2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

Multimodal Trajectory Predictions for Urban Environments Using Geometric Relationships between a Vehicle and Lanes

Atsushi Kawasaki and Akihito Seki

***Abstract*— Implementation of safe and efficient autonomous driving systems requires accurate prediction of the long-term trajectories of surrounding vehicles. High uncertainty in traffic behavior makes it difficult to predict trajectories in urban environments, which have various road geometries. To overcome this problem, we propose a method called lane-based multimodal prediction network (LAMP-Net), which can handle arbitrary shapes and numbers of traffic lanes and predict both the future trajectory along each lane and the probability of each lane being selected. A vector map is used to define the lane geometry and a novel lane feature is introduced to represent the generalized geometric relationships between the vehicle state and lanes. Our network takes this feature as the input and is trained to be versatile for arbitrarily shaped lanes. Moreover, we introduce a vehicle motion model constraint to our network. Our prediction method combined with the constraint significantly enhances prediction accuracy. We evaluate the prediction performance on two datasets which contain a wide variety of real-world traffic scenarios. Experimental results show that our proposed LAMP-Net outperforms state-of-the-art methods.**

I. INTRODUCTION

Predicting the future trajectories of other traffic participants is a crucial task for autonomous vehicles. An autonomous vehicle needs to drive safely under all traffic situations by taking into account the future trajectories of other traffic in order to avoid collisions.

Trajectory predictions for surrounding vehicles involve high uncertainty because we do not know and cannot uniquely determine which direction they will move in a given scenario, such as when arriving at an intersection and changing lanes. Therefore, the prediction results cannot be expressed as a unimodal distribution. The autonomous vehicle needs to take into account the inherent multimodality of the future motion of surrounding vehicles.

Multimodal prediction in urban environments is much more challenging than prediction in limited situations. Urban traffic environments vary in the number and shape of lanes. Previous works have shown the effectiveness of multimodal prediction in limited situations such as intersections [1-5] and lane changing scenarios [6-8]. However, these divide-and-conquer approaches might not scale to cover prohibitively large variations in real-world traffic scenarios. Autonomous driving in the real-world needs to be able to predict trajectories in arbitrary scenarios, but it is unrealistic to collect and train all possible road geometries.

To overcome this problem, we propose a framework for multimodal trajectory predictions in urban environments that can handle arbitrary numbers and shapes of lanes. Most drivers do not drive freely on the road, but select one of multiple lanes and follow that lane. We focus on the prediction for each lane and train a versatile long short-term memory (LSTM) for arbitrary shapes of lanes. The lane geometry is defined as a vector map, and a novel lane feature (LF) is introduced which expresses the generalized geometric relationships between the vehicle state and lanes. Our network takes this feature as the input and is trained to be versatile for arbitrarily shaped lanes. When our network is assigned to each lane, we obatin both future trajectory along each lane and the probability of each lane being selected, as shown in Fig. 1. We call our framework lane-based multimodal prediction network (LAMP-Net).

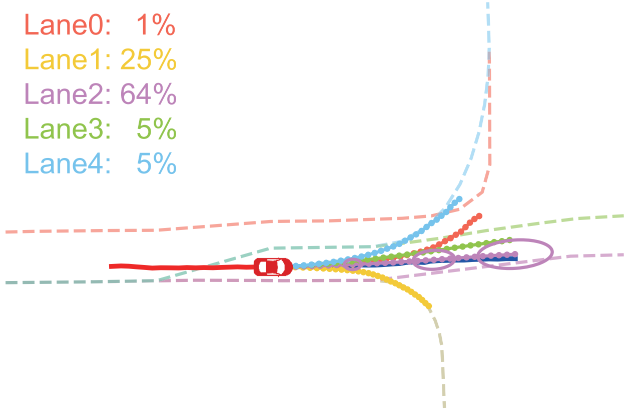


Fig. 1. A result of multimodal trajectory predictions (3 seconds into the future); observed and ground-truth (GT) trajectories are drawn in red and blue, respectively; dashed lines indicate the vector maps of the centerlines, and predicted trajectories are plotted in colors corresponding to the centerlines, with the associated probabilities also shown.

Additionally, we introduce a model-based approach to our network in order to enhance prediction performance. Most learning-based methods predict future trajectory without considering any motion models (e.g., constant acceleration or velocity models). However, most vehicles are constrained by these models. We take into account this constraint in LAMP-Net by tailoring Kalman filter (KF) approach [9]. In [9], LSTM-KF was proposed, which uses LSTM to estimate the parameters for a KF. We propose a tailored LSTM-KF (Tailored-KF) for accurate prediction.

Our method is evaluated on our dataset and a public dataset, which contain a large variety of real-world traffic scenarios. Experiments show that the prediction performance of our model with Tailored-KF is better than that of the model with LSTM-KF. Furthermore, we demonstrate that our

Authors are with the Corporate R&D Center, Toshiba Corporation, Japan {atsushi1.kawasaki|akihito.seki}@toshiba.co.jp

978-1-7281-7395-5/20/$31.00 ©2020 IEEE

method outperforms other state-of-the-art methods.

The contributions of this paper are as follows. 1) We propose a framework for multimodal trajectory predictions in urban environments, which can handle any numbers and shapes of lanes. 2) Inspired by the idea of KF and LSTM, we introduce a vehicle motion model constraint into our neural network, in order to enhance prediction performance. 3) Experiments on both our dataset and a public dataset show that our method outperforms other state-of-the-art methods.

II. RELATED WORKS

Model-based and Learning-based Approaches: The problem of predicting the trajectories of traffic participants has been actively studied in the field of robotics [10]. The related works can be classified into two approaches: model-based and learning-based. In model-based approaches, many researchers have used the constant velocity model, the intelligent driver model, and KF. KF, in particular, has been widely used to predict trajectories with uncertainties. In [11-13], the future velocity and yaw rate required for lane changing and curving were modeled and incorporated into a KF. However, this approach has the disadvantage that parameter tuning is required for each situation.

Machine learning techniques overcome this problem. Classical learning-based methods have been used, such as the Gaussian process model [14], hidden Markov model [15], and Bayesian network [16]. In recent work, recurrent neural networks (RNN), LSTM, and convolutional neural networks (CNNs) have been proposed. In [17], Kim et al. fed sequential vehicle coordinates into an LSTM and produced probabilistic information about the future location. In [8, 18, 19], LSTM-based trajectory prediction methods were proposed that take into account social interactions between different targets. It is well known that deep-learning-based methods perform better than model-based methods.

In other fields, hybrid methods of learning and model-based approaches have also been proposed. In [20], the model parameters of heat diffusion were estimated using a neural network. LSTM-KF [9] can estimate the parameters of KF for human pose estimation. Hybrid methods have widely shown more stable and accurate results. Following the above works, we aim to enhance prediction performance by combining learning- and model-based approaches.

Using Map Information: Map information is expected to facilitate confident prediction of future trajectories in complicated scenarios. We can classify the related prediction algorithms using maps into two approaches: image map-based and vector map-based. In the first approach, road environments are embedded in bird's-eye-view images. In [1, 21], the input images consisted of road surface, lane center lines, and bounding boxes of traffic participants. In [22, 23], a dynamic occupancy grid map was used as the input for a CNN or Convolutional-LSTM [24]. DESIRE [3] and Scene-LSTM [25] utilize real scene images with semantic labels captured by drones or surveillance cameras. However, these image-map based methods have the disadvantage that the prediction accuracy depends on image resolution.

In the second approach, a vector map is used directly. This approach does not suffer from discretization errors. In [4], Hu et al. proposed a multimodal prediction method using dynamic time warping (DTW) distances [26]. They used the DTW distances between observed trajectories and vector maps for lane selection in roundabouts and predicted the trajectories along the lane. In [27], vehicle positions in a Cartesian coordinate system were transformed to the distance along and offset from the centerline. Their method could predict the trajectory along the centerline, but the curvature of the lane could not be considered because the motion of the target vehicle, such as going straight or turning, cannot be discriminated in the transformed coordinate system.

In the present study, we also directly use vector maps which represent the lane centerline. To predict trajectories along a lane without ignoring its geometry, we propose a novel LF that represents the geometric relationships between the vehicle state and the lanes.

Multi-modal Prediction: Several papers addressing the problem of modeling multimodality have been published. The mixture density network (MDN) [28] solves multimodal regression tasks by learning the parameters of a Gaussian mixture model. Zyner et al. [2] proposed a method for predicting multimodal trajectories at roundabouts by combining MDN with RNN. However, MDN is difficult to be trained due to numerical instability when operating in high dimensional spaces. In [3, 4, 27], multimodal predictions were generated by random sampling from multivariate normal distributions using a variational auto encoder (VAE) [29] or conditional variational auto encoder (CVAE) [30]. These methods require unnecessary repeated forward passes to obtain sufficient prediction candidates due to the use of random sampling. In [7, 8], a single network was proposed which produces different outputs for each maneuver. However, these networks can only predict a set of fixed maneuvers (e.g., keep lane, left and right lane changes, and braking).

To solve these problems, we train a versatile LSTM that can predict the trajectory along an arbitrarily shaped lane and the probability of this lane being selected by the target vehicle. By assigning this network to each lane, we get multiple trajectories and associated probabilities.

III. PROPOSED METHOD

In this section, we first introduce the problem setting and notation. The details of LAMP-Net including the multimodal prediction and combination of learning- and model-based approaches are then discussed.

A. Problem Setting

We assume access to a perception module that maps sensor data (LiDAR, radar, or camera) to 3D tracked vehicles. At each timestep t, we have the measured vehicle states $\mathbf{x}_t$, represented as follows:

$$\mathbf{x}_t = \begin{bmatrix} x_t & y_t & \theta_t & v_t & \gamma_t & a_t & \dot{\gamma}_t \end{bmatrix}^T,$$

where x, y, θ are the 2D position and angle in a Cartesian coordinate system, v is the total velocity, γ is the yaw rate,

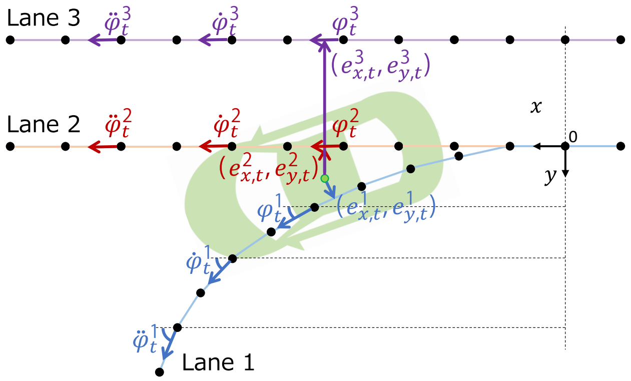


Fig. 2. Example of lane feature (LF) for lanes 1, 2, and 3 at a certain timestep. Black points show the vector maps of the centerlines. Each LF consists of an error vector $(e_{x,t}^m, e_{y,t}^m)$ and the angles of the direction vector of the vector maps $(\varphi_t^m, \dot{\varphi}_t^m, \ddot{\varphi}_t^m)$

a is the total acceleration, and $\dot{\gamma}$ is the yaw acceleration. The inputs to our model are sequential data for the above states at timesteps $t = \{1, \ldots, T_{obs}\}$. The outputs are also sequential states at $t = \{T_{obs+1}, \ldots, T_{pred}\}$. The time gap between consecutive time steps is constant (0.1 s). The coordinate origin is set at $(x_{T_{obs}}, y_{T_{obs}})$, where T_{obs} is the last observed timestep. Our perception module also gives the state-estimation uncertainty as a covariance matrix. We use this covariance matrix at $t = T_{obs}$ for the initial input to the Tailored-KF module and represent it as $\mathbf{P}_{T_{obs}}$. Additionally, we use the vector map of the lane centerlines. The interval between nodes in the vector map is equal to 1 meter.

B. Lane Feature

To handle arbitrary lane shapes, we introduce a novel LF which represents the generalized geometric relationships between the prediction target and lanes. LF is calculated for each lane m at each timestep t. LF is defined as $\mathbf{LF}_t^m = \left[\begin{array}{ccccc} e_{x,t}^m & e_{y,t}^m & \varphi_t^m & \dot{\varphi}_t^m & \ddot{\varphi}_t^m \end{array}\right]$. Fig. 2 shows an example of LF. In the definition, $e_{x,t}^m$ and $e_{y,t}^m$ are the elements of the error vector between the vehicle position and the lane. We define the nodes in the target lane as $\{p_0, \ldots, p_N\}$ and define the closest node to the vehicle as p_n. φ_t^m is the angle of the direction vector from p_n to p_{n+1}; $\dot{\varphi}_t^m$ is from p_{n+2} to p_{n+3}, and $\ddot{\varphi}_t^m$ is from p_{n+4} to p_{n+5}.

Our method can consider the lane geometries ahead of a vehicle position from $\dot{\varphi}_t^m$ and $\ddot{\varphi}_t^m$. For example, vehicles usually slow down before turning. Our method can learn this characteristic by adding these elements.

C. Network Architecture

The detail of LAMP-Net is shown in Fig. 3 (a). We use an LSTM-based encoder-decoder framework [31]. One encoder module consists of an LSTM and embedding functions (LSTM1, ϕ_1, and ϕ_2 in Fig. 3 (a)). We assign a weight-shared encoder module to each lane. The sequential state vector $\mathbf{x}_t$ and lane feature $\mathbf{LF}_t^m$ are concatenated and used as the input.

To estimate the probabilities of lane selection, we use each final encoder LSTM state which can be expected to encode how the vehicle follows the lane. Each LSTM state passes through $\phi_2(\cdot)$ and is converted to a one-dimensional vector in order to remove the order dependency of the lanes. These one-dimensional vectors are concatenated and then fed through the softmax function to produce the lane selection probabilities $\{p_0, \ldots, p_M\}$.

Each decoder module consists of an LSTM, embedding functions, and a Tailored-KF module. We also assign a weight-shared decoder module to each lane. Each hidden state of LSTM2 in Fig. 3 is updated from the hidden state of the LSTM1 corresponding to the lane. The outputs of the decoder at each timestep and for each lane are the predicted state $\mathbf{x}_{t+1}^m$ and the covariance matrix $\mathbf{P}_{t+1}^m$, produced through the Tailored-KF module. $\mathbf{LF}_{t+1}^m$ is calculated by using $\mathbf{x}_{t+1}^m$. $\mathbf{LF}_{t+1}^m$ and $\mathbf{x}_{t+1}^m$ are concatenated and taken as the input to the decoder, recurrently.

D. Tailored-KF

We incorporate a motion model, such as the constant acceleration model or the decay acceleration model, into our neural network by using the tailored KF-based approach (LSTM-KF) [9]. In LSTM-KF [9], the parameters of KF, such as the state transition model, the Jacobian, and noise, are estimated by using LSTMs. This method can estimate all parameters, but the expressive capabilities are weak because the Jacobian and noise are characterized by diagonal matrices. The nondiagonal elements are important factors for accurately expressing the distributions of the uncertainties. For vehicle trajectory prediction, it is appropriate to use the motion model for the transition model. The nondiagonal elements of the Jacobian matrix can be calculated from the formulated transition model. Additionally, we calculate the nondiagonal elements of the noise matrix of the transition model by adding the constraint that all noise originates from acceleration and yaw acceleration.

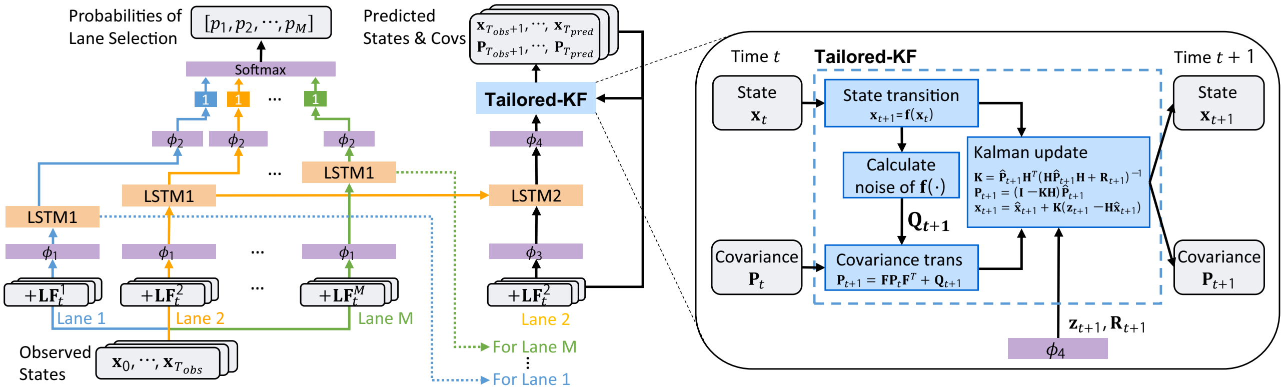
Fig. 3 (b) shows the architecture of the Tailored-KF module. For simplicity of notation, in this subsection, we omit the subscript m (lane label). The transition model is defined as follows:

$$\begin{aligned} \bar{\mathbf{x}}_{t+1} &= \mathbf{f}(\mathbf{x}_t) + \mathbf{w}_t \\ &= \left[\begin{array}{ccccccc} f_1 & f_2 & f_3 & f_4 & f_5 & f_6 & f_7 \end{array}\right] + \mathbf{w}_t, \end{aligned} \tag{1}$$

where

$$\begin{aligned} f_1 &= x_t + (v_t \cos\theta_t)\Delta t + (a_t \cos\theta_t + \gamma_t v_t \sin\theta_t)\Delta t^2/2 \\ f_2 &= y_t + (v_t \sin\theta_t)\Delta t + (a_t \sin\theta_t + \gamma_t v_t \cos\theta_t)\Delta t^2/2 \\ f_3 &= \theta_t + (\gamma_t)\Delta t + (\dot{\gamma}_t)\Delta t^2/2 \\ f_4 &= v_t + (a_t)\Delta t + (-k_a a_t)\Delta t^2/2 \\ f_5 &= \gamma_t + (\dot{\gamma}_p)\Delta t + (-k_{\dot{\gamma}}\dot{\gamma}_p)\Delta t^2/2 \\ f_6 &= a_t + (-k_a a_t)\Delta t + (k_a^2 a_t)\Delta t^2/2 \\ f_7 &= \dot{\gamma}_t + (-k_{\dot{\gamma}}\dot{\gamma}_t)\Delta t + (k_{\dot{\gamma}}{}^2\dot{\gamma}_t)\Delta t^2/2 \\ \mathbf{w}_t &\sim N(\begin{array}{cc} 0 & \mathbf{Q}_t \end{array}). \end{aligned} \tag{2}$$

The acceleration and yaw acceleration are assumed to decay to zero at a rate characterized by the time constants k_a and $k_{\dot{\gamma}}$. In our implementation, k_a and $k_{\dot{\gamma}}$ are set to 0.1. Suppose that the noise of $\mathbf{f}(\cdot)$ has a normal distribution with zero mean


Probabilities of Lane Selection
$[p_1, p_2, \cdots, p_M]$
Softmax
1
1
1
ϕ_2
LSTM1
ϕ_1
$+\mathrm{LF}_t^1$
$+\mathrm{LF}_t^2$
$+\mathrm{LF}_t^M$
Lane 1
Lane 2
Lane M
Observed States
$\mathbf{x}_0, \cdots, \mathbf{x}_{T_{obs}}$
Predicted States & Covs
$\mathbf{x}_{T_{obs}+1}, \cdots, \mathbf{x}_{T_{pred}}$
$\mathbf{P}_{T_{obs}+1}, \cdots, \mathbf{P}_{T_{pred}}$
Tailored-KF
ϕ_4
LSTM2
ϕ_3
$+\mathrm{LF}_t^2$
Lane 2
For Lane M
For Lane 1
Time t
Tailored-KF
Time $t+1$
State $\mathbf{x}_t$
State transition $\mathbf{x}_{t+1}=\mathbf{f}(\mathbf{x}_t)$
Calculate noise of $\mathbf{f}(\cdot)$
$\mathbf{Q}_{t+1}$
Kalman update
$\mathbf{K} = \hat{\mathbf{P}}_{t+1}\mathbf{H}^T(\mathbf{H}\hat{\mathbf{P}}_{t+1}\mathbf{H} + \mathbf{R}_{t+1})^{-1}$
$\mathbf{P}_{t+1} = (\mathbf{I} - \mathbf{KH})\hat{\mathbf{P}}_{t+1}$
$\mathbf{x}_{t+1} = \hat{\mathbf{x}}_{t+1} + \mathbf{K}(\mathbf{z}_{t+1} - \mathbf{H}\hat{\mathbf{x}}_{t+1})$
Covariance $\mathbf{P}_t$
Covariance trans $\mathbf{P}_{t+1} = \mathbf{F}\mathbf{P}_t\mathbf{F}^T + \mathbf{Q}_{t+1}$
State $\mathbf{x}_{t+1}$
Covariance $\mathbf{P}_{t+1}$
$\mathbf{z}_{t+1}, \mathbf{R}_{t+1}$
ϕ_4

(a) Network Architecture

(b)The detail of Tailored-KF

Fig. 3. Overview of the developed network architecture, LAMP-Net.

and covariance matrix $\mathbf{Q}_t$. We dynamically estimate $\mathbf{Q}_t$ based on the constraint that all noise originates from acceleration and yaw acceleration as follows:

$$\begin{aligned} h_{t+1} &= \mathrm{LSTM\_Q}(h_t, \mathbf{x_t}) \\ \left[\begin{array}{cc} \sigma^a_{t+1} & \sigma^{\dot{\gamma}}_{t+1} \end{array}\right] &= \exp(\phi(h_{t+1})) \\ \mathbf{F}_t &= \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}_t} \\ \mathbf{B} &= \left[\begin{array}{ccccccc} 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right]^T \\ \mathbf{Q}_{t+1} &= (\mathbf{F}_{t+1}\mathbf{B})diag(\left[\begin{array}{cc} \sigma^a_{t+1} & \sigma^{\dot{\gamma}}_{t+1} \end{array}\right])(\mathbf{F}_{t+1}\mathbf{B})^T, \end{aligned}$$

where $\phi(\cdot)$ is an embedding function, and h_t is the hidden state and the output of LSTM_Q$(\cdot)$. The variance of the acceleration σ^a_t and that of the yaw acceleration $\sigma^{\dot{\gamma}}_t$ are estimated by using LSTM and exponential function. Other elements of $\mathbf{Q}_t$ are estimated from the Jacobian matrix $\mathbf{F}_t$.

The observation model of Tailored-KF is similar to KF-based models in related works [11, 12]. The model consists of the future desired velocity and desired yaw rate, which act as virtual measurements as follows:

$$\begin{aligned} \mathbf{z}_{t+1} &= \mathbf{H} \cdot \mathbf{x}_{t+1} + \mathbf{v}_{t+1} \\ &= \left[\begin{array}{ccccccc} 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right] \mathbf{x}_{t+1} + \mathbf{v}_{t+1} \\ &= \left[\begin{array}{cc} v^{des}_{t+1} & \gamma^{des}_{t+1} \end{array}\right]^T \\ \mathbf{v}_{t+1} &\sim N(\begin{array}{cc} 0 & \mathbf{R}_{t+1} \end{array}) \\ \mathbf{R}_{t+1} &= diag(\left[\begin{array}{cc} \exp(\bar{\sigma}^v_{t+1}) & \exp(\bar{\sigma}^{\gamma}_{t+1}) \end{array}\right]) \\ &= diag(\left[\begin{array}{cc} \sigma^v_{t+1} & \sigma^{\gamma}_{t+1} \end{array}\right]), \end{aligned} \tag{3}$$

where v^{des} and γ^{des} are the future desired velocity and yaw rate, and $\mathbf{R}$ is their covariance matrix. These values are calculated from the output of $\phi_4(\cdot)$, $\left[\begin{array}{cccc} v^{des}_{t+1} & \gamma^{des}_{t+1} & \bar{\sigma}^v_{t+1} & \bar{\sigma}^{\gamma}_{t+1} \end{array}\right]$.

The Kalman update is the same as in KF. After calculating the Kalman gain, the refined state $\mathbf{x}_{t+1}$ and the covariance $\mathbf{P}_{t+1}$ are obtained. Fig. 3 (b) shows the detailed calculation.

E. Training

Our loss function consists of three elements: L^{prob} is the lane selection loss, L^{traj} is the predicted trajectory loss, and L^{obs} is the loss of the observation model of Tailored-KF. L^{prob} is defined as the cross-entropy losses of the predicted and GT lane labels. In the definition of L^{traj} and L^{obs}, we use only the decoder output on the GT lane label, in contrast with MDN. To train a versatile decoder which can predict the trajectory along the target lane, the outputs on incorrect lane labels are not necessary. This idea is based on the characteristic that most vehicles follow the centerline. L^{traj} is defined as:

$$L^{traj} = \sum_{m=1}^{M} \left(I_{m=m^*} \sum_{t=T_{obs}+1}^{T_{pred}} \mathfrak{L}^{traj}_{m,t} \right)$$

$$\mathfrak{L}^{traj}_{m,t} = -\log P_2(x^*_t, y^*_t \mid (x^m_t, y^m_t), [\mathbf{P}^m_t]_{xy}) \cdot P_1(\theta^*_t \mid \theta^m_t, [\mathbf{P}^m_t]_\theta),$$

where $*$ indicates GT and M is the maximum number of lanes, I_m is a binary indicator function that is equal to 1 if lane label m is GT lane label m^*, and $\mathfrak{L}$ is the negative log-likelihood loss [32] for the predicted states (position and angle). We assume the predicted position meets a bivariate Gaussian distribution parameterized by $N((x^m_t, y^m_t), [\mathbf{P}^m_t]_{xy})$, where $[\mathbf{P}^m_t]_{xy}$ is 2×2 submatrix of $\mathbf{P}^m_t$ characterizing the position and $P_2(\cdot)$ denotes a density function of a bivariate Gaussian distribution. Similarly, $P_1(\cdot)$ is the density function of a univariate Gaussian distribution parameterized by $N(\theta^m_t, [\mathbf{P}^m_t]_\theta)$, where $[\mathbf{P}^m_t]_\theta$ is a diagonal elements of $\mathbf{P}^m_t$ regarding to θ. In L_{obs}, the desired velocity and the desired yaw rate are also optimized as follows:

$$L^{obs} = \sum_{m=1}^{M} \left(I_{m=m^*} \sum_{t=T_{obs}+1}^{T_{pred}} \mathfrak{L}^{obs}_{m,t} \right)$$

$$\mathfrak{L}^{obs}_{m,t} = -\log P_1(v^*_t \mid v^{des,m}_t, \sigma^{v,m}_t) \cdot P_1(\gamma^*_t \mid \gamma^{des,m}_t, \sigma^{\gamma,m}_t).$$

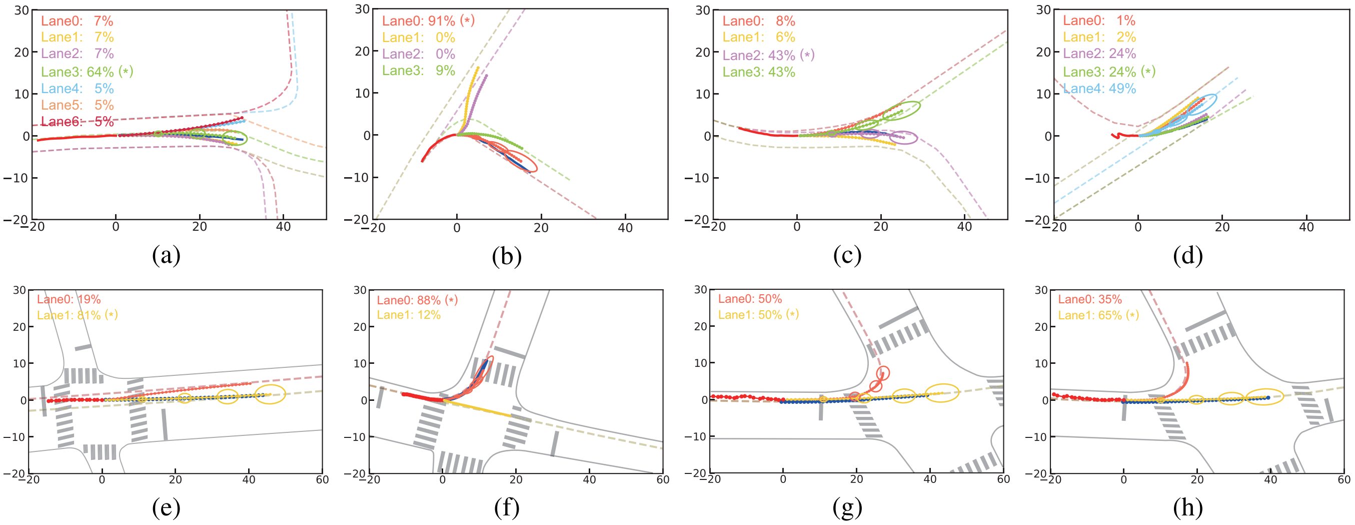
This loss has the benefit of stabilizing the training and enhancing the effect of Tailored-KF.

We train the model using the Adam solver [33] with a learning rate 0.0005. All LSTMs have 16-dimensional states. We also use all embedding layers with 16 dimensions, with tanh activation. The model is implemented using Tensorflow.

IV. EXPERIMENTS

A. Datasets and Metrics

Datasets: We evaluate our method on our dataset and a public dataset that contains a large variety of real-world



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 4. Visualization of the prediction results, as in Fig. 1. The first row shows the results for the validation dataset in Argoverse [27]. The results in the second row are for our dataset. Ellipses indicate standard deviation based on the output covariance matrix and are drawn for only the top trajectory. An asterisk $(*)$ next to a lane selection probability indicates the GT lane label.

		Predicted lane error [m]				**Oracle lane error [m]**			
		Our dataset		Argoverse [27]		Our dataset		Argoverse [27]	
Methods	Multimodal	ADE	FDE	ADE	FDE	ADE	FDE	ADE	FDE
Constant velocity	–	–	–	–	–	2.83	6.28	2.97	6.36
Houenou et al. [34] + kawasaki et.al. [12]	✓	2.15	4.71	2.45	5.03	2.01	4.39	2.20	4.86
Vanilla LSTM	–	–	–	–	–	2.24	4.91	2.03	4.57
Chang et al. [27]	–	–	–	–	–	2.12	3.72	1.91	4.05
Hu et al. [4]	✓	2.21	4.27	2.02	4.56	1.99	3.52	1.76	3.94
LAMP-Net (direct regression)	✓	2.33	4.37	1.94	4.46	2.05	3.60	1.77	3.89
LAMP-Net (with LSTM-KF [9])	✓	1.97	4.07	1.89	4.26	1.78	3.48	1.74	3.80
LAMP-Net (with Tailored-KF)	✓	**1.79**	**3.83**	**1.82**	**4.04**	**1.61**	**3.28**	**1.71**	**3.69**

TABLE I

QUANTITATIVE RESULTS OF ALL METHODS ON TWO DATASETS.

traffic scenarios. Our dataset contains vehicle tracking results and vector maps of the lane centerlines. The states and variances of the tracked vehicles are obtained from multiple sensor modules, which can produce top-down projected 2D bounding boxes.The initial covariance matrix $\mathbf{P}_{T_{obs}}$ can be obtained from these modules. We collected 66 different intersection scenarios from Tokyo, Japan, and trained and tested the model using 55 and 11 intersections, respectively. The number of sequences is 16,896 for training and 3,105 for testing. The length of input sequence is 20 frames (2.0 s), and the prediction time horizon is 40 frames (4.0 s).

Additionally, we used Argoverse [27], which is a public dataset for the vehicle trajectory prediction task. Argoverse consists of 208,272 training sequences and 40,127 validation sequences, with vector maps of the lane centerlines. The input and prediction length are 20 and 30 frames (2.0 and 3.0 s). In the dataset, the vehicle states have only positions. We calculate other states, such as angle and velocity from the time series difference, and set $\mathbf{P}_{T_{obs}}$ to a fixed value of $diag(3.2e^{-1}, 6.4e^{-2}, 2.7e^{-3}, 1.4e^{-1}, 2.5e^{-4}, 1.8e^{-1}, 1.3e^{-5})$.

Metrics: As in related works [18, 25, 27], our method is evaluated using two metrics: average displacement error (ADE) and final displacement error (FDE). ADE is the mean error over all locations of predicted trajectories and GT. FDE is the distance between the final predicted location and the final GT location. Moreover, we compare them in terms of the predicted lane error and the oracle lane error. One is defined as the error in predicted trajectory for the lane label with the highest probability of selection. The other is the error in the predicted trajectory for the GT lane label.

Baselines: We compare the following baselines:

- ConstantVelocity: Using the most recent velocity(v_x, v_y).
- Houenou et al.[34] + Kawasaki et al.[12]: Model-based method. Lane selection probabilities are estimated by the distances from a vehicle to the road boundaries [34]. Trajectories are predicted by modeling a future velocity by fitting observed velocities to a cubic equation [12].
- Vanilla LSTM: LSTM encoder-decoder model that directly predicts the position from past trajectories only.
- Chang et al.[27]: Transforms vehicle positions in a Cartesian coordinate system to distance along the lane centerline and offset from it, and predicts the trajectory by using Enc-Dec LSTM. This method relies on the assumption that the system knows the GT lane label.

- Hu et al.[4]: Estimates the lane-selection probabilities by using DTW distances[26], and predicts the trajectory by a CVAE-based method.
- LAMP-Net: The proposed method. We prepare three models: a direct regression model through ϕ_4, the LSTM-KF [9] model, and the Tailored-KF model.

B. Qualitative Results

Fig. 4 shows examples of the prediction results from our best-performing model. The colors of the centerlines and the predicted trajectories correspond to each other. We can see that each predicted trajectory follows each centerline. These results show that our network is versatile with respect to lane shape and that our LF and our training method works well. Moreover, it can be seen that prediction is possible for scenarios with an arbitrary number of lanes. This is because the network architecture is adaptive to the number of lanes.

In Fig. 4 (a,b,e,f,h), the lane with the highest probability corresponds to the GT lane. In Fig. 4 (g), it is not a coincidence that the two probabilities are equal. These probabilities depend on the input to the encoder. In (g), the vehicle has not arrive at the lane branch, and each lane has the same LF at the observed times, so the probabilities are also equal. Fig. 4 (h) is the scene 10 frames after (g). In (h), the vehicle has not arrived at the lane branch, but the probabilities are different because the LFs are not equal. As mentioned in section III.B, our LF includes the lane angles ahead of vehicle position. Our model can decide which lane a vehicle will select when the vehicle approaches the branch within a certain distance. This decision by our model is similar to human judgment.

Fig. 4 (d) is a failure case, in which the lane with the highest probability does not correspond to the GT lane label. The judgment in this scenario is hard even for humans. We need to use the lane with the highest probability along with the lanes where the probability exceeds a certain threshold.

C. Quantitative Results

The ADE and FDE values were calculated for all methods, and the results are shown in Table I. LAMP-Net (with Tailored-KF) outperforms the other methods on all metrics for all datasets, regardless of whether the GT lane label is known. The methods in the first 2 rows are model-based approaches, those in rows 3 to 7 are learning-based methods, and the final 2 rows are combinations of model- and learning-based approaches. The overall trend shows that the learning-based methods perform better than model-based methods, and the combination methods perform best of all.

We now focus on the oracle lane error. This metric shows the pure performance of trajectory generation. When we compare three learning-based methods, the errors of Chang et al. [27] are the highest. In that method, vehicle positions are transformed from a Cartesian coordinate system to a centerline-based coordinate system. Their method can predict the vehicle trajectory along the centerline but cannot consider the lane shape from the transformed coordinates. Our method and Hu et al.'s method [4] have the advantage of being able to predict trajectories while keeping the lane shape.

Methods	Our dataset	Argoverse [27]
Houenou et al.[34]	0.862	0.751
Chang et al.[4]	0.821	0.767
Ours	**0.927**	0.851
Ours w/o decoder	0.925	**0.854**

TABLE II

COMPARISON OF LANE SELECTION ACCURACY

We next focus on the three methods at the bottom of the table. The performance is slightly improved by adding the original LSTM-KF to the direct regression model. By adding Tailored-KF instead, performance is further improved. Our Tailored-KF fixes the transition function but can estimate nondiagonal elements of the Jacobian and the noise matrix. The advantage of being able to estimate nondiagonal elements improves prediction performance.

The metric for the predicted lane error depends on the lane selection probabilities. Table II shows the accuracy of lane selection. The maximum numbers of lanes in our dataset and Argoverse are 3 and 15, respectively. The lane selection methods in the first 2 rows are model-based approaches. The accuracies of our learning-based methods are higher than those of other works.

The last 2 rows in Table II show an ablation study of our method. We compare the accuracies between a single-task model (ours without decoder in Table II) for lane selection and a multi-task model (ours in Table II) for trajectory prediction and lane selection. There is not much difference between the accuracies obtained from the two models, which shows that the two tasks can coexist harmoniously.

V. CONCLUSION

In this paper, we presented a framework for multimodal trajectory predictions of surrounding vehicles in urban environments. Our framework can handle any number and shape of lanes. We utilized a vector map for lane information and introduced a novel LF that represents the generalized geometric relationships between the vehicle state and the lanes. This feature makes our network versatile for any shape of lanes. Our versatile network can predict both the future trajectory along each lane and the probability of each lane being selected. Moreover, our trajectory prediction method introduces a vehicle motion model constraint into our neural network. We customized LSTM-KF to Tailored-KF by adding this constraint. Our Tailored-KF overcomes the problem of low expressive capability for estimated parameters. Experiments on both our dataset and a public dataset show that our method outperforms other state-of-the-art methods.

ACKNOWLEDGEMENT

We would like to thank Atsuto Maki and Joonatan Mänttäri at KTH Royal Institute of Technology, for collaboration on the early stage of this work. We also thank Tatsuo Kozakaya and Susumu Kubota, at Toshiba Corporation, for proofreading the manuscript.

References

[1] H. Cui, V. Radosavljevic, F. C. Chou, T. H. Linm, T. Nguyen, T. K. Huang, J. Schneider, and N. Djuric, Multimodal trajectory predictions for autonomous driving using deep convolutional networks, arXiv preprint arXiv:1809.10732, 2018.

[2] A. Zyner, S. Worrall, and E. Nebot, Naturalistic driver intention and path prediction using recurrent neural networks, IEEE Transactions on Intelligent Transportation Systems, 2019.

[3] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. S. Torr, and M. Chandraker. Desire: Distant future prediction in dynamic scenes with interacting agents, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 336-345, 2017.

[4] Y. Hu, W. Zhan, L. Sun, and M. Tomizuka, Multi-modal probabilistic prediction of interactive behavior via an interpretable model, IEEE Conference on Intelligent Vehicles Symposium (IV), pp. 557-563, 2019.

[5] O. Makansi, E. Ilg, O. Cicek, and T. Brox, Overcoming limitations of mixture density networks: A sampling and fitting framework for multimodal future prediction, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7144-7153, 2019.

[6] Y. Hu, W. Zhan, M. Tomizuka, A framework for probabilistic generic traffic scene prediction. IEEE Conference on Intelligent Transportation Systems (ITSC), pp. 2790-2796, 2018.

[7] N. Deo and M. M. Trivedi, Multi-modal trajectory prediction of surrounding vehicles with maneuver based lstms, IEEE Conference on Intelligent Vehicles Symposium (IV), pp. 1179-1184, 2018.

[8] N. Deo and M. M. Trivedi, Convolutional social pooling for vehicle trajectory prediction, IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR workshops), pp. 1468-1476, 2018.

[9] H. Coskun, F. Achilles, R. DiPietro, N. Navab, and F. Tombari, Long short-term memory kalman filters: Recurrent neural estimators for pose regularization, IEEE International Conference on Computer Vision (ICCV), pp. 5524-5532, 2017.

[10] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, and K. O. Arras, Human Motion Trajectory Prediction: A Survey, arXiv preprint arXiv:1905.06113, 2019.

[11] B. Kim, and K. Yi, Probabilistic and holistic prediction of vehicle states using sensor fusion for application to integrated vehicle safety systems, IEEE Transactions on Intelligent Transportation Systems, vol. 15, pp. 2178-2190, 2014.

[12] A. Kawasaki and T. Tasaki, Trajectory prediction of turning vehicles based on intersection geometry and observed velocities, IEEE Conference on Intelligent Vehicles Symposium (IV), pp. 511-516, 2018.

[13] D. Petrich, T. Dang, D. Kasper, G. Breuel, and C. Stiller, Map-based long term motion prediction for vehicles in traffic environments, IEEE Conference on Intelligent Transportation Systems (ITSC), pp. 2166-2172, 2013.

[14] Q. Tran and J. Firl, Modelling of traffic situations at urban intersections with probabilistic non-parametric regression, IEEE Conference on Intelligent Vehicles Symposium (IV), pp. 334-339, 2013.

[15] M. Brand, N. Oliver, and A. Pentland, Coupled hidden Markov models for complex action recognition, IEEE Conference on Computer Vision and Pattern Recognition(CVPR), pp. 994-999, 1997.

[16] T. Gindele, S. Brechtel, and R. Dillmann, R, Learning driver behavior models from traffic observations for decision making and planning, IEEE Intelligent Transportation Systems Magazine, vol, 7, pp. 69-79, 2015.

[17] B. Kim, C. M. Kang, S. Lee, H. Chae, J. Kim, C. C. Chung, and J. W. Choi, Probabilistic Vehicle Trajectory Prediction over Occupancy Grid Map via Recurrent Neural Network, IEEE Conference on Intelligent Transportation Systems (ITSC), 2017.

[18] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, Social lstm: Human trajectory prediction in crowded spaces, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 961-971, 2016.

[19] Y. Ma, X. Zhu, S. Zhang, R. Yang, W. Wang, and D. Manocha, Trafficpredict: Trajectory prediction for heterogeneous traffic-agents, Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 6120-6127, 2019.

[20] Y. Long, X. She, and S. Mukhopadhyay, HybridNet: integrating model-based and data-driven learning to predict evolution of dynamical systems, arXiv preprint arXiv:1806.07439 ,2018.

[21] J. Hong, B. Sapp, and J. Philbin, Rules of the road: Predicting driving behavior with a convolutional model of semantic interactions, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 8454-8462, 2019.

[22] M. Schreiber, S. Hoermann, and K. Dietmayer, Long-term occupancy grid prediction using recurrent neural networks, arXiv preprint arXiv:1809.03782, 2018.

[23] S. Hoermann, M. Bach, and Klaus Dietmayer, Dynamic occupancy grid prediction for urban autonomous driving: a deep learning approach with fully automatic labeling, arXiv preprint arXiv:1705.08781, 2017.

[24] X. Shi, Z. Chen, H. Wang, D. Y. Yeung, W. K. Wong, and W. C. Woo, Convolutional LSTM network: A machine learning approach for precipitation nowcasting, Advances in Neural Information Processing Systems (NIPS), pp. 802-810, 2015.

[25] H. Manh and G. Alaghband, Scene-lstm: A model for human trajectory prediction, arXiv preprint arXiv:1808.04018, 2018.

[26] E.J. Keogh and M. J. Pazzani, Scaling up dynamic time warping for datamining applications, The sixth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 285-289, 2000.

[27] M. F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan, and J. Hays, Argoverse: 3D tracking and forecasting with rich maps, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 8748-8757, 2019.

[28] C. M. Bishop, Mixture density networks, Aston University, 1994.

[29] D. P. Kingma, and M. Welling, Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114, 2013.

[30] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling, Semi-supervised learning with deep generative models. Advances in Neural Information Processing Systems (NIPS), pp. 3581-3589, 2014.

[31] K. Cho, B. V. Merrinboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, Learning phrase representations using RNN encoder-decoder for statistical machine translation, arXiv preprint arXiv:1406.1078, 2014.

[32] A. Graves, Generating sequences with recurrent neural networks. arXiv preprint arXiv:1308.0850, 2013.

[33] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980, 2014.

[34] A. Houenou, P. Bonnifait, V. Cherfaoui, and W. Yao. Vehicle trajectory prediction based on motion model and maneuver recognition, IEEE/RSJ Conference on Intelligent Robots and Systems (IROS), pages 4363-4369, 2013.