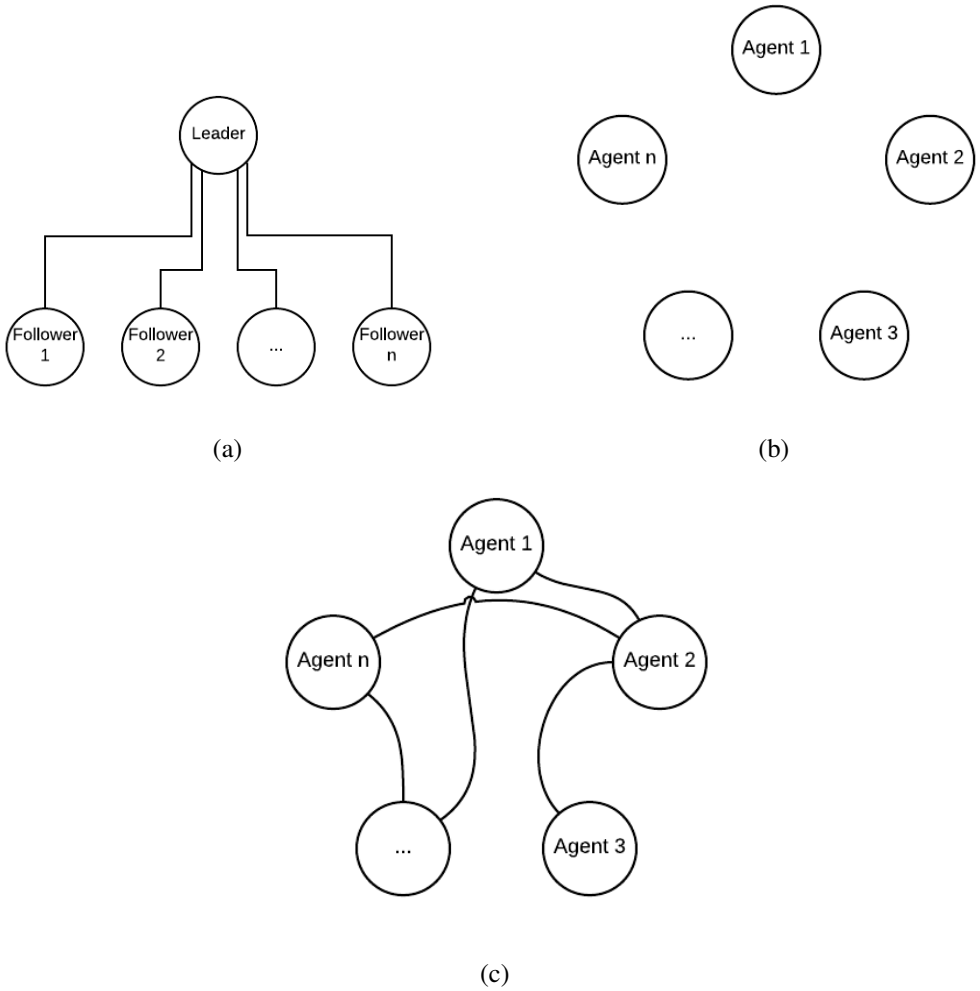


Figure 2.1: (a) A centralized topology (b) A decentralized topology (c) A distributed topology

tions, such as ring or hierarchical [15], they essentially represent different degrees of the distributions presented here.

In UAV cooperation there are also the notions of flocking, as seen in the real world from schools of fish or flocks of birds, formation flying and standoff tracking. A group of agents is said to be flocking when three rules are met [16]. The first is known as cohesion and refers to flockmates attempting to remain nearby other flockmates. The second is separation, which is essentially represented as collision avoidance. The last rule is alignment, where flockmates attempt to match velocity with nearby flockmates. Thus, the only restrictions on a flockmate's relative position to other flockmates is that it remains near the flock without

collision. This in turn implies that a flockmate's neighbors can continuously be changing. This is the main differentation with formation flight, where agents have a specific position in the formation and must maintain that position [17]. Standoff tracking is when the team of UAVs has certain constraints put on their movements. Common examples of this include ensuring that the camera is always fixed onto the target or ensuring that the UAVs maintain a specified distance from the target [18].

2.3 Target Tracking

Target tracking refers to the attempt of maintaining an estimate of the target's position at all times [19]. Target tracking can be divided by whether it is mobile target tracking or mobile *multi*-target tracking. Thus, the distinguishing feature between the two groups is the number of targets. As the thesis aims to put more emphasis on the cooperative state estimation cooperative control aspect of the UAV target tracking problem, it will focus on implementations using a single target, as implementations with multiple targets generally become a task assignment problem [20]. In other words, with multiple targets, the major problem addressed in research is deciding which UAV will track which target(s). In UAV target tracking, there is also distinctions made between tracking a ground target and an aerial target.

Successfully implementing cooperative UAV target tracking is dependent on the implementation of two sub-tasks. They are state estimation and flight coordination [21], both of which require some form of sensing (see Figure 2.2).

2.3.1 Sensing

The ability to track a target well relies heavily upon the sensor suite available on each UAV. The solution presented here makes use of a vision sensor (i.e. camera) oriented towards the target. Using a vision sensor necessitates some form of computer vision in

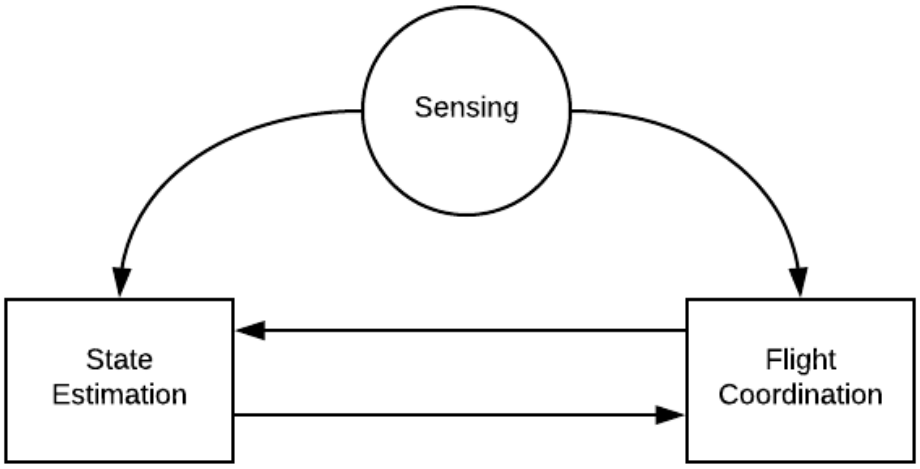


Figure 2.2: Dependencies among the three main sub-problems involved in cooperative UAV target tracking.

order to extract relevant information from video frames, such as identifying and positioning the target in each frame.

2.3.2 State Estimation

Information on both the target and UAV states, usually consisting of some combination of orientation and position, are obtained from sensing data. Final estimates of the states are then obtained from what is known as a sensor fusion algorithm. These algorithms usually consist of some form of vehicle modelling and subsequent integration with sensor data and filtering [22]. The state estimates are then sent to a flight controller or planner for use in flight coordination.

2.3.3 Flight Coordination

With state estimates, flight coordination attempts to determine a flight path that adheres to some predefined constraints. These might include coverage problems, where the UAVs could for example attempt to maintain the largest field of view collectively as possible. Or

they might include problems like the controls involved in ensuring that at least one UAV always has a line of sight with the target. In other words, the process of flight coordination involves controlling the UAVs, as a group or individually, in order to obtain a desired objective. Flight coordination implementations depend heavily on the type of cooperation adopted (distributed, decentralized and centralized).

2.4 Kalman Filtering

There are a variety of possible state estimation algorithms to choose from. For the typical Global Navigation Satellite System (*GNSS*) and Inertial Measurements Unit (*IMU*) fusion, the KF and its derivatives are a common example [23]. On the other hand, when cameras are involved, state estimation implementations other that a KF could include Image Based Navigation (*IBN*) [24], Visual Odometry (*VO*) [25], optic-flow based algorithms [26] or even Terrain Relative Navigation (*TRN*) [27]. Lastly, state estimation based off of range sensors, like LiDAR or ultrasonic are also possibilities [28].

The thesis focuses on Kalman Filter based approaches because for a linear system, the Kalman Filter is the optimal estimator [29], making it one of the more popular state estimation approaches. However, as many applications are not linear, two major derivatives of the Kalman Filter have been produced: the Extended Kalman Filter (*EKF*) [30] and the Unscented Kalman Filter (*UKF*) [31], where more information can be found in their respective references. Both these derivatives aim to approximate nonlinear models, but in different ways. The EKF uses linearization through the use of Jacobians whereas the UKF uses what is known as the Unscented Transform. There have been many studies comparing the performance of the two methods [32] [33] [34] [35]. Although the field of state estimation is still undecided as to which method is the best there are several advantages associated with each. For example, the EKF only provides a first order approximation of the linear system while the UKF provides a second order approximation. On the other

hand, the EKF is much older and has a lot of flavours present in the literature for robust implementations of specific applications, while the UKF does not. This is the main reason for using the EKF as the basis of the state estimation algorithm developed in this thesis.

2.4.1 The Extended Kalman Filter

The formulation is taken from [36]. It begins with the definition of a state space model as

$$\mathbf{x}(k) = f(\mathbf{x}(k-1)) + \mathbf{b}(k)\,, \tag{2.1}$$

$$\mathbf{z}(k) = h(\mathbf{x}(k)) + \mathbf{d}(k)\,, \tag{2.2}$$

where the function f predicts future states based on a model and the function h takes the states to the measurement space. Both $\mathbf{b}(k)$ and $\mathbf{d}(k)$ are Gaussian noise with covariance matrices Q and R, respectively. Furthermore, in this case, the initial state is determined from the measurement $\mathbf{z}(k)$ of the target as

$$\hat{\mathbf{x}}(0) = h^{-1}(\mathbf{z}(0)) \tag{2.3}$$

$$P(k) = P(0)\,. \tag{2.4}$$

Then following equations are carried out in sequence

$$\hat{\mathbf{x}}(k)^{-} = f(\hat{\mathbf{x}}^{+}(k-1)) \tag{2.5}$$

$$P^{-}(k) = F(k)P^{+}(k-1)F(k)^{T} + Q\,. \tag{2.6}$$

Equations (2.5) and (2.6) form what is know as the prediction or a priori step of the filter. Furthermore, $F(k)$ can be found by taking the Jacobian of $f(\cdot)$ as

$$F(k) = \frac{\partial f}{\partial \mathbf{x}}\Big|_{\hat{\mathbf{x}}(k-1)} . \tag{2.7}$$

Next is the measurement update step (also called the a posteriori step), after receiving a measurement $\mathbf{z}(k)$. The relevant equations are

$$K(k) = P^{-}(k)H(k)^{T}\left(H(k)P^{-}(k)H^{T}(k) + R(k)\right)^{-1} , \tag{2.8}$$

$$H(k) = \frac{\partial h}{\partial \mathbf{x}}\Big|_{\hat{\mathbf{x}}^{-}(k)} . \tag{2.9}$$

$$\hat{\mathbf{x}}^{+}(k) = \hat{\mathbf{x}}^{-}(k) + K(k)\left(\mathbf{z}(k) - h(\hat{\mathbf{x}}(k))\right) \tag{2.10}$$

$$P(k)^{+} = \left(I - K(k)H(k)\right)P(k)^{-} . \tag{2.11}$$

Equations (2.5)-(2.11) are repeated for each time step k.

2.5 Dynamic Modelling and System Identification

In the case of traditional MPC (more info in Section 2.6), how well the model represents the vehicle it models impacts the performance of the controller, as MPC uses this model to predict future control inputs needed [37]. There are several different methods that can be used to identify a model but three of the main ones are first principles techniques, experimental and system identification. This report will proceed to explain all three methods and subsequently go into further detail with the first principles and system identification

techniques.

In the first principles domain, the ranges of models vary from nonlinear fully coupled to nonlinear decoupled to linear decoupled and everything in between. Use of a model based on first principles is common. For example, references [38], [39], [40], have all used models derived solely from first principles techniques for control of VTOL vehicles.

The experimental method is a mix of first principles and the determination of physical parameters through experimentation (like drag coefficients for example). VTOL research that has used this type of modelling can be observed in [41], where a nonlinear model for a helicopter was experimentally determined. Another example can be found in [42], where a linear-parameter-varying model was developed for an unmanned helicopter.

The last method to consider is that of system identification, where the model of the vehicle is determined from a desired fit based on the input output relations of the vehicle. Methods for system identification in UAVs differ between time domain and frequency domain techniques. An example of the former is the Matlab System Identification Toolbox, which was used in [43], and approximates a linear model. In the latter, one of the more common tools for VTOL vehicles is the Comprehensive Identification from Frequency Responses *(CIFER)* tool developed by NASA's rotorcraft division [44] [45].

2.5.1 The UAV Model

2.5.1.1 Deriving From First Principles

There are many ways to achieve an accurate dynamic model of quadcopter dynamics. This report will attempt to derive a state space representation for the angular and linear accelerations of the quadrotor in terms of some inputs to be defined. The derivation of the quadcopter dynamics begins with the work presented in [46]. The commonly used quadrotor conventions are depicted in Figure 2.3.

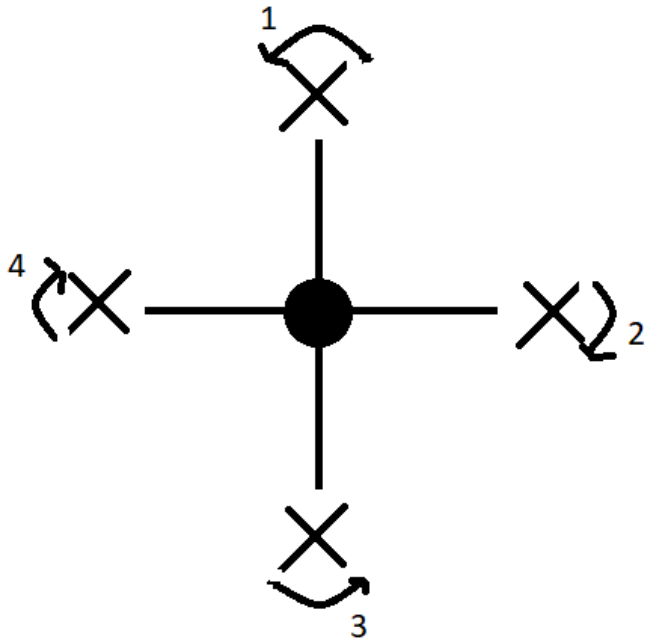


Figure 2.3: The direction of rotation of the individual motors of the UAV. Also the commonly used naming convention of said rotors.

In rigid body dynamics, the motion of the vehicle is defined by the sum of its forces $\mathbf{f}_{tot}$ and that of its torques $\boldsymbol{m}_{tot}$. For the sum of forces, the drag component is identified as $\boldsymbol{f_d}$, the thrust component $\boldsymbol{f_M}$ and the gravitational component $\boldsymbol{f_g}$. They are given by equations (2.12)-(2.14) respectively.

$$\boldsymbol{f_d} = -D_{d,F} \cdot C_b^w \cdot ||\boldsymbol{v}^b - {}^w\boldsymbol{v}^b||(\boldsymbol{v} - {}^w\boldsymbol{v}) \tag{2.12}$$

$$\boldsymbol{f_M} = \begin{bmatrix} 0 & 0 & -\Sigma_{i=1}^4 f_i \end{bmatrix}^T \tag{2.13}$$

$$\boldsymbol{f}_g = m \cdot C_n^b \cdot \begin{bmatrix} 0 & 0 & g \end{bmatrix}^T \tag{2.14}$$

where $\boldsymbol{v}$ and ${}^w\boldsymbol{v}$ are the linear velocities of the quadrotor and the wind respectively. f_i refers to the individual thrust forces of the quadrotor motors. Lastly, m and g are the mass of the quadrotor and the gravitational constant, respectively. $D_{d,F}$ is a matrix of drag coefficients for the force and C_b^w is the transformation matrix between the quadcopter base frame b and

some global frame w. From [47], this transformation matrix is given by:

$$C_b^w = \begin{bmatrix} c_\theta c_\psi & s_\phi s_\theta c_{psi} - c_\phi s_\psi & c_\phi s_\theta c_\psi + s_\phi s_\psi \\ c_\theta s_\psi & s_\phi s_\theta s_{psi} + c_\phi c_\psi & c_\phi s_\theta s_\psi - s_\phi c_\psi \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{bmatrix}^T \tag{2.15}$$

where $c_{(\cdot)}$ and $s_{(\cdot)}$ are the $\sin(\cdot)$ and $\cos(\cdot)$ functions. Furthermore angles θ, ϕ and ψ are the Roll, Pitch and Yaw (*RPY*) angles of the quadrotor, respectively. Next, using Newton's second law

$$\begin{aligned} \boldsymbol{f}_{tot} &= m \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \boldsymbol{f}_g + \boldsymbol{f}_m + \boldsymbol{f}_d \\ &= m \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -\Sigma_{i=1}^4 f_i \end{bmatrix} - D_{d,F} \cdot C_b^w \cdot ||\boldsymbol{v}^b - {}^w\boldsymbol{v}^b||(\boldsymbol{v} - {}^w\boldsymbol{v}) \\ \Rightarrow \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} &= \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} + C_b^w \cdot \begin{bmatrix} 0 \\ 0 \\ -\Sigma_{i=1}^4 f_i/m \end{bmatrix} - \frac{D_{d,F}}{m} \cdot C_b^w \cdot ||\boldsymbol{v}^b - {}^w\boldsymbol{v}^b||(\boldsymbol{v} - {}^w\boldsymbol{v}) \end{aligned} \tag{2.16}$$

where a_x, a_y and a_z represent the x, y and z accelerations of the quadrotor. Similarly, the torques can be further subdivided into the drag moment $\boldsymbol{m}_d$, the electromagnetic moment m_e, the mechanical moment m_m and the moment due to the thrust of the motors $\boldsymbol{m}_M$. They

are given by equations (2.17)-(2.20).

$$\boldsymbol{m}_d = -D_{d,M} \cdot ||\boldsymbol{\omega}^b||\boldsymbol{\omega}^b \tag{2.17}$$

$$m_e = \Psi i_A \tag{2.18}$$

$$m_m = k_T \cdot t_T \tag{2.19}$$

$$\boldsymbol{m}_M^b = \begin{bmatrix} (f_4 - f_2) \cdot l_M \\ (f_1 - f_3) \cdot l_M \\ -m_1 + m_2 - m_3 + m_4 \end{bmatrix} \tag{2.20}$$

where $D_{d,M}$ is the matrix of drag coefficients for torques, $\boldsymbol{w}$ is the angular velocities of the quadrotor, the flux is Ψ, the thrust of a motor is t_T, l_m is the distance between a rotor and the base frame, i_A is the anchor current and m_i is the moment of motor i. For the derivation in this thesis only the moments due to the thrust of the motors are considered because they represent the major components of the total moment acting on the vehicle. Also, we have from [47] that the overall moments can be given by

$$\boldsymbol{m}_{tot}^b = \begin{bmatrix} j_x & 0 & 0 \\ 0 & j_y & 0 \\ 0 & 0 & j_z \end{bmatrix} \begin{bmatrix} \omega_\theta^b \\ \omega_\phi^b \\ \omega_\psi^b \end{bmatrix} + \begin{bmatrix} \theta^b \\ \phi^b \\ \psi^b \end{bmatrix} \times \begin{bmatrix} j_x & 0 & 0 \\ 0 & j_y & 0 \\ 0 & 0 & j_z \end{bmatrix} \begin{bmatrix} \theta^b \\ \phi^b \\ \psi^b \end{bmatrix} \tag{2.21}$$

where $\boldsymbol{m}_{tot}^b$ is the moments in the UAV frame or reference. $j_{(\cdot)}$ are the moments of inertia in their respective directions. Similarly, θ^b, ϕ^b and ψ^b are the euler angles in the robot frame. Neglecting drag equation (2.17) and equating equation (2.20) and (2.21) yields

$$\begin{bmatrix} (f_4 - f_2) \cdot l_m \\ (f_1 - f_3) \cdot l_m \\ -m_1 + m_2 - m_3 + m_4 \end{bmatrix} = \begin{bmatrix} j_x & 0 & 0 \\ 0 & j_y & 0 \\ 0 & 0 & j_z \end{bmatrix} \begin{bmatrix} \omega_\theta^b \\ \omega_\phi^b \\ \omega_\phi^b \end{bmatrix} + \begin{bmatrix} \theta^b \\ \phi^b \\ \psi^b \end{bmatrix} \times \begin{bmatrix} j_x & 0 & 0 \\ 0 & j_y & 0 \\ 0 & 0 & j_z \end{bmatrix} \begin{bmatrix} \theta^b \\ \phi^b \\ \psi^b \end{bmatrix}. \tag{2.22}$$

Rearranging,

$$\begin{bmatrix} \omega_\theta^b \\ \omega_\phi^b \\ \omega_\psi^b \end{bmatrix} = \begin{bmatrix} (j_y - j_z)\phi^b\psi^b/j_x \\ (j_z - j_x)\theta^b\psi^b/j_y \\ (j_x - j_y)\theta^b\phi/j_z \end{bmatrix} + \begin{bmatrix} (f_4 - f_2) \cdot l_m/j_x \\ (f_1 - f_3) \cdot l_m/j_y \\ (-m_1 + m_2 - m_3 + m_4)/j_z \end{bmatrix}. \tag{2.23}$$

For the next step, the small angle approximation is used, which holds true in a nominal range around hovering and which is suitable for most quadrotor flights. With this approximation, the angular accelerations in the body and inertial frames can be equated to finalize our model

$$\begin{bmatrix} \alpha_\phi \\ \alpha_\theta \\ \alpha_\psi \end{bmatrix} \approx \begin{bmatrix} (j_y - j_z)\omega_\theta\omega_\psi/j_x \\ (j_z - j_x)\omega_\phi\omega_\psi/j_y \\ (j_x - j_y)\omega_\phi\omega_\theta/j_z \end{bmatrix} + \begin{bmatrix} (f_4 - f_2) \cdot l_m/j_x \\ (f_1 - f_3) \cdot l_m/j_y \\ (-m_1 + m_2 - m_3 + m_4)/j_z \end{bmatrix} \tag{2.24}$$

Next the inputs to the quadrotor are defined. Traditionally, u_ϕ is defined as the pitch force resulting from $f_4 - f_2$, u_θ as the roll force $f_1 - f_3$, u_ψ is the yawing moment $-m_1 + m_2 - m_3 + m_4$. The final input is the sum of the upwards thrust produced by the quadrotor, u_f. Substituting these new inputs into equations (2.16) and (2.24), the final first principles model is obtained

$$\begin{aligned} \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} &= \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} + C_b^w \cdot \begin{bmatrix} 0 \\ 0 \\ u_f/m \end{bmatrix} \\ &\quad - \frac{D_{d,F}}{m} \cdot \boldsymbol{C}_b^w \cdot |\boldsymbol{v}^b - {}^w\boldsymbol{v}^b|(\boldsymbol{v} - {}^w\boldsymbol{v}) \end{aligned} \tag{2.25}$$

and for the angular accelerations

$$\begin{bmatrix} \alpha_\phi \\ \alpha_\theta \\ \alpha_\psi \end{bmatrix} \approx \begin{bmatrix} (j_y - j_z)\omega_\theta\omega_\psi/j_x \\ (j_z - j_x)\omega_\phi\omega_\psi/j_y \\ (j_x - j_y)\omega_\phi\omega_\theta/j_z \end{bmatrix} + \begin{bmatrix} u_\phi \cdot l_m/j_x \\ u_\theta \cdot l_m/j_y \\ u_\psi/j_z \end{bmatrix} . \tag{2.26}$$

2.5.1.2 Using System Identification

In reality, identifying inertial and mass parameters is a difficult process and dependant on the quadrotor being used. For that reason, researchers also use system identification to develop a model of the quadrotor. For proper identification, one must first have the quadrotor fly over a range of inputs so that the identified model can accurately identify parameters over the desired flight ranges. Thus, for a position controller, having the UAV fly according to step inputs in x and y would suffice. For this flight, the input at time k would be recorded, denoted $\boldsymbol{u}(k)$, and similarly the measured outputs, denoted $\boldsymbol{z}(k)$. Additionally, the states of the system are written as $\boldsymbol{x}(k)$. This derivation is done only in the x coordinate, but a similar procedure is used to identify y coordinate, assuming there is no correlation between the two. Thus, a state-space representation exists as follows

$$\boldsymbol{x}(k) = A\boldsymbol{x}(k-1) + B\boldsymbol{u}(k-1) \tag{2.27}$$

$$\boldsymbol{z}(k) = H\boldsymbol{x}(k) \tag{2.28}$$

where A and B are the dynamic matrix and input matrix, respectively. H is the measurement matrix. The input vector $\boldsymbol{u}(k)$ is defined as

$$\boldsymbol{u}(k) = \begin{bmatrix} u_x & u_y \end{bmatrix}^T \tag{2.29}$$

where x and y are the positions the UAV should fly to, in the global frame, as determined from an MPC controller in this case. For position control in x, two states are identified: the

x position, and its speed v_x. Therefore, equation (2.27) can be expanded as

$$\begin{bmatrix} x \\ v_x \end{bmatrix}_k = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x \\ v_x \end{bmatrix}_{k-1} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} u_x(k-1)\,. \tag{2.30}$$

In this case, a_{ij} and b_i represent the i^{th} and j^{th} entries of the matrices A and B, respectively. There exists different ways to identify these parameters, but this thesis explains the method of linear least squares [48] to find the best fits for the matrices A and B. The algorithm functions by essentially minimizing the error between the model and the outputs. That is, given a matrix of states and inputs, denoted G, a vector of the parameters to be identified $\boldsymbol{w}$, and a vector of outputs to the system in question, $\boldsymbol{l}$, the algorithm attempts to find the optimal $\boldsymbol{w}$ for the optimization problem

$$\arg\min_{\boldsymbol{w}} \frac{1}{2}||G\boldsymbol{w} - \boldsymbol{l}||_2^2\,. \tag{2.31}$$

For the identification procedure, the UAV flies at step input in x, that is for example from 0 m to 1 m in the global frame. Also, assume that n input-output samples are gathered for the flight. Next, for simplicity the identification is split into two sub-problems. More specifically, consider solely the first row of equation (2.30), that is the parameters a_{11}, a_{12} and b_1. A symmetrical problem can be solved for the second row of equation (2.30). With n samples, starting at sample $n = 1$, the output x positions are obtained by simply truncating the first sample

$$\boldsymbol{l} = \begin{bmatrix} z(2) & z(3) & z(4) & \dots & z(n-1) & z(n) \end{bmatrix}^T_{1\times(n-1)}\,. \tag{2.32}$$

Similarly, the matrix G is obtained by truncation of the n^{th} sample

$$G = \begin{bmatrix} x(1) & v_x(1) & u_x(1) \\ x(2) & v_x(2) & u_x(2) \\ x(3) & v_x(3) & u_x(3) \\ \vdots & \vdots & \vdots \\ x(n-2) & v_x(n-2) & u_x(n-2) \\ x(n-1) & v_x(n-1) & u_x(n-1) \end{bmatrix}_{(n-1)\times 3} . \tag{2.33}$$

With these definitions, a vector of unknowns for the first row of the problem is given by

$$\boldsymbol{w}_1 = \begin{bmatrix} a_{11} & a_{12} & b_1 \end{bmatrix}^T_{1\times 3} , \tag{2.34}$$

yielding the final equation

$$\boldsymbol{l}_{(n-1)\times 1} = E_{(n-1)\times 3}\boldsymbol{w}_{3\times 1} . \tag{2.35}$$

The process is then repeated for the second vector of unknowns,

$$\boldsymbol{w}_2 = \begin{bmatrix} a_{21} & a_{22} & b_2 \end{bmatrix}^T_{1\times 3} . \tag{2.36}$$

2.5.2 Target Model

For the target model, only first principles methods are discussed in the thesis, as using multiple target models to match the motion of a single target is one of the major premises of this work. Note however that if there is a priori knowledge of target, as in what type of vehicle it is, then it is possible to use similar system identification methods described in Section 2.5.1.2. In this thesis, there are three models that will be used consistently because they appear most often in target tracking literature. They are the Constant Velocity

(*CV*) [49], Constant Acceleration (*CA*) [50] and Coordinated Turn (*CT*) [51] models. The constant velocity model of a vehicle has the following discrete state space:

$$
\begin{aligned}
{}^{CV}_{g}\mathbf{x}(k) &= \begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix}_k = \begin{bmatrix} x + v_x \cdot T_s \\ y + v_y \cdot T_s \\ v_x \\ v_y \end{bmatrix}_{k-1} \\
\Rightarrow \begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix}_k &= \begin{bmatrix} 1 & 0 & T_s & 0 \\ 0 & 1 & 0 & T_s \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}_{k-1} \begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix}_{k-1} \\
\Rightarrow {}^{CV}_{g}\mathbf{x}_k &= {}^{CV}_{g}F(k-1)\,{}^{CV}_{g}\mathbf{x}(k-1) \\
\Rightarrow {}^{CV}_{g}\mathbf{x}(k) &\triangleq {}^{CV}_{g}f({}^{CV}_{g}\mathbf{x}(k-1))\,,
\end{aligned} \tag{2.37}
$$

where T_s represents the sample time, x and y represent position and $v_{(\cdot)}$ is the speed in the appropriate direction. The subscript g refers to the target.

The second UAV will adopt the CA model, which with similar manipulations to equation (2.37), is given by

$$
\begin{aligned}
{}^{CA}_{g}\mathbf{x}(k) &= {}^{CA}_{g}F(k-1)\,{}^{CA}_{g}\mathbf{x}(k-1) \\
\Rightarrow {}^{CA}_{g}\mathbf{x}(k) &\triangleq {}^{CA}_{g}f({}^{CA}_{g}\mathbf{x}(k-1))\,,
\end{aligned} \tag{2.38}
$$

with

$$ {}^{CA}_{g}\mathbf{x} = \begin{bmatrix} x \\ y \\ v_x \\ v_y \\ a_x \\ a_y \end{bmatrix} \qquad {}^{CA}_{g}F(k-1) = \begin{bmatrix} 1 & 0 & T_s & 0 & T_s^2/2 & 0 \\ 0 & 1 & 0 & T_s & 0 & T_s^2/2 \\ 0 & 0 & 1 & 0 & T_s & 0 \\ 0 & 0 & 0 & 1 & 0 & T_s \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, $$

where $a_{(\cdot)}$ is the vehicle's acceleration in the respective direction.

Lastly, the CT model will be used by the third UAV. Namely, with a turn rate ω, the model is described as follows

$$ \begin{aligned} {}^{CT}_{g}\mathbf{x}(k) &= {}^{CT}_{g}F(w, k-1)\,{}^{CT}_{g}\mathbf{x}(k-1) \\ \Rightarrow {}^{CT}_{g}\mathbf{x}(k) &\triangleq {}^{CT}_{g}f({}^{CT}_{g}\mathbf{x}(k-1)), \end{aligned} \tag{2.39} $$

with

$$ {}^{CT}_{g}\mathbf{x} = \begin{bmatrix} x & y & v_x & v_y & w \end{bmatrix}^{\mathsf{T}} $$

$$ {}^{CT}_{g}F(w, k-1) = \begin{bmatrix} 1 & \frac{\sin\omega T_s}{\omega} & 0 & -\frac{1-\cos\omega T_s}{\omega} & 0 \\ 0 & \cos\omega T_s & 0 & -\sin\omega T_s & 0 \\ 0 & \frac{1-\cos\omega T_s}{\omega} & 1 & \frac{\sin\omega T_s}{\omega} & 0 \\ 0 & \sin\omega T_s & 0 & \cos\omega T_s & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{k-1}, $$

where the value for ω used is that at time step $k-1$. Note that in this case the prediction matrix is nonlinear.

2.6 Model Predictive Control

MPC is has become a popular choice as a practial control scheme [52]. In the general discussion of MPC, a dynamic model is used to predict the future behaviour of the target according to its current state. These states and inputs are propagated through the model over a certain number of time steps, usually called the prediction horizon. Having the desired end-states of the vehicle and the aforementioned predicted states allows for the optimization of a trajectory to have the UAVs follow. Thus, the MPC problem is also commonly viewed as an optimization problem where the inputs needed to reach the desired end-states are minimizers over the entire horizon. Furthermore, by formulating the control as an optimization problem, the solution can be constrained to achieve certain performance standards. A common example could be to constrain the distance of a UAV to the target, known as standoff tracking. There are some drawbacks to MPC however, which relate to the computation time required for an optimal input and the need for a model. One major problem is the nature of the objective (or cost) function being minimized. If the objective function is not convex, it may take longer for the optimization to achieve a globally optimal estimate and in some cases, depending on the type of optimization algorithm used and the nature of the objective function, only local optimality might be guaranteed. On the contrary, if the objective function is convex, it can be guaranteed that there is indeed a single global maximum or minimum [53]. Furthermore, the traditional MPC controller needs a model of the vehicle motion in order to accurately predict the impact of current control decisions. Thus, the control relies heavily on the accuracy of the model and if not identified properly can worsen the controller performance. It should be noted that recent advances in MPC have led to a derivative known as robust MPC, where model and plant mismatch are dealt with through the consideration of process and model uncertainties, resulting in a more stable response [54]. However due to its simplicity of implementation and the amount of previous research done on traditional MPC, its formulation is presented here.

2.6.1 Problem Formulation

The typical optimization based approach consists of N UAVs tracking a target over time. Furthermore, the target has a dynamic model, typically used in the optimization of a cost function, which is denoted here as $\mathfrak{J}$ and which uses the current states of the target and UAVs as well as the inputs to the UAVs to optimize the future inputs of the UAVs over a specified horizon, denoted h. Thus,

$$\mathbf{u}^{\star}(k) = \arg\min_{\mathbf{u}} \mathfrak{J} \tag{2.40}$$

where

$$\mathbf{u} \triangleq \begin{bmatrix} \mathbf{u}(k) & \mathbf{u}(k+1) & \cdots & \mathbf{u}(k+h) \end{bmatrix}^T \tag{2.41}$$

contains the inputs over the horizon required for the UAVs, beginning from time step k

2.6.2 Cost Function

In reality, there are a variety of cost functions currently employed in the literature, depending on the criteria being optimized, as will be seen in Section 5.2. In this section however, a sample MPC implementation for a target tracking UAV is expanded into much greater detail. Although the controller is identified for a single UAV here, it can easily by scaled for the N UAVs in a distributed network. This section has the sole intent of informing the reader of the basics of MPC.

First, assume that the UAV is trying to maintain its position over the target at a constant altitude. This simplifying assumption allows the MPC controller to use the current estimate of the target's xy position as the UAVs desired xy position. Next, a linear model is adopted for the target dynamics, which determines the reference trajectory, as in the following state

space representation

$$ {}_g\mathbf{x}(k) = {}_gA\mathbf{x}(k-1)\,, \tag{2.42} $$

where recall that the subscript g refers to the ground target, $\mathbf{x}(k)$ is state vector of the target, and can vary depending on which model has been adopted (i.e. CV, CA or CT), all at time step k, respectively. Then, given an initial measurement (from an EKF for example) of the ground target, ${}_g\hat{\mathbf{x}}^+(0)$, with a horizon over which to optimize, h, there exists the following relationship

$$ {}_g\S = {}_g\mathcal{A}_g\hat{\mathbf{x}}^+(0) \tag{2.43} $$

where

$$ {}_g\S \triangleq \begin{bmatrix} {}_g\mathbf{x}(1) & {}_g\mathbf{x}(2) & \cdots & {}_g\mathbf{x}(h) \end{bmatrix}^T_{1\times(o\cdot h)} \qquad {}_g\mathcal{A} \triangleq \begin{bmatrix} {}_gA & {}_gA^2 & \dots & {}_gA^h \end{bmatrix}^T_{o\times(o\cdot h)}\,, $$

where o is the number of states in the target model. Thus with equation (2.43), the state of the target and consequently the desired position of the UAV can be predicted at each iteration. For the UAV, similarly a linear model is adopted, as in the one that was adopted through system identification (see Section 2.5.1.2)

$$ \boldsymbol{x}(k) = A\boldsymbol{x}(k-1) + B\boldsymbol{u}(k-1) \tag{2.44} $$

where a similar relationship exists

$$ \S = \mathcal{A}\hat{\mathbf{x}}(0) + \mathcal{B}\sqcap \tag{2.45} $$

where

$$\mathcal{A} = \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^h \end{bmatrix}_{(p\cdot h)\times p} \qquad \mathcal{B} = \begin{bmatrix} B & 0 & \cdots & 0 \\ AB & B & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A^{h-1}B & A^{h-2}B & \cdots & B \end{bmatrix}_{(s\cdot h)\times(s\cdot h)}$$

$$\S = \begin{bmatrix} \mathbf{x}(1) & \mathbf{x}(2) & \cdots & \mathbf{x}(h) \end{bmatrix}^T_{1\times(p\cdot h)} \qquad \sqcap \triangleq \begin{bmatrix} \mathbf{u}(0) & \mathbf{u}(1) & \cdots & \mathbf{u}(h) \end{bmatrix}^T_{1\times(s\cdot h)}$$

where p and s represent the number of states and inputs respectively for the UAV model. With the UAV states and target states over the entire horizon, the cost function to be minimized can be defined accordingly. Defining the reference sequence for the UAV to follow as $\mathbf{x}_r(k) \triangleq {}_g\mathbf{x}(k)$. The cost function at time step k over horizon h is defined as [55]

$$\begin{aligned} \mathfrak{J}(\mathbf{x}(k),\, \mathbf{u}(k),\, {}_g\mathbf{x}(k)) &= (\S - {}_g\S)^T \mathcal{Q} (\S - {}^g\S) + \sqcap^T \mathcal{R} \sqcap \\ &= \begin{bmatrix} \mathbf{x}(k) - {}_g\mathbf{x}(k) \\ \mathbf{x}(k+1) - {}_g\mathbf{x}(k+1) \\ \vdots \\ \mathbf{x}(k+h) - {}_g\mathbf{x}(k+h) \end{bmatrix}^T \begin{bmatrix} Q(1) & 0 & 0 & 0 \\ 0 & Q(2) & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & Q(h) \end{bmatrix} \begin{bmatrix} \mathbf{x}(k) - {}_g\mathbf{x}(k+1) \\ \mathbf{x}(k+2) - {}_g\mathbf{x}(k+2) \\ \vdots \\ \mathbf{x}(k+h) - {}_g\mathbf{x}(k+h) \end{bmatrix} \\ &+ \begin{bmatrix} \mathbf{u}(k) \\ \mathbf{u}(k+2) \\ \vdots \\ \mathbf{u}(k+h) \end{bmatrix}^T \begin{bmatrix} R(1) & 0 & 0 & 0 \\ 0 & R(2) & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & R(h) \end{bmatrix} \begin{bmatrix} \mathbf{u}(k) \\ \mathbf{u}(k+1) \\ \vdots \\ \mathbf{u}(k+h) \end{bmatrix} \end{aligned} \tag{2.46}$$

where $\mathbf{u}(k)$ is the inputs to the UAV at time step k. The weighting matrices $\mathcal{Q}$ and $\mathcal{R}$

associate a cost with certain time steps and parameters as follows

$$\mathcal{Q}_{(p\cdot h)\times(p\cdot h)} = \text{diag}\begin{bmatrix} Q(1) & Q(2) & \cdots & Q(h) \end{bmatrix}$$
$$\mathcal{R}_{(s\cdot h)\times(s\cdot h)} = \text{diag}\begin{bmatrix} R(1) & R(2) & \cdots & R(h) \end{bmatrix}$$

where

$$Q(k)_{p\times p} = \text{diag}\begin{bmatrix} q_1 & q_2 & \cdots & q_p \end{bmatrix}$$
$$R(k)_{m\times m} = \text{diag}\begin{bmatrix} r_1 & r_2 & \cdots & r_s \end{bmatrix}$$

for a UAV with s inputs to optimize and p states. The diagonal elements of these matrices are positive real constants. In most cases however, and as will be the case of this example, the individual weight matrices are usually equal to one another, that is

$$Q(1) = Q(2) = \cdots = Q(h) \triangleq Q\,,$$
$$R(1) = R(2) = \cdots = R(h) \triangleq R\,,\,.$$

Finally, using the above simplification, the final form of the cost function can be written as

$$\begin{aligned}\mathfrak{J}(\mathbf{x}(k),\,\mathbf{u}(k),\,{}_g\mathbf{x}(k)) &= \sum_{l=k}^{k+h}[(\mathbf{x}(l)-\mathbf{x}_r(l))^T Q(\mathbf{x}(l)-\mathbf{x}_r(l)) + \mathbf{u}(l)^T R\mathbf{u}(l)] \\ &= \sum_{l=k}^{k+h}[(\mathbf{x}(l)-{}_g\mathbf{x}(l))^T Q(\mathbf{x}(l)-{}_g\mathbf{x}(l)) + \mathbf{u}(l)^T R\mathbf{u}(l)]\,. \qquad (2.47)\end{aligned}$$

To form the optimization problem with this cost function, note that the optimizer is the inputs to the system, $\mathbf{u}$. We would like to minimize these inputs over the horizon according

to the states $\mathbf{x}$. Thus the problem may be posed as

$$\begin{aligned}\mathbf{u}^{\star} = &\arg\min_{\mathbf{u}} \mathfrak{J}(\mathbf{x}(k),\, \mathbf{u}(k),\, {}_{g}\mathbf{x}(k)) \\ &\text{subject to} \quad \boldsymbol{x}(k) = A\boldsymbol{x}(k-1) + B\boldsymbol{u}(k-1) \\ &\boldsymbol{u}_{min} \leq \mathbf{u} \leq \mathbf{u}_{max} \\ &\boldsymbol{x}(0) = \hat{\boldsymbol{x}}^{+}(0)\end{aligned} \tag{2.48}$$

where u_{min} and u_{max} form the range of allowable inputs [47]. Typically, at each time step, the first optimized input, that is $\mathbf{u}_t^{\star} \in \mathbf{u}^{\star}$ is used as the control input and the optimization process begins once more.

The above formulation is for a single UAV tracking for a target. This formulation is expanded to a Distributed MPC (*DMPC*) for the multi-UAV case in Chapter 5, where now a team of UAVs capable of sharing information track a mobile ground target.

2.7 Intended Research Area

Now that that an overview of the major theoretical areas of the thesis have been specified, this section will narrow the scope of the project accordingly. The general background concepts introduced in this Chapter are listed and the chosen approach is identified in Table 2.1.

Thus, the general area of research can be expressed precisely as: *Distributed EKF estimation and MPC of VTOL UAVs for formation tracking of a mobile ground target through use of both system identification and first principles techniques.*

Although the targeted area has been defined above, based on the resources available there are several more specific constraints and simplifications that were imposed on the developed solutions, for the intention of narrowing the scope of the thesis. Many of these constraints involve the type and the number of UAVs employed. Other constraints may include geometric constraints imposed by sensor placement, such as the camera location

Table 2.1: Problem Summary and Specifications

Sup-problem	Approaches	Chosen
UAV	VTOL, Fixed-wing	VTOL
Distribution	Centralized, Decentralized, Distributed	Distributed
Cooperation	Flocking, Formation, Standoff Tracking	Formation
Target Type	Ground, Air vehicle	Ground Vehicle
Target Number	Single, Multiple	Single
Estimation	KF, EKF, UKF, VO, etc...	EKF
UAV Model	First Principles, System Identification	System Identification
Target Model	First Principles, System Identification	First Principles
Control	MPC, Game Theory, etc...	MPC

and orientation and whether or not these aspects can be varied [56]. Computation power constrains the complexity of algorithms capable of being performed on-line [57]. The airspeeds at which the vehicle is allowed to fly at could be a constraint as well [58]. Constraints imposed by the target should also be considered and is why modelling target motion is a field of research in itself [59]. For example, whether or not the target is evasive can have impacts on the type of solution chosen. These constraints are all addressed in the final solutions.

Chapter 3

Methodologies

This chapter details the specifics of the final implementation details used in this work. Note that as is detailed in Chapter 1, one of the overall aims of the thesis is to provide a comprehensive outdoor multi-UAV testing environment. Although this objective was not achieved, as no outdoor flights were conducted, there were several steps taken in order to progress towards the objective, which are detailed here.

3.1 Implementation

3.1.1 VTOL UAV Choice

A standard VTOL UAV used in the literature is the AscTec Pelican. The reason being that it comes ready to use with the Robot Operating System (*ROS*), a commonly used middleware for robotics. Furthermore, both the hardware and the software can be easily customized to meet the needs of the thesis. Three of these quadrotors were used in both simulation and live experiments.

3.1.2 Software

As the thesis aims to develop primarily a new state estimation algorithm and a new control law for UAV cooperation, there were several other software requirements necessary in order to achieve a fully functioning experimental test bed.

3.1.2.1 UAV Attitude Control

The AscTec Pelican comes pre-installed with a Proportional Derivative (*PD*) attitude controller on the Low Level Processor (*LLP*). This will be the lowest level of control present on the final implementation.

3.1.2.2 UAV Position Control

For individual UAV position control, existing software known as the AscTec Mav Framework will be used. The framework includes a high level position controller which uses input-output linearization accompanied with separate Proportional Integral Derivative (*PID*) control for the longitudinal, lateral and vertical axes of the UAV [60].

3.1.2.3 UAV State Estimation

A 25 state Extended Kalman Filter will be used for on board state estimation. The filter is compatible with the position controller. More information can be found in [61]. The filter performs fusion of the Pelican IMU data with various types of sensors, depending on the application. For example, this thesis performs Optitrack/IMU fusion for indoor flight and GNSS/IMU fusion for outdoor flight.

3.1.2.4 Computer Vision

The aspect requiring computer vision was the identification and interpretation of target camera data. As this thesis does not focus on computer vision, generic algorithms and libraries were used. The library is known as Aruco ROS, and it functions by identifying an Aruco marker of a known type and dimension. From this information it returns the relative position of the marker in the camera frame [62].

3.1.2.5 Target State Estimation

This is one of the two main contributions of the thesis. The target state estimation will be carried out through a multi model DEKF, as detailed in Chapter 4.

3.1.2.6 UAV Trajectory Generation

The UAV trajectories are the second major contribution of the thesis. Through the use of a multi model DMPC, separate UAV trajectories are computed in real time, as detailed in Chapter 5.

3.1.2.7 Ground Control Software

For indoor flights, no particular software was used. In outdoor flights, a readily available software for managing outdoor flights is QGroundControl. However the outdoor implementation never made as far as the implementation of this software.

3.1.2.8 Communication

For distributed estimation and control, a key concept is communication, both how information is communicated and what information should be communicated. Here, only the former is discussed. For indoor flights a wireless network, accompanied with the multi master framework were used. In outdoor flights, each pelican and the ground station were given XBEE modules. The appropriate wrappers for the modules were coded and tested [63]. Note that for the final implementations of the algorithms presented here, each iteration resulted in a UAV sending out on the order of 50 bytes of information.

3.1.3 Hardware

Having experiments implies a need for several hardware components, listed below.

3.1.3.1 Target Choice

For indoor experiments, the Turtlebot was programmed to follow several different paths. For outdoor experimetns, a Remote Controlled (*RC*) Clearpath Husky was prepared. Both robots are compatible with ROS.

3.1.3.2 Sensing

For the Pelican, there are a variety of on-board sensors available for use and integrated into the previously mentioned software packages. More specifically, the Pelican is outfitted with an IMU, a camera and barometer for indoor flights. For outdoor flights, GPS was integrated.

3.1.3.3 Optitrack

Since indoor flights represent a GNSS-denied environment, in order to replicate outdoor flights a positioning system known as Optitrack was used, providing accurate pose data of the Pelican in real-time. The Optitrack system consists of 24 infrared cameras that detect markers attached to the UAVs and target. Through calibration of the system and different marker placement arrangements, each UAV's position and orientation is streamed over a wifi network. The system is able to keep track of the different UAVs based on a marker arrangement of 4-6 markers per UAV, which is defined by the user and differs from UAV to UAV. From the marker arrangement the Optitrack system returns the center of gravity of the UAV, to an accuracy on the order of millimeters [64].

3.1.3.4 Ground Control Station

For indoor flights, a GCS has been added in the form of a laptop with SSH capabilities through a wireless network to which the Pelican automatically connects to. For outdoor flights, a similar laptop was outfitted with an XBEE module for serial communication.

3.2 Validation

The resultant algorithms, written in a combination of C, C++, Python and XML underwent various levels of validation, be it simulation or indoor experiments, with the final intention of an outdoor experiment.

3.2.1 Simulation

The simulation environment used is known as VREP [65], which is compatible with both MATLAB and ROS. Together they form a comprehensive real-time simulation environment.

3.2.2 Experimental Setups

This subsection overviews the different experimental setups used to validate the different algorithms present in this work.

3.2.2.1 Indoor Setup

Indoor flights held at Royal Military College of Canada's (*RMC*) robotics lab. An overview of the different systems involved are given in Figure 3.1 and the actual implementation is shown in Figure 3.2.

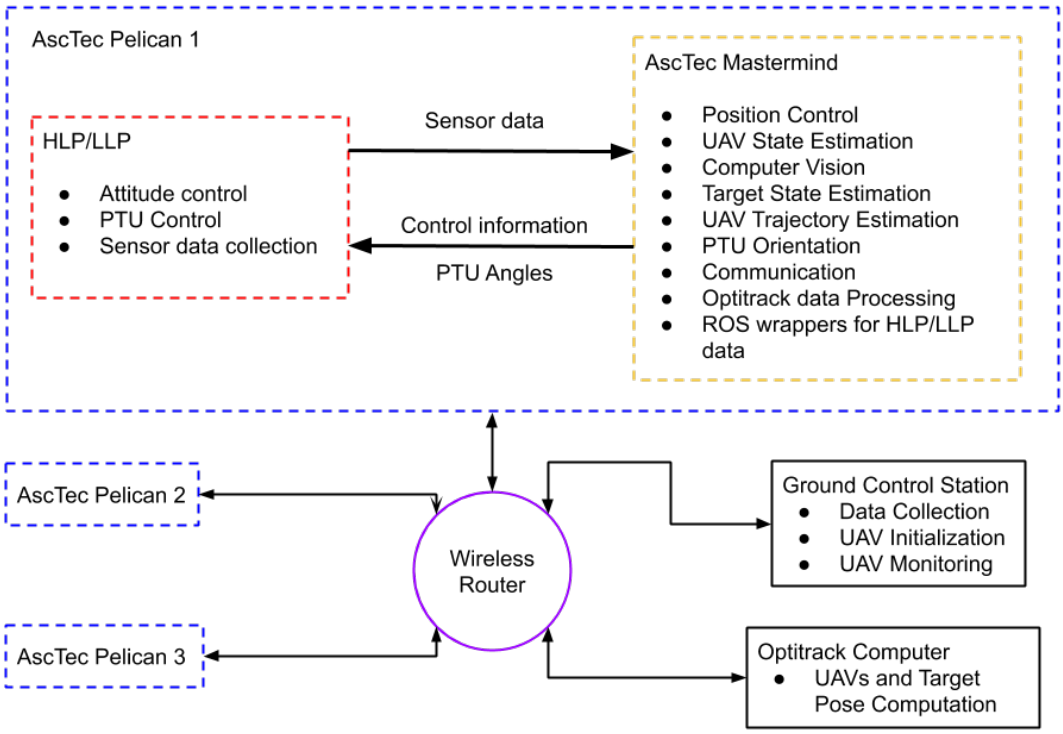


Figure 3.1: Hardware and software implementations for the indoor experimental setup. Each Pelican has two main locations where computation is done, the HLP/LLP unit and the AscTec Mastermind. There is also a ground control station which monitors the UAVs through SSH protocols. The optitrack computer and the inter UAV communications use a multicast interface.

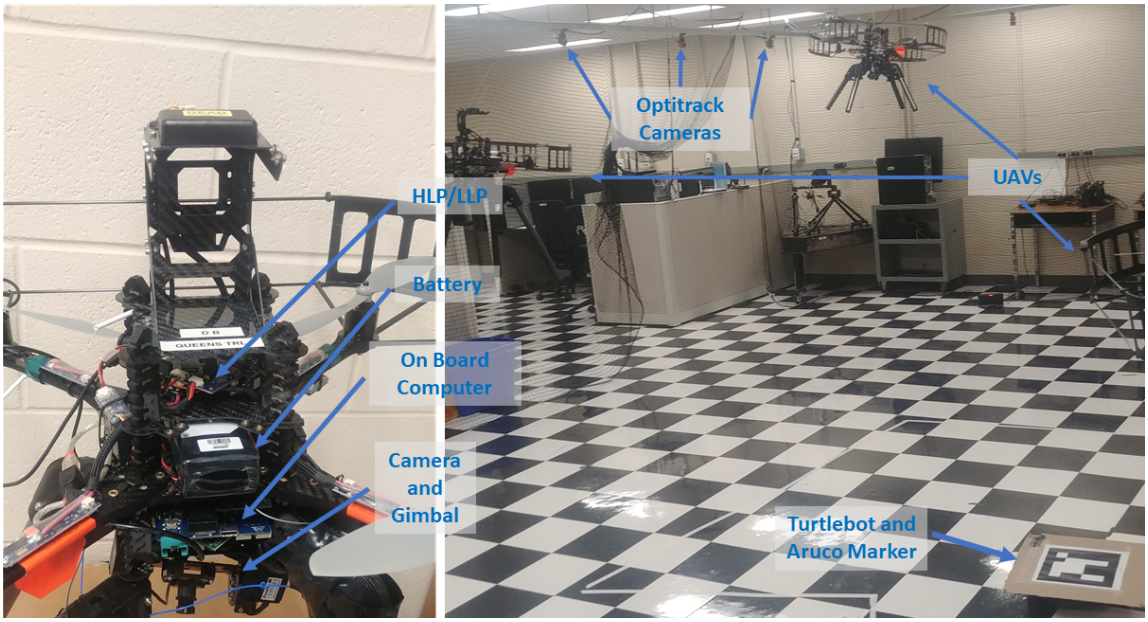


Figure 3.2: On the left, the main hardware components of the AscTec Pelican. The on board computer (AscTec Mastermind) runs the ROS framework and algorithms (Computer Vision, State Estimation, Transform Tree, Multi-Master Framework, Position Control), whereas the High Level and Low Level Processors (*HLP* and *LLP* respectively) handle the attitude control and gimbal orientation.

3.2.2.2 State Estimation Experimental Setup

In this particular problem, a team of 3 VTOL UAVs was considered. The UAVs are equipped with a global positioning sensor, such as GNSS or in the indoor case Optitrack. To detect the target, each UAV has an on-board monocular camera with accompanying computer vision software, capable of determining the relative position of the target in the camera frame. At a height of 1.5m off the ground, looking straight down the field of view of the camera is approximately 1.2m going front to back (x in the body frame) for the UAV and 0.9m going side to side (y in the body frame) for the UAV. Additionally, the camera can be oriented towards the target in both pitch and roll, through use of a Pan-Tilt Unit (*PTU*) (see Section 3.3). Lastly, the UAVs can communicate with each other any necessary information, however lossless communication is not assumed.

The target being tracked is an evasive mobile ground target capable of acceleration, with

a maximum speed not exceeding that of the UAVs. Furthermore, it is assumed that there is a single target of interest to track. The systems and their interactions for a single Pelican are shown in Figure 3.3. All modules are interfaced through ROS. Furthermore, the transform tree node transmits relevant rotation matrices and translation vectors between predefined coordinate frames of interest, like between the UAV frame and the camera frame. Furthermore, the indoor experimental set-up is shown in Figure 3.2

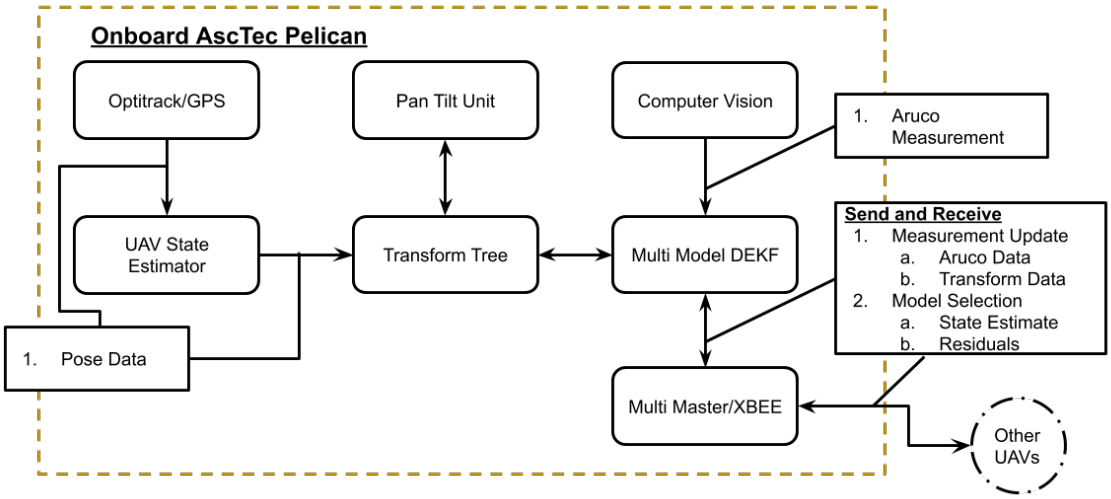


Figure 3.3: System overview for an AscTec Pelican. In indoor tests, Optitrack and the Multi Master framework are used for UAV position data and inter-UAV communication respectively, transmitted through a local wifi network in the lab. GNSS and the XBEE module are the outdoor equivalents.

3.2.2.3 Control Experimental Setup

There are only a few changes between the state estimation and control indoor experimental setups. The resultant system overview is shown in Figure 3.4.

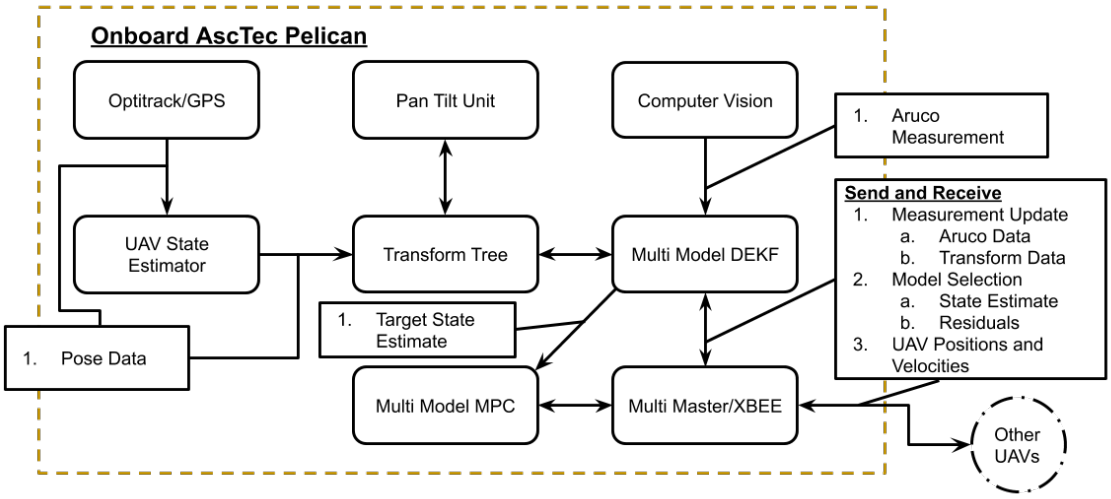


Figure 3.4: For the control setup, the controller being tested, that is the multi model MPC has been added. Furthermore, UAVs now share their own position and velocity estimates.

Now that there is a controller, the UAVs are no longer hanging from the ceiling. Another change to the setup is that UAVs now share their position and velocity estimates as well, for reasons that will be seen in Chapter 5.

3.2.2.4 Simulation Setup

To test the controller with multiple UAVs, VREP in coordination with MATLAB was used. The corresponding simulation setup is shown in Figure 3.5. Furthermore, the general appearance of the simulation environment is shown in Figure 3.6. VREP is a real-time simulation environment where models of UAVs can be imported and flown by specifying attitude commands. Furthermore, these models are represented fully in 3D and can therefore be subject to collisions should they occur.

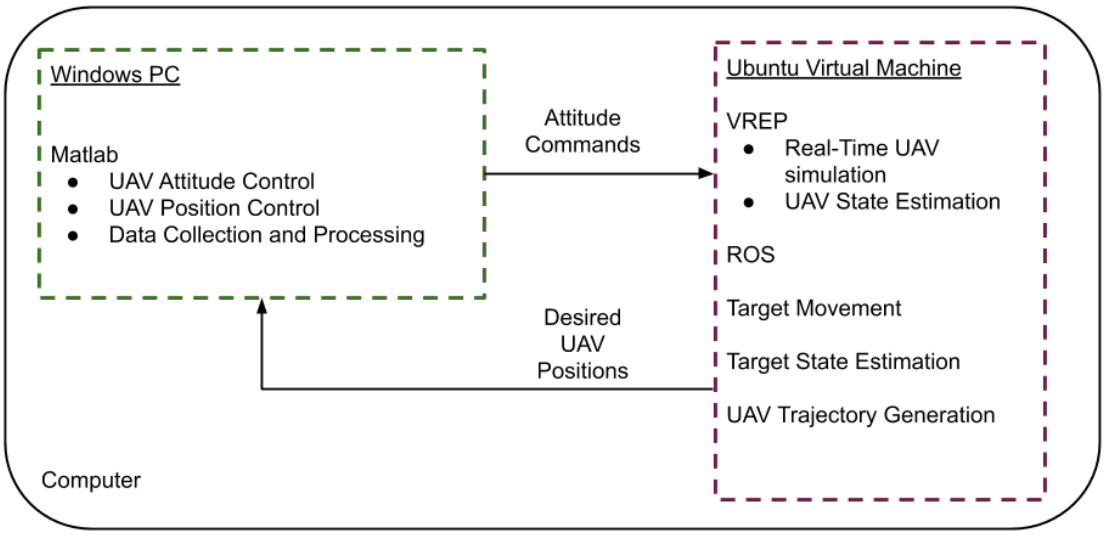


Figure 3.5: In simulation, there is no sensing being done, all relevant information is directly accessible. The simulation takes place on one physical machine running Windows with a Virtual Machine running Ubuntu (for ROS).

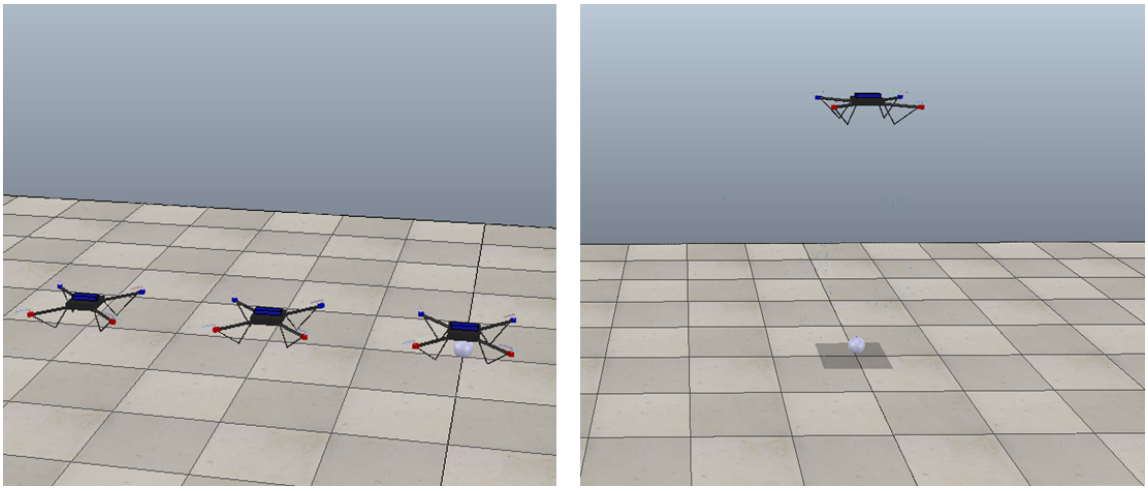


Figure 3.6: The general appearance of the quadrotors and target (blue ball) for the VREP simulation is shown here.

3.3 Pan-Tilt Unit

This section explains the functionality and algorithms in the Pelican's PTU, otherwise known as a 2-axis gimbal. More precisely, the AscTec Pelican has an attachment with

two servo motors whereby the Pelican's camera can be stabilized. Given a desired orientation for the camera, the HLP of the Pelican will attempt to maintain this orientation by accounting for the UAV's pitch and roll. For the purposes of target tracking, it is desirable to keep the target in the center of the frame, in order to minimize occlusions arising from any sudden deviations in the target's trajectory.

In the following formulation, a UAV equipped with a PTU is tracking a target. It detects the target via an on board camera and an accompanying computer vision algorithm which returns the position of the target relative to the camera. These target estimates are then combined in an EKF to obtain the target's position in the world frame. Furthermore, the UAV has access to its own position in the world frame through a separate on-board EKF. Given this information, the gimbal should attempt to ensure that the target remains in the center of the camera frame. To accomplish this, the inputs to the gimbal that should be specified are its roll and pitch, denoted as θ_p and ϕ_p, respectively.

3.3.1 Coordinate Frames

For the purpose of this formulation, the Pelican's frames are organized as shown in Figure 3.7 below.

The translation vectors, that is $\mathbf{t}_b^g$ and $\mathbf{t}_g^c$, are vectors that represent the necessary translation needed to go from one coordinate frame to the other. They are resolved in the final coordinate frame and depend on the current roll and pitch angles of the gimbal.

3.3.2 Transformation Matrices

The overall equation relating the position of the target in the world frame to that in the camera frame is given by

$$ {}_g\mathbf{p}^c = T_g^c T_b^g T_{wg}^b \mathbf{p}^w \,, \tag{3.1} $$

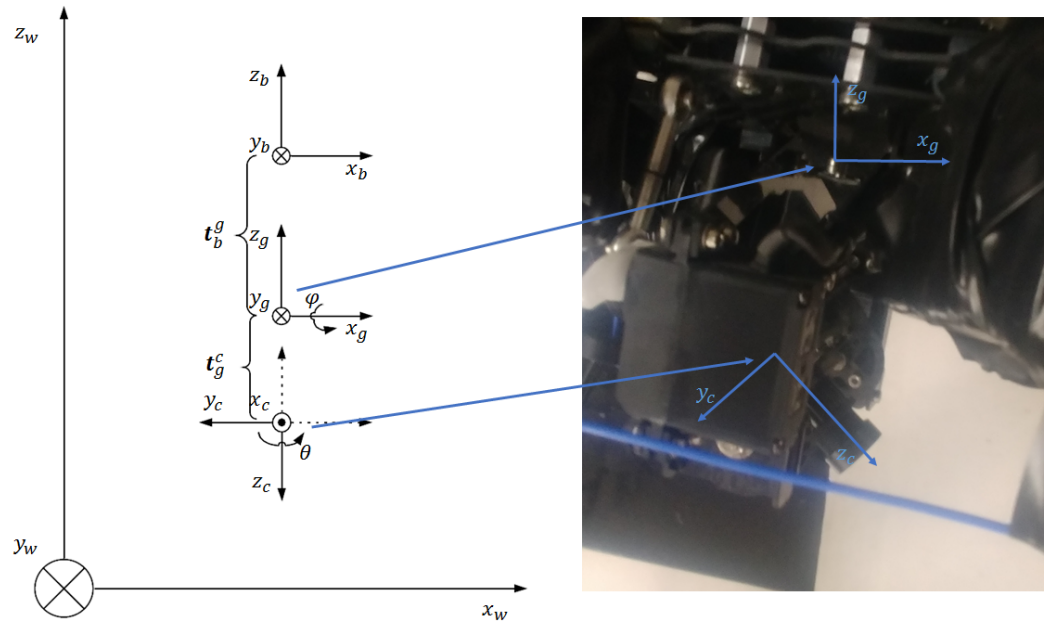


Figure 3.7: The world (w), body (b), PTU (p) and camera (c) frames of reference. Also, the angles of rotation of the gimbal (θ_p and ϕ_p) and associated translation vectors between frames. Note that the gimbal frame (g) is set to a link on the Pelican from which the gimbal rolls. Furthermore, the body frame is not presented in right hand figure; it is set to the centre of gravity of the Pelican, which is right below the battery placement (see Figure 3.2).

where the vector ${}_g\mathbf{p}$ refers to the position of the target in the respective frame. Note that from the on-board EKF estimating the position of the UAV, T_w^b is readily available and does not depend on the gimbal's roll and pitch angles. Thus $T_{wg}^b \mathbf{p}^w$ is known and will be defined as

$$ {}_g\mathbf{p}^b = T_{wg}^b \mathbf{p}^w \,, \tag{3.2} $$

simplifying equation (3.1) to

$$ {}_g\mathbf{p}^c = T_g^c T_b^g \, {}_g\mathbf{p}^b \,. \tag{3.3} $$

As the vector ${}_g\mathbf{p}^c$ is also known, it is the measurement from CV, all that remains to identify are the transformation matrices T_g^c and T_b^g.

3.3.2.1 T_g^c

Going from the gimbal from to the camera frame is a combination of three operations. That is, a change of axis, a rotation and a translation. The corresponding transformation matrix is

$$T_g^c = \begin{bmatrix} C_g^c & -\mathbf{t}_g^c \\ \mathbf{0} & 1 \end{bmatrix} . \tag{3.4}$$

Note that $\mathbf{0}$ represents the corresponding zero vector. The rotational aspect C_g^c of the transformation matrix consists of a change in axis followed by a rotation about the x axis of the camera frame or in other words, the pitch of the camera. The rotation matrix associated with the axis transformation, in accordance with Figure 3.7, is derived from the fact that $x_c = -y_g$, $y_c = -x_g$ and $z_c = -z_g$. Therefore, the consequent rotation matrix is written as follows

$$\begin{aligned} C_g^c &= C_x^c(\phi_p) C_{axis} \\ &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_{\phi_p} & s_{\phi_p} \\ 0 & -s_{\phi_p} & c_{\phi_p} \end{bmatrix} \begin{bmatrix} 0 & -1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \\ &= \begin{bmatrix} 0 & -1 & 0 \\ -c_{\phi_p} & 0 & -s_{\phi_p} \\ s_{\phi_p} & 0 & -c_{\phi_p} \end{bmatrix} . \end{aligned} \tag{3.5}$$

The rotation matrix $C_x^c(\phi_p)$ refers to a rotation in the camera frame about the x axis according to the angle ϕ_p, as detailed in Figure 3.7. The C_{axis} rotation matrix represents the

rotation to change axis according to Figure 3.7. The translation vector $\mathbf{t}_g^c$ must be resolved in the camera frame and therefore depends on the current pitch of the camera. In order to do so, first an original translation vector is defined as

$$\mathbf{t}_g \triangleq \begin{bmatrix} (t_x)_g^c \\ (t_y)_g^c \\ (t_z)_g^c \end{bmatrix} . \tag{3.6}$$

This vector of constants represents the distances in x, y and z required to go from the gimbal frame's origin to that of the camera frame. The distances are measured according to the camera's original frame of reference ($\phi_p = 0$) and are constants. With $\mathbf{t}_g$ defined, the final translation vector is written as

$$\mathbf{t}_g^c = C_x^c(\phi_p)\mathbf{t}_g \, . \tag{3.7}$$

Using the information from equations (3.5) and (3.7), the closed form representation of equation (3.4) is given as

$$\begin{aligned} T_g^c &= \begin{bmatrix} C_g^c & -\mathbf{t}_g^c \\ \mathbf{0} & 1 \end{bmatrix} \\ &= \begin{bmatrix} 0 & -1 & 0 & -(t_x)_g^c \\ -c_{\phi_p} & 0 & -s_{\phi_p} & -(t_y)_g^c c_{\phi_p} - (t_z)_g^c s_{\phi_p} \\ s_{\phi_p} & 0 & -c_{\phi_p} & (t_y)_g^c s_{\phi_p} - (t_z)_g^c c_{\phi_p} \\ 0 & 0 & 0 & 1 \end{bmatrix} . \end{aligned} \tag{3.8}$$

3.3.2.2 T_b^g

Using the same procedure as in Section 3.3.2.1, the transformation matrix in going from the body frame to the gimbal frame is given by

$$T_b^g = \begin{bmatrix} C_b^g & -\mathbf{t}_b^g \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & -(t_x)_b^g \\ 0 & c_{\theta_p} & s_{\theta_p} & -(t_y)_b^g c_{\theta_p} - (t_z)_b^g s_{\theta_p} \\ 0 & -s_{\theta_p} & c_{\theta_p} & (t_y)_b^g s_{\theta_p} - (t_z)_b^g c_{\theta_p} \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.9}$$

3.3.2.3 T_b^c

With equations (3.8) and (3.9), a closed form representation for equation (3.1) can be obtained. First, a final transformation matrix is defined

$$T_b^c = T_g^c T_b^g$$

$$T_b^c = \begin{bmatrix} 0 & -c_{\theta_p} & -s_{\theta_p} & (t_y)_b^g c_{\theta_p} + (t_z)_b^g s_{\theta_p} - (t_x)_g^c \\ -c_{\phi_p} & s_{\phi_p} s_{\theta_p} & -s_{\phi_p} c_{\theta_p} & (t_x)_b^g c_{\phi_p} + (t_z)_b^g c_{\theta_p} s_{\phi_p} - (t_y)_b^g s_{\theta_p} s_{\phi_p} - (t_y)_p^c c_{\phi_p} - (t_z)_g^c s_{\phi_p} \\ s_{\phi_p} & c_{\phi_p} s_{\theta_p} & -c_{\phi_p} c_{\theta_p} & -(t_x)_b^g s_{\phi_p} + (t_z)_b^g c_{\theta_p} c_{\phi_p} - (t_y)_b^g s_{\theta_p} c_{\phi_p} + (t_y)_g^c s_{\phi_p} - (t_z)_g^c c_{\phi_p} \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.10}$$

3.3.3 Solving for PTU Roll and Pitch

Now that T_b^c has been obtained, equation (3.1) is simplified to

$$\overset{{}_g\mathbf{p}^c = T_b^c\, {}_g\mathbf{p}^b}{\Rightarrow} \begin{bmatrix} {}_g(p_x)^c \\ {}_g(p_y)^c \\ {}_g(p_z)^c \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & -c_{\theta_p} & -s_{\theta_p} & (t_y)_b^g c_{\theta_p} + (t_z)_b^g s_{\theta_p} - (t_x)_g^c \\ -c_{\phi_p} & s_{\phi_p} s_{\theta_p} & -s_{\phi_p} c_{\theta_p} & (t_x)_b^g c_{\phi_p} + (t_z)_b^g c_{\theta_p} s_{\phi_p} - (t_y)_b^g s_{\theta_p} s_{\phi_p} - (t_y)_g^c c_{\phi_p} - (t_z)_g^c s_{\phi_p} \\ s_{\phi_p} & c_{\phi_p} s_{\theta_p} & -c_{\phi_p} c_{\theta_p} & -(t_x)_b^g s_{\phi_p} + (t_z)_b^g c_{\theta_p} c_{\phi_p} - (t_y)_b^g s_{\theta_p} c_{\phi_p} + (t_y)_g^c s_{\phi_p} - (t_z)_g^c c_{\phi_p} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}_g(p_x)^b \\ {}_g(p_y)^b \\ {}_g(p_z)^b \\ 1 \end{bmatrix}, \tag{3.11}$$

where a 1 is appended to the position vectors in order to carry out the transformation.

3.3.3.1 Simplifications

With the above system of equations, the desired roll and pitch angles to keep the camera frame centered can be determined accordingly. However, there are some simplifications that can be made based on the physical constraints of the system at hand. First, note that both the components of ${}_g\mathbf{p}^c$ and ${}_g\mathbf{p}^b$ are either measured or fixed to a desired value and can therefore be treated as known constants. More specifically, ${}_g\mathbf{p}^b$ is obtained from a combination of separate EKF estimates for the UAV's pose and the target's position, respectively. For ${}_g\mathbf{p}^c$, however, setting it as

$${}_g\mathbf{p}^c = \begin{bmatrix} 0 & 0 & {}_g(p_z)^c \end{bmatrix}^T \tag{3.12}$$

ensures that the target will be centered in the camera frame. In other words, the desired position of the target in the camera frame is fixed to zero in x and y. For ${}_g(p_z)^c$, this is known once again through EKF estimates.

A second set of simplifications exist with vectors $\mathbf{t}_g$ and $\mathbf{t}_b$. As these are specific to the

system, they are subject to change from Pelican to Pelican. However, they will take the following forms

$$\mathbf{t}_g = \begin{bmatrix} 0 & 0 & (t_z)_g^c \end{bmatrix}^T \tag{3.13}$$

$$\mathbf{t}_b = \begin{bmatrix} 0 & 0 & (t_z)_b^g \end{bmatrix}^T , \tag{3.14}$$

signifying the fact that essentially the camera is located directly under the center of mass of the Pelican.

3.3.3.2 The Desired Roll

In order to solve for the desired roll angle to center the camera on the target, the first equation in the system of equations depicted in equation (3.11) is used

$$\begin{aligned} {}_g(p_x)^c &= -{}_g(p_y)^b c_{\theta_p} - {}_g(p_z)^b s_{\theta_p} + (t_y)_b^g c_{\theta_p} + (t_z)_b^g s_{\theta_p} - (t_x)_g^c \\ \Rightarrow 0 &= -{}_g(p_y)^b c_{\theta_p} + \left(-{}_g(p_z)^b + (t_z)_b^g\right) s_{\theta_p} \\ \Rightarrow {}_g(p_y)^b c_{\theta_p} &= \left(-{}_g(p_z)^b + (t_z)_b^g\right) s_{\theta_p} \\ \Rightarrow \theta_p &= \tan^{-1}\left(\frac{{}_g(p_y)^b}{-{}_g(p_z)^b + (t_z)_b^g}\right) , \end{aligned} \tag{3.15}$$

coupled with the aforementioned simplifications (i.e. ${}^g(p_x)^c = 0$).

3.3.3.3 The Desired Pitch

For the desired pitch, the following two equations in equation (3.11) are used and simplified similarly. For the second equation

$$
\begin{aligned}
&{}_g(p_y)^c = -{}_g(p_x)^b c_{\phi_p} + {}_g(p_y)^b s_{\phi_p} s_{\theta_p} - {}_g(p_z)^b s_{\phi_p} c_{\theta_p} + (t_x)^g_b c_{\phi_p} + (t_z)^g_b c_{\theta_p} s_{\phi_p} - (t_y)^g_b s_{\theta_p} s_{\phi_p} - (t_y)^c_g c_{\phi_p} - (t_z)^c_g s_{\phi_p} \\
&\Rightarrow 0 = -{}_g(p_x)^b c_{\phi_p} + {}_g(p_y)^b s_{\phi_p} s_{\theta_p} - {}_g(p_z)^b s_{\phi_p} c_{\theta_p} + (t_z)^g_b c_{\theta_p} s_{\phi_p} - (t_z)^c_g s_{\phi_p} \\
&\Rightarrow 0 = -{}_g(p_x)^b c_{\phi_p} + ({}_g(p_y)^b s_{\theta_p} - {}_g(p_z)^b c_{\theta_p} + (t_z)^g_b c_{\theta_p} - (t_z)^c_g) s_{\phi_p}
\end{aligned} \tag{3.16}
$$

and similarly for the third equation

$${}^g(p_z)^c = {}^g(p_x)^b s_{\theta_p} + ({}^g(p_y)^b s_{\phi_p} - {}^g(p_z)^b c_{\phi_p} + (t_z)^g_b c_{\phi_p} - (t_z)^c_g) c_{\theta_p} \,. \tag{3.17}$$

Combining equations (3.16) and (3.17) into their own system of equations yields the following

$$
\begin{bmatrix}
{}_g(p_y)^b s_{\phi_p} - {}^g(p_z)^b c_{\phi_p} + (t_z)^g_b c_{\phi_p} - (t_z)^c_g & -{}^g(p_x)^b \\
{}^g(p_x)^b & {}_g(p_y)^b s_{\phi_p} - {}^g(p_z)^b c_{\phi_p} + (t_z)^g_b c_{\phi_p} - (t_z)^c_g
\end{bmatrix}
\begin{bmatrix} s_{\theta_p} \\ c_{\theta_p} \end{bmatrix}
=
\begin{bmatrix} 0 \\ {}^g(p_z)^c \end{bmatrix} ,
\tag{3.18}
$$

which gives a solution for $\sin\phi_p$ and $\cos\phi_p$ as

$$\sin\phi_p = \frac{{}_g(p_x)^b \cdot {}_g(p_z)^c}{\left({}_g(p_y)^b s_{\theta_p} - {}_g(p_z)^b c_{\theta_p} + (t_z)^g_b c_{\theta_p}\right)^2 + ({}_g(p_x)^b)^2} \tag{3.19}$$

$$\cos\phi_p = \frac{\left({}_g(p_y)^b s_{\theta_p} - {}_g(p_z)^b c_{\theta_p} + (t_z)^g_b c_{\theta_p}\right) \cdot {}_g(p_z)^c}{\left({}_g(p_y)^b s_{\theta_p} - {}_g(p_z)^b c_{\theta_p} + (t_z)^g_b c_{\theta_p}\right)^2 + ({}_g(p_x)^b)^2} \,. \tag{3.20}$$

Solving equations (3.19) and (3.20) for ϕ_p yields the following solution

$$\phi_p = \tan^{-1}\left(\frac{{}_g(p_x)^b}{{}_g(p_y)^b s_{\theta_p} - {}_g(p_z)^b c_{\theta_p} + (t_z)^g_b c_{\theta_p}}\right) . \tag{3.21}$$

3.3.4 Yaw Control

As the gimbal only has control over roll and pitch angles concerning the orientation of the camera, it is therefore limited in yaw. In a fixed wing UAV scenario, yawing the UAV such that the camera faces the target would mean a deviation from the UAVs desired trajectory, which is undesirable. However, this is not the case for VTOL UAV, which theoretically can yaw in any direction while maintaining the intended course. Thus, having this capability available, it is desirable to have the gimbal control the roll and the pitch while the UAV itself can control the yaw.

3.3.4.1 Coordinate Frame

Figure 3.8 shows the relevant transformation necessary to point the camera to the target.

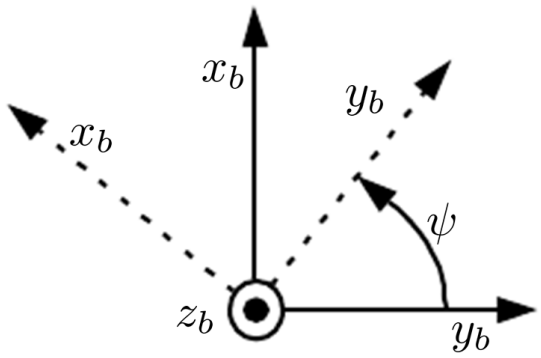


Figure 3.8: Orienting the quadrotor frame so that the camera points at the target, through changes in the yaw angle, ψ.

3.3.5 Transformation Matrices

The overall equation relating the position of the target in the world frame to that of the quadrotor frame is given by

$$ {}_g\mathbf{p}^b = T^b_{wg}\mathbf{p}^w \,, \tag{3.22}$$

where T_w^b is given by

$$T_w^b = \begin{bmatrix} C_w^b & -\mathbf{t}_w^b \\ \mathbf{0} & 1 \end{bmatrix} . \tag{3.23}$$

The consequent rotation matrix is written as follows

$$C_w^b = \begin{bmatrix} c_\psi & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} . \tag{3.24}$$

The translation vector $\mathbf{t}_w^b$ must be resolved in the body frame and therefore depends on the current yaw of the vehicle. In order to do so, first an original translation vector is defined as

$$\mathbf{t}_w \triangleq \begin{bmatrix} (t_x)_w^b \\ (t_y)_w^b \\ (t_z)_w^b \end{bmatrix} . \tag{3.25}$$

This vector is a vector of constants representing the distances in x, y and z required to go from the world frame's origin to that of the body frame. Or, in other words, the current position of the UAV. Thus, the final translation vector is written as

$$\mathbf{t}_w^b = C_z^b(\psi)\mathbf{t}_w . \tag{3.26}$$

Using the information from equations (3.24) and (3.26), the closed form representation to equation (3.23) is given as

$$T_w^b = \begin{bmatrix} C_w^b & -\mathbf{t}_w^b \\ \mathbf{0} & 1 \end{bmatrix}$$
$$= \begin{bmatrix} c_\psi & -s_\psi & 0 & -\left((t_x)_w^b c_\psi - (t_y)_w^b s_\psi\right) \\ s_\psi & c_\psi & 0 & -\left((t_x)_w^b s_\psi + (t_y)_w^b c_\psi\right) \\ 0 & 0 & 1 & -(t_z)_w^b \\ 0 & 0 & 0 & 1 \end{bmatrix} . \tag{3.27}$$

3.3.6 Solving for Yaw

Now that T_w^b has been obtained, equation (3.22) is simplified to

$$ {}_g\mathbf{p}^b = T_{wg}^b \mathbf{p}^w$$
$$\Rightarrow \begin{bmatrix} {}_g(p_x)^b \\ {}_g(p_y)^b \\ {}_g(p_z)^b \\ 1 \end{bmatrix} = \begin{bmatrix} c_\psi & -s_\psi & 0 & -\left((t_x)_w^b c_\psi - (t_y)_w^b s_\psi\right) \\ s_\psi & c_\psi & 0 & -\left((t_x)_w^b s_\psi + (t_y)_w^b c_\psi\right) \\ 0 & 0 & 1 & -(t_z)_w^b \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}_g(p_x)^w \\ {}_g(p_y)^w \\ {}_g(p_z)^w \\ 1 \end{bmatrix} , \tag{3.28}$$

where a 1 is appended to the position vectors in order to carry out the transformation. For the desired yaw, the first two equations in equation (3.28) are used and simplified similarly. For the first equation

$$ {}_g(p_x)^b = {}_g(p_x)^w c_\psi - {}_g(p_y)^w s_\psi - (t_x)_w^b c_\psi + (t_y)_w^b s_\psi \, , \tag{3.29}$$

and similarly for the third equation

$$ {}_g(p_y)^b = {}_g(p_x)^w s_\psi + {}_g(p_y)^w c_\psi - (t_x)_w^b s_\psi - (t_y)_w^b c_\psi \, . \tag{3.30}$$

The only simplification being that since the target should be directly in front of the UAV, which is the direction in which the camera faces, ${}_g(p_y)^b = 0$. Thus, combining equations (3.29) and (3.30) into their own system of equations yields the following

$$\begin{bmatrix} {}_g(p_x)^w - (t_x)^b_w & {}_g(p_y)^w - (t_y)^b_w \\ -{}_g(p_y)^w + (t_y)^b_w & {}_g(p_x)^w - (t_x)^b_w \end{bmatrix} \begin{bmatrix} s_\psi \\ c_\psi \end{bmatrix} = \begin{bmatrix} 0 \\ {}_g(p_x)^b \end{bmatrix}, \tag{3.31}$$

which gives a solution for ψ as

$$\psi = \tan^{-1}\left(\frac{{}_g(p_y)^w - (t_y)^b_w}{{}_g(p_x)^w - (t_x)^b_w}\right). \tag{3.32}$$

Chapter 4

Distributed Multi Model EKF Using T Test

4.1 Opening Remarks

In the world of UAVs, one application that has been thoroughly researched is that of target tracking, where a single UAV or team of UAVs attempt to maintain an estimate of where the target of interest is located. In order to accomplish this, a common technique used is Kalman filtering, where a model of how the target moves is fused with measurement information on the target's location. Therefore, one problem that degrades the estimation is when the target moves in a way that does not fit the model accurately. To combat this, in the single UAV case, some implementations use different models at different times, depending on the target's motion. Presented here is the first attempt of using multiple models in a scenario where a team of UAVs is tracking a target. It combines the T Test method for multiple model estimation with the traditional Distributed Extended Kalman Filtering (*DEKF*) used in multi-UAV scenarios. The algorithm is validated in both indoor and outdoor flights. First, it is shown that having multiple models improves estimation over having a single model. Next, a performance comparison of various methods shows that the T Test provides the best results.

The remainder of the report is divided as follows: Related Works in Section 4.2, Multi Model Distributed Extended Kalman Filter in Section 4.3, Other Fusion Methods in Section 4.4, Occlusions in Section 4.5, Results in Section 4.6, Conclusion in Section 4.7.

4.2 Related Works

This report explores multi-UAV state estimation in a novel manner, through the use of the EKF. Each UAV in the network will use an EKF with a separate model to predict the target's motion, after which their respective estimates will be shared and the best one chosen according to the T Test. Thus, this Multi Model Distributed EKF utilizes ideas from two main areas of research. The first is single agent target tracking using multiple models in bank of EKF filters. The second is a multi agent DEKF using a single model. It is therefore of interest to examine previous works in both of these areas.

4.2.1 Single Agent Multiple Model Estimation

Multiple model estimation that pertains to target tracking with a UAV has several main approaches. One of the more common approaches thus far is the Interchanging Multiple Model approach (*IMM*), which was first introduced in [66]. In this approach, generally at each iteration the probability of each possible model is calculated and final estimate of the target is generated based on these probabilities. The IMM algorithm has been paired with various forms of filters for the purpose of target tracking. For example, in [67], the IMM technique was used in conjunction with the Curbature Kalman Filter (*CKF*) for target tracking purposes. It has also been paired with the UKF for target tracking UAVs [68]. Another common technique used in multiple model scenarios is hypothesis testing, of which the T Test is a variant. One example of hypothesis testing is [69], where a controller is designed to switch models based on a performance index. In [70], a similar hypothesis testing algorithm is paired with a KF. One last method for multiple model estimation, gaining in popularity is the Variable Structure Multiple Model (*VSMM*), where one example is [71]. The idea behind VSMM is to have a varying amount of models to choose from, based on the situation, or to adaptively change the models themselves. For more information on multiple model estimation, a comprehensive survey is done in [72], specifically for target

tracking purposes.

4.2.2 Multi UAV Estimation

In situations where there are multiple agents making observations of the target, several methods as to how to combine these estimates have arisen. One of the more popular being the distributed Kalman filter, where multiple sensors or agents make observations on the same target and come to a consensus on a global estimate [73]. As the DKF is a building block of more complex distributed filters, a formulation adapted from [21], where a team of three ground robots use the DKF to track a mobile ground target, is shown here. The DKF filter attempts to use neighbour estimates to better estimate the target's position. For the formulation, let the set of all robots be defined as $\mathcal{Z}$, with UAV $\zeta \in \mathcal{Z}$. Furthermore, let $\mathcal{J}$ represent the set of neighbours to UAV $i \in \mathcal{I}$, as $j \in \mathcal{J}$. Note that $\mathcal{I} \cap \mathcal{J} = \emptyset$ and that $\mathcal{I}, \mathcal{J} \subset \mathcal{Z}$. Like the standard KF, the DKF uses an Linear Time-Invariant (*LTI*) representation of the target dynamics, namely, as

$$ {}_g\mathbf{x}(k) = {}_gF\,{}_g\mathbf{x}(k-1) + \mathbf{b}(k) \tag{4.1} $$

where ${}_g\mathbf{x}(k)$ is the state vector, ${}_gF$ is the prediction matrix and $\mathbf{b}(k)$ is the noise associated with the prediction and is drawn from a covariance matrix. For the measurement model, complexity is added when fusing measurements from different robots. Say the DKF of a single robot is examined, the measurement equation of said DKF is

$$ {}^i_g\mathbf{z}(k) = {}^i_{fg}H^i_{fg}\mathbf{x}(k) + \mathbf{d}(k) \tag{4.2} $$

where

$$
\begin{aligned}
{}^{i}_{fg}\mathbf{z}(k) &= \begin{bmatrix} {}^{i}_{g}\mathbf{z}(k) & {}^{j}_{g}\mathbf{z}(k) \end{bmatrix}^{\mathsf{T}} \\
{}^{i}_{fg}H &= \begin{bmatrix} {}^{i}_{g}H & {}^{j}_{g}H \end{bmatrix}^{\mathsf{T}} \qquad \forall j \in \mathcal{J}\,.
\end{aligned}
$$

In this case ${}^{i}_{fg}\mathbf{z}(k)$ is the overall vector of measurements for robot i. It consists of robot i's own estimate of the target ${}^{i}_{g}\mathbf{z}(k)$ and its set of neighbours' ($\mathcal{J}$) measurements, ${}^{j}_{g}\mathbf{z}(k)$. Similarly, ${}^{j}_{fg}H$ is the overall measurement matrix of robot i, being a combination of its own measurement matrix ${}^{i}_{g}H$ and its neighbours', ${}^{i}_{g}H$. The prediction step being the same as a traditional KF, the measurement update step is altered from the traditional KF as

$$
\begin{aligned}
{}^{i}_{fg}\hat{\mathbf{x}}^{+}(k) &= {}^{i}_{fg}\hat{\mathbf{x}}^{-}(k) + {}^{i}_{g}P^{-}(k){}^{i}_{fg}H^{\mathsf{T}}{}^{i}_{fg}R^{-1}({}^{i}_{fg}\mathbf{z}(k) - {}^{i}_{fg}H{}^{i}_{fg}\hat{\mathbf{x}}^{-}(k)) \\
&= {}^{i}_{fg}\hat{\mathbf{x}}^{+}(k) + {}^{i}_{g}P^{-}(k) \sum_{\zeta \in \mathcal{J}\cup\mathcal{I}} {}^{\zeta}_{g}H^{\mathsf{T}}{}^{\zeta}_{g}R^{-1}({}^{\zeta}_{g}\mathbf{z}(k) - {}^{\zeta}_{g}H{}^{i}_{g}\hat{\mathbf{x}}^{-}(k))\,. \qquad (4.3)
\end{aligned}
$$

Updating the a priori covariance is computed as

$$
\begin{aligned}
{}^{i}_{g}P^{+}(k)^{-1} &= {}^{i}_{g}P^{-}(k)^{-1} + {}^{i}_{fg}H^{\mathsf{T}}{}^{i}_{fg}R^{-1}{}^{i}_{fg}H \\
&= {}^{i}_{g}P^{-}(k)^{-1} + \sum_{\zeta \in \mathcal{J}\cup\mathcal{I}} {}^{\zeta}_{g}H^{\mathsf{T}}{}^{\zeta}_{g}R^{-1}{}^{\zeta}_{g}H \qquad (4.4)
\end{aligned}
$$

The second term in equation (4.3) is where cooperation between robots is taking place. In the second term, robot i is using the neighbours' measurements of the target state for its own estimate of the target position. Other examples of distributed filtering technique include a distributed curbature information filter paired with the IMM for UAV target tracking [74], a distributed IMM UKF used in a microphone array [75] or even a distributed information filter for target tacking UAVs, where selection criteria determines which UAVs participate in the estimate fusion [76]. A overview of some common distributed filters for multi agent systems is given in [77].

4.3 Multi-Model Distributed Extended Kalman Filter

4.3.1 Distributed Extended Kalman Filter

This subsection addresses the DEKF used on each UAV. The DEKF relies on raw measurements and transform data from the neighbour UAVs within communication range. Thus, the equations presented here are ran simultaneously by each UAV.

4.3.1.1 Filter Equations

Once again, the set of all UAVs be defined as $\mathcal{Z}$, with UAV $\zeta \in \mathcal{Z}$. Furthermore, let $\mathcal{J}$ represent the set of neighbours to UAV $i \in \mathcal{I}$, as $j \in \mathcal{J}$. Note that $\mathcal{I} \cap \mathcal{J} = \emptyset$ and that $\mathcal{I}, \mathcal{J} \subset \mathcal{Z}$. Thus, the equations for the EKF of UAV i are formulated below. The EKFs for subsequent UAVs can be similarly abstracted. The filter begins with the initialization of the state and covariance matrix as

$$ {}^{i}_{g}\hat{\mathbf{x}}(k) = {}^{i}_{g}h^{-1}({}^{i}_{g}\mathbf{z}(0)) \tag{4.5} $$

$$ {}^{i}_{g}P(k) = {}^{i}_{g}P(0)\,. \tag{4.6} $$

In this case, the initial state is determined from the initial measurement ${}^{i}_{g}\mathbf{z}(0)$ of the target. Furthermore, the function $h : \mathbf{x} \rightarrow \mathbf{z}$ takes the states from the absolute coordinate frame to the measurement space, which in this case is the camera frame. Next, a prediction is made using UAV i's model of the target motion (see Section 4.3.2), as

$$ {}^{i}_{g}\hat{\mathbf{x}}^{-}(k) = {}^{i}_{g}f({}^{i}_{g}\hat{\mathbf{x}}^{+}(k-1)) \tag{4.7} $$

and for the covariance matrix

$$ {}^{i}_{g}P^{-}(k) = {}^{i}_{g}F(k){}^{i}_{g}P^{+}(k-1){}^{i}_{g}F^{T}(k) + {}^{i}_{g}Q\,, \tag{4.8} $$

where ${}^{i}_{g}Q$ is the respective covariance matrix for the process noise, which is dependent on the adopted model. Moreover, ${}^{i}_{g}F(k)$ can be found by taking the Jacobian of ${}^{i}_{g}f(\cdot)$ as

$$ {}^{i}_{g}F(k) = \frac{\partial^{i}_{g} f}{\partial^{i}_{g}\mathbf{x}}\Big|_{{}^{i}_{g}\hat{\mathbf{x}}^{+}(k-1)} . \tag{4.9} $$

Next is the update step, after receiving a measurement ${}^{i}_{g}\mathbf{z}(k)$ of the target relative to the camera frame, as well as ${}^{j}_{g}\mathbf{z}(k)$ of the neighbouring UAVS, for $j \in \mathcal{J}$. First comes calculation of the Kalman gain as

$$ {}^{i}_{g}K(k) = {}^{i}_{g}P^{-}(k){}^{i}_{g}H^{T}(k)\left({}^{i}_{g}H(k){}^{i}_{g}P^{-}(k){}^{i}_{g}H^{T}(k) + {}^{i}_{fg}R(k)\right)^{-1} , \tag{4.10} $$

where the covariance matrix ${}^{i}_{g}R(k)$ represents the measurement uncertainty. Note that it changes size based on the measurements available from the other UAVs and from itself. Also, the first order approximation of the measurement function is defined as

$$ {}^{i}_{g}H(k) = \frac{\partial^{i}_{fg} h}{\partial^{i}_{g}\mathbf{x}}\Big|_{{}^{i}_{g}\hat{\mathbf{x}}^{-}(k)} . \tag{4.11} $$

For the update of the state

$$ {}^{i}_{g}\hat{\mathbf{x}}^{+}(k) = {}^{i}_{g}\hat{\mathbf{x}}^{-}(k) + {}^{i}_{g}K(k)\left({}^{i}_{fg}\mathbf{z}(k) - {}^{i}_{fg}h({}^{i}_{g}\hat{\mathbf{x}}^{-}(k))\right) \tag{4.12} $$

is used and finally the covariance update is given by

$$ {}^{i}_{g}P(k)^{+} = \left({}^{i}_{g}I - {}^{i}_{g}K(k){}^{i}_{g}H(k)\right){}^{i}_{g}P(k)^{-} . \tag{4.13} $$

4.3.1.2 Measurement Model

Recall that at each time-step k the UAV i receives either a measurement ${}^{i}_{g}\mathbf{z}(k)$ from the camera of the relative position of the target and/or either one or two measurements ${}^{j}_{g}\mathbf{z}(k)$

from the other UAVs of the target's position in the respective UAVs camera frame, as well as the associated transform in going from the camera to the world frame at the time of measurement. Thus, the measurement function h should map $\mathbf{x} \rightarrow \mathbf{z}$ as

$$ {}^{i}_{g}\mathbf{z}(k) = {}^{i}_{g}h\left({}^{i}_{g}\hat{\mathbf{x}}^{+}(k)\right) \tag{4.14} $$

and thus for ${}^{i}_{g}\mathbf{z}(k)$ can be defined as follows

$$ {}^{i}_{g}h\left({}^{i}_{g}\hat{\mathbf{x}}(k)\right) \triangleq {}^{i}_{g}S\left({}^{i}_{g}C^{w}_{c}(k){}^{i}_{g}\mathbf{p}_{w}(k) + {}^{i}_{g}\mathbf{t}^{w}_{c}(k)\right) \tag{4.15} $$

where C^{w}_{c} is a rotation matrix from the world to the camera frame, $\mathbf{p}_{w}$ is the position of the target in the world frame (${}^{i}_{g}\mathbf{p}(k) \subset {}^{i}_{g}\hat{\mathbf{x}}(k)^{+}$) and $\mathbf{t}^{w}_{c}$ is the translation from the world to the camera frame. S is a projection onto the first two coordinates. Given these definitions and assuming a flat terrain for the vehicle (i.e. target height is negligible), (4.14) can be expanded as

$$ \begin{aligned}
{}^{i}_{g}\mathbf{z}(k) &= {}^{i}_{g}S\left({}^{i}_{g}C^{w}_{c}(k){}^{i}_{g}\mathbf{p}_{w}(k) + {}^{i}_{g}\mathbf{t}^{w}_{c}(k)\right) , \\
\Rightarrow \begin{bmatrix} x_c(k) \\ y_c(k) \end{bmatrix} &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \left(\begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \right) , \\
\Rightarrow \begin{bmatrix} x_c \\ y_c \end{bmatrix} &= \begin{bmatrix} c_{11}x_w + c_{12}y_w + c_{13}z_w + t_x \\ c_{21}x_w + c_{22}y_w + c_{23}z_w + t_y \end{bmatrix} , \\
z_w = 0 \Rightarrow \begin{bmatrix} x_c(k) \\ y_c(k) \end{bmatrix} &= \begin{bmatrix} c_{11}x_w + c_{12}y_w + t_x \\ c_{21}x_w + c_{22}y_w + t_y \end{bmatrix} .
\end{aligned} \tag{4.16} $$

For measurements coming from the cooperating UAVs, the measurement model is the same

$$\begin{aligned} {}^{j}_{g}\mathbf{z}(k) &= {}^{j}_{g}h({}^{i}_{g}\hat{\mathbf{x}}^{+}(k)) \,, \\ &= {}^{j}_{g}S\left({}^{j}_{g}C^{w}_{c}(k)\,{}^{i}_{g}\mathbf{p}_{w}(k) + {}^{j}_{g}\mathbf{t}^{w}_{c}(k)\right) \quad \forall j \in \mathcal{J} \,. \end{aligned} \tag{4.17}$$

With this information, and given the number of measurements made and received at time step k to be η, the final observation is

$${}^{i}_{fg}\mathbf{z}(k)_{m\times 1} \triangleq \begin{bmatrix} \mathbf{z}_{l} \\ \vdots \\ \mathbf{z}_{\eta} \end{bmatrix} \quad \text{for} \quad l = 0, \ldots, \eta \tag{4.18}$$

where $\mathbf{z}_l$ depends on which UAV $\zeta \in \mathcal{Z}$ the observation belongs to, and is defined as

$$\mathbf{z}_{l} = \begin{cases} {}^{i}_{g}\mathbf{z}(k), & \zeta \in \mathcal{I} \\ {}^{j}_{g}\mathbf{z}(k), & \zeta \in \mathcal{J} \\ \emptyset, & \eta = 0 \end{cases} . \tag{4.19}$$

A similar definition of the final measurement function is

$${}^{i}_{fg}h({}^{i}_{g}\hat{\mathbf{x}}^{+}(k))_{m\times 1} \triangleq \begin{bmatrix} h({}^{i}_{g}\hat{\mathbf{x}}^{+}(k))_{l} \\ \vdots \\ h({}^{i}_{g}\hat{\mathbf{x}}^{+}(k))_{\eta} \end{bmatrix} \quad \text{for} \quad l = 0, \ldots, \eta \tag{4.20}$$

where

$$h({}^{i}_{g}\hat{\mathbf{x}}^{+}(k))_{l} = \begin{cases} {}^{i}_{g}h\left({}^{i}_{g}\hat{\mathbf{x}}^{+}(k)\right), & \zeta \in \mathcal{I} \\ {}^{j}_{g}h\left({}^{i}_{g}\hat{\mathbf{x}}^{+}(k)\right), & \zeta \in \mathcal{J} \\ \emptyset, & \eta = 0 \end{cases} . \tag{4.21}$$

Furthermore, the overall measurement covariance matrix ${}^{i}_{fg}R(k)$ is expanded from R, which is for a single observation, η times as

$$ {}^{i}_{fg}R(k)_{2\eta\times 2\eta} \triangleq \begin{bmatrix} R & & 0 \\ & \ddots & \\ 0 & & R \end{bmatrix} . \tag{4.22} $$

4.3.2 Multi-UAV Predictions Models

To achieve a better estimate of the target, each UAV will adopt a different prediction model, similar to an IMM EKF for a single UAV, except in this case the computation is distributed over the team of UAVs. In the end, each UAV will produce a slightly different estimate of the target and the best estimate will be chosen to match the target's actual motion. The prediction models are discussed in 2.5.2, and are the CV, CA and CT models. In a team of three UAVs each adopts a different one.

4.3.3 T Test Selection

In order to select which model is providing the best estimate among the UAVs, the T Test selection method is adapted from [78]. In this method, the residual of each UAV's DEKF is compared and the DEKF that returns the lowest T Test value is then used by all UAVs as the estimate of the target's states for that timestep.

To begin the formulation, imagine that at time step k UAV i may or may not receive UAV j's current target state estimate and the past n residuals from UAV j, spanning time steps $[k-n, k[$. Then in the case of N UAVs, the result of the T Test, ${}^{i}t_{stat}$ is given by

$$ {}^{i}t_{stat}(k) = \frac{{}^{i}\bar{\epsilon}(k) - \bar{\kappa}(k)}{S_d(k)/\sqrt{n}} , \tag{4.23} $$

where the mean of the residuals for UAV i, ${}^{i}\bar{\epsilon}$, is given by

$$ {}^{i}\bar{\epsilon}(k) = \frac{\Sigma_{l=k-n}^{k} || {}^{i}_{fg}\mathbf{z}(l) - {}^{i}_{fg}h({}^{i}_{g}\hat{\mathbf{x}}^{+}(l)) ||_2^2}{n} \tag{4.24} $$

and the ideal mean of the residual $\bar{\kappa}(k) = 0$. The standard deviation of the samples $S_d(k)$ is given by

$$ S_d^2(k) = \frac{\Sigma_{j=1}^{N}(\Sigma_{l=k-n}^{k}(\bar{E}(l) - {}^{j}\epsilon(l))^2)}{n \cdot N - 1}\,, \tag{4.25} $$

with the mean of all residuals

$$ \bar{E}(k) = \frac{E(k)}{n \cdot N} \tag{4.26} $$

and finally the sum of all residuals is given by

$$ E(k) = \Sigma_{j=1}^{N}\Sigma_{l=k-n}^{k} {}^{j}\epsilon(l)\,, \tag{4.27} $$

with the norm of the residual

$$ {}^{j}\epsilon(k) = || {}^{j}_{fg}\mathbf{z}(k) - {}^{j}_{fg}h({}^{j}_{g}\hat{\mathbf{x}}^{+}(k)) ||_2^2\,. \tag{4.28} $$

Thus, at each iteration, UAV i produces at most N t_{stat} values, each associated with the estimate using either the CV, CA or CT models, and the estimate corresponding to the lowest t_{stat} value, termed ${}^{j}_{fg}\hat{\mathbf{x}}^{+}(k)$, is used by the UAV as its current estimate, in UAV control for example.

4.3.4 Overall Multi-Model Distributed EKF Architecture

The overall flow of information is depicted in Figure 4.1. There are essentially three processes running on each UAV. The first and second are the prediction and update of the

DEKF, where individual UAVs send and receive measurement and transform data and fuse it with its prediction model. The third process involves the sharing of the a posteriori estimate and past residuals. For each set of set of estimates and residuals the UAV receives, it computes the corresponding t_{stat} value. The estimate with the lowest value is then used as that UAVs final estimate. Note that the estimate used in the next iteration is not the final estimate corresponding to the T Test but rather the UAVs respective a posteriori estimate, for reasons that are explained in Section 4.6.

4.4 Other Fusion Methods

In this Section, we begin the formulation for a performance comparison between three commonly used data fusion methods for EKFs. The first is the T Test method that was previously explained. The other two, a probability based IMM EKF and a covariance-based ellipse fusion method, are explained here.

4.4.1 IMM-EKF

The goal of this method is to determine how much each model should be trusted at given timestep and operates in essence as a weighted average of the estimates from each model. It is adapted from its single UAV counterpart [79]. This method consists of three main steps: computation of the likelihood function, probability update and state estimate combination.

4.4.1.1 Likelihood Function

The likelihood function is the density function using the residual and innovation covariance information of UAV i's DEKF. Given the residual

$$ {}^{i}_{g}\upsilon(k) \triangleq {}^{i}_{fg}\mathbf{z}(k) - {}^{i}_{fg}h({}^{i}_{g}\hat{\mathbf{x}}^{+}(k)) \tag{4.29} $$

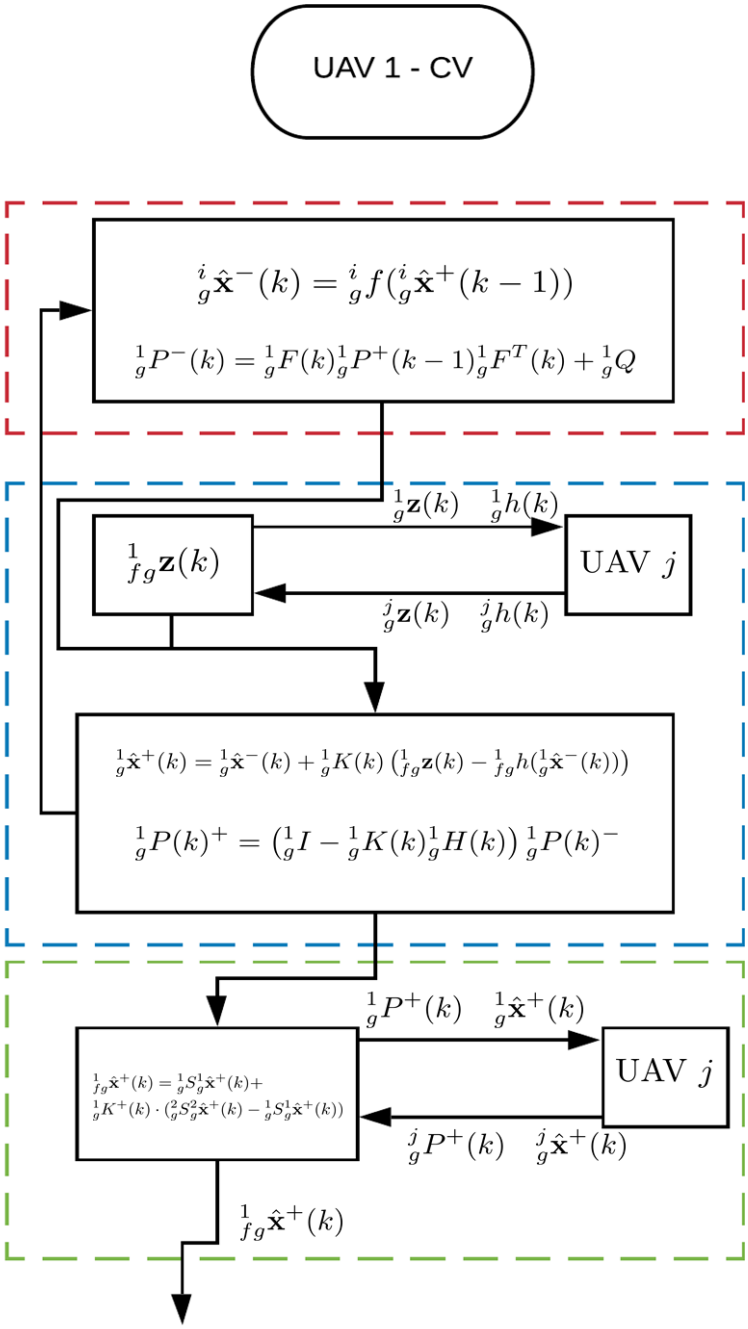


Figure 4.1: Information flow of the multi model dekf with T Test selection. The (-) represents the prediction phase, each UAV using their respective model. The (-) represents the measurement update phase, where each UAV shares their measurements and updates their estimate. Finally, (-) represents the selection phase, through the T Test, where based on the residuals received, a final estimate for the iteration is chosen.

and the innovation covariance

$$ {}^{i}_{g}\Upsilon(k) \triangleq {}^{i}_{g}H(k){}^{i}_{g}P^{-}(k){}^{i}_{g}H(k)^{\mathsf{T}} + {}^{i}_{g}R(k)\,, \tag{4.30} $$

the likelihood function is computed as

$$ {}^{i}_{g}\gamma(k) = \frac{exp\left\{-\frac{1}{2}{}^{i}_{g}\upsilon(k)^{\mathsf{T}}{}^{i}_{g}\Upsilon^{-1}(k){}^{i}_{g}\upsilon(k)\right\}}{(2\pi)^{n/2}\sqrt{{}^{i}_{g}\Upsilon(k)}}\,. \tag{4.31} $$

4.4.1.2 Probability Update

With the likelihood functions from each UAV, the next step is to normalize the values in order to determine which model has the highest probability of being correct for the current timestep. Thus, the model probability is calculated as

$$ {}^{i}_{g}\mu(k|k) = \frac{1}{{}^{i}_{g}c(k)}{}^{i}_{g}\mu(k|k-1){}^{i}_{g}\gamma(k) \tag{4.32} $$

with the normalization factor

$$ {}^{i}_{g}c(k) = \Sigma_{j=1}^{N}{}^{j}_{g}\mu(k|k-1){}^{j}_{g}\mu(k)\,. \tag{4.33} $$

The predicted model probability ${}^{i}_{g}\mu(k|k-1)$ is given by

$$ {}^{i}_{g}\mu(k|k-1) = \Sigma_{j=1}^{N}\rho_{ji}{}^{j}_{g}\mu(k-1|k-1)\,, \tag{4.34} $$

where the entry ρ_{ji} in the matrix ${}^{i}_{g}\rho$ is a coefficient representing the switching probability, modelled as a Markov Process, essentially representing the probability that we will switch models in the next iteration.

4.4.1.3 State Combination

The last process consists of combining the state estimates from the different models according to their computed model probabilities. This is accomplished as

$$ {}^{i}_{fg}\hat{\mathbf{x}}^{+}(k) = \Sigma_{j=1}^{N} {}^{j}_{g}\mu(k|k) {}^{j}_{g}\hat{\mathbf{x}}^{+}(k)\,. \tag{4.35} $$

4.4.1.4 IMM-EKF Information Flow

The flow of information between UAVs for this method can be seen in Figure 4.2. Note that once again each UAV uses its respective a posteriori estimate as its current state estimate in the next time step.

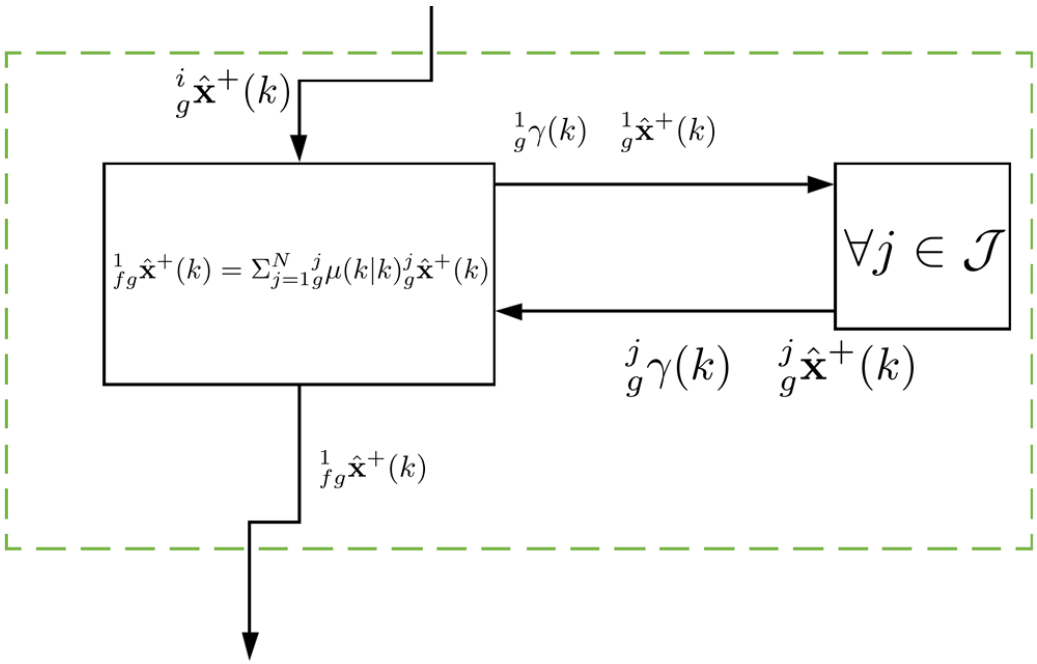


Figure 4.2: Information flow of the IMM DEKF. The (-) and (-) of Figure 4.1 remain unchanged. The (-) represents the weighted phase, where based on the likelihoods received, a final estimate for the iteration is chosen.

4.4.2 Maximum Likelihood Estimation

Another common method commonly used for fusing estimates is what is known as the maximum likelihood estimation, where covariance ellipses are fused together to generate an overall better estimate. The algorithm in this case starts after each UAV has generated their respective estimate (with their respective model), and the estimates are combined along with the covariances. Therefore, models producing better estimates will supposedly have smaller covariances and be trusted more.

4.4.2.1 Formulation

The formulation is taken from [80]. It begins with the calculation of a second Kalman gain, termed K^+. Given the respective covariance matrices of UAVs 1 and 2 as ${}^1_gP^+(k)$ and ${}^2_gP^+(k)$, for the state estimates ${}^1_g\hat{\mathbf{x}}^+(k)$ and ${}^2_g\hat{\mathbf{x}}^+(k)$, this second Kalman gain is calculated as

$$ {}^1_gK^+(k) = {}^1_gS \cdot {}^1_gP^+(k) \cdot \left({}^1_gS^1_gP^+(k) + {}^2_gS^2_gP^+(k)\right)^{-1} . \tag{4.36} $$

The matrices ${}^1_gS$ and ${}^2_gS$ project the appropriate covariance matrix to contain the only common states (i.e. position and velocity). The updated estimate is given as

$$ {}^1_{fg}\hat{\mathbf{x}}^+(k) = {}^1_gS^1_g\hat{\mathbf{x}}^+(k) + {}^1_gK^+(k) \cdot ({}^2_gS^2_g\hat{\mathbf{x}}^+(k) - {}^1_gS^1_g\hat{\mathbf{x}}^+(k)) . \tag{4.37} $$

Note that in the case that multiple estimates are received from multiple UAVs, equations (4.36)-(4.37) are evaluated iteratively for each estimate.

4.4.2.2 Maximum Likelihood Information Flow

The first two processes in Figure 4.1 remain unchanged. However in the weighted process (-) maximum likelihood estimation is performed through a second Kalman gain

calculation. Note once again that the next state used in the DEKF is that made before the maximum likelihood update. The general flow is shown in Figure 4.3.

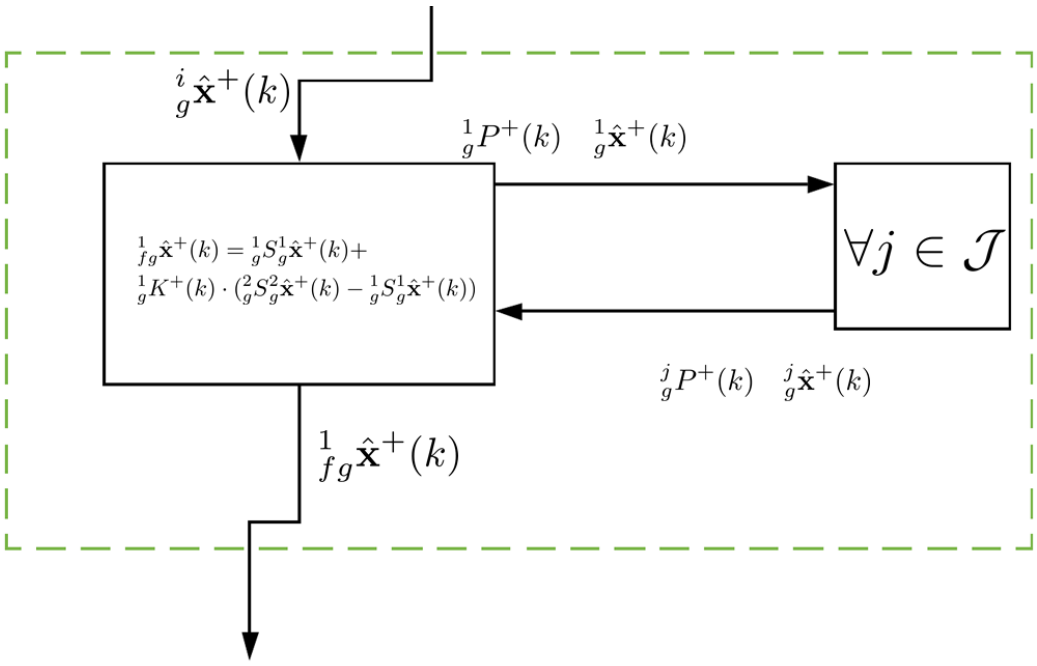


Figure 4.3: Information flow of the maximum likelihood DEKF. The (-) and (-) from Figure 4.1 remain unchanged. The (-) represents the weighted phase, where based on the covariance matrices received, a final estimate for the iteration is chosen.

4.5 Occlusions

An advantage of multi-UAV scenarios is that a UAV need not necessarily rely on its sensors to perform observations. Observations can come from other UAVs within communication range. However, there are still instances when none of the UAVs in the scenario make an observation. In these instances, solely the prediction step in the DEKF is run. Furthermore, in these cases the T Test is invalid because the norm of the residual is not available. Therefore, this thesis explores and contrasts several different selection methods for which model to use for the prediction step in times when no detections are made.

The first selection method is to use the last model at the time of the occlusion. The idea is to keep the estimate with the associated model which was performing best according to

the T Test at the time of the occlusion. This can be formulated as

$$ {}^{i}t_{stat}(k) = {}^{i}t_{stat}(k-1) \quad \forall i \,. \tag{4.38} $$

The second selection method is to use the estimate associated with the model that had the least uncertainty at the time of the occlusion. More precisely, this is achieved by comparing the trace of the covariance matrices in x and y, choosing the model with the lowest trace. This is done through

$$ {}^{i}t_{stat}(k) = tr({}^{i}_{g}S^{i}_{g}P^{+}(k)) \quad \forall i \,, \tag{4.39} $$

and choosing ${}^{i}_{g}\hat{\mathbf{x}}^{+}(k)$ that corresponds to the lowest t_{stat} value. The trace operation is represented by $tr(\cdot)$.

The last selection method tested was computing the eigenvalues of the covariance matrices and choosing the model with the smallest maximum eigenvalue, representing a covariance matrix diverging the slowest, which is the case in times of occlusions. This can be formulated as

$$ {}^{i}t_{stat}(k) = {}^{i}\lambda_{max}(k) \quad \forall i \,, \tag{4.40} $$

where ${}^{i}\lambda_{max}(k)$ is the largest eigenvalue of ${}^{i}_{g}P^{+}(k)$. Once again, the estimate with the smallest t_{stat} value is chosen.

4.6 Results

The results are split into four main sections. The first section aims to show that using a multi model approach improves estimation over using a single model. The second section aims to show that using the T Test as a model selection criteria results is less estimation

error over the other methods presented in this thesis. The third section aims to show that in times of occlusions, choosing to stay with the last identified model leads to the smallest estimate error.

4.6.1 Experimental Set-Up

Indoor tests were conducted in the robotics lab at the Royal Military College of Canada. The lab is equipped with a motion capture system for pose data with millimetre precision. The quadrotors used in the experiments are the AscTec Pelicans, all three of which were hung from the ceiling, in order to decouple the state estimation and control. Furthermore, each quadrotor is equipped with a PTU and an onboard camera, capable of centering the target in the camera frame. The target used is the Turtlebot with an Aruco code (see Figure 3.2) used for identification. Using the Optitrack system, the actual position of the target was compared against the UAV estimates.

4.6.2 Multi-Model vs. Single Model

To compare the performance of the multi model DEKF, the target was made to experience all three of the models in question (CV, CA and CT) for a prolonged period of time. Measurement data of five trials was gathered and both the multi model DEKFs and single model DEKFs were ran using this data. For brevity, we will show the data relevant to one of the trials. The general initialization parameters of the algorithm are presented in Table 4.1. The resultant estimates of the target in the xy plane are shown in Figure 4.4. Figure 4.5 shows the corresponding trial over time. The switching behaviour via the T Test of the multi model DEKF is shown in Figure 4.6.

Table 4.1: Multi Model DEKF Initialization

	UAV 1(CV)	UAV 2 (CA)	UAV 3 (CT)
Rate (hz)	30	30	30
Q	$0.05I_{4\times4}$	$0.05I_{6\times6}$	$0.05I_{5\times5}$
R	$10I_{2\times2}$	$10I_{2\times2}$	$10I_{2\times2}$
P_0	$0.5I_{4\times4}$	$0.5I_{6\times6}$	$0.5I_{5\times5}$
$\hat{\mathbf{x}}_0$	$\begin{bmatrix} h^{-1}({}^g\mathbf{z_0}) \\ 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} h^{-1}({}^g\mathbf{z_0}) \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$	$\begin{bmatrix} h^{-1}({}^g\mathbf{z_0}) \\ 0 \\ 0 \\ 1e^{-5} \end{bmatrix}$
n	10	10	10

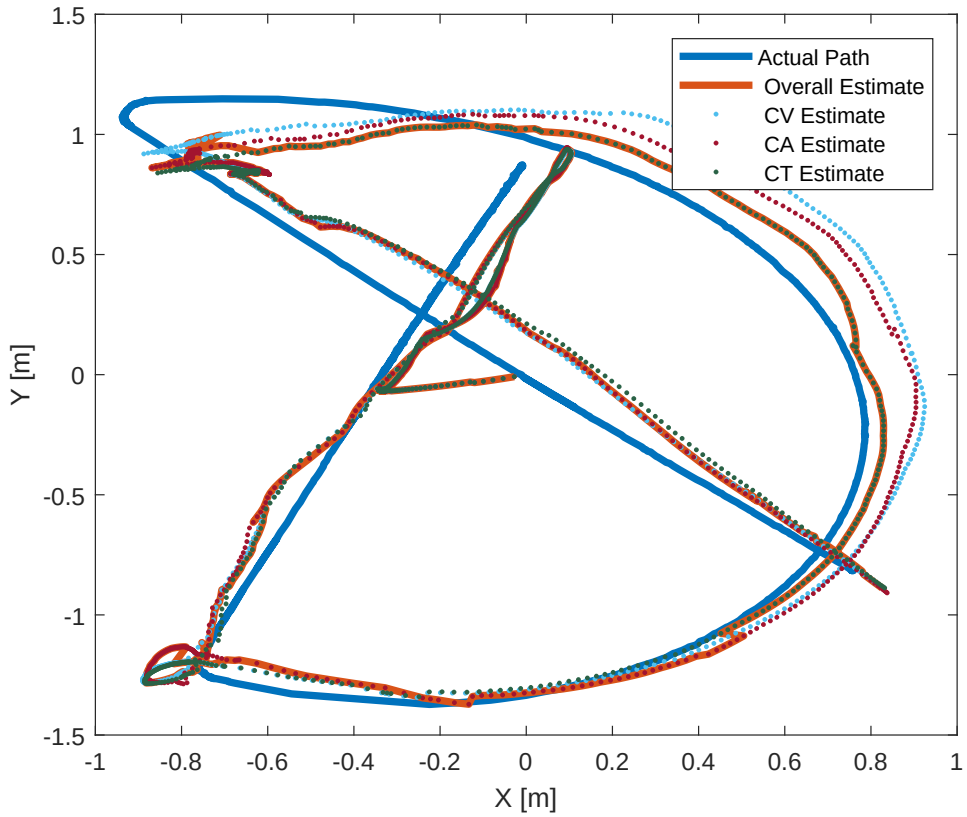
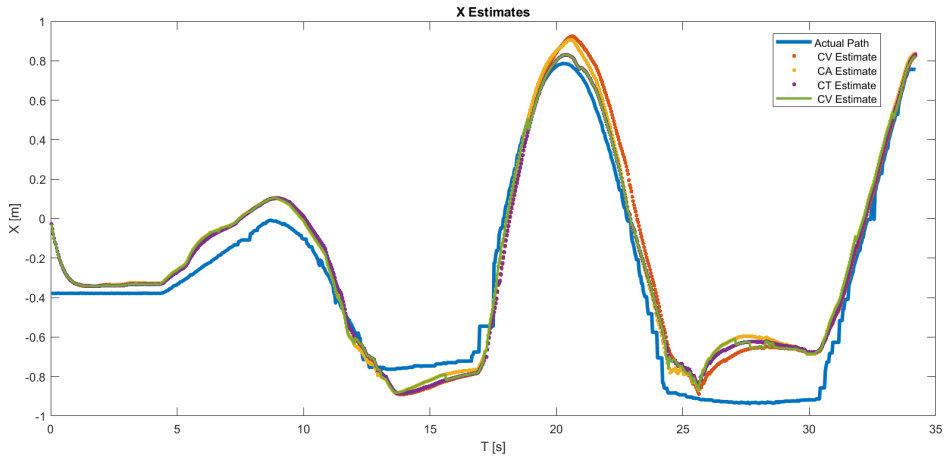
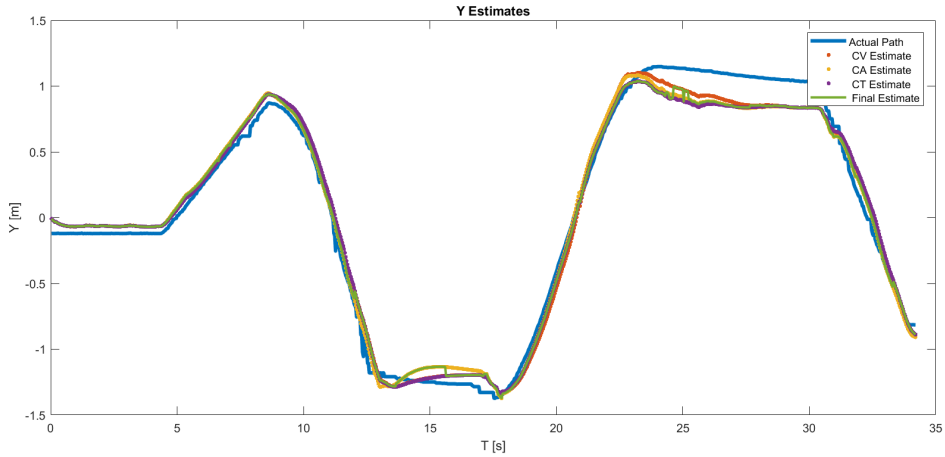


Figure 4.4: Estimates of the multi model DEKF. All UAVs make the same overall estimate of the target, shown as a solid line. Dots represent the estimates made by the individual UAVs.



(a)



(b)

Figure 4.5: x and y estimates of the single model and multi model DEKFs. Solid line is the overall estimate made by each UAV (they all have the same overall estimate), dots represent the estimate made by each UAV, using their respective model.

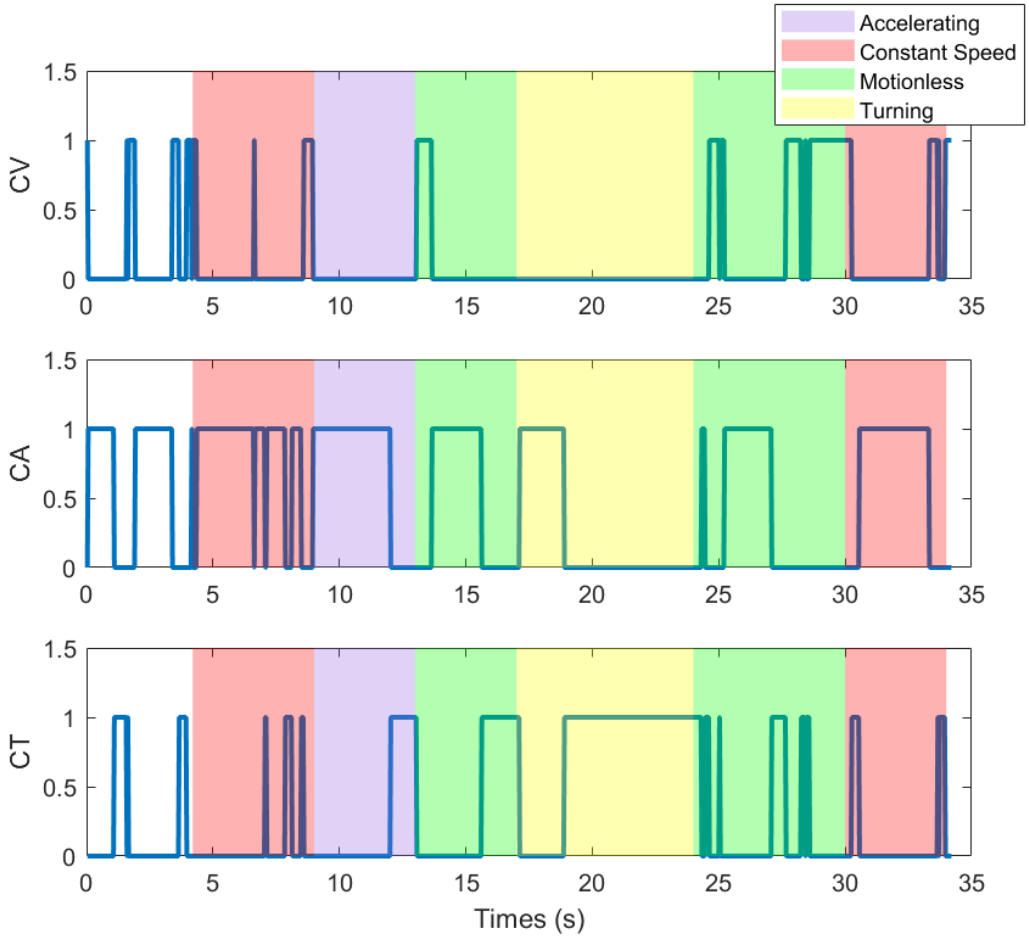


Figure 4.6: T Test results for a target in various states of motion. The first moments of each trial are for initialization and are not considered in the test. The model being trusted by the UAVs is shown by a value of one.

As can be seen in Figure 4.6, the T Test selects the correct model at the correct time. When the target is in a coordinated turn, the multi model DEKF is trusting the CT model estimates over the other two models. Similarly, during purely linear accelerations (purple in the Figure), the CA model is trusted. Furthermore, switches between model occur primarily between changes of motion in the target, which is expected as the multi model DEKF is converging towards the correct model in those times. The only problem arises in distinguishing between the CV and CA models. This could be attributed to the fact that in times of constant speed, acceleration estimates are near zero, in which case the CA model approximates the CV model. To evaluate the performance of the multi model DEKF, the

data was run and compared against when all three UAVs adopted the same model. The error, calculated as the mean error over the five trials, is given by the distance off the estimate is from the Optitrack measurement, and is shown in Table 4.2. To calculate this error, equation (4.41) is used.

$$\text{Mean Error} = \frac{1}{k}\sum_{l=0}^{l=k}\sqrt{({}^{i}_{fg}\hat{x}^{+}(k))^2 + ({}^{i}_{fg}\hat{y}^{+}(k))^2} \tag{4.41}$$

Table 4.2: Mean Estimate Error in Meters.

	CVCVCV	CACACA	CTCTCT	CVCACT
Trial 1	0.169	0.163	0.162	0.153
Trial 2	0.166	0.190	0.175	0.165
Trial 3	0.160	0.166	0.159	0.154
Trial 4	0.259	0.244	0.252	0.241
Trial 5	0.171	0.175	0.169	0.164
Average	0.185	0.187	0.184	0.175

The notation“CVCVCV” indicates that each UAV was using the CV model, whereas “CVCACT” indicates that each UAV used a different model during the test. As is seen in the table, the multi model DEKF experienced less error over the single model equivalents over the trials. Although the improvement is small, it is still significant, as the “CVCACT” algorithm performed best overall and in every individual trial. Furthermore, the fact that the experiments were conducted indoors implies an area for the target to move in of approximately $4\ m^2$ while the UAVs are constrained to a maximum height of 1.5 m. This small scale of the experiment puts limitations on the degree of improvement that can be observed from one algorithm to the next. Consequently, it is believed that outdoor experiments with maneuvering areas and maximum heights several magnitudes larger will induce larger errors and thus larger differences in errors between the different algorithms presented here.

4.6.3 Performance Comparison

The multi model DEKF relies heavily on fusing the estimates in such a manner that emphasis is put on the estimate with the model that best represents the target's motion. In order to do so, two alternatives have been presented here, namely the IMM-EKF and the maximum likelihood methods. It should be noted that these are the first attempts at using these algorithms for multi-UAV scenarios. For this test, similar to Section 4.6.2, 5 trials were run and the algorithms run with the associated measurements from the UAV cameras. However, in this series of tests the target's motion was evasive in nature. The corresponding estimates are shown in Figure 4.7. Furthermore, the relevant initializations are shown in Table 4.3. And the corresponding estimates over time are shown in Figure 4.8. The corresponding T Test results are shown in Figure 4.9.

Table 4.3: Initialization of DEKFs for Various Methods

	Maximum Likelihood	IMM-DEKF	T Test
Rate (hz)	30	30	30
Q	$0.05I$	$0.05I$	$0.05I$
R	$10I$	$10I$	$10I$
P_0	$0.5I$	$0.5I$	$0.5I$
$\hat{\mathbf{x}}_0$	$\begin{bmatrix} h^{-1}({}^g\mathbf{z_0}) \\ \mathbf{0} \end{bmatrix}$	$\begin{bmatrix} h^{-1}({}^g\mathbf{z_0}) \\ \mathbf{0} \end{bmatrix}$	$\begin{bmatrix} h^{-1}({}^g\mathbf{z_0}) \\ \mathbf{0} \end{bmatrix}$
Method Specific	None	$\rho = \begin{bmatrix} 0.99 & 0.01 & 0.009 \\ 0.001 & 0.99 & 0.009 \\ 0.005 & 0.005 & 0.99 \end{bmatrix}$	$n = 10$

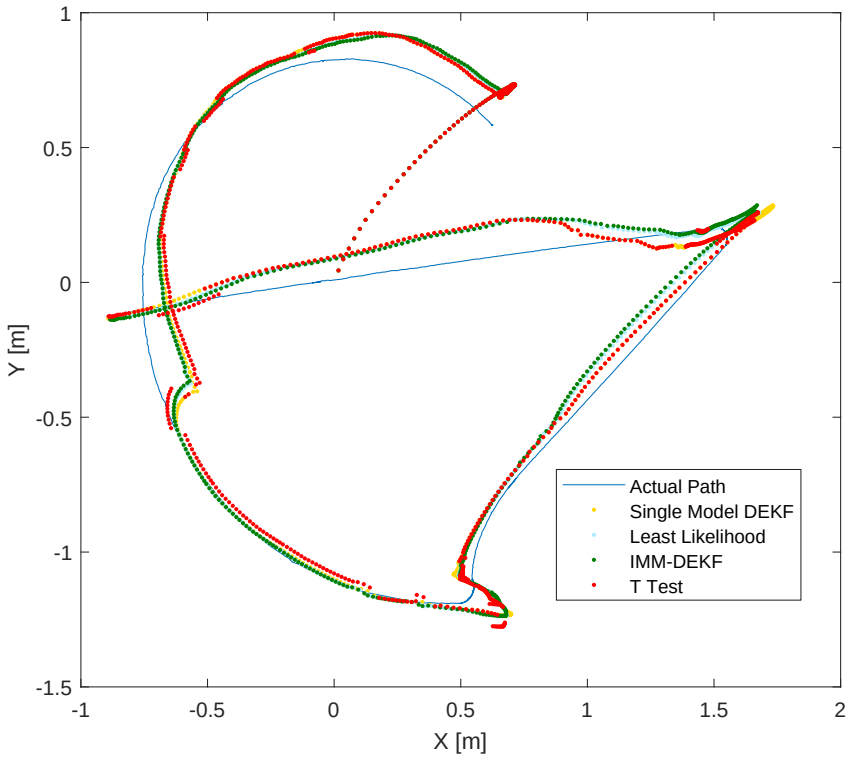
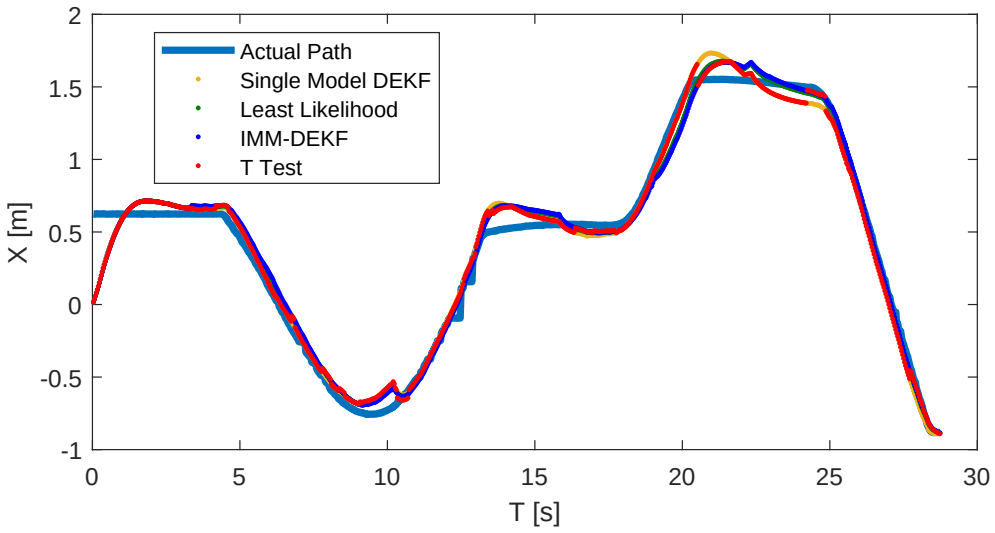
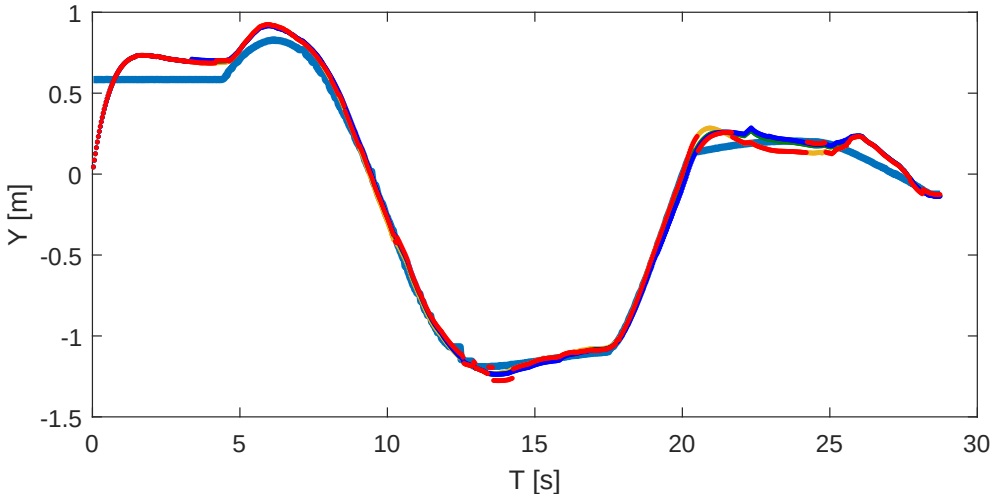


Figure 4.7: Estimates of target position for various methods of estimation.



(a)



(b)

Figure 4.8: Position estimates of the maximum likelihood, IMM-DEKF and T Test methods for fusing estimates. For the single model DEKF, the plot shows the estimates using the CA model.

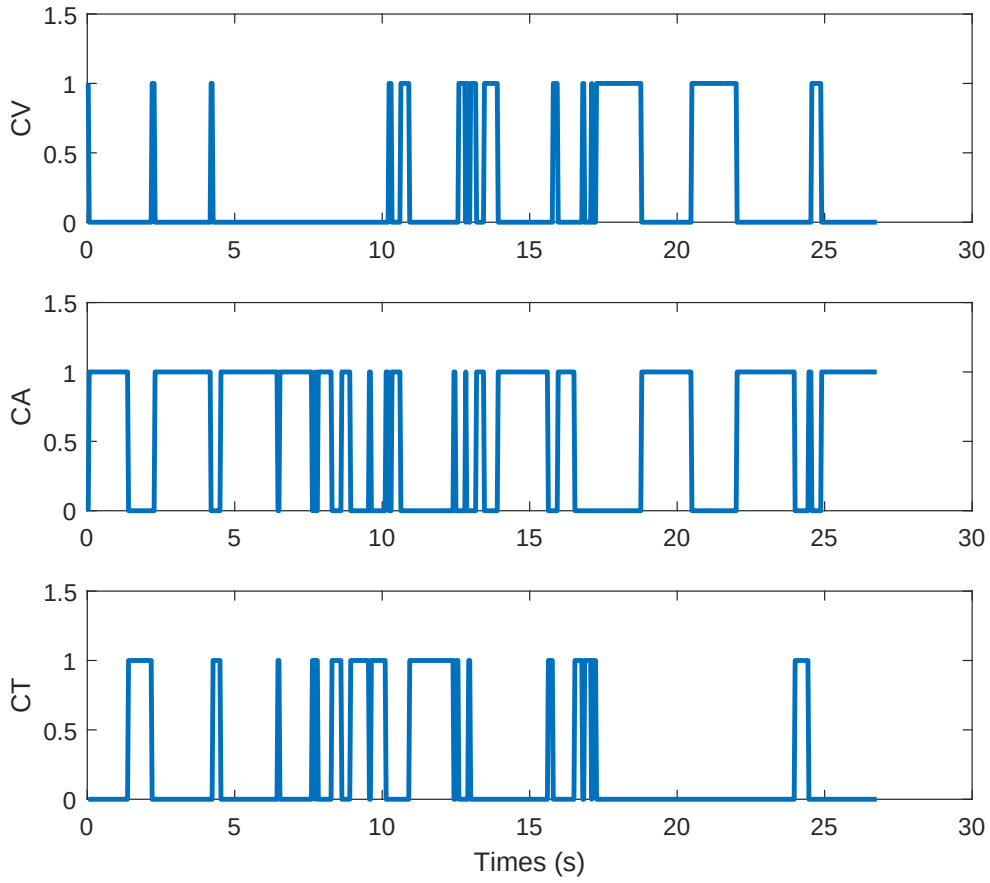


Figure 4.9: T Test model switching over time. The path followed by the target evasive in nature, with multiple changes in its motion over time.

For maximum likelihood estimation, it is relevant to observe the covariance over time (Figure 4.10), which is the primary factor in determining the weight of estimates from different models.

Last, the IMM DEKF model probabilities are shown in Figure 4.11. The values shown in the Figure indicate how much weight is put onto each model when constructing the final estimate.

The error in the estimates, once again calculated as the distance the estimate is from the Optitrack measurements, is shown in Table 4.4.

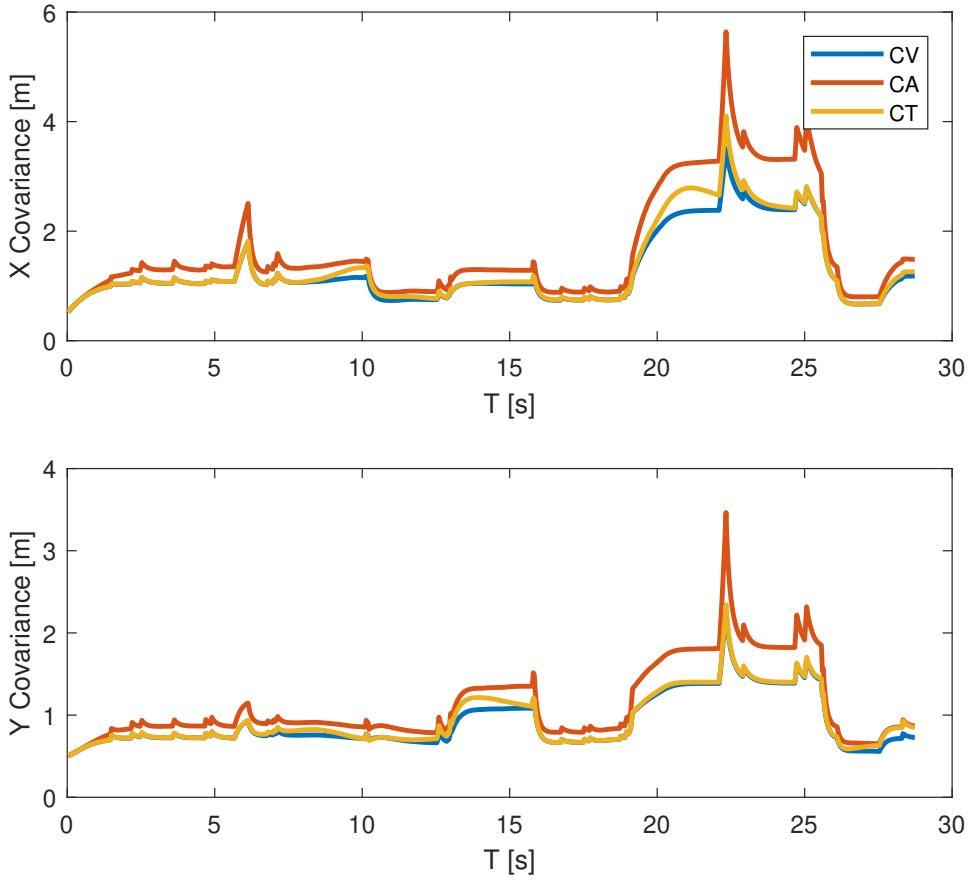


Figure 4.10: Covariances over time of the various models using the maximum likelihood estimation method.

Table 4.4: Mean Estimate Error in meters.

	maximum likelihood	IMM-DEKF	T Test
Trial 1	0.161	0.164	0.157
Trial 2	0.169	0.168	0.158
Trial 3	0.180	0.189	0.171
Trial 4	0.162	0.167	0.157
Trial 5	0.298	0.306	0.270
Average	0.194	0.199	0183

The T Test method does indeed improve the estimation accuracy over the other proposed methods. First, if we examine Figure 4.9, it is seen that that are 36 transitions, showing that the T Test is constantly adapting to what it believes to be the best model.

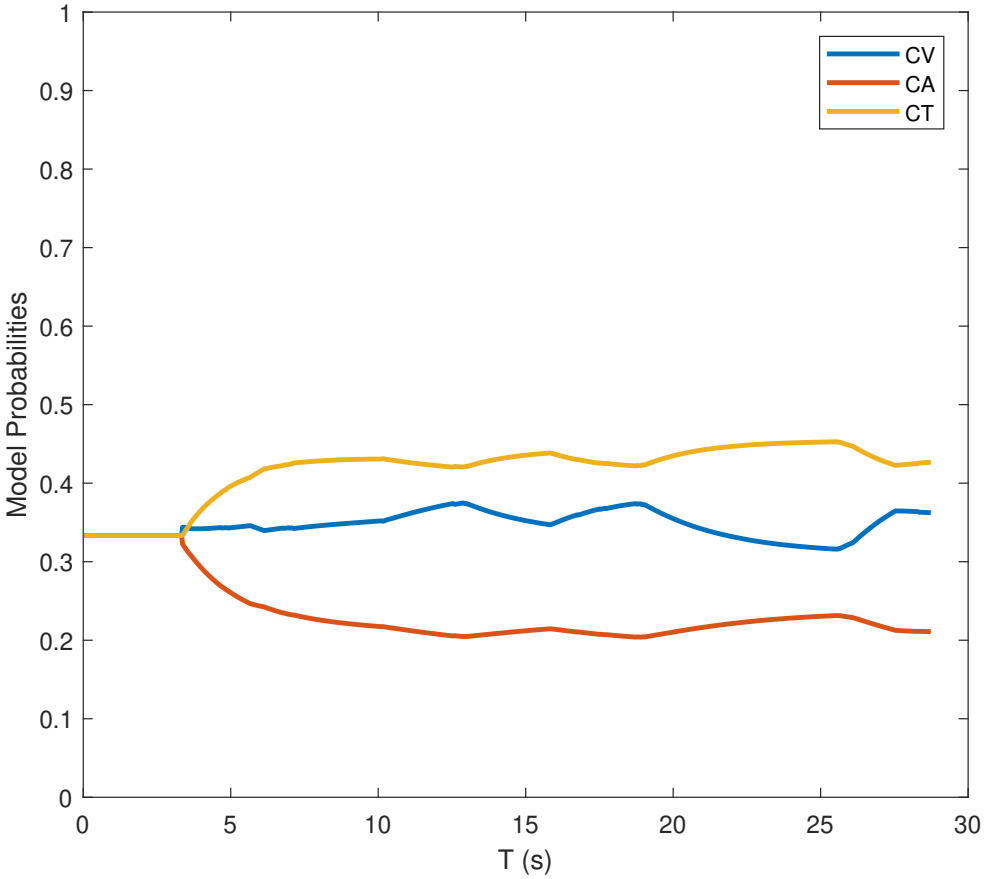


Figure 4.11: Model probabilities over time of the various models using the adpated IMM method. Note that the first 100 samples are used for initialization and for the individual filter to converge before sharing of measurements.

There are advantages and disadvantages to this. The advantage means it can react promptly to a change in target motion. The disadvantage being that the estimates lose smoothness, especially in times when the T Test cannot converge to a model, causing it to switch back and forth between estimates. If we next take a look at Figure 4.10, this is the primary decision factor in weighing the different estimates prior to combining them. The Figure shows that all the covariances move similarly and that in reality they are all close to one another, essentially representing equal weights. Although it does show decent results, there is not enough emphasis on one model at one particular time, which is not desired since this does not represent the optimal estimate made using the optimal model at that time but rather a

weighted average of the different model estimates. Whereas, the T Test will always come to a decision as to which model it believes is performing best, maximum likelihood estimation averages the values of the three models, providing good results over the entire run, but rarely the best result possible at a particular iteration. For the IMM technique the relevant figure is Figure 4.11, where the weights of each model over time are shown. Once again, there are changes in model weights (especially at the beginning while the algorithm converges), but the changes over time are not particularly significant. Recall that these weights are determined primarily from the likelihood function, which in turn depends heavily on the process noise covariance matrix Q. This means that if not accurately initialized, as is the case in practice, it will be harder for the algorithm to converge and react to changes in the likelihood function.

4.6.4 Occlusions

In Section 4.6.2 and Section 4.6.3, the paths followed by the target had no instances where none of the three UAVs were detecting the target. The purpose of this section is to show that keeping the last model chosen by the T Test provides the least amount of estimate error over the other selection methods presented in Section 4.5, that is, a comparison of the trace of the covariance matrix and a comparison of the smallest eigenvalue.

For this particular set of experiments, the target drove a path where for an extended period of time, the target was not detectable by any of the UAVs. Furthermore, the target's motion was similar to Section 4.6.2, where motion was predetermined to follow a specific model for a designated time interval. The initialization is the same as in Table 4.1. The results over 5 trials are shown in Table 4.5.

Table 4.5: Mean Estimate Error in Meters.

	Last Model	Slowest Eigenvalue	Trace of Covariance
Average (During Occlusions)	0.740	0.765	0.765
Average (Entire Path)	0.280	0.281	0.281

The results show that for the most part the three selection methods perform similarly, especially the slowest eigenvalue selection method and the trace of the covariance selection method. Keeping the last model showed the best results over periods of full occlusions. In the end, however, this resulted in only a minor improvement over the entire path.

4.7 Closing Remarks

The Multi Model Distributed EKF is a novel approach to the target tracking problem in general. It links two previously unlinked domains in target tracking. The first is traditional single UAV multi model approaches, where the UAV has banks of filters and all computation is done by the UAV. The second is traditional multi UAV estimation, namely distributed EKFs, where the UAVs share their measurements. Coupled with the T Test selection criteria, the approach presented here distributes the multi model approach, which is traditionally done all on a single UAV, over the various UAVs, having each UAV perform the necessary computation for a single model. At the same time, the UAVs are capable of combining their own target measurements with any measurements received from neighbouring UAVs.

There were three main stages of validation presented here. First, we showed that for a team of three UAVs, the multi model DEKF using the CV, CA and CT models had better estimation over the single model equivalents. Second, the T Test method was shown to best fuse the estimates from different models and UAVs. Third, it was shown that trusting the last model chosen in a time of occlusion will lead to slightly less estimate error. These series of tests show that the Distributed Multi Model approaches, with the appropriate selection criteria, are a valid approach to the multi UAV state estimation. Furthermore, since the models are distributed over the team of UAVs, the computational and memory overhead is minimal as compared to the traditional DEKF. Next steps would be to perform tests where the control is not decoupled from the estimation. Namely, having the target follow a path that forces the UAVs to follow in some capacity, representing a more realistic

tracking scenario.

Chapter 5

Multiple Model Distributed MPC

5.1 Opening Remarks

In this chapter, a DMPC controller is presented for formation flight, using a convex objective, a linear target model and a linear UAV model. The goal of the controller is such that based on which type of motion the target is following, the UAV will choose a model that best fits this motion to propagate down the horizon. The idea is that having a better model for the target will result in a smaller tracking error for the team of UAVs. However, it is shown through a real-time simulation that there is no significant improvement, as compared to the single model equivalent. This MPC controller is then tested and validated in indoor flights. Finally, the controller is adapted to handle the multi-UAV case for formation flight and tested in a real-time simulation. Furthermore, the effects of occlusions on the DMPC controller are investigated.

The remainder of the report is divided as follows: Related Works in Section 5.2, Multi Model Distributed Model Predictive Controller in Section 5.3, Results in Section 5.4, Closing Remarks in Section 5.5.

5.2 Related Works

This section presents the current literature on cooperative control for target tracking UAVs. First, a brief overview of non-optimization based methods is presented. Then, a more in depth summary of current optimization based methods is done.

5.2.1 Non-Optimization Based Cooperative Control

Continuing on their work in [81], Ma and Hovakimyan published [9], a paper that outlined a distributed control strategy for formation flying over a moving ground target. In this implementation, the UAVs have the same linear velocity and maintain a circular formation with the target at the center, through the derivation of a control law. This control is done via the yaw rate of the UAVs and the proposed law consists of two main components. That is, a term for tracking the target and a term for maintaining separation. The tracking term, which has been adapted from [81] to enable a time-varying formation radius, depends on several parameters, including the target and UAV dynamics and desired formation radius. The separation term is given explicitly as well, except for three separate intra-vehicle communication topologies: all-to-all, ring and cyclic pursuit. However, despite the changes in topology, the control law reduces to be proportional to a subset of the UAVs' (depending on the topology) bearings, essentially creating a repellent force between UAVs. The algorithm is tested in simulation for a target with a constant velocity and shown to track the target while at the same time maintaining the formation, which itself is varying.

Another common method employed for general tracking problems is pursuer-evader game theory based approaches. In [82] for example, multi-pursuer single evader distributed control is developed. The evader is assumed to follow closed loop Nash strategies with the Ricatti differential equations approach. The pursuers attempt to follow Nash strategies as well, however since the cooperation between UAVs is distributed, the pursuers are only able to sense within a certain radius and thus do not always have access to required information for guaranteeing a Nash Equilibrium. To overcome this issue, best achievable performance indices are computed through the minimization of a multi-objective function including dependencies on both the pursuer and evader gain matrices, input matrices and the solution to the Ricatti differential equations. The algorithm is tested with three pursuers and an evader in simulation, where the pursuers capture in all tested scenarios.

In [83], a tracking algorithm consisting of coordinated vision based tracking is im-

plemented and tested. Assumptions include constant speed UAVs and that the difference between the commanded and actual yaw rates is bounded. The UAVs attempt to maintain a circular formation around a moving ground vehicle by sharing of only the Line Of Sight (*LOS*) angle between neighbouring UAVs, hence a distributed approach. Furthermore, the solution is tested against communication failures between UAVs. Based on these assumptions, a control law is established for the yaw rate of the vehicles. The simulation is tested with 6 UAVs and a target moving in a straight line. The target is tracked and the formation is maintained.

Cyclic pursuit is another method used for coordinated target tracking and a variant of it has been implemented in [84]. In cyclic pursuit, agents follow other specific agents to attain certain behaviours, such as encirclement in this case. The cyclic pursuit law in this case is modified for target tracking in a distributed cooperation scheme. The agents however are modelled simply as a unicycle. Two sensing topologies are implemented, one with information from a single neighbour and the other is an all-to-all communication scheme. These methods are tested in simulation. Once validated, the author discusses the implementation onto a MAV network, where using hardware in the loop the algorithm is tested on aerial video footage for a stationary target.

As a lot of current methods being employed for multiple UAV target tracking revolve around MPC and optimal control, investigating this area further will identify the current areas of cooperative UAV target tracking that necessitate more research.

5.2.2 Optimization Based Approaches

This sections gives an overview of current optimization based methods being employed in the literature, as they pertain to cooperative UAV target tracking.

5.2.2.1 Recursive Methods

These methods incorporate some form of recursion in their cost function in order to be able to modify it in real-time. The first such implementation to be presented in this thesis is [85], where a swarm of UAVs equipped with Received Signal Strength (*RSS*) sensors are tasked with tracking a moving aerial Radio Frequency (*RF*) source in a decentralized manner. The state estimation of the target is done using an EKF. The target UAV dynamics are modelled as a LTI system, as

$$ {}_g\mathbf{x}(k) = {}_gA_g\mathbf{x}(k-1)\,. \tag{5.1} $$

The states are defined as the position and linear velocities in the x and y directions of the target in a 2D inertial frame. Furthermore, all UAVs are assumed to have the same altitude, including the target. The path planning algorithm functions by use of a Receding Horizon Control (*RHC*) which optimizes about the change in headings the collective group of UAVs need to make in order to track the target over the horizon. The cost function consists of a submatrix of the Fisher Information Matrix (*FIM*), obtained by using the simplifications presented in [86] and which depends on the target dynamics, measurement noise, process noise and measurement error. The minimization problem takes the form of equation (2.40). The overall FIM is constructed from the Joint Likelihood Function (*JLF*) between target measurements and states. The function $\mathfrak{J}(\mathbf{u}(k))$ is found recursively by the

following formula

$$\begin{aligned}
\mathfrak{J}(k+1) &= D^{22}(k) - D^{21}(k) \cdot (\mathfrak{J}(k) + D^{11}(k))^{-1} D^{12}(k) \\
D^{11}(k) &= A^T Q^{-1} A \\
D^{12}(k) &= (D^{21})^T = -A^T Q^{-1} \\
D^{22}(k) &= Q^{-1} + E[G(k+1) R^{-1} G^T(k+1)] \\
G(k) &= \nabla_{{}_g\mathbf{x}(k)} \cdot h^T({}_g\mathbf{x}(k))
\end{aligned} \tag{5.2}$$

where Q and R are the covariance matrices associated with the process and measurement noise respectively. The measurement function is $h(\cdot)$, relating the states to sensor measurements. $E[\cdot]$ is the expectation function. The algorithm is validated only in simulation in two dimensions. Three UAVs track a target UAV and results show that tracking performance is improved with prediction.

In an extension from their work in [87], Hausman published an optimization-based control strategy developed and implemented for tracking a ground target with multiple coordinated UAVs [88]. The UAVs use a camera and the data is processed using OpenCV, where the relevant measurements are used in an EKF for state estimation. The motion model of the target is a constant velocity model. For control, the author forms a cost function that essentially minimizes the uncertainty of the joint target position estimate. It does so by determining the optimal control inputs for the UAVs through the trace of the combined target position covariance over a finite horizon, ${}_gP(k+h)$, which is determined from an EKF. An additional term is also added that acts as a means for collision avoidance. In this case, equation (2.40) can be rewritten as

$$\mathbf{u}^\star(\mathbf{k}) = \arg\min_{\mathbf{u}} \left[\mathfrak{J}(\mathbf{u}(k)) + \mathfrak{c}_a(\mathbf{u}(k))\right] \tag{5.3}$$

where $\mathfrak{c}_a$ is a cost function that accounts for collision avoidance. The cost function $\mathfrak{J}$ is

given by

$$\mathfrak{J}(\mathbf{u}(k)) = \sum_{l=k}^{k+h} \gamma^l \text{tr}\left[{}_gP(l)\right] \tag{5.4}$$

where $0 \leq \gamma \leq 1$ is a discount factor. The overall implementation is tested both in simulation and in a real indoor implementation with multiple quadrotors. In the indoor simulation however, due to proximity of drones from one another, major disturbances from wind currents cause erroneous measurements for certain topologies, especially when one UAV is above the other. The target being tracked was a TurtleBot 2.

5.2.2.2 MPC based methods

Quintero is a researcher who is at the forefront of MPC for UAVs and has been investigating standoff target tracking, which is where the vehicles attempt to maintain a certain distance and orientation with respect to the target. In [89], the implementation and simulation of a distributed cooperative MPC controller for target tracking is conducted. The implementation has the UAVs fly at constant altitude and velocity while tracking a slower target. Moreover, the author has taken into account the possibility of wind, represented as a constant disturbance in the formulation. The dynamics have migrated to a more realistic 4^{th} order system and consequently been discretized using second-order Lie series approximation for the planar position of the UAV and a small angle approximation for $\tan\theta$. The target behaves as a double integrator in simulation. Therefore, it can move evasively as well as accelerate to some bounded velocity. Furthermore, the algorithm models the target dynamics as a LTI system

$${}_g\mathbf{x}(k) = {}_gA_g\mathbf{x}(k-1) + {}_gB_g\mathbf{w}(k)\,. \tag{5.5}$$

An overall discrete state space is then formed by combining the two UAV dynamics with the target dynamics.

The optimality criterion of the robust MPC with Moving Horizon Estimation (*MHE*) has a cost function in the form of a min max optimization as

$$\begin{aligned}\mathfrak{J}({}_c\mathbf{x}(k),\ \mathbf{u}(k),\ {}_g\mathbf{u}(k), {}_c\mathbf{z}(k)) = &\sum_{l=k}^{k+h} \mathfrak{g}({}_c\mathbf{x}(k)) + \sum_{l=k}^{k+h-1} \gamma_u ||\mathbf{u}(k) - \theta(k)||^2 \\ &- \sum_{l=k-L}^{k+h-1} \gamma_d ||{}_g\mathbf{u}(k)||^2 - \sum_{l=k-L}^{h} \gamma_n ||{}_c\mathbf{z}(k) - H_c\mathbf{x}(k)||^2 \end{aligned} \quad (5.6)$$

where the missing notation is displayed in Table 5.1. The idea of a min max optimization is to find the most efficient inputs (i.e. minimization) when the estimates of the states and the effects due to disturbances are at their predicted worse (i.e. maximization). From equation

Table 5.1: MPC/MHE Notation

Notation	Description
${}_c\mathbf{x}$	Combined state vector with both UAV and target states
$\mathbf{u}$	Input roll of UAVs
${}_g\mathbf{u}$	Unknown control sequence of target
${}_c\mathbf{z}$	Measurement vector of UAV and target states
H	Measurement matrix
γ_u , γ_d , γ_n	Positive scalar discount factors
L	Moving horizon used to look back at previous measurements
θ	Roll angle of tracking UAVs
$\mathfrak{g}({}_c\mathbf{x}(k))$	Two-player zero-sum game function

(5.6), we can see the cost of the input error, target movement and measurement error are given by the second, third and fourth terms of the cost function, respectively. The first term has $\mathfrak{g}({}_c\mathbf{x}(k))$ as

$$\mathfrak{g}({}_c\mathbf{x}(k)) = \gamma_1 \frac{({}^1_g)^2({}^2_g r)^2}{({}^1_g r)^2 + ({}^2_g r)^2} + \gamma_2(({}^1_g r)^2 + ({}^2_g r)^2) \quad (5.7)$$

where the first part of the addition is a scalar representing the fused error covariance of the

distance between the UAVs and the target, using positive discount factor γ_1, and where ${}^{i}_{g}r$ represents the distance between UAV i and the target. The second term ensures that UAVs are close enough that the target is detectable by computer vision.

This cost function is then integrated into a min max optimization problem, where the goal is to minimize the objective under worse case conditions of the unknowns, like the target inputs for example. More precisely,

$$\mathbf{u}^{\star} = \min_{\mathbf{u}} \max_{{}_{c}\hat{\mathbf{x}}, {}_{g}\hat{\mathbf{u}}, {}^{w}\hat{\mathbf{v}}} \mathfrak{J}({}_{c}\hat{\mathbf{x}}(k),\, \mathbf{u}(k),\, {}_{c}\hat{\mathbf{u}}(k), {}_{c}\mathbf{z}(k),\, t) \tag{5.8}$$

where recall $\hat{(\cdot)}$ represents the estimate of $(\cdot)$ and the term ${}^{w}\mathbf{v}$ is used to denote the disturbance due to wind present. The algorithm is capable of being run online. Simulations were run for constant velocity and evasive targets.

In a work by Yao, [90], an algorithm developed in [91], a previous work for a single UAV performing standoff tracking was enhanced to encompass cooperation between multiple UAVs in a distributed manner. Using similar assumptions as in [91], three simulated UAVs attempt to track a target in a dynamic environment. The overall cost function for N UAVs takes a centralized approach, as

$$\mathfrak{J}(k) = \sum_{\zeta=1}^{N} {}^{\zeta}\mathfrak{J}(k) \quad \forall \zeta \in \mathcal{Z}\,, \tag{5.9}$$

which is simply the sum of the individual UAV costs. The set $\mathcal{Z}$ with elements ζ represents the set of all UAVs in the implementation. Thus, ${}^{\zeta}\mathfrak{J}(k)$ is expressed as

$$\gamma_T \mathfrak{T}(k) + \gamma_A \mathfrak{A}(k) + \gamma_S \mathfrak{S}(k) \tag{5.10}$$

where the sum of the discount factors γ_T, γ_A and γ_S is one. $\mathfrak{T}(k)$ in equation (5.10) repre-

sents the cost of target tracking and is given by a sum of inner products as follows

$$\mathfrak{T}(k) = \sum_{l=k}^{k+h-1} \mathbf{v}(k)^{\mathsf{T}} \mathbf{v}'(k) \tag{5.11}$$

where $\mathbf{v}$ represents the velocity of a UAV without obstacles and $\mathbf{v}'$ is that with obstacles. The value of $\mathbf{v}$ is determined using a Lyapunov Guidance Vector Field (*LGVF*) method [92], and is given by

$$\mathbf{v}(k) = \delta \mathbf{v}''(k) + {}_g\mathbf{v}(k) \tag{5.12}$$

where $\mathbf{v}''$ denotes the calculated velocity from LGVF and ${}_g\mathbf{v}$ the target velocity. δ is termed the velocity correction coefficient. The velocity after incorporating obstacle avoidance is calculated using a Interfered Fluid Dynamical System (*IFDS*) approach.

The second term in equation (5.10) evaluates the cost of how threatening surrounding obstacles are, and is out of the scope of this text.

Finally, the last term in equation (5.10) puts a cost on the smoothness of the planned path and is calculated by

$$\mathfrak{S}(k) = \frac{1}{h} \sum_{l=k}^{k+h-1} (\gamma_\psi \cdot |\psi(l+1) - \psi(l)| + \gamma_\phi \cdot |\phi(l+1) - \phi(l)|) \tag{5.13}$$

where the UAV heading is denoted as ψ and the pitch of the vehicle as ϕ. The terms γ_ψ and γ_ϕ are discount factors.

The UAVs attempt to maintain a circular formation above the target with equal angular separation. The final path of each UAV is generated by first addressing the formation. That is to say, the formation is controlled by the horizontal velocities of the individual UAVs. To calculate the desired speed, a Lyapunov function based on the desired phase angles (denoted ${}^{\zeta}\omega_\beta$ for UAV ζ) of the respective UAVs is designed. The desired speed of UAV ζ

is the given according to

$$^{\zeta}v = r^{\zeta}\omega_{\beta} \forall \zeta \in \mathcal{Z} \tag{5.14}$$

where r is the radius of the circle followed. Once the horizontal speeds have been collaboratively determined, the separate UAV velocities are calculated based on the LGVF methods in [91], using the respective speeds from equation (5.14). From there, these individual velocities are perturbed first by a term for attraction between UAVs to maintain sensing range then by a final term for obstacle avoidance. This final velocity is used for the path generation. The implementation is tested in simulation with obstacles and an evasive target, using three UAVs. The results show that the UAVs successfully track the target while avoiding obstacles.

A decentralized Learning Based Model Predictive Control (*LBMPC*) used in formation flight with N cooperative UAVs can be found in [93]. The dynamic model used for the ζ^{th} UAV can be written as

$$^{\zeta}\dot{\mathbf{x}}(t) = {}^{\zeta}A^{\zeta}\mathbf{x}(t) + {}^{\zeta}B^{\zeta}\mathbf{u}(t) + g({}^{\zeta}\mathbf{x}(t), {}^{\zeta}\mathbf{u}(t)) \quad \forall \zeta \in \mathcal{Z}\,, \tag{5.15}$$

where $g({}^{\zeta}\mathbf{x}(t), {}^{\zeta}\mathbf{u}(t))$ represents the unmodeled dynamics. It itself is calculated from an oracle which contains the learning parameters in a linear state space form. Furthermore, the communication topology includes only UAV i's neighbours. The MPC optimization is convex and for UAV i takes the form of

$$\begin{aligned} {}^{i}\mathfrak{J}\big({}^{i}\mathbf{u}(t), \mathbf{u}_r(t), {}^{i}\mathbf{x}(t), {}^{j}\mathbf{x}(t), \mathbf{x}_r(t)\big) = & \sum_{l=t}^{t+h-1} \left[{}^{i}\mathbf{x}(l) - {}^{j}\hat{\mathbf{x}}(l) - \mathbf{x}_r(l)\right]^{\mathsf{T}} Q \left[{}^{i}\mathbf{x}(l) - {}^{j}\hat{\mathbf{x}}(l) - \mathbf{x}_r(l)\right] \\ & + \sum_{l=t}^{t+h-1} \left[{}^{i}\mathbf{u}(l) - \mathbf{u}_r(l)\right]^{\mathsf{T}} R \left[{}^{i}\mathbf{u}(l) - \mathbf{u}_r(l)\right] \\ & + \mathfrak{c}_e \end{aligned} \tag{5.16}$$

where Table 5.2 below explains relevant notation. Note for equation (5.16), we have $i \in \mathcal{I}$ for the UAV in question and $j \in \mathcal{J}$ for the set of neighbour UAVs to i. Also since the formulation is continuous, t represents time. In equation (5.16), the cost associated with

Table 5.2: LBMPC Notation

Notation	Description
${}^{j}\hat{\mathbf{x}}$	Estimated states of i's neighbours
$\mathbf{x}_r$, $\mathbf{u}_r$	Reference steady state and input to reach said state, respectively
Q,R	Corresponding weight matrices
$\mathfrak{c}_e$	Cost of the final state error

following the reference is given by the first term and the cost of the inputs is given by the second term. The cost associated with $\mathfrak{c}_e$ is determined by

$$\left[{}^{i}\mathbf{x}(h) - {}^{j}\hat{\mathbf{x}}(h) - \mathbf{x}_r(h)\right]^{\mathsf{T}} Q_h \left[{}^{i}\mathbf{x}(h) - {}^{j}\hat{\mathbf{x}}(h) - \mathbf{x}_r(h)\right] \tag{5.17}$$

where Q_h is the weight of reaching the final desired state over the horizon. Once the optimal control input is solved for UAV i, it is fed through a feedback gain which is chosen to satisfy the discrete-time Ricatti equation. The algorithm is validated in simulation with a team of three UAVs tracking a target moving in a straight line. The UAV formation is such that they lie on a line with the center UAV 10 m in from the outer UAVs.

5.2.2.3 Final Remarks

This survey is by no means an exhaustive list of the current research on single-target cooperative UAV tracking. It does however expand in detail on some of the more common approaches used in today's research. In terms of sensing, some used a form of RF sensors, but most of the papers assumed a vision camera. For the state estimation, a variant of the EKF was definitely the most popular method employed. Of the techniques presented in the survey for cooperative target tracking, that is vector field approaches, flocking, formation

flight, cyclic pursuit, pursuer-evader game theory, standoff tracking and optimisation based, the latter, more specifically MPC, is a popular choice.

In optimisation and MPC there exist several different methods for forming the objective to minimize. Aspects such as the model chosen and the type of optimization all have an impact on the overall implementation. When looking at optimization approaches, it seems the major trade-off in the research presented was the aspect of accuracy for computation time. A more complex model or cost function may yield better results but may not be feasible on an actual vehicle. Furthermore, the solutions presented here have not been shown to be very robust, as the majority have never been implemented in a real time system of UAVs. Considering areas in the field that need to be expanded on, most of the control strategies have been developed for fixed wing UAVs and not VTOL. Moreover, the majority of targets being considered are non-evasive and can be modelled quite easily, saving on computation. However in a real life scenario this would not always be the case and more complex target models would have to be adopted.

5.3 Multi-Model Model Predictive Controller

5.3.1 Formulation

From 2.6.1, the general problem is posed as

$$\mathbf{u}^{\star}(k) = \arg\min_{\mathbf{u}} \mathfrak{J} \tag{5.18}$$

where

$$\mathbf{u} \triangleq \begin{bmatrix} \mathbf{u}(k) & \mathbf{u}(k+1) & \cdots & \mathbf{u}(k+h) \end{bmatrix}^T \tag{5.19}$$

In this particular problem, the UAVs are equipped with a PTU capable of pitching and rolling in order to keep the target centered in the camera frame to ensure adequate detection. Furthermore, as the UAVs in question are VTOL, they also possess the capacity to yaw in

the direction of the target, such that the camera is always facing it, and hover, should the target be stationary. These target measurements are then assumed to be fused through some form of state estimation algorithm, which outputs an overall estimate and an indication of which model best describes the target's motion at that point in time, as does the algorithm described in Chapter 4. It is also assumed that UAVs can communicate over a specified range their respective state estimates of the target and their position. Finally, the heights of the UAVs above the ground are assumed constant. Furthermore, this UAV coordination and control problem aims to specify at each instant the individual UAV positions in x and y that will best satisfy $\mathfrak{J}$, using the principle of tracking through formation flight, where each UAV will attempt to remain in a specific spot from the target. One further constraint on the implementation is that it must be run on board the UAV in real-time. As optimization can be quite computationally intensive, the cost function remains convex to ensure optimal convergence in a timely manner. The MPC law is first derived for a single UAV without cooperation. Then, in the following sections it is expanded so that UAVs must maintain a specified separation from each other.

5.3.2 The Cost Function $\mathfrak{J}$

The optimization problem posed in equation (5.18) can take many forms. To keep the problem convex, $\mathfrak{J}$ for the i^{th} UAV will take the following form

$$\begin{aligned} {}^{i}\mathbf{u}^{\star}(k) = &\arg\min_{{}^{i}\mathbf{u}} {}^{i}\mathfrak{J}({}^{i}\mathbf{u}(k)) \\ &\text{subject to} \quad {}^{i}\boldsymbol{x}(k) = {}^{i}A^{i}\boldsymbol{x}(k-1) + {}^{i}B^{i}\boldsymbol{u}(k-1) \\ &\boldsymbol{u}_{min} \leq {}^{i}\mathbf{u} \leq \mathbf{u}_{max} \\ &{}^{i}\boldsymbol{x}(0) = {}^{i}\hat{\boldsymbol{x}}^{+}(0) \end{aligned} \tag{5.20}$$

with

$$ {}^{i}\mathfrak{J}({}^{i}\mathbf{u}(k)) = \sum_{l=k}^{k+h} [({}^{i}\mathbf{x}(l) - {}^{i}\mathbf{x}_r(l))^{T\,i}Q({}^{i}\mathbf{x}(l) - {}^{i}\mathbf{x}_r(l)) + {}^{i}\mathbf{u}(l)^{T\,i}R^{i}\mathbf{u}(l)] \tag{5.21} $$

where u_{min} and u_{max} form the range of allowable inputs. Matrices A and B are the linear model, determined through system identification, of UAV i. They have input ${}^{i}\mathbf{u}$, which is also the minimizer of the objective function in equation (5.18). Therefore, ${}^{i}\mathbf{u}$ takes the form $\begin{bmatrix} x & y \end{bmatrix}^{\mathsf{T}}$, which are the inputs to the high level position controller for individual UAVs.

As previously mentioned, the premise behind this controller is that of tracking a target in a formation. To accomplish this, the position relative to the target each UAV should maintain is incorporated in the reference trajectory ${}^{i}\mathbf{x}_r(k)$. Recall that the state of UAV i over the horizon is ${}^{i}\mathbf{x}(l)$. To formulate ${}^{i}\mathbf{x}_r(l)$, information about the current target state estimate is used as

$$ \begin{aligned} {}^{i}\mathbf{x}_r(l) &= S_g^i \hat{\mathbf{x}}(l) + {}^{i}\mathbf{r}_r \\ \Rightarrow \begin{bmatrix} {}^{i}x_r \\ {}^{i}y_r \end{bmatrix} &= \begin{bmatrix} {}^{i}_{g}\hat{x} \\ {}^{i}_{g}\hat{y} \end{bmatrix} + \begin{bmatrix} {}^{i}x_r \\ {}^{i}y_r \end{bmatrix} . \end{aligned} \tag{5.22} $$

In equation (5.22), the relative position the UAV should maintain from the target is $\mathbf{r}_r$, specified by x_r and y_r. S projects the target's state vector ${}^{i}_{g}\mathbf{x}(k)$ onto the first two states, and depends on the current model being used. This information is depicted in Figure 5.1 for a team of three UAVs. The target's estimate ${}_{g}\hat{\mathbf{x}}(l)$ is found by propagating the current state estimate, using a priori knowledge on which model is currently performing best, as

$$ {}^{i}_{g}\hat{\mathbf{x}}(l) = {}^{i}_{g}A^{*\,i}_{g}\hat{\mathbf{x}}(l)\,. \tag{5.23} $$

The optimal model representing the target's motion best and the current estimate ${}^{i}_{g}\hat{\mathbf{x}}^{+}(k)$ of the target are chosen at time $l = k$. The model ${}^{i}_{g}A^{\star}$ takes the form of either the CV,

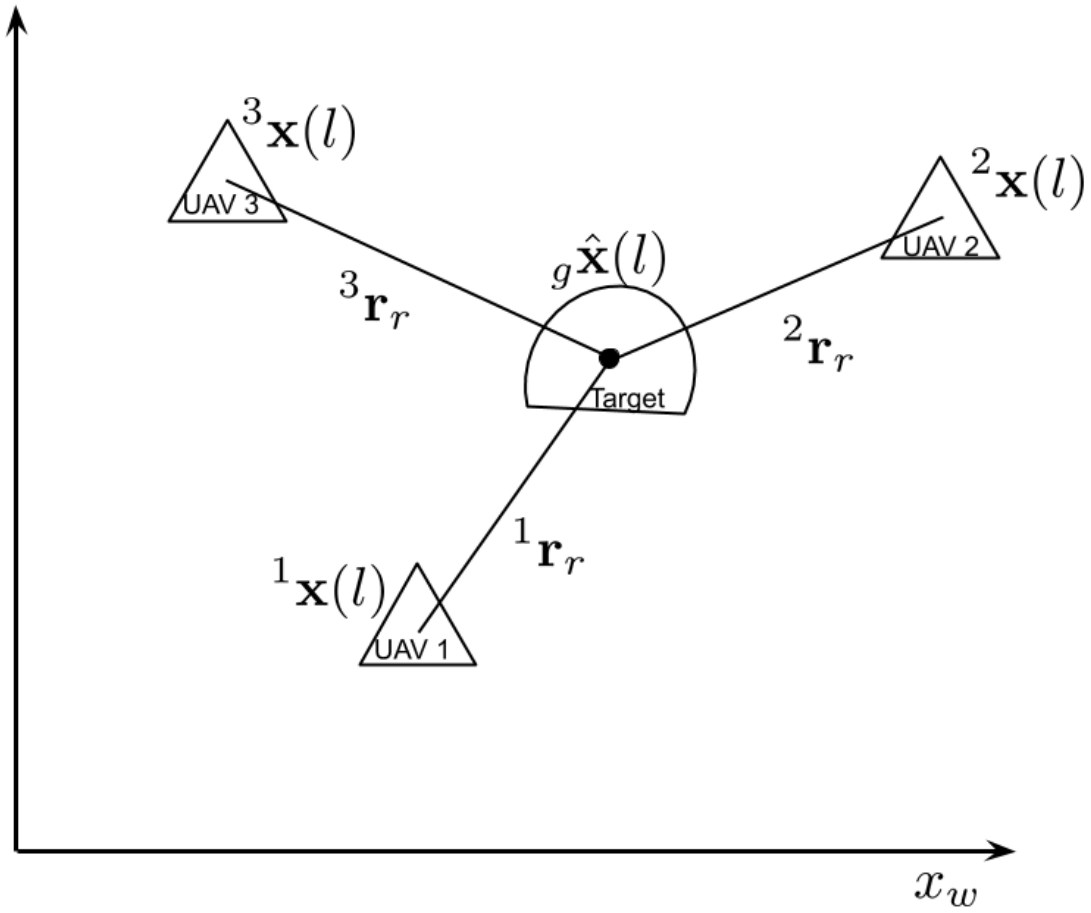


Figure 5.1: Each UAV attempts to maintain its own relative position to the target. By doing so, the UAVs are in formation, as specified by ${}^i\mathbf{r}_r$.

CA or CT models, as presented in Chapter 2, based on the current result of the T Test (See Chapter 4).

5.3.3 **Multi UAV Distributed MPC**

Up to now, the MPC has been formulated for a single UAV to maintain a relative position to the target, as specified by ${}^i\mathbf{r}_r$. To incorporate all UAVs in $\mathcal{Z}$ into the formation, each is given their own respective relative position to follow ${}^\zeta\mathbf{r}_r$, as shown in Figure 5.1. Thus, the formation can take any predefined shape by knowing the desired position each UAV should maintain relative to the target.

The UAVs share their current position and velocity estimate with each other as ${}^{i}\mathbf{x}(k)$. With this information the relative position at time k is calculated as

$$ {}^{ij}\mathbf{r}(k) = S({}^{i}\mathbf{x}(k) - {}^{j}\mathbf{x}(k)) \tag{5.24} $$

where S projects onto the first two coordinates (i.e. x and y). The sequence of states over the horizon for UAV i is calculated directly in the MPC as ${}^{i}\mathbf{x}(l)$, with initial state ${}^{i}\mathbf{x}(k)$ at time $l = k$. In order for UAV i to propagate the state of UAV j over the horizon, it uses its own identified model and UAV j's initial state at time k, ${}^{j}\mathbf{x}(k)$. Thus, the predicted states over the horizon of UAV j, as computed by UAV i, are

$$ {}^{j}\mathbf{x}(l) = {}^{i}A^{j}\mathbf{x}(l-1)\,. \tag{5.25} $$

Similarly, the relative position over the horizon is given by

$$ {}^{ij}\mathbf{r}(l) = S({}^{i}\mathbf{x}(l) - {}^{j}\mathbf{x}(l))\,. \tag{5.26} $$

For cooperation, there are two main goals to achieve. First the UAVs should attempt to maintain their formation, which is achieved through the formulation in Section 5.3.2. The second goal is to avoid collisions between UAVs, that may occur in the case of an evasive target. This is the main formulation of this section. In order to avoid collisions, another quadratic term is added to the overall cost function as

$$ ({}^{i}\mathbf{x}(l) - {}^{ij}\mathbf{x}_s(l))^{T\,ij}Q_s({}^{i}\mathbf{x}(l) - {}^{ij}\mathbf{x}_s(l)) \quad \forall i \in \mathcal{I}\,, \forall j \in \mathcal{J} \tag{5.27} $$

where ${}^{ij}Q_s$ is a weighting matrix similar to Q and R from Section 5.3.2. The term ${}^{ij}\mathbf{x}_s(k)$ represents the separation distance in x and y UAV i should attempt to maintain to avoid a collision with UAV j. Therefore, the term in equation (5.27) is added dynamically to the

overall cost function for each neighbouring UAV $j \in \mathcal{J}$. The separation term is given as follows

$$
\begin{aligned}
{}^{ij}\mathbf{x}_s(k) &= {}^{ij}\mathbf{r}(k) + {}^{ij}\mathbf{r}_s(k) \\
\Rightarrow \begin{bmatrix} {}^{ij}x_s \\ {}^{ij}y_s \end{bmatrix} &= \begin{bmatrix} {}^{i}x \\ {}^{i}y \end{bmatrix} - \begin{bmatrix} {}^{j}x \\ {}^{j}y \end{bmatrix} + \begin{bmatrix} {}^{ij}x_s \\ {}^{ij}y_s \end{bmatrix}
\end{aligned} \tag{5.28}
$$

where

$$
{}^{ij}\mathbf{r}_s(k) = \begin{cases} \begin{bmatrix} 0 & {}^{ij}r_s \end{bmatrix}^{\mathsf{T}}, & \text{if} \quad {}^{i}x - {}^{j}x \geq 0 \wedge {}^{i}y - {}^{j}y \geq 0 \\ \begin{bmatrix} -{}^{ij}r_s & 0 \end{bmatrix}^{\mathsf{T}}, & \text{if} \quad {}^{i}x - {}^{j}x < 0 \wedge {}^{i}y - {}^{j}y \geq 0 \\ \begin{bmatrix} 0 & -{}^{ij}r_s \end{bmatrix}^{\mathsf{T}}, & \text{if} \quad {}^{i}x - {}^{j}x < 0 \wedge {}^{i}y - {}^{j}y < 0 \\ \begin{bmatrix} {}^{ij}r_s & 0 \end{bmatrix}^{\mathsf{T}}, & \text{if} \quad {}^{i}x - {}^{j}x \geq 0 \wedge {}^{i}y - {}^{j}y < 0 \end{cases} . \tag{5.29}
$$

Essentially, when the UAVs are within the minimum separation distance of each other the separation term is added, which guides UAV i around UAV j to the right by following the points defined by ${}^{ij}\mathbf{r}_s(k)$, as shown in Figure 5.2.

5.3.4 Overall Multi Model DMPC

With the definition of the cost function ${}^{i}\mathfrak{J}(\mathbf{u}(k))$ and with the inter UAV separation term defined, the final multi model DMPC for UAV i can be formulated as

$$
\begin{aligned}
{}^{i}\mathbf{u}^{\star}(k) = \arg\min_{{}^{i}\mathbf{u}} \; & {}^{i}\mathfrak{J}({}^{i}\mathbf{u}(k)) \\
& \text{subject to} \quad {}^{i}\boldsymbol{x}(k) = {}^{i}A\,{}^{i}\boldsymbol{x}(k-1) + {}^{i}B\,{}^{i}\boldsymbol{u}(k-1) \\
& \boldsymbol{u}_{min} \leq {}^{i}\mathbf{u} \leq \mathbf{u}_{max} \\
& {}^{i}\boldsymbol{x}(0) = {}^{i}\hat{\boldsymbol{x}}^{+}(0)
\end{aligned} \tag{5.30}
$$

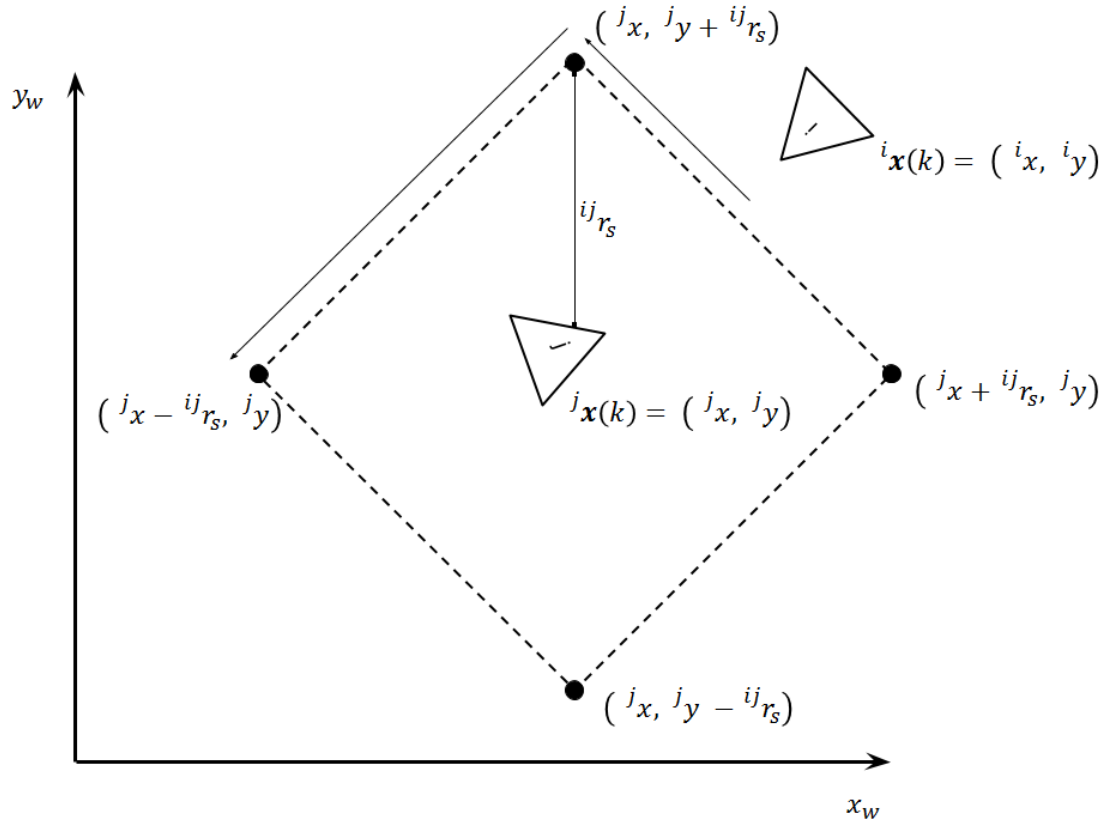


Figure 5.2: In the Figure, UAV i avoids UAV j. In order to accomplish this, once UAV i enters UAV j's separation zone, a separation term is added to UAV i's objective function, altering the overall reference trajectory to avoid UAV j.

with

$$
\begin{aligned}
{}^{i}\mathfrak{J}({}^{i}\mathbf{u}(k)) = \sum_{l=k}^{k+h} [({}^{i}\mathbf{x}(l) - {}^{i}\mathbf{x}_r(l))^{Ti} Q ({}^{i}\mathbf{x}(l) - {}^{i}\mathbf{x}_r(l)) + {}^{i}\mathbf{u}(l)^{Ti} R^{i}\mathbf{u}(l) \\
+ ({}^{i}\mathbf{x}(l) - {}^{ij}\mathbf{x}_s(l))^{Tij} Q_s ({}^{i}\mathbf{x}(l) - {}^{ij}\mathbf{x}_s(l))] \quad \forall i \in \mathcal{I}\,, \forall j \in \mathcal{J}
\end{aligned}
\tag{5.31}
$$

5.4 Results

5.4.1 System Identification

The first step in the MPC design is to choose a model for both the target and the UAVs. In the case of the target, it has already been established that multiple models (CV, CA and

CT) will be used in conjunction to propagate the target states over the horizon. For the UAVs however, a second order system was identified for both the simulation environment and for the AscTec Pelican, using the system identification toolbox in MATLAB .

5.4.1.1 In Simulation

For the VREP simulation, from a hovering position, the underlying position controller of the UAV was given a step input in both x and y. The response of the UAV was measured and the following second order linear model was identified

$$ {}^{i}\mathbf{x}(k) = {}^{i}A\,{}^{i}\mathbf{x}(k-1) + {}^{i}B\mathbf{u}(k)\,, \tag{5.32} $$

where

$$ {}^{i}A = \begin{bmatrix} 0.920 & 0 & 0.084 & 0 \\ 0 & 0.920 & 0 & 0.084 \\ -0.228 & 0 & 0.905 & 0 \\ 0 & -0.228 & 0 & 0.905 \end{bmatrix} {}^{i}B = \begin{bmatrix} 0.0006019 & 0 \\ 0 & 0.0006019 \\ 0.02375 & 0 \\ 0 & 0.02375 \end{bmatrix} \quad {}^{i}\mathbf{x} = \begin{bmatrix} {}^{i}x \\ {}^{i}y \\ {}^{i}v_x \\ {}^{i}v_y \end{bmatrix} . $$

The fit of the model is shown in Figure 5.3. As can be seen, the model approximates the UAV well during its major displacement, but has a higher overshoot than the UAV actually experienced. Furthermore, the steady states are slightly different. This model was then used in the MPC controller and the step response is shown in Figure 5.4. Note that the MPC only begins at five seconds, where beforehand only the position controller of the UAV is functioning. The fluctuations in x and y positions before five seconds arise as the UAV is climbing to a height where it will hover before the MPC is activated, and therefore maintaining its position is more difficult. The MPC controller has some overshoot and a little steady state error. The respective values of ${}^{\zeta}Q$ and ${}^{\zeta}R$ were tuned to 12 and 0.15, for all ζ in $\mathcal{Z}$ respectively.

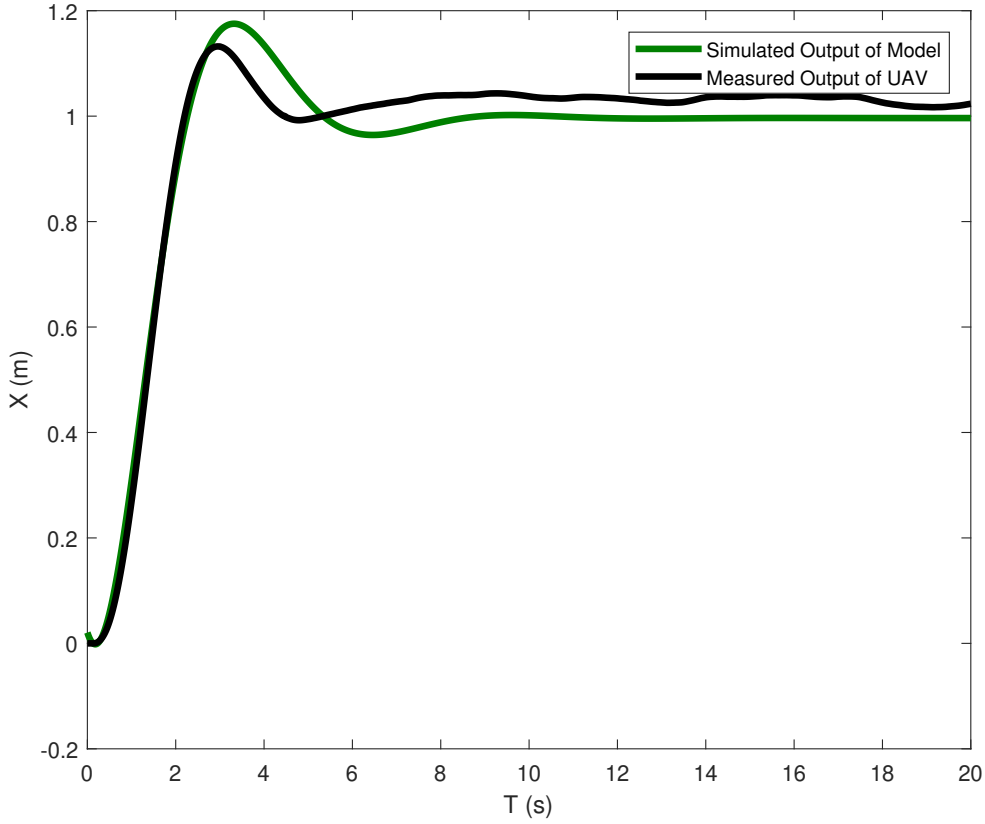


Figure 5.3: A comparison between the actual flight path of the UAV and how well the model resembles this response.

5.4.1.2 AscTec Pelican

For the AscTec Pelican, a similar process was carried out, where the Pelican would reach a desired height and once hovering, a step input in x or y, was given and the outputs recorded. Five trials were done in each x and y, then the system identification tool was used to identify a second order model based on that trial's data. The model identified from each run was validated against each of the other trials, and a performance metric known as the percentage fit was calculated to determine which model performed the best. Table 5.3 shows that the model identified in trial 2 performed best over the five trials. For the UAV's

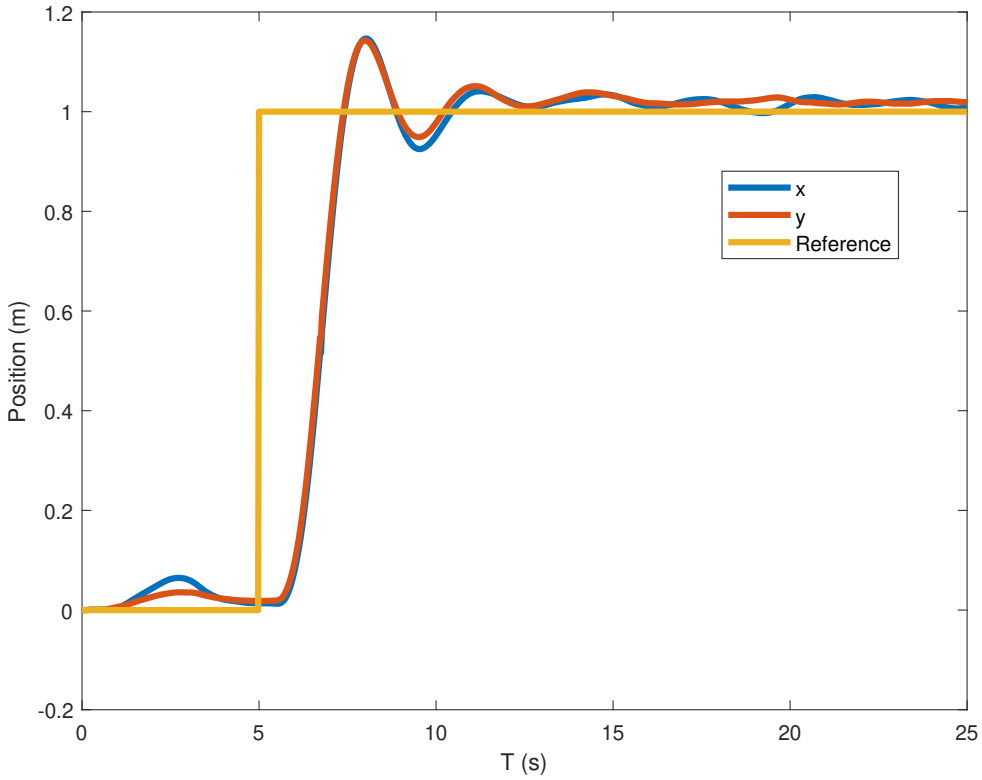


Figure 5.4: The step response of the MPC controller in both x and y. The instants before the step input are the quadrotor trying to maintain its position while it climbs to altitude.

position $x(k)$, and a trial ending at time k_f, the fit percentage is calculated as

$$^{i}FIT = 100 \cdot \sum_{l=0}^{l=k_f} \frac{1 - ||^{i}x(l) - {}^{i}\hat{x}||}{||^{i}x(l) - {}^{i}\bar{x}||} \tag{5.33}$$

where $\bar{(\cdot)}$ represents the mean of $(\cdot)$ over the trial. The fit percentage of a model is a representation of how well the identified model matches the actual flight data collected. A higher fit percentage indicates that the identified model matches the vehicle motion better than a model with a lower fit percentage would. Thus, the model identified in Trial 2 had the best overall average fit against the data from various trials. For example, in Figure 5.5, the models identified in each trial are validated against the flight path of trial 2.

The same procedure was carried out in y and the relevant data is shown in Table 5.4,

Table 5.3: Best Fit Percentages of Pelican Identified Models in X

	Model Identified in Trial				
Trial Data	1	2	3	4	5
1	75	80	76	74	77
2	83	86	83	70	82
3	82	87	85	64	81
4	57	57	53	76	56
5	73	78	75	62	77
Average	74	78	74	69	75

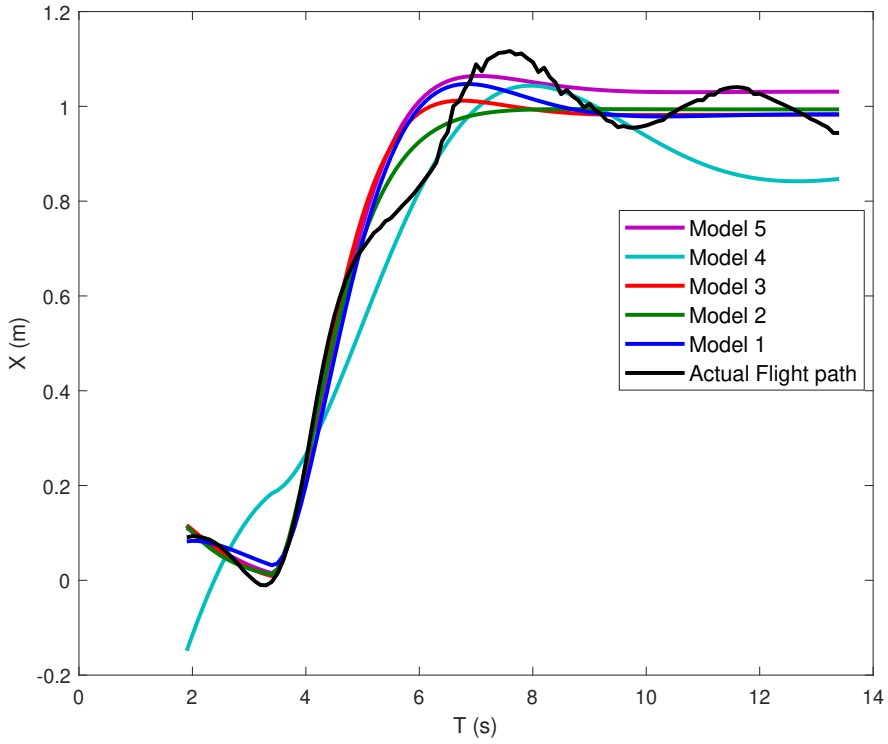


Figure 5.5: The simulated output of each model is compared against the flight data for trial 2. For example, Model 1 refers to the second order linear model identified using trial 1 data.

with $y(k)$, $\bar{y}$ and $\hat{y}$ replacing the corresponding x data in equation (5.33). The best performing model was that identified in trial 4. Combining the x and y models into one overall state space representation, the identified model has

Table 5.4: Best Fit Percentages of Pelican Identified Models in Y

	Model Identified in Trial				
Trial Data	1	2	3	4	5
1	88	83	80	89	72
2	81	84	80	81	70
3	81	84	83	80	71
4	80	82	74	80	74
5	82	71	70	83	76
Average	82.57	81	77	82.68	73

$$ {}^{i}A = \begin{bmatrix} 0.945 & 0 & 0.052 & 0 \\ 0 & 0.943 & 0 & 0.052 \\ -0.151 & 0 & 0.799 & 0 \\ 0 & -0.152 & 0 & 0.853 \end{bmatrix} {}^{i}B = \begin{bmatrix} 0.00268 & 0 \\ 0 & 0.00235 \\ 0.123 & 0 \\ 0 & 0.111 \end{bmatrix} {}^{i}\mathbf{x} = \begin{bmatrix} {}^{i}x \\ {}^{i}y \\ {}^{i}v_x \\ {}^{i}v_y \end{bmatrix}. $$

5.4.2 Multiple Model Comparison

The main goal of the single UAV MPC tests in simulation is to verify whether or not using multiple models for target state propagation has an effect on the tracking performance. For the simulation, a target was made to enter various states of motion. First, it moved at 0.5 m/s in x for 20 seconds. Next, a left turn was made with a turn rate of 0.15 rad/s for 20 seconds. Then, the target accelerated in both x and y at $0.05\ m/s^2$, once again for 20 seconds. Finally, the target remained at rest for the remainder of the simulation. To reduce potential sources of error, the UAV was given perfect estimation of both its own states and the target states, regardless of which model it adopted. Furthermore, the UAV was made to attempt to stay over the target, with ${}^{i}\mathbf{r}_r = \begin{bmatrix} 0 & 0 \end{bmatrix}^{\mathsf{T}}$. The initializations for the simulation are shown in Table 5.5.

The correspoding xy trajectories of the target and UAV using the multi model MPC

Table 5.5: Parameter Initialization of Single UAV MPC in Simulation

Parameter	Value
${}^{i}h$	5
${}^{i}T$	0.2 s
${}^{i}Q$	12
${}^{i}R$	0.15
Input Constraint	$-10 \leq {}^{i}\mathbf{u}(k) \leq 10$
${}^{i}\mathbf{r}_r$	$\begin{bmatrix}0 & 0\end{bmatrix}^{\mathsf{T}}$
Initial UAV position	$\begin{bmatrix}{}^{i}x & {}^{i}y\end{bmatrix}^{\mathsf{T}} = \begin{bmatrix}0 & 0\end{bmatrix}^{\mathsf{T}}$
Initial Target position	$\begin{bmatrix}{}_{g}x & {}_{g}y\end{bmatrix}^{\mathsf{T}} = \begin{bmatrix}1 & 1\end{bmatrix}^{\mathsf{T}}$

approach are shown in Figure 5.6. The control inputs for the same trial are shown in Figure 5.7.

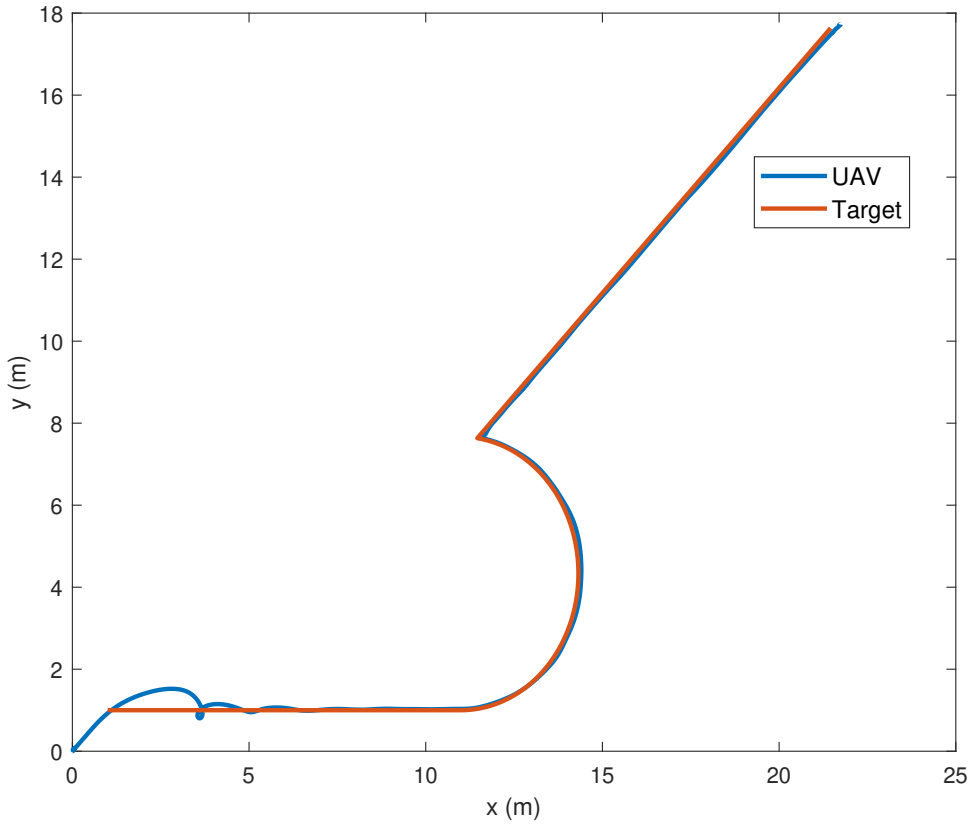


Figure 5.6: Tracking of a mobile ground target using multiple model MPC, in VREP simulation environment for one trial.

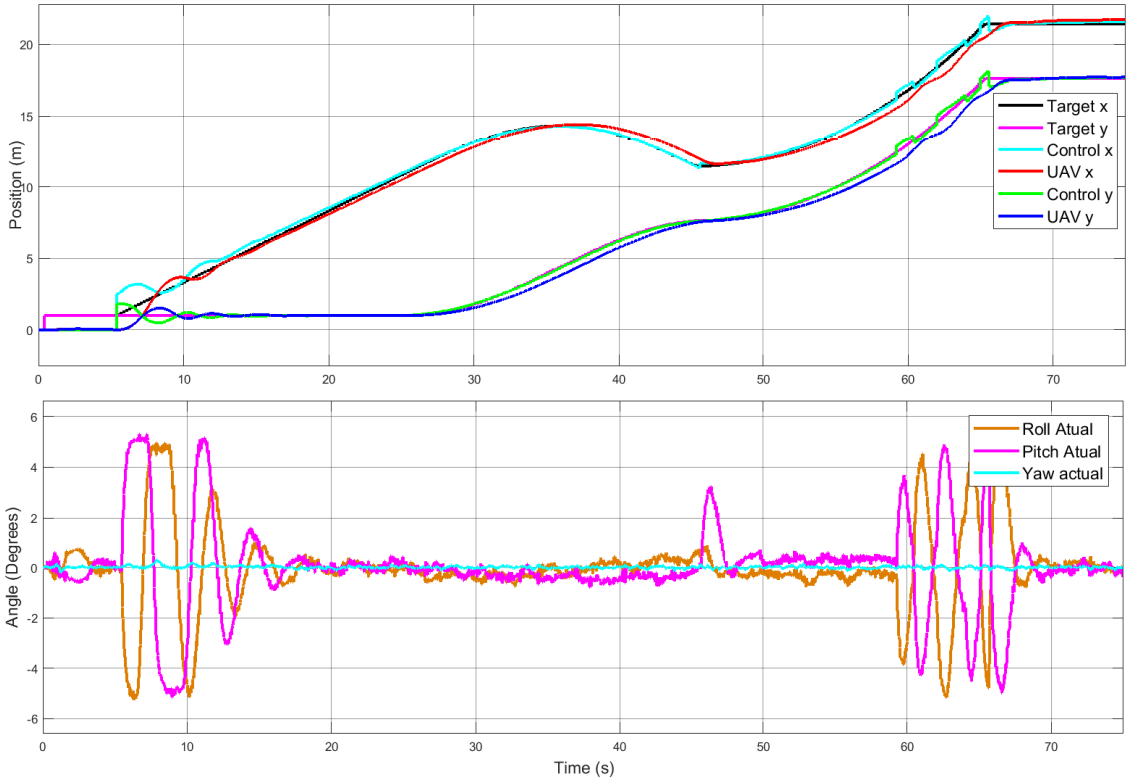


Figure 5.7: In the top figure, the resultant MPC inputs and UAV position. In the bottom figure, lower level angle inputs resulting from MPC generated high level control.

From the figures, it is seen that the UAV accurately tracks the target through the various changes in states of motion. Furthermore, the simulation shows that the low level inputs are controlled as well, except for some saturation in the first few seconds. Saturation of the RPY angles is set at 5°. For the single model comparison, the CV model was used, as it is the most commonly used model in the literature. Both the multiple model MPC and single model MPC went through five trials each. The resultant distance error is shown in Table 5.6. The results show that there is no significant change in tracking performance between the two approaches. Although it is theorized that the multiple model approach should reduce tracking error over a single model approach, this was not the case in simulation. One possible reason could be due the length of the prediction horizon of five, might be too short for their to be any significant deviations between the predictions from different models. Furthermore, the control horizon of DMPC is one, meaning only the first input

is applied to control a UAV. It is not likely that there are any major deviations in the first input between different models. Therefore, more testing should be done in the future with a longer prediction horizon and control horizon.

Table 5.6: Mean Estimate Error in Meters.

	Single Model	Multiple Model
Trial 1	0.28	0.29
Trial 2	0.28	0.29
Trial 3	0.27	0.28
Trial 4	0.30	0.27
Trial 5	0.27	0.29
Average	0.28	0.29

5.4.3 Indoor Target Tracking

For the indoor target tracking test, the UAV used the multiple model DEKF described in Chapter 4, however when there is only one UAV in play, the multiple model DEKF reverts to a traditional EKF, using the model inherent to that UAV. For the target, the Turtlebot, as shown in Figure 3.2 was used, except this time instead of a pre-programmed path it was controlled by the keyboard of a ground computer through SSH protocols. Relevant initialization is shown in Table 5.7. The way the test is set up, the UAV flies from rest to the point $\begin{bmatrix} {}^{i}x & {}^{i}y & {}^{i}z \end{bmatrix}^{\mathsf{T}} = \begin{bmatrix} 0 & 0 & 1.2 \end{bmatrix}^{\mathsf{T}}$. This is where the target is located and is done through only the use of the high level position controller. Furthermore, there is no yaw control at this point either, meaning the UAV maintains a pre-specified yaw. This configuration is maintained for roughly one hundred measurements of the target, allowing the state estimation to converge. After this, both the MPC and the UAV's yaw control is activated, and the UAV flies to a height of 1.5 meters for a larger field of view. The results of the estimation are shown in Figure 5.8. There is some estimation error present, as was described and explained in Chapter 4. This error has an effect on the control inputs and the

Table 5.7: Parameter Initialization of Single UAV MPC Indoor Flight

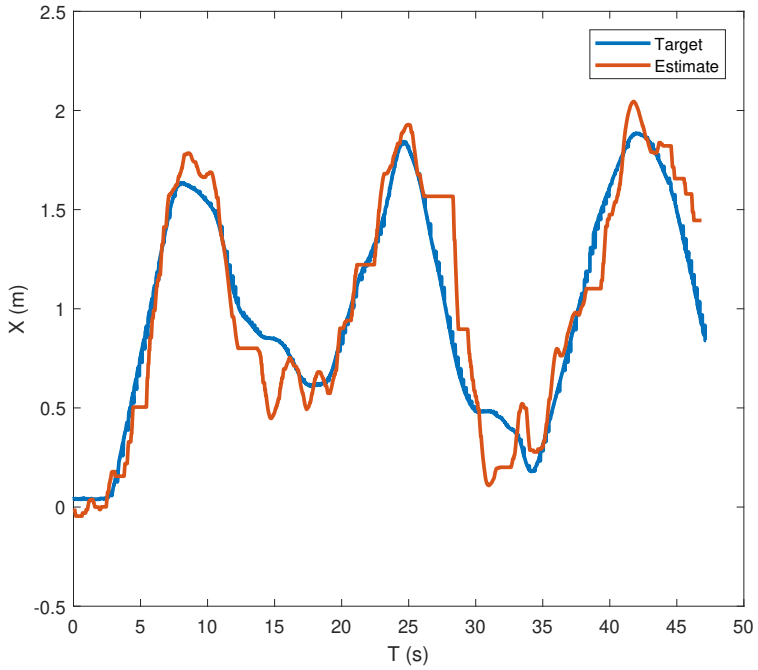
Parameter	Value
UAV Model	CA
${}^{i}h$	5
${}^{i}T$	0.2 s
${}^{i}Q$	12
${}^{i}R$	0.15
Input Constraint	$-1.5 \leq {}^{i}\mathbf{u}(k) \leq 1.5$
${}^{i}\mathbf{r}_r$	$\begin{bmatrix} -0.75 & 0 \end{bmatrix}^\mathsf{T}$
Initial UAV position	$\begin{bmatrix} {}^{i}x & {}^{i}y \end{bmatrix}^\mathsf{T} = \begin{bmatrix} 0 & 0 \end{bmatrix}^\mathsf{T}$
Initial Target position	$\begin{bmatrix} {}_{g}x & {}_{g}y \end{bmatrix}^\mathsf{T} = \begin{bmatrix} 0 & 0 \end{bmatrix}^\mathsf{T}$

final trajectory of the UAV as well, which are shown in Figure 5.9. Recall that the goal of the experiment was for the UAV to stay -0.75 m in x from the target while attempting to maintain the same y as the target. From 5.9(a), it is seen that the UAV does indeed stay behind the target for most of the experiment. The relevant errors of the experiment are shown in Table 5.8.

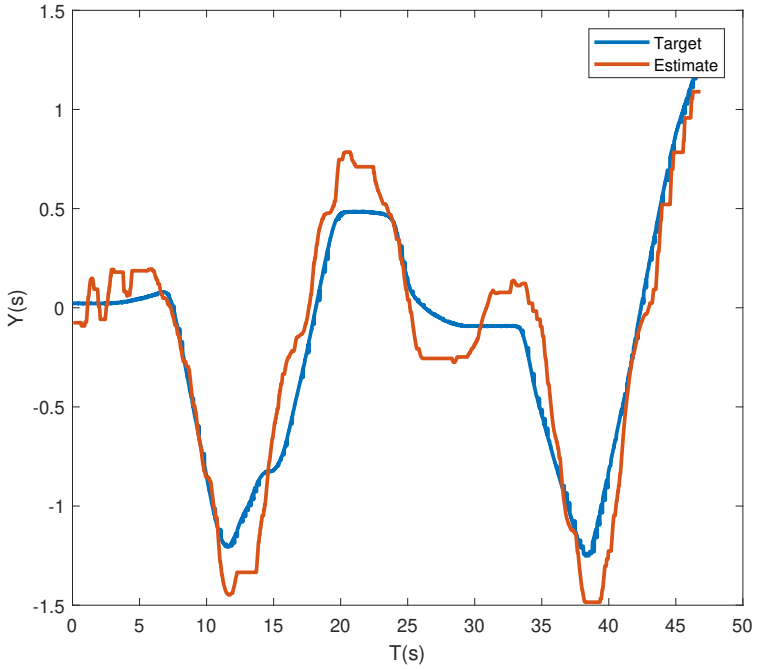
Table 5.8: Mean Tracking and Estimation Errors in Indoor Flight

Estimation Error (distance)	0.25 m
Tracking Error in x (x distance from -0.75m of target)	0.24 m
Tracking Error in y (y distance from 0 m of target)	0.39 m

From Table 5.8, it can be seen that there is some error present in both the estimation and tracking. The tracking errors, as previously mentioned, are influenced by the estimation errors. One other possible source of error may be that the response of the MPC controller is too aggressive, as the UAV overshoots the setpoint in some instances. This is supported by the fact that in the MPC step response (see Figure 5.4) has an overshoot as well.

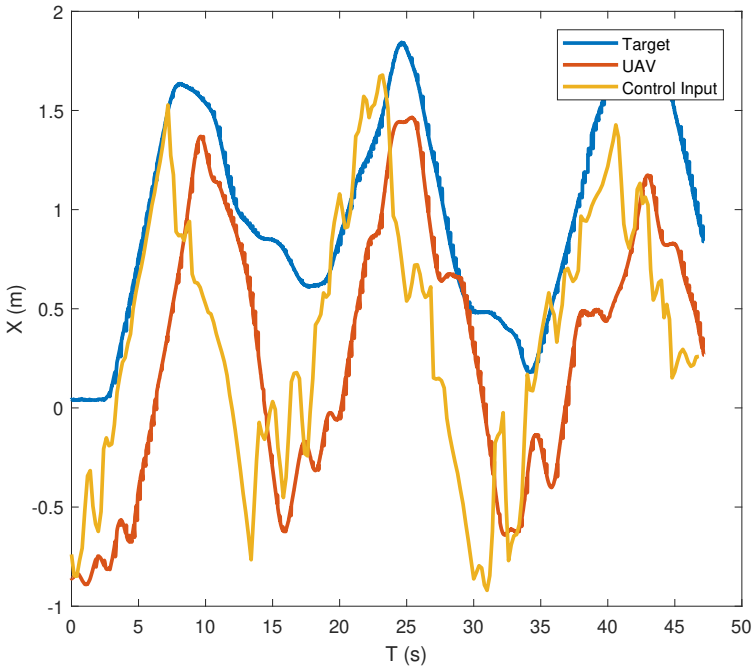


(a)

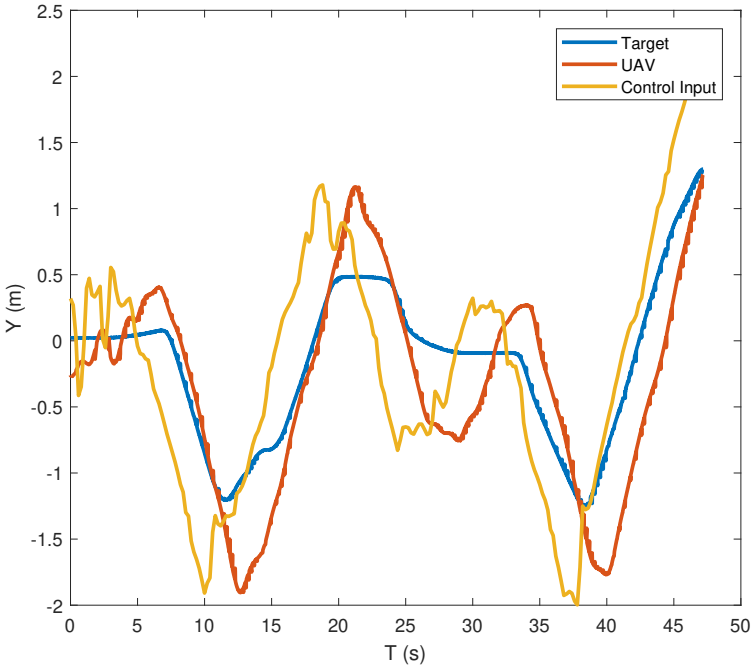


(b)

Figure 5.8: (a) Estimation result in x (b) Estimation result in y



(a)



(b)

Figure 5.9: (a) Control result in x (b) Control result in y

5.4.4 Multi-UAV Simulation

5.4.4.1 Collision Avoidance

To validate the collision avoidance algorithm, two UAVs in the VREP simulation were used, with the relevant initialization shown in Table 5.9. For this test, the target remains

Table 5.9: Parameter Initialization of Collision Avoidance Simulation Scenario

Parameter	UAV 1	UAV 2
${}^{i}h$	5	5
${}^{i}T$	0.2 s	0.2 s
${}^{i}Q$	12	12
${}^{i}Q_s$	6	6
${}^{i}R$	0.15	0.15
Input Constraint	$-2 \leq {}^{1}\mathbf{u}(k) \leq 2$	$-2 \leq {}^{2}\mathbf{u}(k) \leq 2$
${}^{i}\mathbf{r}_r$	$\begin{bmatrix}1 & 1\end{bmatrix}^\mathsf{T}$	$\begin{bmatrix}-5 & -5\end{bmatrix}^\mathsf{T}$
Initial UAV position	$\begin{bmatrix}{}^{1}x & {}^{1}y\end{bmatrix}^\mathsf{T} = \begin{bmatrix}0 & 0\end{bmatrix}^\mathsf{T}$	$\begin{bmatrix}{}^{2}x & {}^{2}y\end{bmatrix}^\mathsf{T} = \begin{bmatrix}5 & 5\end{bmatrix}^\mathsf{T}$
${}^{ij}r_s$	5	5

stationary at its initial position of $\begin{bmatrix}{}_{g}x & {}_{g}y\end{bmatrix}^\mathsf{T} = \begin{bmatrix}1 & 1\end{bmatrix}^\mathsf{T}$ and the UAVs are to adopt the prescribed formation by their respective ${}^{i}\mathbf{r}_r$. Essentially, UAV 1 is to go from $({}^{1}x(0), {}^{1}y(0)) = (0,0) \rightarrow (1,1)$ and UAV 2 is to go from $({}^{2}x(0), {}^{2}y(0)) = (5,5) \rightarrow (-4,-4)$. Thus, UAV 1 is in the direct path of UAV 2. The responses of the UAVs for this scenario are shown in Figure 5.10.

From Figure 5.10, the UAVs successfully avoid each other and assemble the correct formation. It is seen clearly that once UAV 2 enters the zone defined by ${}^{21}r_s$, it begins the avoidance procedure. Note that it is normal that UAV 2 enters this zone, as the separation term in the overall cost function is only added once a UAV enters the separation zone.

5.4.4.2 Multi-UAV Tracking

With properly functioning collision avoidance between UAVs, a tracking scenario similar to that described in Section 5.4.2 is adopted. However, in this case, three UAVs track

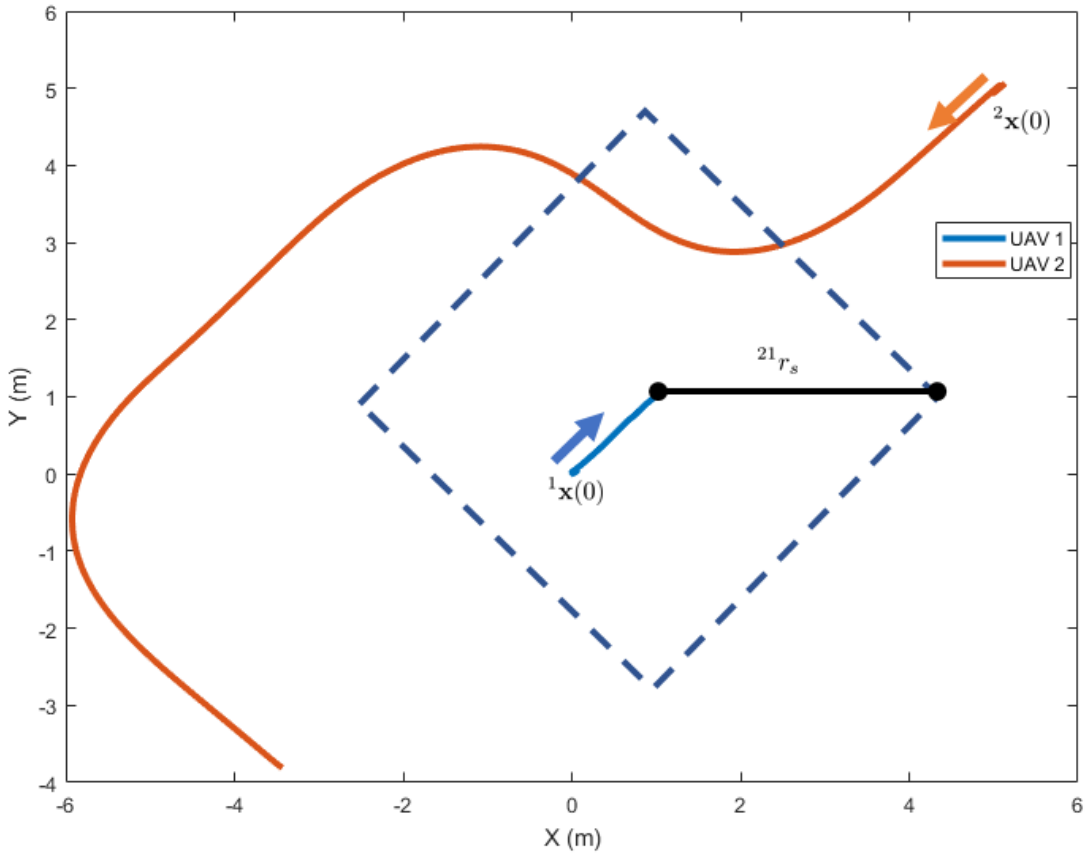


Figure 5.10: In the figure, two UAVs assemble a formation for a stationary target. In order to get to the formation, UAV 2 must avoid UAV 1.

the target in formation. The relevant initializations for the simulation are given in Table 5.10.

Table 5.10: Parameter Initialization of Multi UAV DMPC in Simulation

Parameter	UAV 1	UAV 2	UAV 3
^{i}h	5	5	5
^{i}T	0.2 s	0.2 s	0.2 s
^{i}Q	12	12	12
$^{i}Q_s$	6	6	6
^{i}R	0.15	0.15	0.15
Input Constraint	$-2 \leq {}^{1}\mathbf{u}(k) \leq 2$	$-2 \leq {}^{2}\mathbf{u}(k) \leq 2$	$-2 \leq {}^{3}\mathbf{u}(k) \leq 2$
$^{i}\mathbf{r}_r$	$[0 \quad 0]^\mathsf{T}$	$[10 \quad 10]^\mathsf{T}$	$[-10 \quad -10]^\mathsf{T}$
Initial UAV position	$[^{1}x \quad {}^{1}y]^\mathsf{T} = [0 \quad 0]^\mathsf{T}$	$[^{2}x \quad {}^{2}y]^\mathsf{T} = [10 \quad 10]^\mathsf{T}$	$[^{3}x \quad {}^{3}y]^\mathsf{T} = [-10 \quad -10]^\mathsf{T}$
$^{ij}r_s$	5	5	5

From Table 5.10, the three UAVs form a diagonal line between them, with UAV 1 attempting maintain over the target. UAV 2 should remain 10 m in front and to the right

of the target, whereas UAV 3 should be to the back and left of the target by 10 m. In this scenario, the initial target position is $\begin{bmatrix} {}_{g}x & {}_{g}y \end{bmatrix}^{\mathsf{T}} = \begin{bmatrix} 1 & 1 \end{bmatrix}^{\mathsf{T}}$. Furthermore, the target is given a 15 s head start over the UAVs, forcing them to catch up. The corresponding response of the formation is shown in Figure 5.11 and in the time domain in Figure 5.12.

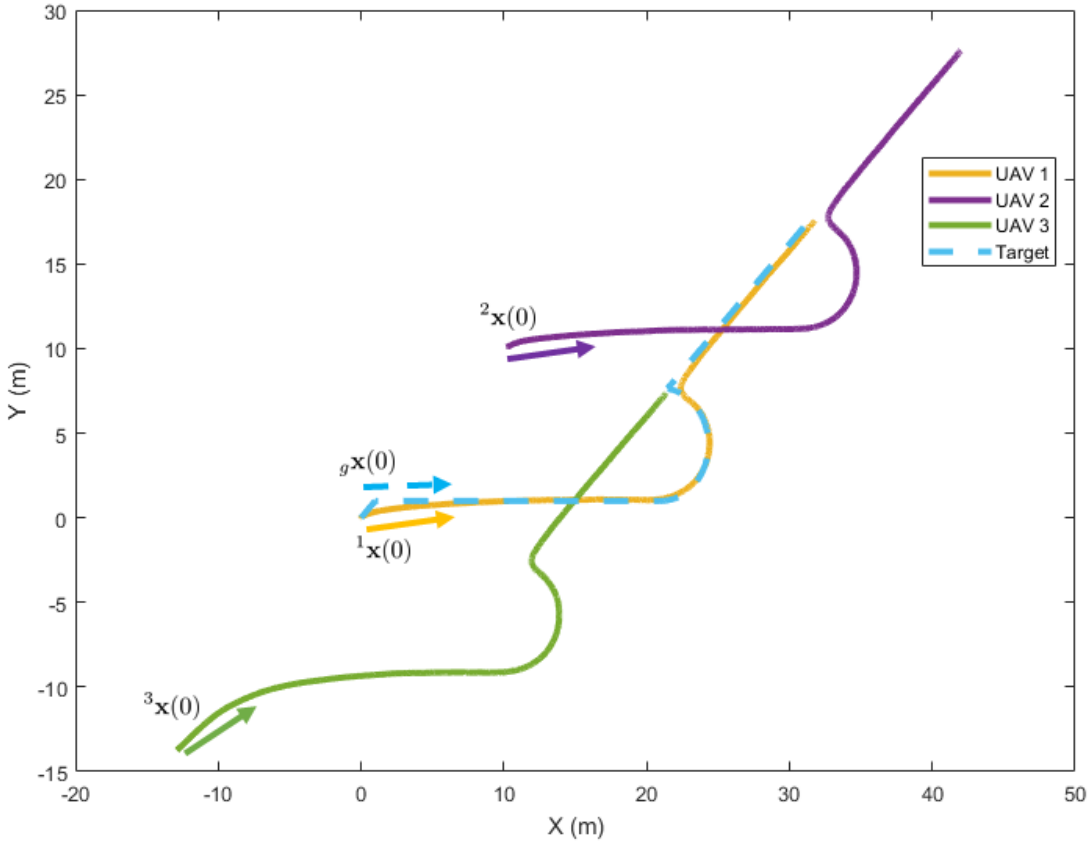
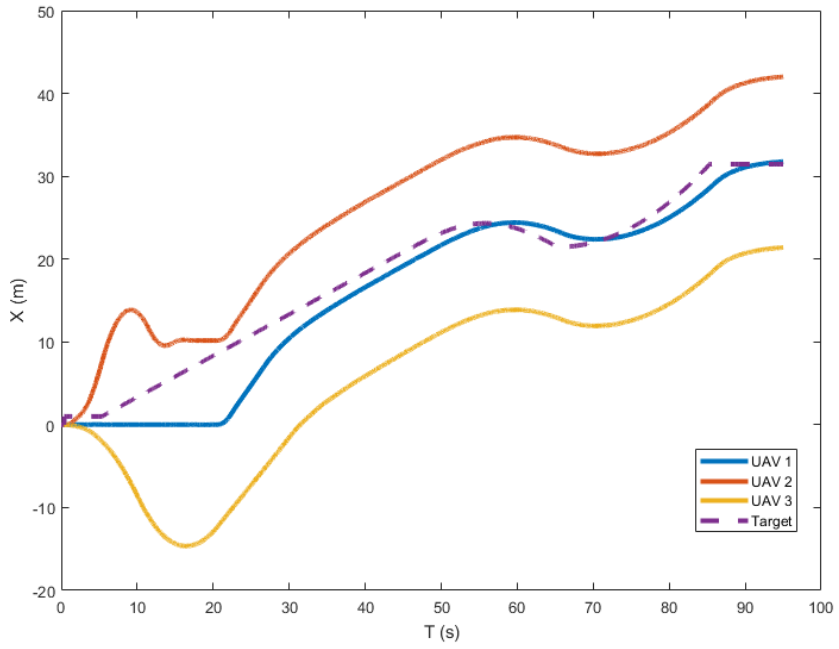
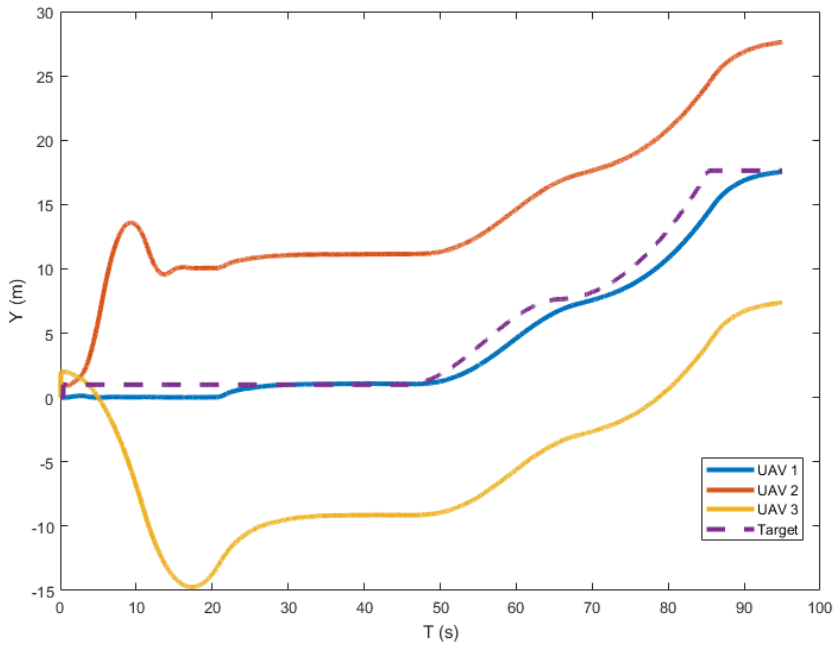


Figure 5.11: In the figure, three UAVs assemble a formation to track an evasive target.

It should be noted that the first 20 seconds of the simulation are left for the UAVs to get to their respective initial positions. Therefore, the DMPC only begins at $t = 20$ s. In the meantime, the target is allowed to move and get away. However, as can be noted from Figures 5.11 and 5.12, the UAVs catch up to the target and maintain the desired formation. The respective mean error of the UAVs from their desired position in the formation is shown in Table 5.11.



(a)



(b)

Figure 5.12: (a) UAV responses in x (b) UAV responses in y

Table 5.11: Mean Tracking Error for a Multi-UAV Target Tracking Simulation

	UAV 1	UAV 2	UAV 3
Tracking Error (distance in meters from ${}^{i}\mathbf{x}_{r}$)	1.42 m	1.18 m	2.13 m

5.4.4.3 Occlusions

Up until now, all tests have been done with the notion that all UAVs have a perfect version of the MMDEKF described in Chapter 4. This implies that each UAV knows perfectly the states of the target and which model (CV, CA or CT) most accurately represents the target's motion. In this section, the effects of occlusions on the control is investigated in order to demonstrate how the proposed overall architecture is more robust than traditional implementations. To accomplish this, a similar simulation as that presented in Section 5.4.4.2 was implemented. The exception however is that for two periods of time, UAV 3 will lose communication with the other UAVs and will lose sight of the target. Essentially, the UAV will undergo a complete occlusion, where only the UAV's last estimate and model chosen can be used to track the target. The relevant initialization is identical to that of Table 5.10.

The occlusions occur for UAV 3 during two instances, the first at $40s \leq t \leq 55s$ and the second at $65s \leq t \leq 75s$. Any other period of time, UAV 3 maintains perfect estimation and sharing of information. The corresponding response of the formation is shown in Figure 5.13 and in the time domain in Figure 5.14.

Once again it should be noted that the first 20 seconds of the simulation are left for the UAVs to get to their respective initial positions. From the figures, it can be seen that UAVs 1 and 2 are not greatly affected by UAV 3 undergoing an occlusion. Conversely, UAV 3 does deviate from the desired formation during times of occlusions. This demonstrates three qualities of the DMPC controller and overall target tracking architecture. First, the distributed nature of the DMPC controller allows the UAVs not undergoing an occlusion (i.e. UAVs 1 and 2) to continue tracking the target despite one of the UAVs (i.e. UAV 3) in the formation veering from the desired formation due to an occlusion. Second, the

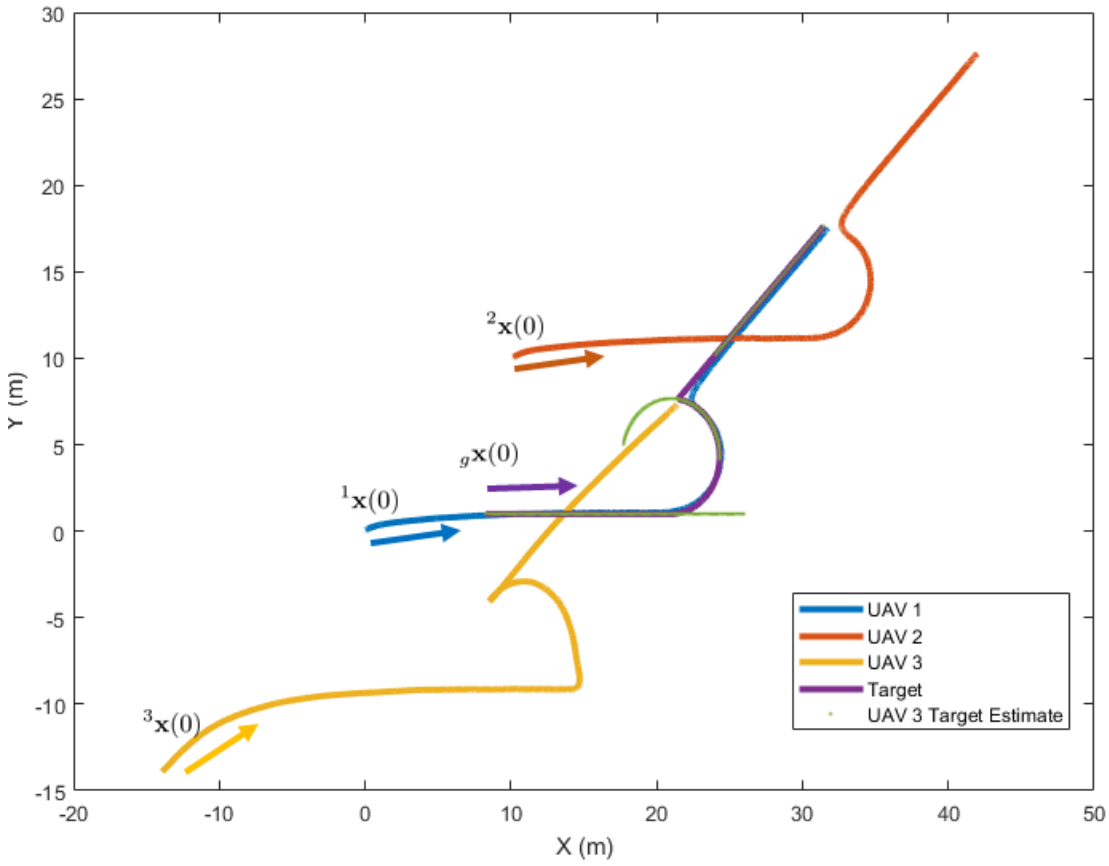
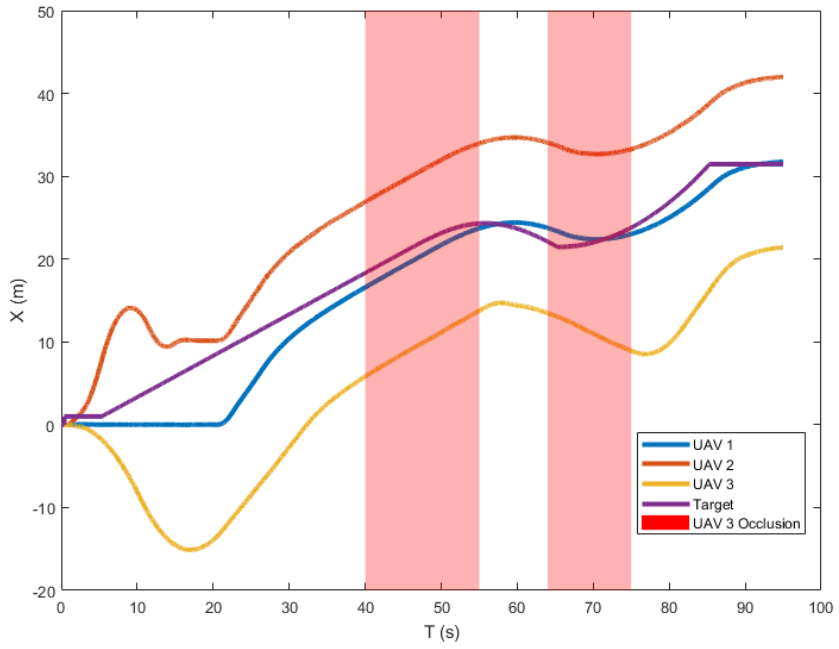


Figure 5.13: In the figure, three UAVs assemble a formation to track an evasive target. UAV 3 undergoes multiple occlusions, as depicted in the figure, through faulty estimation of the target.

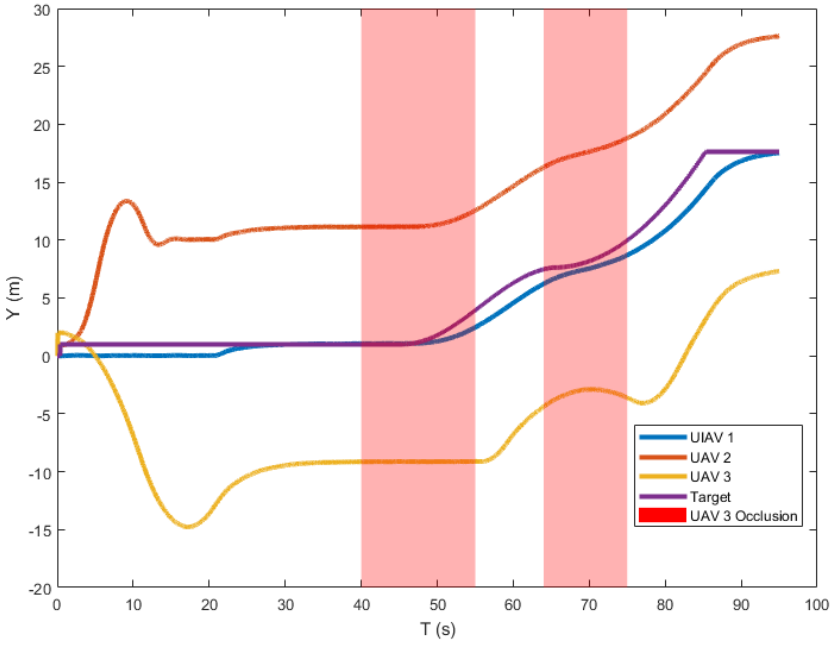
poorer tracking performance of UAV 3 overall validates the proposed architecture and the importance of sharing necessary information, as described in this Chapter and Chapter 4. Last, despite UAV 3 undergoing two occlusions totalling 25 s, it still manages to regain its trajectory, showing that the proposed DMPC controller is stable even with the possibility of occlusions. The respective mean error of the UAVs from their desired position in the formation is shown in Table 5.12. The changes in errors from Table 5.11 to Table 5.12 for

Table 5.12: Mean Tracking Error for a Multi-UAV Target Tracking Simulation During Occlusions

	UAV 1	UAV 2	UAV 3
Tracking Error (distance in meters from $^i\mathbf{x}_r$)	1.44 m	1.15 m	3.25 m



(a)



(b)

Figure 5.14: (a) UAV responses in x (b) UAV responses in y

UAVs 1 and 2 are negligible, whereas the error for UAV 3 has increased significantly.

5.5 Closing Remarks

In this chapter, a two step approach to validating a multiple model distributed MPC controller was followed. The first step consisted of implementing an MPC control law using multiple models for a single UAV maintaining a relative position to a mobile ground target. Simulation in the VREP environment with perfect state estimation showed no difference between the multiple model and single model MPC approaches. However, more tests should be done with better tuning of the parameters. For example, a longer prediction horizon and control horizon. It was hypothesized that the short prediction horizon led to the insignificant differences in model prediction. This algorithm was then validated in an indoor flight test, with the AscTec Pelican and Turtlebot. The second step in the approach came with the formulation of another quadratic term in the overall objective function to maintain separation between UAVs so that a team of UAVs could track a target in formation while avoiding collisions. This distributed multiple model MPC controller was tested in simulation as a team of three UAVs tracked a ground target, both with and without the presence of occlusions, showing the advantages of a distributed network with sharing of measurements.

Chapter 6

Conclusion

In this thesis, a comprehensive investigation into using multiple models for target tracking with a team of UAVs was conducted. First, a complete testing platform of UAVs was designed and implemented for indoor flight and configured for outdoor flight. This meant configuration, implementation and tuning of various necessary components, including UAV state estimation, computer vision, various levels of control, sensing and communication. Some advantages of the implementation are that the UAVs are not assumed to travel at a specific speed and that they can yaw towards the target, two aspects not common in the literature, as they are direct consequences of working with VTOL UAVs.

Second, through real-time indoor tests, it was shown that the multiple model DEKF can be effectively used to estimate the position of a mobile ground target. The novelties in the algorithm include using the T Test as a model selection criteria, the distribution of computation over the network of UAVs that is inherent in multiple model estimation, and finally the complete independence of the algorithm on other UAV information. By the latter what is meant is that a UAV will use a neighbouring UAVs' information if available but does not require it to estimate the target's whereabouts. Conversely, a UAV does not need to detect the target in order to estimate its positions, as neighbouring UAVs can share their measurements.

Last, the control of the UAVs was carried out through distributed MPC. The main novelties of this algorithm was the use of multiple models to predict the target's motion, which used the aforementioned state estimation algorithm to know which model was performing best at which time. Although it was not shown that the multiple model method improves the tracking performance over the single model method, there are several advantages to the im-

plementation. The major advantage is that the algorithm can be run using solely the UAV's on board computation power, as the objective function designed, as well as the constraints implemented in the multi UAV scenario, are all convex, leading to quicker convergence. This is rare among the literature and arises from a second advantage of the implementation, which is a flexible formation flight. As each UAV attempts to maintain a certain position relative to the target, the shape of the formation can be specified by the user and can change over time, depending on the needs of the team of UAVs. Specification of a desired point to maintain and not a specific distance, coupled with completely linear models, are what keep the objective convex.

The area of multiple UAV target tracking is an expanding area with many opportunities for further research. The research conducted here reflects a small subset of the research being done currently in the field. Furthermore, although several contributions were made in the thesis, there still remains more work that can be done to further validate and expand upon the algorithms.

In the robotics implementation, all that remains is the outdoor implementation and some code optimization. All the necessary outdoor modifications have been made and are ready to be tested, both on the component level and the full integration.

On the state estimation side, the T Test selection criteria can be further refined still in order to select the CV model in the appropriate times. Also, to reduce communication needs between UAVs, the T Test selection can be further optimized so that less information needs to be shared. For example, instead of sharing the last n residual norms, some other metric can be used, such as the average of the residuals. Furthermore, real indoor and outdoor flight tests would only further validate the integrity of the algorithm.

The equivalent can be said of the control part of the thesis, where only indoor single UAV tests were conducted. Due to the size of the indoor arena and the turbulence that arises from being indoors, multiple UAV tests were not as feasible. That is why they were carried out in simulation, but outdoor validation should be done as well. Another future area of

further investigation would be to clearly determine whether or not multiple target models improves tracking performance, as only the preliminary ground work was conducted here.

Last, as seen from the indoor flight data, both the control and estimation accuracy can be further improved for an overall smoother tracking performance. Other limitations that arose from the implementation are the fact that the target is assumed to have a constant elevation, which would change in a large scale outdoor implementation. Furthermore, there is no height control done in the MPC formulation, the UAVs maintain a constant height. Last, this work pertains to VTOL UAVs mostly, as some assumptions, like being able to hover for example, cannot be expanded to fixed wing UAVs.

References

[1] Lucintel Brief. Growth opportunity in global uav market. *Las Colinas, USA*, 2011.

[2] Reg Austin. *Unmanned aircraft systems: UAVS design, development and deployment*, volume 54. John Wiley & Sons, 2011.

[3] M Hassanalian and A Abdelkefi. Classifications, applications, and design challenges of drones: a review. *Progress in Aerospace Sciences*, 91:99–131, 2017.

[4] John Villasenor. "drones" and the future of domestic aviation [point of view]. *Proceedings of the IEEE*, 102(3):235–238, 2014.

[5] Y Uny Cao, Alex S Fukunaga, and Andrew Kahng. Cooperative mobile robotics: Antecedents and directions. *Autonomous robots*, 4(1):7–27, 1997.

[6] Andrew W Sanders. Drone swarms. Technical report, US Army School for Advanced Military Studies Fort Leavenworth United States, 2017.

[7] Adam C Watts, Vincent G Ambrosia, and Everett A Hinkley. Unmanned aircraft systems in remote sensing and scientific research: Classification and considerations of use. *Remote Sensing*, 4(6):1671–1692, 2012.

[8] Farid Kendoul. Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems. *Journal of Field Robotics*, 29(2):315–378, 2012.

[9] Lili Ma and Naira Hovakimyan. Cooperative target tracking with time-varying formation radius. In *Control Conference (ECC), 2015 European*, pages 1699–1704. IEEE, 2015.

[10] Richard Wise and Rolf Rysdyk. Uav coordination for autonomous target tracking. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, page 6453, 2006.

[11] Marc D Richards, Darrell Whitley, J Ross Beveridge, Todd Mytkowicz, Duong Nguyen, and David Rome. Evolving cooperative strategies for uav teams. In *Proceedings of the 7th annual conference on Genetic and evolutionary computation*, pages 1721–1728. ACM, 2005.

[12] ML Cummings. Operator interaction with centralized versus decentralized uav architectures. In *Handbook of Unmanned Aerial Vehicles*, pages 977–992. Springer, 2015.

[13] Anthony J Marasco, Sidney N Givigi, Camille Alain Rabbath, and Alain Beaulieu. Dynamic encirclement of a moving target using decentralized nonlinear model predictive control. In *American Control Conference (ACC), 2013*, pages 3960–3966. IEEE, 2013.

[14] Nicolaj Siggelkow and Daniel A Levinthal. Temporarily divide to conquer: Centralized, decentralized, and reintegrated organizational approaches to exploration and adaptation. *Organization Science*, 14(6):650–669, 2003.

[15] RR Negenborn and JM Maestre. Distributed model predictive control: An overview and roadmap of future research opportunities. *IEEE Control Systems*, 34(4):87–97, 2014.

[16] Reza Olfati-Saber. Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Transactions on automatic control*, 51(3):401–420, 2006.

[17] Fabrizio Giulietti, Lorenzo Pollini, and Mario Innocenti. Autonomous formation flight. *IEEE Control Systems*, 20(6):34–44, 2000.

[18] Seunghan Lim, Yeongju Kim, Dongjin Lee, and Hyochoong Bang. Standoff target tracking using a vector field for multiple unmanned aircrafts. *Journal of Intelligent & Robotic Systems*, 69(1-4):347–360, 2013.

[19] Jongrae Kim and Yoonsoo Kim. Moving ground target tracking in dense obstacle areas using uavs. *IFAC Proceedings Volumes*, 41(2):8552–8557, 2008.

[20] Gürdal Arslan, Jason R Marden, and Jeff S Shamma. Autonomous vehicle-target assignment: A game-theoretical formulation. *Journal of Dynamic Systems, Measurement, and Control*, 129(5):584–596, 2007.

[21] Zongyao Wang and Dongbing Gu. Cooperative target tracking control of multiple robots. *IEEE Transactions on Industrial Electronics*, 59(8):3232–3240, 2012.

[22] Guoquan Huang, Michael Kaess, and John J Leonard. Consistent unscented incremental smoothing for multi-robot cooperative target tracking. *Robotics and autonomous systems*, 69:52–67, 2015.

[23] Simon Cooper and Hugh Durrant-Whyte. A kalman filter model for gps navigation of land vehicles. In *Intelligent Robots and Systems' 94.'Advanced Robotic Systems and the Real World', IROS'94. Proceedings of the IEEE/RSJ/GI International Conference on*, volume 1, pages 157–163. IEEE, 1994.

[24] Jun Zhang, Weisong Liu, and Yirong Wu. Novel technique for vision-based uav navigation. *IEEE Transactions on Aerospace and Electronic Systems*, 47(4):2731–2741, 2011.

[25] David Nistér, Oleg Naroditsky, and James Bergen. Visual odometry for ground vehicle applications. *Journal of Field Robotics*, 23(1):3–20, 2006.

[26] Keiji Nagatani, Satoshi Tachibana, Makoto Sofne, and Yutaka Tanaka. Improvement of odometry for omnidirectional vehicle using optical flow information. In *Intelligent*

Robots and Systems, 2000.(IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on, volume 1, pages 468–473. IEEE, 2000.

[27] Boris Sofman, J Bagnell, Anthony Stentz, and Nicolas Vandapel. Terrain classification from aerial data to support ground vehicle navigation. *Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMURI-TR-05-39*, 2006.

[28] Jesse Levinson, Michael Montemerlo, and Sebastian Thrun. Map-based precision vehicle localization in urban environments. In *Robotics: Science and Systems*, volume 4, page 1. Citeseer, 2007.

[29] Mohinder S Grewal. Kalman filtering. In *International Encyclopedia of Statistical Science*, pages 705–708. Springer, 2011.

[30] Gabriel A Terejanu et al. Extended kalman filter tutorial. *Department of Computer Science and Engineering, University at Buffalo. Available online: https://homes. cs. washington. edu/˜ todorov/courses/cseP590/readings/tutorialEKF. pdf (accessed on 16 February 2017)*, 2008.

[31] Eric A Wan and Rudolph Van Der Merwe. The unscented kalman filter. *Kalman filtering and neural networks*, 5(2007):221–280, 2001.

[32] Rambabu Kandepu, Bjarne Foss, and Lars Imsland. Applying the unscented kalman filter for nonlinear state estimation. *Journal of process control*, 18(7-8):753–768, 2008.

[33] Eric A Wan and Rudolph Van Der Merwe. The unscented kalman filter for nonlinear estimation. In *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*, pages 153–158. Ieee, 2000.

[34] M Karimi, M Bozorg, and AR Khayatian. A comparison of dvl/ins fusion by ukf

and ekf to localize an autonomous underwater vehicle. In *Robotics and Mechatronics (ICRoM), 2013 First RSI/ISM International Conference on*, pages 62–67. IEEE, 2013.

[35] Zongwen Xue and Howard Schwartz. A comparison of several nonlinear filters for mobile robot pose estimation. In *2013 IEEE International Conference on Mechatronics and Automation*, pages 1087–1094. IEEE, 2013.

[36] Simon Haykin. *Kalman filtering and neural networks*, volume 47. John Wiley & Sons, 2004.

[37] Mark L Darby and Michael Nikolaou. Mpc: Current practice and challenges. *Control Engineering Practice*, 20(4):328–342, 2012.

[38] Morten Bisgaard, Anders la Cour-Harbo, and Jan Dimon Bendtsen. Adaptive control system for autonomous helicopter slung load operations. *Control Engineering Practice*, 18(7):800–811, 2010.

[39] Pedro Castillo, Alejandro Dzul, and Rogelio Lozano. Real-time stabilization and tracking of a four-rotor mini rotorcraft. *IEEE Transactions on control systems technology*, 12(4):510–516, 2004.

[40] Farid Kendoul, Zhenyu Yu, and Kenzo Nonami. Guidance and nonlinear control system for autonomous flight of minirotorcraft unmanned aerial vehicles. *Journal of Field Robotics*, 27(3):311–334, 2010.

[41] Vladislav Gavrilets, Emilio Frazzoli, Bernard Mettler, Michael Piedmonte, and Eric Feron. Aggressive maneuvering of small autonomous helicopters: A human-centered approach. *The International Journal of Robotics Research*, 20(10):795–807, 2001.

[42] Subodh Bhandari, Richard Colgren, Philipp Lederbogen, and Scott Kowalchuk. Six-dof dynamic modeling and flight testing of a uav helicopter. In *AIAA Modeling and Simulation Technologies Conference and Exhibit*, page 6422, 2005.

[43] David H Shim, H Jin Kim, and Shankar Sastry. Decentralized nonlinear model predictive control of multiple flying robots. In *42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03CH37475)*, volume 4, pages 3621–3626. IEEE, 2003.

[44] Mark B Tischler and Mavis G Cauffman. Frequency-response method for rotorcraft system identification: Flight applications to bo 105 coupled rotor/fuselage dynamics. *Journal of the American Helicopter Society*, 37(3):3–17, 1992.

[45] Farid Kendoul. Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems. *Journal of Field Robotics*, 29(2):315–378, 2012.

[46] Johannes Meyer, Alexander Sendobry, Stefan Kohlbrecher, Uwe Klingauf, and Oskar Von Stryk. Comprehensive simulation of quadrotor uavs using ros and gazebo. In *International conference on simulation, modeling, and programming for autonomous robots*, pages 400–411. Springer, 2012.

[47] Peter T Jardine, Sidney Givigi, and Shahram Yousefi. Parameter tuning for prediction-based quadcopter trajectory planning using learning automata. *IFAC-PapersOnLine*, 50(1):2341–2346, 2017.

[48] Herman O Hartley. The modified gauss-newton method for the fitting of non-linear regression functions by least squares. *Technometrics*, 3(2):269–280, 1961.

[49] Petar M Djuric, Mahesh Vemula, and Mónica F Bugallo. Target tracking by particle filtering in binary sensor networks. *IEEE Transactions on signal processing*, 56(6):2229–2238, 2008.

[50] Chaw-Bing Chang and John Tabaczynski. Application of state estimation to target tracking. *IEEE Transactions on Automatic Control*, 29(2):98–109, 1984.

[51] Xianghui Yuan, Feng Lian, and Chongzhao Han. Models and algorithms for tracking target with coordinated turn motion. *Mathematical Problems in Engineering*, 2014, 2014.

[52] Eduardo F Camacho and Carlos Bordons Alba. *Model predictive control*. Springer Science & Business Media, 2013.

[53] Philip E Gill, Walter Murray, and Margaret H Wright. Practical optimization. *London: Academic Press, 1981*, 1981.

[54] Ali A Jalali and Vahid Nadimi. A survey on robust model predictive control from 1999-2006. In *2006 International Conference on Computational Inteligence for Modelling Control and Automation and International Conference on Intelligent Agents Web Technologies and International Commerce (CIMCA'06)*, pages 207–207. IEEE, 2006.

[55] Lucintel Brief. Growth opportunity in global uav market. *Las Colinas, USA*, 2011.

[56] Yoko Watanabe, Anthony Calise, and Eric Johnson. Vision-based obstacle avoidance for uavs. In *AIAA Guidance, Navigation and Control Conference and Exhibit*, page 6829, 2007.

[57] Ugur Zengin and Atilla Dogan. Real-time target tracking for autonomous uavs in adversarial environments: A gradient search algorithm. *IEEE Transactions on Robotics*, 23(2):294–307, 2007.

[58] Senqiang Zhu, Danwei Wang, and Chang Boon Low. Ground target tracking using uav with input constraints. *Journal of Intelligent & Robotic Systems*, 69(1-4):417–429, 2013.

[59] Zhiyuan Li, Naira Hovakimyan, Vladimir Dobrokhodov, and Isaac Kaminer. Vision-

based target tracking and motion estimation using a small uav. In *Decision and Control (CDC), 2010 49th IEEE Conference on*, pages 2505–2510. IEEE, 2010.

[60] Markus W Achtelik, Michael Achtelik, Stephan Weiss, and Roland Siegwart. Onboard imu and monocular vision based control for mavs in unknown in-and outdoor environments. In *IEEE International Conference on Robotics and Automation (ICRA 2011)*. Eidgenössische Technische Hochschule Zürich, Autonomous Systems Lab, 2011.

[61] Simon Lynen, Markus W Achtelik, Stephan Weiss, Margarita Chli, and Roland Siegwart. A robust and modular multi-sensor fusion approach applied to mav navigation. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 3923–3929. IEEE, 2013.

[62] Sergio Garrido-Jurado, Rafael Muñoz-Salinas, Francisco José Madrid-Cuevas, and Manuel Jesús Marín-Jiménez. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47(6):2280–2292, 2014.

[63] Sergi Hernandez Juan and Fernando Herrero Cotarelo. Multi-master ros systems. *Institut de Robotics and Industrial Informatics*, pages 1–18, 2015.

[64] Natural Point. Optitrack. *Natural Point, Inc*, 2011.

[65] M. Freese E. Rohmer, S. P. N. Singh. V-rep: a versatile and scalable robot simulation framework. In *Proc. of The International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[66] Henk AP Blom and Yaakov Bar-Shalom. The interacting multiple model algorithm for systems with markovian switching coefficients. *IEEE transactions on Automatic Control*, 33(8):780–783, 1988.

[67] Wei Zhu, Wei Wang, and Gannan Yuan. An improved interacting multiple model filtering algorithm based on the cubature kalman filter for maneuvering target tracking. *Sensors*, 16(6):805, 2016.

[68] Lennon Cork and Rodney Walker. Sensor fault detection for uavs using a nonlinear dynamic model and the imm-ukf algorithm. In *2007 Information, Decision and Control*, pages 230–235. IEEE, 2007.

[69] Alexander Scott Campbell and Howard M Schwartz. Multiple model control improvements: hypothesis testing and modified model arrangement. *Control and Intelligent Systems*, 35(3):236–243, 2007.

[70] Fahmida N Chowdhury. Kalman filter with hypothesis testing: A tool for estimating uncertain parameters. *Circuits, Systems and Signal Processing*, 15(3):291–311, 1996.

[71] Linfeng Xu, X Rong Li, and Zhansheng Duan. Hybrid grid multiple-model estimation with application to maneuvering target tracking. *IEEE Transactions on Aerospace and Electronic Systems*, 52(1):122–136, 2016.

[72] X Rong Li and Vesselin P Jilkov. Survey of maneuvering target tracking. part v. multiple-model methods. *IEEE Transactions on Aerospace and Electronic Systems*, 41(4):1255–1321, 2005.

[73] Reza Olfati-Saber. Distributed kalman filtering for sensor networks. In *2007 46th IEEE Conference on Decision and Control*, pages 5492–5498. IEEE, 2007.

[74] Tao Sun and Ming Xin. Multiple uav target tracking using consensus-based distributed high degree cubature information filter. In *AIAA guidance, navigation, and control conference*, page 0347, 2015.

[75] Ye Tian, Zhe Chen, and Fuliang Yin. Distributed imm-unscented kalman filter for

speaker tracking in microphone array networks. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 23(10):1637–1647, 2015.

[76] Venkata Pathuri Bhuvana and Andrea M Tonello. Distributed object tracking based on information weighted uav selection with priory objects. In *2017 25th European Signal Processing Conference (EUSIPCO)*, pages 2046–2050. IEEE, 2017.

[77] Gerasimos Rigatos. Distributed particle filtering over sensor networks for autonomous navigation of uavs. In *Advanced Strategies for Robot Manipulators*. IntechOpen, 2010.

[78] Walter Aburime. Autonomous navigation of unmanned aerial vehicles subjected to time delays. Master's thesis, Carleton University, 2019.

[79] VPS Naidu, Girija Gopalaratnam, and N Shanthakumar. Three model imm-ekf for tracking targets executing evasive maneuvers. In *45th AIAA Aerospace Sciences Meeting and Exhibit*, page 1204, 2007.

[80] Randall C Smith and Peter Cheeseman. On the representation and estimation of spatial uncertainty. *The international journal of Robotics Research*, 5(4):56–68, 1986.

[81] Lili Ma and Naira Hovakimyan. Cooperative target tracking in balanced circular formation: Multiple uavs tracking a ground vehicle. In *American Control Conference (ACC), 2013*, pages 5386–5391. IEEE, 2013.

[82] Wei Lin, Zhihua Qu, and Marwan A Simaan. Multi-pursuer single-evader differential games with limited observations. In *American Control Conference (ACC), 2013*, pages 2711–2716. IEEE, 2013.

[83] Venanzio Cichella, Isaac Kaminer, Vladimir Dobrokhodov, and Naira Hovakimyan. Coordinated vision-based tracking for multiple uavs. In *Intelligent Robots and Sys-*

tems (IROS), 2015 IEEE/RSJ International Conference on, pages 656–661. IEEE, 2015.

[84] Sangeeta Daingade, Arpita Sinha, Aseem Vivek Borkar, and Hemendra Arya. A variant of cyclic pursuit for target tracking applications: theory and implementation. *Autonomous Robots*, 40(4):669–686, 2016.

[85] Farshad Koohifar, Abhaykumar Kumbhar, and Ismail Guvenc. Receding horizon multi-uav cooperative tracking of moving rf source. *IEEE Communications Letters*, 21(6):1433–1436, 2017.

[86] Petr Tichavsky, Carlos H Muravchik, and Arye Nehorai. Posterior cramér-rao bounds for discrete-time nonlinear filtering. *IEEE Transactions on signal processing*, 46(5):1386–1396, 1998.

[87] Karol Hausman, Jörg Müller, Abishek Hariharan, Nora Ayanian, and Gaurav S Sukhatme. Cooperative multi-robot control for target tracking with onboard sensing. *The International Journal of Robotics Research*, 34(13):1660–1677, 2015.

[88] Karol Hausman, Jörg Müller, Abishek Hariharan, Nora Ayanian, and Gaurav S Sukhatme. Cooperative control for target tracking with onboard sensing. In *Experimental Robotics*, pages 879–892. Springer, 2016.

[89] Steven AP Quintero, David A Copp, and Joao P Hespanha. Robust coordination of small uavs for vision-based target tracking using output-feedback mpc with mhe. *Cooperative Control of Multi-Agent Systems: Theory and Applications*, pages 51–83, 2016.

[90] Peng Yao, Honglun Wang, and Zikang Su. Cooperative path planning with applications to target tracking and obstacle avoidance for multi-uavs. *Aerospace Science and Technology*, 54:10–22, 2016.

[91] Peng Yao, Honglun Wang, and Zikang Su. Real-time path planning of unmanned aerial vehicle for target tracking and obstacle avoidance in complex dynamic environment. *Aerospace Science and Technology*, 47:269–279, 2015.

[92] Dale A Lawrence, Eric W Frew, and William J Pisano. Lyapunov vector fields for autonomous unmanned aircraft flight control. *Journal of Guidance, Control, and Dynamics*, 31(5):1220–1229, 2008.

[93] Ahmed T Hafez, Sidney N Givigi, Khaled A Ghamry, and Shahram Yousefi. Multiple cooperative uavs target tracking using learning based model predictive control. In *Unmanned aircraft systems (icuas), 2015 international conference on*, pages 1017–1024. IEEE, 2015.