Adaptive Visual Interaction based Multi-target Future State Prediction for Autonomous Driving Vehicles

Li Du, Zixuan Wang, Leiquan Wang, Zhicheng Zhao, Fei Su,
Bojin Zhuang and Nikolaos V. Boulgouris, *Senior Member, IEEE*

***Abstract*—Predicting the state of dynamic objects in a real traffic environment is a key issue in autonomous driving vehicles. Various approaches have been proposed to learn the dynamics from visual observations with static background. However, minimal research has been conducted in a real traffic environment due to the complicated and changeable scenes. This paper proposes an adaptive multi-target future state prediction (position/velocity) method under autonomous driving conditions. In particular, an adaptive visual interaction method and control mechanism are introduced to overcome the change in the number of objects in continuous driving frames. In addition, a two-stream architecture with stage-wise learning is utilized for accurate object state prediction by simultaneously complementing spatial and temporal information. Experiments on two public challenging datasets namely Udacity (CrowdAI) and Udacity (Autti), demonstrate the effectiveness of the proposed method on multi-target dynamic state prediction in a real traffic environment.**

***Index Terms*—state prediction, autonomous driving, adaptive visual interaction, adaptive prediction control mechanism, two-stream architecture**

I. Introduction

REASONING and predicting the future states (position and velocity) of objects in a dynamic environment to support autonomous driving, is one of the most challenging topics in computer vision and machine learning [1, 2]. Using vision-based methods to calculate the states of obstacles in front of a driving car is an effective approach that enables a driving agent to make wise obstacle avoidance decisions. However, predicting the exact state from the current visual observations is a challenging problem due to the complicated and changeable traffic environment.

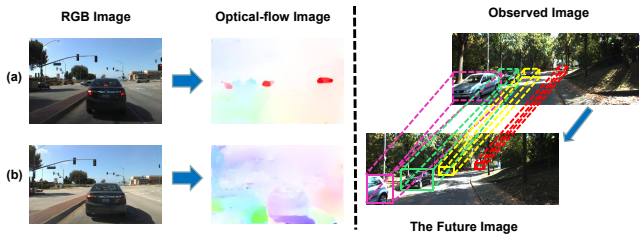


Fig. 1. Dynamic scene analysis. The images in (a) and (b) are captured at two different time points for the same scene. By visualizing the optical-flow features of them separately, we can see that the moving objects in (a) are the vehicles across the street while in (b) is the car in front of the video capture agent.

To overcome this challenge, interaction analogy is currently the most representative solution. It has been proposed to perform the analogy process of predicting with interaction networks [3] and [4]. *Visual Interaction Network* (VIN) [5] is a general purpose model that can support accurate predictions from visual observations. The effectiveness of VIN is demonstrated in static background videos with a fixed number of specific objects. However, autonomous driving vehicles have to deal with various objects under normal living conditions [6]. Furthermore, due to the varied number of objects, the influence of newly detected (or disappeared) objects should also be considered. To the best of our knowledge, nearly all available state prediction methods [4, 7] disregard the existence of dynamic complex interactions.

In the current study, we address the problem of dynamic object state prediction with an adaptive mechanism in autonomous driving. This work is motivated by the observation that determining the states of objects in a single video frame is ambiguous. As shown in Fig.1, RGB frames in (a) and (b) are separately captured at two different time points for the same scene. By visualizing their optical-flow features, we can see that the moving objects in (a) are the vehicles across the street, whereas in (b), the moving object is the vehicle in front of the agent car. Therefore, it is necessary to distinguish the moving states of objects in driving videos, as this may be the key clue for the driving agent to make wise driving decisions. To achieve this objective, we propose to predict objects' states in a future frame in this study.

Manuscript received October 30,2018; revised December 30,2018 and February 29, 2019; accepted March 10, 2019. This work was supported in part by the Chinese National Natural Science Foundation Grant 61532018 and 61471049, and in part by the Graduate Innovation and Entrepreneurship Project of Beijing University of Posts and Telecommunications under Project 2018-YC-A147. (*Corresponding authors: Zhicheng Zhao and Fei Su.*)

Li Du, Zixuan Wang , Zhicheng Zhao and Fei Su are with the School of Information and Communication Engineering and Beijing, Key Laboratory of Network System and Network Culture, Beijing University of Posts and Telecommunications, Beijing, China, (e-mail: duli@bupt.edu.cn; princexuan@bupt.edu.cn; zhaozc@bupt.edu.cn, sufei@bupt.edu.cn).

Leiquan Wang is with the College of computer and communication engineering, China University of Petroleum, Qingdao, China, e-mail:richiewlq@gmail.com.

Bojin Zhuang is withPing An TechnologyShenzhen Co., Ltd, e-mail:zhuangbojin232@pingan.com.cn.

Dr. Nikolaos V. Boulgouris is with the College of Engineering, Design, and Physical Sciences, Brunel University London, Uxbridge, United Kingdom, e-mail:nikolaos.boulgouris@brunel.ac.uk.

Continuous visual information includes spatial and temporal streams, which are collectively known as the two-stream architecture. The spatial stream describes scene and object information in an individual frame. The temporal stream captures dynamic motion information in consecutive frames. The two-stream architecture has been successful in video analysis tasks, such as action recognition [8] and semantic segmentation [9, 10]. However, the two-stream information is nearly utilized in parallel, and thus, it lacks necessary interactions and is inadequate for multi-target future state predictions under real autonomous driving conditions.

Compared with Fast-RCNN-related methods, no region proposal process occurs in Single Shot Detection (SSD) [11], thereby making it less time-consuming when processing large amounts of video data. In TVNet [12], a relaxed formulation of TV-L1 [13] optical-flow optimization was implemented in the form of a neural network that can be trained in an end-to-end fashion. The relaxation of the TV-L1 formulation endowed TVNet with the capacity to adapt to the type of motion features of which it is trained.

We focus on predicting the dynamic states of several typical objects, such as car, bus, and person, which are present in real driving scenes. As shown in Fig.1, the images in (a) and (b) nearly exhibit the same spatial features, but they have different optical flow features caused by varying relative motions in two complicated scenes. Therefore, it is difficult to determine whether it is the spatial or the temporal information in the observed images that is playing a leading role in generating state information for each object. To address this problem, a feedback process from the spatial stream based on TVNet to the temporal stream based on SSD is introduced to dynamically adjust the information generation progress for the observed images. The experimental results on real driving datasets show that utilizing the location information of detected objects to facilitate an accurate temporal motion generator will definitely contribute to dynamic object state prediction.

Human drivers are capable of extracting and representing key visual information, reasoning, and predicting the relations, dynamic interactions of objects in complex dynamic driving scenes. However, such tasks are difficult to accomplish for an autonomous driving system. In this work, we propose a multi-target physical state prediction method that aims to solve future state (position/velocity) prediction under complicated and dynamic driving conditions. In particular, the major contributions of this work are as follows.

- An adaptive visual interaction method is proposed for multi-target state prediction to adapt to dynamic and changing driving environment, without including the constant coordinate channels of an image;
- A two-stream architecture with stage-wise learning progress is utilized for a robust visual description by adaptively complementing spatial and temporal information;
- Extensive experiments are conducted on the Udacity (Crowdai) and Udaticy (Autti) datasets. Mean average precision (Map) is used to evaluate the prediction efficiency of our method for each object, and a visible trajectory description method is introduced to show its continuous prediction ability. The experimental results demonstrate that the proposed method can effectively predict the future states of objects under real autonomous driving conditions.

II. Related Work

State-of-the-art deep learning-based methods have recently surpassed humans in terms of face verification and image classification. However, these methods are incapable of capturing changes in real-world scenes in subsequent periods. To overcome this challenge, many researchers have designed advanced artificial neural networks to mimic human brain. NeuroAnimator is the first physic-based graphic model who made states-to-states predictions and utilized neural networks in the physically realistic animation task [14]. The application of machine learning techniques to physics-based fluid simulations was demonstrated in [15] through the regression forest method. Battaglia et al. [4] proposed a deep neural network - *interaction network* (IN) to analyze how objects in a complex system interact, thereby supporting dynamic prediction and inference about relations in a wide variety of complex real-world domains. Watters et al. introduced the *visual interaction network* [5] for learning the dynamics of a physical system from raw visual observations. Meanwhile, the experimental results in [16] demonstrated that latent representations can be learnt by a perceptual module and an object-based dynamic predictor, which result in accurate dynamic prediction.

References [17] and [18] demonstrated the ability of machine learning approaches for effective feature extraction in data processing. In recent years, great progress has been made by convolutional neural network (CNN) for feature learning and ConvNet based model pre-training in the salient object detection task [19–22]. Meanwhile, numerous image processing-based approaches can predict summary references and generate simple actions from the physical environment. [23] and [24] demonstrated the possibility of predicting the long-term movements of objects from a single image. In [25] and [26], the physical properties of objects in frames and video were learned by training their methods to fit parameters into physical equations. In [1], a deep neural network based method was proposed, which modeled the dynamics of robot interactions directly from images. Both methods are limited by the inherent accurate result from temporal dependency within continuous frames.

Pixel-level future state prediction tasks [27, 28] are always limited to a particular physical domain of interest and focus on short-term interval sequences. The early literature investigated simple predictable motions in relatively small image patches [29–31] and real videos [32–34]. To solve difficulties in the aperture problem [35], the former patchwise method is unable to deal with motion prediction mission for high-resolution videos. ALVINN's autonomous vehicle navigation task, which was based on a simple neural network in [36], was the first attempt to recognize driving action from pixel inputs. The efficiency of a pixel-level method was demonstrated in our prior work [37], which achieved accurate ego-motion predictions for ego-centric vehicle in real time.

Frame-level future state prediction is also known as future frame prediction. It aims to generate future frames by learning dynamic visual patterns from past frames and descriptions. The current video prediction task has extended to full frame prediction. Villegas et al. proposed a hierarchical approach based on Long Short Term Memory Networks (LSTM) [38] and an analogy-based encoder decoder CNN, to make long-term predictions for future frames in [39]. Xue et al. [40] proposed the cross convolutional network, a network structure for future frame synthesis based on a single input frame. The method relies on the modeling of future frames and operates through the combination of suitable kernels with extracted feature maps. NVIDIA verified ALVINN's concept by using a more efficient deep CNN in [41]. However, these methods always produced blurry results when predicting farther into the future, which causes multiple future uncertainties.

Mathieu and LeCun proposed a deep multiscale video prediction method in [42] based on the adversarial training mechanism, which overcomes the inherent blurry predictions caused by the standard mean squared error loss function. Demonstrative experiments were only performed on public human action datasets with a limited frame number. The related frame prediction tasks only consider the moving areas of the images. Yunseok Jang et al. proposed an appearance-motion conditional generative adversarial network in [47], which constrains the generated future video when given specific condition information (appearance and motion). This method constrains how the future may appear when given prior descriptions of future videos. However, obtaining sufficient prior information and forming a uniform description pattern for driving videos are difficult.

Our proposed future state prediction task in autonomous driving scenes was motivated by the physical reasoning learning processes in [3] and [48], in which a two-stream architecture with stage-wise learning strategy is proposed. In contrast with aforementioned methods, we intend to solve the target problem by combining instance detection with visual interaction. Meanwhile, spatial and temporal features are incorporated to eatablish a more reasonable prediction process.

III. Task and Method

A. Task Description and Formulation

VIN [5] was proposed by DeepMind and provides state prediction only for a fixed number of objects in videos with a static background. Our network is designed to adaptively provide state prediction for different objects in accordance with dynamic driving scenes. In addition, the input of VIN includes the constant coordination channels (an x- and y-coordinate mesh grid) of one image, which allows positions to be incorporated through considerable processing methods. Different from the aforementioned procedure, we undertake a different, more challenging, task i.e., to infer the physical states of images without the inclusion process and adaptively predict the future states of objects in ego-centric driving videos.

In this study, we focus on predicting future states: position p and velocity v of typical objects (car, bus, person) in future driving frames. Given an RGB frame-sequence X, our objective is to learn a generic prediction approach for extracting states-set s of all the objects in a future frame x. To solve this problem, we introduce a prediction model f to generate the candidate state set of a future frame. Therefore, our state prediction model can be defined as:

$$P(s|f(X)) : S \times F \rightarrow R \tag{1}$$

where *f(X)* denotes the prediction results of the observed images, and *P* measures the feasibility score of candidate state s under the predicted results *f(X)*.

Motivated by the feasibility of maximizing a posterior for true scenes described in [49], we propose to solve our posterior problem using the *maximum posterior estimation* method. In particular, an objective function is defined to find the optimal parameter θ as follows:

$$\hat{\theta}_{MAP} = \arg\max_{\theta} P\left(s|f(X)\right) = \arg\max_{\theta} P\left(\theta\right) P\left(f(X)|\theta\right). \tag{2}$$

Therefore, our prediction task can be supervised by the following *cross entropy loss*:

$$L_H = -\frac{1}{NK}\sum_{n=0}^{N-1}\left[\sum_{k=1}^{K} s_k^n log f(x_k^n) + (1 - f(x_k^n)) log(1 - s_k^n)\right] \tag{3}$$

where $f(x_k^n)$ is the predicted states of the k-th object in the n-th frame, s_k^n denotes its corresponding ground true states and s_k^n=$(p_k(n), v_k(n))$, *N* indicates the total number of input frame and *K* is the objects' number in the n-th frame. Meanwhile, *K* and *N* are constants, whereas $f(x_k^n)$ and s_k^n are feature vectors.

B. Two-stream Architecture

The two-stream architecture originates from the two-pathway hypothesis of the visual cortex in [50], wherein the ventral stream aims to recognize objects and the dorsal stream captures motion. The recently proposed two-stream architecture has achieved excellent results in various tasks, such as action recognition, semantic segmentation, and action detection. As described in [51], a video can be decomposed into spatial and temporal parts. The spatial part, which is in the form of individual frames, carries information about scenes and objects in the video. Meanwhile, the temporal part carries the dynamic information of video capturers (e.g.,cameras mounted in front of driving vehicles) and the objects in videos.

The demonstration experiment of VIN is based on simple mesh grid blurry images, without any other complicated object in their backgrounds. The driving images in our datasets are full of complex objects in driving scenes. We aim to predict the states of objects in images, and thus, other components can be regarded as noise, which increases the complexity in our task. Obtaining the dynamics of objects by relying only on the static information in an image is difficult. A two-stream learning architecture is proposed as our basic framework to capture complementary appearance information from a still frame and motion information between dynamic frames. As shown in Fig.2, the spatial stream is used to learn the appearance information in a continuous still frame, whereas the temporal stream obtains motion information between frames.

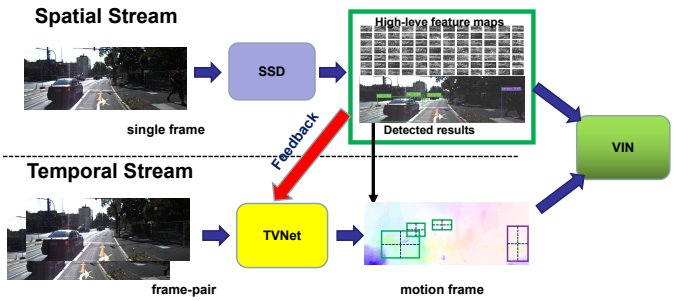


Fig. 2. Two-stream architecture with stage-wise learning. The SSD (in purple) works to sparse the output space of bounding boxes into a set of boxes over different aspect ratios and scales per feature map location. It also generates detection scores for each instance in each default box and produces the best adjustments to each box. The TVNet (in yellow) is applied to generate the optical-flow-like feature of each frame-pair. The VIN (in green) makes prediction based on the output of the the former two parts. (best viewed in color)

The learned high-level semantic features from the two streams are merged as the new input of the subsequent adaptive visual interaction predictor.

Evidently, our proposed two-stream architecture differs from its general architecture. The detection results from the spatial stream contribute to adjusting the training parameters in the temporal stream. The spatial stream locates the coordinates of the detected objects and helps calculate the velocity value for each object. When the detection accuracy of the spatial stream is lower than a fixed threshold, the training parameters (wrap and scale) of TVNet will increase after a feedback process.

1) The spatial stream: performs object detection on individual frames to extract useful static appearance information, which is a vital clue to detecting particular objects in a driving video frame. Considerable current object detection efforts [11, 55–57] use CNN [58] based methods for generic descriptors. We build our model on the recent advanced CNN-based object detection method, namely, SSD, and pretrain it on a large number of object detection datasets, such as the PASCAL dataset [59]. The input for this step is a single frame. The details are described in the *Evaluation* section.

2) The temporal stream: utilizes dynamic motion information in the consecutive frame pairs. Optical flow can arise from the relative motion of objects and the viewer [60, 61], and it is insensitive to the quantization of brightness levels and additive noise [62]. The input for this step is temporal continuous frame pairs. The two continuous frames are stacked to obtain the explicit optical flow displacement fields between them. To get the temporal information of dynamic objects in driving videos, which shows the moving trajectory of objects within the watching scope, the motion generator (TVNet) is used to extract the optical flow-like features of frame pairs by using a trainable CNN. This network imitates the optical flow calculation function of TV-L1 [13] by presenting its optimization iterations as neural layers. Thus, it can be directly used without additional learning.

C. Stage-wise Learning

Our model has three function components: an object detector (SSD), a motion generator (TVNet) and a state predictor (VIN). As depicted in Fig.2, SSD (in purple) parses the output space of bounding boxes into a set of boxes with different aspect ratios and scales per feature map location. SSD also generates detection scores for each object in each default box and produces the best adjustment for each box. TVNet (in yellow) generates the optical flow-like feature for each frame-pair. The VIN (in green) makes prediction based on the outputs of SSD and TVNet.

Video datasets exhibit differences in resolution caused by varying video capturers or driving environments. Such differences may result in unstable object detection ef?ciency for the spatial stream. To solve this problem, we introduce *feedback* processing from the *object detector* to the *motion generator*. When the detection precision of the *object detector* decreases, the initial parameters (wrap and scale) in the *motion generator* increase to a certain extent. By experimentally analyzing large numbers of the generated results from TVNet, we derive the following feedback function:

$$I = \alpha + \lceil \beta \cdot e^{-\frac{\sum_{i=0}^{N-1} S_i}{N-1}} \rceil \tag{4}$$

where I denotes the training parameters (wrap/scale) in the *motion generator*, α stands for the basic iteration, and β indicates the adjusted parameter. N is the total number of detected objects, and S_i is the detection score of the i-th object in the observed frame. The specific parameter setting process of this function is illustrated in *Motion generator configuration* part in the *Model Architecture* section.

The former mentioned reliance of *motion generator* on *object detector*, leading directly to the instability of *motion generator*'s time consuming for frame-pair sequences. The inputs of the *state predictor* originate from the *object detector* and the *motion generator*. Hence, ensuring that the two inputs will be ready at the same time is difficult. Meanwhile, the two-stream architecture results in a larger model compared with a normal single-stream network with numerous layers and parameters. Therefore, it must be time-consuming and computationally expensive in the training stage. Moreover, the number of available autonomous driving datasets for the object detection task is limited, and the detection ability of an *object detector* may be improved by fine-tuning the learned model on newly released datasets. To accelerate the learning process and ensure the learning ability of each component, we apply a *stage-wise* learning process to complete our future prediction task.

First, we train *object detector* to identify the spatial features of each observed frame and object detection results. Then, *motion generator* produces the temporal features of each continuous frame pair following the feedback process. Finally, future predictor is trained to obtain the target results by feeding the output features from the object detector and the motion generator.

IV. Model Architecture

To develop a system that can perform our target task effectively, we establish a deep neural network architecture, namely, the future state prediction model, based on VIN, TVNet, and SSD. The model architecture is shown in Fig.3.

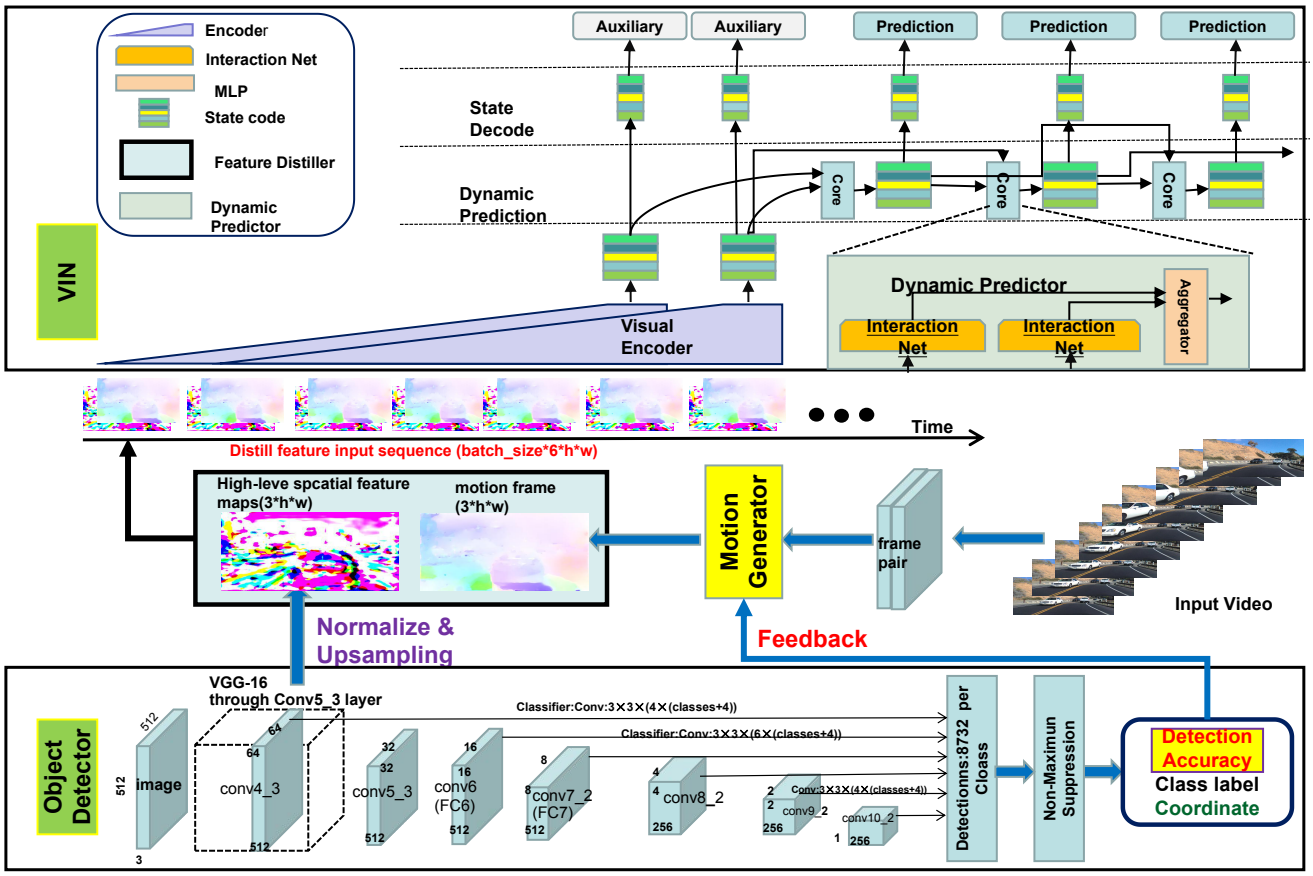


Fig. 3. Model architecture of our proposed method. A fully convolutional filter based *Object Detector* (SSD) is used to generate the box offset, detection accuracy and class label for each object in one frame. The output feature maps from the VGG16(conv4-3) with the size of $(512 \times 64 \times 64)$. To ensure the objects' physical spacial locations extracted from the observed image keeping in constant with its corresponding temporal characters in the motion frame, we apply a normalize upsampling process to further extract a higher normalized spacial feature representation. The newly generated feature is a 3-channel RGB image with the same size as the observed frame, and it will be combined with the 3-channel motion frame from the motion generator to generate a 6-channel feature input. We introduce a *feedback* processing from the *object detector* to the *motion generator*. When the detection accuracy from the *object detector* decrease, the initial parameters in *motion generator* (iteration and scale number) changes following the feedback function. The distilled feature input will be sent into the modified VIN [5], to obtain the states information of the future frame. (best viewed in color).

A. Object Detector Configuration

As depicted in Fig.3, our introduced *object detector* is an SSD with standard network VGG16 (conv4-3) [63] and 6 fully convolutional network layers. The network is based on a 2D convolutional network, and produces output scores for the presence of each object category in each default bounding box and adjusts the box to accurately match the object shape. The standard network aims to conduct high-quality image spatial feature extraction, whereas the added layers produce the vital detection feature maps, including multi-scale feature maps, convolutional predictors, default boxes, and aspect ratios.

The progressively added fully convolutional layers are arranged in descending order by size, which contributes to a multi-scale detection process. Each feature layer (basic network and added layers) produces a fixed set of detection features by using its assigned filters. A $(3 \times 3 \times p)$ kernel functions as a basic element for parameters prediction during the detection period or the shape coordinates the generation phase for the bounding boxes. When a kernel is applied to an $(m \times n)$ image field, we will obtain the corresponding output values.

1) Normalization and Upsampling: A fully convolutional filter is used to generate box offset for each feature map location. The output feature from VGG16 (conv4-3) measures $(64 \times 64 \times 512)$. To ensure that the physical spatial locations from the observed images remain constant with their corresponding temporal characters in the motion frames, we apply normalization and upsampling processes to further distill a higher normalized spatial feature representation for the output from base network.

Firstly, we normalize the 512 feature maps to 64×64 following the standard normal distribution. Secondly, we apply a channel-wise convolution process to fuse the feature maps from 512-channel to 3-channel higher-level feature maps. The feature maps are visualized into a 3-channel image. Finally, a 3-layer upsampling with bilinear interpolation is introduced to expand the 64×64 image to its raw size. The newly generated feature image will be sent to the subsequent *state predictor*.

2) Objective Function: For each detected object in observed frame, the output values from SSD include a 4D location offset, a 1D category score, and its corresponding class label. The score will be used to control the feedback process in motion generator and calculate the object's velocity.

Therefore, the loss function in the detection part is also the sum of localization loss (loc) and confidence loss $(conf)$ as indicated in [11]:

$$L_{det}(x,c,l,g) = \frac{1}{N}\left(L_{conf}(x,c) + \alpha L_{loc}(x,l,g)\right) \quad (5)$$

where N is the number of matched default boxes, L_{loc} is the smooth L1 loss between the predicted box (l) and the ground truth box (g) with the same bounding box regression method used in [11], and L_{conf} is supervised by the softmax loss on the multiple class confidence (c).

B. Motion Generator Configuration

We verify the application of TVNet to our complex feature extraction process for driving videos and introduce it to complete motion generation task in temporal stream.

1) Verification Experiment of TVNet: To demonstrate that TVNet can extract complex motion information from driving videos, we apply it to two typical publicly available datasets: KITTI (Residential) [64] and NVIDIA [41]. The KITTI dataset contains high-resolution images with complicated circumstance information, whereas the NVIDIA consists of relatively lower-resolution images captured under clear driving conditions. We concatenate the frames in each dataset into a continuous video sequence. The frame rate of NVIDIA is 24 f/s, whereas that of KITTI (Residential) is 10 f/s. The resolution of each image in NVIDIA is 455×256, while in KITTI is 1392 × 512. The two motion generators used to extract motion features have the same parameter setting.

The generated motion features are shown in Fig.4. The generated motion images in NVIDIA provide more complete and differential motion information than those generated in KITTI. Thus, we unify the resolution of each image to 455×256. To omit the influence of frame rate on motion feature extraction, we separate the statistics for the failure cases[1] in each motion dataset when the observed images in the two datasets have the same resolution. NVIDIA consists of 2475 frames, whereas KITTI includes 14077 frames. The results show that the failure rates of motion generation for NVIDIA and KITTI are 1.4% (35 in 2475) and 0.085% (12 in 14077), respectively. Therefore, we unify the frame rate in the following learning process for a fair comparison.

2) Feedback Process: Although TVNet can be used directly to generate motion information for dynamic image pairs without any ground truth data for training, the initial parameters (*wrap, scale and iteration*) severely affect the performance of the generated results. When *scale* controls the generation of an approximation field for the multi-scale scheme in a coarse-to-fine brightness linearized method, *wrap* defines the image warping degree in the pixel-wise brightness estimation process, whereas *iteration* denotes the epoch number in the block-wise convolution operations in TVNet. Fig.5 presents typical motion generation examples of driving videos. The generated motion results of the image pairs with a dynamic background contain more complex information than those with a static background. Under general driving conditions, the observed driving agents are always moving, and numerous dynamic backgrounds exist in the captured videos. Therefore, finding a mechanism to adjust the initial parameters for each image pair is necessary to provide useful motion information for the final prediction task.

[1]The failure cases will be shown in the *Appendix*.

Fig. 4. Some typical motion generation examples of observed images between their corresponding neighbor images. (a) Observed image-pair with static background. (b,h) Lane following. (c,g) Turn left. (d) Speed up. (e) Slow down. (f) Turn right.

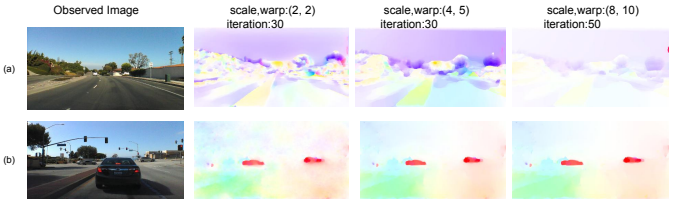


Fig. 5. The comparison results for image-pairs with static and dynamic background. (a) The observed image-pair with dynamic background. (b) The observed image-pair with static background. The variance of parameters have more effects on the motion image-pair with dynamic background than that with static background.

By learning the data-processing strategies in [65] and analyzing the motion generation performance of TVNet with different initial parameters, we develop a feedback function based on detection scores from *object detector*, as expressed in Eq.4. Given that the calculation time consuming of each image depends on the multiple values of *wrap, scale, and iteration*, we uniformly set *iteration* to 30 and I range in $[2, 8]$ to balance efficiency and cost. Meanwhile, we apply a linear regression method to identify the suitable parameters in the feedback function. The basic iteration number α is 2, and the range of parameter β is $[6, 10]$ in our task.

3) Generated Outputs: The output of the *motion generator* is a two-channel optical flow-like motion feature vector and is set to *feature distiller*. We visualize each generated feature vector as an RGB image, stack it with the output feature from

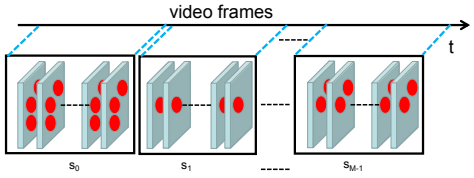


Fig. 6. We divide the continuous video into continuous interval sequences. The number of objects of each frame in the same sequence is equal.

object detector, and send them to *state predictor*. Meanwhile, the velocity of each detected object is calculated based on the detected coordinates from *object detector*. The bounding boxes of all the detected objects are rectangles, and each object moves as a whole. We extract the velocity vector $\vec{ve} = (u, v)$ of the pixel at the physical center of each bounding box and transform it into a constant value using the following equation to precisely obtain the relative velocity value of each object:

$$velocity = \|\vec{ve}\| = \sqrt[2]{u^2 + v^2} \tag{6}$$

where the generated velocity value and the detected coordinates from *object detector* are combined to form the input label for *states predictor*. Therefore, the state predictor in our proposed model can be trained in a semi-supervised manner.

C. Adaptive Prediction Control

In general driving conditions, both driving agent and the objects around it are moving, and thus, the numbers of objects in video are always changing. This phenomenon definitely adds difficulties to model definition in our task. To solve this problem, we propose an adaptive prediction control mechanism that can dynamically control the model's predefinition parameters in our modified VIN.

Motivated by the concepts of calculus and action recognition, we divide the continuous input video into short interval sequences. The number and states of the objects change excessively in each sequence, as shown in Fig.6. For special interval sequences without object that last for less than six frames, we do not perform any process. We can determine the detected category of each object, the number of objects in each frame, and the length of each sequence from the *object detector*. Therefore, we can dynamically predefine the VIN's architecture by using these parameters for each interval sequence. We complete this process by designing an interface class.

To compromise between computation efficiency and detection accuracy, we set a uniform number of objects for each sequence ranges in $[0, 6]$. When the last frame in each sequence is predicted, we directly output the generated state label of the first frame in the next sequence, which benefits from the stage-wise learning process. Therefore, the final prediction result of our task is the sum of the prediction results from all the interval sequences.

$$L_H = -\frac{1}{MNK}\lim_{\lambda\to 0}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left[\sum_{k=1}^{K} s_k^{m,n} log f(x_k^{m,n}) + (1 - f(x_k^{m,n}))log(1 - s_k^{m,n})\right]\Delta\delta_m \tag{7}$$

where $f(x_k^{m,n})$ is the predicted states of the k-th object in the n-th frame of the m-th sequence, and $s_k^{m,n}$ denotes its corresponding ground truth future states. M denotes the sequences' number, N stands for the number of total processing frames, and K is the number of sequences in the n-th frame. Meanwhile, the maximum length (λ) of the longest sequence tends to be zero. $\Delta\delta_m$ is the length of the m-th sequence. Eq. (7) can be regarded as a calculus function when M is larger than 100. Each dataset used in our experiment contains at least 1500 interval sequences.

D. State Predictor Configuration

The model detail of VIN is shown in the supplement section of [5]. The architecture of our adaptive predictor is depicted in Fig.3 and is also constructed with the following three components:

1) Visual Encoder: It encodes the distilled feature inputs into state code sets. We modify this encoder to adapt to our proposed task. Its input is a six-channel vector, including a three-channel high-level spatial feature map from the *object detector* and a three-channel motion frame from the *motion generator*. The visual encoder is composed of several convolution and pooling layers, to extract the states code from the distilled feature input. A state code is a combination of vectors. Each vector stands for one object in the current driving scene, and is a distributed representation of the physical states of the corresponding object. A sliding window is applied to the continuously distilled feature frames to produce a triplet of state codes.

In contrast with the original visual encoder in VIN, the input for this step is the continuously distilled feature batches. Each batch consists of six frames $[f_0, f_1, f_2, f_3, f_4, f_5\ ]$ as constrained by the aforementioned adaptive control mechanism. We concatenate the frames in each feature batch into five image pairs [F1, F2, F3, F4, F5] with an overlapping rate of one. Then, we apply an image pair encoder E_{pair} (described in Fig.7) to image pairs to obtain state codes [s1, s2, s3, s4, s5]. Each code is a length-64 vector. Five two-layer convolutional nets with a kernel size of three following an adaptive-pool2d are applied to each image pair to obtain its state codes. We design seven types of shared linear fully connected layers to adapt to the change in object numbers in various interval sequences. In particular, similar to [5], we use these layers in order to represent state codes as tensors of shape $N_{object} \times 64$, which are structures that can accommodate all N_{object} objects that are present in the current interval sequence. The five generated state codes can then be further concatenated pairwisely in a slot-wise manner into a tensor list $[S1, S2, S3, S4]$ of shape $N_{object} \times 64$ with 39 channels.

2) Dynamic Predictor: This predictor consists of several interaction cores, which produce predicted state code S^{pred} when consecutive state codes $[S1, S2, S3, S4]$ are injected. Interaction Net (IN) [5] takes a state code as input and gives its predicted state code. Relation net is used to concatenate the state code of each object with the others and get its relation dynamics, while the self dynamic of each object is obtained by concatenating its state code slot-wisely with itself. The primary difference of our adaptive predictor from the original one in [5] is its adaptive aggregation over multiple

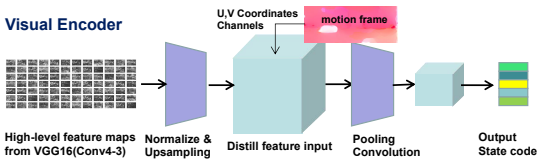


Fig. 7. Visual encoder. The image-pair encoder in visual encoder (in the spirit of [5]). It takes high-level feature maps from the spatial stream and motion frame from the temporal stream as input and output a candidate state code. The two features are stacked along their color-channel dimension be the distill feature input. (best viewed in color)

temporal offsets, which can be assigned based on the number of objects in each sequence, whereas the offsets in the original one are fixed. An adaptive-pool2d following a shared slot-wise multiple layer perception aggregator is introduced to concatenate the candidate future states of all the objects.

3) States Decoder: This decoder converts the predicted state codes from the adaptive dynamic predictor to states representation codes. A states representation code is a combination of the predicted physical states of objects.

V. Evaluation

To evaluate the effectiveness of the proposed prediction framework, we apply our model to two publicly available object detection datasets: Udacity(CrowdAI) and Udacity(Autti) [66]. The Udacity(CrowdAI) dataset involves driving in Mountain View California and neighboring cities under daylight conditions. It consists of 9423 frames and was annotated by CrowdAI via machine learning and by humans. Udacity(Autti) contains additional fields for occlusion. It was annotated entirely by humans using Autti. This dataset is slightly larger with 15,000 frames. The two datasets were collected using Point Grey research cameras operating at a full resolution of 1920 $\times$ 1200 at 2 Hz. Therefore, the state prediction task can also be attributed to predicting the future frame in the subsequent half second.

A. Object Detection Evaluation

The data generator's image transformation function is used to resize the input three-channel RGB images to 256x512. The same image preprocessing and data augmentation methods as SSD's original implementation in [11] are adopted.

Detection includes four categories (background, car, truck, pedestrian). When training our object detector, SSD is fine-tuned by optionally loading the trained weights in an available mature base network. The adopted available pretrained SSD model is trained on two object detection datasets: Pascal VOC (2007) and Pascal VOC (2012). We train the *object detector* on Udacity on Keras deep learning architecture within 100 epochs on a single 8G GPU. The weight decay of our training process is 0.001, and its momentum is 0.9.

We evaluate the detection results using the COCO-style average precision (AP) and the PASCAL-style AP with a single Intersection-over-Union (IoU) threshold of 0.7. In case the overlapping rate of the generated bounding box with its corresponding ground truth for each object is over 70%, it is regarded as correctly detected. To specifically analyze the

TABLE I
Testing IOU values.

Datasets \ APs	AP-min	AP-max	AP-total
Udacity(CrowdAI)	38.94	65.97	50.61
Udacity(Autti)	36.57	52.97	46.30

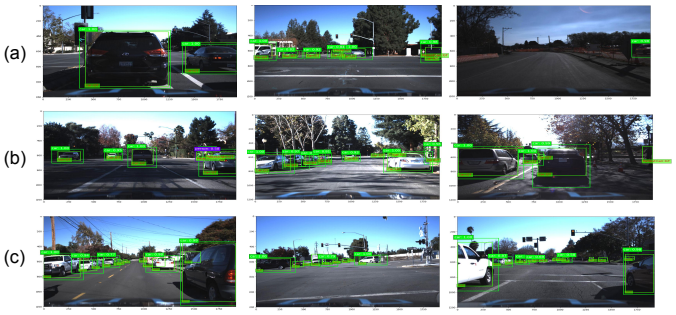


Fig. 8. The detection results of the *object detector*. The images in row (a) are easy cases. The images in row (b) show the general performance of our detector, and row (c) shows three hard cases. The boxes with red taps represent the groundtruth boundings, while the boxes with white tags shows the detected score of all the detected objects. (best viewed in color)

performance of our trained model, we count the IoU of each object in each frame and obtain the average IoU values of the entire dataset, which are listed in Tab.I. At the object level, the average minimum IoU is 0.3894 and the average maximum IoU is 0.6597 for the Udacity(CrowdAI) dataset, whereas the mean value of all the images' IoU is 0.5061 at the frame level. The proposed detection method performs worse in Udacity(Autti) for over occlusions in datasets. Compared with the general object detection tasks with IoU thresholds of 0.5 and 0.6 on COCO [67] and PASCAL [68], respectively, the detection results from our trained model can reach an acceptable level with a more strict threshold of 0.7 on a relatively simpler model architecture.

The detection results of our trained model on Udacity(CrowdAI) dataset are shown in Fig.8. The images in Row (a) show three easy cases with the highest detection precision. They demonstrate the efficiency of the object detector for processing images captured in simple and clear driving environments. The first image in Row (a) shows the complete object, without occlusions between them. The middle row shows objects with a relatively small size in the crossing street, and the last image was captured under dark light condition with an incomplete target. The images in Row (b) contain more objects than those in Row (a), with stable detection recall and presentation. The images in Row (c) contain large numbers of objects and heavy occlusions between objects. The relatively smaller objects are omitted, and the detected results exhibit an acceptable precision. In summary, our trained *object detector* can meet the detection requirement of the proposed task, by providing stable detection results for the subsequent two parts.

Meanwhile, the mean value of average precision for each object in the testing dataset is shown in Tab.II. Compared with typical object detection methods based on Faster-RNN, our trained SSD can yield reasonable detection results. The two

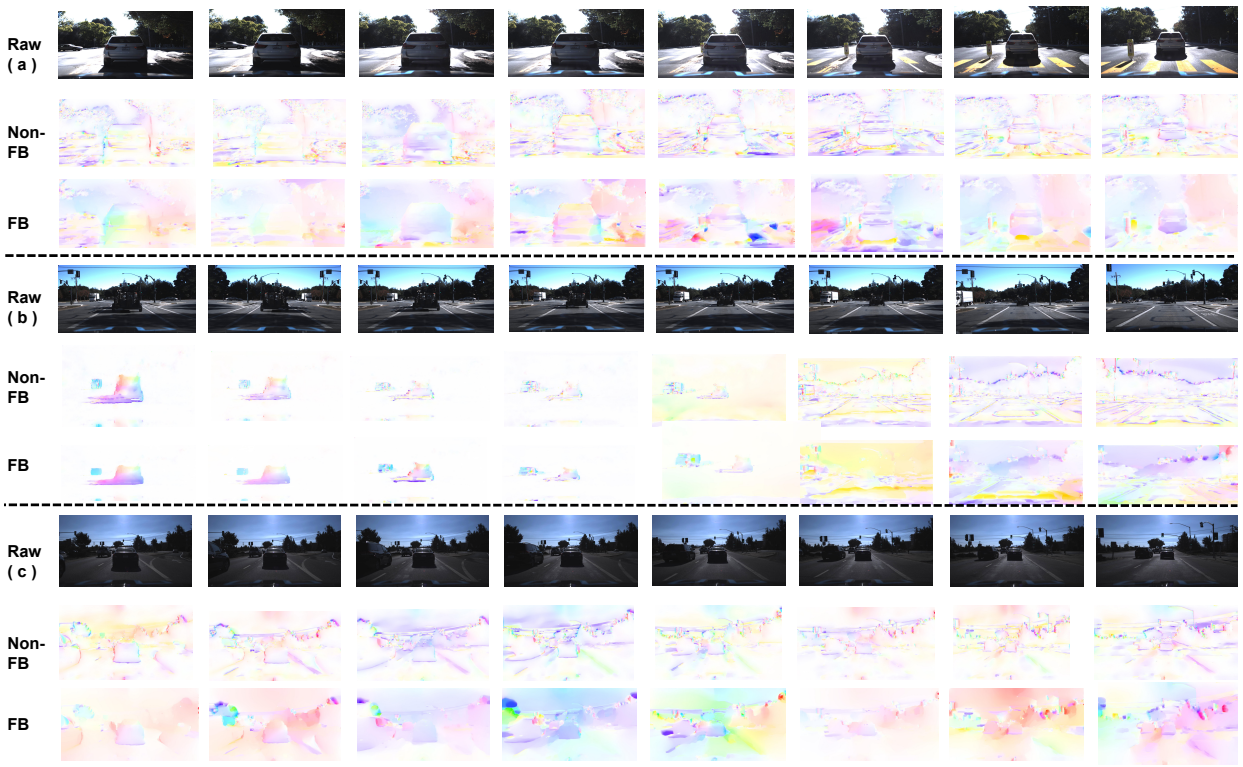


Fig. 9. Comparing results of the motion generation progress on Udacity(Autti). (best viewed in color) The rows denoted by *Raw* are the continuous raw sequences, the rows with *Non-FB* tags are the motion features generated with a set of fixed parameters (*wrap, scale*), while the images in the rows with *FB* tags show the motion images trained with feedback process from the *object detector*.

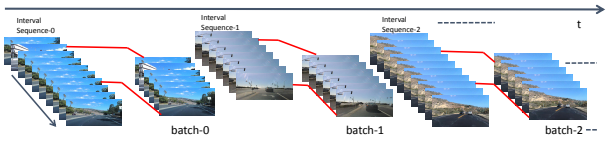


Fig. 10. Batch extraction from each interval sequence. Each batch contains six consecutive frames, with a random start point. The extracted batches will be sent into predictor's *visual encoder* in order.

TABLE II
TESTING MAP RESULTS OF THE OBJECTS.

Method	Dataset	Car	Bus	Person
Faster-RCNN[68]	PASCAL-07+12	75.9	77.5	79.6
Faster-RCNN[68]	COCO+PA07+12	82	81.9	84.1
Proposed method	PA07+12+Udacity	79.8	78.7	69.7

Faster-RCNN methods use region proposal network (RPN) with a complex calculation process, whereas SSD in our method has a simpler architecture. The experimental results show that our trained object detector can efficiently provide reliable detection information for the subsequent parts.

B. Motion Generation with Feedback Evaluation

We extract motion information of continuous Udacity datasets by using TVNet. Comparison of ablation experiments are introduced with two generation processes: with or without feedback from the *object detector* for the related results shown in Fig.9. We randomly extract three sequences from the testing dataset and separately visualize their motion features.

In our task, we pay considerable attention to the spatiotemporal information of the target objects, while all other components are regarded as noises. As shown in Fig.9, the frames in *Raw(a)* contain fewer objects and show a clearer driving road than the other two sequences. The motion generation results without feedback *Non-FB* show substantial specific information, such as car's edge, road lines, and contour of trees, whereas our designed feedback method *FB* generates complete and differential information of objects from other surroundings. For the comparison of the sequences in *Raw(b)* and *Raw(c)*, which are captured in more complex driving conditions and contain more objects, the generated motion images with feedback contain more differential target information than their corresponding *Non-FB* processes.

Moreover, the generated motion features of each object from the feedback contain less noise than the general ones, which indicates that the detection results from the spatial stream contribute to extract much more useful spatial information than the temporal stream and the feedback process plays a vital role in motion generation task. Moreover, the results from the feedback process exhibit a stable performance for continuous sequences, and thus, providing useful temporal information for the final prediction task is not needed. Combining the detection offsets with the motion results, we can obtain velocities of all the objects in the observed image as illustrated in *generated output* part in Section IV.

C. State Prediction Evaluation

Given that the lengths of all interval sequences (illustrated in *Adaptive Prediction Control*) are always changing, we specifically design a method for predicting the future state of a frame following a sequence batch of fixed length. To fairly train all the interval sequences, we continuously extract sequence batches from each interval and send them to *visual encoder*, as shown in Fig.10. We randomly get the starting point of each extracted batch. Hence, the sampling rates of all the interval sequences are equal. Meanwhile, the time-consuming of model training processes are equal to the multiplication of sampling rate, sampling time, and model training consuming of one epoch.

In this work, we design a method for predicting the future states of the *7*-th frame following a sequence batch with six frames, such that the inputs of our modified VIN are continuous batches from all continuous intervals. The shortest interval in our dataset contains nine frames, and thus we set the sampling rate as three to adapt to its maximum sample number. We complete the proposed prediction task based on Pytorch deep learning architecture, by using an 8G GTX1080 GPU. For each learning rate, the training process lasts for 100 epochs. The prediction performance of our method is evaluated by MAPs and standard deviation (SD).

As depicted in Fig.3, the predictor's input is batch sequence stream. Each batch contains six distilled feature frames, and each feature frame is a six-channel spatiotemporal feature combination. To demonstrate the efficiency of our introduced motion generator in the two-stream architecture for future state prediction, we conduct a comparison experiment by feeding three-channel high-level feature images to VIN, and each image only comes from the spatial stream. To fit the input channel of the visual encoder in VIN, we expand the dimensions of single-channel examples, by introducing a convolution layer with padding to generate six-channel feature maps. Moreover, we compare the prediction ability of our modified model with those of typical LSTM network based methods in terms of its perfect long-term dependency learning ability for a continuous data sequence.

LSTM is introduced to function as dynamic predictor in our modified model. The state codes from the *visual encoder* are flattened into two feature streams by the subsequent three fully connected network layers. Then, a two-layer LSTM follows each stream to produce candidate states codes. Multi-Layer perception (MLP) is used to aggregate the candidates to generate the predicted state codes for a future frame. We add the same *state decoder* as that in VIN following MLP, converting the predicted codes to state representation codes. Therefore, the objective function of LSTM-based methods is similar to the one with the IN predictor. We use the same testing dataset with the two predictors. Meanwhile, a comparison experiment with single stream input is also conducted for each method.

Models used to verify the final prediction algorithms are trained with SGD to optimize the overall objective functions mentioned earlier. The initial learning rates are set to 10^{-2}, and decreased to 10^{-3} after one epoch. The related testing results are presented in Tab.III, where the mean average precision of the designed two-stream spatiotemporal model with IN predictor is higher than the spatial only input by 2.92 and surpasses the two-stream LSTM by 13.86. Meanwhile, the two-stream LSTM performs better than the single-stream LSTM with a 1.72 advantage. Therefore, our proposed states prediction model can predict future states, and the two-stream learning architecture helps develop a more efficient prediction process.

TABLE III
TESTING RESULTS OF OUR DEVISED MODEL WITH DIFFERENT PREDICTORS.

Architecture	Predictor	mAP	SD
Spatiotemporal	IN	79.45	5.47
	LSTM	65.59	5.98
Spatial only	IN	76.53	8.50
	LSTM	63.87	9.21

To qualitatively demonstrate the prediction ability of our proposed method, we visualize the predicted states (position) in a future frame by showing the continuous predicted moving trajectories of all the detected objects. Fig.11 illustrates the overall moving trajectories of all the detected objects from a macro perspective. Each predicted layout map at the end of each row contains the predicted trajectory points of all the detected objects. Each point represents the geometric center of the detected bounding corresponds to its detected object. The interval between two neighboring points shows the moving stride between two frames for each detected object. The results show that our proposed method is effective in short-term and long-term state prediction tasks.

(a) shows a short-term video sequence captured at crossing in a related complex city street while the agent is waiting for the green traffic light. The red points depict the dynamics of the car in front of the agent, whereas the green and blue points show the moving trajectory of detected cars in a far place. (b) and (c) show image sequences captured on a high way, where vehicles are traveling at a relatively faster speed than in (a). The results at the end of these two sequences have longer strides compared with the former one in (a). To demonstrate that our adaptive prediction method can make an efficient long-duration prediction, we introduce a long-term sequence and conduct prediction processing on it. The corresponding result on the right of (d) displays the predicted future position information of all the detected objects.

VI. CONCLUSION

In this study, an adaptive visual interaction method for multi-target state prediction in dynamic and changing driving environments is proposed. High-level distilled features are generated through the two-stream architecture. Experimental results on Udacity datasets show that the proposed method can effectively predict the future states of objects under real driving conditions.

Fig. 11. Continuous moving trajectories of the predicted objects. (best viewed in color) The sequence of (a), (b) and (c) show the short-term prediction results of our method, while the sequence (d) is the long-term prediction example.

APPENDIX A
FAILURE CASES OF *motion generator*

The motion generator's failure cases in the two typical datasets, KITTI and NVIDIA, are shown in Fig.12. The failure cases of KITTI originate from a long-term sequence without any dynamic object. The generated motion images show only frame-level relative dynamics, and they come from the special interval sequence removed in the *adaptive prediction control* phase. For NVIDIA, most of the images are captured in scenes with less objects on clear driving roads and with a higher frame rate than KITTI. The higher the frame rate, the less relative differences exist in each frame-pair. Therefore, the two aforementioned reasons result in the failure cases in NVIDIA.

REFERENCES

[1] Agrawal, Pulkit, et al. "Learning to poke by poking: Experiential learning of intuitive physics." Advances in Neural Information Processing Systems. 2016.

[2] A. Lerer, S. Gross, and R. Fergus. "Learning Physical Intuition of Block Towers by Example." International Conference on Machine Learning. 2016: 430-438.

[3] Santoro A, Raposo D, Barrett D G, et al. "A simple neural network module for relational reasoning". Advances in neural information processing systems. 2017: 4967-4976

[4] Battaglia, Peter, et al. "Interaction networks for learning about objects, relations and physics." Advances in neural information processing systems. 2016: 4502-4510.

[5] Watters, Nicholas, et al. "Visual interaction networks: Learning a physics simulator from video." Advances in neural information processing systems. 2017: 4539-4547.

[6] Ma, Zhanyu, et al. "Short utterance based speech language identification in intelligent vehicles with time-scale modifications and deep bottleneck features." IEEE transactions on vehicular technology 68.1 (2019): 121-128.

[7] Chang M B, Ullman T, Torralba A, et al. A Compositional Object-Based Approach to Learning Physical Dynamics[J]. 2017.

[8] Peng, Xiaojiang, and Cordelia Schmid. "Multi-region two-stream R-CNN for action detection." European conference on computer vision. Springer, Cham, 2016: 744-759.

[9] Noh, Hyeonwoo, Seunghoon Hong, and Bohyung Han. "Learning deconvolution network for semantic segmentation." Proceedings of the IEEE international conference on computer vision. 2015.

[10] Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

[11] Liu, Wei, et al. "Ssd: Single shot multibox detector." European conference on computer vision. Springer, Cham, 2016.

[12] Fan, Lijie, et al. "End-to-end learning of motion representation for video understanding." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.

[13] Zach, Christopher, Thomas Pock, and Horst Bischof. "A duality based approach for realtime TV-L 1 optical flow." Joint pattern recognition symposium. Springer, Berlin, Heidelberg, 2007: 214-223.

[14] Grzeszczuk, Radek, Demetri Terzopoulos, and Geoffrey Hinton. NeuroAnimator: fast neural network emula-

Fig. 12. The failure cases of the motion generator on KITTI and NVIDIA datasets. (best viewed in color)

tion and control of physics-based models. University of Toronto, 2000.

[15] Jeong, SoHyeon, et al. "Data-driven fluid simulations using regression forests." ACM Transactions on Graphics (TOG) 34.6 (2015): 199.

[16] Ehrhardt S, Monszpart A, Mitra N J, et al. Learning A Physical Long-term Predictor[J]. 2017.

[17] Zhanyu Ma*, Yuping Lai, W. Bastiaan Kleijn, Liang Wang, and Jun Guo, Variational Bayesian Learning for Dirichlet Process Mixture of Inverted Dirichlet Distributions in Non-Gaussian Image Feature Modeling, IEEE Transactions on Neural Network and Learning Systems (TNNLS), accepted, 2018.

[18] Zhanyu Ma*, Jing-Hao Xue, Arne Leijon, Zheng-Hua Tan, Zhen Yang, and Jun Guo, "Decorrelation of Neutral Vector Variables: Theory and Applications", IEEE Transactions on Neural Networks and Learning Systems (TNNLS), Vol. 29, Issue 1, pp. 129-143, Jan. 2018

[19] Zhang, D., Han, J., Li, C., Wang, J., & Li, X. "Detection of co-salient objects by looking deep and wide." International Journal of Computer Vision 120.2 (2016): 215-232.

[20] Cheng, Gong, Peicheng Zhou, and Junwei Han. "Learning rotation-invariant convolutional neural networks for object detection in VHR optical remote sensing images." IEEE Transactions on Geoscience and Remote Sensing 54.12 (2016): 7405-7415.

[21] Han, Junwei, et al. "Background prior-based salient object detection via deep reconstruction residual." IEEE Transactions on Circuits and Systems for Video Technology 25.8 (2015): 1309-1321.

[22] Han, Junwei, et al. "Representing and retrieving video shots in human-centric brain imaging space." IEEE Transactions on Image Processing 22.7 (2013): 2723-2736.

[23] Mottaghi, Roozbeh, et al. "Newtonian scene understanding: Unfolding the dynamics of objects in static images." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016.

[24] Mottaghi, Roozbeh, et al. "Learning to Predict the Effect of Forces in Images." European Conference on Computer Vision. Springer, Cham, 2016.

[25] Wu, Jiajun, et al. "Physics 101: Learning Physical Object Properties from Unlabeled Videos." BMVC. Vol. 2. No. 6. 2016.

[26] Murphy, Kevin P., Antonio Torralba, and William T. Freeman. "Using the forest to see the trees: A graphical model relating features, objects, and scenes." Advances in neural information processing systems. 2004.

[27] Bhattacharyya, Apratim, et al. "Long-term image boundary extrapolation." arXiv preprint arXiv:1611.08841 (2016).

[28] Fragkiadaki, Katerina, et al. "Learning visual predictive models of physics for playing billiards." arXiv preprint arXiv:1511.07404 (2015).
[29] Sutskever, Ilya, and Geoffrey Hinton. "Temporal-kernel recurrent neural networks." Neural Networks 23.2 (2010): 239-243.
[30] Michalski, Vincent, Roland Memisevic, and Kishore Konda. "Modeling deep temporal dependencies with recurrent grammar cells"".". Advances in neural information processing systems. 2014.
[31] Zhou, Minhua. "Complexity-scalable intra-frame prediction technique." U.S. Patent No. 7,170,937. 30 Jan. 2007.
[32] Igarashi, Katsuji, et al. "Video coding selectable between intra-frame prediction/field-based orthogonal transformation and inter-frame prediction/frame-based orthogonal transformation." U.S. Patent No. 6,324,216. 27 Nov. 2001.
[33] Zhang, Rui, Shankar L. Regunathan, and Kenneth Rose. "Video coding with optimal inter/intra-mode switching for packet loss resilience." IEEE Journal on Selected Areas in Communications 18.6 (2000): 966-976.
[34] Fieguth, Paul, and Demetri Terzopoulos. "Color-based tracking of heads and other mobile objects at video frame rates." Proceedings of IEEE computer society conference on computer vision and pattern recognition. IEEE, 1997.
[35] Nakayama, Ken, and Gerald H. Silverman. "The aperture problemŀI. Perception of nonrigidity and motion direction in translating sinusoidal lines." Vision research 28.6 (1988): 739-746.
[36] Pomerleau, Dean A. "Alvinn: An autonomous land vehicle in a neural network." Advances in neural information processing systems. 1989.
[37] L. Du, W. Jiang, Z. Zhao and F. Su, "Ego-Motion Classification for Driving Vehicle," 2017 IEEE Third International Conference on Multimedia Big Data (BigMM), Laguna Hills, California, USA, 2017, pp. 276-279. doi:10.1109/BigMM.2017.25
[38] Hochreiter, Sepp, and Jrgen Schmidhuber. "Long short-term memory." Neural computation 9.8 (1997): 1735-1780.
[39] Villegas, Ruben, et al. "Learning to generate long-term future via hierarchical prediction." Proceedings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org, 2017.
[40] Xue, Tianfan, et al. "Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks." Advances in Neural Information Processing Systems. 2016.
[41] Bojarski, Mariusz, et al. "End to end learning for self-driving cars." arXiv preprint arXiv:1604.07316 (2016).
[42] Mathieu, Michael, Camille Couprie, and Yann LeCun. "Deep multi-scale video prediction beyond mean square error." arXiv preprint arXiv:1511.05440 (2015).
[43] Kingma, Diederik P., and Max Welling. "Auto-encoding variational bayes." arXiv preprint arXiv:1312.6114 (2013).
[44] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative Adversarial Nets. In NIPS, 2014.
[45] Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." arXiv preprint arXiv:1511.06434 (2015).
[46] Mathieu, Michael, Camille Couprie, and Yann LeCun. "Deep multi-scale video prediction beyond mean square error." arXiv preprint arXiv:1511.05440 (2015).
[47] Jang, Yunseok, Gunhee Kim, and Yale Song. "Video Prediction with Appearance and Motion Conditions." arXiv preprint arXiv:1807.02635 (2018).
[48] Baillargeon, Renee. "A model of physical reasoning in infancy." In C. Rovee-Collier, & LP Lipsitt (Eds.), Advances in infancy research. 1995.
[49] Greig, Dorothy M., Bruce T. Porteous, and Allan H. Seheult. "Exact maximum a posteriori estimation for binary images." Journal of the Royal Statistical Society: Series B (Methodological) 51.2 (1989): 271-279.
[50] Goodale, Melvyn A., and A. David Milner. "Separate visual pathways for perception and action." Trends in neurosciences 15.1 (1992): 20-25.
[51] Simonyan, Karen, and Andrew Zisserman. "Two-stream convolutional networks for action recognition in videos." Advances in neural information processing systems. 2014.
[52] Feichtenhofer, Christoph, Axel Pinz, and Andrew Zisserman. "Convolutional two-stream network fusion for video action recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
[53] Wang, Limin, Yu Qiao, and Xiaoou Tang. "Action recognition with trajectory-pooled deep-convolutional descriptors." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.
[54] Wang, Limin, et al. "Temporal segment networks: Towards good practices for deep action recognition." European conference on computer vision. Springer, Cham, 2016.
[55] Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." Advances in neural information processing systems. 2015.
[56] Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
[57] Salvador, Amaia, et al. "Faster r-cnn features for instance search." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2016.
[58] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems. 2012.
[59] Everingham, Mark, et al. "The pascal visual object classes (voc) challenge." International journal of computer vision 88.2 (2010): 303-338.
[60] Gibson, James J. "The perception of the visual world." (1950).
[61] Gibson, James Jerome. "The senses considered as perceptual systems." (1966).
[62] Horn, B. K. P., and B. G. Schunck. "Determining Optical Flow Artificial Intelligence Vol. 17." (1981): 185-203.
[63] Han, Song, Huizi Mao, and William J. Dally. "Deep compression: Compressing deep neural networks with

pruning, trained quantization and huffman coding." arXiv preprint arXiv:1510.00149 (2015).
[64] Geiger, Andreas, Philip Lenz, and Raquel Urtasun. "Are we ready for autonomous driving? the kitti vision benchmark suite." 2012 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2012.
[65] Ma, Zhanyu, et al. "Variational Bayesian matrix factorization for bounded support data." IEEE transactions on pattern analysis and machine intelligence 37.4 (2015): 876-889.
[66] Udacity self driving car https://github.com/udacity-/self-driving-car.
[67] Lin, Tsung-Yi, et al. "Feature pyramid networks for object detection." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017.
[68] Ren, Shaoqing, et al. "Faster R-CNN: towards real-time object detection with region proposal networks." IEEE Transactions on Pattern Analysis and Machine Intelligence 6.2017: 1137-1149.

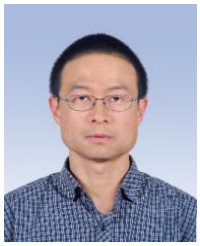
Li Du received the B.S. degree in telecommunication engineering from Inner Mongolia Normal University and the M.S. degree in information and telecommunication engineering from Inner Mongolia University, Hohhot, China, in 2011 and 2015, respectively. She is currently working toward the Ph.D. degree with the school of information and telecommunication, Beijing University of Posts and Telecommunications, Beijing, China. Her research interests include deep learning based automatic driving video prediction, computer vision, object detection.

Zixuan Wang received the B.E. degree in telecommunication engineering from Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2018. Now, he is working toward the M.E. degree in Information and Communication Engineering in BUPT. He is working for the Beijing Key Laboratory of Multimedia and Pattern Recognition, in BUPT. His current research interests include automatic driving, object detection and video prediction via deep learning method.

Leiquan Wang received the Ph.D. degree majoring in Communication and Electrical Systems from BUPT. Now he is a lecturer in college of computer and communication engineering, China University of Petroleum. His current research interests include multimodal fusion, cross modal retrieval and image/video caption.

Zhicheng Zhao is an associate professor of Beijing University of Posts and Telecommunications. He was a visiting scholar at School of Computer Science, Carnegie Mellon University from 2015 to 2016. His research interests are computer vision, image and video semantic understanding and retrieval. He has authored and coauthored more than 60 journal and conference papers.

Fei Su is a female professor in the multimedia communication and pattern recognition lab, school of information and telecommunication, Beijing university of posts and telecommunications. She received the Ph.D. degree majoring in Communication and Electrical Systems from BUPT in 2000. She was a visiting scholar at electrical computer engineering department, Carnegie Mellon University from 2008 to 2009, Her current interests include pattern recognition, image and video processing and biometrics. She has authored and co-authored more than 70 journal and conference papers and some textbooks.



Bojin Zhuang is an senior research fellow of Ping An Technology(Shenzhen) Co., Ltd. His research interests are computer vision, nature language process and optimization theory.

Nikolaos V. Boulgouris is with the Department of Electronic and Computer Engineering of Brunel University London, U.K. From 2004 to 2010 he was an academic member of staff with King's College London, and prior to that he was a researcher with the Department of Electrical and Computer Engineering of the University of Toronto, Canada. Dr. Boulgouris served as Technical Program Chair for the 2018 IEEE International Conference on Image Processing (ICIP). He has served as Senior Area Editor for the IEEE Transactions on Image Processing and as Associate Editor for the IEEE Transactions on Circuits and Systems for Video Technology, from which he received the 2017 Best Associate Editor Award. In the past he served as Associate Editor for the IEEE Transactions on Image Processing and for the IEEE Signal Processing Letters. He was co-editor of the book Biometrics: Theory, Methods, and Applications, which was published by Wiley - IEEE Press, and guest co-editor for two journal special issues. From 2014 to 2019 he served as member of the IEEE Image, Video, and Multidimensional Signal Processing Technical Committee (IVMSP - TC). Dr. Boulgouris is a Senior Member of the IEEE and a Fellow of the Higher Education Academy.