sensors

MDPI

Article

A Two-Stage Data Association Approach for 3D Multi-Object Tracking

Minh-Quan Dao and Vincent Frémont *

LS2N, CNRS, École Centrale de Nantes, 1 Rue de la Noë, 44321 Nantes, France; minh-quan.dao@ec-nantes.fr
* Correspondence: vincent.fremont@ec-nantes.fr; Tel.: +33-255-589-186

Abstract: Multi-Object Tracking (MOT) is an integral part of any autonomous driving pipelines because it produces trajectories of other moving objects in the scene and predicts their future motion. Thanks to the recent advances in 3D object detection enabled by deep learning, track-by-detection has become the dominant paradigm in 3D MOT. In this paradigm, a MOT system is essentially made of an object detector and a data association algorithm which establishes track-to-detection correspondence. While 3D object detection has been actively researched, association algorithms for 3D MOT has settled at bipartite matching formulated as a Linear Assignment Problem (LAP) and solved by the Hungarian algorithm. In this paper, we adapt a two-stage data association method which was successfully applied to image-based tracking to the 3D setting, thus providing an alternative for data association for 3D MOT. Our method outperforms the baseline using one-stage bipartite matching for data association by achieving 0.587 Average Multi-Object Tracking Accuracy (AMOTA) in NuScenes validation set and 0.365 AMOTA (at level 2) in Waymo test set.

Keywords: multi-object tracking; data association; autonomous vehicles

Citation: Dao, M-Q.; Frémont, V. A Two-Stage Data Association Approach for 3D Multi-Object Tracking. *Sensors* **2021**, *21*, 2894. https://doi.org/10.3390/s21092894

Academic Editor: Steven Waslander

Received: 21 March 2021
Accepted: 16 April 2021
Published: 21 April 2021

1. Introduction

Multi-object tracking have been a long standing problem in computer vision and robotics community since it is a crucial part of any autonomous systems. From the early work of tracking with hand-craft features, the revolution of deep learning which results in highly accurate object detection models [1–3] has shifted the focus of the field to the track-by-detection paradigm [4,5]. In the framework of this paradigm, tracking algorithms receive a set of object detection, usually in the form of bounding boxes, at each time step and they aim to link detection of the same object across time to form trajectories.

While image-based methods of this paradigm have reached a certain maturity, 3D tracking is still in its early phase where most of the published approaches are originated from successful 2D exemplars. One popular method is [6] which extends [4] into 3D space. In these works, detections are linked to tracks by solving a bipartite matching with the Hungarian algorithm [7], then states of tracks are updated by a Kalman filter. Taking a similar approach to establishing detection-to-track correspondence, [8] trains a network for calculating the matching cost instead of using the 3D Intersection over Union (IoU). In [9,10], objects' poses in the current and several future frames are predicted by deep neural networks. Thus, tracks can be formed by greedy closest-point matching.

Even though 3D tracking has been progressed rapidly thanks to the availability of standardized large scale benchmarks such as KITTI [11], NuScenes [12], Waymo Open Dataset [13], the focus of the field is placed on developing better object detection models rather than developing better tracking algorithm as shown in Table 1 which presents the performance measured by the AMOTA metric of tracking algorithms following the track-by-detection paradigm and the performance of their object detector measured by mean Average Precision (mAP). AMOTA is a scalar value representing how well the algorithm does in limiting:

- ID switches (IDS): the number of times tracks are associated with wrong detections;
- False Positives (FP): the number of times real objects are missed detected;
- False Negatives (FN): the number of times the tracking algorithm reports tracks in places where there are no real objects present.

There are two trends that can be observed in this table. First, tracking performance experiences a boost when a better object detection model is introduced. Second, the method of AB3DMOT [6] , which uses the Hungarian algorithm on some metrics (e.g., 3D IoU, Mahalanobis distance) to perform data association, Kalman Filters to update tracks' states once they have associated detections, and set of heuristic rules to manage birth and death of tracks, is favored by most recent 3D tracking systems.

Table 1. Summary of tracking methods which details are published in the leader board of NuScenes and Waymo Open Dataset.

Dataset	Method Name	Tracking Method	AMOTA	Object Detector	mAP
NuScenes	CenterPoint [9]	Greedy closest-point matching	0.650	CenterPoint	0.603
	PMBM	Poisson Multi-Bernoulli Mixture filter [14]	0.626	CenterPoint	0.603
	StanfordIPRL-TRI [15]	Hungarian algorithm with Mahalanobis distance as cost function and Kalman Filter	0.550	MEGVII [16]	0.519
	AB3DMOT [6]	Hungarian algorithm with 3D IoU as cost function and Kalman Filter	0.151	MEGVII	0.519
	CenterTrack	Greedy closest-point mathcing	0.108	CenterNet [17]	0.388
Waymo	HorizonMOT [18]	3-stage data associate, each stage is an assignment problem solved by Hungarian algorithm	0.6345	AFDet [19]	0.7711
	CenterPoint	Greedy closest-point matching	0.5867	CenterPoint	0.7193
	PV-RCNN-KF	Hungarian algorithm and Kalman Filter	0.5553	PV-RCNN [20]	0.7152
	PPBA AB3DMOT	Hungarian algorithm with 3D IoU as cost function and Kalman Filter	0.2914	PointPillars and PPBA [21]	0.3530

The reason for AB3DMOT's popularity is that despite its simplicity, it achieves competitive result in challenging datasets at significantly high frame rate (more than 200 FPS). However, such simplicity comes at the cost of the MOT system being vulnerable to false associations due to occlusion or imperfect detections which is case for objects in a clutter or far away from the ego vehicle.

Aware of the shortage of a generic 3D tracking algorithm which can better handle the occlusion and imperfect detections, yet remains relatively simple, we adapt the image-based tracking method proposed by [22] to the 3D setting. Specifically, this method is a two-stage data association scheme. In this scheme, each tracked trajectory is called a tracklet and is assigned a confidence score computed based on how well associated detection matches with tracklet. The first association stage aims to establish the correspondence between high-confident tracklets and detection. The second stage matches the left over detection with the low confident tracklets as well as link low-confident tracklets to high-confident ones if they meet a certain criterion.

In this paper, we make two contributions

- Our main contribution is the adaptation of an image-based tracking method to the 3D setting. In details, we exploit a kinematically feasible motion model, which is unavailable in 2D, to facilitate the prediction of objects' poses. This motion model defines the minimal state vector needed to be tracked.

- Extensive experiment carried out in various datasets proves the effectiveness of our approach. In fact, our better performance, compared to AB3DMOT-style models, show that adding a certain degree of re-identification can improve the tracking performance while keeping the added complexity to the minimum.
- Our implementation is available at https://github.com/quan-dao/track_with_confidence accessed on 21 April 2021.

2. Related Work

A multi-object tracking system in the track-by-detection paradigm consists of an object detection model, a data association algorithm and a filtering method. While the last two components are domain agnostic, object detection models, especially learning-based methods, are tailored to their operation domain (e.g images or point clouds). This paper targets autonomous driving where objects' poses are required thus being interested in 3D object detection models. However, developing such a model is not in the scope of this paper, instead we use the detection result provided by baseline models of benchmarks (e.g., PointPillars of NuScenes) to focus on the data association algorithm and to have a fair comparison. Interested readers are referred to [23] for a review of 3D object detection.

Data association via the Hungarian algorithm was early explored in [24] where a 2-stage tracking scheme was proposed for offline 2D tracking. Firstly, detections are linked frame-by-frame to form tracklets by associate detections to tracklets via solving a LAP with the Hungarian algorithm. The cost matrix of this LAP is computed based on geometric and appearance cue. While the geometric cue is the 2D IoU, the appearance cue is the correlation between two bounding boxes. Secondly, tracklets are associated with each other to compensate trajectory fragments and ID switches due to occlusion. Similar to the previous step, this association is also formulated as a LAP and solved by the Hungarian algorithm.

Due to its batch-processing nature, the method of [24] cannot be applied to online tracking. The authors of [4] overcomes this by eliminating the second stage, which effectively sacrifices the ability of re-identifies objects after a period of occlusion. Despite its simplicity, SORT — the method proposed by [4]–achieves competitive result in MOT15 [25] with lightning-fast inference speed (260 Hz). The success of SORT inspired [6] to adapt it to 3D setting by using 3D IoU as the affinity function. The performance of SORT in 3D setting is later improved in [15] showing the superiority over 3D IoU of the Mahalanobis distance which is the magnitude of difference between the expected detection given the ego vehicle pose and the real detection while taking into account their uncertainty. In [26], the authors integrate the 3D version of SORT into a complete perception pipeline for autonomous vehicles.

The two-stage association scheme is adapted to online tracking in [22] which proposes a confidence score to quantify tracklets quality. Based on this score, tracklets are associated with detections or another tracklets, or terminated. The appearance model learned by ILDA in [22] is improved by deep learning in the follow-up work [27]. Recently, this association scheme is revisited in the context of image-based pedestrian tracking by [28] which proposed to use the rank of the Hankel matrix as tracklets motion affinity. To be specific, this technique estimates a tracklet 's dynamic by a linear regressor taking its previous states as input. In noise-free scenarios, the order of such a regressor (i.e., the number of past states needed to estimate the current state) is equal to the rank of the Hankel matrix which formula can be found in [28]. The intuition behind this technique is that if two tracklets belong the same trajectory, explaining their merged trajectory would require a low order regressor. This technique is popular in image-based tracking despite being prone to deterioration due to noise because of the absence of an accurate motion model in this space. However, objects' motion in 3D can be well explained by their kinematic models. Therefore, our approach employs two different kinematic models for two different categories of objects to have more computationally efficient and accurate motion affinity.

Differently from [22] and its related works, this paper applies the two-stage association scheme to online 3D tracking. In addition, we can provide competitive result despite relying solely on geometric cue to compute tracklet affinity by exploiting the Constant Turning Rate and Velocity (CTRV) motion model which can accurately predict objects position in 3D space by exploiting their kinematic.

3. Method

3.1. Problem Formulation

Online MOT in the sense of track-by-detection aims to gradually grow the set of tracklets $\mathbb{T}_{0:t} = \{\mathcal{T}^i\}_{i=1}^{|\mathbb{T}_{0:t}|}$ by establishing correspondences with the set of detections received at every time step $\mathbb{D}_t = \{d_t^j\}_{j=1}^{|\mathbb{D}|}$ and updating tracklets state accordingly. A detection d_t^j at time step t encapsulates information of an object as a 3D bounding box

$$d_t^j = \begin{bmatrix} x & y & z & \theta & w & l & h \end{bmatrix}^T, \tag{1}$$

here, $[x, y, z]$ is the position of the box's center, θ is its heading direction, and $[w, l, h]$ is its size. It is worth noticing that in the context of autonomous driving, objects are assumed to remain in contact with the ground; therefore, their detections are up-right bounding boxes which orientation is described by a single number — the heading angle. A tracklet is a collection of state vectors corresponding to the same object $\mathcal{T}^i = \{\mathrm{x}_k^i | 0 \leq t_s^i \leq k \leq t_e^i \leq t\}$, here t_s^i, t_e^i are respectively the starting- and ending-time of the tracklet.

The correspondence between $\mathbb{T}_{0:t}$ and $\mathbb{D}_t$ can be formally defined as finding the set $\mathbb{T}_{0:t}^*$ that maximizes its likelihood given $\mathbb{D}_t$.

$$\mathbb{T}_{0:t}^* = \underset{\mathbb{T}_{0:t}}{\mathsf{argmax}}\, \mathrm{p}(\mathbb{T}_{0:t} | \mathbb{D}_t) \tag{2}$$

Due the exponential growth of possible associations between $\mathbb{T}_{0:t}$ and $\mathbb{D}_t$, Equation (2) is computationally intractable after a few time steps. In this paper, such a correspondence is approximated by the two-stage data association proposed by [22] as shown in the following.

3.2. Two-Stage Data Association

3.2.1. Tracklet Confidence Score

The reliability of a tracklet is quantified by a confidence score which is calculated based on how well associated detections match with its states across its life span and how long its corresponding object was undetected.

$$\mathsf{conf}\left(\mathcal{T}^i\right) = \left(\frac{1}{L^i} \sum_{k \in [t_s^i, t_e^i] | v^i(k) = 1} \Lambda\left(\mathcal{T}^i, d_k^j\right) \right) \times \exp\left(-\beta \frac{W}{L_i}\right) \tag{3}$$

where $v^i(k)$ is a binary indicator which takes 1 if the tracklet has a detection d_k^j associated with it at time step k, and 0 otherwise. L^i is the number of time step that the traklet gets associated with a detection. $\Lambda(\cdot)$ is the affinity function which evaluates the similarity between a track and a detection. Its detail will be presented in the following subsection. β is a tuning parameter which takes high value if the object detection model is accurate. $W = t - t_s^i - L^i + 1$ is the number of time step that tracklet was undetected (i.e., did not have associated detection) calculated from its birth to the current time step t.

Applying a threshold τ^c this confidence score divides the set $\mathbb{T}_{0:t}$ into a subset of high-confident tracklets $\mathbb{T}_{0:t}^h = \{\mathcal{T}^{i,h} | \mathsf{conf}\left(\mathcal{T}^i\right) > \tau^c\}$ and a subset of low-confident tracklets $\mathbb{T}_{0:t}^l = \{\mathcal{T}^{i,l} | \mathsf{conf}\left(\mathcal{T}^i\right) \leq \tau^c\}$. These two subsets are the fundamental elements of the two-stage association pipeline showed in Figure 1

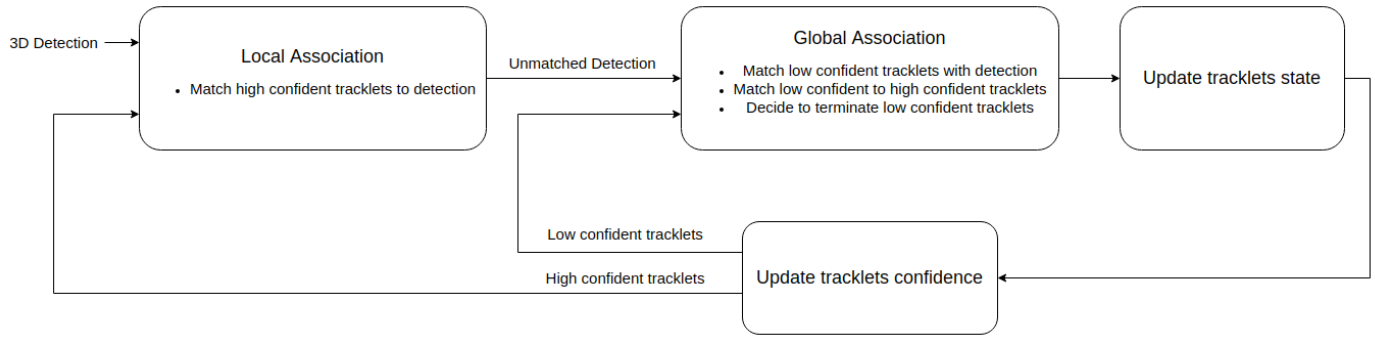


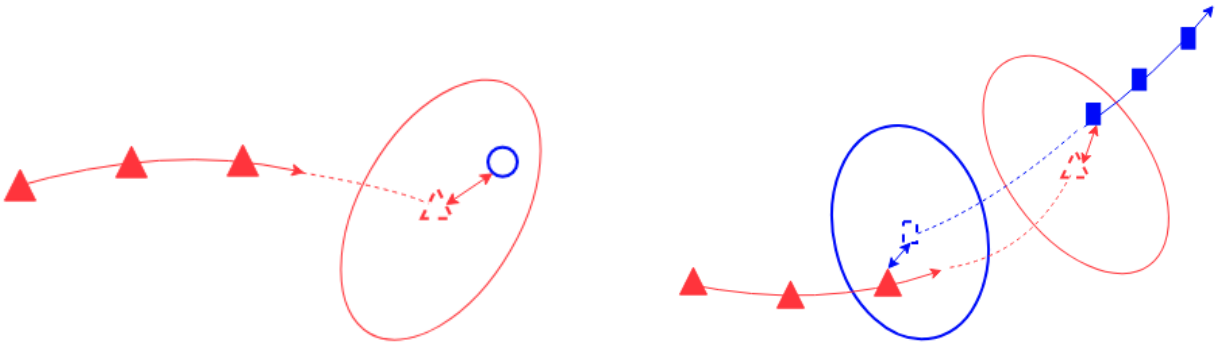
Figure 1. The pipeline of two-stage data association. The first stage — local association establish the correspondences between detections at this time step $\mathbb{D}_t$ and high-confident tracklets $\mathbb{T}^h_{0:t}$. Then, global association stage matches each low-confident tracklets $\mathcal{T}^{i,l}$ with either a high-confident tracklet or a left-over detection, or terminates it.

3.2.2. Affinity Function

Affinity function $\Lambda(\cdot)$ is to compute how similar a detection to a tracklet or a tracklet to another. As mentioned earlier, due to the lack of colorful texture in point cloud, the affinity function used in this work is just comprised of geometric cue. Specifically, it is the sum of position affinity and size affinity.

$$\Lambda(\mathcal{T}^i, d_t^j) = \Lambda(\mathcal{T}^i, d_t^j)^p + \Lambda(\mathcal{T}^i, d_t^j)^s \tag{4}$$

The scheme for computing position affinity between a tracklet and a detection or between two tracklets are shown in Figure 2



(**a**) tracklet-to-detection position affinity (**b**) tracklet-to-tracklet position affinity

Figure 2. The computational scheme of position affinity. The filled triangles (or rectangles) are subsequent states of a tracklet. The colored arrow represents the time order: the closer to the tip, the more recent the state. The triangle (or rectangle) in dash line is the state propagated forward (or backward) in time. The covariance of these propagated states are denoted by ellipses with the same color. The two-headed arrows indicate the Mahalanobis distance. In the subfigure (**a**), the blue circle denotes a detection.

As shown in Figure 2a, the position affinity $\Lambda(\mathcal{T}^i, d_t^j)^p$ between a tracklet $\mathcal{T}^i$ and a detection d_t^j is defined as the Mahalanobis distance between tracklet's last state propagated to the current time step t and the measurement vector z_t^j extracted from d_t^j

$$\Lambda(\mathcal{T}^i, d_t^j)^p = \left(\mathrm{h}\left(\bar{\mathrm{x}}_e^i\right) - \mathrm{z}_t^j\right)^T \cdot \mathbf{S}^{-1} \cdot \left(\mathrm{h}\left(\bar{\mathrm{x}}_e^i\right) - \mathrm{z}_t^j\right) \tag{5}$$

where $\bar{\mathrm{x}}_e^i$ is last state of tracklet $\mathcal{T}^i$ propagated to the current time step using the motion model which will be presented below. $\mathrm{h}(\cdot)$ is the measurement model computing the

expected measurement using the inputted state and the measurement vector z_t^j extracted from d_t^j

$$z_t^j = [x, y, z, \theta]^T \tag{6}$$

The matrix **S** is the covariance matrix of the innovation (i.e., the difference between expected measurement $h(\bar{x}_e^i)$ and its true value z_t^j)

$$\mathbf{S} = \mathbf{H} \cdot \mathbf{P} \cdot \mathbf{H}^T + \mathbf{R} \tag{7}$$

here, $\mathbf{H} = \delta h / \delta x$ is the Jacobian of the measurement model. $\mathbf{P}, \mathbf{R}$ are covariance matrix of $\bar{x}_e^i$ and z_t^j, respectively. These covariance matrices are calculated based on training data using the approach proposed by [15].

In the case of two tracklets $\mathcal{T}^i$ and $\mathcal{T}^j$, assuming $\mathcal{T}^j$ starts after $\mathcal{T}^i$ ended, their motion affinity is, according to Figure 2b, is the sum of

- Mahalanobis distance between the last state of $\mathcal{T}^i$ propagated forward in time and the first state of $\mathcal{T}^j$
- Mahalanobis distance between the first state of $\mathcal{T}^j$ propagated backward in time and the last state of $\mathcal{T}^i$

$$\Lambda(\mathcal{T}^i, \mathcal{T}^j)^p = \Lambda(\mathcal{T}^j, \bar{x}_e^i)^p + \Lambda(\mathcal{T}^i, \bar{x}_s^j)^p \tag{8}$$

here, $\bar{x}_e^i$ is the last state of tracklet $\mathcal{T}^i$ propagated forward in time to the first time step of tracklet $\mathcal{T}^j$, while $\bar{x}_s^j$ is the first state of tracklet $\mathcal{T}^j$ propagated backward in time to the last time step of tracklet $\mathcal{T}^i$. The size affinity $\Lambda(\mathcal{T}^i, d_t^j)^s$ is computed as follows:

$$\Lambda\left(\mathcal{T}^i, d_t^j\right)^s = -\frac{|w_e^i - w_t^j|}{w_e^i + w_t^j} \cdot \frac{|l_e^i - l_t^j|}{l_e^i + l_t^j} \cdot \frac{|h_e^i - h_t^j|}{h_e^i + h_t^j} \tag{9}$$

here, $[w_e^i, l_e^i, h_e^i]$ are the size of the last state of tracklet $\mathcal{T}^i$, while $[w_t^j, l_t^j, h_t^j]$ are the size of the detection d_t^j. In the case of two tracklets $\mathcal{T}^i$ and $\mathcal{T}^j$, assuming $\mathcal{T}^j$ starts after $\mathcal{T}^i$ ended, their size affinity is

$$\Lambda\left(\mathcal{T}^i, \mathcal{T}^j\right)^s = -\frac{|w_e^i - w_s^j|}{w_e^i + w_s^j} \cdot \frac{|l_e^i - l_s^j|}{l_e^i + l_s^j} \cdot \frac{|h_e^i - h_s^j|}{h_e^i + h_s^j} \tag{10}$$

The subscript e, s in Equation (10) respectively denote the ending and starting state of a tracklet.

To reduce the risk of false association, a threshold is applied to the affinity

$$\Lambda(\mathcal{T}^i, d_t^j) = \begin{cases} \Lambda(\mathcal{T}^i, d_t^j), & \text{if } \Lambda(\mathcal{T}^i, d_t^j) < \sigma \\ \infty, & \text{otherwise} \end{cases} \tag{11}$$

3.2.3. Local Association

In this association stage, high-confident tracklets ($\mathbb{T}_{0:t}^h$) are extended by their correspondence in the set of detections $\mathbb{D}_t$. This tracklet-to-detection is found by solving the linear assignment problem characterized by the cost matrix **L** as follows:

$$\mathbf{L} = [l_{i,j}] \in \mathbb{R}^{h \times d}, \text{ with } l_{i,j} = -\Lambda\left(\mathcal{T}^{i,h}, d_t^j\right), \mathcal{T}^{i,h} \in \mathbb{T}_{0:t}^h \tag{12}$$

where h, d are respectively the number of high-confident tracklets and the number of detections. The intuition of this association stage is that because tracklets with high-confident have been tracked accurately for several time steps, the affinity function can identify if a detection is belong to the same object as the tracklet with high accuracy, thus

limiting the possibility of false correspondences. In addition, low-confident tracklets are usually resulted from fragment trajectories or noisy detections, excluding them from this association stage help reduces the ambiguity.

3.2.4. Global Association

As shown in Figure 1, the global association stage carries out the following tasks

- Matching low-confident tracklets with high-confident ones
- Matching low-confident tracklets with detections left over by the local association stage
- Deciding to terminate low-confident tracklets

These tasks are simultaneously solved as a LAP formulated by the following cost matrix

$$\mathbf{G}_{(l+d')\times(h+l)} = \begin{bmatrix} \mathbf{A}_{l\times h} & B_{l\times l} \\ \infty_{d'\times h} & C_{d'\times l} \end{bmatrix} \tag{13}$$

here, l, d are respectively the number low-confident tracklets and detections left over by the local association stage. $\infty_{d'\times h}$ is the matrix of size $d' \times h$ with every element is set to ∞. Recall h is the number of high-confident tracklets. Submatrix $\mathbf{A}$ is the cost matrix of the event where low-confident tracklets are matched with high-confident ones

$$\mathbf{A} = [a_{i,j}] \in \mathbb{R}^{l\times h}, \text{ with } a_{i,j} = -\Lambda\left(T^{i,l}, T^{j,h}\right) \tag{14}$$

Submatrix $\mathbf{B}$ represents the event where low-confident tracklets are terminated.

$$\mathbf{B} = [b_{i,j}] \in \mathbb{R}^{l\times l}, \text{ with } b_{i,j} = \begin{cases} -\log\left(1 - \text{conf}\left(\mathcal{T}^i\right)\right), \text{ if } i = j \\ \infty, \text{ otherwise} \end{cases} \tag{15}$$

Finally, submatrix $\mathbf{C}$ is the cost of the associating low-confident tracklets with detections left over by local association stage.

$$\mathbf{C} = [c_{i,j}] \in \mathbb{R}^{d'\times l}, \text{ with } c_{i,j} = -\Lambda\left(\mathcal{T}^j, d_t^i\right) \tag{16}$$

The solution to the LAP in this stage and in the Local Association stage is the association that *minimize* the cost and can be either found by the Hungarian algorithm for the optimal solution or by a greedy algorithm which iteratively picks and removes correspondence pair with the smallest cost until there is no pair has cost less than a threshold. The detail of this greedy algorithm can be found in [15] or in Section 3.4.

Once a detection is associated with a tracklet, its position and heading is used to update the tracklet's state according to the equation of the Kalman Filter, while its sizes is averaged with tracklet's sizes in the past few time steps to result in updated sizes. Detections do not get associated in the global association stage are used to initialize new tracklets.

3.3. Motion Model and State Vector

Exploiting the fact that objects are tracked in 3D space of a common static reference frame which can be referred to as the world frame, motion of objects can be described by more kinematically accurate models, compared to the commonly used Constant Velocity (CV) model. In this work, we use the Constant Turning Rate and Velocity (CTRV) model to predict motion of car-like vehicles (e.g., cars, buses, trucks), while keeping the CV model for pedestrians.

For car-like vehicles, its state can be described by

$$\mathrm{x} = [x, y, z, \theta, \mathrm{v}, \dot{\theta}, \dot{z}]^T \tag{17}$$

here, $[x, y, z]$ is the location in the world frame of the center of the bounding box represented by the state vector, θ is the heading angle, v is longitudal velocity (i.e., velocity along the heading direction), $\dot{\theta}, \dot{z}$ are respectively velocity of θ and z.

The motion on x-y plane of car-like vehicles can be predicted using CTRV as follows:

$$\mathrm{x}_{t+1} = \mathrm{x}_t + \begin{bmatrix} \frac{\mathrm{v}}{\dot{\theta}}\left(\sin(\theta + \dot{\theta}\Delta t) - \sin(\theta)\right) \\ \frac{\mathrm{v}}{\dot{\theta}}\left(-\cos(\theta + \dot{\theta}\Delta t) + \cos(\theta)\right) \\ \dot{z}\Delta t \\ \dot{\theta}\Delta t \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{18}$$

where Δt is the sampling time. Please note that in Equation (18), z is assumed to evolve with constant velocity. In the case of zero turning rate (i.e., $\dot{\theta} = 0$),

$$\mathrm{x}_{t+1} = \mathrm{x}_t + \begin{bmatrix} \mathrm{v}\cos(\theta) & \mathrm{v}\sin(\theta) & \dot{z}\Delta t & \dot{\theta}\Delta t & 0 & 0 & 0 \end{bmatrix}^T \tag{19}$$

The state vector of a pedestrian is

$$\mathrm{x} = \begin{bmatrix} x & y & z & \theta & \dot{x} & \dot{y} & \dot{z} & \dot{\theta} \end{bmatrix}^T \tag{20}$$

The motion of pedestrians is predicted according to CV model

$$\mathrm{x}_{t+1} = \mathrm{x}_t + \begin{bmatrix} \dot{x} & \dot{y} & \dot{z} & \dot{\theta} & 0 & 0 & 0 & 0 \end{bmatrix}^T \cdot \Delta t \tag{21}$$

3.4. Complexity Analysis

As shown in Figure 1, our data association pipeline is made of four components: Local Association, Global Association, Update Tracklets' States, Update Tracklets' Confidence. This section gives an analysis of the time complexity referred to as complexity of these four components.

Let d and h be the number of detections and the number of high confident tracklets, respectively. The time complexity of the Local Association step is the sum of the complexity of computing the cost matrix $\mathbf{L}$ in Equation (12) and solving the LAP represented by $\mathbf{L}$. Since $\mathbf{L}$ has the size of $h \times d$, the complexity of computing $\mathbf{L}$ is $\mathcal{O}(hd)$.

The LAP represented by $\mathbf{L}$ can be solved by either the Hungarian algorithm or a greedy algorithm [15]. The complexity of the Hungarian algorithm is $\mathcal{O}(hd^2)$. On the other hand, the greedy algorithm is made of two steps presented in Algorithm 1.

The first step of sorting the flattened cost matrix $\mathbf{C} \in \mathbb{R}^{r \times c}$ has the complexity of $\mathcal{O}(rc\log(rc)) = \mathcal{O}(rc\log(c))$ assuming $c > r$. The complexity of the second step in the best case scenario where the for loop is stopped at $k = 0$, meaning there is no valid association, is $\mathcal{O}(1)$. The worst case scenario happens when the For Loop proceeds till the last value of k, which means every possible association has its affinity less than the threshold σ. In this case, the complexity is $\mathcal{O}(|\mathrm{c}^{flat}|) = \mathcal{O}(rc)$. As the result, the complexity of the greedy algorithm is

$$\mathcal{O}(rc\log(c)) + \mathcal{O}(rc) = \mathcal{O}(rc\log(c)) \tag{22}$$

Using Equation (22), the complexity of the Local and Global Association step solved by the greedy algorithm are $\mathcal{O}(hd\log(d))$ and $\mathcal{O}((l + d')(h + l))\log(l + d')$, respectively. Recall l and d' are the number of low-confident tracklets and the number of detections left over by the Local Association step.

The other steps, Update Tracklets' States and Update Tracklets' Confidence, have the linear complexity because they are made of one loop through all tracklets.

Algorithm 1: Greedy algorithm for solving LAP

Input: Cost matrix $\mathbf{C} \in \mathbb{R}^{r \times c}$ and cost threshold σ
Result: List of pairs of indices. Each pair of (i, j) denotes the correspondence between a row i and a column j of the inputted cost matrix $\mathbf{C}$

```
Flatten C into an array c^flat and sort c^flat into the ascending order ;
P = ∅ ;
for k ∈ {0, 1, ..., |c^flat| − 1} do
    if c^flat > σ then
        break ;
    else
        j = k  mod c ;
        i = (k − j) / c;
        if both i and j aren't in any pairs in P then
            add (i, j) to P
        end
    end
end
```

4. Experiments

The effectiveness of our method is demonstrated by benchmarking against SORT-style baseline models on three large scale datasets: KITTI, NuScenes, and Waymo. In addition, we perform an ablation study using NuScenes dataset to better understand the impact of each component on our system's general performance.

4.1. Tuning the Hyper Parameters

There are three hyper parameters in our data association pipeline: the confidence threshold τ^c, the detection model accuracy β in Equation (3), and the affinity threshold σ.

The confidence threshold τ^c is set to 0.5 according to [22]. It is worth noticing that [22] suggests that this parameter does not have any significant effect on the tracking performance. The value of β is chosen empirically such that a high-confident tracklet becomes low-confident after being undetected for three frames.

As observed from experiments, the position affinity $\Lambda(\cdot, \cdot)^p$ is the dominant component in the tracklet-to-detection and tracklet-to-tracklet affinity. Since the position affinity, which is the Mahalanobis distance between expected detection and real detection, is χ^2 distributed, the affinity threshold σ in Equation (11) is chosen according to the percentile of χ^2 distribution where the position affinity resulted from a correct association is expected to fall into. Notice that the degree of freedom of the χ^2 distribution of our interest is 4 due to the dimension of the measurement vector z in Equation (6).

Intuitively, the affinity threshold σ determines how conservative our tracking algorithm is. Small σ makes our algorithm be more skeptical by rejecting detections that are close, but not close enough to the prediction of tracks' states. This works well in the scenario where a large number of false-positive detections presents (e.g., Waymo dataset). However, too small σ can reject correct detections thus deteriorating the tracking performance. The method used for searching for a good value of σ is

- Performs a coarse grid search with the expected percentile of χ^2 distribution in the set $\{10\%, 50\%, 90\%, 95\%, 97.5\%, 99\%\}$ which means the value of σ is in the set $\{0.53, 1.67, 3.89, 4.75, 5.57, 6.64\}$, while keeping the rest of hyper parameters unchanged. Please note that here the value of the threshold σ is just half of the corresponding value in χ^2 Distribution Table. This is because the motion affinity is scaled by half in our implementation to reduce its dominance over the size affinity.
- Once a performance peak is identified at $\hat{\sigma}$, a fine grid search is performed on the set $\{\hat{\sigma} - 0.2, \hat{\sigma} - 0.1, \hat{\sigma}, \hat{\sigma} + 0.1, \hat{\sigma} + 0.2\}$

The resulted value of σ on KITTI, NuScenes, and Waymo are respectively 6.5, 4.5, and 1.5.

4.2. Tracking Results

Evaluation Metrics: Classically, MOT systems are evaluated by the CLEAR MOT metrics [29] which compute tracking performance based on three cores quantities which are the number of False Positives, False Negatives, and ID Switches (the definition of these quantities can be found in Section 1). Intuitively, this set of metrics aims at evaluating a tracker's precision in estimating tracks' states as well as its consistency (i.e., keeping a unique ID for each even in the presence of occlusion). As pointed out by [30] and later by [6], there is a linear relation between MOTA and object detectors' recall rate, as a result, MOTA does not provide a well-rounded evaluation performance of trackers. To remedy this, [6] proposes to average MOTA and MOTP over a range of recall rate, resulting in two integral metrics AMOTA and AMOTP which become the norm in recent benchmarks.

Datasets: To verify the effectiveness of our method, we benchmark it on three popular autonomous driving datasets which offer 3D MOT benchmark: KITTI, NuScenes, and Waymo. These datasets are collections of driving sequences collected in various environment using a multi-modal sensor suite including LiDAR. KITTI tracking benchmark interests in two classes of object which are cars and pedestrians. Initially, KITTI tracking was designed for MOT in 2D images and recently [6] adapts it to 3D MOT. NuScenes concerns a larger set of objects which comprises of cars, bicycles, buses, trucks, pedestrians, motorcycles, trailers. Waymo shares the same interest as NuScenes but groups car-like vehicles into a meta class.

Public Detection: As can be seen in Table 1, AMOTA highly depends on the precision of object detectors. Therefore, to have a fair comparison, the baseline detection results made publicly available by the benchmarks are used as the input to our tracking system. Specifically, we use Point-RCNN detection for KITTI dataset, MEGVII detection for NuScenes, and PointPillars with PPBA detection for Waymo.

The performance of our model compared to the SORT-style baseline model in three popular benchmarks are shown in Table 2.

Table 2. Quantitative performance of our model on KITTI validation set, NuScenes validation set, and Waymo test set. AMOTA is the primary metric of these benchmarks. FP, FN IDS and FRAG are absolute numbers in the case of KITTI and NuScenes, while they are divided by the total number of objects in Waymo. The performance on Waymo is calculated at the difficulty of LEVEL 2.

Dataset	Method	AMOTA↑	AMOTP↓	MT↑	ML↓	FP↓	FN↓	IDS↓	FRAG↓
KITTI (val)	Ours	**0.415**	0.691	NA	NA	766	3721	10	259
	AB3DMOT [6]	0.377	**0.648**	NA	NA	**696**	**3713**	**1**	**93**
NuScenes (val)	Ours	**0.583**	**0.748**	**3617**	1885	13,439	**28,119**	**512**	**511**
	StanfordIPRL-TRI [15]	0.561	0.800	3432	**1857**	**12,140**	28,387	679	606
Waymo (test @ L2)	Ours	**0.365**	**0.263**	NA	NA	**0.089**	**0.533**	0.014	NA
	PPBA-AB3DMOT	0.291	0.270	NA	NA	0.171	0.535	**0.003**	NA

As can be seen, our model consistently outperforms the baseline model in term of the primary metric AMOTA, thus proving the effectiveness of the 2-stage data association. Specifically, the improvements are 10.080%, 3.922%, and 25.430% for KITTI, NuScenes and Waymo, respectively. It is worth noticing that our approach has more track fragmentations (FRAG), 259 compared to 93 of the base line, in KITTI. The reason for this is that at each time step tracklets have no matched detections are not reported by our approach, while the baseline predicts their pose using the constant velocity model (CV) and reports this prediction.

The comparison runtime on KITTI dataset of our tracking algorithm against AB3DMOT [6] is shown in Table 3. Despite the additional complexity added by the second stage of the

data association (i.e., the Global Association step), our approach can achieve a runtime that is close to AB3DMOT on KITTI and exceeds the real-time speed by a large margin. On more challenging datasets, the object detector generates a significantly larger number of detections per frame on average, 57.50 on NuScenes and 264.18 on Waymo, compared to 10.04 of KITTI. This large number of detections enlarges the cost matrix of the Local and Global Association step, thus making the LAPs represented by them more costly to solve. Therefore, the runtime of our approach is reduced to 1.44 frames-per-second (fps) on NuScenes and 0.35 fps on Waymo. This runtime can be greatly improved if our approach is re-implemented in a compiling language such as C++.

Table 3. Comparison of our approach's runtime on KITTI dataset against AB3DMOT's.

Class of Objects	Our Runtime (fps)	AB3DMOT's Runtime (fps)
Car	115	186
Pedestrian	497	424
Cyclist	1111	1189

The qualitative performance on NuScenes is illustrated by drawing the bird-eye view of a scene with tracking result, ground truth objects and detection result accumulated through time as in Figures 3 and 4.

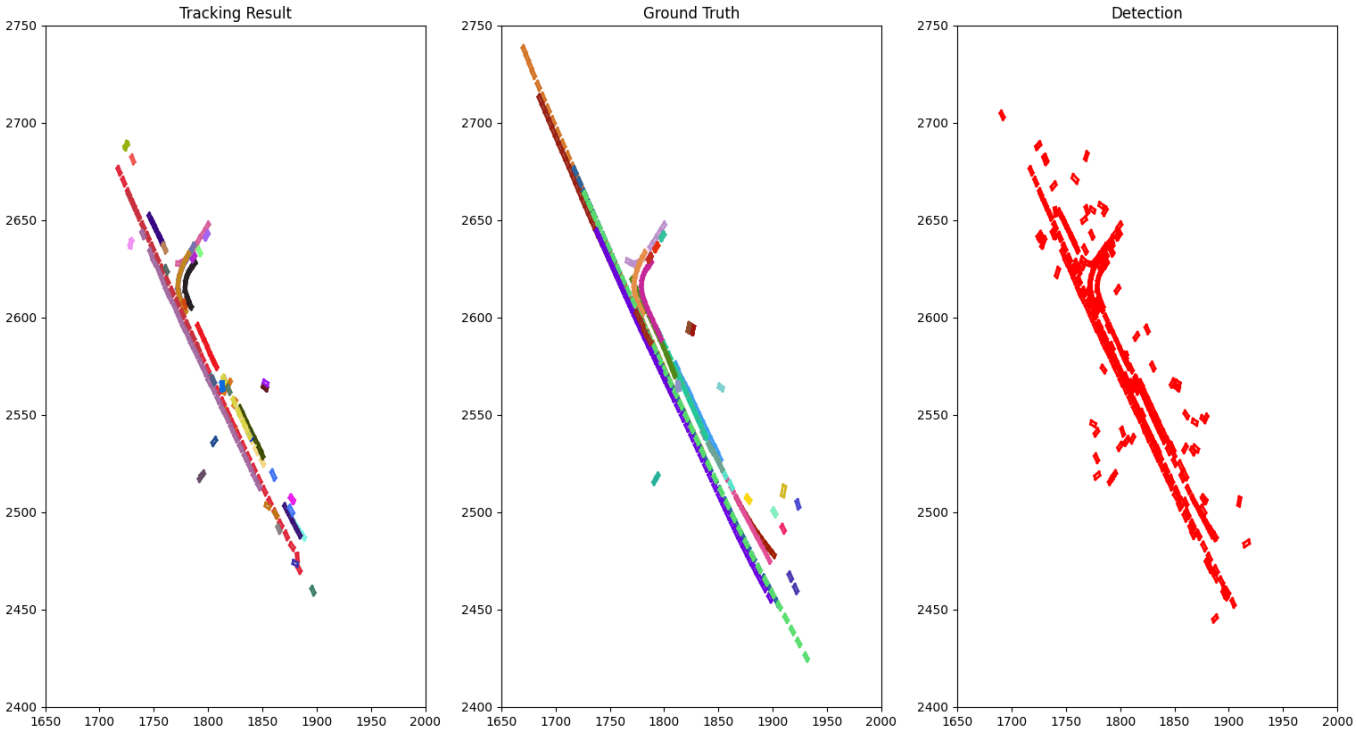


Figure 3. The bird-eye view of the tracking result for class car compared to the ground truth of scene 0796 (NuScenes) accumulated through time. Each rectangle represents a car and each color is associated with a track ID.

The difficulty of the 3D MOT can be appreciated by the noisy detection with several false positives denoted by the clutter in the top of Figure 4-Detection and false negatives, as shown by the absence of one trajectory in the top left corner of Figure 3-Detection.

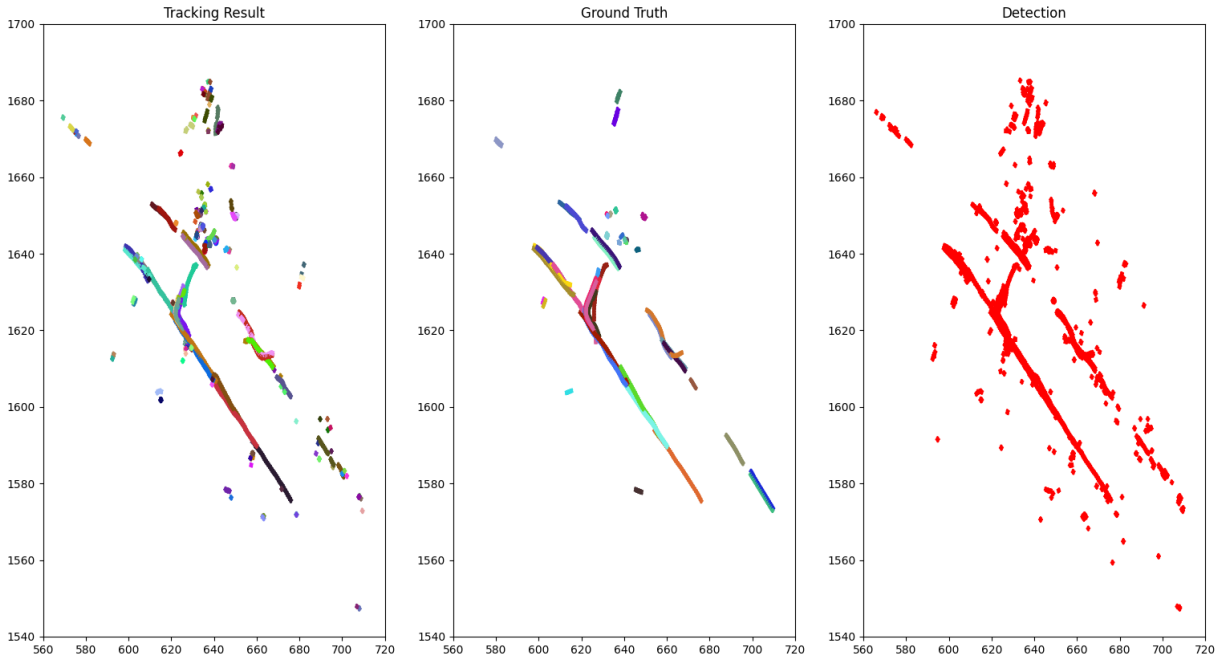


Figure 4. The bird-eye view of tracking result for class pedestrian compared to the ground truth of scene 0103 (NuScenes) accumulated through time. Each dot represents a pedestrian and each color is associated with a track ID.

4.3. Ablation Study

In this ablation study, the default method is the method presented in Section 3 which has

- Two stages of data association (local and global). Each stage is formulated as a LAP and solved by a greedy matching algorithm [15].
- The affinity function the sum of position affinity and size affinity (as in Equation (4)).
- The motion model is Constant Turning Rate and Velocity (CTRV) for car-like objects (cars, buses, trucks, trailers, bicycles) and Constant Velocitiy (CV) for pedestrians.
- As mentioned in Section 4.1, the value of hyperparameters are set as follows: $\beta = 1.35$ (in Equation (3)), tracklet confidence threshold $\tau^c = 0.45$, and the affinity threshold $\sigma = 4.5$ (in Equation (11))

To understand the effect of each component on the system's general performance, we modify or remove each of them and carry out experiment with the rest of the system being kept the same as the default method and the same hyperparameters. The changes and the resulted performance are shown in Table 4.

Table 4. Ablation study using NuScenes dataset.

Method	AMOTA↑	AMOTP↓	MT↑	ML↓	FP↓	FN↓	IDS↓	FRAG↓
Default	0.583	0.748	**3617**	1885	13,439	28,119	512	511
Hungarian for LAP	**0.587**	**0.743**	3609	**1880**	13,667	**28,070**	596	573
No ReID	0.583	0.748	3616	1882	13,429	28,100	**504**	**510**
Global assoc only	0.327	0.924	2575	2244	26,244	38,315	4215	3038
Const Velocity only	0.567	0.781	3483	1966	12,649	29,427	718	606
No size affinity	0.581	0.748	3595	1904	13,423	28,448	512	508
3D IoU as affinity	0.535	0.898	3090	2075	**9168**	33,041	550	528

It can be seen that solving the matching problem (formulated as a LAP) with the Hungarian algorithm instead of the greedy matching algorithm of [15] results in a marginal increase of AMOTA; however, this increased performance comes at the cost of increased execution time since the Hugarian algorithm has higher time complexity (cubic time

compared to quadratic time.). In addition, using Constant Velocity model only reduces the AMOTA by 2.744% compared to the Default setting which shows the effectiveness of the Constant Turning Rate and Velocity model in predicting motion of car-like vehicle. Finally, performing global association only deteriorates the tracking performance confirms the importance of the local association step which significantly reduce the association ambiguity for the second stage.

5. Conclusions and Perspectives

In conclusion, this paper successfully adapted an image-based tracking method to the 3D space. Particularly, extensive experiments carried out in various datasets shows that our two-stage data association pipeline can result in significant improvement in the tracking accuracy by adding a certain degree of re-identification while keeping the added complexity to the minimum. Nevertheless, medium and long-term occlusion remains challenging for our approach due to the fact that the affinity function relies mostly on tracklets position whose prediction's reliability reduces with the length of the prediction horizon. In the domain of image-based MOT, this problem is offend solved by exploiting tracklets' appearance with Siamese networks [31,32]. However, the extension of this method to 3D space is not straightforward due to the lack of color and texture in point cloud. A possibility to resolve this issue is to associate 3D tracklets to 2D object detections, then carry out re-identification in images. Taking a different approach, a recent work in graph neural networks [33] proposes to jointly learn affinity function from point clouds and images.

Author Contributions: Conceptualization, M.-Q.D. and V.F.; methodology, M.-Q.D.; validation, M.-Q.D. and V.F.; formal analysis, M.-Q.D.; investigation, M.-Q.D. and V.F.; resources, V.F.; writing—original draft preparation, M.-Q.D. and V.F.; writing—review and editing, M.-Q.D. and V.F.; visualization, M.-Q.D.; supervision, V.F.; funding acquisition, V.F. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: This research has been conducted as part of the AIby4 project (AI by/for Human, Health and Industry), funded by the French Ministry of Education and Research and the French National Research Agency (ANR-20-THIA-0011).

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

MOT	Multi-Object Tracking
IoU	Intersection over Union
LAP	Linear Assignment Problem
CTRV	Constant Turning Rate and Velocity
CV	Constant Velocity
AMOTA	Average Multi-Object Tracking Accuracy
AMOTP	Average Multi-Object Tracking Precision
MT	Mostly Track
ML	Mostly Lost
FP	False Positive
FN	False Negative
IDS	ID Switches
FRAG	Fragment
FPS	Frames Per Second

References

1. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *39*, 1137–1149. [CrossRef] [PubMed]

2. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In *European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 21–37.
3. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
4. Bewley, A.; Ge, Z.; Ott, L.; Ramos, F.; Upcroft, B. Simple online and realtime tracking. In Proceedings of the 2016 IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, USA, 25–28 September 2016; pp. 3464–3468.
5. Scheidegger, S.; Benjaminsson, J.; Rosenberg, E.; Krishnan, A.; Granström, K. Mono-camera 3d multi-object tracking using deep learning detections and pmbm filtering. In Proceedings of the 2018 IEEE Intelligent Vehicles Symposium (IV), Changshu, China, 26–30 June 2018; pp. 433–440.
6. Weng, X.; Wang, J.; Held, D.; Kitani, K. AB3DMOT: A Baseline for 3D Multi-Object Tracking and New Evaluation Metrics. *arXiv* **2020**, arXiv:2008.08063.
7. Kuhn, H.W. The Hungarian method for the assignment problem. *Nav. Res. Logist. Q.* **1955**, *2*, 83–97. [CrossRef]
8. Liang, M.; Yang, B.; Zeng, W.; Chen, Y.; Hu, R.; Casas, S.; Urtasun, R. PnPNet: End-to-End Perception and Prediction with Tracking in the Loop. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 16–18 June 2020; pp. 11553–11562.
9. Yin, T.; Zhou, X.; Krähenbühl, P. Center-based 3d object detection and tracking. *arXiv* **2020**, arXiv:2006.11275.
10. Luo, W.; Yang, B.; Urtasun, R. Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 3569–3577.
11. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets robotics: The kitti dataset. *Int. J. Robot. Res.* **2013**, *32*, 1231–1237. [CrossRef]
12. Caesar, H.; Bankiti, V.; Lang, A.H.; Vora, S.; Liong, V.E.; Xu, Q.; Krishnan, A.; Pan, Y.; Baldan, G.; Beijbom, O. Nuscenes: A multimodal dataset for autonomous driving. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11621–11631.
13. Sun, P.; Kretzschmar, H.; Dotiwalla, X.; Chouard, A.; Patnaik, V.; Tsui, P.; Guo, J.; Zhou, Y.; Chai, Y.; Caine, B.; et al. Scalability in perception for autonomous driving: Waymo open dataset. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 2446–2454.
14. Garcia-Fernandez, A.F.; Williams, J.L.; Granstrom, K.; Svensson, L. Poisson Multi-Bernoulli Mixture Filter: Direct Derivation and Implementation. *IEEE Trans. Aerosp. Electron. Syst.* **2018**, *54*, 1883–1901. [CrossRef]
15. kuang Chiu, H.; Prioletti, A.; Li, J.; Bohg, J. Probabilistic 3D Multi-Object Tracking for Autonomous Driving. *arXiv* **2020**, arXiv:2001.05673.
16. Zhu, B.; Jiang, Z.; Zhou, X.; Li, Z.; Yu, G. Class-balanced grouping and sampling for point cloud 3d object detection. *arXiv* **2019**, arXiv:1908.09492.
17. Zhou, X.; Wang, D.; Krähenbühl, P. Objects as Points. *arXiv* **2019**, arXiv:1904.07850.
18. Ding, Z.; Hu, Y.; Ge, R.; Huang, L.; Chen, S.; Wang, Y.; Liao, J. 1st Place Solution for Waymo Open Dataset Challenge–3D Detection and Domain Adaptation. *arXiv* **2020**, arXiv:2006.15505.
19. Ge, R.; Ding, Z.; Hu, Y.; Wang, Y.; Chen, S.; Huang, L.; Li, Y. AFDet: Anchor Free One Stage 3D Object Detection. *arXiv* **2020**, arXiv:2006.12671.
20. Shi, S.; Guo, C.; Jiang, L.; Wang, Z.; Shi, J.; Wang, X.; Li, H. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 16–18 June 2020; pp. 10529–10538.
21. Cheng, S.; Leng, Z.; Cubuk, E.D.; Zoph, B.; Bai, C.; Ngiam, J.; Song, Y.; Caine, B.; Vasudevan, V.; Li, C.; et al. Improving 3D Object Detection through Progressive Population Based Augmentation. *arXiv* **2020**, arXiv:2004.00831.
22. Bae, S.H.; Yoon, K.J. Robust online multi-object tracking based on tracklet confidence and online discriminative appearance learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 1218–1225.
23. Arnold, E.; Al-Jarrah, O.Y.; Dianati, M.; Fallah, S.; Oxtoby, D.; Mouzakitis, A. A survey on 3d object detection methods for autonomous driving applications. *IEEE Trans. Intell. Transp. Syst.* **2019**, *20*, 3782–3795. [CrossRef]
24. Geiger, A.; Lauer, M.; Wojek, C.; Stiller, C.; Urtasun, R. 3d traffic scene understanding from movable platforms. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *36*, 1012–1025. [CrossRef]
25. Leal-Taixé, L.; Milan, A.; Reid, I.; Roth, S.; Schindler, K. Motchallenge 2015: Towards a benchmark for multi-target tracking. *arXiv* **2015**, arXiv:1504.01942.
26. Mauri, A.; Khemmar, R.; Decoux, B.; Ragot, N.; Rossi, R.; Trabelsi, R.; Boutteau, R.; Ertaud, J.Y.; Savatier, X. Deep Learning for Real-Time 3D Multi-Object Detection, Localisation, and Tracking: Application to Smart Mobility. *Sensors* **2020**, *20*, 532. [CrossRef] [PubMed]
27. Bae, S.H.; Yoon, K.J. Confidence-based data association and discriminative deep appearance learning for robust online multi-object tracking. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *40*, 595–610. [CrossRef]
28. Yang, H.; Wen, J.; Wu, X.; He, L.; Mumtaz, S. An efficient edge artificial intelligence multipedestrian tracking method with rank constraint. *IEEE Trans. Ind. Inform.* **2019**, *15*, 4178–4188. [CrossRef]

29. Bernardin, K.; Stiefelhagen, R. Evaluating multiple object tracking performance: The CLEAR MOT metrics. *EURASIP J. Image Video Process.* **2008**, *2008*, 1–10. [CrossRef]
30. Leal-Taixé, L.; Milan, A.; Schindler, K.; Cremers, D.; Reid, I.; Roth, S. Tracking the trackers: An analysis of the state of the art in multiple object tracking. *arXiv* **2017**, arXiv:1704.02781.
31. Leal-Taixé, L.; Canton-Ferrer, C.; Schindler, K. Learning by tracking: Siamese CNN for robust target association. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Paris, France, 12 September 2016; pp. 33–40.
32. Chang, S.; Li, W.; Zhang, Y.; Feng, Z. Online siamese network for visual object tracking. *Sensors* **2019**, *19*, 1858. [CrossRef] [PubMed]
33. Weng, X.; Wang, Y.; Man, Y.; Kitani, K.M. Gnn3dmot: Graph neural network for 3d multi-object tracking with 2d-3d multi-feature learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 6499–6508.