Linköping University | Department of Computer and Information Science
Master's thesis, 30 ECTS | Statistics and Machine Learning
2022 | LIU-IDA/STAT-A–22/009–SE

Poisson Multi-Bernoulli Mixture Filter for Multi-Object Tracking in Automotive Systems

Nicolas Taba

Supervisor : Amirhossein Ahmadian
Examiner : Anders Eklund

External supervisor : Suleyman Fatih Kara, Mathias Hallmen

LINKÖPING UNIVERSITY

Linköpings universitet
SE–581 83 Linköping
+46 13 28 10 00 , www.liu.se

Abstract

Multi-Object Tracking (MOT) is the task in which many objects of interest in a scene are identified and tracked through video frames. A main problem in this task is to estimate the number of elements of interest, their position and identity in real-time. A high performing multi-object tracking method in the context of advanced driver-assistance systems is an important step in reaching autonomous vehicle technology. Bayesian filters are a prime candidate to solve this problem as they can express uncertainties on the number of states, the states themselves and the association between detected objects and tracks. This thesis provides empirical evidence of the performance of the Poisson Multi-Bernoulli Mixture (PMBM) filter for multi-object tracking in automotive systems. Objects such as cars and pedestrians are tracked through 150 video scenarios of 20 seconds of varying road scenarios on the NuScenes dataset. The aim of this thesis is to evaluate the performance of the PMBM filter compared to other MOT methods and whether the PMBM filter is competitive against state-of-the-art tracking algorithms. The performance of the PMBM tracking filter is compared against A Baseline for 3D Multi-Object Tracking (AB3DMOT) and 3 state-of-the-art trackers on the NuScenes open dataset. 3 different LiDAR based object detection methods are used as measurement inputs. These are publicly available to be used as an input for the tracking task. The PMBM tracking filter is shown to outperform the baseline on all performance measures and is competitive against state-of-the-art trackers. For the PMBM tracking filter the best Average Multi-Object Tracking Accuracy (AMOTA) is 65.3 and the best Average Multi-Object Tracking Precision (AMOTP) is 68.1 on the validation set of NuScenes using CenterPoint detector.

Acknowledgments

I want to thank my supervisors Suleyman Fatih Kara and Mathias Hallmen at Arriver who helped me navigate the system and who supported me with feedback and encouragement. I would also like to thank Stefan and the RatRods team for welcoming me. I would also like to thank my IDA supervisor Amirhossein Ahmadian for the enlightening discussions and encouragements to push further and investigate.

I want to thank my examiner Anders Eklund and my opponent Syeda Aqsa Iftikhar for the constructive criticism all along my work and challenging it.

I am thankful for all those who supported and helped me during this Master's Program be they in class with me: Alejo, Yuki, Martynas, Keshav, Akshay, Shashi, Rojan, Eleftheria, Rasmus, Siddharth; all the others, out there in Linköping with the whole Ryd crew: Toby, Davide, Maria, Sara, Alex, Giacomo, Andreas, Sarah, Mattias; The Ethologists with the DnD group and garbage TV: Ash, Tomás, Chara, Rebecca and Rita; Pascal for always putting in a good word on my behalf and all the others who helped me along the way. I am very grateful to have met you all and been on this journey all the way to this point.

I would like to thank my family without whose support I would not be where I am and who help me stay grounded. Last but not least, I would like to thank Jeby for keeping me on schedule and healthy and Camille without whom none of this would have been possible.

Contents

List of Figures

List of Tables

1 Introduction

Autonomous vehicles have become a subject of interest in the last decade for the general public in terms of improving road safety and new developments in technology . In order to obtain viable autonomous vehicle technology, high-performing detection of the surrounding environment objects, tracking their dynamic motion and estimating their future position is needed. Detection algorithms have come a long way in the past years thanks to high performance computational resources. Deep learning in real-time and the use of monocular, stereo cameras and LiDAR sensors have also contributed to the increase in detector performance. In the pursuit of full automation of vehicles, one issue to tackle is the tracking of relevant objects in the traffic around the vehicle. These elements are numerous (other vehicles, pedestrians, traffic signs), diverse and changing in number as the autonomous vehicle, also called point-of-view vehicle or ego-vehicle, experiences traffic. Tracking solutions must be able to overcome uncertainties about the number of targets present, their states (trajectories, position) and the uncertainty of associating the correct detected measurement to the appropriate representing object in our tracking scheme [27].

The online multi-object tracking problem is presented broadly in figure 1.1. At each time step, visual data is collected either as RADAR, LiDAR or video information and fed to a detector. This detection model is usually based on deep learning (see section 1.2.1). The detector outputs an application specific state representation of the tracked objects of interest. These can be bounding boxes, information about the position, orientation or appearance (similarity features) of the objects [30]. The represented state of the object can then be evolved or used immediately for data association depending on the tracking approach. During the data association step, each detected state is associated with a previously tracked state. Approaches vary from nearest neighbor matching, cost reduction (Hungarian algorithm or Murty's algorithm) or by similarity score. Once this operation is performed, the state is updated and the information carried by the state is updated to the associated track. Each updated track information is the output of the online multi-object tracking algorithm for that iteration of information collected since the first step.

1.1 Motivation

Multiple Object Tracking (MOT) deals with determining the number and state of objects of interest on the scene after their detection and keeping track of them over consecutive video

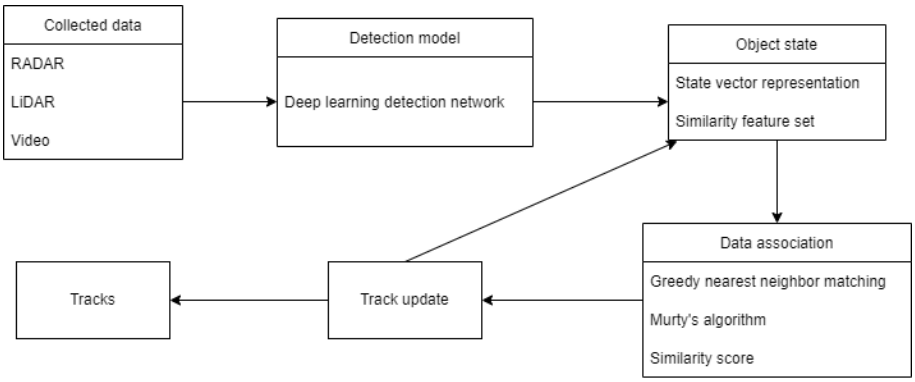


Figure 1.1: Online multi-object tracking problem flow diagram. The listed elements are common approaches used in the field of multi-object tracking and are not an exhaustive list.

frames. Early implementations of trackers to automotive systems have used propagation of detected objects as measurements applied to bayesian filtering methods. More recent approaches using deep learning have combined deep learning for computer vision algorithms for detection feeding into a deep learning tracking network. The latter approach is called end-to-end tracking as it combines both detection and tracking using deep learning. From the perspective of bayesian filters, one of the recently developed approaches in this field for automotive solutions is the Poisson Multi-Bernoulli Mixture (PMBM) filter. This tracking algorithm is based on Bayesian filtering wherein the next state of the object is predicted and then updated as detections are performed successively at each video frame. This filtering method is promising in that it presents good performance at reduced computational costs with no training required in a probabilistic framework. In its framework, the PMBM distributions models both the objects that are detected (false detections and objects of interest detected) as well as the set of undetected objects (occluded, missdetected). The PMBM tracking filter is the first to propose a probabilistic model for these undetected objects and to perform tracking of occluded objects. The PMBM filter is a method that belongs to the category of detection based tracking as it uses processed detector outputs to infer the state of targets; it is online as it updates at each time step and deterministic as the association between measurement and probability distribution yields the same result at each step given that we use the same parameters. [27]

There is a gap in knowledge with regards to the performance of the PMBM filter in real world situations as opposed to simulated data. There is also missing comparisons between the PMBM performance against other tracking methods. Arriver is a Swedish company that specializes in detection, tracking and driving policies for driver assistance systems. Arriver is currently interested in developing knowledge about different tracking algorithms and is the commissioner of this thesis. The thesis aims to investigate the performance of the PMBM tracking filter algorithm in a real life environment.

1.2 Prior work

Multi-object tracking (MOT) is the task that combines detection of targets of interest in a video frame and keep track of identified targets and their trajectories in subsequent frames. MOT is used in different fields like vehicle traffic monitoring for autonomous vehicle technology, crowd monitoring to understand group behavior and interactions or cell movement detection in biology for cell migration and reaction to medication. MOT is composed of two main components: detection of targets on a frame and updating the target tracks [28].

1.2.1 Detection

Although object detection is not the main focus of this thesis, it is an important component of any MOT solution. The stated goal of detection is to identify and discriminate the bounds of objects of interest in a video frame. These targets can be singular objects (vehicles, individual cells) or a cluster of objects (a group of pedestrians walking together, LiDAR point cloud representing an object). State of the art detection approaches rely on neural network architectures and in particular deep convolutional networks for computer vision. They were first applied successfully to real-time applications with high accuracy using so-called "two stage detectors". One of the most popular is called Faster R-CNN [37]. Two stage detectors propose one model to extract regions of objects that are present on a 2D image and another model that classifies and refines the localization of the object on the video frame. These detectors usually come at a greater computational cost than one stage detectors, although some recent advances have shown an improvement in detection using shared features to enable the gradient training to learn more complex features efficiently. More recently, accuracy has improved for one-stage detectors as well as speed in [35] and [36] that cite YOLO9000 and YOLOv3 as one stage detector models. Both of these approaches relied on resolving 2D image data from cameras by processing them only once. Bounding boxes are first assigned as a prior in gradually less fine grid and then refined and classified in an optimization step in a single pass through the network. The difference and speed up compared to multi-stage detection network come from the omission of proposal boxes to be used as a starting point for the regression and classification part of the network.

More recently, some approaches have been proposed to produce bounding boxes for 3D objects using LiDAR and stereo camera data input. They allow for the identification of bounding boxes on 2D or 3D image data after training on pre-processed data [9]. The encoded grid regions are used as inputs for convolutional neural networks (CNN) used for computer vision. The most popular approaches can make use of projections of point clouds onto bird's eye view coordinates [21] in a model called Aggregate View Object Detection (AVOD) model. Depth is encoded in this way as well. The 3D encoding can then be reduced to 2D imaging with a certain amount of feature channels (height, intensity and density)[6][46]. The encoding of 3D point cloud allow the creation of output features about the position of centers, angle of objects with respect to the detector. We thus have a transformation to bird's eye-view output feature set.

One of the most promising approaches encodes the features into point cloud pillars (intensity or density readings) that are then processed by a convolutional neural network and then the data is outputted by a single-shot detection head for the network [22]. This method is called PointPillars. This detection method is used as the baseline to generate the features for the tracking task evaluation on the NuScenes dataset [5]. Another even more successful approach is based on using a 3D feature extractor to reduce the problem to a small number of features for point cloud detection using LiDAR. The features are extracted through a sparse 3D convolution. These features are then used to perform the final predictions in a multi-head network to perform detection. This last method has shown to be the best performing to generate detection dataset to be used for other tasks on NuScenes. This method represents the state-of-the-art in terms of detection to date. [48].

1.2.2 Tracking

Tracking in the sense of MOT deals with trying to answer 3 questions: How many objects of interest are there in view? What is the state of these objects (speed, location, acceleration, angle, type, ...)? To what tracked object does a detector reading belong to? Most of the target tracking approaches have been historically based on probabilistic models applied to filtering. Using a statistical model to explain the probabilistic distribution of states given the measurements collected, one should be able to describe the next state of the target. More recently

another class of popular trackers have come to light in the use of deep learning for tracking [24].

Deep learning approaches use tracking by detection. They combine a neural network for detection and another deep learning network for tracking. The architectures of the tracking networks rely on the use of Recurrent Neural Networks (RNN) and Long-Short Term Memory (LSTM) networks in particular for tracking and data associations [30]. RNNs are able to learn different motion models for the objects detected on a scene and are able to learn one-to-one assignments between frames. When first introduced, this approach was lacking in incorporating robust association strategies such as leveraging appearance (or similarity) as an additional feature [30]. Although this approach was shown to be fast (300 Hz) in its implementation, RNNs require large datasets of various annotated scenarios in order to properly generalize networks after training. In deep learning for tracking two general approaches are used: affinity measures and prediction based approaches. Affinity measures between objects are now used to match appearance, motion or interactions [49] [39]. Here, identities are being kept track of by assigning object identities encoded through either appearance [40] or their motion [8]. Matching patterns between images, motion of objects and other spatio-temporal features can also be used to associate different objects found on different video frames [23].Finally this approach also allows for end-to-end tracking where the network predicts a certain number of steps ahead based on detections made a certain number of previous steps through inference [29].

An issue of approaching tracking from the perspective of neural networks is that these models often fail in term of explainability and in terms of estimating the uncertainties linked to tracking. They provide point estimates to the uncertainties. However, neural networks (NNs) have recently performed very well on a new dataset (NuScenes) and its bi-annual competitions. The best 10 entries rely on NNs according to the tracked metrics [32]. Neural Network training also require an abundance of annotated data in order for training to generalize. The great number of scenarios and updates to the model as new data is generated also require retraining of the model and additional computational resources. The need for all these resources pose the question of the feasibility of integrating this approach in integrated technologies on-board vehicles.

Bayesian filtering is another popular approach that keeps in mind the main uncertainties in tracking tasks. These are namely uncertainties regarding the number of states, the uncertainty around the states of the objects themselves and the uncertainty linked to the data association (identity uncertainty) [41]. Bayesian filters are composed of two steps: prediction and update. At the first step of the filtering algorithm, the probability density of the state of an object is estimated. Then, using a transition density (or motion model), the state density of the object at the next time step is estimated. We observe the state and update the predicted density using the new measurement. This updated density then becomes the prior estimate of the state for the next prediction step. The details of bayesian filters are explained in the theory and background section of this work.

The most popular approaches to MOT using bayesian filtering for tracking in vehicle applications fall under the following categories: **Joint Probabilistic Data Assosciation** (JPDA) filters, **Multi-Hypothesis Tracking** (MHT) and **Random Finite Set** (RFS) tracking[18].

JPDA relies on the association of all track hypothesis of association between measurement and state probability densities and marginal probability distributions calculated under one joint distribution. In this method, all hypothesis for object tracks are considered together under the same joint distribution. This allows for fast computation of a joint probability score to perform tracking. This approach is most popular and efficient in applications where the number of targets is known [38]. In the context of automotive application this approach fails in that the number of objects present in any scenario can be changing. Furthermore, the errors in accuracy are considered as joint which may be sensitive to outliers.

MHT is an approach that relies on reducing the data association problem such that only a limited number of association hypothesis between detected objects and state densities are

considered in the posterior estimation. The true posterior density is approximated by considering only the largest contributors [20]. This approach is heavy computationally as we keep track of all track hypothesis for a few frames before low probability hypothesis are pruned. This can become computationally demanding with more objects on frame. This approach could become intractable in densely populated areas with several different kinds and amount of targets present unless reduction steps are taken to reduce the space of data association combination that are possible.

RFS is a novel approach that relies on RFS theory. This approach has gained traction in the past 10 years and has proven to offer explainable algorithms that perform well in the tasks of tracking [18]. RFS tracking makes use of randomly sized sets of objects whose states are also random variables. The most popular approaches use conjugate prior distributions to find closed form solutions to the filtering prediction and update steps. Techniques such as Multi-Bernoulli Mixture (MBM) filter and δ-Generalized Labelled Multi-Bernoulli (δ-GLMB) are popular under the RFS umbrella and offer a way of modeling detected and clutter (false detections) objects under Multi-Bernoulli distributions. These techniques are said to not measurement driven in that they give rise to distributions that can be described as "phantom" distributions before a newly detected object enter from measurements. The newly born potential targets on scene are generated randomly. In other words, there exists a probability for an object to exist before it has been measured and it does not rise from a potential location from which an object could come into view (detection edges, occluded objects). Both these methods allow us to compute closed form solutions for the filtering problem with slightly different approaches to data association [12].

1.2.3 Poisson Multi-Bernoulli Mixture filter

One recent approach to conjugate prior based filtering algorithms in the field of Random Finite Sets (RFS) is the Poisson Muli-Bernoulli Mixture (PMBM) filter. This filter models detected and undetected objects after measurements using conjugate priors for all distributions involved. This approach has been extensively studied and derived in a seminal paper [12] after its introduction in 2012 [44]. The PMBM filter has now been applied to extended objects (objects that are composed of aggregated detection points such as a point cloud from LiDAR) [45], different approaches to the data association (sets of trajectories)[11] [33] [13] as well as different birth models [4] all on simulated data. Furthermore, the output of this algorithm has been proven to be good in terms of performance and computational time compared to other RFS techniques on simulated data [25] [16].

This algorithm has been applied to a real world scenario for autonomous vehicle tracking in a master's thesis in 2019 [3]. Although the method was fully implemented in the thesis and the work was successful in terms of computational complexity and efficiency on the KITTI dataset [14] [15], the result did not allow for direct empirical comparison between models (namely PMBM filter and NN approaches) due to the small size of the dataset preventing any meaningful training and generalization[3]. Furthermore, the PMBM tracking filter presents some challenges in its implementation in that its parameters are set a priori and usually using some heuristics to cover most cases. Furthermore, the reliance on hypothesis formulation might be detrimental to the speed of execution of this filtering approach compare to more greedy data association algorithms.

This filter presents advantages compared to other Bayesian filters as it models detected and undetected objects in a measurement driven framework while incorporating an evaluation of the uncertainties linked to tracking algorithms. Furthermore, literature related to this filter hint at good performance compared to deep learning approaches with the added benefit of requiring no training. Explainability in conjunction with reduced computational costs, thanks to the use of deterministic data association, make the PMBM filter a prime candidate for a high performing tracking algorithm for autonomous vehicle technology applications [12].

1.3 Aim

There is a lack of empirical study of the PMBM filter applied to real world data using systematic approaches to evaluate the PMBM filter performance against other state of the art solutions to the tracking problem. The aim of this thesis is to evaluate the performance of the PMBM filter compared to other MOT techniques. We implement the PMBM filter on the NuScenes dataset for the first time. We evaluate the performance of the PMBM filter on real world data as well as using different object detection networks. We show that the PMBM is a competitive solution in terms of multi-object tracking and performs similarly to other state of the art solutions. The research questions that will be answered in this work are:

- What is the performance of the PMBM on the NuScenes dataset?
- How does the PMBM performance compare to a baseline tracking method and against other state-of-the-art tracking algorithms?
- How is the performance of the PMBM affected by the quality of the object detector output?

1.4 Delimitations

We will focus on the implementation of the PMBM tracking filter to the NuScenes dataset using the tracking datasets provided. These datasets propose ready-to-use outputs from CNN detection networks. Other open source detector outputs are also used in order to perform performance comparison with other tracking methods. The performance of the PMBM filter will only be compared to other approaches related to the NuScenes dataset (vehicle traffic domain). This work will focus only on the tracking part of the tracker as detector models would warrant a more in-depth work of its own.

1.5 Report structure

We start with chapter 2 on the theoretical framework needed to implement the PMBM filter. We then present a chapter on the implementation of the PMBM filter and methodological considerations to evaluate tracking performance as well as the data that we are using. Results are shown in chapter 4, discussion of these results are led in chapter 5 and finally this work is concluded in chapter 6.

2 Theoretical background

The PMBM filter is a Bayesian filter that relies on probability distributions, Bayesian state estimates and random finite set (RFS) theory. In order to grasp these concepts, we present here the theoretical framework used throughout this work.

2.1 Probability theory

This work focuses on Bayesian interpretation of statistics. In this view, the probability is the degree of belief in an event given certain assumptions (prior belief). This belief is based on prior knowledge and then updated (posterior) upon retrieving additional measurement information (likelihood function). When the prior probability density function has the same functional form than the posterior probability density function, they are said to be conjugate. [2]. The reference [2] is used until section 2.3.

2.1.1 Gaussian distribution

The Gaussian distribution (also called Normal) is a continuous probability distribution for a random variable. Its probability density function is defined by its mean value μ and its variance σ^2:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{2.1}$$

In this thesis, the Gaussian is used to represent the state of the targets. This allows us to take into account uncertainties about the state of the objects itself. Equation 2.1 generalizes to the many dimensional case when $\mathbf{x}$ is a state vector and σ is the determinant of the covariance matrix and σ^2 is the covariance matrix. This is true if the covariance matrix is positive definite (a matrix with all positive eigenvalues) [2].

2.1.2 Poisson distribution

The Poisson distribution is a discrete probability distribution that expresses the number of occurrences of independent events in a time interval. Its probability mass function is defined by the number of occurrences k and the expected value of occurrences λ:

$$f(k,\lambda) = \frac{\lambda^k e^{-\lambda}}{k!} \tag{2.2}$$

This distribution is discussed further in the context of the distribution of undetected objects in 2.3.3

2.1.3 Bernoulli distribution

The Bernoulli distribution is a discrete probability distribution of a random variable representing the probability of a binary outcome. It is defined by a probability p of the event happening:

$$f(k;p) = \begin{cases} p & \text{if } k = 1 \\ 1-p & \text{if } k = 0 \end{cases} \tag{2.3}$$

In the case where there are more than two outcomes, the Bernoulli distribution generalizes to the categorical (multinoulli) distribution. In this distribution, for n different possible outcomes, the sum probabilities p_n of each event add up to 1. The Bernoulli and multinoulli distributions are revisited further in 2.3.4. They are used to model a target's distribution given that the object is either detected ($k = 1$) or not detected ($k = 0$) and to express the probability of a data association hypothesis .

2.2 Bayesian filtering

Bayesian filtering relies on the use of Bayes theorem for its computations. Given a state $\mathbf{x}$, we have a prior belief in our knowledge about this state $p(\mathbf{x})$. The likelihood function $p(\mathbf{z}|\mathbf{x})$ is the probability of obtaining the measurement $\mathbf{z}$ given the state $\mathbf{x}$. Bayes theorem in terms of state vector and measurement vector is then:

$$p(\mathbf{x}|\mathbf{z}) = \frac{p(\mathbf{z}|\mathbf{x})p(\mathbf{x})}{\int p(\mathbf{z}|\mathbf{x})p(\mathbf{x})d\mathbf{x}} \propto p(\mathbf{z}|\mathbf{x})p(\mathbf{x}) \tag{2.4}$$

Bayesian filtering is composed of two steps: a prediction step and an update step. During prediction, the state distribution of the target in the next time step is predicted given the currently available data. During the update step, data is collected and used to update the predicted distribution. The process of Bayesian filtering is presented in figure 2.1.

2.2.1 Prediction

In order to perform the prediction step and obtain the posterior predictive distribution density, the Chapman-Kolmogorov equation is used:

$$p(\mathbf{x}_k|\mathbf{z}_{k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{z}_{k-1})\, d\mathbf{x}_{k-1} \tag{2.5}$$

The predictive distribution is an integral depending on the distribution of the previous state given data available up to the previous time step and $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ which is the transition density. The transition density is also called motion model in the field of Multi-Object Tracking (MOT) and describes the propagation of the object state from one time step to the next.

2.2.1.1 Motion models

Motion models are representations of the dynamics involved in the propagation of the object state in time. They are formally defined as a function applied to a state and some added noise. The matrix representation of a linear motion model is as follows:

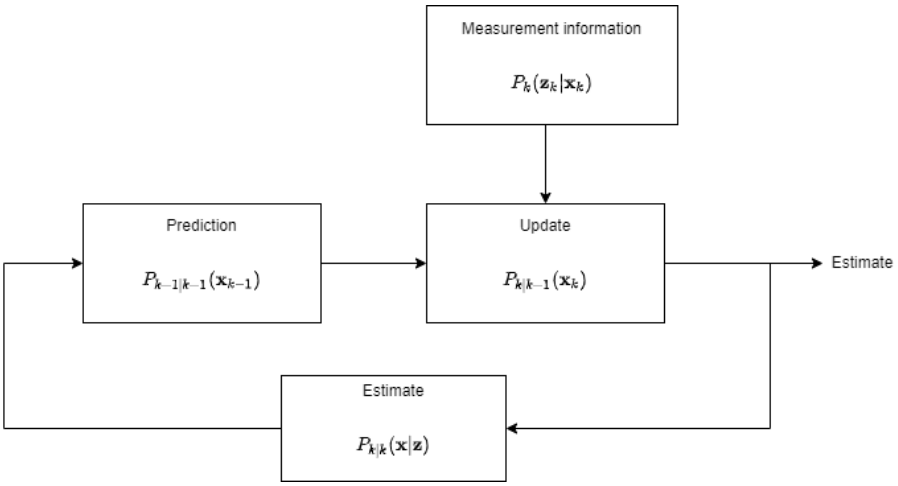


Figure 2.1: Bayesian filter presented schematically. We predict the distribution $P_{k|k-1}(\mathbf{x})$ and update this distribution using information from measurement $P_k(\mathbf{z}_k|\mathbf{x}_k)$. The outputted posterior is the estimate of the state at time step k and used as a prior for the next time step.

$$\mathbf{x}_k = \mathbf{F}\mathbf{x}_{k-1} + \mathcal{N}(0, \mathbf{Q}) \tag{2.6}$$

Q is the variance of the motion noise. Motion models are categorized in linear and non-linear motion models. We will present only the linear motion model that is used in this thesis: the constant velocity motion model.

2.2.1.2 Constant velocity motion model

In this model, the targets are described as having an unchanging velocity between time steps. The velocity may be updated to a new value when measurements are collected. They have as a state vector $\mathbf{x} = [x \quad y \quad \gamma_{pos} \quad \dot{x} \quad \dot{y} \quad \gamma_{vel}]$. This state vector is composed of position and velocity components usually in two dimensions when dealing with MOT as well as rotation components for both the position and the velocity [7]. The noise is usually taken to be gaussian. The position is updated by performing: $([x_{k+1} \quad y_{k+1} \quad \gamma_{k+1}] = \Delta t[\dot{x}_k \quad \dot{y}_k \quad \dot{\gamma}_k])$ and the velocity vector remains unchanged and takes the previous value.

Thus the motion model is defined as:

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.7}$$

When the matrix $\mathbf{F}$ is applied to the state, the state position components are modified with respect to the velocity components. The predicted velocity components remain unchanged during the prediction step. The noise matrix $\mathbf{Q}_{CV}$ for the constant velocity model is:

$$\mathbf{Q}_{CV} = \sigma_Q^2 \begin{bmatrix} \frac{\Delta t^3}{3} & 0 & 0 & \frac{\Delta t^2}{2} & 0 & 0 \\ 0 & \frac{\Delta t^3}{3} & 0 & 0 & \frac{\Delta t^2}{2} & 0 \\ 0 & 0 & \frac{\Delta t^3}{3} & 0 & 0 & \frac{\Delta t^2}{2} \\ \frac{\Delta t^2}{2} & 0 & 0 & \Delta t & 0 & 0 \\ 0 & \frac{\Delta t^2}{2} & 0 & 0 & \Delta t & 0 \\ 0 & 0 & \frac{\Delta t^2}{2} & 0 & 0 & \Delta t \end{bmatrix} \tag{2.8}$$

where Δt is the time increment and σ_Q^2 is a tunable parameter representing the noise in the motion model.

The constant velocity motion model makes the strong assumption that the velocity remains unchanged between time steps. For large enough time steps, this assumption does not hold as objects might change direction or speed between time steps.

2.2.2 Update

The update step is performed using Bayes theorem Eq:2.4 using the predictive distribution Eq:2.5 as a prior:

$$p(\mathbf{x}_k|\mathbf{z}_{1:k}) = \frac{p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_{1:k-1})}{\int p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_{1:k-1})\,d\mathbf{x}_k} \propto p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_{1:k-1}) \tag{2.9}$$

Collected measurement information is included and updates our prior predicted distribution in Eq:2.5. The likelihood function $p(\mathbf{z}_k|\mathbf{x}_k)$ that includes the measurement information is called measurement model.

2.2.3 Measurement model

The measurement model is defined as a function h_k applied to the true state of the object $\mathbf{x}_k$ to which white noise r_k is added to output the detector output measurement $\mathbf{z}_k$:

$$\mathbf{z}_k = h_k(\mathbf{x}_k) + r_k \tag{2.10}$$

This equation can only be evaluated if we have access to the function h_k through estimation. In practice, this function is often set to be a linear transformation and Eq:2.10 becomes:

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathcal{N}(0\ ,\ \mathbf{R}) \tag{2.11}$$

Where R is the noise of the measurement. In this work, the measurement model represents the relationship between the true state of the of the object $\mathbf{x}_k$ and the output of a detector $\mathbf{z}_k$. This linear form allows for straightforward implementation of the update in the Kalman filter.

2.2.4 Kalman filters

The Bayesian filter updates are usually difficult to evaluate due to the normalizing integral in the denominator of bayes theorem Eq:2.4. The Kalman filter is a filter that minimizes the mean square error estimation of this calculation in the case of linear models for motion and measurement [26]. Here we use the motion model transition matrix $\mathbf{F}$, process covariance $\mathbf{P}$, measurement model $\mathbf{H}$ and measurement noise $\mathbf{R}$ to implement the Kalman filter. $\mathbf{Q}$ is still the noise of the motion model as defined in 2.6. We use shorthand notation where measurements from the first timestep to the current timestep are simply marked as k. In the implementation of the Kalman filter, we perform the following operations on state vector $\mathbf{x}$ and measurement vector $\mathbf{z}$.

Using the equations presented in [26], we present the equations needed to use the Kalman filter in the case of linear models for measurement and motion. For the prediction step of the state vector:

$$\mathbf{x}_{k|k-1} = \mathbf{F}\mathbf{x}_{k-1|k-1} \tag{2.12}$$

The predicted covariance for the process is:

$$\mathbf{P}_{k|k-1} = \mathbf{F}\mathbf{P}_{k-1|k-1}\mathbf{F}^T + \mathbf{Q} \tag{2.13}$$

The state prior is then updated such that:

$$\mathbf{x}_{k|k} = \mathbf{x}_{k|k-1} + \mathbf{K}_k\mathbf{y}_k \tag{2.14}$$

The measurement prior is also updated:

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{H})\mathbf{P}_{k|k-1} \tag{2.15}$$

with $\mathbf{y}_k$ being residuals of the measurement and the predicted estimate of the measurement (also called innovation residuals):

$$\mathbf{y}_k = \mathbf{z}_k - \mathbf{H}\mathbf{x}_{k|k-1} \tag{2.16}$$

The update covariance (also called innovation covariance) is the covariance of the residuals of the measurement and the predicted estimate of the measurement:

$$\mathbf{S}_k = \mathbf{R} + \mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^T \tag{2.17}$$

and the Kalman gain is a factor that minimizes the error covariance:

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}^T\mathbf{S}_k^{-1} \tag{2.18}$$

2.3 Random finite sets

Random finite sets (RFS) are used as a representation of varying number of states and measurements in the field of MOT. They have particularly suited property that allow them to represent one-to-one relation between a mathematical model and physical reality. RFS are sets of varying cardinality (number of elements in a set is random) and the elements themselves are random variables. Furthermore, the order of elements in a set is not important. With these properties, a RFS can represent varying number of elements in MOT as well as uncertainties of the states of those elements.

This poses RFSs as a unified probabilistic framework to model all objects in MOT. Using conjugacy between distributions, this theory allows for the derivation of metrics and Bayesian posteriors to account for the fundamental uncertainties in MOT [12]. This section relies on [12] and [3] as references.

2.3.1 Elements of set theory and random finite sets

The union ($\cup$) of two sets is the set of all elements in the first or in the second set (i.e. all elements present in both sets). The intersection ($\cap$) of 2 sets is the set of elements that are in the first and the second set (i.e. elements that are shared between the sets). Two sets are disjoint if their intersection is the empty set. If several sets are disjoint of each other, they are said to be mutually disjoint ($\uplus$ is the mutually disjoint union of sets operator). The cardinality of a set is the number of elements in a set. These definitions are revisited in Fig 2.2. In this figure, set 3, set 4 and the union of set 1 and 2 are mutually disjoint. The cardinality of the union of set 1 and set 2 is 3.

RFS are a randomly sized finite set of random variables. All the elements in the set are unique (no duplicate elements). RFS are invariant to order in the set. Two sets are equal if both sets have the same elements (can also be the empty set). The cardinality and the elements of the set may be distributed according to some probabilistic relation.

2.3.2 Multi-object probability distribution functions

A multi-object probability distribution function (PDF) is a non-negative function on sets that integrates to 1. It is both a description over the elements of the set as well as the distribution of the number of elements of the set (cardinality distribution). They are invariant to order since they describe RFSs

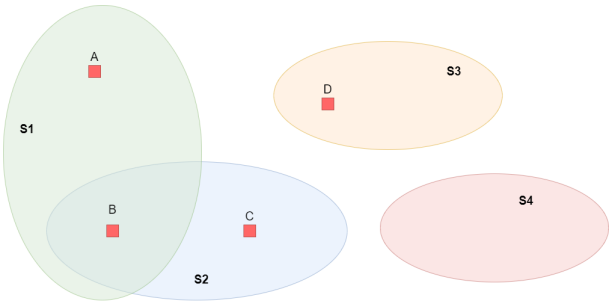


Figure 2.2: Four sets S1, S2, S3 and S4 are presented. The union of S1 and S2 is the set containing points A, B and C. The Intersection of those two set is the set containing B. S3 is disjoint from all other sets and S4 is the empty set.

In order to compute the multi-object PDF, it is important to first set a formula for calculating the PDF of a discrete set that can be separated into mutually disjoint sets using convolutions. The convolution formula that allows to calculate the multi-object PDF for a discrete RFS is:

$$p_{\mathbf{X}}(\mathbf{X}) = \sum_{\mathbf{X}^1 \uplus \cdots \uplus \mathbf{X}^n = \mathbf{X}} \prod_{i=1}^{n} p_{\mathbf{X}^i}(\mathbf{X}^i) \tag{2.19}$$

In this work, sets are marked as capital and bold faced letters. The superscript indicates the index of the set. Equation 2.19 allows, in the case of continuous variables and mutually disjoint subsets, to be used to formulate a set integral:

$$\int f(\mathbf{X})\delta\mathbf{X} = f(\varnothing) + \sum_{i=1}^{\infty} \frac{1}{i!} \int f(\mathbf{x}^1, \ldots, \mathbf{x}^i)\, d\mathbf{x}^1 \ldots\, d\mathbf{x}^i \tag{2.20}$$

Because the cardinality (size) of the set is unknown, we sum over i where i is the number of elements in a set $\mathbf{X}$. In order to clearly state that this integral takes a random finite set as input and outputs a real number, we use $\delta\mathbf{X}$ as integrand as opposed to the usual $d\mathbf{X}$. This set integral allows us in turn to formulate important quantity such as the expected value for RFS:

$$\mathbb{E}[f(\mathbf{X})] = \int f(\mathbf{X}) p_{\mathbf{X}}(\mathbf{X})\delta\mathbf{X} = \sum_{i=0}^{\infty} \frac{1}{i!} \int f(\mathbf{x}^1, \ldots, \mathbf{x}^i) p_{\mathbf{X}}(\mathbf{x}^1, \ldots, \mathbf{x}^i)\, d\mathbf{x}^1 \ldots\, d\mathbf{x}^i \tag{2.21}$$

These set integrals are reused to perform the prediction and update steps for Poisson Multi-Bernoulli Mixture distributions.

2.3.3 Poisson Point Process in RFS

The Poisson Point Process (PPP) is a random process that is defined by its intensity λ. This quantity is the expected number of events or average density of points occurring in a mathematical space like a time interval or some region of space. The number of points in this region is Poisson distributed. The PPP is of interest as it can be used to represent newly appearing (birthed) objects in a region of space that a detector operates in or used to describe the number of undetected objects. The Poisson point process in RFS is formally defined as:

$$p_{\mathbf{X}}(x_1, ..., x_n) = exp(-\overline{\lambda}) \prod_{i=1}^{n} \lambda(x_i) \tag{2.22}$$

where $(-\overline{\lambda})$ is the integral over the intensity function $\lambda(x)$. This intensity function is the Poisson rate.

We can use the PPP to describe the number of undetected objects in a space (cardinality of the set). In this work, the intensity function of the PPP is spatially distributed according to a mixture of Gaussians each representing the state of undetected objects. This allows for statistical model of the set of occluded objects on the scene with regards to their state.

The PPP is also used to describe the first appearance, also called birth, of new objects in the field of view. These newly born objects could either come from an occluded area or appear in the field of view truly for the first time. The birth process in this work also follows a PPP. The intensity or number of births is Poisson distributed and spatially distributed according to a mixture of Gaussians. This modelling of the "birth" of objects allows for objects to (re)appear on screen at this time step.

2.3.4 Multi-Bernoulli Mixture in RFS

For each data association hypothesis associating a set of measurements with a set of distributions, a weight is assigned to this association hypothesis. These weights w^i are normalized and each hypothesis is thus an outcome of a categorical (multinoulli) distribution. Furthermore, each of the Gaussian components of the set of distributions of states are themselves Bernoulli distributed with an existence probability r_j^i.

In the context of RFS, the Multi-Bernoulli Mixture (MBM) of Gaussians is defined as:

$$p_{\mathbf{x}} = \sum_{n=1}^{H} w_h p_{\mathbf{x}}^h(\mathbf{x}) \tag{2.23}$$

where w_h is the weight of the data association hypothesis and $p_{\mathbf{x}}^h(\mathbf{x})$ is the multi-Bernoulli (MB) probability distribution function for the set of Gaussian mixture components that make up the spatial distribution of the states of objects. The summation is performed over data association hypotheses h. This PDF is defined as:

$$p_{\mathbf{x}}^h(\mathbf{x}) = \sum_{\uplus_{i=1}^{N} \mathbf{x}_i = \mathbf{x}} \prod_{j=1}^{N} p_{\mathbf{x}_j}^h(\mathbf{x}_j) \tag{2.24}$$

where $\mathbf{x}_j$ is a set composed of a mixture of Gaussian state vectors and $p_{\mathbf{x}_j}^h(\mathbf{x}_j)$ are Bernoulli probability distribution functions corresponding to the the Bernoulli RFS $\mathbf{X}_i$.

2.4 Poisson Multi-Bernoulli Mixture filter

The PMBM distribution is composed of two distributions that are conjugate distributions [12]). This means that during the prediction, update and final state estimation process, the functional form of the related distributions do not change and remain PMBM distributions.

The PMBM distribution is defined as:

$$\mathcal{PMBM}_{k|k}(\mathbf{X}) = \sum_{\mathbf{X}^d \uplus \mathbf{X}^u = \mathbf{X}} \mathcal{P}_{k|k}(\mathbf{X}^u) \mathcal{MBM}_{k|k}(\mathbf{X}^d) \tag{2.25}$$

As we can see in the previous equation, the PMBM distribution is composed of a PPP $\mathcal{P}_{k|k}(\mathbf{X}^u)$ that describes the distribution over the set of undetected objects and a MBM distribution $\mathcal{MBM}_{k|k}(\mathbf{X}^d)$ that describes the distribution over the set of detected objects. The sub-index $k|k$ represent the indexing of estimation at time step k given all observations up to time k. The PMBM distribution has the advantage that previous equations such as Eq:2.5 and Eq:2.2 generalize naturally in the case of RFS. The Chapman-Kolmogorov equation 2.5 for RFS PMBM becomes:

$$\mathcal{PMBM}_{k+1|k}(\mathbf{X}_k) = \int p(\mathbf{X}_{k+1}|\mathbf{X}_k) \mathcal{PMBM}_{k|k}(\mathbf{X}_k) \delta \mathbf{X}_k \tag{2.26}$$

The Bayes update 2.2 for RFS PMBM is:

$$\mathcal{PMBM}_{k+1|k+1}(\mathbf{X}_{k+1}) = \frac{p(\mathbf{Z}_{k+1}|\mathbf{X}_{k+1})\mathcal{PMBM}_{k+1|k}(\mathbf{X}_{k+1})}{\int p(\mathbf{Z}_{k+1}|\mathbf{X}'_{k+1})\mathcal{PMBM}_{k+1|k}(\mathbf{X}'_{k+1})\delta\mathbf{X}'_k} \tag{2.27}$$

With these tools, we can perform the prediction and update steps accordingly and apply it to the Kalman filter. This aspect is explored further in the following sections for the set of detected objects and the set of undetected objects.

2.4.1 Detected objects in the PMBM filter

To describe the set of detected objects , we use a Gaussian mixture to represent the spatial distribution of states and the components of the vector. For a particular detected object state $\mathbf{x}^d$, we have:

$$\mu_{j,i}(\mathbf{x}^d) = \mathcal{N}(\mathbf{x}^d;\ \bar{\mathbf{x}}^d_{j,i},\ \mathbf{P}^d_{j,i}) \tag{2.28}$$

where $\bar{\mathbf{x}}^d_{j,i}$ is the state estimated mean of $\mathbf{x}^d$ for the i-th state in the j-th association hypothesis and $\mathbf{P}^d_{j,i}$ is the covariance matrix of that state

Using the equations 2.12 and 2.13 for the Kalman filter prediction, we get for the predicted state estimate:

$$\mu_{j,i}(\mathbf{x}^d) = \mathcal{N}(\mathbf{x}^d;\ \mathbf{F}\bar{\mathbf{x}}^d_{j,i},\ \mathbf{F}\mathbf{P}^d_{j,i}\mathbf{F}^T + \mathbf{Q}) \tag{2.29}$$

After the prediction step, the existence probability of an object is updated by multiplying it with the survival probability p_s:

$$r^d_{j,i} = p_s r^d_{j,i} \tag{2.30}$$

For the update step, we must consider two different kinds of input in the set of detected objects: the set newly born objects and misdetection (clutter $c(\mathbf{z})$) and the set objects that were previously detected. These two sets of objects are treated differently. We start with the case of objects that are born onto the scene.

After the update step, the existence probability, if the target has just been born, is updated:

$$r^d_{j,i} = \frac{p_d \sum_{i=1}^{N_u} w^u_{j,i}\mathcal{N}(\mathbf{z};\ \hat{\mathbf{x}}_{j,i}(\mathbf{z}),\ \hat{\mathbf{S}}_{j,i})}{p_d \sum_{i=1}^{N_u} w^u_{j,i}\mathcal{N}(\mathbf{z};\ \hat{\mathbf{x}}_{j,i}(\mathbf{z}),\ \hat{\mathbf{S}}_{j,i}) + c(\mathbf{z})} \tag{2.31}$$

In the context of the Kalman filter, the state distribution for the targets that have been born is updated such that:

$$p(\mathbf{x}_{born}|\mathbf{z}) = \sum_{i=i}^{N_u} \hat{w}_{j,i}\mathcal{N}(\mathbf{x};\ \hat{\mathbf{x}}_{j,i}(\mathbf{z}),\ \hat{\mathbf{P}}_{j,i}) \tag{2.32}$$

where the weight $\hat{w}_{j,i}$ is:

$$\hat{w}_{j,i} \propto w_{j,i}\mathcal{N}(\mathbf{z};\ \hat{\mathbf{x}}_{j,i}(\mathbf{z}),\ \hat{\mathbf{S}}_{j,i}) \tag{2.33}$$

and state estimate $\hat{\mathbf{x}}_{j,i}(\mathbf{z})$ is:

$$\hat{\mathbf{x}}_{j,i}(\mathbf{z}) = \bar{\mathbf{x}}^u_{j,i} + \hat{\mathbf{K}}_{j,i}\hat{\mathbf{S}}^{-1}_{j,i}(\mathbf{z} - \mathbf{H}\bar{\mathbf{x}}^u_{j,i}) \tag{2.34}$$

and covariance estimate $\hat{\mathbf{P}}_{j,i}$ is:

$$\hat{\mathbf{P}}_{j,i} = \mathbf{P}^u_{j,i} - \hat{\mathbf{K}}_{j,i}\hat{\mathbf{S}}^{-1}_{j,i}\hat{\mathbf{K}}^T_{j,i} \tag{2.35}$$

and Kalman gain:

$$\hat{\mathbf{K}}_{j,i} = \mathbf{P}^u_{j,i}\mathbf{H}^T \tag{2.36}$$

and innovation

$$\hat{\mathbf{S}}_{j,i} = \mathbf{H}\mathbf{P}^u_{j,i}\mathbf{H}^T + \mathbf{R} \tag{2.37}$$

In the case that the object state was detected in the previous step, the survival probability becomes:

$$r^d_{j,i} = \frac{r^d_{j,i}(1 - p_d)}{1 - r_{j,i} + r_{j,i}(1 - p_d)} \tag{2.38}$$

where p_d is the detection probability of any object on scene. furhermore, the weights are updated such that:

$$w^d_{j,i} = w^d_{j,i}(1 - r_{j,i} + r_{j,i}(1 - p_d)) \tag{2.39}$$

With regards to the Kalman filter predictions and updates, the equations are similar to the ones presented above but over the set of previously detected objects instead of the set of objects coming from the PPP.

The existence probability becomes 1 since the object is seen:

$$r^d_{j,i} = 1 \tag{2.40}$$

The updated weight estimates is:

$$w^d_{j,i} = w^d_{j,i} r_{j,i} p_d \mathcal{N}(\mathbf{z}; \bar{\mathbf{x}}^d_{j,i}, \hat{\mathbf{S}}_{j,i}) \tag{2.41}$$

2.4.2 Undetected Objects

As discussed in 2.3.3, the PPP is defined by its intensity lambda and the spatial distribution of the points is a mixture of Gaussian in this work. This mixture is defined as:

$$\lambda^u(\mathbf{x}^u) = \sum_{i=1}^{N_u} w^u \mathcal{N}(\mathbf{x}^u; \bar{\mathbf{x}}^u, \mathbf{P}^u) \tag{2.42}$$

Similarly, for new targets born from the PPP, we have:

$$\lambda^b(\mathbf{x}^u) = \sum_{i=1}^{N_u} w^b \mathcal{N}(\mathbf{x}^b; \bar{\mathbf{x}}^u, \mathbf{P}^b) \tag{2.43}$$

During the prediction step of the PMBM filter, the set of all undetected objects is estimated using the following equation:

$$\lambda^u_{k+1|k}(\mathbf{x}^u) = \lambda^b(\mathbf{x}^u) + p_s \sum_{i=1}^{N_u} w^u \mathcal{N}(\mathbf{x}^u; \mathbf{F}\bar{\mathbf{x}}^u_i, \mathbf{F}\mathbf{P}^u\mathbf{F}^T + \mathbf{Q}) \tag{2.44}$$

where p_s is the survival probability of the undetected distribution from one time step to the next. At the update steps, since the measurements are undetected, we simply update the probability of the states remaining undetected by the complement of the survival probability:

$$\lambda^u_{k+1|k+1}(\mathbf{x}^u) = (1 - p_d)\lambda^u_{k+1|k}(\mathbf{x}^u) \tag{2.45}$$

When the weights associated to the distribution of an undetected object becomes smaller than a threshold Γ_s , the distribution is removed and the target "dies".

2.4.3 Hypothesis

At each time step there are 3 ways of associating the measurements: the measurement is a previously detected target, the measurement is a new target or the measurement is a faulty detection (clutter). We establish a set of hypothesis in order to take into account all of these possibilities in the context of MOT and the PMBM tracking filter in particular. This work focuses on the more computationally efficient approach to hypothesis called track-oriented hypothesis as opposed to the hypothesis-oriented approach [17].

In figure 2.3 the track-oriented approach to building a data association hypothesis is presented. At time step 1 two objects are detected and are associated to the track of object x^1 or

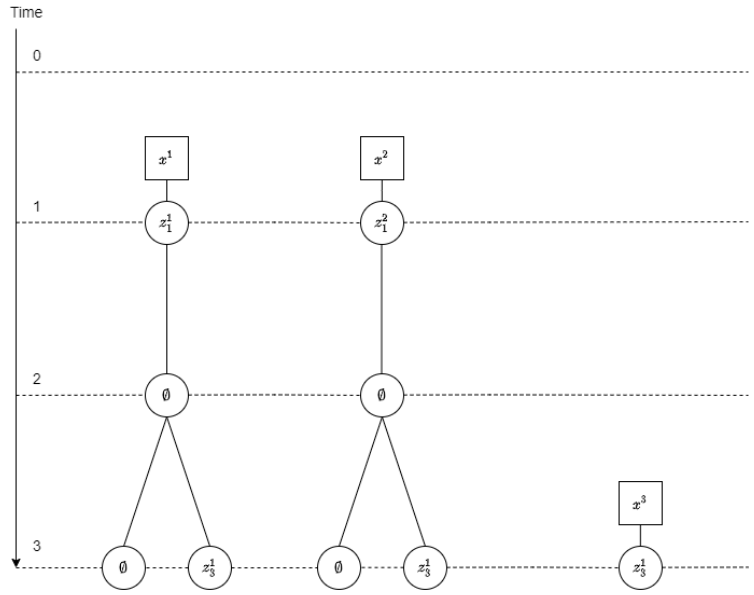


Figure 2.3: Tree structure of track-oriented hypothesis for single targets [3]. This figure presents the global hypothesis for associating a set of measurements after 3 time steps. At time step 3, we may only associate measurement z_3 with either of the potential objects x or a new object x^3.

x^2. At time step 2, there is no detected object. At time step 3, there are three possible data association for the single detection z_3. The measurement z_3 can be associated to the first or second track or it could be the start of a new track x^3. The data association only accepts one of these outcomes. The collection of probabilities for the data association at time step 3 make up the single target hypothesis (STH) for an object detected at time 3. The collection of possible STH at all time step make up a global hypothesis. In this example, at time step 3, there are 3 STH.

To solve the data association problem, we establish a cost matrix for each global hypothesis. Each row of this cost matrix corresponds to a set of measurements at that time step and each column is a previously detected object or a new track. In order to solve the data association, an optimal assignment algorithm solves the cost matrix to find the one with minimal cost.

$$L^h = \begin{bmatrix} -l^{1,1,h} & -l^{1,2,h} & \dots & -l^{1,N^h,h} & -l^{1,0} & \inf & \dots & \inf \\ -l^{2,1,h} & -l^{2,2,h} & \dots & -l^{2,N^h,h} & \inf & -l^{2,0} & \dots & \inf \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ -l^{m_k,1,h} & -l^{m_k,2,h} & \dots & -l^{m_k,N^h,h} & \inf & \inf & \dots & -l^{m_k,0} \end{bmatrix} \tag{2.46}$$

where m_k is the time index at which measurements are made, N^h are the STH and h is the hypothesis index for the global hypothesis. The right side of the matrix represents the new

track hypothesis. We use log weights in our implementation for computational stability:

$$l^{j,i,h} = \log(w_{j,i}^d) - \log(w_{0,i}^d) = log\left(\frac{w_{j,i}^d}{w_{0,i}^d}\right) \tag{2.47}$$

where we have the difference between the single target hypothesis-measurement pair weight and the misdetection weight. New detections have weights calculated according to Eq.2.33:

$$l^{j,0} = \log(w_{j,0}^d) = -log(\lambda_j(\mathbf{z})) \tag{2.48}$$

Murty's algorithm solves the data association problem by finding the best assignment between the measurements and targets that minimize the cost of the matrix. We use this algorithm to create new global hypotheses [31].

2.4.4 Reduction

The previous cost matrix grows exponentially with the number of measurement steps that we make [10] and we wish to reduce the computational complexity of the association problem. In order to do so, three techniques are used: pruning, gating and recycling.

We perform pruning of the global hypothesis, Poisson and MBM distributions. We perform pruning by discarding elements of these hypothesis or distributions when their associated weights are below a threshold Γ. In other words, when $w < \Gamma$ we set the weight to 0.

Measurements that are far from the center of the detected distribution have a very small probability of being associated together. Considering this fact, we can perform an operation called gating wherein we consider only measurements within a certain distance from the distribution. The most common gating procedure is called ellipsoidal gating. In ellipsoidal gating we consider only measurements that are at a smaller distance than a certain number of standard deviation G. The distance is computed using the Mahalanobis distance that takes into account the innovation covariance to encompass uncertainties regarding the predicted state and the measurement:

$$d_M^2 = [\hat{z}_i - z_j]^T \mathbf{S}^-1 [\hat{z}_i - z_j] \tag{2.49}$$

If a measurement is outside of the gate, its data association weight is set to 0.

Some distributions are close to some measurements, but do not end up being associated with any. We wish to discard them from consideration for the purpose of the data association step, but retain that information in case the measurement appears again in a future step. To solve this problem, we use a process called recycling. When recycling, the distributions of detected objects with an existence probability below a threshold Γ_r are moved to the undetected distribution. At this step, the undetected density becomes:

$$\lambda_{k|k}^u(\mathbf{x}^u) = \lambda_{k|k}^u(\mathbf{x}^u) + \sum_{i:r_{k|k}^i < \Gamma^r} w_{j,i}^d r_{k|k}^i \mu_{j,i}(\mathbf{x}_{j,i}) \tag{2.50}$$

Since hypotheses do not need to be unique,we may also merge similar hypothesis without affecting performance. When merging hypothesis (single target or global), their weights have to be added so that they still all add up to 1.

2.4.5 State estimation

The state estimation is straightforward in its implementation. We choose the global hypothesis that has the highest weight (lowest loss). For this hypothesis, the mean value of the states of the MBM are considered to be the output of the filter at this time step and used as a prior for the next iteration of the filter. The mean value is chosen following the maximum likelihood estimate criterion. States are only considered if they have an existence probability that is

greater than a threshold Γ. In the estimation, the survival probability p_s, detection probability p_d and the threshold Γ all influence the length of time that an object can remain undetected and still be linked to a previous track once detected again.

We present the functioning of the PMBM filter schematically in figure 2.4. In this figure, the PPP and MBM components of the PMBM distribution can be considered mostly as independent. The prediction step is performed using a motion model and the update step is performed using the measurement model. The "death" of an object happens when distributions existence probabilities $r_{j,i}$ fall below a threshold Γ_s. New targets are birthed according to the spatial distribution of the PPP and become part of the detected object distribution. The reduction process allows the data association to remain computationally tractable by performing pruning, gating and recycling. Distributions that are not associated with any measurements can be moved to the undetected distribution by recycling.

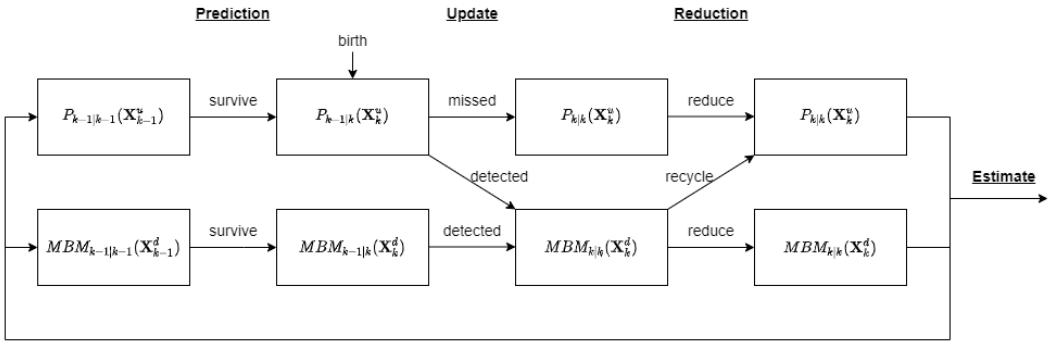


Figure 2.4: Schematic representation of the PMBM filter. The prediction is performed using a motion model and the update step incorporates the measurement information. The recycling process allows to retain information about distributions not associated with any measurements.

3 Method & Data

In this chapter, the implementation of the PMBM tracking filter, the data used and the evaluation methods are discussed.

3.1 PMBM implementation

The PMBM tracking filter is implemented following the pseudocode 1 first presented in [12]. The code for the tracking filter was implemented in Python (3.8) using basic libraries (Numpy, Scipy and Pandas) as well as leveraging some previous work. An implementation of Murty's algorithm for data association in C++ with a Python wrapper [3] is leveraged and the API of the NuScenes dataset that is provided as an open source development kit for dataset users [5] is also used in this implementation. This API allows for easy loading and manipulation of the data as well as automated evaluation of the saved output of the tracker. The downside of using the API is that for any use of the dataset, each split has to be fully loaded in memory. This is memory inefficient but is a requirement of this API and discussed in 4.1.1.

The motion model used is the constant velocity motion model parametrized as in 2.2.1.2. For matrices that require to be inverted in the Kalman filter, we ensure computational stability and retaining the positive semi-definite property by using the following procedure for any matrix $\mathbf{S}$ that must be inverted:

$$\mathbf{S} = \frac{1}{2}(\mathbf{S} + \mathbf{S}^T) \tag{3.1}$$

Log-weights are implemented to avoid underflow and are considered throughout for computational stability. Furthermore rotations are represented as a rotation axis (vector) and angle. This vector representation of the rotation of an object is transformed and reduced to $[-\pi, \pi]$ angle interval with respect to the point-of-view car (ego vehicle). Tracking boxes at the next time step are predicted to be of the same dimension than at the current time step. They then take the average value between the associated measurement detection bounding box and the predicted bounding box during the update. This incurs bounding box size errors when objects (in particular vehicles) turn, but does not affect the position of the center of the object and thus does not affect any performance measure that includes the position of the center of the object in its computation. The PMBM algorithm state estimation outputs only

Algorithm 1 PMBM tracking filter pseudocode

Input: Previous state estimate and measurements Z of current time step
Output: Estimate of the PMBM distribution at this time step
- Perform prediction as presented in 2
for $z \in Z$ **do** ▷ newly detected targets update
    - Perform gating on z with respect to the poisson spatial distribution
    if z is within the gate of at least one component **then**
        - Create a new target in the Multi-Bernoulli Mixture
    end if
end for
for $i = 1 \rightarrow n$ **do** ▷ all targets update
    for $j_i = 1 \rightarrow l_i$ **do** ▷ all single target hypotheses of a given target
        - Create new misdetection hypothesis
        - Perform gating on Z and create new detection hypotheses
    end for
end for
- Recycle unassigned distributions
for all j **do** ▷ all global hypotheses
    - Create cost matrix
    - Run Murty's algorithm with the cost matrix as input to select k new global hypotheses
end for
- Estimate the target states
- Prune the Poisson part by discarding components whose weight is below the survival threshold
- Remove Bernoulli components whose existence probability is below a threshold or do not appear in the pruned global hypotheses

the center of the tracked objects in its output features. We include the bounding boxes for ease of evaluation and qualitative analysis.

Each new object is assigned to a unique track ID. This track ID contains the information about the state of an object (mean value of distribution) and the uncertainty associated (covariance matrix associated with the state). The information about the new object is stored in the single target hypothesis. The association weight, time index of birth, associated measurement index and existence probability is also stored in the single target hypothesis as discussed in the theory section of this thesis. With a growing number of single target hypothesis, it is important to assign a single target ID with respect to a track. Each track contains all the single target hypothesis, the class of the object and the time of birth. The class parameter in this track data container can allow us to use different motion model covariances depending on the class of the object. This organization of track and single target hypothesis allow the use of a lookup table in the global hypothesis step where only the track information and the single target ID are needed to point to the correct information. Furthermore, the memory cost can be reduced by removing single target hypothesis that are not considered in the global hypothesis.

For the first iteration of the algorithm, the cost matrix creation is skipped and all new possible targets associated to a measurement are considered as new tracks. The number of new global hypothesis is chosen by design (see appendixA). The state estimate is computed by finding the global hypothesis with the highest weight and choosing all single target hypothesis that have a probability of existence larger than the existence pruning threshold.

3.2 Parameter tuning

There are two sets of parameters that require tuning. Parameters that concern the Kalman filter process of prediction and update and the other parameters (such as survival probability, detection probability existence pruning threshold, etc...) that concern the other steps of the PMBM tracking algorithm. The Kalman filter process parameters that are tuned are *the motion* and *the measurement noise matrices*. In this implementation, we use the average values for covariance matrices for each class of object of interest over the training set [7] as initial values for those noise covariance matrices. This allows for different initial motion parameters for each class of objects. The other parameters are informed by expert input and tuned further through heuristics. The set of optimal parameters that are used can be found in the appendix A. The initial operating parameters that were found to work for this implementation were heuristically chosen using visual evaluation of a sequence of the training dataset split that presented sufficient complexity. The sequence chosen is a road crossing with passing vehicles (cars and a turning bus) as well as a bicycle in good visibility condition.

No sequence of the evaluation set were used for parameter tuning. Both the Kalman filter parameters and the PMBM algorithm process parameters are tuned on training set sequences.

3.3 Evaluation method

Performance measures must allow the evaluation of the different aspects of multi-object tracking performance. They must allow to evaluate the performance of the tracking approach with respect to the the number of objects of interest on the scene that are correctly identified and tracked, the overlap of the tracking boxes, the correct association between measurement and predicted box, the length of tracking against the ground truth and the loss of the tracked object.

For NuScenes, the evaluation method that is used by the tracking community is a set of performance measures that address the aforementioned requirements called the CLEAR-MOT suite [1]. The important measures of this suite are the AMOTA and AMOTP based on MOTA and MOTP [43].

MOTA: Multi-Object Tracking Accuracy is the measure that combines false positives, missed targets and identity switches at the best recall threshold:

$$MOTA = 1 - \frac{\sum_t (IDS_t + FP_t + FN_t)}{\sum_t TP_t} \tag{3.2}$$

where IDS_t are the identity switches, FP_t are the falsely tracked objects, FN_t are the targets that are missed and TP_t are the ground truth tracked objects at time t. *IDS* occurs when an object is wrongfully assigned as a new target or to the wrong estimated state. This measure gives an indication of the trackers performance in keeping the right track regardless of the precision in the estimation of the position of the tracking boxes. This measure takes into account the identity of the tracked objects. The MOTA can then be averaged over the sequences that are evaluated.

MOTP: Multi Object Tracking Precision is the misalignment errors between the annotated ground truth and the tracking boxes at the best recall threshold:

$$MOTP = \frac{\sum_{i,t} d_{i,t}}{\sum_t TP_t} \tag{3.3}$$

where $d_{i,t}$ is the displacement error between the tracked object i and its associated ground truth. Displacement errors are calculated such as the closest distance between the ground truth bounding box center and the center of the closest tracked object. This measure indicates the performance of the tracker in its estimation of the position of objects of interest independent of its ability to correctly classify and associate objects of interest with the correct trajectory.

The main metrics are AMOTA and AMOTP developped in [5]. These are the integrals over the MOTA and MOTP curves respectively using n-point interpolation (n=40 arbitrarily). Points with recall smaller than 0.1 are not considered as they are typically noisy and can lead to negative values of MOTA [5].Furthermore, the recall value thresholds are calculated by changing the number of ground truth objects that are considered in a sequence in order to match the desired recall threshold.

AMOTA: Average Multi Object Tracking Accuracy is the average over the MOTA metric at different recall thresholds over the entire sequences.

$$AMOTA = \frac{1}{n-1} \sum_{r \in \{\frac{1}{n-1}, \frac{2}{n-1}, \ldots, 1\}} \max(0,\ 1 - \frac{IDS_r + FP_r + FN_r + (1-r) * TP}{r * TP}) \tag{3.4}$$

In equation 3.4, a recall- normalization term $(1-r) * P$ is included in the numerator, the factor r in the denominator. This enforces that the values of the AMOTA span the entire $[0, 1]$ range. TP is the number of true positives for the current class that is evaluated.

AMOTP: Average Multi-Object Tracking Precision is the average over the MOTP metric defined before at different recall thresholds. Here $d_{i,t}$ indicates the position error of a track i at time t and TP_t indicates the number of matches at time t.

$$AMOTP = \frac{1}{n-1} \sum_{r \in \{\frac{1}{n-1}, \frac{2}{n-1}, \ldots, 1\}} \frac{\sum_{i,t} d_{i,t}}{\sum_t TP_t} \tag{3.5}$$

A low AMOTP indicates that the center of tracked objects are properly placed with respect to ground truth annotations. We posit that the diversity of the types of situations and the number of sequences on the evaluation set is sufficient to be representative of the population mean behavior when analyzing the metrics of the different trackers. Furthermore, all trackers do not make available the source code in order to perform the evaluation of the sequences on truncated samples of the evaluation in order to extract the mean and standard deviation of the performance measures within the time limits of this thesis. The conclusions of comparison between methods can be seen as mean behavior indicators when comparing performance.

Other performance measures of interest are:

- **MT**: Mostly Tracked is the number of tracked trajectories that cover at least 80% of the total length of the ground truth track of the object.
- **ML**: Mostly Lost is the number of ground truth trajectories that are covered by the by a track hypothesis for at most 20% of the length of the track.
- **FP**: Number of false positives (clutter detections).
- **FN**: Number of false negatives or targets that are missed and not tracked.
- **IDS**: Identity Switches is the total number of identity switches. This performance measure contains identity information.

The IDS, MT and ML performance measures allow us to identify if a particular tracking approach is tracking the objects of interest for the whole length of sequences. These performance measures are best used when comparing between methods. The FP and FN measures give some indication on the performance of the data association and the filtering of clutter.

Automatic evaluation through the evaluation API outputs these performance measures as well as recall curves against all performance measures and per object class. MOTA/MOTP against recall curves represent the trade-off between a class classification threshold and the performance measure of interest. The discriminating threshold (recall) between one class and the others may affect how many error terms are considered in the calculation of the performance measure.

All methods that are compared using the same evaluation approach on the same ground truth. Fair comparison between tracking methods can thus be performed if the detector outputs used are the same between methods.

3.4 Data

The dataset used in this work is an open source dataset called NuScenes provided by the Motional company [5]. This dataset was inspired by the KITTI dataset [14] [15] with the aim to expand in scope. The KITTI dataset proposes a high sampling rate (30Hz) of 20 sequences of 20 seconds of annotated video data. NuScenes is composed of 1000 samples of 20 seconds and data is collected through various detectors. There are 6 cameras sampling at 12Hz, 1 spinning LiDAR sampling at 21 Hz, 5 RADAR sensors collecting data at 13Hz, 1 Intertial Measurement Unit (IMU) collecting speed and orientation information and GPS sensor.

Data are collected in 4 different locations in Singapore and Boston: Singapore Queenstown, Singapore One North, Singapore Holland Village and Boston Seaport, in different driving conditions such as dense city, suburb, highway, different visibility conditions, weather conditions and combination thereof. The dataset data structure is presented in figure 3.1 with information relevant to the implementation of the PMBM tracking filter.

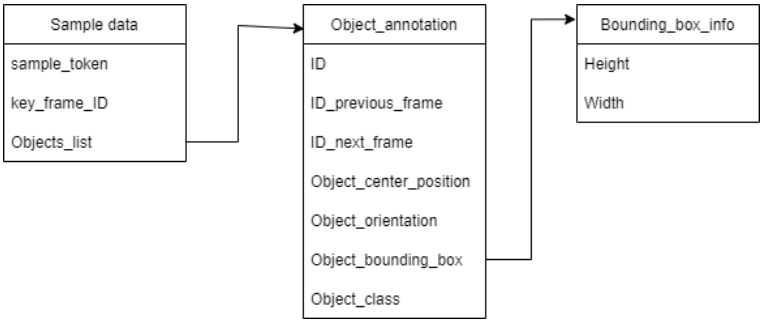


Figure 3.1: Dataset structure. Units of distance are in real-world coordinates and rotations with respect to the driving vehicle.

The classes of interest in the tracking task are cars, pedestrians, trucks, trailers, buses, bicycles and motorcycles. The most common classes are cars and pedestrian. The least frequent class are trucks, trailers and motorcycles. In order to balance class frequency, more scenes including rare classes such as bicycles are added to the dataset. The scenes are annotated by human experts on key frames that are marked as being every 2Hz in frequency. On average there are 40 key frames per scene. These key frames are also used for evaluation.

In this work, the focus is centered on estimating the position, orientation and identity for each object of interest on a scene given detection data consisting of position of the center of object of interest, size of detection boxes and class assignment probabilities. The variables coming from detector output are discussed further in 3.5.

Although the sampling rate is higher, the data that can be used is limited. 40 key frames are made available for the tracking task for each sequence. Furthermore, the ground truth (GT) is only accessible for evaluation on 850 video sequences (training and validation sets) as the last 150 sequences of the test set are reserved for officially competing tracking methods. Finally, due to computational costs, we are only able to perform evaluation and experiments on 150 sequences of the evaluation set. Computational costs and limitations are discussed in 4.1.1.

3.5 Detection dataset

The detector output data used are the output of detection neural networks. In this work, we focus on the use of the MEGVII [48], MapPillars [22] and CenterPoint [47] detections for the NuScenes dataset. These detections use only LiDAR point clouds as inputs and output the detections for the frames of interest at 2Hz. These detections are composed of a classification label and a box regression. We will only use detection sets that use LiDAR data only as this work does not focus on sensor fusion. The data used is thus in real world coordinates and not in image coordinates like RGB image data.

For detections, the performance of detectors is evaluated by the mean average Precision (mAP) and the NuScenes Detection Score (NDS). The mAP is defined as the integral over the precision versus recall curve. In this definition precision and recall are:

$$precision = \frac{TP}{TP + FP} \tag{3.6}$$

$$recall = \frac{TP}{TP + FN} \tag{3.7}$$

Where TP are the true positive detection of objects, FP (false positive) are the falsely detected objects, and FN are the false negative detections. The mAP is thus the averaged value of the different integrals for each object class over different recall values. The NDS is defined as the weighted sum of different types of errors that include translation errors between detection boxes and ground truth, the mAP, orientation errors, velocity errors and accuracy (classification probability) errors. For both of these performance measures, the higher the value, the better the performance. We present in table 3.1 the reported performance [5] [47] of the detections that will be used in this work.

Detection method	mAP ↑	NDS ↑
Megvii	51.9	62.8
CenterPoint	**58.0**	**65.5**
PointPillars	29.5	44.9

Table 3.1: Detection sets and their performance measure

The MEGVII detections are provided by the NuScenes dataset providers. The detections of the MEGVII detection set are the output of a two-staged detector. Features are extracted using a 3D feature extractor and then sent to a region proposal network before a multi-head group performs classification, box regression and orientation estimation[48] as presented in section 1.2.1 of the introduction chapter. These detections are publicly available and provided by the dataset makers. These detections are used in the baseline method AB3DMOT [43] and in StanfordIPRL-TRI tracking methods [7].

CenterPoint detections are also produced using a two-stage detector [47]. A 3D feature extractor network extracts map-view features. Using a detection head (a 2D convolutional neural network), the first stage of the detector produces object centers and 3D box proposals. Finally, using center features a Multi-layer Perceptron (MLP) predicts a confidence score and refines the box regression [47]. These detections are used in the SimpleTrack and in the CenterPointEnsemble tracking methods.

Overall, the CenterPoint detector performs better overall and over all categories of interest for tracking. We will use both methods to compare the PMBM performance against various other tracking methods, but we will favor the CenterPoint detections otherwise.

The detection information that is available is presented in figure 3.2:

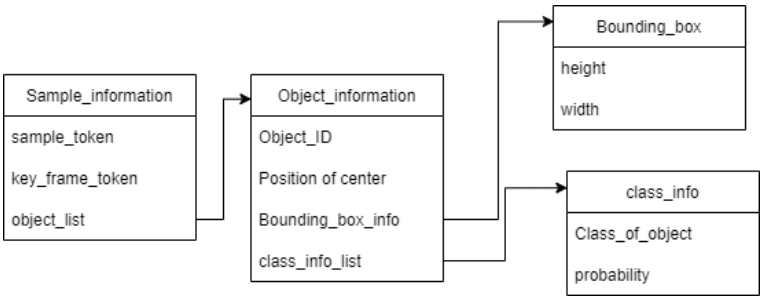


Figure 3.2: Detection data information

3.6 Other tracking methods

The performance of the PMBM is compared to four other tracking methods: A baseline for 3D multi-object tracking (AB3DMOT) [43], StanfordPRL-TRI [7], SimpleTrack [34] and CenterPointEnsemble [47]. AB3DMOT is chosen as a baseline to evaluate the initial performance of the PMBM tracking filter. This baseline is provided by the dataset providers [5]. The other methods are taken in order of best performing as state-of-the-art trackers whose code and results are made available as open source and can be directly compared to the PMBM tracker.

AB3DMOT uses a 3D Kalman filter and MEGVII detections. The data association is performed using Mahalanobis distance with the Hungarian algorithm. This data association algorithm is similar to Murty's algorithm, but retains only the global hypothesis with highest weight and discards all others. Murty's algorithm retains a set number of global hypothesis and then performs further reduction between steps. StanfordPRL-TRI uses a similar setup to AB3DMOT but improves on it by using a greedy nearest-neighbour approach to associate measurements with predicted states [7].

CenterPointEnsemble's tracking approach predicts the position of object centers and an estimate of the velocity from multi-frame data. The data association is performed using a greedy nearest-neighbour approach. SimpleTrack uses the same detections as CenterPointEnsemble. SimpleTrack uses a Kalman filter to predict object states and the Hungarian algorithm to solve the data association problem.

4 Results

This section of the report focuses on presenting results of experiments conducted and the performance of the PMBM tracking filter applied to the NuScenes dataset. All of the experiments are performed on the validation split of the dataset and evaluated on it. One sequence of the training set is used to tune the PMBM process parameters. Results regarding the study of filtering of clutter in detection is followed by evaluation of performance of the PMBM tracking filter with different detections and different tracking methods and finally a short study on the effect of motion and measurement noise on the AMOTP. For all results, we have chosen to multiply the AMOTA, MOTA, AMOTP and MOTP by 100 for ease of reading and comparison. MOTA and AMOTA take values between 0 and 100. MOTP and AMOTP have a minimum value of 0 and are unbounded by the top.

4.1 Experiments

The experiments that are performed in this work aim to gain empirical evidence of the performance of the PMBM tracking filter, the needs in terms of dataset and parameters. This work aims to compare the PMBM filter to baselines and state-of-the-art methods in the context of empirical environment. The experimental setup, limitations and experiments are presented in this section.

4.1.1 Experimental setup and limitations

The experimental machine uses an Intel® core^{TM} i7-10700 CPU @2.90GHz x 16 with 16GB of RAM and hard disk drive of sufficient size to contain all of the dataset (500Gb). The operating system is Ubuntu 20.04 64bits. As explained, the computations are CPU intensive and sequential in nature. The API has very strict requirements in terms of database organization and loads the entire split of the data needed into memory before transforming them into Python objects. This means that large JSON files cannot be loaded and manipulated due to memory limitations. The computations themselves vary in length between 100 and 1000 ms/frame or computations between 30 minutes and more than 1 hour. This work focuses on the validation set of data as this is the only one small enough to give access to the GT with the memory restrictions presented.

These computation limitations make the extensive use of loops over parameter for tuning prohibitive in practice but a systematic heuristic approach allows a thorough investigation of the research questions.

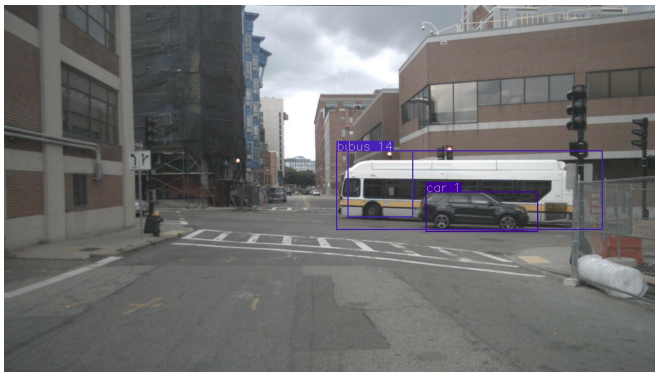
4.1.2 Conducted experiments

A systematic approach is established to answer the research question with study components regarding the performance of the PMBM depending on the detections used, the parameters of the PMBM and how they affect performance and finally the performance of the PMBM compared to other tracking methods.

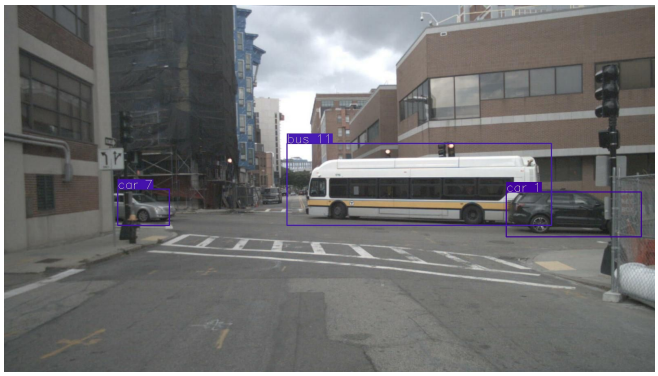
The detections provided contain clutter (false detections) that negatively affect the computation time and may negatively affect the performance of the PMBM. After establishing a set of starting parameters that are advised by domain experts as a starting point for our study, we evaluate the PMBM using different clutter filters on the detections of the best performing detector CenterPoint. The clutter filter threshold sets the limit above which a detector output detection probability must be in order to be considered as a valid object detection. We then set out to compare the performance of the PMBM against other trackers by taking care of using the same detections than those trackers while ensuring that we are using the best clutter filter available. Finally, we study the effect on the performance of the filter when using different sets of parameters for the Kalman filter noises. We use mostly the CenterPoint detections as they are the best performing that are made available. These are the best representation of an empirical detector available. We also ensure that we are only using LiDAR measurements so that when we are comparing between methods, sensor fusion will not enter into consideration and allow us a fair comparison between the PMBM filter and other methods.

4.2 Qualitative analysis

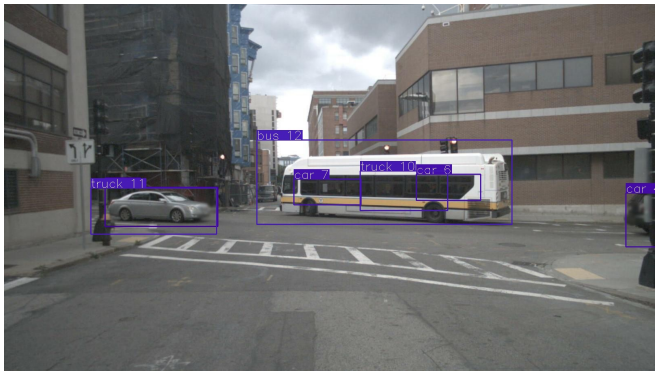
We illustrate the clutter visible as well as thresholded detection where only elements detected with more than 20% probability by the detection network are visible on screen. We include as well the output of the PMBM filter for this sequence of images. This analysis aims to show the successes and failures of the PMBM tracking algorithm. The output shown in the figures of this section are bounding boxes for objects detected or tracked superimposed on camera data. The bounding boxes are obtained as the average between the previous frame bounding box size and the current frame bounding box, such that the bounding box is always estimated to be larger than the true bounding box with respect to the estimate center of the object. The outputs shown here is an example sequence from the training set and was used for initial tuning of parameters:



(a) frame 9

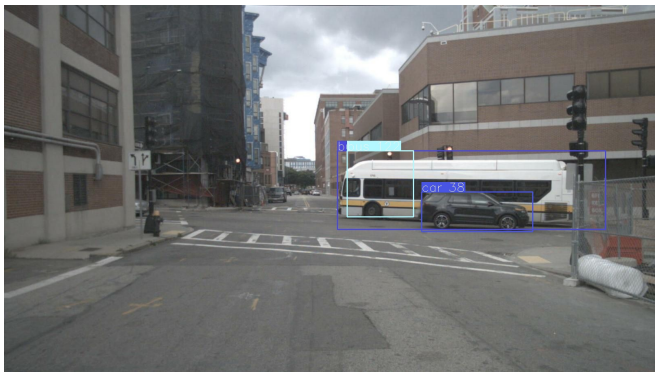


(b) frame 10



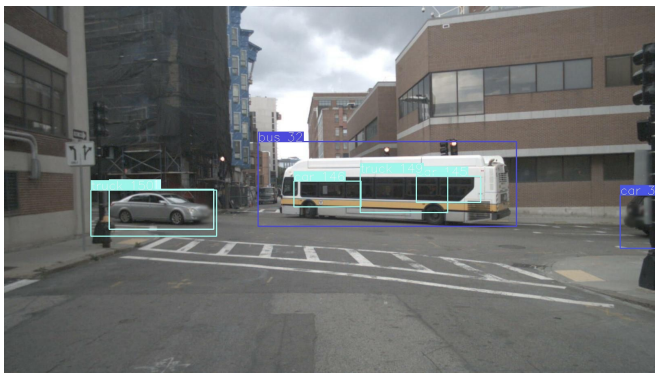
(c) frame 11

Figure 4.1: Three frames from detections with no filtering threshold



(a) frame 9

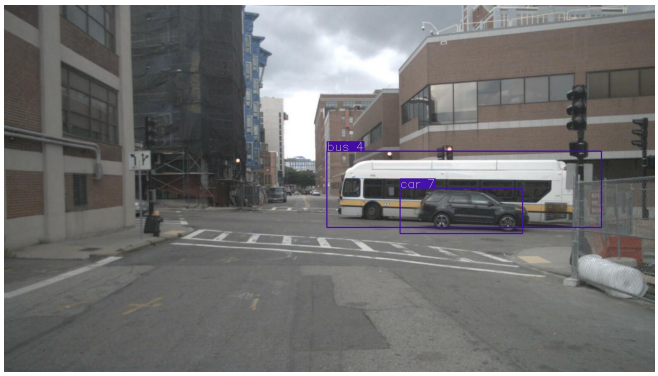
(b) frame 10



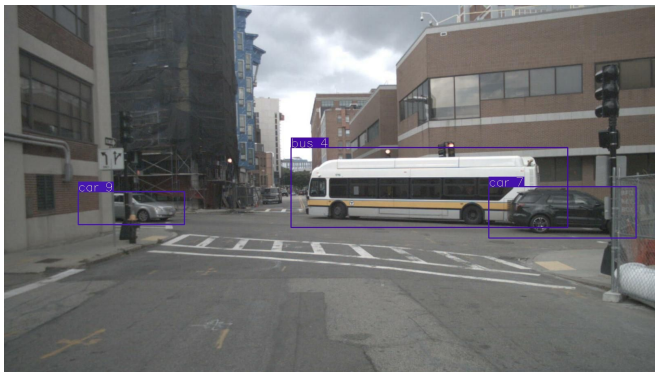
(c) frame 11

Figure 4.2: Three frames from detections filtering at 20% threshold

In figure 4.2, the light blue labeled measurements are clutter detections or incomplete objects on the detector output for detected objects filtered at 20% threshold.



(a) frame 9



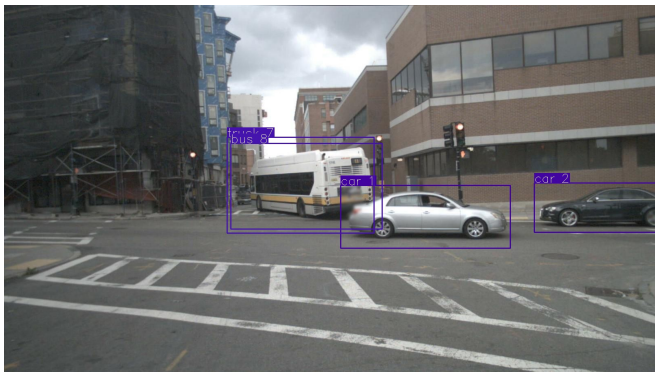
(b) frame 10

(c) frame 11

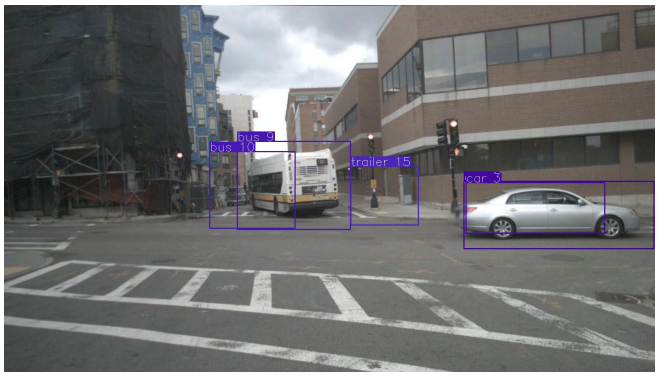
Figure 4.3: Three frames from the output of the PMBM filter

We notice the same amount of clutter on 4.1c and 4.2c and that the PMBM tracking filter handles the clutter properly for that same frame in 4.3c by correctly associating the proper detection with the track that on the next frame.

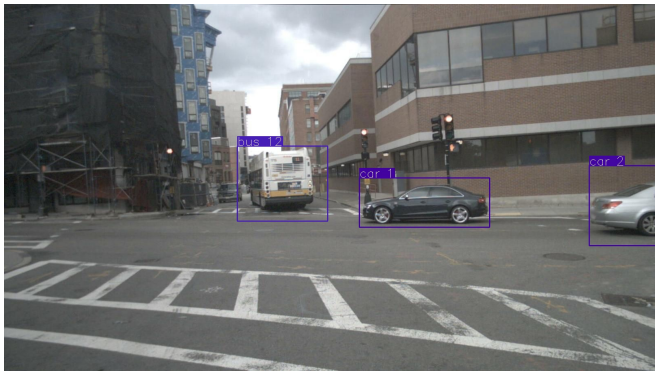
We now present the example results on frames at a further time step that illustrate lagging tracking boxes and identity switches and their respective raw detections in figures 4.4 and 4.5.



(a) frame 15

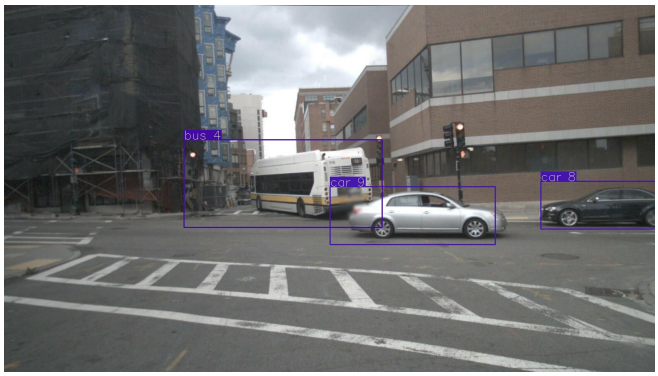


(b) frame 16

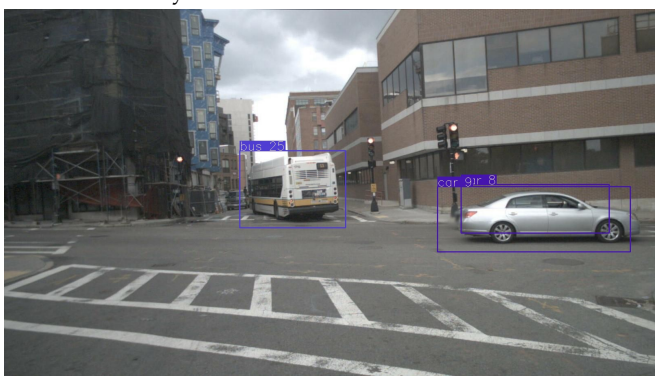


(c) frame 17

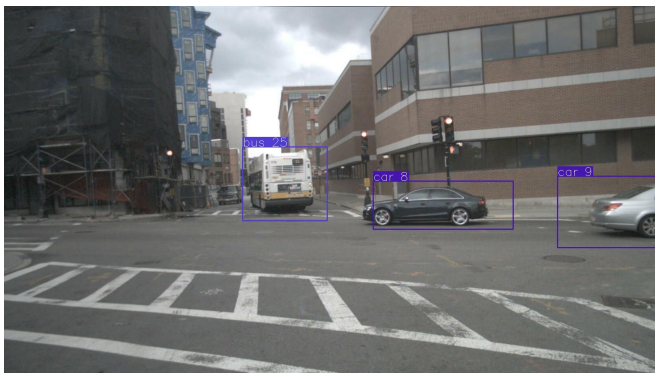
Figure 4.4: Three frames of raw detections



(a) frame 15: lagging tracking boxes, tracking box too large for bus with identity "bus 4"



(b) frame 16: Occluded object considered as detected, bus IDS to "bus 25"



(c) frame 17: lagging tracking box for "car 8"

Figure 4.5: Three frames from the output of the PMBM filter

We can observe in these figures that on frame 15, the tracking box for the bus is much larger than on the raw detection. This is caused by a sudden change in motion (rotation) that deviates from the predicted motion of the object. Futhermore, we observe lagging tracking boxes on the visible grey car. The lag observed for this car in frame 16 can also be attributed to the data association being performed between the clutter (smaller detection box in frame 16) and the predicted tracking box compared to the ground truth box. This will affect the MOTP and AMOTP measures. In the next frame (frame16), we can also observe an IDS from "bus 4" to "bus 25" as well as an occluded car labelled as detected due to a clutter detection allowing the data association. Finally on frame 17, we can observe that the tracking box for "car 8" is lagging. We also see that the occluded black car in frame 16 maintains its identity and is properly tracked.

It should be noted that the output of the PMBM tracking filter are point estimate (the mean value of the Gaussian state distribution) given by a particular global hypothesis. The uncertainty is handled and included in the Kalman process where the state covariance is used in the prediction step and the innovation covariance updates the uncertainty between the measurement and the predicted state estimate. The uncertainty is implicit in the algorithm. The uncertainty does not appear in the state estimation because the cost calculation of the data association algorithm only considers the mean value of the state vector as input to calculate the cost and choose the optimal global hypothesis.

4.3 Effect of clutter on tracking performance

The elements detected above the detection threshold and their effects on the AMOTA and AMOTP of the PMBM algorithm overall and for each class of interest are presented in tables 4.1 and 4.2. Filtering is performed on the CenterPoint detections such that all detections with a detection probability below the filtering threshold are removed from the detection set. The PMBM parameters are the ones found in appendix A.

Thresholding %	overall ↑	bicycle	bus	car	motorcycle	pedestrian	trailer	truck
0	63.9	34.6	**79.9**	**81.1**	63.9	71.7	**51.5**	**64.3**
10	63.9	34.6	**79.9**	**81.1**	63.9	71.8	**51.5**	**64.3**
20	**65.3**	**39.1**	78.2	**81.1**	**66.5**	**78.5**	50.8	62.8
30	63.1	39.4	75.9	77.4	63.5	78.6	48.6	58.6
40	59.4	39.3	73.6	70.8	58.3	75.4	45.1	53.0
50	53.9	35.4	69.1	63.8	51.1	71.5	41.8	44.5

Table 4.1: AMOTA for different clutter filtering levels. AMOTA takes values between 0 and 100 and higher values are better.

Thresholding %	overall ↓	bicycle	bus	car	motorcycle	pedestrian	trailer	truck
0	**61.1**	**57.5**	**61.7**	43.1	**56.8**	**45.8**	100.3	**62.5**
10	**61.1**	**57.5**	**61.7**	**43.0**	**56.8**	**45.8**	100.3	**62.5**
20	68.1	76.7	65.5	46.6	63.8	45.9	**100.1**	70.2
30	77.0	94.5	69.2	55.4	71.3	48.8	114.0	86.1
40	86.6	104.0	73.2	68.3	84.9	56.8	121.3	98.2
50	99.1	121.3	81.1	81.3	101.1	65.5	128.4	114.8

Table 4.2: AMOTP for different clutter filtering levels. AMOTP is unbounded by the top and takes a minimal value of 0 and lower values are better.

Overall, the best filtering threshold for detections is 20% from the perspective of the AMOTA and lower AMOTP for no filtering and 10% filtering threshold. For the remaining experiments, we choose a filtering threshold of 20% on all detections from this point forward. The better performance in terms of AMOTP of lower filtering threshold is attributed with the data association. If more detections are available, more of them could be removed through data association if they only exist within a limited amount of frames. Considerations of computational resources are also taken into account in the choice of threshold for clutter detections. Computations of the PMBM tracking filter were observed to be twice as long for thresholds 0% and 10% than for the 20% threshold. This is attributed to the reduction of the number of possible targets to loop through the algorithm.

4.4 PMBM against other methods and with different detectors

The overall performance of the PMBM tracking filter, AB3DMOT [43], StanfordPRL-TRI [7], SimpleTrack [34] and CenterPointEnsemble [47] are presented here as well as the PMBM performance using different detector models in table 4.3. We use a detection threshold of 20% for all PMBM evaluations. The performance of other tracking methods are available on their code repositories for the evaluation set. The arrows indicated if low (↓) or high (↑) are considered as good.

Tracking method - detector	AMOTA ↑	AMOTP ↓	MOTA ↑	MOTP ↓	MT ↑	ML ↓	IDS ↓	FP ↓	FN ↓
AB3DMOT-Megvii	15.1	150.1	15.4	40.2	1006	4428	9027	15088	75730
StanfordPRL-TRI	55.5	**79.8**	45.9	**35.3**	4294	2184	**950**	17533	33216
PMBM-Megvii	**57.4**	83.0	**50.5**	35.6	**3869**	**1499**	1334	**9752**	**25307**
AB3DMOT-PointPillars	2.9	170.3	4.5	82.4	480	5332	7548	41115	88551
PMBM-PointPillars	**27.6**	**142.2**	**24.4**	**72.1**	**2416**	**2482**	**1965**	**10162**	**43489**
CenterPointEnsemble	66.4	56.6	**56.1**	**32.1**	**4430**	1523	562	13187	20446
SimpleTrack	**68.6**	57.2	**59.2**	32.9	4385	1475	**519**	12983	**19941**
PMBM-CenterPoint	65.3	68.1	56.9	32.8	4202	**1451**	1461	**10981**	21686

Table 4.3: Performance measures for trackers using different detections.

The PMBM tracking filter outperforms all AB3DMOT implementations on all performance measures. The PMBM tracking filter also outperforms the StanfordPRL-TRI filter in terms of MOTA, AMOTA, ML, FP and FN. The PMBM filter underperforms when using the CenterPoint detections overall compared to state-of-the-art tracking filters, but remains competitive in terms of AMOTA and MOTA. On Megvii and CenterPoint detections, the PMBM tracking filter has a considerable difference in performance compared to other state-of-the-art solutions in terms of AMOTP where the PMBM tracking filter performs less well than other methods. This is attributed to the data association process that allows for some association to happen between some clutter and predictions if the detections are closer to the predicted state and that the global hypothesis weight is higher than other global hypothesis that would perform the association between the correct measurement and the predicted state. The MOTA and MOTP is better for the PMBM when compared to the CenterPointEnsemble method. We also observe that for better detector performance, we observe a better AMOTA and AMOTP for the PMBM filter.

We present in table 4.4 the overall AMOTA and AMOTA per class of interest for the different methods presented.

Tracking method - detector	AMOTA overall	Bicycle	Bus	Car	Motorcycle	Pedestrian	Trailer	Truck
AB3DMOT-Megvii	15.1	0	40.8	27.8	8.1	14.1	13.6	1.3
StanfordPRL-TRI	55.5	25.5	**64.1**	71.9	48.1	74.5	**49.5**	51.3
PMBM-Megvii	**57.4**	22.8	**73.9**	**76.6**	**58.1**	**75.3**	36.7	**58.2**
AB3DMOT-PointPillars	2.9	0	6.6	9.4	0	3.9	0	0
PMBM-PointPillars	**27.6**	0	**37.4**	**62.6**	**13.3**	**52.3**	**8.7**	**18.6**
CenterPointEnsemble	66.4	45.7	80.0	84.2	61.5	77.7	50.3	**65.6**
SimpleTrack	**68.6**	**51.0**	**80.4**	**83.8**	**68.3**	**79.2**	**53.0**	64.9
PMBM-CenterPoint	65.3	39.1	78.2	81.1	66.5	78.5	50.8	62.8

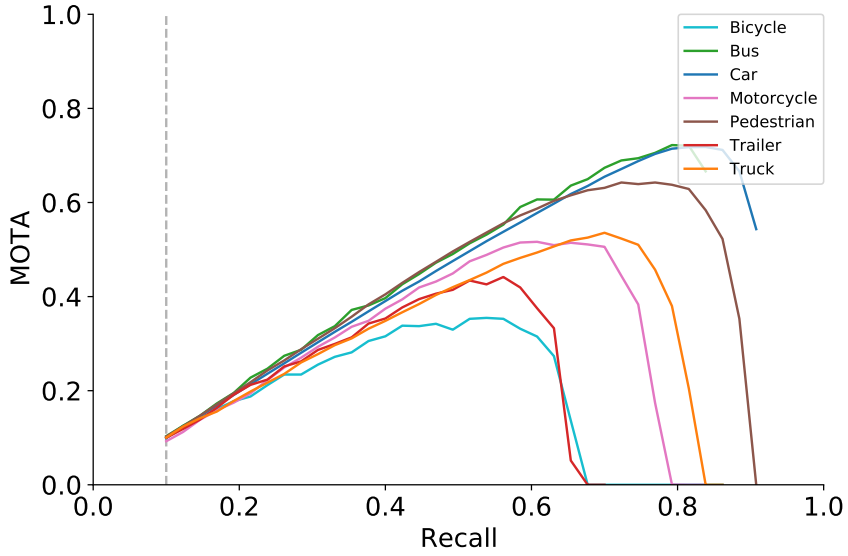
Table 4.4: Overall AMOTA and for each class for different tracking methods

The PMBM tracking filter performs better AMOTA in all classes compared to the baseline method. The PMBM filter also outperforms the StanfordPRL-TRI across all classes except bicycle and trailers. The PMBM remains competitive against other state-of-the-art solutions. This table shows that the data association is performing according to desired behavior.

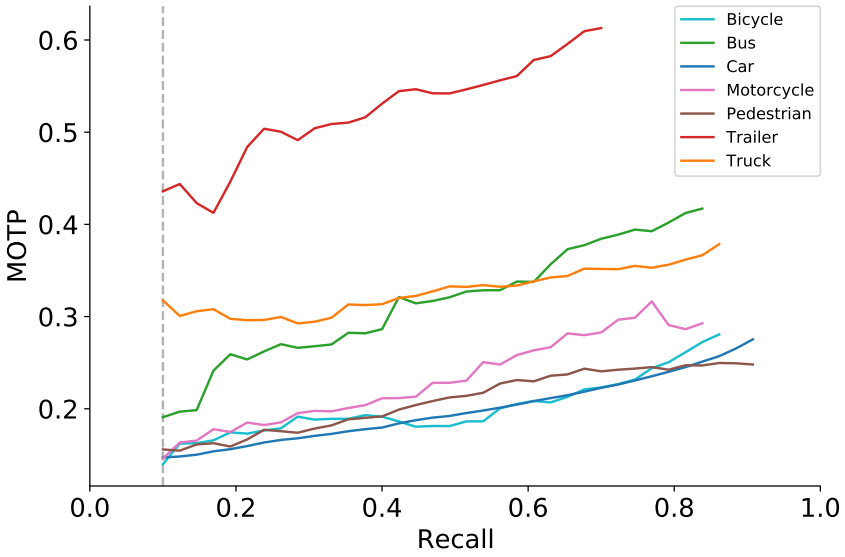
We present the recall curves against MOTA and MOTP for CenterPointEnsemble, SimpleTrack and PMBM-CenterPoint. We otherwise refer the reader to the appendix B for additional figures related to the CenterPoint detections, appendix C for figures related to Megvii detections and appendix D for figures related to PointPillars detections.

We expect the MOTA vs. recall curve to increase and then drop as recall value increases as the amount of positive results is included until the presence of other classes' errors get included in and drop the MOTA performance. The later the drop, the better classifier of that class we have. The higher the value of MOTA, the better tracking, from a track perspective for that class we observe.

For MOTP, the expected behavior is that we would observe an increasing value of MOTP as the value of recall increases because we are including more and more dislocation errors into the MOTP numerator. Sudden changes (increases) in MOTP value would indicate that we are now changing how many classes are included . A sudden decrease would imply that the denominator becomes larger faster than errors are included.

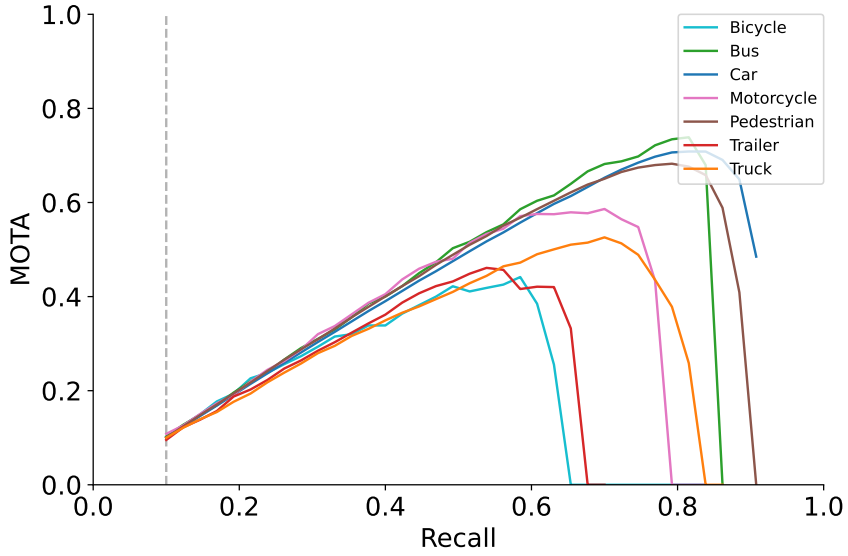


(a) MOTA at different recall threshold

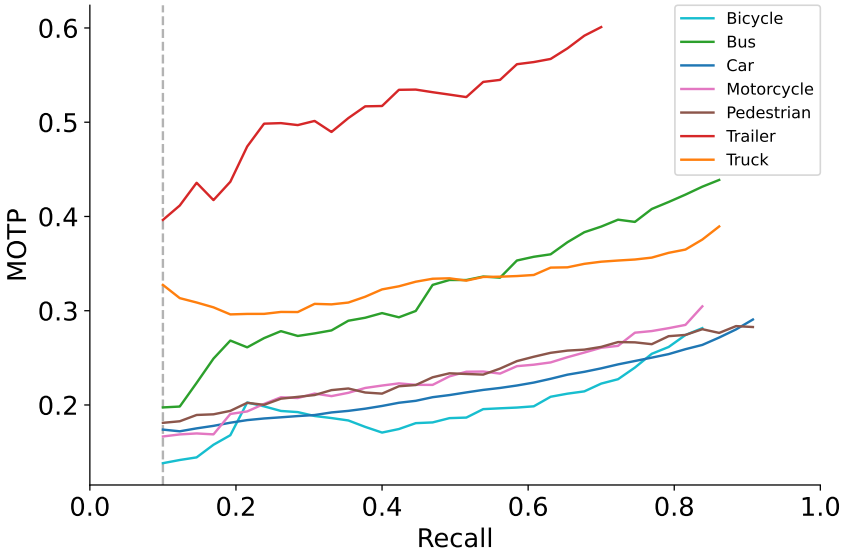


(b) MOTP at different recall threshold

Figure 4.6: Recall curves for CenterPointEnsemble

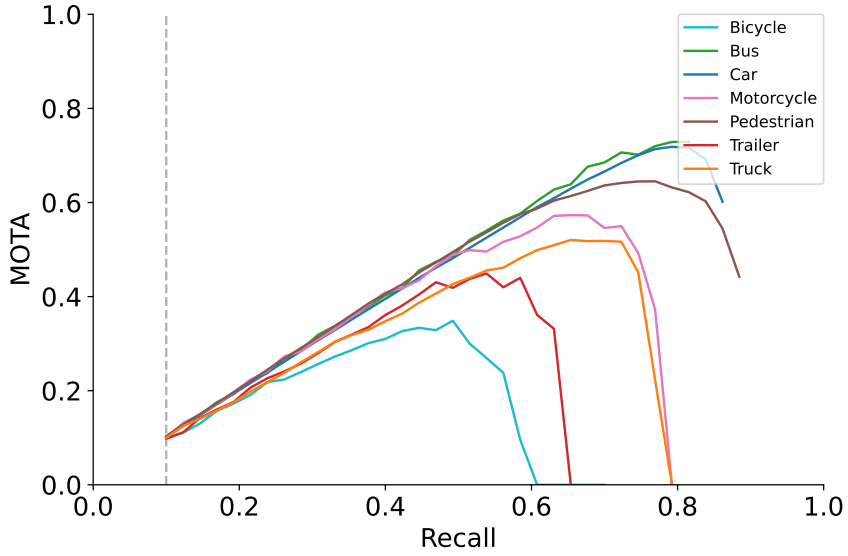


(a) MOTA at different recall threshold

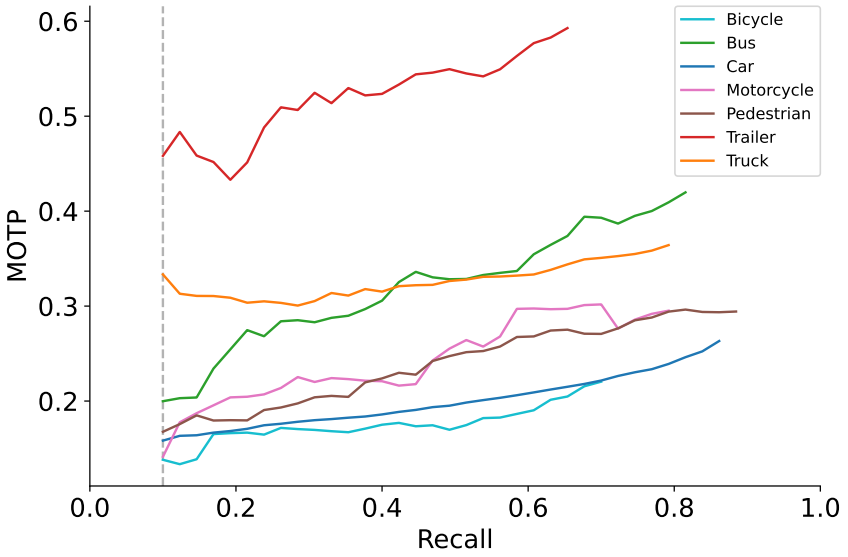


(b) MOTP at different recall threshold

Figure 4.7: Recall curves for SimpleTrack



(a) MOTA at different recall threshold



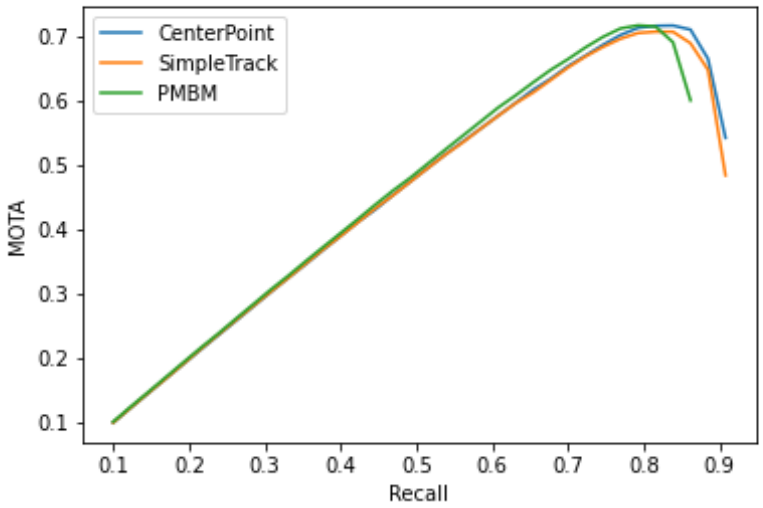
(b) MOTP at different recall threshold

Figure 4.8: Recall curves for PMBM-CenterPoint

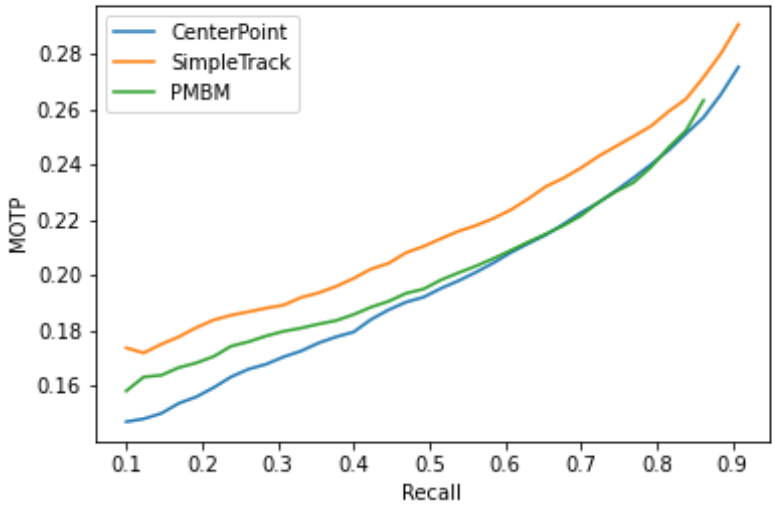
We observe a better MOTA vs. recall curve of the PMBM for pedestrians when comparing with the CenterPointEnsemble MOTA curve. We notice that the best tracked classes according to the MOTA vs. recall curves are cars, buses and pedestrians. This could indicate that the chosen motion models are appropriate for those classes.

The PMBM tracking filter underperforms compared to other state-of-the-art solutions across all classes with respect to the MOTP. The behavior with respect to different classes is the same across methods. This could indicate that the behavior observed can mostly be attributed to the dataset. We further observe that for all methods the MOTP is low (good performance) for classes motorcycle, pedestrian, car and bicycle compared to the other classes (trailer, bus and truck). This indicates that the largest contribution to AMOTP is contained in those three classes. These are rare classes that are over-represented in the dataset as opposed to reality.

For ease of interpretation and comaprison, MOTA and MOTP plots are presented for CenterPointEnsemble, SimpleTrack and PMBM-Centerpoint for the car class on the same figure 4.9:



(a) MOTA at different recall threshold



(b) MOTP at different recall threshold

Figure 4.9: Recall curves comparing CenterPointEnsemble, SimpleTrack and PMBM-CenterPoint for the car class

From these curves, it can be observed that the PMBM underperforms in terms of MOTA at high recall values against the other state-of-the-art methods but is better than SimpleTrack in terms of MOTP. This indicates that the PMBM is competitive with state-of-art alternatives on the NuScenes dataset with little parameter tuning.

4.5 Motion and measurement noise tuning

The values of the the measurement and motion uncertainty are allowed to be reduced and grow in order to study their effects on the MOTP. The values of motion noise (P) and measurement noise (R) are multiplied by a factor indicated on the heat map axes indexes with respect to the parameters found to be optimal in 4.2. We present the results in figures 4.10 and 4.11.

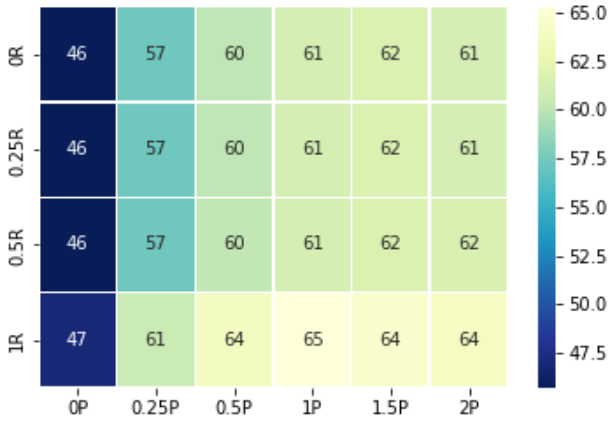


Figure 4.10: AMOTA ↑ of PMBM-CenterPoint for different values of motion noise (P) and measurement noise (R)

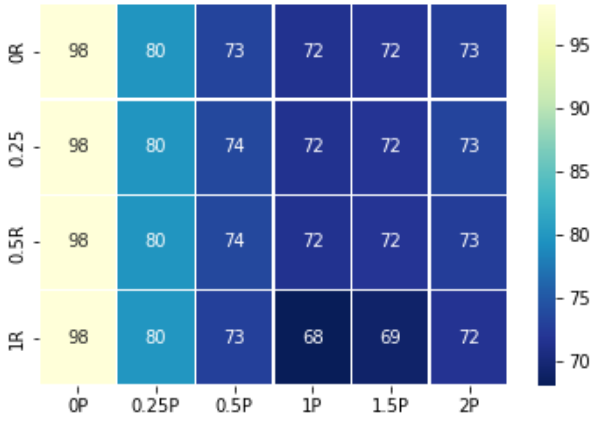


Figure 4.11: AMOTP ↓ of PMBM-CenterPoint for different values of motion noise (P) and measurement noise (R)

For both AMOTA and AMOTP, the change in performance is observed when the motion noise changes with only very small effects when the measurement noise is reduced. The measurement noise was not increased because of matrix singularity arising in some of the sequences. This result seems to indicate that the performance is affected more by the choice of motion model than by our tuned measurement model for this tracking problem. Furthermore, the next conclusion that can be taken from these heat maps is that the increase in motion noise does not improve the performance measures. This indicates that the lagging of tracking boxes (reflected in AMOTP) should be affected by another parameter.

5 Discussion

5.1 Qualitative analysis

The PMBM globally performs as expected from the perspective of the theoretical background. Only one measurement can be associated to a single distribution and objects are tracked properly according to the motion model and measurement model used in this implementation. We identify several shortfalls of the implementation of the PMBM that we are using. The filtering of clutter does not always remove some or all of the clutter observed. This attests to the importance of the good performance of the detector. We also observe that the PMBM suffers from IDS when objects tracked change their motion greatly from expected behavior. These IDSs occur during fast turns and strong deceleration/accelerations between frames. The reason for these IDS is explored in 5.5 and can be summarized by a data association issue with the most likely hypothesis being chosen is the birth of a new object. Finally, the lagging of tracking boxes with respect to the actual position of the tracked object is attributed to a lack of parameter tuning or the presence of clutter. Overlapping clutter detections may influence the update process. The cost matrix may consider one detection rather than another in the optimal global assignment of targets in the data association process, but this may not be the optimal assignment for this particular object.

5.2 Study of clutter on CenterPoint detections

The study of the filtering threshold for clutter detections showed an expected improvement in computation time with an increase in filtering threshold. The higher the filter limit, the less computations are expected and a faster runtime is expected. We also observe that the AMOTA is best at a filtering threshold of 20% but an AMOTP that is overall better at lower filtering threshold. The increase in AMOTA is expected as there are less chances of clutter influencing the data association for that threshold and thus the proper tracks are identified. For the AMOTP, lower thresholds mean that conversely, more clutter measurements that might be closer to the ground truth position are considered for data association. Overall the behavior is expected and we justify the use of a threshold of 20% by the higher AMOTA that confirms that the proper objects are tracked. The AMOTP behavior can also be attributed to other parameters that are presented in 5.5. Although this study was performed on only the best detector available, other detection method might behave differently. For practical reasons,

we choose the threshold of 20% as a rule of thumb in line with domain expert methodology and presented in [34]. In this paper, the author propose an optimal clutter filtering threshold on CenterPoint at 24%.

5.3 PMBM with 3 different detections and the same clutter filtering thresholds

The expected behavior of the PMBM with different detections is that better performing detector should yield a better performance of the PMBM. This is the behavior observed between the performances of all PMBM implementation using different detector outputs across all presented performance measures. This behavior highlights the influence of a good detector on all performance metrics of any tracking filter. We highlight here the importance of a good detector related to the discussion on AMOTP in the previous section 5.2 and in 5.5 to come. A good detector should output less clutter, more precise detections (in the MOTP sense) and reduce the measurement noise. This directly impacts the performance in terms of both MOTA and MOTP. We emphasize here the increase of 8 points when using the CenterPoint detections compared to the Megvii detections and the increase of 38 points between PointPillars and CenterPoint detections for the PMBM. In this work, we are limited in our use of detector by the fact that we use prepared detector outputs that are open-source and readily available to compare with other tracking methods.

5.4 PMBM performance against other tracking methods

In the wider context of comparing the performance measure of the PMBM with respect to other tracking methods, we have made the choice of comparing with tracking methods that made both code and results available for direct and fair comparison. We observe that the PMBM tracking filter outperforms the StanfordPRL-TRI method using Megvii in terms of (A)MOTA, MT and ML performance measures. This implies that the PMBM tracking filter tracks properly the objects of interest in sequences compared to the StanfordPRL-TRI tracker. However, the tracking is imprecise (in the MOTP sense). The (A)MOTP is comparable between methods but the PMBM underperforms across this performance measure. Similarly, this is observed also when comparing with other state-of-the-art methods (CenterPointEnsemble and SimpleTrack) using CenterPoint detections where the (A)MOTP is systematically worse in the case of the PMBM compared to the other two methods. This discrepancy is attributed to a lack of parameter tuning as explained in 5.5 further. We note however that the PMBM slightly outperforms other state-of-the-art methods for the ML performance measure although it displays double IDS and similar MT performance measures. This could be an indication that the explicit modeling of undetected objects in the context of the PMBM is robust to occlusion and appearing objects although the motion model and other parameters are not well optimized. The IDS performance also seem to indicate that the motion model chosen or the data association approach can be further refined. There are strong assumptions attributed to the constant velocity motion model. The assumption is that the velocity of the parameter does not change during the time step increment. This is not always realistic in all traffic situations. This assumption is partly mitigated by applying different motion noise to different tracked object classes. Another possible mitigation method would be to use different motion models for different classes of objects.

We note here that the reported times for estimation are much shorter for greedy methods compared to methods using the Hungarian or Murty's algorithm although the experimental resources might be different. Methods using a greedy approach report 1 ms/frame speeds whereas other data association approach report tens or hundreds of milliseconds per frame as estimation speeds. This poses the question of the need for complex probabilistic model for object representation as opposed to using a good detector, motion model and a greedy

data association process. We argue that the present PMBM tracking filter lacks tuning in its parameters and that the empirical performance is limited by the low frequency of information used. This argument is elaborated on in the discussion of 5.5. Furthermore, explicit handling of occluded objects is not proposed by other methods.

5.5 Motion and measurement noise tuning for MOTP/AMOTP performance increase

In order to expand on the issue of the precision (MOTP and AMOTP), it was important to understand the process of a Bayesian filter. The key components of which are the motion and measurement process noises. A high motion noise allows for more measurements to be included into consideration due to extended motions being possible. With low measurement noise, the expected behavior is that of points further from the center of predicted distributions to not be considered for association (more variation in the speed update). The results obtained are not in line with expected behavior. We observe high variations depending on the motion noise, but not when changing the measurement noise. This seems to indicate that other parameters could be involved in the AMOTP performance measures. Due to the size of the dataset and other computational restrictions it is difficult to evaluate the performance of the PMBM tracking filter with respect to the large parameter space. We have restricted ourselves to studying only the motion noise and measurement noise effects on the AMOTP. A more thorough analysis of the parameters affecting the performance of the PMBM filter with respect to the lagging of tracking boxes requires a search over the parameters presented in A as well as motion noise, measurement noise, filtering threshold, detection method and motion model. Due to limited computational resources and time, this could not be accomplished in this work.

The survival probability, detection probability and existence pruning threshold affect directly how the spatial distributions of objects that are considered exist within a frame and during a sequence. This affects the AMOTP by restricting the data associations possible. The birth intensity, weight and birth gate size affect directly where new objects that enter into frame (or occluded previously) appear in and their dislocation with respect to the ground truth measurements. The number of global hypothesis kept as well as the weight pruning threshold affect the the data associations considered directly and thus influence the AMOTP. The filtering threshold on the detections affects the AMOTP as explained in 5.2 and finally the detection method affects the AMOTP as presented in 5.3. The change of motion model can affect the MOTP/AMOTP performance measure in the prediction step. Different objects will have a "typical" motion that is different depending on the class of the object as well as environmental context. A car on a highway and in a city or parking garage do not exhibit the same behavior. A pedestrian in a suburban area or at a traffic intersection with bicycle and cars are not expected to have the same motion. Changes in motion of objects depending on their class an environment are expected to affect the choice of motion model and thus the predicted state density of these objects. This directly impacts the performance of the tracking filter with respect to the expected position of the tracked object and the data association between the next measurement and the state density. The choice of the motion model also affects the data association. An incorrect motion model (such as the bus turning example) may lead to the most likely data association being that of a new object being born on scene due to the wrong motion being predicted between frames. One could use a non-linear motion model to model motion. This would require to change the Kalman filter to incorporate non-linearities by using either an unscented Kalman filter [42] or the extended Kalman filter [19].

It is noted in [34] that the NuScenes dataset suffers from a lack of annotation frames at higher frequency. This is stated to increase the difficulty of tracking the proper objects when sudden changes in motion and speed are operated between those key frames. This is an

essential problem both to the Kalman filter and to the data association. The Kalman filter predictive performance is affected by the frequency at which we can obtain measurement information. By increasing the frequency, the changes in state from one time step to the next are smaller and easier to predict (in particular motion, rotation and speed). For the data association, the higher frequency of measurement would increase the association performance due to clutter being present only at certain frames and not others, thus removing certain hypothesis from consideration faster. In this same study [34], the detections are augmented to 10Hz and evaluated on the evaluation set. The results are presented in tables 5.1 and 5.2 with the 2Hz results as well as the PMBM implementation on CenterPoint detections.

Tracking method-detector	AMOTA ↑	AMOTP ↓	MOTA ↑	MOTP ↓	MT ↑	ML ↓	IDS ↓	FP ↓	FN ↓
SimpleTrack 10Hz	**69.5**	**54.6**	**60.2**	**32.5**	**4570**	**1366**	**403**	14505	**18120**
SimpleTrack 2Hz	68.6	57.2	59.2	32.9	4385	1475	519	12983	19941
PMBM-CenterPoint 2Hz	65.3	68.1	56.9	32.8	4202	1451	1461	**10981**	21686

Table 5.1: Comparison of performance measures of trackers against 10Hz detection data augmentation.

Tracking method-detector	AMOTA overall	Bicycle	Bus	Car	Motorcycle	Pedestrian	Trailer	Truck
SimpleTrack 10Hz	**69.5**	**50.3**	**79.5**	**83.8**	**74.1**	**80.6**	**52.7**	**65.6**
SimpleTrack 2Hz	68.6	51.0	80.4	83.8	68.3	79.2	53.0	64.9
PMBM-CenterPoint 2Hz	65.3	39.1	78.2	81.1	66.5	78.5	50.8	62.8

Table 5.2: Comparison table of overall AMOTA for each class of interest between 10Hz detection data augmentation and other methods

We can observe an increase in performance (in the 10Hz case) compared to the case where the 2Hz data is used across all metrics and in particular with respect to the AMOTP. The authors argue that the performance of most trackers can be increased through having higher detection frequency.

5.6 Method limitations

The PMBM tracking filter implicitly handles the three uncertainties of the MOT task. The number of state uncertainties are modeled using the RFS cardinality distribution as a probabilistic representation of both the birth, evolution and death process of objects in situation. The uncertainty of the state is encoded in the Kalman filter process where the Gaussian state mixture components have updated covariances during the prediction step and once more during the update step. The uncertainty in identity (or track identification) is encoded in Murphy's algorithm where the loss (distance) is minimized over all detected objects but not necessarily for each and every particular object. The uncertainties themselves are not used at estimation time. Following the maximum likelihood estimate decision criterion for point estimate, the output of the algorithm is the mean value of each Gaussian state. The covariance matrices are saved for each state and used accordingly at the next iteration of the algorithm to perform the prediction step and updated after data association.

Due to the use of a large dataset with various sequences presenting different condition and the use of detector outputs as empirical input, we can assume that this study reflects an empirical investigation of the performance of the PMBM. We will however note that the average performance measures AMOTA and AMOTP typically used in evaluation of tracker performance are heavily dependent on the detectors used as well as the dataset itself. NuScenes was chosen as it presents varied and numerous sequences in different driving situations in an open source format. The performance measures of the PMBM presented in this work are heavily dependent on the previous reasons and will only be indicative of the performance

on this data set. Any generalization of the performance of the PMBM should be investigated in a wider context and with different datasets. Furthermore, the (A)MOTA and (A)MOTP represent the overall performance over the dataset split studied and the assumption is that the errors of the tracker are distributed uniformly across the various sequences used. The MOTA and MOTP are chosen at the recall threshold where they are maximized. This gives an account of the tracker performance, but not its operation in conjunction with the detector. The difference between AMOTA and MOTA or AMOTP and MOTP is seen through the recall curve performance and provides better information about the data and detector. These performance measures should reflect the average population behavior given the sufficient number of sequences and the diversity of traffic situations.

The assumption of uniformly distributed error presents a step forward in the empirical investigation of the PMBM compared to previous empirical study [3]. In this study, detector output was simulated by adding independently and identically distributed (i.i.d) white noise to the ground truth detections and uniformly distributed clutter on 20 sequences of the KITTI dataset. In effect, i.i.d noise was added to the measurement set cardinality and to the ground truth spatial position of objects . We argue that this is not representative of the true cardinality distribution of clutter and too similar to ground truth in the case of measurement noise. This could be interpreted as a bias towards the ground truth when evaluating the PMBM. The authors acknowledge the issue due to a lack of generalization of the implemented detector for that work. These shortcomings are addressed in this study as we use empirical detections that encompass both detected state noise and clutter cardinality distribution in a way that is more faithful to an empirical situation. Furthermore, the large size of the dataset allows for generalization of the detection models.

We argue further that the other performance measures are computed values across the whole evaluation set of data. This implies that the interpretation that we make in this work is based on our understanding of the theoretical framework within which we operate but does not reflect the sequence or frame by frame realities. We reduced the risk of wrong interpretation by choosing to display a sequence that encompasses several shortcomings of the PMBM in a "sufficiently complex" scenario.

We also emphasize that the PMBM tracking algorithm is the only method (to the best of our knowledge) that has an explicit probabilistic representation of occluded objects through the undetected object distribution and proposes a measurement driven handling of the occluded objects and objects born on the scene for the first time. Deep learning tracking method rely mostly on similarity measures or greedy data association methods to solve the occlusion problem. Other Kalman filter methods presented here (AB3DMOT, StanfordPRL-TRI and SimpleTrack) do not have an explicit representation of occluded objects.

5.7 Ethical considerations

This research is commissioned by Arriver in order to study a new and promising MOT filter. Although the aim is focused on tracking, this work has impacts on societal aspects. This tracking filter can be used to increase the safety of road users. If this filter was coupled with a decision policy system, the hope is to reduce road accidents. The application in automotive autonomous systems of such a filter poses the question of confidence and minimum safety thresholds that have to be met. This is inherently an issue that must be taken into account according to ethical and moral considerations that are biased by societal values. The use of distributions to estimate the state of objects of interest could be a good approach to decision making as confidence bounds can be estimated by using the covariance matrix for the different values of interest and inform the model user about the amount of information present to make a decision. However, we must remind ourselves that threshold values must be chosen and unexpected behavior once a product is brought into production could lead to fatal outcomes for road users.

The MOT filter could also be used for surveillance in order to study group behaviors or for security and policing purposes. There are also several military applications in which such a MOT filter could find a use. These are of course dependent on social needs and biases as well. In any of these applications, the model builder's bias as well as the application bias should be taken into account. This means that the PMBM tracking filter could be used for means detrimental to societal betterment and as a tool for control.

6 Conclusion

Multi-Object Tracking (MOT) is the task wherein objects of interest are identified and tracked in measurement sequences of interest. MOT methods try to answer uncertainties regarding the number of objects, the state of the objects of interest and the identity (track) to which they belong. The PMBM tracking filter is a bayesian filter that is proposed to solve the MOT task. Using conjugate distributions over random finite sets for detected and undetected objects, the PMBM tracking filter algorithm predicts the state of objects of interest at future time steps and updates the prediction with the output of an object detector. This thesis answers the following research questions:

- *What is the performance of the PMBM on the NuScenes dataset?* The performance of the PMBM tracking filter was investigated and evaluated depending on detector used and against different other tracking methods. **AMOTA** and **AMOTP** are performance measures commonly used to evaluate the performance of MOT methods on the NuScenes dataset. They respectively represent the performance of tracking by assigning the right object to the right track and the precision of tracking in the sense of how close to the ground truth positions the estimated states are. The best **AMOTA** and **AMOTP** achieved on the validation set using the PMBM tracking filter is **65.3** and **68.8** respectively.
- *How does the PMBM performance compare to a baseline tracking method and against other state-of-the-art tracking algorithms?* We have shown that the PMBM tracking filter can be evaluated on the NuScenes dataset, outperforms the baseline AB3DMOT on all performance measures and performs competitively to other state-of-the-art tracking methods.
- *How is the performance of the PMBM affected by the quality of the object detector output?* Increased performance of the PMBM tracking filter is shown to positively be related to better detectors. The PMBM performance is negatively affected by clutter, but shows good performance in (A)MOTA performance measures.

There are several restricting conditions to the performance of the PMBM filter. These restrictions are partly inherent to the dataset (detection clutter, annotation frequency), to the method (intractability of Murty's algorithm leading to long computational times) and to our tuning of the parameter in the form of a large parameter space that makes optimization difficult within the time constraints of this project. In the previous chapter, aspects of tuning

are discussed. The issue of dislocated center of object with respect to ground truth (AMOTP) remains; although there are indications that the observed large AMOTP is caused by parameters that are not the motion or measurement noise. We also showed that the Mostly Lost performance measure is better than the state of the art solutions. This is a strong indicator that the explicit modeling of undetected objects, that is unique to the PMBM, allows for tracking of occluded objects.

More investigations are required in order to understand the effect of the large parameter space on the AMOTP. Applying different motion models is also a viable way of decreasing the AMOTP as we have shown that the constant velocity motion model used is not robust enough to handle sharp turns and large changes in velocity. We could also apply different motion models depending on location (GPS information) and class of object. The dataset could also be augmented by using the information contained between key frames to refine the prediction of the Kalman filter. This could allow for sudden changes in velocity and rotation to be accounted for. As shown, better detections lead to better tracking performance. Better detectors have less clutter detected and more precise detections (in the MOTP sense) that can be used. In terms of the speed of execution of the algorithm, the bottleneck is the data association process. More studies are needed to investigate the effect of the parameters of the reduction and their effect on computational speed and MOTA/MOTP.

Bibliography

[1] Keni Bernardin and Rainer Stiefelhagen. "Evaluating Multiple Object Tracking Performance: The CLEAR MOT Metrics". In: *J. Image Video Process.* 2008 (Jan. 2008). ISSN: 1687-5176. DOI: 10.1155/2008/246309. URL: https://doi.org/10.1155/2008/246309.

[2] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[3] Erik Bohnsack and Adam Lilja. "Multi-Object Tracking using either Endto-End Deep Learning or PMBM filtering". MA thesis. Chalmers University of Technology, 2019, pp. 1–110.

[4] Per Boström-Rost, Daniel Axehill, and Gustaf Hendeby. "PMBM filter with partially grid-based birth model with applications in sensor management". In: (Mar. 2021). URL: http://arxiv.org/abs/2103.10775.

[5] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. "nuScenes: A multimodal dataset for autonomous driving". In: *CVPR*. 2020.

[6] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. "Multi-View 3D Object Detection Network for Autonomous Driving". In: *CoRR* abs/1611.07759 (2016). arXiv: 1611.07759. URL: http://arxiv.org/abs/1611.07759.

[7] Hsu-Kuang Chiu, Antonio Prioletti, Jie Li, and Jeannette Bohg. "Probabilistic 3D Multi-Object Tracking for Autonomous Driving". In: *CoRR* abs/2001.05673 (2020). arXiv: 2001.05673. URL: https://arxiv.org/abs/2001.05673.

[8] Wongun Choi. "Near-Online Multi-target Tracking with Aggregated Local Flow Descriptor". In: *CoRR* abs/1504.02340 (2015). arXiv: 1504.02340. URL: http://arxiv.org/abs/1504.02340.

[9] Julie Dequaire, Peter Ondrúška, Dushyant Rao, Dominic Wang, and Ingmar Posner. "Deep tracking in the wild: End-to-end tracking using recurrent neural networks". In: *The International Journal of Robotics Research* 37.4-5 (2018), pp. 492–512. DOI: 10.1177/0278364917710543. eprint: https://doi.org/10.1177/0278364917710543. URL: https://doi.org/10.1177/0278364917710543.

[10] Patrick Emami, Panos M. Pardalos, Lily Elefteriadou, and Sanjay Ranka. "Machine Learning Methods for Solving Assignment Problems in Multi-Target Tracking". In: *CoRR* abs/1802.06897 (2018). arXiv: 1802.06897. URL: http://arxiv.org/abs/1802.06897.

[11] Angel F. Garcia-Fernandez and Lennart Svensson. "Tracking Multiple Spawning Targets Using Poisson Multi-Bernoulli Mixtures on Sets of Tree Trajectories". In: *IEEE Transactions on Signal Processing* 70 (2022), pp. 1987–1999. DOI: 10.1109/tsp.2022.3165947. URL: https://doi.org/10.1109%2Ftsp.2022.3165947.

[12] Angel F. Garcia-Fernandez, Jason L. Williams, Karl Granstrom, and Lennart Svensson. "Poisson Multi-Bernoulli Mixture Filter: Direct Derivation and Implementation". In: *IEEE Transactions on Aerospace and Electronic Systems* 54.4 (Aug. 2018), pp. 1883–1901. DOI: 10.1109/taes.2018.2805153. URL: https://doi.org/10.1109%2Ftaes.2018.2805153.

[13] Ángel F. García-Fernández, Lennart Svensson, Jason L. Williams, Yuxuan Xia, and Karl Granström. "Trajectory Poisson multi-Bernoulli filters". In: (Mar. 2020). DOI: 10.1109/TSP.2020.3017046. URL: http://arxiv.org/abs/2003.12767%20http://dx.doi.org/10.1109/TSP.2020.3017046.

[14] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. "Vision meets Robotics: The KITTI Dataset". In: *International Journal of Robotics Research (IJRR)* (2013).

[15] Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012.

[16] Karl Granstrom, Maryam Fatemi, and Lennart Svensson. "Poisson multi-Bernoulli conjugate prior for multiple extended object filtering". In: (May 2016). DOI: 10.1109/TAES.2019.2920220. URL: http://arxiv.org/abs/1605.06311%20http://dx.doi.org/10.1109/TAES.2019.2920220.

[17] Karl Granstrom and Lennart Svensson. *Mutli-Object tracking for automotive systems*. 2019. URL: https://www.edx.org/course/multi-object-tracking-for-automotive-systems.

[18] Karl Granström and Marcus Baum. "Extended Object Tracking: Introduction, Overview and Applications". In: *CoRR* abs/1604.00970 (2016). arXiv: 1604.00970. URL: http://arxiv.org/abs/1604.00970.

[19] Simon J. Julier and Jeffrey K. Uhlmann. "New extension of the Kalman filter to nonlinear systems". In: *Signal Processing, Sensor Fusion, and Target Recognition VI*. Ed. by Ivan Kadar. Vol. 3068. International Society for Optics and Photonics. SPIE, 1997, pp. 182–193. DOI: 10.1117/12.280797. URL: https://doi.org/10.1117/12.280797.

[20] Chanho Kim, Fuxin Li, Arridhana Ciptadi, and James M. Rehg. "Multiple Hypothesis Tracking Revisited". In: *2015 IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 4696–4704. DOI: 10.1109/ICCV.2015.533.

[21] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven Lake Waslander. "Joint 3D Proposal Generation and Object Detection from View Aggregation". In: *CoRR* abs/1712.02294 (2017). arXiv: 1712.02294. URL: http://arxiv.org/abs/1712.02294.

[22] Alex H. Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. "PointPillars: Fast Encoders for Object Detection from Point Clouds". In: *CoRR* abs/1812.05784 (2018). arXiv: 1812.05784. URL: http://arxiv.org/abs/1812.05784.

[23] Laura Leal-Taixé, Cristian Canton-Ferrer, and Konrad Schindler. "Learning by tracking: Siamese CNN for robust target association". In: *CoRR* abs/1604.07866 (2016). arXiv: 1604.07866. URL: http://arxiv.org/abs/1604.07866.

[24] Laura Leal-Taixé, Anton Milan, Konrad Schindler, Daniel Cremers, Ian D. Reid, and Stefan Roth. "Tracking the Trackers: An Analysis of the State of the Art in Multiple Object Tracking". In: *CoRR* abs/1704.02781 (2017). arXiv: 1704.02781. URL: http://arxiv.org/abs/1704.02781.

[25] Guchong Li. "Multiple Model Poisson Multi-Bernoulli Mixture Filter for Maneuvering Targets". In: (2019).

[26] Qiang Li, Ranyang Li, Kaifan Ji, and Wei Dai. "Kalman Filter and Its Application". In: *2015 8th International Conference on Intelligent Networks and Intelligent Systems (ICINIS)*. 2015, pp. 74–77. DOI: 10.1109/ICINIS.2015.35.

[27] Wenhan Luo, Junliang Xing, Anton Milan, Xiaoqin Zhang, Wei Liu, and Tae Kyun Kim. "Multiple object tracking: A literature review". In: *Artificial Intelligence* 293 (Apr. 2021). ISSN: 00043702. DOI: 10.1016/J.ARTINT.2020.103448.

[28] Wenhan Luo, Junliang Xing, Anton Milan, Xiaoqin Zhang, Wei Liu, and Tae Kyun Kim. *Multiple object tracking: A literature review*. Apr. 2021. DOI: 10.1016/j.artint.2020.103448.

[29] Wenjie Luo, Bin Yang, and Raquel Urtasun. "Fast and Furious: Real Time End-to-End 3D Detection, Tracking and Motion Forecasting with a Single Convolutional Net". In: *CoRR* abs/2012.12395 (2020). arXiv: 2012.12395. URL: https://arxiv.org/abs/2012.12395.

[30] Anton Milan, Seyed Hamid Rezatofighi, Anthony R. Dick, Konrad Schindler, and Ian D. Reid. "Online Multi-target Tracking using Recurrent Neural Networks". In: *CoRR* abs/1604.03635 (2016). arXiv: 1604.03635. URL: http://arxiv.org/abs/1604.03635.

[31] Katta G. Murty. "An Algorithm for Ranking all the Assignments in Order of Increasing Cost". In: *Operations Research* 16.3 (1968), pp. 682–687. ISSN: 0030364X, 15265463. URL: http://www.jstor.org/stable/168595.

[32] NuScenes. *NuScenes Tracking Tasks*. 2019. URL: https://www.nuscenes.org/tracking?externalData=all&mapData=all&modalities=Any (visited on 02/10/2022).

[33] Su Pang and Hayder Radha. "Multi-Object Tracking using Poisson Multi-Bernoulli Mixture Filtering for Autonomous Vehicles". In: (Mar. 2021). URL: http://arxiv.org/abs/2103.07783.

[34] Ziqi Pang, Zhichao Li, and Naiyan Wang. "SimpleTrack: Understanding and Rethinking 3D Multi-object Tracking". In: *CoRR* abs/2111.09621 (2021). arXiv: 2111.09621. URL: https://arxiv.org/abs/2111.09621.

[35] Joseph Redmon and Ali Farhadi. "YOLO9000: Better, Faster, Stronger". In: *CoRR* abs/1612.08242 (2016). arXiv: 1612.08242. URL: http://arxiv.org/abs/1612.08242.

[36] Joseph Redmon and Ali Farhadi. "YOLOv3: An Incremental Improvement". In: *CoRR* abs/1804.02767 (2018). arXiv: 1804.02767. URL: http://arxiv.org/abs/1804.02767.

[37] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *CoRR* abs/1506.01497 (2015). arXiv: 1506.01497. URL: http://arxiv.org/abs/1506.01497.

[38] Seyed Hamid Rezatofighi, Anton Milan, Zhen Zhang, Qinfeng Shi, Anthony Dick, and Ian Reid. "Joint Probabilistic Data Association Revisited". In: *2015 IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 3047–3055. DOI: 10.1109/ICCV.2015.349.

[39] Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. "Tracking The Untrackable: Learning To Track Multiple Cues with Long-Term Dependencies". In: *CoRR* abs/1701.01909 (2017). arXiv: 1701.01909. URL: http://arxiv.org/abs/1701.01909.

[40] Siyu Tang, Bjoern Andres, Mykhaylo Andriluka, and Bernt Schiele. "Multi-Person Tracking by Multicut and Deep Matching". In: *CoRR* abs/1608.05404 (2016). arXiv: 1608.05404. URL: http://arxiv.org/abs/1608.05404.

[41] Ba-Ngu Vo, Mahendra Mallick, Yaakov bar-shalom, Stefano Coraluppi, Richard III, Ronald Mahler, and Ba-Tuong Vo. "Multitarget Tracking". In: *Wiley Encyclopedia* (Sept. 2015), pp. 1–25. DOI: 10.1002/047134608X.W8275.

[42] E.A. Wan and R. Van Der Merwe. "The unscented Kalman filter for nonlinear estimation". In: *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*. 2000, pp. 153–158. DOI: 10.1109/ASSPCC.2000.882463.

[43] Xinshuo Weng, Jianren Wang, David Held, and Kris Kitani. "AB3DMOT: A Baseline for 3D Multi-Object Tracking and New Evaluation Metrics". In: *CoRR* abs/2008.08063 (2020). arXiv: 2008.08063. URL: https://arxiv.org/abs/2008.08063.

[44] Jason L. Williams. "Alternative multi-Bernoulli filters (extended version)". In: *CoRR* abs/1203.2995 (2012). arXiv: 1203.2995. URL: http://arxiv.org/abs/1203.2995.

[45] Yuxuan Xia, Karl Granström, Lennart Svensson, Maryam Fatemi, Ángel F. García-Fernández, and Jason L. Williams. "Poisson Multi-Bernoulli Approximations for Multiple Extended Object Filtering". In: (Jan. 2018). URL: http://arxiv.org/abs/1801.01353.

[46] Yan Yan, Yuxing Mao, and Bo Li. "SECOND: Sparsely EMbedded Convolutional Detection". In: *Sensors* 18, no. 10: 3337 (2016). MDPI: 1612.08242. URL: https://www.mdpi.com/1424-8220/18/10/3337.

[47] Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. "Center-based 3D Object Detection and Tracking". In: *CoRR* abs/2006.11275 (2020). arXiv: 2006.11275. URL: https://arxiv.org/abs/2006.11275.

[48] Benjin Zhu, Zhengkai Jiang, Xiangxin Zhou, Zeming Li, and Gang Yu. "Class-balanced Grouping and Sampling for Point Cloud 3D Object Detection". In: *CoRR* abs/1908.09492 (2019). arXiv: 1908.09492. URL: http://arxiv.org/abs/1908.09492.

[49] Ji Zhu, Hua Yang, Nian Liu, Minyoung Kim, Wenjun Zhang, and Ming-Hsuan Yang. "Online Multi-Object Tracking with Dual Matching Attention Networks". In: *CoRR* abs/1902.00749 (2019). arXiv: 1902.00749. URL: http://arxiv.org/abs/1902.00749.

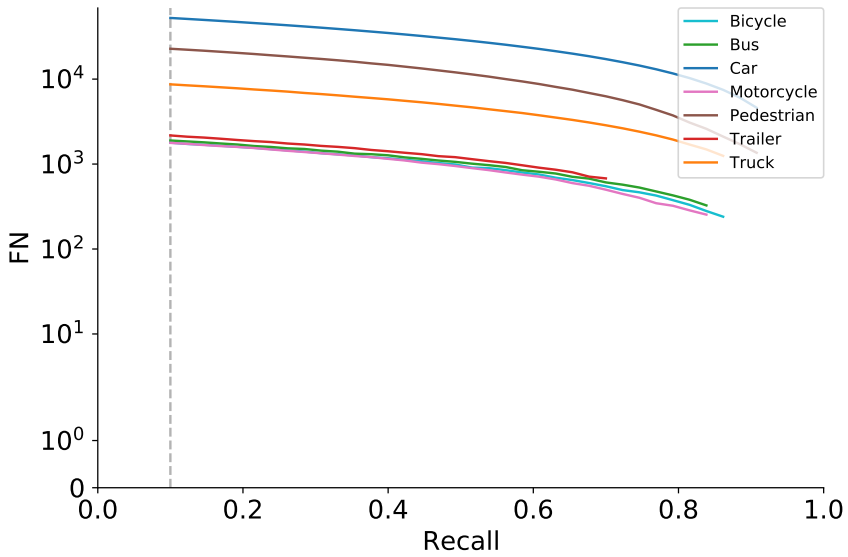
A Parameters

The parameters used for optimal performance of the PMBM filter are found here:

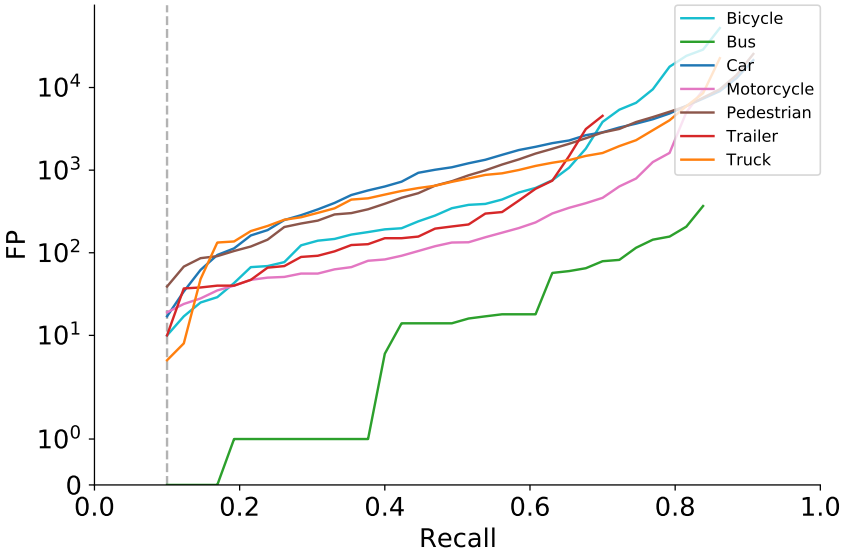
1. Survival probability p_s: 0.85
2. Detection probability p_d: 0.95
3. Poisson birth weight: $ln(0.1)$
4. Poisson birth gate size: 11
5. Poisson log-weights pruning threshold: -5
6. Poisson intensity: 10^{-4}
7. Maximum number of global hypothesis: 5
8. Existence pruning threshold: 10^{-3}
9. Log-weights pruning threshold: -5

The diagonal terms of the covariance matrices for motion/measurement noise depends on the class of the tracked object. The values of these covariance matrix terms are calculated as the mean standard deviation of the respective component on the training set as presented in [7].

B Figures for CenterPoint detector

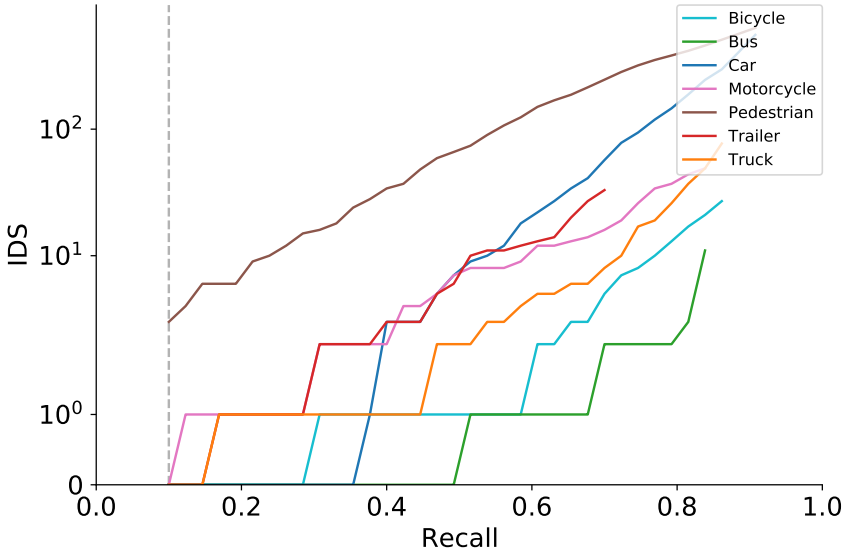


(a) FN at different recall threshold



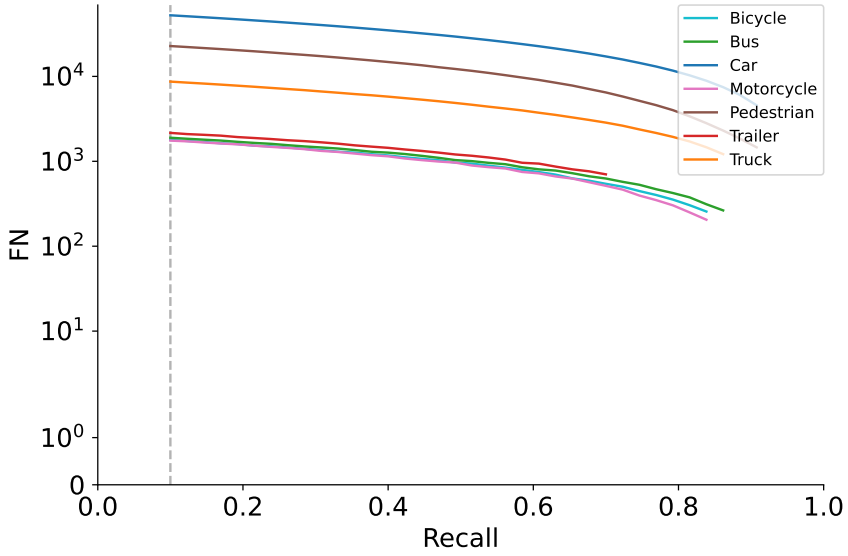
(b) FP at different recall threshold

Figure B.1: Recall curves for CenterPointEnsemble

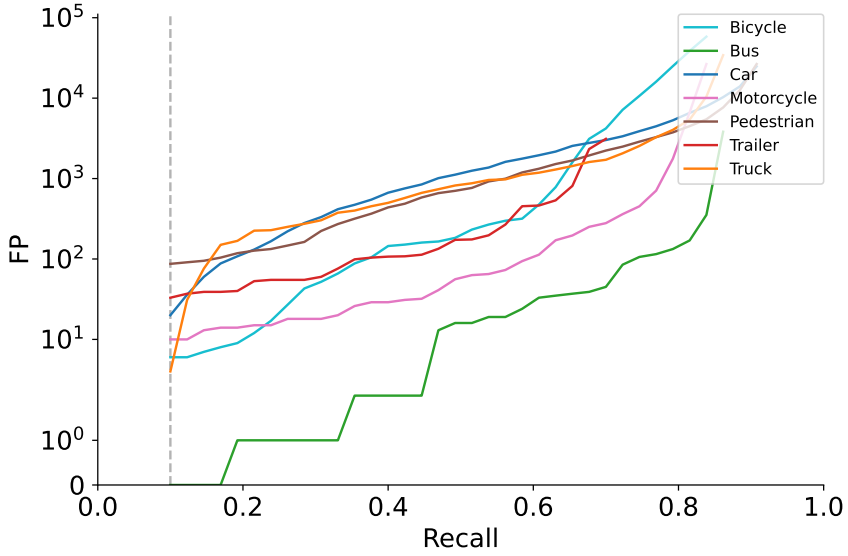


(a) IDS at different recall threshold

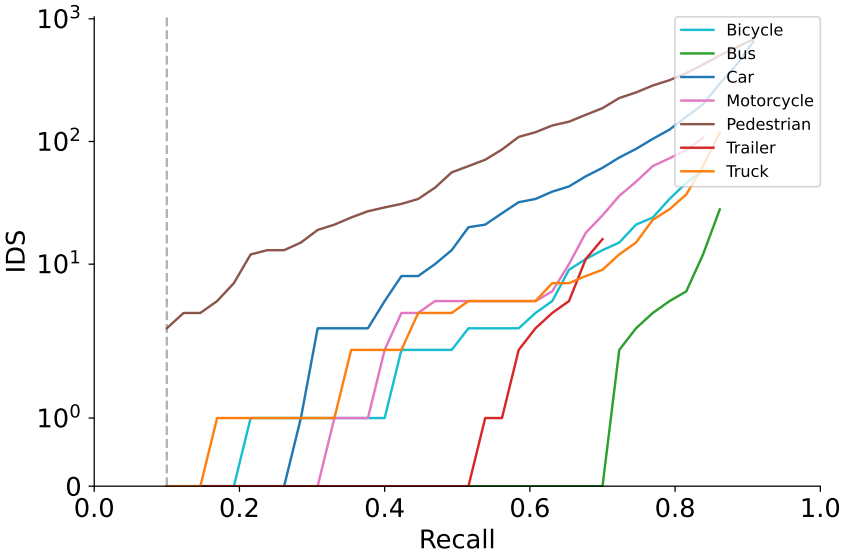
Figure B.2: Recall curves for CenterPointEnsemble



(a) FN at different recall threshold

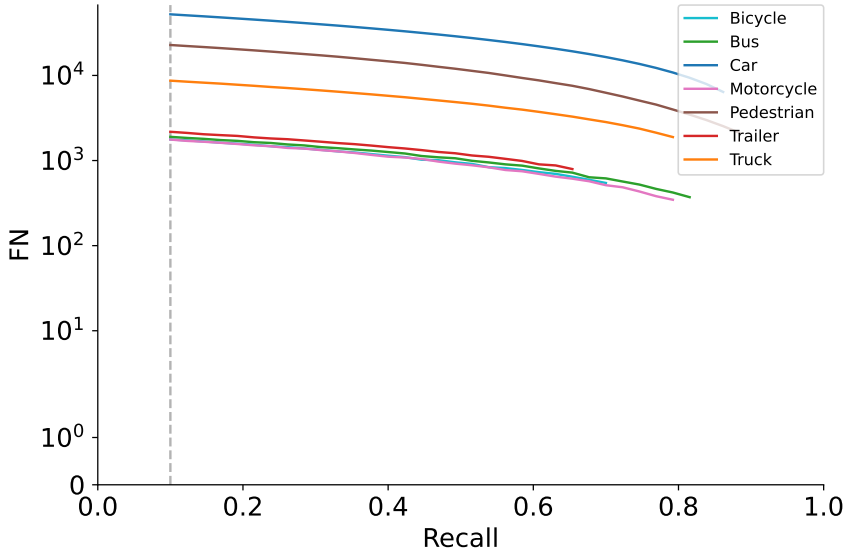


(b) FP at different recall threshold

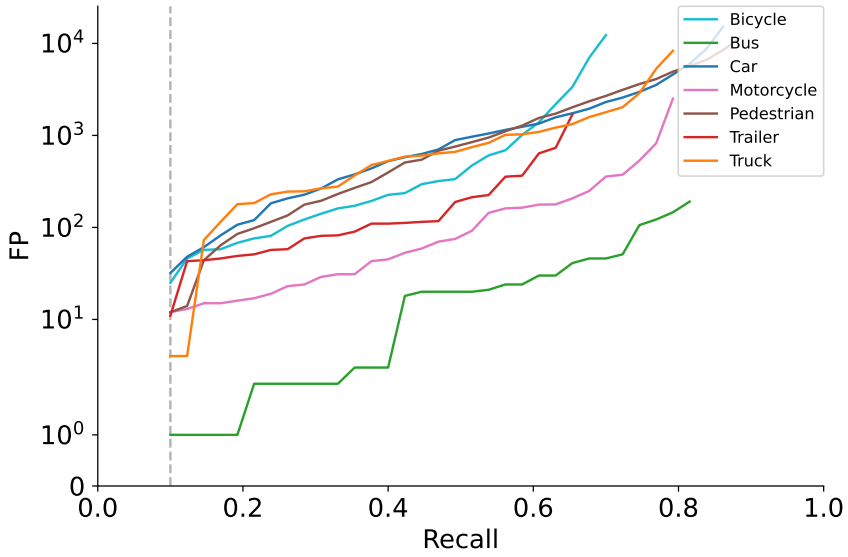


(c) IDS at different recall threshold

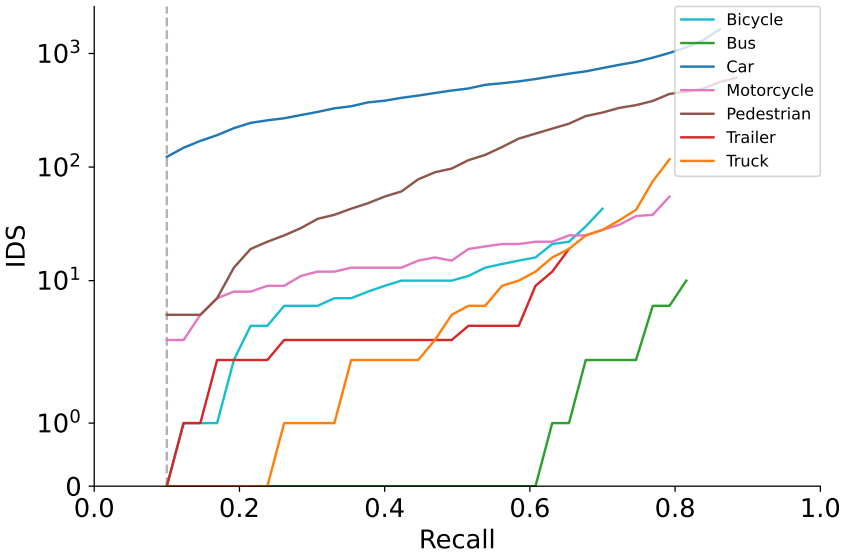
Figure B.3: Recall curves for SimpleTrack



(a) FN at different recall threshold



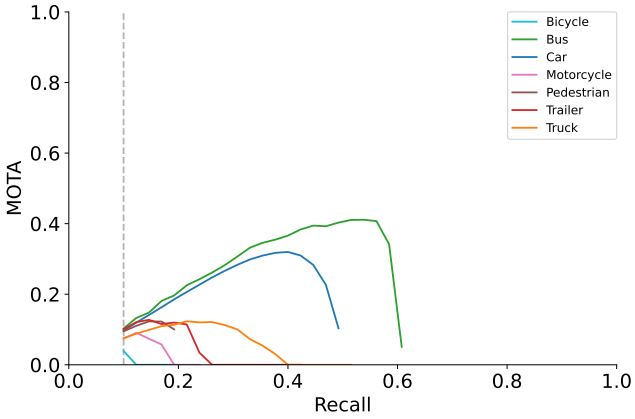
(b) FP at different recall threshold



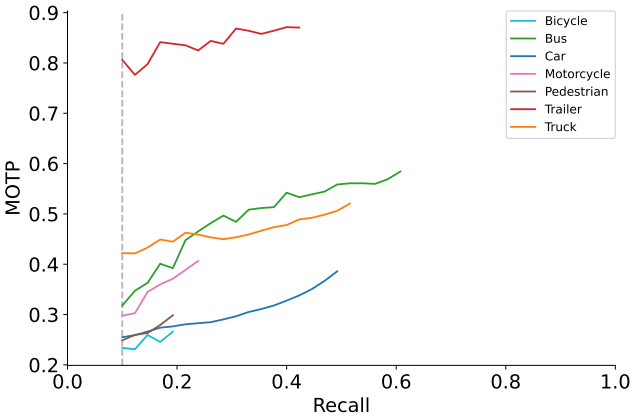
(c) IDS at different recall threshold

Figure B.4: Recall curves for PMBM

C Figures for Megvii detectior

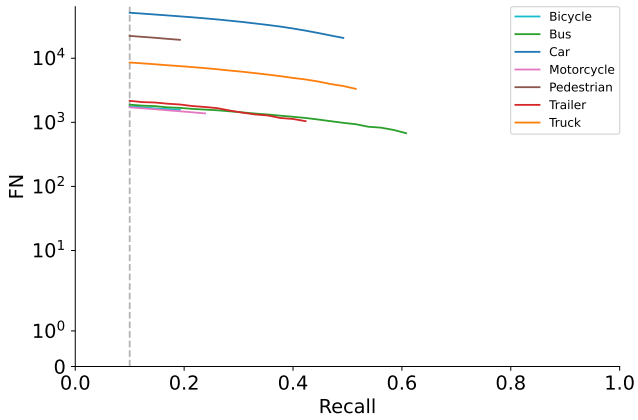


(a) MOTA at different recall threshold

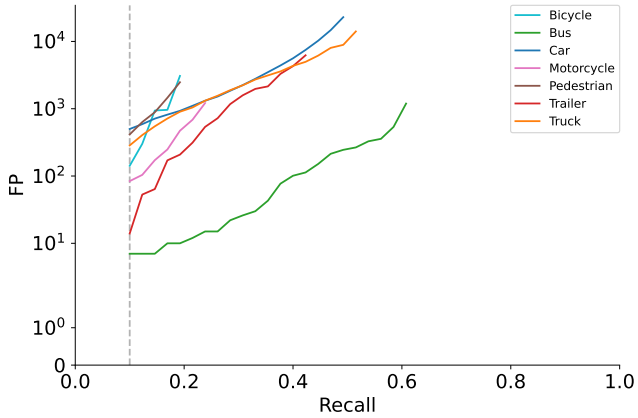


(b) MOTP at different recall threshold

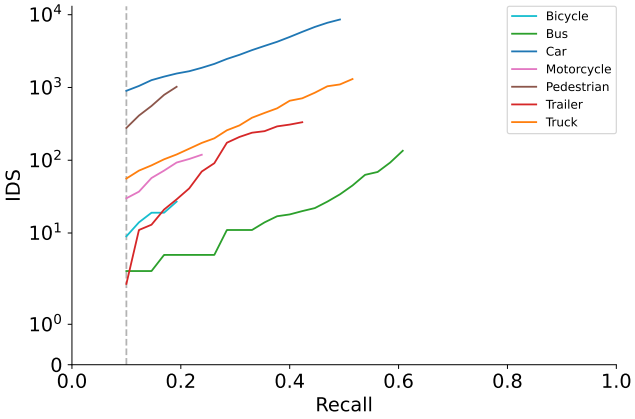
Figure C.1: MOTA & MOTP recall curves for AB3DMOT-megvii



(a) FN at different recall threshold

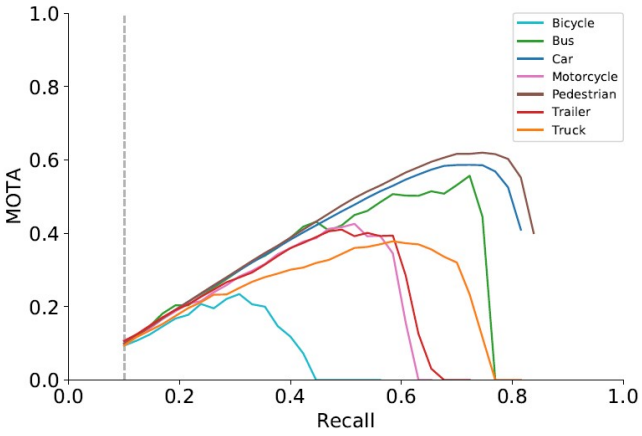


(b) FP at different recall threshold

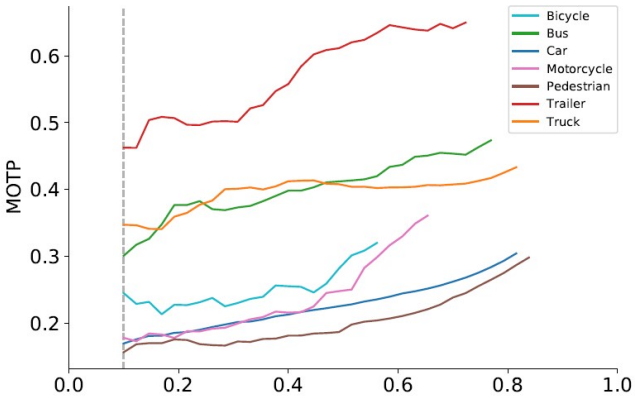


(c) IDS at different recall threshold

Figure C.2: FN, FP & IDS recall curves for AB3DMOT-megvii

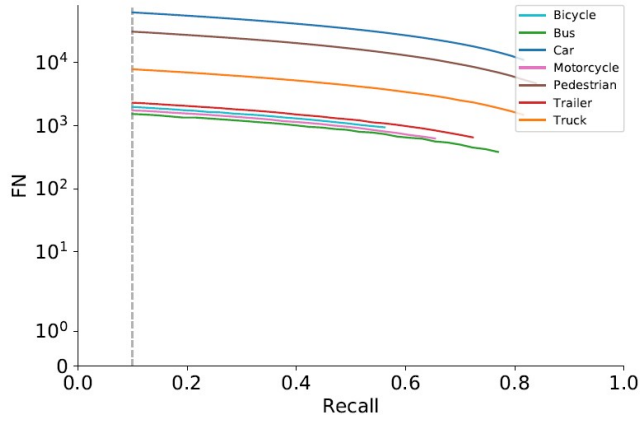


(a) MOTA at different recall threshold

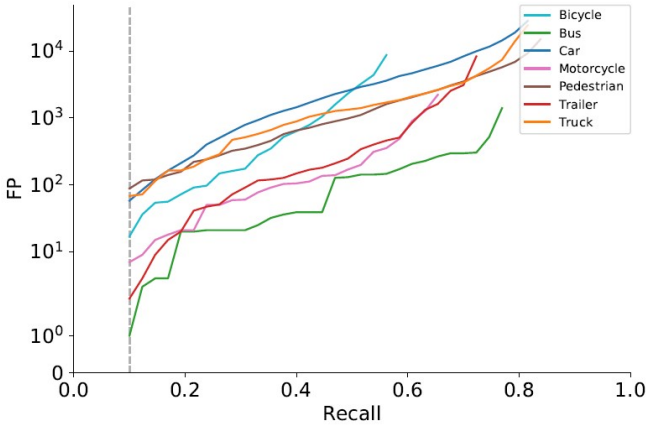


(b) MOTP at different recall threshold

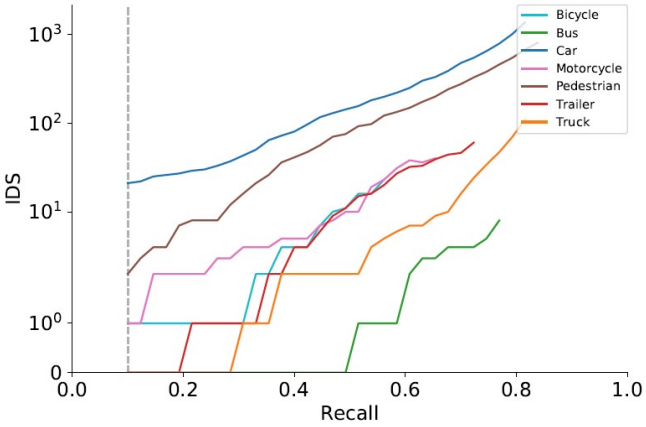
Figure C.3: MOTA & MOTP recall curves for StanfordPRL-TRI



(a) FN at different recall threshold

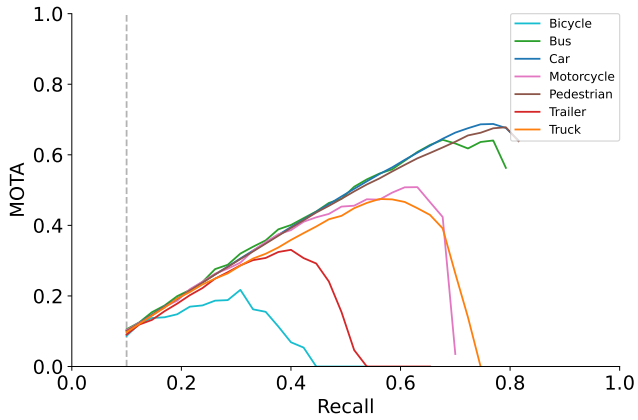


(b) FP at different recall threshold

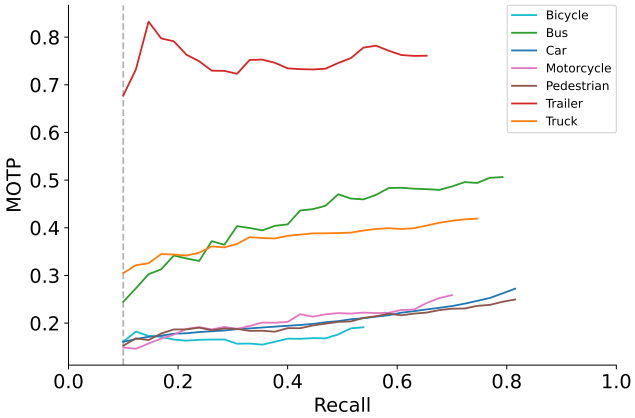


(c) IDS at different recall threshold

Figure C.4: FN, FP & IDS recall curves for StanfordPRL-TRI

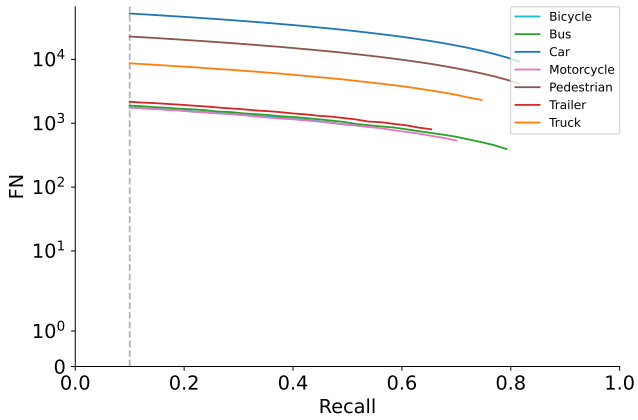


(a) MOTA at different recall threshold

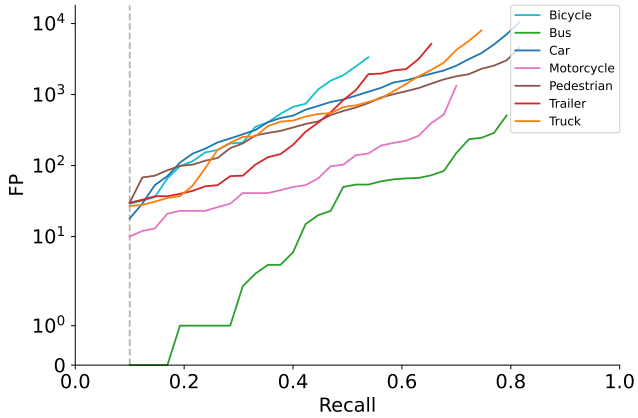


(b) MOTP at different recall threshold

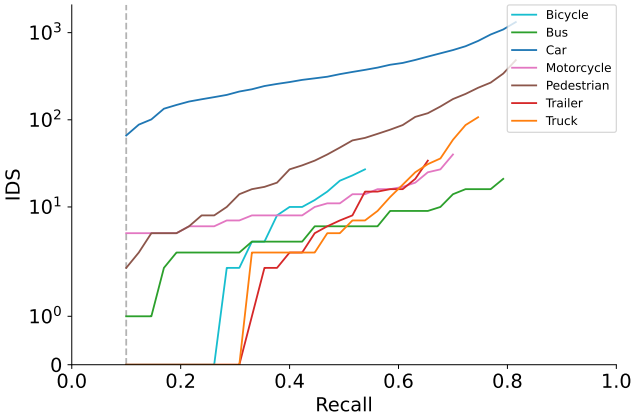
Figure C.5: MOTA & MOTP recall curves for PMBM-megvii



(a) FN at different recall threshold



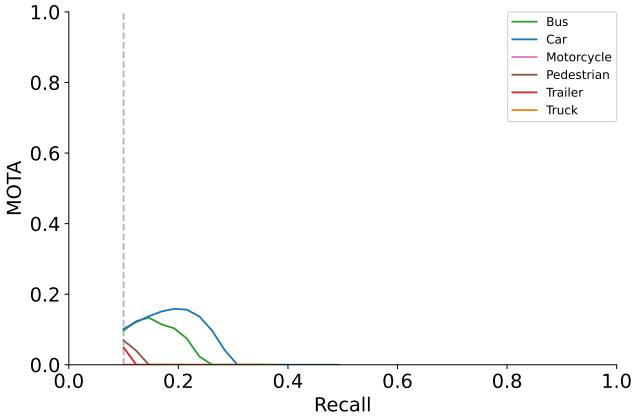
(b) FP at different recall threshold



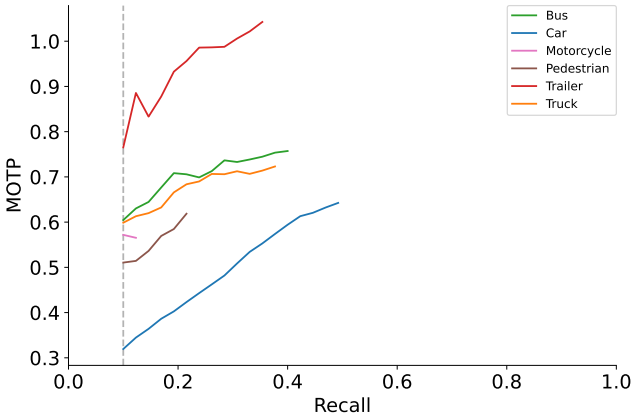
(c) IDS at different recall threshold

Figure C.6: FN, FP & IDS recall curves for PMBM-megvii

D Figures for PointPillars detector

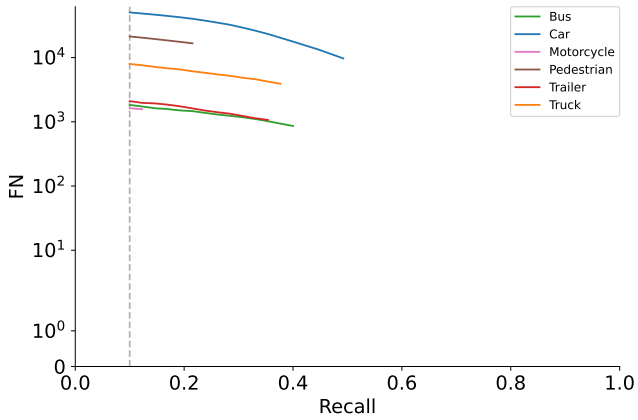


(a) MOTA at different recall threshold

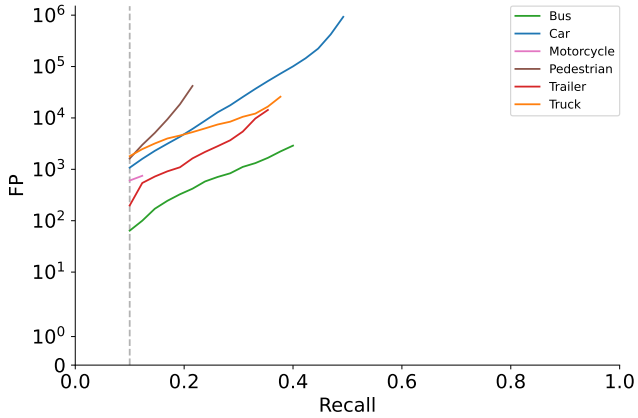


(b) MOTP at different recall threshold

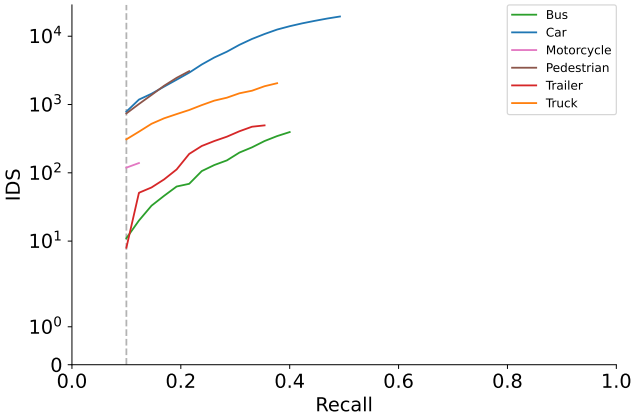
Figure D.1: MOTA & MOTP recall curves for AB3DMOT-PointPillars



(a) FN at different recall threshold

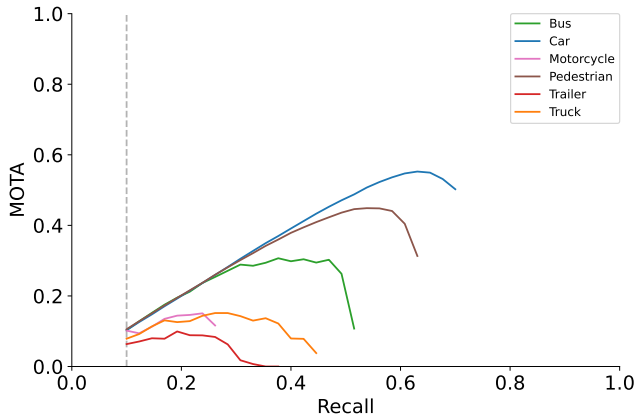


(b) FP at different recall threshold

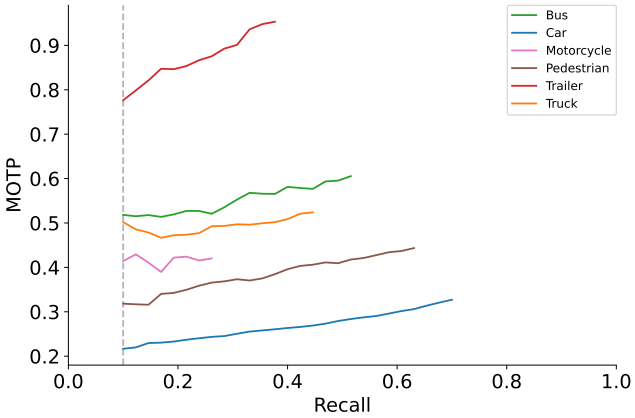


(c) IDS at different recall threshold

Figure D.2: FN, FP & IDS recall curves for AB3DMOT-PointPillars

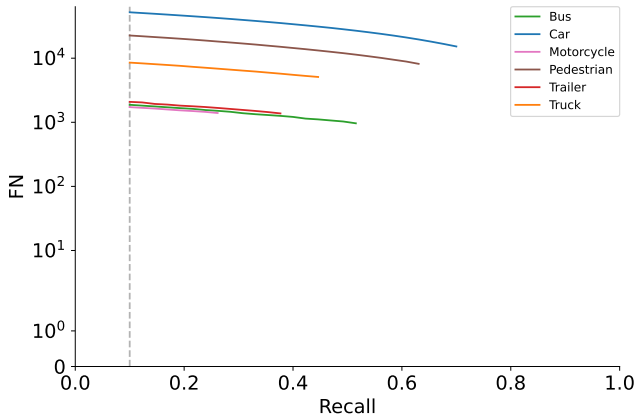


(a) MOTA at different recall threshold

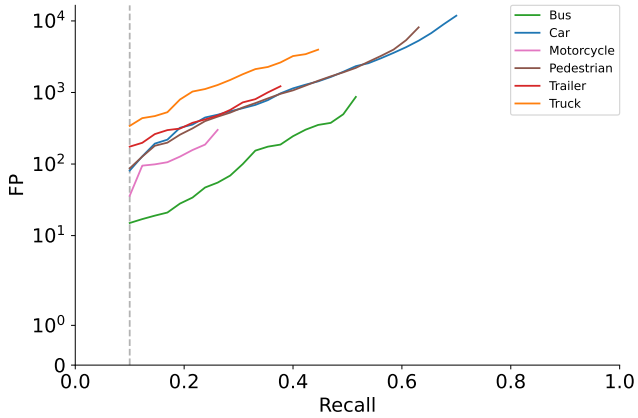


(b) MOTP at different recall threshold

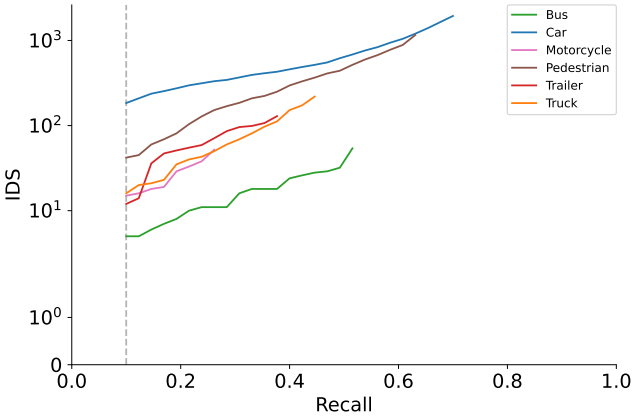
Figure D.3: MOTA & MOTP recall curves for PMBM-PointPillars



(a) FN at different recall threshold



(b) FP at different recall threshold



(c) IDS at different recall threshold

Figure D.4: FN, FP & IDS recall curves for PMBM-PointPillars