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 progresses due to the development of deep learning. However, previous researches are mostly based on single frame input, information between frames is scarcely explored. In this paper, we attempt to leverage the temporal information in streaming data and explore 3D streaming based object detection as well as tracking. Toward this goal, we set up a dual-way network for 3D object detection in keyframes only, and then propagate predictions to non-key frames through a motion based interpolation algorithm guided by temporal information. Our proposed framework is shown to have great improvements on object detection compared to frame-by-frame approach, and also proven to produce competitive results on the KITTI Object Tracking Benchmark, with 76.68% in MOTA and 81.65% 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 data can provide accurate location 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 is always obtained in a streaming fashion, thus it's more natural to perform object detection with streaming data. Compared to single frame, streaming data can provide consistent temporal correlations between consecutive frames for detected features, which can reduce noisy detections 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 also challenging. Last but not least, the sheer numbers of frames that streaming data provided introduce tremendous computational costs, especially for real-time applications.

This paper proposes a **D**ual-way **O**bject **D**etection and **T**racking (**DODT**) framework, as is illustrated in *Figure* 1, to tackle aforementioned problems. We construct our framework based on the following observations: (1) streaming data is huge but redundant, thus we can process keyframes only and propagate predictions to non-key frames. (2) object detection can be benefit from long-term information which can be obtained through cross-correlation between adjacent frames, as demonstrated in [8]. For first observation, we design our framework into a dual-way fashion, thus it can operate two adjacent keyframes simultaneously and convenient for prediction propagation. Moreover, a specific network named *Shared RPN* is proposed to generate 3D proposals to be shared by two detection branches. For second observation, a *Temporal module* is introduced to capture temporal information through cross-correlation in BEV space, and then predicts object co-occurrence and offsets between two keyframes. Notably, different from [8], [9], our correlation operation is performed on proposal level, which leads to a much less computational cost.

To generate predictions of every frame, a motion based interpolation algorithm is developed to propagate keyframe predictions to non-key frames guided by *Temporal module* outputs. Meanwhile, multi-object tracking can also be accomplished through *tracking by detection* [10]. In summary, our contributions are threefold:

- We propose a dual-way framework named DODT, which performs 3D object detection and tracking precisely on streaming data by only processing keyframes.
- A *Temporal module* is proposed to encode temporal information across frame in proposal level, which is much efficient than previous approaches.
- A motion based interpolation algorithm is developed for prediction propagation, which leads to considerable performance improvements in object detection and tracking.

II. RELATED WORK

3D object detection. Currently, most approaches in 3D object detection are usually done in three fronts: image based, point cloud based, and multi-view fusion based. Image based approaches such as Mono3D [1] and 3DOP [2] use camera data only. Since image lacks depth information, handcrafted geometric features are required in these approaches. As for the point cloud based methods, there are usually two ways: voxelization based and projection based, according to how point clouds features are represented. Voxelization based methods such as 3D FCN [11], Vote3Deep [12], VoxelNet [3], utilize a voxel grid to encode features. These

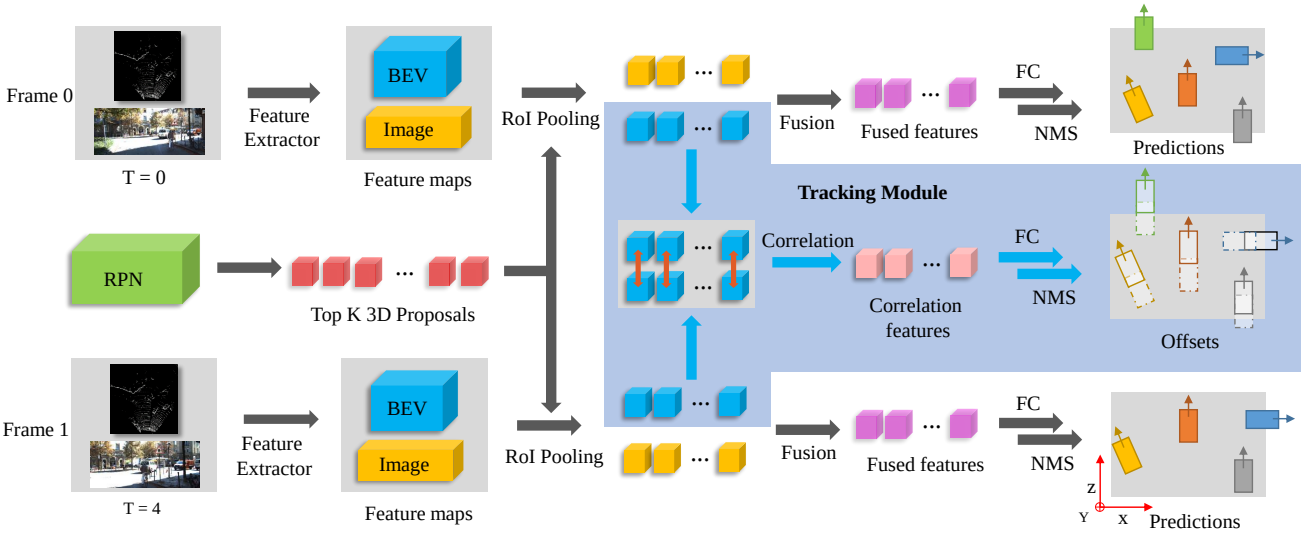

Fig. 1: DODT architecture. Blue area is *Temporal module*, it does not use image features since information in BEV is sufficient for tracking.

approaches suffer from the sparsity of point clouds and enormous computation costs in 3D convolution. While projection based methods such as PIXOR [4], Complex-YOLO [5], Complexer-YOLO [13] attempt to project point clouds to a perspective view (e.g. bird eye view) and apply image-based feature extraction techniques. However, due to the sparsity of point clouds, features after projection are usually insufficient for accurate object detection, especially for small targets. Multi-view fusion based approaches such as F-PointNet [14], MV3D [6], AVOD [7], try to fuse point cloud features with image features for accurate object detection. These methods distinguish from each other mainly on how data is fused. Our detection branches are constructed based on AVOD, but proposal features are enhanced with temporal information.

Video object detection. Nearly all existing methods in video object detection incorporate temporal information on either feature level or final box level. FGFA [15] leverages temporal coherence on feature level, it warps the nearby frames feature maps to a reference frame for feature enhancement according to flow emotion. On the other hand, T-CNN [16], [17] leverages precomputed optical flows to propagate predicted bounding boxes to neighboring frames; Seq-NMS [18] improves NMS algorithm for video by constructing sequences along nearby high-confidence bounding boxes from consecutive frames. They are all utilize temporal information in final box level. There are also a few approaches attempt to learn temporal features between consecutive frames without optical flow. D&T [8] proposes a two branches detection network for object detection and tracking simultaneously in video. The network learns temporal information representation through computing convolutional cross-correlation between frames. Our DODT approach is mainly inspired by D&T, however, we develop this idea to 3D space and restrain correlation operation in proposal level, which reduce computational costs significantly. Moreover, tracking using two adjacent frames often suffers from drift, loss of targets in one frame, etc., our approach can handle these issues well by performing motion based interpolation algorithm.

3D multi-object tracking. Existing 3D multi-object tracking approaches are mostly implemented based on tracking by detection. For example, FaF [19] 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. 3D-CNN/PMBM [20] trained a DNN to detect and estimate the distance to objects from a single image, and then fed the detections to a Poisson multi-Bernoulli mixture tracking filter for 3D tracking. DSM [21] 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 an offline version, while our tracking algorithm is an near online approach.

III. METHODOLOGY

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

A. DODT Model Structure

We aim at performing 3D object detection and tracking on streaming data. To this end, we design DODT into a dual-way network. *Figure* 1 illustrates the whole pipeline. With the help of the dual-way structure, the network can fed with

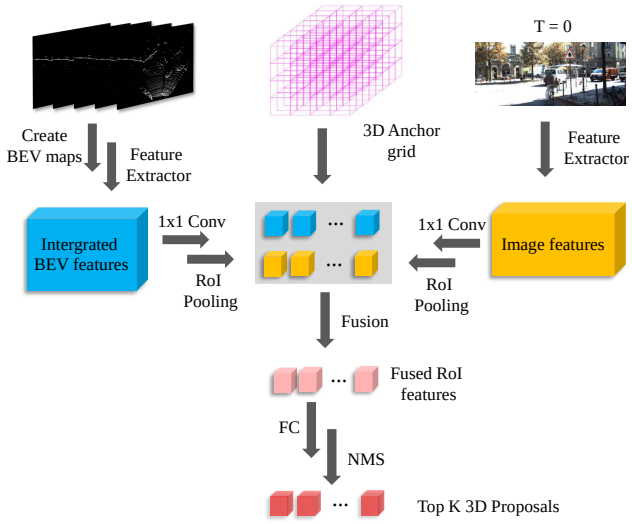

Fig. 2: Shared RPN module.

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 is first fed to a feature extractor to get corresponding feature maps. The feature extractor is designed following the procedure described in AVOD [7], which constructed in an encoder-decoder fashion resulting in a full resolution feature map. To extract feature crops from image and BEV feature maps, we also follow the idea proposed in AVOD. Given a 3D anchor generated by RPN, two view specific ROIs are obtained by projecting the anchor onto the image and BEV feature maps, the corresponding feature crops are extracted and then RoI pooling is performed from two views. After that, a *early fusion* scheme in MV3D [6] is used to combine multi-view features in proposal level. Finally, after applying fully connected layers for classification and box regression, and 2D non-maximum suppression (NMS) for proposals filtering, the final predictions are produced. Meanwhile, fed with BEV feature crops of two branches, *Temporal module* performs correlation operation on feature crop pairs to produce cross-correlation features. The cross-correlation features are then used to predict object co-occurrence and displacement between two keyframes for streaming-level detection and tracking.

The whole network is designed in an end-to-end form. The multi-task objective consists of a cross-entropy loss L_{cls} for classification, a smooth *L1* loss L_{reg} for box regression, and a smooth *L1* loss L_{corr} for displacement regression between corresponding objects across two keyframes. L_{reg} and L_{corr} are normalized by the number of proposals while L_{cls} is normalized by the number of positive proposals.

B. Shared RPN

We transform RPN network in AVOD [7] to our *shared RPN network* (as shown in *Figure* 2), which can generate 3D proposals shared by both detection branches. To ensure proposals generated by our RPN are suitable for both

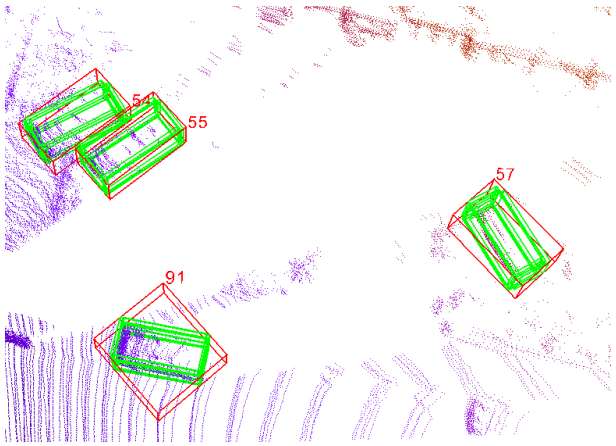

Fig. 3: Five consecutive point clouds in the same coordinate system. Green boxes are default labels of five frames, while red boxes are the axis-aligned new labels for RPN training. The numbers are object id.

keyframes, we create integrated BEV feature maps based on two keyfames and the frames between them. Note that point cloud shows object accurate 3D localization, we can simply transform these frames to a same coordinate system and then fuse them. Since point cloud is extremely sparse and is encoded by projection, this process does not increase any computational cost but enhance the point cloud BEV feature maps greatly. Assume there are five frames in point cloud input, due to object movement, the location of the same object in five frames are shift. For training convenient, we replace original proposal labels with new axis-aligned labels that can cover all five original labels of an object in five frames. *Figure* 3 illustrates these two kind of labels. Though this process enlarges proposal labels, it can ensure that new labels are suitable for proposal prediction in both detection branches. Moreover, to obtain accurate proposals, We fuse BEV feature maps with image data to enhance object appearance features. Since one image contains enough features in a short temporal slice, our module only aggregates the features in first image of a slice. Subsequent processing and operations are similar with RPN module described in AVOD, we refer the reader to [7] for more information.

C. Temporal Module

The *Temporal Module* is demonstrated in *Figure* 1. Given two sets of point cloud BEV features crops $F_t, F_{t+\tau}$, a set of cross frame feature pairs can be constructed as $\{(F_t^i, F_{t+\tau}^i) \mid i \in \{0, 1, ..., N\}\}$, where $F_t^i, F_{t+\tau}^i$ are features extracted by i-th proposal from frame t and frame $t+\tau$ respectively, τ is temporal stride and N is the number of 3D proposals. Note that proposals generated by RPN are shared by two detection branches, thus the set can be obtained conveniently. After the correspondence of feature crops is constructed, correlation operation can be performed on feature pair $(F_t^i, F_{t+\tau}^i)$ to compute cross-correlation features over frames. Once the cross-correlation features are obtained, they are fed to a classification head for object co-occurrence prediction and a regression head for box offsets prediction respectively. Object co-occurrence determines whether an object exists

on both keyframes, it provides an extra guarantee for the successful implementation of the interpolation algorithm. When an object is loss in one of the two keyframes, object co-occurrence shows the probability of a mis-detection, or a track start or end. More details are available in Sec. D.

For a target we have ground truth $d^t = (p_{co}, d_x^t, d_z^t, d_{ry}^t)$ in frame t for *Temporal Module*, and similarly $d^{t+\tau}$ for frame $t+\tau$, denoting object co-occurrence probability and its horizontal and vertical center coordinates as well as orientation in BEV map. p_{co} is 1 if the target exists on both keyframes, 0 otherwise. The box offsets label $\Delta^{t,t+\tau} = (\delta_x^{t,t+\tau}, \delta_z^{t,t+\tau}, \delta_{ry}^{t,t+\tau})$ is then

$$\Delta^{t,t+\tau} = \begin{cases} (\frac{d_x^{t+\tau} - d_x^t}{d_w^t}, \frac{d_z^{t+\tau} - d_z^t}{d_l^t}, \frac{d_{ry}^{t+\tau} - d_{ry}^t}{d_{ry}^t}) & d \in F_t \cap F_{t+\tau} \\ (0.0, 0.0, 0.0) & otherwise \end{cases} \tag{1}$$

where d_w^t, d_l^t are width and height of the object.

D. 3D Streaming Object Detection and Tracking

Since streaming data possesses a lot of redundant information and objects typically move smoothly in time, we can perform detection network on keyframes and propagate results to the rest of the frames by applying Algo. 1. Fed with two detections lists $D^t, D^{t+\tau}$ and a offsets list $\Delta^{t,t+\tau}$ of keyframes, the algorithm first utilizes linear interpolation to calculate detections in intermediate frames if an object exists on both keyframes. The association of detections is determined by *getMatched* function, which performs a 3D IoU based algorithm with a threshold 0.2 to select the best matched detection in second keyframe given a rectified detection $(d_i^t + \delta_i^t)$ in first keyframe. If match failed, the function output none. As for a detection that exists only in one keyframe, if its co-occurrence probability p_{co}^i is less than p_{co}^{max} (0.5 in our experiment), we regard it as a mis-detection in another keyframe, then we apply linear interpolation to propagate the detection to intermediate frames as well as another keyframe using offsets prediction $\delta_i^{t,t+\tau}$. Otherwise we think a trajectory birth or death happens, then a motion model is leveraged to propagate the detections.

The motion model holds the hypothesis that the target velocity is constant within a short time period, independent of camera ego-motion. Thus we can calculate target state in next frame using historical state and increment. In detail, we maintain a global velocity (v_x, v_z, v_{ry}) in BEV space, it will be updated to $(\frac{\delta_x^{t,t+\tau}}{\tau}, \frac{\delta_z^{t,t+\tau}}{\tau}, \frac{\delta_{ry}^{t,t+\tau}}{\tau})$, once a new detection is matched with the trajectory. As for the start or end of a track, since there is no way for us to locate its head or tail with only keyframes predictions, we simply extend the track by F_{ext} (3 in our experiment) frames according to the trend of movement. If the track outsides the range of BEV feature maps, then the extension will be terminated early.

In addition, we observe that a track sometimes suffers from opposite orientation in nearby keyframes, which will lead to inferior performance. To address this issue, we operate a simple orientation correction algorithm during box interpolation and track extension. Consider a track T_{match}^i and its next matched detection D_{match}^i, if the difference of orientation in T_{match}^i and D_{match}^i is grater than $\frac{\pi}{2}$, we add a π to the orientation in D_{match}^i so that its orientation can be roughly consistent with the track.

Multi-object tracking can be accomplished with streaming level object detection simultaneously. Note that after performing interpolation algorithm on keyframe pairs, data association between keyframe pairs is also obtained. We only need to associate keyframe detections to link tracklets in time and build long-term object tubes. Since our method associates next temporal slice to the track at one time, this shows an near online tracking approach.

Algorithm 1: Motion based Interpolation Algorithm

Input:
$D^t = [d_0^t, d_1^t, ..., d_{N_t}^t], D^{t+\tau} = [d_0^{t+\tau}, d_1^{t+\tau}, ..., d_{N_{t+\tau}}^{t+\tau}]$,
$\Delta^{t,t+\tau} = [\delta_0^{t,t+\tau}, \delta_1^{t,t+\tau}, ..., \delta_N^{t,t+\tau}], N = max\{N_t, N_{t+\tau}\}$
Output: $D = [D^t, D^{t+1}, ..., D^{t+\tau}]$
Initialize: $D_{temp} = D^{t+\tau}, D, p_{co}^{max}$
for $d_i^t \in D^t$ **do**
  $d' = getMatched(d_i^t + \delta_i^t, D_{temp})$
  if d' **then**
    $d_i^{t+1}, ..., d_i^{t+\tau-1} = Interpolate(d_i^t, d')$
    remove d' from D_{temp}
  else if $p_{co}^i > p_{co}^{max}$ **then**
    $d_i^{t+1}, ..., d_i^{t+\tau} = Interpolate(d_i^t, \delta_i^t)$
  else
    predict $(d_i^{t+1}, ..., d_i^{t+\tau-1})$ by motion model
if D_{temp} *is not empty* **then**
  for $d_j^{t+\tau} \in D_{temp}$ **do**
    if $p_{co}^j > p_{co}^{max}$ **then**
      $d_j^t, ..., d_j^{t+\tau-1} = Interpolate(d_j^{t+\tau}, \delta_j^{t+\tau})$
    else
      predict $(d_j^{t+1}, ..., d_j^{t+\tau-1})$ by motion model

IV. EXPERIMENTS

A. Datasets and Training

Datasets and Preprocessing. We use KITTI object tracking Benchmark [22] for evaluation. It consists of 21 training sequences and 29 test sequences with vehicles annotated in 3D. For object detection, we split 21 training sequences into two parts according to their sequence number, odd numbered sequences for training and the rest for evaluation. For tracking, we train our model in all 21 training sequences and evaluate on test datasets. Similar to the data preprocessing in AVOD [7], we crop point clouds at $[-40, 40] \times [0, 70] \times [0, 2.5]$ meters along X, Z, Y axis respectively to contain points within the field of camera view. To make the system invariant to the speed of the ego-car, we calculate the displacement of the observer between different frames and translate the coordinates accordingly using IMU data.

We notice that the labels provided in KITTI object tracking datasets are incomplete. For example, first row in *Figure* 4 illustrates several frames and labels in sequence 0. We can

		IoU = 0.5			IoU = 0.7			
Methods	Modules	Easy	Moderate	Hard	Easy	Moderate	Hard	FPS
AVOD[7]	-	90.13 / 90.91	80.00 / 81.79	71.61 / 81.79	76.00 / 90.90	57.23 / 81.73	56.13 / 72.69	10.0
DODT(τ = 1)	-	88.28 / 99.97	85.74 / 90.90	86.14 / 90.89	83.44 / 90.82	67.48 / 90.79	61.24 / 90.80	6.7
DODT(τ = 1)	T	88.32 / **99.99**	86.53 / 90.90	86.71 / **90.90**	83.60 / 90.82	68.93 / 90.80	62.69 / 90.81	5.9
DODT(τ = 1)	M	89.99 / 99.95	87.86 / 90.87	87.81 / 90.86	86.89 / 90.89	73.96 / 90.83	67.07 / 81.79	6.5
DODT(τ = 1)	T + M	**90.63** / 99.95	89.07 / 90.90	88.79 / **90.90**	88.74 / 90.91	75.27 / 90.84	68.75 / 90.57	5.7
DODT(τ = 2)	T + M	90.60 / 99.94	**89.19 / 90.91**	**88.91** / 90.88	**88.90 / 90.92**	**76.64** / 90.85	75.81 / 90.83	8.6
DODT(τ = 3)	T + M	90.61 / 99.98	89.01 / 90.89	88.84 / 90.89	88.81 / 90.91	76.38 / **90.86**	**75.83 / 90.85**	11.4
DODT(τ = 4)	T + M	90.55 / 99.94	88.82 / 90.88	88.34 / 90.87	88.43 / 90.91	75.70 / 90.82	68.75 / 90.82	14.3
DODT(τ = 5)	T + M	87.98 / 90.91	85.57 / 90.87	86.01 / 90.87	81.59 / 90.81	67.30 / 90.76	61.35 / 81.73	17.1
DODT(τ = 6)	T + M	78.77 / 90.75	70.88 / 90.71	71.65 / 81.70	71.71 / 90.44	55.86 / 81.50	56.80 / 81.51	**20.0**

TABLE I: We report AP_{3D}/AP_{BEV} (in %) of the **Car** category on evaluation datasets, corresponding to average precision of the birds-eye view and 3D object detection. T is *Treacking Module*, M is *Motion based interpolation algorithm*. τ is temporal stride.

see that some targets in frame 118 and 120 are not labeled even they can be well observed and are labeled in frame 128. While our model can well predict these objects, as shown in second row. In order to evaluate our model as accurately as possible, we labeled these missing tags manually.

Training and testing. We train our network only for *Car* category temporarily, following most of the super-parameter settings in AVOD [7] during training and testing. The network is trained for 120K iterations with a batch size 1, using an ADAM [23] optimizer with an initial learning rate of 0.0001 that is decayed exponentially every 30K iterations with a decay factor of 0.8. Loss weights for $L_{cls}, L_{reg}, L_{corr}$ are 1.0, 5.0, 1.0 respectively. During proposal generation, anchors with IoU less than 0.3 are considered background and greater than 0.5 are objects. 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

Shared RPN. To evaluate the performance of our *Shared RPN*, we implement a non-shared version of RPN. It predicts proposals for each keyframes independently based on each feature maps. A comparison of proposal prediction accuracy between *Non-shared RPN* and *Shared RPN* is shown in *Figure* II. Result showing *Shared RPN* outperforms *Non-shared RPN* by 0.66% indicates that the shared mechanism in RPN promotes the accuracy of proposal prediction.

Method	Non-shared RPN	Shared RPN
Accuracy(%)	97.81	**98.47**

TABLE II: Comparison of proposal prediction accuracy.

Streaming level detection. The main results on 3D object detection are summarized in *Table* I. We first evaluate the effectiveness of the dual-way structure, *Temporal Module* and our interpolation algorithm in 3D object detection with a temporal stride $\tau = 1$. Several important trends can be observed: **1)** Compared to original AVOD [7] model trained on the same datasets, our DODT model (without *Temporal Module* and interpolation algorithm) shows improvements in all settings with IoU threshold 0.7. These improvements indicate that the introduction of dual-way structure and *Shared RPN* contribute to detection performance significantly. **2)** The introduction of *Temporal Module* and interpolation algorithm is conducive to model performance. *Temporal Module* brings 0.16%, 1.45% and 1.45% gain in *Easy*, *Moderate* and *Hard* setting with a overlap of 0.7 respectively. It shows that the adding of object co-occurrences information contributes to the detection in occluded and truncated objects more than easy ones. While our interpolation algorithm improves model accuracy greatly, by 3.45%, 6.48% and 5.83% in three setting respectively with a overlap of 0.7. These significant improvements indicate that our algorithm works notably well in all settings. It's not surprising because the algorithm has the ability to reject false positives predictions and generate new results by perfecting the trajectories. **3)** *Temporal Module* and interpolation algorithm can work together harmony and additionally improve model accuracy by 1-2% in all setting, this is because the interpolation algorithm can work better with the results of *Temporal Module* (see Sec. D in methodology for details).

Secondly, We investigate the effect of multi-frame input. Specifically, we focus on the effect of different temporal strides τ on inference accuracy and speed. Towards this goal, we train six models with $\tau = \{1, 2, 3, 4, 5, 6\}$ respectively, and then link the predicted detections over time and generate detections in intermediate frames by box interpolation. Results are shown in *Table* I. DODT ($\tau = 2$) shows the best result in *Easy* and *Moderate* settings with IoU = 0.7, by 88.90% and 76.64% in AP_{3D} respectively. While DODT ($\tau = 3$) shows the best performance in *Hard* setting, with 75.84% in AP_{3D}. Compared with the base model DODT ($\tau = 1$), the model performances with $\tau = 2, 3$ are boosted significantly in hard objects, by about 1% in *Moderate* setting and about 7% in *Hard* setting with IoU = 0.7. This gain demonstrates that the detection of truncated and occluded targets can benefit greatly from a large temporal stride. Bottom two rows in *Figure* 4 shows how our interpolation algorithm improves model performance. Boxes in magenta are DODT predictions with $\tau = 3$ while yellow are with $\tau = 1$. The results shows that a larger τ can reduce noise and generate more true positive predictions. However, *Table* I also shows that a too large τ leads to a significant decay of accuracy. As temporal stride increasing, there are more trajectories start or end within two keyframes, thus box interpolation would introduce more false positive predictions.

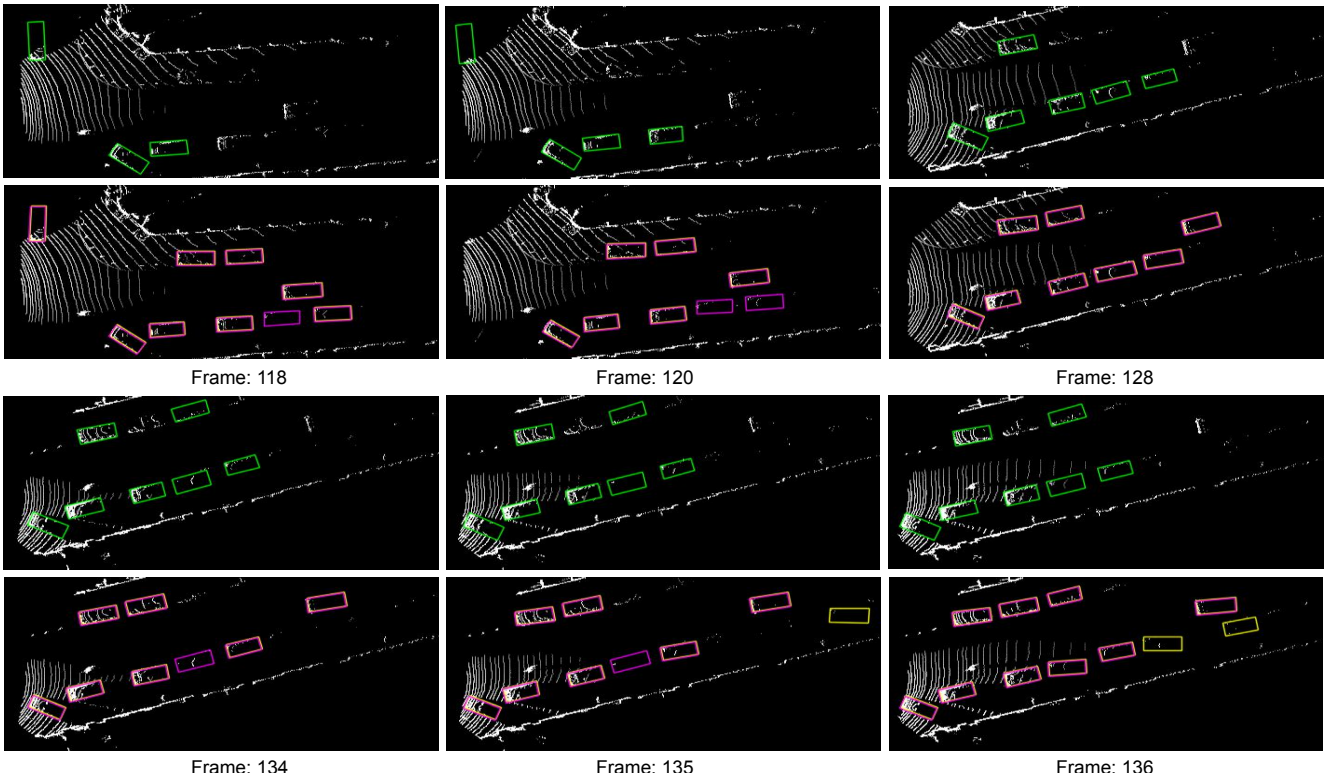

Fig. 4: Visualization of labels and our predictions in sequences 0. Boxes in **Green** are official ground truth, boxes in **Yellow** are results predicted with temporal stride $\tau = 1$, and boxes in **Magenta** are results predicted with temporal stride $\tau = 3$. Best viewed in color.

Methods	Modules	MOTA(%)↑	MOTP(%)↑	MT(%)↑	ML(%)↓	IDS↓	FRAG↓
DODT(τ = 3)	-	72.09	81.84	58.15	12.31	233	756
DODT(τ = 3)	T	73.26	82.38	58.61	12.00	340	699
DODT(τ = 3)	M	74.36	**82.49**	56.66	13.08	**27**	**310**
DODT(τ = 3)	T+M	**76.68**	81.64	**60.77**	**11.69**	63	384

TABLE III: Ablation study on KITTI Tracking test datasets.

Method	MOTA(%)↑	MOTP(%)↑	MT(%)↑	ML(%)↓	IDS↓	FRAG↓	FPS↑
Complexer-YOLO[13]	75.70	78.46	58.00	5.08	1186	2096	100.0
DSM[21]	76.15	83.42	60.00	8.31	296	868	10.0 (GPU)
3D-CNN/PMBM[20]	80.39	81.26	62.77	6.15	121	613	71.4
3DT[24]	**84.52**	**85.64**	**73.38**	**2.77**	377	847	33.3
DODT(ours)	76.68	81.65	60.77	11.69	**63**	**384**	100.0

TABLE IV: Comparison of publicly available methods in the KITTI Tracking Benchmark. The time for object detection is not included in the specified runtime.

We also compute the inference time in streaming level detection. The inference time for two keyframe inputs is 0.175s. As temporal stride τ increasing, the time cost for detection is unchanged, and interpolation time cost increases negligible, thus the inference time is almost the same, but FPS increases by multiple. We chose $\tau = 3$ for our following experiment, which is a good trade-off between inference speed and accuracy.

Multi-object tracking. For multi-object tracking, We first investigate the contribution of different modules to tracking accuracy in KITTI tracking test datasets, results are shown in *Table* III. The results shows that both modules can improve tracking performance. With these modules, our model outperforms base model by a large margin in nearly all tracking metrics (e.g. MOTA by 4.59%, MT by 2.62%, ML by -0.62%, IDS by -170 and FRAG by -372), as *Temporal Module* makes the data association more precise, and box interpolation can help to complete the track notably.

We also compare our approach to publicly available methods of multiple object tracking in 3D in KITTI Tracking Benchmark, results are shown in *Table* IV. We can see that our approach is competitive with the state-of-the-art. For multiple object tracking accuracy, our method outperforms Complexer-YOLO [13] and DSM [21], but behind 3D-CNN/PMBM [20] and 3DT [24]; For trajectory ID-switches and fragmentations, our method outperforms all other methods. Note that 3D-CNN/PMBM utilizes PMBM filter for data association while ours is a simple IoU based method, and 3DT trained their system on a new datasets collected on a realistic 3D virtual environments. Moreover, labels of test datasets maybe incomplete just like the case in training datasets, thus the tracking performance of our approach maybe better than current ones. Also, our approach shows a competitive results in runtime.

V. CONCLUSIONS

We propose DODT, a unified framework for simultaneous 3D object detection and tracking based on streaming data. The network is a dual-way structure and can process two keyframes at the same time. Embedded with a *Temporal Module* to encode the diversity of adjacent keyframes and a motion based interpolation algorithm to generate predictions in non-key frames, our network can overcome the challenge of target occlusion and loss in one frame, and perform object detection and tracking in a very efficient way.

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] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] M. Simon, K. Amende, A. Kraus, J. Honer, T. Samann, H. Kaulbersch, S. Milz, and H. Michael Gross, "Complexer-yolo: Real-time 3d object detection and tracking on semantic point clouds," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] S. Scheidegger, J. Benjaminsson, E. Rosenberg, A. Krishnan, and K. Granström, "Mono-camera 3d multi-object tracking using deep learning detections and pmbm filtering," in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 433–440.

[21] 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.

[22] 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.

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

[24] H.-N. Hu, Q.-Z. Cai, D. Wang, J. Lin, M. Sun, P. Krhenbhl, T. Darrell, and F. Yu, "Joint monocular 3d detection and tracking," 2019.