3D Object Detection and Tracking on Streaming Data

Xusen Guo[1] and Kai Huang[2]

Abstract— **Recent approaches for 3D object detection have made tremendous progress due to the development of neural networks. However, previous researches are mostly single frame based, information between frames is scarcely explored. In this paper, we attempt to leverage the temporal information in streaming data and explore 3D object detection as well as tracking based on multiple frames. Towards this goal, we set up a ConvNet architecture that can associate multi-frame images and multi-frame point clouds to generate accurate 3D detections and trajectories in an end-to-end form. Specifically, a tracking module is introduced to capture objects co-occurrences across time, and a multi-task objective for frame-based object detection and across-frame track regression are used. Our proposed architecture is proven to produce competitive results on the KITTI Object Tracking Benchmark, with 72.21% in MOTA and 82.29% in MOTP respectively.**

I. INTRODUCTION

3D Object detection has received increasing attention over the last few years due to the rapid development of autonomous driving. Compared to 2D image, 3D information can provide accurate localization of targets and characterize their shapes. Current approaches for 3D object detection are mostly carried out in three fronts: image based[1], [2], point clouds based[3], [4], [5], and multi-view fusion based[6], [7]. Most of these approaches have achieved competitive results but are limited to single frame input.

During autonomous driving, data are always generated in a streaming fashion and thus it is more natural to perform object detection from streaming data. Compared to single-frame based approaches, streaming data can provide consistent temporal correlations between consecutive frames for detected features, which can reduce detection noise over time. In addition, truncated and occluded targets can possibly be compensated by subsequent frames within streaming data. Therefore, exploring 3D object detection methods specifically for streaming data is essential and promising.

Performing 3D object detection in streaming data is however complex. First of all, acquiring consistent 3D information between frames is difficult. On the one hand, camera data provide rich appearance features but lack of depth information. On the other hand, though LiDAR can accurately detect the position of object of interest, it is very sparse and thus difficult to determine the appearance of object purely from its point cloud representation. Second, how to correlate the features between individual frames is not obvious. For example, generating 3D scene flow with temporal feature representation will need to determine the corresponding points between frames, which is not straightforward and challenging. Last but not least, the sheer numbers of frames that streaming data provided introduce unaffordable computational costs for frame level detection.

In this paper, we propose a **D**ual-way **O**bject **D**etection and **T**racking (**DODT**) network to tackle the aforementioned problem, its structure is illustrated in *Figure* 1. The network has a RPN module to generate 3D proposals and two detection branches to perform 3D object detection on two adjacent keyframes respectively. To compensate for the sparsity of point cloud data, the network aggregates image features and thus can utilize the strengths of both. Considering the redundancy of features between frames, the 3D proposals are shared by two detection branches. In order to avoid estimating 3D scene flow directly when computing object cross-correlation features, a tracking module is aided to our network for temporal feature encoding. The tracking module uses correlation operation to extract temporal features and predict object displacements over keyframes, but different from traditional ways which performed on whole feature maps, it does correlation on object proposal-level with the help of sharing mechanism in proposals.

For a fast inference speed, we only performs object detection on keyframes and propagate predicted bounding boxes to neighboring frames for a streaming-based detection. By linking detections over time, multi-object tracking can be finished through *tracking by detection* [8]. Note that tracking using two frames often suffers from many problems, such as drift, loss of targets in one frame, etc. We develop a interpolation algorithm driven by motion model to address these problems. The ablation study shows the effectiveness of our interpolation algorithm.

In summary, our contributions are threefold: *(i)* we set up a dual-way network for 3D streaming-based object detection and multi-object tracking in autonomous driving scenarios. *(ii)* Instead of encoding temporal feature using 3D scene flow, we introduce a tracking module to compute convolutional cross-correlation of adjacent frames for temporal feature representation. Our model performs correlation operation on proposal-level, which is much efficient. *(iii)* We develop a interpolation algorithm driven by motion model to solve typical problems in object tracking using two frames. We perform our tracking approach to KITTI Object Tracking Benchmark and obtain competitive results, with 72.21% in MOTA and 82.29% in MOTP respectively.

II. RELATED WORK

3D object detection. Currently, most approaches in 3D object detection can be divided into three types: image based detectors, point cloud based detectors, and fusion based detectors. Images based approaches such as Mono3D [1] and 3DOP [2] use camera data only. Since image lacks depth information, hand-crafted geometric features are required in

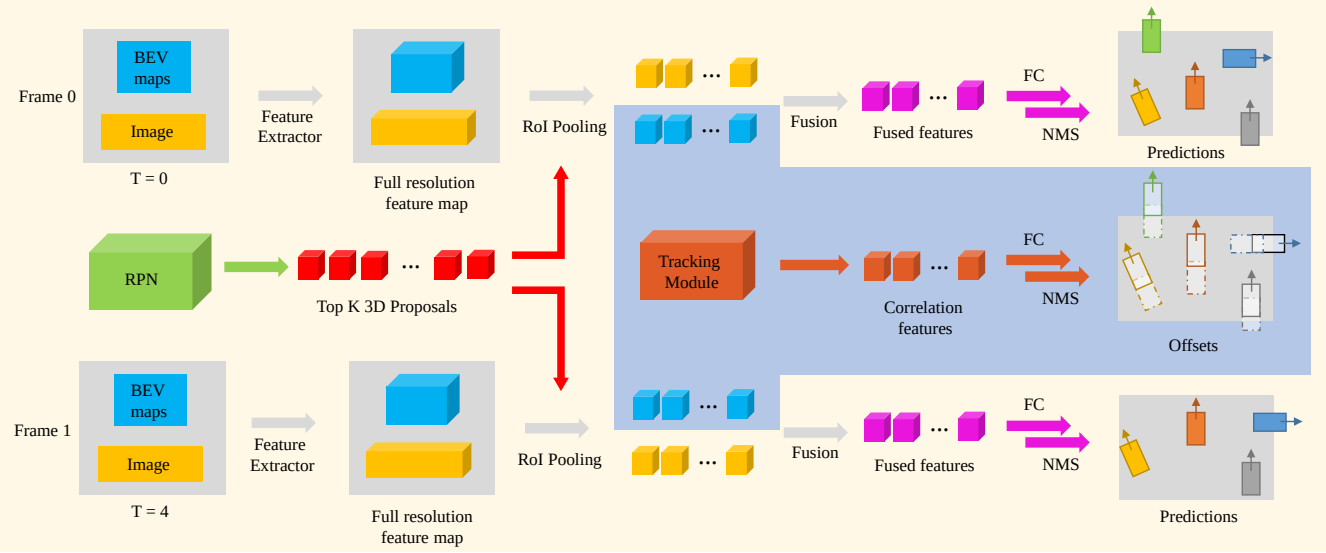

Fig. 1: DODT architecture.

these approaches. Point cloud based methods are usually done in two fronts: voxelization based and projection based, according to how point cloud features is represented. Voxelization based methods utilize a voxel grid representation to encoder the point cloud and then apply 3D convolution for feature extraction. These approaches include 3D FCN [9], Vote3Deep [10], VoxelNet [3], etc. These approaches suffer from the sparsity of point cloud and enormous computation costs in 3D convolution. While projection based methods attempt to project point cloud to a perspective view (e.g. bird eye view) and apply image-based feature extraction techniques, such as PIXOR [4], FaF [11], Comple-YOLO [5], etc. These methods take advantage of the fact that 3D detections in driving scenes are almost on the same plane, thus loss of height information has little influence on performance. However, due to the sparsity of point cloud, features after projection are insufficient for accurate object detection, especially for small targets.

Fusion based approaches such as F-PointNet [12] first extracts the 3D bounding frustum of an object by extruding 2D bounding boxes from image detectors, then consecutively performs 3D object instance segmentation and amodal extent regression to estimate the amodal 3D bounding box. This method works well for indoor scenes and brightly lit outdoor scenes, but are expected to perform poorly in more extreme outdoor scenarios. MV3D [6] extends the image based RPN of Faster R-CNN[13] to 3D and proposes a 3D RPN, then applies feature fusion of images and point clouds to produces accurate 3D detections. However, due to the insufficient information in feature extraction caused by downsampling, it does not work well for small targets. AVOD [7] is similar to MV3D in 3D RPN and feature fusion, but with full resolution feature maps produced by a pyramid architecture, which leads to a great improvement in localization accuracy for small targets. Our detection submodule is somehow similar to AVOD, but enhances proposal's feature with temporal information additionally.

Video object detection. Nearly all existing methods in video object detection incorporate temporal information on either feature level or final box level. FGFA [14] leverages temporal coherence on feature level. The network first applies feature extraction network on individual frames to produce per-frame feature maps, and then enhances features at a reference frame by warping the nearby frames feature maps according to flow emotion. On the other hand, final box level approaches usually utilize temporal information in bounding box post processing. T-CNN [15], [16] leverages precomputed optical flows to propagate predicted bounding boxes to neighboring frames, and then generates tubelets by applying tracking algorithms from high-confidence bounding boxes. Seq-NMS [17] improves NMS algorithm for video by constructing sequences along nearby high-confidence bounding boxes from consecutive frames. While boxes of the sequence are then re-scored to the average confidence and other boxes close to this sequence are suppressed.

Other approaches attempt to learn temporal features between consecutive frames to avoid using optical flow. D&T [18] presents an end-to-end fully convolutional architecture, it uses a detection and tracking based loss for simultaneous detection and tracking in video. In order to learn temporal information representation, the network is fed with multiple frames, and a correlation module is embedded for computing convolutional cross-correlation between frames. Our DODT approach is mainly inspired by D&T, however, we develop this idea to 3D space. Moreover, we constrain correlation operation in proposal-level, which reduce computational costs significantly.

3D multi-object tracking. Existing 3D multi-object tracking approaches are mostly implemented based on tracking by detection. For example, FaF [11] jointly reasons about 3D detection, tracking and motion forecasting taking a 4D tensor created from multiple consecutive temporal frames. It can aggregate the detection information for the past n

timestamps to produce accurate tracklets. DSM [19] first predicts 3D bounding boxes in continuous frames and then associates detections using a *Matching net* and a *Scoring net*, which is similar to our approach. However, their 3D detector is directly single frame based approach MV3D[6], temporal features between frames are mostly ignored. Moreover, their bounding boxes association is done by solving a linear program and is a offline version, while our tracking algorithm is a online approach.

III. METHODOLOGY

In this section we first give an overview of our DODT approach (Sec. A) that generates 3D object detection and tracking results given two adjacent keyframes as inputs. We then introduce the RPN module (Sec. B) that predicts 3D proposals shared by two detection branches. Sec. C shows how tracking module encodes object co-occurrences features and predicts the displacement of corresponding targets in two adjacent keyframes. Sec. D shows how we implement interpolation algorithm and complete 3D streaming object detection and multi-object tracking.

A. DODT Model Structure

We aim at performing 3D object detection and tracking on streaming data. To this end, we design DODT architecture into a dual-way network. *Figure* 1 illustrates the whole architecture. By doubling its inputs, we can feed two adjacent keyframes data simultaneously. The keyframes data consists of an image and point cloud BEV maps (following the procedure described in MV3D[6]). Keyframe data are first fed to feature extractors to get corresponding feature maps respectively. We design the feature extractor following the procedure described in AVOD[7], which constructed in an encoder-decoder version resulting in a full resolution feature map. To extract feature crops for every anchor from image and BEV view feature maps, we also follow the idea proposed in AVOD. Given an 3D anchor generated by RPN, two view specific ROIs are obtained by projecting the anchor onto the BEV and image feature maps, the corresponding feature crops are extracted and performed RoI pooling from two views. After that, a *early fusion* scheme is used to combine multi-view features in proposal-level. Finally, after fully connected layers and the non-maximum suppression (NMS) algorithm, the final detection outputs are produced. Meanwhile, the tracking module uses BEV feature crops of two branches, and performs correlation operation to corresponding crops to produce correlation features. The correlation features are then used to predict proposal-level offsets for streaming-level detection and tracking.

The whole network is designed in an end-to-end form, and the multi-task loss is consist of a classification loss L_{cls}, a regression loss L_{reg} and a correlation loss L_{corr}. L_{corr} scores the displacement regression between corresponding objects across two frames. L_{reg} and L_{corr} are normalized by the number of proposals while L_{cls} is normalized only by positive proposals.

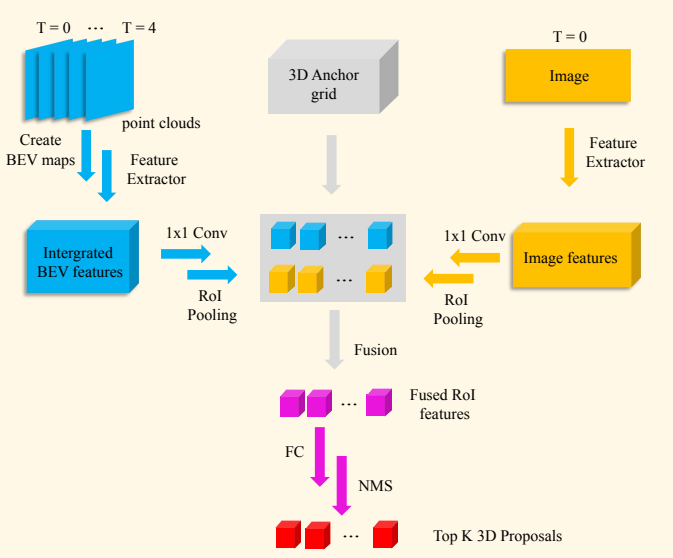

Fig. 2: RPN structure.

B. RPN Module

We develop a RPN network that can generated 3D proposals to both detection branches. The structure is shown in *Figure* 2. To ensure proposals predicted by our RPN are suitable for both keyframes, we create BEV feature maps based on two keyfames as well as their intermediate frames. In this case is 5 frames. Note that point cloud shows object accurate 3D localization, we can simply transform 5 frames to a same coordinate system to fuse them. Since point cloud is extremely sparse and is encoded by projection, this process does not increase any computational cost but enrich the point cloud features. Due to the motion, the location of the same object in 5 frames are different. For training convenient, we compute a axis-aligned label to replace origin 5 labels in 5 frames. *Figure* 3 illustrates the relationship of two labels. We also fused image data to further enrich appearance features. Since one image can contain enough features in a short temporal slice, we use first frame only for image branch. Feature extractor modules, RoI Pooling component, fusion scheme, FC layers and NMS algorithm are all similar to detection branches. While 1x1 convolutional layer before RoI Pooling works as dimensionality reduction, which helps RPN module reduce computational requirements.

C. Tracking Module

Our 3D correlation module is illustrated in *Figure* **??**. Given feature maps of two adjacent key frames, it first performs correlation operation to compute convolutional cross-correlation. Similar to RPN sub-module in AVOD, the module then extracts feature crops via multiview *Crop and Resize* operations guided by aforementioned *Top K* non-oriented region proposals. The feature crops are then fed to a fusion module to generate multiview aggregated features. The fusion module is identical to the one involved in AVOD, which was first introduced in MV3D[6]. The aggregated features are then passed to a fully connected layer for

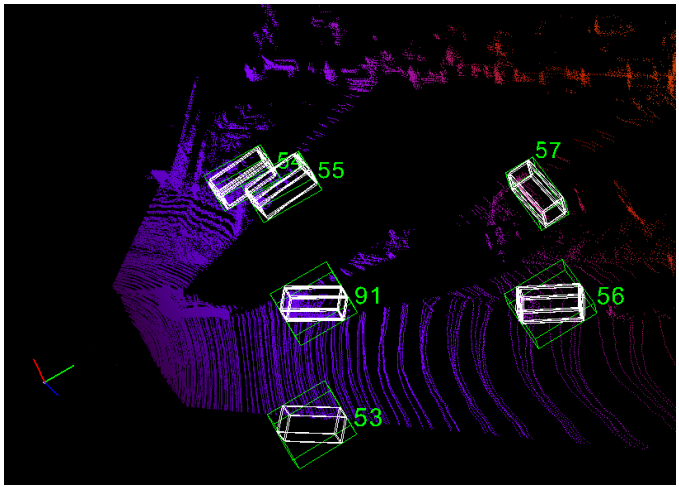

Fig. 3: 5 consecutive point clouds in the same coordinate system. White boxes are ground true labels in 5 frames, while green boxes are the axis-aligned new labels for RPN training. The numbers are object id.

regression. After that, non-maximum suppression (NMS) is applied to ignore redundant, overlapping bounding boxes for the final loss calculation.

Similar to FlowNet [21], we restrict correlation operation to a local neighborhood instead of all possible circular shifts in a feature map. This restriction helps the module avoid large output dimensionality. The correlation operation performs point-wise feature comparison of two feature maps f_t, $f_{t+\tau}$ by

$$\mathcal{C}^{t,t+\tau}(i,j,p,q) = \Big\langle f_t(i,j), f_{t+\tau}(i+p, j+q) \Big\rangle \quad (1)$$

where $p, q \in [-d, d]$ are offsets to compared features in a local square window defined by the maximum displacement d, and i, j are the location of the window center in a feature map. The output is a correlation feature map of size $\mathcal{C} \in \mathbb{R}^{h \times w \times (2d+1) \times (2d+1)}$, where h, w are the height and the width of the feature map.

After applying (1), two correlation feature maps are obtained, one for point cloud $\mathcal{C}_{pc}^{t,t+\tau}$, and one for RGB image $\mathcal{C}_{img}^{t,t+\tau}$. ROI pooling and feature fusion are then performed to produce aggregate feature map $\mathcal{C}_{fusion}^{t,t+\tau} = fusion(\mathcal{C}_{pc}^{t,t+\tau}, \mathcal{C}_{img}^{t,t+\tau})$, which is then flattened and fed to a fully connected layer to predict the transformation $\Delta^{t,t+\tau} = (\Delta_x^{t,t+\tau}, \Delta_y^{t,t+\tau}, \Delta_z^{t,t+\tau}, \Delta_r^{t,t+\tau})$ of the RoIs from t to $t+\tau$. We hold the prior that the size of a target does not change over time, thus the network only needs to predict the variations in the center coordinates and steering angle.

D. 3D Streaming Object Detection and Tracking

Given a sequence of N frames $\{I_f \mid f = 1, ..., N\}$, streaming-based object detection task needs to predict a set of detections D_f for each frame I_f, where $D_f = \{d_f^i \mid i = 1, ...N_f\}$, d_f^i is i^{th} target and N_f is the number of targets in frame f. Note that D_f can also be an empty set when no target is detected in a frame. In 3D object detection, each detection d_i is parametrized as $D_i = (x_i, y_i, z_i, w_i, h_i, l_i, \theta_i, s_i)$, where (x_i, y_i, z_i) corresponds to the coordinate of target center, (w_i, h_i, l_i) corresponds to width, height, length of the target, θ_i the rotation angle in yaw axis and s_i prediction confidence of the target.

We only perform object detection in key frames due to the redundant features in streaming data. Detections of the intermediate frame can be determined leveraging the detection results of its adjacent key frames. Suppose we have two predicted detections set $(D_f, D_{f+\tau})$ of two consecutive key frames $(I_f, I_{f+\tau})$, where τ is temporal stride, the detection result d_{f+t}^i of intermediate frame $I_{f+t} (t \in \{1, \tau - 1\})$ can be obtained by

$$d_{f+t}^i = \mathcal{F}(W_f d_f^i, W_{f+\tau} d_{f+\tau}^i) \quad (2)$$

where $W_f, W_{f+\tau}$ are the corresponding weight coefficients and $\mathcal{F}$ is interpolation function. Note that we can use Equation (2) to generate d_{f+t}^i only when the target exists in D_f and $D_{f+\tau}$ simultaneously. If a target is the start or end of a trajectory in intermediate frames, this method will fail. Although carefully selecting key frames could fix this problem, it is beyond the scope of this work. In this paper we focus on the targets that always exist between two key frames.

For multi-object tracking, we attempt to assign each detection in each frame to a unique trajectory $T_k = \{d_{f_1}^k, d_{f_2}^k, ..., d_{f_{N_k}}^k\}$, utilizing an improved IOU tracker algorithm[20] (detailed description is available in supplementary material). Where k is the trajectory id and N_k is the length of T_k. Unlike multi-object tracking in 2D image which suffers from boxes overlap, detection in 3D has its unique position, any overlap of two detections in 3D means high probability of the same target. Thus a IOU based data association algorithm can also work well in our approach.

IV. EXPERIMENTS

A. Datasets and Training

Datasets. We use KITTI object tracking Benchmark [33] for evaluation. It consists of 21 training sequences and 29 test sequences with vehicles annotated in 3D. Each sequence includes hundreds of point cloud frames captured by Velodyne HDL-64E rotating 3D laser scanner and corresponding RGB images. We split 21 training sequences into two parts according to their sequence number, odd numbered sequences for training datasets and even numbered ones for evaluation datasets. For multi-object tracking evaluation, we train our model in all 21 training sequences.

Datasets preprocessing. Similar to the data preprocessing in [7], we crop point clouds at $[-40, 40] \times [0, 70] \times [0, 2.5]$ meters along Y, X, Z axis respectively to contain points within the field of view of the camera. In KITTI tracking datasets, the observer is an autonomous vehicle, thus the coordinate system of consecutive frames shift due to the movement of the observer over time. Since the location and velocity information of the observer are available in IMU data, one can calculate the displacement of the observer

Method	Runtime (s)	Class	$AP_{3D}(\%)$ Easy	Moderate	Hard	$AP_{BEV}(\%)$ Easy	Moderate	Hard
AVOD[7]	0.08		75.24	55.11	48.58	89.68	72.68	72.66
Bi-AVOD	0.20		81.17	54.96	53.59	90.87	72.68	72.65
Bi-AVOD*	0.20		**83.77**	58.31	57.12	**90.88**	81.71	72.68
Bi-AVOD* (τ = 1)	0.10	Car	77.07	63.43	63.04	90.86	81.76	81.75
Bi-AVOD* (τ = 3)	0.07		77.89	**63.80**	**63.50**	90.83	**81.77**	**81.76**
Bi-AVOD* (τ = 5)	0.04		77.26	62.30	55.79	90.80	72.69	72.67
Bi-AVOD* (τ = 7)	**0.03**		74.85	54.36	53.60	81.77	72.60	72.57

TABLE I: A comparison of the performance between original AVOD, our Bi-AVOD and fine-tuned Bi-AVOD* with different temporal stride τ on our KITTI tracking evaluation datasets.

Method	MOTA(%)	MOTP(%)	MT(%)	ML(%)	IDS	FRAG
AVOD[7]	58.59	81.62	42.44	31.51	**5**	166
Bi-AVOD(ours)	**78.90**	**84.22**	**70.59**	**5.04**	31	**123**

TABLE II: Tracking performance comparison of origin AVOD and our Bi-AVOD on KITTI tracking evaluation datasets.

Method	MOTA(%)	MOTP(%)	MT(%)	ML(%)	IDS	FRAG
CEM[23]	51.94	77.11	20.00	31.54	125	396
RMOT[24]	52.42	75.18	21.69	31.85	50	376
TBD[25]	55.07	78.35	20.46	32.62	31	529
mbodSSP[26]	56.03	77.52	23.23	27.23	**0**	699
SCEA[27]	57.03	78.84	26.92	26.62	17	461
SSP[26]	57.85	77.64	29.38	24.31	7	704
ODAMOT[28]	59.23	75.45	27.08	15.54	389	1274
NOMT-HM[29]	61.17	78.65	33.85	28.00	28	**241**
LP-SSVM[30]	61.77	76.93	35.54	21.69	16	422
RMOT*[24]	65.83	75.42	40.15	9.69	209	727
NOMT[29]	66.60	78.17	41.08	25.23	13	150
DCO-X*[31]	68.11	78.85	37.54	14.15	318	959
mbodSSP*[26]	72.69	78.75	48.77	8.77	114	858
SSP*[26]	72.72	78.55	53.85	**8.00**	185	932
NOMT-HM*[29]	75.20	80.02	50.00	13.54	105	351
SCEA*[27]	75.58	79.39	53.08	11.54	104	448
MDP[32]	**76.59**	82.10	52.15	13.38	130	387
Bi-AVOD(ours)	72.21	**82.29**	**54.61**	15.38	113	523

TABLE III: Tracking performance comparison of publicly available methods in the KITTI Tracking Benchmark.

between different frames and translate the coordinates accordingly. In this way both point clouds and objects labels are on the exact same coordinate system. Note that this transform is important to make the system invariant to the speed of the ego-car.

Training and testing. We train our network for *Car* category only. We follow most of the super-parameter settings in [7] during training and testing. The network is trained for 120K iterations using an ADAM[34] optimizer with an initial learning rate of 0.0001 that is decayed exponentially every 30K iterations with a decay factor of 0.8. During proposal generation, anchors with IoU less than 0.3 are considered background and greater than 0.5 are object. To remove redundant proposals, 2D NMS is performed at an IoU threshold of 0.8 in BEV to keep the top 1024 proposals during training, while at inference time, the top 300 proposals are kept.

B. Results

3D object detection. Both streaming level detection and multi-object tracking require accurate detection results, thus the performance of the network on 3D object detection is significant. We train our Bi-AVOD on our tracking training datasets, and results are evaluated using KITTI object detection evaluation metrics. Since evaluate our model on KITTI tracking testing datasets for 3D object detection is hard, we turn to our evaluation datasets (described in Sec 4.1) instead. To explore the effectiveness of dual-way structure on object detection, we also train original AVOD model on our training datasets. The comparison results on 3D object detection are shown in *Table* I. Our Bi-AVOD achieves 81.15% AP_{3D} in *easy* setting and 53.59% AP_{3D} in *hard* setting, outperforms original AVOD by 5.93% and 5.01% respectively. This gain show that the introduction of dual-way structure and correlation module contributes to 3D object detection significantly. For better performance, we train original AVOD model on KITTI object detection datasets, and then transfer relevant parameters to our Bi-AVOD model for further fine-tuned learning on KITTI tracking datasets. In this way a better performance is obtained. Results are shown in *Table* I, where Bi-AVOD* is the fine-tuned model. Compared to Bi-AVOD which is trained from scratch, the fine-tuned model raises

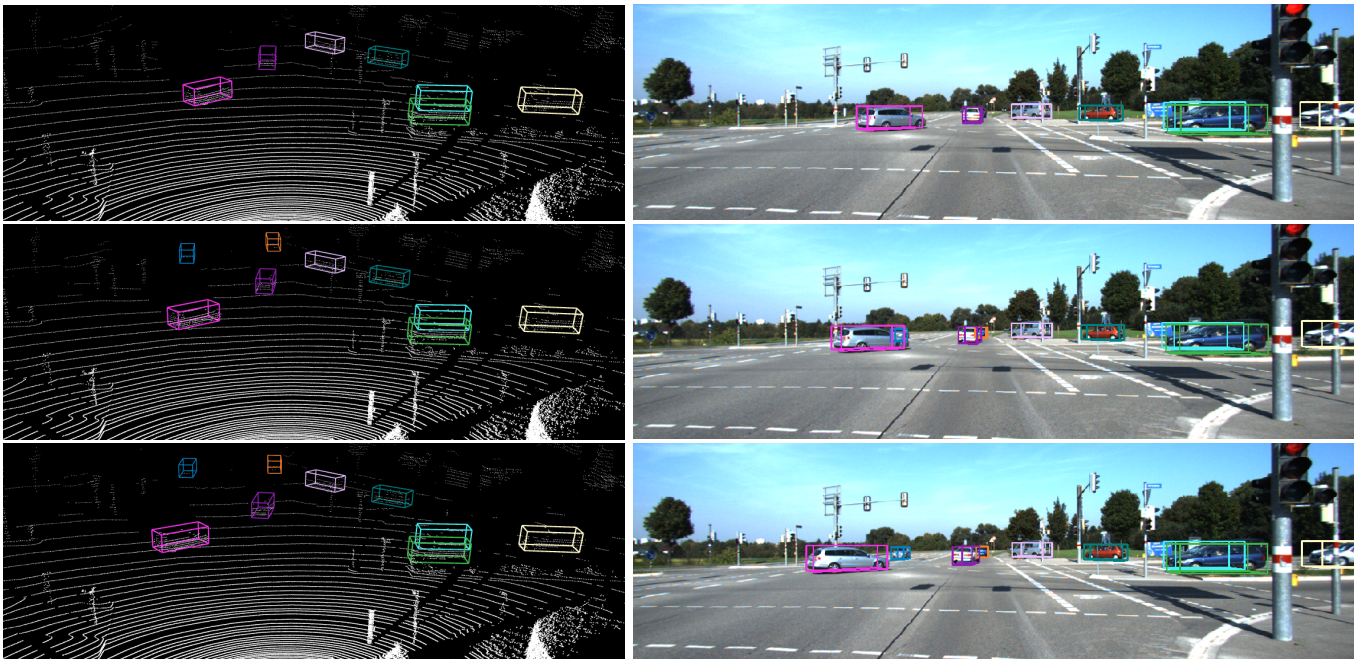

Fig. 4: Visualization of a set of trajectories produced by the tracker. Trajectories are color coded, such that having the same color means it's the same object.

performance substantially to 83.77% AP_{3D} in *easy* setting, 58.31% in *moderate* setting and 57.12% in *hard* setting.

Streaming level detection. With accurate 3D object detection results of adjacent key frames and target displacements, streaming level 3D object detection can be implemented. We investigate the effect of multi-frame input during testing. Specifically, we focus on the effect of different temporal strides τ on inference time and accuracy. Towards this goal, we train five models with $\tau = \{0, 1, 3, 5, 7\}$, and then link the predicted detections over time and generate detections in intermediate frame by box interpolation. Results are shown in *Table* I. Bi-AVOD* ($\tau = 3$) achieves the best result among five models, with 77.89% AP_{3D} in *easy* setting, 63.80% AP_{3D} in *moderate* setting, 63.50% AP_{3D} in *hard* setting. Compared with the based fine-tuned model Bi-AVOD* ($\tau = 0$), the AP scores of Bi-AVOD* ($\tau = 3$) can be boosted significantly (e.g. 3D *moderate* setting by 5.49%, 3D *hard* setting by 6.38%, BEV *moderate* setting by 9.09%, BEV *hard* setting by 9.11%). This gain demonstrates that the detection of truncated and occluded targets can benefit from a large temporal stride. However, there is also a non-ignorable decay on the *easy* setting (by -5.88%), we think it is mainly caused by the failed link at both ends of the trajectories (see Sec. 3.4 for detail). Moreover, *Table* I shows that a too large τ leads to a significant decay of accuracy. This is straightforward as a larger temporal stride introduces more failed trajectories link.

We calculate the inference time in streaming level. Results in *Table* I shows that a larger temporal stride leads to less time cost per frame. Moreover, when τ is larger than 3, our Bi-AVOD network can run faster than origin AVOD in streaming level. We chose $\tau = 3$ for our following experiment, which is a good trade-off between speed and accuracy.

Multi-object tracking. We finally validate our approach on multi-object tracking. To investigate the effect of our correlation module, we compare our approach with original AVOD structure in our evaluation datasets. Performance comparison is shown in *Table* II. We see that Our Bi-AVOD approach outperforms origin AVOD by a large margin in nearly all tracking metrics (e.g. MOTA by 20.31%, MOTP by 2.6%, MT by 28.15%, ML by 26.47%, FRAG by 43). This indicates that our correlation module can improve the performance of multi-object tracking significantly. We also compare our approach to publicly available methods in KITTI Tracking Benchmark. In *Table* III we see that our approach is competitive with the state of the art, outperforming other methods in some of the metrics (MOTP and MT). Note that KITTI only evaluates the metrics in 2D, which does not fully represent the performance of our 3D approach. We also visualize some trajectories produced by our tracker. A example is shown in *Figure* 4. It shows that our approach can generate nice trajectories for most targets, even though those truncated and occluded targets. More examples are available in the supplementary materials.

V. CONCLUSIONS

We propose Bi-AVOD, a unified framework for simultaneous 3D object detection and tracking in streaming data. The network is a dual-way structure and can process two frames at the same time. Embedded with a correlation module to encode the diversity of adjacent frames, our network can perform object detection and tracking in a very efficient way. Our approach achieves accuracy competitive with the state-

of-the-art methods in KITTI Tracking Benchmark. In the future, we plan to improve our approach with a more flexible key frame selection algorithm and explore the mismatch problem of trajectory boundaries.

REFERENCES

[1] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, and R. Urtasun, "Monocular 3d object detection for autonomous driving," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 2147–2156.

[2] X. Chen, K. Kundu, Y. Zhu, H. Ma, S. Fidler, and R. Urtasun, "3d object proposals using stereo imagery for accurate object class detection," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 5, pp. 1259–1272, 2018.

[3] Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4490–4499.

[4] B. Yang, W. Luo, and R. Urtasun, "Pixor: Real-time 3d object detection from point clouds," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7652–7660.

[5] M. Simon, S. Milz, K. Amende, and H.-M. Gross, "Complex-yolo: An euler-region-proposal for real-time 3d object detection on point clouds," in *European Conference on Computer Vision*. Springer, 2018, pp. 197–209.

[6] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1907–1915.

[7] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3d proposal generation and object detection from view aggregation," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1–8.

[8] P. Lenz, A. Geiger, and R. Urtasun, "Followme: Efficient online min-cost flow tracking with bounded memory and computation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4364–4372.

[9] B. Li, "3d fully convolutional network for vehicle detection in point cloud," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 1513–1518.

[10] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner, "Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 1355–1361.

[11] W. Luo, B. Yang, and R. Urtasun, "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*, 2018, pp. 3569–3577.

[12] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum pointnets for 3d object detection from rgb-d data," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 918–927.

[13] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[14] X. Zhu, Y. Wang, J. Dai, L. Yuan, and Y. Wei, "Flow-guided feature aggregation for video object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 408–417.

[15] K. Kang, H. Li, J. Yan, X. Zeng, B. Yang, T. Xiao, C. Zhang, Z. Wang, R. Wang, X. Wang *et al.*, "T-cnn: Tubelets with convolutional neural networks for object detection from videos," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 10, pp. 2896–2907, 2018.

[16] K. Kang, W. Ouyang, H. Li, and X. Wang, "Object detection from video tubelets with convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 817–825.

[17] W. Han, P. Khorrami, T. L. Paine, P. Ramachandran, M. Babaeizadeh, H. Shi, J. Li, S. Yan, and T. S. Huang, "Seq-nms for video object detection," *arXiv preprint arXiv:1602.08465*, 2016.

[18] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Detect to track and track to detect," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3038–3046.

[19] D. Frossard and R. Urtasun, "End-to-end learning of multi-sensor 3d tracking by detection," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 635–642.

[20] E. Bochinski, T. Senst, and T. Sikora, "Extending iou based multi-object tracking by visual information," *AVSS. IEEE*, 2018.

[21] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2758–2766.

[22] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[23] A. Milan, S. Roth, and K. Schindler, "Continuous energy minimization for multitarget tracking," *IEEE TPAMI*, vol. 36, no. 1, pp. 58–72, 2014.

[24] J. H. Yoon, M.-H. Yang, J. Lim, and K.-J. Yoon, "Bayesian multi-object tracking using motion context from multiple objects," in *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2015.

[25] A. Geiger, M. Lauer, C. Wojek, C. Stiller, and R. Urtasun, "3d traffic scene understanding from movable platforms," *Pattern Analysis and Machine Intelligence (PAMI)*, 2014.

[26] P. Lenz, A. Geiger, and R. Urtasun, "Followme: Efficient online min-cost flow tracking with bounded memory and computation," in *International Conference on Computer Vision (ICCV)*, 2015.

[27] J. H. Yoon, C.-R. Lee, M.-H. Yang, and K.-J. Yoon, "Online multi-object tracking via structural constraint event aggregation," in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[28] A. Gaidon and E. Vig, "Online Domain Adaptation for Multi-Object Tracking," in *British Machine Vision Conference (BMVC)*, 2015.

[29] W. Choi, "Near-online multi-target tracking with aggregated local flow descriptor," *ICCV*, 2015.

[30] S. Wang and C. Fowlkes, "Learning optimal parameters for multi-target tracking with contextual interactions," *International Journal of Computer Vision*, 2016.

[31] A. Milan, K. Schindler, and S. Roth, "Detection- and trajectory-level exclusion in multiple object tracking," in *CVPR*, 2013.

[32] Y. Xiang, W. Choi, Y. Lin, and S. Savarese, "Subcategory-aware convolutional neural networks for object proposals and detection," in *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017.

[33] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.