SDP-Net: Scene Flow Based Real-time Object Detection and Prediction from Sequential 3D Point Clouds

Yi Zhang[1*] [0000−0002−4263−7515], Yuwen Ye[1*] [0000−0002−8033−6641], Zhiyu Xiang[2†] [0000−0002−3329−7037], and Jiaqi Gu[1] [0000−0002−4644−6046]

[1] College of Information and Electronic Engineering, Zhejiang University, Hangzhou, China
[2] Zhejiang Provincial Key Laboratory of Information Processing, Communication and Networking, Zhejiang University, Hangzhou, China
xiangzy@zju.edu.cn

Abstract. Robust object detection in 3D point clouds faces the challenges caused by sparse range data. Accumulating multi-frame data could densify the 3D point clouds and greatly benefit detection task. However, accurately aligning the point clouds before the detecting process is a difficult task since there may exist moving objects in the scene. In this paper a novel scene flow based multi-frame network named SDP-Net is proposed. It is able to perform multiple tasks such as self-alignment, 3D object detection, prediction and tracking simultaneously. Thanks to the design of scene flow and the scheme of multi-task, our network is capable of working effectively with a simple network backbone. We further improve the annotations on KITTI RAW dataset by supplementing the ground truth. Experimental results show that our approach greatly outperforms the state-of-the-art and can perform multiple tasks in real-time.

1 Introduction

Object detection is a fundamental task for the safety of autonomous driving [1]. Robust object detection in 3D point clouds faces the challenges caused by sparse range points. Accumulating multiple frame data could densify the point clouds and greatly benefit the detection task. YOLO4D [2] shows that temporal information can improve the accuracy of 3D object detection by applying LSTMs [3]. FAF [4] pre-registers the multi-frame point clouds, uses 3D convolution for multi-frame feature extraction, and implements a multi-task network based on 3D object detection. However, pre-registration of point clouds requires location information or special algorithms like ICP [5] which introduce extra sensor requirement or pre-computation. In fact, accurately aligning is a difficult task since there may exist moving objects in the scene. Traditional point registration methods like ICP assume the static scene and cannot tackle this problem.

* Equal contribution.
† Corresponding author.

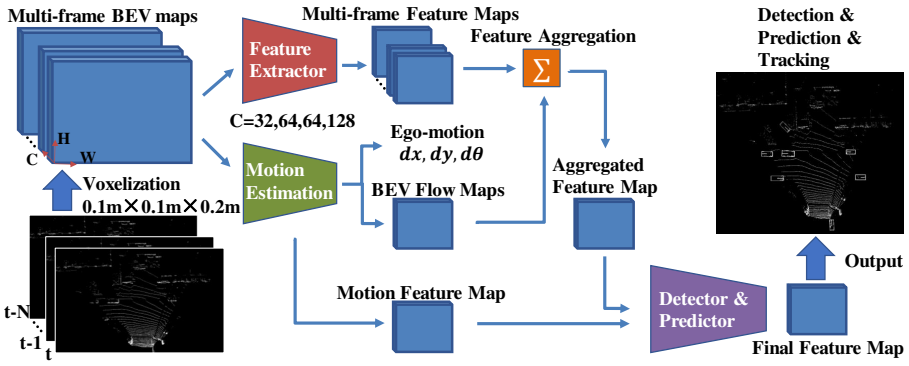

Fig. 1. Architecture of proposed network. We take multi-frame point clouds as input and perform multiple tasks jointly.

Besides detection, deeply understanding the scene's dynamics requires further prediction and tracking of the objects. When the observing platform itself is also moving, self-localization is also required to describe all of the motion dynamics in a unified world frame. These tasks are usually implemented separately. We believe it would be beneficial to fulfill them within a multi-task network. However, some challenges exist in realizing this idea. On the one hand, tracking by network requires multi-frame input, which means larger computation burden and difficulty of real-time implementation. If we still want the network to perform in real-time, only those with simple backbones can be considered. On the other hand, dealing with this complex multi-frame situation requires the network be sufficiently strong, so that all of the tasks can be correctly carried out. The dilemma reveals the real challenge when developing a real-time multi-frame multi-task network for perceiving the dynamic environment.

In this paper we propose SDP-Net, a novel fast and high-precision multi-frame multi-task network based on the bird's eye view (BEV) scene flow. It is able to complete the tasks of self-alignment, 3D object detection, prediction and tracking jointly from BEV. As shown in Fig. 1, we directly construct BEV maps from the multi-frame point clouds, and estimate the flow map and the ego-motion from them. After feature aggregation guided by BEV flow map, we detect objects, predict their motion and perform object tracking. The entire network uses 2D convolutions and shares features across multiple tasks to get real-time performance. We conduct the experiments on KITTI [6, 7] RAW dataset and apply automatic method and manual fine-tuning to complete the annotations of unlabeled objects. Experimental results show that we can extract the motion information of objects effectively, and perform these tasks accurately. Importantly, we can perform accurate object detection even based on a simple backbone, achieving the state-of-the-art performance and outperforms the similar networks by a large margin. The entire approach can run at 12 FPS, meeting the real-time requirement. In summary, the key contributions of this paper are:

- We propose a BEV scene flow based multi-frame alignment method, which could estimate the correspondences of 3D features between consecutive frames and adaptively aggregate multiple frame information to benefit the 3D object detection task;
- We propose a multi-frame multi-task network based on the proposed BEV scene flow. Multiple related tasks like self-alignment, ego-motion estimation, object detection, object prediction and tracking are integrated in a unifying framework. These tasks complement each other and contribute achieving better performance than a single task;
- We improve the KITTI [6, 7] RAW dataset by supplementing the ground truth, increasing the number of annotations by 10.58% for more accurate and reliable evaluation. We only use the provided ground truth in KITTI RAW dataset to guarantee the quality of the supplemented annotations;
- Our method is implemented end-to-end and verified on KITTI RAW dataset. The experimental results demonstrated that our network can run in real-time and achieve the superior performance on the tasks as detection and tracking.

2 Related Work

2.1 Single-frame 3D Object Detection

In recent years, 3D object detection in a single frame has made great progress. MV3D [8] and AVOD [9] merge the information of RGB image and LIDAR point clouds to detect 3D objects. VoxelNet [10] and SECOND [11] detect objects by extracting voxel features in point clouds. F-PointNet [12] generates 2D object region proposals in RGB images to get the frustum point clouds, then applies PointNet [13] for 3D object detection. PointRCNN [14] uses PointNet++ [15] to extract features and optimizes the detection by box refinement. To improve the real-time performance, Complex-YOLO [16] build BEV map from 3D point clouds and apply YOLOv2 [17] to build a fast single-stage 3D object detector. PIXOR [18] and PIXOR++ [19] use occupancy maps of the point clouds as input, and perform 3D object detection with higher precision. PointPillars [20] applies PointNet to encode point clouds, and uses SSD [21] for object detection. MODet [22] extracts features in BEV and adopts prediction errors regression to improve accuracy. STD [23] uses spherical anchor and predict IoU branch to get great performance localization and classification. Fast Point R-CNN [24] combines RefinerNet to do a further fusion. PV-RCNN [25] and SA-SSD [26] combine voxel and points features to further improve the detection accuracy.

2.2 Multi-frame 3D Object Detection

The performance of object detection can be improved by utilizing temporal information. In the field of 2D video object detection, T-CNN [27] and MCMOT [28] use post-processing techniques to improve the detection results. Associated LSTM [29] and Bottleneck-LSTM [30] extract the multi-frame features with

LSTMs. FGFA [31] shows that the features of multiple frames can be fused through the guidance of optical flow to improve the detection performance. 3D object detection can benefit from similar ideas. Complexer-YOLO [32] applies multi-object tracking based on 3D object detection to increase accuracy and robustness. YOLO4D aggregates the 3D LIDAR point clouds over time as a 4D tensor, then applies LSTMs on YOLO3D [33] to extract temporal features and shows the advantages of incorporating the temporal dimension. FAF [4] pre-registers the multi-frame point clouds before input, then extracts multi-frame feature through 3D convolutions, and outputs the results of 3D object detection, prediction and tracking simultaneously. However, according to the same authers' work [19], FAF [4] still has lower object detection accuracy when compared with a faster 3D single-frame detector such as PIXOR. The reason partly lies in the misalignment error of moving objects in the scene, which is difficult to be removed by rigid body registration methods like ICP.

2.3 Object Prediction and Tracking

For moving objects, prediction and tracking can be further applied to describe the objects' behaviour in near future. When the observing platform itself is also moving, self-localization is also required to describe all of the motion dynamics in a unified world frame. Usually, these tasks are fulfilled separately.

Self-localization with 3D Point Clouds. Most of localization methods are presented in the field of SLAM. Given point clouds of different frames, DeepMapping [34] estimates poses and converts the registration problem to the unsupervised binary occupancy classification. L3-Net [35] uses PointNet to extract features, and applies 3D convolution and RNNs to output poses. [36] uses LIDAR Intensity Map to build embeddings, and estimates the localization by convolutional matching between the embeddings. PointFlowNet [37] estimates the 3D scene flow to endow the detected object with spatial velocity.

Prediction and Tracking. In order to predict the future location of objects, [38] proposes a system based on acceleration motion model and maneuver recognition respectively. [39] classifies vehicle motion with a Bayesian network for trajectories prediction. Given the detection results of the current and the past frames, the motion model or the appearance features is widely applied in data association [40, 41]. Besides Hungarian algorithm [42], deep association networks [43, 44] can also be employed. In order to track objects in 3D space, [45] proposes a 2D-3D Kalman filter to use the information of the image and 3D points jointly. AB3DMOT [46] extends the state space of Kalman filter with 3D object parameters, and performs fast and accurate object tracking without CNNs.

3D object detection, prediction and tracking are highly relevant tasks. Instead of completing these tasks separately, it would be a better way to fulfill them within a multi-task network. Inspired by [4], we take advantages of temporal information within multiple frames and perform object detection, prediction and tracking simultaneously. Different from [4], we estimate the BEV scene flow from multiple frames directly and do not require the frames to be pre-aligned. We can also perform an extra task of self-localization for the ego-vehicle.

3 SDP-Net

3.1 Network Structure

As shown in Fig. 1, the entire network is mainly composed of four modules, i.e., feature extraction, motion estimation, feature aggregation, and object detection and prediction. We build binary occupancy BEV map from point clouds of each frame, then perform feature extraction and motion estimation. During motion estimation, we estimate the BEV flow maps of multiple frames and output the 2D ego-motion relative to the previous frame. In feature aggregation module, we warp the feature maps of the past frames according to the corresponding BEV flow maps, and get the aggregated feature map with adaptive weights. In the module of object detection and prediction, we fuse the aggregated feature map and the object motion information, detect objects in the current frame and predict their motion offset in the future frames. Finally, the tracking can be carried out by association with simple overlap area of the current detection and the past predictions. In the following we describe our input representation first, and then introduce each module of the network in detail.

Point Cloud Processing. By constructing a BEV map, we can represent 3D space effectively and use 2D convolution for fast feature extraction. We take the point clouds from the current and the past N frames as input, and construct BEV maps separately. Following [22], we make a rough ground plane estimation to correct the height of point clouds. Then, we perform voxelization on point clouds, and build binary representation of each voxel. For each frame, we construct a binary occupancy tensor with a voxel size of 0.1m×0.1m×0.2m to speed up the following computation. When there is a point in the corresponding voxel, the value of the voxel is 1, otherwise the value is 0.

Feature Extractor. In the feature extraction stage, we apply a fairly simple network for real-time purpose. As shown in Fig. 1, for each BEV map, we perform only 4 convolutional layers and 3 max-pooling layers with 32, 64, 64 and 128 channels respectively, which result in an 8× downsampled feature map.

BEV Flow based Motion Estimation. Point clouds are sparse and unevenly distributed in 3D space, it is difficult to make fast and accurate scene flow estimation for each 3D point. Considering that most of the objects in the scene stay in the ground, we focus on estimating the horizontal motions only. Given all of the binary voxel pillars in the scene, we focus on obtaining accurate motion flow for voxels on objects.

As shown in Fig. 2, the BEV flow map describes the relative motion of the pixels on each object in X and Y directions. For each pixel in the BEV flow map, if it belongs to an object, the value is the corresponding motion offset; otherwise it belongs to the background and the value is required to be zero. For computing efficiency, an 8× downsampled flow map is used.

The resulting BEV flow maps represent the motion of objects between the current and the past N frames. Based on them, the features of objects in the past frames can be warped into the current frame by:

$$F^{(t-n)\to t}(p) = F^{t-n}(p + flow^{t-n}(p)) \ . \tag{1}$$

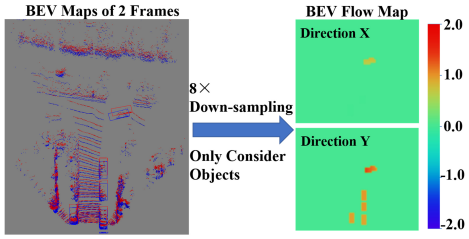

Fig. 2. BEV map (left) and the corresponding flow map (right). Voxel pillars of two neighboring frames are marked in blue and red respectively in BEV map. The BEV flow map contains the motion of the pixels on each object in direction X and Y.

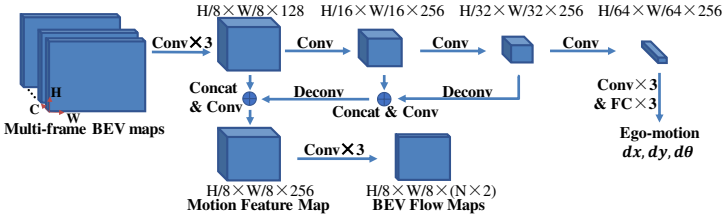

Fig. 3. Detailed structure of motion estimation module.

where $1 \leq n \leq N$, $F^{(t-n)\to t}$ is the resulting feature map warped from frame $t-n$ to t, and $flow^{t-n}(p)$ is the computed flow offset of the pixel p between frame t and $t-n$. For possible non-integer coordinates, bilinear interpolation is used for warping.

The detailed network structure of motion estimation module is illustrated in Fig. 3. Instead of performing estimation for every two frames, we concatenate the BEV maps of the current and the past N frames and extract features directly. Then, we estimate the BEV flow maps relative to the past N frames and the ego-motion on two branches simultaneously. On the BEV flow map estimation branch, we use pyramid structure to fuse features at different scales with up to 32× down-sampling. Then, after 3 layers of further convolution with 64, 32 and $N\times2$ channels, a tensor of $H/8 \times W/8 \times (N \times 2)$ corresponding to the BEV flow maps of the past N frames is obtained. The BEV flow map of each frame has 2 channels, which represent the motion offset relative to current frame in X and Y directions respectively. Another branch of this module is to estimate the ego-motion. On this branch, we perform convolution and fully-connected operations based on the 64× downsampled feature map, and estimate the motion offset $(dx, dy, d\theta)$, which represents the translation and rotation of ego-vehicle.

BEV flow branch and self-localization branch are essentially extracting the motion information of the foreground and background respecticely. They share most of the layers in the sub network, which can speed up their convergence. We can have the ground truth for these two tasks from the labeled objects and GPS respectively for supervised training.

Feature Aggregation. Because the same object have different coordinates on the successive feature maps, direct aggregation will cause severe misalignment. We use BEV flow maps to guide the warping of multi-frame features in BEV and align the features of the same object as shown in Eq. (1).

In order to further suppress the impact of residential alignment error, we apply adaptive weights to the warped feature maps before aggregation. We use a 3-layer convolution operation with shared parameters on the warped feature maps, extract 1024-dimensional feature vectors at each pixel, and calculate the cosine similarity [31] with the feature vectors of current frame. Then, the normalized weights are computed as:

$$w^{t-n}(p) = \frac{exp(s^{t-n}(p))}{\sum_j exp(s^{t-j}(p))} \ . \tag{2}$$

where $0 \leq n \leq N$, $0 \leq j \leq N$, $s^{t-n}(p)$ is the cosine similarity between the 1024-dimensional feature vectors of current frame t and past frame $t-n$ at coordinate p, w^{t-n} is the normalized weight map of frame $t-n$. Finally, the aggregated feature map $\bar{F}$ can be computed as:

$$\bar{F} = \sum\nolimits_n w^{t-n} * F^{(t-n)\to t} \ . \tag{3}$$

where $*$ represents elementwise multiplication. By accumulating multi-frame features together, the features of objects are strengthened.

Object Detection, Prediction and Tracking. We simultaneously detect objects and predict their motion offset in the next M frames in the object detection and prediction stage. As shown in Fig. 4, we apply pyramid structure and dense-block [47] on the aggregated feature map for further feature extraction. The output of the detection branch is the object classification confidence c_{obj}, location offset (t_x, t_y), object size (l, w) and object orientation ($sin\theta$, $cos\theta$), which are further defined as [22]:

$$\begin{aligned} &t_x = \frac{x_{gt} - x_{grid}}{grid}, t_y = \frac{y_{gt} - y_{grid}}{grid}, \\ &l = log(l_{gt}), w = log(w_{gt}), \theta_{gt} = atan2(\frac{sin\theta}{cos\theta}) \ . \end{aligned} \tag{4}$$

where $(x_{gt}, y_{gt}, l_{gt}, w_{gt}, \theta_{gt})$ is the ground truth, (x_{grid}, y_{grid}) is the grid location of each pixel in feature map, and $grid$ is the size of each pixel in feature map. In practice, $grid$ is set to 0.8 meter.

We further introduce the *Motion Feature Map*, i.e., the feature block before the final *BEV Flow Map* in Fig. 3, to produce the prediction of the objects' location in the upcoming frames. FAF [4] uses several predefined anchors and describes the prediction with location offset, size and heading on future frame. In contrast, we don't use any anchors and only predict the relative motion of objects for the sake of efficiency. The motion offset is described as:

$$d_{tx}^{t+m} = t_x^{t+m} - t_x, d_{ty}^{t+m} = t_y^{t+m} - t_y, d_{\theta}^{t+m} = \theta^{t+m} - \theta \ . \tag{5}$$

where $1 \leq m \leq M$, (t_x, t_y, θ) and $(t_x^{t+m}, t_y^{t+m}, \theta^{t+m})$ represent the corresponding positions of the same object in the current frame t and the future frame $t+m$.

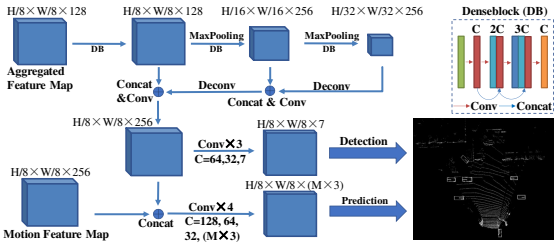

Fig. 4. Detailed structure of object detection and prediction module.

With the current detection and the past predictions at hand, it is straight forward to carry out the object tracking by simply checking the Intersection-Over-Union (IoU) on them. We first track the current objects based on the prediction of frame $t-1$ with a certain IoU threshold. If no association could be set up, we further check the corresponding predictions from frame $t-2$ to $t-M$ in order to set up the association.

3.2 Loss Function

SDP-Net implements motion estimation, object detection, prediction and tracking jointly in BEV. Therefore, as shown in Eq.(6), the loss function L is mainly composed of motion estimation loss L_{motion}, object detection loss L_{det}, and object prediction loss L_{pred}. The weights among them are determined experimentally with $\lambda_1 = 0.125$, $\lambda_2 = 0.1$, $\lambda_3 = 1.25$.

$$L = \lambda_1 L_{motion} + \lambda_2 L_{det} + \lambda_3 L_{pred} \ . \tag{6}$$

Motion Estimation Loss. The motion estimation loss L_{motion} consists of BEV flow estimation loss L_{flow} and ego-motion loss L_{pose}:

$$L_{motion} = L_{flow} + L_{pose} \ . \tag{7}$$

We apply L2 loss and increase the weights of pixels with objects for L_{flow}:

$$L_{flow} = \frac{1}{N_t} \sum\nolimits_p \lambda_4 |\widehat{flow}(p) - flow(p)|^2 \ . \tag{8}$$

where N_t is the total number of pixels in the BEV flow map, $\widehat{flow}(p)$ and $flow(p)$ seperately represent the estimation and ground truth on the coordinate p. In practice, if coordinate p belongs to an object, $\lambda_4 = 3$, if not, $\lambda_4 = 0.01$. Note that we get the ground truth of flow from supplemented annotations of KITTI RAW dataset instead of GPS, which will be described in section 4.1.

We use smooth L1 loss for L_{pose} as:

$$L1(x) = \begin{cases} 0.5x^2, & if|x| \leq 1, \\ |x| - 0.5, & otherwise. \end{cases} \tag{9}$$

$$L_{pose} = L1(|\widehat{dx} - dx|) + L1(|\widehat{dy} - dy|) + \lambda_5 L1(|\widehat{d\theta} - d\theta|) \ . \tag{10}$$

where $(\widehat{dx}, \widehat{dy}, \widehat{d\theta})$ and $(dx, dy, d\theta)$ represent the estimation and ground truth respectively, and we set λ_5=100 in practice.

Object Detection Loss. Object detection loss consists of classification loss L_{cls} and regression loss L_{reg}. We use focal loss as [48] for classification to alleviate the impact from sample imbalance, and consider all of the pixels in feature map to get L_{cls}. We use smooth L1 loss for regression. Total object detection loss:

$$L_{det} = L_{cls} + \frac{1}{N_p} \sum L_{reg} \ . \tag{11}$$

where N_p is the number of pixels corresponding to the positive samples and L_{reg} is the smooth L1 loss of $(t_x, t_y, l, w, sin\theta, cos\theta)$ for object regression.

Object Prediction Loss. We estimate object motion offset $(d_{tx}, d_{ty}, d_\theta)$ on the same pixels as object detection in feature map. We apply smooth L1 loss and consider all the positive samples in the future M frames:

$$L_{pred} = \frac{1}{N_p} \sum \sum\nolimits_m [L1(|\widehat{d_{tx}^{t+m}} - d_{tx}^{t+m}|) + L1(|\widehat{d_{ty}^{t+m}} - d_{ty}^{t+m}|) + \lambda_6 L1(|\widehat{d_\theta^{t+m}} - d_\theta^{t+m}|)] \ . \tag{12}$$

where N_p is the number of pixels corresponding to the positive samples, $1 \leq m \leq M$, and we set λ_6=100 in practice.

4 Experiments

4.1 Dataset and Implementation Details

KITTI RAW Dataset. We use KITTI RAW dataset for network training and evaluation. The KITTI RAW dataset contains 37 labeled sequences collected in different scenes with a total of 12,486 frames. The annotations of each frame include RTK-GPS data, RGB images, 64-line LIDAR point clouds and object labels with tracking information. The KITTI RAW dataset contains 34278 labeled vehicles, which makes up the vast majority of labeled object type. The other objects like bicycles and pedestrians are unevenly distributed in the sequence and have only 1550 and 2574 labels respectively, which are not enough for network training and objective evaluation. Therefore, like FAF and PIXOR, we only consider vehicles in the experiments.

Ground Truth Supplementing. The KITTI RAW dataset does not annotate objects that look highly occluded in the RGB images, although many of them can still be observed in BEV space. Since one motivation of our network is to improve the detection of those hard objects by multi-frame fusion, failing to label these occluded objects will confuse the network and degrade the credibility of evaluation. Therefore, we develop a method to complete these missed ground truth. We use GPS of ego-vehicle to project the ground truth bounding boxes from the next frame back to the current frame. If the projected bounding box contains more than 3 points in the current frame while has less than 0.1 of IoU with any of the bounding boxes in the current frame, it means the same object in the current frame are missed due to the partial occlusion and can be supplemented. Then, we manually check the increased annotations and

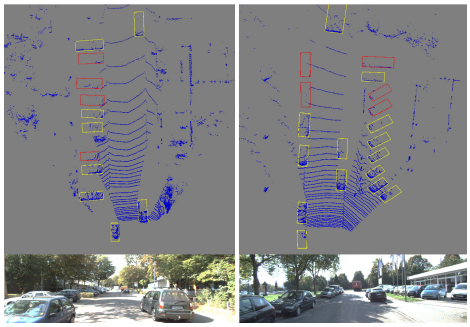

Fig. 5. Two examples of ground truth supplementing. The original annotations are marked with yellow, while the supplemented are marked with red in BEV. The RGB images below show the corresponding scenerios, respectively.

adjust the incorrect bounding boxes to eliminate the annotation error. Finally, we successfully expand the total number of vehicles from 34278 to 37906, with an increase of 10.58%. As shown in Fig. 5, for the original KITTI annotations, a certain number of objects that can be clearly observed in BEV space are missed. After ground truth supplementing, most of these highly occluded objects are found back and added. It makes our supplemented raw dataset a more difficult benchmark than the original KITTI.

Training Setup. Besides current frame, we consider the 4 recent frames of point clouds within the range of forward length Y $\in$ [0, 51.2m], width X $\in$ [-28.8m, 28.8m] and height Z $\in$ [-2m, 1m]. Thus our input is a 512×576×15×5 binary tensor which consists of X, Y, Z and time. And we predict objects' motion in 5 future frames. Then, we randomly split the KITTI RAW dataset for training and evaluation. The training set contains 24 sequences with a total of 7410 frames and the test set contains 13 sequences with a total of 5076 frames. We use NVIDIA GTX1080Ti GPU and Intel i7 CPU for both network training and evaluation. We set the initial learning rate to 0.001, the batch size to 4, the decay step to 150k, and the decay rate to 0.5. We follow [22] to apply data augmentation in training. We train the motion estimation module for 30 epochs first, then fix the module parameters, and train other modules for 60 epochs.

4.2 Evaluation Results

In the following, we evaluate the task performance of BEV flow estimation, ego-motion estimation, object detection and object tracking, respectively.

BEV Scene Flow Estimation. Some qualitative results of the estimated BEV flow maps as well as the corresponding detected objects are shown in Fig. 6, where we can see that the BEV flow of each object is well learned. The accuracy of the scene flow will dertermin the alignment of the multi-frame feature, which will further affect the performance of object detection. We compare our BEV flow estimation with the ICP or GPS based point clouds registration, as well

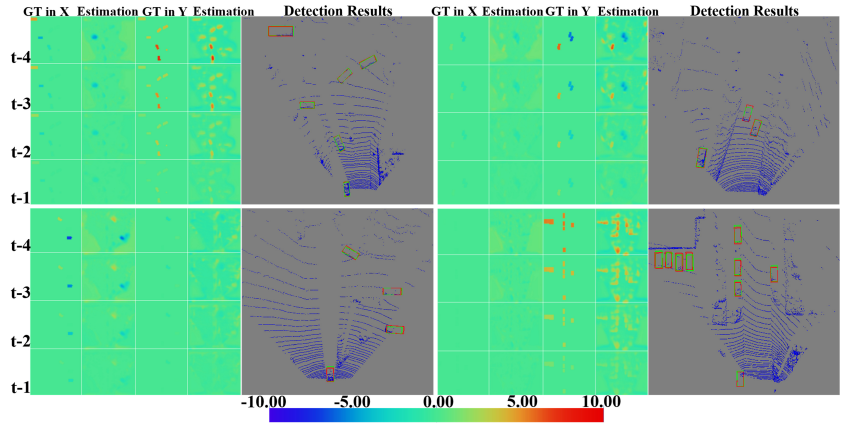

Fig. 6. Illustration of four groups of results each of which consists of a composite BEV flow image (left) and the object detection results for the current frame (right). In the BEV flow image the ground truth and the estimated flow map in X and Y direction relative to the time from t-1 to t-4 are further illustrated separately side by side. The object detection results are shown here just for reference, where the ground truth is labeled in red and the output of our network are marked with green.

Table 1. The mean alignment error of voxels in object.

Method	Dir-Agg	ICP	GPS	Ours
Mean alignment error (pixels)	2.04	0.84	0.76	0.67

as direct aggregation without any registration (Dir-Agg). The metric of mean alignment error of objects on the feature map is used, and the results are shown in Table 1. As expected, direct feature aggregation on original multi-frame point clouds produces large alignment error. Methods based on point clouds registration (ICP) can reduce the error by aligning the static foregrounds. Our method performs the best thanks to its capability of aligning the features in both static and moving objects at the same time. It even results in smaller error than the method based on GPS which shows poor performance in aligning moving objects.

Ego-motion Estimation. We further output the self-motion offset results from the motion estimation branch. We calculate the mean error of all 13 sequences on KITTI RAW test set. The resulting translational and rotational errors of ICP are 0.26 meter and 0.003 rad., comparing with ours 0.13 meter and 0.003 rad., respectively. It indicates that our module can effectively estimate ego-motion from multiple frames.

Object Detection. We first compare our approach with other methods and then conduct the ablation study. We are not able to directly compare our approach with the multi-frame approaches such as YOLO4D [2] and FAF [4], because they are neither open-source nor ranked on the public benchmark. The most relevant approach, FAF, is only evaluated on the TOR4D [4] dataset. How-

Table 2. Performance comparison of different methods on KITTI BEV benchmark.

Method	Times(ms)	mAP(%)		
		easy	moderate	hard
PIXOR [18]	35	83.97	80.01	74.31
C-YOLO [32]	60	77.24	68.96	64.95
SDP-Net-s	**12**	**86.57**	**81.93**	**75.83**

Table 3. Performance comparison of different methods on KITTI RAW test set.

Method	Input	Modality	Times(ms)	mAP(%)		
				easy	moderate	hard
F-Net [12]	Single-frame	Lidar&Img	170	89.60	82.49	70.90
MODet [22]	Single-frame	Lidar	50	90.20	87.49	84.27
SDP-Net-s	Single-frame	Lidar	**12**	86.05	81.91	79.00
ICP-based	Multi-frame	Lidar	ICP+59	90.01	84.73	82.60
SDP-Net	Multi-frame	Lidar	82	**91.51**	**88.29**	**84.74**

ever, according to the same authors' work [19], the detection accuracy of FAF is lower than PIXOR, which is a single-frame detector and ranked on KITTI benchmark. Complexer-YOLO [32] applies multi-object tracking based on a single-frame object detector, and also ranks on KITTI benchmark. Therefore, we build and train a single frame version of our model, termed SDP-Net-s for preliminary comparing purpose on KITTI benchmark. To build this baseline, we take the single-frame point clouds as input, removing all of the rest modules except the main backbone with a simplified feature pyramid structure and detection head. Then, we train SDP-Net-s on the KITTI benchmark dataset and compare it with these methods. As shown in Table 2, it can be found that our single-frame baseline has competitive performance compared with these two methods.

With this idea in mind, we further compare our approach with other multi-frame and single-frame approaches on our supplemented KITTI RAW test set. We follow the evaluation criteria of KITTI benckmark to set the IoU threshold to 0.7 and obtain the mean average precision (mAP) on three difficulty levels (easy, moderate and hard). In the hard level, all of the labeled objects are considered.

For the multi-frame approach, we build an ICP-based network for comparison. We remove the flow estimation module in our network and pre-register different frames of point clouds with ICP before input. Then, we apply the same feature extractor and adaptive weights to perform object detection. For the single-frame approaches, we train and evaluate F-PointNet [12] and MODet [22] which have more complex backbones than our approach. Considering that F-PointNet does not have an open-source 2D object detector, we use ground truth of 2D bounding boxes to get the frustum point clouds during training and evaluation. MODet is a single-stage high-precision object detector which has the state-of-the-art performance with higher detection accuracy compared to Point-RCNN [14] and PointPillars [20] on KITTI BEV benckmark.

Table 4. Performance comparison on different objects on KITTI RAW test set.

Method	Input	mAP(%)			
		Static&<30m	Static&>30m	Moving&<30m	Moving&>30m
F-Net [12]	Single-frame	83.93	48.94	70.67	45.17
MODet [22]	Single-frame	90.67	**66.57**	88.23	57.79
SDP-Net-s	Single-frame	87.34	56.60	84.20	53.58
ICP-based	Multi-frame	90.71	62.71	82.04	52.76
SDP-Net	Multi-frame	**91.54**	66.38	**89.28**	**60.75**

Table 5. Ablation study of SDP-Net on KITTI RAW test set.

Input	BEV flow map	Adaptive weights	mAP(%)		
			easy	moderate	hard
Single-frame	\	\	86.05	81.91	79.00
Multi-frame	\	Applied	83.00	81.19	79.23
Multi-frame	Applied	\	88.36	84.53	82.79
Multi-frame	Applied	Applied	**91.51**	**88.29**	**84.74**

As shown in Table 3, comparing with our single-frame baseline SDP-Net-s, F-PointNet performs worse in hard level, where lots of occluded or distant objects exist. The ICP-based network pre-registers the point clouds and has a certain improvement over our single-frame baseline. Our full approach SDP-Net outperforms the ICP-based approach by a large margin. It can adaptively aggregate the multi-frame features with unregistered point clouds input. Although residual networks with deeper network backbone are used in MODet [22], our approach still get better performance than it with a simpler backbone.

We further list mAPs of these methods according to the state and distance of objects, as shown in Table 4. For static objects, ICP-based method can perform accurate feature alignment, which is better than the single-frame method. But for moving objects, it has poor performance because of feature misalignment. Our approach uses BEV flow map to guide feature aggregation and achieves the best detection performance for most of the moving and static objects.

Ablation Study. We conduct the ablation study for SDP-Net by with or without BEV scene flow estimation or adaptive weights aggregation module. As shown in Table 5, without the guidance of BEV flow maps, the multi-frame performance is even lower than the single frame due to the feature misalignment. The adaptive weights can also help with the detection task by further reducing the residual misalignment error.

Prediction and Tracking. While detecting objects of the current frame, we also predict their motion in the next 5 frames. We compute the average distance error between the center of the predicted objects and the ground truth. The resulting mean errors in the next 5 frames are 0.18 m, 0.22 m, 0.27 m, 0.34 m and 0.39 m, respectively, which validate the accuracy of object prediction task.

We further track the objects based on the detection and prediction results, as described in section 3.1. Some qualitative results are illustrated in Fig. 7, where

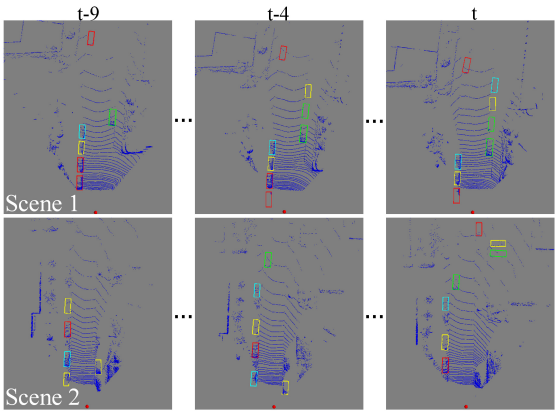

Fig. 7. Two scenes of object tracking, where the successive tracking results are listed in a row with time t-9, t-4 and t, respectively. The same object in each scene is marked with the same color.

Table 6. Results of object tracking on KITTI RAW test set.

Method	MOTA	MOTP	MT	ML
AB3DMOT [46]	78.89	84.69	74.22	6.52
SDP-Net	**82.29**	**84.96**	**81.05**	**4.66**

we can see the tracking task is fulfilled robustly. We further quantitatively evaluate the tracking performance with the metric of Multiple Object Tracking Accuracy (MOTA), Multiple Object Tracking Precision (MOTP), Mostly-Tracked (MT) and Mostly-Lost (ML) following KITTI protocol, and set IoU to 0.5 for association and 0.8 score for thresholding. As shown in Table 6, our approach can perform object tracking with better performance than AB3DMOT [46].

5 Conclusions

In this paper, we propose a novel scene flow based multi-frame multi-task network for 3D object detection and tracking. To the best our knowledge, ours is the first work which can adaptively integrating the multi-frame temporal laser data to enhance the object detection and prediction in real-time. We estimate the scene dynamics through BEV flow and then adaptively aggregate the features from multiple frames. Multiple related tasks such as ego-motion estimation, object detection, prediction and tracking are jointly performed in real time within our network. We achieve the superior performance than the state-of-the-art on the improved and more difficult KITTI RAW test set. In the future, we plan to test our approach on more datasets with more object categories, and incorporate more complex networks like PointNet for better feature extraction.

Funding: The work is supported by NSFC-Zhejiang Joint Fund for the Integration of Industrialization and Informatization under grant No. U1709214.

References

1. Levinson, J., Askeland, J., Becker, J., Dolson, J., Held, D., Kammel, S., Kolter, J.Z., Langer, D., Pink, O., Pratt, V., et al.: Towards fully autonomous driving: Systems and algorithms. In: IV. (2011)
2. El Sallab, A., Sobh, I., Zidan, M., Zahran, M., Abdelkarim, S.: YOLO4D: A spatio-temporal approach for real-time multi-object detection and classification from lidar point clouds. In: NIPS Workshops. (2018)
3. Hochreiter, S., Schmidhuber, J.: Long Short-Term Memory. Neural computation **9** (1997) 1735–1780
4. Luo, W., Yang, B., Urtasun, R.: Fast and Furious: Real Time End-to-End 3D Detection, Tracking and Motion Forecasting with a Single Convolutional Net. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, IEEE (2018) 3569–3577
5. Besl, P.J., McKay, N.D.: Method for registration of 3-D shapes. In: Sensor fusion IV: control paradigms and data structures. Volume 1611., International Society for Optics and Photonics (1992) 586–606
6. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, IEEE (2012)
7. Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: The kitti dataset. The International Journal of Robotics Research **32** (2013) 1231–1237
8. Chen, X., Ma, H., Wan, J., Li, B., Xia, T.: Multi-View 3D Object Detection Network for Autonomous Driving. In: CVPR, IEEE (2017)
9. Ku, J., Mozifian, M., Lee, J., Harakeh, A., Waslander, S.L.: Joint 3D Proposal Generation and Object Detection from View Aggregation. In: IROS, IEEE (2018)
10. Zhou, Y., Tuzel, O.: VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection. In: CVPR, IEEE (2018)
11. Yan, Y., Mao, Y., Li, B.: SECOND: Sparsely Embedded Convolutional Detection. Sensors **18** (2018) 3337
12. Qi, C.R., Liu, W., Wu, C., Su, H., Guibas, L.J.: Frustum PointNets for 3D Object Detection from RGB-D Data. In: CVPR, IEEE (2018)
13. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE (2017)
14. Shi, S., Wang, X., Li, H.: PointRCNN: 3D Object Proposal Generation and Detection From Point Cloud. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE (2020)
15. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In: NIPS. (2017)
16. Simony, M., Milzy, S., Amendey, K., Gross, H.M.: Complex-YOLO: An Euler-region-proposal for Real-time 3D Object Detection on Point Clouds. In: ECCV. (2018)
17. Redmon, J., Farhadi, A.: YOLO9000: Better, Faster, Stronger. In: IEEE Conference on Computer Vision and Pattern Recognition, IEEE (2017) 6517–6525
18. Yang, B., Luo, W., Urtasun, R.: PIXOR: Real-time 3D Object Detection from Point Clouds. In: CVPR, IEEE (2018)
19. Yang, B., Liang, M., Urtasun, R.: HDNET: Exploiting HD Maps for 3D Object Detection. In: CoRL. (2018)

20. Lang, A.H., Vora, S., Caesar, H., Zhou, L., Yang, J., Beijbom, O.: PointPillars: Fast Encoders for Object Detection From Point Clouds. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE (2019)
21. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: SSD: Single Shot MultiBox Detector. In: ECCV. (2016)
22. Zhang, Y., Xiang, Z., Qiao, C., Chen, S.: Accurate and Real-Time Object Detection Based on Bird's Eye View on 3D Point Clouds. In: 3DV. (2019)
23. Yang, Z., Sun, Y., Liu, S., Shen, X., Jia, J.: STD: Sparse-to-Dense 3D Object Detector for Point Cloud. In: ICCV, IEEE (2019)
24. Chen, Y., Liu, S., Shen, X., Jia, J.: Fast Point R-CNN. 2019 IEEE/CVF International Conference on Computer Vision (ICCV) (2019) 9774–9783
25. Shi, S., Guo, C., Jiang, L., Wang, Z., Shi, J., Wang, X., Li, H.: PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE (2020)
26. He, C., Zeng, H., Huang, J., Hua, X.S., Zhang, L.: Structure Aware Single-stage 3D Object Detection from Point Cloud. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE (2020)
27. Kang, K., Li, H., Yan, J., Zeng, X., Yang, B., Xiao, T., Zhang, C., Wang, Z., Wang, R., Wang, X., et al.: T-CNN: Tubelets with Convolutional Neural Networks for Object Detection from Videos. IEEE Transactions on Circuits and Systems for Video Technology **28** (2017) 2896–2907
28. Lee, B., Erdenee, E., Jin, S., Nam, M.Y., Jung, Y.G., Rhee, P.K.: Multi-Class Multi-Object Tracking using Changing Point Detection. In: ECCV. (2016)
29. Nam, H., Han, B.: Learning Multi-Domain Convolutional Neural Networks for Visual Tracking. In: CVPR, IEEE (2016)
30. Liu, M., Zhu, M.: Mobile Video Object Detection with Temporally-Aware Feature Maps. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE (2018)
31. Zhu, X., Wang, Y., Dai, J., Yuan, L., Wei, Y.: Flow-Guided Feature Aggregation for Video Object Detection. In: 2017 IEEE International Conference on Computer Vision (ICCV), IEEE (2017)
32. Simon, M., Amende, K., Kraus, A., Honer, J., Samann, T., Kaulbersch, H., Milz, S., Michael Gross, H.: Complex-YOLO: Real-time 3D Object Detection on Point Clouds. In: CVPR Workshops. (2019)
33. Ali, W., Abdelkarim, S., Zidan, M., Zahran, M., El Sallab, A.: YOLO3D: End-to-end real-time 3D Oriented Object Bounding Box Detection from LiDAR Point Cloud. In: ECCV. (2018)
34. Li, D., Chen, F.: DeepMapping: Unsupervised Map Estimation From Multiple Point Clouds. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE (2020)
35. Lu, W., Zhou, Y., Wan, G., Hou, S., Song, S.: L3-Net: Towards Learning Based LiDAR Localization for Autonomous Driving. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE (2020)
36. Barsan, I.A., Wang, S., Pokrovsky, A., Urtasun, R.: Learning to Localize Using a LiDAR Intensity Map. In: CoRL. (2018)
37. Behl, A., Paschalidou, D., Donne, S., Geiger, A.: PointFlowNet: Learning Representations for Rigid Motion Estimation From Point Clouds. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE (2019)
38. Houenou, A., Bonnifait, P., Cherfaoui, V., Yao, W.: Vehicle Trajectory Prediction based on Motion Model and Maneuver Recognition. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE (2013)

39. Schreier, M., Willert, V., Adamy, J.: Bayesian, maneuver-based, long-term trajectory prediction and criticality assessment for driver assistance systems. In: IEEE International Conference on Intelligent Transportation Systems, IEEE (2014)
40. Wojke, N., Bewley, A., Paulus, D.: Simple Online and Realtime Tracking with a Deep Association Metric. In: ICIP. (2017)
41. Choi, W.: Near-Online Multi-target Tracking with Aggregated Local Flow Descriptor. In: ICCV, IEEE (2015)
42. Kuhn, H.W.: The hungarian method for the assignment problem. Naval research logistics quarterly **2** (1955) 83–97
43. Xu, Y., Ban, Y., Alameda-Pineda, X., Horaud, R.: DeepMOT: A Differentiable Framework for Training Multiple Object Trackers. arXiv preprint arXiv:1906.06618 (2019)
44. Voigtlaender, P., Krause, M., Osep, A., Luiten, J., Sekar, B.B.G., Geiger, A., Leibe, B.: MOTS: Multi-Object Tracking and Segmentation. In: CVPR, IEEE (2019)
45. Osep, A., Mehner, W., Mathias, M., Leibe, B.: Combined Image- and World-Space Tracking in Traffic Scenes. In: 2017 IEEE International Conference on Robotics and Automation (ICRA), IEEE (2017)
46. Weng, X., Kitani, K.: A baseline for 3d multi-object tracking. arXiv preprint arXiv:1907.03961 (2019)
47. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely Connected Convolutional Networks. In: CVPR, IEEE (2017)
48. Ma, C., Huang, J.B., Yang, X., Yang, M.H.: Hierarchical convolutional features for visual tracking. In: ICCV, IEEE (2015)
49. Ross, T.Y.L.P.G., Dollár, G.K.H.P.: Focal Loss for Dense Object Detection. In: IEEE Transactions on Pattern Analysis and Machine Intelligence, IEEE (2017)
50. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In: NIPS. (2015)
51. Lee, N., Choi, W., Vernaza, P., Choy, C.B., Torr, P.H., Chandraker, M.: DESIRE: Distant Future Prediction in Dynamic Scenes with Interacting Agents. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE (2017)
52. Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., Savarese, S.: Social LSTM: Human Trajectory Prediction in Crowded Spaces. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE (2016)
53. Deo, N., Rangesh, A., Trivedi, M.M.: How would surround vehicles move? a unified framework for maneuver classification and motion prediction. IEEE Transactions on Intelligent Vehicles **3** (2018) 129–140
54. Kim, B., Kang, C.M., Kim, J., Lee, S.H., Chung, C.C., Choi, J.W.: Probabilistic Vehicle Trajectory Prediction over Occupancy Grid Map via Recurrent Neural Network. In: ITSC. (2017)
55. Baser, E., Balasubramanian, V., Bhattacharyya, P., Czarnecki, K.: Fantrack: 3d multi-object tracking with feature association network. In: IV. (2019)
56. Frossard, D., Urtasun, R.: End-to-end Learning of Multi-sensor 3D Tracking by Detection. In: ICRA, IEEE (2018)