Self-Driving Car Final Competition

Team BNN:
Sheng Cheng Lee
Jung Han Chen
Chao Chun Hsu

Introduction

3D multi-object tracking is essential for autonomous driving. Its aim is to estimate the location, orientation, and scale of all the objects in the environment over time. By taking temporal information into account, a tracking module can filter outliers in frame-by-frame object detectors and be robust to partial or full occlusions. Thereby, it promises to identify the trajectories of different categories of moving objects, such as pedestrians, bicycles, and cars. The resulting trajectories may then be used to infer motion patterns and driving behaviors for improved forecasting. This in turn helps planning to enable autonomous driving.

In this competition, we approach the 3D multi-object tracking problem with a **Kalman Filter**. We model the state of each object with its 3D position, orientation and scale as well as linear and angular velocity. For the prediction step, we use a process model with **constant velocity**. For the update step, we consider the detections provided by a baseline tracker as measurement.

For data association between the predict and actual object detection, we use two approaches: (i) **Mahalanobis distance**; (ii) **2D Intersection-Over-Union (2D-IOU)**. Differences between these two approaches are that Mahalanobis distance takes into account the uncertainty about the predicted object state as provided by the Kalman Filter in form of the state covariance matrix. Moreover, the Mahalanobis distance can provide distance measurement even when prediction and detection do not overlap. In this case, the 2D-IOU gives zero which prevents any data association. The results of these two approaches will be compared below.

Correctly choosing the initial state and noise covariance matrices is fundamental for filter convergence. We extract the statistics of the training data to perform this initialization. This also ensures that our experiments on the validation and test set do not use any future or ground-truth information.

Tracking Architecture

We use 3D object detection results as measurements. At each timestep, we use two approaches: (i) **Mahalanobis distance**; (ii) **2D Intersection-Over-Union (2D-IOU)** to compute the distance between object detections and predictions. Given this distance, we perform data association. The Kalman Filter then updates the current state estimates. It uses a **constant velocity model** for predicting the mean and covariance of the state in the next time step.

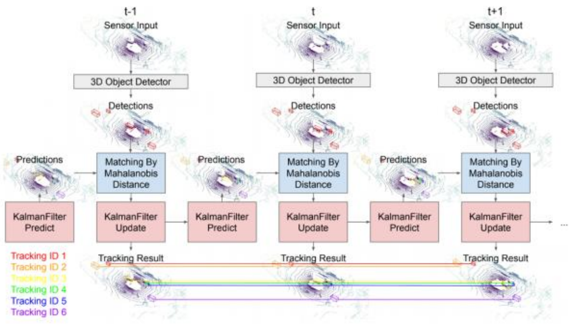


Figure from: Hsu-kuang Chiu, Antonio Prioletti, Jie Li, and Jeannette Bohg. Stanford University, Toyota Research Institute. Probabilistic 3D Multi-Object Tracking for Autonomous Driving

Detection

The precomputed 3D detections were computed on the Argoverse dataset using the method described in Class-balanced Grouping and Sampling for Point Cloud 3D Object Detection, with detection range increased to 100 meters in each direction and pruned to ROI to match Argoverse annotation policy.

Overall network architecture is presented in Figure below, which is mainly composed of 4 parts: (i) **Input Module**; (ii) **3D Feature Extractor**; (iii) **Region Proposal Network**; (iv) **Multi-group Head network**. Together with improvements on data augmentation, loss function, and training procedure, we not only make it perform 10 categories' 3D object detection, velocity and attribute prediction simultaneously, but also achieve better performance than perform each category's detection respectively.

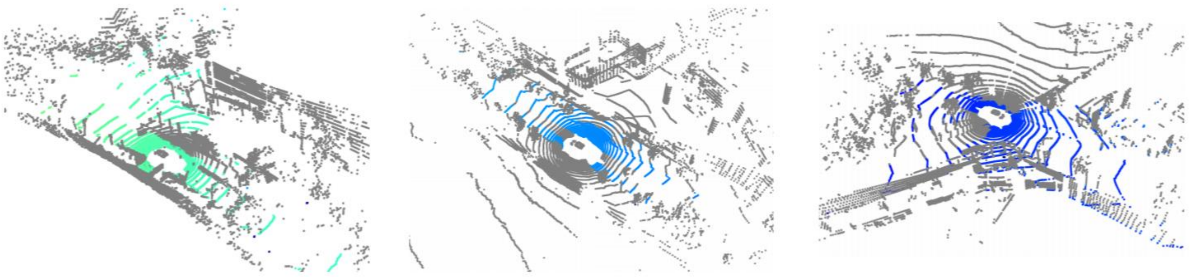


Figure from: Benjin Zhu , Zhengkai Jiang, Xiangxin Zhou, Zeming Li, and Gang Yu Megvii. Research Institute of Automation, Chinese Academy of Sciences Tsinghua University. Class-balanced Grouping and Sampling for Point Cloud 3D Object Detection

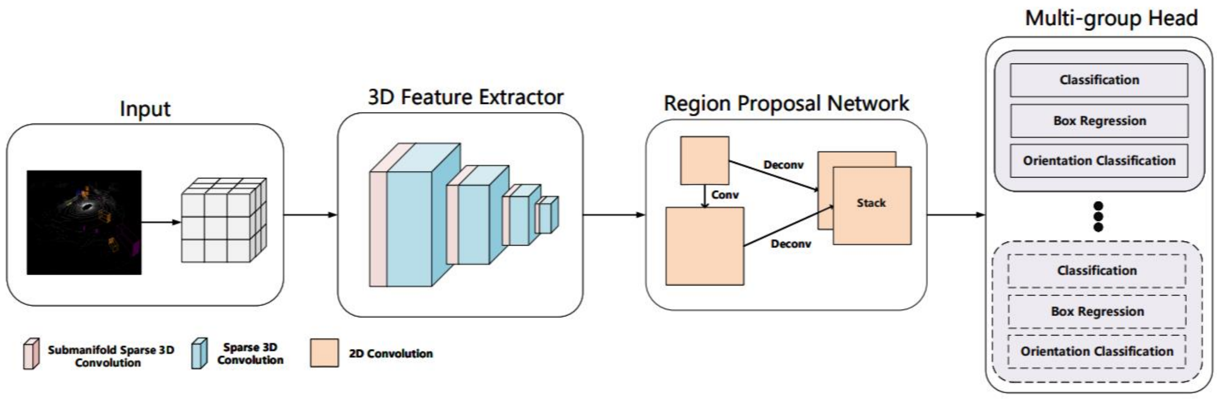


Figure: Network Architecture. 3D Feature Extractor is composed of submanifold and regular 3D sparse convolutions. Outputs of 3D Feature Extractor are of 16× downscale ratio, which are flatten along output axis and fed into following Region Proposal Network to generate 8× feature maps, followed by the multi-group head network to generate final predictions. Number of groups in head is set according to grouping specification.

Covariance

Since the provided baseline method do not adjust the covariance of the Kalman Filter, the resulted output is apparently not the best result this method can achieve. We utilize the covariance values found by the team, whose method got the first place in NuScenes tracking competition, to adjust the uncertainty in KF, which did not get a notable improvement.

The following video is recorded using the visualization tool by TA which shows the same scene in the testing data.

Baseline Not Modified

Baseline with Modified Covariance

Baseline with Different Covariances for Car and Pedestrian

Data Association

In data association, we need to design a data association mechanism to decide which detection to pair with a predicted object state and which detections to treat as outliers. In argo baseline, they adopt 2D-IOU to measure the affinity between predictions and detections. However, 2D-IOU has some drawbacks, especially when the tracked object is in small size, such as pedestrians. For pedestrians, their 2D-IOU could be 0 even if the prediction and the detection are very close but do not overlap. In our approach, we replace 2D-IOU method by using Mahalanobis distance. This distance m measures the difference between predicted detections and actual detections weighted by the uncertainty about the prediction as expressed through the innovation covariance.

$$m = \sqrt{(\mathbf{o}_{t+1} - \mathbf{H}\hat{\mu}_{t+1})^T \mathbf{S}_{t+1}{}^{-1} (\mathbf{o}_{t+1} - \mathbf{H}\hat{\mu}_{t+1})}.$$

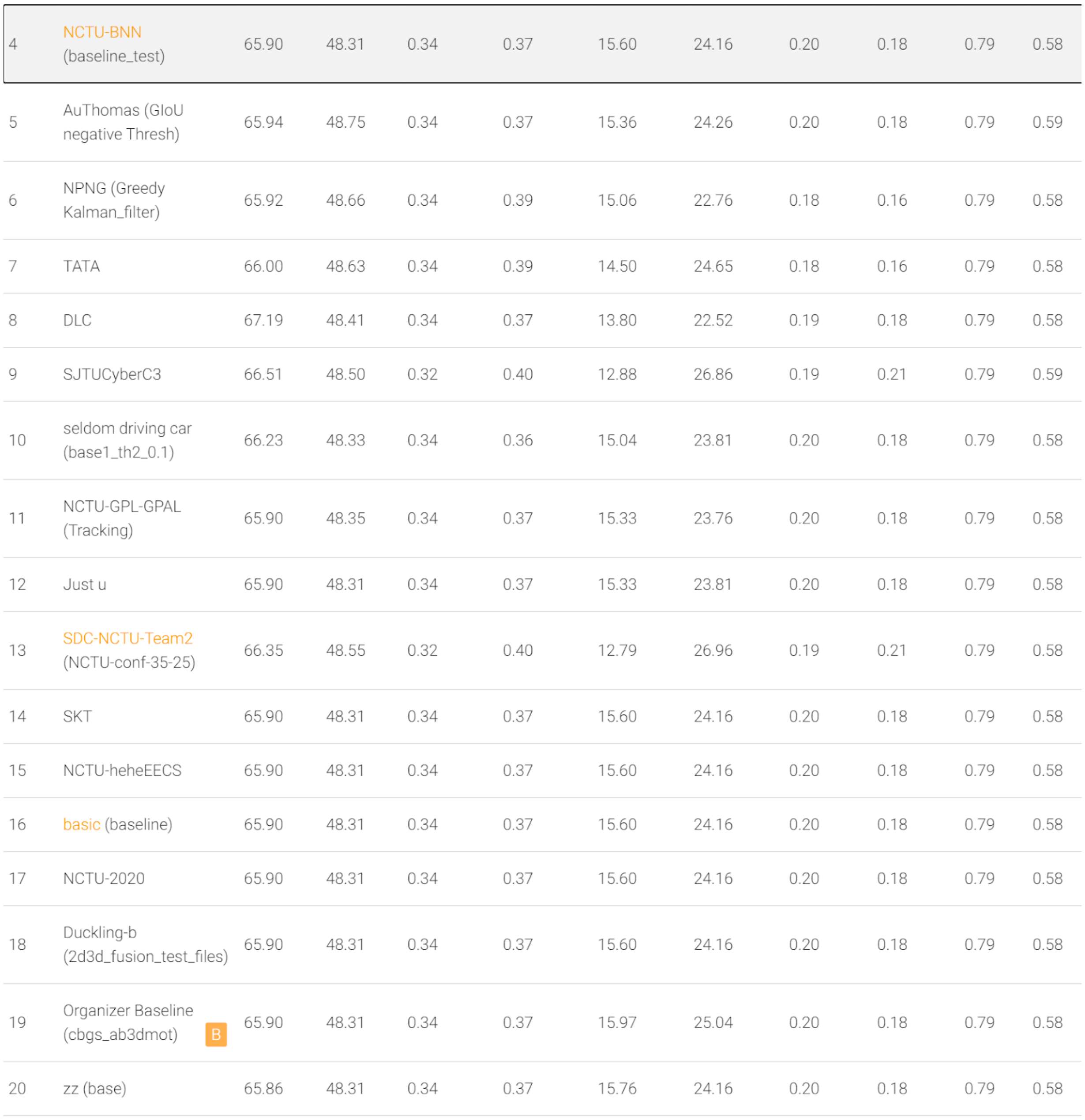
Given the distances between all predictions and detections, we solve a bipartite matching problem to find the optimal pairing. We have tried two method, greedy algorithm and Hungarian algorithm. However, we compared two method, the two results were almost similar. Finally, we use Hungarian algorithm and set the suitable distance threshold.

Links below show our result using two different threshold in Hungarian algorithm. It is obvious that the result effects a lot by choosing the reasonable threshold.

Threshold=2

Threshold=10

Best Result



4	NCTU-BNN (baseline_test)	65.90	48.31	0.34	0.37	15.60	24.16	0.20	0.18	0.79	0.58
5	AuThomas (GIoU negative Thresh)	65.94	48.75	0.34	0.37	15.36	24.26	0.20	0.18	0.79	0.59
6	NPNG (Greedy Kalman_filter)	65.92	48.66	0.34	0.39	15.06	22.76	0.18	0.16	0.79	0.58
7	TATA	66.00	48.63	0.34	0.39	14.50	24.65	0.18	0.16	0.79	0.58
8	DLC	67.19	48.41	0.34	0.37	13.80	22.52	0.19	0.18	0.79	0.58
9	SJTUCyberC3	66.51	48.50	0.32	0.40	12.88	26.86	0.19	0.21	0.79	0.59
10	seldom driving car (base1_th2_0.1)	66.23	48.33	0.34	0.36	15.04	23.81	0.20	0.18	0.79	0.58
11	NCTU-GPL-GPAL (Tracking)	65.90	48.35	0.34	0.37	15.33	23.76	0.20	0.18	0.79	0.58
12	Just u	65.90	48.31	0.34	0.37	15.33	23.81	0.20	0.18	0.79	0.58
13	SDC-NCTU-Team2 (NCTU-conf-35-25)	66.35	48.55	0.32	0.40	12.79	26.96	0.19	0.21	0.79	0.58
14	SKT	65.90	48.31	0.34	0.37	15.60	24.16	0.20	0.18	0.79	0.58
15	NCTU-heheEECS	65.90	48.31	0.34	0.37	15.60	24.16	0.20	0.18	0.79	0.58
16	basic (baseline)	65.90	48.31	0.34	0.37	15.60	24.16	0.20	0.18	0.79	0.58
17	NCTU-2020	65.90	48.31	0.34	0.37	15.60	24.16	0.20	0.18	0.79	0.58
18	Duckling-b (2d3d_fusion_test_files)	65.90	48.31	0.34	0.37	15.60	24.16	0.20	0.18	0.79	0.58
19	Organizer Baseline (cbgs_ab3dmot) B	65.90	48.31	0.34	0.37	15.97	25.04	0.20	0.18	0.79	0.58
20	zz (base)	65.86	48.31	0.34	0.37	15.76	24.16	0.20	0.18	0.79	0.58

Conclusion

By running the baseline code and modify the method of data association, we had more sense in tracking compared to the in-class knowledge. Although the approaches we implemented didn’t come out good, but we have a strong conceptual understanding of the tracking pipeline and also tried our best to optimize the tracking mission. There are also lots of algorithms and methods that can be implemented in tracking. We may keep searching and implementing the new approaches until the competition is ended.

Bonus

Since TA provides the NCTU dataset, we gave it a try and found that by only utilize the ab3dmot method, we can get a decent result. The reason may be that the precomputed detection data was actually manually labelled.

First Scene

Second Scene