MULTI-TARGET TRACKING AND IDENTITY MANAGEMENT USING MULTIPLE MOBILE SENSORS

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Chiyu Zhang

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

May 2020

Purdue University

West Lafayette, Indiana

TABLE OF CONTENTS

Page

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

ABSTRACT

Zhang, Chiyu. PhD, Purdue University, May 2020. Multi-target Tracking and Identity Management Using Multiple Mobile Sensors. Major Professor: Inseok Hwang.

Due to their rapid technological advancement, mobile sensors such as unmanned aerial vehicles (UAVs) are seeing growing application in the area of multi-target tracking and identity management (MTIM). For efficient and sustainable performance of a MTIM system with mobile sensors, proper algorithms are needed to both effectively estimate the states/identities of targets from sensing data and optimally guide the mobile sensors based on the target estimates. One major challenge in MTIM is that a target may be temporarily lost due to line-of-sight breaks or corrupted sensing data in cluttered environments. It is desired that these targets are kept tracking and identification, especially when they reappear after the temporary loss of detection. Another challenging task in MTIM is to correctly track and identify targets during track coalescence, where multiple targets get close to each other and could be hardly distinguishable. In addition, while the number of targets in the sensors' surveillance region is usually unknown and time-varying in practice, many existing MTIM algorithms assume their number of targets to be known and constant, thus those algorithms could not be directly applied to real scenarios.

In this research, a set of solutions is developed to address three particular issues in MTIM that involves the above challenges: 1) using a single mobile sensor with a limited sensing range to track multiple targets, where the targets may occasionally lose detection; 2) using a network of mobile sensors to actively seek and identify targets to improve the accuracy of multi-target identity management; and 3) tracking and managing the identities of an unknown and time-varying number of targets in clutter.

1. INTRODUCTION

1.1 Background and Motivations

Tracking multiple moving targets is an important application of multi-sensor scheduling in both military and civil contexts, where sensors mainly accomplish two tasks: state estimation, i.e., the sensors autonomously estimate and predict the states of the targets; and identity management, i.e., the sensors differentiate the identities of the targets [1–3]. For example, consider a scenario where hostile targets move together with decoys in an adversarial environment. In this scenario, sensors, particularly mobile sensors such as unmanned aerial vehicles (UAVs) and cameras/telescopes, need not only to estimate the target states, but also to identify and track/follow the hostile targets.

Multi-target state estimation and tracking has been actively studied in the past few decades [4–6], however, there are still a few practical issues worth further attention but have not been seriously considered: one issue is that the targets may occasionally be lost due to the line of sight (LOS) broken by buildings, trees, and other occlusions, or broken by the targets moving out of the sensor's detection range; another one is that under circumstances where the number of targets is large but the number of sensors is limited, one sensor needs to be able to autonomously follow and keep track of one or more target(s). One major challenge in multi-target state estimation comes from the computational cost, especially complex situations such as losing targets are involved [7, 8]. For mobile sensors such as UAVs, it is desirable to develop a low-cost state estimation algorithm due to the limited computational power [9]; moreover, the estimation algorithm should be able to provide sufficiently accurate position estimates of the temporarily-lost targets so that their tracks are not completely lost. Another challenge regards to the problem of tracking multiple targets with a single mobile

sensor, where a proper guidance objective function must be constructed for the sensor to make decisions in a reasonable and efficient manner, e.g., the decision process is based on minimizing the overall tracking uncertainty; specifically, the objective function should introduce a tracking performance metric that can be evaluated and optimized quickly.

Multi-target identity management, on the other hand, is an area with few studies. However, identity management plays a significant role in a target tracking system; e.g., an air traffic management system needs to identify all the aircraft to safely manage the traffic; and a missile defense system needs to identify the actual missiles from decoys, etc. The target identification problem may not be a major concern when all the targets keep broadcasting their identity information, which is an extremely ideal case. In practice, even in a friendly environment where the targets communicate with the sensors, the target identity information can be unavailable due to communication losses; moreover, in a hostile situation, the targets do not communicate with the sensors at all. Then, it falls upon the sensors to actively identify the targets. In practice, we may receive direct target identity information from sensors. For example, a high resolution camera can identify a target by observing its color, shape, etc. This target identity information can be effectively and efficiently incorporated for accurate target identity management [10, 11]. Unfortunately, there is no study in how sensors should be scheduled to actively and optimally identify the targets. The major challenge lies within the optimization problem formulation of sensor scheduling. It is necessary that the optimal solutions computed make realistic senses. For example, seeking to identify an uncertain target can be considered as a reasonable solution, while trying to identify a target that is already sufficiently certain cannot. To achieve this, we need a well-posed objective function to describe our knowledge of the target identities. If no feasible optimal solution exists, a baseline solution needs to be provided to prevent the algorithm from terminating abruptly. In addition, when mobile sensors are considered, the time cost of controlling the sensors poses another challenge. For instance, if a UAV equipped with a narrow-ranged camera is used to identify targets,

then it takes time for the UAV to fly close to the target to identify it once the target is selected. This time cost must be incorporated to the optimization problem in a predictive manner. Moreover, an additional challenge rises since the prediction error is almost inevitable during the predictive sensor scheduling process, which can degrade the tracking performance. For example, consider using a UAV to pursue and identify targets, it is likely that the target to be identified is not within the UAV's own sensor observation range as predicted when the UAV follows the optimal control sequence, making the optimal solution no longer valid. Then, it is necessary to compensate the prediction error during the sensor control phase so that the optimal solution from the optimal sensor scheduling phase keeps its validity.

Another major limit of the current multi-target identity management is the lack of capability to handle an unknown and time-varying number of targets. In practice, the number of targets is unknown and time-varying to the tracking systems, as targets may move in or out in the system (e.g., aircraft takeoff/landing), or lose/regain detection, etc. Despite the research progress in tracking and estimating an unknown and time-varying number of targets [12, 13], the current identity management algorithms are not capable of handling this situation for two reasons: first, the state-of-art multi-target state trackers [13] do not explicitly calculate the target-measurement association, therefore they do not provide necessary information of identity management and cannot be used by the identity management algorithms; and second, the current mathematical framework of the identity management is incomplete in that it is only able to process a fixed number of targets.

Motivated by the limits and incompleteness of the current multi-target tracking and identity management (MTIM) studies, we aim to develop an integrated multi-sensor algorithm to address the aforementioned issues and challenges.

1.2 Works and Contributions

In this research, we develop a set of algorithms to solve three particular issues in tracking multiple moving targets and managing their identities.

Firstly, we consider the problem of tracking multiple targets with temporary loss of detection due to line-of-sight breaks and limited sensing range. For the problem, we propose a new multi-target state tracking algorithm which is composed of an estimation algorithm able to track lost targets efficiently and a mobile sensor guidance algorithm for multi-target tracking. The estimation algorithm is based on the Gaussian sum approximation of multi-variate functions [14–17]. It is proved that multi-variate functions defined over compact subsets of a real vector space can be approximated by the weighted sum of a set of Gaussian distributions [14], and such approximation uniformly converges as the number of Gaussian distributions increases. Led by this property, we construct a non-linear likelihood function when the 'no detection' event of a target happens; instead of formulating the likelihood function through the probabilistic approach as was used in some previous works [18], we define an event-based likelihood function which is then approximated by the Gaussian sum. Using a linear target motion model with Gaussian noise, we show that in the 'no detection' case, the posterior target probability density function (PDF) is also a Gaussian sum, and thus it is reasonable to consider only the first two moments of this distribution. This way, our algorithm keeps tracking and estimating the multiple targets regardless of whether they are lost; instead of numerically computing the posterior and prior target PDFs, only the mean and covariance matrix need to be calculated and stored, reducing the computation cost; in addition, the estimator smoothly switches to the standard Kalman filter once the target is back within the detection range of the sensor. Given the state estimation and prediction of a target, the tracking information entropy in the future can be viewed as a random variable with the probability of detection. We construct the guidance objective function based on the expectation of the information entropy.

Secondly, the problem of using multiple mobile sensors to actively seek and identify multiple targets is considered. We develop an algorithm to solve the problem with a decentralized sensor network composed of two types of sensors: primary sensor and secondary sensor. The primary sensor, which has a large detection range, tracks the states of the targets. In reality, the primary sensor can be a stationary long-range radar, a mobile high-altitude UAV equipped with a wide-angle camera, or a satellite, etc. The secondary sensor, on the other hand, is assumed to look at only one target at a time; however, it is able to identify the target by measuring its shape or other characteristics (consider a stationary telescope, a narrow-ranged camera mounted on a mobile UAV, etc.). The proposed algorithm involves one primary sensor and multiple secondary sensors. The primary sensor estimates the states of the targets, and sends the target state estimates to the secondary sensors; the secondary sensors utilize the incoming state estimates of the targets to keep track of the targets' identities. For each secondary sensor, we develop an optimal sensor scheduling algorithm which solves an optimization problem of identifying the target to minimize the cumulative expectation of the uncertainty of the targets' identities over a given time window. To prevent the algorithm from terminating abruptly when no feasible solution is found for the optimization problem, a baseline solution is also calculated and used whenever necessary. Eventually, the optimal sensor scheduling algorithm selects the optimal target to be identified according to not only the uncertainty about the target identities, but also the time cost to identify the target, as well as the corresponding control sequence of the secondary sensor. The proposed algorithm is a decentralized one since the secondary sensors do not need to synchronize with each other, and each secondary sensor solves its own optimization problem.

Thirdly, we study the problem of tracking and managing the identities of an unknown and time-varying number of targets. For this problem, we propose a generalized MTIM algorithm, called GMPHD-IM algorithm, which extends the capabilities of the state-of-art MTIM algorithm and is able to track and manage the identities of an unknown and time-varying number of targets. In order to manage the identities

of an unknown and time-varying number of targets, we introduce a new mathematical framework for multi-target identity management. The new framework can keep track of the identities of an unknown and time-varying number of targets, and update the target identities whenever there is available target identity information so that the uncertainty in all the target identities is reduced. In addition, we also develop a multi-target state estimator to estimate the states of the targets as well as to efficiently compute necessary information for multi-target identity management.

In summary, the main contribution of this research is the development of integrated multi-target tracking and identity management algorithms which can address the limitations in the state-of-art studies. The algorithm framework we propose is a general solution, i.e., the mathematical formulation does not consider the specific types of sensors or targets. In real applications where the types of sensors and targets are given, one can choose the corresponding models for the algorithm.

The rest of this dissertation is organized as follows. In Chapter 2, a multi-target state estimation and tracking algorithm is proposed and demonstrated by illustrative simulation results. In Chapter 3, a multi-target identity management algorithm for a fixed number of targets with mobile-sensors is proposed and demonstrated by simulation results. In Chapter 4, we propose a new identity management algorithm for an unknown and time-varying number of targets and demonstrate the algorithm with simulation results. The conclusions of the work is summarized in Chapter 5.

2. MULTI-TARGET STATE ESTIMATION AND TRACKING WITH TEMPORARY LOSS OF DETECTION

2.1 Background

Recursive Bayesian estimation (RBE) [19] forms the theoretical basis of target state estimation, where the posterior probability density function (PDF) of the target state is recursively constructed using available information such as measurement data, and is propagated according to the target motion model. Based on the RBE framework, different estimation algorithms, such as the Kalman filter (and its variants) and Monte Carlo methods, are developed [20–24]. The conventional RBE framework requires the target measurement input for each recursion step and does not account for the situation where the target is lost. To track a lost target, the estimation algorithm should keep running even without the target measurement. Dead-reckoning (keep propagating the target state until a new measurement is obtained) is simple and widely used in application and experiments [25,26]. This method relies mainly on the modeling of target motion, which can differ from the target's true movement; and it ignores the information conveyed by the event of 'no detection' that the probability of the target in the observable region can be excluded if no measurement is received from that area. Led by this idea, unified RBE search and tracking (S&T) techniques are proposed through the definition of a unified non-linear likelihood function and objective for both no-detection (search) and detection (tracking) cases [7, 8, 27–29]. All of these approaches construct their target motion PDF on a grid basis; however, they are either computationally complex or could provide a rough description of target motion. In addition, although the entire algorithms are non-linear, they usually yield a near-Gaussian PDF during the detection and tracking processes; those algorithms keep computing the PDF values cell by cell regardless of this near-Gaussian

characteristics which can be exploited to improve the efficiency. To overcome these limitations, we develop a new RBE algorithm with a new construction and approximation of likelihood function for the event of no-detection. Therefore, the proposed multiple maneuvering target tracking algorithm has a good tracking performance with great improvement of computation efficiency.

The majority of the research works on the multi-target tracking problem relies on the assumption that the number of available sensors is greater than or equal to the number of targets. In this case, the decision-making strategy of the sensors is then formulated as an area surveillance problem [30–32], where the objective function usually desires that the sensors cover a large surveillance area [33] or reduce the tracking error [34]. In situations where there are fewer sensors than targets, it is often considered that the sensors have a large sensor coverage area such that all the targets can be covered simultaneously all the time. The guidance objective function is then formed as a task assignment problem: to allocate regions to the sensors [35], or to find a target-covering sequence that can be accomplished in minimum time [36]. However, the circumstance that the number of available sensors is limited and the sensors cannot cover all the targets at the same time exists, and cannot be ignored. Despite one proposed strategy [37], where the sensor actively gives up tracking the targets that are unable to be accurately tracked, research has barely been done in this area. Our work is motivated by the practical significance and the lack of research efforts in the problem. The key to the problem is to formulate a proper guidance objective function that incorporates the tracking performance metrics of both the lost and detected targets. In the field of multi-target tracking, information entropy (as a representation of the uncertainty) is widely used to construct tracking objectives [38–42], where the objective function is usually formulated as to minimize the predicted information entropy based on the current observation of the targets. However, in the situation of one range-limited sensor vs. multiple targets, it is inevitable that some targets are not observed at some point. The possibility that some targets are not detected in the future needs to be considered when constructing the objective function. One way

to account for the possibility is to add a multiplier that represents the probability of detection [43]. Based on our proposed estimation algorithm which has the ability to track lost targets, we propose the following strategy to track multiple targets with one range-limited sensor: whenever it is unable to detect all the targets, the sensor tries to cover the targets alternately, i.e., leave some of the targets lost and cover the rest of the targets, and go back to the lost targets later.

The main contributions of this chapter lie in the consideration of a specific type of multi-target tracking problem that is practically significant yet not seriously/widely considered by the community: there are fewer sensing agents than targets, and the sensing agents cannot cover all the targets simultaneously; and in the development of computationally efficient multi-target state estimation and optimal guidance algorithms to solve the problem. Specifically, we develop a new RBE algorithm that is able to track lost targets more efficiently through the proposed approximation approach. While most existing RBE algorithms for target tracking assume that the targets are always detected, i.e., they do not consider the possibility of losing targets [21–24], the proposed RBE algorithm explicitly considers the event that a target is lost, which makes the target tracking problem challenging and does not allow to directly apply the existing algorithms. In this chapter, we formulate the lost-target event as a non-linear likelihood function. Since the evaluation of this non-linear likelihood function is complex and computationally demanding, we develop an approximation approach using Gaussian sum to improve the efficiency of the proposed RBE algorithm. In addition, by utilizing the output of the proposed RBE algorithm, we formulate an optimal guidance problem to minimize the uncertainty about the targets' positions; the optimal guidance problem enables a single UAV to effectively track multiple targets which may not be covered by a UAV at the same (i.e., some targets are within the UAV's sensing range but others are out of the sensing range), which has not been seriously studied before.

The rest of this chapter is organized as follows: We first formulate the problem and briefly introduce the RBE framework and the Gaussian sum approximation in

Section 2.2. The details of the proposed algorithm are presented in Section 2.3, including the proposed RBE estimation technique and the optimal guidance law. The proposed multiple maneuvering target tracking algorithm is demonstrated with illustrative numerical examples in Section 2.4.

2.2 Problem Formulation

This section formulates the target and the primary sensor models, and the RBE framework and the Gaussian sum approximation.

The j th target's motion is described by its state vector $x_j(t)$ at time t, whose discrete-time motion model is given by

$$x_j(t+1) = f_j(x_j(t), u_j(t), w_j(t)) \tag{2.1}$$

where $u_j(t)$ and $w_j(t)$ are the input and the system noise at time t, respectively. The target's motion in (2.1) is considered as a memoryless Markov process. In this chapter, we consider the targets as moving ground vehicles. Thus, for a single target, its state is a 4-dimensional vector describing the vehicle's ground position and speed of two orthogonal directions.

The states of the targets are measured by a primary sensor's onboard sensor with a circular detection area whose radius is denoted as $r^{(ps)}$. As long as a target is within the detection range, it will be detected and localized by the sensor. For the j th target, its measurement $z_j(t)$ at time t is given by:

$$z_j(t) = Hx_j(t) + v(t) \tag{2.2}$$

where $v(t)$ is white noise whose covariance is R. Figure 2.1 provides a visual description of the target tracking scenario considered.

At the tth time step, the discrete-time motion model of the primary sensor is given by

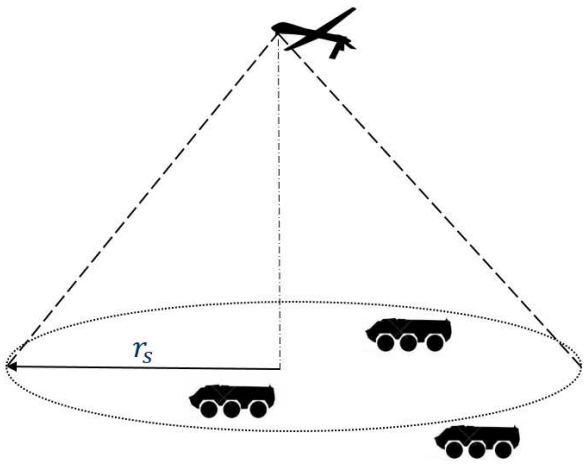


Fig. 2.1.: Multiple Maneuvering Target Tracking Scenario

$$x^{(ps)}(t+1) = f^{(ps)}(x^{(ps)}(t), u^{(ps)}(t)) \tag{2.3}$$

where $x^{(ps)}(t)$ and $u^{(ps)}(t)$ represent the primary sensor's state and control input at time t.

2.2.1 Recursive Bayesian Estimation

For simplicity, in this section we omit the subscript j of x_j that denotes the j th target. Let $d(t)$ be the random variable of observation indicating whether a target is detected by the sensor at time t, i.e. $d(t) \in \{$'detection', 'no detection'$\}$. Denote the observation history up to time t as $D(t) = \{d_1, d_2, \cdots, d(t)\}$. RBE then computes the target state PDF recursively by two stages: update and prediction.

- *Update.* The update stage computes the posterior PDF $p(x(t)|D(t))$ from the incoming observation $d(t)$, and the last predicted PDF $p(x(t)|D(t-1))$ which is given by the observation history $D(t-1)$:

$$p(x(t)|D(t)) = K\Lambda(x(t)|d(t))p(x(t)|D(t-1)) \tag{2.4}$$

where $\Lambda(x(t)|d(t)) = p(d(t)|x(t), D(t-1))$ represents the likelihood of $x(t)$ given $d(t)$, and K is the normalization factor.

- *Prediction.* The prediction stage derives the target state PDF of the next time step $p(x(t+1)|D(t))$ from the current estimated PDF $p(x(t)|D(t))$. Since the target motion is modeled as a memoryless Markov process, the prediction can be computed by the Chapman-Kolmogorov equation [44]:

$$p(x(t+1)|D(t)) = \int p(x(t+1)|x(t))p(x(t)|D(t))dx(t) \tag{2.5}$$

where $p(x(t+1)|x(t))$ is computed by propagating the target's current state $x(t)$ to the next state $x(t+1)$ with the target's motion model in (2.1).

2.2.2 Gaussian Sum Approximation

Consider multi-variate function $p(x)$ of vector x, which satisfies the following properties:

- $p(x)$ is Riemann integrable
- $\int_x p(x)dx < \infty$
- $\forall x,\ p(x) \geq 0$

It is proved that $p(x)$ can be approximated by a weighted sum of Gaussian PDFs [14]:

$$p(x) \approx \sum_{i=1}^{n} \alpha_i \mathcal{N}(x; \mu_i, P_i) \tag{2.6}$$

where $\mathcal{N}(x; \mu_i, P_i)$ is a Gaussian distribution whose mean and covariance are μ_i and P_i, respectively. μ_i is selected from the sample space of x, and once it is determined, the weight α_i can be calculated from $p(\mu_i)$.

2.3 Algorithm Development

In this section, we present in detail the proposed multi-target tracking algorithm, which is composed of two components: the multi-target state estimation and the optimal guidance law.

2.3.1 RBE Target State Estimation

In this section, we consider a linear target motion model with Gaussian noise. For simplicity, the subscript j that denote sthe j th target state is omitted. A likelihood function of the target state $\Lambda(x(t)|d(t) =$ 'no detection') is constructed, and the RBE equations are then derived by approximating $\Lambda(x(t)|d(t) =$ 'no detection') using the Gaussian sum. Specifically, it is shown that if a linear target motion model with Gaussian noise is applied, the proposed approach can be unified with Kalman filter. An implementation of the algorithm based on the target position grid is also presented in this section.

Likelihood Function in No Detection Case and Approximated Estimation

When a target is detected at time t, the likelihood function $\Lambda(x(t)|d(t) =$ 'detection') in (2.4) can be calculated using the sensor measurement $z(t)$. However, when the target is not detected and no measurement is received, the likelihood function $\Lambda(x(t)|d(t) =$ 'no detection') is not defined. Therefore, our primary task is to give a definition of $\Lambda(x(t)|d(t) =$ 'no detection'), i.e., the likelihood of a target's state given the event of 'no detection'. The definition of $\Lambda(x(t)|d(t) =$ 'no detection') is based on the definition of the 'detectable region'. The detectable region is the area within the primary sensor's detection range; and a measurement is generated only if the target is in this detectable region. Given the primary sensor's state as $x^{(ps)}(t)$, the detectable region is uniquely determined, and denote the detectable region as $\mathcal{X}_D(x^{(ps)}(t))$. The likelihood

of the target state when there is no detection is defined as 0 inside the detectable region and uniformly distributed outside the detectable region:

$$\Lambda(x(t)|d(t) = \text{'no detection'}) = \begin{cases} 0 & x(t) \in \mathcal{X}_D(x^{(ps)}(t)) \\ \eta & x(t) \in \mathcal{X} \cap \bar{\mathcal{X}}_D(x^{(ps)}(t)) \end{cases} \tag{2.7}$$

where $\mathcal{X}$ is the finite operational area and η is a positive constant number. When the 'no detection' event is given, it can be interpreted as the target being unlikely within the sensor's coverage, and the above definition in (2.7) is the mathematic expression of the interpretation. Then, according to equation (2.6), the likelihood function can be approximated by the Gaussian sum:

$$\Lambda(x(t)|d(t) = \text{'no detection'}) \approx \sum_{i=1}^{n} \alpha_i \mathcal{N}(x(t); \mu_i, P_i) \tag{2.8}$$

where μ_i and P_i denote the mean and covariance of the ith Gaussian PDF, respectively. Considering a linear motion model with Gaussian noise, we assume that the prior probability $p(x(t)|D(t-1))$ is a normal distribution whose mean and covariance are $x(t|t-1)$ and $P(t|t-1)$, respectively. By denoting $\mu = x(t|t-1)$ and $P = P(t|t-1)$, we have

$$p(x(t)|D(t-1)) = \mathcal{N}(x(t); \mu, P) \tag{2.9}$$

By (2.4), the update stage of RBE can be approximated as:

$$p(x(t)|D(t)) \approx K \sum_{i=1}^{n} \alpha_i \mathcal{N}(x(t); \mu_i, P_i) \mathcal{N}(x(t); \mu, P) \tag{2.10}$$

To calculate the state estimate and covariance, we introduce the following lemma which is a well-known conclusion in the Gaussian sum literature [45]:

Lemma 2.3.1 *Let* $x(t),\ \mu,\ \mu_i \in \mathbb{R}^q$, $K, \alpha_i \in \mathbb{R}$; *and let* $P,\ P_i \in \mathbb{R}^{q \times q}$ *be positive definite matrices. Then*

$$K\sum_{i=1}^{n}\alpha_i\mathcal{N}(x(t);\mu_i,P_i)\mathcal{N}(x(t);\mu,P)=K\sum_{i=1}^{n}\alpha_iK_i\mathcal{N}(x(t);\hat{\mu}_i,\hat{P}_i) \tag{2.11}$$

where

$$\begin{aligned}\hat{\mu}_i&=(P_i^{-1}+P^{-1})^{-1}(P_i^{-1}\mu_i+P^{-1}\mu)\\ \hat{P}_i&=(P_i^{-1}+P^{-1})^{-1}\\ K_i&\in\mathbb{R}\end{aligned} \tag{2.12}$$

Let $\beta_i = K\cdot\alpha_iK_i$ be a weight; then the two steps of the approximated RBE are expressed as follow:

$$\begin{aligned}p(x(t)|D(t))&=\sum_{i=1}^{n}\beta_i\mathcal{N}(x(t);\hat{\mu}_i,\hat{P}_i)\\ p(x(t+1)|D(t))&=\sum_{i=1}^{n}\beta_i\int p(x(t+1)|x(t))\mathcal{N}(x(t);\hat{\mu}_i,\hat{P}_i)dx(t)\end{aligned} \tag{2.13}$$

In particular, if a linear target motion model is applied, then the integral in (2.13) can be evaluated with Kalman filter. Consider the following linear target motion model:

$$x(t+1)=F_tx(t)+w(t) \tag{2.14}$$

where F_t is a system matrix and $w(t)$ is white noise whose covariance matrix is $Q(t)$. Then, the prediction step in (2.13) becomes:

$$p(x(t+1)|D(t))=\sum_{i=1}^{n}\beta_i\mathcal{N}(x(t+1);F_t\hat{\mu}_i,F_t\hat{P}_iF_t^T+Q(t)) \tag{2.15}$$

It is still computationally demanding to calculate the exact PDFs in (2.13) and (2.15); however, it is reasonable to consider only the means and covariance matrices during the iteration. This is because the first and second moments of a PDF are essentially projections to the polynomial basis of the inner-product space (denoted as $\mathcal{L}^{(2)}$) of square integrable functions defined over the following inner-product:

$$\langle f(x), g(x) \rangle = \int_{-\infty}^{\infty} f(x)g(x)dx \tag{2.16}$$

Therefore, the use of mean and covariance is in fact a second-order approximation of the PDF in the context of $\mathcal{L}^{(2)}$. In fact, considering the existing well-established Gaussian sum reduction methods based on different approximation criteria [46–48], our way of approximation essentially can be viewed as a lower-order solution to those algorithms.

Since the update and prediction PDFs are both Gaussian sums, their means and covariance matrices can be analytically calculated. Let $\hat{x}(t)$ and $\hat{P}(t)$ be the conditional mean and covariance of the posterior PDF $p(x(t)|D(t))$. Then we have:

$$\begin{aligned} \hat{x}(t) &\triangleq \int x(t)p(x(t)|D(t))dx(t) = \sum_{i=1}^{n} \beta_i \hat{\mu}_i \\ \hat{P}(t) &\triangleq E\left[(x(t) - \hat{x}(t))(x(t) - \hat{x}(t))^T\right] \\ &= \sum_{i=1}^{n} \beta_i \left[\hat{P}_i + (\hat{\mu}_i - \hat{x}(t))(\hat{\mu}_i - \hat{x}(t))^T\right] \end{aligned} \tag{2.17}$$

Given the linear target motion model in (2.14), the predicted mean $x_{t+1|k}$ and covariance $P(t+1|t)$ can be obtained from (2.17) using Kalman filter:

$$x_{t+1|k} = F_t \hat{x}(t), \;\; P(t+1|t) = F_t \hat{P}(t) F_t^T + Q(t) \tag{2.18}$$

It can be shown that the mean and covariance given in (2.18) are exactly the mean and covariance of the PDF in (2.15). By the approximation above, the proposed method can be unified with the standard Kalman filter, since regardless of whether a target is detected or lost, only the mean and covariance need to be calculated and stored.

Implementation Based on Position Grid

The main factor that influences the complexity of the proposed algorithm is the number of Gaussian PDFs we used, which grows exponentially with the dimension of

the target state space. Although the target state is modeled as a 4-dimensional vector, we are mainly interested in its position information when giving a tracking command to the UAV; in addition, when a target is lost, there will be no speed information. Therefore, it is sufficient to discretize and apply the Gaussian sum approximation to only the position space, and find alternative ways to update the speed estimate.

When a target is lost, we apply (2.17) to the target position space and let the position estimate at time t be $\hat{p}(t)$ and its covariance $\Sigma(\hat{p}(t), \hat{p}(t))$. Let Δt be the sampling time. To obtain the speed estimate $\hat{v}(t)$ at time t, we use the current and previous position estimate:

$$\hat{v}(t) = \frac{1}{\Delta t}(\hat{p}(t) - \hat{p}(t-1)) \tag{2.19}$$

Then, the covariance matrices $\Sigma(\hat{v}(t), \hat{v}(t))$ and $\Sigma(\hat{v}(t), \hat{p}(t))$ can be computed as:

$$\begin{aligned} & \Sigma(\hat{v}(t), \hat{v}(t)) = E[\hat{v}(t)\hat{v}(t)^T] - E[\hat{v}(t)E\hat{v}(t)^T] \\ = \; & \frac{1}{\Delta t^2}\big(E[(\hat{p}(t) - \hat{p}(t-1))(\hat{p}(t) - \hat{p}(t-1))^T] \\ & - E[\hat{p}(t) - \hat{p}(t-1)]E[(\hat{p}(t) - \hat{p}(t-1))^T]\big) \\ = \; & \frac{1}{\Delta t^2}\big(\Sigma(\hat{p}(t), \hat{p}(t)) + \Sigma(\hat{p}(t-1), \hat{p}(t-1)) \\ & - \Sigma(\hat{p}(t), \hat{p}(t-1)) - \Sigma(\hat{p}(t), \hat{p}(t-1))^T\big) \end{aligned} \tag{2.20}$$

$$\begin{aligned} & \Sigma(\hat{v}(t), \hat{p}(t)) = E[\hat{v}(t)\hat{p}(t)^T] - E[\hat{v}(t)]E[\hat{p}(t)^T] \\ = \; & \frac{1}{\Delta t}\big(E[(\hat{p}(t) - \hat{p}(t-1))\hat{p}(t)^T] - E[\hat{p}(t) - \hat{p}(t-1)]E[\hat{p}(t)^T]\big) \\ = \; & \frac{1}{\Delta t}\left(\Sigma(\hat{p}(t), \hat{p}(t)) - \Sigma(\hat{p}(t), \hat{p}(t-1))^T\right) \end{aligned} \tag{2.21}$$

The covariance matrices $\Sigma(\hat{p}(t), \hat{p}(t))$ and $\Sigma(\hat{p}(t-1), \hat{p}(t-1))$ are given directly from the position update. To obtain $\Sigma(\hat{p}(t), \hat{p}(t-1))$, note that from the previous section we have:

$$\begin{aligned}\hat{x}(t) &= \sum_{i=1}^{n} \beta_i \hat{\mu}_i = \sum_{i=1}^{n} \beta_i \hat{P}_i^{-1}(P_i^{-1}\mu_i + P_{t|t-1}^{-1} x_{t|t-1}) \\ &= \sum_{i=1}^{n} \beta_i \hat{P}_i^{-1}(P_i^{-1}\mu_i + P_{t|t-1}^{-1} F_{k-1}\hat{x}(t-1))\end{aligned} \tag{2.22}$$

Since μ_i is deterministic, the covariance of μ_i and $\hat{x}(t-1)$ are 0 and the covariance of $\hat{x}(t)$ and $\hat{x}(t-1)$ can be computed as:

$$\begin{aligned}\Sigma(\hat{x}(t), \hat{x}(t-1)) &= \Sigma(\sum_{i=1}^{n} \beta_i \hat{P}_i^{-1} P_{t|t-1}^{-1} F_{k-1}\hat{x}(t-1), \hat{x}(t-1)) \\ &= P_\Sigma \Sigma(\hat{x}(t-1), \hat{x}(t-1)) P_\Sigma^T, \\ \text{where } P_\Sigma &= \sum_{i=1}^{n} \beta_i \hat{P}_i^{-1} P_{t|t-1}^{-1} F_{k-1}\end{aligned} \tag{2.23}$$

where $\Sigma(\hat{x}(t-1), \hat{x}(t-1)) = \hat{P}_{t-1}$ is the estimate covariance at time $t-1$. $\Sigma(\hat{p}(t), \hat{p}(t-1))$ is then the position components of $\Sigma(\hat{x}(t), \hat{x}(t-1))$. The complete state estimation and covariance can be obtained by combining the position and speed components together.

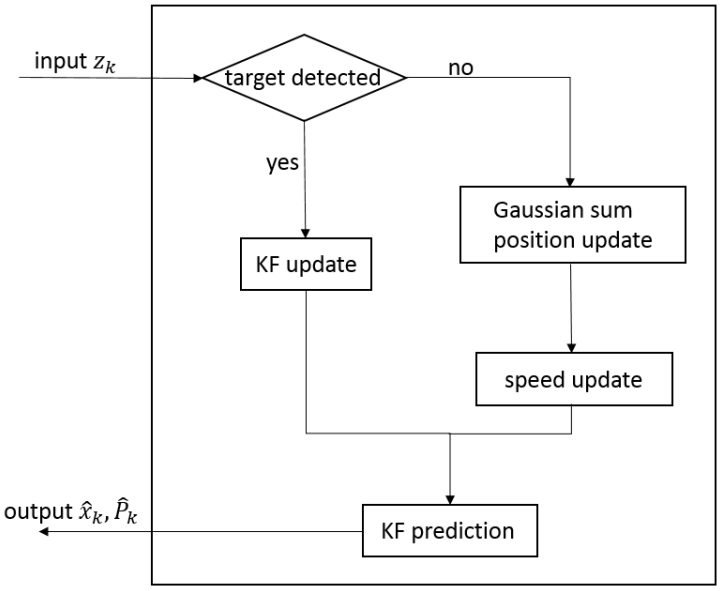


Fig. 2.2.: Overall Estimation Algorithm Structure

As a result, the overall structure of the proposed estimation algorithm is shown in Figure 2.2. For each sampling time step, if a target is detected, an estimate of its state is given by the standard Kalman filter; otherwise the position estimate is calculated by (2.17) first, and then the speed estimate is updated using the formulas in this section. In the position estimation of a lost target, the Gaussian PDFs are constructed on a grid basis, i.e. the 2-dimensional area is discretized to a grid map; and the means and covariance matrices of the Gaussian PDFs are given by the grid points and sizes, respectively. Therefore, the grid density implies the number of Gaussian PDFs used in the algorithm; a higher grid density will lead to more intensive computation, and vice versa. Whether a target is detected or lost, the mean and covariance of the target state estimation will always be provided and the standard Kalman filter is applied to the prediction.

2.3.2 Multi-target Tracking Guidance

In this section, we first formulate a guidance objective function for multi-target tracking by minimizing the expected information entropy. The objective function is then transformed by eigenvalue decomposition to improve the computational efficiency. The primary sensor guidance law is eventually formulated as an optimization problem of seeking the optimal command input subject to the sensor's kinematic constraints.

Information Entropy and Objective Function

Information entropy is essentially a metric of the uncertainty of a system: a higher entropy value means that we have less amount of information about the system [49]. In the context of target tracking, it is a matter of fact that a large amount of information is obtained if a target is detected, and is lost if a target is lost. Therefore, minimizing the predicted information entropy is, in some sense, equivalent to maximizing the number of detected targets in a predictive manner.

Let us consider the prediction covariance of the j th target $P_j(t+1|t)$. The predicted update covariance of the j th target is a Bernoulli variable:

$$P_j(t+1|t+1) = \begin{cases} P_j(t+1|t) - P_j(t+1|t)H^T(HP_j(t+1|t)H^T + R)^{-1}HP_j(t+1|t) & \text{target } j \text{ is detected} \\ P_j(t+1|t) & \text{otherwise} \end{cases} \tag{2.24}$$

where H and R are the sensor measurement and covariance matrices. The probability that the j th target is detected $\pi_{(D,j)}(x^{(ps)}(t+1))$ is a function of the sensor's state and can be calculated according to the prediction of the j th target given in:

$$\pi_{(D,j)}(x^{(ps)}(t+1)) = \iint\limits_{x_j \in \mathcal{X}_D(x^{(ps)}(t+1))} \mathcal{N}(x_j; x_j(t+1|t), P_j(t+1|t))dx_j \tag{2.25}$$

It should be noted that the integral (2.25) in fact represents the probability that the j-th target is within the primary sensor's detectable region, given the sensor's position at $x^{(ps)}(t+1)$. Then, the expected update covariance is given by:

$$\bar{P}_j(t+1|t+1) = P_j(t+1|t) - \pi_{(D,j)}(x^{(ps)}(t+1))P_j(t+1|t)H^T(HP_j(t+1|t)H^T+R)^{-1}HP_j(t+1|t) \tag{2.26}$$

The information entropy of a Gaussian distribution, regardless of the constants, is the logarithm of the determinant of the covariance matrix. Hence for the linear target motion models with Gaussian noise, the expected information entropy from the j th target can be written as $\log\left|\bar{P}_j(t+1|t+1)\right|$. As a result, the objective function is formulated as a weighted sum of the expected information entropy of all targets:

$$J = \sum_{j=1}^{N} W_j \log\left|\bar{P}_j(t+1|t+1)\right| \tag{2.27}$$

where N is the number of the targets. The purpose of using the weighted sum is to increase the priority of the lost targets. The longer a target is lost, the higher

priority it has. Let t_j^L be the cumulative time that the j th target has been lost: t_j^L starts increasing as the target is lost, and is reset to 0 when the target is re-detected. The weight term W_j of the j th target is constructed as a function of t_j^L with the following properties: (1) $W_j(0) = 0$; (2) W_j increases as t_j^L increases.

Objective Function Transformation

The evaluation of the objective function (2.27) is mainly the calculation of the following determinant:

$$\left|P_j(t+1|t)-\pi_{(D,j)}(x^{(ps)}(t+1))P_j(t+1|t)H^T(HP_j(t+1|t)H^T+R)^{-1}HP_j(t+1|t)\right| \quad (2.28)$$

The computation cost of calculating the determinant can be high especially when iterative optimization methods are applied, since they usually involve a large number of function evaluations [50]. In addition, the computation cost grows with the number of targets. Therefore, it is desirable that (2.28) can be transformed for faster calculation. For brevity, we consider two matrices P and $\tilde{P}$ and a scalar π in this section, where P is positive definite, $\tilde{P}$ is semi-positive definite, and π is non-negative. Then, (2.28) can be written as:

$$\left|P - \pi\tilde{P}\right| \quad (2.29)$$

To transform (2.29), we first introduce the following lemma:

Lemma 2.3.2 *Let $P \in \mathbb{R}^{q\times q}$ and $\tilde{P} \in \mathbb{R}^{q\times q}$ be two Hermite matrices. Let $\pi \in \mathbb{R}$. If both P and $\tilde{P}$ are invertible, then*

$$\left|P - \pi\tilde{P}\right| = \left|\Lambda_{\tilde{P}}\right| \cdot \prod_i (\lambda_i^* - \pi) \quad (2.30)$$

where $\Lambda_{\tilde{P}} \in \mathbb{D}^{q\times q}$ is a diagonal matrix and $\lambda_i^* \in \mathbb{R}$.

Proof We first do eigenvalue decomposition to $\tilde{P}$:

$$\tilde{P} = U\Lambda_{\tilde{P}}U^T \tag{2.31}$$

where U is unitary and $\Lambda_{\tilde{P}}$ is diagonal. Substituting $\tilde{P}$ with its eigenvalue decomposition, we have:

$$\begin{aligned} \left|P - \pi\tilde{P}\right| &= \left|P - \pi U\Lambda_{\tilde{P}}U^T\right| \\ &= \left|U^T PU\Lambda_{\tilde{P}}^{-1} - \pi I\right| \cdot \left|\Lambda_{\tilde{P}}\right| \end{aligned} \tag{2.32}$$

To evaluate the determinant $\left|U^T PU\Lambda_{\tilde{P}}^{-1} - \pi I\right|$, we consider its characteristic equation:

$$\left|(\lambda + \pi)I - U^T PU\Lambda_{\tilde{P}}^{-1}\right| = 0 \tag{2.33}$$

where λ are the eigenvalues of $U^T PU\Lambda_{\tilde{P}}^{-1} - \pi I$. By denoting the eigenvalues of $U^T PU\Lambda_{\tilde{P}}^{-1}$ as λ^*, we have $\lambda = \lambda^* - \pi$. Since the determinant of a matrix is the product of its eigenvalues, we have the following equation:

$$\left|P - \pi\tilde{P}\right| = \left|\Lambda_{\tilde{P}}\right| \cdot \prod_i (\lambda_i^* - \pi) \tag{2.34}$$

where λ_i^* is the i th eigenvalue of $U^T PU\Lambda_{\tilde{P}}^{-1}$. ■

Since in (2.29) P is positive definite and $\tilde{P}$ is semi-positive definite, they are intrinsically Hermite and the lemma can be applied. However, it should be noted that in Lemma 2.3.2 we assume $\tilde{P}$ to be invertible which may not be the case in reality. For the non-invertible cases, we add a small perturbation to $\tilde{P}$ before the eigenvalue decomposition:

$$\tilde{P} = \tilde{P} + \varepsilon I \tag{2.35}$$

where ε is a small positive scalar and I is the identity matrix. It should be noted that the small perturbation added to $\tilde{P}$ is in fact a tradeoff between stability and accuracy. Since $\tilde{P}$ is not guaranteed to be invertible, it is likely that the algorithm

terminates when the inverse of $\tilde{P}$ does not exist. To prevent the algorithm from this unexpected termination, a small perturbation is added to $\tilde{P}$ only when it is non-invertible. In addition, the perturbation is only effective in one single optimal guidance step, and does not affect the estimation performance. Then, using Lemma 2.3.2, the objective function can be converted to a form which can be efficiently evaluated:

$$J = \sum_{j=1}^{N} W_j \Big\{ \log \left| \Lambda_{\tilde{P},j} \right| + \sum_i \log \left(\lambda^*_{i,j} - \pi_{(D,j)}(x^{(ps)}(t+1)) \right) \Big\} \tag{2.36}$$

It should be noted that in the original objective (2.27), the matrix determinant is a function of the optimization variable $x^{(ps)}(t+1)$, and needs to be repeatedly calculated for different $x^{(ps)}(t+1)$ values within one optimization step. On the other hand, in the transformed objective function (2.36), the matrix operations can be done before the optimization iteration starts, since they are not related to $x^{(ps)}(t+1)$; during the optimization iteration we only need to evaluate the scalar function $\pi_{(D,j)}(x^{(ps)}(t+1))$ instead of the determinants. The efficiency improvement is significant since the optimization typically involves a large number of function evaluations. For example, consider a case that during one optimization step the objective function needs to be calculated for 100 different $x^{(ps)}(t+1)$ values. Then (2.27) requires 100 determinant calculations, while (2.36) only needs 2 eigenvalue decompositions plus 100 scalar function evaluations, which is more efficient. Although the computational complexities of using (2.27) and (2.36) both grow linearly with the number of function evaluations needed, using (2.36) does improve the optimization efficiency proportionally.

Combining the objective function (2.36) and the primary sensor constraints, we can formulate a multi-target tracking guidance law as an optimization problem to find the next waypoint and heading angle command:

$$\begin{aligned} \min_{x^{(ps)}(t+1),\theta} \quad & \sum_{j=1}^{N} W_j \Big\{ \log \left| \Lambda_{\tilde{P},j} \right| + \sum_i \log \left(\lambda^*_{i,j} - \pi_j(x^{(ps)}(t+1)) \right) \Big\} \\ \text{s.t.} \quad & x^{(ps)}(t+1) = f^{(ps)}(x^{(ps)}(t), u^{(ps)}(t)) \end{aligned} \tag{2.37}$$

In summary, the proposed algorithm has two components: the multi-target RBE state estimation and the optimal primary sensor guidance. At each time step, the estimation algorithm generates the estimated and the predicted states of the targets, whether these targets are being detected or not. Then, using the predicted targets states, the algorithm computes the expected covariance matrices of all the targets, and a guidance command is calculated to minimize the overall expected information entropy, which can be computed from the expected covariance matrices. The algorithm is executed recursively until the tracking mission is terminated.

2.4 Numerical Examples

This section presents numerical demonstration of the proposed approach with two illustrative test scenarios, where the targets to be tracked follow randomly generated trajectories and the sensor measurements are corrupted by random noise.

In the first scenario, only one target is tracked, and some occlusion areas that block the line of sight (LOS) are added in the map so that the target is bound to be lost at some point. There are two goals for this test: (1) validate the ability of the proposed estimation algorithm to track a lost target; and (2) study the techniques of discretizing the map and construct the Gaussian PDFs. The second scenario applies the proposed approach to a 3-target tracking mission in an open area (no LOS blocks). Based on the results in the first scenario, an optimized map discretization scheme is used to construct the Gaussian PDFs. As a comparison, we also simulate the tracking mission using the dead-reckoning estimation and the proposed guidance law. The performance of the proposed multi-target tracking algorithm is demonstrated via this test. All the simulations are done on a laptop with a i5-3210M CPU. The CPU has four 2.50GHz computation cores.

2.4.1 Target and Sensor Modeling

Let $x_j(t)$ and $y_j(t)$ be the position coordinates of the j th target at time t. The target trajectory is generated by the following model [27]:

$$\begin{aligned} x_j(t+1) &= x_j(t) + \Delta t \cdot v(t)\cos\theta(t) \\ y_j(t+1) &= y_j(t) + \Delta t \cdot v(t)\sin\theta(t) \end{aligned} \tag{2.38}$$

where $v(t)$ and $\theta(t)$ are random speed and direction so that they describe a random motion of the target; and Δt is a time increment.

For the estimator, we use the constant velocity model [51]:

$$x(t+1) = F_{cv}x(t) + G_{cv}w(t) \tag{2.39}$$

where

$$\begin{aligned} F_{cv} &= \mathrm{diag}[F_1, F_1], \quad G_{cv} = \mathrm{diag}[G_1, G_1] \\ F_1 &= \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix}, \quad G_1 = \begin{bmatrix} T_s^2/2 \\ T_s \end{bmatrix} \end{aligned} \tag{2.40}$$

$w(t) \sim \mathcal{N}(0,1)$ is unit white noise. T_s is the measurement sampling time and is set as one second in the tests. Note that the target tracking model differs from the target's trajectory generation model in (3.20), because in real applications the real motion of the target may not necessarily be available to the estimator.

A fixed-wing UAV is used as the sensor to track the targets. The cruise speed of the UAV is 165 knots (306 tm/h), and its minimum turning radius is 21 m.[1] The simulation time step is set to be 1 s, therefore the turning angle bound per one time step is 2.02 rad (about 116°). A video camera with a detection range of 120 m is mounted on the UAV. The camera is able to provide the position measurements of the targets subject to unbiased white noise with a covariance of $diag(10m, 10m)$, which

[1]The UAV specifications are set to simulate high-speed surveillance UAVs such as MQ/Wing Loong series.

means that the state measurement subjects to an standard deviation of 10m in both directions.

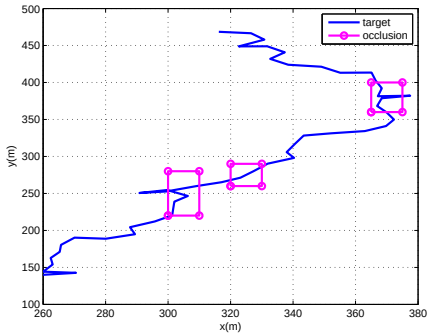


Fig. 2.3.: Test Scenario 1: Single Target Tracking

The weight term W_j in the guidance objective function is set to be:

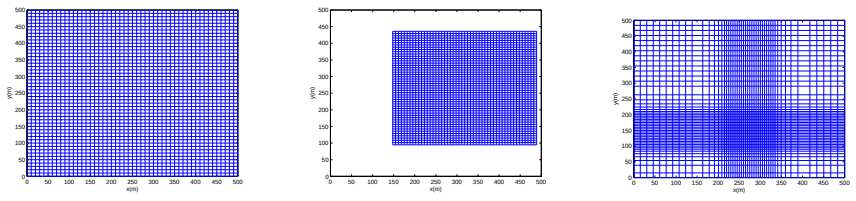
$$W_j = exp(t_j^L) - 1 \tag{2.41}$$

where t_j^L is the cumulative time steps that target j has been lost. Then, as the j-th target being lost, the priority to find the target grows exponentially. Since the guidance optimization problem is non-linear, we apply a light-weighted genetic algorithm to solve it [50].

In the simulation tests, the Gaussian sum terms are constructed as follows: we first discretize the map into rectangular grids; then, for each grid, a Gaussian sum term is constructed with its mean value set to be the center of the grid and its standard deviation set to be the size of the grid. For example, consider a 1-dimensional case, where a grid $[10, 20]$ is constructed on the x axis; then, the corresponding Gaussian sum term is $\mathcal{N}(x; 15, 10)$. It should be noted that there exist multiple ways of discretizing the map, and several of them are studied through simulation in the next section.

2.4.2 Scenario 1: Single-target Tracking with Blocked LOS

The test scenario is shown in Figure 2.3, where the blue solid line is the random target trajectory, and the circled magenta lines are the areas where the LOS is blocked. In this section, we propose and test 3 different ways of discretizing the map and constructing the Gaussian sum terms. The first way is to discretize the entire map uniformly. It is the most straightforward and easiest way to implement. A major drawback in this approach is that the grid points that are far away from the target have little contribution, since the prior probability values of these points are near-zero. As an improvement, the second proposed discretization technique only considers the 5-σ ellipsoid of the target's last prediction and uniformly discretize the area. In this method, the algorithm discretizes a dynamic area that changes over time. It should be noted that if the target is lost for a long time the ellipsoid will grow large as well as the covariance matrix, and this method will degenerate to discretizing the entire map area. The third approach still discretizes the entire map area, however, in contrast to uniform discretization, we derive such a grid that is dense near the UAV position and sparse far away. This approach pays more attention to the area close to the UAV, especially at the boundary of the the UAV's sensor range, where the likelihood given by (2.7) is discontinuous. Figure 2.4 gives an example of those different grid structures given the same number of grid points. For the sake of easy implementation, we use rectangular areas instead of the 5-σ ellipsoid and the sensor range circle.



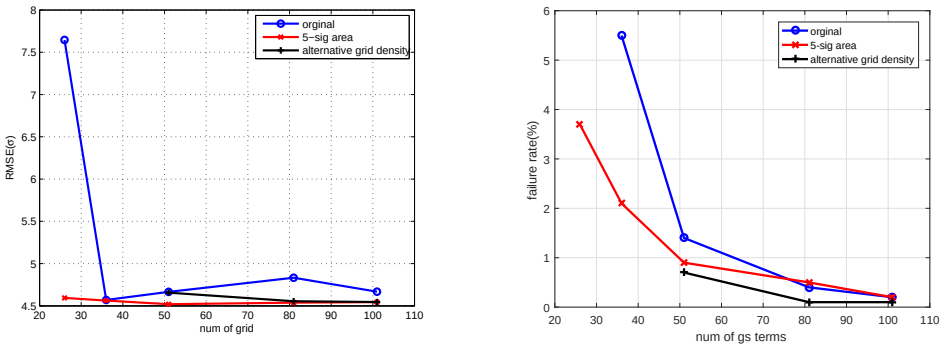
(a) original method (b) 5-σ area (c) alternative density

Fig. 2.4.: Examples of Different Grid Structures

We test these three discretization methods using different grid densities (number of grid points). For each grid density, 1000 Monte Carlo simulations of the test scenario have been performed, and the failure rate of the 1000 simulations is calculated. A failure is defined as: once the target is lost, it is unable to be re-captured till the end of the simulation. The failure rate is the total number of failures divided by 1000. We also calculate the normalized root-mean-square (RMS) error of the position estimation. The normalized RMS error is given by:

$$\text{err} = (\hat{x}(t) - x(t))^T \hat{P}(t)^{-1} (\hat{x}(t) - x(t)) \tag{2.42}$$

where we take the difference of the estimated and the real target positions and normalize it by the estimation covariance. It should be noted that in the calculation of average RMS error, only the successful test cases are considered.



(a) RMS error

(b) failure rate

Fig. 2.5.: Performance of Different GS Term Construction Methods

Figure 2.5 presents the average performance of the three proposed discretization methods with different grid densities. The circles, cross and plus signs correspond to the data points of the first, second and third methods, respectively. A summary of the simulation results is given in Table 2.1. It can be observed that with an appropriate grid density, our estimation algorithm can track a target with an average estimation

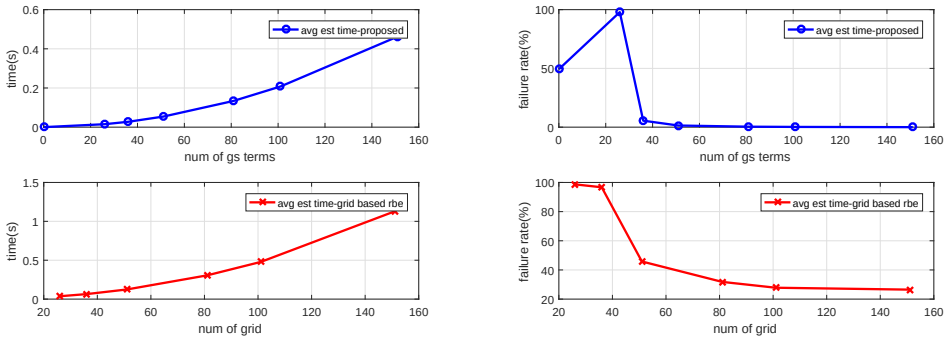
error bounded by the 5-σ ellipsoid, whether the target is detected or lost. In addition, the failure rate is below 2% with an appropriate grid density.

Table 2.1.: Summarized GSF Simulation Results

Test	GS Term Construction	# GS Terms	RMSE(σ)	Failure Rate (%)
1	whole map	25x25	7.7	98
2	whole map	50x50	4.6	1.5
3	whole map	100x100	4.6	0.2
4	5-σ area	25x25	4.6	3.8
5	5-σ area	50x50	4.5	0.9
6	5-σ area	100x100	4.5	0.2
7	alternative density	50x50	4.6	0.8
8	alternative density	100x100	4.5	0.2

As for the comparison of the different discretization methods, the third approach has excessively high failure rates at low grid densities and therefore their results are not included. Despite those discarded data points, one can observe that in comparison to the first construction method, the second and third methods generally have comparable precision and reliability improvements when the same grid density is used because they apply the limited resource to the areas which are more important and avoid those areas with negligible effects. In other word, these two methods can achieve similar performance level with fewer cost. Generally speaking, the second method (discretizing the 5-σ area) has the best balance of tracking error and reliability.

In addition, we show that our proposed estimation approach is more efficient and reliable than the grid-based direct numerical RBE [27]. Figure 2.6 compares the estimation time cost and failure rate of our algorithm using uniform map discretization to those of the numerical RBE. Table 2.2 summarizes the simulation results. As can be seen, our unified algorithm is overall more efficient and more reliable. For example,



(a) time comparison (b) failure rate comparison

Fig. 2.6.: Comparison of the Proposed Estimation and the Numerical RBE

Table 2.2.: RBE/GSF Simulation Results

Test	Algorithm	# of Grid/GS Terms	1-Step Time Cost(s)	Failure Rate (%)
1	RBE	25x25	0.2	95
2	RBE	30x30	0.25	92
3	RBE	50x50	0.3	48
4	RBE	80x80	0.4	38
5	RBE	150x150	1.2	30
6	GSF	25x25	0.02	98
7	GSF	30x30	0.04	5.5
8	GSF	50x50	0.1	1.5
9	GSF	80x80	0.18	0.2
10	GSF	160x160	0.42	0

if we require the tracking failure rate below 30%, the numerical RBE needs about 80×80 grids, which leads to the average estimation time cost of 0.4s per step; on the other hand, our proposed approach only needs about 30×30 GS terms, resulting the average estimation time cost of less than 0.05s per step. Further discussion can be found in our previous work.

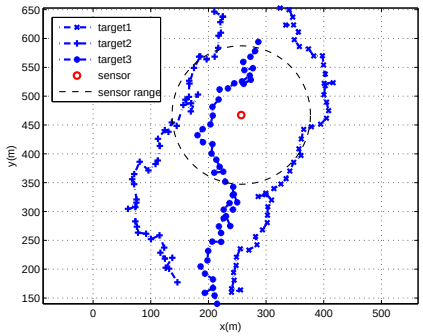


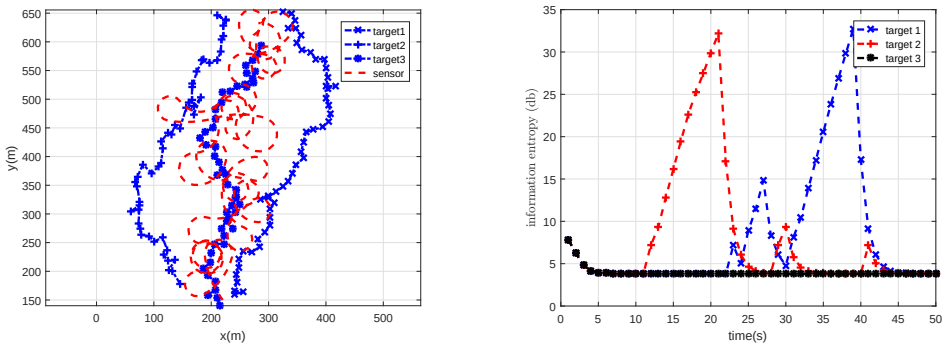
Fig. 2.7.: Test Scenario 2: Multi-target Tracking

2.4.3 Scenario 2: Multi-target Tracking

Figure 2.7 shows the multi-target tracking scenario; the figure illustrates the target trajectories as well as the sensor range. As a common situation in the context of multi-target tracking [43], although the targets move toward a similar direction, they can be distant during the mission. In our scenario, the maximum separation range of the targets is about $260m$. Two algorithm are tested in this scenario. The first one applies our proposed estimation and guidance algorithms. As a performance benchmark, the second one uses the dead-reckoning estimation and the proposed guidance law. Based on the study in the previous section, in the first test algorithm, we discretize the 5-σ area to a 50×50 grid space whenever a Gaussian sum needs to be constructed.

The test results of the first algorithm are presented in Figure 2.8. The UAV's trajectory is given by the dashed red line in Figure 2.8(a). From the trajectory it can be observed that whenever the UAV is unable to cover all the 3 targets, it tries to cover 2 of them first and then search for the lost one. A more quantified description of the UAV's behavior can be obtained from Figure 2.8(b), where the information entropy of the targets are given. The information entropy of the j th target at time t is a linear function of the logarithm of the determinant of its estimation covariance:

$$entropy = \log \left| P_j(t) \right| + b \tag{2.43}$$

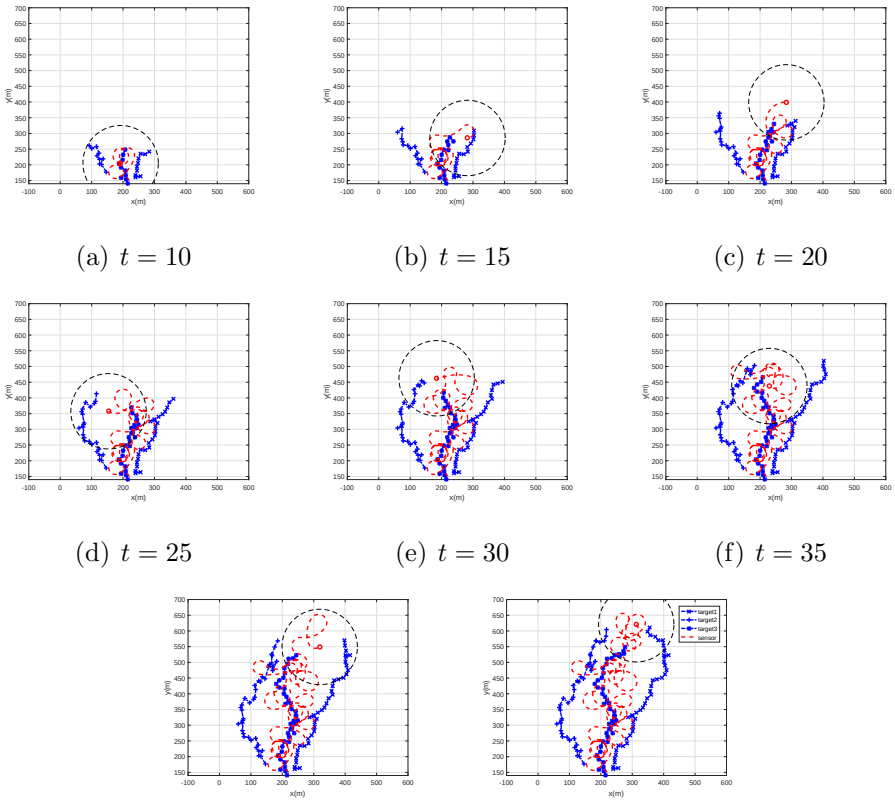


(a) trajectories (b) information entropy

Fig. 2.8.: Test Results of Algorithm 1: Proposed State Estimation Algorithm and Guidance Law

where b is constant. As a target being lost, its information entropy increases quickly; once the target is re-detected, its information entropy drastically drops. In Figure 2.8(b) one can observe the alternate loss of targets 1 and 2 from the rises and falls of the information entropy values. To better illustrate this point, Figure 2.9 shows the motions of the UAV and the targets at multiple time steps of the simulation result in Figure 2.8(a). The black dashed lines in the figures represent the sensor detection range. As can be seen, the targets move away from each other so that the sensor cannot cover all of them simultaneously; however, the UAV follows the desired tracking strategy such that it minimizes the uncertainty by tracking as many targets as possible (two targets in the scenario), and goes back for the lost target from time to time.

Figure 2.10 shows the optimization time costs of using the original objective function (2.27) and the transformed objective function (2.36) w.r.t the number of function evaluations. The data points are collected from the 3-target simulation case. As can be seen, typically one step of optimization involves hundreds of times of function evaluations; although the time cost grows linearly for both objective functions as we



(a) $t = 10$ (b) $t = 15$ (c) $t = 20$

(d) $t = 25$ (e) $t = 30$ (f) $t = 35$

(g) $t = 40$ (h) $t = 45$

Fig. 2.9.: Trajectories of Targets and the Tracking UAV

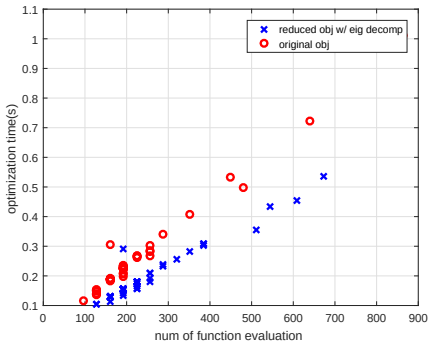
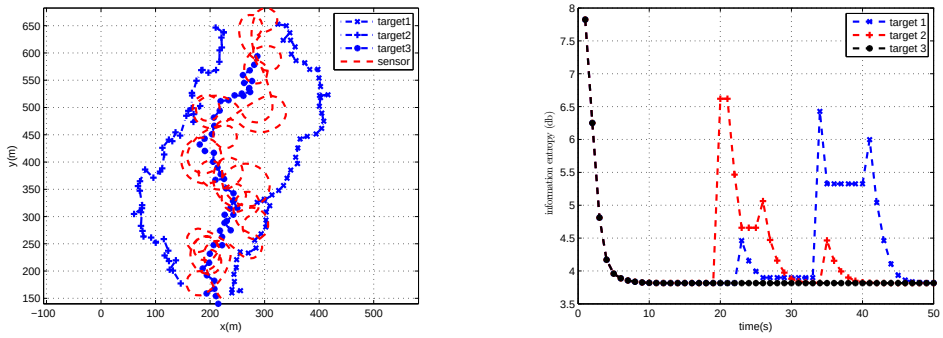


Fig. 2.10.: Optimization Time Cost

discussed previously, the transformed objective function that we propose is significantly more efficient.



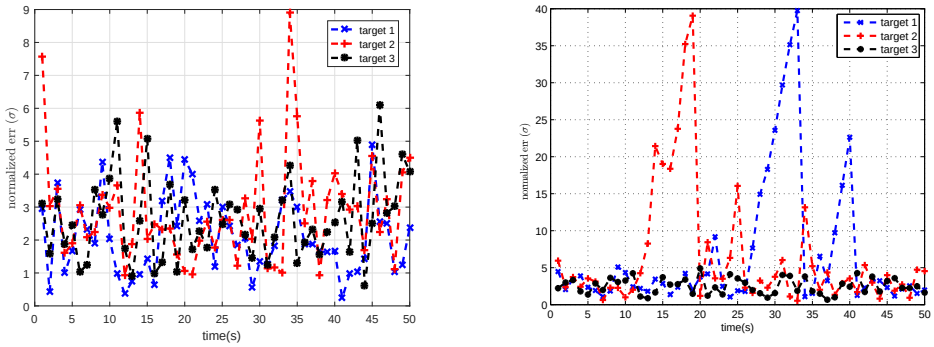
(a) trajectories

(b) information entropy

Fig. 2.11.: Test Results of Algorithm 2: Dead-reckoning and the Proposed Guidance Law

Figure 2.11 shows the test results of the second algorithm. The UAV's trajectory in Figure 2.11(a) indicates that it still follows the desired tracking strategy since it uses the proposed guidance law. However, the second algorithm does not track the targets as effectively as the first algorithm does, since the dead-reckoning estimation introduces higher estimation errors, especially when a target is lost. The information entropy has lower values in the second algorithm, however, this does not indicate that the second algorithm is more certain about the targets' location and thus better. The low information entropy values are due to the fact that the proposed Gaussian sum estimation introduces more uncertainty than the dead-reckoning estimation when a target is lost. In fact, given the large estimation error the dead-reckoning method has, it actually provides a highly optimistic estimation with a much smaller confidence area than the actual situation, which shows ineffective tracking performance.

Figure 2.12 and Table 2.3 compares the normalized RMS error of all the targets of the two estimation methods. The dead-reckoning estimation produces excessively high errors especially when a target is lost, therefore the tracking can be considered as 'failure' at those points. On the other hand, the proposed estimation algorithm produces a relatively low estimation error, even in the cases that a target is lost.



(a) RMS error of the proposed RBE (b) RMS error of dead-reckoning

Fig. 2.12.: RMS error comparison

Table 2.3.: Multi-target Tracking Simulation Results

Test	# of Targets	Algorithm	# of Grid/GS Terms	Max. Error(σ)
1	3	GSF	50x50	9
2	3	Dead-reckoning	N/A	40

3. MULTI-TARGET IDENTITY MANAGEMENT WITH MOBILE SENSORS

3.1 Background

In this chapter, we propose a new sensor scheduling algorithm to actively and optimally seek target identity information for multi-target identity management (MIM). Managing multiple target identities is essentially associating the target position estimates with their identities over time. In the past decades, approaches such as Multiple Hypothesis Tracking (MHT) [52, 53], Joint Probabilistic Data Association (JPDA) [6, 54, 55], and the Markov Chain Monte Carlo (MCMC) approach [56–58] were applied to the identity association problem. While the conventional data association techniques try to assign the target estimates to tracks, they do not explicitly tell whether different tracks belong to the same target identity, and thus may give incorrect identity association results especially, when targets get close to each other within the sensor's resolution range or cross their paths (known as track coalescence); in addition, the exponential complexity in associating the target estimates with the target identities is another problem of the above data association approaches. Recent years have seen studies addressing the above issues, which result in techniques of tracking labeled targets [59–62] and approximations to keep the computational complexity in check [63–67]. However, these general data association approaches do not consider a unique feature in the context of identity management: the local target identity information. This information may be available at any time during the identity management; it can be an observation of the target's shape or color from the sensors, or an transmission from an aircraft's identity transponder, etc. In the identity management scenarios, it is desired that this target identity information can be facilitated for multiple target tracking and identity management, whenever available.

As a solution to the identity management problem, the identity belief matrix, which stores and updates the probabilistic distributions of the target identities, was introduced [68]. Based on the concept, the Multiple-Target Tracking and Identity Management (MTIM) algorithm framework has been proposed to not only associate and manage the target identities efficiently over time with polynomial complexity, but also incorporates the local identity information whenever available [10, 69–72]. The local information incorporation plays a significant role in identity management, especially during the track coalescence cases. Almost all of the current studies assume that the local information is obtained frequently; it has not been seriously considered how the local information should be acquired efficiently when multiple sensors are available, which can be achievable due to the rapid development of sensor networks in the field of target tracking [73,74]. Specifically, let's consider a sensor that can identify the targets once they are within its surveillance range, and then, the problem of 'how to acquire the local information' becomes a sensor scheduling problem of which target the sensor should identify next and how to control the sensor to identify the selected target, so that all the targets can be identified and their identities tracked correctly and efficiently over time. The sensor scheduling problem for MTIM is composed of two key elements: optimal sensor scheduling (to decide which target to be identified next) and sensor control (to drive the sensor to identify the selected target). The target to be identified should be selected optimally based on the current knowledge of the target identities; for example, identifying a target whose identity is already known is meaningless and wasting available resources, while identifying a target whose identity is most unclear could be an optimal decision.

In summary, the main contribution of this chapter is to develop an MIM framework with a robust closed-loop sensor scheduling algorithm, which, for the first time, considers actively seeking the target identity information so that the targets' identities (along with the target states) are effectively and efficiently tracked and managed over time. The algorithm framework we propose is a general solution, i.e., the mathematical formulation does not consider the specific types of sensors or targets. In

real applications where the types of sensors and targets are given, one can choose the corresponding models for the algorithm.

The rest of this chapter is organized as follow: We present the mathematical definitions used in the chapter and introduce the basic concepts of identity management in Section 3.2. In Section 3.3, we first formulate the MIM algorithm with one secondary sensor, and then extend it to a decentralized network of multiple secondary sensors. The performance of the algorithm is demonstrated in Section 3.4.

3.2 Preliminaries

In this section, we present the mathematical notations and definitions of the problem used in the chapter. We also give a brief introduction of the basics of identity management, including the identity belief matrix, and its propagation and update procedures.

3.2.1 Sensor and Target Models

In this chapter, the primary sensor is assumed to be stationary and able to keep track of targets within its sensing range. The secondary sensors, on the other hand, are assumed to be mobile (e.g., UAVs). We denote the state of the m-th secondary sensor at time t as $x_m^s(t)$, and its equations of motion as:

$$x_m^s(t+1) = f_m^s(x_m^s(t), u_m(t)) \tag{3.1}$$

where $u_m(t)$ is the control input to the sensor at time t. It should be noted that the motion model of the secondary sensor (3.1) can be either a kinematic model or a dynamic model, depending on the actual application. For example, consider that we use a UAV equipped with a camera as the secondary sensor, then we may either use the dynamic equations for (3.1), with $u_m(t)$ being the dynamic input such as throttle, or assume that the lower level controller is already available and use the kinematic equations for (3.1), with $u_m(t)$ being the waypoint commands.

Denote the state of the j-th target at time t as $x_j(t)$, $j = 1, 2, ...N$, where N is the number of targets which is assumed to be known and fixed. In this chapter, we use the kinematic models for target state tracking [51]:

$$x_j(t+1) = f_j(x_j(t), w_j(t)) \tag{3.2}$$

where $w_j(t)$ is white noise. The state estimates of the targets are provided by the primary sensor at each time step. The estimate of the j-th target at time t is denoted as $\hat{x}_j(t)$.

3.2.2 Identity Management Basics

Identity Belief Matrix

The identity belief matrix $B(t)$ at the t-th time step is defined as an $N \times N$ doubly-stochastic matrix (a non-negative matrix whose row and column sums are 1), where N is the number of targets (columns) and the number of possible identities (rows). The entry of $B(t)$, denoted as $b_{ij}(t)$, represents the probability that the j-th target estimate $\hat{x}_j(t)$ has the i-th identity. The j-th column of $B(t)$, denoted as $\boldsymbol{b}_j(t)$, is then the probability distribution of $\hat{x}_j(t)$'s identity.

As time goes on, the identity belief matrix can be propagated with the new target estimates. This is accomplished by introducing the mixing matrix $M(t)$, which is also an $N \times N$ doubly-stochastic matrix. Its entry $m_{ij}(t)$ represents the probability of $\hat{x}_j(t)$ being originated from $\hat{x}_i(t-1)$, i.e., the estimates $\hat{x}_j(t)$ and $\hat{x}_i(t-1)$ belong to the same target identity. [1] The mixing matrix $M(t)$ is a collection of marginal association probabilities per se, and in theory can be calculated from the joint association probability [75]. In practice, however, to avoid the intensive calculation of the joint association probability, heuristic approaches can be applied by utilizing the dynamic

[1] Note that at different times t_1 and t_2, $M(t_1)$ and $M(t_2)$ are assumed to be statistically independent with each other.

information of the targets [68]. The propagation of the identity belief matrix can be described as:

$$B(t+1) = B(t)M(t+1) \tag{3.3}$$

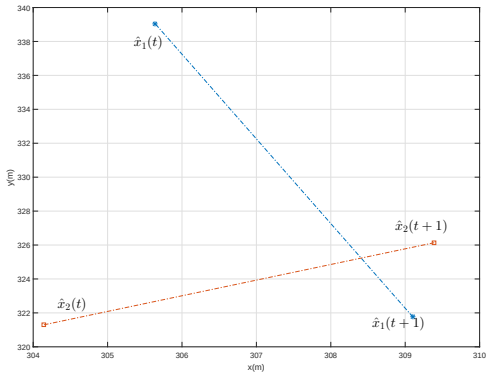


Fig. 3.1.: State Estimates for Two-target Example

For example, consider the two-target identity management example shown in Figure 3.1. In this example, we assume that target 1's ID is 'A' and target 2's ID is 'B'.[2] At time t, the target state estimates are given by $\hat{x}_1(t)$ and $\hat{x}_2(t)$, respectively, and if we assume that the probability of 'target 1's ID is B' is 0.2, then the belief matrix is:

$$B(t) = \begin{bmatrix} 0.8 & 0.2 \\ 0.2 & 0.8 \end{bmatrix}$$

At time $t+1$, the target estimates $\hat{x}_1(t+1)$ and $\hat{x}_2(t+1)$ get close to each other within the resolution range of a sensor so that they are almost equally likely to be originated from either of the previous estimates. For instance, suppose that there is a 60% probability that the target state estimates $\hat{x}_1(t)$ and $\hat{x}_1(t+1)$ belong to the same target, then, the mixing matrix is:

[2]In practice, ID can be a flight number such as 'BA2490', or an aircraft model such as 'Boeing 747', or friend or foe, depending on the application

$$M(t) = \begin{bmatrix} 0.6 & 0.4 \\ 0.4 & 0.6 \end{bmatrix}$$

Then, the belief matrix $B(t+1)$ at time $t+1$ can be updated as:

$$B(t+1) = B(t)M(t) = \begin{bmatrix} 0.56 & 0.44 \\ 0.44 & 0.56 \end{bmatrix}$$

In summary, one can recursively compute the identity belief matrix at each time. However, as shown in the example, the update procedure in (3.3) does not reduce the uncertainty about the target identities [68]. Therefore, local information about the targets' identities is needed to reduce the uncertainty in the target identities.

Local Information Update

As we previously stated, local information is the information about the target identities obtained from sensors. To clearly explain the idea, let us consider a 3-target system with 3 possible IDs: 'A', 'B' and 'C', and initially we have a uniform belief of the identity (i.e., we do not know who is who):

$$B = \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}$$

Suppose that at some instant, it is identified that the 3rd target's ID is 'C', then the local information is represented as a column vector $\begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$, and the prior identity belief matrix with the incoming local information becomes:

$$B = \begin{bmatrix} 1/3 & 1/3 & 0 \\ 1/3 & 1/3 & 0 \\ 1/3 & 1/3 & 1 \end{bmatrix}$$

This prior belief matrix violates the doubly-stochastic property, and therefore it should be logically scaled to:

$$B = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 1/2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Note that the above matrix is the updated belief matrix using the local information. As the example shows, the local information on a target can change the global information on all the targets' identities. Thus, the identity management of all the targets can be effectively and efficiently performed. Although quite trivial in this case, updating an arbitrary belief matrix is no easy task. The Sinkhorn scaling [76] is applied to scale any prior belief matrix to satisfy the doubly-stochastic property. The computational efficiency of the scaling procedure can be improved by taking polynomial approximation [72].

3.3 Multi-target Identity Management Algorithm

In this section, we first introduce the proposed MIM algorithm with one primary sensor and one secondary sensor to better illustrate the proposed algorithm. Then, we extend the MIM algorithm to multiple secondary sensors.

The overall structure of the proposed algorithm with one secondary sensor is shown in Figure 3.2. The primary sensor performs the multi-target tracking and the multi-target identity management tasks; the secondary sensor, on the other hand, tracks and identifies the targets through the sensor optimization and control processes. Each of the rectangular blocks inside the sensor blocks in Figure 3.2 represents an algorithm component.

3.3.1 Primary Sensor

Without loss of generality, in this chapter we assume the primary sensor to be stationary and have a sufficiently large detection range to detect all the targets. We

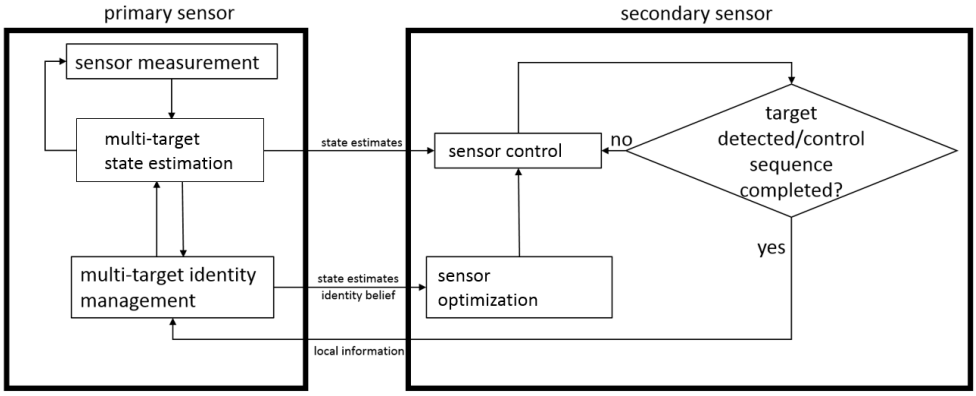


Fig. 3.2.: Overall Algorithm Structure

assign the primary sensor to perform the multi-target state estimation and identity management tasks.

At each time step, the primary sensor measures the states of the targets and assigns the measurements using the information provided by the identity management block to generate the state estimates of the targets. The state estimates provide necessary information for the primary sensor to perform the identity management task. In addition, the state estimates are also transmitted to the secondary sensor for its optimization and control. The primary sensor is able to generate the target estimates at each time step via the state estimation block. Since the algorithms running on the primary sensor are not the major concern of this chapter, we do not discuss them in detail. The algorithm architecture we propose is a general framework, therefore, any suitable multi-target tracking and measurement association algorithms [1,13,20,21,77] can be applied to the state estimation block. In practice, one may also use multiple primary sensors to jointly track multiple targets [78–81]. However, the primary goal of this chapter is to demonstrate the connection between the primary sensor and the secondary sensor, and therefore we only consider a single primary sensor for better illustration of the proposed method.

The identity management block stores an $N \times N$ identity belief matrix $B(t)$ where N is the number of targets and their identities, and updates it overtime. At each

time step, the identity management block first calculates the target state mixing matrix $M(t)$ using the target state estimates. The calculation of $M(t)$ is essentially associating measurements to the target estimates, and various approaches have been developed [68, 69, 71] and can be applied directly. Then, the belief matrix $B(t)$ is updated with the mixing matrix $M(t)$ via the update equation (3.3).

In addition to updating $B(t)$ with $M(t)$ at each time, the identity management block also accepts the local information provided by the secondary sensor whenever the secondary sensor is able to identify a target, and updates $B(t)$ with the local information as previously explained.

3.3.2 Secondary Sensor: Overview

The secondary sensor is assumed to be able to identify one target at a time whenever the target is within its sensor range. It is also assumed that the secondary sensor is able to estimate its own states.

Whenever the secondary sensor is un-occupied (e.g., after initialized or previous tasks completed), the sensor optimization block obtains the target state estimates and the identity management results from the primary sensor to find an optimal solution composed of the target to be identified next and a sequence of control inputs for the secondary sensor to capture the selected target. Once a solution is found, the sensor control block applies not only the control inputs from the sensor optimization block, but also the correction inputs calculated from the target state estimates provided by the primary sensor. The secondary sensor keeps pursuing the selected target until the control input sequence is completly executed (regardless of whether the target is identified or not at the end), before it starts a new sensor optimization operation. Whenever a target is identified, the secondary sensor sends the corresponding local information to the primary sensor, in the form of a column vector as introduced in Section 3.2.2. The design of the secondary sensor algorithm blocks is the major content of this chapter, and is thoroughly discussed below.

3.3.3 Secondary Sensor: Sensor Optimization

The sensor optimization block collects the target state estimates and the identity management information from the primary sensor, and solves an optimization problem to find the next target to be identified as well as the corresponding control input sequence. If no feasible optimal solution is found, the sensor optimization block provides a baseline solution. As we are considering only one secondary sensor for now, the superscript m denoting the m-th secondary sensor is omitted in this section.

Optimization Problem Formulation

In probabilistic target tracking scenarios, the concept of minimizing the expectation of a statistical cost function which represents the uncertainty level is a widely adopted information-driven approach [82–85]. Since the identity belief matrix $B(t)$ is essentially a matrix of probability distributions, one can calculate the uncertainty levels of the distributions (e.g., information entropy) and use them to quantify the knowledge of the target identities. In this chapter, we adopt the similar information-driven idea to formulate the optimization problem. By using the Shannon entropy [86], we define the total information entropy of the identity belief matrix $B(t)$ as:

$$H[B(t)] = \sum_{j=1}^{N} -\boldsymbol{b}_j(t)^T \log \boldsymbol{b}_j(t) = \sum_{j=1}^{N} \sum_{i=1}^{N} -b_{ij}(t) \log b_{ij}(t) \tag{3.4}$$

where $\boldsymbol{b}_j(t)$ represents the j-th column of $B(t)$ and $b_{ij}(t)$ is the i-th entry of $\boldsymbol{b}_j(t)$. It should be noted that $H[B(t)]$ represents the overall identity uncertainty of the entire system composed of N targets: the larger $H[B(t)]$ means that we are more uncertain about the target identities. Furthermore, if we know all the target identities, then $H[B(t)]$ reaches to its lower bound 0. Then, the sensor optimization problem is to find the next target to be identified such that the expectation of $H[B(t)]$ given that the target is identified is minimized. To express the expectation of $H[B(t)]$, we first define a random matrix $B(t|j,i)$ as the identity belief matrix B updated by the local

information that 'the ID of the j-th target has the i-th identity' at time t. Then, the expectation of $H[B(t)]$ given that the j-th target is identified, can be written as:

$$E\Big[H[B(t)]\big|j\Big] = \sum_{i=1}^{N} b_{ij}(t) H[B(t|j,i)] \tag{3.5}$$

i.e., the sum of all possible Shannon entropies from the j-th target being identified, weighted by the probability that the ID of the j-th target has the i-th identity.

To formulate the optimization problem, it also needs to be considered that the time gap between an optimal control sequence being found and the control sequence being completely executed, since it takes time for the secondary sensor to move and identify the selected target. The idea we use in this chapter is to treat the sensor motion as a constraint of the optimization problem, and investigate the cumulative performance of the objective function (3.5). Such idea is well developed and commonly adopted in the area of optimal control [87–89]. Specifically, we formulate the sensor optimization problem as below:

$$\begin{aligned}
\text{find} \quad & j, u(t), u(t+1), ..., u(t+T-1) \\
\min \quad & J = \sum_{\tau=0}^{T-1} E\Big[H[B(t+\tau)]\big|j\Big] \\
\text{s.t.} \quad & x^s(t+\tau+1) = f^s(x^s(t+\tau), u(t+\tau)) \\
& u(t+\tau) \in [u_{\text{inf}}, u_{\text{sup}}] \\
& \hat{x}_i(t+\tau+1) = f_i(\hat{x}_i(t+\tau), w_i(t+\tau)) \\
& B(t+\tau+1) = B(t+\tau) M(t+\tau+1) \\
& M(t+\tau+1) = f_M(\hat{x}_i(t+\tau), \hat{x}_i(t+\tau+1)) \\
& i = 1, 2, ..., N \\
& \tau = 0, 1, ..., T-1
\end{aligned} \tag{3.6}$$

The optimization problem in (3.6) aims to find a target to be identified and the corresponding sensor control sequence to minimize the cumulative expectation of the Shannon entropy over a given time window T. The sensor motion is constrained by

its motion model and control input limits. It should be noted that one can use either kinematic models or dynamic models as the sensor constraints, as discussed in Section 3.2.1. The prediction of the target states is also addressed as the constraints of the optimization problem. The target state prediction is necessary for the optimization problem to calculate the expected identity belief matrix. Since the actual dynamics of the target motion is generally unknown, we use the kinematic propagation models to predict the target states [51]. The identity belief matrix $B(t+\tau+1)$ and the mixing matrix and $M(t+\tau)$ are propagated using the target states prediction. We first propagate the mixing matrix $M(t+\tau)$ by associating the predicted target states $\hat{x}_i(t+\tau+1)$ with the previous target states $\hat{x}_i(t+\tau)$; the mixing matrix is then used to propagate the identity belief matrix $B(t+\tau+1)$, from which the Shannon entropy is calculated.

The mixed-integer optimization problem (3.6) is, however, not well posed, since the problem is to find a set of continuous variable $u(t), u(t+1), ...$ and an integer index j which represents the j-th column of a matrix, and it is difficult to optimize a matrix index [50]. To make the problem better posed, we first introduce a set of binary variables $Y_{j\tau} \in \{0,1\}$, where $j = 1, 2, ..., N$ and $\tau = 0, 1, ..., T-1$. The expression $Y_{j\tau} = 1$ represents the j-th target is identified by the secondary sensor at time τ, and $Y_{j\tau} = 0$ indicates that the j-th target is not identified by the secondary sensor. Then, the objective function in (3.6) can be written as:

$$J = \sum_{\tau=0}^{T-1} \left(Y_{j\tau} E\Big[H[B(t+\tau)] \Big| Y_{j\tau} = 1 \Big] + (1 - \sum_{j=1}^{N} Y_{j\tau}) H[B(t+\tau)] \right) \tag{3.7}$$

The term $(1 - \sum_{j=1}^{N} Y_{j\tau}) H[B(t+\tau)]$ in (3.7) represents the cost function when no target is identified at time τ: when $Y_{j\tau} = 0, \forall j$, the Shannon entropy of the belief matrix given that no target is identified is calculated; and when $Y_{j\tau} = 1, \exists j$, the term is 0. To mathematically describe the definition of $Y_{j\tau}$, additional constraints are included in the optimization problem:

$$Y_{j\tau} \in \{0,1\},\ \forall j,\tau \tag{3.8}$$

$$\sum_{j=1}^{N} Y_{j\tau} \leq 1,\ \forall \tau \tag{3.9}$$

$$f^{(Y)}(Y_{j\tau}, \hat{x}_j(t+\tau), x^s(t+\tau)) \leq 0,\ \forall \tau \tag{3.10}$$

The constraint (3.8) is the definition of the binary variable $Y_{j\tau}$, and the constraint (3.9) represents the assumption that at most one target at a time can be identified by the secondary sensor. The constraint (3.10) is a function of $Y_{j\tau}$, and the target and secondary sensor states such that the target should be identified by the sensor whenever $Y_{j\tau} = 1$. For example, assume a target is identified as long as it is within the sensor range, and denote the sensor range as r_s, and then the constraint (3.10) can be expressed as:

$$Y_{j\tau}\big(||\hat{x}_j(t+\tau) - x^s(t+\tau)||^2 - r_s^2\big) \leq 0,\ \forall j,\tau \tag{3.11}$$

It should be noted that the inequality (3.11) is an example of how the inequality (3.10) can be formulated, and the definition of (3.10) does not require that the sensor can always perfectly identify the targets. In fact, when the sensor missed/false detection is considered, the constraint (3.10) can be constructed accordingly by introducing corresponding factors such as the probability of detection, etc. With the additional variables and constraints, the optimization problem (3.6) can be rewritten as:

$$
\begin{aligned}
\text{find} \quad & S = \{Y_{j\tau}, u(t), u(t+1), ..., u(t+T-1)\} \\
\min \quad & J(S) = \sum_{\tau=0}^{T-1} \Bigg(Y_{j\tau} E\Big[H[B(t+\tau)] \big| Y_{j\tau} = 1 \Big] + \\
& (1 - \sum_{j=1}^{N} Y_{j\tau}) H[B(t+\tau)] \Bigg) \\
\text{s.t.} \quad & x^s(t+\tau+1) = f^s(x^s(t+\tau), u(t+\tau)) \\
& u(t+\tau) \in [u_{\text{inf}}, u_{\text{sup}}] \\
& \hat{x}_j(t+\tau+1) = f_j(\hat{x}_j(t+\tau), w_j(t+\tau)) \\
& B(t+\tau+1) = B(t+\tau) M(t+\tau+1) \\
& M(t+\tau+1) = f_M(\hat{x}_j(t+\tau), \hat{x}_j(t+\tau+1)) \\
& Y_{j\tau} \in \{0, 1\} \\
& \sum_{j=1}^{N} Y_{j\tau} - 1 \leq 0 \\
& f^{(Y)}(Y_{j\tau}, \hat{x}_j(t+\tau), x^s(t+\tau)) \leq 0 \\
& \forall j \in \{1, 2, ..., N\} \text{ and } \tau \in \{0, 1, ..., T-1\}
\end{aligned}
\tag{3.12}
$$

where S denotes the set of all optimization variables. The problem (3.12) is a dynamic programming problem with constraints, and numerical methods can be applied to solve the problem in practice [90].

It should be noted that although the problem (3.12) is posed in a simple yet illustrative environment, it can be extended to complex real environments. To consider practical situations, such as sensor positioning limits, field of view limits, line of sight blocks, etc., one simply need to formulate these conditions as additional constraints to the optimization problem. To study the solution properties of the optimization problem (3.12), we first introduce the following lemma:

Lemma 3.3.1 *Given an arbitrary identity belief matrix $B^{(0)}$ with its j-th row denoted as $\boldsymbol{b}_j$, and a local information vector $\boldsymbol{l}_{j_0}$ which identifies the j_0-th target, i.e., $\boldsymbol{l}_{j_0}^T = [0, ..., 0, 1, 0, ..., 0]^T$. Denote $B^{(2)}$ as the identity belief matrix updated from $B^{(0)}$ with*

$\boldsymbol{l}_{j_0}$, then the Shannon entropy of the updated belief matrix $B^{(2)}$ is no greater than that of the original belief matrix $B^{(0)}$:

$$H[B^{(2)}] \leq H[B^{(0)}]$$

and the equality holds if and only if the j_0-th target is already identified before the local information is incorporated.

Proof To prove the lemma, we separately discuss the 2 cases that the j_0-th target is identified or un-identified before $\boldsymbol{l}_{j_0}$ is obtained.

Case 1 The j_0-th target is already identified. In this case, the incorporation of $\boldsymbol{l}_{j_0}$ does not change the identity belief matrix, thus we have $B^{(2)} = B^{(0)}$ and $H[B^{(2)}] = H[B^{(0)}]$.

Case 2 The identity of the j_0-th target is unknown. We first denote the belief matrix that replaces the j_0-th column of $B^{(0)}$ with $\boldsymbol{l}_{j_0}$ as $B^{(1)}$, where $B^{(1)}$ is the unscaled version of the updated belief matrix $B^{(2)}$ such that $B^{(1)}$ violates the row sum and column sum constraints of the identity matrix. Then, it is obvious that $H[B^{(1)}] < H[B^{(0)}]$, since $-\boldsymbol{l}_{j_0}^T \log \boldsymbol{l}_{j_0} < -\boldsymbol{b}_{j_0}^T \log \boldsymbol{b}_{j_0}$ when the j_0-th target's identity is unknown a priori.

According to the Sinkhorn scaling algorithm [76], for all entries of the matrices $B^{(1)}$ and $B^{(2)}$ (denoted as $b_{ij}^{(1)}$ and $b_{ij}^{(2)}$, respectively), we have the following equation:

$$b_{ij}^{(2)} = d_{1i} b_{ij}^{(1)} d_{2j}$$

where d_{1i} and d_{2j} are scaling factors. Furthermore, specifically for the local information incorporation in this chapter, we have $d_{1i} > 1$ and $d_{2j} > 1$ for all i and j. This is because the scaling factors are in fact the ratios of the desired row/column sums (which are 1) to the actual row/column sums of $B^{(1)}$ (which are less than 1 since the local information substitutes some of the entries of $B^{(1)}$ for 0) [76]. Therefore, by defining $t_{ij} = d_{1i} d_{2j}$, we have:

$$
\begin{aligned}
H[B^{(2)}] &= \sum_{j=1}^{N}\sum_{i=1}^{N} -b_{ij}^{(2)} \log b_{ij}^{(2)} = \sum_{j=1}^{N}\sum_{i=1}^{N} -k_{ij} b_{ij}^{(1)} \log(t_{ij} b_{ij}^{(1)}) \\
&= \sum_{j=1}^{N}\sum_{i=1}^{N} -k_{ij} b_{ij}^{(1)} \log b_{ij}^{(1)} - k_{ij} b_{ij}^{(1)} \log k_{ij} \\
&< \sum_{j=1}^{N}\sum_{i=1}^{N} -b_{ij}^{(1)} \log b_{ij}^{(1)} \\
&= H[B^{(1)}]
\end{aligned}
$$

The inequality holds since $t_{ij} > 1$ and the term $t_{ij} b_{ij}^{(1)} \log k_{ij}$ is strictly positive. This completes the proof as we have $H[B^{(2)}] < H[B^{(1)}] < H[B^{(0)}]$, when the identity of the j_0-th target is unknown a priori.

Combining Cases 1 and 2, the lemma is proved. ■

Intuitively, Lemma 3.3.1 indicates the fact that identifying a target reduces the uncertainty of the identity belief matrix, thereby the uncertainty of the entire system composed of N targets. With Lemma 3.3.1, we now define two sets of feasible solutions to the optimization problem (3.12): $\mathcal{S}_0 = \{S | Y_{j\tau} = 0, \forall j, \tau\}$ and $\mathcal{S}_1 = \{S | Y_{j\tau} = 1, \exists j, \tau\}$. $\mathcal{S}_0$ represents the set all feasible solutions such that no target is identified during the optimal control sequence, and $\mathcal{S}_1$ represents the set of all feasible solutions such that at least one target is identified at some instance. Then, we have the following theorem:

Theorem 3.3.2 *Let S^* be the optimal solution to the problem (3.12), and let $\mathcal{S}_0$ and $\mathcal{S}_1$ defined as above. If $\mathcal{S}_1 \neq \emptyset$, then $S^* \in \mathcal{S}_1$.*

Proof If $\mathcal{S}_0 = \emptyset$, the theorem holds. Otherwise, let S_0 and S_1 be two feasible solutions to the problem (3.12), where $S_0 \in \mathcal{S}_0$ and $S_1 \in \mathcal{S}_1$. Moreover, without loss of generality, we assume that $\forall Y_{j\tau} \in S_1$, $Y_{j_0\tau_0} = 1$ and $Y_{j\tau} = 0, \forall j\tau \neq j_0\tau_0$. Then, by the definition of the cost function (3.7), we have:

$$
J(S_1) - J(S_0) = E\Big[H[B(t+\tau)] \big| Y_{j_0\tau_0} = 1 \Big] - H[B(t+\tau)]
$$

since all the other terms in the cost functions $J(S_1)$ and $J(S_0)$ are identical and can be canceled out. By Lemma 3.3.1, it can be deduced that:

$$E\Big[H[B(t+\tau)]\big|Y_{j_0\tau_0}=1\Big] < H[B(t+\tau)] \tag{3.13}$$

Therefore, we have $J(S_1) < J(S_0)$. The inequality means that any feasible solutions from the set $\mathcal{S}_1$ is better than all the feasible solutions from the set $\mathcal{S}_0$, which concludes the proof. ■

Theorem 3.3.2 indicates the optimality of the problem (3.12): as long as it is feasible, solving the optimization problem (3.12) always yields to the identification of a target at some point within the given time window to minimize the Shannon information entropy of the targets' identities.

Robustness and Convergence

When no feasible solution exists for the optimization problem (3.12), we apply the following baseline strategy as a robustness safeguard to prevent the algorithm from abruptly terminating: pursue the target that is currently most uncertain (greedy optimization), i.e., at time t, the secondary sensor starts to pursue the j-th target with the highest $-\boldsymbol{b}_j(t)^T \log \boldsymbol{b}_j(t)$. In fact, the strategy is equivalent to minimizing the objective function (3.7) with the time window of $T = 0$, regardless of the constraints and the control inputs, and perform the pursuit control separately. The pursuit control, however, is not the major concern of this chapter, and various pursuit strategies have been solidly developed and can be applied to a range of kinematic/dynamic objects [91, 92].

It can be shown that the performance of applying the proposed optimization strategy along with the robustness safeguard is guaranteed under the following assumptions:

(a) No prior information about the target identities is initially available.

(b) The targets do not get close within the sensor's resolution limit after the algorithm begins.

(c) The effect of $M(t)$ is negligible (i.e., $M(t)$ can be considered as a unit matrix).

Assumptions (b) and (c) in fact consider an ideal case where the targets do not interact with each other. In practice, however, whenever two targets interact with each other, e.g., get close within the sensor's resolution limit, it can be considered as a 're-initialization' of the algorithm, and the assumptions will hold until the targets start to interact again. Then, under the assumptions, we have the following theorem:

Theorem 3.3.3 *Given assumptions (a), (b) and (c), the identity belief matrix $B(t)$ updated by the secondary sensor converges to the true identity in at most N optimization steps, where N is the number of targets.*

Proof By Lemma 3.3.1, once the j-th target is identified, it will not be revisited given the assumptions, since identifying any other uncertain targets results in a lower cost function value, making it a better solution to problem (3.12) than re-identifying the j-th target. Note that the previous statement holds whether the solution is found from the optimization problem (3.12) or from the robustness safeguard. Either way, all targets will be visited exactly once in N optimization steps, i.e., all the target identities will be certain in at most N optimization steps. ■

Note that Theorem 3.3.3 assumes the perfect identification of a target. In the cases where the target identification is not perfect, i.e., the local information generated by the secondary sensor is a belief of the target's identities which is uncertain to some extent, the true identity belief is no longer achievable. However, the identity belief matrix still converges to a minimum uncertainty level based on the 'quality' (uncertainty) of the local information within N optimization steps, as can be seen in the simulation results.

3.3.4 Secondary Sensor: Sensor Control

If there is no feasible solution to the optimization problem (3.12), the sensor control block pursues the target given by the robustness safeguard. Whichever the pursuit strategy is used, the control block uses the target state information provided by the primary sensor to pursue the target. On the other hand, if a solution to the optimization problem (3.12) is found, denoted as $S^* = \{u^*(t+\tau), Y^*_{j\tau}\}$, where $j = 1, 2, ..., N$ and $\tau = 0, 1, ..., T-1$, the sensor control block applies the control inputs until any target is identified, and once there is a target identified, the algorithm loops back to the sensor optimization block. To interpret the control strategy mathematically, we first define:

$$\tau_0 = \inf\{\tau \big| Y^*_{j\tau} = 1\}$$
$$j_0 \in \{j \big| Y^*_{j\tau_0} = 1\}$$

Then, the control strategy can be described as: whenever an optimal solution S^* is given, the sensor control block applies the control sequence $u^*(t), u^*(t+1), ..., u^*(t+\tau_0)$ and identifies the j_0-th target at time $t+\tau_0$. The above control strategy requires the j_0-th target needs to be identified by the secondary sensor at time $k+\tau_0$, which is generally not guaranteed due to the imprecise target motion prediction. Thus, additional control inputs are required so that the prediction error can be compensated during the sensor control. The control strategy we apply is to keep the relative position between the j_0-th target and the secondary sensor as planned, i.e., feed the prediction error of the j_0-th target to the secondary sensor to calculate correction control inputs in addition to the planned control input sequence. Let $u^*(t_s)$, $t_s = t, t+1, ..., t+\tau_0$ be the control inputs in the optimal solution S^*, and $u(t_s)$ be the actual control inputs applied in the sensor control block. Then, the proposed idea is essentially calculating the correction control inputs $u^c(t_s)$ from the prediction error such that:

$$u(t_s) = u^*(t_s) + u^c(t_s)$$

This way, the control inputs obtained from solving the optimization problem (3.12) are still applied, along with a sequence of correction terms. Next, we show that the prediction error can be compensated by calculating $u(t_s)^c$ from appropriate feedback controllers, if the linear sensor motion models are used for the secondary sensor optimization/control. Denote the actual state of the secondary sensor at time t_s, $(t_s = t, t+1, ..., t+\tau_0)$ as $x^s(t_s)$, the predicted secondary sensor state generated by the optimal control input $u^*(t_s)$ at time t_s as $x^{s*}(t_s)$, and the deviation between them as $d^s(t_s)$, i.e.,:

$$\begin{aligned} x^s(t_s+1) &= f^s(x^s(t_s), u(t_s)) \\ x^{s*}(t_s+1) &= f^s(x^{s*}(t_s), u^*(t_s)) \\ x^s(t_s) &= x^{s*}(t_s) + d^s(t_s) \end{aligned} \tag{3.14}$$

Similarly, let the actual and the predicted states of the j_0-th target, and the deviation between them at time t be $x_{j0}(t_s)$, $x_{j0*}(t_s)$, and $d_{j0}(t_s)$, respectively. Then, we also have:

$$x_{j0}(t_s) = x_{j0*}(t_s) + d_{j0}(t_s) \tag{3.15}$$

Let $d_p^s(t_s)$ and $d_{p,j0}(t_s)$ be the position components of $d^s(t_s)$ and $d_{j0}(t_s)$, respectively. Then, the proposed strategy is then in fact controlling the deviation $d_p^s(t_s) \to d_{p,j0}(t_s)$ through the inputs $u(t_s) = u^*(t_s) + u^c(t_s)$. Particularly, if the linear motion model is used for the secondary sensor, i.e.,:

$$x^s(t_s+1) = A^s x^s(t_s) + B^s u(t_s)$$

Then, combining the equations in (3.14), we have the propagation equation of the deviation between the actual and the predicted states of the secondary sensor:

$$d^s(t_s+1) = A^s d^s(t_s) + B^s u^c(t_s) \tag{3.16}$$

Then, the proposed strategy is equivalent to controlling the linear system (3.16) with an input to the position components. As long as the system (3.16) is stablizable,

the goal can be achieved by designing appropriate feedback controllers, e.g., PID controllers or other feedback controllers for linear systems for $u^c(t_s)$ [93]. Furthermore, we would like to make a few additional remarks: (1) the proposed control strategy only requires the motion model of the secondary sensor to be linear, and the target motion prediction model makes no difference to the conclusion; (2) the proposed strategy can be extended to nonlinear sensor motion models through proper linearization procedures, as has been extensively studied in the area of nonlinear systems and control [94]; and (3) in practice, the 'actual' target and sensor states used by the feedback controller are generally the estimates of the states when the measurements are corrupted by noise.

3.3.5 Extension to Multiple Secondary Sensors

In practice, a single secondary sensor may have limited mobility and operation range, which could make our algorithm's performance degrade in scenarios with a large operation area and numerous targets. To improve the proposed algorithm's capability and practicality for larger-scale applications, we extend the algorithm to multiple secondary sensors. Generally, managing multiple sensors involves selecting/assigning sensors to their tasks, and let the sensors plan their works independently (for decentralized networks) [95, 96]. In this section, we propose a strategy of secondary sensor selection and independent planning.

Assume the operation area is bounded and known a priori, we divide it into multiple sectors. Each secondary sensor is assigned to a sector: the sensor assigned to a sector can identify the targets within the sector. Whenever a secondary sensor is un-employed, it obtains the state and identity information of all the targets within its assigned sector from the primary sensor. Figure 3.3 shows the modified algorithm architecture of the primary/secondary sensors. The newly added 'target selection' block performs the task of selecting and sending the target information to the secondary sensor. The communication strategy is identical between the primary sensor and

each secondary sensor, and the secondary sensors operate independently, as shown in Figure 3.4.

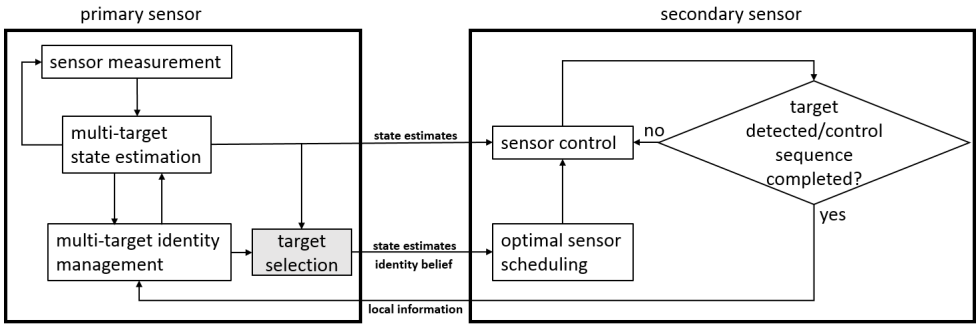


Fig. 3.3.: Algorithm Structure and Sensor Communication Strategy

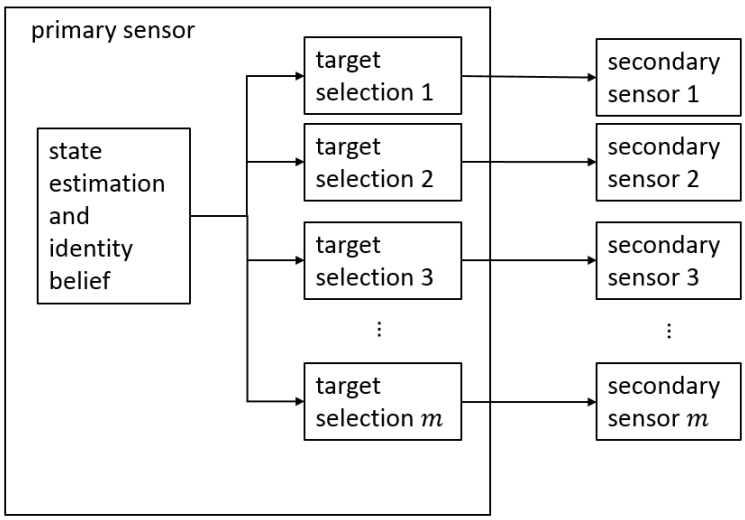


Fig. 3.4.: Multiple Secondary Sensor Operation Structure

After a secondary sensor obtains the target information, it solves an optimization problem to find the next target to be identified. We extend the optimization problem (3.12) with additional constraints to address the setup that the secondary sensor is assigned to a sector. The optimization problem solved by the m-th secondary sensor is formulated as follow:

$$
\begin{aligned}
\text{find} \quad & S = \{Y_{m,j\tau}, u_m(t), u_m(t+1), ..., u_m(t+T-1)\} \\
\min \quad & J(S) = \sum_{\tau=0}^{T-1} \Bigg(Y_{m,j\tau} E\Big[H[B_m(t+\tau)] \Big| Y_{m,j\tau} = 1 \Big] + \\
& (1 - \sum_{j=1}^{N_m} Y_{m,j\tau}) H[B_m(t+\tau)] \Bigg) \\
\text{s.t.} \quad & x_m^s(t+\tau+1) = f_m^s(x_m^s(t+\tau), u_m(t+\tau)) \\
& u_m(t+\tau) \in [u_m^{\inf}, u_m^{\sup}] \\
& g_m^s(x_m^s(t+\tau), u_m(t+\tau)) < 0 \\
& \hat{x}_j(t+\tau+1) = f_j(\hat{x}_j(t+\tau), w_j(t+\tau)) \\
& B_m(t+\tau+1) = B_m(t+\tau) M_m(t+\tau+1) \\
& M_m(t+\tau+1) = f_M(\hat{x}_j(t+\tau), \hat{x}_j(t+\tau+1)) \\
& Y_{m,j\tau} \in \{0,1\} \\
& \sum_{j=1}^{N_m} Y_{m,j\tau} - 1 \leq 0 \\
& f_m^{(Y)}(Y_{m,j\tau}, \hat{x}_j(t+\tau), x^s(t+\tau)) \leq 0 \\
& \forall j \in \{1,2,...,N_m\} \text{ and } \tau \in \{0,1,...,T-1\}
\end{aligned}
\tag{3.17}
$$

where N_m is the number of targets within the m-th secondary sensor's assigned sector. B_m is the identity belief matrix of the N_m targets, which is not square but $N \times N_m$. M_m is the $N_m \times N_m$ mixing matrix of the N_m targets. In addition to the sensor's motion model and control input limits, its motion is also constrained by the assigned sector which is defined by the function $g_m^s(\cdot)$. The problem (3.17) solved by the m-th secondary sensor only optimizes its own (instead of global) information entropy within its assigned sector, since the secondary sensors are independent of each other and therefore have no information from each other.

Whenever the optimization problem is solved, the secondary sensor follows the same control strategy proposed in Section 3.3.4 to identify the target. The extended

algorithm is decentralized in that the secondary sensors do not need to synchronize with each other, and each secondary sensor solves its own optimization problem.

It should be noted that the mathematical guarantees we obtained in Section 3.3.3 may not hold when the targets keep jumping between sectors. However, we show in simulation that the algorithm performs well in the scenarios with common target motion patterns.

3.4 Numerical Examples

In this section, we demonstrate the performance of the proposed algorithm, with single and multiple secondary sensors, via illustrative numerical examples. The targets in the simulation scenarios are assumed to be ground targets (e.g., ground vehicles). In some test scenarios, the targets are set to be non-maneuvering and move with constant speeds, while in other test scenarios the targets are maneuvering and their trajectories are randomly generated. As a benchmark, we also run the simulation cases without the secondary sensors to identify the targets, i.e., only the multi-target state estimation and the identity belief matrix propagation are performed. Since the primary sensor still performs tracking and data association, these benchmark cases are similar to the conventional multi-target tracker, where local identity information is not incorporated.

3.4.1 Sensor Modeling

We use a stationary sensor with a detection area that covers the entire map (about 1km$\times$1km) as the primary sensor, and fast-moving UAVs equipped with fixed narrow-ranged sensors as the secondary sensors.

At each time step, the primary sensor measures all the target positions with the measurement noise covariance:

$$\begin{bmatrix} (10)^2 & 0 \\ 0 & (10)^2 \end{bmatrix}$$

which means the standard deviation of the position error is 10m in both dimensions. Since the multi-target tracking block of the primary sensor is not the major concern of the chapter, we use the standard Kalman filter [20] to track the targets, with the constant velocity target state propagation model [51]:

$$x(t+1) = F_{cv}x(t) + G_{cv}w(t) \tag{3.18}$$

where

$$F_{cv} = \text{diag}[F_1, F_1], \quad G_{cv} = \text{diag}[G_1, G_1]$$
$$F_1 = \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix}, \quad G_1 = \begin{bmatrix} T_s^2/2 \\ T_s \end{bmatrix}$$

$w(t) \sim \mathcal{N}(0,1)$ is unit white noise, and T_s is the measurement sampling time which is set to be 1s. The constant velocity target propagation model (3.18) is also used by the optimization problem (3.12).

Each secondary sensor (UAV) flies at a constant cruise speed of 65tts (about 30m/s) with a minimum turn radius of 21m.[3] It is assumed that the UAV has a closed-loop position controller so that given a waypoint command input, the UAV is able to fly to the waypoint. Then, the sensor (UAV) motion model used in the optimization can be written as the following kinematic model:

$$x^s(t+1) = A^s x^s(t) + u(t)$$

where $A^s = diag(1,1)$ is the unit matrix. The input $u(t)$ is the essentially the relative position (regarding to the UAV's current position) of the next waypoint; $u(t)$ is bounded since the waypoint should be reachable by the UAV in 1 time step. It should be noted that the kinematic model is only used for the optimization, and the

[3]This is a common setup for a large variety of UAVs, e.g., see http://drones.cnas.org/drones/.

simulation of the UAV trajectory uses a line-of-sight waypoint guidance model [97]. For the sensor control, the correction input $u^c(t)$ is given by the following feedback controller:

$$u^c(t) = K(d_{p,j0}(t) - d_p^s(t)) \tag{3.19}$$

where K is the control gain and its value is set to be 0.9 in this section. The detection range of the sensor mounted on the UAV is set to be 30m, i.e., the UAV is able to identify one target within 30m at each time. Due to the short detection range of the sensor, the UAV needs to move close to a target in order to identify it, as will be shown in the results.

3.4.2 Single Secondary Sensor and Non-maneuvering Targets

The test scenario is shown in Figure 3.5, where 5 targets move at different yet constant velocities so that their motion can be predicted accurately. The time duration of the scenario is 100s. We use this scenario to test the performance of the algorithm under both the perfect and partial target identification conditions.

Case 1: Perfect Target Identification

In this case, the secondary sensor's target identification is assumed to be perfect such that the true target identity can be detected. We test the proposed algorithm with different optimization time windows T in the problem (3.12). In addition, two different initial conditions are applied to the simulation tests: (1) the worst case, i.e., the identities of all the targets are initially unknown so that the probability of the target identities are uniformly distributed at the beginning; and (2) the best case, i.e., all the targets' identities are initially known.

Figure 3.6 shows the trajectories of the targets and the secondary sensor (UAV) at multiple time instances during the scenario with an optimization time window of $T = 15$s, given the worst case initial condition. The trajectory of the tracking UAV

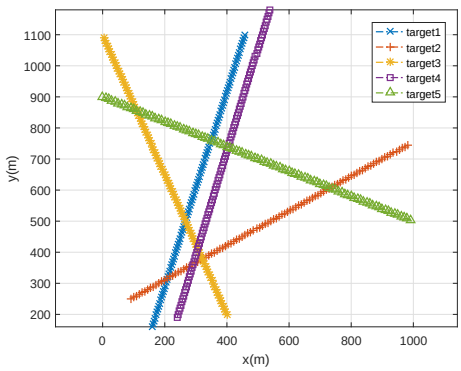


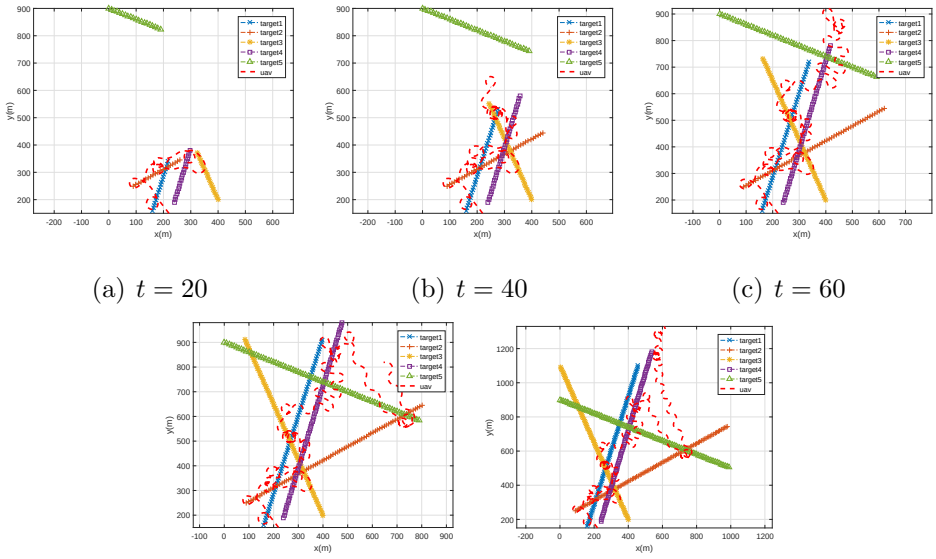
Fig. 3.5.: Target Trajectories of the Test Scenario

is given by the red dashed line. Figure 3.6 shows that the secondary sensor always moves to and identify the targets that get too close (thereby losing their identities). Moreover, between the time instances $t = 50$ and $t = 80$, the 2nd target (given by the orange trajectory) and the 5th target (given by the green trajectory) start to move close to each other, although they are still far away and differentiable between the time instances. However, the algorithm is able to predict the behavior of those two targets and commands the UAV to start to move to the area where the two targets possibly get close.

To study the performance of the algorithm, we first quantify the performance using the KL divergence [98]. Namely, let I be the true identity matrix, which is essentially an identity belief matrix whose entries are composed of only 0s and 1s. We study the KL divergence between the true identity I and the identity belief matrix B:

$$D_{KL}(I||B) = -\sum_{j=1}^{N}\sum_{i=1}^{N} I_{ij} \log \frac{B_{ij}}{I_{ij}}$$

It should be noted that a small KL divergence $D_{KL}(I||B)$ indicates that our identity belief matrix is close to the true identity (and hence certain), and vice versa. Furthermore, $D_{KL}(I||B) = 0$ means all the targets are correctly identified. Figure 3.7 shows the KL divergence $D_{KL}(I||B)$ over time corresponding to the simulation result



(a) $t = 20$ (b) $t = 40$ (c) $t = 60$

(d) $t = 80$ (e) $t = 100$

Fig. 3.6.: Trajectories of the Targets and the Tracking UAV

shown in Figure 3.6. As shown by the red line, the KL divergence starts high and drops every time a target is identified. In addition, the KL divergence converges to 0, i.e., all the targets are correctly identified after the secondary sensor performs 4 identifications, as predicted by Theorem 3.3.3. In addition, Figure 3.7 also presents the identity management result without using the proposed secondary sensor identification algorithm, given the worst case initial condition. As shown by the blue line, the KL divergence does not decrease from the start, i.e., the targets' identities can not be differentiated. However, the proposed algorithm using a secondary sensor can reduce the uncertainty of the targets' identities, and eventually all the targets' identities are correctly identified and maintained.

Figure 3.8(a) gives the probabilities of each target having each of the identities along the test duration, in the test case using the proposed secondary sensor algorithm. One can observe a mixing of the probabilities of targets 3 and 4 between the time instances $t = 20$ and $t = 30$, corresponding to the peak value of the red line in Figure 3.7. However, the identity belief is restored by the algorithm quickly

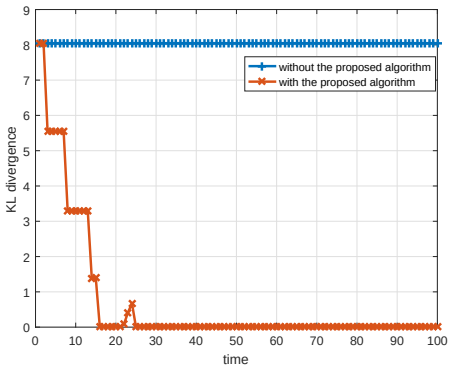
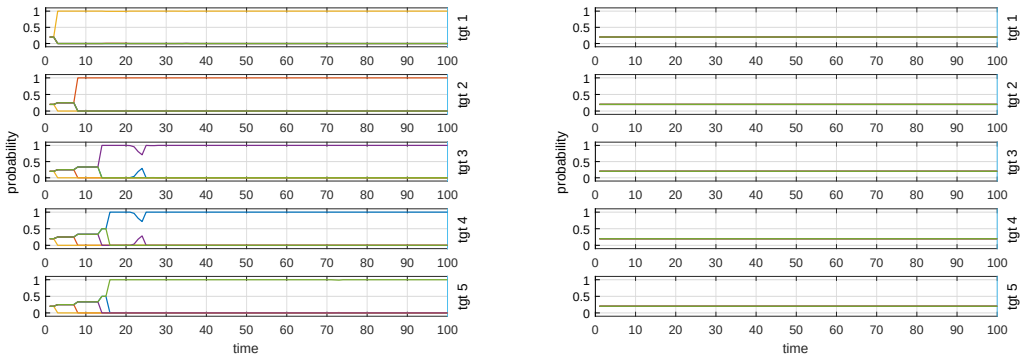


Fig. 3.7.: Quantified Performance Measured in KL Divergence



(a) with the secondary sensor identification (b) without the secondary sensor identification

Fig. 3.8.: Probabilities of the Targets' Identities with the Worst Initial Condition

after the mixing of the probabilities appears, as the secondary sensor identifies the uncertain targets. Moreover, it should be noted that the probability mixing rarely happens even though the targets get close at multiple time instances. This is because for most of the mixing situations, the algorithm successfully predicts them and keeps identifying the targets when they start to have even the slightest uncertainties, and the correct identity belief matrix is able to be maintained. On the other hand, as shown by Figure 3.8(b), the probabilities of the targets' identities remain uniformly distributed in the test case without using the proposed algorithm, which corresponds to the non-decreasing KL divergence in Figure 3.7.

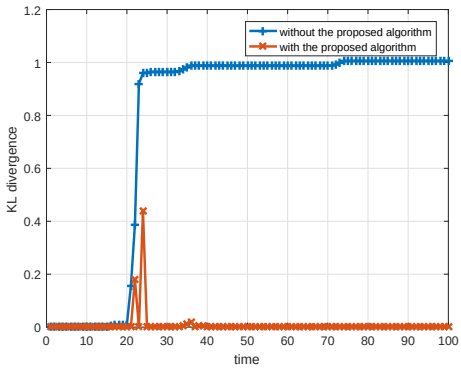
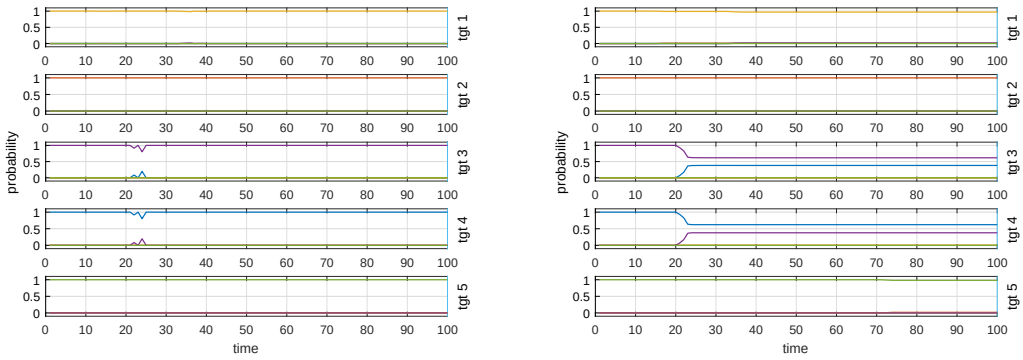


Fig. 3.9.: Quantified Performance Measured in KL Divergence



(a) with the secondary sensor identification (b) without the secondary sensor identification

Fig. 3.10.: Probabilities of the Targets' Identities with the Best Initial Condition

We also run the simulation scenario with and without the proposed algorithm given the best case initial condition, where the targets' identities are known from the start. As shown by the blue line in Figure 3.9, when the proposed secondary sensor identification algorithm is not used, the KL divergence increases and converges to a non-zero value, i.e., even though the target identities are initially known, they can be mixed due to the interactions between the targets and cannot be recovered without the secondary sensor identifying them. On the other hand, as shown by the red line, the targets' identities can be accurately tracked over time when the proposed secondary sensor identification algorithm is applied. The probabilities of the targets' identities

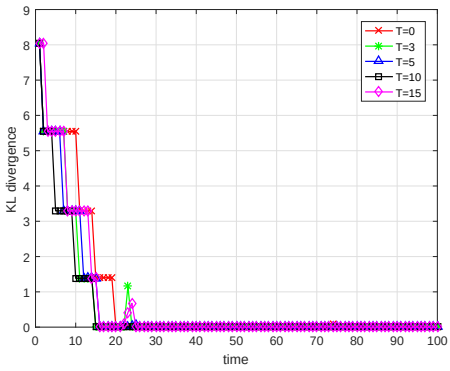


Fig. 3.11.: Performance with Different Optimization Time Window

given the best case initial condition are shown in Figure 3.10. Similar to those in the worst case, there is a mixing of the probabilities of targets 3 and 4 between the time instances $t = 20$ and $t = 30$ as the two targets cross their tracks, even though their identities are initially known. The mixing of the probabilities corresponds to the rapid increase of the KL divergence in Figure 3.9. As Figure 3.10 shows, the mixing of the probabilities can be successfully recovered by the secondary sensor identification.

By calculating the KL divergence $D_{KL}(I||B)$, Figure 3.11 compares the performance of the algorithm run with different optimization time windows T, given the worst case initial condition. It should be noted that the case of $T = 0$ means that the algorithm does not solve the optimization problem (3.12) and only applies the baseline pursuit strategy (i.e., pursuit the currently most uncertain target). As the figure shows, the proposed algorithm always converges in certain time steps as described by Theorem 3.3.3. However, the actual number of time steps needed for convergence varies depending on the choice of T. Applying only the baseline pursuit strategy takes more time steps to converge, since the control inputs generated by the baseline strategy are not guaranteed to be as optimal as those obtained from solving the optimization problem (3.12). Furthermore, the test result shows that increasing the time window T does not necessarily improve the performance. This is because the prediction error still occurs in this scenario due to the sensor measurement noise,

and therefore the larger time window T is, the more the prediction error will be cumulated, thereby degrading the performance. On the other hand, smaller T values may cause the algorithm less likely to find the feasible optimal solutions and apply the baseline pursuit strategy more often, and results in reduced performance.

Table 3.1.: Average Usage of the Baseline Pursuit

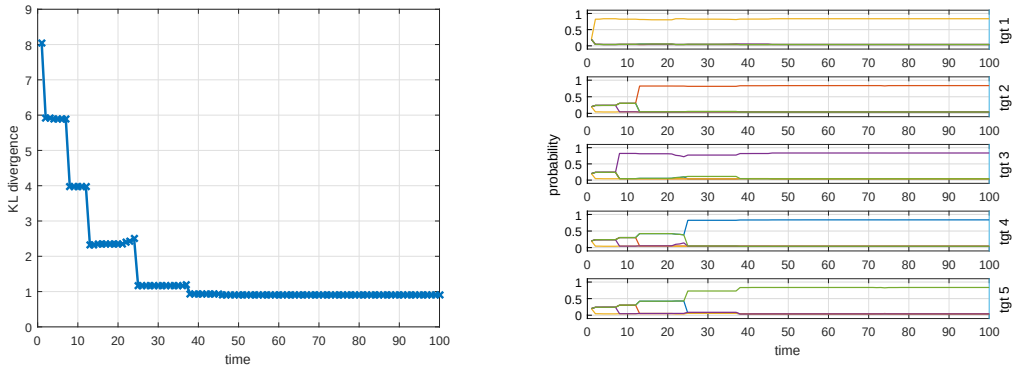
T	Baseline Pursuit Usage (%)
0	100
3	65
5	46
10	12
15	9

Table 3.1 shows the usage of the baseline pursuit strategy when different time windows T are used. The usage is defined as the number of time steps during the scenario that the UAV applies the control inputs generated from the baseline pursuit strategy. In the table, we present the usage as a percentage of the number of time steps out of the total time steps of the test scenario. In addition, we study the average percentage of 100 Monte-Carlo tests due to the random measurement noise we introduced in the simulation. The baseline pursuit usage of the algorithm with $T = 0$ is 100%, as the algorithm does not apply the optimization technique at all. Moreover, for smaller time windows T, larger percentages of baseline pursuit usage are resulted, validating our claim that the optimal solutions are less likely to be found in those cases.

Case 2: Partial Target Identification

This case demonstrates the robustness and sensitivity of the proposed algorithm w.r.t. imperfect local information. In this case, the secondary sensor identification is

assumed to be partial. Each time the secondary sensor identifies a target, it returns a belief p to the targets' true identity, i.e., the local information is in the form of $[(1-p)/N, ..., (1-p)/N, p, (1-p)/N, ..., (1-p)/N]^T$. The simulations are done with the worst case initial condition, i.e., the probabilities of the targets' identities are uniformly distributed from the start. The optimization window for the tests of the scenario is set to be $T = 10$s.



(a) KL divergence

(b) probabilities of the targets' identities

Fig. 3.12.: Performance Under Partial Target Identification

Figure 3.12 shows the performance of the algorithm with $p = 0.8$. The KL divergence converges in 5 optimization steps. Although the KL divergence does not converge to 0 (where all the targets' identities are certainly obtained), it converges to a minimum value such that each target has a probability of 0.8 to be its true identity, which is the best achievable result given the target identification condition.

The performance of the algorithm for different p values, measured in KL divergence, is shown in Figure 3.13. In all of the cases, the algorithm converges within no more than 5 optimization steps. In addition, the value to which the KL divergence converges depends on the value of p. A larger p value (i.e., more accurate sensor) leads to a smaller final KL divergence, which means the algorithm always performs better with more accurate target identifications.

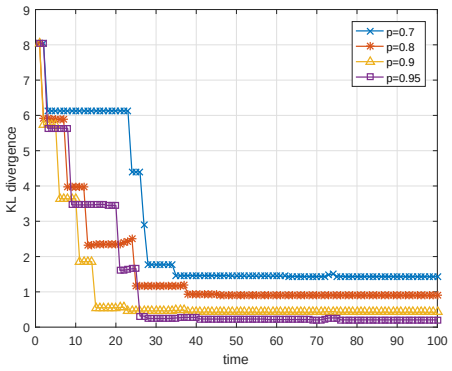


Fig. 3.13.: KL Divergence Given Different p Values

3.4.3 Single Secondary Sensor and Maneuvering Targets

In this scenario, 5 maneuvering targets are to be tracked, whose trajectories are shown in Figure 3.14. The targets move along randomly generated trajectories given by [27]:

$$\begin{aligned} x_j(t+1) &= x_j(t) + \Delta t \cdot v(t)\cos\theta(t) \\ y_j(t+1) &= y_j(t) + \Delta t \cdot v(t)\sin\theta(t) \end{aligned} \tag{3.20}$$

where $v(t)$ and $\theta(t)$ are the random speed and direction so that they describe a random motion pattern that cannot be perfectly predicted by the target propagation models. Δt is a time increment. The duration of the test scenario is 100s. For the scenario, we test and compare the results of the algorithm with and without the proposed feedback correction control input $u(t_s)^c$. The simulations are done with the worst case initial condition, i.e., the probabilities of the targets' identities are uniformly distributed at the start. The optimization window for the tests of the scenario is set to be $T = 10$s. The target identification is assumed to be perfect in this scenario.

We first present the trajectory of the tracking UAV in Figure 3.15 to illustrate the performance of the complete algorithm (i.e., with the proposed feedback control) in the test scenario. As can be observed, the proposed algorithm can track and identify

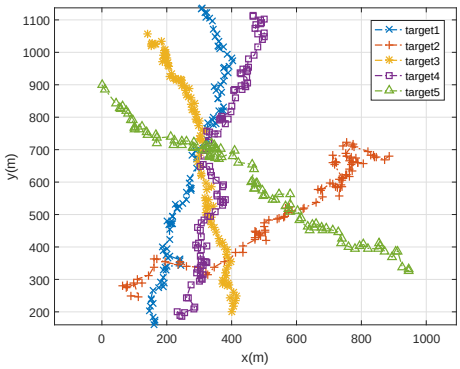
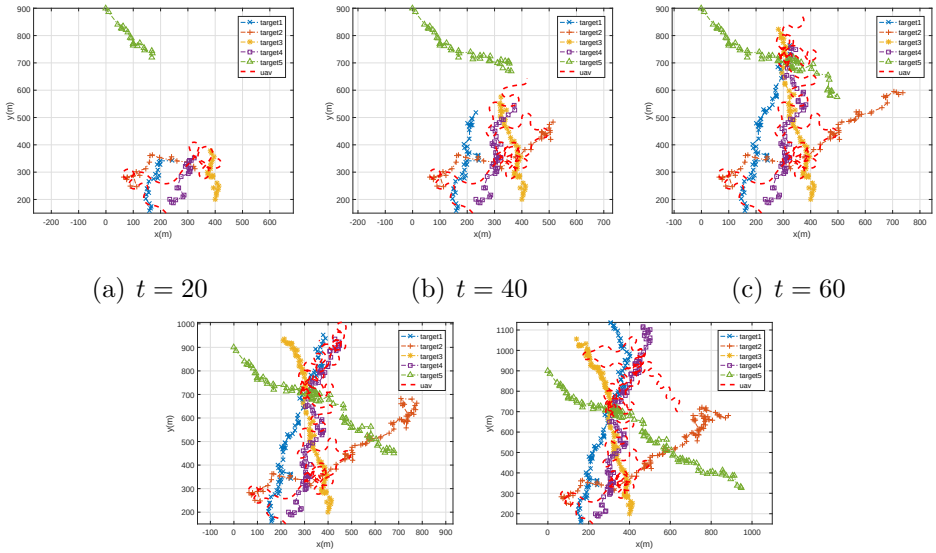


Fig. 3.14.: Target Trajectories of the Test Scenario

the targets accurately, especially when the targets heavily clutter together between time instances $t = 50$ and $t = 70$, in the case that the target motion cannot be accurately predicted.



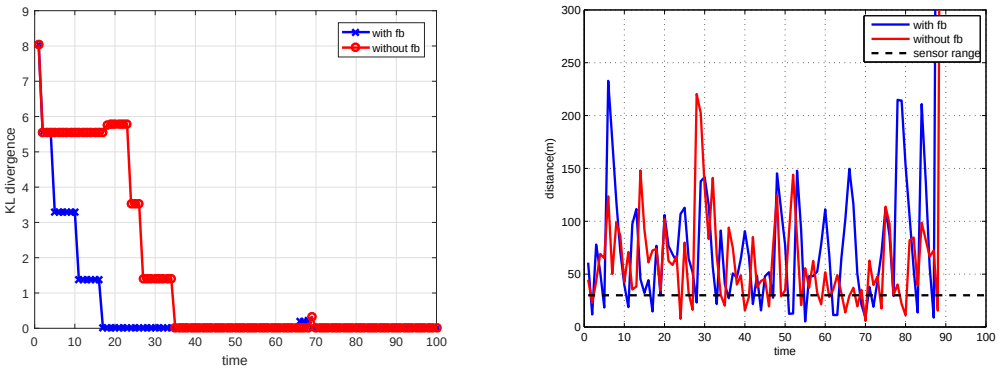
(a) $t = 20$ (b) $t = 40$ (c) $t = 60$

(d) $t = 80$ (e) $t = 100$

Fig. 3.15.: Trajectories of the Targets and the Tracking UAV (with the Feedback Control)

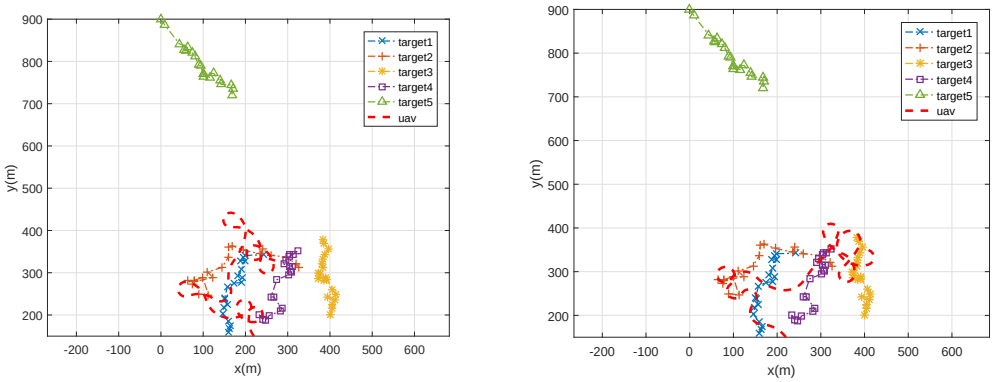
In Figure 3.16, we compare the performance of the algorithm with and without the proposed feedback correction. Figure 3.16(a) presents the KL divergence $D_{KL}(I||B)$ over the test duration. Although the algorithm converges in both cases with and without the feedback correction, it takes more time for the case without the feedback to converge. The reason behind is that in the case without the feedback, it occasionally happens that the target to be identified moves away from the predicted position when the tracking UAV arrives at that position, so that the target is in fact not identified. In the test scenario, the 'missed identification' situation happens around the time instance $t = 20$, where $D_{KL}(I||B)$ of the without feedback case even slightly increases due to the interaction of two targets. Figure 3.16(b) shows the distance between the tracking UAV and the target to be identified (which is not consistent as the algorithm outputs different targets to be identified from time to time). The black dashed line indicates the detection range of the sensor on the tracking UAV, i.e., a target is identified only if its distance to the tracking UAV is shorter than the detection range. Throughout the test duration, the peaks of the distance occur as a new target needs to be identified (which is usually far away from the tracking UAV's current position), and ideally the distance should decrease until it is smaller than the sensor detection range, as constrained by the optimization problem (3.12). However, in the case without feedback (shown by the red line), a few local minimum points of the distance are over the sensor range. This fact indicates that a target is not actually identified when an optimal control sequence is fully executed, especially around the time instance $t = 20$, which explains the non-converging behavior of $D_{KL}(I||B)$ in Figure 3.16(a).

A more illustrative explanation is given in Figure 3.17, where we show the targets and the UAV trajectories of both the with and without feedback cases up to the time instance $t = 20$. As can be observed, close to $t = 20$, the 2nd target (orange) starts to maneuver toward the 4th target (purple), which is not predicted accurately by the algorithm. In the case without the feedback correction (Figure 3.17(a)), the tracking UAV keeps moving according to the plan and misses the targets, and thus



(a) KL divergence (b) distance to the target-to-identify

Fig. 3.16.: Performance with/without the Feedback Correction



(a) without feedback (b) with feedback

Fig. 3.17.: Targets and UAV Trajectories up to $t = 20$

their identities. On the other hand, in the case with the feedback (Figure 3.17(b)), the tracking UAV changes its course accordingly and is able to identify the targets.

3.4.4 Multiple Secondary Sensor and Non-maneuvering Targets

As shown in Figure 3.18, in this scenario, the map is divided into 4 sectors, and each sector is assigned to a secondary UAV (i.e., each UAV stays in its sector and identifies targets within the sector). The 4 secondary sensors are used to track 7

moving targets. The time duration of the scenario is 100s. The identities of the targets are initially unknown so that the probabilities of the target identities are uniformly distributed at the beginning. The optimization window for the tests of the scenario is set to be $T = 10$s. The target identification is assumed to be perfect in this scenario.

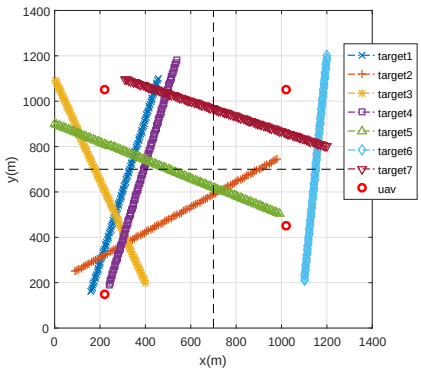
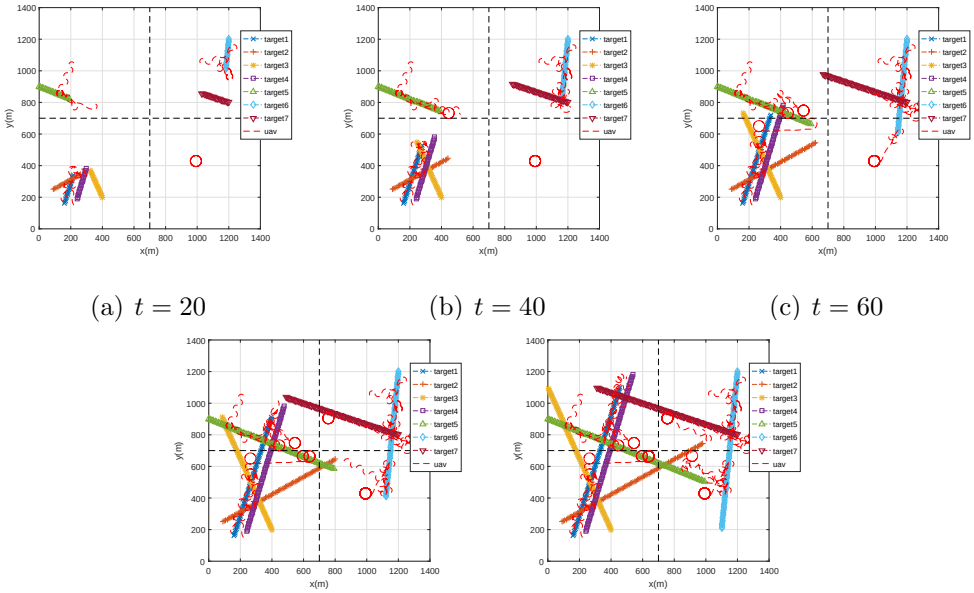


Fig. 3.18.: Target Trajectories of the Test Scenario

Figure 3.19 shows the trajectories of the targets and the secondary sensors (UAVs) at multiple time instances during the scenario. The trajectories of the tracking UAVs are given by the red dashed lines. Initially, there are no targets in the low-right sector, and the UAV assigned to the sector loiters until a target comes in (about $t = 60$). Whenever targets move close, for example, target 2 (the orange trajectory) and target 5 (the green trajectory) between the time instances $t = 60$ and $t = 80$, the tracking UAVs assigned to the sectors where the targets are commanded to move and identify them.

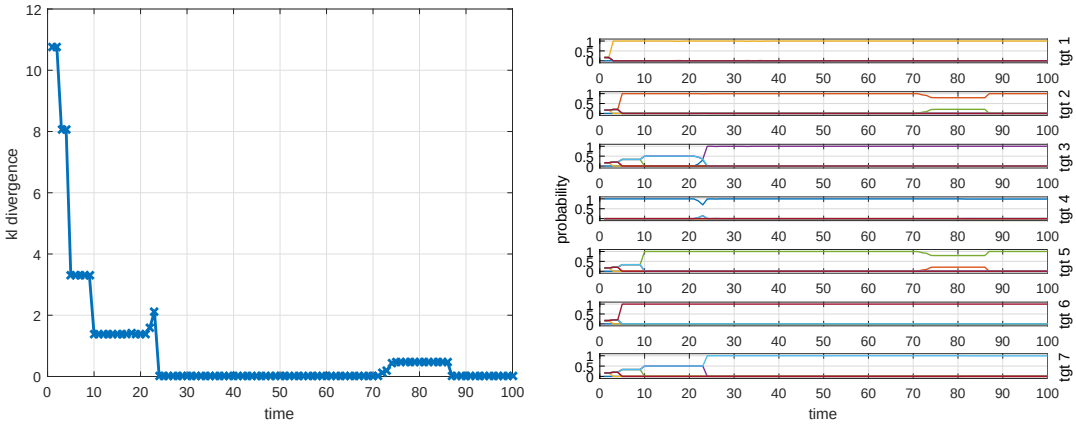
Figure 3.20(a) shows the KL divergence $D_{KL}(I||B)$ over time. Starting with no initial identity knowledge, all targets are correctly identified at $t = 25$ as the KL divergence converges to 0. The target identities become mixed again at $t = 70$ since some targets get close at the time; however, they are re-identified soon after and the KL divergence converges back to 0 again.



(a) $t = 20$ (b) $t = 40$ (c) $t = 60$

(d) $t = 80$ (e) $t = 100$

Fig. 3.19.: Trajectories of the Targets and the Tracking UAVs



(a) KL divergence (b) probabilities of the targets' identities

Fig. 3.20.: Algorithm Performance

Figure 3.20(b) presents the probabilities of each target having each of the identities along the test duration. One can observe a mixing of the probabilities of targets 2 and 5 between the time instances $t = 70$ and $t = 90$, corresponding to the increase of

the KL divergence in Figure 3.20(a), which happens as the two targets get close while cross the sector boarder. However, the identity belief is restored by the algorithm quickly as the secondary sensors identify the uncertain targets.

3.4.5 Multiple Secondary Sensor and Non-maneuvering Targets: Large Scale Application

The test scenario is shown in Figure 3.21, where 24 targets (trajectories are given by the black dash-dot lines) move at different yet constant velocities. Same to scenario 1, the map is divided into 4 sectors as shown by the black dashed lines, and each sector is assigned to a UAV. The initial positions of the UAVs are shown by the red circles. The time duration of the scenario is 100s. The identities of the targets are initially unknown so that the probability of the target identities are uniformly distributed at the beginning. The optimization window for the tests of the scenario is set to be $T = 10$s. The target identification is assumed to be perfect in this scenario.

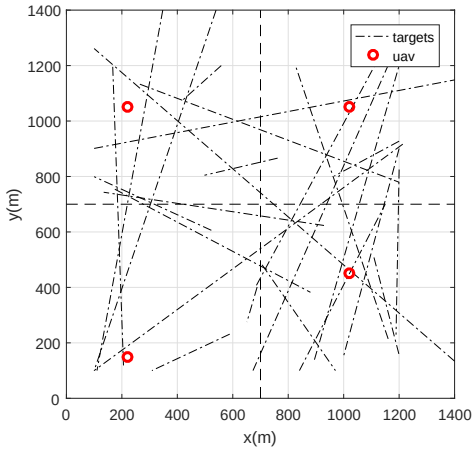


Fig. 3.21.: Target Trajectories of the Test Scenario

The test results are demonstrated in Figure 3.22, which shows the KL divergence between the identity belief matrix computed by our proposed algorithm and the true identity. As the tracking UAVs move to identify the targets, the KL divergence,

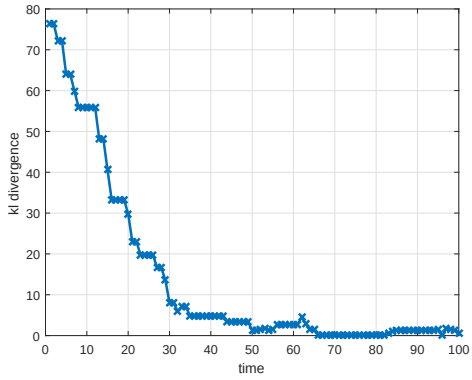
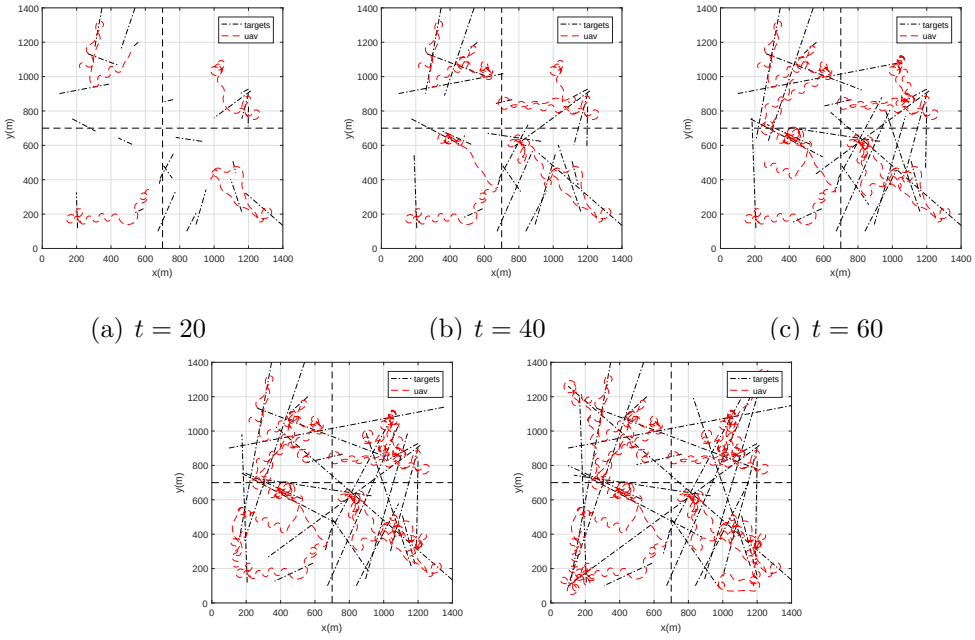


Fig. 3.22.: Quantified Identity Tracking Performance Measured in KL Divergence

starting from a high value at the beginning since no initial knowledge on the target identities is available, converges to 0 as time proceeds. Occasionally the target identities get mixed again after they have been already identified, however, the sensors are able to re-identify them. The results show that the proposed algorithm is able to effectively handle the scenarios with a large operation area and numerous targets, provided the targets do not jitter between the sector boundaries. Figure 3.23 shows the trajectories of the targets and the secondary sensors (UAVs) at multiple time instances during the scenario. The trajectories of the tracking UAVs are given by the red dashed lines.

3.4.6 Multiple Secondary Sensor and Maneuvering Targets

As shown in Figure 3.24, in this scenario, 4 secondary sensors are used to track 7 constantly-maneuvering targets. The target trajectories are generated through the same way in Section 3.4.3. The time duration of the scenario is 100s. The optimization window for the tests of the scenario is set to be $T = 10$s. Similar to the previous simulation settings, the identities of the targets are initially unknown, and the target identification is assumed to be perfect.



(a) $t = 20$ (b) $t = 40$ (c) $t = 60$

(d) $t = 80$ (e) $t = 100$

Fig. 3.23.: Trajectories of the Targets and the Tracking UAVs

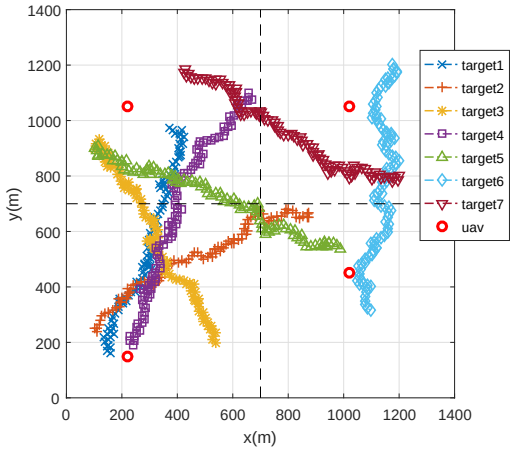
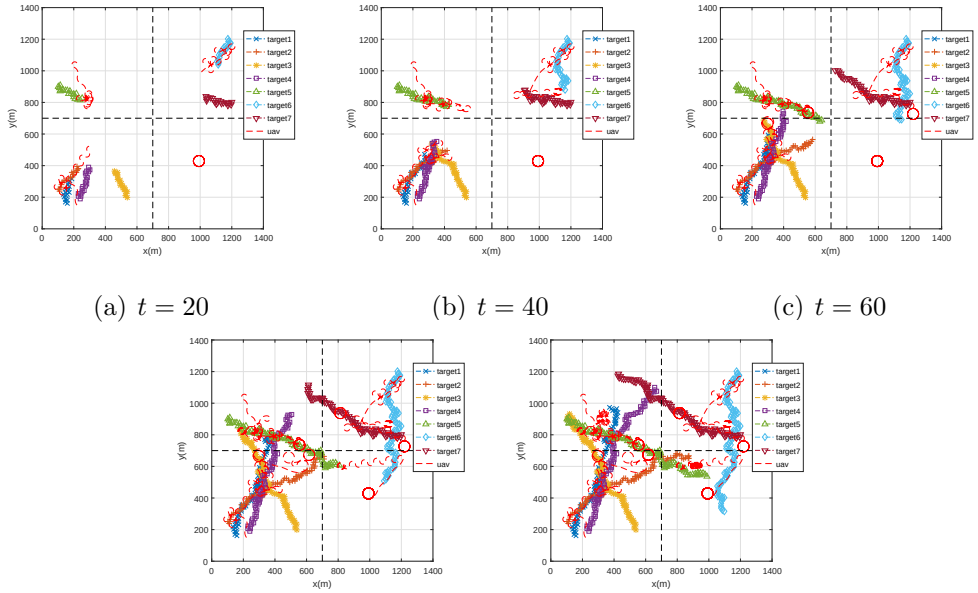


Fig. 3.24.: Target Trajectories of the Test Scenario

Figure 3.25 shows the trajectories of the targets and the secondary sensors (UAVs) at multiple time instances during the scenario. The trajectories of the tracking UAVs are given by the red dashed lines.



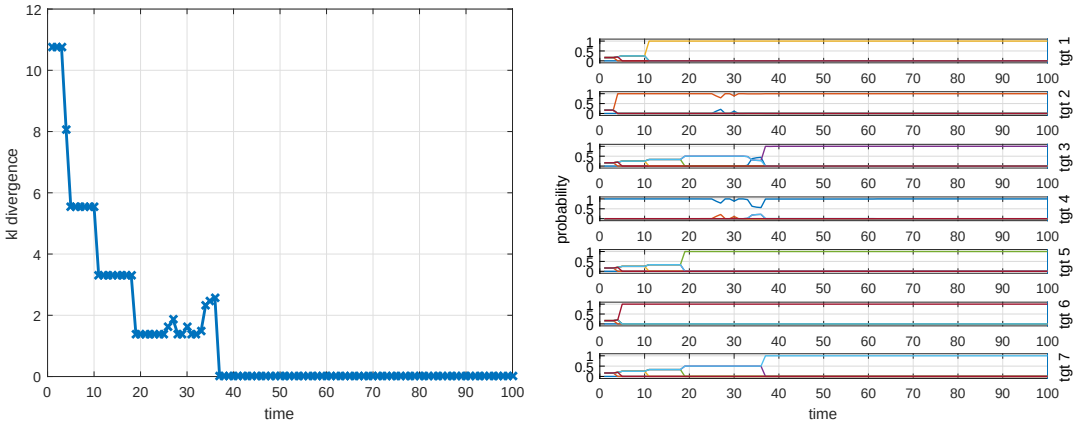
(a) $t = 20$ (b) $t = 40$ (c) $t = 60$

(d) $t = 80$ (e) $t = 100$

Fig. 3.25.: Trajectories of the Targets and the Tracking UAVs

Figure 3.26(a) shows the KL divergence $D_{KL}(I||B)$ over time. Starting with no initial identity knowledge, all targets are correctly identified at $t = 40$ as the KL divergence converges to 0. Figure 3.26(b) presents the probabilities of each target having each of the identities along the test duration. One can observe the interaction of the identity probabilities between targets 2, 3, and 4 around $t = 30$, which is recovered quickly after.

The simulation result shows that the multi-sensor version-ed algorithm is also able to handle the disturbance in the target state estimate/prediction. However, our test results indicate that the algorithm's performance still relies on a decent target state estimation model. For example, the algorithm no longer works properly if the constant velocity motion model is applied in a scenario where targets constantly perform turnings with varying turning rates, as the algorithm is unable to provide accurate target state estimate/prediction; in this case, one need to select a more appropriate target motion model according to [51].



(a) KL divergence

(b) probabilities of the targets' identities

Fig. 3.26.: Algorithm Performance

4. MULTI-TARGET IDENTITY MANAGEMENT OF UNKNOWN AND TIME-VARYING NUMBER OF TARGETS

4.1 Background

In practice, the number of targets is unknown and time-varying because targets may move into and/or out of the surveillance region of the tracking system (e.g., aircraft takeoff/landing), or the tracking system can lose or regain targets due to clutter or occlusion, etc., making the task of multi-target tracking and identity management even more challenging. Despite the substantial development in sensing and tracking techniques, there is no algorithm available for simultaneous target tracking and identity management for an unknown and time-varying number of targets; it is this chapter's emphasis to fill the gap in the area of study. In this chapter, we propose a new multi-target identity management algorithm, named GMPHD-IM algorithm, which can keep track of an unknown and time-varying number of targets and their identities simultaneously in a cluttered environment.

While joint probabilistic data association filters (JPDAF) have been traditionally used for scenarios where the number of targets is assumed to be known and fixed [1, 77, 99], recent research interests have been leaning to tracking an unknown number of targets, which leads to the probability hypothesis density (PHD) filter and its approximated implementation with linear-model assumptions, the Gaussian mixture probability hypothesis density (GMPHD) filter [12, 13]. In general, however, multi-target tracking algorithms do not explicitly keep track of the identities of different targets. More recent algorithms attempt to perform track association or labeled-target tracking [62, 100]; while these algorithms effectively associate the estimated target states with tracks/labels, they still do not explicitly tell how target

identities should be assigned to tracks or labels, and thus could give incorrect identity management results. For example, these algorithms could generate multiple tracks for one target, or assign the same label to multiple targets, especially in the case of track coalescence. Another missing piece in the implicit identity management performed by traditional multi-target tracking algorithms is the incorporation of target identity information (e.g., transmissions from an aircraft identity transponder, an observation from sensors such as camera, etc.), which may occasionally be available in real tracking scenarios, especially with the emerging application of (heterogeneous) sensor networks in multi-target tracking [74, 101].

To simultaneously keep tracking both targets' states and identities, the multi-target tracking and identity management (MTIM) algorithms have been proposed based on JPDAF and identity belief matrix [10, 69, 71]. The MTIM algorithms use an identity belief matrix to probabilistically represent the identities of all the targets. They use JPDAF to track multiple targets in clutter and calculate the target-measurement association information, which is used to propagate the identity belief matrix over time. The target identity information (i.e., local information) on some targets can also be incorporated whenever available to reduce uncertainties in the identities of all the targets within the sensor's surveillance range. Despite the research progress in tracking an unknown and time-varying number of targets with the GMPHD filter, the current MTIM algorithms are not capable of handling this situation for two reasons: first, the GMPHD filter does not explicitly calculate the target-measurement association and therefore cannot be directly applied to provide necessary information for identity management; and second, the current mathematical framework of the identity management is incomplete in that it is only able to process a known and fixed number of targets.

In this chapter, we propose a new multi-target identity management algorithm, called GMPHD-IM algorithm, which is able to simultaneously track and manage the identities of an unknown and time-varying number of targets. We introduce a generalized mathematical framework of identity management which can deal with

an unknown and time-varying number of targets. Within the proposed mathematical framework, the GMPHD-IM algorithm can actively manage the identities of the targets over time by maintaining the target identity belief (i.e., the probability distributions of the targets' identities) through a recursive propagation-update procedure. The target identity belief is propagated with the target-measurement association output from multi-target tracking algorithms. In this chapter, we use the modified-covariance GMPHD (MC-GMPHD) filter from [102] for better tracking performance, and develop a method to calculate the target-measurement association probabilities from the weights of the Gaussian mixture terms without introducing additional computational complexity. Whenever there is available target identity information on some targets, the algorithm updates the target identity belief with the information such that the uncertainties in the identities of all the targets are reduced. The uncertainties in the identities is calculated from the statistical entropy of the identity belief matrix. The target identity belief, together with the target state estimates generated by the MC-GMPHD filter, forms the output of the proposed algorithm. One major application of the proposed algorithm is to track and identify aircraft in air traffic control (ATC), or unmanned aircraft in UAS traffic management (UTM), which can help improve the safety while decreasing the human operator's workload.

The rest of this chapter is organized as follow. We first present the preliminary concepts and the overall architecture of the proposed GMPHD-IM algorithm in Section 4.2. Then, in Section 4.3, we give the mathematical formulation of the new identity management framework. The details of the GMPHD-IM algorithm components are presented in Section 4.4. The performance of the algorithm is demonstrated with illustrative numerical examples in Section 4.5.

4.2 Preliminaries and Algorithm Overview

In this section, we first present the preliminaries of the basic multi-target identity management concepts. Then, an overview of the proposed GMPHD-IM algorithm is given.

4.2.1 Identity Management Basics

An *identity belief vector* of a target is a column vector which represents the probability distribution that the target having a particular identity. Naturally a target has only one identity, therefore the sum of all elements in the identity belief vector should be 1. Suppose we consider the same identity sample space for all targets in the surveillance system, i.e., the identity belief vectors of all targets have the same dimension. Then, the identity belief vectors can be combined as one *identity belief matrix*. For example, let us consider two target estimates $\hat{x}_1(t)$ and $\hat{x}_2(t)$ at time t, and three identities 'A', 'B' and 'C'. Then, the identity belief matrix is in the form:

$$\begin{bmatrix} 0.8 & 0.1 \\ 0.2 & 0.7 \\ 0.0 & 0.2 \end{bmatrix}$$

where the first column is the identity belief vector of $\hat{x}_1(t)$, which is the probability distribution that $\hat{x}_1(t)$ having ID 'A', 'B' and 'C', receptively. The main task of multi-target identity management is to propagate the identity belief matrix over time and update it whenever there is available target identity information on some targets.

Propagation: The identity belief matrix is propagated using the association information between the current and previous target estimates. The association information is in the form of a *mixing matrix*, whose element represents the probability that a current target estimate is originated from a previous target estimate. For example, let us consider three target estimates $\hat{x}_1(t)$, $\hat{x}_2(t)$ and $\hat{x}_3(t)$ at time t, and two target

estimates $\hat{x}_1(t-1)$ and $\hat{x}_2(t-1)$ at time $t-1$. Then, the mixing matrix is in the form:

$$\begin{bmatrix} 0.8 & 0.1 & 0.01 \\ 0.1 & 0.8 & 0.01 \end{bmatrix}$$

In this example, the entry at $(1,1)$ means that the probability that $\hat{x}_1(t)$ is originated from $\hat{x}_1(t-1)$ is 0.8.

Update: The identity belief matrix should be updated with target identity information on some targets to reduce uncertainties in the identities of all the targets within the sensor's surveillance region, whenever such information is available. Target identity information usually comes from transmissions from an aircraft identity transponder, an observation from sensors such as camera, etc. Such target identity information on a target can be represented as an identity belief vector, i.e., a column vector which is defined as *local information*.

The formal mathematical definitions of the identity belief matrix and mixing matrix, along with the propagation and update equations, are presented in Section 4.3.

4.2.2 GMPHD-IM Algorithm Overview

In this chapter, we propose the GMPHD-IM algorithm to manage the identities of an unknown and time-varying number of targets as well as estimate their states. Figure 4.1 shows the multi-target identity management algorithm structure for each time step. As mentioned previously, the identity management recursively propagates and updates the identity belief matrix.

The identity belief matrix propagation requires the mixing matrix, which contains the target association information and is in general obtained from multi-target tracking and estimation algorithms. In this chapter, we use the MC-GMPHD filter from [102] for target state estimation, and augment it to calculate the mixing matrix during the estimation process. Figure 4.2 shows the steps of the MC-GMPHD filter and mixing matrix building. The MC-GMPHD filter applies the standard GM-

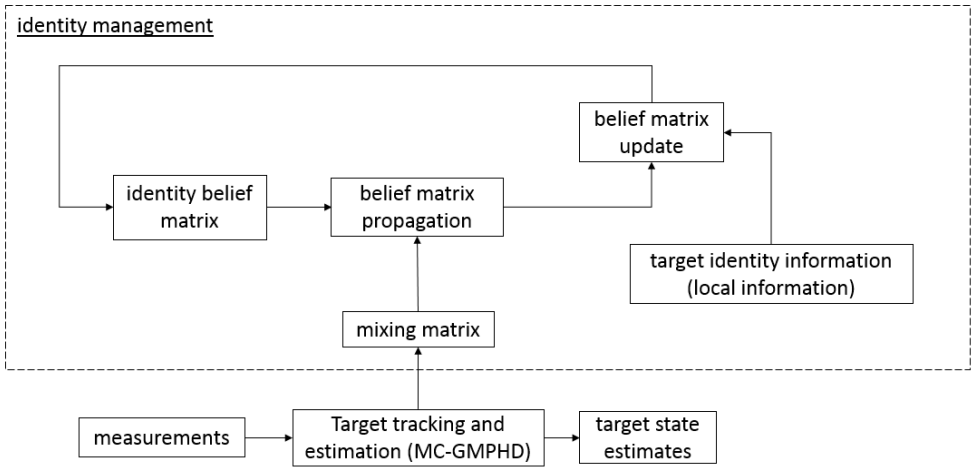


Fig. 4.1.: GMPHD-IM Algorithm Architecture

PHD prediction, pruning and extraction processes; however, for the measurement update, the MC-GMPHD uses a modified-covariance equation which explicitly considers the origin of the measurement (i.e., whether the measurement is a true target measurement or clutter). As the MC-GMPHD filter updates the target states with new measurements, we construct the mixing matrix using the information from the MC-GMPHD update process. The mixing matrix is then pruned as the MC-GMPHD performs its own pruning process.

After the identity belief matrix propagation, if there is available local information, we update the identity belief matrix such that the overall uncertainty level is reduced. The identity belief matrix, after update and/or local information incorporation, along with the target state estimates are the output of the proposed algorithm. The details of the GMPHD-IM algorithm are presented in Section 4.4

4.3 Mathematical Framework of Identity Management

In this section, we propose a mathematical framework of managing the identities of an unknown and time-varying number of targets. Consider the following problem. At time t, there are N_t targets whose states are estimated by the target tracking system.

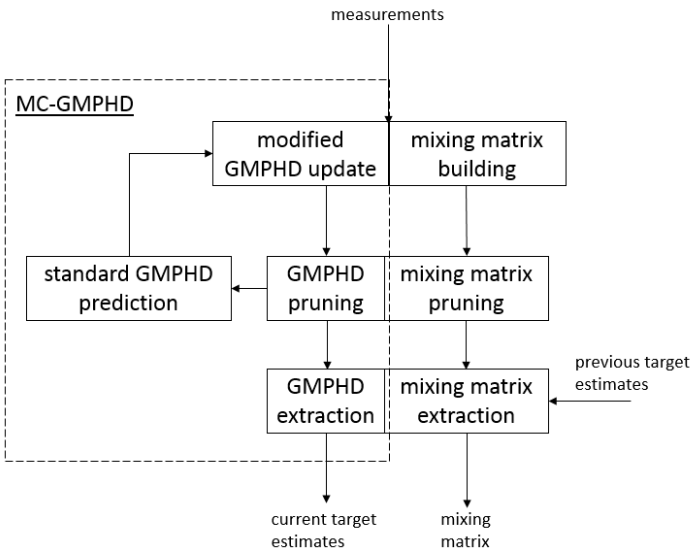


Fig. 4.2.: MC-GMPHD and Mixing Matrix Building

Meanwhile, there are L_t possible identities for the N_t targets considered by the identity management system. For each target whose identity is unclear, the possible identity shall theoretically have infinitely many candidates; however, it is only tractable if the identity management system considers a finite number of more-likely candidates and ignore the less-likely ones. The number of identity candidates considered for one target is generally more than one, and therefore we have $L_t \geq N_t$. The goal of multi-target identity management is to maintain and update the target identity information over time and incorporate local information when available. It should be noted that for the scenario in which $L_t = N_t = N$ is constant, the framework proposed in this section becomes identical to the current identity management algorithm [68], making the latter a special case.

4.3.1 Identity Management Formulation

At time t, consider the set of target state estimates $\hat{\mathcal{X}}(t) = \{\hat{x}_j(t)|j = 1, 2, ..., N_t\}$, and the set of possible enumerated target identities $\mathcal{I}(t) = \{1, 2, ..., L_t\}$. The core

idea of the identity management is to calculate the possibility that a target whose state estimate is $\hat{x}_j(t) \in \hat{\mathcal{X}}(t)$ has identity $i \in \mathcal{I}(t)$.

To formulate the multi-target identity management problem, we start by defining the identity belief matrix $B(t)$.

Definition 4.3.1 *The identity belief matrix $B(t)$, whose entry $b_{ij}(t)$ represents the probability that $\hat{x}_j(t) \in \hat{\mathcal{X}}(t)$ has identity $i \in \mathcal{I}(t)$, is an $L_t \times N_t$ non-negative matrix satisfying the following properties:*

1. $\sum_{i=1}^{L_t} b_{ij}(t) = 1$, *i.e., column stochastic.*
2. $\sum_{j=1}^{N_t} b_{ij}(t) \leq 1$, *i.e., the row sums are no greater than one.*

The j-th column $\boldsymbol{b_j}(t)$ of $B(t)$ is called the identity belief vector of $\hat{x}_j(t)$.

$$B(t) = [\boldsymbol{b_1}(t), \boldsymbol{b_2}(t), ..., \boldsymbol{b_{N_t}}(t)] \in [0,1]^{L_t \times N_t}$$

where

$$\boldsymbol{b_j}(t) = \begin{bmatrix} p(\hat{x}_j(t) \text{'s ID is } 1) \\ p(\hat{x}_j(t) \text{'s ID is } 2) \\ \vdots \\ p(\hat{x}_j(t) \text{'s ID is } L_t) \end{bmatrix} \in [0,1]^{L_t \times 1}$$

Indeed, if we consider $\hat{x}_j(t)$'s ID as a random variable, its sample space is infinitely large. The column stochastic property in Definition 4.3.1 represents the idea that only a finite number of identities are being considered and all other less-likely candidates are ignored. The row sums, on the other hand, are generally no greater than one since each target has only one identity and $L_t \geq N_t$. Note that in the case where $L_t = N_t$, the row sums are also constrained to one, and thus $B(t)$ becomes doubly-stochastic. We denote the 'residue' vector of the row sums of $B(t)$ as $\bar{\boldsymbol{b}}(t)$:

$$\bar{\boldsymbol{b}}(t) = [\bar{b}_1(t), \bar{b}_2(t), ..., \bar{b}_{L_t}(t)]^T$$

where $\bar{b}_i(t) = 1 - \sum_{j=1}^{N_t} b_{ij}(t)$ represents the probability that $\hat{x}_j(t)$'s ID is **not** i, $\forall \hat{x}_j(t)$. Next, we define the mixing matrix $M(t)$.

Definition 4.3.2 *The mixing matrix $M(t)$ is an $N_{t-1} \times N_t$ non-negative matrix whose entry $m_{ij}(t)$ represents the probability of $\hat{x}_j(t)$ being originated from $\hat{x}_i(t-1)$. The matrix $M(t)$ satisfies the following properties:*

1. $\sum_{i=1}^{N_{t-1}} m_{ij}(t) \leq 1.$
2. $\sum_{j=1}^{N_t} m_{ij}(t) \leq 1.$

i.e., both row and column sums of $M(t)$ do not exceed one.

Conceptually, $M(t)$ is a matrix of association probabilities of all target estimates between two adjacent time instances. It should also be noted that in general $M(t)$ can be calculated from measurement association during multi-target tracking [10]. The constraint that the row sums of $M(t)$ do not exceed one essentially means that each target at time $t-1$ has a possibility of being disappeared from the target tracking system at time t. The column sums of $M(t)$ are also required not to be greater than one because it represents the possibility that targets at time t could be new targets that are not originated from any targets at time $t-1$. Note that when $N_{t-1} = N_t = N$ is constant over time, the matrix $M(t)$ also becomes doubly-stochastic. We denote the 'residue' vector of the column sums of $M(t)$ as $\bar{\boldsymbol{m}}(t)$:

$$\bar{\boldsymbol{m}}(t) = [\bar{m}_1(t), \bar{m}_2(t), ..., \bar{m}_{N_t}(t)]^T$$

where $\bar{m}_j(t) = 1 - \sum_{i=1}^{N_{t-1}} m_{ij}(t)$ represents the probability that $\hat{x}_j(t)$ is **not** originated from $\hat{x}_i(t-1)$ $\forall \hat{x}_i(t-1)$, i.e. $\hat{x}_j(t)$ represents a new target.

The main body of the multi-target identity management is to recursively update the targets' identity information. When new target estimates come at time $t+1$, the first step is to associate these new estimates with the prior identity set $\mathcal{I}(t)$. By the law of total probability, $\forall \hat{x}_j(t+1) \in \hat{\mathcal{X}}(t+1), \hat{x}_k(t) \in \hat{\mathcal{X}}(t)$ and $i \in \mathcal{I}(t)$, we have

$$\begin{aligned} &p(\hat{x}_j(t+1)\text{'s ID is } i) = \\ &\sum_{k=1}^{N_t} p(\hat{x}_j(t+1) \text{ is originated from } \hat{x}_k(t)) \cdot p(\hat{x}_k(t)\text{'s ID is } i) + \\ &p(\hat{x}_j(t+1) \text{ is a new target}) \cdot p(\text{the new target's ID is } i) \end{aligned} \tag{4.1}$$

On the left hand side of (4.1), $p(\hat{x}_j(t+1)\text{'s ID is } i)$ is by definition $b_{ij}(t+1)$. On the right hand side, the first term can be expressed as $\sum_{k=1}^{N_t} b_{ik}(t) m_{kj}(t+1)$. In addition, the term $p(\hat{x}_j(t+1)$ is a new target) is by definition $\bar{m}_j(t+1)$. The term p(the new target's ID is i), on the other hand, cannot be readily expressed. However, we do know that the probability that a new target at time $t+1$ whose ID is i is related to the probability that no current target at time t has ID i, which, by definition, is $\bar{b}_i(t)$. Therefore, we model $p(\hat{x}_j(t+1)$ is a new target) as $\lambda_M \bar{b}_i(t)$, where λ_M is a modeling parameter which can be viewed as a 'rate of occurrence' of the event that a new target has a specific ID. Then, the propagation equation (4.1) can be written as

$$b_{ij}(t+1) = \sum_{k=1}^{N_t} b_{ik}(t) m_{kj}(t+1) + \lambda_M \bar{b}_i(t) \bar{m}_j(t+1) \tag{4.2}$$

and we have

$$B(t+1) = B(t)M(t+1) + \lambda_M \bar{\boldsymbol{b}}(t) \bar{\boldsymbol{m}}(t+1)^T \tag{4.3}$$

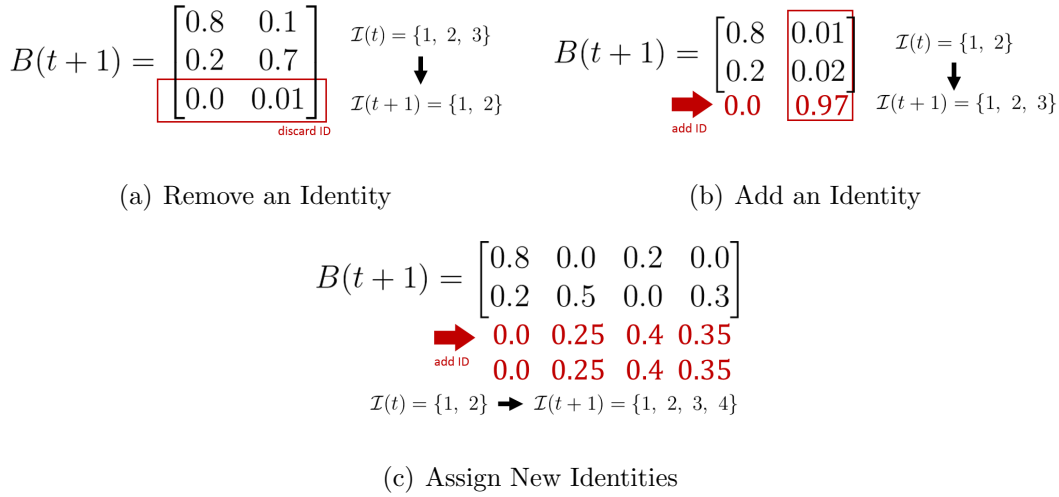
In the case of the number of targets being constant, one can set the modeling parameter $\lambda_M = 0$ and the propagation equation (4.3) becomes identical to the propagation equation used by the current identity management algorithm [68].

The belief matrix $B(t+1)$ by now only has target identity information of the prior identity set $\mathcal{I}(t)$. Next, we need to update the identity set as well as the

belief matrix correspondingly. In this section, we introduce the general idea of this updating process. A detailed description of the algorithm implementation is presented in Section 4.4.

The identity set and the identity belief matrix are updated in a way of removing IDs that are unlikely to appear in the current targets and adding/assigning new IDs to the current targets. The update of the belief matrix and the identity set takes a three-step procedure. First, $\forall i \in \mathcal{I}(t)$, we check the i-th row sum $\bar{b}_i(t+1)$ of the belief matrix $B(t+1)$. If the row is sufficiently small (i.e., less than a threshold), it means that the probability that $\hat{x}_j(t+1)$'s ID is i is negligible $\forall \hat{x}_j(t+1) \in \hat{\mathcal{X}}(t+1)$, i.e., ID i is unlikely to appear. Therefore, we remove ID i from the prior identity set $\mathcal{I}(t)$, and correspondingly remove the i-th row in $B(t+1)$. Next, $\forall \hat{x}_j(t+1) \in \hat{\mathcal{X}}(t+1)$, we check the j-th column sum of $B(t+1)$. If the column sum is sufficiently small (i.e., less than a threshold), it means that the probability that $\hat{x}_j(t+1)$'s ID is in the prior identity set $\mathcal{I}(t)$ is negligible. Therefore, we expand the identity set with a new ID assigned to $\hat{x}_j(t+1)$ and correspondingly, a new row is appended to $B(t+1)$. After the above two steps, we check the latest identity belief matrix to see whether the number of rows is smaller than that of columns. Since each target must have one identity, this case means that there are new targets whose IDs are not assigned in the system, but we do not know which of the targets are the new ones. Therefore, we expand the identity set with new IDs and assign the probability of these new IDs to all $\hat{x}_j(t+1) \in \hat{\mathcal{X}}(t+1)$. Correspondingly, new rows will be appended to the identity belief matrix so that the number of rows and columns are even. Figure 4.3 shows illustrative examples of updating the identity set and the identity belief matrix by removing, adding and assigning identities.

The identity belief matrix $B(t+1)$ now contains the target identity information of the updated identity set $\mathcal{I}(t+1)$. However, after the above updating process, there is no guarantee that the new $B(t+1)$ satisfies the row and column sum constraints in Definition 4.3.1, especially the column-stochastic constraint. The statistical reason behind is that the probability that an ID does not belong to any current targets is not



(a) Remove an Identity

(b) Add an Identity

(c) Assign New Identities

Fig. 4.3.: Examples of Updating Identity Belief Matrix and Identity Set

strictly 1 as we remove the ID and the corresponding matrix row during the update. Consequently, the sample space of $\hat{x}_j(t+1)$'s ID is not complete, and the column sums of the identity belief matrix are usually less than 1. However, in the proposed identity management framework, we are ignoring this sufficiently small probability and considering the current target identity set $\mathcal{I}(t+1)$ as a complete sample space (which is being updated over time). Thus, it is both statistically necessary and reasonable to rescale the matrix $B(t+1)$ to meet the row and column sum constraints. In this chapter, we develop a matrix rescaling approach that rescales any non-negative matrix to a matrix satisfying the constraints in Definition 4.3.1. The details of the rescaling approach is described in later sections.

4.3.2 Local Information Incorporation

In practice, the local information may come from different sources, such as manual actions from a human operator, transmissions from the targets, observations from sensors [103], or calculation from the target dynamic behaviors [69], and it can be incorporated to update the identity belief matrix.

In the multi-target identity management framework, the local information is expressed as an identity belief vector whose entry represents the marginal probability of the target's identities. Consider an example of two targets and two identities 'A' and 'B' at time t. Then, the set of target state estimates is $\hat{\mathcal{X}}(t) = \{\hat{x}_j(t)|j = 1, 2\}$, and the identity set $\mathcal{I}(t) = \{1,\ 2\}$ where 1 and 2 are the enumerated identities of 'A' and 'B', respectively. Suppose the identity belief matrix at time t is

$$B(t) = \begin{bmatrix} 0.6 & 0.4 \\ 0.4 & 0.6 \end{bmatrix}$$

Note that $B(t)$ is doubly-stochastic in this example since it is square. The local information for target 2 (corresponding to the 2nd column of $B(t)$) may arrive in the form

$$\boldsymbol{l}_2(t) = \begin{bmatrix} 0.1 \\ 0.9 \end{bmatrix}$$

which means target 2 has ID 'B' with a probability of 0.9. Theoretically, this information can be used to update $B(t)$ through Bayesian normalization. In practice, however, the normalization approach is infeasible due to its exponential complexity [68]; and as an alternative, it is proposed to replace the corresponding column in the belief matrix with the local information and then rescale the matrix to meet the row/column sum requirements [68]. It should be noted that this process can be viewed as an approximation of the Bayesian update. In the example above, the updated and rescaled belief matrix is

$$B(t) = \begin{bmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{bmatrix}$$

The local information may not necessarily contain only the information for the identities in the current target identity set, but also contain the information for the identities that are not included at present. Following the example above, the local

information for target 2 (corresponding to the 2nd column of $B(t)$) may also arrive in the form

$$\boldsymbol{l}_2(t) = \begin{bmatrix} 0.0 \\ 0.8 \\ 0.2 \end{bmatrix}$$

which means target 2 has a new ID (neither 'A' nor 'B') with a probability of 0.2. To incorporate this information, the target identity set needs to be first expanded. Then, the column of $B(t)$ is replaced by the local information and becomes

$$\begin{bmatrix} 0.6 & 0.0 \\ 0.4 & 0.8 \\ 0.0 & 0.2 \end{bmatrix}$$

In general, it is not trivial to rescale an arbitrary non-negative matrix (e.g., the one above). The rescaling approach we develop in this chapter is also applied to the local information incorporation.

4.3.3 Rescaling of Identity Belief Matrix

The matrix rescaling problem has been much studied by mathematicians in the past decades [104–106]. The most common case in the matrix rescaling problem, known as Sinkhorn rescaling [76] which aims to rescale a non-negative square matrix to a doubly-stochastic one, is widely seen in various statistical applications including the current identity management algorithms. In this chapter, however, a more general problem occurs which is to rescale a non-negative rectangular matrix to a column stochastic one with inequality row sum constraints. To make the chapter concise and easy-to-follow, in this section we only present the conclusions and our approach to solve the matrix scaling problem. The detailed mathematical justifications are included in Section 4.6.1.

In general, rescaling an arbitrary $L \times N$ non-negative matrix B is to find a matrix $B' = \delta D_1 B D_2$, where δ is a positive scalar, D_1 is $L \times L$ and D_2 is $N \times N$, both are diagonal matrices, and B' satisfies the prescribed row and column sum constraints. One measure to describe how 'statistically close' the two matrices B' and B are is the Kullback-Leibler (KL) distance [98]:

$$D_{KL}(B'||B) = \sum_{j=1}^{N} \sum_{i=1}^{L} b'_{ij} \log \frac{b'_{ij}}{b_{ij}} \tag{4.4}$$

The following convention is used throughout this chapter: $0 \log 0 = 0$, $0 \log \frac{0}{0} = 0$, and $b \log \frac{b}{0} = \infty$ if $b > 0$ [69]. We show that for an arbitrary identity belief matrix $B(t)$ whose dimension is $L_t \times N_t$, finding the rescaling matrix which satisfies the constraints in Definition 4.3.1 is the same as solving a convex optimization problem of minimizing the KL distance:

Theorem 4.3.1 *The solution to the optimization problem (4.5), if exists, rescales the $L_t \times N_t$ matrix $B(t)$ to a column-stochastic matrix.*

$$\begin{aligned}
&\textit{find} \quad b'_{ij}(t) \\
&\textit{min} \quad \sum_{j=1}^{N_t} \sum_{i=1}^{L_t} b'_{ij}(t) \log \frac{b'_{ij}(t)}{b_{ij}(t)} \\
&\textit{s.t.} \quad \sum_{j=1}^{N_t} b'_{ij}(t) \leq 1 \\
&\qquad \sum_{i=1}^{L_t} b'_{ij}(t) = 1 \\
&\qquad b'_{ij}(t) \geq 0 \ \textit{if}\ b_{ij}(t) > 0 \\
&\qquad b'_{ij}(t) = 0 \ \textit{if}\ b_{ij}(t) = 0 \\
&\qquad \forall i \in \mathcal{I}(t), \forall j \in \{1, 2, ..., N_t\}
\end{aligned} \tag{4.5}$$

The formal definition of 'rescaling a matrix', along with a mathematically rigorous expression of Theorem 4.3.1 and its proof, are given in Appendix 4.6.1. Note that the problem (4.5) is convex and can be solved efficiently by optimization approaches

such as interior-point method [107]. Particularly, for the special case where $L_t = N_t$, it is easy to show that the row sum inequality constraints in problem (4.5) converge to the equality constraints. The converged optimization problem, which is considered as the belief matrix rescaling problem in the current identity management algorithm, is proved to be the same as Sinkhorn rescaling [72].

4.4 GMPHD-IM Algorithm

A detailed explanation of the GMPHD-IM algorithm is presented in this section. As shown in Figure 4.1, the multi-target identity management algorithm comprises two stages: propagation and update. The propagation of the identity belief matrix requires the mixing matrix which is computed from the multi-target tracking and estimation. In this chapter, we use the MC-GMPHD filter [102] to perform the multi-target tracking and estimation. The MC-GMPHD filter is a GMPHD variant with a modified covariance update equation, and it generates N_t current target state estimates at time t. We augment the MC-GMPHD filter so that it simultaneously computes an $N_{t-1} \times N_t$ mixing matrix using the state estimates of N_{t-1} targets from the previous time step and K_t cluttered measurements from the current time step. The update stage checks whether the system needs to remove existing IDs or assign new IDs, and updates the identity belief matrix accordingly. In addition, the identity belief matrix is also updated whenever there is local identity information available. Eventually, the GMPHD-IM algorithm outputs the identity belief matrix as well as the current target estimates.

4.4.1 Identity Belief Matrix Propagation

At time t, given the previous set of target state estimates $\hat{\mathcal{X}}(t-1) = \{\hat{x}_j(t-1)|j = 1, 2, ..., N_{t-1}\}$ and the current set of measurements $\mathcal{Z}(t) = \{z_k(t)|k = 1, 2, ..., K_t\}$, we use the MC-GMPHD filter to provide the current set of target state estimates $\hat{\mathcal{X}}(t) = \{\hat{x}_j(t)|j = 1, 2, ..., N_t\}$. In the mean time, we also develop a method to

compute the mixing matrix $M(t)$ within the MC-GMPHD framework. While the MC-GMPHD filter recursively updates a Gaussian mixture (a weighted sum of Gaussian components), from which the target state estimates can be extracted, the mixing matrix is constructed meanwhile using the weights of the Gaussian mixture.

Over time, the MC-GMPHD filter updates itself with new measurements, and performs pruning and extraction to select a number of Gaussian distributions from the Gaussian mixture [13], each of which represents the distribution of a target state. As the MC-GMPHD filter performs the pruning and extraction processes, similar actions are taken by the mixing matrix as well. After the processes, the mixing matrix is normalized by rescaling. Eventually, for each target state estimate computed by the MC-GMPHD filter, its association probabilities with all the previous target state estimates are also given in the form of mixing matrix $M(t)$, which is passed to the identity management propagation.

Original GMPHD Filter

The GMPHD filter is theoretically based on the finite set statistics (FISST) [12, 108]. Provided the random finite set (RFS) of the multi-target state $\mathcal{X}(t)$ and the RFS of the measurement $\mathcal{Z}(t)$, FISST gives the recursive Bayesian filtering of the multi-target prior density $f(\mathcal{X}(t)|\mathcal{Z}_{1:t-1})$ and posterior density $f(\mathcal{X}(t)|\mathcal{Z}_{1:t})$ as:

$$\begin{aligned} f(\mathcal{X}(t)|\mathcal{Z}_{1:t-1}) &= \int f(\mathcal{X}(t)|\mathcal{X}(t-1)) f(\mathcal{X}(t-1)|\mathcal{Z}_{1:t-1}) \delta\mathcal{X}(t-1) \\ f(\mathcal{X}(t)|\mathcal{Z}_{1:t}) &= \frac{f(\mathcal{Z}(t)|\mathcal{X}(t)) f(\mathcal{X}(t)|\mathcal{Z}_{1:t-1})}{\int f(\mathcal{Z}(t)|\mathcal{X}(t)) f(\mathcal{X}(t)|\mathcal{Z}_{1:t-1}) \delta\mathcal{X}(t)} \end{aligned} \tag{4.6}$$

where $\mathcal{Z}_{1:t} = \{\mathcal{Z}(1), \mathcal{Z}(2), ..., \mathcal{Z}(t)\}$ is the sequence of the measurement RFS history and $f(\mathcal{X}(t)|\mathcal{X}(t-1))$ is the state transition density. To make the problem computationally tractable, the GMPHD filter propagates and updates the posterior intensity, which is the first order statistical moment of the posterior multi-target state.

Given the previous RFS of multi-target state $\mathcal{X}(t-1)$, the current-time multi-target state RFS $\mathcal{X}(t)$ is given by the union of surviving targets $S(\mathcal{X}(t-1))$, spawned targets $B(\mathcal{X}(t-1))$, and spontaneous births $\Gamma(t)$:

$$\mathcal{X}(t) = S(\mathcal{X}(t-1)) \cup B(\mathcal{X}(t-1)) \cup \Gamma(t) \tag{4.7}$$

Assume that $\mathcal{X}(t)$ is a Poisson process, i.e., the cardinality of $\mathcal{X}(t)$ is subject to a Poisson distribution and the elements $x(t) \in \mathcal{X}(t)$ are independent and identically distributed, the prior intensity $I(x(t)|\mathcal{Z}_{1:t-1})$ and the posterior intensity $I(x(t)|\mathcal{Z}_{1:t})$ can be recursively calculated as follow [109]:

$$\begin{aligned} I(x(t)|\mathcal{Z}_{1:t-1}) &= \gamma(x(t)) + \int p_S f(x(t)|x(t-1))dx(t-1) \\ &\quad + \int \beta(x(t)|x(t-1))I(x(t-1)|\mathcal{Z}_{1:t-1})dx(t-1) \\ I(x(t)|\mathcal{Z}_{1:t}) &= (1-p_D)I(x(t)|\mathcal{Z}_{1:t-1}) \\ &\quad + \sum_{z_k(t)\in\mathcal{Z}(t)} \frac{p_D p(z_k(t)|x(t))I(x(t)|\mathcal{Z}_{1:t-1})}{\lambda c(z_k(t)) + \int p_D p(z_k(t)|\eta(t))I(\eta(t)|\mathcal{Z}_{1:t-1})d\eta(t)} \end{aligned} \tag{4.8}$$

where p_D is the probability of detection, p_S is the probability that a target still exists, $\beta(x(t)|x(t-1))$ and $\gamma(x(t))$ represent the intensities of target spawn and birth, respectively, and $\lambda c(z_k(t))$ is the intensity of measurement clutter, described by the Poisson distribution. The GMPHD filter assumes linear Gaussian models for both target dynamics and measurement, i.e.,

$$f(x(t)|x(t-1)) \sim \mathcal{N}(x(t); F_{t-1}x(t-1), Q_{t-1})$$
$$p(z_k(t)|x(t)) \sim \mathcal{N}(z_k(t); H_t x(t), R_t)$$

where $\mathcal{N}(\cdot; m, P)$ denotes a normal distribution with mean m and covariance matrix P. F_{t-1} is the state transition matrix; Q_{t-1} is the process noise covariance; H_t is the measurement matrix and R_t is the measurement noise covariance. In addition, the target spawn and birth intensities are assumed in the form of Gaussian mixture:

$$\begin{aligned}\gamma(x(t)) &= \sum_{i=1}^{J_{\gamma,t}} \omega_{\gamma,t}^{(i)} \mathcal{N}(x(t); m_{\gamma,t}^{(i)}, P_{\gamma,t}^{(i)}) \\ \beta(x(t)|x(t-1)) &= \sum_{j=1}^{J_{\beta,t}} \omega_{\beta,t}^{(j)} \mathcal{N}(x(t); F_{\beta,t-1}^{(j)} x(t-1) + d_{\beta,t-1}^{(j)}, Q_{\beta,t-1}^{(j)})\end{aligned} \tag{4.9}$$

where $J_{\gamma,t},\ \omega_{\gamma,t}^{(i)}, m_{\gamma,t}^{(i)}$ and $P_{\gamma,t}^{(i)}$ are given model parameters that determine the shape of the birth intensity; similarly, $J_{\beta,t},\ \omega_{\beta,t}^{(j)},\ F_{\beta,t-1}^{(j)},\ d_{\beta,t-1}^{(j)}$ and $Q_{\beta,t-1}^{(j)}$ are parameters determining the shape of the spawn intensity of a target given its previous state $x(t-1)$. It is shown [13] that with the above assumptions, the prior and posterior intensities in (4.8) can also be expressed in the form of Gaussian mixture. Let the posterior intensity at time $t-1$ be:

$$I(x(t-1)|\mathcal{Z}_{1:t-1}) = \sum_{i=1}^{J_{t-1}} \omega_{t-1}^{(i)} \mathcal{N}(x(t-1); m_i(t-1), P_i(t-1))$$

which has J_{t-1} Gaussian components. Then, the prior intensity $I(x(t)|\mathcal{Z}_{1:t-1})$ is given by:

$$I(x(t)|\mathcal{Z}_{1:t-1}) = \gamma(x(t)) + I_{S,t|t-1}(x(t)) + I_{\beta,t|t-1}(x(t)) \tag{4.10}$$

where $I_{S,t|t-1}(x(t)),\ I_{\beta,t|t-1}(x(t))$ and $\gamma(x(t))$ represent the intensities of target survival, spawn and birth, respectively, all of which can be written in the form of Gaussian mixture:

$$I_{S,t|t-1}(x(t)) = p_S \sum_{i=1}^{J_{t-1}} \omega_{t-1}^{(i)} \mathcal{N}(x(t); F_{t-1} m_i(t-1), Q_{t-1} + F_{t-1} P_i(t-1) F_{t-1}^T) \tag{4.11}$$

$$\begin{aligned}I_{\beta,t|t-1}(x(t)) &= \sum_{i=1}^{J_{t-1}} \sum_{j=1}^{J_{\beta,t}} \omega_{t-1}^{(i)} \omega_{\beta,t}^{(j)} \mathcal{N}(x(t); m_{\beta,t|t-1}^{(i,j)}, P_{\beta,t|t-1}^{(i,j)}) \\ m_{\beta,t|t-1}^{(i,j)} &= F_{\beta,t-1}^{(j)} m_i(t-1) + d_{\beta,t-1}^{(j)} \\ P_{\beta,t|t-1}^{(i,j)} &= Q_{\beta,t-1}^{(j)} + F_{\beta,t-1}^{(j)} P_i(t-1) [F_{\beta,t-1}^{(j)}]^T\end{aligned} \tag{4.12}$$

and $\gamma(x(t))$ given in (4.9). The prior intensity can be re-written in the form of Gaussian mixture:

$$I(x(t)|\mathcal{Z}_{1:t-1}) = \sum_{i=1}^{J_{t|t-1}} \omega_{t|t-1}^{(i)} \mathcal{N}(x(t); m_i(t|t-1), P_i(t|t-1)) \tag{4.13}$$

The posterior intensity at time t, updated with the measurement set $\mathcal{Z}(t)$, is also in the form of Gaussian mixture:

$$I(x(t)|\mathcal{Z}_{1:t}) = (1-p_D) I(x(t)|\mathcal{Z}_{1:t-1}) + \sum_{z_k(t)\in\mathcal{Z}(t)} I_D(x(t)|z_k(t)) \tag{4.14}$$

where $I_D(x(t)|z_k(t))$ is the intensity updated by measurement $z_k(t)$ and is calculated as follow:

$$\begin{aligned}
I_D(x(t)|z_k(t)) &= \sum_{i=1}^{J_{t|t-1}} \omega_t^{(i,k)} \mathcal{N}(x(t); m_{i,k}(t), P_i(t)) \\
\omega_t^{(i,k)} &= \frac{p_D \omega_{t|t-1}^{(i)} p(z_k(t)|m_i(t|t-1))}{\lambda c(z_k(t)) + \sum_{j=1}^{N_{t|t-1}} p_D \omega_{t|t-1}^{(j)} p(z_k(t)|m_j(t|t-1))} \\
p(z_k(t)|m_i(t|t-1)) &= \mathcal{N}(z_k(t); H_t m_i(t|t-1), H_t P_i(t|t-1) H_t^T) \\
m_{i,k}(t) &= m_i(t|t-1) + K_i(t)(z_k(t) - H_t m_i(t|t-1)) \\
P_i(t) &= (I - K_i(t) H_t) P_i(t|t-1) \\
K_i(t) &= P_i(t|t-1) H_t^T (H_t P_i(t|t-1) H_t^T + R_t)^{-1}
\end{aligned} \tag{4.15}$$

To obtain target estimates from the cluttered measurements and reduce the computational cost, the GMPHD filter performs pruning and extraction processes after it updates the posterior intensities. The GMPHD pruning process reduces the number of Gaussian components propagated to the next time step, preventing the number of Gaussian components from increasing infinitely over time. During the pruning process, the Gaussian components that are close to each other are merged to a single Gaussian; in addition, the Gaussian components with weak weights are discarded. The extraction process selects N_t Gaussian components from the J_t remaining Gaussian components after the pruning process based on their weights, whose means are

used as the target state estimates. In this chapter, we use the same pruning and extraction processes as those of the original GMPHD filter [13].

MC-GMPHD Filter

A well-known issue of the GMPHD filter is the loss of target estimates, especially in heavily-cluttered circumstances. As there is no explicit data association, the GM-PHD filter does not explicitly tell the origin of a measurement (i.e., whether it comes from a target or clutter). More specifically, at each time step, the GMPHD filter generates Gaussian components with the same estimation covariance for each individual measurement, regardless of its origin. As a result, the weight of a previously extracted Gaussian component is likely to decrease and the GMPHD filter prefers a spawn or birth Gaussian component (which may in fact come from clutter) to the previously extracted target. Such issue not only results in inaccurate target state estimates, but also leads to incorrect identity management results as the extracted spawn or birth Gaussian components are considered as new targets and are likely to be assigned with new identities.

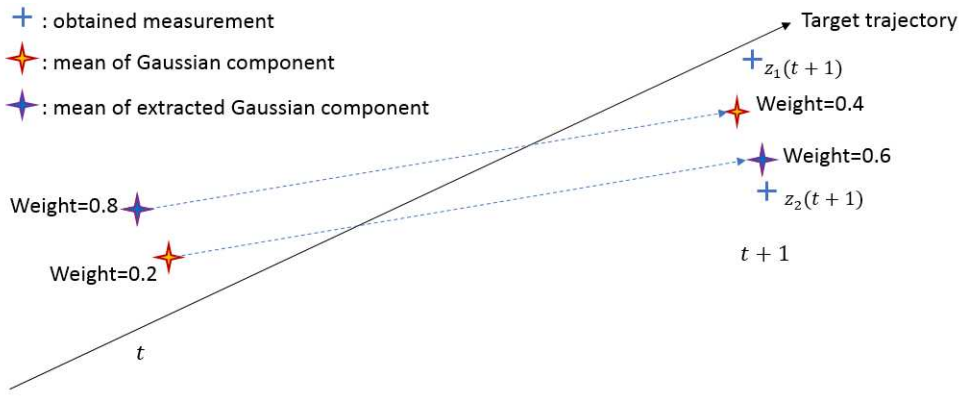


Fig. 4.4.: Illustrative Example of the GMPHD Issue

Figure 4.4 illustrates the GMPHD issue with a single-target tracking example. Suppose at time $t+1$, two measurements are obtained, where $z_1(t+1)$ is an actual

target measurement and $z_2(t+1)$ is a false measurement (clutter). In this example, the GMPHD filter tends to extract the Gaussian component updated by the false measurement $z_2(t+1)$ at time $t+1$ since it has a higher weight. One main reason is that the GMPHD filter does not weigh $z_1(t+1)$ over $z_2(t+1)$ according to the prediction of the Gaussian components, as no explicit data association is involved. Eventually, this leaves an inaccurate target estimate; in addition, the mixing matrix $M(t+1)$ is 0 in this case and a new identity will be falsely assigned to the target.

The GMPHD issue has recently caught research interests and various improvements to the GMPHD filter have been proposed, such as the refined GMPHD filter [110], the improved GMPHD filter [111] and the Gaussian mixture cardinalized PHD [112]. These methods are either heuristic approaches or suffer heavy computational loads. In this chapter, we use a modified-covariance GMPHD (MC-GMPHD) filter that we proposed in our recent work [102], in which the origin uncertainty of a measurement (whether the measurement is originated from a target or clutter) is explicitly considered when updating the Gaussian components. For brevity, we only show the conclusions and essential equations in this section. The detailed formulation and mathematical justification can be found in Section 4.6.2 and [102].

We first define the following events for the k-th measurement $z_k(t) \in \mathcal{Z}(t)$ at time t:

- $E^k_{T,t}$: the event that $z_k(t)$ is originated from a target.
- $E^k_{F,t}$: the event that $z_k(t)$ is originated from a false measurement (i.e., clutter).

The events are assumed to be time-independent, i.e., E^k_{T,t_1} and E^k_{T,t_2} are independent for all $t_1 \neq t_2$, and the same applies to $E^k_{F,t}$. In addition, we assume that the false measurements are uniformly distributed in the surveillance region, and the number of false measurements K is subjected to a Poisson distribution with density λ:

$$\mu_F(K) = \frac{(\lambda V_G)^K}{K!} e^{-\lambda V_G} \tag{4.16}$$

where V_G is the volume of the surveillance region. The pedestal of the MC-GMPHD filter is to calculate the probability $p(E^k_{T,t})$, given the K_t measurements and the $J_{t|t-1}$ predicted Gaussian components. To describe the condition of the predicted Gaussian components, we define the random variable $D_{i,k}(t)$, which is the normalized distance square (NDS) between the k-th measurement and the i-th predicted Gaussian component at time t:

$$D_{i,k}(t) = \nu^T_{i,k}(t) S^{-1}_i(t) \nu_{i,k}(t) \tag{4.17}$$

where $\nu_{i,k}(t) = z_k(t) - H_t m_i(t|t-1)$ whose covariance $S_i(t) = H_t P_i(t|t-1) H^T_t + R_t$. Then, the conditional probability of $E^k_{T,t}$ can be defined and calculated as:

$$p_{i,k}(t) = p(E^k_{T,t}|D_{i,k}(t), K_t) = \frac{f(D_{i,k}(t), E^k_{T,t}, K_t)}{f(D_{i,k}(t), E^k_{T,t}, K_t) + f(D_{i,k}(t), E^k_{F,t}, K_t)} \tag{4.18}$$

where $f(D_{i,k}(t), E^k_{T,t}, K_t)$ and $f(D_{i,k}(t), E^k_{F,t}, K_t)$ are the joint probability density functions (pdf) whose expressions are given in Appendix 4.6.2. By the law of total probability, it is shown [102, 113] that when the measurement origin uncertainty is explicitly considered, the GMPHD covariance update in (4.15) is given by:

$$\begin{aligned} P_{i,k}(t) = {} & p_{i,k}(t) P_i(t|E^k_{T,t}) + (1 - p_{i,k}(t)) P_{i,k}(t|E^k_{F,t}) \\ & + p_{i,k}(t)(1 - p_{i,k}(t)) K_i(t) \nu_{i,k}(t) \nu^T_{i,k} K^T_i(t) \end{aligned} \tag{4.19}$$

where $P_i(t|E^k_{T,t}) = (I - K_i(t) H_t) P_i(t|t-1)$ is the conditioned covariance given $E^k_{T,t}$ and is the same as the standard Kalman filter covariance update; and $P_{i,k}(t|E^k_{F,t})$ is the conditioned covariance given $E^k_{F,t}$ and is given by:

$$P_{i,k}(t|E^k_{F,t}) = P_i(t|t-1) + (\alpha_{i,k}(t) - 1) K_i(t) S_i(t) K^T_i(t) \tag{4.20}$$

where the expression of parameter $\alpha_{i,k}(t)$ is given in Appendix 4.6.2. It should be noted that (4.19) is essentially conditioned by the NDS $D_{i,k}(t)$. That being said, instead of treating all measurements indifferently, the new covariance update (4.19) checks the probability that a measurement is originated from a target according to

the predicted Gaussian components. As a result, the Gaussian components updated by the true target measurements are likely to have higher weights, and hence more accurate target sate estimates and mixing matrices.

Mixing Matrix Construction

The mixing matrix is essentially a collection of the normalized association scores between the current N_t target estimates and the previous N_{t-1} estimates, and is used to propagate the identity belief matrix over time. One approach to obtain the mixing matrix is to calculate it during the multi-target tracking/data association process and feed it to the identity management process [68, 69]. This approach attaches the multi-target tracking algorithms to the identity management algorithms and formulates a systematic and simultaneous solution to both multi-target tracking and multi-target identity management problems. The existing algorithms that can deal with only a known and constant number of targets calculates the mixing matrix in three steps [69]. Suppose at time t, there are K_t new cluttered measurements, i.e., $\mathcal{Z}(t) = \{z_1(t), z_2(t), ..., z_{K_t}(t)\}$. The multi-target tracking/data association process such as JPDAF first calculates the association scores between the K_t measurements and the previous N_{t-1} target estimates; these association scores form an $N_{t-1} \times K_t$ matrix. Then, as the multi-target tracking algorithm selects the measurements and updates N_t target estimates from the measurements, the corresponding elements are selected from the $N_{t-1} \times K_t$ matrix, forming an un-normalized $N_{t-1} \times N_t$ association matrix, i.e., it does not necessarily satisfy the row and column sum constraints (note that $N_{t-1} = N_t = N$ in the conventional identity management case). Finally, the mixing matrix is obtained by rescaling the un-normalized association matrix.

Whereas no explicit data association is involved in the GMPHD framework, the similar idea of constructing the mixing matrix can still be adopted. The starting point is that the calculated weights during the GMPHD update, as described by (4.15), can be viewed as an analogy to the association scores between the new measurements and

the previous Gaussian components; then, as the GMPHD extraction process selects the target estimates from the Gaussian components, the corresponding weights can be extracted as the association scores between the current and previous target estimates and form the un-normalized association matrix, from which the mixing matrix can be acquired by rescaling. One major advantage of this approach of constructing the mixing matrix is that it preserves the GMPHD filter's computational efficiency by avoiding explicit data association, while dealing with the unknown and time-varying number of targets in clutter.

To construct the mixing matrix, we start with the prior intensity $I(x(t)|\mathcal{Z}_{1:t-1})$. By (4.10), the prior intensity can be divided into three separate cases: the predictions of the target birth $\gamma(x(t))$, the target spawn $I_{\beta,t|t-1}(x(t))$ and the target survival $I_{S,t|t-1}(x(t))$. Since the prior intensity is a linear combination of Gaussian components from all the three cases, the posterior intensity updated by the new measurements can also be separated into the same three cases:

$$\begin{aligned} I_D(x(t)|\mathcal{Z}(t)) &= I_{\gamma,D}(x(t)|\mathcal{Z}(t)) + I_{\beta,D}(x(t)|\mathcal{Z}(t)) + I_{S,D}(x(t)|\mathcal{Z}(t)) \\ I_{\cdot,D}(x(t)|\mathcal{Z}(t)) &= \sum_{k=1}^{K_t} I_{\cdot,D}(x(t)|z_k(t)) \end{aligned} \tag{4.21}$$

where the Gaussian mixtures $I_{\gamma,D}(x(t)|\mathcal{Z}(t))$, $I_{\beta,D}(x(t)|\mathcal{Z}(t))$ and $I_{S,D}(x(t)|\mathcal{Z}(t))$ are the updated intensities of the three cases: target birth, target spawn and target survival, respectively. Let $\mathcal{G}_\gamma$, $\mathcal{G}_\beta$ and $\mathcal{G}_S$ be the sets of the Gaussian components of $I_{\gamma,D}(x(t)|\mathcal{Z}(t))$, $I_{\beta,D}(x(t)|\mathcal{Z}(t))$ and $I_{S,D}(x(t)|\mathcal{Z}(t))$, respectively. Each Gaussian component in the update (4.15) belongs to one of the three sets.

By definition, the mixing matrix only concerns the association information with the previous target estimates, therefore, only the Gaussian components in $\mathcal{G}_S$ need to be considered for the construction of the mixing matrix. Following (4.15), the intensity $I_{S,D}(x(t)|z_k(t))$, as the intensity of target survival updated by the measurement $z_k(t)$, can be calculated as:

$$
\begin{aligned}
I_{S,D}(x(t)|z_k(t)) &= \sum_{i=1}^{J_{t-1}} \omega_t^{(i,k)} \mathcal{N}(x(t); m_{i,k}(t), P_{i,k}(t)) \\
\omega_t^{(i,k)} &= \frac{p_D \omega_{t|t-1}^{(i)} p(z_k(t)|m_i(t|t-1))}{\lambda c(z_k(t)) + \sum_{j=1}^{N_{t|t-1}} p_D \omega_{t|t-1}^{(j)} p(z_k(t)|m_j(t|t-1))} \\
p(z_k(t)|m_i(t|t-1)) &= \mathcal{N}(z_k(t); H_t m_i(t|t-1), H_t P_i(t|t-1) H_t^T) \\
m_{i,k}(t) &= m_i(t|t-1) + K_i(t)(z_k(t) - H_t m_i(t|t-1)) \\
K_i(t) &= P_i(t|t-1) H_t^T (H_t P_i(t|t-1) H_t^T + R_t)^{-1}
\end{aligned}
\tag{4.22}
$$

where $P_{i,k}(t)$ is the modified covariance update of MC-GMPHD which is given by (4.19). Note that the Gaussian components $\mathcal{N}(x(t); m_{i,k}(t), P_{i,k}(t))$ correspond uniquely to those in (4.15) that form the set $\mathcal{G}_S$, only with a different way of indexing, i.e., there exists an index mapping $f_t^{(I)}(\cdot) : [1, J_t] \to [1, J_{t-1}] \times [1, K_t]$ such that

$$
f_t^{(I)}(j) = \begin{cases} (i,k) & if\ \mathcal{N}(x(t); m_j(t), P_j(t)) \in \mathcal{G}_S \\ \emptyset & otherwise \end{cases}
\tag{4.23}
$$

Each index pair (i, k) essentially represents the Gaussian approximation of the association hypothesis such that the i-th surviving Gaussian component is associated with the k-th measurement in the FISST framework[1] [13, 108]. Therefore, it is reasonable to consider the weight $\omega_t^{(i,k)}$ as an association score of the hypothesis, which is analogous to the association probability between the $z_k(t)$ and $\mathcal{N}(x(t-1); m_i(t-1), P_i(t-1))$, and construct the following matrix of association scores:

$$
W(t) = [w_{ik}(t) = \omega_t^{(i,k)}] \in [0,1]^{J_{t-1} \times K_t}
\tag{4.24}
$$

As the MC-GMPHD filter performs the pruning and extraction processes, similar actions to $W(t)$ are necessary. When the pruning process merges or discards the updated Gaussian components, it is essentially the association hypotheses that are

[1]which is the emulating idea of conventional data association in FISST to avoid the combinatorial calculation of association probabilities.

merged or discarded, and therefore the corresponding association scores in $W(t)$ also need to be changed. Similarly, the extraction process in fact selects the association hypotheses as it extracts the Gaussian components, and the association scores need to be extracted accordingly.

The MC-GMPHD pruning process has two outcome cases: discard and merge. For the discard case, if a Gaussian component $\mathcal{N}(x(t); m_{i,k}(t), P_{i,k}(t))$ from the set $\mathcal{G}_S$ is discarded, it conveys the idea that the hypothesis that $z_k(t)$ is associated with $\mathcal{N}(x(t-1); m_i(t-1), P_i(t-1))$ is discarded; therefore, in the association score matrix $W(t)$, the corresponding score w_{ik} is set to 0. As for the merge case, assume the Gaussian components in the set $\mathcal{G}_S$ with indices $(i_m, k_m) \in \mathcal{L}_m \subset \mathcal{L}$, where $\mathcal{L} = \{(i,k)|i \in \{1,2,...,J_{t-1}\}, k \in \{1,2,...,K_t\}\}$, are merged with any Gaussian components in the sets $\mathcal{G}_\gamma$ and $\mathcal{G}_\beta$ into a single Gaussian component, then the multiple association hypotheses represented by all $(i_m, k_m) \in \mathcal{L}_m$ are considered to be a single one. In the context of identity management, however, if the merged Gaussian component is eventually extracted as a target estimate, it may be originated from any of the Gaussian components indexed by $(i_m, k_m) \in \mathcal{L}_m$ and therefore share the same ID with these Gaussian components. In order to represent the idea above in the mixing matrix, the following procedure is adopted. Let the set of merged Gaussian components be $\mathcal{G}_M$, and the number of merged Gaussian components $J_M = |\mathcal{G}_M|$. Then, for the merged Gaussian components, we define the following matrix:

$$\tilde{W}(t) = [\tilde{\boldsymbol{w}}_1, ..., \tilde{\boldsymbol{w}}_{j_m}, ..., \tilde{\boldsymbol{w}}_{J_M}] \tag{4.25}$$

where $j_m \in \{1,2,...,J_M\}$, $\tilde{\boldsymbol{w}}_{j_m} = [\tilde{w}_{ij_m}]^T \in \{0, \omega_t^{(j_m)}/J_M\}^{J_{t-1}\times 1}$ is a column vector, $\omega_t^{(j_m)}$ is the merged weight of the j_m-th Gaussian component, and

$$\tilde{w}_{ij_m} = \begin{cases} \omega_t^{(j_m)}/J_M & if\ \exists k\ s.t.\ (i,k) \in \mathcal{L}_m \\ 0 & otherwise \end{cases} \tag{4.26}$$

i.e., the association score is evenly distributed among all the truncated Gaussian components. Note that by the definitions above, there exists an index mapping $f_t^{(M)}(\cdot) : [1, J_t] \to [1, J_m]$ such that

$$f_t^{(M)}(j) = \begin{cases} j_m & if\ \mathcal{N}(x(t); m_j(t), P_j(t)) \in \mathcal{G}_M \\ \emptyset & otherwise \end{cases} \tag{4.27}$$

Eventually, two association matrices $W(t)$ and $\tilde{W}(t)$ are created after the GMPHD update and pruning. The step-by-step procedure of constructing the two association matrices is shown in Algorithm 1.

As the MC-GMPHD filter performs the extraction process and selects Gaussian components, the corresponding weights are also selected from either $W(t)$ or $\tilde{W}(t)$ to form the mixing matrix $M(t)$. The extraction process may extract Gaussian components from either of the four sets $\mathcal{G}_\gamma,\ \mathcal{G}_\beta,\ \mathcal{G}_S$ and $\mathcal{G}_M$. If a Gaussian component in either $\mathcal{G}_\gamma$ or $\mathcal{G}_\beta$ as a target state estimate, by definition, the estimate should have 0 association probability with any previous estimates. Therefore, a column of all 0s is added to $M(t)$. On the other hand, if a Gaussian component in $\mathcal{G}_S$ or $\mathcal{G}_M$ is extracted, one needs to find its association probability with the previous target state estimates, i.e., the previously extracted Gaussian components. Note that at time t, the GMPHD extraction process in fact creates an index mapping $f_t^{(E)}(\cdot) : [1, J_t] \to [1, N_t]$:

$$f_t^{(E)}(i) = \begin{cases} j & if\ m_i(t) = \hat{x}_j(t) \\ \emptyset & otherwise \end{cases} \tag{4.28}$$

Then, if the j^*-th Gaussian component in $\mathcal{G}_S$ is extracted as the j-th target estimate $\hat{x}_j(t)$, we extract a sub-column from the matrix $W(t)$ as the j-th column of $M(t)$ (denoted as $\boldsymbol{m}_j(t)$):

$$\boldsymbol{m}_j(t) = [w_{ik}]^T,\ \forall i\ s.t.\ f_{t-1}^{(E)}(i) \neq \emptyset \tag{4.29}$$

Algorithm 1 Construct association matrices at time t

Given: sets of Gaussian components $\mathcal{G}_\gamma$, $\mathcal{G}_\beta$ and $\mathcal{G}_S$, and the posterior intensity $I(x(t)|\mathcal{Z}_{1:t})$

for j in $1 : J_t$ **do**

  if $f_t^{(I)}(j) \neq \emptyset$ **then**

    $(i, k) \leftarrow f_t^{(I)}(j)$

    $w_{ik}(t) \leftarrow \omega_t^{(i,k)}$

  end if

end for

Perform GMPHD pruning and obtain the set $\mathcal{G}_M$

//update Gaussian component sets

$\mathcal{G}_S \leftarrow \mathcal{G}_S - \mathcal{G}_S \cap \mathcal{G}_M$

$\mathcal{G}_\gamma \leftarrow \mathcal{G}_\gamma - \mathcal{G}_\gamma \cap \mathcal{G}_M$

$\mathcal{G}_\beta \leftarrow \mathcal{G}_\beta - \mathcal{G}_\beta \cap \mathcal{G}_M$

for j in $1 : J_t$ **do**

  if $f_t^{(M)}(j) \neq \emptyset$ **then**

    $j_m \leftarrow f_t^{(M)}(j)$

    Construct $\tilde{\boldsymbol{w}}_{j_m}$ according to (4.26)

  end if

  if $f_t^{(I)}(j) \neq \emptyset$ and $\mathcal{N}(x(t); m_j(t), P_j(t))$ is discarded by pruning **then**

    $(i, k) \leftarrow f_t^{(I)}(j)$

    $w_{ik}(t) \leftarrow 0$

  end if

end for

$\tilde{W}(t) \leftarrow \{\tilde{\boldsymbol{w}}_{j_m}\}$

return $W(t),\ \tilde{W}(t)$

where k is obtained from the mapping $f_t^{(I)}(j^*)$. Similarly, if the j^*-th Gaussian component in $\mathcal{G}_M$ is extracted as the j-th target estimate, we extract a sub-column from the matrix $\tilde{W}(t)$:

$$\boldsymbol{m}_j(t) = [\tilde{w}_{ik}]^T,\ \forall i\ s.t.\ f_{t-1}^{(E)}(i) \neq \emptyset \tag{4.30}$$

where $k = f_t^{(M)}(j^*)$. Algorithm 2 describes the steps of extracting the mixing matrix $M(t)$ at time t.

Remark 1 *By (4.22), the column sum of $W(t)$ is less than 1. Therefore, a good extraction weight threshold, e.g., 0.5 as the most general case [13], guarantees that only at most one Gaussian component corresponding to each column of $W(t)$ will be extracted. This essentially indicates that each measurement is generated by at most one target.*

Remark 2 *Since the mixing matrix $M(t)$ is obtained from the weights of Gaussian components, the proposed algorithm does not involve any additional computational complexity from explicit data association.*

The extracted mixing matrix $M(t)$ is $N_{t-1} \times N_t$ whose element $m_{ij}(t)$ essentially represents the association score between $\hat{x}_i(t-1)$ and $\hat{x}_j(t)$, calculated by the procedure described above. However, $M(t)$ may not necessarily satisfy its row and column sum constraints and needs to be re-scaled. For this, we first augment the current $M(t)$ matrix with its 'residue' of column sums:

$$\begin{gathered} M^{(1)}(t) = [M(t); \bar{\boldsymbol{m}}(t)^T],\ \bar{\boldsymbol{m}}(t) = [\bar{m}_1(t), \bar{m}_2(t), ..., \bar{m}_{N_t}(t)]^T \\ \bar{m}_j(t) = \begin{cases} 0,\ if\ \sum_{i=1}^{N_{t-1}} m_{ij}(t) > 1 \\ 1 - \sum_{i=1}^{N_{t-1}} m_{ij}(t),\ otherwise \end{cases} \end{gathered} \tag{4.31}$$

Then, the augmented $(N_{t-1}+1) \times N_t$ matrix $M^{(1)}(t)$ is rescaled to a column stochastic matrix by solving the following optimization problem that we derived in this work:

Algorithm 2 Extract mixing matrix $M(t)$ at time t

Given: matrices $W(t)$ and $\tilde{W}(t)$, and the posterior intensity $I(x(t)|\mathcal{Z}_{1:t})$

for j in $1, 2, ..., J_{t-1}$ **do**

  if $f_{t-1}^{(E)}(j) \neq \emptyset$ **then**

    $\boldsymbol{j_0} \leftarrow [\boldsymbol{j_0}, f_{t-1}^{(E)}(j)]$

  end if

end for

Perform GMPHD extraction and update the mapping $f_t^{(E)}(\cdot)$

for j in $1, 2, ..., J_t$ **do**

  if $f_t^{(E)}(j) \neq \emptyset$ **then**

    if $f_t^{(I)}(j) \neq \emptyset$ **then**

      $(i, k) \leftarrow f_t^{(I)}(j)$

      $\boldsymbol{m} \leftarrow w_{\boldsymbol{j_0}k}$

    else if $f_t^{(M)}(j) \neq \emptyset$ **then**

      $k \leftarrow= f_t^{(M)}(j)$

      $\boldsymbol{m} \leftarrow \tilde{w}_{\boldsymbol{j_0}k}$

    else

      $\boldsymbol{m} \leftarrow \mathbf{0}$

    end if

    $M(t) \leftarrow [M(t) \mid \boldsymbol{m}]$

  end if

end for

return $M(t)$

$$
\begin{aligned}
\text{find} \quad & m_{ij}^{(2)}(t) \\
\min \quad & \sum_{j=1}^{N_t} \sum_{i=1}^{N_{t-1}+1} m_{ij}^{(2)}(t) \log \frac{m_{ij}^{(2)}(t)}{m_{ij}^{(1)}(t)} \\
\text{s.t.} \quad & \sum_{j=1}^{N_t} m_{ij}^{(2)}(t) \leq 1 \\
& \sum_{i=1}^{N_{t-1}+1} m_{ij}^{(2)}(t) = 1 \\
& m_{ij}^{(2)}(t) \geq 0 \text{ if } m_{ij}^{(1)}(t) > 0 \\
& m_{ij}^{(2)}(t) = 0 \text{ if } m_{ij}^{(1)}(t) = 0 \\
& \forall i \in \{1, 2, ..., N_{t-1} + 1\}, \forall j \in \{1, 2, ..., N_t\}
\end{aligned}
\tag{4.32}
$$

The reason that we rescale the augmented matrix $M^{(1)}(t)$ to a column stochastic matrix is that the objective function in problem (4.32) represents the KL distance only if the column stochastic constraint is satisfied. Only then, the rescaled matrix $M^{(2)}(t)$, as the solution to problem (4.32) is 'closest' to $M^{(1)}(t)$ from a statistical perspective. The rescaled mixing matrix $M(t)$ is then the first N_{t-1} rows of $M^{(2)}(t)$. The identity belief matrix is then propagated over time using the mixing matrix following the propagation equation (4.3).

4.4.2 Identity Belief Matrix Update

After the propagation of the identity belief matrix, it is determined whether an existing ID needs to be removed (i.e., delete a row from the identity belief matrix) or new IDs need to be added (i.e., add rows to the belief matrix). If the i-th row sum of $B(t)$, denoted as $\sum_j b_{ij}$, is smaller than a threshold, we remove ID i by deleting the i-th row from $B(t)$. On the other hand, if the j-th column sum of $B(t)$, denoted as $\sum_i b_{ij}$, is smaller than a threshold, we add a new ID by appending the matrix $B(t)$ with a new row whose entries are set to 0 except that the j-th entry is set as $1 - \sum_i b_{ij}$. After the above steps, if the number of rows of $B(t)$ is less than the number

of columns, we add more new IDs by appending multiple rows to $B(t)$ to make the matrix square. The new belief matrix is then rescaled to meet the row and column sum constraints.

Following the definitions and notations in Section 4.3, and let L_t and N_t be the number of rows and columns of the identity belief matrix $B(t)$, respectively, Algorithm 3 describes the procedure of updating the identity belief matrix.

In addition, the identity belief matrix can also be updated when there is available local information on the identities of some targets. The purpose of updating the identity belief matrix with local information is to reduce the uncertainties in the identities of all the targets within the sensor's surveillance range as identity management keeps track of all the targets' identities. As described in Section 4.3, the local information is in the form of a probability distribution vector of the identities, and can be used to update the identity belief matrix by replacing the corresponding column in the identity belief matrix and then rescale the matrix. However, not every local information is necessarily useful (i.e., is able to reduce the uncertainties in the identities) and should be used to update the identity belief matrix. To determine the 'usefulness' of local information, we use the statistical entropy of a probability distribution vector $\boldsymbol{f} \in [0,1]^N$, defined as

$$H[\boldsymbol{f}] = \sum_{i=1}^{N} -f_i \log f_i \tag{4.33}$$

as an uncertainty measure of $\boldsymbol{f}$. Then, the uncertainty in the identities of all targets is defined by the average statistical entropy of the identity belief matrix $B(t)$:

$$\tilde{H}[B(t)] = \frac{1}{N_t} \sum_{j=1}^{L_t} H[\boldsymbol{b_j}(t)] \tag{4.34}$$

which is used as an uncertainty measure in the identity. Since the purpose of local information incorporation in the identity management is to reduce the uncertainties in the targets' identities, a piece of local information should be considered as 'useful' only if it is able to reduce the statistical entropy of the identity belief matrix. Therefore,

Algorithm 3 Update identity belief matrix $B(t)$ at time t

Given: propagated identity belief matrix $B(t) = B(t-1)M(t) + \lambda\bar{\boldsymbol{b}}(t-1)\bar{\boldsymbol{m}}(t)^T$

for i in 1:L_t **do**

  if $\sum_{j=1}^{N} b_{ij} \leq$ row_thres **then**

    Remove i-th row from $B(t)$

  end if

end for

for j in 1:N_t **do**

  if $\sum_{i=1}^{L} b_{ij} \leq$ col_thres **then**

    new_row $\leftarrow [0, ..., 0, 1 - \sum_{i=1}^{L} b_{ij}, 0, ..., 0]$

    $B(t)$.append(new_row)

  end if

end for

if $L_t < N_t$ **then**

  rows_needed $\leftarrow N - L$

  new_row $\leftarrow [1 - \sum_{i=1}^{L} b_{i1}, ..., 1 - \sum_{i=1}^{L} b_{ij}, 1 - \sum_{i=1}^{L} b_{iN}]$

  new_row $\leftarrow$ new_row/rows_needed

  for i in 1:rows_needed **do**

    $B(t)$.append(new_row)

  end for

end if

$B(t) \leftarrow$ rescale $B(t)$ by solving optimization problem (4.5)

return $B(t)$

a piece of local information will be used to update the identity belief matrix only if the resulting identity belief matrix $B'(t)$ has smaller statistical entropy, i.e.:

$$\tilde{H}[B'(t)] < \tilde{H}[B(t)] \tag{4.35}$$

The procedure of updating the identity belief matrix $B(t)$ with local information is shown in Algorithm 4.

Algorithm 4 Update $B(t)$ with local information at time t

Given: matrix $B(t)$, local information $\boldsymbol{l}_j$ about target j

$B'(t) \leftarrow [\boldsymbol{b_1}(t), ..., \boldsymbol{b_{j-1}}(t), \boldsymbol{l}_j, \boldsymbol{b_{j+1}}(t), ..., \boldsymbol{b_{N_t}}(t)]$

$B'(t) \leftarrow$ rescale $B'(t)$ by solving optimization problem (4.5)

if $\tilde{H}[B'(t)] < \tilde{H}[B(t)]$ **then**

  $B(t) \leftarrow B('t)$

end if

return $B(t)$

4.5 Numerical Examples

In this section, we demonstrate the performance of the GMPHD-IM algorithm with illustrative numerical simulations. Three test scenarios are studied in this section. In the first test scenario, we use the GMPHD-IM algorithm to manage the identities of several targets, some of which may appear or disappear during the multi-target tracking mission. We show the GMPHD-IM algorithm's capability of managing the identities as well as estimating the states of an unknown and time-varying number of targets. The second and third test scenarios address two particularly challenging cases in multi-target identity management: scenario 2 presents a case of two-target coalescence, while scenario 3 shows a case where a new target spawns from the current target. With the two scenarios we demonstrate the GMPHD-IM algorithm's capability of managing the identities and updating the identity beliefs whenever there is

available local information. In the two test scenarios, we assume that there is available local information at specific time instances. The targets in all test scenarios are assumed to be 2D targets, i.e., only the positions and velocities in the x and y directions are considered.

4.5.1 Target and Sensor Models

In the simulation cases, a fixed-position sensor (e.g., a Radar station) is used to track and manage the identities of all targets. The sensor is assumed to cover the entire surveillance area and is able to detect targets in the area with a probability of detection of 0.98. At each time step, the sensor provides cluttered target position measurements with noise covariance:

$$\begin{bmatrix} (10)^2 & 0 \\ 0 & (10)^2 \end{bmatrix}$$

i.e., the standard deviation of the target position measurement error is 10m in both x and y directions. The measurement clutter is generated randomly per time instance. At time t, the target state vector in the MC-GMPHD filter is given by $x(t) = [p_x, p_y, \dot{p}_x, \dot{p}_y]^T$, where p_x and p_y represent the positions in the x and y directions, respectively. The MC-GMPHD filter uses the constant velocity model for target state propagation [51]:

$$x(t+1) = F_{cv}x(t) + G_{cv}w(t) \tag{4.36}$$

where

$$F_{cv} = \begin{bmatrix} 1 & 0 & T_s & 0 \\ 0 & 1 & 0 & T_s \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad G_{cv} = \begin{bmatrix} T_s^2/2 & 0 \\ 0 & T_s^2/2 \\ T_s & 0 \\ 0 & T_s \end{bmatrix}$$

$w(k) \sim \mathcal{N}(0, \sigma)$ is a white noise, and T_s is the measurement sampling time which is set to be 1 second. The rest of the MC-GMPHD filter and identity management parameters are given as follows. The truncation and merge thresholds in the pruning process are set as 10^{-5} and 4, respectively. The weight threshold of extraction is set to 0.5. The probability of target survival is 0.99, and the weight of the spawned and birth target are both 0.1. Note that the above parameters are a common setup in the GMPHD studies [13]. The threshold to add a new identity, as described in Section 4.3 and Algorithm 3, is set to 0.2. The threshold to remove an identity from the set of identities is set to 0.02.

4.5.2 Scenario 1: An Unknown and Time-varying Number of Targets

The scenario contains four targets that may appear/disappear during the simulation, so that the number of targets is unknown and time-varying. Hence, the existing identity management algorithm [69] is unable to handle the case. The purpose of this test scenario is not only to show the proposed GMPHD-IM's capability of managing the identities of an unknown and time-varying number of targets, but also to study the performance/limits of the algorithm in practice (i.e., various measurement clutter levels). Therefore, we run the proposed algorithm with two different clutter densities: the average number of clutters at each time instance is set to 40 and 100, respectively.

The trajectories of the targets are shown in Figure 4.5. The total simulation time is set to 55 seconds. Initially, there are only targets 1 and 2. Target 3 appears at time 22 and disappears at time 52, while target 4 appears at time 32 and remains until the end of simulation.

Figure 4.6 shows the algorithm output with an average clutter number per time step set at 40. The MC-GMPHD filter outputs two ghost targets (false alarms) at time 39 and 47, respectively, and mis-detects target 1 at time 40, as shown in Figure 4.6(a). Figure 4.6(b) shows the target identity beliefs calculated by the GMPHD-IM algorithm, where the four plots correspond to the identity beliefs of target 1 to target

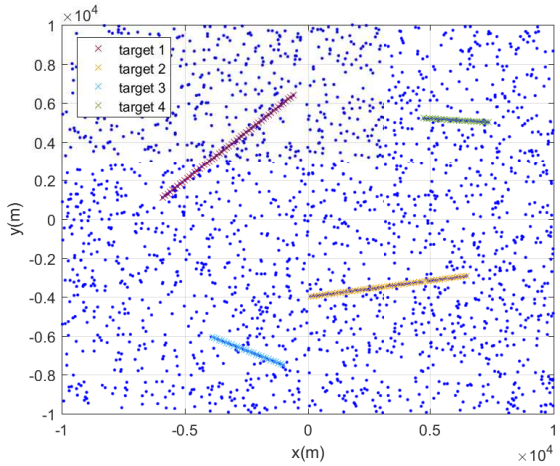
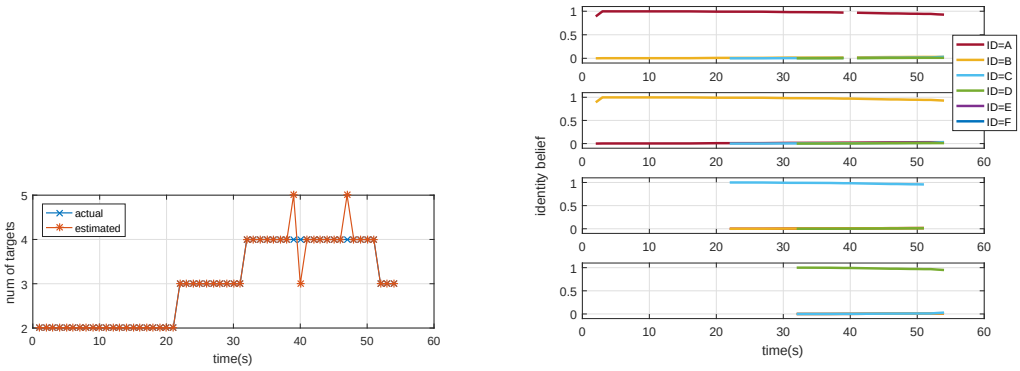


Fig. 4.5.: Target Trajectories and Clutter of Scenario 1

4 from top to bottom. Initially, there are only two targets and the identity belief matrix $B(t)$ is 2×2, where the identities of targets 1 and 2 are known as 'A' and 'B', respectively. As target 3 appears at time 22, a new ID 'C' is assigned to the target and added to the identity set $\mathcal{I}(t)$ and the dimension of $B(t)$ becomes 3×3. Similarly, ID 'D' is assigned to target 4 when it appears at time 32 and the dimension of $B(t)$ becomes 4×4. Two identities 'E' and 'F' are assigned to the ghost targets briefly when they appear at time 39 and 47, respectively. The two identities are removed right after the ghost targets disappear in one time step. At time 52, target 3 disappears; however, ID 'C' is kept in $\mathcal{I}(t)$ and the dimension of $B(t)$ becomes 4×3 since then. Regardless, the identities of targets 1, 2 and 4 are still correctly tracked.

Figure 4.7 shows the identity management output with an average clutter number per time step set at 100 (i.e., a more difficult case). In general, the targets are more likely to get mis-detected due to the higher clutter density. In this example, target 1 is temporarily undetected at time 13, 20 and 40, and target 2 is not detected at time 46, which results in the frequent change of number of estimated targets in Figure 4.7(a). When target 1 is re-detected at time 14 and 21, it is assigned to new IDs ('C' and 'D'). On the other hand, the same ID was preserved when target 1 and target 2 are re-detected at time 41 and 47, respectively. It should be noted that whether a target



(a) Number of Targets

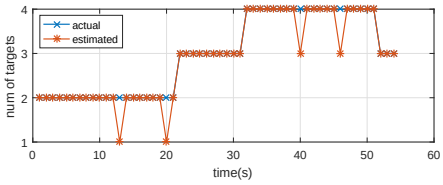
(b) Target Identity Beliefs

Fig. 4.6.: Identity Management Results from GMPHD-IM for Scenario 1 (Avg. Clutter Number=40)

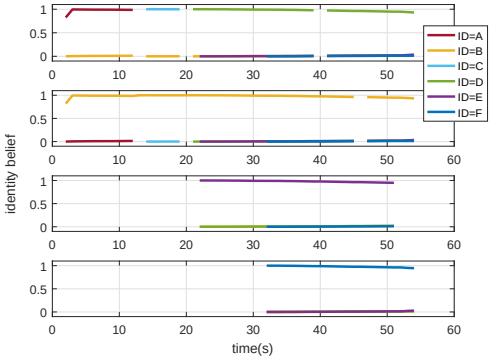
is assigned a new ID or its previous ID upon re-detection depends on various factors such as the density of measurement clutters, how close the clutters are to the real target, etc. However, the GMPHD-IM algorithm can keep track of the ID assigned to the re-detected target, as shown in Figure 4.7(b). Therefore, when local information about the re-detected target is available, the GMPHD-IM algorithm is able to update itself with the correct target ID. Figure 4.7(c) demonstrates a case in which the system receives local information at time 42 (shortly after the re-detection of target 1) such that the probability of target 1 having ID 'A' is 0.99. The GMPHD-IM algorithm correctly updates the target identity beliefs, removing the redundant IDs 'C' and 'D', and keeps track of the correct IDs.

4.5.3 Scenario 2: Two-Target Coalescence

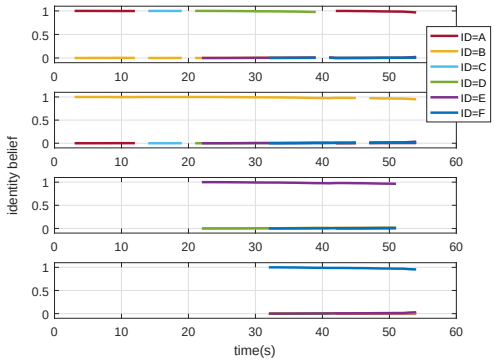
This scenario contains two targets that coalesce with each other and then separate. The trajectories of the targets are shown in Figure 4.8. The total simulation time is set to 55 seconds, and the average number of measurement clutters per time step is set to 40. Initially, the two targets start at positions $(3000, -2000)$ and $(3000, 2000)$



(a) Number of Targets



(b) Target Identity Beliefs without Local Information Update



(c) Target Identity Beliefs with Local Information Update

Fig. 4.7.: Identity Management Results from GMPHD-IM for Scenario 1 (Avg. Clutter Number=100)

(right side of the figure) and move toward the directions shown by the black arrows, respectively. The two targets get close around $t = 20$ and separate again around $t = 40$. During the coalescence period, the two targets are too close to be distinguished and the MC-GMPHD filter returns only one target estimate. Therefore, this scenario has challenges from both target coalescence and the time-varying number of targets. We use this scenario to show the GMPHD-IM algorithm's capability of managing the identities of the coalesced targets and more importantly, updating the identity beliefs

of the coalesced targets whenever there is available local information. We assume that at time 40, the system receives local information on target 1 such that the probability of target 1 having ID 'A' is 0.99.

In addition to running the simulation with the proposed GMPHD-IM algorithm, we also run the simulation with the MC-GMPHD tracker [100, 102] for comparison. The MC-GMPHD tracker adds a data association process to the MC-GMPHD filter to obtain the tracks of the targets.

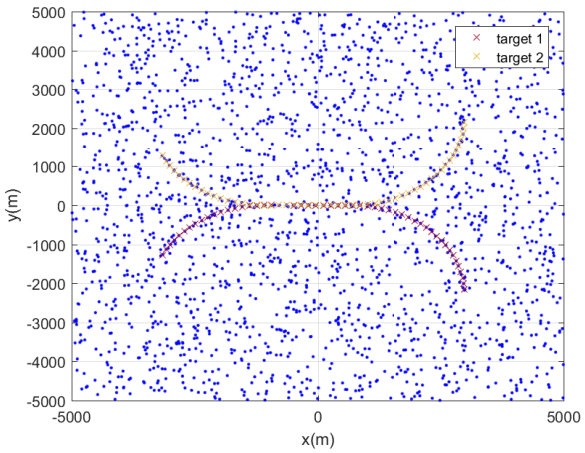
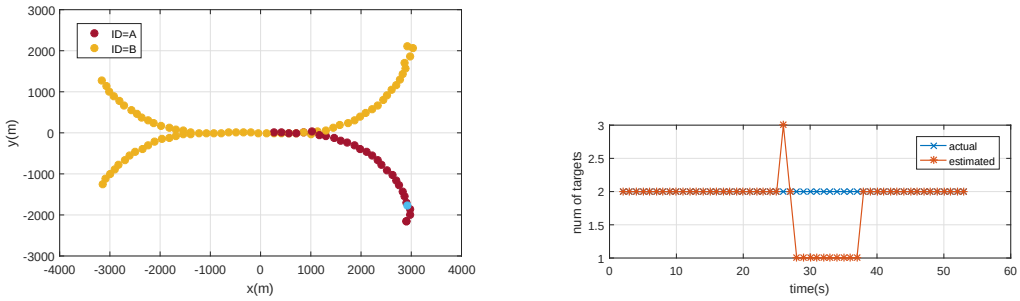


Fig. 4.8.: Target Trajectories and Clutter of Scenario 2

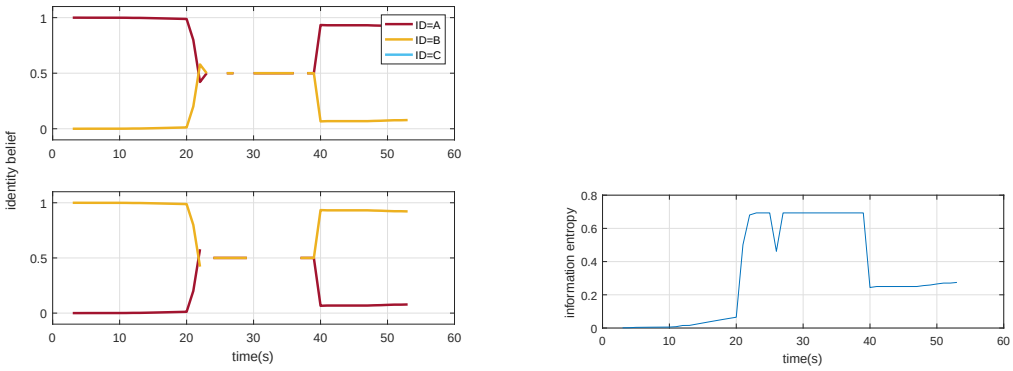
We first present the MC-GMPHD tracker's output for this scenario. Figure 4.9(a) shows the target state estimates computed by the MC-GMPHD tracker, colored by the track IDs assigned to them. At first, the two targets are well separated and two IDs are correctly assigned to their estimates, respectively. When the two targets coalesce, the MC-GMPHD tracker returns only one target estimate from time 27 through time 38, and only one track ID is assigned to the target ('B' in this case) without considering the possibility that the target estimate may as well come from the other track. As a result, when the two targets separate again, both targets are assigned with the same track ID. Moreover, the MC-GMPHD tracker is unable to update the track IDs with local information, when it information comes in the form of probability measurements (e.g., 'the probability of target 1 having ID 'A' is 0.99').



(a) GMPHD Estimates with Assigned Labels (b) Number of Targets

Fig. 4.9.: Target Tracking and Labeling from MC-GMPHD Tracker for Scenario 2

The output of the proposed GMPHD-IM algorithm for this test scenario is shown in Figure 4.10. Initially, the two targets are well-separated and detected, and are correctly assigned their IDs 'A' and 'B'. A ghost target appears near the starting location of target 1 at time 26, and a new ID 'C' is temporarily assigned to the target which is immediately removed after the ghost target disappears. The new ID does not affect the identity belief of the two real targets since the ghost target is far away from the two targets. The two targets start to merge around time 20, and their identity beliefs mingle with each other. It should be noted that even though there is only one target estimate between time 27 and 38, the GMPHD-IM algorithm preserves both IDs and indicates that the target estimate has a 0.5 probability of being each of the IDs, which is a correct representation of track coalescence. Then, when the local information 'the probability of target 1 having ID 'A' is 0.99' is available at time 40, the GMPHD-IM algorithm incorporates the information and correctly updates the identity beliefs immediately, as shown in Figure 4.10(a). The uncertainties in the identities of all targets, as defined by (4.34), is shown in Figure 4.10(b). The uncertainty increases rapidly when the target coalescence starts. As the GMPHD-IM algorithm updates with the local information, the uncertainty rapidly decreases to a lower level, however, it does not reduce to 0 since the local information itself is not certain about target 1's identity.



(a) Target Identity Beliefs (b) Uncertainties in Identities of All Targets

Fig. 4.10.: Identity Management Results from GMPHD-IM for Scenario 2

4.5.4 Scenario 3: Target Spawn

In this scenario, we demonstrate the GMPHD-IM algorithm's capability of managing and updating the identity beliefs with available local information in the situation where a new target spawns from the current target. Figure 4.11 shows the target trajectory of the scenario. The total simulation time is set to 55 seconds, and the average number of measurement clutters per time step is set to 40. Initially, there is only one target (target 1). At time 37, a new target (target 2) spawns from the original target. Since the spawned target is extremely close to the original target, it is hard to be distinguished. We assume that at time 42, the system receives local information such that the probability of target 1 having ID 'A' is 0.99. This scenario has particular practical significance such as a missile or a small unmanned aircraft detached from a large aircraft. Again, we run the simulation with both the proposed GMPHD-IM algorithm and the MC-GMPHD tracker for comparison.

Figure 4.12(a) shows the target estimates by the MC-GMPHD tracker, colored by the track IDs assigned to them. After the spawned target is detected by the MC-GMPHD tracker, it is incorrectly assigned with the same track ID as that of the

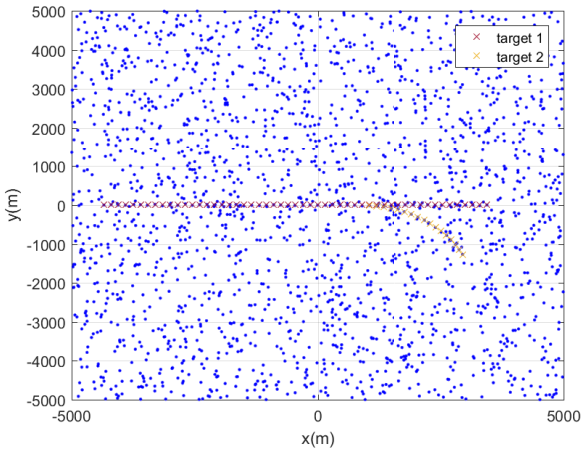
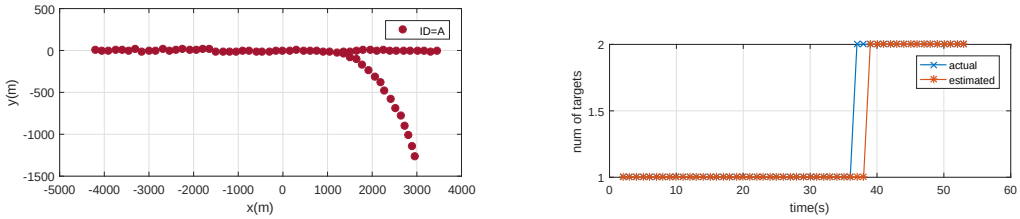


Fig. 4.11.: Target Trajectories and Clutter of Scenario 3

original target since the two targets are too close, disregarding the fact that one of the two targets is a new one and should be assigned with a new ID.

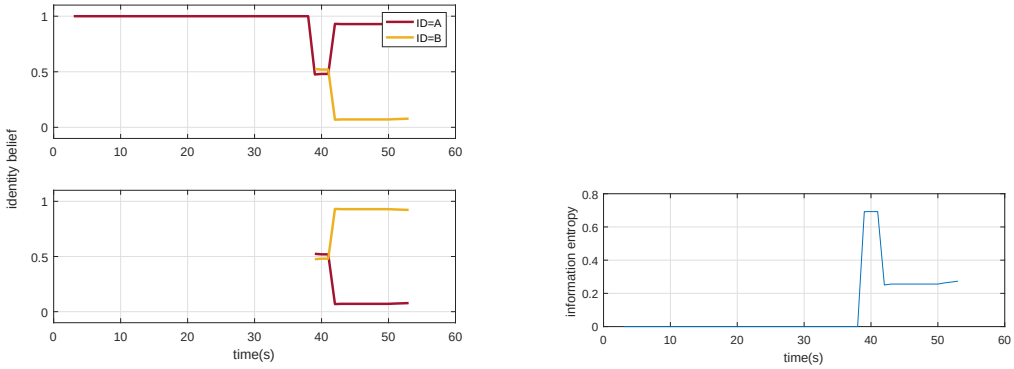


(a) GMPHD Estimates with Assigned Labels (b) Number of Targets

Fig. 4.12.: Target Tracking and Labeling from MC-GMPHD Tracker for Scenario 3

The output of the GMPHD-IM algorithm for the test scenario is shown in Figure 4.13. When the spawned target appears, the algorithm correctly assigns a new ID to it. However, without any additional information, the identity beliefs of the two IDs are almost even. When the local information is available at time 42, the GMPHD-IM algorithm is then able to incorporate the information and update the identity beliefs correctly, as shown in Figure 4.13(a). Figure 4.13(b) shows the uncertainties in the identities of all targets. A sudden increase in the uncertainty appears

when the new target spawns since the two targets are hardly distinguishable. The GMPHD-IM algorithm then can drastically reduces the uncertainty with the local information incorporation.



(a) Target Identity Beliefs (b) Uncertainties in Identities of All Targets

Fig. 4.13.: Identity Management Results from GMPHD-IM, Scenario 3

4.6 Proofs and Equations

4.6.1 Matrix Rescaling

The matrix rescaling problem aims to rescale a positive matrix (i.e., all elements are non-negative, and there are no rows or columns of zeros) with prescribed row and column sum constraints. As the most commonly applied case, the Sinkhorn algorithm [76] rescales a positive square matrix to a doubly-stochastic matrix (all row and column sums are 1). The general case is to rescale an arbitrary-dimensional rectangular matrix with inequality row and column sum constraints, which has been well studied by mathematicians [105, 106]. In this section, we show that the general matrix rescaling problem is equivalent to solving a particular optimization problem, which can be applied to our identity management problem.

Although the term 'rescaling' makes an intuitive sense, it is still desired to have a mathematical definition of 'rescaling', i.e., what it means to say one matrix is

proportional to another. To begin with, we let $\boldsymbol{x} > \mathbf{0}$, for $\boldsymbol{x}$ being either a vector or a matrix, which means that every component of $\boldsymbol{x}$ is positive, while $\boldsymbol{x} \geq \mathbf{0}$ means every component of $\boldsymbol{x}$ is non-negative, and $M = 1, 2, ..., i, ..., m$ and $N = 1, 2, ..., j, ..., n$

A *problem* is defined as a pair $(\boldsymbol{p}, \boldsymbol{\sigma})$, where $\boldsymbol{p} = \{p_{ij}\} \geq \mathbf{0}$ is an $m \times n$ matrix containing no rows or columns of zeros, and $\boldsymbol{\sigma} = (\boldsymbol{r}^-, \boldsymbol{r}^+, \boldsymbol{c}^-, \boldsymbol{c}^+, h)$ is a vector with two m-vectors $\boldsymbol{r}^- = \{r_i^-\} \geq \mathbf{0}$ and $\boldsymbol{r}^+ = \{r_i^+\} > \mathbf{0}$, two n-vectors $\boldsymbol{c}^- = \{c_j^-\} \geq \mathbf{0}$ and $\boldsymbol{c}^+ = \{c_j^+\} > \mathbf{0}$, and a positive scalar h. The set of allocation $R(\boldsymbol{\sigma})$ is defined as

$$R(\boldsymbol{\sigma}) = \{\boldsymbol{f} = \{f_{ij}\} \geq \mathbf{0} : \\ r_i^- \leq f_{iN} \leq r_i^+, i \in M; c_j^- \leq f_{Mj} \leq c_j^+, j \in N; f_{MN} = h\}$$

where $f_{MN} = \sum_{M \times N} f_{ij}$. Moreover, we have two subsets of $R(\boldsymbol{\sigma})$ such that $R^+(\boldsymbol{p}, \boldsymbol{\sigma}) \subset R^0(\boldsymbol{p}, \boldsymbol{\sigma}) \subset R(\boldsymbol{\sigma})$, where

$$R^0(\boldsymbol{p}, \boldsymbol{\sigma}) = \{\boldsymbol{f} \in R(\boldsymbol{\sigma}) : f_{ij} = 0 \text{ if } p_{ij} = 0\}$$

$$R^+(\boldsymbol{p}, \boldsymbol{\sigma}) = \{\boldsymbol{f} \in R(\boldsymbol{\sigma}) : f_{ij} = 0 \text{ if and only if } p_{ij} = 0\}$$

The 'proportional rescaling' of a matrix is defined through the following. Let δ be a scalar, $\boldsymbol{\lambda} = \{\lambda_i\}$, $\boldsymbol{\mu} = \{\mu_j\}$ and $\boldsymbol{p} = \{p_{ij}\}$. Then, $\delta\boldsymbol{\lambda}\boldsymbol{p}\boldsymbol{\mu}$ represents the matrix $\{\delta\lambda_i p_{ij}\mu_j\}$. A matrix $\boldsymbol{f}$ is said to be a *fair share matrix* for a problem $(\boldsymbol{p}, \boldsymbol{\sigma})$ if

$$\boldsymbol{f} = \delta\boldsymbol{\lambda}\boldsymbol{p}\boldsymbol{\mu}, \boldsymbol{f} \in R(\boldsymbol{\sigma})$$

for some $\delta > 0$,$\boldsymbol{\lambda} > \mathbf{0}$, and $\boldsymbol{\mu} > \mathbf{0}$ satisfying:

- $\lambda_i > 1$ implies $f_{iN} = r_i^-$, and $\lambda_i < 1$ implies $f_{iN} = r_i^+$.
- $\mu_j > 1$ implies $f_{Mj} = c_i^-$, and $\mu_j < 1$ implies $f_{Mj} = c_j^+$.

An axiomatic justification is provided in [105] to show that the above definition is a reasonable method of describing the proportional matrix rescaling. In addition, it is shown that such a fair share matrix uniquely exists if and only if $R^+(\boldsymbol{p}, \boldsymbol{\sigma}) \neq \emptyset$.

Given a matrix $\boldsymbol{p} \geq 0$, let S be the set of index (i, j) for which $p_{ij} > 0$, and $\bar{S}$ be its complement. Consider the following optimization problem:

$$
\begin{aligned}
\text{find} \quad & x_{ij} \\
\min \quad & \sum_{j=1}^{N}\sum_{i=1}^{M} x_{ij}\log\frac{x_{ij}}{p_{ij}} \\
\text{s.t.} \quad & r_i^- \le x_{iN} \le r_i^+ \\
& c_j^- \le x_{Mj} \le c_j^+ \\
& x_{MN} = h \\
& x_{ij} \ge 0, \forall (i,j) \in S \\
& x_{ij} = 0, \forall (i,j) \in \bar{S}
\end{aligned}
\tag{4.37}
$$

Then, the following theorem holds:

Theorem 4.6.1 *The solution to the optimization problem (4.37) is the fair share matrix of problem* $(\boldsymbol{p}, \boldsymbol{\sigma})$.

Proof The solution to the optimization problem (4.37), denoted as $\boldsymbol{f}$, minimizes the Lagrangian:

$$
\begin{aligned}
L(\boldsymbol{x}, \boldsymbol{\alpha}^-, \boldsymbol{\alpha}^+, \boldsymbol{\beta}^-, \boldsymbol{\beta}^+, \nu) = & \sum_S x_{ij}\log\frac{x_{ij}}{p_{ij}} + \sum_M \alpha_i^-(r_i^- - x_{iN}) \\
& + \sum_M \alpha_i^+(x_{iN} - r_i^+) + \sum_N \beta_j^-(c_j^- - x_{Mj}) \\
& + \sum_N \beta_j^+(x_{Mj} - c_j^+) + \nu(h - x_{MN})
\end{aligned}
\tag{4.38}
$$

where the non-negative Karush-Kuhn-Tucker (KKT) multipliers $\boldsymbol{\alpha}^-, \boldsymbol{\alpha}^+$ for the row constraints, $\boldsymbol{\beta}^-, \boldsymbol{\beta}^+$ for the column constraints, and ν for the constraint of the sum of all matrix entries. By the first order conditions, we have

$$
\begin{aligned}
& f_{ij} = p_{ij}\exp\{\alpha_i^- - \alpha_i^+ + \beta_j^- - \beta_j^+ + \nu - 1\} \text{ for } (i,j) \in S \\
& f_{ij} = 0 \text{ for } (i,j) \in \bar{S}
\end{aligned}
$$

Let $\lambda_i = \exp\{\alpha_i^- - \alpha_i^+\}$, $\mu_j = \exp\{\beta_j^- - \beta_j^+ i\}$, and $\delta = \exp\{\nu - 1\}$. Then, we have

$$f_{ij} = \delta \lambda_i p_{ij} \mu_j$$

If $\lambda_i > 1$, then necessarily $\alpha_i^- > 0$, which implies $f_{iN} = r_i^-$ by the KKT conditions; if $\lambda_i < 1$, $\alpha_i^+ > 0$ and $f_{iN} = r_i^+$. The column constraints can be interpreted analogously. Therefore, by definition, $\boldsymbol{f}$ is a fair share matrix.

Moreover, since the objective function of optimization problem (4.37) is strictly convex, the fair share matrix $\boldsymbol{f}$ is a unique solution. ■

In the context of identity management, we aim to find the colum stochastic fair share matrix of an $L \times N$ positive matrix. Correspondingly, in the optimization problem (4.37), we have $r_i^- = 0$, $r_i^+ = 1$, $c_j^- = c_j^+ = 1$, and natrually $h = N$. If the matrix is square, i.e., $L = N$, it is obvious that the fair share matrix must also satisfy the row stochastic constraint, and the optimization problem (4.37) is equivalent to Sinkhorn rescaling.

4.6.2 GMPHD Covariance Update with Measurement Origin Uncertainty

We first define some commonly-used notations for this section:

n dimension of the target state
K_t number of measurements at time t
p_D probability of detection
V_D volume of an ellipsoid whose gate size is $\sqrt{D}$
V_G the size of surveillance area

Define the following events for the k-th measurement $z_k(t) \in \mathcal{Z}(t)$ at time t:

- $E_{T,t}^k$: the event that $z_k(t)$ is originated from a target.
- $E_{F,t}^k$: the event that $z_k(t)$ is originated from a false measurement (i.e., clutter).

Assume the number of false measurements K is subjected to a Poisson distribution with density λ:

$$\mu_F(K) = \frac{(\lambda V_G)^K}{K!} e^{-\lambda V_G} \tag{4.39}$$

Let $D_{i,k}(t)$ be the normalized distance square (NDS) between the k-th measurement and the i-th predicted Gaussian component at time t:

$$D_{i,k}(t) = \nu_{i,k}^T(t) S_i^{-1}(t) \nu_{i,k}(t) \tag{4.40}$$

where $\nu_{i,k}(t) = z_k(t) - H_t m_i(t|t-1)$ whose covariance $S_i(t) = H_t P_i(t|t-1) H_t^T + R_t$. It can be shown that $D_{i,k}(t)$ is subjected to the following Gaussian distribution:

$$N(D_{i,k}(t)) = \frac{1}{\sqrt{(2\pi)^n |S_i(t)|}} e^{-\frac{D_{i,k}(t)}{2}} \tag{4.41}$$

Then, the conditional probability $p_{i,k}(t) = p(E_{T,t}^k | D_{i,k}(t), K_t)$ is given by:

$$p_{i,k}(t) = \frac{f(D_{i,k}(t), E_{T,t}^k, K_t)}{f(D_{i,k}(t), E_{T,t}^k, K_t) + f(D_{i,k}(t), E_{F,t}^k, K_t)} \tag{4.42}$$

The joint distribution $f(D_{i,k}(t), E_{T,t}^k, K_t)$ is calculated as:

$$\begin{aligned} f(D_{i,k}(t), E_{T,t}^k, K_t) = & \binom{K_t - 1}{k-1} \left(\frac{D_{i,k}(t)}{V_G}\right)^{\frac{n}{2}(k-1)} \left(1 - \left(\frac{D_{i,k}(t)}{V_G}\right)^{\frac{n}{2}}\right)^{K_t - k} \\ & \mu_F(K_t - 1) \frac{n V_{D_{i,k}(t)}}{2 D_{i,k}(t)} N(D_{i,k}(t)) U(D_{i,k}(t); (0, V_G]) p_D \end{aligned} \tag{4.43}$$

where $U(x; S)$ is a unit step function such that:

$$U(x; S) = \begin{cases} 1 & if\ x \in S \\ 0 & otherwise \end{cases} \tag{4.44}$$

The joint distribution $f(D_{i,k}(t), E_{F,t}^k, K_t)$ is calculated as:

$$\begin{aligned} f(D_{i,k}(t), E_{F,t}^k, K_t) = & (1 - p_D) f_{c_k}(D_{i,k}(t) | K_t) \mu_F(K_t) \\ & + p_D (1 - p_R(D_{i,k}(t))) f_{c_k}(D_{i,k}(t) | K_t - 1) \mu_F(K_t - 1) \\ & + p_D p_R(D_{i,k}(t)) f_{c_{k-1}}(D_{i,k}(t) | K_t - 1) \mu_F(K_t - 1) \end{aligned} \tag{4.45}$$

where $p_R(D_{i,k}(t))$ is the probability that a target exists in the region with size $\sqrt{D_{i,k}(t)}$ [114]. The conditional distribution $f_{c_k}(D_{i,k}(t)|K_t)$ is given by:

$$f_{c_k}(D_{i,k}(t)|K_t) = \frac{nK_t}{2D_{i,k}(t)} \binom{K_t - 1}{k-1} \left(\frac{D_{i,k}(t)}{V_G}\right)^{\frac{nk}{2}} \left(1 - \left(\frac{D_{i,k}(t)}{V_G}\right)^{\frac{n}{2}}\right)^{K_t - k} \tag{4.46}$$

Provided the probability $p_{i,k}(t)$ in equation (4.42), we can get the new GMPHD covariance update as:

$$\begin{aligned} P_{i,k}(t) = p_{i,k}(t)P_i(t|E^k_{T,t}) + (1 - p_{i,k}(t))P_{i,k}(t|E^k_{F,t}) \\ + p_{i,k}(t)(1 - p_{i,k}(t))K_i(t)\nu_{i,k}(t)\nu^T_{i,k}K^T_i(t) \end{aligned} \tag{4.47}$$

where $P_i(t|E^k_{T,t}) = (I - K_i(t)H_t)P_i(t|t-1)$ is the same as the standard Kalman filter covariance update; $P_{i,k}(t|E^k_{F,t})$ is given by:

$$P_{i,k}(t|E^k_{F,t}) = P_i(t|t-1) + (\alpha_{i,k}(t) - 1)K_i(t)S_i(t)K^T_i(t) \tag{4.48}$$

where

$$\begin{aligned} \alpha_{i,k}(t) &= \frac{\begin{array}{c}\lambda(1 - p_D C_T)V_{D_{i,k}(t)}(V_G - V_{D_{i,k}(t)}) + p_D C_T(1 - p_R(D_{i,k}(t)) \\ (K_t - k)V_{D_{i,k}(t)} + p_D p_R(D_{i,k}(t))C_T(k-1)(V_G - V_{D_{i,k}(t)})\end{array}}{\begin{array}{c}\lambda(1 - p_D)V_{D_{i,k}(t)}(V_G - V_{D_{i,k}(t)}) + p_D(1 - p_R(D_{i,k}(t)) \\ (K_t - k)V_{D_{i,k}(t)} + p_D p_R(D_{i,k}(t))(k-1)(V_G - V_{D_{i,k}(t)})\end{array}} \\ C_T &= \frac{\int_0^{D_{i,k}(t)} q^{n/2} e^{-q/2} dq}{n \int_0^{D_{i,k}(t)} q^{n/2-1} e^{-q/2} dq} \end{aligned} \tag{4.49}$$

The proof of equations (4.43), (4.45) and (4.49) are established in our previous work [102].

5. SUMMARY

In this research, we studied problems in the multi-target tracking and identity management (MTIM) area. Particularly, we proposed solutions to three MTIM issues and demonstrated our solutions with illustrative numerical examples.

First, for the problem of tracking multiple targets with a limited sensor range and occasional target loss, we developed a new multi-target state estimator for mobile sensors which considers the information gain/loss from lost targets. A guidance objective function is formulated to minimize the tracking uncertainty of all the targets. The objective function is essentially following such strategy that whenever it is unable to cover all the targets, the sensor tries to cover some of the targets while leaving the rest lost, and goes back to track the lost target(s) later. To keep track of the lost targets, the proposed algorithm uses a Gaussian-sum based estimation technique. The estimation technique is able to provide relatively accurate position estimates for lost targets at a low computational cost. It was shown that the proposed tracking algorithm can be unified with the standard Kalman filter if a linear target motion model is applied.

Second, for the problem of seeking target identity information in MTIM, we proposed a decentralized sensor scheduling algorithm which selects targets to identify by solving an optimization problem to reduce the identity information uncertainty. We developed an algorithm for the secondary sensor to efficiently identify the targets while considering the time cost for the secondary sensor to reach the selected target for identification. The sensor optimization and control of the secondary sensor, namely, which target to be identified and how to control the sensor to identify the selected target, are formulated as an optimization problem to minimize the uncertainty of the target identities subject to the sensor assignments and dynamics constraints. In addition, the control of the secondary sensor is designed to be robust to abrupt

algorithm termination due to the nonexistence of an optimal solution and state estimation error. Moreover, the proposed algorithm is a general solution regardless of the types of targets or sensors.

Third, we proposed a new identity management algorithm, called GMPHD-IM algorithm, which is able to simultaneously track and manage identities of an unknown and time-varying number of targets in clutter. The GMPHD-IM algorithm can actively manage the identities of the detected targets over time by maintaining the target identity belief through a recursive propagation-update procedure. The propagation of the target identity belief requires the target-measurement association output from multi-target tracking algorithms. For this, we used a modified-covariance GMPHD (MC-GMPHD) filter and developed a method to calculate the target-measurement association from the weights of the Gaussian mixture terms without introducing additional computational complexity. Whenever there is available target identity information (i.e., local information) on some targets, the algorithm updates the target identity belief with the information such that the uncertainties (represented by the statistical entropy) in the identities of all the targets are reduced. The target identity belief, together with the target state estimates generated by the MC-GMPHD filter, forms the output of the algorithm.

REFERENCES

REFERENCES

[1] Y. Bar-Shalom, "Multitarget-multisensor tracking: advanced applications," *Norwood, MA, Artech House, 1990, 391 p.*, 1990.

[2] S. Challa and G. W. Pulford, "Joint target tracking and classification using radar and esm sensors," *IEEE transactions on Aerospace and Electronic Systems*, vol. 37, no. 3, pp. 1039–1055, 2001.

[3] E. P. Blasch, *Derivation of a belief filter for high range resolution radar simultaneous target tracking and identification*. Wright State University, 1999.

[4] D. E. Clark and J. Bell, "Multi-target state estimation and track continuity for the particle phd filter," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 43, no. 4, 2007.

[5] H. Sidenbladh, "Multi-target particle filtering for the probability hypothesis density," *arXiv preprint cs/0303018*, 2003.

[6] T. E. Fortmann, Y. Bar-Shalom, and M. Scheffe, "Multi-target tracking using joint probabilistic data association," in *Decision and Control including the Symposium on Adaptive Processes, 1980 19th IEEE Conference on*. IEEE, 1980, pp. 807–812.

[7] F. Bourgault, T. Furukawa, and H. F. Durrant-Whyte, "Optimal search for a lost target in a bayesian world," in *Field and service robotics*. Springer, 2006, pp. 209–222.

[8] T. Furukawa, F. Bourgault, B. Lavis, and H. F. Durrant-Whyte, "Recursive bayesian search-and-tracking using coordinated uavs for lost targets," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*. IEEE, 2006, pp. 2521–2526.

[9] H. Chao, Y. Cao, and Y. Chen, "Autopilots for small unmanned aerial vehicles: A survey," *International Journal of Control, Automation and Systems*, vol. 8, no. 1, pp. 36–44, 2010.

[10] I. Hwang, H. Balakrishnan, K. Roy, J. Shin, L. Guibas, and C. Tomlin, "Multiple-target tracking and identity management," in *Sensors, 2003. Proceedings of IEEE*, vol. 1. IEEE, 2003, pp. 36–41.

[11] I. Hwang, K. Roy, H. Balakrishnan, and C. Tomlin, "A distributed multiple-target identity management algorithm in sensor networks," in *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, vol. 1. IEEE, 2004, pp. 728–734.

[12] R. P. Mahler, "Multitarget bayes filtering via first-order multitarget moments," *IEEE Transactions on Aerospace and Electronic systems*, vol. 39, no. 4, pp. 1152–1178, 2003.

[13] B.-N. Vo and W.-K. Ma, "The gaussian mixture probability hypothesis density filter," *IEEE Transactions on signal processing*, vol. 54, no. 11, pp. 4091–4104, 2006.

[14] H. W. Sorenson and D. L. Alspach, "Recursive bayesian estimation using gaussian sums," *Automatica*, vol. 7, no. 4, pp. 465–479, 1971.

[15] D. L. Alspach and H. W. Sorenson, "Nonlinear bayesian estimation using gaussian sum approximations," *Automatic Control, IËEE Transactions on*, vol. 17, no. 4, pp. 439–448, 1972.

[16] D. Mušicki and R. J. Evans, "Measurement gaussian sum mixture target tracking," in *Information Fusion, 2006 9th International Conference on*. IEEE, 2006, pp. 1–8.

[17] I. Bilik and J. Tabrikian, "Maneuvering target tracking in the presence of glint using the nonlinear gaussian mixture kalman filter," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 46, no. 1, pp. 246–262, 2010.

[18] K. Wyffels and M. Campbell, "Negative information for occlusion reasoning in dynamic extended multiobject tracking," *IEEE Transactions on Robotics*, vol. 31, no. 2, pp. 425–442, 2015.

[19] N. Bergman, "Recursive bayesian estimation," *Department of Electrical Engineering, Linköping University, Linköping Studies in Science and Technology. Doctoral dissertation*, vol. 579, 1999.

[20] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Fluids Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[21] S. J. Julier and J. K. Uhlmann, "New extension of the kalman filter to nonlinear systems," in *AeroSense'97*. International Society for Optics and Photonics, 1997, pp. 182–193.

[22] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," *Signal Processing, IEEE Transactions on*, vol. 50, no. 2, pp. 174–188, 2002.

[23] N. Gordon, D. Salmond, and C. Ewing, "Bayesian state estimation for tracking and guidance using the bootstrap filter," *Journal of Guidance, Control, and Dynamics*, vol. 18, no. 6, pp. 1434–1443, 1995.

[24] A. Ryan and J. K. Hedrick, "Particle filter based information-theoretic active sensing," *Robotics and Autonomous Systems*, vol. 58, no. 5, pp. 574–584, 2010.

[25] V. N. Dobrokhodov, I. I. Kaminer, K. D. Jones, and R. Ghabcheloo, "Vision-based tracking and motion estimation for moving targets using unmanned air vehicles," *Journal of guidance, control, and dynamics*, vol. 31, no. 4, pp. 907–917, 2008.

[26] C. Teuliere, L. Eck, and E. Marchand, "Chasing a moving target from a flying uav," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*. IEEE, 2011, pp. 4929–4934.

[27] T. Furukawa, L. C. Mak, H. Durrant-Whyte, and R. Madhavan, "Autonomous bayesian search and tracking, and its experimental validation," *Advanced Robotics*, vol. 26, no. 5-6, pp. 461–485, 2012.

[28] H. Yu, R. W. Beard, M. Argyle, and C. Chamberlain, "Probabilistic path planning for cooperative target tracking using aerial and ground vehicles," in *American Control Conference (ACC), 2011*. IEEE, 2011, pp. 4673–4678.

[29] K. Cook, E. Bryan, H. Yu, H. Bai, K. Seppi, and R. Beard, "Intelligent cooperative control for urban tracking," *Journal of Intelligent & Robotic Systems*, vol. 74, no. 1-2, pp. 251–267, 2014.

[30] B. Grocholsky, J. Keller, V. Kumar, and G. Pappas, "Cooperative air and ground surveillance," *IEEE Robotics & Automation Magazine*, vol. 13, no. 3, pp. 16–25, 2006.

[31] A. Sinha, T. Kirubarajan, and Y. Bar-Shalom, "Autonomous surveillance by multiple cooperative uavs," in *Optics & Photonics 2005*. International Society for Optics and Photonics, 2005, pp. 59 131V–59 131V.

[32] J. Capitan, L. Merino, and A. Ollero, "Cooperative decision-making under uncertainties for multi-target surveillance with multiples uavs," *Journal of Intelligent & Robotic Systems*, pp. 1–16, 2015.

[33] H. Oh, S. Kim, H.-s. Shin, and A. Tsourdos, "Coordinated standoff tracking of moving target groups using multiple uavs," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 51, no. 2, pp. 1501–1514, 2015.

[34] S. Ragi and E. K. Chong, "Decentralized guidance control of uavs with explicit optimization of communication," *Journal of Intelligent & Robotic Systems*, vol. 73, no. 1-4, pp. 811–822, 2014.

[35] E. Adamey and U. Ozguner, "A decentralized approach for multi-uav multitarget tracking and surveillance," in *SPIE Defense, Security, and Sensing*. International Society for Optics and Photonics, 2012, pp. 838 915–838 915.

[36] Z. Tang and U. Ozguner, "Motion planning for multitarget surveillance with mobile sensor agents," *IEEE Transactions on Robotics*, vol. 21, no. 5, pp. 898–908, 2005.

[37] N. Miller and J. Rogers, "Simultaneous tracking of multiple ground targets from a single multirotor uav," in *AIAA Atmospheric Flight Mechanics Conference*, 2014, p. 2670.

[38] J. Ousingsawat and M. E. Campbell, "Optimal cooperative reconnaissance using multiple vehicles," *Journal of Guidance Control and Dynamics*, vol. 30, no. 1, pp. 122–132, 2007.

[39] G. M. Hoffmann, S. L. Waslander, and C. J. Tomlin, "Distributed cooperative search using information-theoretic costs for particle filters, with quadrotor applications," in *Proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2006, pp. 21–24.

[40] G. M. Hoffmann and C. J. Tomlin, "Mobile sensor network control using mutual information methods and particle filters," *IEEE Transactions on Automatic Control*, vol. 55, no. 1, pp. 32–47, 2010.

[41] V. Jilkov, X. Li, and D. DelBalzo, "Best combination of multiple objectives for uav search & track path optimization," in *Information Fusion, 2007 10th International Conference on*, 2007, pp. 1 – 8.

[42] R. R. Pitre, X. R. Li, and R. Delbalzo, "Uav route planning for joint search and track missions: An information-value approach," *Aerospace and Electronic Systems, IEEE Transactions on*, no. 3, pp. 2551–2565, 2012.

[43] A. Sinha, T. Kirubarajan, and Y. Bar-Shalom, "Autonomous ground target tracking by multiple cooperative uavs," in *2005 IEEE Aerospace Conference*. IEEE, 2005, pp. 1–9.

[44] S. M. Ross, *Introduction to Probability Models*. Academic press, 2014.

[45] J. L. Williams, "Gaussian mixture reduction for tracking multiple maneuvering targets in clutter," AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING AND MANAGEMENT, Tech. Rep., 2003.

[46] J. L. Williams and P. S. Maybeck, "Cost-function-based gaussian mixture reduction for target tracking," in *Proceedings of the sixth international conference of Information fusion*, vol. 2, 2003, pp. 1047–1054.

[47] D. J. Salmond, "Mixture reduction algorithms for target tracking," in *State Estimation in Aerospace and Tracking Applications, IEE Colloquium on*. IET, 1989, pp. 7–1.

[48] A. R. Runnalls, "Kullback-leibler approach to gaussian mixture reduction," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 43, no. 3, 2007.

[49] R. M. Gray, "Entropy and information," in *Entropy and Information Theory*. Springer, 1990, pp. 21–55.

[50] J. Arora, *Introduction to Optimum Design*. Academic Press, 2004.

[51] X. R. Li and V. P. Jilkov, "Survey of maneuvering target tracking. part i. dynamic models," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 39, no. 4, pp. 1333–1364, 2003.

[52] D. Reid, "An algorithm for tracking multiple targets," *IEEE transactions on Automatic Control*, vol. 24, no. 6, pp. 843–854, 1979.

[53] S. S. Blackman, "Multiple hypothesis tracking for multiple target tracking," *IEEE Aerospace and Electronic Systems Magazine*, vol. 19, no. 1, pp. 5–18, 2004.

[54] K.-C. Chang and Y. Bar-Shalom, "Joint probabilistic data association for multitarget tracking with possibly unresolved measurements and maneuvers," *IEEE Transactions on Automatic Control*, vol. 29, no. 7, pp. 585–594, 1984.

[55] R. J. Fitzgerald, "Development of practical pda logic for multitarget tracking by microprocessor," in *American Control Conference, 1986.* IEEE, 1986, pp. 889–898.

[56] N. Bergman and A. Doucet, "Markov chain monte carlo data association for target tracking," in *Acoustics, Speech, and Signal Processing, 2000. ICASSP'00. Proceedings. 2000 IEEE International Conference on*, vol. 2. IEEE, 2000, pp. II705–II708.

[57] S. Oh, S. Sastry, and L. Schenato, "A hierarchical multiple-target tracking algorithm for sensor networks," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on.* IEEE, 2005, pp. 2197–2202.

[58] S. Oh, S. Russell, and S. Sastry, "Markov chain monte carlo data association for multi-target tracking," *IEEE Transactions on Automatic Control*, vol. 54, no. 3, pp. 481–497, 2009.

[59] Y. Bar-Shalom, P. K. Willett, and X. Tian, *Tracking and Data Fusion.* YBS publishing, 2011.

[60] M. Mallick, S. Coraluppi, C. Carthel, V. Krishnamurthy, and B. Vo, "Multitarget tracking using multiple hypothesis tracking," in *Integrated Tracking, Classification, and Sensor Management: Theory and Applications.* Wiley Online Library, 2012, pp. 165–201.

[61] S. J. Davey and D. A. Gray, "Integrated track maintenance for the pmht via the hysteresis model," *IEEE transactions on Aerospace and Electronic Systems*, vol. 43, no. 1, pp. 93–111, 2007.

[62] B.-N. Vo, B.-T. Vo, and D. Phung, "Labeled random finite sets and the bayes multi-target tracking filter," *IEEE Transactions on Signal Processing*, vol. 62, no. 24, pp. 6554–6567, 2014.

[63] S. Maskell, M. Briers, and R. Wright, "Fast mutual exclusion," in *Signal and Data Processing of Small Targets 2004*, vol. 5428. International Society for Optics and Photonics, 2004, pp. 526–536.

[64] P. Horridge and S. Maskell, "Real-time tracking of hundreds of targets with efficient exact jpdaf implementation," in *2006 9th International Conference on Information Fusion.* IEEE, 2006, pp. 1–8.

[65] K. Romeo, D. F. Crouse, Y. Bar-Shalom, and P. Willett, "The jpdaf in practical systems: Approximations," in *Signal and Data Processing of Small Targets 2010*, vol. 7698. International Society for Optics and Photonics, 2010, p. 76981I.

[66] B.-N. Vo, B.-T. Vo, and H. G. Hoang, "An efficient implementation of the generalized labeled multi-bernoulli filter," *IEEE Transactions on Signal Processing*, vol. 65, no. 8, pp. 1975–1987, 2016.

[67] M. Beard, B. T. Vo, and B.-N. Vo, "A solution for large-scale multi-object tracking," *arXiv preprint arXiv:1804.06622*, 2018.

[68] J. Shin, L. J. Guibas, and F. Zhao, "A distributed algorithm for managing multi-target identities in wireless ad-hoc sensor networks," in *Information Processing in Sensor Networks*. Springer, 2003, pp. 223–238.

[69] I. Hwang, H. Balakrishnan, K. Roy, and C. Tomlin, "Multiple-target tracking and identity management with application to aircraft tracking," *Journal of guidance, control, and dynamics*, vol. 30, no. 3, pp. 641–653, 2007.

[70] S. Oh, I. Hwang, K. Roy, and S. Sastry, "A fully automated distributed multiple-target tracking and identity management algorithm," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2005, p. 5838.

[71] S. Oh, I. Hwang, and S. Sastry, "Distributed multitarget tracking and identity management," *Journal of guidance, control, and dynamics*, vol. 31, no. 1, pp. 12–29, 2008.

[72] H. Balakrishnan, I. Hwang, and C. J. Tomlin, "Polynomial approximation algorithms for belief matrix maintenance in identity management," in *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, vol. 5. IEEE, 2004, pp. 4874–4879.

[73] J. Liu, M. Chu, and J. E. Reich, "Multitarget tracking in distributed sensor networks," *IEEE Signal Processing Magazine*, vol. 24, no. 3, pp. 36–46, 2007.

[74] M. Bocca, O. Kaltiokallio, N. Patwari, and S. Venkatasubramanian, "Multiple target tracking with rf sensor networks," *IEEE Transactions on Mobile Computing*, vol. 13, no. 8, pp. 1787–1800, 2014.

[75] I. J. Cox and S. L. Hingorani, "An efficient implementation of reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 18, no. 2, pp. 138–150, 1996.

[76] R. Sinkhorn, "Diagonal equivalence to matrices with prescribed row and column sums," *The American Mathematical Monthly*, vol. 74, no. 4, pp. 402–405, 1967.

[77] X. R. Li and V. P. Jilkov, "Survey of maneuvering target tracking. part v. multiple-model methods," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 41, no. 4, pp. 1255–1321, 2005.

[78] P. Dames, "Distributed multi-target search and tracking using the phd filter," in *2017 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*. IEEE, 2017, pp. 1–8.

[79] S. Oh, L. Schenato, P. Chen, and S. Sastry, "Tracking and coordination of multiple agents using sensor networks: System design, algorithms and experiments," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 234–254, 2007.

[80] S. Papaioannou, P. Kolios, T. Theocharides, C. G. Panayiotou, and M. Polycarpou, "Jointly-optimized searching and tracking with random finite sets," *IEEE Transactions on Mobile Computing*, 2019.

[81] Y. Fu, Q. Ling, and Z. Tian, "Distributed sensor allocation for multi-target tracking in wireless sensor networks," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, no. 4, pp. 3538–3553, 2012.

[82] A. K. Gostar, R. Hoseinnezhad, and A. Bab-Hadiashar, "Multi-bernoulli sensor control for multi-target tracking," in *2013 IEEE Eighth International Conference on Intelligent Sensors, Sensor Networks and Information Processing*. IEEE, 2013, pp. 312–317.

[83] S. Papaioannou, P. Kolios, T. Theocharides, C. G. Panayiotou, and M. M. Polycarpou, "Probabilistic search and track with multiple mobile agents," in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2019, pp. 253–262.

[84] C. Kreucher, A. O. Hero, and K. Kastella, "A comparison of task driven and information driven sensor management for target tracking," in *Proceedings of the 44th IEEE Conference on Decision and Control*. IEEE, 2005, pp. 4004–4009.

[85] J. L. Williams, J. W. Fisher III, and A. S. Willsky, "Sensor management for multiple target tracking with heterogeneous sensor models," in *Signal Processing, Sensor Fusion, and Target Recognition XV*, vol. 6235. International Society for Optics and Photonics, 2006, p. 62350F.

[86] C. E. Shannon, "A mathematical theory of communication," *Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[87] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[88] D. P. Bertsekas, D. P. Bertsekas, D. P. Bertsekas, and D. P. Bertsekas, *Dynamic programming and optimal control*. Athena scientific Belmont, MA, 1995, vol. 1, no. 2.

[89] M. T. Spaan, "Partially observable markov decision processes," in *Reinforcement Learning*. Springer, 2012, pp. 387–414.

[90] O. Sundström, D. Ambühl, and L. Guzzella, "On implementation of dynamic programming for optimal control problems with final state constraints," *Oil & Gas Science and Technology–Revue de lInstitut Français du Pétrole*, vol. 65, no. 1, pp. 91–102, 2010.

[91] N. A. Shneydor, *Missile guidance and pursuit: kinematics, dynamics and control*. Elsevier, 1998.

[92] T. Shima and O. M. Golan, "Head pursuit guidance," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 5, pp. 1437–1444, 2007.

[93] K. Ogata, *Modern control engineering*. Prentice hall India, 2002, vol. 4.

[94] H. K. Khalil, *Nonlinear control*. Prentice Hall, 2014.

[95] A. O. Hero and D. Cochran, "Sensor management: Past, present, and future," *IEEE Sensors Journal*, vol. 11, no. 12, pp. 3064–3075, 2011.

[96] M. Jiang, W. Yi, and L. Kong, "Multi-sensor control for multi-target tracking using cauchy-schwarz divergence," in *2016 19th International Conference on Information Fusion (FUSION)*. IEEE, 2016, pp. 2059–2066.

[97] G. H. Elkaim, F. A. P. Lie, and D. Gebre-Egziabher, "Principles of guidance, navigation, and control of uavs," *Handbook of Unmanned Aerial Vehicles*, pp. 347–380, 2015.

[98] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[99] Y. Bar-Shalom and X.-R. Li, *Multitarget-multisensor tracking: principles and techniques*. YBs London, UK:, 1995, vol. 19.

[100] K. Panta, D. E. Clark, and B.-N. Vo, "Data association and track management for the gaussian mixture probability hypothesis density filter," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 45, no. 3, pp. 1003–1016, 2009.

[101] M. Kushwaha, S. Oh, I. Amundson, X. Koutsoukos, and A. Ledeczi, "Target tracking in heterogeneous sensor networks using audio and video sensor fusion," in *2008 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*. IEEE, 2008, pp. 14–19.

[102] D. Kim, C. Kwon, and I. Hwang, "Gaussian mixture probability hypothesis density filter against measurement origin uncertainty," *Signal Processing*, p. 107448, 2020.

[103] C. Zhang and I. Hwang, "Multi-target identity management with decentralized optimal sensor scheduling," *European Journal of Control*, 2020.

[104] M. Idel, "A review of matrix scaling and sinkhorn's normal form for matrices and positive maps," *arXiv preprint arXiv:1609.06349*, 2016.

[105] M. L. Balinski and G. Demange, "An axiomatic approach to proportionality between matrices," *Mathematics of Operations Research*, vol. 14, no. 4, pp. 700–719, 1989.

[106] ——, "Algorithms for proportional matrices in reals and integers," *Mathematical Programming*, vol. 45, no. 1-3, pp. 193–210, 1989.

[107] S. Mehrotra, "On the implementation of a primal-dual interior point method," *SIAM Journal on optimization*, vol. 2, no. 4, pp. 575–601, 1992.

[108] R. P. Mahler, *Statistical multisource-multitarget information fusion*. Artech House Norwood, MA, 2007, vol. 685.

[109] ——, "A theoretical foundation for the stein-winter probability hypothesis density (phd) multitarget tracking approach," ARMY RESEARCH OFFICE ALEXANDRIA VA, Tech. Rep., 2000.

[110] M. Yazdian-Dehkordi and Z. Azimifar, "Refined gm-phd tracker for tracking targets in possible subsequent missed detections," *Signal Processing*, vol. 116, pp. 112–126, 2015.

[111] H. Zhang, L. Gao, M. Xu, and Y. Wang, "An improved probability hypothesis density filter for multi-target tracking," *Optik*, vol. 182, pp. 23–31, 2019.

[112] B.-T. Vo, B.-N. Vo, and A. Cantoni, "Analytic implementations of the cardinalized probability hypothesis density filter," *IEEE transactions on signal processing*, vol. 55, no. 7, pp. 3553–3567, 2007.

[113] X. R. Li, "Tracking in clutter with strongest neighbor measurements. i. theoretical analysis," *IEEE Transactions on Automatic Control*, vol. 43, no. 11, pp. 1560–1578, 1998.

[114] T. L. Song, D. G. Lee, and J. Ryu, "A probabilistic nearest neighbor filter algorithm for tracking in a clutter environment," *Signal Processing*, vol. 85, no. 10, pp. 2044–2053, 2005.

VITA

VITA

Chiyu Zhang received his B.S. degree in Aeronautics and Astronautics from Tsinghua University, Beijing, China in 2013, and his M.S. degrees in Aeronautics and Astronautics from Purdue University in 2015. He began to pursue a doctorate degree in Aeronautics and Astronautics at Purdue University since 2015. His research focuses on multi-target tracking, estimation and identity management, and guidance/control/planning of mobile sensors on robotics and unmanned vehicles.