AutoTrajectory: Label-free Trajectory Extraction and Prediction from Videos using Dynamic Points

Yuexin Ma$^{\star 1}$, Xinge Zhu$^{\star 2}$, Xinjing Cheng^{3}
Ruigang Yang^{3}, Jiming Liu^{1}, and Dinesh Manocha^{4}

[1] Hong Kong Baptist University [2] Chinese University of Hong Kong
[3] Inceptio [4] University of Maryland at College Park
yuexinma93@gmail.com

Abstract. Current methods for trajectory prediction operate in supervised manners, and therefore require vast quantities of corresponding ground truth data for training. In this paper, we present a novel, label-free algorithm, AutoTrajectory, for trajectory extraction and prediction to use raw videos directly. To better capture the moving objects in videos, we introduce dynamic points. We use them to model dynamic motions by using a forward-backward extractor to keep temporal consistency and using image reconstruction to keep spatial consistency in an unsupervised manner. Then we aggregate dynamic points to instance points, which stand for moving objects such as pedestrians in videos. Finally, we extract trajectories by matching instance points for prediction training. To the best of our knowledge, our method is the first to achieve unsupervised learning of trajectory extraction and prediction. We evaluate the performance on well-known trajectory datasets and show that our method is effective for real-world videos and can use raw videos to further improve the performance of existing models.

1 Introduction

For intelligent agents like robots and autonomous vehicles, it is crucial to be able to forecast neighboring traffic-agents' future trajectories for navigation and planning applications. Trajectory prediction for dynamic objects has been widely studied and is an active area of research. Some traditional methods for trajectory prediction are based on motion models such as Bayesian networks [25], Kalman filters [2], Gaussian process regression models [18], etc. These methods can deal with simple scenarios with very few moving instances, but are limited in complex real-world scenarios with many instances or agents interacting with each other. Recurrent Neural Network (RNN) and its variant long short-term Memory (LSTM) have become an effective way for trajectory prediction due to its ability to model non-linear temporal dependencies in sequence learning and generation [31, 6]. Based on these networks, recent works are able to achieve

*Equal contribution

good accuracy on predictig trajectories for pedestrians [1,17], vehicles [24,33], and heterogeneous traffic-agents [29]. However, all of the above methods operate in supervised manners, which rely heavily on labeled trajectory data. One general method to get a trajectory dataset [35,26] is to label consecutive positions of moving traffic-agents (pedestrians or vehicles) on fixed-view videos and then transfer the trajectory from the image coordinate system to a real-world coordinate system. Labeling consecutive objects from videos is complex and expensive [19]. There is a great demand for an unsupervised learning method to alleviate the dependence on annotations by simply taking raw videos as input and automatically extracting trajectories for training prediction network.

The most pivotal and challenging task for label-free trajectory extraction is capturing the moving objects, which we also call dynamic instances, from videos without any supervision. There are some related problems that also need to learn the motion dynamics of objects from videos, like activity prediction [27], video prediction [21,30], and object tracking [19,11]. However, we found they did not perform well for common bird's-eye view videos (sometimes, we may just see the heads and shoulders of pedestrians). For such videos, it is difficult to distinguish instances for the network just by appearance and structure features, while the above methods all rely on these features. To extract trajectories for dynamic instances in videos, we need consider not only the appearance and structure features in spatial space, but also the dynamic features (consecutive motions of objects) in temporal space. Our work is based on this consideration.

Main Results: In this paper, we propose a label-free learning-based method AutoTrajectory for trajectory extraction and prediction to overcome the above difficulties. To better capture the motion dynamics of moving objects in the video, we use the concept of *dynamic points*, which can focus on dynamic locations on images. These points are derived by keeping the spatial appearance and structure consistent via self-image reconstruction and maintaining the temporal dynamic features to be consistent in consecutive frames. Because our target is to get trajectories of instances, then we use optical flow and clustering algorithms to aggregate dynamic points to instances and extract trajectories by the matching method. Finally, we use these trajectories to train the trajectory prediction network. The whole process uses no labels. Our approach contains four main parts, including dynamic point modeling, dynamic-to-instance aggregation, trajectory extraction, and trajectory prediction. The main contributions of our work are:

- We propose a label-free trajectory extraction and prediction pipeline, which can extract trajectories of dynamic instances from raw videos directly and train a prediction network.
- We propose a novel forward-backward dynamic-point extractor, which could capture dynamic features in consecutive images.
- We propose a dynamic-to-instance mechanism, which could aggregate dynamic locations to instances.
- Our method is effective and has good scalability. With more raw videos, our method can also improve existing methods in a semi-supervised manner.

2 Related Work

2.1 Trajectory Prediction

Classical model-based approaches for trajectory prediction [25, 2, 18, 28] focus on the inherent motion regularities of objects themselves. However, the motion of dynamic objects in the real world is diverse and can be governed by many factors, like neighboring objects' motion states and the environment. These methods are limited in modeling complex scenarios. Recently, RNN and its variant LSTM have achieved great success in modeling sequence prediction tasks [31, 6]. Based on these basic networks, many prediction approaches [1, 17, 24, 33, 29, 9, 8, 43, 39, 7, 38, 48, 32, 44] have outperformed classical methods in real-world benchmarks. However, these supervised methods require large-scale, well-annotated trajectory data. Two main ways to generate the data are labeling moving instances from fixed-view videos and LiDAR point clouds. Both ways are expensive and time-consuming. Even though there are a lot of videos captured by street or commodity cameras, they cannot be used to improve the prediction performance without annotation. We try to solve this problem by using unsupervised manners.

2.2 Supervised Multi-object Tracking

Except for manual labeling, another possible solution for getting trajectories from videos is using current SOTA trackers. However, most modern trackers[5, 4, 13, 40, 42, 49, 46] follow the tracking-by-detection paradigm. The performance depends largely on the detector used to find the objects as the tracking targets and the detector requires large-scale labeled data. Besides, the tracker is always trained for fixed-categories, which is hard to adapt to other domains. Recent trend in multi-object tracking is combining both detection and tracking in one framework[15, 37, 3]. However, they do not overcome the above limitations. Our approach focuses on exploring the nature of video, *i.e.*, the dynamic information, which is naturally category-free and works well on all domains.

2.3 Unsupervised Learning for Dynamic Modeling

To extract trajectories from sequential frames, a crucial step is learning the motion dynamics of the video. Many works have explored unsupervised methods for dynamic modeling for videos to solve different problems [27]. Based on keypoint-based representation [20], the video prediction approach [30] could decouple pixel generation from dynamic prediction. [21] combines keypoints and extra action classes to help generate action conditioned image sequences. Inspired by the function of keypoint on video prediction and generation, we designed dynamic point. For unsupervised tracking, unsupervised single object tracking is the mainstream [36, 47, 45]. However, they cannot handle the scenes with multiple objects. For unsupervised multi-object tracking, the pioneering work AIR [12] proposes a VAE-based framework to detect objects from individual images through inference, which is followed by [22, 19]. [11] makes use of spatially invariant computations and representations to exploit the structure of objects in videos. In our initial attempts, we applied these unsupervised methods to locate dynamic instances on pedestrian videos directly but got poor results. The primary reason is that the above methods rely on structure and appearance features of objects, which are not applicable for trajectory extraction from bird's-eye view videos, where the these features are not very obvious.

3 Our Approach

3.1 Problem Definition

Given raw videos without any annotations, our task is to obtain a trajectory predictor in an unsupervised manner. We solve this problem by two main steps: trajectory extraction and trajectory prediction. For trajectory extraction, the input is raw videos captured by street cameras, and the output is $R = \{r_1, r_2, ..., r_n\}$, where R denotes all trajectories of moving objects in the videos. The trajectory for the ith object is defined as a set of discrete positions in the real-world coordinate system: $r_i = \left\{p_i^{t_{start}}, p_i^{t_{start}+1}, ..., p_i^{t_{end}}\right\}$, where $[t_{start}, t_{end}]$ denotes the time interval when the object occurs in the video. For the trajectory prediction, the extracted trajectories R acts as the dataset for training and validating the prediction network. The predictor observes objects' trajectories of an time interval and predicts their trajectories in the following period, like observing trajectory of 3s and predicting the trajectory for the next 5s. Without any label, we finally compute a trajectory prediction predictor.

3.2 Method Overview

We propose a label-free pipeline to generate the trajectory and then train the trajectory predictor. Specifically, our approach consists of four components: Dynamic-Point Modeling, Dynamic-to-Instance Aggregation, Instance Matching, and Trajectory Prediction. The first three parts form the unsupervised trajectory extraction. We show the pipeline in Fig. 1. In what follows, we will present these components in details.

3.3 Dynamic-Point Modeling

This part performs the unsupervised discovery of the dynamic points. Given a sequence of images, including Image$_1$ (I_1), Image$_{t-1}$ (I_{t-1}), Image$_t$ (I_t), and Image$_{t+1}$ (I_{t+1}), our objective is to capture K pixel locations, namely dynamic points $\Phi \in \mathbb{R}^{K\times 2}$, which correspond to the locations of moving regions in I_t.

The detailed networks are shown in Fig. 1**(1)**. The first image provides the background and layout features. Two pairs of consecutive images are used to capture the dynamic points in I_t. Both background features and the dynamic-point gaussian heatmaps are used to reconstruct the image (I_t). The learning objective $\mathcal{L}$ then consists of two parts, consistency loss $\mathcal{L}_C$ and reconstruction

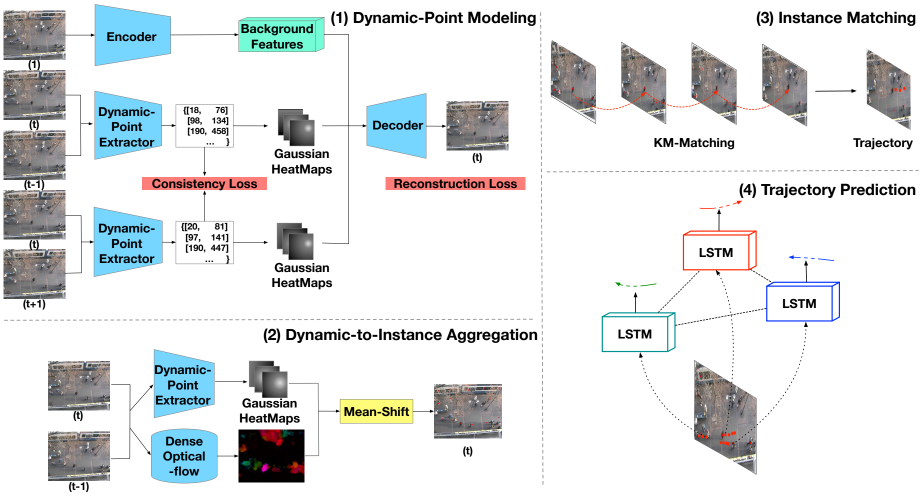


Fig. 1. Pipeline and main components of AutoTrajectory. Specifically, the first three components form the unsupervised trajectory extraction.

loss $\mathcal{L}_R$, to regularize the dynamic points extraction and image reconstruction, respectively. The total objective is formulated as $\mathcal{L} = \mathcal{L}_R + \beta\mathcal{L}_C$.

Forward-Backward Dynamic-Point Extractors. Keypoints are known as natural representations of objects. Some methods for video prediction [21, 30] encode single frames to keypoints to make the representation spatially structured and then generate videos. For the trajectory extraction from bird's eye view videos (Fig. 2(a)), the movement features in the temporal space are very important due to the limited appearance and structure features. Thus, we extend keypoints to dynamic points by utilizing more consecutive infomation in the temporal space. Dynamic-point extractors use two consecutive images to capture the dynamic points Φ. Two sets of images are applied in both forward (*i.e.* from $t-1$ to t) and backward (*i.e.* from $t+1$ to t) directions to keep the dynamic points of I_t consistent. The consistency loss is a location-wise MSE loss.

$$\mathcal{L}_C = ||(\Phi_{forward} - \Phi_{backward})||_2^2. \tag{1}$$

Gaussian Heatmaps. After obtaining dynamic points $\Phi \in \mathbb{R}^{K\times 2}$, we use gaussian heatmaps $\mathcal{H} \in \mathbb{R}^{H\times W\times K}$ to encode these points Φ into pixel representation, which is more suitable as the input for the convolutional reconstruction network. We first normalize the dynamic points via Softmax (*i.e.* Φ^* after normalization). Then each dynamic point is replaced with a gaussian function:

$$\mathcal{H} = \exp(-\frac{1}{2\sigma^2}\|\Phi - \Phi^*\|^2), \tag{2}$$

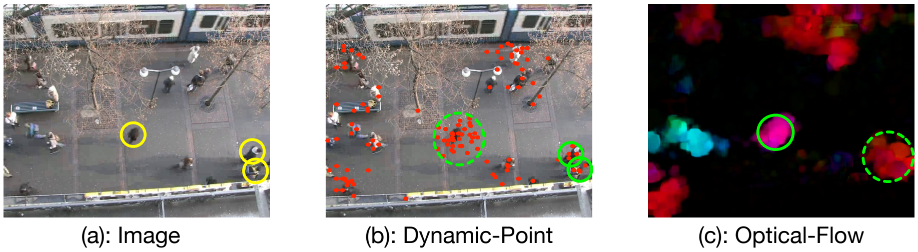


Fig. 2. A sample of image with dynamic points and optical flow. Yellow circles denote the pedestrians. Green dashed circles denote the poor instance-level representations. Solid green circles indicate the better instance-level descriptions.

where σ is a fixed standard deviation. The result $\mathcal{H} \in \mathbb{R}^{H \times W \times K}$ is the gaussian heatmap that describes the dynamic locations; it is also used as an input to the decoder network.

Decoder. The decoder network utilizes background and layout features and dynamic-point heatmaps to reconstruct the image (*i.e.* the reconstructed image is I_t^*). The reconstruction loss is a pixel-wise L2 loss:

$$\mathcal{L}_R = \|(I_t^* - I_t)\|_2^2. \tag{3}$$

In this way, the objective could induce the representation of dynamic points for reconstructing the specific image in an unsupervised manner. Meanwhile, image reconstruction can make full use of the appearance and structure information in the spatial space, which is a complement to the focus on dynamic motions.

3.4 Dynamic-to-Instance Aggregation

Dynamic points could detect dynamic locations on images, while trajectories originate from instances. After acquiring the well-trained dynamic-point extractor in the previous step, we aim to group these dynamic points to get the instance-level location information. Intuitively, the solution is to cluster the dynamic points to instance points directly. However, the dynamic points have some characteristics: it shows better instance-level information (distinguishing different objects well) when multiple objects are close to each other while shows loose when solo object occurs.

We tackle this problem by introducing the optical flow into the instance-level information collection. An example of the dynamic points and optical flow is shown in Fig. 2. We can observe that the dynamic points correspond to a better instance-level representation, when multiple objects are in close proximity (solid green circles in (b)). The optical flow shows the compact representation for solo objects (solid green circle in (c)). It shows that that dynamic points and optical flow are complementary.

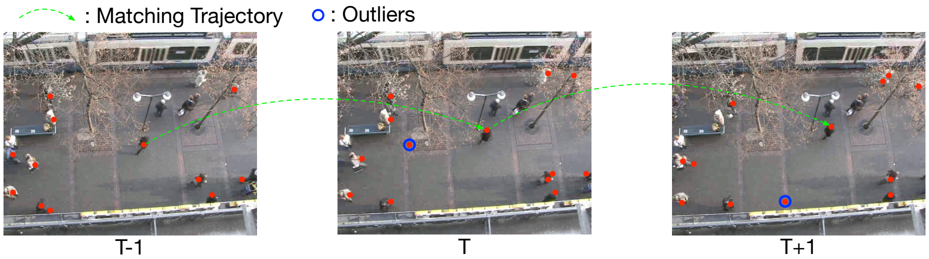


Fig. 3. An example of instance matching. Green dashed line denotes the instance points matching across timesteps. Blue circles denote the outliers of the instance points (also mean missmatching points).

Specifically, we use a pair of consecutive images (I_t and I_{t-1}) to extract the dynamic point representation and optical flow, training the dynamic-point extractor in step 1 and applying unsupervised optical flow method [14]. The gaussian heatmaps are upsampled to the original image size via bilinear interpolation. Then both gaussian heatmaps and optical flow are concatenated as the input to the clustering method, *i.e.* mean-shift, to get the cluster centers, which are the coordinates of objects.

Region of Validity. Since there exist invalid regions for moving objects in images (railway in Fig. 2) and the background is static for a fixed camera, we apply the region of validity to filter these outliers located in the invalid regions. We show the details in the experiment section.

3.5 Instance Matching

The instance points obtained from the clustering method are independent across time. To obtain the trajectory, we perform cross-time instance matching. The basic idea is to establish a cost matrix between two consecutive images where each entry indicates the distance between two instance points across two images. Then we apply the Kuhn-Munkres (KM) algorithm to calculate the minimum-cost matching. To better incorporate the appearance feature, we also use the RGB information as a part of distance. The final distance function is designed as $\mathcal{D}_{ij} = dist(P_i, Q_j) + \lambda rgb(P_i, Q_j)$, where P_i and Q_j are two instance points from two images. $dist(\cdot)$ is the Euclidean distance and $rgb(\cdot)$ is the L1 distance.

Specifically, the cost matrix $\mathcal{C} \in \mathbb{R}^{M \times N}$ is defined as the all-to-all distance between two images, where M and N indicate the number of instance points in two images. We use the KM algorithm with the cost matrix $\mathcal{C}$ to get the minimum-cost matching. The workflow is shown in Fig. 3. The matching pair from the KM algorithm is specified as a *true* pair if its distance is less than the pre-defined threshold $\mathbb{D}$, otherwise it is a *false* pair. Note that there exist

http://software.clapper.org/munkres/

some outliers that do not match any point. We label these outliers with blue circles. To handle these points, we apply some specific methods to filter them. For the points in image T, if we cannot find the former matching points in image $T-1$ but can find the matching points in image $T+1$, we label these points as the starting points of the sequence, otherwise we label them as outliers. This bidirectional filter benefits the precision of cross-time matching.

3.6 Trajectory Prediction

After extracting the trajectories in the pixel coordinate system, we transfer them to the real-word coordinate system and use them as the dataset to train and validate the prediction network in the last stage. At any time t, the status for the ith dynamic instance can be represented as the location $p_i = (x_i^t, y_i^t)$. The task for the prediction network is to observe the status of all the dynamic instances in the time interval $[1 : T_{obs}]$ and then predict their discrete positions at $[T_{obs}+1 : T_{pred}]$. We have highlighted many learning-based works in Section 2.1 and these methods can be directly used in our approach. Because the datasets we use are human crowd videos, we utilize some classical LSTM-based approaches for pedestrian trajectory prediction in our experiments to verify the effectiveness of our unsupervised method.

3.7 Optimization

In the proposed approach, dynamic-point modeling and trajectory prediction stages have trainable parameters, and the other two stages are non-parametric. The whole workflow is stage-by-stage. We first train the dynamic-point modeling part. An ADAM optimizer with learning rate = 1e-4 is used for optimization. β is 0.5, σ is 0.1 and $\lambda = 0.2$. Then we apply the well-trained dynamic-point extractor to access the dynamic points. After dynamic-to-instance and instance matching, we get the extracted trajectories. For the trajectory prediction part, we follow the settings in the original paper to train the network optimizer, including the observation and prediction length.

3.8 Network Architecture

Dynamic-Point Modeling. For the dynamic-point extractor, we use the basic block (Conv2d + BatchNorm2d +Leaky Relu) in VGG [41] as the unit. The sizes of Conv2d are: [64, 128, ‘M’, 256, 256, ‘M’, 512, 512, ‘M’, 512, 512], where ‘M’ denotes the MaxPooling and each number indicates the size of one unit. For the encoder, we use a structure similar to the dynamic-point extractor. For the decoder part, we use the reverse setting of the encoder to keep the output and input size consistent. The detailed setting is [512, 512, ‘U’, 256, 256, ‘U’, 256, 256, ‘U’, 128, 64], where ‘U’ denotes the bilinear upsampling.

4 Experiments

4.1 Implementation Details

For trajectory prediction, we use several LSTM-based models, including Vanilla-LSTM, Occupancy-LSTM (O-LSTM), and Social LSTM (S-LSTM) [1]. They are trained by ground truth data before. In our approach, we use our extracted trajectories to train these models. Following the original setting in S-LSTM, we filter our extracted trajectories by removing the trajectories with lengths less than 20 frames (8 seconds). We set K=180 so that the dynamic points could distribute all moving objects.

Evaluation Metrics. We evaluate our performance on three aspects: detected instance points, extracted trajectories, and predicted trajectories.

We introduce recall and precision to test the quality of instance points extracted from Dynamic-to-Instance Aggregation. We give the detailed explanation as follows. (1) True-Positive instance points: instance points where the distance between detected instance points and the ground-truth points is less than the threshold $\mathcal{D}$. (2) Recall: the ratio of True-Positive points to all ground-truth points. (3) Precision: the ratio of True-Positive points to all detected instance points. We term them Ins-Recall and Ins-Precision, respectively.

We also apply recall and precision to test qualities of extracted trajectories. The True-Positive trajectories are defined as: trajectories where the average distance between extracted trajectories and ground truth trajectories across timesteps is less than the threshold $\mathcal{E}$. The definition of recall and precision is similar to the statement above. We term them Gen-Recall and Gen-Precision, respectively. Note that there exist some conditions where one detected instance point (or trajectory) corresponds to several ground truth points (or trajectories), or vice versa. We use the KM algorithm to get the minimum cost matching. Both precision and recall are calculated on average. We set $\mathcal{E} = 1.5$ and $\mathcal{D} = 1.5$.

Similar to prior work [1], we use two popular evaluation metrics for predicted trajectory evaluation: (1) Average Displacement Error (ADE): Average L2 distance between predicted trajectory and the ground truth over all timesteps. (2) Final Displacement Error (FDE): The distance between the predicted final destination and the true final destination in the ground truth. Besides the comparison between our unsupervised method and supervised methods, we also conduct semi-supervised experiments by using our extracted trajectories as extra data to train supervised models.

4.2 Datasets

For the dynamic-point modeling part, we use two publicly available datasets: ETH [34] and UCY [23] as the training data. These two datasets are captured by fixed-cameras. Although there are some other datasets containing videos of traffic scenarios such as KITTI [16] and Argoverse [10], the videos are all captured in drivers' view. The camera is moving and and they do not provide the homograph matrix for each frame, which is not infeasible for our method.

We follow Social LSTM [1] to split the video to frames at every 0.4 seconds. For the trajectory prediction stage, we need to convert pixel coordinates to real-world coordinates to train these LSTM-based methods. Therefore, the extrinsic matrix is required to transfer the pixel coordinates to the real-world coordinates. From the open-source codebase , it can be found that only three scenes (UCY-Zara01, UCY-Zara02, and UCY-University) have complete transform matrixes. We thus use these three scenes for trajectory prediction.

Table 1. Evaluation results of detected instance points. We compare the proposed method with the unsupervised tracking [19] method and unsupervised keypoint modeling method [30]. '-' indicates the model cannot converge in the dataset

Metric	Ins-Precision					Ins-Recall				
Dataset	ETH	Hotel	Univ	Zara1	Zara2	ETH	Hotel	Univ	Zara1	Zara2
Un-Tracking [19]	8.3%	-	-	19.6%	21.4%	12.7%	-	-	10.1%	14.8%
Un-Keypoint [30]	16.8%	11.2%	-	33.1%	36.7%	14.1%	14.6%	-	39.4%	41.0%
Ours	**47.9%**	**37.1%**	**36.4%**	**58.7%**	**60.3%**	**58.3%**	**42.0%**	**31.4%**	**63.1%**	**67.9%**

4.3 Results

Experimental Results for Instance Points. We first evaluate the extracted instance points on various datasets. Since there is no annotation in any of the datasets, we use the unsupervised object tracking algorithm [19] and the keypoint-based video prediction algorithm [30] as baseline methods. From Table 1, several phenomena can be found: 1) in all datasets, the proposed dynamic-point modeling and dynamic-to-instance aggregation achieve consistently better performance than unsupervised tracking and unsupervised keypoint modeling; 2) for Hotel and Univ (where there are a large number of moving instances), unsupervised tracking method cannot converge while our method remains generalizable; 3) unsupervised keypoint modeling method without considering the sequential temporal information also performs poor (even does not converge in Univ dataset), while our method exploits the temporal consistency and achieves decent performance for all videos. Hence, for unsupervised tracking and keypoing modeling methods, it is difficult for them to extract dynamic instances from these videos, which are in bird's-eye view containing limited appearance and structure features. Instead, the proposed dynamic-point modeling and dynamic-to-instance aggregation could better handle the difficulties.

Visualization for the decoder. To investigate the performance of the decoder part in dynamic-point modeling, we visualize the reconstructed images in Fig. 1 in the supplementary material. It can be observed that the moving pedestrians are well captured and reconstructed, even with a large number of moving objects. The reconstructed images are also real and decent. Hence, it can be found

https://github.com/trungmanhhuynh/Scene-LSTM

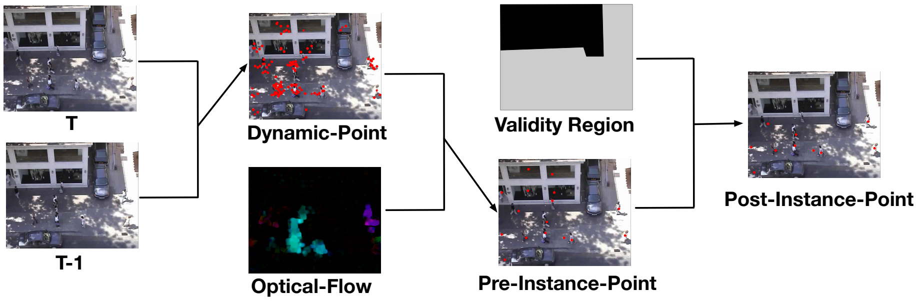


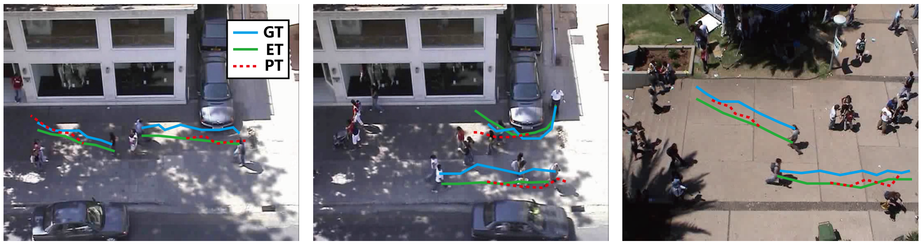
Fig. 4. Visualization for the output of each step in dynamic-to-instance aggregation. For the image of the valid region, the grey color denotes the valid part while the black color indicates the invalid region.

that our dynamic modeling does capture the dynamic information and could reconstruct the input image.

Visualization for each step in dynamic-to-instance aggregation. To give a more intuitive description, we visualize the output of each step during instance-point extraction in Fig. 4. Specifically, we first use Image (**T**) and Image (**T-1**) to extract the dynamic points. Then both dynamic points and optical flow are used to get the pre-instance points. Due to some invalid regions (buildings, railways) for pedestrians, we constrain these instance points with the valid region map. Because the background for a fixed-camera is static, it is easy to circle the valid region on just one frame. After that, we obtain the post-instance points.

Experimental Results for extracted trajectories. We use Gen-Recall and Gen-Precision to test the performance of extracted trajectories. Three datasets, including Zara1, Zara2, and Univ, are reported. The results are shown in the following; Gen-Precision of Zara1, Zara2 and Univ is 49.1%, 53.7%, and 23.7% respectively. Gen-Recall of Zara1, Zara2 and Univ is 52.9%, 54.4%, and 20.6% respectively. Our method could generate about a half number of trajectories similar to the ground truth for general videos. We visualize some extracted trajectories in Fig. 5. For very crowded scene (Univ), the performance drops due to the mismatching of instances. We show some bad cases in Fig. 2 in supplementary material. When multiple pedestrians meet, the error of instance matching occurrs and the trajectories of these pedestrians are biased in the wrong direction. It is also a fundamental obstacle for multi-object tracking methods.

Experimental Results for Trajectory Prediction. Because the datasets (Zara1, Zara2, and Univ) we use are about pedestrians, we test the extracted trajectories with three popular models for predicting trajectories of pedestrians, including LSTM, O-LSTM, and S-LSTM. We use a popular evaluation method, the leave-one-out approach, to test the trajectory prediction part, where we train



(a) (b) (c)

Fig. 5. Visualization for trajectory prediction. We display three examples with the ground truth trajectory (GT in green line), the extracted trajectory by our method (ET in blue line), and the predicted trajectory by our method (PT in red dashed line).

Table 2. Experimental results of trajectory prediction. We use three popular models to test the extracted trajectories, where O-LSTM and S-LSTM are both from Social-LSTM [1]. LSTM(sup), O-LSTM(sup), and S-LSTM(sup) indicate these models in a supervised manner. The unit for ADE and FDE is meters

Metric	ADE			FDE		
Dataset	Univ	Zara1	Zara2	Univ	Zara1	Zara2
LSTM	0.936	0.729	0.742	1.512	1.24	1.338
O-LSTM	0.875	0.511	0.579	1.427	0.947	1.092
S-LSTM	0.892	0.477	0.495	1.45	0.911	1.03
LSTM (sup)	0.52	0.43	0.52	1.25	0.93	1.09
O-LSTM (sup)	0.35	0.22	0.28	0.90	0.46	0.58
S-LSTM (sup)	0.27	0.22	0.25	0.77	0.48	0.50

on 2 scenes and test on the remaining one. We follow settings from prior works to observe the trajectory for 8 timesteps (3.2s) and predict the trajectory of 12 timesteps (4.8s). We use our extracted trajectories in the training process and test with the ground truth. The results of trajectory prediction are shown in Table. 2. The performance on Univ is worse than the other two scenes because it is more complex with a crowd of moving objects. We also display the performances of LSTM, O-LSTM, and S-LSTM with supervision. We can see that the supervised method performs better than our unsupervised methods. It is mainly because our extracted trajectories are not smooth (Fig. 5) as the ground truth and sometimes we have bad cases (Fig. 2 in supplementary material). However, for our unsupervised method without any label, the ADE is about half meter and FDE is about one meter, it still has good practical significance.

Visualization for trajectory prediction. In Fig. 5, We show several examples to display the ground-truth trajectory, our extracted trajectory, and our predicted trajectory. From the visualization, we can find that the extracted trajectories mainly focus on the centers of moving objects, which demonstrates that our generated instance points can capture the main dynamic information of moving objects. After training on extracted trajectories, our trajectory predictor can also work on true trajectories, which also illustrates the usefulness of our extracted trajectories in an unsupervised manner.

Semi-supervised training for trajectory prediction. To show the capability of our extracted trajectories in improving current supervised prediction models, we conduct semi-supervised experiments. We first use the ground truth data of Zara1 to train the model. Then we use extracted trajectories from other datasets as extra data to further train the model. Table 3 shows the results of testing on Zara2. We can see that adding more our extracted trajectories in the training process will make the prediction results more accurate. It illustrates our method is feasible in using large-scale raw videos to improve current models.

Table 3. Results of Semi-supervised training

Dataset	Zara1		+Univ(Gen)		+Univ(Gen)+Zara2(Gen)	
Method	LSTM	S-LSTM	LSTM	S-LSTM	LSTM	S-LSTM
ADE	0.598	0.347	0.578	0.341	0.521	0.320
FDE	1.25	0.69	1.157	0.687	1.094	0.659

4.4 Ablation Study

In this section, we perform several ablation studies to investigate the effectiveness of different components of the proposed approach. We train the dynamic-point modeling part with all five scenes and test the performance on Zara1 and Zara2.

Components of Clustering. For the dynamic-to-instance aggregation part, we use two types of dynamic information as the features, *i.e.* gaussian heatmaps and optical flow. From Table 4, it can be found after removing the dynamic points and optical flow, the performance of instance points is about 20% worse. Additionally, the model without dynamic points performs worse than the model without optical flow, which also demonstrates that dynamic points play a more important role in the instance-point extraction.

Forward *vs.* Backward Extractors. In the dynamic-point modeling part, we apply a forward-backward cycle extractor to keep the dynamic points consistent in cycle timesteps. We try to remove one of them to perform the ablation study. From Table 4, it can be observed that removing the forward extractor or removing the backward extractor will decrease the performance. Both forward and backward extractors are important ingredients in the instance-point extraction.

Consistency Loss. Moreover, we remove the consistency loss between the forward and backward extractors to check the effect. The results in Table 4 show that the consistency loss further boosts the forward-backward extractors (about 3%-4%) during the instance-point extraction.

Scalability. To verify the scalability of the proposed dynamic-point modeling, we compare the model trained with all five scenes to the model trained with only two scenes (Zara1 and Zara2). The results in Table 4 show that more video data improves the performance. It also demonstrates that our methods keep good scalability and take full advantage of large-scale video data.

Table 4. Ablation studies for instance-point extraction. We make several variants to investigate the effectiveness of different components

Metric	Ins-Precision		Ins-Recall	
Dataset	Zara1	Zara2	Zara1	Zara2
Ours w/o Dynamic-Point	38.3%	39.8%	44.1%	48.2%
Ours w/o Optical Flow	40.7%	43.4%	49.9%	53.1%
Ours w/o Forward Extractor	46.8%	50.1%	54.4%	59.8%
Ours w/o Backward Extractor	52.1%	56.8%	57.8%	62.2%
Ours w/o Consistency Loss	56.2%	58.0%	59.1%	60.4%
Ours w/ only-two-scenes	52.3%	53.9%	58.1%	61.6%
Ours	**58.7%**	**60.3%**	**63.1%**	**67.9%**

4.5 Limitations and Future Work

Although the proposed method works in an unsupervised manner, there also exist some limitations. 1) The whole framework is not end-to-end. We train these learnable components one by one. 2) There are some hyper-parameters, which need fine-tuning when training with different datasets. 3) Since there is no target category, our method might focus on the dynamic part of non-target category, such as a car in the pedestrian trajectory dataset. We visualize some badcases in the supplementary materials. In the future work, we aim to incorporate the category-aware memory and template into the dynamic modeling to further distinguish different categories. And we will also explore dynamic point-based approach on drivers' view videos.

5 Conclusion

In this paper, we propose a complete pipeline for label-free trajectory extraction and prediction. To our knowledge, this is the first time unsupervised trajectory extraction and prediction have been explored. We make full use of the spatial consistency by image reconstruction and the temporal dynamic consistency by sequential frames to capture moving regions in videos through dynamic points. To extract trajectories at the instance-level, we also propose a novel aggregation approach to cluster dynamic points to instance points by compensating with optical flow. Without any supervision, our method uses raw videos to extract trajectories and train trajectory prediction networks. The experiments show the effectiveness and scalability of our approach.

References

1. Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Li, F.F., Savarese, S.: Social lstm: Human trajectory prediction in crowded spaces. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 961–971 (2016)
2. Başar, T.: A new approach to linear filtering and prediction problems (2001)
3. Bergmann, P., Meinhardt, T., Leal-Taixe, L.: Tracking without bells and whistles. In: Proceedings of the IEEE international conference on computer vision. pp. 941–951 (2019)
4. Bertinetto, L., Valmadre, J., Henriques, J.F., Vedaldi, A., Torr, P.H.: Fully-convolutional siamese networks for object tracking. In: European conference on computer vision. pp. 850–865. Springer (2016)
5. Bewley, A., Ge, Z., Ott, L., Ramos, F., Upcroft, B.: Simple online and realtime tracking. In: 2016 IEEE International Conference on Image Processing (ICIP). pp. 3464–3468. IEEE (2016)
6. Cao, C., Liu, X., Yang, Y., Yu, Y., Wang, J., Wang, Z., Huang, Y., Wang, L., Huang, C., Xu, W., Ramanan, D., Huang, T.S.: Look and think twice: Capturing top-down visual attention with feedback convolutional neural networks. 2015 IEEE International Conference on Computer Vision (ICCV) pp. 2956–2964 (2015)
7. Chai, Y., Sapp, B., Bansal, M., Anguelov, D.: Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. ArXiv **abs/1910.05449** (2019)
8. Chandra, R., Bhattacharya, U., Roncal, C., Bera, A., Manocha, D.: Robusttp: End-to-end trajectory prediction for heterogeneous road-agents in dense traffic with noisy sensor inputs. In: CSCS '19 (2019)
9. Chandra, R., Guan, T., Panuganti, S., Mittal, T., Bhattacharya, U., Bera, A., Manocha, D.: Forecasting trajectory and behavior of road-agents using spectral clustering in graph-lstms. ArXiv **abs/1912.01118** (2019)
10. Chang, M.F., Lambert, J., Sangkloy, P., Singh, J., Bak, S., Hartnett, A.T., Wang, D., Carr, P., Lucey, S., Ramanan, D., Hays, J.: Argoverse: 3d tracking and forecasting with rich maps. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 8740–8749 (2019)
11. Crawford, E., Pineau, J.: Exploiting spatial invariance for scalable unsupervised object tracking. ArXiv **abs/1911.09033** (2019)
12. Eslami, S.M.A., Heess, N.M.O., Weber, T., Tassa, Y., Szepesvari, D., Kavukcuoglu, K., Hinton, G.E.: Attend, infer, repeat: Fast scene understanding with generative models. In: NIPS (2016)
13. Fang, K., Xiang, Y., Li, X., Savarese, S.: Recurrent autoregressive networks for online multi-object tracking. In: 2018 IEEE Winter Conference on Applications of Computer Vision (WACV). pp. 466–475. IEEE (2018)
14. Farnebäck, G.: Two-frame motion estimation based on polynomial expansion. In: SCIA (2003)
15. Feichtenhofer, C., Pinz, A., Zisserman, A.: Detect to track and track to detect. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3038–3046 (2017)
16. Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: The kitti dataset. The International Journal of Robotics Research **32**, 1231 – 1237 (2013)
17. Gupta, A., Johnson, J.E., Fei-Fei, L., Savarese, S., Alahi, A.: Social gan: Socially acceptable trajectories with generative adversarial networks. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition pp. 2255–2264 (2018)

18. Hall, M.A.: Correlation-based feature selection for machine learning (2003)
19. He, Z., Li, J., Liu, D., He, H., Barber, D.: Tracking by animation: Unsupervised learning of multi-object attentive trackers. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 1318–1327 (2018)
20. Jakab, T., Gupta, A., Bilen, H., Vedaldi, A.: Conditional image generation for learning the structure of visual objects. ArXiv **abs/1806.07823** (2018)
21. Kim, Y., Nam, S., Cho, I.S., Kim, S.J.: Unsupervised keypoint learning for guiding class-conditional video prediction. ArXiv **abs/1910.02027** (2019)
22. Kosiorek, A.R., Kim, H., Posner, I., Teh, Y.W.: Sequential attend, infer, repeat: Generative modelling of moving objects. In: NeurIPS (2018)
23. Leal-Taixé, L., Fenzi, M., Kuznetsova, A., Rosenhahn, B., Savarese, S.: Learning an image-based motion context for multiple people tracking. 2014 IEEE Conference on Computer Vision and Pattern Recognition pp. 3542–3549 (2014)
24. Lee, N., Choi, W., Vernaza, P., Choy, C.B., Torr, P.H.S., Chandraker, M.K.: Desire: Distant future prediction in dynamic scenes with interacting agents. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 2165–2174 (2017)
25. Lefevre, S., Laugier, C., Guzman, J.I.: Exploiting map information for driver intention estimation at road intersections. 2011 IEEE Intelligent Vehicles Symposium (IV) pp. 583–588 (2011)
26. Lerner, A., Chrysanthou, Y., Lischinski, D.: Crowds by example. Comput. Graph. Forum **26**, 655–664 (2007)
27. Luo, Z., Peng, B., Huang, D.A., Alahi, A., Fei-Fei, L.: Unsupervised learning of long-term motion dynamics for videos. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 7101–7110 (2017)
28. Ma, Y., Manocha, D., Wang, W.: Autorvo: Local navigation with dynamic constraints in dense heterogeneous traffic. arXiv preprint arXiv:1804.02915 (2018)
29. Ma, Y., Zhu, X., Zhang, S., Yang, R., Wang, W., Manocha, D.: Trafficpredict: Trajectory prediction for heterogeneous traffic-agents. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 6120–6127 (2019)
30. Minderer, M., Sun, C., Villegas, R., Cole, F., Murphy, K., Lee, H.: Unsupervised learning of object structure and dynamics from videos. ArXiv **abs/1906.07889** (2019)
31. Palaz, D.: Towards end-to-end speech recognition (2016)
32. Pan, J., Sun, H., cheng Xu, K., Jiang, Y., Xiao, X., Hu, J., Miao, J.: Lane attention: Predicting vehicles' moving trajectories by learning their attention over lanes. ArXiv **abs/1909.13377** (2019)
33. Park, S., Kim, B., Kang, C.M., Chung, C.C., Choi, J.W.: Sequence-to-sequence prediction of vehicle trajectory via lstm encoder-decoder architecture. 2018 IEEE Intelligent Vehicles Symposium (IV) pp. 1672–1678 (2018)
34. Pellegrini, S., Ess, A., Gool, L.V.: Improving data association by joint modeling of pedestrian trajectories and groupings. In: ECCV (2010)
35. Pellegrini, S., Ess, A., Schindler, K., Gool, L.V.: You'll never walk alone: Modeling social behavior for multi-target tracking. 2009 IEEE 12th ICCV pp. 261–268 (2009)
36. Piekniewski, F., Laurent, P.A., Petre, C., Richert, M., Fisher, D., Hylton, T.: Unsupervised learning from continuous video in a scalable predictive recurrent network. ArXiv **abs/1607.06854** (2016)
37. 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)

38. Rhinehart, N., McAllister, R., Kitani, K.M., Levine, S.: Precog: Prediction conditioned on goals in visual multi-agent settings. ArXiv **abs/1905.01296** (2019)
39. Sadeghian, A., Kosaraju, V., Sadeghian, A., Hirose, N., Savarese, S.: Sophie: An attentive gan for predicting paths compliant to social and physical constraints. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 1349–1358 (2018)
40. Sharma, S., Ansari, J.A., Murthy, J.K., Krishna, K.M.: 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)
41. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR **abs/1409.1556** (2014)
42. Tang, S., Andriluka, M., Andres, B., Schiele, B.: Multiple people tracking by lifted multicut and person re-identification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3539–3548 (2017)
43. Tang, Y., Salakhutdinov, R.: Multiple futures prediction. In: NeurIPS (2019)
44. Wang, M., Shi, D., Guan, N., Zhang, T., Wang, L., Li, R.: Unsupervised pedestrian trajectory prediction with graph neural networks. 2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI) pp. 832–839 (2019)
45. Wang, N., Song, Y., Ma, C., Zhou, W., Liu, W., Li, H.: Unsupervised deep tracking. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 1308–1317 (2019)
46. Xu, J., Cao, Y., Zhang, Z., Hu, H.: Spatial-temporal relation networks for multi-object tracking. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3988–3998 (2019)
47. Zhang, S., Huang, J.B., Lim, J., Gong, Y., Wang, J., Ahuja, N., Yang, M.H.: Tracking persons-of-interest via unsupervised representation adaptation. International Journal of Computer Vision **128**, 120 – 96 (2017)
48. Zhao, T., Xu, Y., Monfort, M., Choi, W., Baker, C., Zhao, Y., Wang, Y., Wu, Y.N.: Multi-agent tensor fusion for contextual trajectory prediction. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 12118–12126 (2019)
49. Zhu, J., Yang, H., Liu, N., Kim, M., Zhang, W., Yang, M.H.: Online multi-object tracking with dual matching attention networks. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 366–382 (2018)