2018 IEEE Intelligent Vehicles Symposium (IV)
Changshu, Suzhou, China, June 26-30, 2018

Mono-Camera 3D Multi-Object Tracking Using Deep Learning Detections and PMBM Filtering

Samuel Scheidegger*†, Joachim Benjaminsson*†, Emil Rosenberg†, Amrit Krishnan*, Karl Granström†
*Zenuity, †Department of Electrical Engineering, Chalmers University of Technology
*{firstname.lastname}@zenuity.com, karl.granstrom@chalmers.se

***Abstract*—Monocular cameras are one of the most commonly used sensors in the automotive industry for autonomous vehicles. One major drawback using a monocular camera is that it only makes observations in the two dimensional image plane and can not directly measure the distance to objects. In this paper, we aim at filling this gap by developing a multi-object tracking algorithm that takes an image as input and produces trajectories of detected objects in a world coordinate system. We solve this by using a deep neural network trained to detect and estimate the distance to objects from a single input image. The detections from a sequence of images are fed in to a state-of-the art Poisson multi-Bernoulli mixture tracking filter. The combination of the learned detector and the PMBM filter results in an algorithm that achieves 3D tracking using only mono-camera images as input. The performance of the algorithm is evaluated both in 3D world coordinates, and 2D image coordinates, using the publicly available KITTI object tracking dataset. The algorithm shows the ability to accurately track objects, correctly handle data associations, even when there is a big overlap of the objects in the image, and is one of the top performing algorithms on the KITTI object tracking benchmark. Furthermore, the algorithm is efficient, running on average close to 20 frames per second.**

I. Introduction

To enable a high level of automation in driving, it is necessary to accurately model the surrounding environment, a problem called environment perception. Data from onboard sensors, such as cameras, radars and lidars, has to be processed to extract information about the environment needed to automatically and safely navigate the vehicle. For example, information about both the static environment, such as road boundaries and lane information, and the dynamic objects, like pedestrians and other vehicles, is of importance. The focus of this paper is the detection and tracking of multiple dynamic objects, specifically vehicles.

Dynamic objects are often modeled by state vectors, and are estimated over time using a multi-object tracking (MOT) framework. MOT denotes the problem of, given a set of noisy measurements, estimating both the number of dynamic objects, and the state of each dynamic object. Compared to the single object tracking problem, in addition to handling measurement noise and detection uncertainty, the MOT problem also has to resolve problems like object birth and object death[1]; clutter detections[2]; and unknown measurement origin.

A recent family of MOT algorithms are based on random finite sets (RFSs) [1]. The probability hypothesis density (PHD) [2] filter, and the cardinalized PHD (CPHD) [3] filter, are two examples of moment approximations of the multi-object density. The generalized labeled multi-Bernoulli (GLMB) [4], [5] and the Poisson multi-Bernoulli mixture (PMBM) [6], [7] filters are examples of MOT filters based on multi-object conjugate priors; these filters have been shown to outperform filters based on moment approximation. A recent comparison study published in [8] has shown that the filters based on the PMBM conjugate prior both achieves greater tracking performance, and has favourable computational cost compared to GLMB, hence we use the PMBM filter in this work.

All of the aforementioned MOT algorithms takes sets of object estimates, or *detections*, as their input. This implies that the raw sensor data, e.g., the images, should be pre-processed into detections. The recent development of deep neural networks has lead to big improvement in fields of image processing. Indeed, considerable improvements have be achieved for the object detection problem, see, e.g., [9], [10], which is is crucial to the tracking performance.

Convolutional neural networks (CNNs) [11] have shown to vastly outperform previous methods in image processing for tasks such as classification, object detection and semantic segmentation. CNNs make use of the spatial relation between neighbouring pixels in images, by processing data in a convolutional manner. Each layer in a CNN consists of a filter bank with a number of convolutional kernels, where each element is a learnable parameter.

The most common approach for object detection using deep neural networks is region-based CNNs (R-CNNs). R-CNNs are divided into two parts; a region proposal network (RPN), followed by a box regression and classification network. The RPN takes an image as input, and outputs a set of general object proposals, which are fed into the following classification and box regression network. The box regression and classification network will refine the size of the object and classify it into one of the object classes. This type of deep neural network structure is used in, e.g., Fast R-CNN [12], and later in the

[1]Object birth and object death is when an object first appears within, and departs from, the ego-vehicle's surveillance area, respectively.

[2]Clutter detections are false detections, i.e., detections not corresponding to an actual object.

978-1-5386-4452-2/18/$31.00 ©2018 IEEE

improved Faster R-CNN [9]. Another approach to the object detection problem is you only look once (YOLO) [10]. Here, the region proposal step is omitted, and the box regression and classification are applied directly on the entire image.

Most previous work on object detection in mono camera data is limited to positioning object in the image plane. However, there are some examples of work on 3D object detection in mono camera data. In [13], the distance to the detected object is recovered by using the fact that the object should be on the ground plane, while [14] uses geometrical constraints and a deep neural network to regress the dimensions of the object. Another approach to 3D object detection is taken in [15], where the object detections are classified to a set of 3D vehicle templates that are then fitted to the 2D detections.

In the automotive industry, monocular camera is a well studied and commonly used type of sensors for developing autonomous driving systems. A monocular camera is a mapping between 3D world coordinates and 2D image coordinates [16] where, in contrary to, e.g., radars and lidars, distance information is lost. Previous work on object tracking using monocular camera data is restricted to tracking in the image-plane, see, e.g., [17], [18], [19], for some recent work.

To achieve a high level of vehicle automation, 2D tracking in the image plane is not adequate. Instead, we need to track objects in world coordinates in order to obtain the relative pose between the ego vehicle the detected objects, information that is crucial for automatic decision making and control. We refer to this as 3D tracking.

The contribution of this paper is a multi-vehicle 3D tracking algorithm, that takes as input monocular camera data, and outputs vehicle estimates in world coordinates. The presented 3D tracking filter has two main components: a detector and an object tracking filter. The detector is a deep neural network trained to from an input image not only extract a 2D bounding box for each detected object, but also to estimate the distance from the camera to the object. This is achieved by using object annotations in lidar data during the learning of the network parameters. The object tracking filter is a state-of-the-art PMBM object tracking filter [6], [7] that processes the detections and outputs estimates. The tracking filter is computationally efficient, and handles both false detections and missed detections. For each object, a position, as well as kinematical properties such as velocity, are estimated. The proposed MOT algorithm is evaluated using the image sequences from the publicly available KITTI tracking dataset [20], and the results show that accurate 3D tracking is achieved.

The paper is structured as follows. In Section II, we give a problem formulation and present an overview of the algorithm. In Section III we present the object detection, and in Section IV we present object tracking. The results of an experimental evaluation using data sequences from the KITTI dataset are presented in Section V, and the paper is concluded in Section VI.

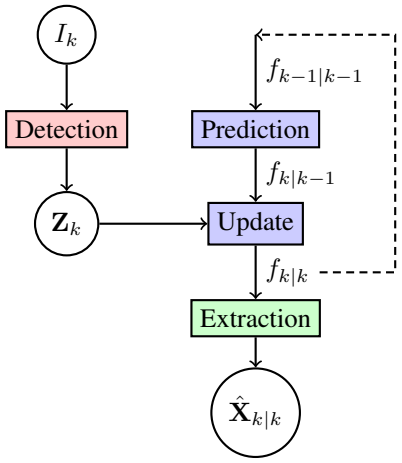


Fig. 1. Algorithm overview. The MOT algorithm has two modules, detection (left, red) and tracking (right, blue/green). The tracking modules consists of a recursive tracking filter (prediction+update) and an object extraction.

II. Problem formulation and algorithm overview

The KITTI object tracking dataset [20] contains data from multiple sensors, e.g., four cameras and a lidar sensor. The data from such sensors can be used for environment perception, i.e., tracking of moving objects and mapping of the stationary environment. In this work we focus on data from a forward looking camera, with the objective to track the other vehicles that are in the environment.

Each vehicle is represented by a state vector $\mathbf{x}$ that contains the relevant information about the object. For 3D object tracking, the following state vector is used,

$$\mathbf{x} = \begin{bmatrix} x & y & z & v_x & v_y & v_z & w & h \end{bmatrix}^T, \tag{1}$$

where (x, y, z) is the 3D position in world coordinates, (v_x, v_y, v_z) is the corresponding velocity, and (w, h) is the width and height of the object's bounding box in the camera image. The position and velocity describes the tracked object's properties of interest; the width and height of the bounding box are used for evaluation analogue to the KITTI object tracking benchmark [20].

The number of vehicles in the environment is not known, and changes with time, so the task is to estimate both the number of vehicles, as well as each vehicle's state. The vehicles at time step k are represented by a set $\mathbf{X}_k$ that contains the state vectors of all vehicles that are present in the vicinity of the ego-vehicle. The set of vehicles $\mathbf{X}_k$ is modeled as a Random Finite Set (RFS) [1]. That is, the number of objects, or the cardinality of the set, is modeled as a time varying discrete random variable and each object's state is a multivariate random variable.

The problem addressed in this paper is the processing of the sequence of images I_k into a sequence of estimates $\hat{\mathbf{X}}_{k|k}$ of the set of vehicles,

$$I_0, I_1, \ldots, I_k \qquad \Rightarrow \qquad \hat{\mathbf{X}}_{0|0}, \hat{\mathbf{X}}_{1|1}, \ldots, \hat{\mathbf{X}}_{k|k}, \tag{2}$$

where the sub-indices denote time. In other words, we wish to process the image sequence to gain information at each time step about the number of vehicles (the cardinality of the set $\mathbf{X}$), and the state of each vehicle. The proposed MOT algorithm has two main parts: object detection, and object tracking; an illustration of the algorithm is given in fig. 1.

In the detection module, each image is processed to output a set of object detections $\mathbf{Z}_k$,

$$I_k \quad \xRightarrow{\text{Detection}} \quad \mathbf{Z}_k. \tag{3}$$

The set of detections $\mathbf{Z}_k$, where each $\mathbf{z}_k^i \in \mathbf{Z}_k$ is an estimated object, is also modeled as a RFS. The detection is based on a CNN, which is presented in detail in Section III.

The tracking module takes the image detections as input and outputs an object set estimate; it has three parts: prediction, update, and extraction. Together, the prediction and the update constitute a tracking filter that recursively estimates a multi-object set density,

$$\mathbf{Z}_k \quad \xRightarrow{\text{PMBM filter}} \quad f_{k|k}(\mathbf{X}_k|\mathbf{Z}^k), \tag{4}$$

where $\mathbf{Z}^k$ denotes all measurement sets up to time step k, $\{\mathbf{Z}_t\}_{t\in(0,k)}$. Specifically, in this work we estimate a PMBM density [6]. The Chapman-Kolmogorov prediction

$$f_{k|k-1}(\mathbf{X}_k|\mathbf{Z}^{k-1}) = \int g(\mathbf{X}_k|\mathbf{X}_{k-1}) f_{k-1|k-1}(\mathbf{X}_{k-1}|\mathbf{Z}^{k-1}) \delta\mathbf{X}_{k-1}, \tag{5a}$$

predicts the PMBM density to the next time step using the multi-object motion model $g(\mathbf{X}_{k+1}|\mathbf{X}_k)$. We use the standard multi-object motion model [1], meaning that $g(\cdot|\cdot)$ models a Markovian process for objects that remain in the field of view, combined a Poisson point process (PPP) birth process.

Using the set of detections $\mathbf{Z}_k$ and the multi-object measurement model $h(\mathbf{Z}_k|\mathbf{X}_k)$, the updated PMBM density is computed using the Bayes update

$$f_{k|k}(\mathbf{X}_k|\mathbf{Z}^k) = \frac{h(\mathbf{Z}_k|\mathbf{X}_k) f_{k|k-1}(\mathbf{X}_k|\mathbf{Z}^{k-1})}{\int h(\mathbf{Z}_k|\mathbf{X}_k) f_{k|k-1}(\mathbf{X}_k|\mathbf{Z}^{k-1}) \delta\mathbf{X}_k}, \tag{5b}$$

We use the standard multi-object measurement model [1], in which $h(\mathbf{Z}_k|\mathbf{X}_k)$ models noisy measurement with detection uncertainty, combined with PPP clutter.

The final part of the tracking is the object extraction, where object estimates are extracted from the PMBM density,

$$f_{k|k}(\mathbf{X}_k|\mathbf{Z}^k) \quad \xRightarrow{\text{Extraction}} \quad \hat{\mathbf{X}}_{k|k} \tag{6}$$

The tracking is described further in Section IV. The integrals in (5) are set-integrals, defined in [1].

III. Object Detection

In this section, we describe how deep learning, see, e.g., [21], is used to process the images $\{I_t\}_{t=0}^k$ to output sets of detections $\{\mathbf{Z}_t\}_{t=0}^k$. For an image I_t with a corresponding set of detections $\mathbf{Z}_t$, each detection $\mathbf{z} \in \mathbf{Z}_t$ consists of a 2D bounding box and a distance from the camera center to the center of the detected object,

$$\mathbf{z} = \begin{bmatrix} x_{min} & y_{min} & x_{max} & y_{max} & d \end{bmatrix}^T, \tag{7}$$

where (x_{min}, y_{min}) and (x_{max}, y_{max}) are the pixel positions of the top left and bottom right corner of the bounding box, respectively, and d is the distance from the camera to the object. The bounding box encloses the object in the image. Using this information, the angle from the camera center to the center of the detected object can be inferred. This, together with the camera-to-object-distance d, allows the camera to be transformed into a range/bearing sensor, which is suitable for object tracking in 3D world coordinates.

The object detection is implemented using a improved version of the network developed in [22]. The network can be divided into two parts; the first part can be viewed as a feature extractor, and the second part consists of three parallel output headers. The feature extractor is identical to the DRN-C-26 [23] network, with the exception that the last two classification layers have been removed. The last two layers are structured for the original classification task of DRN-C-26, which is not suitable in this work.

To represent objects using a bounding box and its distance, the network has three different types of output: classification score, bounding box and distance. Each header in the network has two 1×1 convolutional layers and finally a sub-pixel convolutional layer [24], upscaling the output to 1/4th of the input image resolution. The bounding box header has 4 output channels, representing the top left and bottom right corner of the bounding box, the distance header has one output channel, representing the distance to the object, and the classification header has an additional softmax function and represents the different class scores using one-hot encoding, i.e., one output channel for each class, where each channel represents the score for each class, respectively. For each pixel in the output layer there will be an estimated bounding box, i.e., there can be more than one bounding box per object. To address this, Soft-NMS [25] is applied. In this step, the box with the highest classification score is selected and the score of boxes intersecting the selected box are decayed according to a function of the intersection over union (IOU). This process is repeated until the classification score of all remaining boxes are below a manually chosen threshold.

TABLE I
Table of Notation

- Minor non-bold letter, e.g., a, b, γ, denote scalars.
- Minor bold letters, e.g., $\mathbf{x}$, $\mathbf{z}$, $\boldsymbol{\xi}$, denote vectors.
- Capital non-bold letters , e.g., M, F, H, denote matrices.
- Capital bold letters, e.g., $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$, denote sets.
- $|\mathbf{X}|$ denotes the cardinality of set $\mathbf{X}$, i.e., the number of elements in $\mathbf{X}$.
- $\uplus$ denotes disjoint set union, i.e., $\mathbf{X} \uplus \mathbf{Y} = \mathbf{Z}$ means $\mathbf{X} \cup \mathbf{Y} = \mathbf{Z}$ and $\mathbf{X} \cap \mathbf{Y} = \emptyset$.
- $[h(\cdot)]^{\mathbf{X}} = \prod_{\mathbf{x}\in\mathbf{X}} h(\mathbf{x})$ and $[h(\cdot)]^{\emptyset} = 1$ by definition.
- $\langle a; b\rangle = \int a(x)b(x)\mathrm{d}x$, the inner product of $a(x)$ and $b(x)$.

The feature extractor is pre-trained on ImageNet [26], and the full network is fine-tuned using annotated object labels from the KITTI object data set [20]. The objects in the KITTI object data set are annotated with 3D bounding boxes. By projecting the 3D bounding boxes into the camera image, pixel positions (x_{min}, y_{min}) and (x_{max}, y_{max}) for the 2D bounding boxes in the image plane are obtained. Camera-to-object distances d are computed as the Euclidean distances from the camera to the centres of the 3D bounding boxes. The network is tuned using stochastic gradient descent with momentum. The task of classification used a cross entropy loss function, while bounding box regression and distance estimation used a smooth L1 loss function [12].

IV. Object tracking

To associate objects between consecutive frames and filter the object detections from the neural network, a PMBM tracking filter is applied. Both the set of objects $\mathbf{X}_k$ and the set of image detections $\mathbf{Z}_k$ are modeled as RFSs. The purpose of the tracking module is to process the sequence of detection sets, and output a sequence of estimates $\hat{\mathbf{X}}_{k|k}$ of the true set of objects. We achieve this by using a PMBM filter to estimate the multi-object density $f_{k|k}(\mathbf{X}_k|\mathbf{Z}_k)$, and to extract estimates from this density.

In this section, we first present some necessary RFS background, and the standard point object models that are used to model both the object motion, as well as the detection process. Then, we present the PMBM filter.

A. RFS *background*

In this work, two types of RFSs are important: the PPP and the Bernoulli process. A general introduction to RFS is given in, e.g., [1].

1) Poisson point process: A PPP is a type of RFS where the cardinality is Poisson distributed and all elements are independent and identically distributed (IID). A PPP can be parametrized by an intensity function, $D(\mathbf{x})$, defined as

$$D(\mathbf{x}) = \mu f(\mathbf{x}). \tag{8}$$

The intensity function has two parameters, the Poisson rate $\mu > 0$ and the spatial distribution $f(\mathbf{x})$. The expected number of set members in a PPP S is $\int_{\mathbf{x}\in S} D(\mathbf{x})\mathrm{d}\mathbf{x}$.

The PPP density is

$$f(\mathbf{X}) = e^{-\langle D(\mathbf{x}),1\rangle} \prod_{\mathbf{x}\in\mathbf{X}} D(\mathbf{x}) = e^{-\mu} \prod_{\mathbf{x}\in\mathbf{X}} \mu f(\mathbf{x}). \tag{9}$$

The PPPs are used to model object birth, undetected objects and clutter measurements.

2) Bernoulli process: A Bernoulli RFS is a RFS that with the probability r contains a single element with the probability density function (PDF) $f(\mathbf{x})$, and with the probability $1-r$ is empty:

$$f(\mathbf{X}) = \begin{cases} 1-r, & \mathbf{X} = \emptyset \\ rf(\mathbf{x}), & \mathbf{X} = \{\mathbf{x}\} \\ 0, & |X| > 1 \end{cases}. \tag{10}$$

It is suitable to use a Bernoulli RFS to model objects in a MOT problem, since it both models the object's probability of existence r, and uncertainty in its state $\mathbf{x}$.

In MOT, the objects are typically assumed to be independent [6]. The disjoint union of a fixed number of independent Bernoulli RFSs, $\mathbf{X} = \uplus_{i\in\mathbb{I}}\mathbf{X}^i$, where $\mathbb{I}$ is an index set, is a multi-Bernoulli (MB) RFS. The parameters $\{r^i, f^i(\cdot)\}_{i\in\mathbb{I}}$ defines the MB distribution.

A multi-Bernoulli mixture (MBM) density is a normalized, weighted sum of MB densities. The MBM density is entirely defined by $\{w^j, \{r^{j,i}, f^{j,i}(\cdot)\}_{i\in\mathbb{I}^j}\}_{j\in\mathbb{J}}$, where $\mathbb{J}$ is an index set for the MBs in the MBM, w^j is the probability of the jth MB, and $\mathbb{I}^j$ is the index set for the Bernoulli distributions. In a MOT problem, the different MBs typically corresponds to different data association sequences.

B. Standard models

Here we present the details of the standard measurement and motion models, under Gaussian assumptions.

1) Measurement model: Let $\mathbf{x}_k^i$ be the state of the ith vehicle at the kth time step. At time step k, given a set of objects $\mathbf{X}_k = \{\mathbf{x}_k^i\}_{i\in\mathbb{I}}$, the set of measurements is $\mathbf{Z}_k = (\uplus_{i\in\mathbb{I}}\mathbf{W}_k^i)\uplus\mathbf{K}_k$, where $\mathbf{W}_k^i$ denotes the set of object generated measurements from the ith object, $\mathbb{I}$ is an index set and $\mathbf{K}_k$ denotes the set of clutter measurements. The set $\mathbf{K}_k$ is modeled as a PPP with the intensity $\kappa(\mathbf{z}) = \lambda c(\mathbf{z})$, where λ is the Poisson rate and the spatial distribution $c(\mathbf{z})$ is assumed to be uniform.

Assuming an object is correctly detected with probability of detection p_D. If the object is detected, the measurement $\mathbf{z} \in \mathbf{W}_k^i$ has PDF $\phi_\mathbf{z}(\mathbf{x}_k^i) = \mathcal{N}(\mathbf{z}; a(\mathbf{x}_k^i), R)$, where $a(\mathbf{x}_k^i)$ is a camera measurement model. The resulting measurement likelihood is

$$\ell_\mathbf{Z}(\mathbf{x}) = p(\mathbf{Z}|\mathbf{x}) = \begin{cases} 1-p_\mathrm{D}, & \mathbf{Z} = \emptyset \\ p_\mathrm{D}\phi_\mathbf{z}(\mathbf{x}), & \mathbf{Z} = \{\mathbf{z}\} \\ 0, & |\mathbf{Z}| > 1 \end{cases}. \tag{11}$$

As can be seen in eq. (11), if multiple measurements are associated to one object this will have zero likelihood. This is a standard point object assumption, see, e.g., [1].

Because of the unknown measurement origin[3], it is necessary to discuss data association. Let the measurements in the set $\mathbf{Z}$ be indexed by $m \in \mathbb{M}$,

$$\mathbf{Z} = \{\mathbf{z}^m\}_{m\in\mathbb{M}}, \tag{12}$$

and let $\mathcal{A}^j$ be the space of all data associations A for the jth predicted global hypothesis, i.e., the jth predicted MB. A data association $A \in \mathcal{A}^j$ is an assignment of each measurement in $\mathbf{Z}$ to a source, either to the *background* (clutter or new object) or to one of the existing objects indexed by $i \in \mathbb{I}^j$. Note that $\mathbb{M} \cap \mathbb{I}^j = \emptyset$ for all j. The space of all data associations for the jth hypothesis is $\mathcal{A}^j = \mathcal{P}(\mathbb{M} \cup \mathbb{I}^j)$, i.e., a data association

[3]An inherent property of MOT is that it is unknown which measurements are from object and which are clutter, and among the object generated measurements, it is unknown which object generated which measurement. Hence, the update must handle this uncertainty.

$A \in \mathcal{A}^j$ is a partition of $\mathbb{M} \cup \mathbb{I}^j$ into non-empty disjoint subsets $C \in A$, called index cells[4].

Due to the standard MOT assumption that the objects generate measurements independent of each other, an index cell contains at most one object index and at most one measurement index, i.e., $|C \cap \mathbb{I}^j| \leq 1$ and $|C \cap \mathbb{M}| \leq 1$ for all $C \in A$. Any association in which there is at least one cell, with at least two object indices and/or at least two measurement indices, will have zero likelihood because this violates the independence assumption and the point object assumption, respectively. If the index cell C contains an object index, then let i_C denote the corresponding object index, and if the index cell C contains a measurement index, then let m_C denote the corresponding measurement index.

2) Standard dynamic model: The existing objects—both the detected and the undetected—survive from time step k to time step $k+1$ with probability of survival p_{S}. The objects evolve independently according to a Markov process with Gaussian transition density $g(\mathbf{x}_{k+1}|\mathbf{x}_k) = \mathcal{N}(\mathbf{x}_{k+1}; b(\mathbf{x}_k), Q)$, where $b(\cdot)$ is a constant velocity (CV) motion model. New objects appear independently of the objects that already exist. The object birth is assumed to be a PPP with intensity $D^b_{k+1}(\mathbf{x})$, defined in eq. (9).

C. PMBM *filter*

In this section, the time indexing has been omitted for notational simplicity. The PMBM filter is a combination of two RFSs, a PPP to model the objects that exist at the current time step, but have not yet been detected and a MBM to model the objects that have been detected previously at least once. The set of objects can be divided into two disjoint subsets, $\mathbf{X} = \mathbf{X}^d \uplus \mathbf{X}^u$, where $\mathbf{X}^d$ is the set of detected objects and $\mathbf{X}^u$ is the set of undetected objects. The PMBM density can be expressed as

$$f(\mathbf{X}) = \sum_{\mathbf{X}^u \uplus \mathbf{X}^d = \mathbf{X}} f^u(\mathbf{X}^u) \sum_{j \in \mathbb{J}} w^j f^j(\mathbf{X}^d), \tag{13a}$$

$$f^u(\mathbf{X}^u) = e^{-\langle D^u(\mathbf{x}); 1 \rangle} \left[D^u(\cdot)\right]^{\mathbf{X}^u}, \tag{13b}$$

$$f^j(\mathbf{X}^d) = \sum_{\uplus_{i \in \mathbb{I}^j} \mathbf{X}^i = \mathbf{X}^d} \prod_{i \in \mathbb{I}^j} f^{j,i}(\mathbf{X}^i), \tag{13c}$$

where

- $f^u(\cdot)$ is the PPP density for the set of undetected objects $\mathbf{X}^u$, where $D^u(\cdot)$ is its intensity.
- $\mathbb{J}$ is an index set of MBM components. There are $|\mathbb{J}|$ MBs, where each MB corresponds to a unique global data association hypothesis. The probability of each component in the MBM is denoted as w^j.
- For every component j in the MBM, there is an index set $\mathbb{I}^j$, where each index i corresponds to a potentially detected object $\mathbf{X}^i$.
- $f^{j,i}(\cdot)$ are Bernoulli set densities, defined in eq. (10). Each MB corresponds to a potentially detected object with a probability of existence and a state PDF.

The PMBM density in eq. (13) is defined by the involved parameters,

$$D^u, \{(w^j, \{(r^{j,i}, f^{j,i})\}_{i \in \mathbb{I}^j})\}_{j \in \mathbb{J}}. \tag{14}$$

Further, the PMBM density is an MOT conjugate prior [6], meaning that for the standard point object models (Sections IV-B1 and IV-B2), the prediction and update in eq. (5) both result in PMBM densities. It follows that the PMBM filter propagates the multi-object density by propagating the set of parameters.

In this work, we assume that the birth intensity D^b is a non-normalized Gaussian mixture. It follows from this assumption that the undetected intensity D^u is also a non-normalized Gaussian mixture, and all Bernoulli densities $f^{j,i}$ are Gaussian densities. Below, we present the parameters that result from the prediction and the update, and we present a simple method for extracting target estimates from the set of parameters. To compute the predicted and updated Gaussian parameters, we use the UKF prediction and update, respectively, see, e.g., [27, Ch. 5].

1) Prediction: Given a posterior PMBM density with parameters

$$D^u, \{(w^j, \{(r^{j,i}, f^{j,i})\}_{i \in \mathbb{I}^j})\}_{j \in \mathbb{J}}, \tag{15}$$

and the standard dynamic model (Section IV-B2), the predicted density is a PMBM density with parameters

$$D^u_+, \{(w^j_+, \{(r^{j,i}_+, f^{j,i}_+)\}_{i \in \mathbb{I}^j})\}_{j \in \mathbb{J}}, \tag{16a}$$

where

$$D^u_+(\mathbf{x}) = D^b(\mathbf{x}) + p_{\mathrm{S}} \left\langle D^u; g \right\rangle, \tag{16b}$$

$$r^{j,i}_+ = p_{\mathrm{S}} r^{j,i}, \tag{16c}$$

$$f^{j,i}_+(\mathbf{x}) = \left\langle f^{j,i}; g \right\rangle, \tag{16d}$$

and $w^j_+ = w^j$. For Gaussian mixture intensity D^u, and Gaussian densities $f^{j,i}$, the predictions $\langle \cdot; g \rangle$ in eq. (16) are easily computed using the UKF prediction, see, e.g., [27, Ch. 5].

2) Update: Given a prior PMBM density with parameters

$$D^u_+, \{(w^j_+, \{(r^{j,i}_+, f^{j,i}_+)\}_{i \in \mathbb{I}^j_+})\}_{j \in \mathbb{J}_+}, \tag{17}$$

a set of measurements $\mathbf{Z}$, and the standard measurement model (Section IV-B1), the updated density is a PMBM density

$$f(\mathbf{X}|\mathbf{Z}) = \sum_{\mathbf{X}^u \uplus \mathbf{X}^d = \mathbf{X}} f^u(\mathbf{X}^u) \sum_{j \in \mathbb{J}_+} \sum_{A \in \mathcal{A}^j} w^j_A f^j_A(\mathbf{X}^d), \tag{18a}$$

$$f^u(\mathbf{X}^u) = e^{-\langle D^u; 1 \rangle} \prod_{\mathbf{x} \in \mathbf{X}^u} D^u(\mathbf{x}), \tag{18b}$$

$$f^j_A(\mathbf{X}^d) = \sum_{\uplus_{C \in A} \mathbf{X}^C = \mathbf{X}} \prod_{C \in A} f^j_C(\mathbf{X}^C), \tag{18c}$$

[4]For example, let $\mathbb{M} = (m_1, m_2, m_3)$ and $\mathbb{I} = (i_1, i_2)$, i.e., three measurements and two objects. One valid partition of $\mathbb{M} \cap \mathbb{I}$, i.e., one of the possible associations, has the following four cells $\{m_1\}, \{m_2, i_1\}, \{m_3\}, \{i_2\}$. The meaning of this is that measurement m_2 is associated to object i_1, object i_2 is not detected, and measurements m_1 and m_3 are not associated to any previously detected object, i.e., measurements m_1 and m_3 are either clutter or from new objects.

where the weights are

$$w_A^j = \frac{w_+^j \prod_{C \in A} \mathcal{L}_C}{\sum_{j' \in \mathbb{J}} \sum_{A' \in \mathcal{A}^{j'}} w_+^{j'} \prod_{C' \in A'} \mathcal{L}_{C'}}, \tag{18d}$$

$$\mathcal{L}_C = \begin{cases} \kappa + p_{\mathrm{D}} \left\langle D_+^u ; \phi_{\mathbf{z}^{m_C}} \right\rangle & \text{if } C \cap \mathbb{I}^j = \emptyset, C \cap \mathbb{M} \neq \emptyset, \\ 1 - r_+^{j,i_C} p_{\mathrm{D}} & \text{if } C \cap \mathbb{I}^j \neq \emptyset, C \cap \mathbb{M} = \emptyset, \\ r_+^{j,i_C} p_{\mathrm{D}} \left\langle f_+^{j,i_C} ; \phi_{\mathbf{z}^{m_C}} \right\rangle & \text{if } C \cap \mathbb{I}^j \neq \emptyset, C \cap \mathbb{M} \neq \emptyset, \end{cases} \tag{18e}$$

the densities $f_C^j(\mathbf{X})$ are Bernoulli densities with parameters

$$r_C^j = \begin{cases} \frac{p_{\mathrm{D}} \left\langle D_+^u ; \phi_{\mathbf{z}^{m_C}} \right\rangle}{\kappa + p_{\mathrm{D}} \left\langle D_+^u ; \phi_{\mathbf{z}^{m_C}} \right\rangle} & \text{if } C \cap \mathbb{I}^j = \emptyset, C \cap \mathbb{M} \neq \emptyset, \\ \frac{r_+^{j,i_C}(1 - p_{\mathrm{D}})}{1 - r_+^{j,i_C} p_{\mathrm{D}}} & \text{if } C \cap \mathbb{I}^j \neq \emptyset, C \cap \mathbb{M} = \emptyset, \\ 1 & \text{if } C \cap \mathbb{I}^j \neq \emptyset, C \cap \mathbb{M} \neq \emptyset, \end{cases} \tag{18f}$$

$$f_C^j(\mathbf{x}) = \begin{cases} \frac{\phi_{\mathbf{z}^{m_C}}(\mathbf{x}) D_+^u(\mathbf{x})}{\left\langle D_+^u ; \phi_{\mathbf{z}^{m_C}} \right\rangle} & \text{if } C \cap \mathbb{I}^j = \emptyset, C \cap \mathbb{M} \neq \emptyset, \\ f_+^{j,i_C}(\mathbf{x}) & \text{if } C \cap \mathbb{I}^j \neq \emptyset, C \cap \mathbb{M} = \emptyset, \\ \frac{\phi_{\mathbf{z}^{m_C}}(\mathbf{x}) f_+^{j,i_C}(\mathbf{x})}{\left\langle f_+^{j,i_C} ; \phi_{\mathbf{z}^{m_C}} \right\rangle} & \text{if } C \cap \mathbb{I}^j \neq \emptyset, C \cap \mathbb{M} \neq \emptyset, \end{cases} \tag{18g}$$

and the updated PPP intensity is

$$D^u(\mathbf{x}) = (1 - p_{\mathrm{D}}) D_+^u(\mathbf{x}). \tag{19}$$

For Gaussian mixture intensity D_+^u, and Gaussian densities $f_+^{j,i}$, the updates $\langle \cdot ; \phi \rangle$ in (18) are easily computed using the UKF update, see, e.g., [27, Ch. 5].

3) Extraction: Let the set of updated PMBM parameters be

$$D^u, \{(w^j, \{(r^{j,i}, f^{j,i})\}_{i \in \mathbb{I}^j})\}_{j \in \mathbb{J}}. \tag{20}$$

To extract a set of object estimates, the hypothesis with highest probability is chosen,

$$j^\star = \arg\max_{j \in \mathbb{J}} w^j. \tag{21}$$

From the corresponding MB, with parameters

$$\{(r^{j^\star,i}, f^{j^\star,i})\}_{i \in \mathbb{I}^{j^\star}}, \tag{22}$$

all Bernoulli components with probability of existence $r^{j^\star,i}$ larger than a threshold τ are selected, and the expected value of the object state is included in the set of object estimates,

$$\hat{\mathbf{X}} = \left\{ \hat{\mathbf{x}}^{j^\star,i} \right\}_{i \in \mathbb{I}^{j^\star} : r^{j^\star,i} > \tau}, \tag{23a}$$

$$\hat{\mathbf{x}}^{j^\star,i} = E_{f^{j^\star,i}} \left[\mathbf{x}^{j^\star,i} \right] = \int \mathbf{x} f^{j^\star,i}(\mathbf{x}) d\mathbf{x}. \tag{23b}$$

V. Experimental results

A. Setup

For evaluation, the KITTI object tracking dataset [20] is used. The datasets consists of 21 training sequences and 29 testing sequences that were collected using sensors mounted on a moving car. Each sequence has been manually annotated with ground truth information, e.g., in the images, objects from the classes *Car*, *Pedestrian* and *Cyclist* have been marked by bounding boxes. In this work, the training dataset was split into two parts; one for training the CNN, and one for validation. The sequences used for training are 0, 2, 3, 4, 5, 7, 9, 11, 17 and 20, and the remaining ones are used for validation.

B. Evaluation

In this work we are primarily interested in the 3D tracking results. However, the KITTI testing sequences evaluate the tracking in 2D, hence we present results in both 2D and 3D. Performance is evaluated using IOU of the image plane bounding boxes and Euclidean distance as distance measurements, respectively. For a valid correspondence between a ground truth (GT) object and an estimated object, the 2D IOU has to be at least $50\,\%$, and the 3D Euclidean distance has to be within $3\,\mathrm{m}$, for the 2D and 3D evaluation, respectively. The performance is evaluated using the CLEAR MOT performance measures [28], including MOT accuracy (MOTA), MOT precision (MOTP), with addition of mostly tracked (MT), mostly lost (ML), identity switches (IDS) and fragmentations (FR) from [29], and F1 score (F1), precision (PRE), recall (REC) and false alarm rate (FAR). Note that, for the 2D IOU measure, a larger value is better, whereas for the 3D Euclidean distance, lower is better.

C. Results

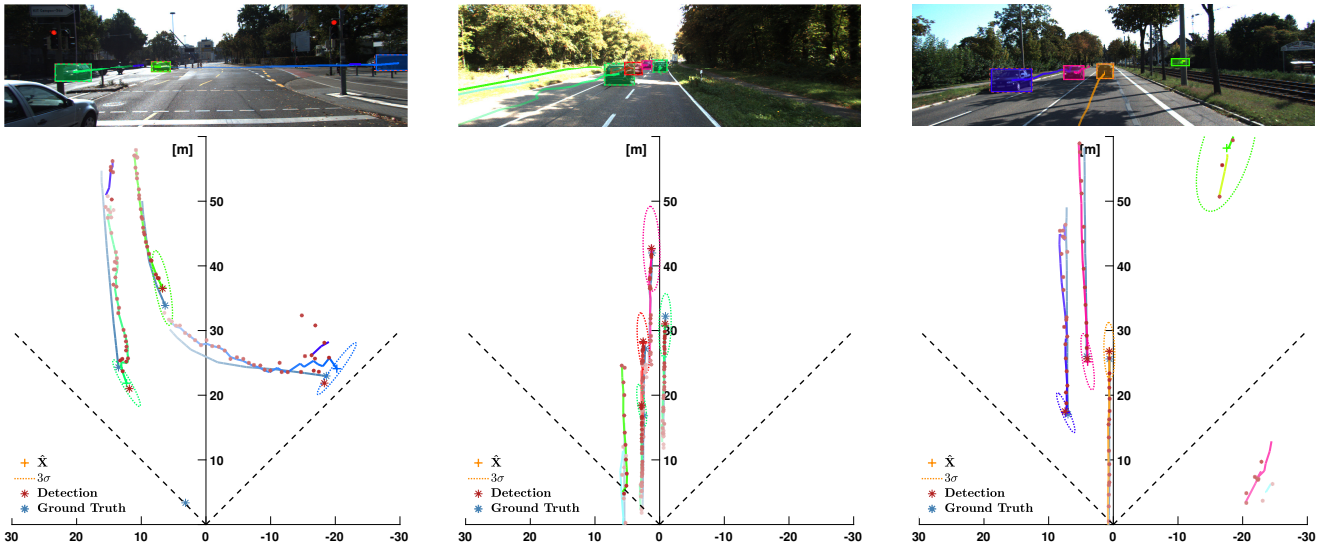
Examples of the 3D tracking results are shown in fig. 2. The three examples show that the tracking algorithm successfully estimates the states of vehicles moving in the same direction as the ego-vehicle, vehicles moving in the opposite direction, as well as vehicles making sharp turns in intersections. In dense scenarios, such as in fig. 2b, there are big overlaps between the bounding boxes; this is handled without problem by the data association. Noteworthy is that the distance estimates are quite noisy. Sometimes this leads to incorrect initial estimates of the velocity vector, as can be seen at the beginning of the track of the oncoming vehicle labelled purple in fig. 2c. However, the tracking filter quickly converges to a correct estimate. Videos of these, and of additional sequences, can be seen at https://goo.gl/AoydgW.

Quantitative results from the evaluation on the validation sequences are shown in table II. Noteworthy is the low amount of identity switches, in both 2D and in 3D. Comparing the raw CNN detections and the MOT algorithm, the MOT precision is lower, and the MOT recall is higher, leading to an F1 score that is higher for the MOT than for the CNN; in other words, the overall object detection performance is slightly improved.

The runtime of the algorithm is on average in total $52\,\mathrm{ms}$, $38\,\mathrm{ms}$ for the detection network and $14\,\mathrm{ms}$ for the tracking algorithm, on a Nvidia Tesla V100 SXM2 and a single thread on a $2.7\,\mathrm{GHz}$ Intel Core i7.

D. KITTI MOT benchmark

The MOT algorithm was also evaluated in 2D using the test sequences on the KITTI evaluation server. For these results, the full training set was used for training the detection CNN. The results are presented in table III; at the time of submission our algorithm was ranked 4th in terms of MOTA among the published algorithms. Note that, even if not reaching the same MOTA performance, the runtime of our algorithm is one magnitude faster and has a significantly lower number of identity switches than the two algorithms with higher MOTA.



(a) (b) (c)

Fig. 2. Object estimates shown both projected to the image plane (top) and in top-view (bottom). The dotted tracks illustrate the GT and tracked object estimates at previous time steps. The color of the tracks corresponds to the identity of the tracked object.

TABLE II
RESULTS OF THE EVALUATION OF TRACKING PERFORMANCE IN 2D AND 3D ON THE *Car* CLASS. ↑ AND ↓ REPRESENTS THAT HIGH VALUES AND LOW VALUES ARE BETTER, RESPECTIVELY. THE BEST VALUES ARE MARKED WITH BOLD FONT.

	Method	MOTA↑	MOTP↑	MT↑	ML↓	IDS↓	FR↓	F1↑	PRE↑	REC↑	FAR↓
2D	CNN	–	**82.04 %**	74.59 %	3.78 %	–	–	91.16 %	**95.72 %**	87.02 %	**9.08 %**
	MOT	**81.23 %**	81.63 %	**76.22 %**	3.78 %	19	107	**91.26 %**	94.76 %	**88.02 %**	11.46 %
	Method	MOTA↑	MOTP↓	MT↑	ML↓	IDS↓	FR↓	F1↑	PRE↑	REC↑	FAR↓
3D	CNN	–	111.39 cm	45.95 %	**10.27 %**	–	–	73.53 %	**78.74 %**	68.97 %	**41.90 %**
	MOT	**47.20 %**	**110.73 cm**	**48.65 %**	11.35 %	20	166	**73.86 %**	78.18 %	**70.00 %**	44.32 %

TABLE III
KITTI MOT BENCHMARK [20] RESULTS FOR *Car* CLASS. ↑ AND ↓ REPRESENTS THAT HIGH VALUES AND LOW VALUES ARE BETTER, RESPECTIVELY. THE BEST VALUES ARE MARKED WITH BOLD FONT. [17] IS TUSIMPLE, [30] IS RRC-IIITH, [19] IS IMMDP AND [31] IS MCMOT-CPD. ONLY RESULTS OF PUBLISHED METHODS ARE REPORTED.

	MOTA↑	MOTP↑	MT↑	ML↓	IDS↓	FR↓	FPS↑[a]
[17]	**86.6 %**	84.0 %	72.5 %	6.8 %	293	501	1.7
[30]	84.2 %	**85.7 %**	**73.2 %**	**2.8 %**	468	944	3.3
[19]	83.0 %	82.7 %	60.6 %	11.4 %	172	**365**	5.3
Our	80.4 %	81.3 %	62.8 %	6.2 %	**121**	613	73
[31]	78.9 %	82.1 %	52.3 %	11.7 %	228	536	**100**

[a] The time for object detection is not included in the specified runtime.

VI. CONCLUSION

This paper presented an image based MOT algorithm using deep learning detections and PMBM filtering. It was shown that a CNN and a subsequent PMBM filter can be used to detect and track objects. The algorithm successfully can track multiple objects in 3D from a single camera image, which can provide valuable information for decision making and control.

ACKNOWLEDGMENT

This work was partially supported by the Wallenberg Autonomous Systems and Software Program (WASP).

REFERENCES

[1] R. Mahler, *Statistical Multisource-Multitarget Information Fusion*. Norwood, MA, USA: Artech House, Inc., 2007.

[2] ——, "Multitarget Bayes filtering via first-order multitarget moments," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1152–1178, October 2003.

[3] ——, "PHD filters of higher order in target number," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 43, no. 4, pp. 1523–1543, October 2007.

[4] B. T. Vo and B. N. Vo, "Labeled Random Finite Sets and Multi-Object Conjugate Priors," *IEEE Transactions on Signal Processing*, vol. 61, no. 13, pp. 3460–3475, July 2013.

[5] S. Reuter, B. T. Vo, B. N. Vo, and K. Dietmayer, "The Labeled Multi-Bernoulli Filter," *IEEE Transactions on Signal Processing*, vol. 62, no. 12, pp. 3246–3260, June 2014.

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

[7] A. F. García-Fernández, J. Williams, K. Granström, and L. Svensson, "Poisson multi-bernoulli mixture filter: direct derivation and implementation," *IEEE Transactions on Aerospace and Electronic Systems*, 2018.

[8] Y. Xia, K. Granström, L. Svensson, and A. F. G. Fernández, "Performance Evaluation of Multi-Bernoulli Conjugate Priors for Multi-Target Filtering," in *2017 20th International Conference on Information Fusion (Fusion)*, July 2017, pp. 1–8.

[9] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, June 2017.

[10] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 779–788.

[11] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2323, 1998.

[12] R. Girshick, "Fast R-CNN," in *2015 IEEE International Conference on Computer Vision (ICCV)*, December 2015, pp. 1440–1448.

[13] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, and R. Urtasun, "Monocular 3D Object Detection for Autonomous Driving," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2147–2156, 2016.

[14] A. Mousavian, D. Anguelov, J. Košecká, and J. Flynn, "3D bounding box estimation using deep learning and geometry," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, pp. 5632–5640.

[15] F. Chabot, M. Chaouch, J. Rabarisoa, C. Teulière, and T. Chateau, "Deep MANTA: A coarse-to-fine many-task network for joint 2D and 3D vehicle analysis from monocular image," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, pp. 1827–1836.

[16] A. Z. Richard Hartley, *Multiple View Geometry*, 2nd ed. New York, NY, USA: Cambridge University Press, 2004.

[17] W. Choi, "Near-Online Multi-target Tracking with Aggregated Local Flow Descriptor," in *2015 IEEE International Conference on Computer Vision (ICCV)*, December 2015, pp. 3029–3037.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 770–778.

[19] Y. Xiang, A. Alahi, and S. Savarese, "Learning to Track: Online Multi-object Tracking by Decision Making," in *2015 IEEE International Conference on Computer Vision (ICCV)*, Dec 2015, pp. 4705–4713.

[20] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2012, pp. 3354–3361.

[21] Y. LeCun, Y. Bengio, G. Hinton, L. Y., B. Y., and H. G., "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[22] A. Krishnan and J. Larsson, "Vehicle detection and road scene segmentation using deep learning," 2016.

[23] F. Yu, V. Koltun, and T. Funkhouser, "Dilated Residual Networks," may 2017. [Online]. Available: http://arxiv.org/abs/1705.09914

[24] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 1874–1883.

[25] N. Bodla, B. Singh, R. Chellappa, and L. S. Davis, "Soft-NMS - Improving Object Detection with One Line of Code," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017, pp. 5562–5570.

[26] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2009, pp. 248–255.

[27] S. Särkkä, *Bayesian Filtering and Smoothing*. Cambridge University Press, 2013.

[28] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: The CLEAR MOT metrics," *Eurasip Journal on Image and Video Processing*, vol. 2008, 2008.

[29] Y. Li, C. Huang, and R. Nevatia, "Learning to associate: HybridBoosted multi-target tracker for crowded scene," in *2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2009, pp. 2953–2960.

[30] S. Sharma, J. A. Ansari, J. K. Murthy, and K. M. Krishna, "Beyond Pixels: Leveraging Geometry and Shape Cues for Online Multi-Object Tracking," 2018. [Online]. Available: http://arxiv.org/abs/1802.09298

[31] B. Lee, E. Erdenee, S. Jin, M. Y. Nam, Y. G. Jung, and P. K. Rhee, "Multi-class multi-object tracking using changing point detection," in *Computer Vision – ECCV 2016 Workshops*, G. Hua and H. Jégou, Eds. Cham: Springer International Publishing, 2016, pp. 68–83.