Robust 3D Detection in Traffic Scenario with Tracking-Based Coupling System

Zhuoli Zhou, Shitao Chen, Rongyao Huang, Nanning Zheng

Robust 3D Detection in Traffic Scenario with Tracking-based Coupling System*

Zhuoli Zhou[1,2], Shitao Chen[1,2], Rongyao Huang[1,2], and Nanning Zheng[1,2]

[1] Institute of Artificial Intelligence and Robotics,
Xi'an Jiaotong University, Xi'an, Shaanxi, P.R.China
[2] National Engineering Laboratory for Visual Information Processing and
Applications, Xi'an Jiaotong University, Xi'an, Shaanxi, P.R.China
{zjsx5408, chenshitao, hryglory}@stu.xjtu.edu.cn;
nnzheng@mail.xjtu.edu.cn

Abstract. Autonomous driving is conducted in complex scenarios, which requires to detect 3D objects in real time scenarios as well as accurately track these 3D objects in order to get such information as location, size, trajectory, velocity. MOT (Multi-Object Tracking) performance is heavily dependent on object detection. Once object detection gives false alarms or missing alarms, the multi-object tracking would be automatically influenced. In this paper, we propose a coupling system which combines 3D object detection and multi-object tracking into one framework. We use the tracked objects as a reference in 3D object detection, in order to locate objects, reduce false or missing alarms in a single frame, and weaken the impact of false and missing alarms on the tracking quality. Our method is evaluated on kitti dataset and is proved effective.

Keywords: Multi-object Tracking · LiDAR 3D detection · Autonomous Vehicles.

1 INTRODUCTION

In recent years, autonomous driving has gradually attracted people's eyes and entered a rapid development period. Object detection and multi-object tracking technology are important components of autonomous driving technology, with which autonomous vehicles can understand the surrounding environment and make decisions. Autonomous driving usually integrates with multiple sensors, and the rich sensor information fused by multiple sensors can enhance robustness. For example, the camera can obtain the RGB texture information of the object but cannot accurately obtain the depth and 3D position information of the target. LiDAR sensor can obtain the position information of the object in the 3D space, not the texture information. By combining the information of camera

* This work was supported by the National Natural Science Foundation of China(NO.61773312,61790563).

Fig. 1. Proposed 3D detection and tracking coupling system result. Left top image and left bottom image shows the 2D detection results and the raw points cloud as input. Right image shows the 3D detection and multi-object tracking result. Objects' trajectories are represented as deep green curve, green numbers are IDs of each object.

and LiDAR, the object's RGB texture information and position information in 3D space can be obtained at the same time and the accuracy of object detection can be enhanced.

As mentioned above, the effect of the object detection algorithm has been greatly improved, but the false detection or miss detection are still urgent problems to be solved. The current mainstream object detection algorithms only consider a single frame, ignoring the connection between the upper and lower frames. Actually, the object detection of autonomous driving usually consumes a period of time. Therefore, the upper and lower frame information is beneficial to object detection. It can not only reduce the false alarms and missing alarms in a single frame, but also can locate objects in the current frame through the historical positions. To a large extent, multi-object tracking depends on the result of object detection. Namely, an efficient object detection can improve multi-object tracking. Combining these two concepts is an interesting research direction.

In this work, we propose a 3D detection and tracking coupling system to complete 3D object detection and multi-object detection tasks. We take the advantage of mature 2D object detectors and project the 2D boxes onto 3D phase to filter the frustum range of point clouds. Then we use the prediction of objects' 3D boxes which have been tracked to locate and segment the objects points in frustum point clouds. We associate the objects in this frame with the tracked objects, and determine whether false alarms or missing alarms would occur according to the tracked objects and handle if it occurs.

2 RELATED WORK

2.1 Object Detection

In this section, we will briefly review the object detection and multi object tracking. Recent years, The emergence and development of region of interests (RoIs)-based CNNs [6], [16] has generated high-confident candidates to detect,

and has greatly improved the performance of 2D Object Detection. However, 2D object detection is still insufficient in more complex scenarios, such as autonomous driving and robot, since collecting 3D data is easier than before with the help of LiDAR and other sensors. 3D object detection draws more attention, because it is more challenging and complicated than 2D version. The 3D object detection can be divided in two main categories, including detection based on raw point clouds and detection based on data conversion or combination.

Methods working on raw point clouds The VoxelNet-style methods [22], [21], [10] try to solve the issue about instance segmentation and T-Net alignment part before predicting, but they have a drawback of object unawareness in 3D point clouds. PointNet[13], PointNet++[14] propose a novel type of network architecture that predicts and segments instance directly based on 3D raw point clouds. PointPillars [10] explores pillar shape instead of the mainstream voxel design to aggregate features. PointRCNN [18] generates a 3D solution directly from the point cloud in a bottom-up manner, which has a higher recall rate than the previous method.

Methods working on data conversion or other data combination In MV3D [3], the LiDAR point clouds were projected to bird eye view (BEV), and then processed by a Faster-RCNN [16]. To generate more reliable 3D object proposals in MV3D, AVOD[9] fuses the multi-modal features. Some existing methods also use RGB-D or RGB data to improve the performance. ComplexerYOLO[19] is the first method introducing semantic segmentation into 3D object detection, which generates a better voxelized sematic point cloud used in 3D predictions afterward. F-PointNet[12] segments point cloud based on the 2D image detection result.

2.2 Multi-object Tracking

3D MOT systems are frameworks trying to detect and associate multiple identical objects in different frames. Most MOT systems follow tracking-by-detection paradigm [2][5][17], which has two steps. One is 3D object detection and the other is data association. The latter problem could be tackled from various perspectives like min-cost flow [5] [11], Markov decision processes (MDP) [20], partial filtering [2]. However, most of these methods are not trained in an end-to-end manner thus many parameters are heuristic (e.g., weights of costs). Therefore, they are susceptible to local optima. DSM [5] proposes an end-to-end tracking and matching method by accurately solving linear programming. It is rarely considered to optimize the detection part through tracking part. [8] boosts the bottom-up object detection with information integrating the top down knowledge about tracking. This method is experimentally validated in inner-city traffic scenes. Inspired by that, we consider the object detection and association as a whole, which means that we use object detection to support association and improve detection after tracking.

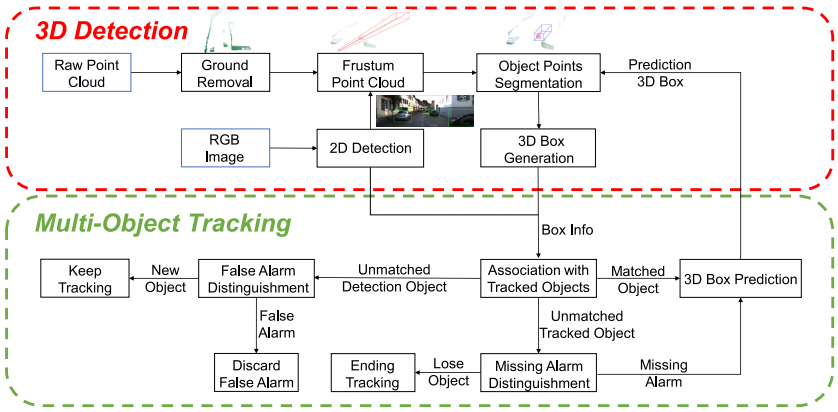

Fig. 2. Method overview. Our input data is processed in three steps. We remove ground of the raw point cloud and detect 2D boxes on image, and then generate frustum point clouds of 2D boxes. We reference the estimate 3D boxes of tracked objects and segment object points. After segmentation, we estimate the 3D boxes and push 3D boxes and 2D box info together into the tracking management. In the MOT part, detection objects are associate with tracked objects. then we handle the unmatched detection and tracked objects and predict matched objects' box.

3 3D Detection and Tracking Coupling System

3.1 Tracking-based 3D Detection

Frustum Point Cloud Generation The framework of our system is shown in Fig.2. Like the Frustum PointNets network, we first obtain the amodal 2D boxes and categories of objects on the RGB image through the proposed 2D object detector. Based on the known camera projection matrix R_{cam} and camera-to-lidar transformation matrix, we project 2D bounding boxes into the LiDAR coordinate frame and get frustums of each box. Before generating the frustum point cloud, We first preprocess the point cloud to remove the ground [7]. The purpose of ground points removal is to avoid the ground points that are considered to belong to objects located by the tracked objects. Then we filter out the non-ground points in each frustum generated by the 2D boxes.

Point Cloud Segmentation and 3D Box Estimation Let $c_i^t \in C^t$ represents the 2D detection result in t frame, $f_i^t \in F^t$ represents the point cloud of frustums in t frame. According to the vehicle positioning and heading information and the transformation of the IMU coordinate system to LiDAR coordinate system, the point cloud of the frustums is transformed into the world coordinated system. Meanwhile we predict status of the stably tracked objects $x_j \in X$ and get their positions, orientations and sizes in the world coordinated system. If the predicted object bounding box $\tilde{x}_j^t \in \tilde{X}^t$ intersects with a frustum point cloud f_i^t, the tracked objects x_i are associated with the 2D detection objects c_i^t. Since the

points of different objects in 3D space are naturally separated, we can segment object's points by the predicted bounding box. Considering the prediction error, we expand the bounding box appropriately. The formulas to judge whether point P belong to pbjects is as follows:

$$\begin{aligned} sin(\theta) * (P_y - B_y) + cos(\theta) * (P_x - B_x) > B_y - \lambda * cov_y^{1/2} - l/2 \\ sin(\theta) * (P_y - B_y) + cos(\theta) * (P_x - B_x) < B_y + \lambda * cov_y^{1/2} + l/2 \\ cos(\theta) * (P_x - B_x) - sin(\theta) * (P_y - B_y > B_x - \lambda * cov_x^{1/2} - w/2 \\ cos(\theta) * (P_x - B_x) - sin(\theta) * (P_y - B_y < B_x + \lambda * cov_x^{1/2} + w/2 \end{aligned} \tag{1}$$

Here P_x, P_y represent coordinates of the point, B_x, B_y, l, w, θ represent the center of the box, length, width and heading angle, cov_x, cov_y is the prediction error of box center got from covariance which is calucated in extended kalman filter when estimate the states of the box, λ is a parameter to control the influence from uncertainty of the center of the box. To get the 3D bounding box from the segmented object's points, we remove the points to the points center coordinate system by subtracting the x, y means of the object's points' position. Referring to Frustum PointNets [12], a preprocessed transformer network and box regression PointNets [13] are used to estimate object's amodal 3D bounding box. Since no tracking information for the first frame and the first observed objects, we apply Frustum PointNets to create 3D object bounding box.

3.2 Multi-object Tracking Management

The frustum point clouds f_i^t and the predicted 3D boxes $\tilde{x}_j^t$ are not always one-to-one correspondence. Therefore, the 2D detected objects c_i^t and the tracked 3D objects x_j need to be associated respectively. For stable tracked objects, we apply EKF to estimate the position and orientation of the objects' boxes in the current frame and use them as points segmentation input. In addition, we distinguish disappearance and appearance of objects with false alarms and missing alarms to deal with the latter two cases.

Objects Association For some reasons, such as occlusion, two or more frustum areas projected by 2D boxes may have intersection areas, or two predicted boxes may intersect with the same frustum point cloud. We need to match 2D detection boxes $c_i^t \in C^t$ with 3D objects $x_j \in X$ which they are not one-to-one associated. Because the motion of objects in video has continuity, the 2D bounding box of the same object in two adjacent frames will have a similar position and size. Besides, the objects which are occluded and further having little 2D bounding boxes. Thus we calculate the IoU between 2D box c_i^t in current frame and 2D box c_j^{t-1} that has been associated to the 3D object x_j in the last frame. We match the 2D object box c_i^t which has larger IoU and tracked object x_j together. For the associated 2D object c_i^t and tracked 3D objects x_j, we save the 2D bounding box, category, 3D bounding box and current frame ID in the queue of the tracking management's objects x_j.

Tracked Object Prediction We apply extended kalman filter to predict the state of stably tracked objects in current frame, and update the entire state of each object based on its corresponding 3D box state. We predict objects' state on the world coordinate system to make no effect on the movement of vehicle. In order to predict the state of objects more accurately, we use a constant acceleration and constant angular velocity model. We formulate the state of the 3D objects as an 10-dimensional vector $(x, y, z, \theta, vx, vy, vz, ax, ay, w)$, the variables vx, vy, vz, ax, ay represent the velocity and acceleration, w represents the angular velocity of the object, Δt represents time interval between two frames. The extended kalman filter updating model formulas are as follows:

$$x_{estimate} = x + \Delta t * vx + \Delta t^2 * ax/2 \qquad vx_{estimate} = vx + \Delta t * ax \tag{2}$$

$$y_{estimate} = y + \Delta t * vy + \Delta t^2 * ay/2 \qquad vy_{estimate} = vy + \Delta t * ay \tag{3}$$

$$z_{estimate} = z + \Delta t * vz \tag{4}$$

$$\theta_{estimate} = \theta + \Delta t * w \tag{5}$$

As a result, the element x, y, z, w of the predicted state and 3D bounding box size estimated in the last frame will be used in points segmentation as the input.

Miss and False Detection Handling As the existing objects will disappear from the field of view and new objects will enter the detection area, we set status for objects to manage the tracked objects. For the objects tracked more than five frames, we consider the objects' status is stably tracked. We continuously track the stably tracked objects, record their trajectory, and predict their position and heading. If the stably tracked object loses less than three frames, we keep predicting its position and giving a hypothetical trajectory. If the objects are associated again, we consider there is a missing alarm for detection. The missing alarm may be caused by false negative 2D detection or there are no points in the area that predict object box intersecting with frustum. First, we apply RTS algorithm to smooth object's trajectory and get more accurate location in lost frame. Let's assume object X^{ST} is tracked in frame $(0, N)$ while lost in frame $j \in (0, N)$, $[Z_1, \cdots, Z_{j-1}, Z_{j+1}, \cdots, Z_N]$ is the object detection input in other frame, we need to calculate the optimal state estimate of object $\tilde{X}_j$ in frame j. The RTS smoothing algorithm is mainly reflected in the backward filtering process, we save state vector and variance matrix estimates and predictions value $\tilde{X}_F(k \mid k)$,$\tilde{X}_F(k \mid k-1)$,$\tilde{P}_F(k \mid k)$,$\tilde{P}_F(k \mid k-1)$ (k means for frame k) calculated in extended kalman filter for every frame. We initialize the smoother, let:

$$\tilde{X}_S(N \mid N) = \tilde{X}_F(N \mid N) \qquad \tilde{P}_S(N \mid N) = \tilde{P}_F(N \mid N) \tag{6}$$

Subscript S indicates optimal smoothing and subscript F indicates kalman filter. In frame j the smoothing gain of RTS smoothing algorithm is as follows:

$$\overline{K_S(j)} = P_F(j \mid j)\Phi_{j+1,j}^T P_F^{-1}(j+1 \mid j) \tag{7}$$

In the formula (7), $\Phi_{j+1,j}$ is the jacobian matrix in extended kalman filter. And the smooth state vector and variance matrix in frame j are updated as:

$$\tilde{X}_S(j \mid N) = \tilde{X}_F(j \mid j) + \overline{K_S(j)}[\tilde{X}_S(j+1 \mid N) - \tilde{X}_F(j+1 \mid j)] \tag{8}$$

$$\tilde{P}_S(j \mid N) = \tilde{P}_F(j \mid j) + \overline{K_S(j)}[\tilde{P}_S(j+1 \mid N) - \tilde{P}_F(j+1 \mid j)]\overline{K_S(j)}^T \tag{9}$$

We can get more accurate position and heading of objects in frame from the smooth state estimate vector $X_S(j \mid N)$, and take it as a prediction input to segment objects point cloud and generate 3D box. If there is no segmented point, we use the predicted value as a result. In addition, if the stably tracked objects miss more than five frames, we suppose the objects are out of range and stop tracking, thereby we distinguish objects disappearance and missing detection, and supply the detection.

For new detected objects and unassociated objects, we set their status as trackable. If the object misses after being detected only one frame, we consider that is a false alarm and discard that. For objects which are continuously detected, the status is updated to tracked.

4 Experiments

In this section, we present experiments we have performed and analyze the results. We evaluate our methods and compare with the other multi-object tracking methods. Then we will show examples to prove that our method works.

4.1 Qualitative Evaluation

Our method is tested on the challenging KITTI Benchmark[1]. We choose Recurrent Rolling Convolution [15] as 2D detection input, and train the F-PointNets network and box estimation network in our framework on KITTI 3D detection dataset. Our method need continuous frame information, so, we evaluate the proposed detection and multi-object tracking framework on tracking dataset. To evaluate the performance of our method, we adopt the MOT metrics. We compare our approach with three MOT methods which also use LiDAR. Tab.1 shows the multi-object tracking evaluation results of our methods and other methods on test set. Our method has close performance with FANTrack on MOTA,MT,PT and ML. It indicates that our method has a similar performance on tracking accuracy and lost targets with FANTrack. Our method have a lower FRG value, which shows that our method has effect on distinguishing and supplying missing alarm. The reason for the low MOTP value may be that sometimes the tracked segment objects' points only based on predict boxes are not precise enough.

In addition to reduce the impact of the detector on performance we compare our methods with AB3DMOT using F-PointNets as 3D detectors on KITTI train set. Tab.2 shows the results of AB3DMOT using PointRCNN as input, AB3DMOT using F-PointNets as input and our methods. We can see that the detector has a great impact on the tracking results. Our method performs better than AB3DMOT on MOTA and MOTP if using a similar detector.

Table 1. Results on the KITTI tracking test set

Method	MOTA	MOTP	Recall	Precision	MT	PT	ML	IDS	FRG
Complexer-YOLO[19]	75.70	78.46	85.32	95.18	58.00	36.92	**5.08**	1186	2092
FANTrack[4]	77.72	82.33	83.66	96.15	62.62	28.62	8.77	150	812
AB3DMOT	**83.84**	**85.24**	**88.32**	96.98	**66.92**	**21.69**	11.38	**9**	224
Ours	77.22	79.00	82.91	**97.40**	62.16	29.19	8.65	145	**205**

Table 2. Compared with AB3DMOT(2) on the KITTI tracking train set

Method	MOTA	MOTP	Recall	Precision	MT	PT	ML	IDS	FRG
AB3DMOT-PointRCNN	83.35	78.43	92.17	93.86	75.68	20.54	3.24	0	30
AB3DMOT-FPointNets	76.04	78.36	80.54	**97.60**	56.22	35.14	8.65	**6**	**50**
Ours	**77.47**	**78.84**	**84.16**	96.79	**65.41**	**30.81**	**3.78**	104	193

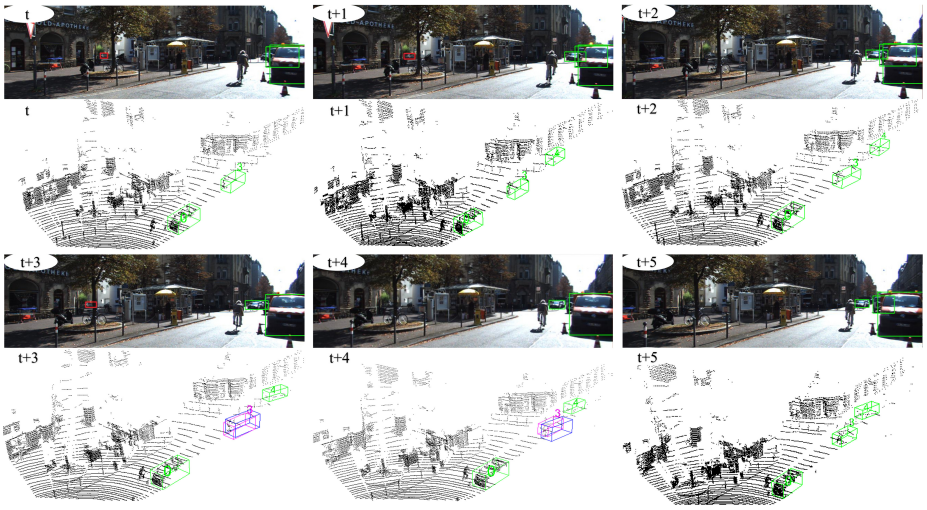

Fig. 3. Instance of where missing alarm and false alarm has been detected and correctly handled. The green 2D boxes are true positive results, red 2D boxes in frame $t, t+1$ and $t+3$ are discarded, because no points belong to objects in the frustum. The green 3D boxes are true positive 3D detection obtained from 2D detection and predicted 3D boxes. The purple 3D boxes in frame $t+3$ and $t+4$ are supplied objects of which 2D detection is missing. The blue 3D boxes in frame $t+3$ and $t+4$ are ground truth.

4.2 Performance

Fig. 3 shows an example about the handle of missing alarm and false alarm in our framework. The object with ID 3 is tracked stably before frame $t+3$, but lost 2D detection in frame $t+3$ and $t+4$. We keep tracking the object and predict its position and trajectory. It is associated again in frame $t+5$ and frames after $t+5$, so we consider missing alarm of the object with ID 3 happend. We smooth the predicted location in frame $t+3$ and $t+4$ by object's location after frame

$t+5$ and supply its 3D detection in frame $t+3$ and $t+4$. The purple 3D boxes are the results and the blue 3D boxes are ground truth. In frame $t, t+1$ and $t+3$,the red 2D boxes are discarded as false alarm for lacking LiDAR points.

5 Conclusion

In this paper, we present a 3D detection and tracking coupling framework. We achieve 3D detection and multi-object tracking through a 2D detection result. And our framework can effectively reduce miss alarm and false alarm in a single frame. Our method still has a long way to go. Though our method and 2D detector are independent, more precise 2D detector can bring superior performance. The points segmentation method and 3D box estimate network in our framework can also be further improved. We consider to improve object detection performance and focus more on the connection between tracking and object detection, because both are important to autonomous driving. Our future work will include improving the 2D detector and box estimation network, making better using of tracking information, and fully fusing camera and LiDAR data.

References

1. Andreas Geiger, P.L., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: CVPR (2012)
2. Breitenstein, M.D., Reichlin, F., Leibe, B., Koller-Meier, E., Van Gool, L.: Online multiperson tracking-by-detection from a single, uncalibrated camera. IEEE transactions on pattern analysis and machine intelligence **33**(9), 1820–1833 (2010)
3. Chen, X., Ma, H., Wan, J., Li, B., Xia, T.: Multi-view 3d object detection network for autonomous driving. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1907–1915 (2017)
4. E. Baser, V. Balasubramanian, P.B., Czarnecki, K.: Fantrack: 3d multi-object tracking with feature association network. In: IEEE Intelligent Vehicles Symposium (2019)
5. Frossard, D., Urtasun, R.: End-to-end learning of multi-sensor 3d tracking by detection. In: 2018 IEEE International Conference on Robotics and Automation (ICRA). pp. 635–642. IEEE (2018)
6. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 1440–1448 (2015)
7. Himmelsbach, M., v. Hundelshausen, F., Wuensche, H.J.: Fast segmentation of 3d point clouds for ground vehicles. In: IEEE Intelligent Vehicles Symposium, Proceedings. pp. 560–565 (2010)
8. Himmelsbach, M., Wuensche, H.J.: Tracking and classification of arbitrary objects with bottom-up/top-down detection. In: 2012 IEEE Intelligent Vehicles Symposium. pp. 577–582. IEEE (2012)
9. Ku, J., Mozifian, M., Lee, J., Harakeh, A., Waslander: Joint 3d proposal generation and object detection from view aggregation. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 1–8. IEEE (2018)
10. Lang, A.H., Vora, S., Caesar, H., Zhou, L., Yang, J., Beijbom, O.: Pointpillars: Fast encoders for object detection from point clouds. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 12697–12705 (2019)

11. Lenz, P., Geiger, A., Urtasun, R.: Followme: Efficient online min-cost flow tracking with bounded memory and computation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4364–4372 (2015)
12. Qi, C.R., Liu, W., Wu, C., Su, H., Guibas, L.J.: Frustum pointnets for 3d object detection from rgb-d data. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 918–927 (2018)
13. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 652–660 (2017)
14. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: Advances in neural information processing systems. pp. 5099–5108 (2017)
15. QiongYan, J.X.J.W.J., LiXu, Y.W.: Accurate single stage detector using recurrent rolling convolution. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2017)
16. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in neural information processing systems. pp. 91–99 (2015)
17. Sharma, S., Ansari, J.A., Murthy, J.K., Krishna: Beyond pixels: Leveraging geometry and shape cues for online multi-object tracking. In: 2018 IEEE International Conference on Robotics and Automation (ICRA). pp. 3508–3515. IEEE (2018)
18. Shi, S., Wang, X., Li, H.: Pointrcnn: 3d object proposal generation and detection from point cloud. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–779 (2019)
19. Simon, M., Amende, K., Kraus, A., Honer, J., Samann, T., Kaulbersch, H., Milz, S., Michael Gross, H.: Complexer-yolo: Real-time 3d object detection and tracking on semantic point clouds. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. pp. 0–0 (2019)
20. Xiang, Y., Alahi, A., Savarese, S.: Learning to track: Online multi-object tracking by decision making. In: Proceedings of the IEEE international conference on computer vision. pp. 4705–4713 (2015)
21. Yan, Y., Mao, Y., Li, B.: Second: Sparsely embedded convolutional detection. Sensors **18**(10), 3337 (2018)
22. Zhou, Y., Tuzel, O.: Voxelnet: End-to-end learning for point cloud based 3d object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4490–4499 (2018)