2021 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom)

# Polarmask-Tracker: Lightweight Multi-Object Tracking and Segmentation Model for Edge Device

Xiaoyun Dong[†*], Zhenchao Ouyang[*†], Zeling Guo[‡*], Jianwei Niu[*†]
[*] Hangzhou Innovation Institution, Beihang University,
Chuanghui Road #18, Binjiang, Zhejiang 310000, China
[†] State Key Laboratory of Virtual Reality Technology and Systems,
[‡] Sino-French Engineer School, Beihang University,
Xueyuan Road #37, Haidian, Beijing 100191, China
Email: ouyangkid@buaa.edu.cn

***Abstract*—The image or video input from the camera is one of the important data sources for unmanned vehicles to perceive the environment. However, the 2D/3D bounding box can only provide a very coarse approximation because one box often contains other targets and background. In order to solve the problem of precise target tracking and computing limitations of edge devices, this paper proposes Polarmask-Tracker, a lightweight segmentation-based multi-object tracking network for vehicular edge devices. Polarmask-Tracker extended the lightweight Polarmask segmentation head with tracking vector. The polar mask replaces the traditional mask prediction by regression of a group of fixed edge points in polar coordinate system, which can greatly optimize the computational complexity and regression difficulty of the mask. With an additional tracking vector branch generated based on mask, the model can learn tracking tasks in an end-to-end manner. Finally, we further accelerated the entire model based on TensorRT and achieve real-time tracking on mobile edge computing platform. Different from previous evaluations on the ImageNet and COCO datasets, this study uses the KITTI tracking dataset to extend the instance segmentation task to segmentation tracking, also called MOTS. At the same time, the target scales captured from the autonomous vehicle camera are usually smaller, which also brings additional challenges. Evaluations on NVidia Jetson AGX show that the final Polarmask-Tracker can achieve 122.55 FPS, 46.57 mAP for mask segmentation, 56.418 HOTA for tracking.**

***Index Terms*—Segmentation, tracking, deep learning, MOTS, autonomous vehicle**

## I. Introduction

Developing accurate, efficient and robust visual perception modules to replace human driver is essential for intelligent mobile platforms, such as self-driving vehicle, unmanned aerial vehicle and service robot [1]. Related modules can be integrated into smart roadside infrastructures to assist autonomous driving on the road [2] as a part of V2X network system. Benefit from the early development of computing [3], [4], sensors, Internet of Things (IoT) and big data technologies [5], [6], currently it is very convenient to obtain large-scale annotated vision task datasets [7]–[9] and carry out end-to-end model design, training and evaluation [10]–[13].

Early deep learning based Multi-Object detection and Tracking (MOT) tasks mainly focus on predicting 2D or 3D rectangular bounding boxes and related classes [14], [15]. However, the bounding box level annotations contains both object and backgrounds when the targets block each other [16]–[18]. Therefore, related tracking systems can only offer limited performances. On the other hand, self-driving vehicle in the road scene needs to interact with other moving targets, which requires high safety and accuracy. Pixel-level segmentation is usually needed to help improve the accuracy of target recognition and inter-frame matching. Recent research tendencies extend bounding-box to segmentation with tracking, and propose multi-object tracking and segmentation (MOTS) task [19]–[21].

However, the regression and prediction of pixel-level masks and instance ID is much complex than bounding boxes with fixed corners and center [22], [23]. Because masks often contains irregular outline, indefinite number of corners and unstable center. This also restricts the deployment of MOTS models on in-vehicle edge devices and provides real-time tracking results. After the milestone Mask-RCNN [24] was proposed, a series of optimizations aimed at backbone network optimization [20], [25]–[29], segmentation head [26], [30], [31] improvement, overall calculation acceleration [32]–[34] and model compression [35]–[37] to improve the segmentation network, but the research on MOTS tasks has just begun. In special situations, such as highways, autonomous vehicles need to track and predict the motion trajectory of the targets at a long distance [38], [39]. The target scale is smaller than normal MOTS tasks, this also pose new challenge for the design of lightweight and real-time model architectures.

For the purpose of designing real-time and light-weight MOTS model for vehicular edge computing devices, this paper proposed the Polarmask-Tracker that combines Resnet101 backbone with Polarmask [40] segmentation head. We introduced the single-shot Polarmask to further accelerate and simplify the regression of the mask. Polarmask can be considered as a manner to reduce the dimensionality of the irregular mask regression problem to fixed corner points (fixed vector of radius-angle tuple from local polar coordinates). Although the polar representation based mask is still more complicated than bounding box, but compared with the traditional point-wise mask generation, the efficiency is greatly improved. An additional tracking vector branch with Tripletloss is then attached

Xiaoyun Dong and Zhenchao Ouyang are co-first authors of the article.

978-0-7381-2646-3/21/$31.00 ©2021 IEEE

DOI 10.1109/ISPA-BDCloud-SocialCom-SustainCom52081.2021.00100

based on the mask generation for end-to-end inter-frame tracking. Yolact-Edge tries to fuse key frame convolutional weight and use TensorRT to accelerate the overall deduce speed based on the Yolact. And finally, we deployed the Polarmask-Tracker on NVidia Jetson AGX platform, evaluated it based on the self-driving scenario dataset of KITTI MOTS, and accelerate the overall model based on TensorRT. The contribution of this paper are summarized as follows:

- We replaced the per-instance coefficients mask generation strategy of Yolact-Edge with a simplified Polarmask representation to achieve fast instances segmentation.
- An additional tracking vector branch is added to extend the segmentation model to MOTS task, and enabled model learning follows the end-to-end fashion. To further optimize the tracking performance, the Tripletloss and the introduction of random negative samples are used to optimize the inter-class distance of the generated tracking vector.
- The TensorRT is adopt to accelerate the overall Polarmask-Tracker model on both GPU server and NVidia Jetson AGX platform.

The rest of this paper is organized as follows. Section II briefly reviews the recent studies on segmentation model refinement and related progress of MOTS. The detailed model of Polarmask-Tracker is presented in Section III. Section IV evaluates the model on the public self-driving dataset of KITTI MOTS. Section V discusses the current model limitation and possible improvements in the future.

## II. Related works

### A. lightweight Segmentation

Compare with normal computer version tasks [41], [42], segmentation offers more detailed detection results, such as panoramic, instance or semantic-level segmentation. However, the computation of pixel-level classification also comparatively complicated, which is the main bottleneck that mainly restricts the deployment of related CNN models on mobile terminals with low power consumption and computing power. To achieve lightweight and real-time, while taking into account segmentation accuracy, recent research can be summarized into three aspects: optimization model backbone network, head segmentation strategy, and special learning mode.

Among the many improvements, the most universal solution is to improve feature extraction, reorganization, and squeezing through model backbone optimization (for most deep learning tasks). At the same time, the efficient backbone network of other image processing tasks can be directly used for optimizing segmentation tasks. Fully Convolutional Network (FCN) [43]–[45] removed the fully connected layer, back-propagation [46] and pooling modules to reduce calculation and details lost by downsampling. U-Net encoder-decoder model [47] and feature pyramid network (FPN) [48] try to optimize the final mask prediction based on multiple scales of feature maps. Other improvements, such as incorporation of dilated convolution [49], attention mechanism [50], PONO feature [20] or deformable convolution [51], [52], mainly focus on improve the efficiency and effectiveness of convolutional encoding, and will more or less increase the overall computational load. YolactEdge [32] reused the convolutional features of specify the layer from keyframes, this helps alleviate the computational load brought by the layer with a large amount of floating-point operations. However, it lacks a reasonable key frame selection strategy.

Early segmentation head refinement tries to introduce Conditional Random Fields (CRFs) [53]–[55] and Markov Random Field (MRFs) [56], [57] as probabilistic graphical models. CRFs and MRFs can learn and predict the pixel-level classification with spatial association of features, and achieve better localization property and accurate boundaries. CenterMask [58] attached a multi-task segmentation head on the CNN backbone, and predict heat-map and offset of the target center, and use size, shape and saliency to refine the target BBoxes and masks. In this way, the additional process time on NMS and region proposal augmentation can be saved. The prediction head of Yolact [30] uses parallel subtasks of prototype masks generation and per-instance mask coefficients prediction as YOLO do for detection task, and linearly combining the prototypes with the mask coefficients as final result. They also allows previous removed detections to suppress other detections and speed up traditional NMS. Yolact++ [26] first incorporated deformable convolutions on small scale feature maps, and refined the prediction head with optimized anchor scales, aspect ratios and fast mask re-scoring branch [59]. Polarmask crafty converts the pixel segmentation into a problem of predicting fixed boundary points in the polar coordinate system, which greatly reduces the computational load of the head. The only drawback is that the polar mask cannot deal with target with non-convexity polygon outline or holes. Combining of the anchor-free and proposal-free object detection [60] with spatial attention-guided (SAG)-mask together, another Centermask [61] only need to deploy segmentation on each detected box. This also heavily simplified the loss design and training phase.

Although the above model claims to be able to obtain real-time segmentation speed, only YolactEdge provides speed estimates for edge computing devices (i.e., 30 FPS on Jetson AGX Xavier and 170 FPS on RTX 2080 Ti) with TensorRT optimization.

### B. MOTS models

The most simple way to extend MOTS model from MOT is by adding a segmentation sub-branch to the prediction head, and use the mask to reduce the mutual interference between foreground targets and background features. Moreover, the Track R-CNN [19] further utilized 3D convolutions to generate temporally enhanced context feature from continues frames. A new tracking-by-points paradigm is presented in PointTrack [62], the model first transformed image into un-ordered 2D point cloud representation and learning instance embedding from randomly selected points with point-wise MLP encoding

modules. Considering that the MLP network architecture used is very shallow, the entire network speed is very efficient.

The APOLLO MOTS dataset with higher instance density is released along this work. PanopticTrackNet [63] introduced a shared backbone with the 2-way FPN to deal with continues image frames, and a complex multi-object head, i.e., semantic segmentation, instance segmentation and instance tracking. In the end of their model, a Multi-Object Panoptic Tracking (MOPT) fusion module combine all the results from each branches and generate semantic labels, instance IDs and tracking ID all together. Finally vision-based PanopticTrackNet can achieve 146ms (<10Hz) per frame on the unity-based virtual KITTI 2 dataset with a 4 RTX 2080ti GPU server. Combining a modified variational autoencoder (VAE) with the Mask R-CNN [24], the one-encoder-three-decoder (namely the auxiliary branch, proposal branch and augment branch) model is used to achieve video MOTS task. In order to ensure efficiency, the three decoder branches shared the same weights but perform different tasks, i.e., future frame prediction, connect object-level information over time, and aggregates pixel-level features. Considering the complexity of the model, Mask R-CNN and the added VAE module, the speed of the model may not be fast. In [64], the authors introduce Gaussian mixture probability hypothesis density (GMPHD) filter is used to associate and process the results of interframe segmentation (from Mask R-CNN). The GMPHD filter is used to build position and motion affinity and the Kernelized correlation filter (KCF) is used to build appearance affinity for the targets in consecutive frames. The entire back-end detection and tracking are independent of the segmentation model, so the parameters of the relevant filters are designed based on manual tuning.

## III. Polarmask-Tracker

The most fundamental purpose of designing Polarmask-Tracker is to meet the requirements of lightweight, real-time multi-target tracking for mobile edge devices in the vehicle environment. For this purpose, we extended the Polarmask segmentation head with an additional tracking vector branch for MOTS task. In detail, we removed the polar centerness branch but directly used the centroid function based on the generated heat-map-per-class. The boundary points of polar mask is then predicted based on the centroid, and the fine-tuned mask features are used to generate the tracking vector with a Tripletloss function, the final model architecture is illustrated in Fig. 1. Considering the Resnet101 backbone and FPN are common paradigms for deep learning models, we only introduce the optimization and improvement of segmentation and tracking head in detail.

In general, the MOTS head can be mainly divided into three main functions, coarse prediction, fine tuned mask generation and tracking vector generation. We also accelerate the whole model with TensorRT for vehicular edge computing platform after determining the overall framework.

### A. Coarse heat-map prediction

As we generate FPN from the Resnet101 backbone, we first predict a coarse prediction for each target class with a binary heat-map at different scales of the feature maps. Because accurate mask prediction needs to fit a complex loss function and brings computational load. The binary heat-map only offers coarse masks of each target, and are then used for generate centroids. This can be easily achieved using off-the-shelf OpenCV contour extraction functions.

Commonly, the target center and 2D bounding box regression are the basic branches for a multi-target head. The Polarmask also needs the information of the center point to place the local coordinate system. Both the bounding box and center regression are removed in our model for the following two reasons. First, the overall MOTS task is too complicated, the final loss function is a combination of bounding box/width-height, centerness, mask, class and tracking distance. A complex objective function not only increases the difficulty of model training, that is, using the same scale parameters to fit more sub-objectives, but also increases the computational load. Second, we compared the calculated centroid from the heat-map and the prediction results from an additional centerness branch, the center coordinates are very similar and have little effect on the segmentation results. Third, the polar mask representation can be treat as a upgraded version of bounding box corners or width-height pair, we process it in the refined segmentation step.

### B. Refined Polarmask prediction

The Polarmask generation module places the polar coordinate at the target centroid $(x_c, y_c)$ generated based on the coarse heat-map result, and predicts a refined fixed mask coordinates as array of Radius ($R$) at fixed Theta ($\theta$): $[R, \theta]$. While the $\theta$ rotates clockwise and increases from 0 to $2\pi$ ($\theta \in [0, 2\pi]$). Starting from the $0°$, $n$ rays with fixed angle interval/angle resolution $\Delta\theta = 10°$ (we use the same configuration as in [40]) are used to find the $n$ intersections with the target contour. Hence, Polarmask predict 36 points for each target, and the optimized target mask can be obtained by connecting these 36 points from $0°$ in a clockwise order (as shown in Fig. 2).

Fig. 3 illustrated a comparison between a generated Polarmask with $30°$ angle resolution (that is a 12-corner-mask) and groundtruth mask. Moreover, bounding box can be regarded as the most extreme case with an angular resolution of $90°$. In practice, the smaller the angular resolution used, the more likely it is to predict results close to GT, and vice versa. However, more corner points also mean an increase in the complexity of calculation and model training.

And the IoU (the ratio of interaction area over union area between the predicted mask and ground truth) and related IoU Loss can be defined as Eq. 1 and Eq. 2. Where the $N$ is the total number of corner points of the Polarmask, the mask area can be treat as the total areas of $N$ triangle (with fixed $\theta = 2\pi/N$), $r_{min}$ and $r_{max}$ are the minimum and maximum

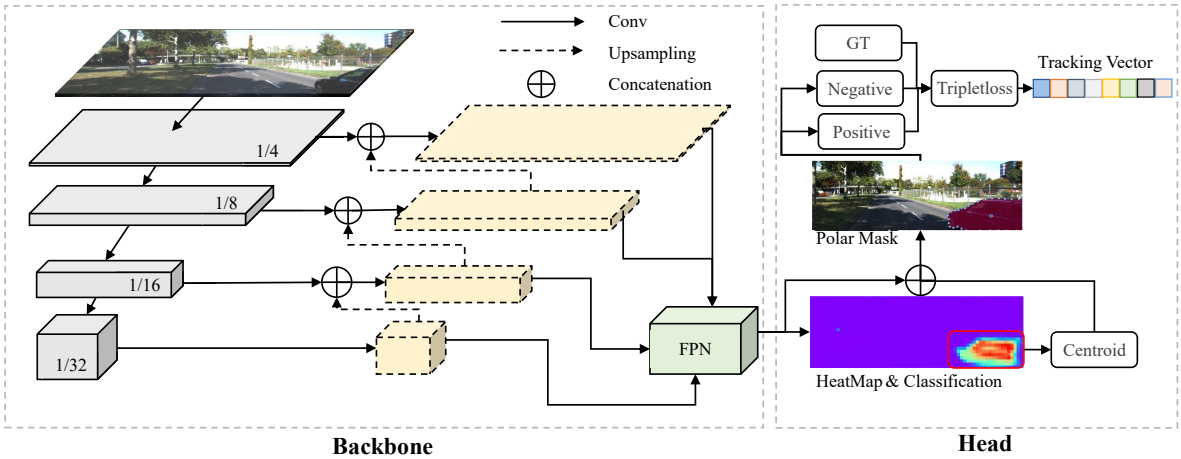



Fig. 1. Polarmask-Tracker with Resnet101 as backbone and generate feature pyramid network (FPN) for segmentation and tracking head. The head first predicts a heat-map for each class of the target, and calculates centroid. The centroid is used for prediction of the polar masks. Finally, we use the fine-tuned mask features from consecutive frames to training the Tripletloss and generate tracking vectors for each target.

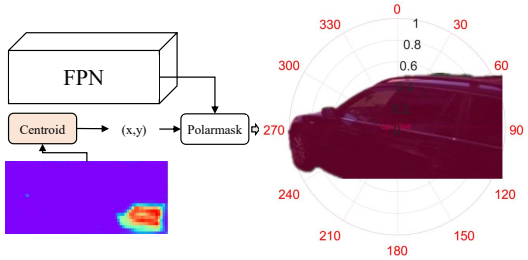



Fig. 2. Refined Polarmask prediction tries to predict an array of corners ($[R, \theta]$) based on the polar coordinate at centroid $(x_c, y_c)$ with FPN features.

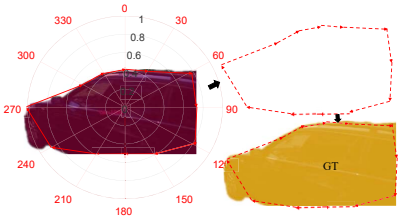



Fig. 3. Example of 12-point-Polarmask (dotted lines) compare with the GT (yellow mask).

values of the target ray radius $d$ and predicted ray radius $d^*$, respectively.

$$IOU = \lim_{N \to \infty} \frac{\sum_{i=1}^{n} \frac{1}{2} r_{min}^2 \Delta\theta_i}{\sum_{i=1}^{n} \frac{1}{2} r_{max}^2 \Delta\theta_i} \tag{1}$$

$$Polar\ IOU\ Loss = log \frac{\sum_{i=1}^{n} r_{min}}{\sum_{i=1}^{n} r_{max}} \tag{2}$$

As mentioned before, the Polarmask representation has a drawback that it cannot deal with non-convexity polygon or target with holes. Luckly, in self-driving scenario, the targets are mainly convexity polygons, and the vehicle is a rigid body with prominent contours. Therefore, the combination of Polarmask instance segmentation and tracking is perfect for on-road target tracking.

### C. Tripletloss-based tracking vector

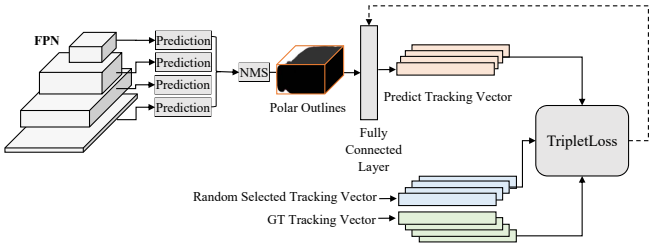



Fig. 4. Tripletloss based tracking vector generation module.

*1) **Tracking vector**:* As the backbone generate a FPN with four scales of feature maps ($[B, H/S, W/S, C]$, $B$ is the batch size, $H$ and $W$ are the width and Height of the image, $S$ is scale of the feature map, and $C$ is the channel number), we have four polar masks at each level. The problem is how to embedding masks with different scales and outlines into fixed tracking vectors.

To achieve this, we simply cropped each mask with an adjacency rectangle and resized to a fixed size (the median length and width of the BBox obtained by statistics of the data set). Features outside the mask range are filled with 0. Then the feature maps of the mask are concatenated along the channel dimension as a combined feature map. A fully connected layer is then used to generate a 128-dimension tracking vector.

*2) **Tripletloss**:* We introduce the Tripletloss [65] Re-ID task to refine the generation of the tracking vector in the embedding space, and train this module in batch mode. This helps the model not only learning metrically close features belong to the same targets, but also learning metrically distant features that apart different targets. We refer the Lifted Embedding loss as Eq. 3 presented in [65] as our Tripletloss. Where $D(a, b)$ as the distance function between vector $a$ and vector $b$, and we refer squared Euclidean distance $D(f_\theta(x_i), f_\theta(x_j))$ as distance. $x_a^i$ corresponds to the $a$-th frame of the $i$-th target, and $m$ is the margin. $P$ and $N$ correspond to the positive and negative targets in current batch.

$$\pounds_{LG}(\theta; X) = \sum_{i=1}^{P} \sum_{a=1}^{N} [log \sum_{p=1, p \neq a}^{N} e^{D(f_\theta(x_a^i), f_\theta(x_p^i))} - log \sum_{j=1, j \neq i}^{P} \sum_{n=1}^{N} e^{m - D(f_\theta(x_a^j), f_\theta(x_n^j))}]_+. \quad (3)$$

For implementation details, we select four consecutive frames as a batch for NVidia GeForce 2080ti during training. Therefore, we have four positive vectors for each target, and a group of 0 4 negative vectors (for there may no other targets in current batch). Each tracking vector is a 128-length array.

*3) **Hungary tracking**:* The elements for current Hungary algorithm matrix is determined base on the number of tracks, i.e., the targets in current frame. Once a target is missing or added in the next frame, a similarity matrix is constructed based on the tracking trajectory and the targets in the current frame, and the first $n$-th targets with the highest similarity are selected as the input of the Hungarian algorithm. The Euclidean distance is used to measure the target distance.

## IV. Experimental Study

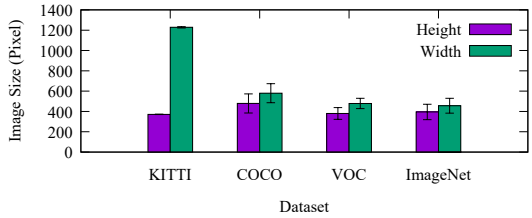
### A. MOTS dataset

MOTS task for self-driving vehicle is often more challenge than normal computer vision researches, because the target size (both bounding boxes and masks) are smaller while the camera input are much larger. We have carried out statistical analysis on the image and target size of KITTI MOTS [19], and compared it with related public datasets, i.e., COCO [8], VOC2012 [66] and ImageNet [7]. From Fig. 5(a), it can be seen that the width of the images for each dataset are very close, however, the height of the images of KITTI MOTS is about twice of the others. And from the bottom, Fig. 5(b), the boxplot (the y-axis uses log coordinates with base 2) of the kilo-pixel number (#$k$) of 2D Bounding Box (BBox) and mask are illustrated, respectively. Commonly, the BBox is the external quadrilateral of the corresponding mask, therefore, it will contain the pixels of the background and other targets, and the amount of pixels is larger. The Q1, median and Q3 of #$k$ pixels per label of KITTI MOTS has a lower scales (i.e., mean=6.3k and median=3k for BBox, mean=4.1k and median=1.7k for mask) than the other three datasets. The relatively large-scale input increases the computational pressure of the platform, while the features of smaller-scale targets are relatively limited, which increases the difficulty of recognition. Please refer [1] for detail.

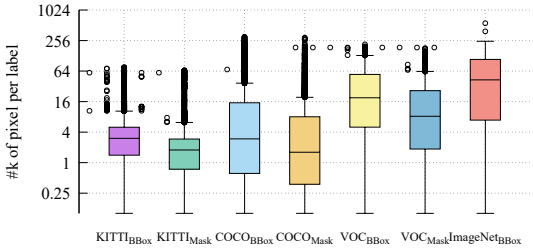
For the safety reasons, the designed model for self-driving platforms needs to efficiently deal with large field of view input images ($>$ 10 Hz), while ensure the detection and tracking of all small targets as much as possible.

### B. Segmentation result

Table I lists the model performances on instance segmentation task and 2D BBox prediction are also contained. The overall segmentation and tracking time is calculated for the frame per second (FPS) of Polarmask-Tracker, while the others only considered segmentation. It can be seen that without TensorRT acceleration, the Polarmask achieves the highest




(a) Compare of image sizes from different dataset.




(b) Distributions of the BBox and Mask.

Fig. 5. Comparison of the input image size (upper), 2D BBox and Mask pixel (bottom) of KITTI MOTS and normal computer vision dataset, i.e., COCO, VOC2012 and ImageNe.

FPS (97.09) on KITTI MOTS dataset. Our Polarmask-Tracker (modified based on Polarmask to deal with MOTS task) is only slightly slower (93.81 FPS), however, when deploying the TensorRT (**Float16** version) the FPS achieves nearly 631, this is about $\times$6.7 speed-up ratio, $\sim$ 1968 times faster than Track R-CNN and $\sim$ 47 times faster than our previous research [20]. As we removed the BBox and centerness prediction branch, the 2D BBox boxes are generated based on the mask results. The mean-Average Precision (mAP) and the AP50 (only count the targets with IoU threshold larger than 0.5) of our model also achieves the second highest place (43.91 mAP and 76.05 $AP_{50}$ for BBox, 46.57 mAP and 76.05 $AP_{50}$ for Mask).

TABLE I
Segmentation results of the state-of-the-art models.

| **Method** | **FPS** (2080Ti) | **Instance Segmentation** | | | |
|---|---|---|---|---|---|
| | | BBox | | Mask | |
| | | mAP | $AP_{50}$ | mAP | $AP_{50}$ |
| Yolact [30] | 35.60 | 34.17 | 55.01 | 32.59 | 52.61 |
| Yolact-Edge [32] | 52.08 | 29.74 | 52.40 | 29.85 | 48.77 |
| Polarmask [40] | **97.09** | 27.43 | 57.01 | 27.70 | 55.90 |
| Track R-CNN [19] | 3.12 | 41.06 | 68.27 | 46.04 | 68.25 |
| [20] | 13.37 | **57.83** | **78.93** | **57.40** | **77.86** |
| Ours | 93.81 | **43.91** | **76.05** | **46.57** | **76.05** |
| Ours (w TensorRT) | **630.91** | 43.85 | 76.05 | 46.57 | 76.05 |

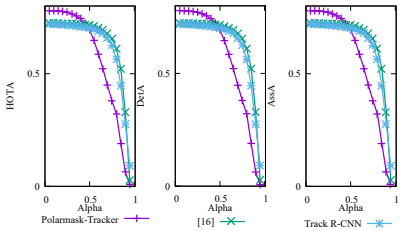



Fig. 6. Curves of HOTA, DetA and AssA for the compared models.

TABLE II
TRACKING PERFORMANCES ON THE KITTI MOTS DATASET.

| Model | HOTA | DetA | AssA | DetRe | DetPr | AssRe | AssPr | LocA | RHOTA | HOTA(0) | LocA(0) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| *Polarmask+Center* | 19.939 | 55.268 | 7.4066 | 61.635 | 69.685 | 13.843 | 19.604 | 77.884 | 21.222 | 27.778 | 70.415 |
| *Polarmask+Embed* | 56.418 | 57.497 | 56.505 | 63.984 | 71.46 | 61.814 | 75.407 | 78.866 | 59.843 | 77.867 | 72.266 |
| *TrackRcnn* | 62.332 | 65.079 | 60.152 | 72.582 | 80.486 | 62.946 | 88.105 | 87.403 | 66.009 | 71.876 | 84.628 |
| [20] | 63.839 | 65.89 | 61.994 | 68.161 | 88.817 | 74.773 | 74.722 | 88.804 | 64.995 | 72.362 | 87.306 |

Fig. 7. Prediction of circular masks when dealing incomplete large vehicles with Polarmask-Tracker.

$$\begin{aligned} HOTA_\alpha &= \sqrt{\frac{\sum_{c\in\{TP\}} A(c)}{|TP|+|FN|+|FP|}} \\ A(c) &= \frac{|TPA(c)|}{|TPA(c)|+|TNA(c)|+|FPA(c)|} \\ HOTA &\approx \frac{1}{19}\sum_{\alpha\in[0.1:0.1:0.9]} HOTA_\alpha \end{aligned} \tag{4}$$

TABLE III
MODEL ACCELERATION BASED REFINE SEGMENTATION HEAD.

| Head | BBox | | Mask | | FPS |
|---|---|---|---|---|---|
| | mAP | AP-50 | mAP | AP-50 | FPS |
| **Yolact** | 34.17 | 52.61 | 32.59 | 52.61 | 35.6 |
| **Polarmask** | 43.70 | 72.39 | 46.32 | 72.39 | 91.17 |
| **Polarmask-refine** | **43.91** | **76.05** | **46.57** | **76.05** | 93.81 |

TABLE IV
MODEL ACCELERATION BASED ON TENSORRT.

| TRT | BBox | | Mask | | FPS | |
|---|---|---|---|---|---|---|
| | mAP | $AP_{50}$ | mAP | $AP_{50}$ | 2080Ti | AGX |
| **FP32** | 43.91 | 76.05 | 46.57 | 76.05 | 93.81 | 20.42 |
| **FP16** | 43.85↓ | 76.05 | 46.57 | 76.05 | 630.91↑ | 122.55 ↑ |
| **INT8** | 43.21↓ | 74.27↓ | 46.06↓ | 75.01↓ | **674.39**↑ | **127.06**↑ |

Moreover, both of the two indicators are even much higher than the original Polarmask (27.43 mAP and 57.01 $AP_{50}$). We summarized the results as the following three reasons. First, we simplified the multi-task head by removing the centerness and 2D BBox regression branches, this greatly reduces the difficulty of fitting the model to complex tasks. The Polarmask also contains a complicated polar centerness regression module we do not considered. Second, we optimized the FPN based on the statistical results (as shown in Fig. 5) of the input image and target sizes of the KITTI MOTS dataset. Considering the on-road targets are commonly smaller and the input image of KITTI is larger, the FPN only consists four different scales of feature maps (1/4 $\sim$ 1/32) as shown in Fig. 1. Third, we use a robust centroid generation strategy by selecting [4, 8, 16, 32] pixels around the true centroid point for [1/4, 1/8, 1/16, 1/32] feature maps. This operation helps mask prediction rejects a lot FP candidates.

### C. Tracking result

Table II compares two different version of Polarmask-Tracker and other two MOTS model. The Polarmask+Center (refer to the tracking strategy of CenterTrack [67]) uses the refined centroid based on Polarmask as tracking reference and greedy algorithm; and the Polarmask+Embed is the final version of Polarmask-Tracker presented in Section III. Except the previous proposed MOTS measurement from [19], we further introduced the Higher Order Tracking Accuracy (HOTA) that can make a more comprehensive assessment of the tracking task. HOTA (defined as Eq. 4) also takes the localization accuracy of tracking results into consideration, the only shortage is that it relies on BBox instead of mask.

From Table II, when replacing the traditional mask prediction head the Polarmask-Tracker achieves much fast FPS and better segmentation, but almost all tracking indicators are lower than the comparison models. However, as we illustrate HOTA, DetA and AssA (considering the limited space, only three of the indicators are shown) under all threshold $\alpha$ (as shown in Fig. 6), we found that when $\alpha \leq 0.5$, the Polarmask-Tracker achieves better performances on all indicators. We found that when the target is at the edge of the image, it tends to predict a circular mask smaller than Ground Truth (as shown in Fig. 7).

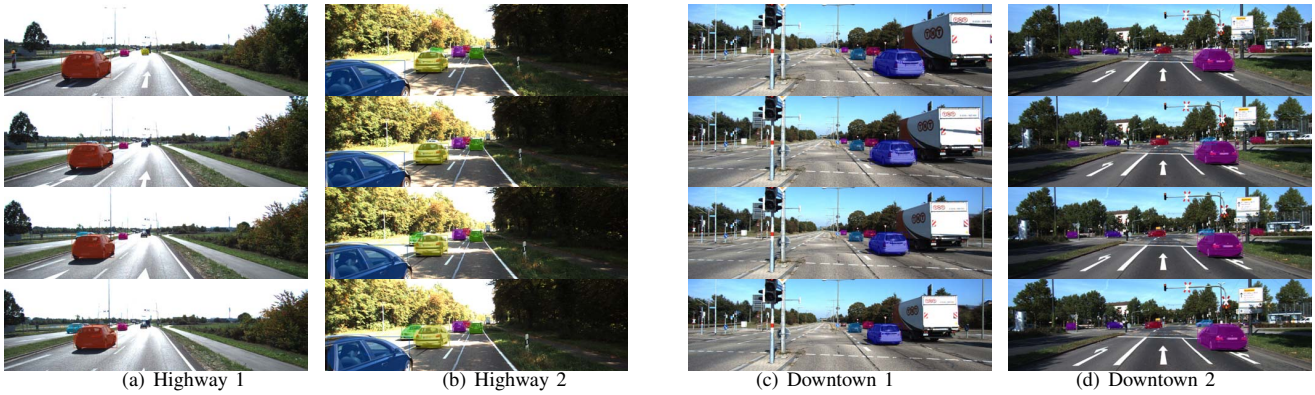
Fig. 8 illustrates four randomly selected tracking subsequences, two from highway and two from downtown, from our model. Visualized results show that our model can accurately segment the vehicles, and some of the vehicles can be tracked again after they are almost completely occluded, i.e., the vehicles moving in the opposite direction with cyan mask in Fig. 8(a) and green mask in Fig. 8(b). Only one failed tracking target with purple mask in the fourth frame of Fig. 8(c) for too long distance.

### D. Ablation studies

We also conducted two simple ablation studies based on segmentation benchmarks because the index is more intuitive.

Table III compares three different segmentation head with Resnet101 backbone, i.e., Yolact, Polarmask (adapted to KITTI input) and Polarmask-refine (removed centreness and BBox branch), and the Polarmask-refine achieves the best segmentation results.

Table IV compares three different TensorRT acceleration patterns for the final Polarmask-Tracker, i.e., Float32 (FP32), Float16 (FP16) and Integer8 (INT8). FP32 is the original version. When using FP16 the model FPS increased from 93.81 to 630.91 with minor drop; but when using INT8, the increment is not significant compare to FP16 (+43 FPS). When



(a) Highway 1 (b) Highway 2 (c) Downtown 1 (d) Downtown 2

Fig. 8. Visualization of the tracking results (two in highway and two in downtown) on KITTI MOTS.

deployed the Polarmask-Tracker with TensorRT on AGX, the origional FPS increased from 20.42 FPS to 122.55 FPS for FP16 and 127.06 FPS for INT8 (Because the model follows an end-to-end neural network design, traditional time complexity estimation is no longer applicable). Since there is a slight drop for the mask prediction (BBox is generated based on mask) when using INT8 mode, we prefor to use FP16 without loss of accuracy.

## V. Conclusion

To achieve end-to-end real-time and lightweight MOTS model we extend the Polarmask based segmentation model with tracking vector embedding. We further improved the segmentation and tracking head by removing the BBox and centerness branches, and introduce the Tripletloss to refine tracking vector generation. With TensorRT acceleration our final model can achieve 630 FPS on 2080ti server and 122.55 FPS on AGX. However, the Polarmask representation is difficult to predict complete contour information when the target is incomplete (i.e., occluded and at the edge of the image, etc.), affects the subsequent tracking results, especially when using large IOU threshold. We plan to further improve the mask prediction in the future.

## Acknowledgment

This work has been supported by China Postdoctoral Science Foundation (2020M681798), Qianjiang Excellent Post-Doctoral Program (2020Y4A001), 2020 Zhejiang Postdoctoral Research Project (ZJ2020011), and Chongqing Autonomous Unmanned System Development Foundation and Key Technology Strategic Research Project (2020-XZ-CQ-3).

## References

[1] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 3354–3361.

[2] Z. Zhao, K. Wang, N. Ling, and G. Xing, "Edgeml: An automl framework for real-time deep learning on the edge," in *Proceedings of the International Conference on Internet-of-Things Design and Implementation*, 2021, pp. 133–144.

[3] J. Niu, Y. Gao, M. Qiu, and Z. Ming, "Selecting proper wireless network interfaces for user experience enhancement with guaranteed probability," *Journal of Parallel and Distributed Computing*, vol. 72, no. 12, pp. 1565–1575, 2012.

[4] J. Niu, C. Liu, Y. Gao, and M. Qiu, "Energy efficient task assignment with guaranteed probability satisfying timing constraints for embedded systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 8, pp. 2043–2052, 2013.

[5] K. Gai and M. Qiu, "Reinforcement learning-based content-centric services in mobile sensing," *IEEE Network*, vol. 32, no. 4, pp. 34–39, 2018.

[6] ——, "Reinforcement learning-based content-centric services in mobile sensing," *IEEE Network*, vol. 32, no. 4, pp. 34–39, 2018.

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[8] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Euro. conf. on comp. vision*, 2014, pp. 740–755.

[9] 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.

[10] P. Dendorfer, A. Osep, A. Milan, K. Schindler, D. Cremers, I. Reid, S. Roth, and L. Leal-Taixé, "Motchallenge: A benchmark for single-camera multiple target tracking," *International Journal of Computer Vision*, vol. 129, no. 4, pp. 845–881, 2021.

[11] S. Manen, M. Gygli, D. Dai, and L. Van Gool, "Pathtrack: Fast trajectory annotation with path supervision," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 290–299.

[12] L. Wen, D. Du, Z. Cai, Z. Lei, M.-C. Chang, H. Qi, J. Lim, M.-H. Yang, and S. Lyu, "Ua-detrac: A new benchmark and protocol for multi-object detection and tracking," *Computer Vision and Image Understanding*, vol. 193, p. 102907, 2020.

[13] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, "Bdd100k: A diverse driving dataset for heterogeneous multitask learning," in *IEEE/CVF conf. on computer vision and pattern recognition*, 2020, pp. 2636–2645.

[14] J. Son, M. Baek, M. Cho, and B. Han, "Multi-object tracking with quadruplet convolutional neural networks," in *IEEE conf. on computer vision and pattern recognition*, 2017, pp. 5620–5629.

[15] E. Baser, V. Balasubramanian, P. Bhattacharyya, and K. Czarnecki, "Fantrack: 3d multi-object tracking with feature association network," in *IEEE Intell. Vehicles Sym. (IV)*, 2019, pp. 1426–1433.

[16] Z. Ouyang, J. Niu, Y. Liu, and M. Guizani, "Deep cnn-based real-time traffic light detector for self-driving vehicles," *IEEE transactions on Mobile Computing*, vol. 19, no. 2, pp. 300–313, 2019.

[17] L. Zhang, Y. Li, and R. Nevatia, "Global data association for multi-object tracking using network flows," in *IEEE Conf. on Computer Vision and Pattern Recognition*. IEEE, 2008, pp. 1–8.

[18] Z. Zhao, Z. Jiang, N. Ling, X. Shuai, and G. Xing, "Ecrt: an edge computing system for real-time image-based object tracking," in *ACM Conf. on Embedded Networked Sensor Systems*, 2018, pp. 394–395.

[19] P. Voigtlaender, M. Krause, A. Osep, J. Luiten, B. B. G. Sekar, A. Geiger, and B. Leibe, "Mots: Multi-object tracking and segmentation," in *IEEE/CVF Conf. CVPR*, 2019, pp. 7942–7951.

[20] X. Dong, J. Niu, J. Cui, Z. Fu, and Z. Ouyang, "Fast segmentation-based object tracking model for autonomous vehicles," in *International Conference on Algorithms and Architectures for Parallel Processing*. Springer, 2020, pp. 259–273.

[21] L. Porzi, M. Hofinger, I. Ruiz, J. Serrat, S. R. Bulo, and P. Kontschieder, "Learning multi-object tracking and segmentation from automatic annotations," in *IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, 2020, pp. 6846–6855.

[22] R. Yao, G. Lin, S. Xia, J. Zhao, and Y. Zhou, "Video object segmentation and tracking: A survey," *ACM Trans. on Intelligent Systems and Technology (TIST)*, vol. 11, no. 4, pp. 1–47, 2020.

[23] S. Minaee, Y. Y. Boykov, F. Porikli, A. J. Plaza, N. Kehtarnavaz, and D. Terzopoulos, "Image segmentation using deep learning: A survey," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2021.

[24] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *IEEE int'l conf. on computer vision*, 2017, pp. 2961–2969.

[25] G. Sharma, D. Liu, S. Maji, E. Kalogerakis, S. Chaudhuri, and R. Měch, "Parsenet: A parametric surface fitting network for 3d point clouds," in *European Conf. on Computer Vision*. Springer, 2020, pp. 261–276.

[26] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, "Yolact++: Better real-time instance segmentation," *arXiv preprint arXiv:1912.06218*, 2019.

[27] R. R. Jha, G. Jaswal, D. Gupta, S. Saini, and A. Nigam, "Pixisegnet: pixel-level iris segmentation network using convolutional encoder–decoder with stacked hourglass bottleneck," *IET Biometrics*, vol. 9, no. 1, pp. 11–24, 2020.

[28] Y. Yuan, X. Chen, and J. Wang, "Object-contextual representations for semantic segmentation," *arXiv preprint arXiv:1909.11065*, 2019.

[29] C. Peng, X. Zhang, G. Yu, G. Luo, and J. Sun, "Large kernel matters–improve semantic segmentation by global convolutional network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4353–4361.

[30] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, "Yolact: Real-time instance segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 9157–9166.

[31] S. Hao, Y. Zhou, and Y. Guo, "A brief survey on semantic segmentation with deep learning," *Neurocomputing*, vol. 406, pp. 302–321, 2020.

[32] H. Liu, R. A. R. Soto, F. Xiao, and Y. J. Lee, "Yolactedge: Real-time instance segmentation on the edge (jetson agx xavier: 30 fps, rtx 2080 ti: 170 fps)," *arXiv preprint arXiv:2012.12259*, 2020.

[33] W. Chen, X. Gong, X. Liu, Q. Zhang, Y. Li, and Z. Wang, "Fasterseg: Searching for faster real-time semantic segmentation," *arXiv preprint arXiv:1912.10917*, 2019.

[34] A. Milioto and C. Stachniss, "Bonnet: An open-source training and deployment framework for semantic segmentation in robotics using cnns," in *IEEE Int'l Conf. ICRA*, 2019, pp. 7094–7100.

[35] A. Holliday, M. Barekatain, J. Laurmaa, C. Kandaswamy, and H. Prendinger, "Speedup of deep learning ensembles for semantic segmentation using a model compression technique," *Computer Vision and Image Understanding*, vol. 164, pp. 16–26, 2017.

[36] Y. Liu, K. Chen, C. Liu, Z. Qin, Z. Luo, and J. Wang, "Structured knowledge distillation for semantic segmentation," in *IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, 2019, pp. 2604–2613.

[37] M. Siam, C. Jiang, S. Lu, L. Petrich, M. Gamal, M. Elhoseiny, and M. Jagersand, "Video object segmentation using teacher-student adaptation in a human robot interaction (hri) setting," in *Int'l Conf. ICRA*, 2019, pp. 50–56.

[38] G. Prabhakar, B. Kailath, S. Natarajan, and R. Kumar, "Obstacle detection and classification using deep learning for tracking in high-speed autonomous driving," in *IEEE region 10 symposium (TENSYMP)*, 2017, pp. 1–6.

[39] S. Dixit, S. Fallah, U. Montanaro, M. Dianati, A. Stevens, F. Mccullough, and A. Mouzakitis, "Trajectory planning and tracking for autonomous overtaking: State-of-the-art and future prospects," *Annual Reviews in Control*, vol. 45, pp. 76–86, 2018.

[40] E. Xie, P. Sun, X. Song, W. Wang, X. Liu, D. Liang, C. Shen, and P. Luo, "Polarmask: Single shot instance segmentation with polar representation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 12 193–12 202.

[41] P. Yang, L. Kong, M. Qiu, X. Liu, and G. Chen, "Compressed imaging reconstruction with sparse random projection," *ACM Transactions on Multimedia Computing, Communications, and Applications*, 2021.

[42] M. Qiu, W. Dai, and A. Vasilakos, "Loop parallelism maximization for multimedia data processing in mobile vehicular clouds," *IEEE Transactions on Cloud Computing*, vol. 7, no. 1, pp. 250–258, 2016.

[43] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[44] Z. Ouyang, J. Cui, X. Dong, Y. Li, and J. Niu, "Saccadefork: A lightweight multi-sensor fusion-based target detector," *Information Fusion*, vol. 1, p. 1, 2021.

[45] F. Milletari, N. Navab, and S.-A. Ahmadi, "V-net: Fully convolutional neural networks for volumetric medical image segmentation," in *IEEE int'l conf. on 3D Vision (3DV)*, 2016, pp. 565–571.

[46] M. Mostajabi, P. Yadollahpour, and G. Shakhnarovich, "Feedforward semantic segmentation with zoom-out features," in *IEEE conf. on computer vision and pattern recog.*, 2015, pp. 3376–3385.

[47] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang, "Unet++: A nested u-net architecture for medical image segmentation," in *Deep learning in medical image analysis and multimodal learning for clinical decision support*. Springer, 2018, pp. 3–11.

[48] Y. Xiong, R. Liao, H. Zhao, R. Hu, M. Bai, E. Yumer, and R. Urtasun, "Upsnet: A unified panoptic segmentation network," in *IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, 2019, pp. 8818–8826.

[49] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 801–818.

[50] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, Z. Fang, and H. Lu, "Dual attention network for scene segmentation," in *IEEE/CVF Conf. on Computer Vision and Pattern Recog.*, 2019, pp. 3146–3154.

[51] X. Zhu, H. Hu, S. Lin, and J. Dai, "Deformable convnets v2: More deformable, better results," in *IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, 2019, pp. 9308–9316.

[52] J. Deng, L. Wang, S. Pu, and C. Zhuo, "Spatio-temporal deformable convolution for compressed video quality enhancement," in *AAAI Conf. on Artificial Intell.*, vol. 34, no. 07, 2020, pp. 10 696–10 703.

[53] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," *arXiv preprint arXiv:1412.7062*, 2014.

[54] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr, "Conditional random fields as recurrent neural networks," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1529–1537.

[55] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE trans. on pattern analysis and machine intell.*, vol. 40, no. 4, pp. 834–848, 2017.

[56] Z. Liu, X. Li, P. Luo, C.-C. Loy, and X. Tang, "Semantic image segmentation via deep parsing network," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1377–1385.

[57] C. Wang, M. de La Gorce, and N. Paragios, "Segmentation, ordering and multi-object tracking using graphical models," in *IEEE 12th Int'l Conf. on Computer Vision*, 2009, pp. 747–754.

[58] Y. Wang, Z. Xu, H. Shen, B. Cheng, and L. Yang, "Centermask: single shot instance segmentation with point representation," in *IEEE/CVF Conf. on Computer Vision and Pattern Recog.*, 2020, pp. 9313–9321.

[59] Z. Huang, L. Huang, Y. Gong, C. Huang, and X. Wang, "Mask scoring r-cnn," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 6409–6418.

[60] Z. Tian, C. Shen, H. Chen, and T. He, "Fcos: Fully convolutional one-stage object detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 9627–9636.

[61] Y. Lee and J. Park, "Centermask: Real-time anchor-free instance segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 13 906–13 915.

[62] Z. Xu, W. Zhang, X. Tan, W. Yang, H. Huang, S. Wen, E. Ding, and L. Huang, "Segment as points for efficient online multi-object tracking and segmentation," in *European Conference on Computer Vision*. Springer, 2020, pp. 264–281.

[63] J. V. Hurtado, R. Mohan, W. Burgard, and A. Valada, "Mopt: Multi-object panoptic tracking," *arXiv preprint arXiv:2004.08189*, 2020.

[64] Y.-m. Song and M. Jeon, "Online multi-object tracking and segmentation with gmphd filter and simple affinity fusion," *arXiv preprint arXiv:2009.00100*, 2020.

[65] A. Hermans, L. Beyer, and B. Leibe, "In defense of the triplet loss for person re-identification," *arXiv preprint arXiv:1703.07737*, 2017.

[66] M. Everingham and J. Winn, "The pascal visual object classes challenge 2012 (voc2012) development kit," *Pattern Analysis, Statistical Modelling and Computational Learning, Tech. Rep*, vol. 8, 2011.

[67] X. Zhou, V. Koltun, and P. Krähenbühl, "Tracking objects as points," in *Euro. Conf. on Comp. Vision*. Springer, 2020, pp. 474–490.