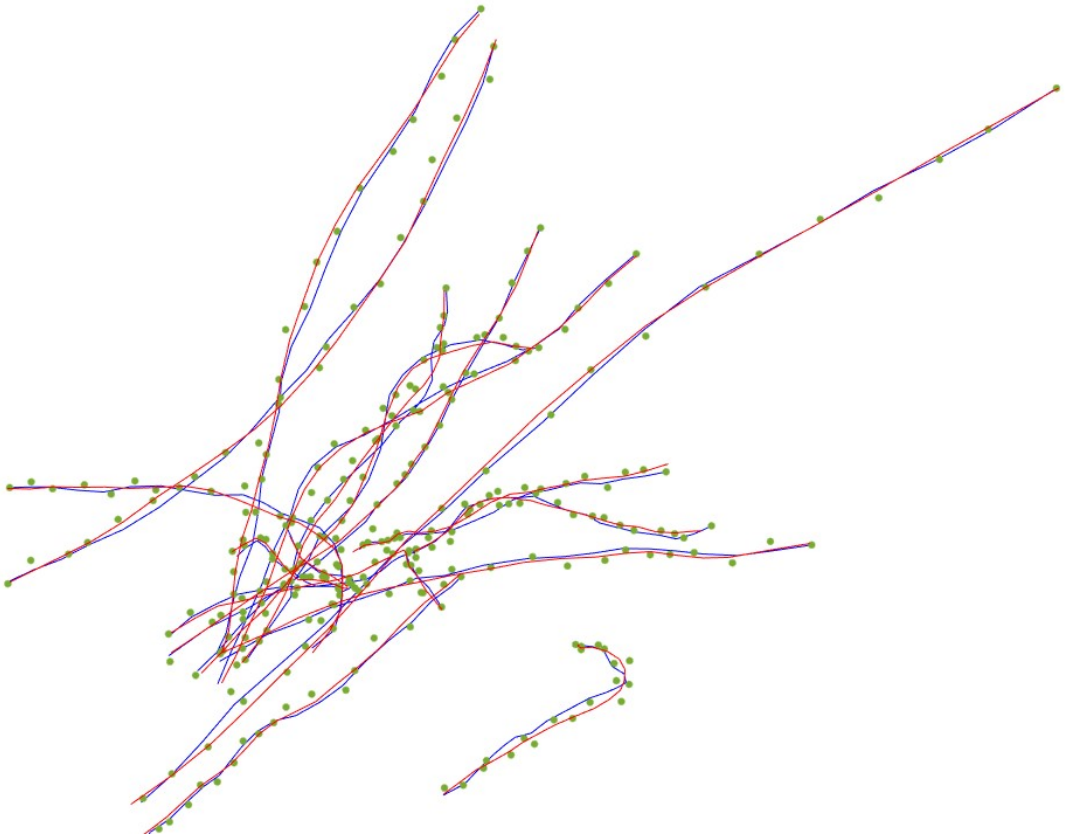


Multi-Object Tracking Based on Multi-Sensor Multi-Bernoulli Densities Fusion

Master's thesis in Systems, Control and Mechatronics

JINGKAI ZHOU
QINGHUAN LIU

Department of Electrical Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2023
www.chalmers.se

Master's Thesis 2023

Multi-Object Tracking Based on Multi-Sensor Multi-Bernoulli Densities Fusion

JINGKAI ZHOU
QINGHUAN LIU

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Division of Signal Processing and Biomedical Engineering Chalmers
University of Technology
Gothenburg, Sweden 2023

Multi-Object Tracking Based on Multi-Sensor Multi-Bernoulli Densities Fusion
JINGKAI ZHOU
QINGHUAN LIU

Supervisor: Yuxuan Xia, Department of Electrical Engineering
Daniel Johansson, Volvo Trucks
Felix Gustavsson, Volvo Trucks
Alexander Greger, Volvo Trucks
Examiner: Lennart Svensson, Department of Electrical Engineering

Master's Thesis 2023
Department of Electrical Engineering
Division of Signal Processing and Biomedical Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Sweden
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2023

Multi-Object Tracking Based on Multi-Sensor Multi-Bernoulli Densities Fusion
JINGKAI ZHOU
QINGHUAN LIU
Department of Electrical Engineering
Division of Signal Processing and Biomedical Engineering
Chalmers University of Technology

Abstract

Advanced Driver Assistance Systems (ADAS) utilizes multiple sensors to help drivers deal with complicated traffic situations. In practice, we need multi-sensor algorithms that can accurately estimate object states, such as positions and velocities, while operating under a limited computational budget.

In this thesis, we propose a multi-sensor fusion algorithm using multi-Bernoulli densities to obtain the fused multi-object densities and estimate the object trajectories. The density of each object is estimated from the single object state fusion, such as covariance intersection, safe fusion, etc. First, we perform data association to associate the densities from different sensors that correspond to the same object. Then, based on the most likely association, the multi-object fusion problem can be decomposed into several single object state fusions. Once the fused multi-object densities are obtained, object trajectories, if desired, can be constructed by backward simulation without object label information. The efficacy of this work has been validated using simulation data as well as real data collected by a Volvo testing truck.

The result shows that our proposed algorithm can accurately associate data in the sense of fewer missed and false detections, achieve accurate multi-sensor multi-object fusion, and the response time is also acceptable.

Keywords: Multi-object tracking, Data association, Sensor fusion, Trajectory estimation.

Acknowledgements

We would like to thank our supervisor, Yuxuan Xia, at Chalmers University of Technology, for his valuable advice and comprehensive discussions with us during the thesis. We would also like to thank our supervisors, Daniel Johansson, Felix Gustavsson and Alexander Greger, at Volvo Trucks. They have given us important empirical advice on data processing and have handled many complex data, without which our thesis would have taken much longer. We would like to thank Shenting Xie and Sixuan Pei for being our opponents in our final presentation and giving us valuable feedback on our thesis. Finally, we would like to express our sincere gratitude to our examiner Lennart Svensson for his suggestions, which have contributed to the rigour of this report.

Jingkai Zhou & Qinghuan Liu, Gothenburg, June 2023

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

AA	Arithmetic Average
ADAS	Advanced Driver Assistance Systems
CC	Cross Covariance
CF	Centralized Fusion
CI	Covariance Intersection
CT	Coordinate Turn
CV	Constant Velocity
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
EKF	Extended Kalman Filter
FoVs	Field of Views
GOSPA	Generalized Optimal Sub-Pattern Assignment Metric
KF	Kalman Filter
KLD	Kullback-Leibler Divergence
MAP	Maximum a Posterior
MB	Multi-Bernoulli
MBM	Multi-Bernoulli Mixture
MCMC	Markov Chain Monte Carlo
MOT	Multi-Object Tracking
NLL	Negative Log-Likelihood
PDF	Possibility Density Function
PMB	Poison Multi-Bernoulli
PMBM	Poison Multi-Bernoulli Mixture
PPP	Poisson Points Process
RFSs	Random Finite Sets
RTS	Rauch-Tung-Striebel
SF	Safe Fusion
SOT	Single-Object Tracking
SVD	Singular Value Decomposition
T2TF	Track-to-Track Fusion

Nomenclature

Below is the nomenclature of indices, sets, parameters, and variables that have been used throughout this thesis.

Indices

i,j	Indices for objects, trajectories, hypothesis and weight matrices
m,n,u,v	Indices for entries in the matrix
k	Time instance

Sets

$\mathbf{X}$	Set of trajectories
π	Multi-trajectory density
γ	Assignment between estimates and ground truth
Γ	Set of all possible assignments between estimates and ground truth

Parameters

A	Dynamic matrix
Q	Dynamic noise matrix
H	Measurement matrix
R	Measurement noise matrix
c	Maximum allowable localization error in GOSPA
α	Factor for the cardinality mismatching penalty in GOSPA
p	Factor for the outliers penalty in GOSPA

Variables

x	Object state
P	Object state's uncertainty
q	Object dynamic noise
y	Measurement of object's state
e	Measurement noise
K	Kalman gain
C,C'	Cost matrix for solving the 2D assignment problem
a	Activator of each entry in cost matrix (can only be 1 or 0)
w	Weight of the local hypothesis
r	Existence probability of the local hypothesis
X	Single object trajectory
p	Single trajectory density of the local hypothesis

Contents

List of Figures

List of Tables

1
Introduction

This chapter provides an introduction to the thesis. Following that, the aim and objective are stated to provide clarity on the specific problem that the thesis seeks to address and the approach to be taken. Additionally, a review of related work is conducted, discussing the strengths and weaknesses of existing approaches and establishing the relevance of this thesis in relation to them. Furthermore, a comprehensive explanation of the limitations and scope of this thesis is provided. Lastly, the final section offers an outline of the thesis, providing a structured overview of its subsequent sections and content.

1.1 Background

Advanced Driver Assistance Systems (ADAS) is the technology that helps drivers control and navigate their vehicles more safely and comfortably. ADAS uses sensors such as cameras and radars to monitor the surroundings, detect hazards, and provide timely warnings or automated actions. ADAS features include adaptive cruise control, lane departure warning, forward collision warning, and driver monitoring systems. Within the mentioned strategies, the fusion of multi-object densities for multi-object tracking (MOT) is the key technic.

The vehicle's perceptual capabilities can be enhanced by using multiple sensors, enabling a more advanced perception system. In the fusion of multi-object densities, two architectures are commonly employed: Centralized Fusion (CF) and Track-to-Track Fusion (T2TF) [1]. CF architecture often faces computational limitations as it necessitates the central processor to handle the processing of all raw data. In contrast, T2TF architecture integrates track estimation, typically in the form of object densities with ID information, obtained from processing the sensor's detections instead of directly incorporating raw detections [2].

In the fusion of multi-object densities, it is important to avoid double-counting the information from different sensors. Additionally, effectively handling densities originating from different fields of view (FoVs) poses a significant challenge. These requirements give rise to a data association problem, which can be approached by solving a 2D assignment problem. By finding the optimal assignment with the minimum total assignment cost, an object detected by one sensor can be mapped to its corresponding object detected by another sensor. This assignment is commonly referred to as a "fusion map" [3].

According to the best fusion map, the assigned densities from different sensors can be combined into a fused object density. Several widely employed methods for single-object fusion exist, such as cross-covariance, covariance intersection, and covariance union [4]. These methods facilitate the integration of information from different sensors, resulting in a consolidated representation of the object.

In the context of MOT, it is insufficient to solely fuse the (unlabeled) object densities within individual time instances. Estimating these objects' trajectories over a specific time interval is necessary. In single-sensor tracking scenarios, trajectories are typically inferred based on object labels. However, this approach encounters challenges when dealing with multiple sensors, as the same object may be assigned different labels by different sensors, a problem commonly known as label mismatch. To overcome this issue, it is crucial to develop a trajectory estimation method capable of handling sets of unlabeled multi-object densities. Our objective is to construct a flexible and less restrictive fusion for the MOT problem.

1.2 Related Work

This section contains several widely-used methods related to data association and multi-object densities fusion.

1.2.1 T2TF

T2TF has emerged as a promising approach for improving the robustness and accuracy of multi-sensor MOT systems. T2TF techniques aim to fuse information from multiple object tracks to enhance object association and tracking performance. T2TF relies on the use of single-object fusion methods and several of them can be implemented, e.g. covariance intersection [5] [6] [7], safe fusion [8] [9], arithmetic average fusion [10] [11] and cross-covariance fusion [7] [12]. In [7], covariance intersection and cross-covariance fusion are compared. The simulation result shows that the cross-covariance fusion returned the best fusion quality w.r.t. root mean square error, and the runtime of the cross-covariance fusion is the lowest. In [9], the safe fusion is compared to the covariance intersection, and the result shows that the safe fusion obtains estimates with better consistency.

1.2.2 Data Association Techniques

Data association techniques play a crucial role in T2TF. Most data association problems are treated as 2D linear assignment problems, where the best assignment can be found using, e.g. the Hungarian algorithm [13]. The main differences between these assignment problems exist in the formation of the cost matrix. In addition, as seen from [14], the association problem also contains information on object labels. Moreover, some clustering methods can also be implemented for data association.

These techniques all aim to associate the data using the information within single-time instances.

Using historical information, the Multisensor Track-To-Track Association method [15] associates tracks based on the concept of "historical distance". The historical distance between two tracks contains past estimates to make the track association more reasonable. The algorithm clusters the tracks that correspond to the same object by finding the minimal historical distance in an iterative fashion.

1.2.3 Deep Learning-based Approaches

Deep learning-based approaches have also been applied to T2TF in MOT [16]. These methods leverage the power of deep neural networks to learn feature representations and perform track fusion. In this survey [17], track associations are determined based on similarity measures or distance metrics. Deep learning-based T2TF methods have shown promising results in handling complex MOT scenarios and achieving state-of-the-art performance. Aiming at the fusion of multiple trajectory densities using deep learning, a transformer-based fusion method is introduced to output a set of fused global object densities [18].

1.3 Aim

The aim of this thesis is to develop a fusion structure for multi-sensor multi-Bernoulli densities and enable the tracking of multiple objects. The structure incorporates techniques such as data association, single-object densities fusion, and object trajectory estimation.

Firstly, the fusion structure needs to accurately and effectively estimate the densities of multiple objects in the surrounding environment. These densities, describing the moving objects within (different) sensor FoVs, are then fused together under the correct data association. Secondly, we evaluate four single-object fusion methods using appropriate evaluation metrics and select the most effective one for subsequent trajectory estimation tasks. Finally, utilizing the fused multi-object densities, we employ backward simulation to estimate the smoothed trajectories of the objects.

1.4 Scope and Limitations

This thesis is subject to certain constraints, which may affect the thesis's outcomes. Firstly, the raw data from the sensors is inaccessible, limiting the project in the domain of T2TF. Secondly, the trajectory estimation can only perform partially online. To clarify, assuming at a time of interest in a time interval, trajectory estimation requires that the fused multi-object densities in this time interval are all accessible. Moreover, only moving objects are considered when processing the sensor and reference data, while stationary objects are excluded through pre-processing. Addition-

ally, the proposed structure operates under the assumption of time synchronization of the sensor data, achieved during an inaccessible pre-processing step.

1.5 Thesis Outline

The remainder of the master thesis is structured in the following manner. Chapter 2 introduces the relevant theories employed in our thesis, including Bayesian estimation theory, random finite set theory, etc. Chapter 3 details the methodologies employed in our fusion structure, encompassing the pre-processing step, data association methods, single-object fusion techniques, trajectory estimation methods, and evaluation methods. Chapter 4 outlines the simulation setups and evaluation results under different scenarios. Chapter 5 presents the results obtained from real-world reference data. Subsequently, Chapter 6 discusses the result of the simulation and real-world testing. Finally, Chapter 7 presents the conclusion.

2
Theory

This chapter presents the background theory, including the state space representation, Bayesian estimation and object density representation related to MOT algorithms.

2.1 State Space Modelling

State space representation is a mathematical modelling technique used in control theory, physics, engineering, and many other fields to describe the behaviour of a system over time. In this thesis, the object's behaviour is represented by a state vector that describes the object's current state and how this object evolves over time.

2.1.1 Object State Representation

In this thesis, we would like to estimate multiple objects' positions and velocities in two dimensions under Cartesian coordinates. Therefore, we write the state x_k of an object at time k and represent it as a vector with an uncertainty matrix in the form of Gaussian,

$$\mu_k = \begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \end{bmatrix}, P_k = diag(\begin{bmatrix} \sigma_x & \sigma_{\dot{x}} & \sigma_y & \sigma_{\dot{y}} \end{bmatrix}), \tag{2.1}$$

where μ represents the mean of this state density, and P represents its uncertainty. In addition, we denote the set of objects existing at time k as $\boldsymbol{x}_k$.

2.1.2 Motion Model

Given a state of an object at time instance k, the object state at the next time instance $k+1$ can be predicted by using a motion model. In general, this motion model can be represented as an affine function.

$$x_{k+1} = f\left(x_k, u_k, q_k\right), \tag{2.2}$$

where f is the dynamic function, u_k is the control input at time k, and q_k represents the noise input.

For linear and Gaussian motion models, (2.2) can be represented in the matrix form

$$x_{k+1} = A_k x_k + B_k u_k + q_k, \quad q_k \sim \mathcal{N}(0, Q_k), \tag{2.3}$$

where A_k is the dynamic matrix at time k, B_k is the input matrix and Q_k denotes the noise matrix.

2.1.3 Measurement Model

A measurement model is a representation of the state of the system being observed, which is used to estimate the state of the system based on the sensor measurements obtained from the environment. The equation for the measurement model at time k is generally given as

$$y_k = h(x_k, e_k), \tag{2.4}$$

where h represents the function that transforms the states into measurements, and e_k represents the measurement noise.

Under the assumption of linearity and Gaussian distribution, the matrix representation of the measurement process is,

$$y_k = H x_k + e_k, \quad e_k \sim \mathcal{N}(0, R_k), \tag{2.5}$$

where H is the measurement matrix, which maps the object state space into the measurement space, and R_k is the measurement noise matrix.

2.2 Bayesian Estimation

In object tracking, Bayesian estimation is a common approach used to estimate the state of a moving object based on noisy sensor measurements. To apply Bayesian estimation in MOT, the probabilistic models of the object's motion should be first developed, i.e. the motion and measurement densities aforementioned in Subsections 2.1.2 and 2.1.3. It is assumed that we have a prior distribution over the object state, which captures the knowledge about its probable location and motion before observing the new measurements. As new measurements come in, the object's prior distribution will be updated using Bayes' rule. This results in a posterior distribution that describes the updated probability of the object's state incorporating the measurement information. This posterior updating process can be represented as a Bayesian network.

In Figure 2.1, we make two assumptions. First, the current state at time k is only dependent on the previous state at time $k-1$. That is, the object moves according to a Markov process. Second, the current measurement is only dependent on the current state. Therefore, $p(x_k|x_{1:k-1})$ and $p(y_k|x_{1:k})$ can be, respectively, simplified as

$$p(x_k|x_{1:k-1}) = p(x_k|x_{k-1}), \tag{2.6}$$

and

$$p(y_k|x_{1:k}) = p(y_k|x_k). \tag{2.7}$$

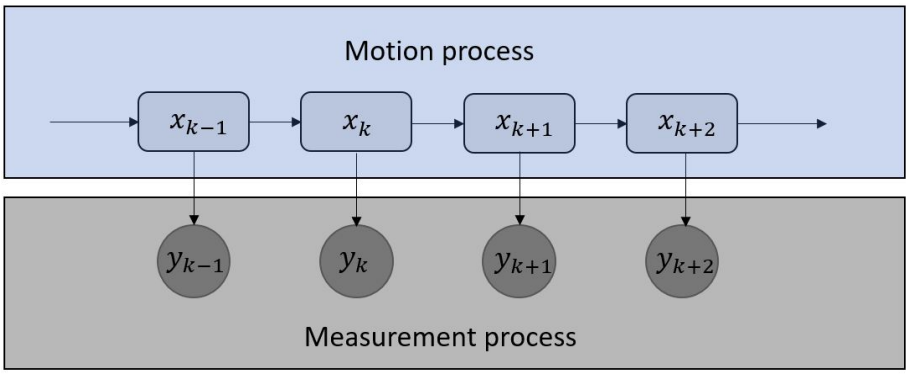


Figure 2.1: A Bayesian network with motion and measurement processes that are conditionally dependent.

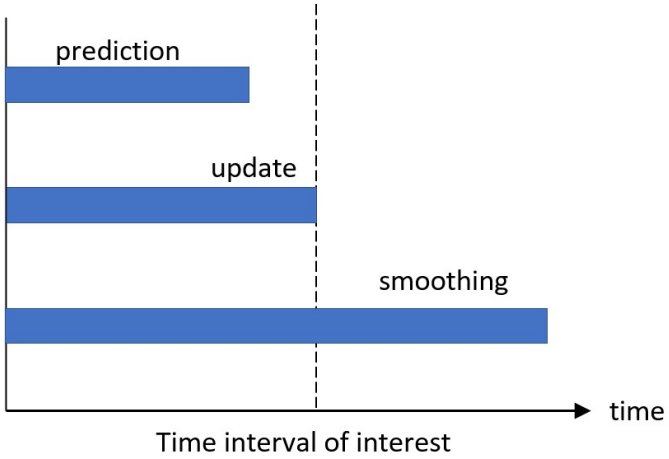


Figure 2.2: Relationship between state's prediction, update, and smoothing in time sequence.

2.2.1 Prediction, Update, and Smoothing

The Bayes filter recursively computes the object filtering density by performing prediction and update. After obtaining all the object filtering estimates in a given time interval, Bayesian smoothing is a tool that uses the motion model to further improve the accuracy of object state estimation at earlier time steps. Figure 2.2 shows the time relation diagram on prediction, update, and smoothing.

In the prediction step, the object state is predicted given the previous posterior density $p\left(x_{k-1}|y_{1:k-1}\right)$ at time $k-1$ using the Chapman-Kolmogorov equation. The explicit expression is

$$\begin{aligned} p\left(x_k|y_{1:k-1}\right) &= \int p\left(x_{k-1}, x_k|y_{1:k-1}\right) dx_{k-1} \\ &= \int p\left(x_k|x_{k-1}, y_{1:k-1}\right) p\left(x_{k-1}|y_{1:k-1}\right) dx_{k-1} \\ &= \int p\left(x_k|x_{k-1}\right) p\left(x_{k-1}|y_{1:k-1}\right) dx_{k-1}, \end{aligned} \tag{2.8}$$

where $p\left(x_k|x_{k-1}\right)$ describes the motion model (2.3).

After prediction, the object state is updated with the measurement y_k received at time k using the Bayes rule. The measurement update equation is

$$\begin{aligned} p\left(x_k|y_{1:k}\right) &= p\left(x_k|y_k, y_{1:k-1}\right) \\ &= \frac{p\left(x_k|y_k, y_{1:k-1}\right)}{p\left(y_k|y_{1:k-1}\right)} p\left(x_k|y_{1:k-1}\right) \qquad (2.9) \\ &\propto p\left(y_k|x_k, y_{1:k-1}\right) p\left(x_k|y_{1:k-1}\right) \\ &= p\left(y_k|x_k\right) p\left(x_k|y_{1:k-1}\right), \qquad (2.10) \end{aligned}$$

where $p\left(y_k|x_k\right)$ is the measurement likelihood parameterized by (2.6), $p\left(y_k|y_{1:k-1}\right)$ is the normalizing constant, and $p\left(x_k|y_{1:k-1}\right)$ is the predicted density obtained from (2.8).

After performing recursive predicting and updating using all the measurements in the time interval $1:T$, smoothing seeks to find the density $p\left(x_k|y_{1:T}\right)$ at time k where $k \leq T-1$. Some common approaches for smoothing are RTS smoothing, two-filter smoothing and backward simulation. In this thesis, we perform backward simulation by recursively drawing samples from the backward kernel. From the Markov property, given x_{k+1}, x_k is independent from the measurement sequence $y_{k+1:T}$. As a consequence, the backward smoothing kernel $p(x_k|x_{k+1}, y_{1:T})$ can be written as $p(x_k|x_{k+1}, y_{1:k})$, which can be further expressed as

$$\begin{aligned} p\left(x_k|x_{k+1}, y_{1:k}\right) &= \frac{p\left(x_k, x_{k+1}|y_{1:k}\right)}{p\left(x_{k+1}|y_{1:k}\right)} \\ &= \frac{p\left(x_{k+1}|x_k, y_{1:k}\right) p\left(x_k|y_{1:k}\right)}{p\left(x_{k+1}|y_{1:k}\right)} \\ &= \frac{p\left(x_{k+1}|x_k\right) p\left(x_k|y_{1:k}\right)}{p\left(x_{k+1}|y_{1:k}\right)}. \qquad (2.11) \end{aligned}$$

2.2.2 Kalman Filter

Kalman filter (KF) provides the Bayesian optimal filtering solution for linear and Gaussian models. In Kalman filtering, both the predicted and filtering densities are Gaussian, parameterized by mean $\hat{x}_{k|k'}$ and covariance $P_{k|k'}$, where $k' = k-1$ for prediction and $k' = k$ for update. The prediction step is given by

$$\hat{x}_{k|k-1} = Ax_{k-1}, \qquad (2.12)$$

$$P_{k|k-1} = AP_{k-1}A^T + Q, \qquad (2.13)$$

The update step is given as

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} K_k v_k, \tag{2.14}$$
$$P_{k|k} = P_{k|k-1} - K_k S_k K_k^T, \tag{2.15}$$
$$K_k = P_{k|k-1} H_k^T S_k^{-1}, \tag{2.16}$$
$$v_k = y_k - H_k \hat{x}_{k|k-1}, \tag{2.17}$$
$$S_k = H_k P_{k|k-1} H_k^T + R_k, \tag{2.18}$$

where K_k is the Kalman gain, v_k is the innovation, and S_k is the innovation covariance.

2.2.3 Backward Simulation for Smoothing

In a time interval $1:T$, smoothing makes use of the measurement information in the entire time interval. One approach to performing smoothing is using backward simulation, which employs a forward-backward idea. First, the object filtering densities are obtained in the forward filtering direction. Then particle representation of smoothed object densities is obtained by sequentially drawing samples from a backward smoothing kernel. In general, the joint posterior density $p\left(x_{1:T} \mid y_{1:T}\right)$ can be factorized as

$$p\left(x_{1:T} \mid y_{1:T}\right) = \left(\prod_{k=1}^{T-1} p\left(x_k \mid x_{k+1}, y_{1:k}\right)\right) p\left(x_T \mid y_{1:T}\right), \tag{2.19}$$

where $p\left(x_k \mid x_{k+1}, y_{1:k}\right)$ is the backward kernel. After initializing the density $p\left(x_T \mid y_{1:T}\right)$ at the last time instance T, the process of backward simulation can be done by drawing samples of x_k from the kernel $p\left(x_k | x_{k+1}, y_{1:k}\right)$ successively from time $T-1$ to time 1.

For the linear Gaussian dynamic model, the backward simulation kernel can be written in matrix form, which follows,

$$p\left(x_k \mid x_{k+1}, y_{1:k}\right) = \mathcal{N}\left(x_k; \mu_k, M_k\right) \tag{2.20}$$

with simulated mean vector and covariance matrix,

$$\mu_k = \hat{x}_{k|k} + P_{k|k} A^\top E_k^{-1} \left(x_{k+1} - A\hat{x}_{k|k}\right), \tag{2.21}$$
$$M_k = P_{k|k} - P_{k|k} A^\top E_k^{-1} A P_{k|k}, \tag{2.22}$$
$$P_{k+1|k} = Q + A P_{k|k} A^\top, \tag{2.23}$$

where $\hat{x}_{k|k}$ and $P_{k|k}$, $k \in 1 \ldots T$ are the densities obtained from forward filtering.

2.3 Random Finite Sets

Random finite sets (RFSs) are random variables used to represent sets with a finite number of elements. The elements in an RFS are order-irrelevant and element

non-repeatable, e.g. $X_1 = \{a, b\}$ is identical to $X_2 = \{b, a\}$ and one cannot write $X_3 = \{a, b, a\}$. In addition, the set operations like intersection $\cap$, union $\cup$, and cardinality $|\cdot|$ are all applicable to RFS.

In MOT, both the number of objects and their states may change over time and need to be estimated, making RFS an appealing tool for building mathematical models. Two building blocks for RFS-based MOT methods are the Poisson point process (PPP) and the Bernoulli RFS.

2.3.1 Poisson Point Process

For a PPP, its cardinality follows a Poisson distribution, meaning that events are randomly distributed and occur independently. The multi-object density of a PPP $\mathbf{x} = \{x_1, ..., x_n\}$ is [19]

$$f(\mathbf{x}) = e^{-\lambda} \prod_{i=1}^{n} \lambda f(x_i), \tag{2.24}$$

parameterized by the Poisson rate λ and the single object density $f(x_i)$. Note that since the elements in a PPP are i.i.d. their single object densities are the same. In MOT, PPP is often used to model newborn objects and clutter measurements.

2.3.2 Bernoulli RFS

A Bernoulli RFS $\mathbf{x}$ has density [19]

$$f(\mathbf{x}) = \begin{cases} 1 - r, & \mathbf{x} = \emptyset \\ rf(x), & \mathbf{x} = \{x\} \\ 0, & |\mathbf{x}| \geq 2 \end{cases}, \tag{2.25}$$

where $r \in [0, 1]$ represents the existence probability of an object, $f(x)$ is the single-object density of this object.

2.3.3 Multi-Bernoulli and Multi-Bernoulli Mixture

A multi-Bernoulli is a disjoint union of independent Bernoulli RFSs. Its density can be written as

$$f(\mathbf{x}) = \sum_{\uplus_{i \in \mathbb{I}} \mathbf{x}_i = \mathbf{x}} \prod_{i \in \mathbb{I}} f_i(\mathbf{x}_i), \tag{2.26}$$

where each Bernoulli RFS is indexed by $i \in \mathbb{I}$, and $\uplus$ denotes the disjoint union.

The Multi-Bernoulli Mixture (MBM) RFSs are used to model the set of the detected objects under data association uncertainties, where each mixture component represents a different data association hypothesis. The multi-object density of an MBM is written as [20]

$$f(\mathbf{x}) = \sum_{j \in \mathbb{J}} w_j \sum_{\uplus_{i \in \mathbb{I}_j} \mathbf{x}_i = \mathbf{x}} \prod_{i \in \mathbb{I}_j} f_{j,i}(\mathbf{x}_i), \tag{2.27}$$

where w_j is the weight of each MB indexed by $j \in \mathbb{J}$, and it satisfies that $\sum_{j \in \mathbb{J}} w_j = 1$.

2.3.4 Poisson Multi-Bernoulli Mixture

A Poisson Multi-Bernoulli Mixture (PMBM) is a disjoint union of a PPP and an MBM [20], and its multi-object density can be expressed as

$$f(\mathbf{x}) = \sum_{\mathbf{x}^u \uplus \mathbf{X}^d = \mathbf{x}} f^u\left(\mathbf{x}^u\right) f^d\left(\mathbf{x}^d\right), \tag{2.28}$$

where $\mathbf{x}^u$ represents the PPP undetected objects with density (2.24), and $\mathbf{x}^d$ represents the MBM detected objects with density (2.27). If there is only a single MB component in the MBM (i.e. $|\mathbb{J}| = 1$), then the PMBM reduces to a PMB.

2.3.5 PMBM filter and PMB filter

PMBM provides a closed-form solution for multiple object filtering with standard multi-object models with PPP birth [21] [22]. The resulting filter is the PMBM filter, and the PMB filter is an efficient approximation of the PMBM filter [22]. In the PMBM density representation, the PPP describes the potential objects that are hypothesized to exist but have never been detected, and the MBM describes targets that have been detected at least once.

2.4 Set of Trajectories

A trajectory is a sequence of object states at consecutive time steps. A single trajectory can be represented as

$$X = \left(t, x^{1:\nu}\right), \tag{2.29}$$

where t is the trajectory start time and ν represents the length of this trajectory. The density of single-trajectory density is denoted as $\pi\left(t, x^{1:\nu}\right)$, where $\pi\left(\cdot\right)$ is a real-value function that integrates to one. Similar to the representation of RFSs, the set of trajectories in a time interval $\alpha:\gamma$ can be represented as

$$\mathbf{X}_{\alpha:\gamma} = \{X_1, X_2, ..., X_n\}. \tag{2.30}$$

The PPP of the $\mathbf{X}_{\alpha:\gamma}$ has density

$$\pi\left(\mathbf{X}_{\alpha:\gamma}\right) = e^{-\langle\lambda,1\rangle}[\lambda(\cdot)]^{\mathbf{X}_{\alpha:\gamma}}, \tag{2.31}$$

where $\lambda(\cdot)$ is the Poisson intensity, $\langle f, g\rangle$ denotes the inner product $\int f(x)g(x)dx$, and the Bernoulli process of the $\mathbf{X}_{\alpha:\gamma}$ has density

$$\pi\left(\mathbf{X}_{\alpha:\gamma}\right) = \begin{cases} 1-r, & \mathbf{X}_{\alpha:\gamma} = \emptyset \\ rp(X), & \mathbf{X}_{\alpha:\gamma} = \{X\} \\ 0, & \text{otherwise} \end{cases} \tag{2.32}$$

Similarly, the MB process of the $\mathbf{X}_{\alpha:\gamma}$ has the density in a similar form to (2.26).

3
Methods

This chapter presents the methods employed in our proposed multi-object fusion algorithm, including data association and single-object fusion methods, and then trajectory estimation is performed based on the fused multi-object densities.

The focus of this thesis lies on tracking movable objects around a testing truck with one camera and five radars equipped, a practical challenge at Volvo Trucks. Each sensor individually processes the raw object detections and provides track-level information. The objective is to effectively leverage the information from several sensors to achieve MOT. In the context of T2TF, this requires precise multi-object densities and accurate estimation of object trajectories.

There are several challenges related to this problem, and the first one is related to the data association problem. It is important to associate single-object densities that represent the same object from different sensors. In addition, we observed that in practice the tracking system may report more than one track estimate for objects moving across the edges of different FoVs. Therefore, a clustering algorithm is needed as a pre-processing step to group closely-spaced track estimates for each sensor. Furthermore, in order to enhance the adaptability of this multi-object fusion technique, different single-object fusion methods can be applied to different working scenarios. However, this introduces a challenge in determining a suitable evaluation tool for assigning each method to its specialized scenario.

Currently, most of the multi-object fusion methods only focus on fusing multi-object densities at individual time instances, without providing trajectory information. To construct trajectories in single sensor tracking, one may make use of the labelling information and connect object state estimates with the same unique label. This simple labelling strategy may work well in some cases, but it is not reliable due to track fragmentation and track switches in challenging scenarios. Furthermore, in multi-sensor MOT, the same object may be assigned different labels by different sensors, a problem commonly known as label mismatch. Thus, the labelling information can be misleading and may give rise to unreasonable trajectory estimates. To construct trajectories in the context of multi-sensor fusion, we use backward simulation for sets of trajectories [23], which does not require any labelling information.

To address the mentioned problems, we develop our fusion structure as illustrated in Figure 3.1.

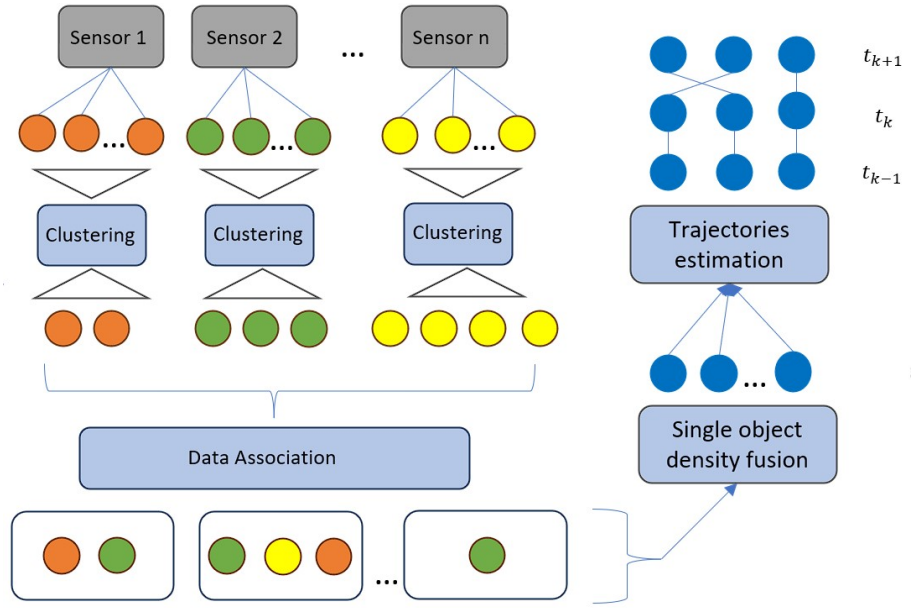


Figure 3.1: A diagram of fusion structure containing clustering, data association, single object fusion and trajectory estimation where each circle represents a single object density.

3.1 Sensory Information

There are six sensors mounted on the testing truck, including five radars and one camera. The approximate layout of sensors is shown in Figure 3.2.

The sensors' feedback is highly integrated, including objects' positions, velocities, accelerations, headings, and their corresponding covariances in Cartesian coordinates. The object dynamic states referring to the environment, e.g. movable or stationary, are also included in the feedback. The information of objects' positions, velocities and their corresponding covariances can be utilised to form the object state representation shown in Subsection 2.1.1.

3.2 Multi-object Fusion

3.2.1 Pre-processing

The output from each sensor captures objects' existence uncertainties via existence probabilities. Therefore, the multi-object density can be regarded as a multi-Bernoulli. In the real data, we observe that the existence probabilities are all comparatively high, usually above 0.9, even around 90% of existence probabilities are above 0.98. Through empirical observation, we found that the object estimates with existence probability below 0.9 typically represent false detections. Therefore, we choose not to take false detections by removing object estimates with existence probability below 0.9 for further implementation.

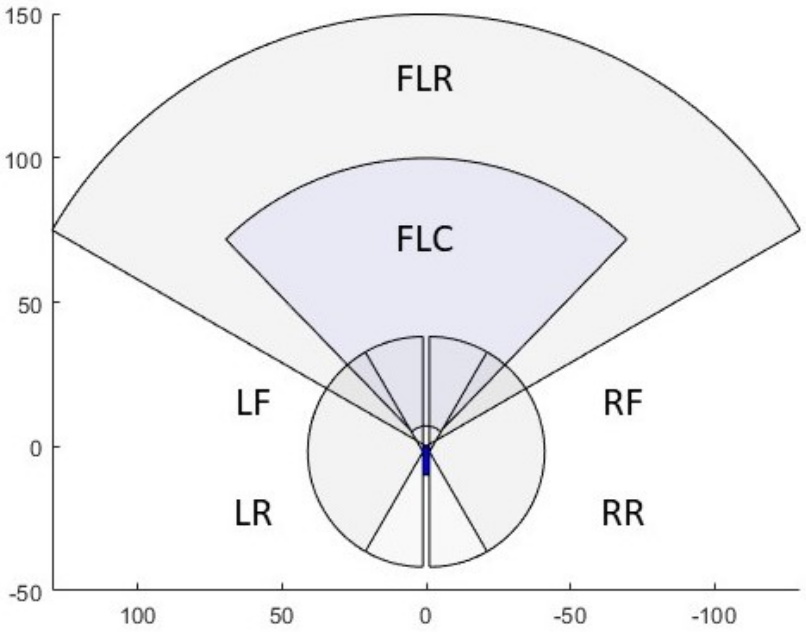


Figure 3.2: The sensor layout. There are six sensors on the testing truck, and the circular segments represent the FoVs of sensors. FLR: Front-looking radar, FLC: front-looking camera, LF: left-front-looking radar, LR:left-rare-looking radar, RF: right-front-looking radar, RR:right-rare-looking radar.

During the sensing process, it is common for an object to be detected more than once when it crosses the edges of different FoVs. To address this, a pre-processing step is employed to cluster sensor data using DBSCAN [24] prior to data association. Unlike many other clustering methods, DBSCAN does not require the number of clusters as an input parameter. DBSCAN identifies clusters based on the minimum number of points τ required to form a dense region and the maximum distance ϵ between two points for them to be considered part of the same cluster.

DBSCAN clusters objects by assigning input objects to core objects, border objects, and noise objects. DBSCAN starts by randomly selecting an object from the set of unassigned input objects and finding all objects within a distance ϵ. The selected objects are marked as core objects if the number of found objects exceeds or equals τ. Next, DBSCAN expands the clustering by iterating all objects within ϵ of the core object and repeating the process until no more objects can be added. Those objects that are not classified as core objects but within ϵ of a core object are marked as border objects and assigned to the cluster. Finally, objects not assigned to any cluster are marked as noise. After assigning all the objects, the process stops.

The characteristics of the data, such as the object densities, influence the choice

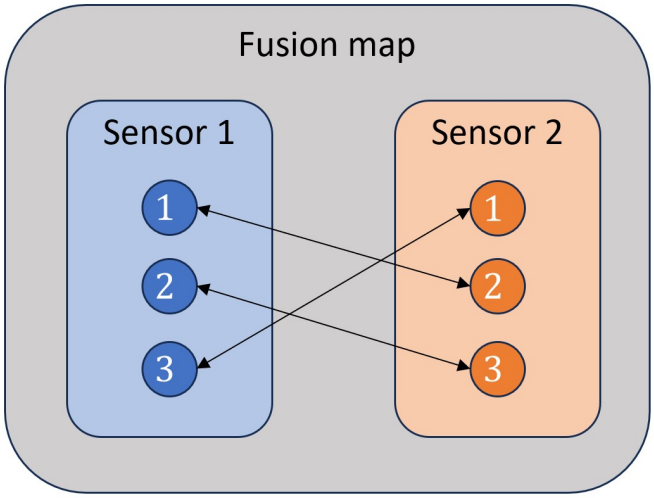


Figure 3.3: An example shows the fusion map where each circle represents an object density, and each double arrow indicates a mapping between one pair of object densities from two different sensors.

of the parameters τ, ϵ. In this thesis, τ is set to 1, meaning each detection may correspond to a single object. The parameter ϵ has been tuned according to the testing scenario.

3.2.2 Data Association

Since the sensors report objects' positions with covariances along with their existence probabilities, we can treat the object estimates from a sensor as a multi-Bernoulli distribution. Then, the association between the object densities from two different sensors is needed for single-object fusion. The data association problem can be addressed by introducing a fusion map indicating the densities' mappings from two sensors, as shown in Figure 3.3. To build the fusion map, the Bernoulli densities from two sensors are measured by the similarity in terms of Kullback-Leibler divergence (KLD) [25]. The best fusion map between two sensors can be found by the sum of the lowest KLDs between pairs of Bernoulli densities, which can be formulated as a 2D assignment problem. For the fusion of more than two sensors, the global optimal fusion map can be found by taking those two steps in pairs in a sequential manner.

The KLD between two distributions $p(x)$ and $q(x)$ is defined as follows:

$$D_{KL}(p||q) = \int p(x) \log \frac{p(x)}{q(x)} dx. \tag{3.1}$$

The value of KLD will be 0 if two single-object deities are identical. If the PDFs of the distributions $p(\cdot)$ and $q(\cdot)$ are in the form of the Bernoulli RFSs density with k dimensional multivariate normal distribution, the KLD between $p(\cdot)$ and $q(\cdot)$ is

given by [26]

$$D_{KL}(p||q) = (1-r_p)\log\frac{1-r_p}{1-r_q} + r_p\log\frac{r_p}{r_q} + \frac{r_p}{2}\left[tr(P_q^{-1}P_p) - \log\frac{|P_p|}{|P_q|} - k + (x_p-x_q)^T P_q^{-1}(x_p-x_q)\right], \quad (3.2)$$

where Bernoulli density $p(\cdot)$ is parameterized by existence probability r_p, mean x_p, and covariance P_p, and Bernoulli density $q(\cdot)$ by existence probability r_q, mean x_q, and covariance P_q. The notation $|\cdot|$ denotes the matrix determinant and $tr(\cdot)$ denotes the trace of the matrix.

Since the KLD is asymmetric, the following symmetrized version of the two KLDs, defined as

$$J(p,q) = \frac{1}{2}\left(D_{KL}(p||q) + D_{KL}(q||p)\right), \quad (3.3)$$

is used to measure the distance between two densities. The cost matrix C is constructed by calculating the KLD between every pair of object densities. Three padded matrixes are added to make the square cost matrix. Assume the cost matrix C is of size $n \times m$, which means there are n single-object densities in sensor 1 and m single-object densities in sensor 2. Three padded matrixes $D^1_{n\times n}$, $D^2_{m\times m}$ and $\mathbf{0}_{m\times n}$ make a square cost matrix. Let C' be the augmented cost matrix in size of $(m+n)\times(m+n)$ as follows:

$$C' = \begin{bmatrix} C_{n\times m} & D^1_{n\times n} \\ D^2_{m\times m} & \mathbf{0}_{m\times n} \end{bmatrix}, \quad (3.4)$$

where the padded matrix D^1 and D^2 are square matrices, where the diagonal elements are threshold while all other entries are set to $+\infty$. To avoid associating densities that are too far away, a threshold, denoting the maximum allowable assignment cost [27], is applied to the cost matrix C'. The threshold is the maximum allowable assignment cost. Then, the best fusion map between sensor 1 and sensor 2 can be found by solving the following optimal assignment problem

$$\begin{aligned} &\arg\min_{\mathrm{a}} \sum_{u=1}^{|\mathbb{I}_1|}\sum_{v=1}^{|\mathbb{I}_2|} a_{u,v}C'_{u,v} \\ &\text{subject to } \sum_{t=1}^{|\mathbb{I}_u|} a_{u,v} = 1, \quad \forall u, \\ &\qquad \sum_{u=1}^{|\mathbb{I}_1|} a_{u,v} = 1, \quad \forall v, \\ &\qquad a_{u,v} \in \{0,1\}, \end{aligned} \quad (3.5)$$

where $C'_{u,v}$ is the entry at the u-th row and v-th column of the cost matrix, and $a_{u,v}$ indicates whether the u-th object and t-th object in the cost matrix C' are associated. Note that only the entries within block C indicate the association between u-th single-object density in sensor 1 and v-th single-object density in sensor 2. The

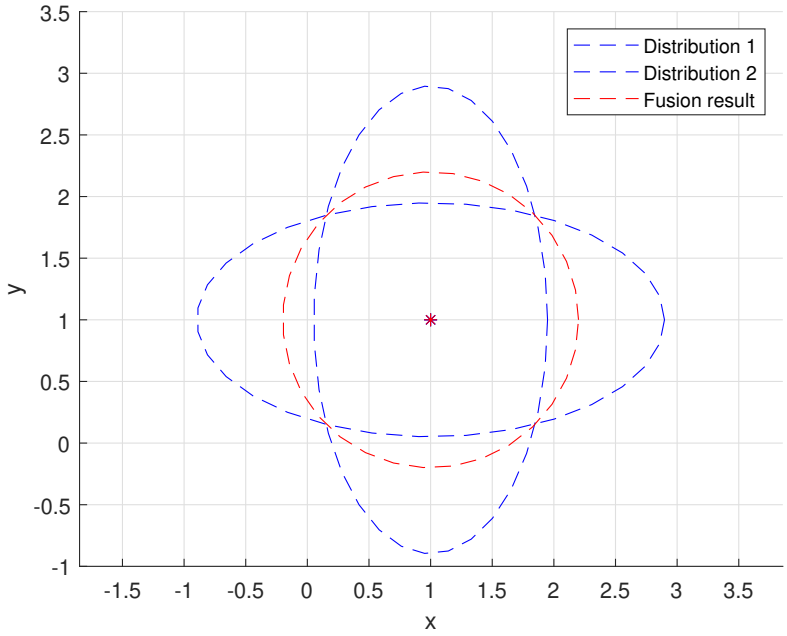


Figure 3.4: The illustration of CI fusion method

association in block D^1 means the unassigned single-object densities in sensor 1, and similarly, the association in block D^2 means the unassigned single-object densities in sensor 2. The unassigned single-object densities may associate with single-object densities from other sensors. If a single-object density remains unassigned after the whole process, then this is regarded as an individual object that has been detected by only one sensor. To solve this 2D assignment problem, we can use, e.g. Munkres' algorithm [28] [29]. Finally, the data association among six sensors can be found by sequentially solving this assignment problem five times in a predetermined order.

3.3 Single-object Density Fusion Algorithms

Four single-object density fusion methods are used to merge single-object densities, which are covariance intersection, safe fusion, arithmetic average, and cross-covariance fusion, respectively. To help understand the differences between these four fusion methods, we compare their fusion results with a toy example: the fusion of two Gaussian distributions with the same mean but different covariances.

3.3.1 Covariance Intersection

Covariance intersection (CI) [5] [6] works for fusing two Gaussian distributions. Given two Gaussian distributions with mean μ_1 and μ_2 and covariance P_1 and P_2 respectively, the fused estimation $\hat{\mu}$ and $\hat{P}$ is formed from the following convex combination,

$$\hat{P}^{-1} = \omega P_1^{-1} + (1-\omega)P_2^{-1}, \tag{3.6}$$

$$\hat{P}^{-1}\hat{\mu} = \omega P_1^{-1}\mu_1 + (1-\omega)P_2^{-1}\mu_2, \tag{3.7}$$

where $\omega \in [0,1]$. Figure 3.4 compares the fusion result with the original distributions. We can see that the fused covariance ellipse crosses four intersection points of two covariance ellipses.

As an extension, CI can also handle cases with the merging of more than two Gaussian distributions. Given N single-object Gaussian densities with mean $\mu_1, \cdots, \mu_N$ and covariance $P_1, \cdots, P_N$ respectively. The mean $\hat{\mu}$ and covariance $\hat{P}$ of the fused Gaussian density can be computed by [6]:

$$\hat{P}^{-1} = \sum_i \omega_i P_i^{-1}, \tag{3.8a}$$

$$\hat{P}^{-1}\hat{\mu} = \sum_i \omega_i P_i^{-1}\mu_i, \tag{3.8b}$$

where w_i is the weight of the i-th Gaussian density, and it satisfies that $\sum_{i=1}^{N} w_i = 1$. The fused existence probability $\hat{r}$ can be obtained as follows:

$$C = \frac{\left|2\pi\hat{P}\right|^{\frac{1}{2}}}{\prod^i |2\pi P_i|^{\frac{\omega_i}{2}}} e^{\left[\frac{1}{2}\hat{\mu}^T\hat{P}^{-1}\hat{\mu} - \sum^i \omega_i \mu_i^T P_i^{-1}\mu_i\right]} \tag{3.9}$$

$$\hat{r} = \frac{C\prod^i r_i^{\omega_i}}{\prod^i (1-r_i)^{\omega_i} + C\prod^i r_i^{\omega_i}} \tag{3.10}$$

where r_i is the existence probability of the i-th Bernoulli density.

3.3.2 Safe Fusion

An intuitive way to fuse two Gaussian densities is to pick one density with the smaller uncertainty, which is the general idea of safe fusion (SF) [8] [9]. The detailed algorithm for fusing two multi-variate Gaussian distributions with mean μ_1 and μ_2 and covariance P_1 and P_2 is shown below,

1. Find the information matrix (inverse of the covariance matrix) $I_1 = P_1^{-1}$ and $I_2 = P_2^{-1}$.
2. Singular value decomposition (SVD): $I_1 = U_1 D_1 U_1^T$.
3. SVD: $D_1^{-1/2} U_1^T I_2 U_1 D_1^{-1/2} = U_2 D_2 U_2^T$.
4. Transformation matrix: $T = U_2^T D_1^{1/2} U_1$.
5. State transformation: $\bar{\mu}_1 = T\mu_1$ and $\bar{\mu}_2 = T\mu_2$. Now the covariance of new distributions are $\bar{P}_1 = I$ and $\bar{P}_2 = D_2^{-1}$ respectively.
6. For each component $i = 1, 2, \cdots, k$, let

$$\bar{\mu}^i = \begin{cases} \bar{\mu}_1^i & \text{if } D_2^{ii} < 1 \\ \bar{\mu}_2^i & \text{if } D_2^{ii} \geq 1 \end{cases}, \tag{3.11a}$$

$$D^{ii} = \begin{cases} 1 & \text{if } D_2^{ii} < 1 \\ D_2^{ii} & \text{if } D_2^{ii} \geq 1 \end{cases}, \tag{3.11b}$$

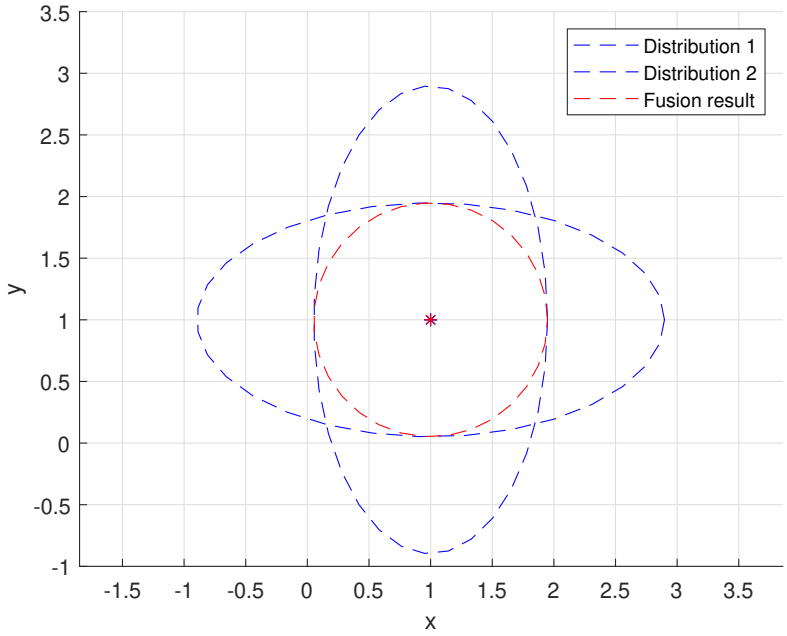


Figure 3.5: The SF result illustration

where D is a diagonal matrix. The superscript ii on D means the i-th element on the diagonal.

7. Inverse the state transformation to get the estimate

$$\hat{\mu} = T^{-1}\bar{\mu}, \tag{3.11c}$$

$$\hat{P} = T^{-1}D^{-1}T^{-T}. \tag{3.11d}$$

We proceed to make explanations on these steps. The operation of SVD in step 2 transforms the first covariance's ellipsoid into a unit circle. The second SVD operation in step 3 rotates the coordinate system, so the second covariance's ellipsoid can be aligned to the coordinate axes. In step 6, the variables with a smaller variance are chosen. In Figure 3.5, we can notice that the fusion result is the combination of variables with a smaller variance.

When there are more than two single-object densities, fusing can be done in a sequential manner, which is order dependent. When fusing Gaussian densities of Bernoulli components, the fusion order can be determined by ranking the object existence probabilities in descending order. For the fusion of existence probability, we use the same method as in the CI fusion method, referring to (3.10).

3.3.3 Arithmetic Average Fusion

The Arithmetic Average (AA) fusion [10] [11] computes the arithmetic average over a finite number of single-object densities. Given two Gaussian distributions with the means, covariances and weights as μ_1 and μ_2, P_1 and P_2 and ω_1 and ω_2 respectively,

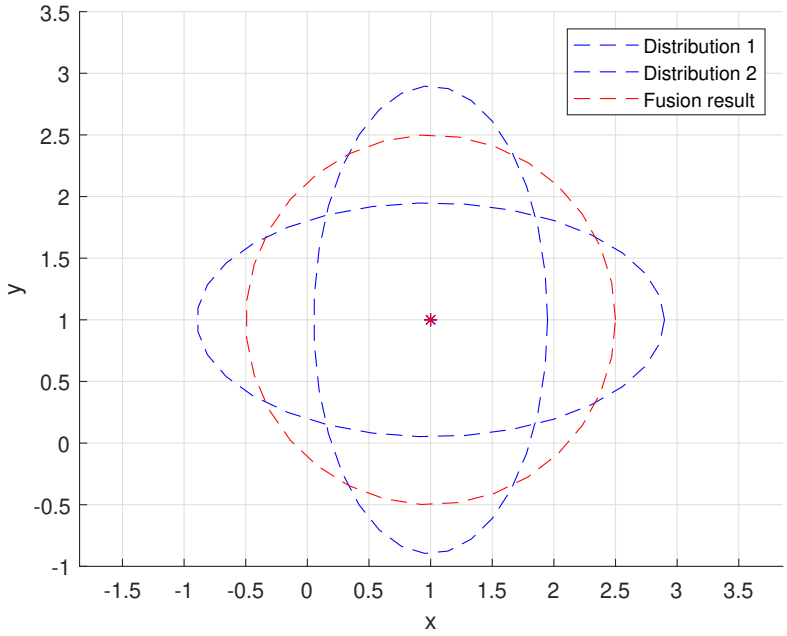


Figure 3.6: The AA result illustration

the fused density $\hat{\mu}$ and $\hat{P}$ is formed from the arithmetic average as follows:

$$\hat{\mu} = \omega_1\mu_1 + \omega_2\mu_2, \tag{3.12a}$$

$$\hat{P} = \omega_1\left(P_1 + (\hat{\mu} - \mu_1)(\hat{\mu} - \mu_1)^T\right) + \omega_2\left(P_2 + (\hat{\mu} - \mu_2)(\hat{\mu} - x_2)^T\right), \tag{3.12b}$$

$$\hat{r} = \omega_1 r_1 + \omega_2 r_2. \tag{3.12c}$$

Figure 3.6 illustrates a fusion example of AA, where it can be seen that the fusion result has larger covariance than each density.

Given N distributions $f_1(x), \cdots, f_N(x)$ with weights $\omega_1, \cdots, \omega_N$, the process of AA with more than two densities is

$$f_{AA}(x) = \sum_i \omega_i f_i(x), \tag{3.13}$$

where $\sum_i w_i = 1$. If there are more than 2 Gaussian distributions, the mean and covariance of the fused Gaussian distribution can be calculated as

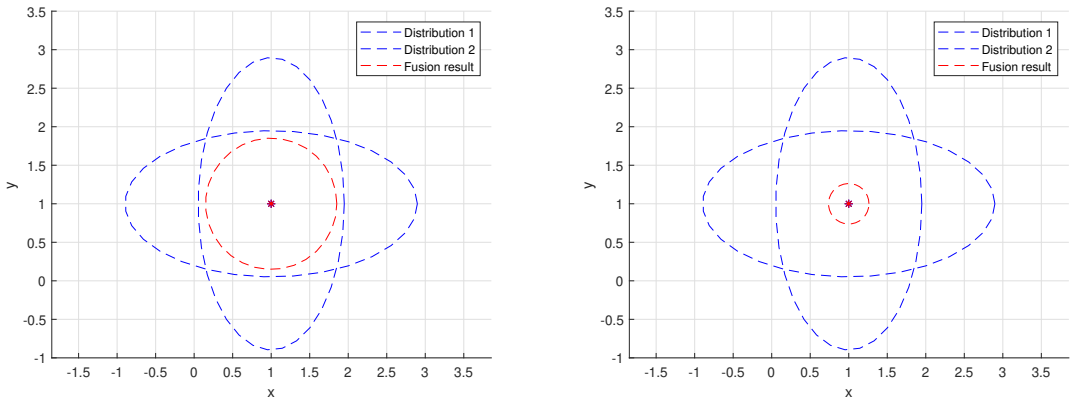
$$\hat{\mu} = \sum_i \omega_i\mu_i, \tag{3.14a}$$

$$\hat{P} = \sum_i \omega_i\left(P_i + (\hat{\mu} - \mu_i)(\hat{\mu} - \mu_i)^T\right), \tag{3.14b}$$

$$\hat{r} = \sum_i \omega_i r_i. \tag{3.14c}$$

3.3.4 Cross-covariance Fusion

The Cross-covariance (CC) fusion method uses cross-correlation between object densities [4] [12]. Given two Gaussian distributions with mean μ_1 and μ_2 and covariance



(a) The CC fusion result when $\rho = 0.01$ **(b)** The CC fusion result when $\rho = 0.99$

Figure 3.7: Illustration of the fusion result with two different correlation coefficients.

P_1 and P_2, the fused density $\hat{\mu}$ is calculated as follow:

$$\hat{\mu} = \mu_1 + \chi(\mu_2 - \mu_1), \tag{3.15}$$

where

$$\chi = (P_1 - P_{12})U_{12}^{-1}, \tag{3.16}$$

$$U_{12} = P_1 + P_2 - P_{12} - P_{12}^T, \tag{3.17}$$

where P_{12} is cross-covariance matrix. Note that the CC between two object densities from two sensors can be accurately calculated if we have full knowledge of how each sensor works, which is not always the case. An alternative way to approximate the CC matrix is using the entry-wise product. The CC of two covariance matrix P_1 and P_2 can be approximated as [4]

$$P_{12} \approx \rho\sqrt{P_1 \cdot P_2}, \tag{3.18}$$

where ρ is the correlation coefficient. The Figure 3.7 shows the fusion result when $\rho = 0.01$ and $\rho = 0.99$. The fusion covariance ellipse is within the range of the covered area, and a larger correlation coefficient leads to a smaller fusion covariance. For scenarios where the states are 2D Cartesian position vectors, $\rho \approx 0.4$ is a good approximation [4]. The covariance matrix $\hat{P}$ is calculated using

$$\hat{P} = P_1 - (P_1 - P_{12})U_{12}^{-1}(P_1 - P_{12})^T \tag{3.19}$$

Supposing there are N densities, the matrix P_c, which includes each density's covariance and their cross-covariance, can be expressed as:

$$P_c = \begin{bmatrix} P_1 & P_{12} & \cdots & P_{1N} \\ P_{21} & P_2 & \cdots & P_{2N} \\ \cdots & & \ddots & \cdots \\ P_{N1} & P_{N2} & \cdots & P_N \end{bmatrix}, \tag{3.20}$$

where the $P_1, \ldots, P_N$ on the matrix diagonal are covariance of each object density, and other parts are cross-covariance between two densities. The fused Gaussian density has mean and covariance [30]

$$\hat{x} = (\mathbf{I}^T P^{-1} \mathbf{I})^{-1} \mathbf{I}^T P_c^{-1} \mathbf{X}, \tag{3.21a}$$

$$\hat{P} = (\mathbf{I}^T P^{-1} \mathbf{I})^{-1}, \tag{3.21b}$$

where $\mathbf{I} = [I\ I \cdots\ I]^T$ is composed of $Nk \times k$ identity matrix, and k is the object state dimension, I is $k \times k$ identity matrix, and $\mathbf{X}$ is a $kN \times 1$ vector with all object's mean. The fused existence probability can be found similarly by noting that a Bernoulli variable with existence probability r has mean r and variance $P_r = r(1-r)$. The correlation factor ρ can be 0, so the cross-covariance matrix between each existence probability is $\mathbf{0}$. The (3.20) can be rewritten as:

$$P_{rc} = diag\left(P_{r_1}, \cdots, P_{r_N}\right), \tag{3.22}$$

where P_{r_i} is the variance of each existence probability. The fused existence probability can be found by

$$\hat{r} = (\mathbf{I}^T P^{-1} \mathbf{I})^{-1} \mathbf{I}^T P_{rc}^{-1} \mathbf{X}_r, \tag{3.23}$$

where $\mathbf{X}_r$ is a $N \times 1$ vector with all existence probabilities.

3.4 Multi-object Trajectory Estimation

The object information from the sensors contains the object labels, which can be used to form object trajectories. However, if we have two or more objects with different labels, it is unclear how to determine the label of the fused object density. One approach to address this is to reassign object labels based on the best fusion map [14]. However, this method may face difficulties in challenging scenarios where initially well-separated objects move closely and then separate again. In such cases, this track labelling procedure may encounter problems and result in unrealistic trajectory estimates. A more appealing approach to multi-object trajectory estimation is using backward simulation [23] [31], which can directly work on unlabelled multi-object densities.

Based on the fused multi-object densities at each time step, trajectory estimation acts as a particle smoother. As discussed in Section 2.2.3, multi-object trajectory estimation using backward simulation consists of a forward filtering process and a backward smoothing process. The fused multi-object densities can be viewed as output from the forward filtering process. For backward smoothing, we sequentially draw samples of sets of trajectories from the backward kernel backwards in time. This will be elaborated in the following subsections.

3.4.1 Backward Simulation for Sets of Trajectories

The objective is to obtain the multi-trajectory posterior density $\pi_{1:T|T}\left(X_{1:T}\right)$ from multi-Bernoulli densities using backward simulation. As mentioned in Section 2.2.3,

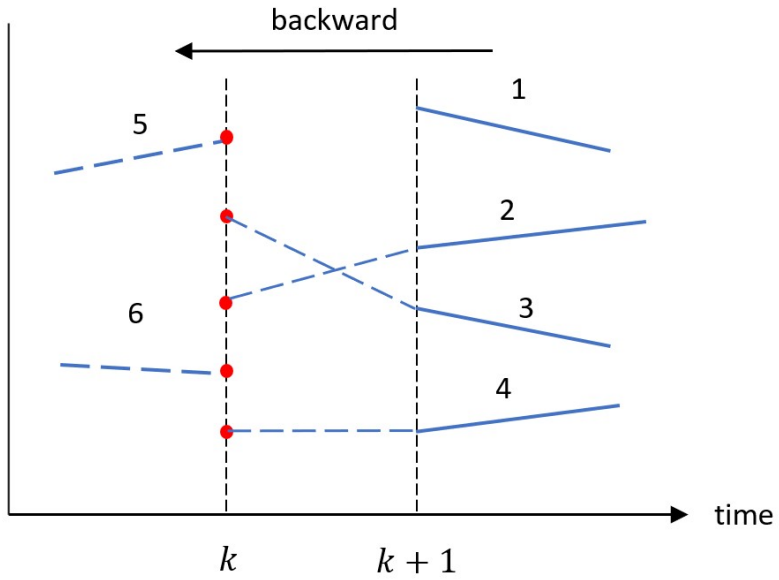


Figure 3.8: An example shows the backward simulation results (dashed line). There are 4 trajectories existing at time $k+1$ and 5 objects at time k, where trajectory 1 is ended at time k; trajectories $2-4$ are updated by the objects at time k; trajectories 5 and 6 initialize two new trajectories; and all the trajectories starting after time $k+1$ remain unaltered.

backward simulation is a sequential process from the last time instance T to the initial time instance. In each step, we consider four possible local hypotheses for a set of trajectories. Assume that we have a set of trajectories at time $k+1$ and an MB set of object densities at time k. The first local hypothesis is that a subset of the trajectories is updated by a subset of the object set, meaning that these trajectories are extended to time k; the second local hypothesis is that a subset of the trajectories is ended at time $k+1$; the third local hypothesis is that a subset of the object set initializes a new set of trajectories at time k; the last local hypothesis is that a subset of the trajectories remains unaltered, meaning that these trajectories start at or after time $k+1$. Figure 3.8 shows an example of four possible results when performing the backward simulation from time $k+1$ to time k.

A global hypothesis is a collection of local hypotheses, explaining the association of each trajectory and object density. Each local hypothesis can be represented as a_i, and it can be one of the four local hypotheses mentioned above. Therefore, a global hypothesis can be represented as a set $h = \{a_1, a_2, ..., a_n\}$. By initializing the trajectories at the last time instance T (by sampling the MB density at time T) and sequentially drawing samples of trajectories from the backward simulation kernel (2.19) till time 1, we can obtain a sample of the set of trajectories in time interval $1:T$. If we run the complete backward simulation S times, we then obtain a particle representation of the posterior multi-trajectory density in time interval $1:T$.

3.4.2 Gaussian Implementation for Backward Simulation

This subsection shows the Gaussian implementation of backward simulation. The following assumptions are made:

- Each object has a constant survival probability p^S,
- Object state dynamic function is linear Gaussian, i.e. $x_{k+1} = \mathcal{N}(\cdot; Ax_k, Q)$ (see Section 2.3),
- The i-th Bernoulli component at time k has existence probability $r^i_{k|k}$ and Gaussian single-object density with mean $x^i_{k|k}$ and covariance $P^i_{k|k}$,
- The set of newborn objects at each time step is a PPP with intensity

$$\lambda^B_k(x) = \sum_{i=1}^{N^b_k} w^{b,i}_k \mathcal{N}\left(x; x^{b,i}_k, P^{b,i}_k\right), \tag{3.24}$$

  where N^b_k is the number of Gaussian components, and each component has weight $w^{b,i}_k$.

The different local hypotheses can be represented as

- The hypothesis, corresponding to the case that the trajectory ended at time step $k+1$, is given by

$$w^{i,1}_{k:T|T} = 1 - r^i_{k|k} + r^i_{k|k}\left(1 - p^S\right), \tag{3.25a}$$

$$r^{i,1}_{k:T|T} = \frac{r^i_{k|k}\left(1 - p^S\right)}{w^{i,1}_{k:T|T}}, \tag{3.25b}$$

$$p^{i,1}_{k:T|T}\left(t, x^{1:\nu}\right) = \delta_k[t]\delta_1[\nu]\mathcal{N}\left(x^1; x^i_{k|k}, P^i_{k|k}\right), \tag{3.25c}$$

  where $\delta_k[\cdot]$ represents the Kronecker delta function centred at k.
- The hypothesis, corresponding to the case that the Bernoulli component, $i \in \{1, \ldots, n_{k|k}\}$, is updated by trajectory $Y^j = \left(t^j, y^{1:\nu^j}\right), j \in \{1, \ldots, m\}$, is given by

$$w^{i,j+1}_{k:T|T} = r^i_{k|k} p^S \mathcal{N}\left(y^1; Ax^i_{k|k}, P^i_{k+1|k}\right), \tag{3.26a}$$

$$r^{i,j+1}_{k:T|T} = 1, \tag{3.26b}$$

$$p^{i,j+1}_{k:T|T}\left(t, x^{1:\nu}\right) = \delta_k[t]\delta_{\nu^j+1}[\nu]\delta_{y^{1:\nu_j}}\left(x^{2:\nu}\right) \times \mathcal{N}\left(x^1; x^i_{k|T}, P^i_{k|T}\right), \tag{3.26c}$$

$$x^i_{k|T} = x^i_{k|k} + G^i\left(y^1 - Ax^i_{k|k}\right), \tag{3.26d}$$

$$P^i_{k|T} = P^i_{k|k} - G^i A P^i_{k|k}, \tag{3.26e}$$

$$G^i = P^i_{k|k} F^T \left(P^i_{k+1|k}\right)^{-1}, \tag{3.26f}$$

$$P^i_{k+1|k} = AP^i_{k|k}A^T + Q. \tag{3.26g}$$

- The hypothesis of the i-th trajectory Bernoulli component, $i \in \{n_{k|k}+1, \ldots, n_{k|k}+m\}$ corresponding to the case that the object with trajectory $Y^j = \left(t^j, y^{1:\nu^j}\right)$

was first detected at time step $k+1$, is given by

$$w_{k:T|T}^{i,2} = \sum_{i=1}^{N_{k+1}^b} w_{k+1}^{b,i} \mathcal{N}\left(y^1; x_{k+1}^{b,i}, P_{k+1}^{b,i}\right) \tag{3.27a}$$

$$r_{k:T|T}^{i,2} = 1, \tag{3.27b}$$

$$p_{k:T|T}^{i,2}\left(t, x^{1:\nu}\right) = \underline{w}_{k:T|T^i}^{i,2} \delta_{(t^j, y^{1:\nu_j})}\left(t, x^{1:\nu}\right). \tag{3.27c}$$

- The hypothesis of the i-th trajectory Bernoulli component, $i \in \{n_{k|k}+1, \ldots, n_{k|k}+m\}$ corresponding to the case that the object with trajectory $Y^j = \left(t^j, y^{1:\nu^j}\right)$ remains unaltered, has $w_{k:T|T}^{i,1} = 1, r_{k:T|T}^{i,1} = 1$ and $p_{k:T|T}^{i,1}(X) = \delta_{Y^j}(X)$.

In principle, we need to enumerate every possible global hypothesis to draw samples, which can be computationally expensive. One possible solution is to only keep the global hypotheses with non-negligible weights [23]. This can be addressed by solving a ranked assignment problem using Murty's algorithm [32] [33]. More specifically, our goal is to sample an association that assigns m trajectories at time $k+1$ to $n_{k|k}$ Bernoulli densities at time k. The assignment problem is formulated by constructing a cost matrix,

$$C = -\log \begin{bmatrix} W_1 & W_2 \end{bmatrix}, \tag{3.28a}$$

$$W_1^{(j,i)} = \frac{w_{k:T|T}^{i,j+1}}{w_{k:T|T}^{i,1}}, \tag{3.28b}$$

$$W_2 = \operatorname{diag}\left(w_{k:T|T}^{n_{k|k}+1,2}, \ldots, w_{k:T|T}^{n_{k|k}+m,2}\right), \tag{3.28c}$$

where the (j,i)-th entry in $W_1 \in \mathbb{R}^{m \times n_{k|k}}$ represents the cost by assigning the j-th object state to the corresponding i-th trajectory and $W_2 \in \mathbb{R}^{m \times m}$ is a diagonal matrix where each entry on the diagonal represents the new trajectories are initialized by the object state at time k. Then by running Murty's algorithm, we obtain M-best assignments with non-negligible weights and we draw a sample from these M-best assignments.

In addition to pruning negligible global hypotheses, we also compute the squared Mahalanobis distance (SMD) between the first state y_1 in a trajectory and the predicted density of $\mathcal{N}\left(x; x_{k|k}^i, P_{k|k}^i\right)$

$$\text{SMD}\left(y_1; x_{k|k}^i, P_{k|k}^i\right) = \left(y_1 - A x_{k|k}^i\right)^T \left(A P_{k|k}^i A^T + Q\right)^{-1} \left(y_1 - A x_{k|k}^i\right) \tag{3.29}$$

to a threshold σ. If this distance is larger than σ, we set the corresponding entry in the cost matrix (3.28a) to 0.

3.5 Evaluation

To evaluate the performance of MOT algorithms, we need to measure the similarity between the estimated set of object states and the ground truth. Intuitively, an

MOT performance measure should capture localization errors for properly detected objects, cost for missed detections, and cost for false detections. In this thesis, two performance measures are used to evaluate MOT algorithms: GOSPA metric [34] and NLL [35].

3.5.1 GOSPA

Let X be the set of true object states and Y be the set of object estimates. Let $d(x, y)$ denote the Euclidean distance between x and y, and $d^{(c)} = \min(c, d(x, y))$. Also let Π_n be the set of all permutations of $\{1, \cdots, n\}$ for any $n \in \mathbb{N}$ where $\pi \in \Pi_n$ is a sequence $(\pi(1), \cdots, \pi(n))$. The GOSPA metric is defined as

$$d_p^{(c,\alpha)}(X, Y) \triangleq \left(\min_{\pi \in \Pi_{|Y|}} \sum_{i=1}^{|X|} d^{(c)} \left(x_i, y_{\pi(i)} \right)^p + \frac{c^p}{\alpha} (|Y| - |X|) \right)^{\frac{1}{p}}. \tag{3.30}$$

There are three parameters in GOSPA: c stands for the maximum allowable localization error, p gives a penalty for outliers, and α is a factor for the cardinality mismatching penalty, which is suggested to be set as 2 for evaluating MOT performance in [34]. Given a ground truth $x \in X$, object estimate $y \in Y$, and $d(x, y) \geq c$, then according to $d^{(c)} = \min(c, d(x, y))$, the distance between x and y is c. The penalty for assigning ground truth x with object estimate y is c^p, and c^p/α otherwise. If we set $\alpha = 2$, the cost of the unassigned object would be the same as the cost of that object being associated with another object at distance c. Computing the GOSPA metric ($\alpha = 2$) can be formulated as a 2D assignment problem:

$$d_p^{(c,2)}(X, Y) = \left[\min_{\gamma \in \Gamma} \left(\sum_{(i,j) \in \gamma} d\left(x_i, y_j\right)^p + \frac{c^p}{2} (|X| + |Y| - 2|\gamma|) \right) \right]^{\frac{1}{p}}, \tag{3.31}$$

where γ is an assignment set between the estimates and the ground truth, and Γ is a set of all possible assignment sets γ. Note that $|\gamma|$ is the number of properly assigned objects, and that $|X| - |\gamma|$ and $|Y| - |\gamma|$ represent the number of missed and false objects, respectively. Consequently, GOSPA allows for the decomposition of the total estimation error into localization errors for properly detected objects and errors for missed and false detections, respectively.

3.5.2 NLL

GOSPA metric does not take into account the uncertainty information when evaluating the MOT performance, and it has two hyper-parameters c and p that need to be chosen. As a comparison, NLL also evaluates the object state uncertainty estimates and has no hyperparameters. Given the true set of object states Y and the multi-object posterior density $f(\cdot)$, the NLL is defined as

$$\text{NLL}(Y, f) = -\log f(Y). \tag{3.32}$$

Note that our posterior density is an MB. To assess performance using the NLL measure, we introduce a PPP component, whose intensity is a Gaussian distribution

centred at the origin with a large covariance. The NLL evaluates PMB densities in three parts, including matched objects, missed objects and false detections. Let the PPP intensity function be $\lambda(\cdot)$, and let r_i and $p_i(\cdot)$ represent the existence probability and single-object density of i-th Bernoulli components, respectively. The following algorithm can be used to approximately calculate the NLL [35]:

$$\begin{aligned} \mathrm{NLL}\left(Y, f_{\mathrm{PMB}}\right) \approx \min_{\gamma \in \Gamma} & \underbrace{-\sum_{(i,j)\in\gamma} \log\left(r_i p_i\left(y_j\right)\right)}_{\text{Localization}} \\ & \underbrace{-\sum_{i\in\mathbb{F}(\gamma)} \log\left(1-r_i\right)}_{\text{False detections}} + \underbrace{\int \lambda\left(y'\right) \mathrm{d}y' - \sum_{j\in\mathbb{M}(\gamma)} \log \lambda\left(y_j\right)}_{\text{Missed objects}}, \end{aligned} \tag{3.33}$$

where γ represents a possible assignment between the estimates and the ground truth, $\mathbb{M}$ represents the ground truth that is not assigned to any estimates, and the $\mathbb{F}$ denotes the estimates that are not assigned to any ground truth. The localization part measures how well the Bernoulli components explain the matched ground truths. The false detections part captures the existence uncertainty of unmatched object densities. Finally, the missed object part evaluates how well the PPP component explains the unmatched ground truths.

The NLL of PMB can be interpreted as the sum of the likelihoods of all possible assignments between the ground truths and either PPP component or one of the Bernoulli components of the estimates. It is generally intractable to compute the exact NLL due to the large number of possible assignments. However, usually, only a few of these likelihoods of assignments contribute most to the exact sum. This means that the likelihood can be approximated by finding the best assignment between the ground truths and the estimates and omitting all other possible assignments. Finding the assignment γ with the minimum cost can be formulated as a 2D assignment problem, and the corresponding cost matrix is calculated as

$$C_{i,j} = \begin{cases} -\log\left(\frac{p_i(y_j)}{1-r_i} r_i\right), & \text{if } i \leq m \\ -\log \lambda\left(y_j\right), & \text{if } i = j + m \\ \infty, & \text{otherwise}, \end{cases} \tag{3.34}$$

where $C_{i,j}$ is the (i,j)-th entry of cost matrix C, and m is the number of estimates.

4

Simulation Results

In this chapter, simulations are set up based on the knowledge from the testing trucks. Then the settings of the simulation and result comparisons are shown.

4.1 Simulation Setting

In this section, we introduce the simulation settings for MOT evaluation. Four sensors are placed in a 150 m× 150 m area. Each sensor's feedback is an MB density representing the set of objects within its FoVs. At each time step, we perform multi-object density fusion to fuse the densities from each sensor. Then, we apply backward simulation on the fused multi-Bernoulli density at each time step to obtain trajectory estimates.

4.1.1 Sensor FoVs Setup

We set the average overlapping area for four sensors to reflect the real truck configuration as shown in Figure 3.2, where some areas can be surveilled by multiple sensors, and some just by single sensors. The sensors' layout for simulation is shown in Figure 4.1.

4.1.2 Ground Truth Creation

We assume that the objects' movements follow a constant velocity (CV) model. The following steps generate the ground truth:

1. Specify the initial position and velocity of each object. The object state at time k can be written as

$$x_k = \begin{bmatrix} x & \dot{x} & y & \dot{y} \end{bmatrix}^T. \tag{4.1}$$

2. Determine the time interval and simulation duration. For convenience, we set the time interval and the duration as $T_s = 1$ and 20 respectively.
3. Generate the object densities. The state of the object at time k can be sequentially generated using the motion model (2.3). During the simulation, we set the dynamic matrix A_k as time-invariant

$$A = I_2 \otimes \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix}, \tag{4.2}$$

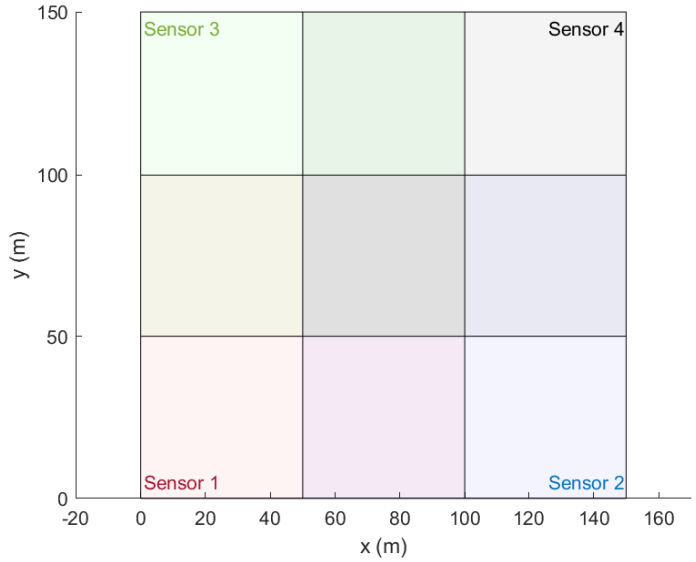


Figure 4.1: The sensor layout contains nine different blocks, four blocks at the corners covered by only one sensor, one block at the centre covered by all four sensors, and the rest four blocks covered by two sensors.

where I_2 is an identity matrix of size 2, $\otimes$ denotes the Kronecker product, and T_s represents the time interval of this discrete process. And we set the input matrix B_k as zero. The noise is defined as $q \sim \mathcal{N}(0, Q)$ where $Q = \sigma_q^2 I_2 \otimes \begin{bmatrix} \frac{T_s^3}{3} & \frac{T_s^2}{2} \\ \frac{T_s^2}{2} & T_s \end{bmatrix}$, and we set the σ_q fairly as 0.3.

4. The generated object densities at consecutive time steps can be viewed as a set of sequences. Given a sequence of object densities, at each time k, we sample a random number from a uniform distribution and compare it with $p_i = 0.92$. If the sample is larger than p_i, we remove this single density. Using this trick, we introduce random birth and death into this sequence of object densities, which makes the ground truth more similar to the real-world setting.

In addition to the CV model, we implement the coordinate turn (CT) motion model to create ground truth with nonlinear object motions. With the object states $x_k = \begin{bmatrix} x & y & v & \theta & \omega \end{bmatrix}^T$ where x and y represent the position, v is the velocity, θ is the heading direction, and ω is the turn rate, we set the dynamic function $f(x_k, u_k, q_k)$ in Section 2.2 as time-invariant and omit the input u_k. The dynamic function is

$$f(x, q) = \begin{bmatrix} x + T_s v \cos(\theta) \\ y + T_s v \sin(\theta) \\ v \\ \theta + T_s \omega \\ \omega \end{bmatrix} + q, \tag{4.3}$$

where $q \sim \mathcal{N}(0, Q)$ is the process noise in CT model with $Q = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & T_s\sigma_v^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & T_s\sigma_\omega^2 \end{bmatrix}$.

4.1.3 Object Estimate Densities Creation

For creating the object estimate densities for the simulation, we assume that the objects can be estimated around the ground truth. This can be achieved by drawing a sample from a Gaussian distribution whose mean is located at the ground truth with a constant covariance matrix, which can be viewed as a measurement of estimation accuracy. The covariance of this sample is set by sampling from a Wishart distribution, which allows the generation of random covariance matrices that can capture the desired real-world statistical properties. This is achieved by setting the sigma parameter of Wishart based on the knowledge of real-world sensors' covariance and by limiting the degree of freedom at a normal level. Figure 4.2 shows four examples of the simulated tracks.

4.2 MOT Simulation Evaluation

The multi-object multi-sensor fusion algorithm, described in Section 3.2 is integrated with the four different single-object fusion methods described in Section 3.3, resulting in four different implementations. We evaluate their performance in scenarios with different sensor FoVs and the number of objects using GOSPA and NLL.

4.2.1 Simulation Result with Different Number of Objects

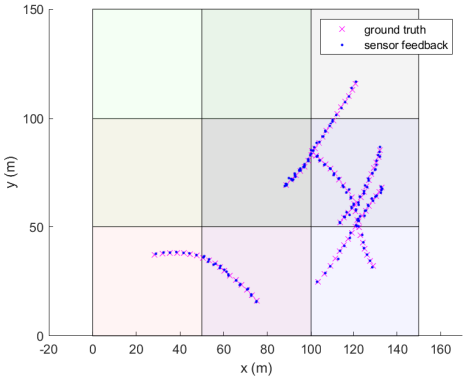
First, we keep the sensors' FoVs fixed as shown in Figure 4.1, and conduct two simulations with 5 and 20 objects, respectively, and each object has random initial positions and velocities. The evaluation results are obtained by averaging a number of 100,000 times Monte Carlo simulations. The results under 5 and 20 objects are shown in Table 4.1 and Table 4.2, respectively.

Table 4.1: GOSPA and NLL values of four implementations with 5 objects.

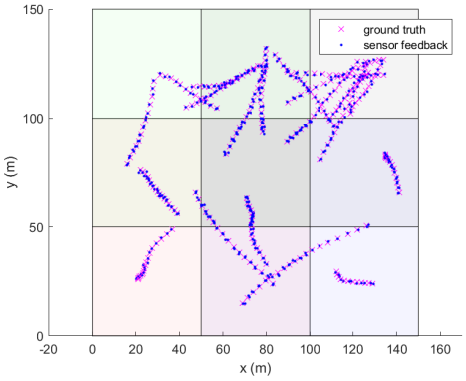
	CI	SF	AA	CC
GOSPA	0.9817 ± 0.1007	$1.2108 \pm \mathbf{0.0869}$	0.9817 ± 0.1007	$\mathbf{0.9692} \pm 0.1051$
NLL	8.4053 ± 2.6080	$15.1347 \pm \mathbf{2.4953}$	$\mathbf{8.1641} \pm 2.5828$	11.7661 ± 2.5077

Table 4.2: GOSPA and NLL values of four implementations with 20 objects.

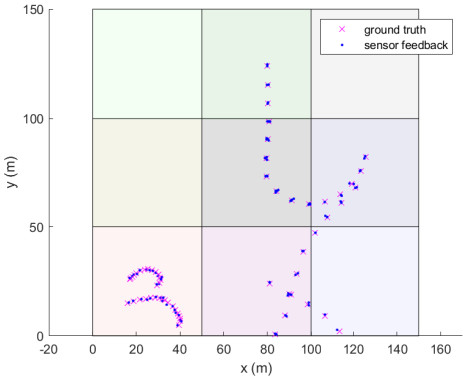
	CI	SF	AA	CC
GOSPA	1.9722 ± 0.1141	2.4153 ± 0.1040	1.9722 ± 0.1141	$\mathbf{1.9502} \pm 0.1172$
NLL	31.9725 ± 4.8912	57.4180 ± 4.7032	$\mathbf{31.0877} \pm 4.8450$	44.7085 ± 4.7162



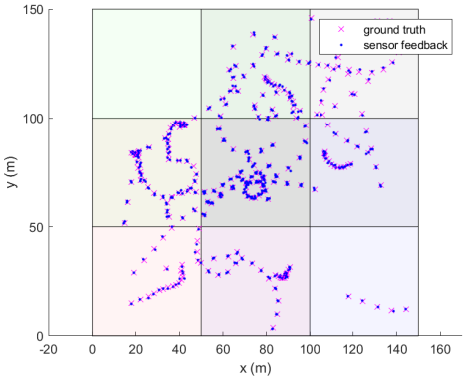
(a) Randomly generated 5 trajectories using CV model.



(b) Randomly generated 20 trajectories using CV model.



(c) Randomly generated 5 trajectories using CT model.



(d) Randomly generated 20 trajectories using CT model.

Figure 4.2: Examples of randomly generated trajectories in four different simulation settings.

4.2.2 Simulation Result with Different Overlapping areas

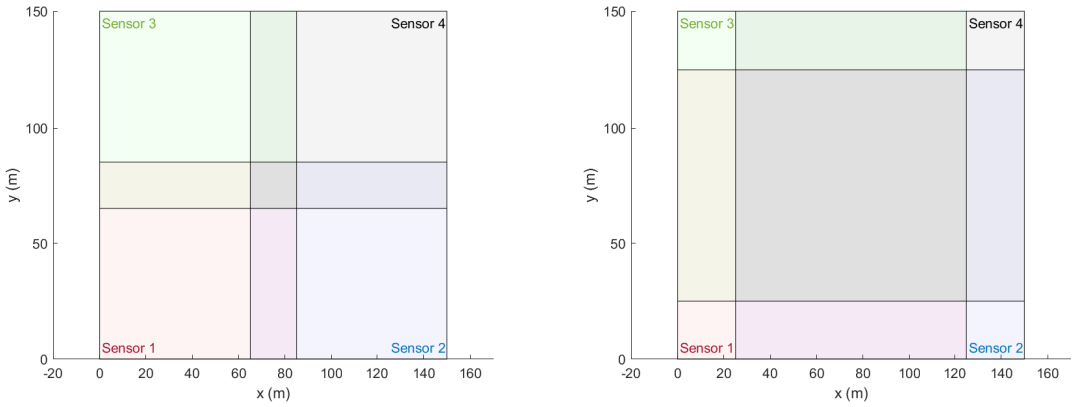
We also evaluate the robustness of the multi-object fusion algorithm under two different sensors' FoVs configurations, and in both cases, we set the number of trajectories as 5. The evaluation results are obtained by averaging a number of 100,000 times Monte Carlo simulations. The results of three different cases with small, medium, and large overlapping areas are shown in Table 4.3, 4.1, and 4.4, respectively.

Table 4.4: GOSPA and NLL values of four implementations in a scenario with a larger overlapping area.

	CI	SF	AA	CC
GOSPA	0.7213 ± 0.0728	1.2185 ± 0.09671	0.7213 ± 0.0728	$\mathbf{0.6190 \pm 0.0699}$
NLL	$\mathbf{1.8620} \pm 1.1840$	15.1644 ± 2.5903	$1.9628 \pm \mathbf{0.9246}$	5.9861 ± 2.0167

Table 4.3: GOSPA and NLL values of four implementations in a scenario with a smaller overlapping area.

	CI	SF	AA	CC
GOSPA	1.0870 ± 0.0962	$1.2097 \pm \mathbf{0.08340}$	1.0870 ± 0.09623	$\mathbf{1.0836} \pm 0.09787$
NLL	11.4350 ± 2.6708	$15.1213 \pm \mathbf{2.4486}$	$\mathbf{11.2689} \pm 2.6710$	13.4188 ± 2.4831



(a) Small overlapping area **(b)** Large overlapping area

Figure 4.3: Two different settings for the sensors' FoVs overlapping areas.

4.2.3 Simulation Result with Different Motion Model

We also evaluate the fusion performance when the object trajectories are generated using the CT model. We choose the sensor layout as Figure 4.1, and we consider four different scenarios with 5, 10, 15, and 20 objects. The evaluation results are obtained by averaging a number of 100,000 times Monte Carlo simulations. The result of the simulation with 5 trajectories is shown in Table 4.5.

Table 4.5: GOSPA and NLL values of four implementations with CT model and 5 objects.

	CI	SF	AA	CC
GOSPA	1.2486 ± 0.2346	1.4512 ± 0.2082	1.2486 ± 0.2346	$\mathbf{1.2369} \pm 0.2379$
NLL	8.0111 ± 5.0117	13.5588 ± 5.1780	$\mathbf{7.8215} \pm 4.9827$	10.7542 ± 5.0881

Table 4.7: GOSPA and NLL values of four implementations with CT model and 15 objects.

	CI	SF	AA	CC
GOSPA	2.3764 ± 0.3143	2.6880 ± 0.2778	2.3764 ± 0.3143	$\mathbf{2.3600} \pm 0.3173$
NLL	22.7095 ± 7.3046	36.1679 ± 8.2424	$\mathbf{22.2723} \pm 7.2524$	29.3310 ± 7.6685

Table 4.6: GOSPA and NLL values of four implementations with CT model and 10 objects.

	CI	SF	AA	CC
GOSPA	1.8606 ± 0.2830	2.1285 ± 0.2521	1.8606 ± 0.2830	$\mathbf{1.8461} \pm 0.2863$
NLL	15.2951 ± 6.3094	25.1431 ± 6.7127	$\mathbf{14.9541} \pm 6.2712$	20.1593 ± 6.5033

Table 4.8: GOSPA and NLL values of four implementations with CT model and 20 objects.

	CI	SF	AA	CC
GOSPA	2.8250 ± 0.3354	3.1676 ± 0.2999	2.8250 ± 0.3354	$\mathbf{2.8078} \pm 0.3385$
NLL	30.7134 ± 8.7723	47.0626 ± 9.8334	$\mathbf{30.1637} \pm 8.6975$	38.8617 ± 9.1875

4.2.4 Simulation Result with a Combination of Low-quality Sensors and High-quality Sensors

In addition, we evaluate the performance of the multi-object multi-sensor fusion algorithm in scenarios where sensors have different qualities. In particular, we manually set the sensor 1 and the sensor 4 with big measurement covariance, which represents the larger uncertainty about the object estimate. The simulation is conducted with the CV model with 5 trajectories, and the results are obtained by averaging a number of 100,000 times Monte Carlo simulations. The result is shown in Table 4.9.

Table 4.9: GOSPA and NLL values of four implementations with a combination of low-quality and high-quality sensors.

	CI	SF	AA	CC
GOSPA	2.6378 ± 0.8274	$\mathbf{2.3573} \pm 0.7316$	$3.1566 \pm \mathbf{0.7226}$	2.4653 ± 0.8887
NLL	$\mathbf{10.2655} \pm 4.0854$	16.0120 ± 11.1145	15.0920 ± 12.2601	$14.5136 \pm \mathbf{3.1087}$

4.3 Trajectory Estimation

Trajectory estimates are obtained by applying backward simulation, described in Section 3.4, to the fused multi-Bernoulli multi-object densities. Figure 4.4 shows an example of trajectory estimation, where there are 5 objects moving from the left-bottom to the right-top of the surveillance area. In the first scenario, we create the ground truth without random birth and death. It can be seen from the results that, in general, the trajectory estimates of different objects can be correctly separated even when objects are moving in close distance, see the left-bottom corner in Figure 4.4. Figure 4.5 and Figure 4.6 show two more complex examples of trajectory estimation, where we bring back the random birth and death. In addition, in Figure 4.6, we manually set the objects closer to be more challenging.

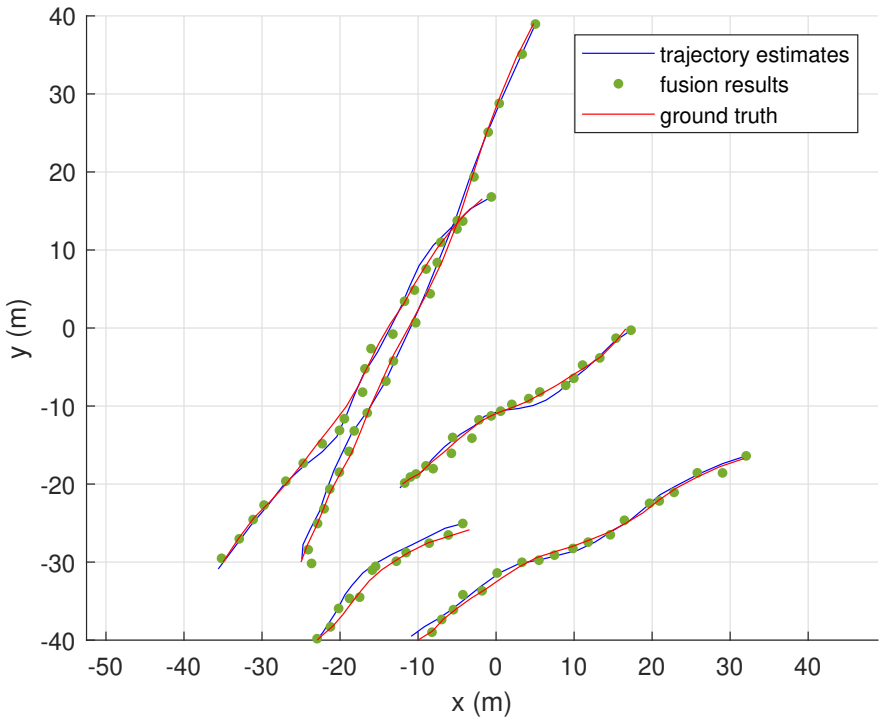


Figure 4.4: Trajectory estimations of 5 objects, where the green dots represent the fusion results, the blue line segments denote the trajectory estimates, and the ground truth trajectories are marked as red line segments.

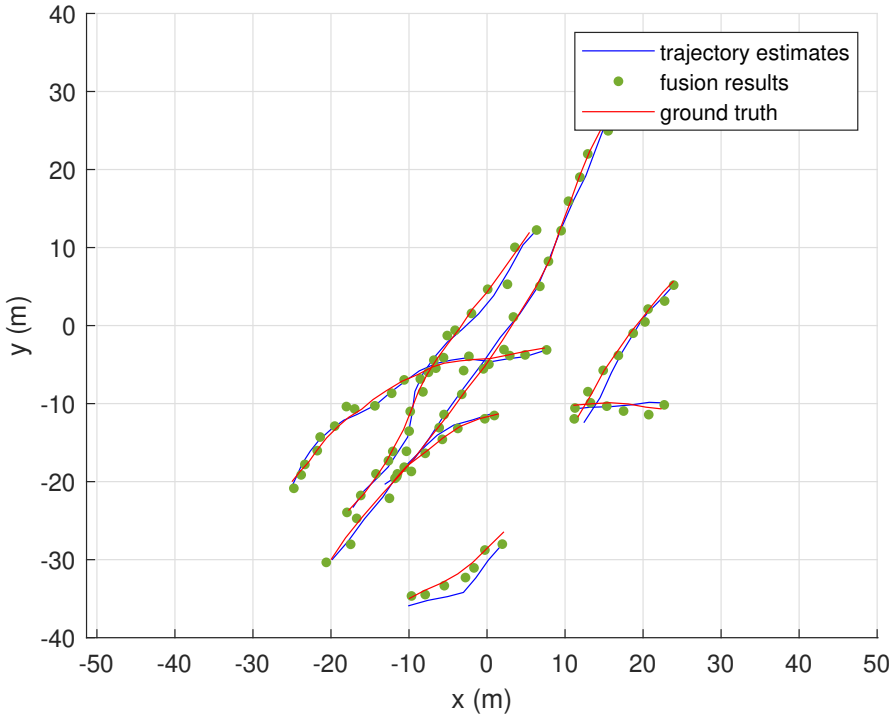


Figure 4.5: Trajectory estimations on 5 objects with random birth and random death.

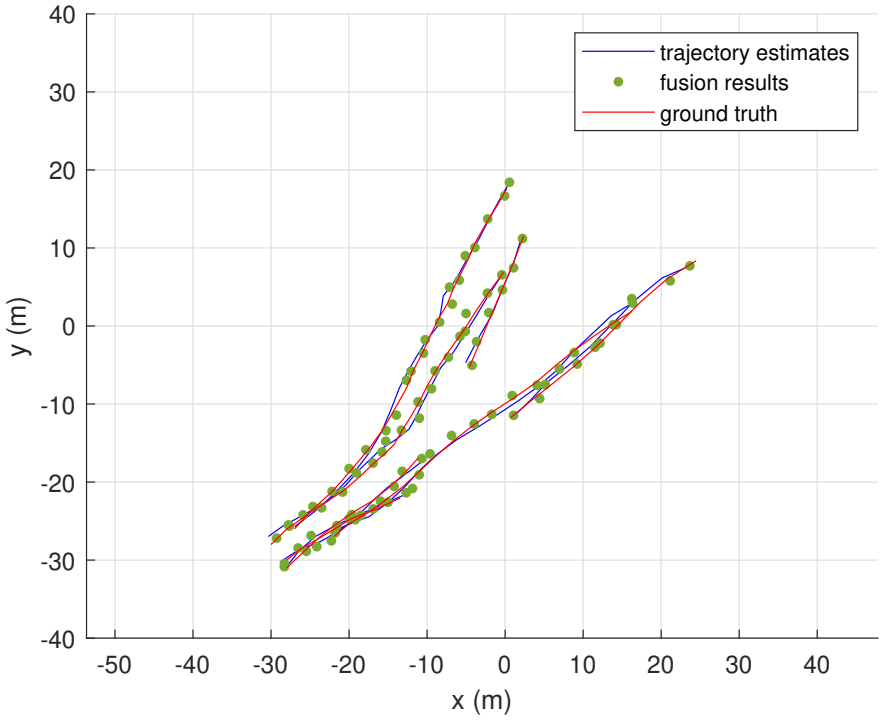


Figure 4.6: Trajectory estimations on 5 objects. We set the random birth and death and make the objects closer.

5
Experimental Results

In this chapter, we evaluate the four different implementations using real-world data. In the scenario being considered, a Volvo V60 car is overtaking a moving testing truck and cutting in from the left rear to its left front. The fusion results obtained from four different implementations are compared to the ground truth, and the fusion performance is evaluated using GOSPA and NLL. In addition, the running times of these algorithms are compared.

5.1 Real-world Sensor Data

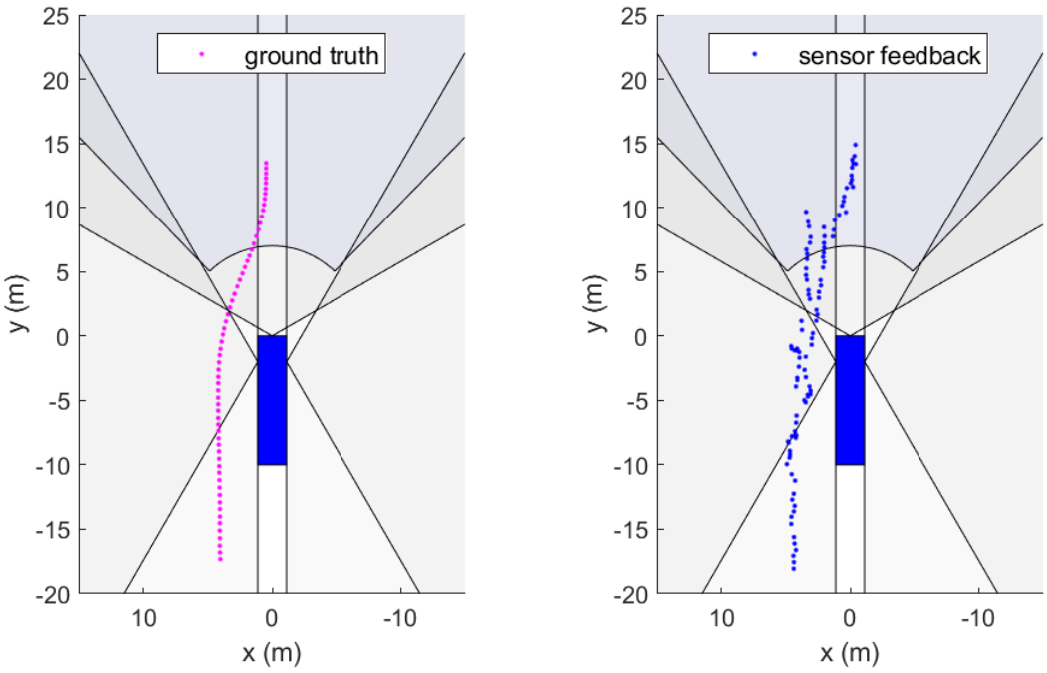
This real-world evaluation is powered by the Volvo reference system, including multiple real-world datasets and ground truth, with both multi-object and single-object data. However, due to time constraints and imperfect reference data, we can only evaluate our fusion algorithm under a single-object scenario. In practice, we clip a representative scenario, in which a Volvo V60 car overtakes an ego truck. As shown in Figure 5.1a, this car crosses the FoVs of four different sensors during the entire process, a scenario that could be considered representative for testing our developed fusion algorithm. In addition, the experimental evaluation, as shown in Figure 5.1b, only uses valid objects densities that are moving and with high existence confidence.

Usually, the sensor is set to return position on the middle of the closest side of this object. Therefore, the object's size can be approximated based on the measurement of this vehicle's closest side length and width length. In the reference system, the data that we get is a point attached to the back of the car. Thus, a position transformation is needed to compensate for this bias. It is also worth mentioning the time synchronization between the sensor data and the ground truth. In our testing case, the ground truth is updated more frequently than the sensor data. Therefore, it is always possible to find the corresponding ground truth for any sensor data at any given time.

5.2 Experimental Evaluation

5.2.1 Fusion Result Visualization

Figure 5.2 shows the multi-object fusion results tested using four different single-object fusion methods.



(a) Ground truth **(b)** Valid feedback from sensors

Figure 5.1: The scenario in the real-world testing data, where a car takes over the testing truck (blue rectangular) on the road.

5.2.2 GOSPA Evaluation Result

For this experimental evaluation, we set the parameters of GOSPA as $p = 2$, $c = 8$, and $\alpha = 2$. The average GOSPA of four different single-object fusion methods is shown in Table 5.1.

Table 5.1: Average GOSPA value of four single-object fusion algorithms

	CI	SF	AA	CC
GOSPA	**0.8127**	0.8630	0.8737	0.8351

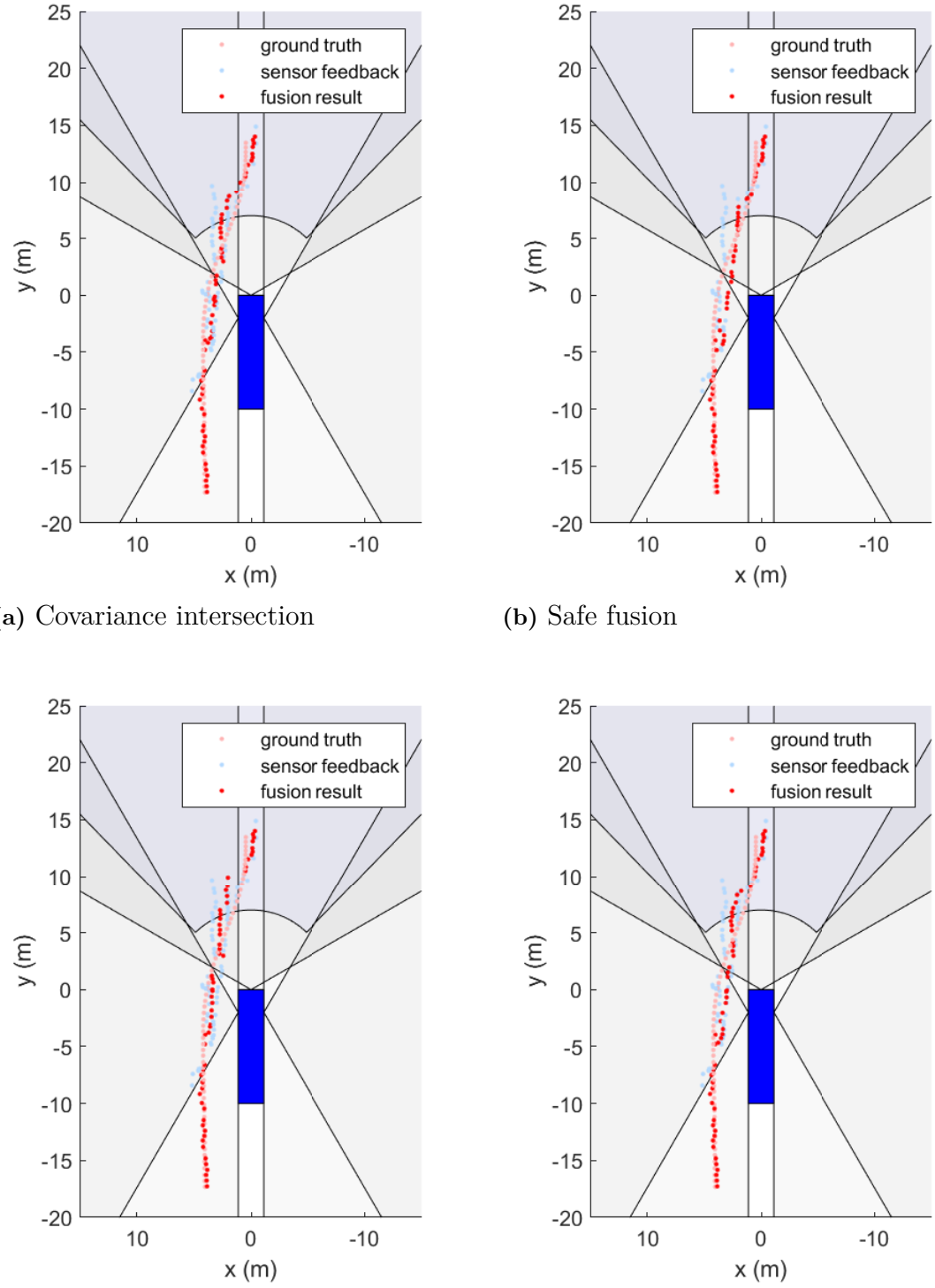
In order to illustrate the comparison at different time steps, Figure 5.3 and Figure 5.4 show the GOSPA and NLL values during the overtaking process.

5.2.3 NLL Evaluation Result

The result showing the NLL of four different single-object fusion methods is given in Table 5.2.

Table 5.2: Average NLL value of four single-object fusion methods

method	CI	SF	AA	CC
NLL	3.5433505	4.8674531	**3.5174546**	4.3920736



(a) Covariance intersection

(b) Safe fusion

(c) Arithmetic average fusion

(d) Cross-covariance fusion

Figure 5.2: The fusion result from four different single-object fusion algorithms.

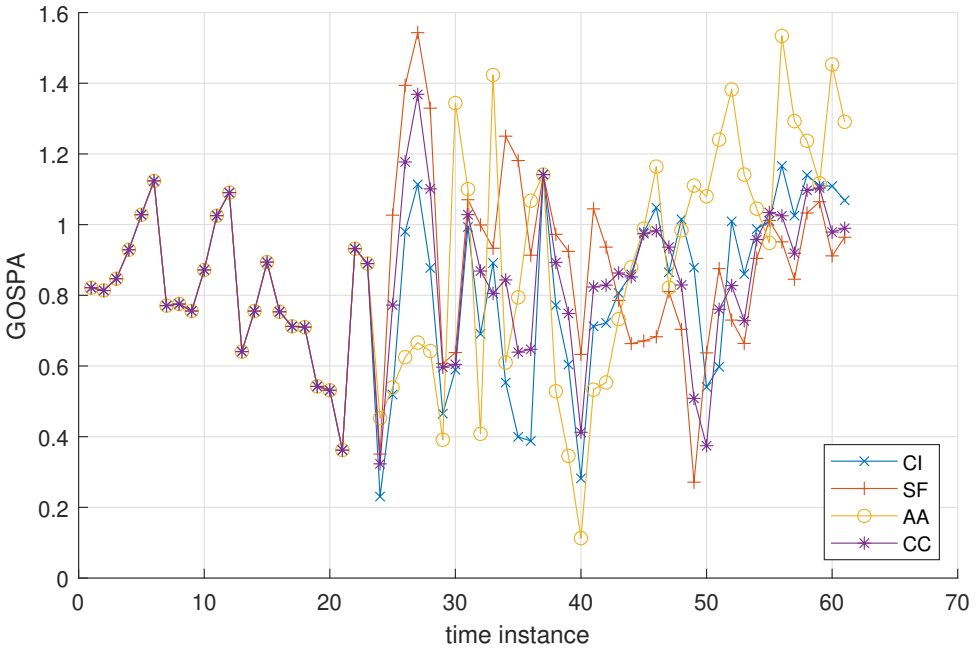


Figure 5.3: GOSPA values over 61 time instances of four different single-object fusion algorithms.

5.2.4 Computational Complexity

To evaluate the computational complexity, we record the running time for each implementation[1]. The runtime is measured by running the entire multi-object fusion process (including preprocessing, data association, and single-object density fusion algorithms) 10000 times and takes the average. The result is shown in Table 5.3.

Table 5.3: Average time consumption in the simulations

method	CI	SF	AA	CC
time (s)	0.02915	0.03062	**0.02811**	0.02954

5.3 Trajectory Estimation

The trajectory can be estimated after getting the fusion result in real data. Figure 5.5 illustrates the trajectory result.

[1]The implementation is carried out under MATLAB 2020a powered by Intel®Core™ i7-11850H processor with 32 GB DDR4 memory.

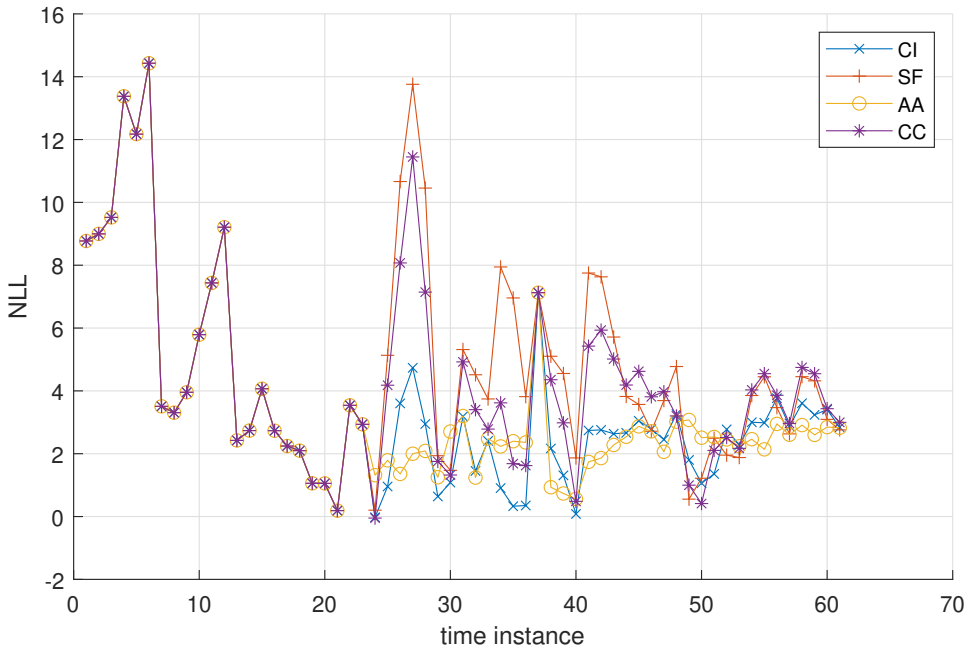


Figure 5.4: NLL values over 61 time instances of four different single-object fusion algorithms.

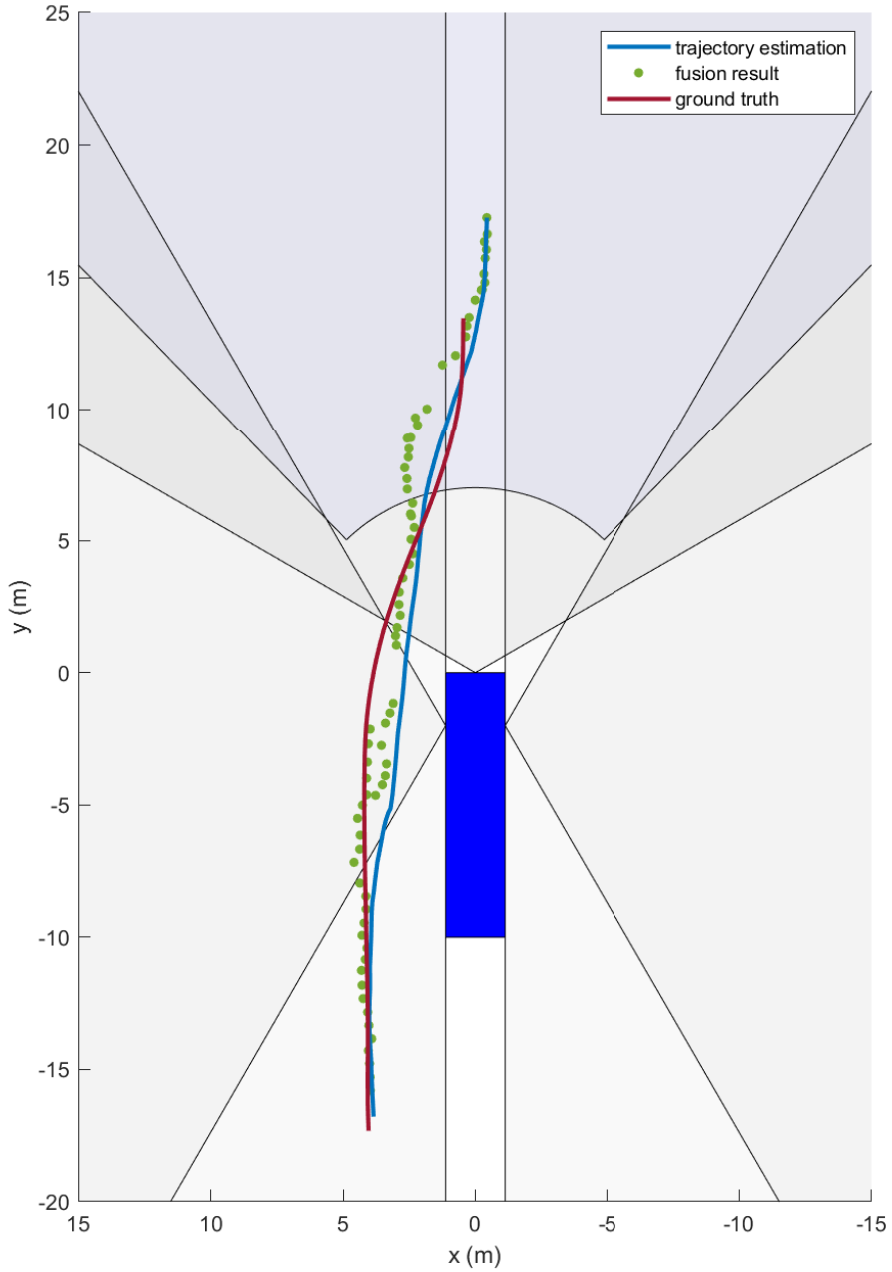


Figure 5.5: The green dots are the fusion results, blue line segments are the trajectory estimations, and ground truth trajectories are marked as red line segments.

6
Discussion

6.1 Simulation and Experimental Comparison

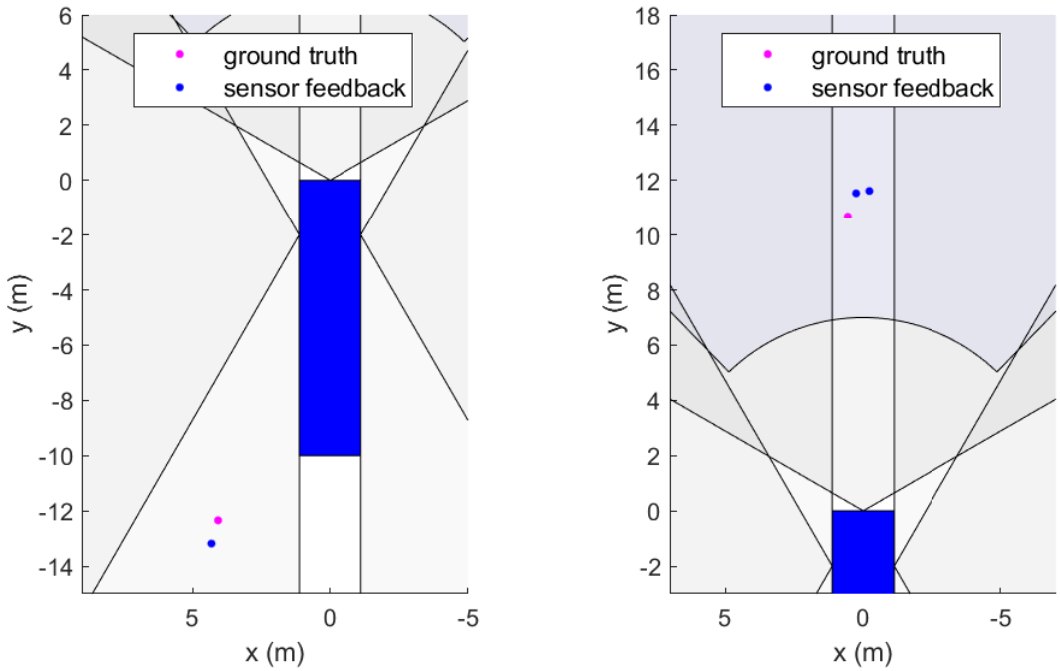
The performance of the simulation surpasses that of the real-world experiment significantly. This discrepancy can be attributed to the configuration of the simulation, which inherently influences its outcomes. However, it should be noted that the simulation falls short of capturing the complete intricacy of the real world.

After analyzing the real-world data, it has been observed that the sensors' low-quality output led to imprecise fusion results. This problem becomes evident when examining specific time instances, as depicted in the accompanying Figure 6.1. Under extreme working conditions, e.g. in Figure 6.1a, only one sensor reports a density (the mean of this density is marked as a blue point). However, this feedback is approximately one meter away from the ground truth. This is caused by the estimation compensation. We transfer all the densities in terms of their mean points all located at the rear-middle point of the object, however, without good knowledge of the size of the object, the compensation might drift away from the ground truth. In Figure 6.1b, there are two densities reported from the sensors, one from the camera and one from the front-looking radar. However, due to the unpredictable time difference, both densities drift away from the ground truth even if we have good compensation.

A noticeable disparity in trajectory estimation can be observed between Figure 4.4 and Figure 5.5. In the simulation, the error levels for both longitude and latitude velocities are closely set, resulting in acceptable velocities for accurate trajectory estimation. However, in the real-world scenario, only the longitude velocity can be deemed reliable, while the reliability of latitude velocity is compromised due to the limitations of the radars. Consequently, in Figure 5.5, the discrepancy between the estimated and ground truth trajectories is acceptable in terms of longitude, but the estimation of latitude is not satisfactory. With improved sensor quality, the trajectory estimation is expected to be more accurate.

6.2 Choice of Single-object Fusion Method

It is hard to draw a concrete conclusion on which single-object fusion method significantly outperforms others. The selection of the method relies on the sensor quality. In cases where it is common to encounter situations where one sensor consistently provides tracks of significantly higher quality than others, i.e. accurate detections



(a) Inaccurate densities returned from left-rear radar

(b) Inaccurate densities mainly returned in front of ego truck

Figure 6.1: Inaccurate sensors densities in two specific time instances

with small covariance, the SF algorithm is a wise choice (see Subsection 4.2.4). However, under this thesis configuration, the qualities of the sensors are at the same level, and the tracks are fused around the ground truth with averaging weights. As a consequence, AA and CI fusion yield better results.

6.3 Simplification in Real-world Applications

In some situations, the covariance estimation is less important. The AA fusion can be a good choice for this application. From the experimental and simulation result, AA required less computational time and the result is acceptable. The AA fusion can directly compute the fused mean estimate by taking the weighted sum. As a comparison, other algorithms require more complex matrix calculation, including SVD or inversion of the covariance matrix. This is more computationally expensive than arithmetic average fusion. In conclusion, the arithmetic average fusion can be a choice for less complex applications.

Another problem that can be seen is that the camera's position estimation is inaccurate. Table 6.1 is the result with the absence of the camera's feedback. Compared to Table 5.1 and 5.2, it can be seen from Table 6.1 that the GOSPA value becomes lower after ignoring the camera's feedback. The covariance intersection algorithm performs better with more accurate sensor feedback.

Table 6.1: GOSPA and NLL values if camera data are ignored

	CI	SF	AA	CC
GOSPA	**0.7920**	0.8544	0.8242	0.8333
NLL	**3.5796**	4.8482	3.6265	4.3793

6.4 Future work

Due to time limitations, we could not fully leverage all the available information, such as object classification and bounding boxes. One possible future work direction is to perform data association based on classification relationships by incorporating object classification information. Additionally, it is also interesting to enhance the understanding of object tracking by introducing another dimension that indicates the spatial location of tracks on the object, which can be achieved through the utilization of bounding boxes.

Furthermore, our data association formulation was limited within individual time instances. However, it may be possible to establish historical relationships between adjacent time instances. This helps to build the data association that cannot be correctly reflected in terms of spatial information. These historical relationships may be accomplished using a graph-based data association approach [36].

7
Conclusion

In this thesis, we propose a multi-sensor multi-Bernoulli densities fusion algorithm with implications for both theoretical understanding and practical applications in the field of ADAS. The result shows that the objects can be tracked accurately under the given experimental environment. Additionally, this fusion algorithm is robust and can be implemented with multiple sensor setups. The trajectories estimated from the fused multi-object densities are accurate compared to the ground truth. As an extension, four single-object fusion methods make this multi-sensor multi-Bernoulli densities fusion algorithm flexible to be adapted to different types of sensors.

Bibliography

[1] C.-Y. Chong, S. Mori, W. Barker, and K.-C. Chang, "Architectures and algorithms for track association and fusion," *IEEE Aerospace and Electronic Systems Magazine*, vol. 15, no. 1, pp. 5–13, Jan. 2000, Conference Name: IEEE Aerospace and Electronic Systems Magazine, ISSN: 1557-959X. DOI: 10.1109/62.821657.

[2] C. Knill, A. Scheel, and K. Dietmayer, "A direct scattering model for tracking vehicles with high-resolution radars," in *2016 IEEE Intelligent Vehicles Symposium (IV)*, Jun. 2016, pp. 298–303. DOI: 10.1109/IVS.2016.7535401.

[3] B. Wang, W. Yi, R. Hoseinnezhad, S. Li, L. Kong, and X. Yang, "Distributed fusion with multi-Bernoulli filter based on generalized covariance intersection," *IEEE Transactions on Signal Processing*, vol. 65, no. 1, pp. 242–255, Jan. 2017, Conference Name: IEEE Transactions on Signal Processing, ISSN: 1941-0476. DOI: 10.1109/TSP.2016.2617825.

[4] S. Matzka and R. Altendorfer, "A comparison of track-to-track fusion algorithms for automotive sensor fusion," in vol. 35, Mar. 25, 2009, pp. 69–81, ISBN: 978-3-540-89858-0. DOI: 10.1007/978-3-540-89859-7_6.

[5] S. Julier and J. Uhlmann, "A non-divergent estimation algorithm in the presence of unknown correlations," in *Proceedings of the 1997 American Control Conference (Cat. No.97CH36041)*, ISSN: 0743-1619, vol. 4, Jun. 1997, 2369–2373 vol.4. DOI: 10.1109/ACC.1997.609105.

[6] S. J. Uhlmann Jeffrey K., "General decentralized data fusion with covariance intersection," in *Handbook of Multisensor Data Fusion*, 2nd ed., Num Pages: 26, CRC Press, 2009, ISBN: 978-1-315-21948-6.

[7] K. Yang, Y. Bar-Shalom, and P. Willett, "Track-to-track fusion with cross-covariances from radar and IR/EO sensor," in *2019 22th International Conference on Information Fusion (FUSION)*, Jul. 2019, pp. 1–5. DOI: 10.23919/FUSION43075.2019.9011439.

[8] G. Fredrik, "Safe fusion of dependent estimates," in *Statistical Sensor Fusion*, 2:2, pp. 32–36, ISBN: 978-91-44-07732-41.

[9] J. Nygards, V. Deleskog, and G. Hendeby, "Safe fusion compared to established distributed fusion methods," in *2016 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, Baden-Baden, Germany: IEEE, Sep. 2016, pp. 265–271, ISBN: 978-1-4673-9708-7. DOI: 10.1109/MFI.2016.7849499. [Online]. Available: http://ieeexplore.ieee.org/document/7849499/ (visited on 01/24/2023).

[10] T. Li, X. Wang, Y. Liang, and Q. Pan, "On arithmetic average fusion and its application for distributed multi-Bernoulli multitarget tracking," *IEEE Trans-*

actions on Signal Processing, vol. 68, pp. 2883–2896, 2020, Conference Name: IEEE Transactions on Signal Processing, ISSN: 1941-0476. DOI: 10.1109/TSP.2020.2985643.

[11] T. Li, Y. Xin, Y. Song, E. Song, and H. Fan, *Some statistic and information-theoretic results on arithmetic average density fusion*, Dec. 31, 2021. arXiv: 2110.01440[cs,math,stat]. [Online]. Available: http://arxiv.org/abs/2110.01440 (visited on 05/10/2023).

[12] Y. Bar-Shalom, "On the track-to-track correlation problem," *IEEE Transactions on Automatic Control*, vol. 26, no. 2, pp. 571–572, Apr. 1981, Conference Name: IEEE Transactions on Automatic Control, ISSN: 1558-2523. DOI: 10.1109/TAC.1981.1102635.

[13] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1, pp. 83–97, 1955, ISSN: 1931-9193. DOI: 10.1002/nav.3800020109. (visited on 05/25/2023).

[14] L. Gao, G. Battistelli, and L. Chisci, "Fusion of labeled RFS densities with minimum information loss," *IEEE Transactions on Signal Processing*, vol. 68, pp. 5855–5868, 2020, Conference Name: IEEE Transactions on Signal Processing, ISSN: 1941-0476. DOI: 10.1109/TSP.2020.3028496.

[15] A. Houenou, P. Bonnifait, V. Cherfaoui, and J.-F. Boissou, "A track-to-track association method for automotive perception systems," in *2012 IEEE Intelligent Vehicles Symposium*, Alcal de Henares , Madrid, Spain: IEEE, Jun. 2012, pp. 704–710, ISBN: 978-1-4673-2118-1 978-1-4673-2119-8 978-1-4673-2117-4. DOI: 10.1109/IVS.2012.6232261. [Online]. Available: http://ieeexplore.ieee.org/document/6232261/ (visited on 01/26/2023).

[16] W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, and T.-K. Kim, "Multiple object tracking: A literature review," *Artificial Intelligence*, vol. 293, p. 103 448, Apr. 1, 2021, ISSN: 0004-3702. DOI: 10.1016/j.artint.2020.103448. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0004370220301958 (visited on 05/08/2023).

[17] Y. Xu, X. Zhou, S. Chen, and F. Li, "Deep learning for multiple object tracking: A survey," *IET Computer Vision*, vol. 13, no. 4, pp. 355–368, 2019, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1049/iet-cvi.2018.5598, ISSN: 1751-9640. DOI: 10.1049/iet-cvi.2018.5598. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1049/iet-cvi.2018.5598 (visited on 05/18/2023).

[18] L. Li, C. Dai, Y. Xia, and L. Svensson, "Deep fusion of multi-object densities using transformer," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Jun. 2023, pp. 1–5. DOI: 10.1109/ICASSP49357.2023.10096214.

[19] R. P. S. Mahler, *Advances in Statistical Multisource-Multitarget Information Fusion*. Artech House, Aug. 1, 2014, 1167 pp., Google-Books-ID: zPBaBAAAQBAJ, ISBN: 978-1-60807-798-4.

[20] K. Granström, M. Fatemi, and L. Svensson, "Poisson multi-Bernoulli mixture conjugate prior for multiple extended target filtering," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, no. 1, pp. 208–225, Feb. 2020, Con-

ference Name: IEEE Transactions on Aerospace and Electronic Systems, ISSN: 1557-9603. DOI: 10.1109/TAES.2019.2920220.

[21] J. L. Williams, "Marginal multi-Bernoulli filters: RFS derivation of MHT, JIPDA, and association-based MeMBer," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 51, no. 3, pp. 1664–1687, 2015.

[22] Á. F. García-Fernández, J. L. Williams, K. Granström, and L. Svensson, "Poisson multi-Bernoulli mixture filter: Direct derivation and implementation," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 54, no. 4, pp. 1883–1901, 2018.

[23] Y. Xia, L. Svensson, Á. F. García-Fernández, J. L. Williams, D. Svensson, and K. Granström, "Multiple object trajectory estimation using backward simulation," *IEEE Transactions on Signal Processing*, vol. 70, pp. 3249–3263, 2022, Conference Name: IEEE Transactions on Signal Processing, ISSN: 1941-0476. DOI: 10.1109/TSP.2022.3184794.

[24] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, ser. KDD'96, Portland, Oregon: AAAI Press, Aug. 2, 1996, pp. 226–231. (visited on 05/09/2023).

[25] S. Kullback and R. A. Leibler, "On information and sufficiency," *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, 1951, Publisher: Institute of Mathematical Statistics, ISSN: 0003-4851. [Online]. Available: https://www.jstor.org/stable/2236703 (visited on 05/11/2023).

[26] M. Fontana, A. F. Garcia-Fenandez, and S. Maskell, "Bernoulli merging for the Poisson multi-Bernoulli mixture filter," in *2020 IEEE 23rd International Conference on Information Fusion (FUSION)*, Rustenburg, South Africa: IEEE, Jul. 2020, pp. 1–8, ISBN: 978-0-578-64709-8. DOI: 10.23919/FUSION45008.2020.9190443. [Online]. Available: https://ieeexplore.ieee.org/document/9190443/ (visited on 06/30/2023).

[27] L. Gao, G. Battistelli, and L. Chisci, "Fusion of labeled RFS densities with different fields of view," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 58, no. 6, pp. 5908–5924, 2022.

[28] J. Munkres, "Algorithms for the assignment and transportation problems," *Journal of the Society for Industrial and Applied Mathematics*, vol. 5, no. 1, pp. 32–38, Mar. 1957, ISSN: 0368-4245, 2168-3484. DOI: 10.1137/0105003. [Online]. Available: http://epubs.siam.org/doi/10.1137/0105003 (visited on 05/11/2023).

[29] F. Bourgeois and J.-C. Lassalle, "An extension of the Munkres algorithm for the assignment problem to rectangular matrices," *Communications of the ACM*, vol. 14, no. 12, pp. 802–804, Dec. 1, 1971, ISSN: 0001-0782. DOI: 10.1145/362919.362945. [Online]. Available: https://dl.acm.org/doi/10.1145/362919.362945 (visited on 05/12/2023).

[30] H. Chen, T. Kirubarajan, and Y. Bar-Shalom, "Performance limits of track-to-track fusion versus centralized estimation: Theory and application [sensor fusion]," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 2,

pp. 386–400, Apr. 2003, Conference Name: IEEE Transactions on Aerospace and Electronic Systems, ISSN: 1557-9603. DOI: 10.1109/TAES.2003.1207252.

[31] Y. Xia, L. Svensson, Á. F. García-Fernández, K. Granström, and J. L. Williams, "Backward simulation for sets of trajectories," in *2020 IEEE 23rd International Conference on Information Fusion (FUSION)*, IEEE, 2020, pp. 1–8.

[32] K. G. Murty, "Letter to the editor—an algorithm for ranking all the assignments in order of increasing cost," *Operations Research*, vol. 16, no. 3, pp. 682–687, Jun. 1968, Publisher: INFORMS, ISSN: 0030-364X. DOI: 10.1287/opre.16.3.682. [Online]. Available: https://pubsonline.informs.org/doi/abs/10.1287/opre.16.3.682 (visited on 05/12/2023).

[33] D. F. Crouse, "On implementing 2D rectangular assignment algorithms," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 52, no. 4, pp. 1679–1696, Aug. 2016, Conference Name: IEEE Transactions on Aerospace and Electronic Systems, ISSN: 1557-9603. DOI: 10.1109/TAES.2016.140952.

[34] A. S. Rahmathullah, Á. F. García-Fernández, and L. Svensson, "Generalized optimal sub-pattern assignment metric," in *2017 20th International Conference on Information Fusion (Fusion)*, Jul. 2017, pp. 1–8. DOI: 10.23919/ICIF.2017.8009645.

[35] J. Pinto, Y. Xia, L. Svensson, and H. Wymeersch, "An uncertainty-aware performance measure for multi-object tracking," *IEEE Signal Processing Letters*, vol. 28, pp. 1689–1693, 2021, Conference Name: IEEE Signal Processing Letters, ISSN: 1558-2361. DOI: 10.1109/LSP.2021.3103488.

[36] D. Touska, K. Gkountakos, T. Tsikrika, K. Ioannidis, S. Vrochidis, and I. Kompatsiaris, "Graph-based data association in multiple object tracking: A survey," in *MultiMedia Modeling*, D.-T. Dang-Nguyen, C. Gurrin, M. Larson, *et al.*, Eds., ser. Lecture Notes in Computer Science, Cham: Springer Nature Switzerland, 2023, pp. 386–398, ISBN: 978-3-031-27818-1. DOI: 10.1007/978-3-031-27818-1_32.

Department of Electrical Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY