Final Report: 3D Multi-Object Tracking for Autonomous Vehicles

Tara Sadjadpour
Stanford University
tsadja@stanford.edu

Abstract

In this work, we explore 3D multi-object tracking under the tracking-by-detection paradigm. Our goal is to take the state-of-the-art tracking algorithm which won the nuScenes competition [8] and adapt it to work with the Waymo Perception dataset. Furthermore, we examine ways that we can improve the algorithm by analyzing experimental results we achieve on the Waymo dataset, as well as conducting a literature review on this area to find alternative techniques. Ultimately, through various permutations of our setup, we achieve competitive results when compared to tracking algorithms that use the same initial detections as input. Additionally, our literature review leads us to decide that we should adapt transformer-based approaches that are successful in 2D multi-object tracking to the 3D case to improve the algorithm. [1]

1. Introduction

3DMOT is an important task for autonomous driving vehicles. In the computer vision community, this is an area of great concern, and there are several high-profile competitions in this area from nuScenes (run by the company Motional, formerly known as NuTonomy) and Waymo. The goal of 3DMOT is to estimate the location, orientation, and scale of objects within the vehicle's viewpoint over time. The main paradigm used in 3DMOT is called tracking-by-detection, whereby 3DMOT systems use detections provided by object detectors to track each object in a given trajectory, or tracklet, over time. Ideally, a 3DMOT system should adeptly use temporal information to track objects over consecutive frames, be robust to occlusions, filter outliers provided by object detectors, and perform data association between object detections and predicted object states to generate tracklets correctly. Using these learned trajectories, the autonomous vehicle can improve its planning abilities by inferring motion patterns and driving behaviors of surrounding pedestrians, cyclists, and vehicles.

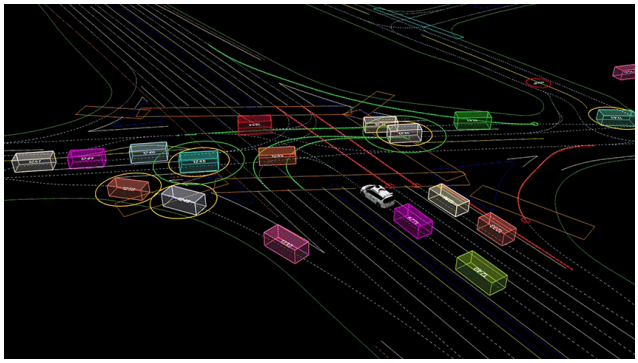

Figure 1. 3DMOT shown in busy intersection. Image taken from Ara Intelligence Blog.

In this work, we aim to improve an existing state-of-the-art 3D multi-object tracking (3DMOT) algorithm that uses the tracking-by-detection framework [8]. There also exists a version of this algorithm that handles multi-modal data [7], but we do not address this version of the algorithm for the project. The 3DMOT algorithm of interest is available on GitHub and is assessed on the nuScenes dataset [2]. Our goals and overall approach for this project can be expressed in two parts: 1) we want to re-write the official GitHub associated with [8] so we can apply the algorithm to the Waymo Perception dataset [13] to see how the algorithm performs on new data; 2) we want to conduct a thorough literature review in the areas of differentiable filtering and data association to figure out new directions for improving the existing state-of-the-art. This research is being conducted under the supervision of Professor Jeannette Bohg, and our partners at Toyota Research Institute (TRI).

2. Background

The 3DMOT algorithm presented in [8] won the nuScenes 3DMOT challenge at the NIPS 2019 AI Driving Olympics. The algorithm uses the detections generated by MEGVII [20], which won the nuScenes 3D Detection Challenge held in the CVPR 2019 Workshop on Autonomous

[1] Please do not post this final report on the class website, as this is an ongoing research project. Also, I have shared my private GitHub repository containing the code for this project with Krishnan.

	KITTI	NuScenes	Argo	Ours
Scenes	22	1000	113	1150
Ann. Lidar Fr.	15K	40K	22K	230K
Hours	1.5	5.5	1	6.4
3D Boxes	80K	1.4M	993k	12M
2D Boxes	80K	–	–	9.9M
Lidars	1	1	2	5
Cameras	4	6	9	5
Avg Points/Frame	120K	34K	107K	177K
LiDAR Features	1	1	1	2
Maps	No	Yes	Yes	No
Visited Area (km^2)	–	5	1.6	76

Figure 2. Comparison between Waymo Perception dataset (labeled as "Ours" column) and other datasets, specifically nuScenes. Figure taken from [13].

Driving. For tracking, the authors use a vanilla Kalman Filter. Given each object detection's 3D bounding box, treated as an uncertain measurement, the Kalman filter aims to predict the state of each detection in the next time stamp. The MEGVII detections and Kalman filter predictions are then matched using a greedy algorithm for data association to create the tracklets for each tracked object. The main contributions of this paper are considered to be: 1) its technique for covariance matrix initialization in the Kalman filter, and 2) its use of Mahalanobis distance for data association.

Additionally, the Waymo Perception Dataset is of interest to us, because it offers more information than the nuScenes dataset. In summary, Waymo uses 5 lidar sensor as opposed to 1 lidar sensor in nuScenes; this leads to denser point clouds, which should make the initial object detections provided to the 3DMOT algorithm more accurate. Furthermore, Waymo provides 1000 scenes for training and validation, where each scene spans 20 seconds and has a sampling rate of 10 frames per second. On the other hand, nuScenes provides 850 scenes for training and validation that are each 20 seconds, but the sampling rate is only 2 frames per second. As a result, Waymo is the only dataset with more annotations than nuScenes. This makes the Waymo Perception dataset more rigorous in testing the 3DMOT algorithm. This information is summarized in Figure 2.

3. Approach

Based on the framework we have thus far in the project, our goal is to complete two tasks: 1) use the algorithm from [8] to generate results with the Waymo Perception dataset [13], as our funders at TRI would like for us to move away from nuScenes and towards this dataset; and 2) explore literature on differentiable filtering to potentially replace the vanilla Kalman filter with a more sophisticated technique for tracking, while exploring state-of-the-art data association algorithms to discover ways to replace the greedy algorithm currently being used in [8].

For the first part of the problem involving the evaluation on Waymo Perception data, I will be doing pure software engineering. It is important to note that this is not a trivial task, and it is an ongoing process within the TRI organization. General hurdles with the Waymo dataset include bad data format, difficult pre-processing, and poor documentation with weak explanations. Furthermore, moving the existing algorithm from nuScenes to Waymo is hard, because the GitHub only used nuScenes, and thus, the authors do all of their operations using the nuScenes dev-kit provided by Motional, the company that made the nuScenes dataset. The nuScenes dev-kit has many built-in functions that streamline data processing and evaluation, but Waymo does not provide any such functions. My main approach is to focus on reading [13] to better understand the dataset, playing with small toy examples on a few frames from one scene to learn how to handle the data, and finally making my own "Waymo dev-kit" and rewriting the algorithm to accommodate my new "Waymo dev-kit".

For running the algorithm on the Waymo dataset, we follow the same procedure as [8], where the training data statistics are used to initialize the covariance matrices in the Kalman Filter and we hand-tune the Mahalanobis distance threshold for deciding which detections should be used for matching in the data association step.

For the second part of the problem involving the literature review, I intend to read many papers in the differentiable filtering and data association domains. If a paper does not seem helpful to me, I will discard it, but if I think it has useful ideas that could improve the existing algorithm, then I will write a summary on it as useful background for future directions, and present it to my supervisors.

For the literature review, we expect to find papers that can inform us on directions we can pursue to further improve the existing state-of-the-art. My main metric for evaluating this part of my project is based on feedback I receive in weekly meetings with Jeannette and bi-weekly meetings with our TRI partners. I will present the literature that will ultimately influence our new direction in the Literature Review section.

4. Experiments

We will evaluate our performance on the Waymo Perception dataset's validation split. Note that we will use the baseline PointPillars+PPBA detections [6] provided in the Waymo 3DMOT challenge as input to our Kalman filter. We will evaluate the performance of the 3DMOT algorithm for Waymo data using the primary metric used in the Waymo 3DMOT challenge, Multiple Object Tracking Ac-

Model	MOTA ↑	MOTP ↓
PPBA AB3DMOT	**0.3600**	**0.1812**
Ours (T=2)	0.2965	0.1813
Ours (T=3)	0.3368	0.1822
Ours (T=8)	0.3514	0.1832
Ours (T=11)	0.3549	0.1827
Ours (T=13)	0.3536	0.1828
Ours (T=20)	0.3499	0.1828

Table 1. Results on vehicles object type. T indicates the Mahalanobis distance threshold we set.

Model	MOTA ↑	MOTP ↓
PPBA AB3DMOT	**0.2730**	0.3431
Ours (T=2)	0.2098	**0.3412**
Ours (T=3)	0.2370	0.3423
Ours (T=8)	0.2495	0.3426
Ours (T=11)	0.2458	0.3417
Ours (T=13)	0.2439	0.3417
Ours (T=20)	0.2392	0.3421

Table 2. Results on pedestrian object type. T indicates the Mahalanobis distance threshold we set.

Model	MOTA ↑	MOTP ↓
PPBA AB3DMOT	**0.2413**	**0.2844**
Ours (T=2)	0.1252	0.2961
Ours (T=3)	0.1472	0.3011
Ours (T=8)	0.1456	0.3058
Ours (T=11)	0.1511	0.3030
Ours (T=13)	0.1495	0.3026
Ours (T=20)	0.1499	0.3043

Table 3. Results on cyclist object type. T indicates the Mahalanobis distance threshold we set.

curacy (MOTA) [1]. Essentially, MOTA penalizes false positives, misses, and mismatches over the objects present in all timestamps. Additionally, as a secondary metric, I will assess the performance on Multiple Object Tracking Precision (MOTP) [1], which is a distance metric that shows the ability of the tracker to estimate precise object positions. Thus, we want to maximize MOTA, while minimizing MOTP. The data is labeled as either L1 (level 1) or L2 (level 2). The criteria to be in a specific difficulty level can depend on both the human labelers and the object statistics. In the literature and on the Waymo 3DMOT challenge leaderboard, an algorithm's ranking depends on its L2 MOTA performance, because the metrics for L2 are cumulative and thus include L1.

Please note that we are unable to produce visualizations for the Waymo dataset's tracking challenge. The MOTA and MOTP metrics are computed using Waymo's code after we create submission files in the correct format. However, the Waymo organization did not create any way to make visualizations, unlike nuScenes which has a built-in visualization tool for its competition. For this reason, the literature produced on the Waymo dataset only provides tables with quantitative results, and visualizations of the tracks do not exist. This makes it harder to develop intuition about the results beyond the quantitative metrics, and it is a limitation that should be taken into consideration for readers.

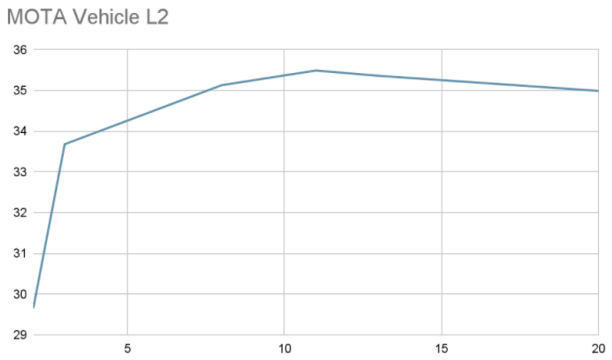

Figure 3. MOTA for vehicles versus Mahalanobis threshold value. The y-axis is the MOTA in percentage form, while the x-axis is the threshold value.

First, we get the results for PPBA+PointPillars using various Mahalanobis thresholds for the data association algorithm. We compare against PPBA AB3DMOT[2], which is the state-of-the-art algorithm when PPBA+PointPillars are the input detections. The results are shown in Tables 1, 2, and 3. My initial intuition was that the threshold was the cause of my poor results, when I was using threshold of 11, just as [8]. Thus, I tried many thresholds as can be seen in the tables. However, in Figure 3, we see that the peak performance was actually at threshold of 11. Furthermore, even when I made the threshold extremely small at 2, which means a high percentage of objects would be discarded from being eligible to become part of a track, the accuracy dropped by less than 6%. When I made the threshold 3, which is still very small, the loss in accuracy is still less than 2%. When I made the threshold extremely large at 20, the loss was not heavily affected either.

These initial experiments told me that the threshold was not the source of my poor results. As a result, I turned my attention towards the covariance matrix initialization. When I was looking at covariance initializations in other works, I noticed that in the GitHub page for [9], the authors used the nuScenes covariance initialization from the original implementation of the algorithm I am working on [8] even though they are evaluating on Waymo data. On the other hand, my results in Tables 1, 2, and 3 are based on

[2]Found on https://waymo.com/open/challenges/entry/?challenge=TRACKING_3D&emailId=aff61d2b-db83&timestamp=1591033253743188

Model	MOTA ↑	MOTP ↓
PPBA AB3DMOT	0.3600	**0.1812**
Ours (T=11)	**0.3815**	0.1817

Table 4. Results on vehicles object type for new covariance initialization.

Model	MOTA ↑	MOTP ↓
PPBA AB3DMOT	0.2730	0.3431
Ours (T=11)	**0.2821**	**0.3397**

Table 5. Results on pedestrian object type for new covariance initialization.

statistics I calculated with PPBA+PointPillars training predictions for the covariance initialization. At first I found it strange that a covariance initialization based on statistics from a completely different dataset worked well. However, upon closer inspection, this makes sense. The covariance initialization from the original algorithm's implementation are based on the MEGVII nuScenes detector [20]. The MEGVII detector has accuracy of 52.8% on nuScenes, whereas PPBA+PointPillars has accuracy of only 35.3% on Waymo. There are 3 matrices that we initialize for the Kalman Filter: process noise uncertainty (Q), estimate uncertainty (P), and measurement uncertainty (R). Q is initialized with the ground truth training statistics, while P and R are initialized with the difference between ground truth and prediction training statistics. When we have low accuracy detectors like PPBA+PointPillars, we have high estimate and measurement uncertainty, which makes the Kalman Filter have a slower estimate convergence.

Thus, as an experiment to test my theory about the covariance matrix initialization, I used the PPBA+PointPillars detections, but my covariance matrx initialization comes from the MEGVII nuScenes covariance initialization used in the original algorithm [8]. Note that this is okay to do, since the dimensionality of the state is the same for both implementations, and ultimately, the covariance matrix initialization is a heuristic, i.e. many papers just set the diagonal entries to some constant value.

The results in Tables 4, 5, and 6 show significant improvement in the primary metric MOTA for our model compared to the results in Tables 1, 2, and 3. For vehicles we have a 3% gain in performance, a 4% gain for pedestrians, and a 3% gain for cyclists. We beat PPBA AB3DMOT for MOTA of vehicles and pedestrians, but we still do poorly for the cyclist. Ultimately, these results show that the algorithm is very sensitive to the covariance matrix initialization, which motivates us to consider finding alternative, more robust methods through our literature review.

Model	MOTA ↑	MOTP ↓
PPBA AB3DMOT	**0.2413**	**0.2844**
Ours (T=11)	0.1804	0.2920

Table 6. Results on cyclist object type for new covariance initialization.

5. Literature Review

For the literature review, I initially started by focusing on differentiable filtering [10], and more specifically differentiable factor graphs which have a smoothing capability [12, 16]. My initial idea was to replace the vanilla Kalman filter with a differentiable factor graph, because it also provides smoothing capabilities that allows us to retrospectively correct information that we predicted at previous time-steps. However, there are several hurdles here: 1) the differentiable factor graph has only been shown to work on single object tracking [16]; 2) if we were to use the differentiable factor graph for multi-object tracking, we would have to consider either making one dynamic factor graph that adds components as new objects arise or having multiple factor graphs with each focusing on one object in the scene. I spoke to the first author from [16] in a conference call about these two problems, and he told me that he had worked on this problem for a long time, but found no reasonable solution to deal with the 2 aforementioned problems with the differentiable factor graph. Based on this input, I decided to move away from the differentiable filtering aspect of the problem, and my advisor and TRI partners approved of this choice.

Currently, I have turned my focus to data association, which aims to find the best way to match the tracked object predictions (characterized with mean and innovation covariance matrix) and the detections provided by our detector. The existing algorithm uses a greedy approach, but newer papers show more nuanced and promising techniques [5, 9, 18, 19]. For the sake of space, I will focus on [9, 18], as these are the two approaches from which I am most interested in re-using ideas.

In [9], the authors propose a two-stage data association approach, which shows improvements over the standard approach of using linear mapping with bipartite graphs to do data association. The authors present their results on the KITTI, nuScenes, and Waymo datasets. The performance of their approach is assessed using the metric called Average Multi Object Tracking Accuracy (AMOTA) [14], which averages over the MOTA metric at different recall thresholds. The authors show improvements in AMOTA for all three datasets against the state-of-the-art. Specifically, for nuScenes, they compare against the model from [8], and achieve a 2.2% boost in AMOTA accuracy (theirs is 0.583, while [8] was 0.561). The two-stage approach is an adaptation of a 2D image-based tracking approach to 3D space

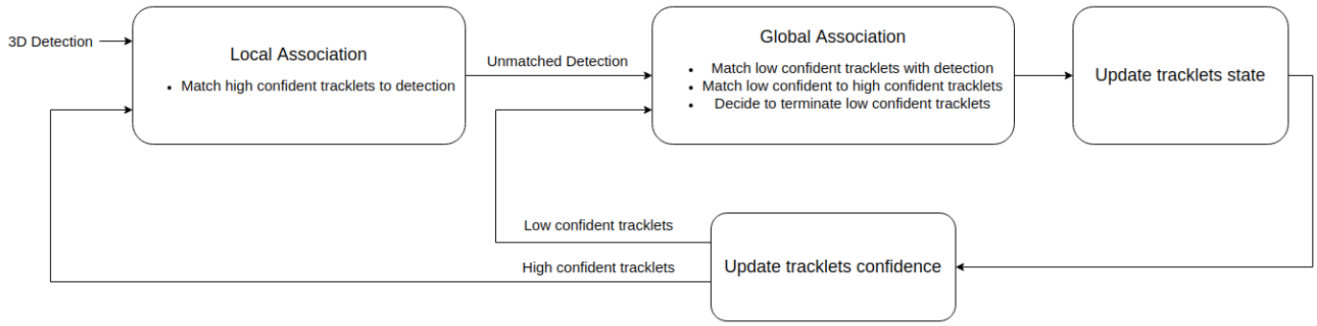

Figure 4. Two-stage data association algorithm. Figure taken from [9].

using motion models. The algorithm's first stage is a local association, which matches high confidence tracklets with detections. The second stage, known as the global association, takes the unmatched detections from the first stages, as well as the low confidence tracklets, as input. Three possible outcomes occur: low confidence tracklets are matched with detections, merged with high confidence tracklets, or discarded. Then the tracklet states and confidences are updated to create a new set of high confidence and low confidence tracklets to iterate through the two-stage process again. The authors conclude that this technique performs well on datasets with good confidence and decisiveness – 2 qualities that are valued in the AMOTA metric. The main weakness of the algorithm lies in dealing with medium- and long-term occlusions. The authors suggest using Siamese networks to deal with these longer periods of occlusion, but the main issue with such an approach is adapting it from the 2D domain to the 3D domain [4, 11]. Alternatively, the authors suggest using graph neural networks to jointly learn the affinity function from point clouds and images [15]. Please note that upon reading the GitHub associated with this paper, I realized that the authors of this work completed the Waymo analysis using statistics from the nuScenes dataset. I contacted the first author over email about this, and he told me that they did not want to go through the trouble of writing code to find the covariance initialization for Waymo training data, as this is a cumbersome task. He concluded his email saying, "I used NuScenes covariance for Waymo as I've experienced that the algorithm is not too sensitive with covariance as long as the magnitude is comparable." This is important to take into consideration, since the Experiments section has shown to us that the covariance matrix initialization is very sensitive.

Additionally, in [18], the authors introduce MOTR, an end-to-end multi-object tracking framework, inspired by DETR [3], an end-to-end object detection framework introduced in 2020. According to the authors, past techniques adopt simple heuristics to characterize various aspects of temporal motion modeling. This results in techniques that are not end-to-end and create fragmentations in the temporal motion modeling by splitting this process into 3 parts: 1) Re-ID to measure appearance variance; 2) Kalman filtering to measure position variance; and 3) post-processing with IoU to match detections and tracks. MOTR does not use explicit associations and learns the temporal variance with respect to appearance and position in an end-to-end framework. MOTR is inspired by the machine translation framework and DETR. In machine translation, the goal is to predict a sequence using sentence features and a hidden state that is iteratively updated. Similarly, we can think of the MOT framework as predicting sequences, i.e. object trajectories. Furthermore, DETR uses object queries (positional embeddings) to help predict object detections. Analogously, we can use track queries to help predict object sequences in MOT.

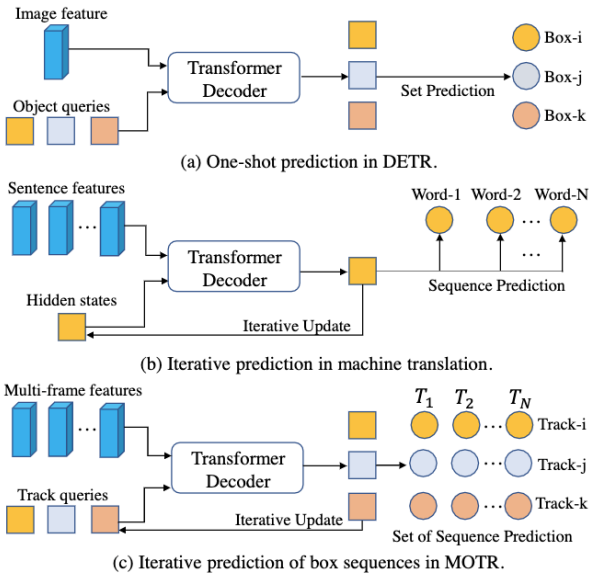

Figure 5. MOTR algorithm, inspired by DETR and machine translation. Figure taken from [18].

Combining these two approaches, the authors create the MOTR approach, where the input to a transformer decoder is the multi-frame input and track query, where each track query corresponds to an object track. MOTR then iteratively updates hidden states, which are the track queries, using self-attention and cross-attention. For each track query, MOTR then outputs a sequence prediction. There are two problems facing MOTR. First, it must learn how to track 1 object with 1 track query. This is solved using TALA (tracklet-aware label assignment) during training where detect and track queries are input to the transformer decoder and interact with each other through self-attention, which suppresses detect queries that are giving detections for already tracked objects, i.e. eliminating duplicates. The second problem is how to deal with newborn and dead objects. This is solved by updating the detect query in each frame and using it to inform the track query. Finally, a problem that frequently arises in this domain is tracking long-term temporal motion accurately. This is addressed with 2 strategies. First, the authors introduce CAL (collective average loss), which gives the overall loss of the whole video sequence normalized by the number of objects. This contrasts with previous approaches, such as the Kalman Filter, which updates parameters only using frame-by-frame loss, i.e. 2 frames, due to the Hidden Markov Model assumption. The second strategy is to use TAN (temporal aggregation network), which enhances temporal relations and provides contextual priors for track queries by using multi-head attention to aggregate historical information from previous frames. Ultimately, the authors achieve state-of-the-art results on the MOT16 dataset.

6. Conclusion

Based on the experimental results I got, as well as the ideas presented in the literature review, there are a few concrete ideas I have for how we can improve [8] to perform better on the Waymo dataset. On the surface level, we could consider using a better object detector than the PointPillars+PPBA baseline provided by the Waymo competition. One example is the CenterPoint detector [17], which has detection accuracy of 71.93% on the Waymo dataset. This is more than double the accuracy of the PointPillars+PPBA baseline, which is only 35.3%. I am currently running experiments with CenterPoint, but my cluster was slower than usual and my computing resource was not able to finish these results in time for the report.

Additionally, since we see that the algorithm's performance is highly sensitive to covariance matrix initialization and we would like to have an end-to-end trainable framework, we are interested in doing away with the Kalman filter altogether. We would like to use a transformer in an end-to-end framework just as [18]. The second idea I like from [9] is to study ways to adapt successful approaches in the 2D problem domain to our 3D problem. The main difficulty in this transition is that in 3D we only are provided with lidar point clouds, whereas in 2D we get much richer information through texture and color provided in the RGB images. Thus, an idea that comes from this line of thinking is that we could have a multi-modal input that takes the lidar point cloud information as well as the 2D camera images to learn more about our environment, as well as better integrate successful ideas from data association in 2D. Specifically, we would like to adapt the success that [18] had with MOTR in 2D to work well with the 3D scenario.

References

[1] K. Bernardin, A. Elbs, and R. Stiefelhagen. Multiple object tracking performance metrics and evaluation in a smart room environment. In *Sixth IEEE International Workshop on Visual Surveillance, in conjunction with ECCV*, volume 90. Citeseer, 2006.

[2] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020.

[3] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020.

[4] S. Chang, W. Li, Y. Zhang, and Z. Feng. Online siamese network for visual object tracking. *Sensors*, 19(8):1858, 2019.

[5] J. P. Chen, F. Obermeyer, V. Lyapunov, A. Uber, L. Gueguen, and N. Goodman. Joint triangulation and mapping via differentiable sensor fusion.

[6] S. Cheng, Z. Leng, E. D. Cubuk, B. Zoph, C. Bai, J. Ngiam, Y. Song, B. Caine, V. Vasudevan, C. Li, et al. Improving 3d object detection through progressive population based augmentation. In *European Conference on Computer Vision*, pages 279–294. Springer, 2020.

[7] H.-k. Chiu, J. Li, R. Ambrus, and J. Bohg. Probabilistic 3d multi-modal, multi-object tracking for autonomous driving. *arXiv preprint arXiv:2012.13755*, 2020.

[8] H.-k. Chiu, A. Prioletti, J. Li, and J. Bohg. Probabilistic 3d multi-object tracking for autonomous driving. *arXiv preprint arXiv:2001.05673*, 2020.

[9] M.-Q. Dao and V. Frémont. A two-stage data association approach for 3d multi-object tracking. *Sensors*, 21(9):2894, 2021.

[10] A. Kloss, G. Martius, and J. Bohg. How to train your differentiable filter. *Autonomous Robots*, pages 1–18, 2021.

[11] L. Leal-Taixé, C. Canton-Ferrer, and K. Schindler. Learning by tracking: Siamese cnn for robust target association. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 33–40, 2016.

[12] J. Pöschmann, T. Pfeifer, and P. Protzel. Factor graph based 3d multi-object tracking in point clouds. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10343–10350. IEEE, 2020.

[13] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2446–2454, 2020.

[14] X. Weng and K. Kitani. A baseline for 3d multi-object tracking. *arXiv preprint arXiv:1907.03961*, 1(2):6, 2019.

[15] X. Weng, Y. Wang, Y. Man, and K. M. Kitani. 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*, pages 6499–6508, 2020.

[16] B. Yi, M. Lee, A. Kloss, R. Martín-Martín, and J. Bohg. Differentiable factor graph optimization for learning smoothers. *arXiv preprint arXiv:2105.08257*, 2021.

[17] T. Yin, X. Zhou, and P. Krahenbuhl. Center-based 3d object detection and tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11784–11793, 2021.

[18] F. Zeng, B. Dong, T. Wang, X. Zhang, and Y. Wei. Motr: End-to-end multiple-object tracking with transformer. *arXiv preprint arXiv:2105.03247*, 2021.

[19] Y. Zhang, P. Sun, Y. Jiang, D. Yu, Z. Yuan, P. Luo, W. Liu, and X. Wang. Bytetrack: Multi-object tracking by associating every detection box. *arXiv preprint arXiv:2110.06864*, 2021.

[20] B. Zhu, Z. Jiang, X. Zhou, Z. Li, and G. Yu. Class-balanced grouping and sampling for point cloud 3d object detection. *arXiv preprint arXiv:1908.09492*, 2019.