Multi-target Tracking via Mixed Integer Optimization

by

Zachary Clayton Saunders

B.S. Operations Research, United States Air Force Academy (2014)

Submitted to the Sloan School of Management
in partial fulfillment of the requirements for the degree of

Master of Science in Operations Research

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2016

Author .....
Sloan School of Management
May 13, 2016

Certified by .....
Sung-Hyun Son
Assistant Group Leader, Lincoln Laboratory Group 36
Thesis Supervisor

Certified by .....
Dimitris Bertsimas
Boeing Professor of Operations Research
Co-Director, Operations Research Center
Thesis Supervisor

Accepted by .....
Patrick Jaillet
Dugald C. Jackson Professor
Department of Electrical Engineering and Computer Science
Co-Director, Operations Research Center

Multi-target Tracking via Mixed Integer Optimization

by

Zachary Clayton Saunders

Submitted to the Sloan School of Management
on May 13, 2016, in partial fulfillment of the
requirements for the degree of
Master of Science in Operations Research

Abstract

Given a set of detections of targets over several time periods, we address in this paper the multi-target tracking problem (MTT) of optimally assigning detections to targets and estimating the trajectory of the targets over time. MTT has been addressed in the literature via predominantly probabilistic methods. In contract we propose mixed integer optimization (MIO) models and local search algorithms that are (a) scalable as they provide near optimal solutions for red six targets and ten time periods in milliseconds to seconds, (b) general as they make no assumptions on the data, (c) robust as they can accommodate missed and false detections of the targets, (d) easily implementable as they use at most two tuning parameters. We evaluate the performance of the new methods using a new metric for complexity of an instance and find that they provide high quality solutions reliably and fast for a large range of scenarios, providing a promising approach to the area of MTT.

Thesis Supervisor: Sung-Hyun Son
Title: Assistant Group Leader, Lincoln Laboratory Group 36

Thesis Supervisor: Dimitris Bertsimas
Title: Boeing Professor of Operations Research
Co-Director, Operations Research Center

Acknowledgments

I would like to thank everyone who played a role in making this opportunity possible, everyone who supported me throughout this process, and everyone who influenced this project in any manner. Though I unfortunately do not have the space to thank each of you by name, I am immensely grateful for each and every one of you and the impacts you have had on me and this work.

I would like to thank my advisor, Professor Dimitris Bertsimas, for his ongoing support and guidance throughout this project. Without your direction and ideas none of this would have been possible. I thank you for constantly challenging me to push myself and for propelling me to expand my academic prowess beyond what I could have thought possible for myself. Additionally, I would like to thank Shimrit Shtern for her guidance and counseling on this project as well. Thank you for taking the time to pour over my error ridden scripts, edit this paper, and provide further guidance at critical points of this project.

Thank you to everyone at Lincoln Laboratories who played a role in making this degree possible. To Mr John Kuconis, thank you for facilitating this opportunity and generously sponsoring me through a military fellowship. To my advisor, Sung-Hyun Son, I also want to thank you for providing me the opportunity to be a Lincoln Laboratory Military Fellow in Group 36. Additionally, thank you for introducing me to the MTT problem and encouraging me to explore a field of the Air Force that was new to me, something that will no doubt pay dividends in my future as an officer in the Air Force. To Steve Relyea, thank you for meeting with me regularly and helping me weed through the details of the MTT problem, repeatedly scratching out ideas on paper. Your advice and insight was critical to both getting this project up and running and keeping it on track throughout the process. Furthermore, I would like to thank the Lincoln Laboratories LLGrid team for their support in running my experiments. As a first time Linux user, your assistance was instrumental in running my simulations and gathering my results.

Finally, I want to thank my friends and family. To my family, Mom, Dad, Tess,

Mathew, and Molly, thank you for your endless love and support and for constantly reminding me that hard work and due diligence always pays off. Without your continual support I would not be where I am today. I would also like to thank the truly extraordinary students at the MIT Operations Research Center who not only provided me with lasting friendships but also supported me academically both in the classroom and on this project.

CONTRACT ACKNOWLEDGMENT: This material is based upon work supported by the Air Force under Air Force Contract No. FA8721-05-C-0002 and/or FA8702-15-D-0001. The views expressed in this document are those of the author and do not reflect the official policy or position of the United States Air Force, the United States Department of Defense or the United States Government.

Contents

List of Figures

THIS PAGE INTENTIONALLY LEFT BLANK

List of Tables

THIS PAGE INTENTIONALLY LEFT BLANK

Chapter 1

Introduction

Multi-target tracking is the problem of estimation the state of multiple dynamic objects, referred to as *targets* over a fixed window of time. At various points of time within the window, the targets are observed in a *scan*, resulting in a set of *detections*. From these detections, the multi-target tracking problem aims to extract information about target dynamics.

Solutions to this problem are sought across many civilian and military applications including but not limited to ballistic missile and aircraft defense, space applications, the movement of ships and ground troops, autonomous vehicles and robotics, and air traffic control. Each application has unique attributes and assumptions, and various algorithms have been developed for each. As a result, the field of multi-target tracking has expanded to numerous research venues, and there is a wide range of literature on the topic. A complete overview of all MTT methods, including the classes of algorithms and their variants as well as additional methods not discussed in this paper, can be found in [22]. For a more exhaustive overview of estimation techniques, filtering, gating, and more please see [3] and [4].

The field of multi-target tracking faces two primary challenges: (i) data association and (ii) trajectory estimation. Given a set of sensor detections, the data association problem consists of assigning the detections to a set of targets. Alternatively, this can be viewed as a labeling problem in which each detection needs to be labeled with a target identifier. The association problem is further complicated when sensors

fail to report detections (missed detection) or incorrectly report detections (false alarm), resulting in ambiguity in the number of existing targets. The trajectory estimation problem consists of estimating the state space of a target (*i.e.*, position, velocity, acceleration, size, etc.) from the associated detections of the aforementioned assignment problem. Even when all of the associations are known, the estimation problem is challenging due to the presence of measurement noise. The two problems of data association and trajectory estimation are closely related and dependent on one another.

Some classical algorithms treat the data association and trajectory estimation problems separately using a combination of probabilistic approaches to determine data associations and filters to estimate trajectories. One such algorithm is the global nearest neighbor (GNN). The GNN algorithm is a naive 2-D assignment algorithm, which evaluates one scan of detections at a time, globally assigning the nearest detection at each scan [7]. Once the data association has been determined, the detections are often passed through one of numerous filters, most commonly a Kalman filter [15], which updates the trajectory estimates before the algorithm progresses forward to the next scan. This process repeats sequentially through each scan of data.

Modern algorithms in the field of multi-target tracking are most commonly statistically based, often relying on heavy probabilistic assumptions about the underlying target dynamics or detection process. The two most prevalent statistical algorithms in the field of multi-target tracking are the Multiple Hypothesis Tracker (MHT) and the Joint Probability Data Association Filter (JPDAF) and their numerous variants and extensions. Both classes of algorithms attempt to solve the data association problem by generating a set of potential hypotheses, or possible detection-to-track assignments. Here a *track* is a set of labeled detections belonging to the same target. Probabilities are assigned to each hypothesis based on the likelihood of the trajectory's existence, and numerous approaches for accomplishing this task have been proposed.

The MHT, first proposed by Reid in [23], assigns likelihood values to hypotheses using a Bayesian maximum a posteriori estimator, which requires probabilistic assumptions on both object dynamics and detection process. This algorithm is gen-

erally considered to be the modern standard for solving the data association problem. Many variants have been proposed for implementation which leverage techniques such as clustering, gating, hypothesis selection, hypothesis pruning, and merging of state estimates. Many of these methods are summarized in [8].

While the MHT has seen various forms of success, it faces several key challenges. Namely, the curse of dimensionality and complexity. The number of possible hypotheses grows exponentially with the number of potential tracks and the number of scans. Consequently, it is considered intractable for large scenarios. Moreover, the MHT might require extensive tuning and thus may be difficult to implement in practice, in addition to being computationally expensive. For these reasons, it is generally considered to be one of the most complex MTT algorithms.

A Probability Data Association (PDA) takes a Bayesian approach to solving the data association problem by finding detection-to-target assignment probabilities via a posterior PDF, which again requires heavy assumptions on object dynamics and the detection process. In similar fashion, a Joint PDA (JPDA) assigns probabilities that are computed *jointly* across all targets. The JPDAF is an algorithm which implements the JPDA along with filters and estimation methods as discussed previously in [3].

A limited number of optimization based algorithms have been applied to solve the MTT problem, most of which attempt to solve by mapping the measurement set onto a trellis and seek the optimal measurement association sequence. Some examples include the Multi-Target Viterbi [27] and an extension in [11] which formulates [27] as a network flow, reducing the solve time from exponential to polynomial. Still others, in particular [20], have suggested adaptations of this approach that output a single best set of tracks, or a list of best sets of tracks, similarly to the MHT.

Compared to the number of statistically based algorithms in the MTT literature, optimization based algorithms are relatively scarce. In fact, most occurrences of optimization in the MTT literature propose the use of optimization to leverage statistical algorithms, in particular, the MHT. For example, integer optimization has been used to improve MHT hypothesis selection by solving an assignment problem which chooses the best hypothesis, but only after costs have been assigned (statistically

based) and hypotheses have been pruned [18]. Somewhat similarly, linear optimization has also been used to assist in the hypothesis selection process for the MHT [10]. Still, other attempts aim to improve the MHT hypothesis selection process via Lagrangian relaxation [21].

More recently, Andriyenko and Schindler have proposed formulating the MTT problem as a minimization of a continuous energy [2], and then again as a minimization of discrete-continuous energy [1]. These algorithms aim to more accurately represent the nature of the problem, but sacrifice interpretability for complexity in the process. Rather than formulating the problem to lend it easily to traditional global optimization methods, the authors aim to leverage the use of optimization techniques to find strong local minima of their proposed energy objective, and they achieve strong results in doing so. However, this approach calls for the use of several parameters that must be tuned and few recommendations are provided for how to go about such a tuning process. Additionally, these methods require initialization heuristics to begin the solving process, which is in itself complicated to implement and is not directly connected to the optimization problem solved.

In this paper, we propose the use of mixed integer optimization (MIO) to formulate and solve the multi-target tracking problem. Although MIOs are generally thought to be intractable (NP-Hard), in many practical cases near optimal solutions and even optimal solutions to these problems can be obtained very efficiently [16]. This can be attributed to the fact that MIO solvers have seen significant performance improvements in recent years due to advancements in both methodology and hardware. The development of new heuristic methods, discoveries in cutting plane theory, and improved linear optimization methods have all contributed to improvements in performance [6]. Modern solvers such as Gurobi and CPLEX have been shown to perform extremely well on benchmark tests. In the past six years alone, Gurobi has seen performance improvements by a factor of 48.7 [12]. CPLEX saw improvements by a factor of 29,000 from 1991 to 2007 [19]. From 1994 to 2014, the growth of supercomputing power as recorded by the TOP500 list has improved by a factor of 567839 [26]. Thus, the total combined effective improvement of software

and hardware advancements is on the scale of 800 billion times in the past 25 years [5].

The literature is also lacking in performance metrics for the evaluation of MTT algorithms. There is no standard method of measuring scenario complexity or algorithm performance as a function of this complexity. In many cases, only the sensor's detection noise is taken into account and other factors such as target density are negated. Recent work [24] proposes a mathematically rigorous performance metric for measuring the distance between ground truth and estimated track, but there is not much attention given to the complexity of generated scenarios. In this paper, we also suggest measures of complexity and performance which are related to the ones suggested in [24], but we show the value in relating a complexity measure to performance measures, namely that it allows us to evaluate the data association and trajectory estimation problems separately. We evaluate the methods suggested in this paper using these complexity and performance measures on two simulated experiments.

The main contributions of this paper are as follows:

(i) We introduce a simple interpretable MIO model which solves the data association and trajectory estimation problems simultaneously for a sensor with no detection ambiguity. The model does not require assumptions on data generation or any tuning of parameters. This MIO is practically tractable, in the sense that it obtains high quality solutions in reasonable amount of time for the considered applications.

(ii) We propose a simple local search heuristic, motivated by the optimization problem, which provides feasible solutions to this problem and show how it can be used as warm start to the MIO in order to improve the quality of the solutions obtained as well as the running time. This heuristic is highly scalable and parallelizable, solving in milliseconds.

(iii) We extend this basic MIO model and corresponding heuristic initialization algorithm for the case of detection ambiguity, i.e., the case where there are both missed detections and false alarms, keeping interpretability while only adding

two tunable parameters, as well as provide general guidelines as how to tune these parameters.

(iv) We present a new measure of complexity for the data association problem and show how it allows scenario generation and performance to be measured separately in each of their own natural demands. We also discuss a simplified measure of performance for the trajectory estimation problem.

The paper structure is as follows. We begin with a description of the MTT problem as we wish to model it in Chapter 2. In Chapter 3 we introduce a simple MIO formulation for a sensor with no detection ambiguity and extend it to a generalized formulation. Then we present a randomized local search heuristic in Chapter 4, which we use as a warm start for the MIO. In Chapter 5 we discuss extensions to both the MIO model and the heuristic for the case of detection ambiguity. In order to quantify the performance of our suggested methods, we develop metrics for measuring scenario complexity and algorithm performance in Chapter 6. Experimental methods and computational results are presented in Chapter 7, including results for scenarios both with and without detection ambiguity. Finally, we summarize our contributions and identify future work in Chapter 8.

General Notations: Unless specified otherwise, $\|\cdot\|$ is used to indicate the ℓ_1 norm, and $|\cdot|$ refers to element wise absolute value.

Chapter 2

Problem Description

In this paper, we restrict our exploration of the MTT problem to the automatic tracking of multiple, independent point targets using a single sensor. A *target* is the object of interest. A point target's only identifiable attributes are its state space, which we restrict to position and velocity. The state space fully defines the field of *trajectories*, or paths along which targets travel. A *detection* is collected from each target at sequential scans. Detections are subject to noise. We treat two general scenarios: with and without detection ambiguity.

When there is no detection ambiguity, the sensor produces exactly one detection for each target in each scan, and there is no other source of detections. Therefore, the number of detections in each scan exactly equals the number of targets in existence. Under these conditions, the data association problem reduces to a one-to-one assignment problem. Our basic optimization model, presented in Chapter 3 aims to model this variant of the MTT problem.

Detection ambiguity refers to a more complex case where the sensor generates both false alarms and missed detections. A *false alarm* occurs when a detection is collected when in fact no target exists. This could be the result of measurement error or difficulties in the sensor's signal processing. A *missed detection* occurs when a data point is not collected in a given scan where a target does in fact actually exist. Therefore, in presence of detection ambiguity, the number of detections in each scan could be higher or lower than the actual number of existing targets, and the number of

targets can not be immediately deduced from the number of detections. Under these conditions, each detection can be assigned to either a target, in the same manner as before, or classified as a false alarm. Furthermore, we wish to identify the location (scan and target ID) of a missed detection. In Chapter 5 we make extensions of the formulation of our basic optimization model to a robust formulation that deals with this detection ambiguity, and we will refer to this formulation as the robust MIO model.

Throughout the paper we make the following assumptions:

Assumption 1.

(i) All targets have constant velocity. i.e., targets do not maneuver and no outside forces act on them.

(ii) Each target's dynamics are independent of one another.

(iii) The number of targets remains constant throughout the window of observation, i.e., there is no birth/death of targets.

(iv) The detection errors are independent of one another.

Notation: We observe P targets over a fixed time window in which T scans are collected. Without loss of generality, and for ease of notation, we assume the scans arrive at a fixed rate of 1Hz, such that the set of scans can be time stamped by $\{1, 2, ..., T\}$. The i^{th} detection of the t^{th} scan is indicated by x_{it}, such that a scan of data at time t is the unordered set of detections $\mathcal{X}_t = \{x_{1t}, x_{2,t}, ..., x_{P,t}\}$. The data for the problem is the ordered set of scans $\boldsymbol{\mathcal{X}} = (\mathcal{X}_1, \mathcal{X}_2, ..., \mathcal{X}_T)$. The state space of target trajectories is paramaterized by a true initial position α_j^{true} and a true constant velocity β_j^{true}.

Chapter 3

MIO Model

In this chapter, we deal with the case of no detection ambiguity. Therefore, we add the following, more restrictive assumptions, to those presented in Assumption 1:

Assumption 2.

(i) The sensor generates exactly one detection for each target at each scan i.e., no missed detections.

(ii) The sensor does not generate any additional detections i.e., no false alarms.

A consequence of Assumption 2 is that the number of detections at each scan will be constant and equal to the number of targets. This seemingly simple point is critical to developing models in the case of no detection ambiguity. We begin constructing our MIO model by introducing decision variables that define data associations as well as estimated trajectories. Using these decision variables, we then develop an objective function that mathematically quantifies the value of the model decisions. Finally, we restrict these variables using constraints that force the model to find solutions that are feasible for the MTT problem as we have defined it. A simple model is first developed step by step in the coming sections before generalized formulation is presented.

3.1 Decision Variables

The data association and trajectory estimation problems require unique decision variables. Because these two problems lie in different domains, the variables we use to represent these decisions also differ. First, we introduce *continuous* decision variables $\alpha_j \in \mathbb{R}^n$ and $\beta_j \in \mathbb{R}^n$ to represent the estimated initial position and velocity, respectively, of each trajectory j. In our interpretation of the MTT problem we allow the trajectory parameters to lie anywhere in the real-continuous domain. For the data association problem, we wish to assign detections to trajectories, a naturally discrete problem. Therefore, we introduce binary decision variables y_{itj} to indicate whether detection x_{it} is assigned to trajectory j or not:

$$y_{itj} = \begin{cases} 1, & \text{if detection } x_{it} \text{ is assigned to trajectory } j, \\ 0, & \text{otherwise.} \end{cases} \tag{3.1}$$

Next, we use these decision variables to develop an objective function to score the solutions found by the model.

3.2 Objective Function

Here, we would like to develop a function that quantifies the quality of a feasible solution. Our goal is to produce a single measure for both the data association and the trajectory estimation problems. For a given assignment and a given estimated trajectories we define the quality of the estimation as the distance of each detection from the estimated position of its associated trajectory. Let $\hat{x}_{jt}$ denote the estimated location of target j at time t, then, if at scan t detection x_{it} is associated with trajectory j then, the distance

$$\|x_{it} - \hat{x}_{jt}\|,$$

is the measure of the quality of estimation for trajectory j at scan t, and the total estimation quality for a given association will be given as

$$\sum_{j=1}^{P}\sum_{t=1}^{T}\left\|\sum_{(i,j)\in\mathcal{A}_t} x_{it} - \hat{x}_{jt}\right\|, \tag{3.2}$$

where $\mathcal{A}_t$ is pairs of detection-trajectory assignments made for scan t.

We can now separate the problem into two parts: given an assignment finding the estimated trajectories which minimizes (3.2), and finding the assignment which results in the best estimated trajectories. Recall that each trajectory is defined by two parameters, α_j^{true} and $\beta_j^{\text{true}}t$ such that the true location is given as

$$\bar{x}_{jt} = \alpha_j^{\text{true}} + \beta_j^{\text{true}}t. \tag{3.3}$$

Thus, an estimated trajectory can analogously be defined by α_j and β_j such that its estimated location at the time of scan t is given by

$$\hat{x}_{jt} = \alpha_j + \beta_j t. \tag{3.4}$$

Therefore, given a full assignment $\mathcal{A} = (\mathcal{A}_1, \ldots, \mathcal{A}_T)$, the trajectory which has the best estimation error is given by the solution of the following optimization problem

$$\underset{\alpha_j,\beta_j}{\text{minimize:}} \sum_{j=1}^{P}\sum_{t=1}^{T}\left\|\sum_{(i,j)\in\mathcal{A}_t} x_{it} - (\alpha_j + \beta_j t)\right\|. \tag{3.5}$$

Notice that under the current assumptions, in which there is no detection ambiguity, (3.5) is the cost of assignment $\mathcal{A}$.

Now we turn to the problem of choosing the assignment, based on this measure. To this end we formulate the assignment cost (3.5) in terms of our decision variables. Note that $(i,j) \in \mathcal{A}_t$ if and only if $y_{itj} = 1$, and because all detections will be assigned

to a trajectory and vice versa, the following equivalence holds

$$\sum_{(i,j)\in\mathcal{A}_t} x_{it} = \sum_{i=1}^{P} y_{itj} x_{it}. \tag{3.6}$$

Making the appropriate substitutions, the cost of an assignment described by variables y_{itj} is given as

$$\underset{\alpha_j,\beta_j}{\text{minimize:}} \ \sum_{j=1}^{P}\sum_{t=1}^{T} \left\| \sum_{i=1}^{P} y_{itj} x_{it} - (\alpha_j + \beta_j t) \right\|. \tag{3.7}$$

Therefore, in order to find the assignment with the lowest cost, we are left to minimize cost (3.7) over all assignments, and obtain the following final objective:

$$\underset{y_{itj},\alpha_j,\beta_j}{\text{minimize:}} \ \sum_{j=1}^{P}\sum_{t=1}^{T} \left\| \sum_{i=1}^{P} y_{itj} x_{it} - (\alpha_j + \beta_j t) \right\|. \tag{3.8}$$

At this point it is necessary to discuss the advantages and disadvantages of the two natural distance measures (norms) that will be considered: the ℓ_1 and the ℓ_2 norms. The ℓ_1 norm has the advantage that it can be reformulated using linear optimization (through the addition of continuous variables and constraints), and it is well known to be more robust to outliers. Furthermore, existing algorithms for MIO are more well developed for linear rather than quadratic optimization. However, the ℓ_2 norm squared form, which is equivalent to the residual sum of squares (RSS), has the advantage that it can be quickly computed using simple linear algebra, making it more amenable to a heuristic. This concept will be discussed further in Chapter 4.

Because of the computational benefits of linear optimization over quadratic optimization, we choose to formulate the objective using the ℓ_1 norm. Therefore, we now show how (3.8) can be reformulated using linear optimization in the case of the ℓ_1

norm by introducing continuous variables $\psi_{jt} \in \mathbb{R}^n$ and the following constraints:

$$\sum_{i=1}^{P} y_{itj} x_{it} - \alpha_j - \beta_j t \leq \psi_{jt}, \qquad \forall j, t, \tag{3.9}$$

$$-\left(\sum_{i=1}^{P} y_{itj} x_{it} - \alpha_j - \beta_j t\right) \geq \psi_{jt} \qquad \forall j, t. \tag{3.10}$$

The resulting objective function for the case of the ℓ_1 norm would then be:

$$\underset{\psi_{jt}}{\text{minimize:}} \sum_{j=1}^{P} \sum_{t=1}^{T} \psi_{jt}. \tag{3.11}$$

3.3 Constraints

In addition to the constraints used to linearize the objective function, we also require standard assignment constraints to ensure that only one detection is assigned to a target and vice versa. Specifically, for each scan, each detection x_{it} must be assigned to exactly one target j:

$$\sum_{j=1}^{P} y_{itj} = 1 \qquad \forall i, t. \tag{3.12}$$

Similarly, for each scan, each target must be assigned exactly one detection:

$$\sum_{i=1}^{P} y_{itj} = 1 \qquad \forall j, t. \tag{3.13}$$

3.4 Overall Formulation

Integrating all of these elements together, we arrive at the following MIO model:

$$
\begin{aligned}
\underset{\psi_{jt}}{\text{minimize:}} \quad & \sum_{j=1}^{P}\sum_{t=1}^{T}\psi_{jt} && && (3.14)\\
\text{subject to:} \quad & \sum_{j=1}^{P} y_{itj} = 1 && \forall i,t\\
& \sum_{i=1}^{P} y_{itj} = 1 && \forall j,t\\
& \sum_{i=1}^{P} y_{itj}x_{it} - \alpha_j - \beta_j t \leq \psi_{jt} && \forall j,t\\
& -\left(\sum_{i=1}^{P} y_{itj}x_{it} - \alpha_j - \beta_j t\right) \geq \psi_{jt} && \forall j,t\\
& y_{itj} \in \{0,1\} \quad \forall i,t,j\\
& \alpha_j \in \mathbb{R}^n \quad \forall j, \quad \beta_j \in \mathbb{R}^n \quad \forall j.
\end{aligned}
$$

This formulation is simple in the sense that it involves few variables and constraints, making it highly interpretable and easily implementable. However, it has the disadvantage of being ill suited for extensions to detection ambiguity because it heavily relies on the fact that exactly one of the detections at each scan is associated to a target. This forces the term $\sum_{i=1}^{P} y_{itj}x_{it}$ to always be equal to one of the detections. However, in the case of detection ambiguity, this no longer holds true, since there might be trajectories which are not associated with a detection in a given scan, resulting in an unintended cost to the assignment. Therefore, in the following section we present a more generalized formulation, which is amenable to scenarios with detection ambiguity.

3.5 Generalized Formulation

Here we modify (3.14) so that it can be easily extended to handle false alarms and missed detections that occur in scenarios with detection ambiguity. We previously identified that the main issue with (3.14) arises from the fact that $\sum_{i=1}^{P} y_{itj} x_{it}$ will always incur a cost in the objective function. However, we only wish to account for this assignment cost when a target has actually been assigned a detection. To this end, we introduce a new variable z_{jt} to substitute for this term. We force this variable to take the value x_{it} when $y_{itj} = 1$ and some arbitrary number when $y_{itj} = 0$.

$$z_{jt} = \begin{cases} x_{it}, & \text{if } y_{itj} = 1, \\ \textit{free}, & \text{otherwise.} \end{cases}$$

In the case of no detection ambiguity, $\sum_{i=1}^{P} y_{itj} = 1$ will always hold true as forced by (3.13), and as a result, we recover the original objective function exactly because z_{jt} will always take on the value of exactly one of the x_{it}.

Additionally, this logical condition can be translated to a model constraint through the following constraint:

$$M_t(1 - y_{itj}) \geq |z_{jt} - x_{it} y_{itj}| \qquad \forall i, t, j. \tag{3.15}$$

where $M_t = \max_j |x_{it}|$ for each scan. Furthermore, we can linearize (3.15) by substituting it for the following two linear constraints:

$$\begin{aligned} x_{it} y_{itj} + M_t(1 - y_{itj}) &\geq z_{jt} \qquad \forall i, t, j, \\ x_{it} y_{itj} - M_t(1 - y_{itj}) &\leq z_{jt} \qquad \forall i, t, j. \end{aligned}$$

We must also adjust (3.8) to account for this change as follows:

$$\underset{z_{jt}, \alpha_j, \beta_j}{\text{minimize:}} \quad \sum_{j=1}^{P} \sum_{t=1}^{T} \|z_{jt} - \alpha_j - \beta_j t\|. \tag{3.16}$$

This objective can be linearized in the same fashion as before by again introducing continuous variables ψ_{jt} and additional constraints as follows:

$$\underset{\psi_{jt}}{\text{minimize:}} \quad \sum_{j=1}^{P}\sum_{t=1}^{T}\psi_{jt} \tag{3.17}$$

$$z_{jt} - \alpha_j - \beta_j t \leq \psi_{jt} \qquad \forall i, j, t, \tag{3.18}$$

$$-(z_{jt} - \alpha_j - \beta_j t) \geq \psi_{jt} \qquad \forall i, j, t. \tag{3.19}$$

Again, we consolidate these elements together and arrive at the following generalized MIO model:

$$\begin{aligned}
\underset{\psi_{jt}}{\text{minimize:}} \quad & \sum_{j=1}^{P}\sum_{t=1}^{T}\psi_{jt} && && (3.20)\\
\text{subject to:} \quad & \sum_{j=1}^{P} y_{itj} = 1 && \forall i, t\\
& \sum_{i=1}^{P} y_{itj} = 1 && \forall j, t\\
& x_{it}y_{itj} + M_t(1 - y_{itj}) \geq z_{jt} && \forall i, t, j\\
& x_{it}y_{itj} - M_t(1 - y_{itj}) \leq z_{jt} && \forall i, t, j\\
& z_{jt} - \alpha_j - \beta_j t \leq \psi_{jt} && \forall i, j, t\\
& -(z_{jt} - \alpha_j - \beta_j t) \geq \psi_{jt} && \forall i, j, t\\
& y_{itj} \in \{0, 1\} \quad \forall i, t, j\\
& \alpha_j \in \mathbb{R}^n \quad \forall j, \quad \beta_j \in \mathbb{R}^n \quad \forall j, \quad z_{jt} \in \mathbb{R}^n \quad \forall j, t.
\end{aligned}$$

Note that (3.14) and (3.20) are exactly identical formulations when detection ambiguity does not exist. Throughout the remainder of this paper, we refer to (3.14) as the *basic* MIO. In Chapter 5 we will extend (3.20) to account for false alarms and missed detections under sensor ambiguity.

Chapter 4

Local Search Heuristic

In this chapter, we present a detailed description of a heuristic that finds high quality feasible solutions for problem (3.20). These solutions can be used as a warm start to the MIO, providing a performance boost to the MIO. The heuristic leverages the power of randomized local search methods to find locally optimal solutions.

The fundamental concept behind randomized local search methods is to begin with a randomized starting point and through local improvements converge to a locally optimal solution. By applying this scheme to a growing number of randomized starting points, the probability of reaching high quality solutions, or even the globally optimal solution, increases.

We now detail the heuristic mechanism for a single starting point. The heuristic *starting point* is a randomized solution to the assignment problem, that satisfies the assignment equations (3.12) and (3.13). This is done for each scan, by choosing a permutation over the detections uniformly at random, and associating to trajectories by the order of the permutation, i.e., if in the random permutation detection i is at location j than it will be assigned to trajectory j. The assignment cost, temporarily denoted by f, of this starting point is then calculated by solving (3.7). After initializing the heuristic starts a *sweep* through the scans, i.e., a single pass through the scans. At each stage of the sweep two detections are randomly selected from the current scan and they exchange assignments, an operation we call a *swap*, generating a new feasible solution. The assignment cost of this new solution is calculated, and if

it improves the current cost it is kept and otherwise it is rejected. The heuristic will continue to conduct sweeps, until a full sweep is completed without accepting a single swap, in which point it terminates. The full description of the heuristic algorithm for a single starting point is given in Figure 4-1, which is also referred to as the *basic* heuristic throughout the course of this paper.

Algorithm 1 Randomized local search with heuristic swaps

Input: $\boldsymbol{\mathcal{X}}$, P, T
Output: f, y
Initialization : Assign random initial assignment to y^0
Calculate $\alpha_j, \beta_j \quad \forall j$
Calculate f^0 - the cost of assignment y^0
swapped $\leftarrow true$
$k \leftarrow 1$
while swapped **do**
  swapped $\leftarrow false$
  for t in $\{t_1, t_2, ..., T\}$ **do**
    Randomly choose $j, m \in \{1, \ldots, P\}$
    Find i, l such that $y^{k-1}_{itm} = 1$ and $y^{k-1}_{ltj} = 1$
    Swap such that $y^k_{itj} \leftarrow 1$, $y^{k-1}_{itm} \leftarrow 0$, $y^k_{ltm} \leftarrow 1$ and $y^k_{ltj} \leftarrow 0$
    Calculate f^k the cost of assignment y^k as well as $\alpha_j, \beta_j, \alpha_m, \beta_m$
    if $(f^k \geq f^{k-1})$ **then**
      $y^k \leftarrow y^{k-1}$
    else
      swapped $\leftarrow true$
    end if
  end for
  $k \leftarrow k + 1$
end while
return f^k, y^k

Figure 4-1: Pseudocode for heuristic for a single starting point.

The goal of any heuristic is to find good feasible solutions in an efficient manner. In particular, the goal of our heuristic is to find good feasible solution for the MIO formulation, which can serve as a warm start for the MIO. In Chapter 3 we discussed our choice to use the ℓ_1 norm over the ℓ_2 norm for use in the objective of our MIO models. We now turn to discuss why the ℓ_2 is the preferred choice for use in this

heuristic. The two main areas of concern are 1) efficiency of the algorithm and 2) quality of the solution.

In the case of the MIO, the preferred objective function was the ℓ_1 norm because it lent itself easily to linear optimization solvers which have known performance advantages over quadratic optimization solvers. However, in the case of the heuristic, the objective function is computed given an assignment, i.e., we need to solve problem (3.7), to obtain the specific assignment's cost, rather than the more difficult problem (3.8). Furthermore, this objective needs to be recomputed after each swap, even if it is eventually rejected, and hundreds to thousands of swaps may be carried out for a single starting point. This fact makes the computational cost of this calculation critical to the scalability of the heuristic. Computing the ℓ_1 norm objective would require solving an linear optimization problem. Even though this linear optimization problem can be computed quite quickly by state of the art optimization solvers, the ℓ_2 norm squared, or RSS, can be computed by simple linear algebra in a fraction of the time, as shown in [14]. Therefore, with respect to efficiency, the ℓ_2 norm is the clear choice for use in the heuristic.

When judging the quality of the heuristic solution, we must look at its purpose. Since it will serve as a warm start for the MIO, which uses the ℓ_1 norm in its objective, we would assume that better solutions will be obtained by using the same norm for the heuristic objective. However, the heuristic will converge to high quality solutions, and since both norms represent measures of distance, and hence are correlated, we assume that a high quality solution as measured by the ℓ_2 norm is also a high quality solution as measured by the ℓ_1 norm. Thus, the choice of the ℓ_2 over the ℓ_1 norm might not significantly degrade the solution quality.

Therefore, although the use of ℓ_2 norm runs the risk of obtaining solutions which are not necessarily extremely good solutions for the ℓ_1 norm objective the potential loss in solution quality is far outweighed by the guaranteed efficiency improvements afforded by the ℓ_2 norm. Finally, the heuristic described above can be parallelized by running partitions of the starting points on separate cores, meaning that a large number of starting points can run in a relatively short amount of time. Increasing

the number of starting points greatly reduces the potential for the heuristic to get stuck at a poor quality solution. Therefore, we make the choice to use the ℓ_2 norm in the objective function of the heuristic.

Chapter 5

Extensions to Detection Ambiguity

We transition to treat the case of detection ambiguity. Specifically, we now allow for false alarms, the instance in which a detection is triggered when no target exists, and missed detections, the instance in which a target exists but no detection is generated. Consequently, the number of detections at each scan varies, and the number of targets we wish to track is now ambiguous. To state this explicitly, we introduce additional notation for the case of detection ambiguity. Let n_t be the number of detections at scan t. We denote

$$N_0 = \min_t n_t$$

as the largest number of detections across all scans, and similarly, we denote

$$N_1 = \max_t n_t$$

as the smallest number of detections across all scans. The only assumption we make in the case of detection ambiguity is that the true number of targets falls somewhere in the range of $[N_0, N_1]$. Particularly, we replace Assumption 2 with the following less restrictive assumption.

Assumption 3.

(i) The sensor generates at most one detection for each target at each scan i.e.,

there can be missed detections.

(ii) The sensor can generate detections which do not originate from any target i.e., there can be false alarms.

(iii) The number of true targets P satisfies $N_0 \leq P \leq N_1$.

Several new challenges emerge in the case of detection ambiguity. First, we now need to estimate the number of targets. We denote the number of estimated targets as P_{est} Second, now each detection must either be assigned to a target, as before, or classified as a false alarm, not both. Finally, for each target we want to identify the scans in which it was not detected.

In this chapter, we develop an approach that reduces the difficulty of estimating the number of targets inherent to this new problem by two different approaches. In the first approach, the number of targets can be determined via the optimization model itself, through the use of additional decision variables and constraints. Alternatively, the second approach we formulate an MIO model that assumes a fixed number of targets P and then use the power of parallelization to run that model with all possible values of P which satisfy Assumption 3, choosing the solution for P that minimizes the total objective function value. These two approaches are equivalent, in the sense they have the same optimal solution set.

However, as it turns out, the first approach leads to a model which is not tractable for practical use, while the second results in smaller more tractable models, which as we stated can be solved in parallel. Therefore, we turn our focus to discuss the second approach and defer the interested reader to Appendix A for a complete discussion of the first approach. We begin by extending our basic MIO model to the case of detection ambiguity before discussing necessary adaptations to the basic heuristic, which will also assume a fixed number of targets and provide good quality warm start solutions to its corresponding MIO.

5.1 Robust MIO with Fixed Number of Targets

In this section, we extend (3.20) to account for missed detections and false alarms through the addition of decision variables and constraints. The objective function is also updated to reflect the necessary additions.

5.1.1 Decision Variables

We need to establish new variables for identifying false alarms as well as missed detections. Toward this goal, we introduce new binary decision variables F_{it} to indicate whether or not a detection x_{it} is a false alarm.

$$F_{it} = \begin{cases} 1, & \text{if detection } i \text{ at time } t \text{ is a false alarm,} \\ 0, & \text{otherwise.} \end{cases}$$

Likewise, we introduce binary decision variables M_{jt} to indicate whether or not trajectory j has a missed detection at time t.

$$M_{jt} = \begin{cases} 1, & \text{if detection for trajectory } j \\ & \text{at time } t \text{ is a missed detection,} \\ 0, & \text{otherwise.} \end{cases}$$

5.1.2 Objective Function

We can easily extend (3.16) to account for detection ambiguity by introducing penalties θ and ϕ for each missed detection and false alarm, respectively. This implies a linear penalty function, meaning that every missed detection (false alarm) is contributes the same penalty to the objective function. Therefore, we simply need to penalize the total number of false alarms and missed detections in the objective function. If we denote the total number of false alarms TF and total number of

missed detections TM, the resulting objective function takes the form:

$$\underset{\psi_{jt}}{\text{minimize:}} \sum_{j=1}^{P} \sum_{t=1}^{T} \psi_{jt} + \theta \cdot TF + \phi \cdot TM. \tag{5.1}$$

As a general rule, we would expect to increase θ or ϕ, as the number of expected false alarms or missed detections decreases, respectively. A more exhaustive discussion on the insight behind these penalties, in addition to recommendations for tuning them can be found in Appendix A.

5.1.3 Constraints

Finally, we must restrict the set of feasible solutions to satisfy the assumptions made earlier. We accomplish this by simply modifying (3.12) and (3.13) to account for false alarms and missed detections, respectively. Specifically, all detections must either be assigned to a trajectory j or to a false alarm:

$$\sum_{j=1}^{P} y_{itj} + F_{it} = 1 \qquad \forall i, t. \tag{5.2}$$

All trajectories j must either be assigned a detection or a missed detection:

$$\sum_{i=1}^{n_t} y_{itj} + M_{jt} = 1 \qquad \forall j, t \tag{5.3}$$

Here we see why it was necessary to generalize the simple model to (3.20). When $M_{jt} = 1$, we have $\sum_{i=1}^{n_t} y_{itj} = 0$ by (5.3). Therefore (3.15) does not restrict z_{jt} at all and it is obvious, from the structure of the objective function, that in this case $z_{jt} = \alpha_j - \beta_j t$ is the optimal solution, since it results in no change to the objective score (no estimation penalty), which is precisely the desired effect. To state this explicitly, we have:

$$z_{jt} = \begin{cases} x_{it}, & \text{if } y_{itj} = 1, \\ \alpha_j - \beta_j t, & \text{otherwise,} \end{cases}$$

in the case of detection ambiguity.

Lastly, to properly penalize false alarms and missed detections in the objective function, we must force TF (TM, respectively) to equal the sum of all false alarms (missed detections, respectively).

$$\sum_{i=1}^{n_t} \sum_{t=1}^{T} F_{it} = TF$$

$$\sum_{j=1}^{P} \sum_{t=1}^{T} M_{jt} = TM \tag{5.4}$$

5.1.4 Full Formulation

Merging all of these elements together we arrive at our MIO model for a case of detection ambiguity and a fixed number of targets. We refer to this model as the

robust MIO.

$$
\begin{aligned}
g(P) = \underset{\psi_{jt}}{\text{minimize:}} \quad & \sum_{j=1}^{P}\sum_{t=1}^{T}\psi_{jt} + \theta \cdot TF + \phi \cdot TM \\
\text{subject to:} \quad & \sum_{j=1}^{P} y_{itj} + F_{it} = 1 \qquad \forall i,t \\
& \sum_{i=1}^{n_t} y_{itj} + M_{jt} = 1 \qquad \forall j,t \\
& \sum_{i=1}^{n_t}\sum_{t=1}^{T} F_{it} = TF \\
& \sum_{j=1}^{P}\sum_{t=1}^{T} M_{jt} = TM \\
& x_{it}y_{itj} + M_t(1-y_{itj}) \geq z_{jt} \qquad \forall i,t,j \\
& x_{it}y_{itj} - M_t(1-y_{itj}) \leq z_{jt} \qquad \forall i,t,j \\
& z_{jt} - \alpha_j - \beta_j t \leq \psi_{jt} \qquad \forall j,t \\
& -(z_{jt} - \alpha_j - \beta_j t) \leq \psi_{jt} \qquad \forall j,t \\
& y_{itj} \in \{0,1\} \quad \forall i,t,j \\
& \alpha_j \in \mathbb{R}^n \quad \forall j, \quad \beta_j \in \mathbb{R}^n \quad \forall j \\
& z_{jt} \in \mathbb{R}^n, \quad \forall j,t.
\end{aligned} \tag{5.5}
$$

P_{est} would then be given by

$$P_{\text{est}} = \min_{N_o \leq P \leq N_1} g(P),$$

and the trajectories and assignments will correspond with the optimal solution of the corresponding model.

5.2 Heuristic with Fixed Number of Targets

The heuristic for the scenario with ambiguity follows closely from the heuristic developed under the scenario without ambiguity, with a few key differences. In the first

place, the process for establishing a random starting point during the initialization process requires refinement. Initial solutions should allow for false alarms and missed detections. With equal probability, each detection is randomly classified as a false alarm or selected to receive a target assignment. Once a detection has been identified for assignment, assignments are randomly selected uniformly across all targets. The remaining scans for which trajectories do not receive an assignment are identified as missed detections.

Once a starting point has been initialized, the heuristic progresses in much the same manner as before. Again, it sweeps through all scans continuously making random swaps. However, the swapping process also change.Specifically, false alarms and missed detections must be included in the switches. but the robust heuristic randomly chooses from the following options when making swaps:

1. Switch detection assignments between two existing targets.

2. Switch the detection assignment of an existing target with a false alarm.

3. Switch the detection assignment of an existing target with a missed detection for a different existing target.

4. Move the detection assignment of an existing target to a false alarm and replace it with a missed detection.

5. Move a false alarm into the location of a missed detection for an existing target.

Similar to the basic heuristic, this robust extension will accept the switch/move if the objective score improves, and reject the switch/move otherwise, and terminates once it completes a full sweep without accepting a single swap.

The framework of this new heuristic, also referred to as the *robust* heuristic, is the same as the one presented in Figure 4-1 with the appropriate modifications in the initialization and swapping steps.

Due to the increase in potential combinations of solutions, we expect this variant of the heuristic to run slightly slower. Though this effect is mitigated by the fact

that this variant is just as parallelizable as the basic heuristic. Next, we shift our discussion to focus on measuring the complexity and performance of our approaches.

Chapter 6

Scenario Complexity & Performance Metrics

In order to measure the quality of solution obtained by any MTT algorithm one needs to have a measure of both performance and scenario complexity. Unfortunately, as stated in [22], there does not exist a unified approach for measuring scenario complexity, nor does there exist clear measures of performance for each of the trajectory estimation and data association problems. In this paper, we argue that the data association problem has a natural performance metric but lacks a measure of complexity, while the trajectory estimation problem has a natural measure of complexity but lacks a clear performance metric, and thus we construct the missing measures for each.

It is natural to assume that the difficulty of a scenario is highly correlated with a sensor property which quantifies the deviation of the detections from the true targets. Thus in order to discuss these measures we must first define σ to be a measure of this sensor property, quantifying how noisy the sensor detections are, and in most cases is the standard deviation of the detection error. We will show how σ can be used to define complexity measures for different scenarios.

6.1 Data Association

In the case of the data association problem, the preferred performance metric often used in practice is % accuracy, *i.e.*, the number of correct detection assignments out of the number of possible correct assignments. For the case without sensor ambiguity, the number of possible assignments is simply the total number of detections, or equivalently, the number of targets multiplied by the number of scans:

$$Accuracy = \frac{\#\text{ correct assignments}}{\text{Total }\#\text{ of detections}} = \frac{\#\text{ correct assignments}}{PT}.$$

In the case of sensor ambiguity, however, the number of possible correct assignments requires a deeper explanation. To develop a better understanding, we consider our goal, which is to correctly assign detections to targets and identify both false alarms and missed detections. With this in mind, we define the number of possible correct assignments as the number of targets multiplied by the number of scans plus the number of false alarms:

$$Accuracy = \frac{\#\text{ correct assignments}}{PT + \#\text{ False Alarms}}.$$

Whereas accuracy serves as a good measure of performance for data association, there does not exist a corresponding measure of complexity which comparatively measures the difficulty of the data association problem. We argue that σ alone is not the best measure of difficulty for the data association problem. For example, for a scenario with very close target trajectories it may be difficult to ascertain data associations even for small σ values, and similarly with high enough σ values even widely spaced targets could be difficult to differentiate. Therefore, we introduce a metric ρ to quantify this complexity. For ease of notation in developing this metric we first define D_{ijt} as the distance between one true trajectory i and another true trajectory j:

$$D_{ijt} = \|\alpha_i^{\text{true}} + \beta_i^{\text{true}} t - \alpha_j^{\text{true}} + \beta_j^{\text{true}} t\|.$$

Additionally, we define a variable c_{ijt} that will take the value of 1 if the distance between trajectory i and trajectory j is greater than some monotonically increasing function of σ which we will denote by $h(\sigma)$:

$$c_{ijt} = \begin{cases} 1, & \text{if } D_{ijt} > h(\sigma), \\ 0, & \text{otherwise.} \end{cases}$$

Then the difficulty of a scenario in the sphere of the data association problem is quantified by the complexity measure ρ, which is the proportion of detection pairs that fall within a closely defined proximity to each other:

$$\rho = \frac{\sum_{t=1}^{T} \sum_{i<j} c_{ijt}}{\binom{P}{2} T}.$$

This metric has several desirable attributes. First and foremost, it falls within the range of $[0, 1]$, identical to the range of accuracy, making it easily comparable. Secondly, it is easy to understand and interpret. Higher values of ρ indicate easier scenarios because fewer targets are within close proximity for a shorter amount of time, and vice versa. Finally, as we have defined it, ρ has an inverse relationship with σ, which means that it serves as a connection between scenario generation and performance measuring processes. While σ can be used more naturally for scenario generation, where it is useful as a parameter for signal noise, ρ can be calculated after the fact and used to quantify the difficulty of the scenario as it pertains to the data association problem.

6.2 Trajectory Estimation

In the case of the trajectory estimation problem, the preferred complexity metric often used in practice is σ itself. Increasing the noise may often lead to stronger bias in the trajectory estimation, especially in scenarios with fewer scans, and results in a deteriorated quality of the estimation. Therefore, we believe that σ is the correct

metric for use in measuring the difficulty of the trajectory estimation problem.

However, establishing a performance metric for the trajectory estimation problem is necessary. We choose to implement a metric which captures the core goal of the trajectory estimation problem, that is to estimate a trajectory as close as possible to the true ground track.

$$\delta = \frac{\sum_{t=1}^{T} \sum_{j=1}^{P} \|\bar{x}_{jt} - \hat{x}_{jt}\|}{PT}.$$

We match the true trajectories to the estimated trajectories using an one-to-one assignment problem which can be formulated using linear optimization. See Appendix C for more details. Lower values of δ correspond to higher performance because the distance between the estimated and true ground trajectories is smaller.

In the next chapter, we will see how these measures of complexity and performance are useful in quantifying the strengths and weaknesses of our methods.

Chapter 7

Experimental Simulations & Computational Results

We evaluate our approaches on a wide variety of simulated scenarios, comparing the results against two benchmark solutions. For the first benchmark, we randomly generate detection assignments, including randomly assigning false alarms and missed detections for the case of detection ambiguity. This solution will be referred to as the *random* solution. In the second benchmark, the detection assignments are known perfectly, meaning that all assignments are exactly correct including the classification of false alarms and identification of missed detections. This solution is referred to as the *ideal* solution, however, it is only ideal in relation to the data association problem, but it provides a means for bounding the expected error in the trajectory estimation problem. Note that we do not compare our methods to any known MTT algorithms, such as the MHT or JPDAF, due to the complexity in tuning and the overhead of implementation of these algorithms.

In the literature, there does not exist a clearly defined comprehensive set of standard test scenarios, as pointed out by [22], which also notes that two types of scenarios of particular importance include crossing trajectories and parallel trajectories. Because we would like to test our methods across scenarios with a wide range of complexity, for both the data association and trajectory estimation problems, it is necessary to create scenarios using both methods. With this in mind, we choose to

generate scenarios of both trajectory types using a simple methodology that will be outlined next in our discussion on experimental methods.

We run two separate experiments, one with detection ambiguity and one without. Both experiments, including the scenario generation process, heuristic, and MIO, were implemented in the development software *julia* 0.4.3 [25] using the optimization package *JuMP* [17]. The optimization software Gurobi 6.5.0 [13] was used to solve the MIOs, and the optimization processes was restricted to the use of a single core. Each simulation was run on a single compute node of the unclassified TX-Green cluster located at Lincoln Laboratories. The cluster utilizes DL165 G7 compute nodes, consisting of 2.2 GHz compute cores, with 8 GB of RAM each, for a total peak performance of 77.1 TFLOPS [9].

We begin by outlining our experimental methods for scenarios without detection ambiguity and discuss the results of our approaches on these scenarios.

7.1 Scenarios without Detection Ambiguity

In order to evaluate scalability of our algorithms we test our methods across a range of scenarios with varying numbers of targets and scans. In particular we consider $P \in \{4, 6, 8, 10\}$ targets and $T \in \{4, 6, 8, 10\}$ scans. The scans are collected at a rate of 1 Hz. The cartesian product of P and T creates 16 unique scenario sizes. We generate 10 unique crossing scenarios and 10 unique parallel scenarios of each size.

To generate trajectories, we first establish a state space as the segment $[-\tau, \tau]$. For our experiments, we elected for $\tau = 20$. Two points within this state space are selected to define a trajectory, where the first is referred to as the trajectory's *initial position* and the second as the trajectory's *final position*. To generate crossing trajectories, the initial and final positions are randomly selected from within the full range of the state space. To generate parallel trajectories, the state space is divided into P equal non overlapping segments, such that within the ith segment we randomly select the initial and final positions for target i. This ensures the generation of trajectories that

do not cross or overlap, but will remain within close proximity of each other.

For each scenario, we randomly generate 10 realizations of data by first perturbing each true position measurement by an error $\epsilon \sim \mathcal{N}(0, \sigma)$ with $\sigma \in \{0.1, 0.5, 1.0, 2.0, 3.5, 5.0\}$, where σ represents the noise parameter. Adding the detection error to the true position results in a detection:

$$x_{it} = \alpha_i^{\text{true}} + \beta_i^{\text{true}} t + \epsilon.$$

Scans $\mathcal{X}_t$ are simulated by randomizing the order of x_{it} for each t. Each unique $\boldsymbol{\mathcal{X}}$ generated is referred to as a *simulation*. For each such simulation, we run the heuristic with a range of number of starting points $N \in \{100 \;\; 1,000 \;\; 10,000\}$, and use each of these solutions as a warm start for the MIO. The optimization process is set to terminate after 3T seconds, with solutions collected at intervals of $\{1, T, 2T, 3T\}$ seconds.

At the conclusion of the experiment, we calculated the difficulty of each scenario, the accuracy of each solution, and the trajectory estimation error δ. When measuring the difficulty of scenarios in terms of ρ, we propose the use of $h(\sigma) = 2\sigma$, since it is difficult to distinguish detections which lie between target trajectories that are closer.

7.1.1 Scenario Generation

We begin with a discussion on the relationship between ρ and σ and show how this relationship benefits both scenario generation and complexity measuring by allowing each to occur in their own natural domain. Figure 7-1 shows the relationship between σ and ρ for the 20 scenarios simulated in our experiments. The plot is broken down by scenario type between crossing and parallel trajectories.

Remember that higher values of ρ indicate a lower proportion of detections within very close proximity to one another. We note that the parallel method of scenario generation clearly generates easier scenarios, as measured by ρ. This suggests that it may be more difficult to discern correct associations for crossing scenarios than for parallel scenarios. In addition, we can conclude from Figure 7-1 that σ and ρ are

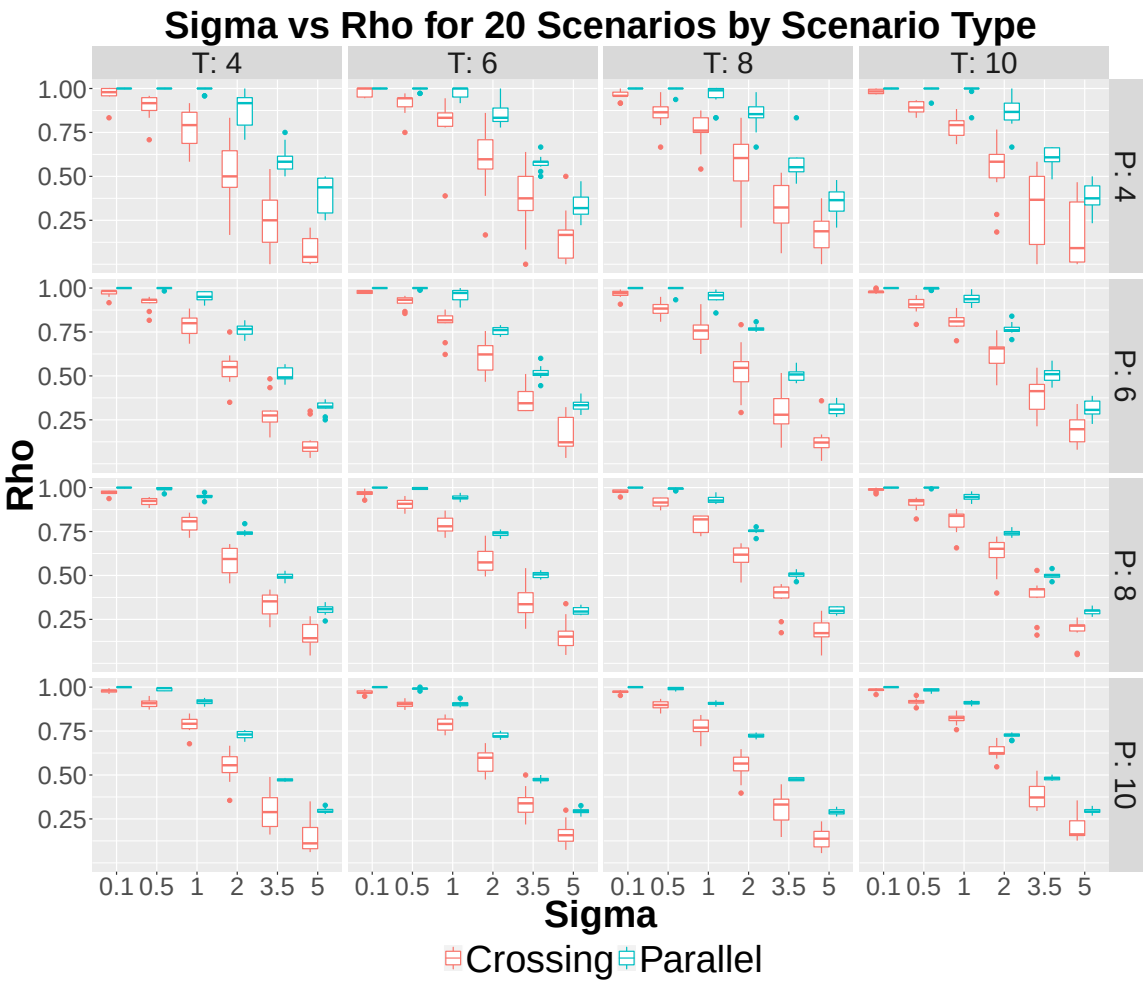

Figure 7-1: Relationship between σ and ρ summarized by scenario type for all 20 generated scenarios in this experiment.

highly correlated, and the higher σ is the lower ρ becomes, which in to be expected from the definition of ρ.

We also note that the range of ρ values corresponding to a single value of σ decreases as the number of targets increases. In other words, as the number of targets increases, it becomes more difficult to generate a wider variety of ρ values from a given σ. For example, for the crossing scenarios with $T = 10$ and $\sigma = 5$, in the case of four targets the range of ρ is approximately while $[0, 0.5]$, while in the case of ten targets it is limited to $[0.125, 0.375]$. Most likely, this is a result of using a fixed state space, as the number of target increases the flexibility of positioning the track decreases.

Although the variety of difficulty measures decreases as the number of targets increases, the measure ρ still provides a meaningful measure of difficulty for the data association problem.

7.1.2 The Basic Heuristic Scalability

We transition to evaluate the scalability of the heuristic run times. Table 7.1 summarizes the minimum, mean, and maximum run times of the heuristic for a single starting point, arranged by the number of targets (P) and number of scans (T). Times are shown in milliseconds.

		Basic Heuristic Run Times (in milliseconds)		
P	T	Min	Mean	Max
4	4	0.07	0.10	0.18
4	6	0.18	0.24	0.38
4	8	0.34	0.45	0.62
4	10	0.58	0.76	1.02
6	4	0.11	0.15	0.25
6	6	0.31	0.39	0.58
6	8	0.64	0.81	1.05
6	10	1.24	1.56	2.02
8	4	0.14	0.19	0.30
8	6	0.46	0.57	0.86
8	8	0.95	1.24	1.58
8	10	2.07	2.53	3.37
10	4	0.19	0.25	0.41
10	6	0.63	0.80	1.03
10	8	1.44	1.84	2.44
10	10	2.96	3.73	4.56

Table 7.1: Heuristic run times (in milliseconds) for a single starting point.

Close examination shows that the heuristic scales more efficiently with increases in the number of targets than increases in the number of scans. For example, increasing from four to six scans for four targets increases the computational cost by 140%, while increasing from four to six targets for four scans increases the computational cost by 50%. The same trend holds true across all targets and scans. Although the heuristic

scales more efficiently in P than T, the heuristic does not exceed 5 milliseconds in all cases per starting point.

The true scalability of the heuristic is fully realized when we consider the power of parallelization. By running the heuristic on several processors, we can reduce the number of starting points run on each processor, and in turn reduce the total running time. As a result, given enough processors, the heuristic can run several thousand starting points and still find solutions in a fraction of a second. To illustrate this, say that we intend to run 50,000 heuristic starting points for a scenario with six targets and six scans. The average run time for a single starting point of this size is about 0.4 milliseconds. Running all of these starting points in sequence would require approximately 20 seconds of run time, however, those same starting points parallelized onto 100 processors would only require a run time of 0.2 seconds. Thus, the run time of the heuristic can be reduced to meet the efficiency needs of the system, subject only to the limitation of available processors.

In order to determine the appropriate number of starting points for the heuristic, we examine the MIO objective value of heuristic solutions for various number of starting points, recall that regardless of the number of starting points we will always choose the solution with the best objective function value. For ease of notation, let f_H be the MIO objective score of the heuristic solution and f_I be the MIO objective score of the ideal solution, which as a reminder is the solution in which the data association problem is exactly correct. We then compute the ratio of f_H/f_I, providing us a normalized measure for comparing the objective score across several scenarios. Figure 7-2 plots f_H/f_I (log scale) against σ.

In all cases, increasing the number of starting points improves the heuristic solution as compared to the ideal solution. For scenarios of four targets, the improvement is greatest, on the order of about five times improvement from 100 to 10,000 starting points. However, for scenarios with more targets, there is only slight improvement in objective value with increases in the number of starting points. This suggests that in order to improve MIO objective value, the number of necessary starting points increases with P, likely due to the exponential increase in combinatorial solutions as

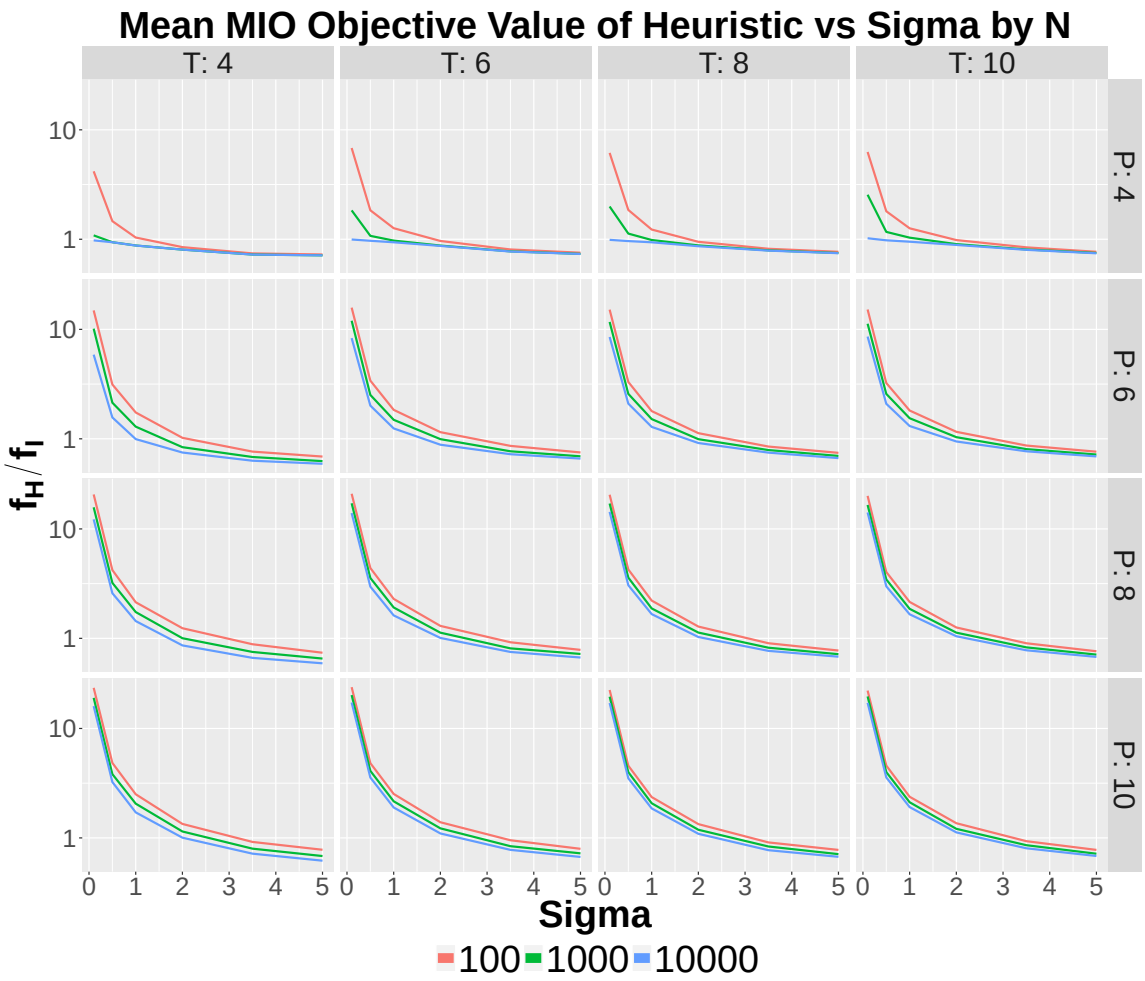

Figure 7-2: Quality of heuristic solution as compared to the ideal solution's MIO objective value summarized by number of starting points.

the number of targets increases. In any event, we have seen that there is not a significant difference in heuristic performance for the range of N values that we explored. Therefore, for simplification as we move forward in our analysis, we will restrict our discussions of the heuristic to $N = 1,000$.

It is also interesting to note that in some instances the ratio f_H/f_I falls below the value 1, indicating that the heuristic outperforms the ideal solution, as measured by the objective function. This occurs for higher values of σ, for example, when the value of σ is larger than 2 regardless of the value of P and T. To explain this phenomenon, we must recall that the ideal solution is simply ideal in the sphere of data association, while the MIO objective function serves to provide a measure of solution quality in

regards to both data association and trajectory estimation. This seems to suggest that it may be necessary to tradeoff correct data associations in order to improve the trajectory estimation in cases where there is more detection noise.

With these points in mind we continue our analysis to evaluate solution quality for both the data association problem and trajectory estimation for both the heuristic and MIO solutions.

7.1.3 Data Association

We shift our focus to analyze the performance of both the basic heuristic and the basic MIO model in regards to the data association problem. Figure 7-3 plots the mean accuracy of both solutions against ρ, our measure of difficulty for data association. The heuristic shown on the plot was initialized with 1,000 starting points and the MIO model was provided this solution as a warm start. Note that the results for the MIO model do not include the solution at $3T$ seconds, since it exhibited little to no improvement over the MIO solution at $2T$ seconds. The ideal solution, which always achieves an accuracy of 1.0, has also been excluded for the sake of clarity.

Figure 7-3 shows that the quality of the associations found by both the heuristic and the MIO are indeed highly correlated with the scenario complexity parameter ρ, and the number of correct association in both increase as ρ reaches 1. Where for 4 and 6 targets the association is extremely close or exactly the ideal association, when ρ is equal to 1. In contrast the random association is unaffected by the value of ρ and it accuracy is always lower than the heuristic accuracy.

In general, it seems that running the MIO for T or less seconds is optimal, since longer running times do not improve the accuracy attained. Specifically, for scenarios of four targets, the best accuracy is reached after 1 second, while for scenarios of eight and ten targets, the best accuracy is reached after T seconds.

In scenarios of all sizes, the heuristic improves over the random solution, suggesting that the heuristic finds good quality solutions. In fact, for scenarios of four targets, the heuristic performs equally as well as the MIO model across all numbers of scans. While for scenarios with more targets, the heuristic does not perform as well as the

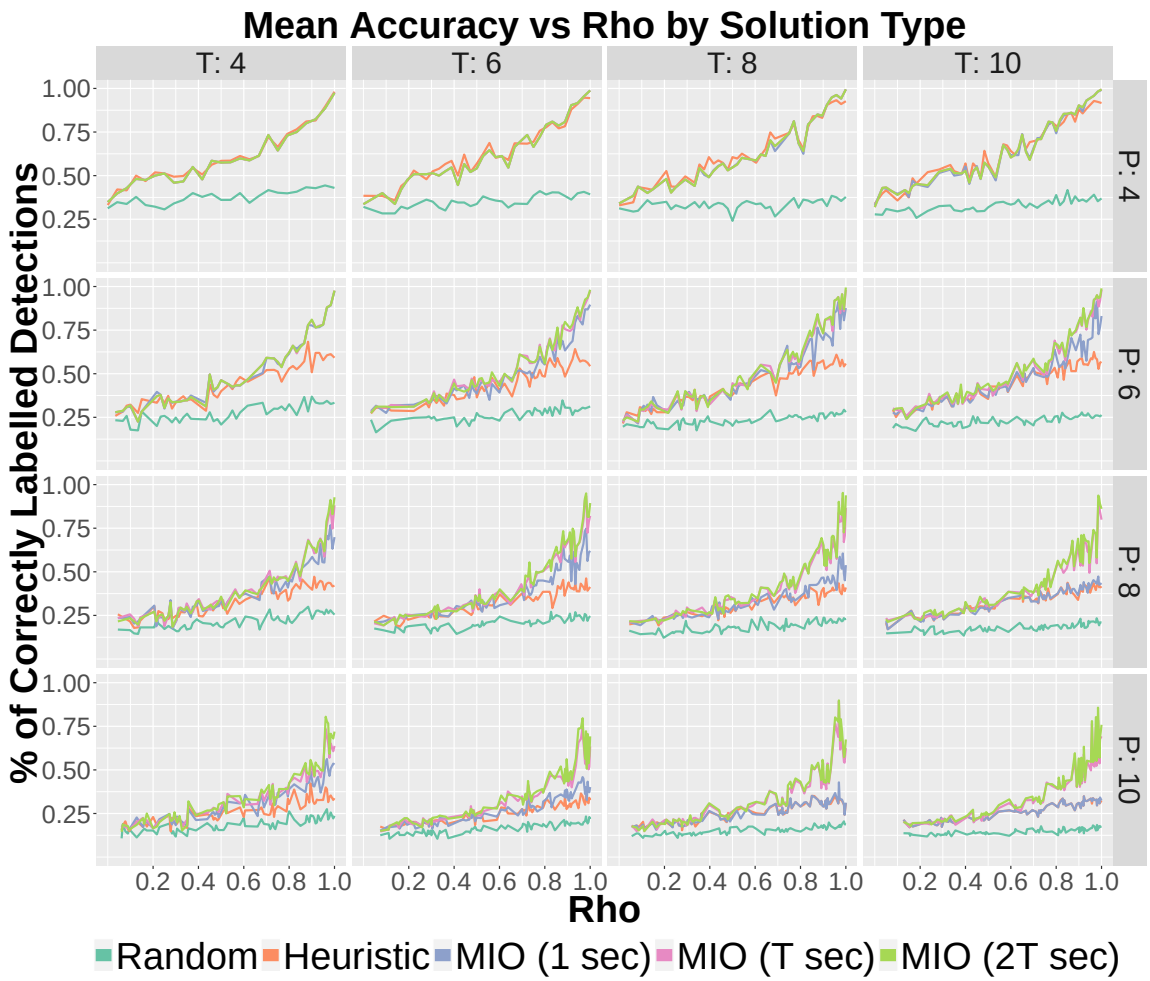

Figure 7-3: Accuracy of MIO compared against the heuristic and a randomized solution.

MIO, it still provides good solutions from which the MIO benefits. In particular, for scenarios with eight and ten targets the MIO after T seconds provides about a 30% improvement over the heuristic solution. Moreover, it seems that as the number of targets increases the accuracy deteriorates, due to the added combinatorial difficulty of the association problem.

Both algorithms appear to scale well with increases in the number of scans. In fact, the accuracy of MIO solutions appear to improve as the number of scans increases. For example, in the case of ten targets, the accuracy of the MIO after T (or $2T$) seconds is higher for eight and ten scans than for four and six scans, especially for high values of ρ. This could suggest that the MIO benefits from additional scans,

likely a result of the additional information gained from increasing the number of detections. For a fixed number of targets, there appears to be no change in heuristic solution quality as the number of scans changes.

7.1.4 Trajectory Estimation

Next, we evaluate the performance of the basic heuristic and MIO through the lens of trajectory estimation. As discussed previously, we are interested in comparing δ, our proxy for ground track error, against σ, our measure of difficulty for trajectory estimation, in order to analyze performance of in the sphere of estimation. Figure 7-4 plots σ against δ for each of the previous solution types in addition to the ideal solution.

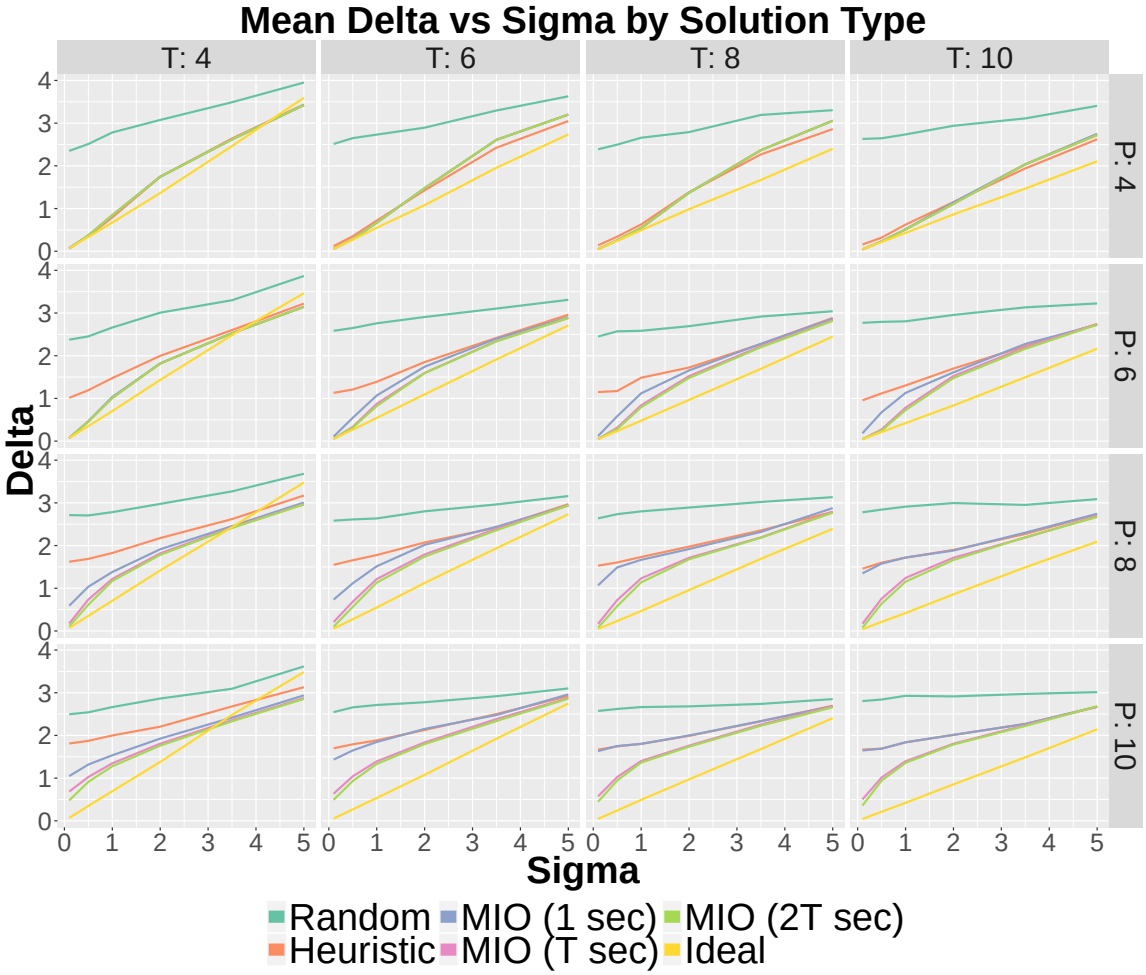

Figure 7-4: Trajectory estimation performance

Remember that lower values of delta correspond to trajectory estimations that are closer to that of the true ground track. We note that as σ approaches zero, both the heuristic and MIO trajectory estimates approach the trajectory estimates of the ideal solution, suggesting that both algorithms find very high quality trajectory estimates in these cases. Similar to the data association, it can be seen that the best estimates are reached after 1 second for scenarios with four targets and after T seconds for all other scenarios.

The heuristic and MIO outperform the ideal solution in scenarios with only four scans and high values of σ. However, the algorithms do not outperform the ideal solution for larger numbers of scans. Moreover, the ideal solution estimation changes linearly with sigma, but the slope of this linear dependence decreases as the number of scans increases. This suggests that as the number of scans approaches infinity, the estimated trajectories given by the ideal association will converge to the true ground tracks, even for large sigma values. This is also true for the MIO and heuristic solutions but to a lesser extent, which can be seen by the fact they move further away from the ideal solution when the number of scan increase, showing the tradeoff between added information and computational complexity.

We that the MIO and heuristic solutions estimation error have a bounded distance from the ideal one, and is always outperforms the random solution even for large values of σ. Which implies that though we do not necessarily get the optimal solution, the guiding objective prevents us from finding solution which are very poor. In particular, the gap between the ideal estimation error and the heuristic/MIO estimation error remains relatively constant across all values of P for a fixed number of scans.

7.1.5 Summary of Results

We have shown that in the case of no detection ambiguity

- The heuristic is highly scalable, and we can finds good quality solutions in fractions of a second using parallelization.

- Using these solutions as a warm start, the MIO achieves high quality solutions

to both the data association and trajectory estimation measures after T or fewer seconds.

- The MIO is scalable with respect to increases in both the number of targets and scans. We have also identified that there exists a tradeoff between making correct data associations and improved trajectory estimation, particularly in cases of high signal to noise ratios.

- Increasing the number of scans, while adding computational complexity to the model, helps to obtain better solutions.

With these points in mind, we advance to discussing the case of scenarios with detection ambiguity.

7.2 Scenarios with Detection Ambiguity

Here we extend our discussion to analyze the performance of our methods on scenarios with detection ambiguity. We first summarize our experimental methods before discussing performance of both the robust heuristic and the robust MIO in the spheres of both the data association and trajectory estimation problems.

This experiment serves as an extension of the basic one, in order to test the performance of our algorithms under detection ambiguity. We use the same scenarios generated from the basic experiment, however due to the additional difficulty inherent with detection ambiguity, we limit the range of signal noise to $\sigma \in \{0.1, 0.5, 1.0, 2.0\}$, choosing to exclude the extreme cases of signal noise. In addition, we simulate both missed detections and false alarms. A detection is removed with probability, γ, and we consider $\gamma \in \{0.2, 0.15, 0.1, 0.05\}$. We do not allow empty scans. For each scan, we generate false alarms according to a poisson distribution with parameter, λ, and false alarm locations are randomly selected uniformly within the state space. We consider $\lambda \in \{0.1, 0.5, 0.1, 2.0\}$. The false alarms are then added to $\mathcal{X}_t$ and the detection order of $\mathcal{X}_t$ is randomly shuffled in the same manner as the first experiment.

Once the data has been generated, we follow the same sequence of steps as outlined for the basic experiment, running the heuristic first and then feeding the solution into the MIO as a warm start. Note that the heuristic is only initialized with 1,000 starting points, as determined from the results of the basic experiment. Once again, the optimization process was set to terminate after $2T$ seconds, with solutions collected at intervals of $\{1, T, 2T\}$ seconds. Prior to the running of this experiment, we performed a mini experiment and used the results to tune the penalties θ and ϕ. A summary of the exact penalties used along with an explanation of the insight behind them, can be found in Appendix A.

7.2.1 Robust Heuristic

Table 7.2 summarizes the minimum, mean, and maximum run times of the heuristic from the robust heuristic for a single starting point, arranged by the number of estimated targets (P_{est}) and number of scans (T). Times are shown in milliseconds.

As expected, the robust heuristic requires longer run times than the basic heuristic, due to the increase in combinatorial solutions to the assignment problem. Comparing Table 7.1 and Table 7.2 we see that robust run times for four estimated targets are roughly four times longer than the corresponding run times in the basic heuristic for a fixed number of scans. However, the magnitude of this effect appears to decay as the number of targets increases. For example, in the case of eight targets, the robust heuristic times are only about twice that of the basic heuristic. This suggests that the robust heuristic still scales well with increases in the number of targets. It is also important to note that the run time variance in the robust case is much wider than the one for the basic case. However, parallelization might somewhat relieve this issue enabling the actual runtime to e closer to the average.

As exhibited by the basic heuristic, the scalability with regard to P_{est} is better than with regard to T. For example, increasing from eight to ten scans for eight targets increases the computational cost by 140%, while increasing from eight to ten targets for eight scans increases the computational cost by 50%. Here, we must remind the reader, that for each scenario we actually have to check several P_{est} values, which

$P_{\text{estimated}}$	T	Robust Heuristic Run Times (in milliseconds) Min	Mean	Max
2	4	0.15	0.23	0.41
2	6	0.42	0.56	0.93
2	8	0.77	1.04	2.24
2	10	1.27	1.73	3.07
4	4	0.15	0.34	1.04
4	6	0.50	0.94	2.69
4	8	1.09	1.88	3.87
4	10	2.12	3.25	7.20
6	4	0.14	0.42	0.96
6	6	0.57	1.29	4.45
6	8	1.33	2.66	5.82
6	10	2.53	4.61	9.4
8	4	0.16	0.50	1.10
8	6	0.60	1.59	3.46
8	8	1.38	3.37	6.87
8	10	2.63	5.84	12.40
10	4	0.18	0.55	1.10
10	6	0.72	1.82	3.98
10	8	1.53	3.96	8.18
10	10	3.42	6.93	13.93
12	4	0.16	0.56	0.99
12	6	0.99	1.95	3.96
12	8	1.74	4.33	8.69
12	10	3.40	7.71	15.10

Table 7.2: Robust heuristic run times (in milliseconds) for a single starting point.

requires additional processing time and parallelization. In our simulations, we found that the range of possible P_{est} values was generally limited to six values.

7.2.2 Number of Targets

Perhaps the most important goal of an MTT algorithm in the case of detection ambiguity is to correctly estimate the number of targets. Hence, we begin our analysis of detection ambiguity by evaluating the difference between the true and estimated

number of targets. Explicitly, we define:

$$P_{\text{diff}} = P_{\text{true}} - P_{\text{est}}, \tag{7.1}$$

where P_{est} is the number of estimated targets and P_{true} is the number of true targets. Note that $P_{\text{difference}} = 0$ indicates that we have correctly estimated the number of targets. When $P_{\text{difference}} < 0$, we have overestimated the number of targets, whereas when $P_{\text{difference}} > 0$, we have underestimated the number of targets. Figure 7-5 plots the distribution of P_{diff} for scenarios with four targets and eight scans, and for comparison, Figure 7-6 plots the same result for scenarios of eight targets and eight time scans.

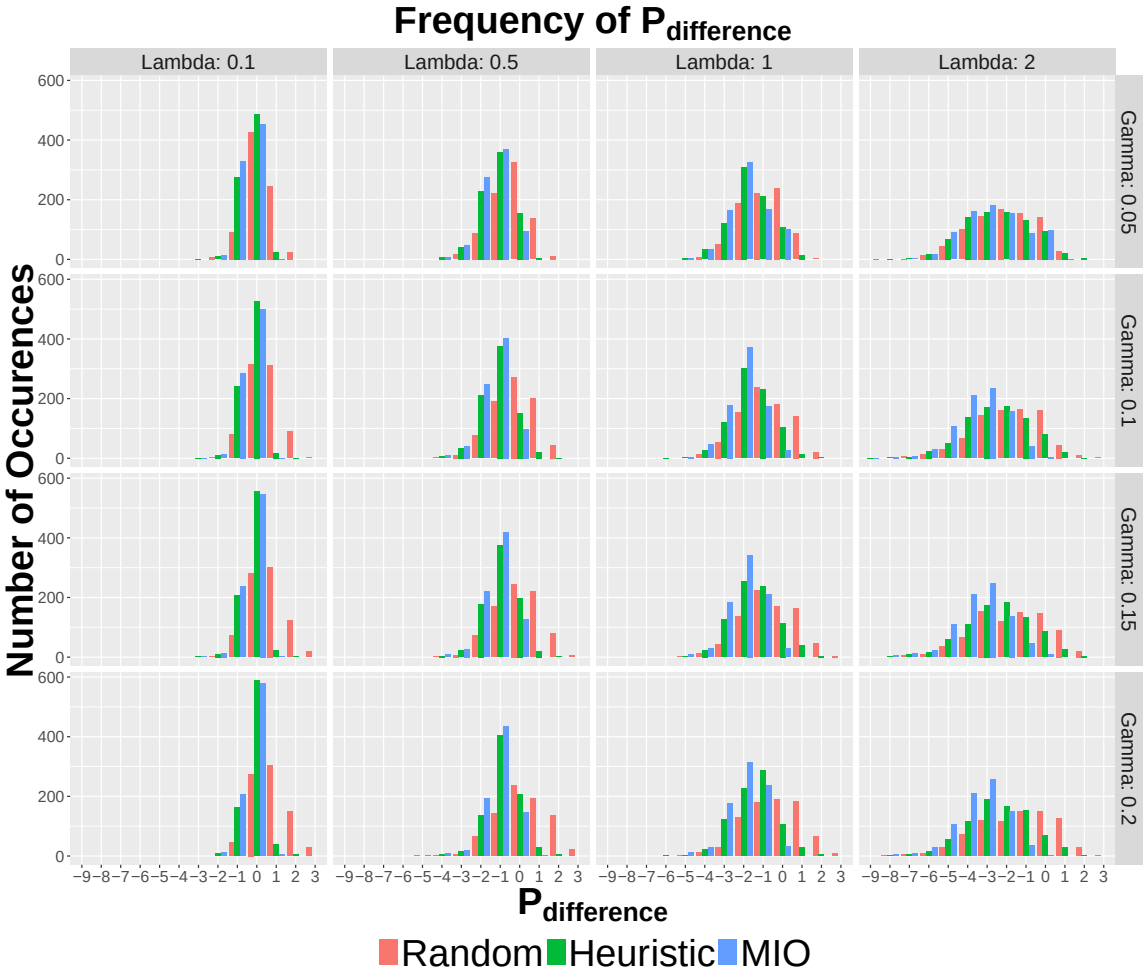

Figure 7-5: Distribution of the difference in true and estimated number of targets for scenarios with 4 targets and 8 scans, arranged by γ and λ.

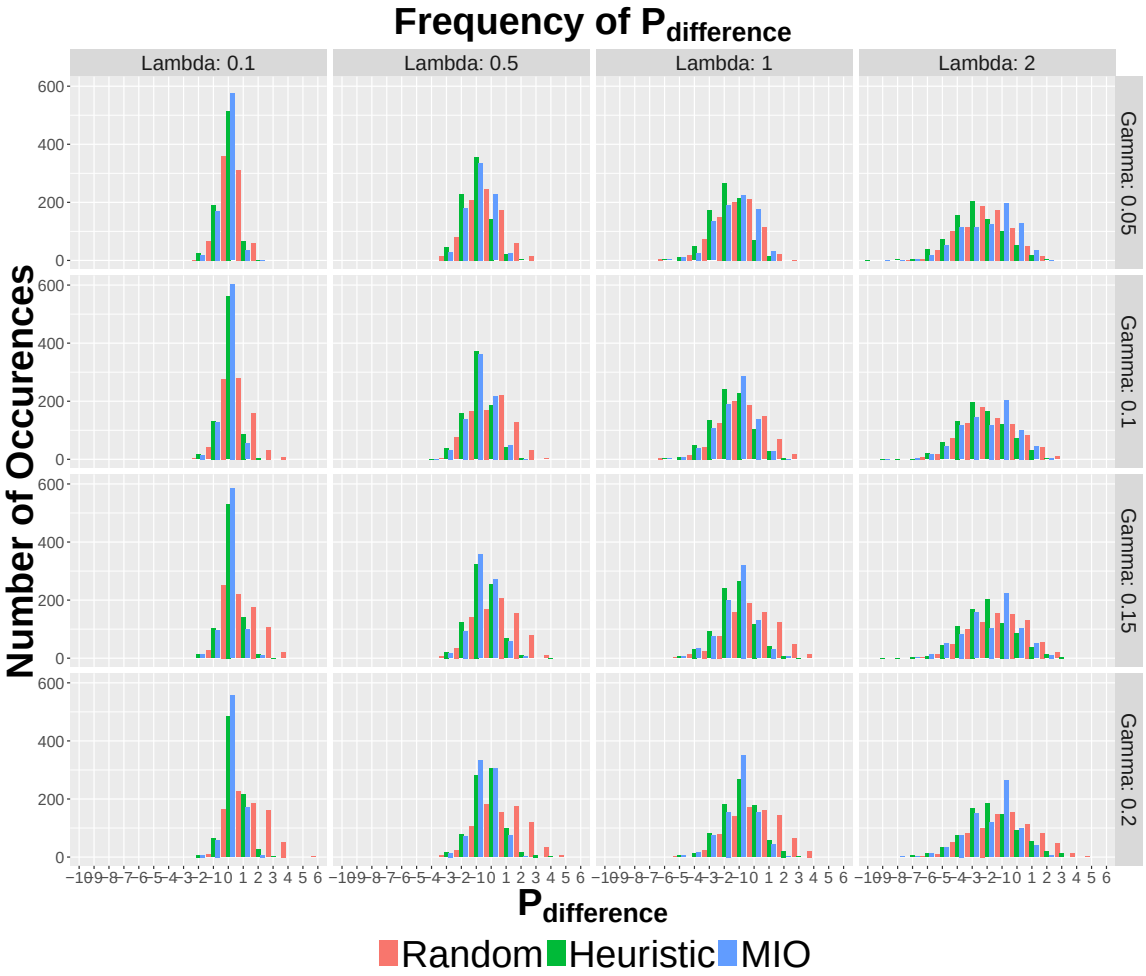

Figure 7-6: Distribution of the difference in true and estimated number of targets for scenarios with 8 targets and 8 scans, arranged by γ and λ.

We see that both the robust heuristic and the robust MIO estimate the number of targets correctly a very high proportion of the time in both scenario sizes when $\lambda = 0.1$. As λ increases, we see a tendency to overestimate the number of targets. This trend appears to be more prominent in Figure 7-5, meaning that the scenario with four targets has a higher tendency to overestimate for large values of λ than the scenario with eight targets. Further examination shows that both solutions have slightly fewer false alarms than the ideal solution for scenarios with eight targets, and this effect is slightly more exaggerated for the scenarios with four targets. This suggest that θ was set too high. Furthermore, the results show that the both algorithms identified too many missed detections, suggesting that the missed detection penalty

ϕ was too low. Altogether, this suggests that the algorithms likely took a preference to create additional trajectories and fill them with missed detections rather than classify detections as false alarms, ultimately resulting in a tendency to overestimate the number of targets.

7.2.3 Data Association

Knowing that we tend to overestimate the number of targets with the given penalties, we move on in our analysis to measuring the accuracy of our robust approaches. Figures 7-7 and 7-8 plot accuracy against the difficulty metric, ρ, for scenarios of four and eight targets, respectively. Both scenarios have eight scans and both figures have been arranged by γ and λ.

Similar to the performance of the basic heuristic, we again see that the robust heuristic improves greatly over that of a random solution. In fact, for scenarios with four targets, the robust heuristic performs closely to the MIO after one second. We see that the MIO after 1 second offers improvement over the heuristic, but running the MIO for any longer than 1 second does not appear achieve significant improvements. This story changes when we look at Figure 7-8 for the scenario with eight targets. Here we see that running the MIO for 1 second offers little to no improvement over the heuristic. Although the heuristic still improves over the random solution, it does not perform as well as for the scenario with four targets. That being said, running the MIO for T seconds offers significant improvement. Again we see that running the MIO for 2T seconds offers little to no further improvement.

Compared to the scenarios with no detection ambiguity, the solution quality appears to remain robust in the event of detection ambiguity, even for large scenarios. For example, we saw from the eight target and eight scan element of Figure 7-3 that accuracy topped out around 90% for scenarios with ρ close to 1. Now, we see from Figure 7-8 that when $\lambda = 0.1$ and $\gamma = 0.05$, we achieve accuracy levels within 5% for the MIO after T seconds. This is a key result because in the case of detection ambiguity we have not assumed a known number of targets.

Furthermore, the robust approaches appear more sensitive to higher false detection

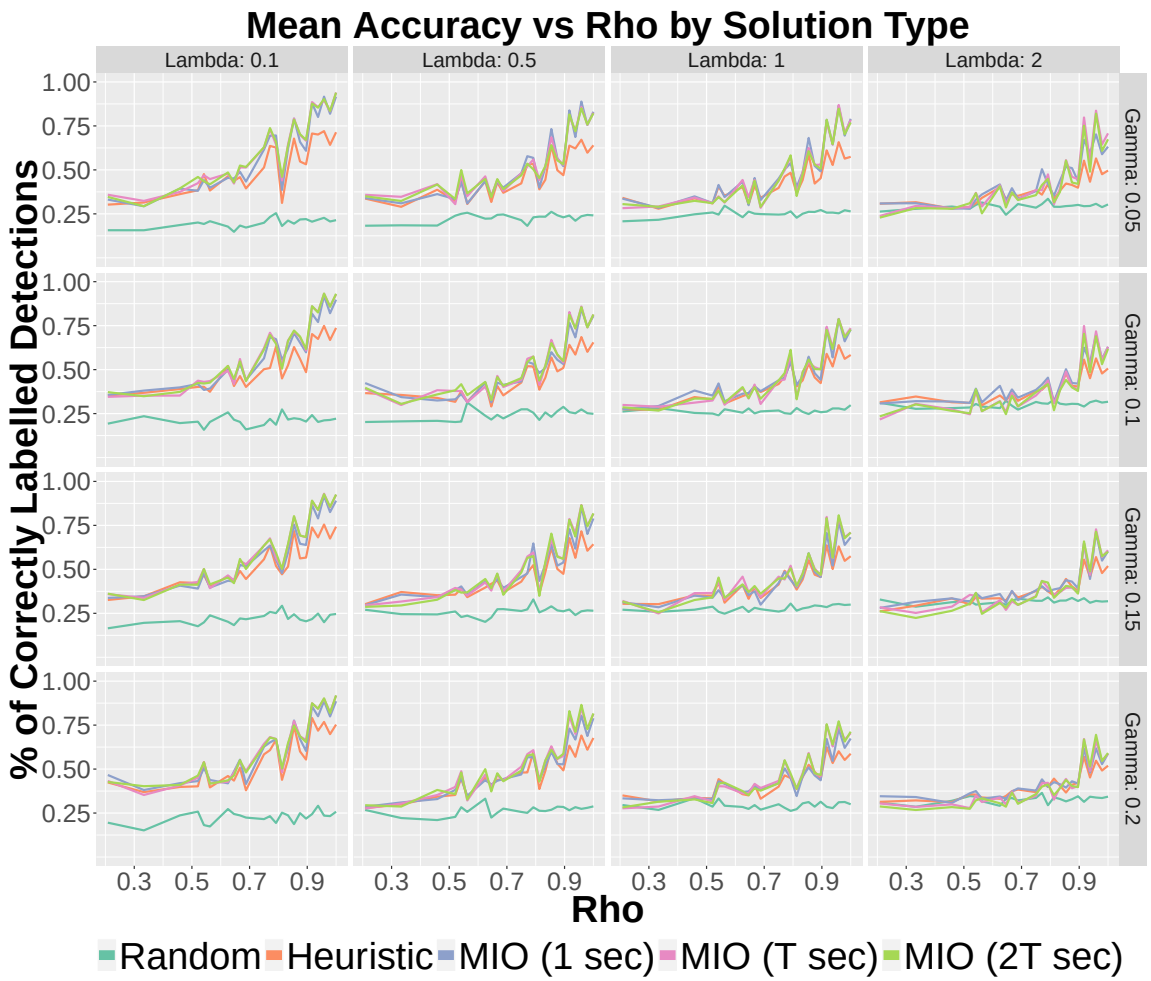

Figure 7-7: Accuracy of robust heuristic and MIO as compared to random solutions for scenarios of 4 targets and 8 scans, arranged by γ and λ.

rates than higher missed detection probabilities. The MIO after T seconds achieves accuracy levels of almost 65% when increasing the false alarm rate λ to 2.0 while fixing γ at 0.05. In contrast, the same solution achieves accuracy levels above 75% when increasing the missed detection probability to $\lambda = 0.2$, fixing λ at 0.1. Thus, we conclude that our approaches are more robust to changes in the missed detection probability than changes in the false alarm rate.

7.2.4 Trajectory Estimation

We conclude our analysis of the robust approaches with a discussion on their performance in the sphere of the trajectory estimation problem. Figures 7-9 and 7-10

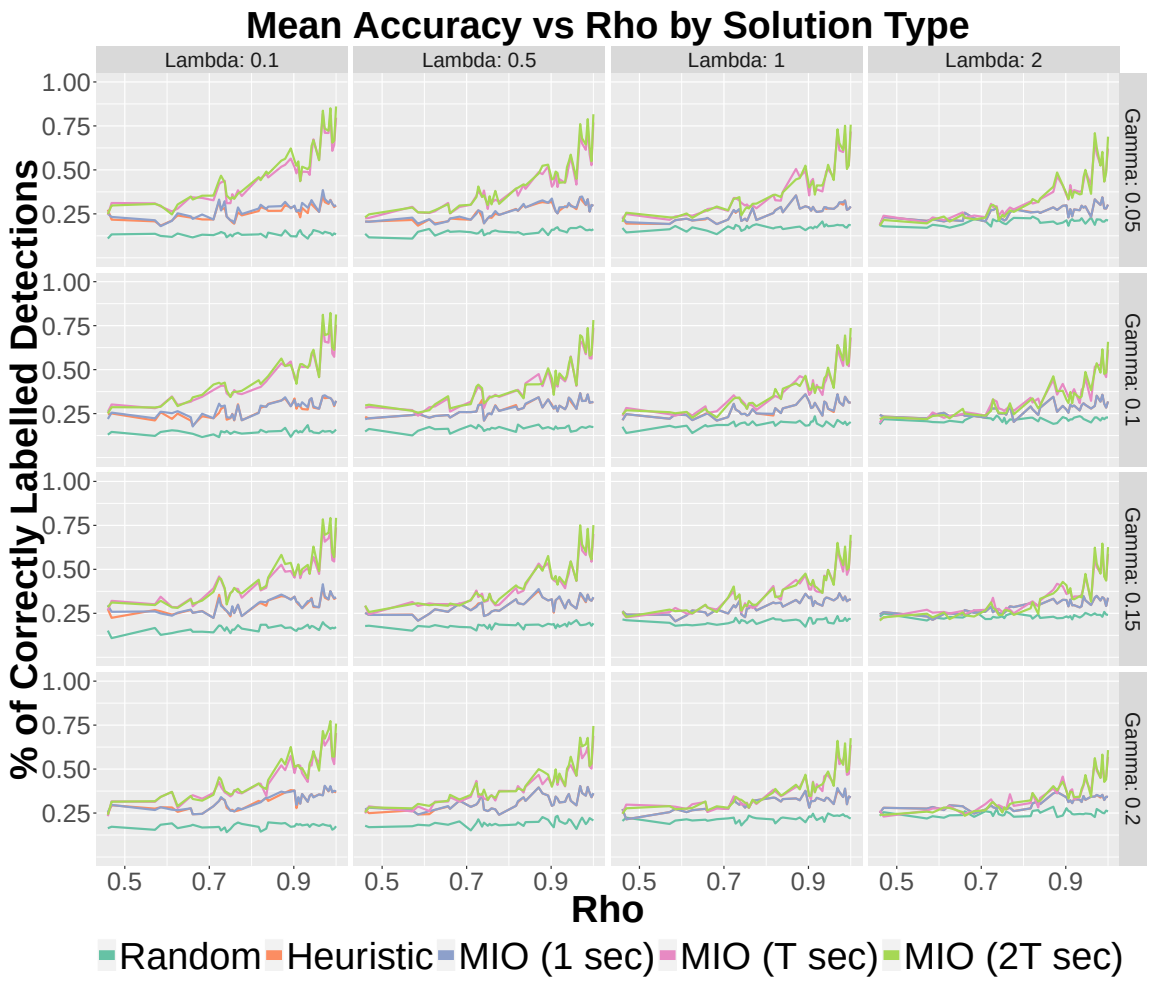

Figure 7-8: Accuracy of robust heuristic and MIO as compared to random solutions for scenarios of 8 targets and 8 scans, arranged by γ and λ.

plot the δ performance metric against the difficulty metric, σ, for scenarios of four and eight targets, respectively. Again, both scenarios have eight scans and both figures have been arranged by γ and λ. Note that the range on σ has been reduced from $[0.1, 5.0]$ in the previous section for scenarios without detection ambiguity to $[0.1, 2.0]$ for this section, where the scenarios now have detection ambiguity.

Again, we measure against the basic approaches by comparing Figure 7-9 with the 4 target and 8 scan element of Figure 7-4. We see that the robust approaches do not drastically reduce in performance for the easiest robust scenario of $\gamma = 0.05$ and $\lambda = 0.1$. However, the gap in performance between the ideal solution and the solutions of the robust algorithms grows wider with increases in σ, something that

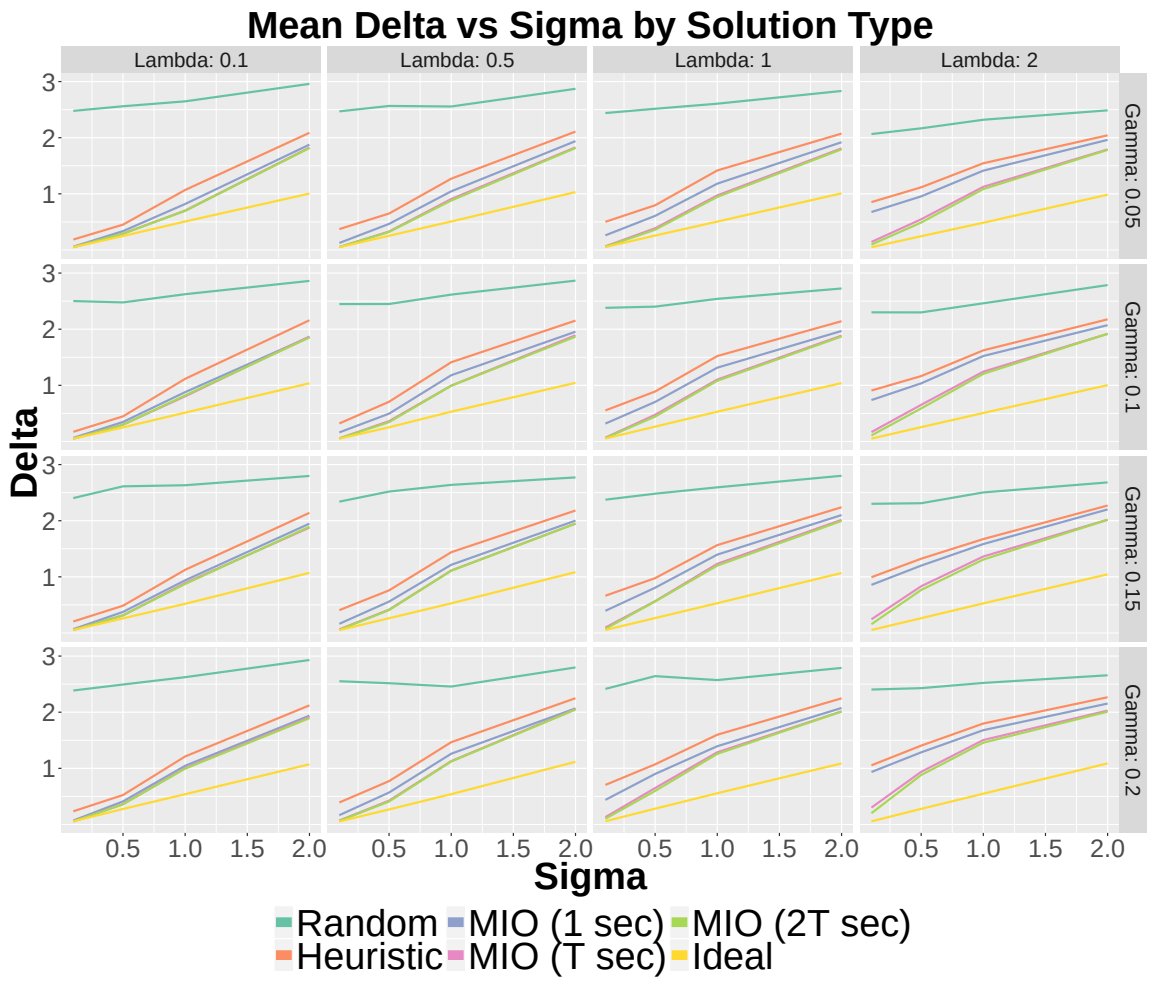

Figure 7-9: δ of robust heuristic and MIO as compared to random solutions for scenarios of 4 targets and 8 scans.

is expected but was not as sizable in the basic experiment. Therefore, the robust approaches may be less robust to increases in σ in scenarios with detection ambiguity. However, it also appears that these methods are more robust to increases in the false alarm rate λ when it comes to trajectory estimation than they were when it came to data association, especially in the larger scenario shown in Figure 7-10. We conclude that our robust methods are fairly robust to both increases in the false alarm rate and decreases in trajectory estimation under detection ambiguity, but increasing the signal noise degrades the performance of our methods more so in scenarios with detection ambiguity than in scenarios without detection ambiguity.

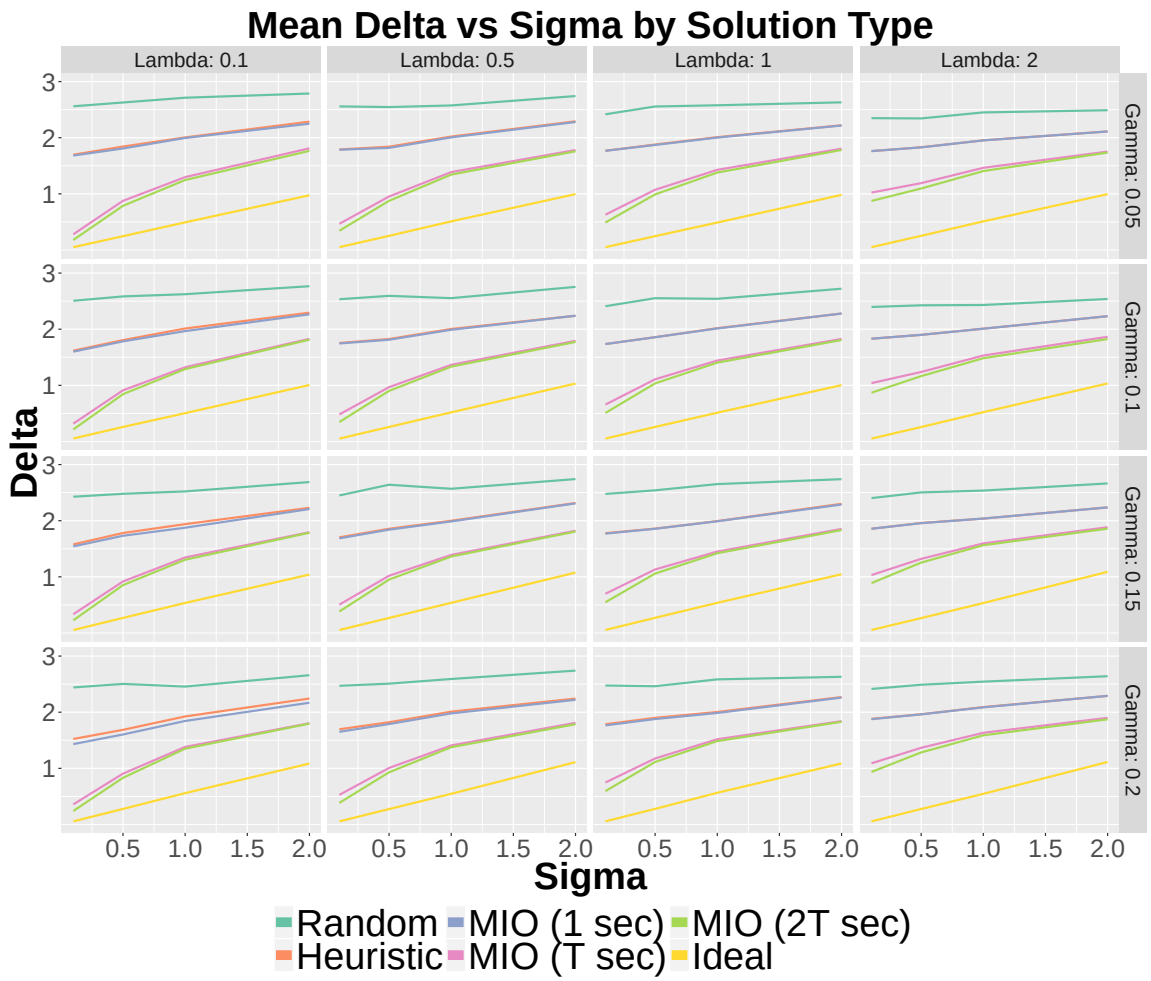

Figure 7-10: δ of robust heuristic and MIO as compared to random solutions for scenarios of 8 targets and 8 scans.

7.2.5 Detection Ambiguity Summary

We have shown that in the case of detection ambiguity:

- The case of detection ambiguity provides two additional challenges: the assignment problem is more complex for a fixed number of targets, and an added problem of deciding the correct number of target. We saw that tuning the parameters we can find the correct number of targets in most cases, even for a relatively large scenario, and given that number the detection ambiguity decreases performance by 5%-10%, for the case of low false alarm rate and any detection probability, with respect to the same scenarios with no detection ambiguity.

- The parameters are more easily tunable for various missed detection probability than for false alarm rates. This may be caused by the fact that missed detection, similarly to estimation errors, is dependent on the number of targets, while the number false alarm only depends on the false alarm rate.

Chapter 8

Summary and Future Work

In summary, we presented multi-target tracking approaches which jointly solve the problems of data association and trajectory estimation via global optimization methods using a single objective function. Toward this goal, we proposed the use of a randomized local search heuristic as a warm start for a mixed integer optimization model, and we did so for scenarios with and without detection ambiguity. We accomplish this without the need of a trajectory bank nor the prior computation of trajectory hypotheses. We showed that the heuristic finds good quality feasible solutions very quickly. We also show that the MIO offers improvement over the heuristic and this improvement is found in real time for applications considered by this paper.

The proposed methods show potential for use in an online algorithm with a sliding window, suggesting one possible area of future work. Implementing the heuristic as sliding window algorithm would very likely mitigate this scaling effect in regards to T. Rather than solve all scans in a single batch at once, a sliding window algorithm solves a subset of scans, or a smaller window, and advances through all scans sequentially. As the window progresses forward through the scans,“soft" decisions are made meaning that the heuristic would begin with the decisions from the previous solution. As scans pass beyond the horizon and out of the sliding window, the decisions become fixed and we refer to them as “hard" decisions. This process continues until all scans have been processed. The run times of a sliding window variant of the heuristic would not exhibit the curse of dimensionality in T since the number of scans remains constant.

Additionally, the heuristic is likely to produce higher quality solutions as a result of these “soft" decisions of previous steps since it is starting from a solution which is likely to be better than a completely random solution.

Additionally, future research could explore the use of more complex penalty functions. One possible idea would be the use of piecewise linear functions which would require further expansion of our formulations. Other areas of future work could seek to relax some of the scenario based assumptions made in Assumption 1, in particular, extensions to non-linear trajectories or the birth/death of targets.

Appendix A

Detection Ambiguity Penalty Values

Here we provide recommendations for the tuning of penalty parameters θ and ϕ. We begin with an explanation grounded in logic. It can be shown that as the false alarm rate λ increases, the number of expected false alarms also increases. Therefore, it stands to reason that as a general rule of thumb the false alarm penalty θ should decrease as λ increases. Similarly, the number of expected missed detections increases as the missed detection probability γ increases, and so too the missed detection penalty should decrease. Then it follows logically that the missed detection penalty ϕ should increase as γ decreases. Furthermore, it is convenient to reason that the value of both of these penalties should somehow be tied to the value of σ. This is due to the fact that as the noise increases, expect a higher standard deviation for the detections. Thus, when assigning false alarms and missed detections, then error will naturally be higher in the objective function. To combat this effect, we should consider increasing both penalties as σ increases. Through examination we found these logical concepts to generally hold true across a variety of scenario sizes and difficulties. Using this insight as well as the results of a mini experiment, we tuned our penalties for the robust experiment. The false alarm penalties used are shown in Figure A.1 and the missed detection penalties are shown in Figure A.2.

	σ			
λ	0.1	0.5	1.0	2.0
0.1	1.7	2.6	3.1	3.5
0.5	1.1	1.9	2.3	2.5
1.0	0.9	1.2	1.6	1.8
2.0	0.5	0.9	0.9	1.0

Table A.1: False alarm penalties (θ) as a function of λ and σ.

		σ			
λ	γ	0.1	0.5	1	2
0.10	0.05	0.20	0.50	0.80	0.70
0.10	0.10	0.10	0.30	0.50	0.50
0.10	0.15	0.10	0.20	0.40	0.40
0.10	0.20	0.10	0.10	0.30	0.40
0.50	0.05	0.20	0.50	0.80	0.80
0.50	0.10	0.20	0.30	0.50	0.60
0.50	0.15	0.20	0.25	0.40	0.40
0.50	0.20	0.10	0.20	0.30	0.40
1.00	0.05	0.30	0.70	0.80	0.80
1.00	0.10	0.20	0.40	0.50	0.60
1.00	0.15	0.20	0.25	0.40	0.40
1.00	0.20	0.10	0.20	0.30	0.40
2.00	0.05	0.30	0.70	0.90	1.00
2.00	0.10	0.20	0.50	0.60	0.60
2.00	0.15	0.20	0.25	0.40	0.50
2.00	0.20	0.10	0.20	0.30	0.40

Table A.2: Missed detection penalties (ϕ) as a function of λ, γ, and σ.

Appendix B

Robust MIO With Number of Targets as a Decision Variable

For completeness we take the time to present an alternative approach to solving the MTT problem with detection ambiguity. This MIO model is an extension to (5.5) that directly determines the number of targets via optimization by incorporating additional variables and constraints into the framework of the formulation.

B.1 Decision Variables

Rather than assuming a fixed number of targets for the model, we now allow this decision to be made by the model itself. Toward this goal, we introduce a new binary decision variable w_j to indicate whether or not trajectory j corresponds to an existing target:

$$w_j = \begin{cases} 1, & \text{if trajectory } j \text{ exists,} \\ 0, & \text{otherwise.} \end{cases}$$

Of important note, any detections assigned to a non-existing target would be considered to be a false alarm.

B.2 Objective Function

We can utilize the same objective function from (5.1), except for slight adjustments needed to account for the possibility that some trajectories may not exist. Explicitly, the number of possible trajectories is now N_1 so the objective should be adjusted to sum over j the full range of j from 1 to N_1:

$$\underset{\psi_{jt}}{\text{minimize:}} \quad \sum_{j=1}^{N_1} \sum_{t=1}^{T} \psi_{jt} + \theta \cdot TF + \phi \cdot TM$$

B.3 Constraints

In the same fashion, most constraints remain similar to their original counterparts in (5.5), with the except that the summations must be adjusted appropriately. For example, we adjust (5.2) and (5.4) as follows:

$$\sum_{j=1}^{N_1} y_{itj} + F_{it} = 1 \qquad \forall i, t,$$

$$\sum_{j=1}^{N_1} \sum_{t=1}^{T} M_{jt} = TM.$$

By the same accord the RHS of (5.3) no longer equals 1 because some trajectories may not exist. Therefore we say that all *existing* trajectories must either be assigned a detection or a missed detection, which implies the following constraint:

$$\sum_{i=1}^{n_t} y_{itj} + M_{jt} = w_j \qquad \forall j, t. \tag{B.1}$$

Moreover, we restrict α_j and β_j to be zero if trajectory j does not exist. This ensures only existing trajectories are penalized in the objective function.

$$|\alpha_j| + |\beta_j| \leq M_0 w_j \qquad \forall j.$$

This model is symmetric between various w_j variables. Such symmetry in models

is an undesirable quality because multiple optimal solutions may exist, resulting in a less efficient formulation. We can actually reduce the symmetry that is inherent to this formulation. Since $N_0 \leq P \leq N_1$, we can set $w_j = 1$ for all $j = 1, \ldots, N_0$, which leaves us with only $N_1 - N_0$ additional binary variables. Adding the constraints

$$w_{N_0+1} \geq ... \geq w_{N_1},$$

further reduces the number of equivalent solution and increases the efficient resolvability of the model.

B.4 Full Formulation

Incorporating these additional variables and constraints, we arrive at the following complete alternative formulation.

$$
\begin{aligned}
\underset{\psi_{jt}}{\text{minimize:}} \quad & \sum_{j=1}^{N_1}\sum_{t=1}^{T}\psi_{jt} + \theta\cdot TF + \phi\cdot TM \\
\text{subject to:} \quad & \sum_{j=1}^{N_1} y_{itj} + F_{it} = 1 \qquad \forall i,t \\
& \sum_{i=1}^{n_t} y_{itj} + M_{jt} = 1 \qquad \forall j = 1,...,N_0,t \\
& \sum_{i=1}^{n_t} y_{itj} + M_{jt} = w_j \qquad \forall j = N_0,...,N_1,t \\
& \sum_{i=1}^{n_t}\sum_{t=1}^{T} F_{it} = TF \\
& \sum_{j=1}^{N_1}\sum_{t=1}^{T} M_{jt} = TM \\
& w_{N_0+1} \geq ... \geq w_{N_1} \\
& |\alpha_j| + |\beta_j| \leq M_0 w_j \qquad \forall j \\
& x_{it}y_{itj} + M_t(1-y_{itj}) \geq z_{jt} \qquad \forall i,t,j \\
& x_{it}y_{itj} - M_t(1-y_{itj}) \leq z_{jt} \qquad \forall i,t,j \\
& z_{jt} - \alpha_j - \beta_j t \leq \psi_{jt} \qquad \forall j,t \\
& -(z_{jt} - \alpha_j - \beta_j t) \leq \psi_{jt} \qquad \forall j,t \\
& y_{itj} \in \{0,1\} \quad \forall i,t,j \\
& \alpha_j \in \mathbb{R}^n, \quad \beta_j \in \mathbb{R}^n, \quad w_j \in \mathbb{R}^n \quad \forall j \\
& z_{jt} \in \mathbb{R}^n, \quad \forall j,t.
\end{aligned}
$$

B.5 Extension of Robust Heuristic

In order to initialize the above MIO we need to explore the entire state space which include specifying the number of targets. Such a heuritic will be identical to that proposed in Chapter 5, except the number of target used will also be randomly selected during the initialization process. As in all previous cases, the warm start will be chosen among the different heuristic solutions based on the objective function value.

THIS PAGE INTENTIONALLY LEFT BLANK

Appendix C

Trajectory Assignment Pairing

In order to analyze the performance of a multi-target tracking algorithm, we must first find the best matching of the true trajectories of the scenario and the estimated trajectories of the algorithm solution. Put differently, we wish to find a set of assignment pairings which match true and estimated trajectories. Here we present a linear optimization model which solves for the globally optimal assignment pairings of true and estimated trajectories. In addition, we extend the assignment problem to scenarios with detection ambiguity, where the algorithm may overestimate or underestimate the number of targets. When this situation arises, a few minor adjustments are necessary to find the optimal assignment pairing.

The goal of this assignment problem is to optimally assign pairs of true trajectories i to estimated trajectories j if there exists such a pairing to be made. Only a single set of decision variables are needed to determine if the true trajectory i should be assigned to the estimated trajectory j or not.

$$y_{ij} = \begin{cases} 1, & \text{if true trajectory } i \text{ is assigned} \\ & \text{to estimated trajectory } j, \\ 0, & \text{otherwise.} \end{cases}$$

Remember that we denote the true position of trajectory i at scan t with $\bar{x}_{it}$ and the estimated position of trajectory j at scan t with $\hat{x}_{jt}$. Then the cost c_{ij} of assigning

true trajectory i to estimated trajectory j is the norm distance between these two trajectories as measured at each scan.

$$c_{ij} = \sum_{t=1}^{T} \|\bar{x}_{it} - \hat{x}_{jt}\|$$

If we denote the true number of targets as P_{true} and the estimated number of targets as P_{est} then the objective of the integer optimization model would be:

$$\underset{y_{ij}}{\text{minimize:}} \quad \sum_{i=1}^{P_{\text{true}}} \sum_{j=1}^{P_{\text{est}}} c_{ij} y_{ij}$$

When the number of true targets is equal to the number of estimated targets ($P_{\text{true}} = P_{\text{est}} = P$), we simply require two equality constraints to ensure that each true trajectory i is assigned to exactly one estimated trajectory j and vice versa.

$$\sum_{i=1}^{P} y_{ij} = 1 \qquad \forall j = 1, ..., P \tag{C.1}$$

$$\sum_{j=1}^{P} y_{ij} = 1 \qquad \forall i = 1, ..., P \tag{C.2}$$

However, when the number of estimated trajectories differs from the number of true trajectories, these constraints must be modified slightly. In the case where the number of true targets exceeds the estimated number of targets ($P_{\text{true}} \geq P_{\text{est}}$), we restrict each true trajectory i to the assignment of *at most* one estimated trajectory j, and Equation C.1 is modified to:

$$\sum_{i=1}^{P_{\text{true}}} y_{ij} \leq 1 \qquad \forall j = 1, ..., P_{\text{est}}.$$

On the contrary, when the number of estimated targets exceeds the true number of targets ($P_{\text{true}} \leq P_{\text{est}}$), then we restrict each estimated trajectory j to the assignment

of *at most* one true trajectory i, and Equation C.2 is modified to:

$$\sum_{j=1}^{P_{\text{est}}} y_{ij} \leq 1 \qquad \forall i = 1, ..., P_{\text{true}}.$$

In summary, the generalized integer optimization assignment model is presented below.

$$\begin{aligned}
\underset{y_{ij}}{\text{minimize:}} \quad & \sum_{i=1}^{P_{\text{true}}} \sum_{j=1}^{P_{\text{est}}} c_{ij} y_{ij} \\
\text{subject to:} \quad & \sum_{i=1}^{P_{\text{true}}} y_{ij} = 1 \qquad \forall j = 1, ..., P_{\text{est}} \\
& \sum_{j=1}^{P_{\text{est}}} y_{ij} = 1 \qquad \forall i = 1, ..., P_{\text{true}} \\
& y_{ij} \in \{0, 1\} \quad \forall i = 1, ..., P_{\text{true}}, j = 1, ..., P_{\text{est}}
\end{aligned}$$

This model is vital to scoring the performance of an MTT algorithm's solution because it ensures the globally optimal pairing of true and estimated trajectories.

THIS PAGE INTENTIONALLY LEFT BLANK

Bibliography

[1] A. Andriyenko, K. Schindler, and S. Roth. Discrete-continuous optimization for multi-target tracking. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1926–1933, June 2012.

[2] Anton Andriyenko and Konrad Schindler. Multi-target tracking by continuous energy minimization. In *Computer Vision and Pattern Recognition (CVPR)*, 2011.

[3] Y. Bar-Shalom and X.R. Li. *Multitarget-multisensor Tracking: Principles and Techniques*. Yaakov Bar-Shalom, 1995.

[4] Yaakov Bar-Shalom and Xiao-Rong Li. *Estimation with Applications to Tracking and Navigation*. John Wiley & Sons, Inc., New York, NY, USA, 2001.

[5] B Bertsimas and J. Dunn. Optimal trees. 2015. submitted for publication.

[6] Robert E. Bixby. Mixed-integer programming: It works better than you may think. `http://www.ferc.gov/CalendarFiles/20100609110044-Bixby,%20Gurobi%20Optimization.pdf`, 2010. Accessed 4 April 2016s.

[7] S.S. Blackman. *Multiple-target Tracking with Radar Applications*. Radar Library. Artech House, 1986.

[8] S.S. Blackman. Multiple hypothesis tracking for multiple target tracking. *Aerospace and Electronic Systems Magazine, IEEE*, 19(1):5–18, Jan 2004.

[9] Nadya Bliss, Robert Bond, Jeremy Kepner, Hahn Kim, and Albert Reuther. Interactive grid computing at lincoln laboratory. *MIT Lincoln Laboratory Journal*, 16(1), 2006.

[10] Craig Carthel and Stefano Coraluppi. Multi-hypothesis sonar tracking. In *Fusion 2004: Seventh International Conference on Information Fusion*, 2004.

[11] D. Castanon. Efficient algorithms for finding the k best paths through a trellis. *IEEE Transactions on Aerospace and Electronic Systems*, 26(2):405–410, Mar 1990.

[12] Inc. Gurobi Optimization. Gurobi 6.5 performance benchmarks. `http://www.gurobi.com/pdfs/benchmarks.pdf`, 2015. Accessed 4 April 2016s.

[13] Inc. Gurobi Optimization. Gurobi optimizer reference manual, 2015.

[14] C. Heij, P. de Boer, P.H. Franses, T. Kloek, H.K. van Dijk, and A.E.U. Rotterdam. *Econometric Methods with Applications in Business and Economics*. OUP Oxford, 2004.

[15] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[16] Andrea Lodi. *50 Years of Integer Programming 1958-2008.From the Early Years to the State-of-the-Art*. Springer Berlin Heidelberg, 2010.

[17] Miles Lubin and Iain Dunning. Computing in operations research using julia. *INFORMS Journal on Computing*, 27(2):238–248, 2015.

[18] C.L. Morefield. Application of 0-1 integer programming to multitarget tracking problems. *Automatic Control, IEEE Transactions on*, 22(3):302–312, Jun 1977.

[19] George Nemhauser. Integer programming: Global impact. `https://smartech.gatech.edu/bitstream/handle/1853/49829/presentation.pdf`, 2013. Accessed 4 April 2016.

[20] R. Perry, A. Vaddiraju, and K. Buckley. Multitarget list viterbi tracking algorithm. In *Signals, Systems amp; Computers, 1998. Conference Record of the Thirty-Second Asilomar Conference on*, volume 1, pages 436–440 vol.1, Nov 1998.

[21] Aubrey P. Poore and Nenad Rijavec. A lagrangian relaxation algorithm for multidimensional assignment problems arising from multitarget tracking. *SIAM Journal on Optimization*, 3(3):544–563, 1993.

[22] G.W. Pulford. Taxonomy of multiple target tracking methods. *Radar, Sonar and Navigation, IEE Proceedings* -, 152(5):291–304, October 2005.

[23] D.B. Reid. An algorithm for tracking multiple targets. *Automatic Control, IEEE Transactions on*, 24(6):843–854, Dec 1979.

[24] B. Ristic, B. N. Vo, D. Clark, and B. T. Vo. A metric for performance evaluation of multi-target tracking algorithms. *IEEE Transactions on Signal Processing*, 59(7):3452–3457, July 2011.

[25] N. Jakovcevic Stor, I. Slapnicar, and J. L. Barlow. Accurate eigenvalue decomposition of real symmetric arrowhead matrices and applications. *Linear Algebra and Its Applications*, 464:62–89, 2015.

[26] Top500 Supercomputer Sites. Performance development. `http://www.top500.org/statistics/perfdevel/`, 2015. Accessed 4 April 2016.

[27] J. K. Wolf, A. M. Viterbi, and G. S. Dixon. Finding the best set of k paths through a trellis with application to multitarget tracking. *IEEE Transactions on Aerospace and Electronic Systems*, 25(2):287–296, Mar 1989.