SearchTrack: Multiple Object Tracking with Object-Customized Search and Motion-Aware Features

Zhong-Min Tsai[1*]
vtsai01@cmlab.csie.ntu.edu.tw
Yu-Ju Tsai[1*]
r06922009@cmlab.csie.ntu.edu.tw
Chien-Yao Wang[2]
kinyiu@iis.sinica.edu.tw
Hong-Yuan Liao[2]
liao@iis.sinica.edu.tw
Youn-Long Lin[3]
ylin@cs.nthu.edu.tw
Yung-Yu Chuang[1]
cyy@csie.ntu.edu.tw

[1] National Taiwan University
Taipei, Taiwan
[2] Institute of Information Science, Academia Sinica
Taipei, Taiwan
[3] National Tsing Hua University
Hsinchu, Taiwan

Abstract

The paper presents a new method, SearchTrack, for multiple object tracking and segmentation (MOTS). To address the association problem between detected objects, SearchTrack proposes object-customized search and motion-aware features. By maintaining a Kalman filter for each object, we encode the predicted motion into the motion-aware feature, which includes both motion and appearance cues. For each object, a customized fully convolutional search engine is created by SearchTrack by learning a set of weights for dynamic convolutions specific to the object. Experiments demonstrate that our SearchTrack method outperforms competitive methods on both MOTS and MOT tasks, particularly in terms of association accuracy. Our method achieves 71.5 HOTA (car) and 57.6 HOTA (pedestrian) on the KITTI MOTS and 53.4 HOTA on MOT17. In terms of association accuracy, our method achieves state-of-the-art performance among 2D online methods on the KITTI MOTS. Our code is available at https://github.com/qa276390/SearchTrack.

1 Introduction

Recently, deep learning has contributed significantly to advances in various core computer vision tasks, including object detection and image segmentation. Nevertheless, there are still a number of important tasks that remain challenging. The tracking of multiple objects is one

*These authors contributed equally to this work.

of them, as noted by Voigtlaender et al. [18]. Multiple-object tracking (MOT) and multiple-object tracking and segmentation (MOTS) have received increased attention in recent years. MOT requires the tracking of objects using bounding boxes, while MOTS requires pixel-level accuracy. In the MOTS problem, detection, segmentation, and tracking have to be considered simultaneously [18].

A main challenge of the MOTS task is the association of instances of the same object at different times. A number of factors make it challenging, including object deformations, changes in view, differences in illumination, occlusions, and ambiguities, among others. Object appearance and object motion are popular cues for resolving the association between detected object instances. The appearance cue is the most popular, and there have been many attempts to derive unique and invariant representations for objects from pixel values. TrackR-CNN [18] and PointTrack [23] are notable examples. TrackR-CNN, based on Mask R-CNN [5], uses region-of-interest (ROI) for cropping the feature map to generate re-identification (re-ID) feature encoding appearance cues for each instance. PointTrack converts the input frame into a 2D point cloud representation and separates the point cloud into foreground and background for learning instance embeddings. Object motion is another popular cue for tracking, providing the object's location over time. As the tracked object in MOT(S) is highly correlated across consecutive frames, tracing its trajectory is helpful for tracking. SORT [3] is a motion-based MOT tracker, which uses a Kalman filter to predict the object's motion and finds matches between objects using the Hungarian algorithm.

In this paper, we propose a new MOT(S) architecture, SearchTrack, which integrates both object appearance and motion cues to resolve the association problem. In order to better utilize the motion cue, the motion predicted by the Kalman filter is encoded together with the appearance feature in order to produce a motion-aware feature. In order to identify the association for a given object, we propose an object-customized search. Inspired by CondConv [24] and CondInst [16], our model learns a set of convolution kernels for searching the given object in the current frame. The resultant convolution kernels are applied to the motion-aware feature to generate a probability map indicating the spatial likelihood of locating the given object in the current frame. To achieve good search results, dynamic convolution kernels are expected to encode the query object's characteristics, such as its appearance, relative position, and shape. Despite that similar ideas have been explored in different contexts, our main contribution is the overall framework for integrating them into the context of MOTS. Experiments show that the proposed method outperforms competitive methods on popular MOT and MOTS benchmarks, particularly in terms of association accuracy.

2 Related Work

Online trackers generally focus on either motion or appearance cues, with a greater focus on the latter. Some attempts combine both cues, but they tend to do so in a simplistic manner. Methods are divided into three categories based on how they utilize cues.

Motion Cues. SORT [3] uses the Kalman filter [7] to predict object motion and estimate the future location of the tracklets. It then associates newly detected tracklets with the highest IOU overlap using the Hungarian algorithm. Since SORT only utilizes motion cues, it can achieve 260 fps inference speed and is effective in some scenarios. Given good detection results and the high frame-rate input, IOU-Tracker [4] eliminates the motion prediction algorithm and uses only the location of objects to compute overlap between the tracklets for tracking. In more challenging situations, such as crowded scenes, these methods may fail

due to the lack of information about the appearance of the objects. To address the problem, Deep SORT [20, 21] considers appearance features from a deep neural network and associates objects using the object location overlap between frames.

Appearance Cues. Most trackers [18, 19, 26] extract the re-ID features from the regions proposed by a detector. FairMOT [27] shows the advantage of the center-based re-ID methods. The re-ID feature similarity is computed between tracklets and detections to assign the identities. CenterTrack [29] proposes a simultaneous detection and tracking algorithm and links objects implicitly in adjacent frames using point-based object representations. SiamMOT [13] formulates the Region Proposal Network and Siamese-based MOT trackers and deals with the similarity of object appearance. These methods can handle challenging cases such as identity losing and re-appearing. CCPNet [22] presents a data augmentation strategy, continuous copy-paste, to deal with the limited number of instances in consecutive raw frames. However, its training uses two external datasets, and others cannot train their models using the augmentation strategy unless the pre-processed datasets become available.

Combination of both cues. In addition to appearance, FairMOT also takes into account motion cues at the time of inference. After the model has been processed, it performs the location prediction using motion cues. In a similar way to Deep SORT, FairMOT adds the similarity of two parts with a handcrafted ratio as the new similarity. There are two drawbacks to these methods. The first problem is that the combination ratio is fixed regardless of the size, category, and density of objects in the scene. In addition, they consider the appearance of an object and its motion as two separate sources of information.

Our approach integrates both appearance and motion cues in a more unified and learnable manner. This results in more robust association results for different scenes, object classes, and sizes, as they are adapted simultaneously. In our experiments, we found that the importance of motion differed when tracking cars and pedestrians. As a reasonable explanation, a car's motion is considered to be rigid body motion, whereas pedestrians' motion is considered to be non-rigid body motion. Unlike the former, which is an overall body movement, the latter is an articulated movement that is difficult to master.

3 SearchTrack

Our MOTS method, SearchTrack, is point-based and compatible with most point-based detection methods. Our current implementation is built on the CenterNet detector [28], which identifies objects by their center points. For segmentation, we adapt CondInst [16]. On top of the point-based detector, we propose an object-customized search method to address the object association problem. For better utilizing motion cues, the Kalman filter predicts object position, which supplements the motion-aware feature.

Fig. 1 provides an overview of SearchTrack. Our method takes two images as input at a time, and these two images need not be adjacent to each other in our training process. At time t, an image of the current frame I^t and a previous frame $I^{t-\delta}$ are given. Since the previous frame $I^{t-\delta}$ has already been processed, it contains instances with the information of tracked identities, $T^{t-\delta} = \{b_1^{t-\delta}, ..., b_i^{t-\delta}, ...\}$. Each detected object $b = (\mathbf{p}, \mathbf{m}, w, \theta, id)$ has attributes including the center location $\mathbf{p}$, the segmentation mask $\mathbf{m}$, the detection confidence w, the customized weights θ, and the unique identity id.

In the first step, both images I^t and $I^{t-\delta}$ go through the same backbone network to obtain feature maps f^t and $f^{t-\delta}$. The feature map f^t of the current frame is then fed to the detection and segmentation branch to produce a list of newly detected instances $\tilde{T}^t$ (including bounding

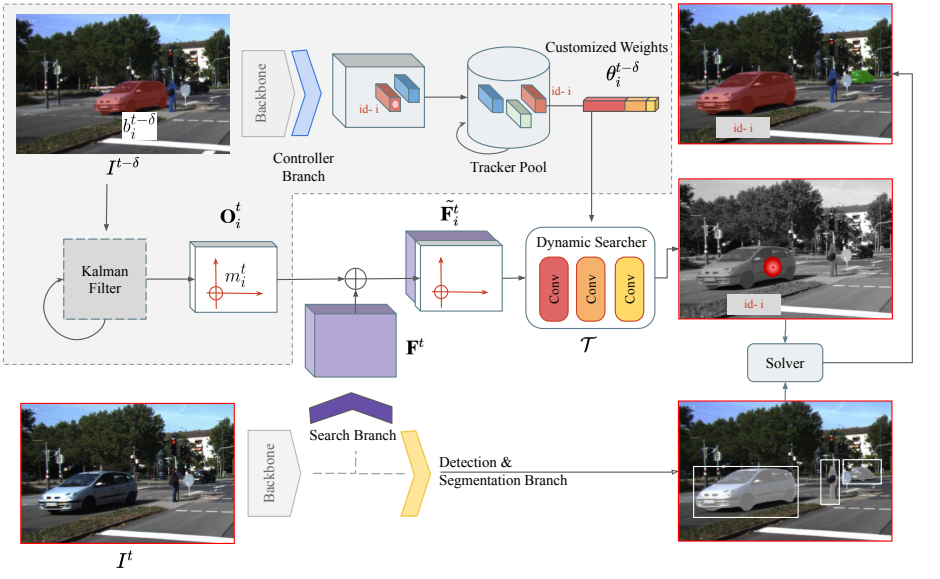

Figure 1: **Overview of SearchTrack.** A shared backbone network extracts features from the current frame I^t and the previous frame $I^{t-\delta}$. The extracted feature of I^t is fed into the detection and segmentation branch for obtaining a set of candidate objects. It is also fed to the search branch to form a search feature map $\mathbf{F}^t$. For a query object $b_i^{t-\delta}$ in the previous frame, we obtain its customized weight $\theta_i^{t-\delta}$ and predicted location m_i^t from its Kalman filter. The location m_i^t is encoded in a motion map $\mathbf{O}_i^t$. Combining $\mathbf{O}_i^t$ and $\mathbf{F}^t$ gives us the motion-aware motion map $\tilde{\mathbf{F}}_i^t$. The customized weight $\theta_i^{t-\delta}$ realized a customized searcher $\mathcal{T}$ for the object $b_i^{t-\delta}$. By taking $\tilde{\mathbf{F}}_i^t$ as input, $\mathcal{T}$ outputs a response map indicating where the object could locate. Finally, a solver matches the boxes from the search branch and the candidate objects from the detection and segmentation branch for resolving association among objects.

boxes and masks) without assigning identities. The core problem is to resolve the association problem between $T^{t-\delta}$ and $\tilde{T}^t$. For resolving this problem, the feature map f^t is also fed to the search branch for extracting a compact feature map $\mathbf{F}^t$ for later search.

For each detected object $b_i^{t-\delta}$ in the previous frame, we aim to search its position at the current frame. For better utilizing object motion, a Kalman filter is maintained separately for each detected object. For the object $b_i^{t-\delta}$, its position at the current frame is predicted by the associated Kalman filter. The position is then encoded as a motion map $\mathbf{O}_i^t$. By concatenating the motion map $\mathbf{O}_i^t$ with the feature map $\mathbf{F}^t$, we obtain the motion-aware feature map $\tilde{\mathbf{F}}_i^t$ for the object b_i and feed it into the dynamic searcher.

For the previous frame, its feature map $f^{t-\delta}$ is fed to the controller branch to obtain a feature map, called dynamic weight map $\theta^{t-\delta}$. Each pixel of the map contains a vector encoding a set of convolution weights customized for an object locating at that location, if there is any. For the detected object b_i, we obtain its customized weights $\theta_i^{t-\delta}$ and feed it to the dynamic searcher.

The dynamic searcher takes $\theta_i^{t-\delta}$ and uses it to form a set of convolution kernels as the search kernels for the object b_i. With these convolution kernels, the dynamic searcher $\mathcal{T}$ becomes a customized CNN for searching the object b_i. By passing the motion-aware feature

$\tilde{\mathbf{F}}^t$ through $\mathcal{T}$, we obtain a response map, in which each pixel value indicates the probability that object b_i appears at that pixel for the current frame. With the response map, we can find the detected instance in $\tilde{T}^t$, which most likely corresponds to the object b_i and thus resolve the association problem.

3.1 Detection and Segmentation Branch

Our method is point-based. The detection results of region-based detectors may cause ambiguity during training since a single rectangular region could correspond to multiple identities in a crowded scene. This shortcoming of region-based detectors was also noted by FairMOT [27]. We have therefore decided to adopt a point-based approach and build our MOTS architecture on top of a point-based detector, CenterNet [28] in the current implementation. As the segmentation branch, we adapt dynamic convolution from CondInst [16].

The detection and segmentation branch takes as input a single image $I \in \mathbb{R}^{W_I \times H_I \times 3}$ and generates a detection representation set $\{(\mathbf{p_j}, \mathbf{s_j})\}_j^{N-1}$ for each object class $c \in \{0, ..., C-1\}$. The detection consists of two attributes: the center point $\mathbf{p} \in \mathbb{R}^2$ signifies the location of each object, and the size $\mathbf{s} \in \mathbb{R}^2$ gives the height and width of the object's bounding box from regression. Furthermore, the model also generates two low-resolution maps: the heat map $Y \in [0,1]^{W \times H \times C}$ and the size map $S \in [0,1]^{W \times H \times 2}$, where $W = W_I/R$ and $H = H_I/R$ with the downsampling ratio $R = 4$ in our implementation. Each center of the detected object will be represented as local maximum $\mathbf{p} \in \mathbb{R}^2$ in the heat map Y along with the detection confidence $\omega = Y_\mathbf{p}$ and the object size $\mathbf{s} = S_\mathbf{p}$, where $Y_\mathbf{p}$ and $S_\mathbf{p}$ are the values of Y and S at the point p. For segmentation, a segmentation mask is associated with each detected object.

3.2 Motion-aware Feature Map

As mentioned, a compact search feature map $\mathbf{F}^t \in \mathbb{R}^{W \times H \times C_{search}}$ is generated by the search branch connected to the backbone branch, which takes I^t as input. We empirically sets the number of channels C_{search} to 16 as it achieves a good balance between computation cost and performance. To leverage object motion for tracking, we choose to use the Kalman filter framework [3] as the motion model. We approximate the displacements between frames of each object with a linear constant velocity model, independent of the motions of other objects and the camera. For each object, a Kalman filter is maintained for predicting its locations in the following frames. By using the Kalman filter to predict the center location m_i^t of the object $b_i^{t-\delta}$ at time t, we can produce a motion map $\mathbf{O}_i^t \in \mathbb{R}^{W \times H \times 2}$ that contains offset vectors from the position m_i^t, $\mathbf{O}_i^t(\mathbf{p}) = \mathbf{p} - m_i^t$. By concatenating $\mathbf{O}_i^t$ and $\mathbf{F}^t$, we obtain the motion-aware feature map $\tilde{\mathbf{F}}_i^t \in \mathbb{R}^{W \times H \times (C_{search}+2)}$ for the object b_i. The motion map provides a strong cue for predicting the object motion for the association and significantly improves the performance, particularly for non-rigid bodies.

3.3 Dynamic Searcher

A core problem of multiple object tracking is to resolve association. SearchTrack addresses this problem by performing object-customized searches through the dynamic search engine. Given a detected instance $b_i^{t-\delta}$ at time $t-\delta$, its customized weights $\theta_i^{t-\delta}$ is adopted as convolution weights to search for the particular instance in frame I^t globally. Formally,

$$\mathbf{R}_i^t = \mathcal{T}(\tilde{\mathbf{F}}_i^t; \theta_i^{t-\delta}), \tag{1}$$

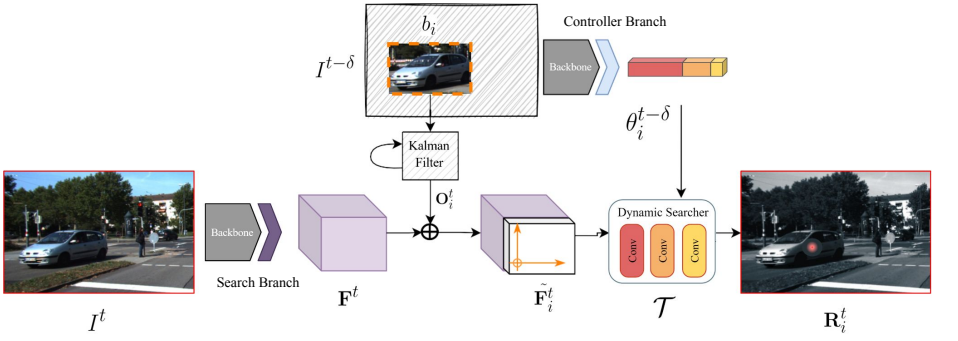

Figure 2: **The search process for finding object association**. Given the input image I^t and a detected object b_i at time $t-\delta$, we obtain the feature map $\mathbf{F}^t$ from the search branch and dynamic weight $\theta_i^{t-\delta}$ from the controller branch. Note that both branches share the same backbone for their first parts. Next, we generate the motion map $\mathbf{O}_i^t$ according to the Kalman filter's prediction for the object. We then integrate $\mathbf{O}_i^t$ and $\mathbf{F}^t$ to have the motion-aware feature map $\tilde{\mathbf{F}}_i^t$ for the object. The dynamic weight $\theta_i^{t-\delta}$ is fed into the dynamic searcher $\mathcal{T}$ to realize a customized searcher for the given object. By taking $\tilde{\mathbf{F}}_i^t$ as input, $\mathcal{T}$ outputs an association response map $\mathbf{R}_i^t$ for the object in the current frame. The peak of the map indicates the center of the object b_i in I^t.

where $\tilde{\mathbf{F}}_i^t$ is the motion-aware feature map customized for the object b_i and $\mathcal{T}$ is the learnable fully convolutional network (FCN) tracker with parameters $\theta_i^{t-\delta}$. These parameters are the dynamic weights generated by the controller head at the center of $b_i^{t-\delta}$. The output of this network is a response map $\mathbf{R}_i^t \in [0,1]^{W\times H}$ that gives the association possibility of b_i in I^t. The supplementary document provides more details regarding the model architecture and the number of parameters for the dynamic weight θ.

By finding the peak in the response map, we determine the location of b_i^t and take the peak value as the association confidence v_i^t. Note that the instance b_i^t is not completed until the bounding box size and segmentation branch is applied. In the case that the instance $b_i^{t-\delta}$ is visible in I^t, our tracker $\mathcal{T}$ should give a high confidence score at the location where b_i^t is located. Fig. 2 illustrates the association process in the overall model depicted in Fig. 1.

Fig. 3 gives an example of the response map. Given the objects in the previous frame and the current frame, the searcher generates a response map corresponding to each object. The green intensity value of each pixel indicates the probability that the object appears at that pixel. The cyan points indicate the previous object centers, and the magenta points represent the object center predicted by the Kalman Filter.

The procedure in Eq. (1) is applied multiple times, once for each detected object $b_i^{t-\delta}$. Nevertheless, each frame only needs to pass through the backbone network one time, resulting in a significant reduction in computation time. Without using many parameters, the dynamic searcher shows that a compact FCN with dynamically-generated filters can outperform ROI-based trackers such as TrackR-CNN.

(a) Frame t-1 (b) Frame t (c) Object ID: 4 (d) Object ID: 2

Figure 3: **The response maps from the dynamic searcher.** (a) The tracking and segmentation results from the previous frame at time $t-1$. (b) The current frame at time t. (c) The response map for the object with ID 4 at time t. (d) The response map for the object with ID 2 at time t. There are two detected objects with ID 2 and 4 at frame $t-1$ as shown in (a). The response map of a previously detected object indicates the likelihood that the object appears at each pixel at time t. Taking the object with ID 4 as an example, the green intensity in the response map (c) shows the corresponding likelihood of the object. The peak will be taken as the detected position of the object. For (c) and (d), the cyan point indicates the object center in the previous frame at time $t-1$. The magenta point denotes the predicted object center provided by the Kalman Filter at time t. For clarity, the relevant region is enlarged and shown on the top-right corner.

3.4 Loss Function

The overall loss function of SearchTrack can be formulated as:

$$L_{total} = L_{heatmap} + L_{mask} + L_{search}, \tag{2}$$

where L_{mask} is the loss of the CondInst segmentation branch; $L_{heatmap}$ is the focal loss [8, 9] defined in the CenterNet,

$$L_{heatmap} = \frac{1}{N}\sum_{xyc}\begin{cases}(1-Y_{xyc})^{\alpha}log(Y_{xyc}) & \text{if } Y^*_{xyc}=1\\ (1-Y^*_{xyc})^{\beta}(Y_{xyc})^{\alpha}log(1-Y_{xyc}) & \text{otherwise,}\end{cases} \tag{3}$$

where $Y \in [0,1]^{W\times H\times C}$ is the output heatmap and $Y^* \in [0,1]^{W\times H\times C}$ is the ground-truth heatmap corresponding to the objects. N is the number of objects, and $\alpha = 2$ and $\beta = 4$ are hyperparameters of the focal loss.

We define the loss L_{search} as

$$L_{search} = \sum_i L_{focal}(\mathcal{T}(\tilde{\mathbf{F}}^t_i;\theta^{t-\delta}_i), R^{*t}_i) \tag{4}$$

and L_{focal} is a reduced version of Eq. (3) defined as:

$$L_{focal}(R,R^*) = \sum_{xy}\begin{cases}(1-R_{xy})^{\alpha}log(R_{xy}) & \text{if } R^*_{xy}=1\\ (1-R^*_{xy})^{\beta}(R_{xy})^{\alpha}log(1-R_{xy}) & \text{otherwise,}\end{cases} \tag{5}$$

where $R^* \in [0,1]^{W\times H}$ is the ground truth response map corresponding to the objects. The ground truth map is formed by rendering a Gaussian at the true center position. $\alpha = 2$ and $\beta = 4$ are hyperparmeters of the focal loss.

		Pedestrian					Car				
	time	HOTA↑	DetA↑	AssA↑	LocA↑	sMOTSA↑	HOTA↑	DetA↑	AssA↑	LocA↑	sMOTSA↑
TrackRCNN [23]	0.5	41.9	53.8	33.8	78.0	47.3	56.6	69.9	46.5	86.6	67.0
GMPHD_SAF [23]	0.08	49.3	65.5	38.3	83.8	62.9	55.1	77.0	39.8	88.7	75.4
PointTrack [23]	0.05	54.4	62.3	48.1	83.3	61.5	62.0	79.4	48.8	88.5	78.5
ReMOTS* [23]	3	58.8	68.0	52.4	84.2	66.0	71.6	78.3	66.0	89.3	75.9
Ours	0.19	57.6	63.7	53.1	80.9	60.6	71.5	76.8	67.1	88.0	74.9

Table 1: **Comparison with leading 2D methods on the KITTI MOTS leaderboard.** "*" denotes **offline** methods. The "time" column reports inference time (in seconds) for a single frame, which was reported by the authors. We compare the published leading **online** 2D methods and one **offline** 2D method on the leaderboard. Our method achieves the highest HOTA score among all **online** 2D methods. Also, our association accuracy (AssA) outperforms all the compared methods. We highlight the best and the second best in each column.

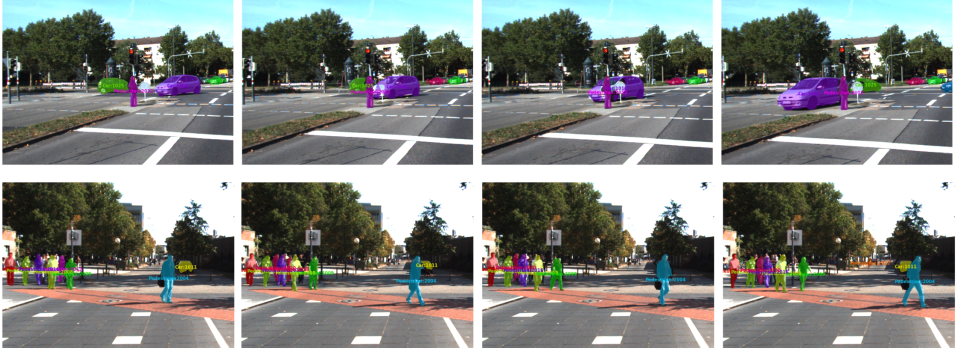

Figure 4: **Results of SearchTrack on KITTI MOTS**. Tracking ids are coded in colors.

4 Experiments and Results

We evaluate the proposed tracker on both the MOT and MOTS tasks using a popular MOTS benchmark, KITTI MOTS [23], and a popular MOT benchmark, MOT17 [11]. The primary evaluation metric is HOTA [6, 10], which is the **primary metric** in KITTI MOTS and MOT17 for comparisons and ranking since 2021. Additionally, we consider other metrics when appropriate, including sMOTSA, MOTA, IDF1, DetA, AssA, and LocA. The supplementary document provides more details regarding the datasets and metrics.

4.1 Results

KITTI MOTS. Table 1 compares SearchTrack with some competitive methods on the KITTI MOTS test set. In order to make a fair comparison, we only select the leading **published** 2D methods since some of the leading methods on the leaderboard are 3D and utilize additional information. The "time" column of Table 1 reports the average inference time (in seconds) for each method on a single frame. As far as inference time is concerned, our method is also competitive. Fig. 4 shows some visual results.

Our SearchTrack outperforms all the state-of-the-art online 2D methods in both pedestrian and car categories. Our method outperforms the state-of-the-art PointTrack method [23] significantly: for the pedestrian class, our method provides 3.2% gain on HOTA and 5% gain on AssA; and for the car class, our method provides 9.5% gain on HOTA and 18.3% gain

Method	HOTA↑	DetA↑	AssA↑	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDSW↓
Tracktor++ [4]	44.8	44.9	45.1	56.3	55.1	21.1%	35.3 %	8866	235449	1987
Visual-Spatial [4]	46.4	45.3	47.9	56.8	58.3	22.8%	37.4 %	11567	230645	1320
CenterTrack [29]	48.2	49.0	47.8	61.5	59.6	26.4%	31.9 %	14076	200672	2583
TMOH [25]	50.4	49.6	50.9	62.1	62.8	26.9%	31.4 %	10951	201195	1897
SiamMOT [23]	-	-	-	65.9	63.3	34.6%	23.9%	18098	170955	3040
PermaTrack [17]	54.2	58.0	51.2	73.1	67.2	42.3%	19.1%	24557	123508	3571
Ours	53.4	55.6	51.6	68.0	65.7	39.1%	21.1%	25651	150786	4254

Table 2: **Results on the MOT17 test set with public detection.** We compare leading published **online** methods on the leaderboard. The results show that SearchTrack achieves the best association accuracy (AssA) among all the compared trackers. We highlight the best and the second best in each column.

Figure 5: **Results of SearchTrack on MOT17**. Tracking ids are coded in colors.

on AssA. In addition to the online methods, SearchTrack achieves comparable performance to the offline 2D method ReMOTS [25] (0.7 % gain for pedestrian and 1.1% gain for cars on AssA) with a significant speed advantage (0.19s vs. 3s per frame). The results show that our method has significantly improved object association for tracking with a reasonable inference speed.

MOT17. Table 2 compares our method with the methods on the MOT17 leaderboard with the public detection setting. The MOT17 public detection setting provides the detection results and mainly tests the tracker's ability to associate objects for tracking. We use the public detection configuration with the same setting as CenterTrack and pretrain on the CrowdHuman dataset [12]. For this comparison, although SearchTrack is 0.8 behind PermaTrack [17] on HOTA, it provides the superior association accuracy performance as measured by AssA. Our tracker is competitive with state-of-the-art methods on the MOT task. Fig. 5 shows some tracking results on the MOT17 dataset.

4.2 Ablation Studies

We have conducted ablation studies to justify our main design choices in the proposed architecture, including the motion-aware feature and the segmentation branch. The experiments are mainly performed on KITTI MOTS. The supplementary gives more ablation studies.

	Pedestrian			Car		
	HOTA↑	DetA↑	AssA↑	HOTA↑	DetA↑	AssA↑
feature without motion	43.9 %	**56.4 %**	34.4 %	77.1 %	79.3 %	75.5 %
motion-aware feature	**59.4 %**	56.0 %	**63.5 %**	**78.7 %**	**79.4 %**	**78.5 %**

Table 3: **Ablation study on the motion-aware feature using the KITTI MOTS validation set.** We evaluate the importance of encoding motion into the feature map for search. The results show the motion information is important, especially for the pedestrian class. With the motion-aware feature, the association accuracy (AssA) and HOTA are significantly improved, especially for non-rigid pedestrians.

	Pedestrian				Car			
	HOTA↑	DetA↑	AssA↑	LocA↑	HOTA↑	DetA↑	AssA↑	LocA↑
w/o segmentation	55.9 %	55.5 %	56.4 %	82.2 %	73.9 %	**73.7 %**	74.3 %	87.3 %
w/ segmentation	**58.2 %**	**56.8 %**	**59.8 %**	**83.8 %**	**74.5 %**	73.3 %	**76.0 %**	**87.6 %**

Table 4: **Impact of the segmentation branch on tracking using the KITTI MOTS validation set.** By jointly training with the segmentation task, the tracking performance can be improved. Furthermore, because the backbone is shared for these tasks, the knowledge of foreground and background also benefits the tracking task.

Motion-aware feature. Table 3 compares the feature maps with and without encoding motion information on the KITTI MOTS validation set. The feature map without motion information only takes appearance cues into account. Its association performance is weak, especially for the pedestrian class. It shows that using only appearance cues is often insufficient for a non-rigid body such as a pedestrian. The motion-aware feature map significantly boosts the association accuracy for the pedestrian class.

The segmentation branch. To investigate the synergy between tracking and segmentation, we experiment with training with and without the segmentation branch. For analysis, we evaluate in bounding box level on KITTI MOTS and train from scratch, avoiding the impacts from the pre-trained model. Table 4 shows that tracking performance is significantly improved for most evaluation metrics when augmenting the segmentation branch into architecture. The results validate that tracking can benefit from the dense pixelwise annotations and better separation of foreground and background. Note that we do not enable the segmentation branch in our architecture when performing the MOT task.

5 Conclusion

This paper proposes an online point-based tracker named SearchTrack that simultaneously considers object appearance and motion cues to address the MOTS problem. To resolve the association problem, SearchTrack employs an object-customized search network. Also, by maintaining a Kalman filter for each object, we encode the predicted location into the motion-aware feature map as the input to the customized searcher. SearchTrack outperforms the state-of-the-art online 2D methods on the KITTI MOTS benchmark. Additionally, our method outperforms previous methods in terms of association accuracy on the MOT benchmark. As far as we know, SearchTrack is the first one-stage point-based MOTS framework. Moreover, it is efficient, straightforward, and more accurate than the state-of-the-art methods.

Acknowledgments. This work was supported in part by MOST 110-2634-F-002-051. We thank to National Center for High-performance Computing (NCHC) for providing computational and storage resources.

References

[1] Favyen Bastani, Songtao He, and Samuel Madden. Self-supervised multi-object tracking with cross-input consistency. *Advances in Neural Information Processing Systems*, 2021.

[2] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[3] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, 2016.

[4] Erik Bochinski, Volker Eiselein, and Thomas Sikora. High-speed tracking-by-detection without using image information. In *International Workshop on Traffic and Street Surveillance for Safety and Security at IEEE AVSS 2017*, 2017.

[5] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, 2017.

[6] Arne Hoffhues Jonathon Luiten. Trackeval. https://github.com/JonathonLuiten/TrackEval, 2020.

[7] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 1960.

[8] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European conference on computer vision (ECCV)*, 2018.

[9] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, 2017.

[10] Jonathon Luiten, Aljosa Osep, Patrick Dendorfer, Philip Torr, Andreas Geiger, Laura Leal-Taixé, and Bastian Leibe. Hota: A higher order metric for evaluating multi-object tracking. *International Journal of Computer Vision*, 2020.

[11] Anton Milan, Laura Leal-Taixé, Ian Reid, Stefan Roth, and Konrad Schindler. Mot16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831*, 2016.

[12] Shuai Shao, Zijian Zhao, Boxun Li, Tete Xiao, Gang Yu, Xiangyu Zhang, and Jian Sun. Crowdhuman: A benchmark for detecting human in a crowd. *arXiv preprint arXiv:1805.00123*, 2018.

[13] Bing Shuai, Andrew Berneshawi, Xinyu Li, Davide Modolo, and Joseph Tighe. Siammot: Siamese multi-object tracking. In *CVPR*, 2021.

[14] Young-min Song and Moongu Jeon. Online multi-object tracking and segmentation with gmphd filter and simple affinity fusion. *arXiv preprint arXiv:2009.00100*, 2020.

[15] Daniel Stadler and Jurgen Beyerer. Improving multiple pedestrian tracking by track management and occlusion handling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[16] Zhi Tian, Chunhua Shen, and Hao Chen. Conditional convolutions for instance segmentation. In *European Conference on Computer Vision*, 2020.

[17] Pavel Tokmakov, Jie Li, Wolfram Burgard, and Adrien Gaidon. Learning to track with object permanence. In *ICCV*, 2021.

[18] Paul Voigtlaender, Michael Krause, Aljosa Osep, Jonathon Luiten, Berin Balachandar Gnana Sekar, Andreas Geiger, and Bastian Leibe. MOTS: Multi-object tracking and segmentation. In *CVPR*, 2019.

[19] Zhongdao Wang, Liang Zheng, Yixuan Liu, Yali Li, and Shengjin Wang. Towards real-time multi-object tracking. In *European Conference on Computer Vision*, 2020.

[20] Nicolai Wojke and Alex Bewley. Deep cosine metric learning for person re-identification. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018.

[21] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *2017 IEEE International Conference on Image Processing (ICIP)*, 2017.

[22] Zhenbo Xu, Ajin Meng, Wei Yang, and Liusheng Huang. Continuous copy-paste for one-stage multi-object tracking and segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.

[23] Zhenbo Xu, Wei Zhang, Xiao Tan, Wei Yang, Huan Huang, Shilei Wen, Errui Ding, and Liusheng Huang. Segment as points for efficient online multi-object tracking and segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[24] Brandon Yang, Gabriel Bender, Quoc V Le, and Jiquan Ngiam. Condconv: Conditionally parameterized convolutions for efficient inference. *Advances in Neural Information Processing Systems*, 2019.

[25] Fan Yang, Xin Chang, Chenyu Dang, Ziqiang Zheng, Sakriani Sakti, Satoshi Nakamura, and Yang Wu. Remots: Self-supervised refining multi-object tracking and segmentation. *arXiv preprint arXiv:2007.03200*, 2020.

[26] Fengwei Yu, Wenbo Li, Quanquan Li, Yu Liu, Xiaohua Shi, and Junjie Yan. Poi: Multiple object tracking with high performance detection and appearance feature. In *European Conference on Computer Vision*, 2016.

[27] Yifu Zhang, Chunyu Wang, Xinggang Wang, Wenjun Zeng, and Wenyu Liu. Fairmot: On the fairness of detection and re-identification in multiple object tracking. *International Journal of Computer Vision*, 2021.

[28] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. In *arXiv preprint arXiv:1904.07850*, 2019.

[29] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. *ECCV*, 2020.