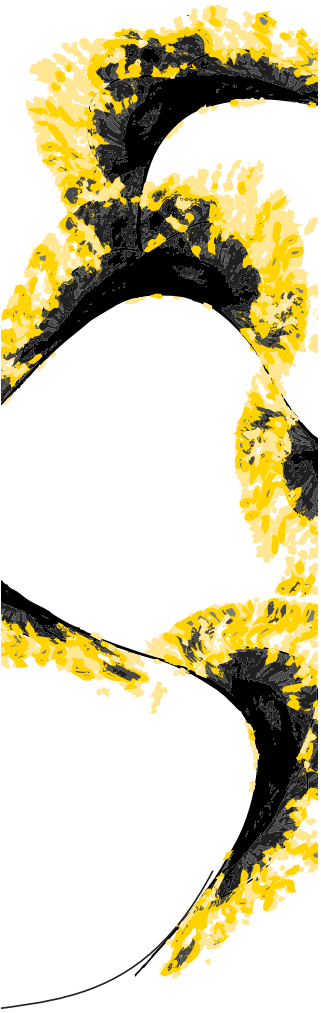


RAM ROBOTICS AND MECHATRONICS

HETEROGENEOUS COOPERATIVE TARGET TRACKING USING NONLINEAR MODEL PREDICTIVE CONTROL

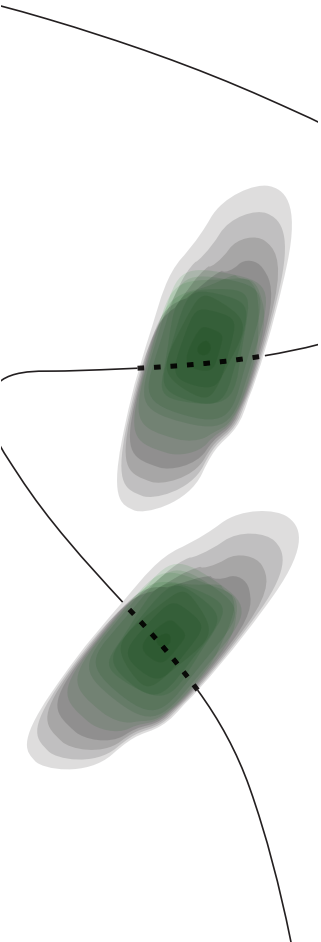
M.P.W. (Max) Kivits

MSC ASSIGNMENT

Committee:
A. Franchi, Ph.D HDR
ir. H. Das
M. Jacquet, MSc
dr. V.V. Lehtola

September 2021

061RaM2021
Robotics and Mechatronics
EEMathCS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands



UNIVERSITY OF TWENTE. | TECHMED CENTRE UNIVERSITY OF TWENTE. | DIGITAL SOCIETY INSTITUTE

Abstract

The autonomy of robots is limited by the accuracy of the information of their environment. The availability of accurate state estimates of the environment is crucial. This work researches how to obtain these state estimates and proposes a method to cooperatively track moving targets using a team of heterogeneous agents.

A state estimation scheme fuses the noisy intermittent target measurements and its covariance is used to quantify the estimation quality. To this end, a C++ implementation of an Intermittent Kalman filter in the Genom framework is constructed.

A NMPC method from previous work is adapted to include a the Intermittent Kalman filter covariance trace as perception objective. The proposed NMPC is able to compute the actuator inputs for a team of heterogeneous generic multi-rotors in order to minimize their collective state estimation covariance, whilst honoring actuator constraints. This removes the need for a cascaded control scheme.

The proposed method is validated using simulation experiments. The NMPC shown to drive two quadrotor sensing agent into a configuration that minimizes their state estimate covariance beyond what it can achieve using a single quadrotor. The method is able to tracking moving targets using both single and dual quadrotors, with a control frequency of over 600Hz and 300Hz respectively.

Acknowledgment

I would like to thank Martin Jacquet and Hemjyoti Das for their supervision, advice, and good company. I thank Antonio Franchi for his feedback, friendly advice, and providing me with the opportunity to do this research. I want to thank Ville Lehtola for agreeing to be the external member to my graduation committee. A big thanks to my parents, Esther and Eric for their continuous support and effort to understand this work. Finally I want to thank my Sophie for sharing my happiness and unconditionally supporting me throughout.

Contents

1 Introduction

1.1 Motivation

The goal of robotics is to design machines to assist or substitute humans in difficult tasks. To enable machines to properly interface with a physical process it is necessary to have information regarding the process state. Improving the ability of machines to measure a target's process state is a crucial step to improve the ability and autonomy of robots. To reduce uncertainty when inferring a process state, measurements from multiple sensing agents must be combined. One interesting question then is how to position the different sensing agents in order to maximize information yield. This is further complicated when considering sensing agents that have heterogeneous movement and sensing capabilities, or when the number of sensing agents and tracking targets increase. Some applications of this field are environmental monitoring[1], search and rescue [2], cooperative localization and mapping [3] and object pose estimation [4]. Perception and control are both essential for robots to perform these tasks autonomously. These problems are very widely studied in the robotics literature, but often considered as separate problems. However robot perception is often done by vision-based sensors or cameras. The accuracy of these cameras is strongly affected by their relative position and motion. A camera provides better quality images if it is in close proximity of a target and image quality is negatively affected by motion blur due to the motion of the measuring platform. Additionally, successfully achieving a motion objective requires the sensing agent to have a good quality estimate of its own state. Hence it makes sense to design a control method that is able to consider a joint perception and motion objective.

When controlling the sensing agents it is critical they are able to honor constraints on both their perception and actuator domains, as not doing so may lead to inability to fulfill the mission or create hazardous situations to people around. Recently *Model Predictive Control* (MPC) has gained popularity in aerial robotics. MPC is a control policy that tries to find the optimal control inputs for the controlled system that drive the system states to some objective by minimizing a cost function, over a receding horizon, whilst satisfying a set of constraints. The structure of this cost function can be very generic which allows MPC to be employed for a wide variety of control objectives. The ability of MPC to plan ahead in time and consider constraints on the different perception and actuator domains for multiple heterogeneous agents make it well suited for the problem of cooperative target tracking. *Nonlinear Model Predictive Control* (NMPC) is a variant of MPC that uses a nonlinear system model instead of a linear one to predict future system states. Most real-world systems are inherently nonlinear, but since computing the optimal control for nonlinear models is harder, NMPC trades off increased prediction accuracy for an increase in computational complexity. UAVs are nonlinear systems so NMPC theory is applicable to the control of these platforms.

The use of heterogeneous agents increases the difficulty of finding the correct control policy, as compared to the case where all agents have uniform capabilities. However when correctly implemented, the controller able to deal with heterogeneous agents will outperform a method only capable to handle homogeneous agents using a similar level of equipment. Consider a team of UAV agents tasked to measure the position of a static ball in 2D space. A strategy could be to equip each agent with a sensor capable of measuring both positional states of the target. A more efficient strategy would be to equip each agent with a sensor capable of measuring either the X or Y target position and fuse these measurements into a single estimate of the target state.

This fusion of multiple measurements into a single state estimate is the canonical state estimation problem and is typically solved using linear quadratic estimation or kalman filtering. A *Kalman Filter* (KF) is a method that provides a probability distribution of an unknown variable, conditioned on previous measurements and their measurement accuracy. By considering all past target state measurements the estimation can be improved, and when all errors are assumed Gaussian the KF can be proven to be the optimal linear estimator.

This reports presents an attempt to combine the above-mentioned systems to integrate a perception objective in a UAV controller. A more complete problem statement is given below.

1.2 Problem Statement

1.2.1 Main Research Question

The main research question of this thesis is:

How can we use Nonlinear Model Predictive Control to cooperatively track a target by minimizing the fused target estimation uncertainty of a heterogeneous sensing team? (1.1)

1.2.2 Sub Questions

To investigate the main research question multiple sub questions are presented and detailed below. The first is:

How do I fuse multiple intermittent target measurements into a single state estimate? (1.2)

Any system which has multiple agents that provide target measurements needs form a consensus on the target state estimate. How do the different state estimation methods compare?

What is the advantage of MPC over other cooperative tracking methods? (1.3)

Multiple different control strategies for cooperative tracking exist. What are their strengths and weaknesses of each method and why use MPC?

What is the advantage of using nonlinear motion models? (1.4)

Nonlinear MPC extends the MPC method to be able to consider nonlinear motion models at the cost of increased complexity. What is the advantage of using such models and is the trade-off worth it?

How do the different NMPC perception objectives compare? (1.5)

NMPC find the control inputs that optimize a cost function over a finite horizon subject to constraints. What perception objectives can be added to this cost function that results in cooperative tracking behavior of the sensing agents? How do these objectives compare?

What is the stability of my proposed NMPC method? (1.6)

Receding horizon predictive control using nonlinear models and constraints is a nonlinear feedback control method. This implies there is a risk of the resulting behavior becoming unstable. A common critique of NMPC methods is the lack of stability guarantees. For linear MPC results from Linear Quadratic optimal control could be used. This raises the question of what can be said about the stability of this nonlinear method?

How is the system affected when the assumptions on perfect modelling, localization and communication are relaxed? (1.7)

When constructing the system it is assumed the agents have perfect knowledge of all sensing agent and target models, the location of other sensing agents and perfect communication to the central controller without delay. These assumptions can be very coarse and unfeasible in real physical systems. What happens when these assumptions are violated?

How can the system be adapted to function in a distributed manner? (1.8)

In this work a centralized controller will be constructed. To improve its scalability with the number of sensing agents a distributed version run locally on each agent could be considered. What challenges arise during the conversion to a distributed method?

1.3 Contribution

This work makes the following contributions:

1) A literature review on Cooperative Target Tracking.

2) The extension of an existing NMPC with a KF covariance perception objective to drive a team of heterogeneous agents into a configuration that minimizes their collective target estimation covariance.

3) A C++ implementation of an Intermittent Kalman filter in the Genom framework

1.4 Structure of Report

The next chapter contains a small theoretical background of Kalman Filters and Model Predictive Control. Chapter 3 is a literature review with a discussion of relevant theory and state of the art. The fourth chapter contains a detailed description of the method. Chapter 5 presents the full system architecture. Experimental Setup and results to validate the method are given in chapter 6. Chapter 6 also contains a discussion of the results. The report is concluded in chapter 7.

2 Background

This chapter contains a summary of prerequisite knowledge on Kalman Filters and Model Predictive Control.

2.1 State Estimation

State estimation deals with finding the optimal estimate of an unknown target state given a set of past imperfect measurements of this target. Figure 2.1 shows the concept of a state estimation scheme. As the concept of states are very general, state estimation is widely applied to a large number of different fields like chemical engineering, automotive and weather prediction. In control theory a state observer is often employed to estimate the unknown states of a plant using its output. A crude state estimation method could be a linear extrapolator that takes the last state measurement and assumes it to be constant over time. Other linear extrapolation methods exists but, under the assumption of state measurements perturbed by Gaussian noise, the Kalman Filter is proven to be the best possible linear estimator.

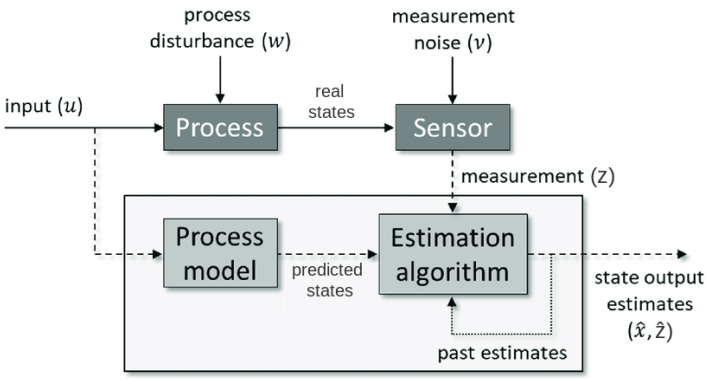
2.1.1 Kalman Filters

A commonly used and effective state estimation method is the Kalman Filter. The Kalman Filter is a recursive filter that uses measurements perturbed by Gaussian noise to estimate the internal state of a process of interest. Table 2.1.1 shows the used symbols. $\mathbf{x}$, $\mathbf{u}$ and $\mathbf{z}$ are all vectors of size N where N is the number of states to track. All other bold symbols are square matrices of size $N \times N$. Subscript indicate timing, $\hat{\mathbf{x}}_{k|k-1}$ is the predicted state at time k, using measurements up until time $k-1$. The KF state consists of $\mathbf{x}$ and $\mathbf{P}$.

Table 2.1: Symbols used in this section

Symbol	Definition
$\mathbf{x}$	Internal (hidden) process state
$\mathbf{z}$	State measurement
$\mathbf{H}$	Measurement model
$\mathbf{R}$	Measurement covariance
$\hat{\mathbf{x}}$	State estimate
$\mathbf{P}$	State estimate covariance
$\mathbf{A}$	State transition model
$\mathbf{B}$	Input response model
$\mathbf{u}$	Input
$\mathbf{Q}$	Process Noise
$\mathbf{K}$	Kalman gain
$\hat{\mathbf{y}}$	Innovation
$\mathbf{S}$	Innovation covariance

Figure 2.1: Generic state estimation scheme, adapted from [5]



It is assumed the internal state at the discrete time instant k is linearly related to the state at $k-1$, with additional zero-mean Gaussian noise terms $\mathbf{w}_k$ and $\mathbf{v}_k$.

$$\mathbf{x}_{k|k-1} = \mathbf{A}_k \mathbf{x}_{k-1|k-1} + \mathbf{B}_k \mathbf{u}_k + \mathbf{w}_k \tag{2.1}$$

$$\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k \tag{2.2}$$

$$\mathbf{w}_k \sim \mathcal{N}(0, \mathbf{Q}_k) \tag{2.3}$$

$$\mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R}_k) \tag{2.4}$$

Although not required, the KF filtering steps are commonly divided into two alternating phases, the predict and update phase. During the prediction phase the new state estimate $\hat{\mathbf{x}}_{k|k-1}$ and covariance $\mathbf{P}_{k|k-1}$ are predicted using the previous state estimate $\hat{\mathbf{x}}_{k-1|k-1}$ and covariance $\mathbf{P}_{k-1|k-1}$.

Predict

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{A}_k\hat{\mathbf{x}}_{k-1|k-1} + \mathbf{B}_k\mathbf{u}_k \tag{2.5}$$

$$\mathbf{P}_{k|k-1} = \mathbf{A}_k\mathbf{P}_{k-1|k-1}\mathbf{A}_k^T + \mathbf{Q}_k \tag{2.6}$$

Then the new measurements $\mathbf{z}_k$ are integrated into the state estimate $\hat{\mathbf{x}}_{k|k}$ and its covariance $\mathbf{P}_{k|k}$. First the innovation $\hat{\mathbf{y}}_k$ and the innovation covariance $\mathbf{S}_k$ are calculated. These reflect the residual in the state after prediction, the difference between the predicted and measured state. Next the Kalman Gain is found. The Kalman Gain is a measure of the quality of the measurement relative to the prediction and takes values ranging from 0 to 1. When no measurements are present the calculation of the innovation and Kalman gain can be skipped and the state prediction is used as filter output.

Update

$$\hat{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1} \tag{2.7}$$

$$\mathbf{S}_k = \mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^T + \mathbf{R}_k \tag{2.8}$$

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^T\mathbf{S}_k^{-1} \tag{2.9}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\hat{\mathbf{y}}_k \tag{2.10}$$

$$\mathbf{P}_{k|k} = (I - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_{k|k-1} \tag{2.11}$$

The continuous time version of the KF is also known as the Kalman-Bucy filter. Optimality of the Kalman Filter assumes Gaussian noise and linear models. In some situations the linear model is an insufficient model and non-linear models are required. Two common KF extensions are used in this scenario; the Extended Kalman Filter (EKF) and Unscented Kalman Filter (UKF). These and other nonlinear state estimation techniques, including learning based approaches, are outside the scope of this background. The interested reader is refereed to [5].

2.2 Model Predictive Control

Model Predictive Control (MPC) is a control paradigm of increasing popularity that consists of a range of control methods that obtain the optimal control inputs for a system. MPC distinguishes itself from the other optimal control methods by using explicit models of the system to predict the systems output over a future horizon and applying a receding strategy that displaces this horizon towards the future at each timestep. MPC is considered a mature technique for slow linear systems like those found in the process industry. Recent developments have allowed for successful application of MPC to faster nonlinear systems like UAVs. Figure 2.2 shows a basic MPC control block diagram, figure 2.3 shows an example MPC control response $u(t)$ and system output $y(t)$ over the planning horizon.

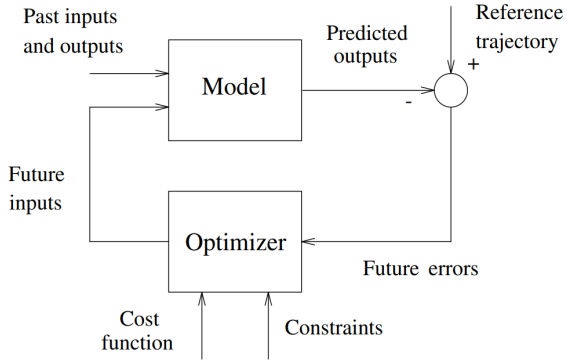


Figure 2.2: Basic MPC structure, taken from [6]

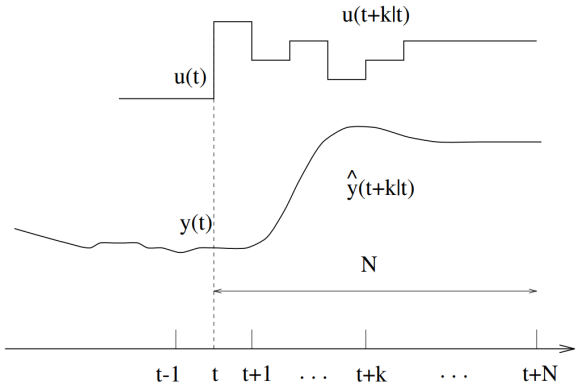


Figure 2.3: MPC strategy, taken from [6]

An example Model Predictive Control problem is shown in equation 2.12. Here is asked to find the system inputs $\mathbf{u}$ that minimize the cost function J for all N discrete time steps k over a receding horizon of length T.

$$\min_{u_0 \dots u_{N-1}} J(\mathbf{y},\mathbf{u}) := \sum_{k=0}^{N} w_{y_i}(r_i - y_i)^2 + \sum_{k=0}^{N} w_{u_i}(u_i)^2 \tag{2.12}$$

$$\text{s.t.} \quad \mathbf{x}_0 = \mathbf{x}(t) \tag{2.13}$$

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k) \tag{2.14}$$

$$\mathbf{y}_{k+1} = g(\mathbf{x}_k) \tag{2.15}$$

$$\underline{\mathbf{y}_k} \leq \mathbf{y}_k \leq \overline{\mathbf{y}}_k \tag{2.16}$$

Where x are the system states, y is the system output, r the output reference and w are weights modulating the relative importance of the two objectives. Equation 2.13 to 2.16 showcase another useful property of MPC, its ability to take various constraint in account during the optimization.

Constraints are present in any real system, actuators have a limited field of action, pipes or wires have limited flow rates and sensors have limited FOV. In order for a control system to be effective and able to operate near these system boundaries it is necessary these constraints are considered. MPC, like many predictive control strategies, intrinsically takes these constraints into account during optimization. In the above optimization example 4 common constraints are considered. Equation 2.13 to 2.15 constrain the system states and output to evolve according to the system state space model. 2.16 represents a band constraint on the system output. Other common constraints are limits on the system input $\mathbf{u}_k$ and its time rate of change $\mathbf{u}_k - \mathbf{u}_{k-1}$.

By adding these constraints the optimization becomes increasingly harder as the objective function becomes more complex. MPC optimization can usually not be done explicitly and instead rely on numerical methods such as Quadratic Programming (QP) to obtain a solution. qpOASES [7] is an effective QP solver for MPC algorithms.

3 Literature Review

Since the main theme of this thesis is the use of a perception objective in a NMPC to cooperatively track a target, a few relevant sets of literature are identified and summarized below. The first is the field of optimal sensor placement as a precursor to cooperative target tracking. This is extended to moving sensors in the next section. An attempt to generalize challenges like cooperative target tracking has produced research on the *Active Information Acquisition* (AIA) problem. The following section on Model Predictive Control summarizes research on the application of MPC to the AIA problem. The last section contains a review of existing works using nonlinear dynamical models to control the sensing agents.

First a small summary of different state estimation methods is presented.

3.1 State Estimation and Sensor Fusion

All target tracking methods require a state estimation scheme to reach a consensus on the current state of the target. If tracking is done by multiple agents cooperating some sensor fusion method is additionally required to combine the information each of the agents gather in some optimal way. In this section a few common state estimation methods are shortly summarized.

3.1.1 Bayesian Filtering

Recursive Bayesian Estimation or Bayes filter is a probabilistic approach for the recursive estimation of a probability density filter (pdf) of a random variable over time. It uses an observation and motion model together with noisy measurements to track the evolution of the target variable over time. The target variable is modeled as a hidden Markov model, assuming, given the previous states, the current state is independent of earlier ones and the current measurement is dependent only upon the current state. In order to implement the filter in practice often additional assumptions are required.

3.1.2 Kalman Filter

First derived in 1960 the Kalman filter is a Bayes filter which assumes the pdf of all variables are Gaussian. The filter reduces to two sequential steps, the predict and update step. This filter is explained in more detail in section 2.1. The Kalman filter is well known and used in various settings and a vast range of applications. The Kalman filter is proven to be the optimal linear quadratic estimator in the sense it yields the smallest mean square estimation error. The assumption of Gaussian pdfs only holds for linear motion and observation models an will merely approximate the Bayes filter

3.1.3 Nonlinear Kalman Filters

Unfortunately most real systems are non-linear. The Extended Kalman filter (EKF) has been developed in an attempt to generalize the KF to nonlinear systems. The EKF exploits a linearisation step and applies the Kalman filtering scheme after linearising the nonlinear motion models around the current operating point. A requirement for this is that the nonlinear models must be differentiable.

Although the EKF is extremely widely used, for instance in GPS. This linearisation step is the origin of many difficulties including divergence and tuning difficulties. Extensions of the EKF exists that attempt to alleviate this problem. One of these is the Unscented KF (UKF). This version should be considered when tracking states of a highly nonlinear system or when derivatives do not exist.

3.1.4 Particle Filter

Particle filtering is a Monte Carlo method to represent the posterior distribution of a target random variable. Like the EKF it is an approximation of a Bayes filter. The filter represents the pdf of the random variable using

a finite set of weighted samples or particles, where the weight represents the probability of that particle being samples from the pdf. At each iteration a random set of samples is drawn from the current estimate of the pdf. These are propagated using the (possibly nonlinear) target motion models. When an observation is made all particle weights are modulated based on how well the observed data correlates with the hypotheses the particle represents. When the weight of a particle tends to zero it is culled from the set as it badly models the current pdf and is no longer of interest.

Similarly to the Kalman type filters the particle filter returns a state estimate and covariance. Unlike the KF it is able to deal with multimodal distributions. The particle filter should not be used in large scale systems due to the requirement of a sufficiently large size of the random sampling set.

3.1.5 Probability Hypothesis Density Filter

Another Bayes filter type able to track multiple targets over time is the Probability Hypotheses Density (PHD) filter. A PHD is a first order statistical moment of a random finite set, a known quantity for many of the popular distributions. The PHD filter models the multiple target states as a Poisson multi-object process with intensity equal to the PHD of the random target variable. This yields the optimal Poisson approximation of the assumed target variable distribution, optimal in the sense it minimizes a popular probability distance metric, the Kullback-Leibler divergence [8]. The PHD filter is able to give a measure of the amount of targets but cannot do data association. A big advantage of the PHD filter is its capability to handle appearing and disappearing objects. It performs well in relatively simple scenarios with high probability of detection and low clutter. The PHD filter assumes all targets are independently Poisson distributed, in reality this is often a very coarse approximation and can reduce the accuracy of the filter.

3.2 Optimal Sensor Placement

A problem that is closely related to cooperative tracking is optimal sensor placement. This field deals with the question how to position sensors in a sensor network such that measurements of a certain quantity of interest are optimal. This sensor placement challenge can be reasonably said to be a simplification of cooperative tracking as it has a similar goal but assumes stationary instead of dynamic sensors. A large body of literature exists on this problem as it has been extensively studied for at least three decades [9]. As the optimal placement of sensors is applicable to a wide range of disciplines and applications, research from many different fields exists. I will highlight a few methods to provide some intuition.

Santpal Singh Dhillon and Krishnendu Chakrabarty present a resource-bounded optimization framework for sensor resource management under the constraints of sufficient grid coverage of the sensor field [10]. As a simplification they assume the observation area or sensor field is made up of grid points whose granularity depends on the desired solution accuracy. They further assume a probabilistic sensor detection model where the probability of detection decreases exponentially with the distance of the sensor to the target. The rate of this decrease models sensor quality, the detection probability is 1 when sensor and target coincide. Between every two grid points in the sensor field two detection probabilities are assigned, one for each sensor detecting the target at the other grid point. Trivially these two probabilities are the same, but asymmetry in these values can occur when obstacles are present in the sensor field. A height difference can for instance cause the elevated sensor to be able to more easily detect a target in a lower point. A sensor placed at the lower point is much less able to detect a target at the elevated position. These are very general assumptions and they can be readily applied to many different applications including the tracking of a target using cameras. Two different algorithms are presented that take a sensor grid field and the corresponding set of detection probabilities and determine the minimal number of sensors and their locations such that every grid point is covered with a minimum confidence level. Both algorithms can be solved in polynomial time and, under the above assumptions and independently of the detection model, are able to significantly outperform uniform sensor placement. The first method attempts to maximize the average coverage of the grid points by placing sensors on the sensing grid and recalculating the average detection probability until

a threshold is satisfied or the maximum number of sensors is exceeded. The second method attempts to maximize the coverage of the grid point that is covered least effectively by instead placing sensors at the location of minimal detection probability. The strengths of this method are its relative simplicity and quick solving time for smaller grids. As it uses very generic models the methods are able to generalize well to different fields. However this might also cause the method to be insufficiently able to deal with complexities arising from specific fields, such as camera field of view. It is assumed camera quality can be modeled by a single parameter. The method assumes the sensor grid is 2 dimensional and quickly becomes intractable for large sensor grids.

A less generic and more complex method is proposed by S.Y. Chen and Y.F. Li [4]. They attempt to find the optimal sequence of stereo camera viewpoints that enable an accurate 3D reconstruction of an object and its features in a time efficient manner. This work also discusses other functions, with the initialization of the viewpoint number and their distribution and viewpoint evolution being most relevant to this work. Being an optimization method, first a viewpoint number is estimated based on object size and camera field of view and these are uniformly distributed on a sphere surrounding the object. These initial sensor viewpoints are mapped onto a sensor placement graph where the vertices represent viewpoints and the edges encode the shortest collision-free paths. Some viewpoints will be not contain new information. In the next step these redundant graph vertices are pruned to optimize the viewpoint planning using a Hierarchical Genetic Algorithm (HGA). HGA are a subclass of evolutionary algorithm that use biologically inspired operations like mutation to iteratively find solutions to optimization problems. It requires the solution to be represented using a genetic model, this is done by designing a chromosome with the viewpoints as parametric genes and the topology as control genes. After initialization the fitness of each viewpoint is assessed using a fitness evaluation function. The viewpoint fitness level is then used as selection probability for the propagation to a new generation using a crossover and mutation method. At each generation constraints on visibility, field of view, viewing angle, focus and occlusion are tested and the viewpoint plan is rejected if any are violated. After obtaining a sufficiently fit sensor placement graph the shortest path through all viewpoints is computed using Christofides algorithm. The method has been experimentally validated and tested using a real robot. The ability to take into account constraints on camera FOV is an advantage. However knowledge on the geometry of the object under investigation is required a priori. Since it is a optimization based method no guarantees are given on the quality of the solution. The method can be slow to compute, during testing the authors report computation times ranging from 5 minutes to 15 hours. But since this is on old hardware and the same viewpoint plan can be used for multiple similar objects the method could still be practical today.

For further information on the optimal sensor placement of static sensors the interested reader is addressed to a survey spanning multiple disciplines [11]. When sensors are able to move and reposition to allow for better target state measurements, the complexity increases.

3.3 Cooperative Target Tracking

Due to advancements in sciences such as microelectronics and battery technologies, it is increasingly effective to replace the stationary sensors with mobile ones. Literature on this *cooperative target tracking* problem is more recent and similarly to sensor placement, it is studied from different disciplines. Where stationary sensors might fail to track a dynamic target, mobile sensors can adjust their position and account for target movement. This new challenge is sometimes split into two sub problems, motion planning and sensor control. Motion planning deals with the calculation of the optimal sensing positions at each point in time and sensor control is the design of an appropriate control policy to drive the sensors to these positions.

Nikolay Atanasov describes the problem of reducing uncertainty about a physical process by designing optimal sensing trajectories for a team of mobile sensing agents as an Active Information Acquisition problem [12]. They show that, under the assumption of linear Gaussian sensing and mobility models, the classical

principle of separation between the estimation and control holds. This implies with the use of a state estimation scheme the problem is reduced to a deterministic optimal control problem for which many solving methods exist. Both greedy and non-greedy solutions are discussed. Where non-greedy (or non-myopic) methods plan over a time horizon, greedy methods only consider the optimal move for the next time instant. They show greedy methods are have attractive scaling properties w.r.t the optimization time horizon length and number of samples. The greedy methods have the disadvantage of having no optimality guarantees and it is shown by example that greedy planning is worse, even for problems with static targets and a small optimization horizon. A discussion on the choice of perception objective is included. Target estimation performance can be quantified in multiple different ways, including T-optimality ($\mathrm{tr}(P_k)$), D-optimality($\det(P_k)$) and E-optimality ($\lambda_{\min}(P_k)$). They present evidence the conditional entropy or D-optimality is preferable in case of Gaussian variables. A non-greedy method is presented in more detail in [13]. In this work a MPC based decentralized approach is used to compute trajectories for four differential drive robots to perform Simultaneous Localization And Mapping (SLAM). They show their method can be applied to SLAM and can, with approximation strategies, achieve linear complexity in the number of sensors.

Next sections details a number of different approaches to the cooperative target tracking problem.

3.3.1 Gradient Methods

Gradient based methods define a cost function based on some target perception goal and use a gradient feedback control law to drive the agents to states where target information is maximized.

In 2006, Timothy H. Chung et al presented a paper on a decentralized motion planning algorithm for the distributed sensing of a noisy dynamical process by multiple cooperating mobile sensor agents in 2D space [14]. Under the specific assumptions of a range-bearing sensor with linear measurement model and a linear target model perturbed by Gaussian noise, a cost function is constructed representing the sensing quality. Two iterations of this cost functions are discussed, the first is a simple scheme based on the fusion of local target estimates, the second uses a Kalman filter run locally on each agent. At each timestep the local target estimate and covariance matrix are updated. A period of time is allotted to allow the communication of these quantities to neighbors and all received information is fused into a global state estimate and covariance. Next the cost function gradient is calculated locally to determine the next optimal sensing location and the agent is moved there. The effect of imperfect communication between the sensing agents is also investigated. A decrease in performance is observed when communication is restricted, a total communication restriction causes agents to act independently, losing all benefit of cooperative tracking. Their gradient control law can also be used to control a heterogeneous sensing team. This paper showcases a relatively simple way to achieve good results on decentralized motion planning. Interestingly when using their method the agents express a teamwork behavior and are able to coordinate together to maximize the overall sensing quality. As their method is designed to work for 2D space it is necessary to question how well it generalizes to 3D space, and if it does, how the performance and time complexity of the method is affected. Another consideration is their sensor assignment method. They use a simple Round robin sensor assignment strategy which they mention is almost never optimal. Finally a known issue of gradient methods is their tendency to get stuck on local minima, and for a complex system the fused estimate covariance matrix might be quite irregular.

A similar method was used by Peng Yang et al. [15]. They consider the problem of maintaining a target state estimate for multiple mobile sensors and move each agent so as to maximize the expected information from its sensor relative to the current estimate uncertainty. They construct a dynamic average consensus estimator that requires an agent to only communicate to its one-hop neighbors to maintain a local estimate of the global target state. Both range-bearing and range-only sensors with a linear measurement model and a diagonal measurement noise covariance in local frame are considered. Similarly to the above work two approaches are tried, a one-time measurement and Kalman filter approach. With the assumption of perfect sensor localization they show the performance of their distributed method yields cooperative behavior for

both approaches and approximates that of a centralized approach after an initial transient. It can handle collaboration of range-bearing sensors with range-only sensors and can be used to track multiple targets assuming each sensor is able to measure and distinguish all targets, avoiding the sensor assignment issue altogether. The method scales well with increasing number of sensing agents, requires limited communication between agents and is able to avoid the single failure point problem of centralized solutions. Similar to the previous work only 2 spatial dimensions are considered and cannot be trivially adapted to 3. The dynamic average consensus requires multiple gains to tune. Another consideration is the requirement of each agent to know all target and sensing agent model parameters.

3.3.2 Search and Sampling Methods

Other types of methods exist, like the Monte Carlo Tree Search method presented by Ke Sun et al. [16]. Addressing the related problem of stochastic motion planning, specifically how to navigate a mobile robot equipped with a LIDAR, they model the full system as a Partially Observable Markov Decision Process. This model assumes system dynamics are governed by a Markov Decision Process, but the system state can not be directly observed and must be acquired by the fusion of noisy sensor data. In a Markov Decision Process sensing agent states are mapped to actions by a control policy that maximizes a reward function. A POMDP control policy instead maps observations or *beliefs* to actions. Using this type of system model allows them to find a control policy for the sensing agents online. They show their method improves upon existing POMDP methods, successfully navigating the LIDAR robot to achieving a higher total reward and success rate in completing the navigational task.

This method requires both the state space and actions space to be discretized. In the above example the action space is a set containing only 6 possible values, allowing only for very rough control of the platform linear and rotational velocity and speed. This problem becomes much more pronounced when considering higher dimensional state and actions spaces, quickly becoming intractable to compute. If a small collection of actions is sufficient to competently describe the sensing agent action possibilities and the state or belief space can be kept small the above works show this method can have definite advantages. Many different existing strategies can be leveraged to find PODMP control policies and for finite-horizon POMDPs the optimal value functions is proven to be piecewise-linear and convex, allowing dynamic programming techniques to find optimal control policies.

In 2018 Brent Schlotfeldt et al. publish a paper on reducing uncertainty about a physical process by designing sensing trajectories for a team of robots [17]. In their paper a search-based planning method for active information gathering is developed which prunes uninformative trajectories from the search space while providing suboptimality guarantees and decentralizes the planning across multiple robots via coordinate descent. They later improve upon the robustness of their method, increasing resilience to communication failures in [18].

In a recent work Xiaoyi Cai et al. [19] optimize the trade-off between information gain and energy cost by augmenting the objective function with an energy term. Their method can plan trajectories for a heterogeneous team of sensor-equipped robots to reduce uncertainty about a dynamical process, using a distributed planning approach based on local search. The perception objective is posed as a minimization of the determinant of the mutual information between target states and observations of these target states.

in 2019 B. Schlotfeldt et al. present a method to plan trajectories for robots equipped with sensors to track a target by formulating the problem as a deterministic planning problem and computing the optimal solution using the A* search algorithm [20]. To effectively use A* they derive a heuristic function based on the upper bounds of a Kalman filter estimate covariance matrix and prove it is an admissible heuristic. They show their method is able to find a trajectory that is optimal in the sense it quickly minimizes the covariance of the target state estimates.

This method is able to compute the sensing trajectories for multiple agents that quickly minimize the estimation covariance for multiple targets in a 2D space with simple obstacles. Although computation is done

offline, admissible sensing trajectories for the above scenario can be found in the order of seconds. In order to use search algorithms the path must first be represented using a tree like data structure. This places limits on both the system states and action space of the sensing agents, only allowing very coarse control of the sensing agents between each step. Continuous time sensing agent motion models can be considered but will cause exponential growth in the number of nodes and requires additional modifications to remain tractable.

Sampling based methods like RRT and RRT* can be used in an attempt to relieve some of the disadvantages of search based methods. One of these methods is presented by G. Hollinger and G. Sukhatme in their 2014 work [21]. They propose a sampling based motion planning algorithm for generating maximally informative trajectories for mobile robots subject to motion constraints.

Pratap Tokekar et al. study the challenge of tracking the most number of mobile targets and accurately estimating their positions using a team of quadrotor sensing agent [22]. They prove the two objectives conflict and pose limits on the number of tracked targets as a function of a desired estimation accuracy. They formulate the problem as a Maximum Group Coverage problem and propose a greedy approximating algorithm to select the best trajectories from a set of possible trajectories generated by a sampling based method, best in the sense either the estimation quality or number of tracked targets is maximized whilst a minimum bound on the other is guaranteed. They show their method is able to coordinate a team of quadrotor agents to cooperatively track a number of moving targets.

In a paper by Jun Chen and Philip Dames a different method is presented. Four distributed algorithms are introduced to enable a team of robots to safely search and track a time-varying number of targets. For state estimation a distributed Probability Hypothesis Density filter is maintained across the sensing team. Sensing space is divided among the sensing agents based on their self-localization covariance using Voronoi partitioning and robots are driven to regions of high uncertainty and avoid collision [23] by feedback control toward the weighted centroid of their Voronoi cells. Their method is able to track a large time-varying number of targets across a specified area of 2d space using a large number of sensing agents whilst guaranteeing collision avoidance. The sensing team has imperfect self-localization but the performance of the method approximates that of a centralized one.

3.3.3 Linear Quadratic Gaussian Control

A closely related problem accompanied by a large body of literature is the Linear Quadratic Gaussian (LQG) control problem of determining a state feedback control law that minimizes the expected value of a quadratic cost function for a continuous or discrete time linear system driven by additive white noise. A classical result is the separation principle which states LQG reduces to a combination of a Kalman filter and a Linear Quadratic Regulator (LQR).

Philipp Foehn and Davide Scaramuzza propose such a LQR for unified control of a micro aerial vehicles (MAV) rotational and translation states [24]. Their cascaded method linearises the MAV dynamics around its current state and is able to compute feasible trajectories onboard an ARM processor at a minimum rate of 10Hz using only onboard visual inertial odometry (VIO). They show their method remains stable whilst hovering or tracking a trajectory under abrupt reference jumps and disturbances.

In a tangentially related work on system identification A. Iannelli and R. Smith use a LQR scheme to design an optimal state feedback control policy which seeks to balance the exploitation of an uncertain plant with information gaining moves that reduce the uncertainty in the plant dynamics [25]. Although this work deals with system identification it showcases the applicability of LQR to an information gathering objective.

All previous cooperative target tracking methods are *greedy* methods, during optimization only the next time instant is considered. A Linear quadratic regulator has the advantage of being a *non-greedy* or *non-myopic* method, LQR optimizes over an infinite or finite time horizon. This allows the method to design control inputs that are optimal in the sense they minimize the total cost over time.

As the name implies, a LQR is designed to control linear systems. This is an important consideration as controlling a nonlinear system using LQR is only possible by linearising the system dynamics, reducing optimality and accuracy of the method. LQR in its original state also does not consider input and state constraints. Receding Horizon Control techniques like Model Predictive Control have been specifically developed to be able to deal with these constraints.

3.4 Model Predictive Control

In one of the precursors to this work, Baljinder Sign Bal [26], identified the fitness of MPC to the cooperative tracking problem. He proposed a centralized method for two heterogeneous robots to cooperatively track a moving target using MPC to minimize the entropy of the estimation of the target's state provided by an intermittent measurement Kalman filter (IKF) adapted from [27]. In a simulation experiment a 2D linear ground and aerial agent dynamics with limited Field of View (FOV) are required to cooperate to track a target with linear dynamics and an unknown input. They show the addition of a IKF covariance perception objective yields cooperative tracking behavior and minimizes the target estimation covariance. Implemented in Matlab, his method is not quite able to run real time. This combined with the 2D and linear agent dynamics assumptions indicate promising further research directions.

In a similar work, Chang Liu and Karl Hendrick present a MPC based path planning approach for a ground mobile robot to autonomously search and track a moving target [28]. They add a similar perception objective to the MPC cost function based on the trace of the IKF estimation covariance. Using simulation experiments they show their method is able to generate trajectories for a robot to search and track a target moving through 2d space.

All previous works consider the agent dynamics to be linear, an assumption that becomes less effective when more complex and performance intensive trajectories are required due to the increased significance of the unmodeled aerodynamic effects.

3.4.1 Nonlinear Model Predictive Control

In [29], Kamel et al. compare a Nonlinear Model Predictive Control (NMPC) scheme from their previous work to a linear MPC (LMPC), by evaluating hovering performance, step response, and aggressive trajectory tracking of a micro aerial vehicle under both nominal conditions and external wind disturbances. They find the NMPC method performs similarly on hover tasks around the LMPC operating point but has slightly better disturbance rejection capacity when 11m/s wind is present. They show during a step-response experiment the NMPC method has much better performance with a 40% quicker rise time for the same overshoot. They attribute this to the superior ability of the NMPC to operate the actuators around their saturation point as compared to the LMPC method. During aggressive trajectory tracking the NMPC achieves a 30% better RMS tracking error compared to the LMPC. Interestingly the NMPC method achieves an average computation time of 0.45s, A 5 times increase over the LMPC method which they attribute to the efficacy of the real time iteration scheme implemented in the NMPC.

In [30], Davide Falanga et al. present the a real-time perception-aware model predictive control framework for quadrotors. It is able to satisfy the system dynamics and require control inputs within the limits of the platform, whilst optimizing perception objectives for sensing by maximizing the visibility of a point of interest and minimizing its velocity in the image plane. The perception objective achieved by the addition of two cost terms that depends on the distance from center and velocity of the feature in the image plane.

G. Li et al present the first NMPC method for trajectory tracking with quadrotors with a suspended payload [31]. Their method considers constraints on nonlinear system dynamics and actuator constraints and is able to fly a quadrotor up to 4m/s whilst minimizing the image plane velocity of the payload in a downward facing camera. To avoid instability they enforce cable tautness using a novel constraint on cable tension. The perception objective is implemented by constraining the distance between the payload and the center of the camera FOV.

Often the MPC is used to plan trajectories for the sensing agents and it is assumed a low-level controller is present to translate these into actuator inputs. When using this type of cascaded control scheme the physical actuator constraints of the platform must first be transformed to constraints on the MPC output trajectory. D. Bicego et al. argue any usage of such fictitious constraints results in a reduced control performance w.r.t the real dynamic potential of the robot [32] They propose an online NMPC method for multi-rotor aerial systems with arbitrarily positioned and oriented rotors which can simultaneously address local reference trajectory planning and tracking problems. Unlike most other works they use a unified nonlinear model that attempts to capture the whole robot dynamics and explicitly consider real physical actuator dynamics and limits. They present an extensive experimental validation of their method and show it is able to control multiple different platforms close to their actuation limits. They show their method is able to handle a rotor failure and can exploit a Multi Directional Thrust platforms capability to modulate the propeller tilt angles, yielding a very energy efficient method.

Martin Jacquet et al. extend this work by including a perception objective in the NMPC in order to maintain visibility of a feature of interest [33]. They propose an onboard NMPC method capable of directly computing the torque inputs of generic multi-rotors to comply with a reference trajectory, modulating through user-defined weights the less important states to comply with a perception objective. Simulation and hardware experiments show their method is able to track a target to comply with its visibility constraints and exploit the full action span of both standard quadrotors and fully-actuated platforms whilst obeying actuator constraints.

Guillem Torrente et al. use a different strategy to increase the efficacy of their linear motion model by augmenting it with a grouped aerodynamics forces term approximated using a Gaussian Process (GP) [34]. The GP attempts to correct for the unmodeled non-linear aerodynamic effects which are especially dominant during high speed aggressive maneuvers. To do this a learning problem is set up to find a mapping from body frame velocities to body frame acceleration disturbances. They prove their augmented linear model is able to approximate a significant part of the dynamics (70% reduced tracking error) normally not captured by a linear model whilst being efficient to evaluate.

These papers all assume perfect communication between robot and controller. Barbara Barros Carlos et al. propose an efficient position controller for quadrotors based on real-time NMPC with time-delay compensation and bounds enforcement on the actuators [35]. To compensate for the time-delay a state predictor is used to predict the current robot state by propagating past states through system dynamics. They show the system is able to run in real-time on an offboard computer.

When multiple agents must cooperate to track a target they necessarily need information on the states of the other agents. When it can no longer be assumes this information is present, such as when designing a distributed control scheme, methods to predict the other agents states must be explored. This can be done using a state estimation scheme similar to the one used for the target. H. Zhu et al present a recurrent neural network based method [36] that attempts to predict the states of other agents to avoid collision, using current and past local observations. They show when using a MPC to do local path planning their method is able to approximate the performance of a centralized method. This method has the advantage of requiring less communication between agents, scaling better with an increasing number of sensing agents.

4 Method

This section contains a description of the models and tools used for the remainder of this work. How these are applied to yield insight into the questions posed in section 1.2 is summarized in section 5.

4.1 Sensing Agent Dynamics

This section describes the sensing agent motion model used by the MPC to predict future agent states. In order to preserve the generality of the method sensing agents are all assumed to be Generically-tilted Multirotors (GTM), taken from [33] and derived by Giulia Michieletto et al in [37]. For completeness and convenience they are repeated below.

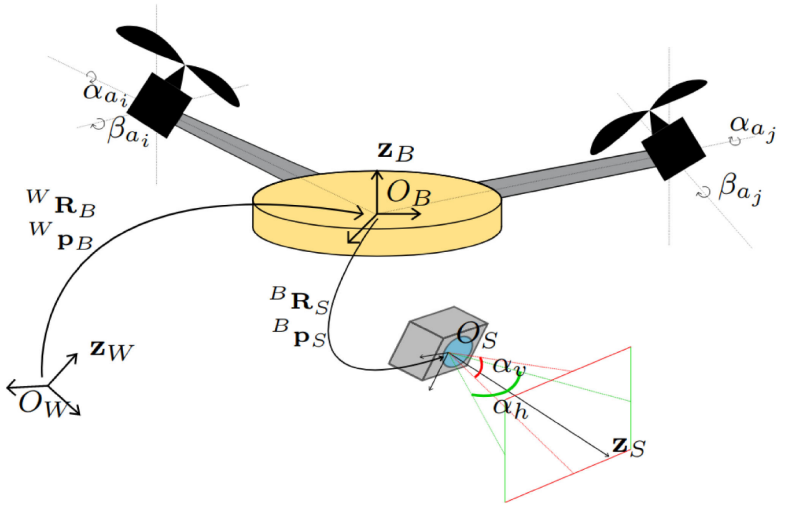


Figure 4.1: GTM schematic showing a single agents' body and camera frames and their relating quantities. Taken from [33]

As depicted in figure 4.1, the inertial world frame and body frame of the sensing agent are respectively denoted by $\mathcal{F}_W$ and $\mathcal{F}_B$. ${}^W\mathbf{p}_B$ is defined as the position of the body frame origin given in world frame coordinates. ${}^W\mathbf{q}_B$ is a unit quaternion that represents the orientation of the body frame with respect to world frame. By using quaternions to represent GTM orientation the singularities present in other methods such as Euler angles and rotation matrices are avoided. Note the propeller angles α_a and β_a define the configuration of each of the n actuators.

The state $\mathbf{x}$ and system inputs $\mathbf{u}$ of a GTM sensing agent with n propellers is defined as

$$\mathbf{x} = \begin{bmatrix}\mathbf{p}^\top \mathbf{q}^\top \mathbf{v}^\top \boldsymbol{\omega}^\top \boldsymbol{\gamma}\end{bmatrix}^\top \in \mathbb{R}^{13+n} \tag{4.1}$$

$$\dot{\boldsymbol{\gamma}} = \mathbf{u} \tag{4.2}$$

$\mathbf{v} \in \mathbb{R}^3$ and $\boldsymbol{\omega} \in \mathbb{R}^3$ respectively depict the translational and rotational velocities of O_{B_i} in world frame. $\boldsymbol{\gamma}$ is a vector containing the n forces produces by the n propellers.

The state derivative $\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u})$ is defined as

$$\dot{\mathbf{p}} = \mathbf{v}, \tag{4.3}$$

$$\dot{\mathbf{q}} = \frac{1}{2}\begin{bmatrix} 0 \\ \boldsymbol{\omega} \end{bmatrix} \otimes \mathbf{q}, \tag{4.4}$$

$$\begin{bmatrix} \dot{\mathbf{v}} \\ \dot{\boldsymbol{\omega}} \end{bmatrix} = \begin{bmatrix} m\mathbf{I}_3 & \mathbf{O}_3 \\ \mathbf{O}_3 & \mathbf{J} \end{bmatrix}^{-1} \left(\begin{bmatrix} -mg\mathbf{z}_W \\ -\boldsymbol{\omega} \times \mathbf{J}\boldsymbol{\omega} \end{bmatrix} + \begin{bmatrix} \mathbf{q} \otimes \mathbf{G}_f \boldsymbol{\gamma} \otimes \mathbf{q}^* \\ \mathbf{G}_t \boldsymbol{\gamma} \end{bmatrix} \right), \tag{4.5}$$

where m is the agent mass and $\mathbf{J}$ is the agent rotational inertia. $\mathbf{I}_3$ and $\mathbf{O}_3$ respectively denote the identity and zero matrix $\in \mathbb{R}^{3\times3}$. $\dot{\mathbf{v}}$ and $\dot{\boldsymbol{\omega}}$ are the translational and angular accelerations of the GTM frame w.r.t. world frame. g is the gravitational acceleration constant, $\mathbf{z}_W$ is the unit vector aligned with the gravitational force in world frame. $\otimes$ denotes the quaternion Hamilton product. $\mathbf{G}_f$ and $\mathbf{G}_t \in \mathbb{R}^{3\times n}$ are the force and torque *allocation matrices* that map the propeller forces to the their resulting forces and torques in body frame.

This models all relevant dynamics for a single agent. Unless specified it is assumed every sensing agent uses this model. When relevant the i_{th} agent state is identified as ${}^{i}x$. The model assumes aerodynamic forces other than the propeller thrust and drag forces are negligible. This generic formulation for the agent dynamics allow for heterogeneity by varying model parameters between different agents or posing different constraints for each.

4.2 Target State Estimation

Target state estimation is done by means of a Kalman Filter. The MPC requires a target state estimate at the start of every new optimization cycle. Due to FOV restrictions target measurements might not always be available. An Intermittent Kalman Filter (IKF) is required to fuse the intermittent measurement signals from all sensing agents and provide the MPC with a target state estimate. Figure 4.2 details the IKF structure and signals. The next section contains a description of the models used by the IKF. Then the IKF system equations are detailed.

Table 4.1: The three IKF target state transition models

Constant Position	Constant Velocity	Constant Acceleration
$\mathbf{A}\mathbf{x}_k^t := \mathbf{I}_3 \begin{bmatrix} x^t \\ y^t \\ z^t \end{bmatrix}$	$\begin{bmatrix} \mathbf{I}_3 & \text{diag}(dt) \\ \mathbf{O}_3 & \mathbf{I}_3 \end{bmatrix} \begin{bmatrix} x^t \\ y^t \\ z^t \\ \dot{x}^t \\ \dot{y}^t \\ \dot{z}^t \end{bmatrix}$	$\begin{bmatrix} \mathbf{I}_3 & \text{diag}(dt) & \text{diag}(dt^2) \\ \mathbf{O}_3 & \mathbf{I}_3 & \text{diag}(dt) \\ \mathbf{O}_3 & \mathbf{O}_3 & \mathbf{I}_3 \end{bmatrix} \begin{bmatrix} x^t \\ y^t \\ z^t \\ \dot{x}^t \\ \dot{y}^t \\ \dot{z}^t \\ \ddot{x}^t \\ \ddot{y}^t \\ \ddot{z}^t \end{bmatrix}$

4.2.1 Target Model

For the target dynamics only linear models are considered, to keep the computation from becoming intractable. Three well-known models are compared; Constant position, velocity and acceleration, listed in table 4.1. The target is modeled by a discrete linear time state space model with additional white noise, given by

$$\mathbf{x}_k^t = \mathbf{A}\,\mathbf{x}_{k-1}^t + \omega_k \tag{4.6}$$

$$\omega_k \sim \mathcal{N}(0, {}^{i}\mathbf{Q}_k) \tag{4.7}$$

Where $\mathbf{x}_k^t$ is the IKF target state estimation at at time k and $\mathbf{A}$ is the state transition model.

4.2.2 Process Noise

The process noise is a measure of uncertainty present in the system. Table 4.2 shows the process noise Q for the 3 different state transition matrices in table 4.1. dt is the IKF filter time period in seconds. Note this assumes velocity and acceleration measurements will not be available and instead have to be inferred from

position derivatives. To approximate the deterioration of uncertainty during this process the process noise is increased by an order of magnitude for each subsequent derivative.

Table 4.2: Process noise used for each of the three state transition models

Constant Position	Constant Velocity	Constant Acceleration
$\text{diag}(dt)$	$\begin{bmatrix}\text{diag}(dt) & 0\\ 0 & \text{diag}(10dt)\end{bmatrix}$	$\begin{bmatrix}\text{diag}(dt) & 0 & 0\\ 0 & \text{diag}(10dt) & 0\\ 0 & 0 & \text{diag}(100dt)\end{bmatrix}$

4.2.3 Measurement model

It is assumed measurements are taken according to a linear measurement model, perturbed by a multivariate zero-mean Gaussian Noise v. Note it is assumed only the 3 positional states are measured. If required by the target model the target velocities and accelerations are inferred by the IKF. The measurement model is given by

$$ {}^{i}\mathbf{z}_k = \mathbf{H}_k\, {}^{t}\mathbf{x}_k^{\text{pos}} + v_k \tag{4.8} $$

$$ v_k \sim \mathcal{N}(0, {}^{i}\mathbf{R}_k) \tag{4.9} $$

Where ${}^{i}\mathbf{z}_k \in \mathbb{R}^3$ depicts the target state as measured by sensor i at time k, $\mathbf{H}_k \in \mathbb{R}^{3\times[3,6,9]}$ is the observation model and ${}^{i}\mathbf{R}_k \in \mathbb{R}^{3\times3}$ is the sensor measurement covariance. For ease of notation the superscript i is dropped for the remainder of this work, unless specifically mentioned z_k denotes a single target measurement at time k.

Observation Model

The observation model for each of the target models is given in table 4.3. Note that this is analogous to assuming sensors can only measure target position and have no constant bias.

Table 4.3: Observation model used for each of the three target models

Constant Position	Constant Velocity	Constant Acceleration
$\mathbf{H} := \mathbf{I}_3$	$\begin{bmatrix}\mathbf{I}_3 & \mathbf{0}_3\end{bmatrix}$	$\begin{bmatrix}\mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3\end{bmatrix}$

Note $\mathbf{I}_3$ is the 3x3 identity matrix.

4.2.4 The Intermittent Kalman Filter

The Intermittent Kalman Filter equations are inspired by the IKF presented in [27] and [38]. Similarly the IKF state at time k is defined as

$$ \hat{\mathbf{x}}_{k|k} := \mathbb{E}\left[\mathbf{x}_k \mid \mathbf{z}_k, \gamma_k\right] \tag{4.10} $$

$$ \mathbf{P}_{k|k} := \mathbb{E}\left[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1})(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1})^T \mid \mathbf{z}_k, \gamma_k\right] \tag{4.11} $$

Where $\hat{\mathbf{x}}_{k|k}$ denotes the target state estimate with covariance $\mathbf{P}_{k|k}$. The availability of a target measurement ${}^{i}\mathbf{z}_k$ at time k and sensor i is modeled by the binary random variable ${}^{i}\gamma_k$. This *Communication variable* γ_k denotes the availability of a target measurement z_k at time k as

$$ \gamma_k = \begin{cases} 1, & z_k \text{ received} \\ 0, & z_k \text{ not received} \end{cases} \tag{4.12} $$

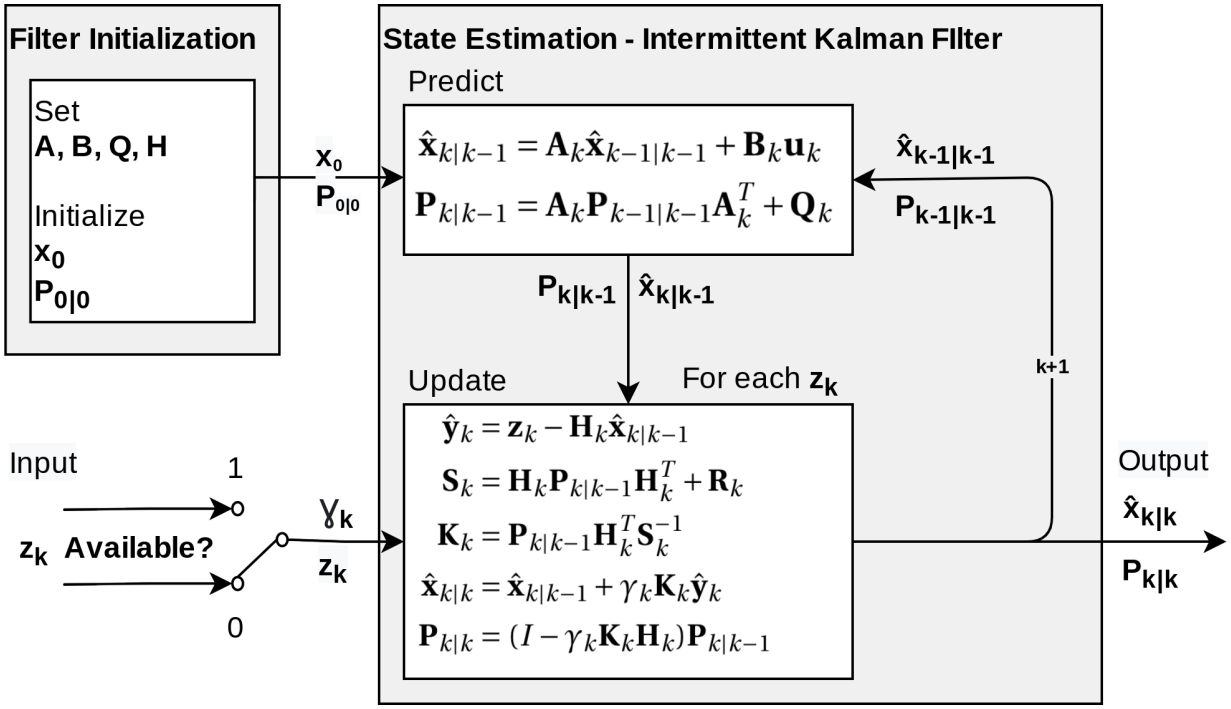


Figure 4.2: Block diagram description of the Intermittent Kalman Filter used for target state estimation

The following completely describes the IKF system:

Predict

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{A}_k\hat{\mathbf{x}}_{k-1|k-1} + \mathbf{B}_k\mathbf{u}_k \tag{4.13}$$

$$\mathbf{P}_{k|k-1} = \mathbf{A}_k\mathbf{P}_{k-1|k-1}\mathbf{A}_k^T + \mathbf{Q}_k \tag{4.14}$$

Update

$$\hat{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1} \tag{4.15}$$

$$\mathbf{S}_k = \mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^T + \mathbf{R}_k \tag{4.16}$$

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^T\mathbf{S}_k^{-1} \tag{4.17}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \gamma_k\mathbf{K}_k\hat{\mathbf{y}}_k \tag{4.18}$$

$$\mathbf{P}_{k|k} = (I - \gamma_k\mathbf{K}_k\mathbf{H}_k)\mathbf{P}_{k|k-1} \tag{4.19}$$

Note that for $\gamma_k = 0$ these equations exactly correspond to propagating the previous state when no measurement is available. At every discrete time instance k the IKF propagates the state estimate using the predict equations. Since the IKF is a linear filter the superposition property allows a separate update step to be called for each separate target measurement z_k. This fuses all available target measurements with the previous state estimate to yield the optimal target state estimate at time k.

4.3 Optimization Objectives and Constraints

The objective is to find the optimal actuator inputs $\mathbf{u}$ for each member a of a team of sensing agents in order to maximize the quality of their combined target state estimate $\hat{\mathbf{x}}^t$ in the near future. This type of constrained optimization problem where an objective function is minimized over a receding horizon using the plant model to predict future system states is a typical Model Predictive Control problem. As one of the main advantages of MPC is the ability to honor constraints during optimization the MPC will directly compute the low-level inputs for the rotor speed controllers instead of computing sensing trajectories and relying on a secondary low level controller to achieve them. Avoiding this cascaded scheme allows to directly

enforce limitations on the motor thrust forces and their rates and ensures constraints on the platform are not violated. The reader is referred to [32] for a more in-depth explanation.

In this section the different objective function terms and constraints are detailed. How the optimization is solved at each iteration is detailed in section 5.2.

The discrete-time optimization problem to be solved is given below, note all constraints are discretized using a 4th-order Runge-Kutta integrator detailed in section 5.2. For each agent a for a target t over the receding horizon T, sampled in N shooting points, at a given discrete instant h, is given by

$$\min_{\substack{{}^{a}\mathbf{x}_0 \dots \mathbf{x}_N \\ \mathbf{u}_0 \dots \mathbf{u}_{N-1}}} J_s := J_{\text{ref}} + J_{\text{perception}} + J_{\text{avoidance}} \tag{4.20}$$

$$\text{s.t.} \quad {}^{a}\mathbf{x}_0 = {}^{a}\mathbf{x}_h \tag{4.21}$$

$${}^{a}\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k) \tag{4.22}$$

$$\mathbf{y}_k = h(\mathbf{x}_k, \mathbf{u}_k) \tag{4.23}$$

$$\underline{\gamma} \leq \gamma_k \leq \overline{\gamma} \tag{4.24}$$

$$\underline{\dot{\gamma}}_k \leq u_k \leq \overline{\dot{\gamma}}_k \tag{4.25}$$

$${}^{\text{fus}}\mathbf{R}_k = g\left({}^{a}\mathbf{x}_k, {}^{t}\mathbf{x}_k\right) \tag{4.26}$$

$$\mathbf{K}_k = \kappa\left(\mathbf{P}_{k|k-1}, {}^{a}\mathbf{x}_k, {}^{t}\mathbf{x}_k\right) \tag{4.27}$$

$$\mathbf{P}_{k|k} = \rho\left(\mathbf{P}_{k-1|k-1}, {}^{a}\mathbf{x}_k, {}^{t}\mathbf{x}_k\right) \tag{4.28}$$

$${}^{t}\mathbf{x}_{k|k} = \chi\left(\mathbf{P}_{k-1|k-1}, {}^{a}\mathbf{x}_k, {}^{t}\mathbf{x}_k\right) \tag{4.29}$$

Where J_s is the cost function detailed in the sections below. ${}^{a}\mathbf{x}_k$ and ${}^{t}\mathbf{x}_k$ are the agent and target states. $\mathbf{y}_k$ are the agent output states. ${}^{\text{fus}}\mathbf{R}_k$ is the fused target measurement covariance as discussed in section 4.3.3. Depending on the exact perception objective term $J_{\text{perception}}$, the Kalman filter constraints 4.27 to 4.29 can be omitted. Details on this and the different perception objectives and cost function terms can be found in the next sections.

4.3.1 Reference Tracking

Reference tracking is achieved by the addition of a reference objective j_{ref} to the optimization cost function. This term should minimize the distance of agent system states to a reference. This Distance is calculated using the Euclidean distance for all except the quaternion attitude states. Because of the unit quaternion equality $q = -q$, geodesic distance in the unit quaternion manifold $d(\mathbf{q}_1, \mathbf{q}_2) = ||log(\mathbf{q}_1 \otimes \mathbf{q}_2^*)||$ is a more appropriate distance measure. For ease of notation only the Euclidean distance is shown. The system output y is mapped from the system states by h as

$$J_{\text{ref}} := \sum_{k=0}^{N} ||{}^{a}\mathbf{y}_k - {}^{a}\mathbf{y}_{r\,k}||_Q^2 \tag{4.30}$$

With

$$\mathbf{y} = \left[\mathbf{p}^T \mathbf{q}^T \dot{\mathbf{p}}^T \boldsymbol{\omega}^T \ddot{\mathbf{p}}^T \dot{\boldsymbol{\omega}}^T\right] \tag{4.31}$$

$$\mathbf{y}_r = \left[\mathbf{p}_r^T \mathbf{q}_r^T \dot{\mathbf{p}}_r^T \boldsymbol{\omega}_r^T \ddot{\mathbf{p}}_r^T \dot{\boldsymbol{\omega}}_r{}^T\right] \tag{4.32}$$

Where $\mathbf{y}_r$ is the reference trajectory to track set by an external component. Q is a weighting matrix to adjust the relative importance of closely tracking each agent state. Higher weight is commonly given to attitude states to prevent the UAV from flipping over during maneuvers and thereby increasing stability.

4.3.2 Collision Avoidance

Collision of the sensing agents with terrain or each other is likely to be catastrophic. To prevent this collision a repulsion field term can be added to the cost function, i.e. similar to [39] as

$$J_{\text{avoidance}} := \sum_{k=0}^{N} ||{}^{a}\mathbf{x}_k^{\text{pos}} - {}^{o}\mathbf{x}_k^{\text{pos}}||^2 \tag{4.33}$$

Where ${}^{a}\mathbf{x}_k^{\text{pos}}$ are the sensing agent positional states at time k and ${}^{o}\mathbf{x}_k^{\text{pos}}$ are positional states where obstacles are present, all in expressed in world frame. This is a *soft constraint* on collision avoidance. The controller is still allowed to send the agents into obstacles but will incur a hefty cost for doing so.

A safer method is to restrict the sensing agents states to a obstacle free subset. This is a *hard constraint* as the controller is guaranteed to not produce any trajectories that lead to collision, provided the position of all obstacles are correctly identified. In practice the position of obstacles is not perfectly known and a state estimator is required to track detected obstacles. The covariance of this estimator can then be used to ensure agent collision avoidance to an arbitrary certainty. This type of constraint can be rewritten as an inequality constraint by considering a sphere of radius r_a centered on the agent at ${}^{a}\mathbf{x}_k^{\text{pos}}$ and a sphere centered on each object at ${}^{o}\mathbf{x}_k^{\text{pos}}$ with radius r_o relative to the estimation uncertainty [40]. The radius of the agent sphere can then be checked for collision against nearby obstacle spheres. This yields the following inequality constraint to be checked for collision avoidance:

$$C({}^{a}\mathbf{x}_k^{\text{pos}}, {}^{0}\mathbf{x}_k^{\text{pos}}, r_a, r_o) := \text{dot}({}^{o}\mathbf{x}_k^{\text{pos}} - {}^{a}\mathbf{x}_k^{\text{pos}}\ ,\ {}^{o}\mathbf{x}_k^{\text{pos}} - {}^{a}\mathbf{x}_k^{\text{pos}}) < (r_a + r_o)^2 \tag{4.34}$$

Often a combination of the two previous methods is used to guarantee collision avoidance and nudge the sensing agents away from nearby obstacles and each other.

Note that for the remainder of this work these techniques have not been used, no obstacle avoidance is implemented thus far. It is mentioned here for completeness and to indicate how it could be achieved in a later stage of development.

4.3.3 Perception Objective

This section discusses the perception objective term $J_{\text{perception}}$. This objective should have a strong positive correlation with the quality of the target state estimate. Two candidates are identified and discussed below. Both require the MPC to be able to predict the measurement covariance of future target measurements, this is discussed first.

Prediction of Measurement Covariance

The measurement covariance is a non-negative quantity associated with a specific target measurement. It denotes the uncertainty present in the target measurement and relates to the measurement quality. It is usually published by sensors alongside their measurement data, however during optimization both these quantities need to be predicted. As discussed in section 5.3 the target measurement prediction is handled by the external IKF. To predict the corresponding measurement covariance it is first assumed the intrinsic camera measurement uncertainty $\mathbf{D}_k$ is shaped as a 3D prolate ellipsoid at the target location with largest semi-axis z pointing along the world z unit vector.

$$\mathbf{D}_k := \begin{bmatrix} f_{xy} & 0 & 0 \\ 0 & f_{xy} & 0 \\ 0 & 0 & f_z \end{bmatrix} \tag{4.35}$$

Where f_z and f_{xy} respectively denote the measurement uncertainty along the principal and remaining axis. At every time instant k the measurement covariance ${}^{a}\mathbf{R}_k$ of agent a is obtained by rotating $\mathbf{D}_k$ using ${}^{W}R_B^T$,

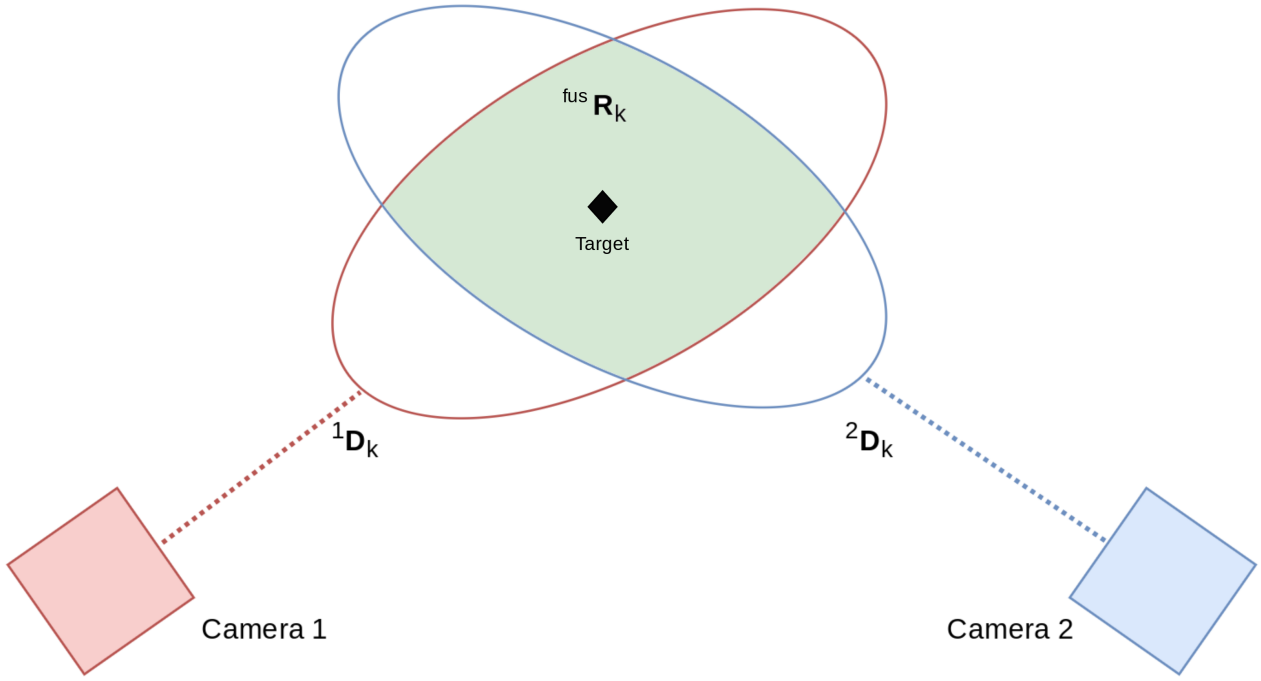


Figure 4.3: 2D illustration of the uncertainty ellipsoids of two cameras and their fused measurement covariance.

aligning the largest principal axis with the vector t_c^v pointing from the center of the agent camera frame to the target using

$$^a\mathbf{R}_k = {}^W R_B \mathbf{D}_k {}^W R_B^T \tag{4.36}$$

This requires an estimate of the target and agent positions at all N shooting points in the finite horizon. The target position estimate is provided by the IKF component. It is assumed agents have perfect information on the location of the whole sensing team and the camera is fixed to the agent body. At each time instant and for each agent ${}^W R_B$ is recalculated using Rodrigues rotation formula

$${}^W R_B = I + \tilde{v} + \tilde{v}^2 \frac{1-\cos\theta}{\sin\theta^2} \tag{4.37}$$

Where $\tilde{[.]}$ is the skew operator. v is the normal of the plane through t_c^v and the ${}^t e_z$, the tag unit z vector. Note this does not holds when $\cos\theta = -1$. This situation is never reached when using a downward facing camera but when this assumption no longer holds this situation should be checked for.

Fused Measurement Covariance

The first perception objective candidate is the fused measurement covariance of the current sensor measurements. It will for the remainder of this work be referred to as the R objective. Using the fused measurement covariance as optimization objective forces the agents to consider the current and future measurements obtained by the rest of the sensing team.

The fused measurement covariance, as illustrated in 2D by figure 4.3, is the shared volume of the measurement covariance of all agents. Following [15] this is computed using

$$^{\text{fus}}\mathbf{R}_k^{-1} = g\left({}^a\mathbf{x}_k, {}^t\mathbf{x}_k\right) = \sum_{i=1}^{A} {}^i\mathbf{R}_k^{-1} \tag{4.38}$$

For all agents i in the sensing team A. The corresponding perception objective term is given by:

$$J_{\text{perception}} := \sum_{k=0}^{N} tr(\mathbf{R}_k) \tag{4.39}$$

A scalar function is required to map the estimate covariance to a scalar cost. The trace is used as a quick measure of the upper bound on the total measurement variance. Note that for the single agent case no fusion is required and the measurement covariance can be directly used.

Kalman Filter Covariance

Analogous to previous work done by Baljinder Signh Bal [26], the second perception objective uses a Kalman Filter covariance and is given by:

$$J_{\text{perception}} := \sum_{k=0}^{N} \text{tr}(\mathbf{P}_{k|k}) \tag{4.40}$$

Where $\mathbf{P}_{k|k}$ is a Kalman Filter covariance matrix of the target estimate at time k, conditioned on all measurements up to and including time k. For the remainder of this work this will referred to as the P objective. Similarly to the R objective the trace is used to map $\mathbf{P}_{k|k}$ to a scalar upper bound of its variance.

This use of a KF covariance matrix as perception objective requires the NMPC to be able to predict future target measurements and a method to predict the KF future target state estimate and its covariance. This is discussed in section 4.4.

Differences between R and P Objectives

This section details my hypotheses on the difference in performance of the R and P perception objectives. Note this is hypothetical but in agreement with findings on a similar topic in [14] and [15]. The first hypothesis is both methods will induce cooperative tracking behavior. The second is that, given an adequate process model and non-infinite process noise, the KF covariance method has a smaller lower bound compared to the fused measurement covariance. Finally I conjecture the KF covariance method will induce trajectories for the sensing agent team. These hypotheses are tested in sections **??** and **??**.

Tracking a KF state in the MPC component requires the MPC state to be extended by 9 elements when using a constant position state transition model, 3 for the target state estimate and 6 for its covariance. When using the more realistic constant acceleration model an extension of 29 elements is required. As both methods require the fused measurement covariance the KF covariance method is strictly more computationally expensive. Since the NMPC directly computes low level sensing agent inputs speed is an important parameter. This is an important disadvantage of the KF covariance method.

For insight into the difference between the two approaches and the basis for the hypotheses consider a comparison of the methods in a two hypothetical scenarios. When the KF process noise approaches infinity, the two methods converge. In this case the prediction covariance will be extremely large, causing the prediction to be fully discarded in favor of any incoming target measurements. This implies the KF covariance equals the fused measurement covariance at each time step. This can be viewed as the worst case scenario for the KF covariance method as there will be no advantage over simply considering the fused measurement covariance even though extra computation is required.

In a second scenario the KF process noise is zero. Here the KF estimate covariance retains all past information and will, given adequately positioned measurements and enough time, converge to a sphere with radius equal to the smallest diagonal entry in the fused measurement covariance matrix. This requires the sensing team to move around over time and take target measurements from different angles, inducing sensing trajectories.

The process noise and accuracy of the plant model are dominant factors in deciding which method to use. If a plant has low process noise and a good model of it is used the KF covariance perception objective is

expected to outperform the fused measurement covariance method after a small time period. With a less competent model and increased process noise the advantages of a KF cannot be fully utilized and the fused measurement covariance method should be considered.

To fully leverage the predictive capabilities of the MPC framework the KF covariance perception objective is used in the remainder of this work.

4.4 Embedding a Kalman Filter in MPC

In order to be able to use the KF covariance matrix as a perception objective information on this quantity and its evolution in time must be available to the MPC. This is similar to the requirement that agent dynamics are known when tracking the agent states inside the MPC. As discussed in section 5.2, a continuous time version of the KF is required instead of the previously derived discrete version. More formally, the state and covariance propagation maps ρ and χ as listed in constraints 4.27 to 4.28 are required. Since these depend on the Kalman gain information on how this gain is computed must also be present in the MPC.

These mappings can be obtained by borrowing results from theory on the Kalman-Bucy filter, the continuous time Kalman filter. Note the common separation into a prediction and update step does not hold in continuous time. The Kalman-Bucy filter assumes a continuous time state space model:

$$\frac{d}{dt}\mathbf{x}(t) = \mathbf{A}(t)\mathbf{x}(t) + \mathbf{B}(t)\mathbf{u}(t) + \mathbf{w}(t) \tag{4.41}$$

$$\mathbf{z}(t) = \mathbf{H}(t)\mathbf{x}(t) + \mathbf{v}(t) \tag{4.42}$$

With $\mathbf{Q}(t)$ and $\mathbf{R}(t)$ the power spectral densities of the added Gaussian noise terms $\mathbf{w}(t)$ and $\mathbf{v}(t)$. To prevent adding a large number of states to the MPC a constant position model $\mathbf{A} = 0$ is used for the internal KF. This yields the following results for the Kalman-Bucy filter state and the ρ, χ and κ maps:

$$\mathbf{x}(t) = [x(t), y(t), z(t)] \tag{4.43}$$

$$\mathbf{K}_k = \mathbf{P}(t)\mathbf{H}^T(t)\,{}^{\text{fus}}\mathbf{R}^{-1}(t) \qquad = \kappa\left(\mathbf{P}_{k|k-1}, {}^{a}\mathbf{x}_k, {}^{t}\mathbf{x}_k\right) \tag{4.44}$$

$$\frac{d}{dt}\hat{\mathbf{x}}(t) = \mathbf{w}(t) \qquad = \chi\left(\mathbf{P}_{k-1|k-1}, {}^{a}\mathbf{x}_k, {}^{t}\mathbf{x}_k\right) \tag{4.45}$$

$$\frac{d}{dt}\mathbf{P}(t) = \mathbf{Q}(t) - \mathbf{K}(t)\mathbf{R}(t)\mathbf{K}^T(t) \qquad = \rho\left(\mathbf{P}_{k-1|k-1}, {}^{a}\mathbf{x}_k, {}^{t}\mathbf{x}_k\right) \tag{4.46}$$

4.5 Full Optimal Control Problem

For convenience, the full optimal control problem statement for the two proposed perception objectives and constraints are repeated below, and the various constraints acting on the system are detailed. Details on how this problem is solved at each time step will be given later, in section 5.2.

The full state vector is obtained by vertically concatenating the N individual agent states given by 4.1 with the states of the KF embedded in the NMPC given by 4.4 as listed in the section above. This results in the full NMPC state vector as

$$\mathbf{x} = [{}^{1}\mathbf{x} \ldots {}^{N}\mathbf{x}, {}^{t}\mathbf{x}] \tag{4.47}$$

Where $\mathbf{x}$ is the full NMPC state vector, ${}^{N}\mathbf{x}$ is the state vector of agent N and ${}^{t}\mathbf{x}$ is the target estimation state vector of the KF embedded in the NMPC.

Fused Measurement Covariance Objective

$$\min_{\substack{{}^{a}\mathbf{x}_0 \dots {}^{a}\mathbf{x}_N \\ \mathbf{u}_0 \dots \mathbf{u}_{N-1}}} J_s := \sum_{k=0}^{N} ||{}^{a}\mathbf{y}_k - {}^{a}\mathbf{y}_{rk}||_Q^2 + \sum_{k=0}^{N} tr(\mathbf{R}_k) + \sum_{k=0}^{N} ||{}^{a}\mathbf{x}_k^{\text{pos}} - {}^{o}\mathbf{x}_k^{\text{pos}}||^2 \tag{4.48}$$

$$\text{s.t.} \quad {}^{a}\mathbf{x}_0 = {}^{a}\mathbf{x}(t) \tag{4.49}$$

$${}^{a}\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k) \tag{4.50}$$

$$\mathbf{y}_k = h(\mathbf{x}_k, \mathbf{u}_k) \tag{4.51}$$

$$\underline{\gamma} \leq \gamma_k \leq \overline{\gamma} \tag{4.52}$$

$$\underline{\dot{\gamma}}_k \leq u_k \leq \overline{\dot{\gamma}}_k \tag{4.53}$$

$$\mathbf{R}_k = g\left({}^{a}\mathbf{x}_k, {}^{t}\mathbf{x}_k\right) \tag{4.54}$$

$$\tag{4.55}$$

KF Covariance Objective

$$\min_{\substack{{}^{a}\mathbf{x}_0 \dots {}^{a}\mathbf{x}_N \\ \mathbf{u}_0 \dots \mathbf{u}_{N-1}}} J_s := \sum_{k=0}^{N} ||{}^{a}\mathbf{y}_k - {}^{a}\mathbf{y}_{rk}||_Q^2 + \sum_{k=0}^{N} tr(\mathbf{P}_{k|k}) + \sum_{k=0}^{N} ||{}^{a}\mathbf{x}_k^{\text{pos}} - {}^{o}\mathbf{x}_k^{\text{pos}}||^2 \tag{4.56}$$

$$\text{s.t.} \quad {}^{a}\mathbf{x}_0 = {}^{a}\mathbf{x}(t) \tag{4.57}$$

$${}^{a}\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k) \tag{4.58}$$

$$\mathbf{y}_k = h(\mathbf{x}_k, \mathbf{u}_k) \tag{4.59}$$

$$\underline{\gamma} \leq \gamma_k \leq \overline{\gamma} \tag{4.60}$$

$$\underline{\dot{\gamma}}_k \leq u_k \leq \overline{\dot{\gamma}}_k \tag{4.61}$$

$$\mathbf{R}_k = g\left({}^{a}\mathbf{x}_k, {}^{t}\mathbf{x}_k\right) \tag{4.62}$$

$$\mathbf{K}_k = \kappa\left(\mathbf{P}_{k|k-1}, {}^{a}\mathbf{x}_k, {}^{t}\mathbf{x}_k\right) \tag{4.63}$$

$$\mathbf{P}_{k|k} = \rho\left(\mathbf{P}_{k-1|k-1}, {}^{a}\mathbf{x}_k, {}^{t}\mathbf{x}_k\right) \tag{4.64}$$

$${}^{t}\mathbf{x}_{k|k} = \chi\left(\mathbf{P}_{k-1|k-1}, {}^{a}\mathbf{x}_k, {}^{t}\mathbf{x}_k\right) \tag{4.65}$$

Constraints

This section details the constraints listed in the MPC optimization problem. Constraint 4.57 forces the system to honor the initial conditions on the agent states. 4.58 constraints the agents states to evolve according to the agent dynamics f detailed in 4.5. Constraint 4.59 forces the system output to evolve according to the system output map h defined by 4.31 in the section below. These constraints together with the initial conditions limit the sensing agent states and outputs to the subset of states possible to reach by the agent.

Constraints 4.60 and 4.61 are band constraints on the actuator forces $\boldsymbol{\gamma}$ and its time derivative. These attempt to capture the physical limitations on the motor velocity and acceleration due to respectively friction on the rotation axis and the inertia of the propeller. Note that together with the system dynamics 4.58 these are the only physical system constraints. $\underline{\gamma}_k$, $\overline{\gamma}_k$ and their derivatives vary for different platform configurations and require an identification campaign as shown in [32].

Finally 4.64 and 4.62 are constraints on the evolution of the perception measures. g denotes the mapping from agent and target states at time k to the measurement covariance $\mathbf{R}_k$ for that agent, discussed in section 4.3.3. ρ denotes the map from agent and target states and the previous KF target state estimate covariance $\mathbf{P}_{k-1|k-1}$ to the new one. This is discussed in section 4.3.3.

5 Architecture

This section lists the different frameworks and methods used to construct the IKF and solve the OC problem given in section 4.5.

5.1 GenoM Framework

All system components have been written in C++ using the GenoM framework. GenoM or *Generator of Modules* is a tool to design real-time software components or modules, developed at LAAS-CNRS [41]. GenoM allows you to capture different operational functions into independent modules and manages their communication and control flow. It is able to operate asynchronous components with different real-time constraints. Another advantage of using GenoM is its middleware independency, GenoM modules can be compiled to work with different middleware like ROS or Pocolibs and for a range of target platforms. Genom generates a description and limited documentation for every module. It is available under a open source BSD license at [1].

5.1.1 Pocolibs

The GenoM modules are compiled to use Pocolibs or *Portability and Communication Libraries* for communication between modules. Pocolibs is a real-time system communication tool comparable to other middleware like ROS. GenoM is able to compile to work with other middleware but Pocolibs is used because it is best supported. Pocolibs is available through Openrobotics[2].

5.2 NMPC

The NMPC module used in the simulation experiments is adapted from a NMPC Genom module previously developed and constructed by Martin Jacquet in [33]. In turn this is based on a C++ implementation of the MATMPC software [42]. To solve the optimal control problem, the dynamic and cost equations and their Jacobian and Hessian are translated to C code using CasADi [43]. These are then discretized using a 4th order Runge Kutta integrator. An explicit solution is found using qpOASES, a Quadratic Programming solver software package especially suited for MPC optimization [7]. A more detailed description of qpOASES is given in appendix **??**. As detailed in section **??** the NMPC directly computes torque level inputs for both quadrotor and hexarotor agents. It allows a UAV agent to track a reference trajectory and modulate it to maintain visibility of a target. It is also able to track moving targets without external input.

5.2.1 Stability

The time it takes qpOASES to find a sufficiently accurate solution to the QP problem varies, so the NMPC control frequency also varies. This raises questions about formal guarantees on the stability of the method, a common critique of MPC. An analysis of control step computation time of this NMPC method during experiments [33] show it is able to compute a control step within 1.7ms 95% of the time, within 4.5ms 99% of the time and with sparse outliers not exceeding 22ms. To investigate stability of the method the control frequency of the NMPC was artificially downgraded to 40ms and it was shown the platform was still able to fly and perform simple maneuvers, showing the capability of the method to compute torque level control inputs of the UAV onboard in real time.

5.3 IKF

An Intermittent Kalman Filter module has been constructed in C++ using the GenoM framework. It is able to provide a target state estimate and covariance at 1kHz, using the fused intermittent target measurements

[1] https://git.openrobots.org/projects/genom3
[2] https://www.openrobots.org/wiki/pocolibs

Table 5.1: Genom modules using in simulation experiments and their functions

Genom Module	Function Description
maneuver	Global reference trajectory planner
mrsim	Multi-rotor simulator of all tk3-mikrokopter features and communication
pom	UKF agent state estimator using IMU and Mocap
optitrack	Interfaces real or simulated Optitrack with Genom
rotorcraft	Low-level hardware interface, interprets rotor velocities
nhfc	Near hovering flight controller, used during takeoff and recovery
camgz	Streams the Gazebo camera images on Genom
arucotag	Detects Arucotags in a camera frame and outputs its 6D pose
NMPC	NMPC controller module as discussed in section 5.2
IKF	Intermittent Kalman Filter as detailed in section 5.3
tagcontrol	Allows control of the target tag from the GenoM layer
joystick	Allows control of the target tag using a joystick

from the ArUco component described in section 5.4. A linear constant acceleration target model is assumed. When requested the IKF publishes an extrapolation of the future target positions. This is used to seed the KF embedded in the NMPC component at the start of every optimization cycle. For more details on the IKF see section 4.2.

For the remainder of this work the IKF or Intermittent Kalman Filter refers to the external IKF component providing target state estimates and its extrapolation to the NMPC. The embedded Kalman Filter refers to the internal KF embedded in the NMPC component, the covariance of which is used as a perception objective.

5.4 Other Components

TeleKyb is a free and open source software framework used for UAV development. Among drivers for devices and robots from various hardware manufactures, TeleKyb provides a high-level closed-loop robotic controller for mobile robots that can be extended dynamically with modules for state estimation, trajectory planning, processing, and tracking. These are all written as GenoM modules, table 5.1 lists the ones used in this project. They are available on the Openrobotics Git[3].

5.5 Physics Simulator

The free open source 3D physics simulator Gazebo [44] is used to simulate the sensing agents and target. Gazebo offers the ability to accurately and efficiently simulate populations of robots in complex indoor and outdoor environments. A simulator eases development and testing, and when required, the TeleKyb interfaces allow for rapid exchange of the simulated physics environment to the real world.

5.5.1 Sensing Agent

Quadrotors are used in the simulated experiments. A list of quadrotor model parameters is given in table 5.2. All sensing agents are equipped with a downward facing camera to provide target measurements. Due to the hardware interface, the quadrotor sensing agent can be quickly swapped for a Tilt-Hex, a fully actuated hexarotor.

[3] https://git.openrobots.org/projects/telekyb3

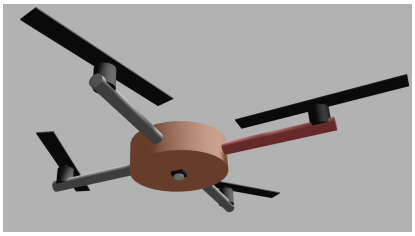


Figure 5.1: Quadrotor sensing agent used in experiments, note the downward facing camera

Parameter	Value
Mass	1.39 [kg]
I_{xy}	0.015 [kg m^2]
I_z	0.007 [kg m^2]
Arm lengths	0.23 [m]
Rotor Mass	7 [g]
Rotor Radius	15 [cm]
Rotor I_{xy}	4e-5 [kg m^2]
Rotor I_z	8e-5 [kg m^2]
Rotor Thrust Coefficient	5.9e-4
Rotor Power Coefficient	1e-5

Table 5.2: Physical Model parameters of the rigid-body quadrotor

5.5.2 Target

A type of fiducial markers, ArUco [45] are used as tracking targets. Fiducial markers commonly serve as reference points in robotics because of the relative ease of detection and the ability of common computer vision algorithms to robustly extract their pose. ArUco fiducial markers are used as target since they are specialized for camera pose estimation and a Genom module that is able to extract the 6D tag pose is already available.

5.5.3 Gazebo Plugins

Two plugins are required to interface Gazebo with the GenoM modules. Optitack-gazebo emulates an OptiTrack motion capture system and streams the agent and target poses, which the optitrack GenoM module retrieves and forwards to pocolibs. The Optitrack-gazebo plugin is available via RobotPKG [4]. The tagcontrol gazebo plugin receives target tag control inputs from the tagcontrol GenoM module and sets the appropriate tag velocity in Gazebo.

5.6 Full System Architecture

The full system architecture including all GenoM modules and the physics simulator interface is illustrated in figure 5.2. At time k the NMPC component receives the current UAV states ${}^a\mathbf{x}_k$ from the pom component and their corresponding references ${}^a\mathbf{x}_{\text{ref}}$ from the maneuver component. It also receives the current target state estimate $\hat{\mathbf{x}}$ and its N extrapolations over the finite horizon of length T. From these the NMPC component computes 4 rotor control inputs $\mathbf{u}_k$ for each of the quadrotor UAV sensing agents. Note in the case of two quadrotors $\mathbf{u}_k$ contains 8 elements, a vertical concatenation of the two row control input vectors. Gazebo simulates both the quadrotor agents and the target tag. The agent state ${}^a\mathbf{x}_k$ is tracked by the optitrack component. This also tracks the target tag to provide the ground truth values. The arucotag component extracts the tag position from the camera data obtained from the camgazebo component. The IKF fuses these two target measurements $\mathbf{z}_k$ into a single estimate of the target state.

[4] http://robotpkg.openrobots.org/robotpkg/simulation/optitrack-gazebo/

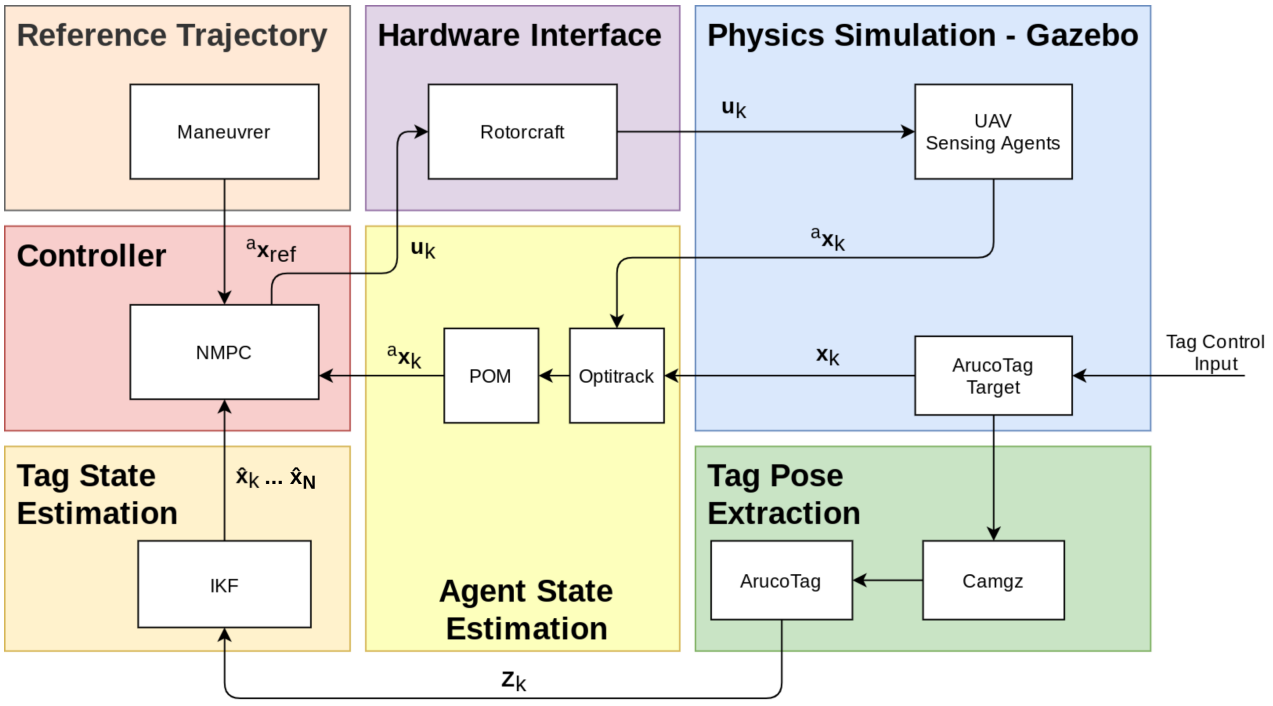


Figure 5.2: Block diagram representing the full system architecture

6 Validation and Discussion

This section details the simulation setup used to validate the proposed method. For clarity results are grouped into sections and subsequently discussed. The objective is to show the proposed method is capable of cooperatively tracking a target using heterogeneous agents. In all experiments the sensing agents are quadrotors, as detailed in section 5.5.1.

To validate the proposed method three sub-objectives are investigated. First is the ability of the proposed method to put both single and dual quadrotors to their optimal sensing configuration. This is discussed in section 6.2. Second is the ability of the method to exploit an additional member of the sensing team to yield a lower target estimate covariance when using two quadrotors instead of one. This is discussed in section 6.3. Finally the ability of the proposed method to track moving targets is validated in section 6.4. To investigate the stability of the proposed method control period statistics are analyzed in section 6.5. First the IKF target state estimation performance is investigated.

6.1 IKF

A target state estimation method is required to fuse the intermittent target measurements at each time instance into a single target state estimate. It also provides the estimate covariance, a quantity by which the overall quality of the estimate can be judged. The quality of the IKF state estimation is assessed by comparing the IKF estimates to ground truth. During these experiments the IKF assumes a constant acceleration model for the target.

As illustrated by figure 6.1, the IKF performance test setup contains two quadrotor sensing agents statically suspended 2m above ground. Each is equipped with a downward facing camera. An ArUco target marker is placed on the ground and is moved with constant velocity. The tag velocity is controlled by a joystick and can be instantaneously changed over the course of the test. Note that by design the tag will move outside of the camera FOV. This is done to gauge the predictive performance of the IKF when no target measurements are present.

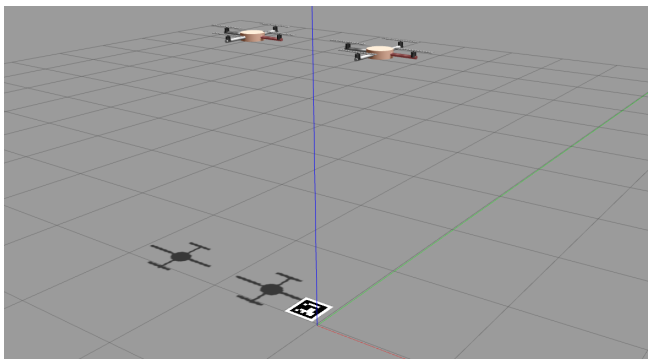


Figure 6.1: Gazebo test setup to benchmark the IKF component, note the UAVs are statically suspended

Figures 6.2 to 6.4 show the IKF estimation of respectively target position, velocity and acceleration. Note the increase in the target estimate covariance encoded by the color shift towards red when the tag leaves camera FOV.

These results show the IKF is able to track the tag and continues to provide estimates of the tag position after the tag has left camera FOV. As expected, the prediction error grows quickly after the change of velocity outside camera FOV.

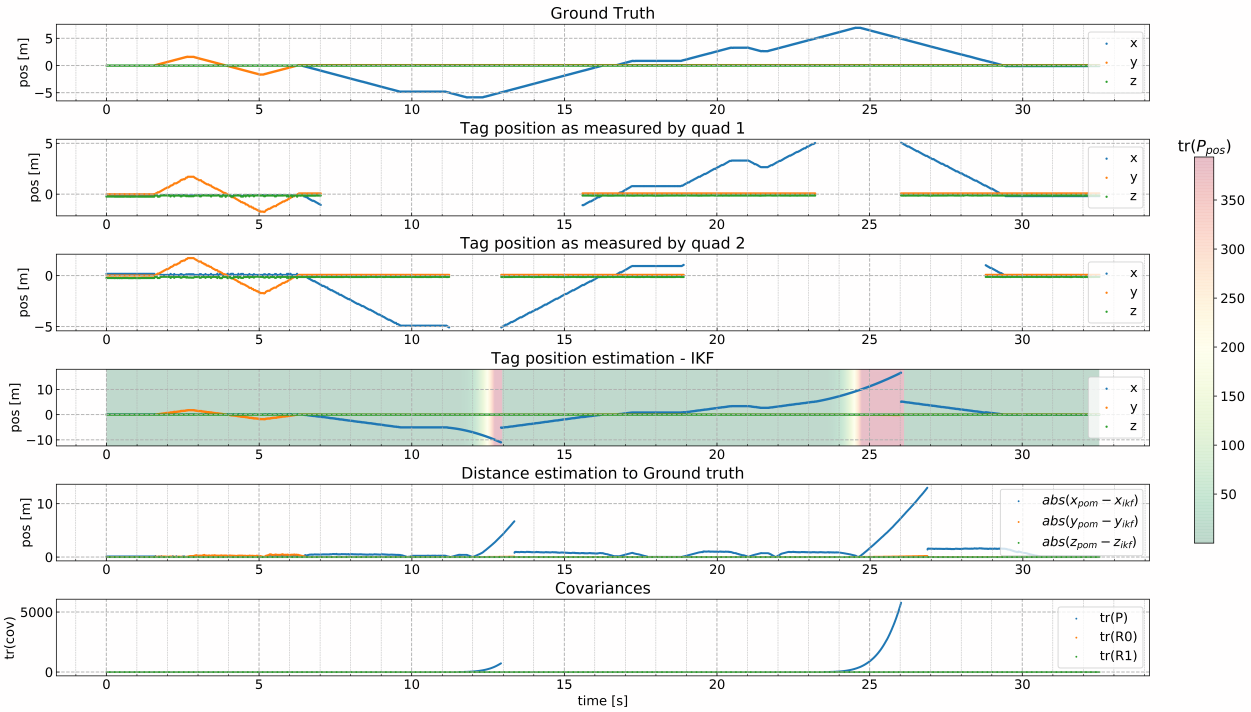


Figure 6.2: Tag position as measured by POM, ARUCO, IKF and IKF distance from ground truth

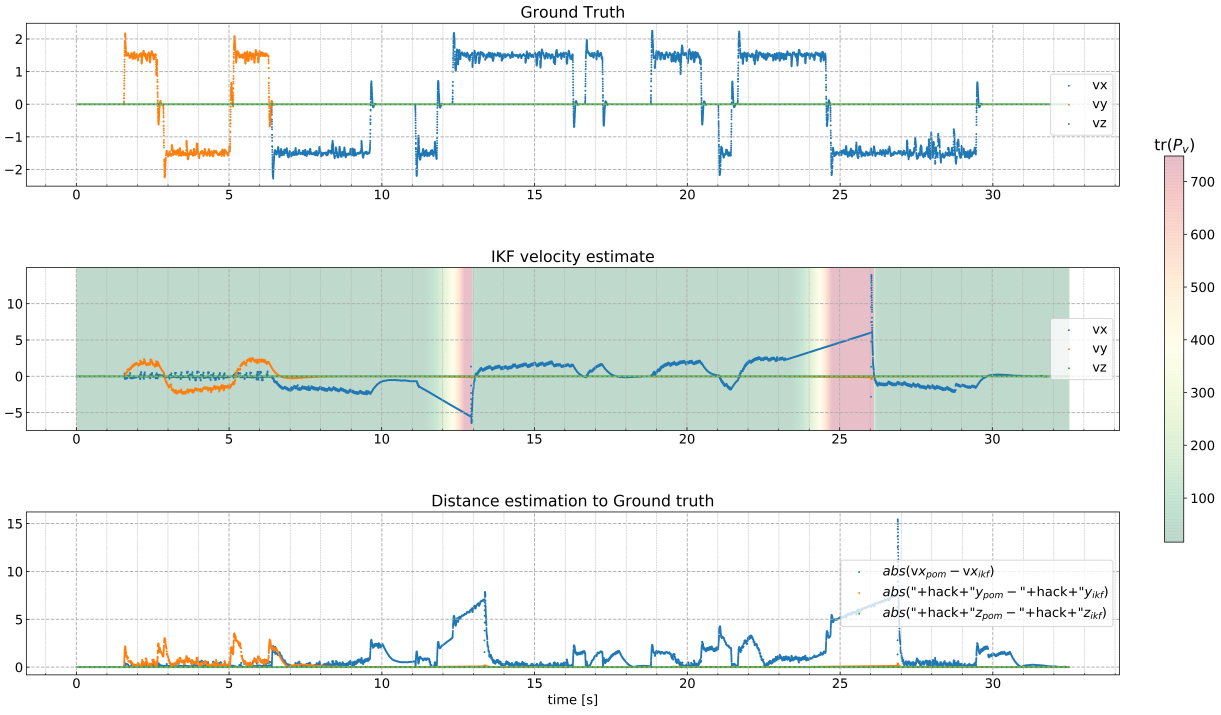


Figure 6.3: IKF tag velocity estimation compared to ground truth

The IKF assumes a constant acceleration model for the target tag. This can be prohibitive assumption, especially so when tracking a highly erratic non linear target. In this case an extended Kalman filter scheme could be implemented. When tracking multiple targets the proposed IKF requires a priori information of the amount of targets and their initial state. It is also unable to handle a time-varying number of targets.

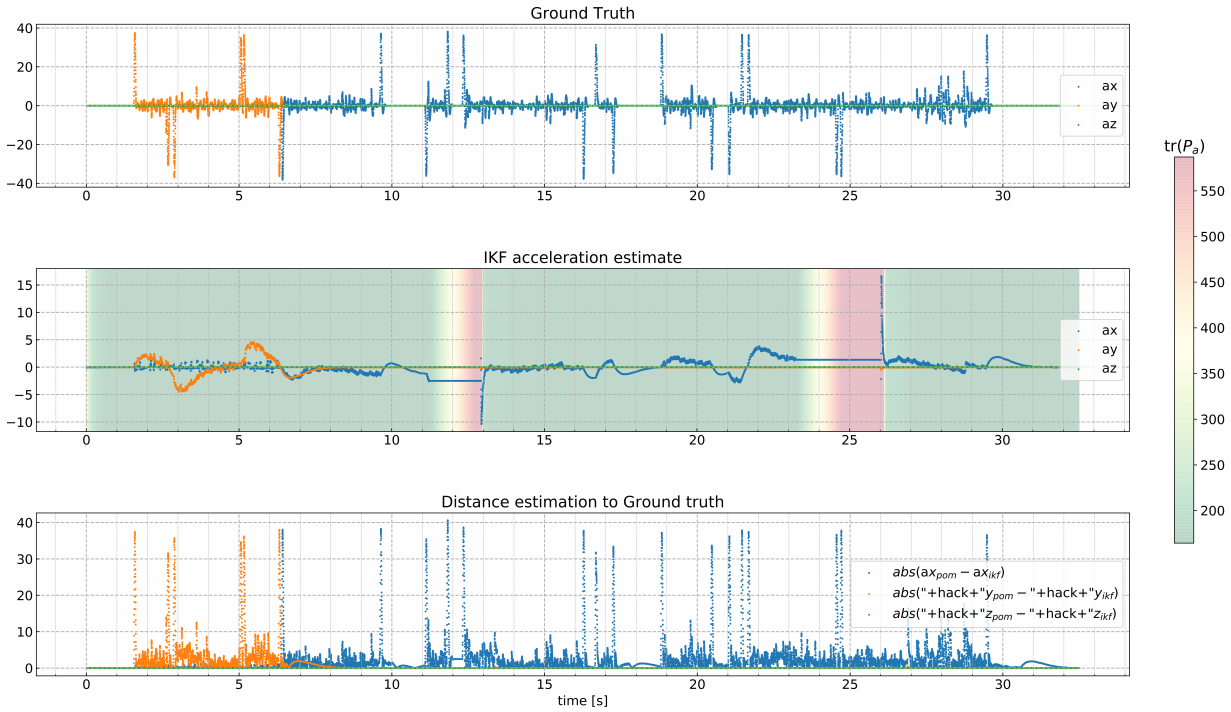


Figure 6.4: IKF tag acceleration estimation compared to ground truth

These limitations can be avoided by switching to another state estimation method such as the Probability Hypotheses Density filter as described in [8].

Next the validation experiment results for the proposed NMPC method are discussed.

6.2 Optimal Sensing Configuration

This section aims to demonstrate the proposed methods capability to drive agents to a configuration that minimizes their collective target state estimate covariance. Unless otherwise specified for the remainder of this chapter the KF covariance perception objective and weights listed in table 6.1 are used. Note that the attitude states have 4 weights associated because it consists of quaternions.

6.2.1 Configuration - Single Quadrotor

In this experiment a single quadrotor agent is brought to a hovering condition near the target tag. The target tag is fixed to ground. The quadrotor position is initialized such that the tag is inside camera FOV when hovering at a height of 2m. After reaching a stable hovering condition the perception objective is

Table 6.1: Perception and Reference objective weights used during experiments

Reference - weights	x	y	z	
position	0.01	0.01	0.1	
attitude	0.75	0.75	0.75	0.75
linear velocity	0.25	0.25	0.5	
angular velocity	0.5	0.5	0.5	
linear acceleration	1e-5	1e-5	1e-5	
angular acceleration	1e-5	1e-5	1e-5	

enabled. By setting the x and y reference positions to the current position they are effectively low passed. This is done as a quick way to better observe control actions computed by the NMPC originating from the perception objective. The height reference is kept at 2m.

Figure 6.5 show the x y and z position of the quadrotor during transition from 4 different initial hovering conditions to the optimal sensing location. The bottom plot shows the NMPC cost evolution of the fourth run. The color denotes the trace of the KF covariance matrix, note this decreases during the experiment.

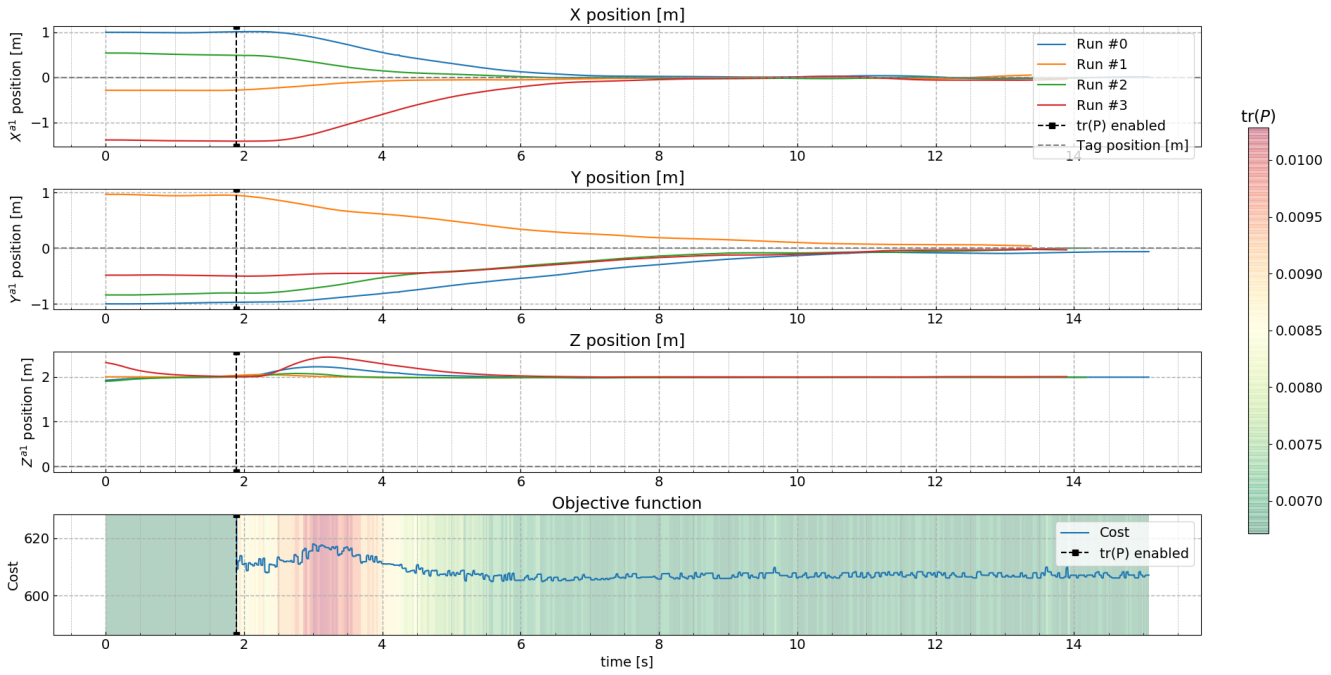


Figure 6.5: Evolution of NMPC cost and quadrotor position during the single quadrotor sensing configuration experiment

As the perception objective is enabled, the agent is driven to a position directly above the tag. This is the optimal sensing position for the single agent, as it is the closest position to the tag whilst still conforming to the height setpoint of 2m. This shows the method is able to control a single quadrotor agent from different initial positions into its optimal sensing location with respect to the target.

6.2.2 Configuration - Two identical Quadrotors

The optimal camera configuration for two cameras with identical covariance matrices requires the rays projected by their image points to be orthogonal [46]. During the experiment the height setpoint of both agents is kept at 2m and deviations from their reference attitudes incurs a heavy cost to promote stability. Combining this information yields the optimal sensing configuration for this experiment puts each agent at the height closest to 2m while still keeping the target inside its FOV, each on the opposite side of a circle centered on the tag with a stable attitude.

The setup for this experiment is similar to the one presented in 6.2.1 except now 2 identical quadrotors are used. The 2 identical quadrotors are initialized at a different starting position. After reaching a hovering position the perception objective is again enabled and the x and y position references set to the quadrotors current position. The height reference is set at 2m. During the experiment the quadrotors are repeatedly dragged away, effectively causing a step disturbance in their x and y position.

Figure 6.6 shows the configuration reached by the two quadrotors in absence of external disturbances.

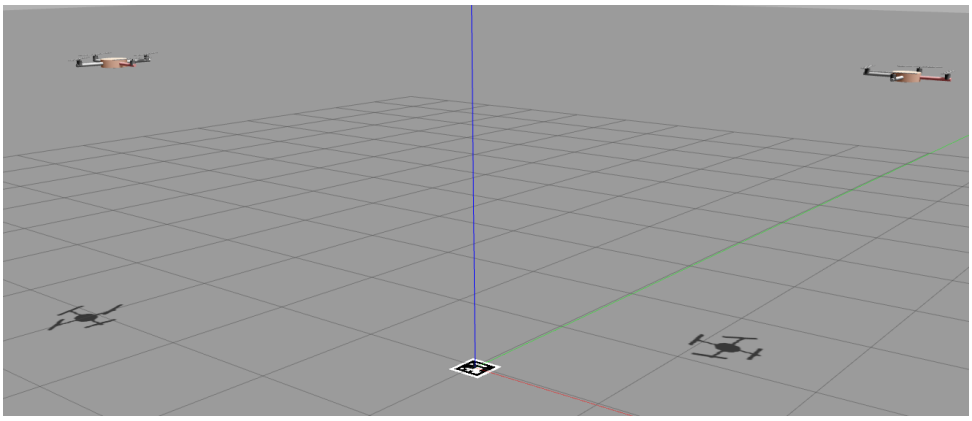


Figure 6.6: The proposed NMPC method controls two quadrotors, driving them to their optimal sensing configuration

Data of this experiment is visualized in figure 6.7, showing the position of the agents, the normalized dot product of the world frame vectors from each quadrotor to the target and the cost evolution over time. The normalized dot product is used as a measure of orthogonality, two vectors are orthogonal if their dot product is zero. After every disturbance the quadrotor x and y positions converge to points on opposite sides of a circle centered on the tag at a height of 2m. This is confirmed by looking at the evolution of the normalized dot product, which tends to zero a few seconds after every disturbance. From the bottom figure can be observed the disturbances cause an increase in the Kalman filter covariance trace and subsequently the NMPC cost. The proposed method drives the quadrotors back into a state where the covariance trace is lower.

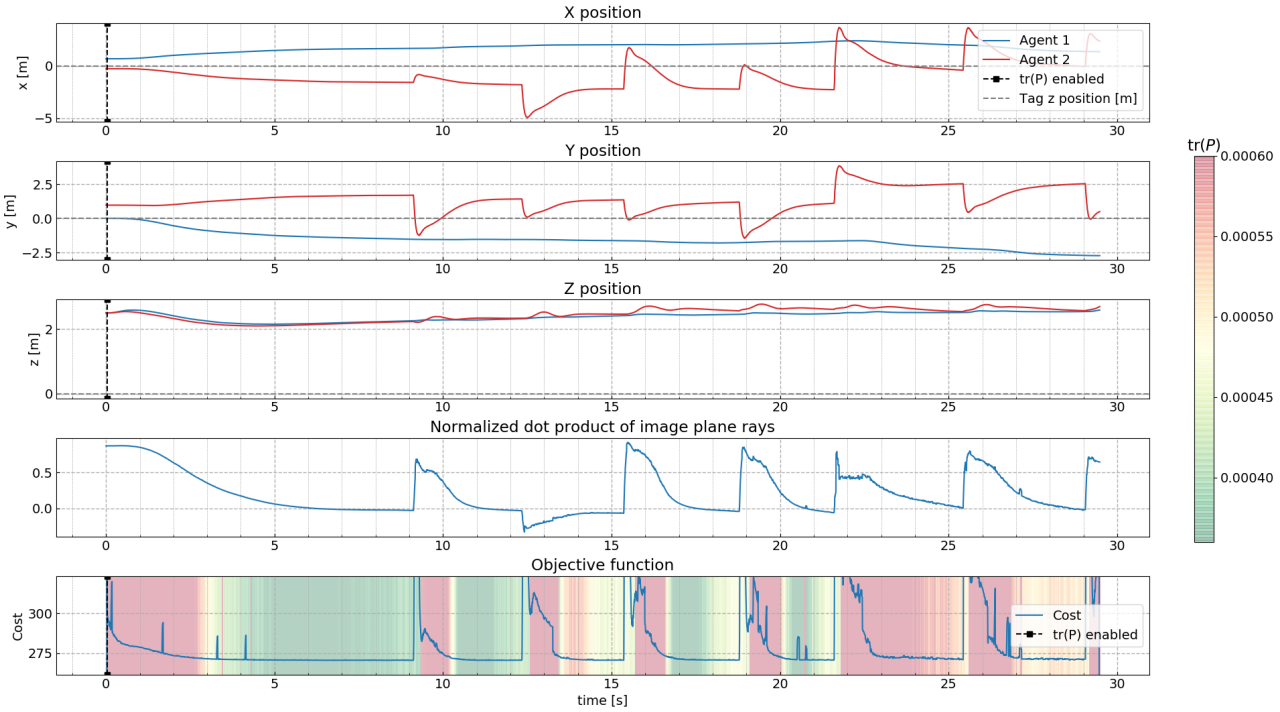


Figure 6.7: The (x,y,z) coordinates of the two quadrotors during the optimal configuration experiment. Fourth plot shows the evolution of the normalized dot product of the vectors from quadrotor to target. Bottom plot shows the evolution of the NMPC cost over time, color represents the trace of the Kalman filter covariance.

Figure 6.8 shows the attitude of both agents remains stable during the experiment, indicating the ability of the method to stabilize the quadrotor agents even in presence of step disturbances of multiple meters.

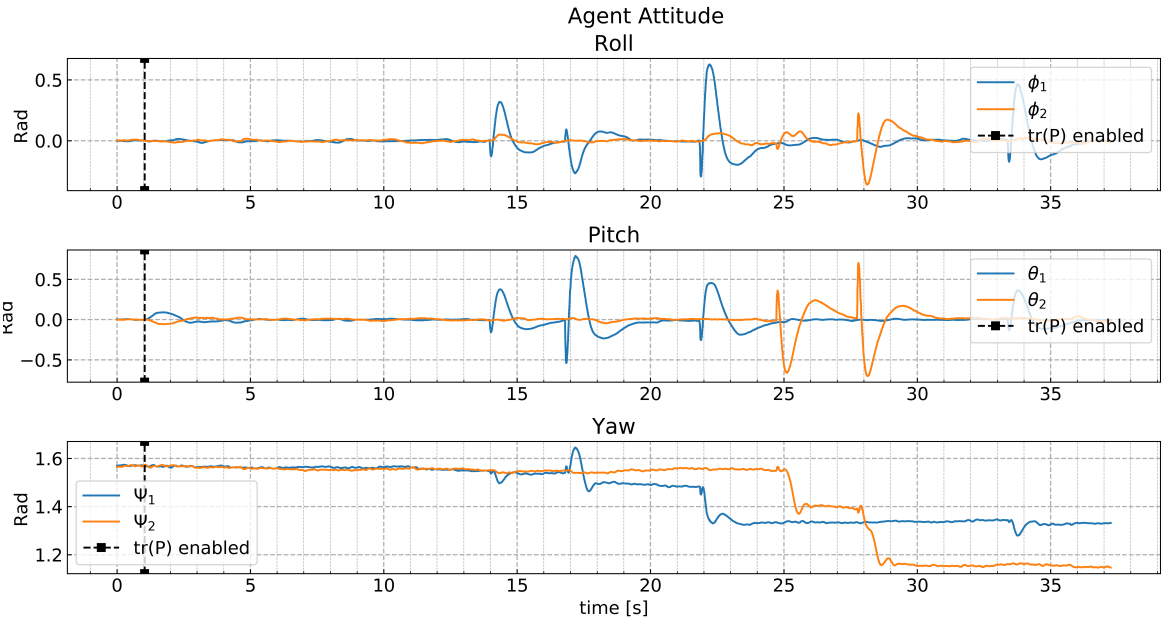


Figure 6.8: Attitude of both quadrotors during the optimal sensing configuration experiment

These observations show the proposed method is able to drive the 2 identical quadrotors to the optimal sensing configuration, optimal in the sense it minimizes their collective target estimation covariance whilst complying at best with their possibly conflicting state setpoints.

6.3 Exploiting extra quadrotor

This section aims to prove the proposed method is able to converge to a lower estimate covariance when using two quadrotors as compared to a single one, to show it is able to constructively utilize the extra resources. Two simulations similar to the previous ones are conducted, one with a single quadrotor sensing agent and one with two identical quads. After the quadrotors reach a stable hovering condition the perception objective is enabled and their position references are again freed. After some time the agents will converge to the optimal sensing position as described in the previous section. The trace of the target estimate covariance for both cases is compared.

Figure 6.9 shows a comparison of the IKF and NMPC embedded KF target estimate covariance for both the single and double quadrotor case. As expected, the IKF and KF embedded in the NMPC have near identical covariances. In both cases, the mean of the target estimate covariance trace improved by an order of magnitude. This indicates an improved estimation quality when using two quads.

Unexpectedly there are significant outliers visible in figure 6.9. For the two quad case these could be explained by the tag moving outside the FOV of one quad. The optimal sensing configuration for two quads puts the target at the edge of their FOV, this could cause one to lose sight for a moment. When the tag is at the FOV edge the measurement covariance degrades quickly. The arucotag component calculates measurement covariance based on distance of the camera to the target, distance of the tag from the optical center and relative orientation of the tag w.r.t the camera. More details on this can be found in section 4B of [33]. These effects are not modeled in the NMPC. In the NMPC is assumed the measurement covariance is an ellipsoid pointing along the camera target vector, unaffected by any of the above distortions. This

To further investigate these effects, the evolution of the KF covariance trace is compared for four different setups where the quadrotors are statically suspended, as illustrated by figure 6.10. Table 6.2 shows the computed mean and standard deviation of the measurement and KF covariances for each of these setups over 60 seconds. Note that during the optimal configuration experiment in section 6.2, the quads converge to static setup a and c when using 1 or 2 quads respectively. Setup b aims to give insight into the increase in measurement covariance when measuring a tag in the corner of the camera FOV. Setup d attempts to compensate for this by rotating the two quads such that the tag is in the center of their FOV.

A few observations are made. As expected, the covariances of the IKF and KF embedded in the NMPC component are almost identical. The measurement covariances in the case of two quadrotors are near identical. This shows the two quads are controlled to symmetrical positions, a desired behavior when two identical quadrotors are used.

Comparing experiment A and B, the measurement covariance traces of experiment A is 51% worse. The mean of the KF covariance trace of the single agent corner case is increased by a factor of 12.8 compared to the centered case. This indicates a significant degradation in measurement covariance and subsequently estimation quality when a target is measured at the edge of camera FOV.

In a static test the estimation covariance trace for two agents is improved by a factor of 17.2 over the single centered agent case. The standard deviation is improved by 3 orders of magnitude. Note that this improvement is obtained in spite of the 51% decreased individual measurement covariance, as shown by the results of experiment A and B. This indicates the method is effectively able to exploit the second quad, controlling both to locations of decreased individual measurement covariance in order to improve the overall target estimation quality.

Tilting the quads such that the target is in their optical centers reduces the individual measurement covariance traces by a factor of 2.75. This yields a 31 fold reduction in mean KF covariance trace compared to two identical quads in upright position. This shows the improvement obtained by measuring the tag in the FOV center is completely negated by the strong increase in measurement covariance when the tag is oblique with respect to the camera.

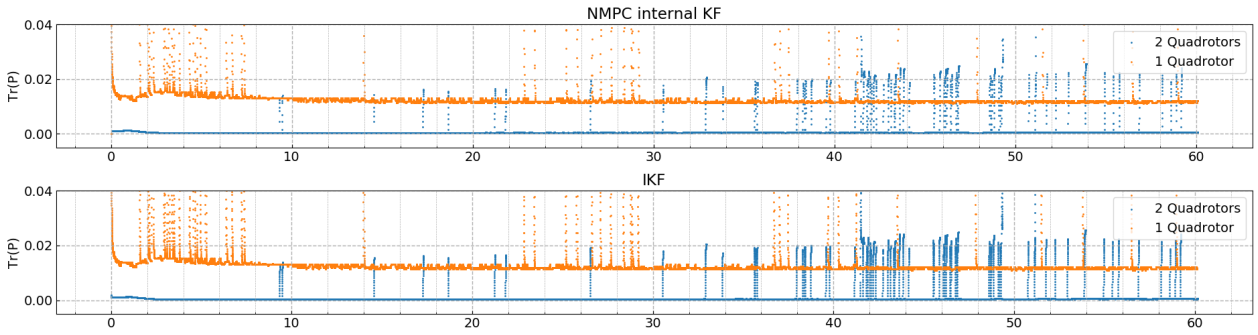


Figure 6.9: The trace of the IKF and embedded KF covariance over time using one and two quadrotors.

For both the single and dual quad case, the proposed method is able to drive the agents into their optimal sensing configuration, also in the presence of external disturbances. These locations are optimal in the sense they minimize the collective target estimation covariance, and agree with theory. These observations prove the proposed method is able to exploit the extra quadrotor to achieve a lower estimation covariance as compared to the single agent case.

Two observations are made so far. Firstly the proposed method can control two agents to put them in their optimal sensing configuration. Second, for two quadrotor agents this optimal sensing position achieves a lower bound on the trace of the target estimation covariance as compared to a single quadrotor. From this can be concluded the proposed NMPC method is able to produce cooperative behavior between two

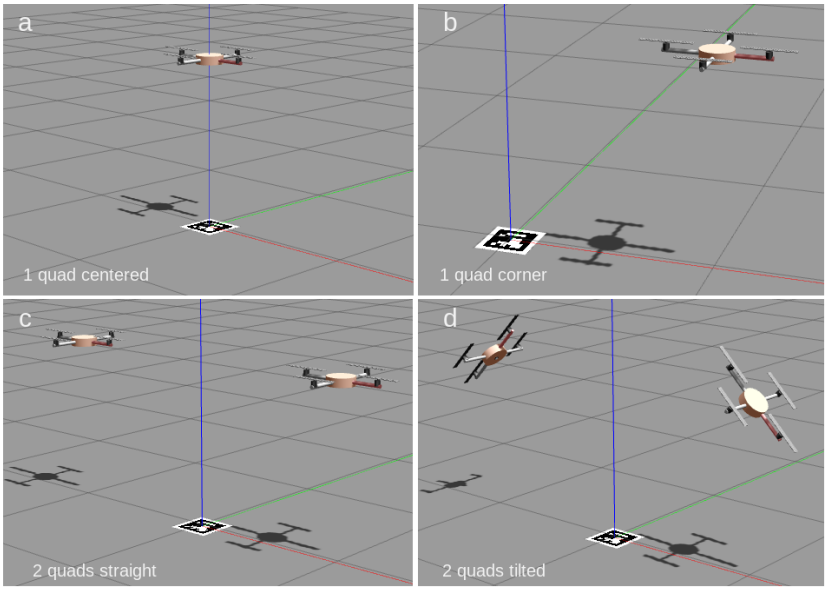


Figure 6.10: Four static test setups to compare KF covariance, quad height is set to 1m. a) contains a single quad centered on the target. b) contains a single quad shifted 1m along the x axis. c) contains two quads in a configuration similar to 6.2.2 d) contains two quads in the same configuration, but now titled such that the tag is in their FOV center

Table 6.2: Measurement and KF covariance Mean and standard deviation of figure 6.9 and four static experiments depicted figure 6.10

	IKF		NMPC		Static test - IKF			
	1 quad	2 quads	1 quad	2 quads	**A** - 1 center	**B** - 1 corner	**C** - 2 straight	**D** - 2 tilted
mean(tr(R0))	1.37e-01	2.73e-01	1.37e-01	2.73e-01	2.80e-03	5.62e-03	5.62e-03	1.55e-02
mean(tr(R1))	×	2.87e-01	×	2.87e-01	×	×	5.59e-03	1.55e-02
mean(tr(P))	**1.59e-02**	**1.24e-03**	**1.58e-02**	**1.26e-03**	**1.32e-03**	**2.00e-03**	**7.67e-05**	**2.43e-03**
std(tr(P))	2.78e-02	4.31e-03	2.76e-02	4.29e-03	1.76e-07	4.52e-07	2.44e-10	6.53e-07

quadrotor agents, and is able to constructively utilize the extra quadrotor. Next the ability of the proposed method to track moving targets is investigated.

6.4 Cooperative Target Tracking

The following experiments aim at demonstrating the capability of the proposed method to cooperatively track a moving target. A setup similar to the one detailed in 6.2 is used, except now the target is moved. After reaching a stable hovering condition, enabling the perception objective and setting the agents (x,y) position references equal to their current positions, the target is moved at constant velocity.

6.4.1 Tracking - Single Quadrotor

First the tracking capabilities of the proposed method using a single quadrotor is assessed. Figure 6.11 shows the evolution of the quad and tag position during the experiment. The NMPC method is able to control the single quad such that it tracks the tag, continuously attempting to converge to the single quad optimal sensing configuration, which is directly above the tag. The bottom plot in figure 6.11 depicts the cost and IKF covariance trace, showing the tag is kept inside the quads FOV during the experiment. Note change in tag movement is associated with an increasing cost as the tag start to move outside the quads FOV and thus degrades the estimation covariance trace.

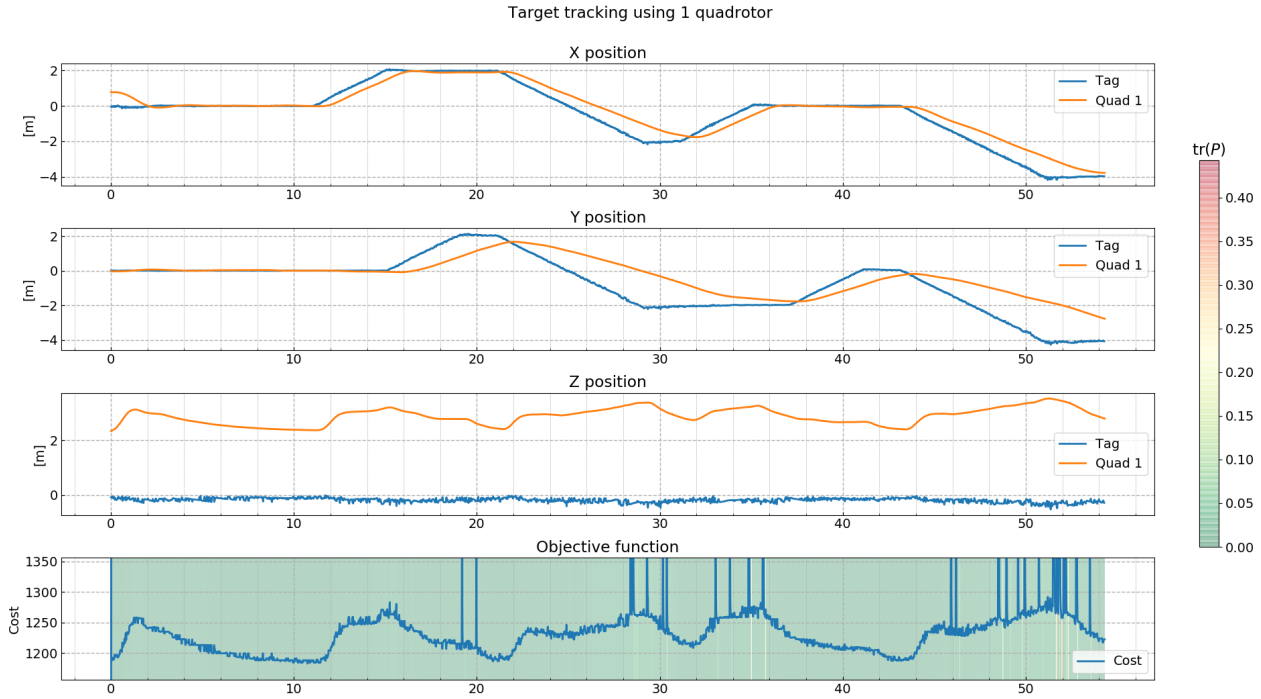


Figure 6.11: (x,y,z) position of the quad and the target during target tracking. Bottom plot shows the evolution of the cost over time, color denotes the covariance trace

As can be observed from figure 6.12 the control input mostly stays within the constraints during tracking. This plot shows the ability of the method to drive the sensing agent close to its actuator limits when the tag rapidly accelerates.

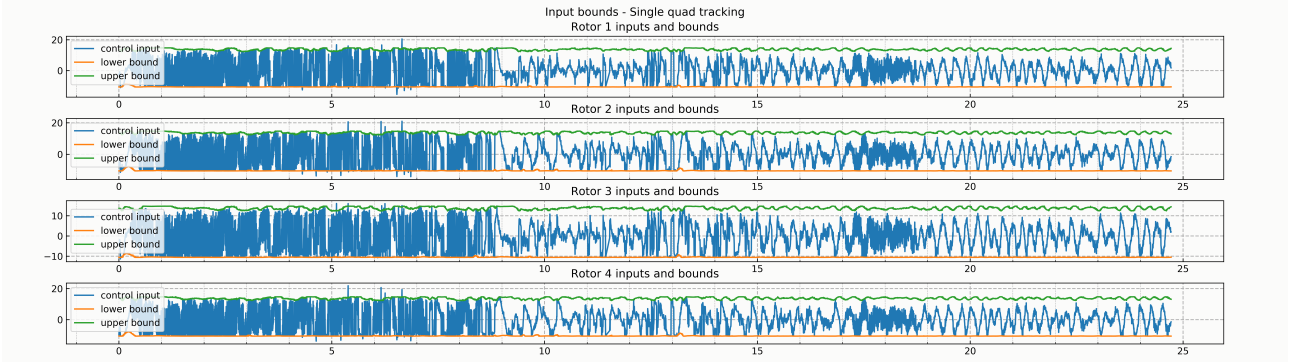


Figure 6.12: Control input and bounds during single agent tracking experiment

These results show the proposed method is able to track a moving target using a single quad. Next two quadrotor UAVs are used to cooperatively track a moving target.

6.4.2 Cooperative Tracking - Two Identical Quadrotors

Figure 6.13 shows the (x,y,z) position of the two quadrotors and the target tag when cooperative tracking is attempted using two identical quadrotor UAVs. As before, the two quads initially move to the optimal sensing configuration. When the tag is moved a response is induced in both quadrotors such that the tag is kept inside both FOVs. Similar to results obtained in section 6.2.2, the quads are driven to opposite equidistant positions of the tag around their setpoint height of 2m. At 55 seconds in the experiment rapid acceleration

in the tag causes it to move outside the FOV of the first quad. As can be observed from the bottom plot of figure 6.13, this causes a large increase in the estimate covariance trace and cost. At 58 seconds the first quad recovers vision of the tag and the covariance lowers again.

The two top plots in figure 6.14 show the measurements from both quadrotors during the experiment. The third plot shows the tag position estimation of the IKF and its covariance. The bottom plot shows the individual measurement covariances. It can be observed the IKF is able to produce a low position estimate covariance even when the individual measurement covariances degrade due to the tag moving to the FOV edges. Near the end of the experiment the first quad loses sight of the tag multiple times but is subsequently controlled back to a position where target sight is restored. There in a steady state error present in the tag z position measurement coming from the aruco component. This is likely due to an initialization error.

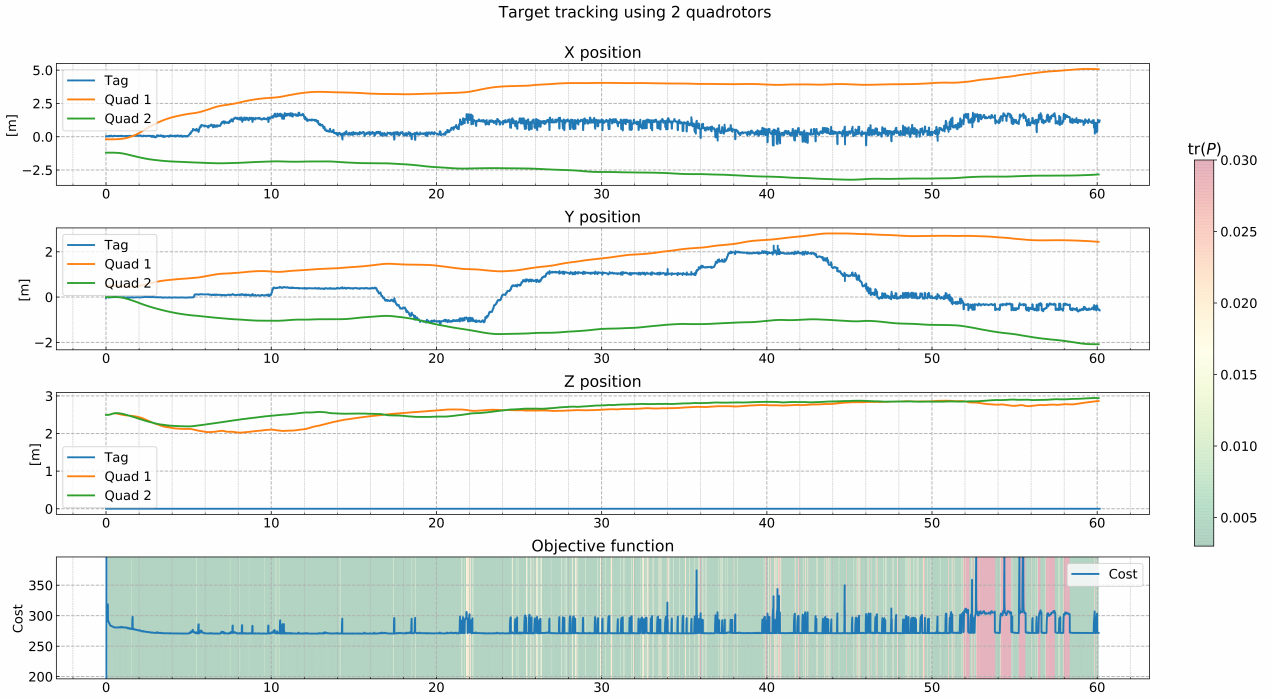


Figure 6.13: (x,y,z) position of the two quads and the target during target tracking. Bottom plot shows the evolution of the cost over time, color denotes the covariance trace

These results show the potential of the proposed method to track a moving target using a single quadrotor, and to cooperatively track a moving target using two quadrotors.

6.5 Computation Time

A crucial parameter of the stability of the proposed NMPC method is its ability to compute control inputs sufficiently fast. Figure 6.15 presents a percentage histogram of the computation time distribution of the single and dual quadrotor experiments in this chapter, binned in 0.5ms intervals. Note this covers a total flight time of over 3 minutes. When using a single quadrotor the mean computation time is 1.63 ms with a standard deviation of 0.98. When two agents are used the mean computation time is 3.36, a 106% increase, with a standard deviation of 1.64, a 66% increase. These values translate to an average control frequency of 613Hz and 298Hz respectively. The addition of the extra quadrotor sensing agent causes both the mean and standard deviation of the control period to increase. For the single agent case 0.038% of control periods exceeded 10ms, 0.412% for the dual quad case. In [33] was found a 40ms control period still allowed for stable flight and simple maneuver. From this can be concluded the proposed method is within the stability margins.

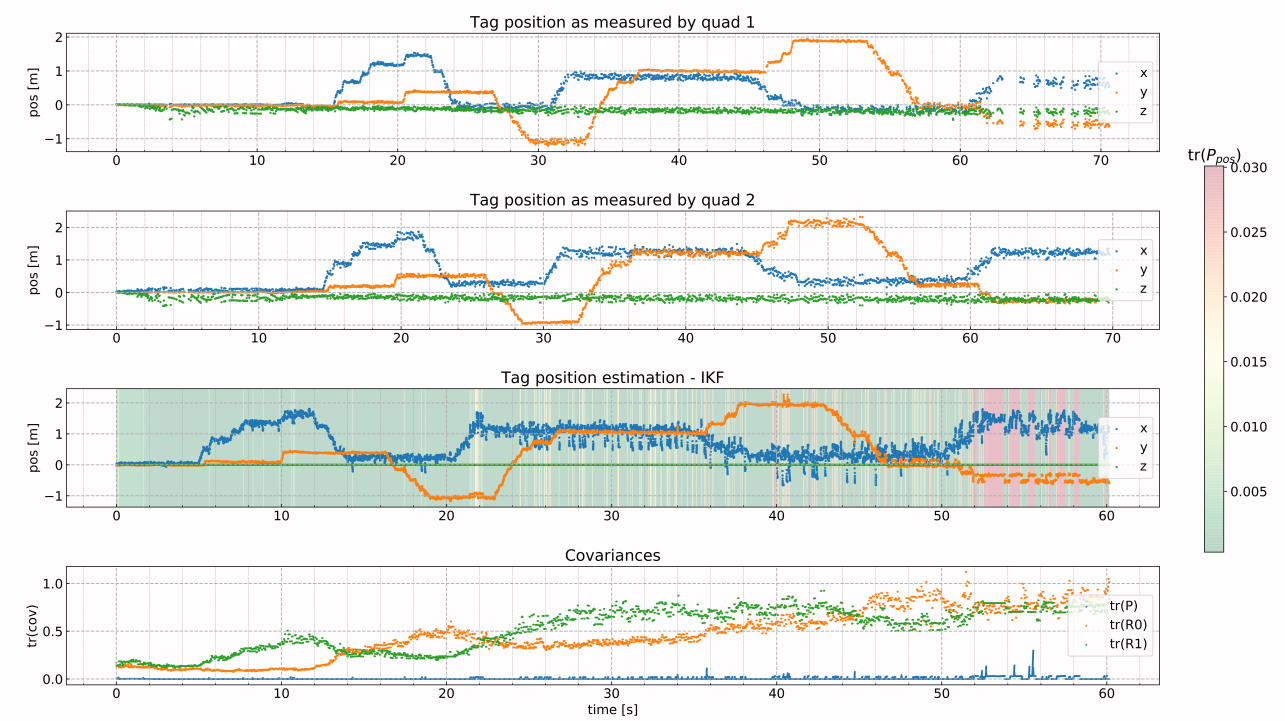


Figure 6.14: (x,y,z) position of the target tag over time. Top plot shows ground truth. Plot two and three show tag position measurement from the first and second quad respectively. Fourth plot shows the IKF tag state estimation. Fifth plot shows its difference to ground truth. Bottom plot shows the measurement covariance trace for both quads and the IKF tag estimation covariance trace.

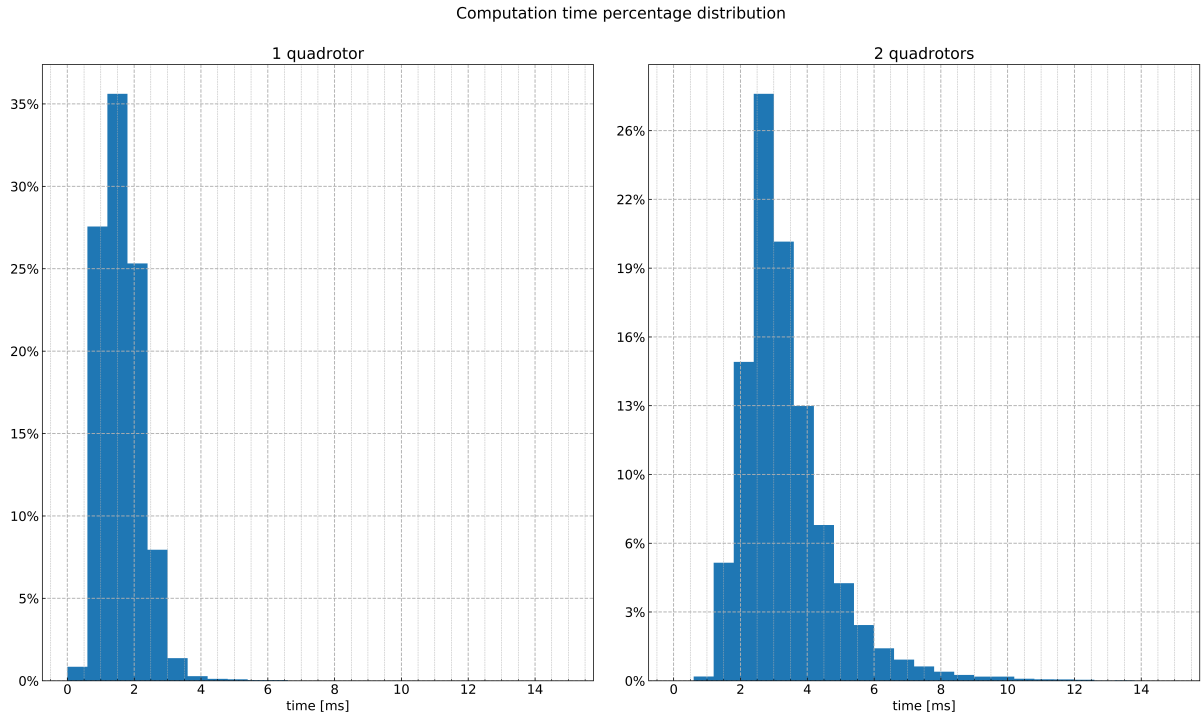


Figure 6.15: Histogram of the NMPC control period percentage distribution during the previous experiments

7 Conclusion

This work aims to answer how we can use Nonlinear Model Predictive Control (NMPC) to cooperatively track a target by minimizing the fused target estimation uncertainty of a heterogeneous sensing team. A literature review of the surrounding State of the Art has been conducted. It has been found MPC is well suited to cooperative tracking problem. It is able to consider the future trajectory of both the target and sensing agents, can compute low-level inputs for non-linear systems and honor FOV perception constraints.

Cooperative target tracking requires a state estimation method. In order to fuse the intermittent measurements coming from one or multiple agents, an Intermittent Kalman Filter implementation inspired by [27] and [38] has been implemented in C++ using the GenoM framework.

In this work a MPC method is proposed for a team of heterogeneous sensing agents to cooperatively track moving targets over time. By introducing a perception objective based on the estimation covariance of a Kalman filter, inspired by previous work [26]. Using results from [33], a Nonlinear Model Predictive Controller has been adapted to include this perception objective. The proposed centralized NMPC method is able to compute the torque level inputs for multiple generic multirotor vehicles in order to cooperative track a target whilst honoring physical actuator constraints.

The proposed NMPC method is able to drive both 1 and 2 identical quadrotors to their optimal sensing configuration, optimal in the sense it minimizes their collective target estimation covariance whilst complying at best with their state setpoints. When using two quadrotors the average estimate covariance is within 20% of the single quadrotor case. It is shown the proposed method is capable of tracking a moving target with both a single and dual quadrotor sensing team. No formal stability proof is given, however statistical evidence indicates the control period of the proposed method is well within the stability margin. The NMPC is able to compute control inputs at an average rate of over 600Hz for a single quadrotor and 300Hz for two quadrotors.

7.1 Limitations and Future Research

No observations have been made so far to support the hypothesis that minimizing the KF covariance objective yields sensing trajectories.Although the proposed NMPC should be able to exploit heterogeneity in the sensing team, no evidence supporting this claim has been presented. Future researchers could look to explore this.

Because the method heavily exploits both the dynamical model of the agents and the target these need to be redefined whenever sensing agent or target characteristics change. The NMPC cost function contains many weights that must be tuned again when a model changes or a different system response is required.

Being a common critique of MPC methods, stability and robustness of the method could be formally investigated by adapting results LQ control theory. To improve robustness an automatic weight tuning method similar to [47] could be employed. The lack of formal stability guarantees necessitate the implementation of failure handling strategies.

Because the computational requirements are large compared to other simpler or greedy methods the onboard use of this method is limited to a subset of generic multi-rotor vehicles that have sufficient onboard processing power.

Future work could include a more rigorous exploration on the choice of perception objective. In this work the Kalman filter covariance is used but the minimization of this appears not to produce sensing trajectories for the agents. As detailed in [12] Ch. 2.3, information theory can provide other means to quantify the perception quality like the mutual information between target state and a measurement to quantify the information gain of a sensor measurement.

When two quadrotors are controlled, the control frequency halves. This indicates poor scaling with increasing number of sensing agents. This can be improved by changing to a distributed scheme. In order to switch to a distributed method the perfect team knowledge and communication assumptions no longer hold and need to be relaxed. Currently it is assumed all agents have perfect knowledge on the state of the other sensing agents and no communication between them is necessary since state estimation and control are both centrally computed. This no longer holds when switching to a distributed scheme, requiring agents to communicate their measurements and reach a consensus on the target state. This could be done using a dynamic consensus filter as detailed in [15].

This also implies each agent must run a separate state estimation scheme to track the other sensing team members. Communication between agents must be taken into account and the effect of time delay on performance and stability should be investigated. Another assumption that could be relaxed is the assumption of perfect model information. Currently it is required each agent knows the model of every other agent. A local system identification method could be used to build up a model of the sensing team members, for instance using a data-driven method analogous to [34]. In this work it is also assumed the target model is known, this could instead be inferred by the agents during operation.

Additional testing is required to see if the method is able to split the tracking task between agents when two targets are tracked. In this work only a single target is considered, future research could attempt to increase this number. To do so the current state estimation scheme must either be extended with data association, or a different state estimation method capable of dealing with multiple targets could be developed. A candidate could be the PHD filter [48] which can deal with a time-varying number of targets.

Bibliography

[1] N. E. Leonard, D. A. Paley, R. E. Davis, D. M. Fratantoni, F. Lekien, and F. Zhang, "Coordinated control of an underwater glider fleet in an adaptive ocean sampling field experiment in monterey bay," *Journal of Field Robotics*, vol. 27, no. 6, pp. 718–740, 2010.

[2] C. Luo, A. P. Espinosa, D. Pranantha, and A. De Gloria, "Multi-robot search and rescue team," *9th IEEE International Symposium on Safety, Security, and Rescue Robotics, SSRR 2011*, pp. 296–301, 2011.

[3] M. Corah, C. O'Meadhra, K. Goel, and N. Michael, "Communication-efficient planning and mapping for multi-robot exploration in large environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1715–1721, 2019.

[4] S. Y. Chen and Y. F. Li, "Automatic Sensor Placement for Model-Based Robot Vision," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 34, no. 1, pp. 393–408, 2004.

[5] R. Alexander, G. Campani, S. Dinh, and F. V. Lima, "Challenges and opportunities on nonlinear state estimation of chemical and biochemical processes," *Processes*, vol. 8, no. 11, pp. 1–27, 2020.

[6] E. F. Camacho and C. Bordons, *Model predictive control*, 2019, vol. 197.

[7] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, "qpOASES: a parametric active-set algorithm for quadratic programming," *Mathematical Programming Computation*, vol. 6, no. 4, pp. 327–363, dec 2014. [Online]. Available: https://link.springer.com/article/10.1007/s12532-014-0071-1

[8] B. N. Vo and W. K. Ma, "The Gaussian mixture probability hypothesis density filter," Tech. Rep. 11, 2006.

[9] C. K. Cowan and P. D. Kovesi, "Automatic sensor placement from vision task requirements," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 3, pp. 407 – 416, 1988.

[10] S. S. Dhillon and K. Chakrabarty, "Sensor placement for effective coverage and surveillance in distributed sensor networks," *IEEE Wireless Communications and Networking Conference, WCNC*, vol. 3, no. C, pp. 1609–1614, 2003.

[11] S. L. Padula and R. K. Kincaid, "Optimization Strategies for Sensor and Actuator Placement," *Contractor*, vol. 23681, no. April, pp. 1–12, 1999. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.30.99&rep=rep1&type=pdf

[12] N. A. Atanasov, "Active information acquisition with mobile robots," *ProQuest Dissertations and Theses*, p. 195, jan 2015. [Online]. Available: http://ezproxy.library.usyd.edu.au/login?url=https://search.proquest.com/docview/1761164196?accountid=14757

[13] N. Atanasov, J. Le Ny, K. Daniilidis, and G. J. Pappas, "Decentralized active information acquisition: Theory and application to multi-robot SLAM," in *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2015-June, no. June. Institute of Electrical and Electronics Engineers Inc., jun 2015, pp. 4775–4782.

[14] T. H. Chung, J. W. Burdick, and R. M. Murray, "A decentralized motion coordination strategy for dynamic target tracking," in *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2006, 2006, pp. 2416–2422.

[15] P. Yang, R. A. Freeman, and K. M. Lynch, "Distributed cooperative active sensing using consensus filters," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2007, pp. 405–410.

[16] K. Sun, B. Schlotfeldt, G. J. Pappas, and V. Kumar, "Stochastic Motion Planning under Partial Observability for Mobile Robots with Continuous Range Measurements," *IEEE Transactions on Robotics*, vol. 37, no. 3, pp. 979–995, 2021.

[17] B. Schlotfeldt, D. Thakur, N. Atanasov, V. Kumar, and G. J. Pappas, "Anytime Planning for Decentralized Multirobot Active Information Gathering," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1025–1032, apr 2018.

[18] B. Schlotfeldt, V. Tzoumas, D. Thakur, and G. J. Pappas, "Resilient Active Information Gathering with Mobile Robots," *IEEE International Conference on Intelligent Robots and Systems*, pp. 4309–4316, mar 2018. [Online]. Available: http://arxiv.org/abs/1803.09730

[19] X. Cai, B. Schlotfeldt, K. Khosoussi, N. Atanasov, G. J. Pappas, and J. P. How, "Non-Monotone Energy-Aware Information Gathering for Heterogeneous Robot Teams," in *IEEE International Conference on Robotics and Automation*, jan 2021. [Online]. Available: http://arxiv.org/abs/2101.11093

[20] B. Schlotfeldt, N. Atanasov, and G. J. Pappas, "Maximum Information Bounds for Planning Active Sensing Trajectories," in *IEEE International Conference on Intelligent Robots and Systems*, 2019, pp. 4913–4920.

[21] G. A. Hollinger and G. S. Sukhatme, "Sampling-based robotic information gathering algorithms," *International Journal of Robotics Research*, vol. 33, no. 9, pp. 1271–1287, jun 2014. [Online]. Available: https://journals.sagepub.com/doi/full/10.1177/0278364914533443

[22] P. Tokekar, V. Isler, and A. Franchi, "Multi-target visual tracking with aerial robots," in *IEEE International Conference on Intelligent Robots and Systems*. Institute of Electrical and Electronics Engineers Inc., oct 2014, pp. 3067–3072.

[23] J. Chen and P. Dames, "Collision-free distributed multi-target tracking using teams of mobile robots with localization uncertainty," Tech. Rep., 2020.

[24] P. Foehn and D. Scaramuzza, "Onboard State Dependent LQR for Agile Quadrotors," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2018, pp. 6566–6572. [Online]. Available: https://youtu.be/8OVsJNgNfa0

[25] A. Iannelli and R. S. Smith, "A Multiobjective LQR Synthesis Approach to Dual Control for Uncertain Plants," *IEEE Control Systems Letters*, vol. 4, no. 4, pp. 952–957, oct 2020.

[26] B. Singh Bal, "Cooperative target tracking for heterogeneous robots," UNIVERSITÀ DI BOLOGNA, Bologna, Tech. Rep., 2019.

[27] C. Yang, J. Zheng, X. Ren, W. Yang, H. Shi, and L. Shi, "Multi-Sensor Kalman Filtering with Intermittent Measurements," *IEEE Transactions on Automatic Control*, vol. 63, no. 3, pp. 797–804, 2018.

[28] C. Liu and J. K. Hedrick, "Model predictive control-based target search and tracking using autonomous mobile robot with limited sensing domain," in *Proceedings of the American Control Conference*. Institute of Electrical and Electronics Engineers Inc., jun 2017, pp. 2937–2942.

[29] M. Kamel, M. Burri, and R. Siegwart, "Linear vs Nonlinear MPC for Trajectory Tracking Applied to Rotary Wing Micro Aerial Vehicles," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 3463–3469, 2017.

[30] D. Falanga, P. Foehn, P. Lu, and D. Scaramuzza, "PAMPC: Perception-Aware Model Predictive Control for Quadrotors," *IEEE International Conference on Intelligent Robots and Systems*, pp. 5200–5207, 2018.

[31] G. Li, A. Tunchez, and G. Loianno, "PCMPC : Perception-Constrained Model Predictive Control for Quadrotors with Suspended Loads using a Single Camera and IMU," 2021.

[32] D. Bicego, J. Mazzetto, R. Carli, M. Farina, and A. Franchi, "Nonlinear Model Predictive Control with Enhanced Actuator Model for Multi-Rotor Aerial Vehicles with Generic Designs," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 100, no. 3-4, pp. 1213–1247, dec 2020.

[33] M. Jacquet and A. Franchi, "Motor and Perception Constrained NMPC for Torque-Controlled Generic Aerial Vehicles," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 518–525, 2021.

[34] G. Torrente, E. Kaufmann, P. Fohn, and D. Scaramuzza, "Data-Driven MPC for Quadrotors," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3769–3776, 2021. [Online]. Available: http://arxiv.org/abs/2102.05773

[35] B. B. Carlos, T. Sartor, A. Zanelli, G. Frison, W. Burgard, M. Diehl, and G. Oriolo, "An Efficient Real-Time NMPC for Quadrotor Position Control under Communication Time-Delay," *16th IEEE International Conference on Control, Automation, Robotics and Vision, ICARCV 2020*, pp. 982–989, 2020.

[36] H. Zhu, F. M. Claramunt, B. Brito, and J. Alonso-Mora, "Learning Interaction-Aware Trajectory Predictions for Decentralized Multi-Robot Motion Planning in Dynamic Environments," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2256–2263, 2021. [Online]. Available: http://arxiv.org/abs/2102.05382

[37] G. Michieletto, M. Ryll, and A. Franchi, "Fundamental Actuation Properties of Multirotors: Force-Moment Decoupling and Fail-Safe Robustness," *IEEE Transactions on Robotics*, vol. 34, no. 3, pp. 702–715, 2018.

[38] B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M. I. Jordan, and S. S. Sastry, "Kalman filtering with intermittent observations," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1453–1464, 2004.

[39] A. Budiyanto, A. Cahyadi, T. B. Adji, and O. Wahyunggoro, "UAV obstacle avoidance using potential field under dynamic environment," *ICCEREC 2015 - International Conference on Control, Electronics, Renewable Energy and Communications*, pp. 187–192, 2015.

[40] B. Lindqvist, S. S. Mansouri, A. A. Agha-Mohammadi, and G. Nikolakopoulos, "Nonlinear MPC for Collision Avoidance and Control of UAVs with Dynamic Obstacles," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6001–6008, 2020.

[41] A. Mallet, C. Pasteur, M. Herrb, S. Lemaignan, and F. Ingrand, "GenoM3: Building middleware-independent robotic components," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2010, pp. 4627–4632.

[42] Y. Chen, M. Bruschetta, E. Picotti, and A. Beghi, "Matmpc-A matlab based toolbox for real-time nonlinear model predictive control," *2019 18th European Control Conference, ECC 2019*, pp. 3365–3370, jun 2019.

[43] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi: a software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, 2019.

[44] N. Koenig and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator," *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 3, pp. 2149–2154, 2004.

[45] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014. [Online]. Available: http://dx.doi.org/10.1016/j.patcog.2014.01.005

[46] C. Beder and R. Steffen, "Determining an initial image pair for fixing the scale of a 3D reconstruction from an image sequence," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 4174 LNCS. Springer Verlag, 2006, pp. 657–666.

[47] M. B. Shadmand, S. Jain, and R. S. Balog, "Autotuning Technique for the Cost Function Weight Factors in Model Predictive Control for Power Electronic Interfaces," *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 7, no. 2, pp. 1408–1420, 2019.

[48] P. M. M. Dames, "Distributed multi-target search and tracking using the PHD filter," *Autonomous Robots*, vol. 44, no. 3-4, pp. 673–689, mar 2020. [Online]. Available: https://doi.org/10.1007/s10514-019-09840-9