Convolution Neural Network-based Lane Change Intention Prediction of Surrounding Vehicles for ACC

Donghan Lee[1*], Youngwook Paul Kwon[1*], Sara McMains[1], and J. Karl Hedrick[1]

***Abstract*— Adaptive cruise control is one of the most widely used vehicle driver assistance systems. However, uncertainty about drivers' lane change maneuvers in surrounding vehicles, such as unexpected cut-in, remains a challenge. We propose a novel adaptive cruise control framework combining convolution neural network (CNN)-based lane-change-intention inference and a predictive controller. We transform real-world driving data, collected on public roads with only standard production sensors, to a simplified bird's-eye view. This enables a CNN-based inference approach with low computational cost and robustness to noisy input. The predicted inference of traffic participants' lane change intention is utilized to improve safety and ride comfort with model predictive control. Simulation results based on driving scene reconstruction demonstrate the superior performance of inference using the proposed CNN-based approach, as well as enhanced safety and ride comfort.**

I. INTRODUCTION

Adaptive Cruise Control (ACC) systems, introduced in the early 1990s, are present in many passenger vehicles today [1]. ACC systems focus entirely on maintaining the space between the ego car and a preceding car or tracking a desired speed that a driver selects. The main benefit of ACC systems is reducing driver fatigue from manually operating brake and gas pedals. Moreover, ACC is a crucial part of emerging autonomous driving systems. However, several limitations still remain, including the lane change prediction problem. The challenge is when a vehicle swerves into the ego car's lane from an adjacent lane and is not recognized by the system until it is fully inside of the ego car's lane. This can make a driver feel uncomfortable and unsafe. Most drivers will cancel the ACC system by pushing the brake pedal in such scenarios. If the system can predict the motions of surrounding vehicles, such as lane keeping and lane changing, it will obtain crucial information to overcome the aforementioned issues.

An overview of existing approaches for the motion prediction of surrounding vehicles can be found in [2]. They propose a classification into three motion models: *physics-based, maneuver-based, and interaction-aware motion models*. First, *physics-based motion models* propagate the vehicle's motion based on the laws of physics. In this class, constant kinematic models are widely utilized because they can easily infer the future motion of traffic participants under a few assumptions [3]. For instance, the Constant Turn Rate and Velocity (CTRV) models take into account the variation around the z-axis by introducing the yaw angle and yaw rate variables in the vehicle state vector [4], [5], [6]. However, these models are limited to short-term motion prediction, since they are not able to anticipate any change in the motion of vehicles due of drivers' maneuvers such as lane changes [2], [7].

Many researchers tackle the long-term prediction issue based on *maneuver-based motion models*. A wide range of machine learning techniques are utilized in order to predict vehicle maneuvers in this class. One such method is a hidden Markov model (HMM), which is used for lane change prediction given features such as lateral position of the target vehicle with respect to the center-line of the nearest lane [8]. Support vector machines [9] and random forests [10] are also popular data-driven approaches for maneuver intention estimation. Probabilistic Multi-layer perceptron (MLP) is proposed for lateral motion prediction in [3]. Based on a set of three representative trajectories for each target lane, the MLP model provides probabilities of how likely a vehicle will follow each trajectory and each lane with high speed data sets. A limitation of methods in this class is the assumption that vehicles move independently from each other, which does not hold in practice because motion by one vehicle will necessarily influence the maneuvers of other vehicles.

Interaction-aware motion models consider inter-vehicle dependencies, i.e. the motion of a vehicle is assumed to be influenced by the motion of other vehicles in the scene. In [11], coupled hidden Markov models (CHMMs) offer a way to model multiple interacting processes without a conflict with the Markov condition. However, as the number of possible pairwise dependencies grows in the context of complex traffic situations, the complexity quickly becomes intractable. There is some work that reduces the computational complexity of the problem based on the assumption of asymmetric dependencies [12], [13]. However, computations are still expensive and not compatible with real-time tasks.

In this paper, we propose a novel approach to infer traffic participants' lane-change-intentions based on a convolution neural network (CNN) for enhancing ACC. The lane change prediction model provides the intent of drivers of surrounding vehicles to a predictive controller, which is in charge of guaranteeing drivers' safety and ride comfort. The contribution of our work is twofold. The main contribution is that the inter-vehicle dependencies are innately taken into account with real-time motion prediction. The proposed CNN-based approach is motivated by the recent successful results of CNN-based methods in various image-related tasks (object detection [14], image retrieval [15], image segmenta-

[1]Donghan Lee, Youngwook Paul Kwon, Sara McMains, and J. Karl Hedrick are with the Department of Mechanical Engineering, University of California, Berkeley, CA 94720, USA. {donghan.lee@, young@, mcmains@, khedrick@me.}berkeley.edu

*The authors contributed equally.

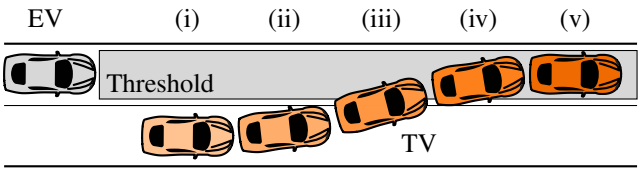

Fig. 1. Ego vehicle (EV) and a target vehicle (TV) cutting in

tion [16]), capturing complex and non-linear patterns in data sets. Even though one could train a CNN for lane change prediction directly from camera images, the process would require a large number of annotated images and a network with many layers such as the 19-layer VGGnet [17] used for classifying natural images. Hence, we introduce a compact, binary, and simplified bird's-eye view (SBV), which enables us to utilize a CNN-based approach, and at the same time, to train a model using real data sets with significantly reduced computational cost.

The secondary contribution is that only one front-facing radar sensor and one camera sensor, which are standard equipment on currently manufactured cars with ACC, are used for the inference of the lane change motions. This is quite unlike previous work where additional Lidar sensors [8] or multiple radar and camera sensors [18] are utilized to estimate the behavior of the surrounding traffic participants. This implies that the proposed algorithm can be applied to currently manufactured cars with driver assistance systems such as ACC and lane keeping assistance.

II. PROBLEM DEFINITION

A target merging from adjacent lanes is chosen as a primary target (PT) in commercial ACC systems such as those from Mando Corp. only when the target fully gets into the threshold corresponding to step (iv) in Fig. 1. This late PT detection becomes critical in scenarios where a car is cutting into the ego vehicle's lane without keeping a proper safety distance or when it is decelerating while changing lanes. In this study, we do not consider high speed cases where the speed of the EV is over 50 miles per hour, since nearly 74% of crashes related to this challenge occurred on roadways less then or equal to 45 miles per hour [19]. Usually, at high speed the EV has enough distance to decelerate its speed to avoid collision with merging vehicles compared to low speed scenarios.

III. METHODS

A. System Description

The proposed system architecture is shown in Fig. 2. In the "Environmental Perception" module, sensor fusion combines measurements from radar and camera sensors to provide relative positions and relative velocities from the ego vehicle (EV) to target vehicles (TVs) in neighboring lanes. In addition, lane information in terms of the lateral distance from the EV to the lane markers and two degree-3 polynomials (one for each lane) is transferred to the "Simplified Bird's-eye View (SBV)" module. In the SBV module, we reconstruct a simplified driving scene based on the observations from

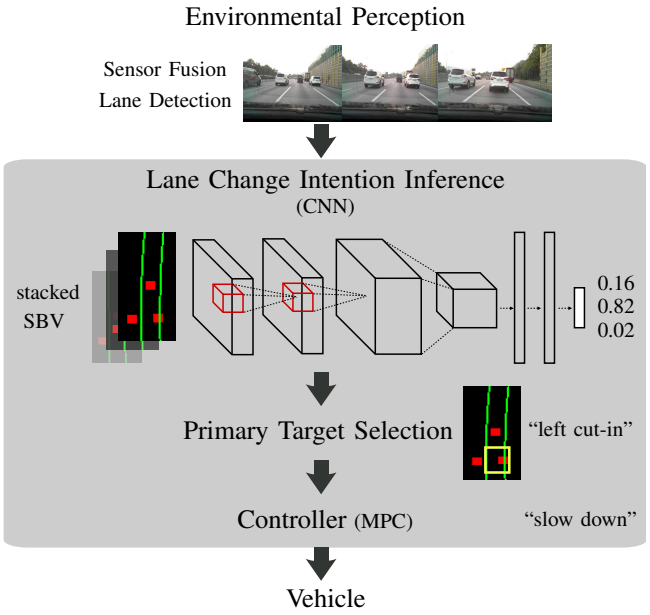

Fig. 2. The proposed adaptive cruise control framework with an example scenario

the "Environmental Perception" module. Then, the CNN-based "Lane Change Intention Inference" module computes the most likely lane change intention of TVs. Lastly, the "Primary Target Selection" module chooses the PT as the reference for the controller at the current time, and transfers its relative distance and the relative velocity to a predictive controller.

B. Sensor Fusion Data and Simplified Bird's-eye View

Real world driving data[1] was collected during straight driving on public roads with three different cars which each have one radar sensor and one camera sensor installed. Mando Corp.'s implementation of sensor fusion for more accurate relative positions and velocities compared to single sensor cases uses a probabilistic track-to-track association algorithm to determine which tracks pertain to the same real-world object. We take our "Environmental Perception" output from this commercial system.

We generate an SBV from these outputs. Specifically, at any given time frame, the SBV is a *binary* 2-channel ($h \times w \times 2$) image. Since we do not consider high speed scenarios, the $50m \times 10m$ physical region ($50m$ to the forward direction, and $5m$ to the left and right side with regard to the EV) is determined as the size of the SBV. Hence, one pixel in a SBV corresponds to $h/50 \times w/10$ physical square meters. There are two channels, a vehicle channel and a lane channel. We mark a vehicle ($4m \times 1.6m$) in the vehicle channel, and draw lane information (third-degree polynomials) in the lane channel with the width of two pixels.

The inference of the lane change intent should be made by context, i.e., inputs for the inference at a certain moment should include earlier frames as well as current frame. Therefore, for inference at frame t, we stack a priori n_f

[1]Mando Corp., an automotive part manufacturing company in S.Korea, provided the data sets.

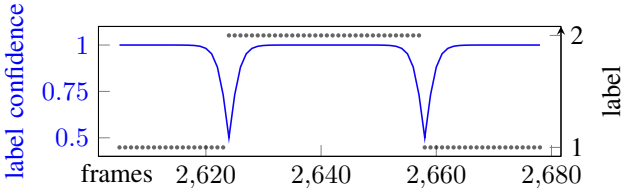

Fig. 3. Label confidence. A left cut-in takes place in this example. The black dots show labels for each frame, and the blue solid line represents the label confidence. The frame interval is 0.05 seconds.

SBVs with δ_f frame intervals, i.e., SBVs at frame $t, t-\delta_f, ...,$ and $t-(n_f-1)\delta_f$. We denote these stacked SBVs, a $h \times w \times 2n_f$ binary matrix, as x_t, which is the input of the proposed inference network.

An example stacked SBVs is shown in Fig. 2 ($h = 250, w = 100, c_h = 8, c_w = 8, n_f = 3$). Note that a SBV is a binary image even though we visualize each SBV as a color image with the vehicle channel in red, and the lane channel in green in Fig. 2.

C. Lane Change Intention Inference

We define a CNN-based network f that consists of a set of weights, W. The network can be thought of as a function that takes a $h \times w \times 2n_f$ binary (stacked) SBV x_t at frame t as input, and outputs the probability distribution $\hat{y}_t = (\hat{y}_t^{(1)}, \hat{y}_t^{(2)}, \hat{y}_t^{(3)})$ for the three classes, *lane keeping* (label 1), *left cut-in* (label 2), and *right cut-in* (label 3). This is denoted as:

$$\hat{y}_t = f(x_t|W)$$
$$x_t \in \{0,1\}^{h\times w\times 2n_f}, \quad \hat{y}_t \in [0,1]^3, \quad \sum_{i=1}^{3} \hat{y}_t^{(i)} = 1.$$

Using x_t and its one-hot encoded[2] 3-tuple label y_t, we train the network so as to find the optimal W^* that minimizes a loss function as follows:

$$W^* = \underset{W}{\operatorname{argmin}} \frac{1}{n}\sum_{t=1}^{n} \text{loss}(y_t, \hat{y}_t)$$
$$= \underset{W}{\operatorname{argmin}} \frac{1}{n}\sum_{t=1}^{n} \text{loss}(y_t, f(x_t|W)).$$

Since our SBV-based inputs are small, sparse, and binary, we do not need complex neural network architectures such as designed for natural images. We experimentally chose six layers, as illustrated in Fig. 2. The convolution and fully connected layers have rectified linear unit (ReLU) activation, except for the final fully connected layer. To avoid over-fitting, we add dropout [20] to fully connected layers except for the last one. By taking the soft-max function of the final layer, a 3-tuple probability distribution $\hat{y}$ is acquired, namely, the posterior probability corresponding to *lane keeping*, *left cut-in*, and *right cut-in*.

We modify the standard cross entropy loss typically used for classification problems to address the uncertainty in the label assignment in this application. Recall that a label is assigned to each frame by a human expert. However, there is uncertainty near the boundaries of the label transitions and noise in the actual labeling of data. Thus, error will likely occur near label boundaries [21]. Hence, we assign to each frame not only a label but also a *label confidence*. At a given frame t, we find the closest human-labeled earlier and later transition moments, $t_{earlier}$ and t_{later}, respectively. For example, in Fig. 3, frame 2640 has $t_{earlier} = 2640 - 2623$ and $t_{later} = 2658 - 2640$ (because the transitions in the human-expert labels occur at frames 2623 and 2658). A label confidence l_t at frame t is defined as follows:

$$l_t = \min(\text{sigmoid}(t_{earlier}), \text{sigmoid}(t_{later}))$$

where $\text{sigmoid}(x) = 1/(1+e^{-sx})$ with s the coefficient of the saturation speed. Since $\text{sigmoid}(x)$ is 0.5 at $x = 0$, and saturates to 1 as x increases, a frame further from any transition moment gets a label confidence closer to 1. We plot example label confidences in Fig. 3 with a solid blue curve. The coefficient s is 1 in this paper since it makes sense to allow around 0.2 second windows for the transition. (Recall that the frame interval is 0.05 seconds, and $\text{sigmoid}(4) = 0.98$.)

We use label confidences as weights in the cross-entropy loss function since it is reasonable for the training examples with lower label confidence (i.e. in the transition stages) to get lower penalties for incorrect predictions. In addition, we add an L_2 regulation term to avoid over-fitting. Finally, the loss function is as follows:

$$\text{loss}(y_t, \hat{y}_t) = \frac{1}{n}\sum_{t=1}^{n}(l_t \sum_{j=1}^{3} y_t^{(j)} \log \hat{y}_t^{(j)}) + \lambda||W||_2$$

where W is the set of all trainable weights in the network; λ is a coefficient for the regulation term; $y_t^{(j)}$ is the j-th element of y_t; and n is the size of the training set.

We train the network for 50,000 iterations using a stochastic gradient method with batch size 150, starting learning rate 0.004, and decay rate 0.95 every 10,000 iterations. With the trained W^*, for a given input x_t, its 3-class probability distribution inference $\hat{y}_t$ is given as the feed-forward output of the network, i.e., $\hat{y}_t = f(x_t|W^*)$. The predicted mode $\hat{c}_t$ corresponding to x_t is the index of $\hat{y}_t$ with the maximum probability, i.e,

$$\hat{c}_t = \underset{i}{\operatorname{argmax}}\, \hat{y}_t^{(i)}. \tag{1}$$

These inference outputs, $\hat{y}_t$ and $\hat{c}_t$, are fed into the *Primary Target Selection* discussed in the next section.

D. Primary Target Selection

Based on the inference from the lane change intention inference model, it is necessary to choose the PT for the controller. First, we partition all target vehicles (TVs) in the SBV into three groups: those in the adjacent right lane, in the adjacent left lane, and in the EV's lane, based on the positions of the lane markers and each TV. Then, the relevant targets (RTs) include:

- Left cut-in RT (LRT):=the closest vehicle among the TVs in the adjacent right lane whose $\hat{y}_t^{(2)} \geq y_{th}$,

[2]E.g., an integer label '3' becomes $[0, 0, 1.0]$

- Right cut-in RT (RRT):=the closest vehicle among the TVs in the adjacent left lane whose $\hat{y}_t^{(3)} \geq y_{th}$,

where y_{th} denotes the threshold for the probability distribution inference. Moreover, the front target (FT) is defined as the closest vehicle among the TVs in the EV's lane. If the FT and LRT exist in a scenario, the relative position and velocity of the PT is computed as follows:

$$\begin{aligned} p_t^p &= (1-\hat{y}_t^{(2)})p_t^f + \hat{y}_t^{(2)} p_t^r \\ v_t^p &= (1-\hat{y}_t^{(2)})v_t^f + \hat{y}_t^{(2)} v_t^r \end{aligned} \tag{2}$$

where $\{p_t^p, v_t^p\}$, $\{p_t^f, v_t^f\}$, and $\{p_t^r, v_t^r\}$ represent the longitudinal position and the longitudinal velocity of the PT, FT, and RT at instant time t, respectively. In case that the FT and RRT exist in the SBV, $\hat{y}_t^{(2)}$ is simply replaced by $\hat{y}_t^{(3)}$ in Eq. (2). In addition, if the FT does not exist in a driving scene, p_t^f is set to be zero, and v_t^f is replaced by v_{set}, which is the desired speed that a driver selects in Eq. (2).

E. Controller

Model predictive control (MPC) is utilized for the controller. MPC iteratively solves a finite-horizon constrained optimal control problem: after computing the optimal control inputs over a finite planning horizon, the controller implements the first input and then computes a new set of control inputs, starting from the updated state. The longitudinal motion of the EV is described as follows:

$$\begin{aligned} \mathbf{x}_{k+1|t} &:= \begin{bmatrix} p_{k+1|t} \\ v_{k+1|t} \\ a_{k+1|t} \end{bmatrix} = \begin{bmatrix} p_{k|t} + T_s v_{k|t} + 0.5T_s^2 a_{k|t} \\ v_{k|t} + T_s a_{k|t} \\ (1-\frac{T_s}{\tau})a_{k|t} + \frac{T_s}{\tau} u_{k|t} \end{bmatrix} \\ &:= f(\mathbf{x}_{k|t}, u_{k|t}), k = 0, \ldots, N-1 \end{aligned} \tag{3}$$

where $p_{k|t}$, $v_{k|t}$, $a_{k|t}$ denote, at any time instant t, the predicted longitudinal position, velocity and acceleration of the EV at time $t+k$, respectively; T_s denotes the sampling time; τ represents the time constant; and u_t is a control input from the controller at time instant t. This longitudinal motion model includes the acceleration lag, which is mapped to the actual longitudinal acceleration of the car by a first-order delay. To design the MPC controller, we first obtain the future motion of the PT. The motion of the PT is assumed to be governed by a kinematic model with the constant velocity (CV) assumption as follows:

$$\begin{bmatrix} p_{k+1|t}^p \\ v_{k+1|t}^p \end{bmatrix} = \begin{bmatrix} p_{k|t}^p + T_s v_{k|t}^p \\ v_{k|t}^p \end{bmatrix}, k = 0, \ldots, N-1$$

where $p_{k|t}^p$, $v_{k|t}^p$ are, at any time instant t, the predicted longitudinal position and velocity of the PT at time $t+k$. Although the CV assumption is not realistic, the results in [22] indicate that there is almost no difference in the performance between the CV model and state-of-the-art approaches in cases that the horizon is less than 2 seconds. Since the horizon in our controller is chosen as 0.5 seconds, the CV assumption is reasonable for this application.

The optimal control input u_t can be obtained at time t by solving the following finite horizon constraint optimal control problem that incorporates the kinematics of the EV in Eq. (3) and the safety and comfort requirement:

$$\min_{\mathbf{u}_t, \epsilon} \sum_{k=0}^{N-1} J_{TE} + J_{DC} + \rho\epsilon^2 \tag{4a}$$

$$\text{s.t.} \quad \mathbf{x}_{k+1|t} = f(\mathbf{x}_{k|t}, u_{k|t}) \tag{4b}$$

$$d_{k|t}^{des} = v_{k|t}\tau_{\text{gap}} + d_{\text{safe}} \tag{4c}$$

$$\Delta u_{k|t} = u_{k|t} - u_{k-1|t} \tag{4d}$$

$$\epsilon v_{min}^u + u_{min} \leq u_{k|t} \leq \epsilon v_{max}^u + u_{max} \tag{4e}$$

$$\epsilon v_{min}^{\Delta u} + \Delta u_{min} \leq \Delta u_{k|t} \leq \epsilon v_{max}^{\Delta u} + \Delta u_{max} \tag{4f}$$

$$\epsilon v_{min}^a + a_{min} \leq a_{k|t} \leq \epsilon v_{max}^a + a_{max} \tag{4g}$$

$$d_{\text{safe}} \leq p_{k|t}^p - p_{k|t} \tag{4h}$$

$$0 \leq v_{k|t} \leq v_{\text{set}} \tag{4i}$$

$$k = 0, \ldots, N-1$$

$$\mathbf{x}_{0|t} = \mathbf{x}_t, \quad u_{-1|t} = u_{t-1}, \quad \epsilon \geq 0 \tag{4j}$$

where J_{TE} in Eq. (4a) represents the cost for tracking errors including distance and velocity errors, and J_{DC} in Eq. (4a) is the cost for driver comfort based on control inputs, rate of control inputs, and accelerations of the EV defined as:

$$\begin{aligned} J_{TE} &:= w_d((p_{k|t}^p - p_{k|t}) - d_{k|t}^{des})^2 + w_v(v_{k|t}^p - v_{k|t})^2 \\ J_{DC} &:= w_u(u_{k|t})^2 + w_{\Delta u}(\Delta u_{k|t})^2 + w_a(a_{k|t})^2 \end{aligned}$$

and $\mathbf{u}_t$ denotes the control input sequence $(u_t, \ldots, u_{t+N-1})$ to be optimized over; $d_{k|t}^{des}$ represents the desired distance from Eq. (4c) with a time gap τ_{gap}; the comfort constraints Eq. (4e), Eq. (4f) and Eq. (4g) are imposed as soft constraints with high penalty ρ on the slack variable ϵ; Eq. (4h) and Eq. (4i) are the safety constraints; and v_{set} is the constant speed chosen by a driver. The optimization problem Eq. (4) is a quadratic program, which we solve using GUROBI [23].

IV. RESULTS

80 individual driving cases of lane changes were collected on public roads. We focus specifically on short segments of data (around $10-20$ seconds each) that involve a lane change maneuver by a TV in a neighboring lane. The segment start time is usually chosen to be a few seconds before the TV initiates the lane change and the end time is when the TV completes the lane change. We randomly choose seven driving cases as a test set, and randomly divide 90% and 10% of the rest into a training set and a validation set, respectively. Due to the relatively small number of data sets, data augmentation is implemented by (1) horizontal flipping (label 2 and 3 are switched), and (2) vertical translation randomly within [0, 20] (all labels are preserved).

To decide various hyper parameters of the inference network, we tested seven configurations on the validation set. Then, with the best model, we evaluate the proposed framework with the test set based on the reconstructed driving scene for a fair performance comparison.

Test	SBV				Network		Accuracy	wF
	δ_f	n_f	h	w	Archi.[3]	λ		
T1	8	5	50	50	C1/F3	0.01	93%	.874
T2	10	5	100	50	C1/F3	0.01	88%	.850
T3	4	10	50	100	C1/F3	0.01	91%	.834
T4	8	5	50	50	C2/F3	0.01	**95%**	**.877**
T5	8	5	50	50	C3/F3	0.01	92%	.859
T6	8	5	50	50	C1/F3	0.1	92%	.835
T7	8	5	50	50	C1/F3	1	93%	.837

TABLE I

VALIDATION SET EVALUATION: EXPERIMENT SETTING AND PERFORMANCE

A. Lane Change Intention Inference

Among the huge number of possible hyper parameter combinations, we select seven candidate configurations to test (Table I). The *Accuracy* is computed as:

$$\text{Accuracy } (\%) \ = \frac{\sum_{t=1}^{T} \mathbb{1}(c_t, \hat{c}_t)}{T} \times 100, \qquad (5)$$

where c_t and $\hat{c}_t$, respectively, denote the true mode (determined by a human expert) and the predicted mode (Eq. (1)) at time step t, and $\mathbb{1}(a, b) = 1$ if $a = b$, else 0. Since test set is imbalanced (naturally lane-keeping cases are more frequent), we calculated the weighted average of F_1 score (also known as F-measure) for each mode in the *wF* column. The score is the harmonic mean of precision and recall. Readers can refer to [24] for more details of the F_1 score.

The tests T1, T2, and T3 are distinct in the SBV-related parameters. The aspect ratio of the SBV in T2 is relatively more similar to the actual aspect ratio ($50m \times 10m$) than the others. We gradually exaggerate the horizontal aspect ratio (width to height) to T1 and T3 due to the importance of horizontal movement in lane changing motion. We found that the intermediate configuration (T1) worked best for the SBV-related parameters. By fixing the SBV-related parameters, the T1, T4, and T5 are distinct in the network architecture. We gradually increase the network size from T1 to T4 and T5. T1, T6, and T7 are distinct in regulation coefficient, which did not significantly affect the results. Ultimately, we chose configuration T4 for simulation tests on the test set.

B. Driving Scene Reconstruction

For the reconstruction of driving scenes for the test set, it is necessary to know the position of the TV with respect to an inertial reference frame. We set the inertial reference frame at the center of gravity of the EV at the time instant when a driving scene starts; then we compute the position of the EV using a kinematic model with observations such as the speed of the EV and the relative heading angle with respect to the lane. Finally, the position of the TV is computed by adding the relative distance to the EV. Note that we only consider a driving scene on a straight road in this reconstruction due to the limitation of estimating the positions of EV with on-board sensors such as wheel-speed sensors. Fig. 4 shows the inertial coordinates to find the global positions of the TV. Then, the global positions of TVs and the positions of the lane markers with respect to the EV are passed to the SBV and lane-change-intention inference algorithm. After selecting the PT, the position and the velocity of the PT are transferred to the controller. Finally, the motion of the EV is simulated using Eq. (3).

[3]The details are as follows: "C1/F3": C(6,32)-P-F(1024)-F(1024)-F(3), "C2/F3": C(6,32)-C(1,16)-P-F(1024)-F(1024)-F(3), "C3/F3": C(6,64)-P-C(3,32)-P-C(1,32)-P-F(1024)-F(1024)-F(3) where $C(w, x)$: a convolutional layer with x $w \times w$ filters, P: 2x2 max-pooling, $F(x)$: a fully connected layer with x output neurons.

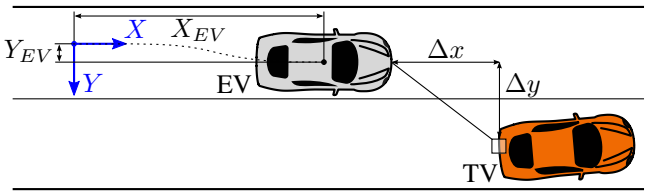

Fig. 4. Illustration of the utilized coordinates. The EV's global position is given by X_{EV} and Y_{EV}. The longitudinal and lateral position of the TV with respect to the EV is denoted by Δx and Δy.

Metric	Unit	CNN-ACC	HMM-ACC	Com-ACC
max_a	m/s^2	**1.73 ± 0.54**	1.79 ± 0.53	1.76 ± 0.58
max_j	m/s^3	**3.94 ± 1.46**	6.31 ± 2.54	7.67 ± 3.18
TTCi	$1/s$	**0.20 ± 0.07**	0.21 ± 0.07	0.21 ± 0.07
Accuracy	%	**89.87 ± 6.97**	81.14 ± 16.56	-

TABLE II

TEST SET EVALUATION: MEAN AND STANDARD DEVIATIONS OF EACH METRIC FOR THE THREE APPROACHES

C. ACC Simulation

We compare the performance of the proposed ACC approach (denoted as *CNN-ACC*) with the commercial solution described in Section II (denoted as *Com-ACC*) and ACC based on probabilistic inference using a hidden Markov model (denoted as *HMM-ACC*). The details of *HMM-ACC* are covered in [25]. For a fair evaluation, the same control strategy with the parameters in Section III-E is used for all three approaches. We first define metrics for the evaluation of the aforementioned methods with each test data set as follows:

- Absolute vales of the maximum deceleration, max_a
- Absolute vales of the maximum negative jerk, max_j, defined as the rate of the change of deceleration
- Maximum inverse time-to-collision, *TTCi*, defined as the ratio of the relative speed and the relative distance of the EV with respect to the PT
- Accuracy of the inference, *Accuracy*, defined in Eq. (5)

Metrics max_a and max_j represent the ride comfort of the ACC system, where lower values correspond to a smoother behavior. Metric *TTCi* characterize the safety, where a lower value indicates lower collision risk.

Table II shows the test set evaluation results for the three approaches. The average max_a and max_j of the *CNN-ACC* are smaller than for other two methods, which implies that the *CNN-ACC* provides superior ride comfort. However, the three approaches have quite similar *TTCi*. This may be the case since the chosen test scenarios are not risky. i.e. the EV has enough space to decelerate its speed without steep

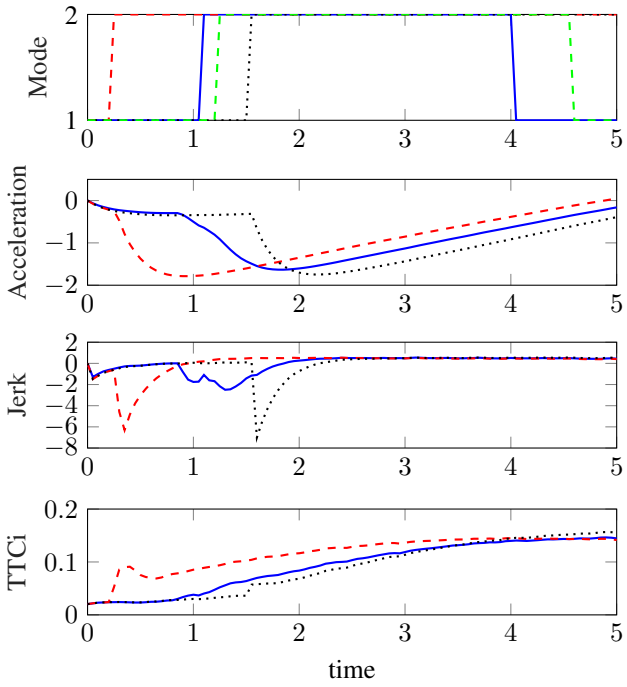

Fig. 5. Comparison of the most likely mode, acceleration, jerk, and inverse time-to-collision for a given three approaches: *CNN-ACC* (blue), *HMM-ACC* (red, dashed), and *Com-ACC* (black, dash-dotted). The dashed green line on the first plot represents the human label.

deceleration. The *Com-ACC* declares a RT as the PT only when it fully crosses into the EV's lane, so the *Accuracy* of the *CNN-ACC* is only compared with the *HMM-ACC* because the *Com-ACC* is not based on inferring a driver's intent inference. The *CNN-ACC* achieves 8% higher *Accuracy* than *HMM-ACC*. One example result is highlighted in Fig. 5. The *HMM-ACC* declares the RT as the PT earlier than the human label while the the most likely mode of the *CNN-ACC* is triggered around 1 second, which is similar to the human expert's decision. Moreover, the *CNN-ACC* declares the RT as the PT 0.5 seconds earlier than the *Com-ACC*. Referring back to Fig. 1, this time gap roughly corresponds to detecting the vehicle's lane change intention in phase (ii) rather than (iv). Both early and late declaration with *HMM-ACC* and *Com-ACC* result in larger jerk than with *CNN-ACC*.

V. CONCLUSIONS

We present a novel adaptive cruise control framework, in which CNN-based lane-change-intention prediction is integrated with a model predictive control-based speed controller. A simplified bird's-eye view enables a computationally efficient CNN-based approach. The reconstructed driving scene is utilized for the evaluation of the proposed method, and the results show the superior performance in terms of accuracy and driver comfort compared with a HMM-based inference method and a commercial solution. In future work, we plan to add more data sets to improve the robustness of the proposed methods in diverse scenarios. Moreover, real-experiments on public roads will be performed.

REFERENCES

[1] L. Xiao and F. Gao, "A comprehensive review of the development of adaptive cruise control systems," *Vehicle System Dynamics*, vol. 48, no. 10, pp. 1167–1192, 2010.

[2] S. Lefèvre, D. Vasquez, and C. Laugier, "A survey on motion prediction and risk assessment for intelligent vehicles," *Robomech Journal*, vol. 1, no. 1, p. 1, 2014.

[3] S. Yoon and D. Kum, "The multilayer perceptron approach to lateral motion prediction of surrounding vehicles for autonomous vehicles," in *Intelligent Vehicles Symposium*. IEEE, 2016, pp. 1307–1312.

[4] R. Schubert, E. Richter, and G. Wanielik, "Comparison and evaluation of advanced motion models for vehicle tracking," in *Information Fusion*. IEEE, 2008, pp. 1–6.

[5] N. Kaempchen, K. Weiss, M. Schaefer, and K. C. Dietmayer, "Imm object tracking for high dynamic driving maneuvers," in *Intelligent Vehicles Symposium*. IEEE, 2004, pp. 825–830.

[6] P. Lytrivis, G. Thomaidis, and A. Amditis, "Cooperative path prediction in vehicular environments," in *Intelligent Transportation Systems*. IEEE, 2008, pp. 803–808.

[7] A. Houenou, P. Bonnifait, V. Cherfaoui, and W. Yao, "Vehicle trajectory prediction based on motion model and maneuver recognition," in *Intelligent Robots and Systems (IROS)*. IEEE, 2013, pp. 4363–4369.

[8] A. Carvalho, A. Williams, S. Lefèvre, and F. Borrelli, "Autonomous cruise control with cut-in target vehicle detection," in *International Symposium on Vehicle Control (AVEC)*, 2016.

[9] P. Kumar, M. Perrollaz, S. Lefevre, and C. Laugier, "Learning-based approach for online lane change intention prediction," in *Intelligent Vehicles Symposium*. IEEE, 2013, pp. 797–802.

[10] J. Schlechtriemen, F. Wirthmueller, A. Wedel, G. Breuel, and K.-D. Kuhnert, "When will it change the lane? a probabilistic regression approach for rarely occurring events," in *Intelligent Vehicles Symposium*. IEEE, 2015, pp. 1373–1379.

[11] M. Brand, N. Oliver, and A. Pentland, "Coupled hidden markov models for complex action recognition," in *CVPR*, 1997, pp. 994–999.

[12] T. Christopher, "Analysis of dynamic scenes: Application to driving assistance," Ph.D. dissertation, Institut National Polytechnique de Grenoble-INPG, 2009.

[13] M. Liebner, M. Baumann, F. Klanner, and C. Stiller, "Driver intent inference at urban intersections using the intelligent driver model," in *Intelligent Vehicles Symposium*. IEEE, 2012, pp. 1162–1167.

[14] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *CVPR*, 2016.

[15] M. Paulin, M. Douze, Z. Harchaoui, J. Mairal, F. Perronin, and C. Schmid, "Local Convolutional Features with Unsupervised Training for Image Retrieval," in *ICCV*, 2015, pp. 91–99.

[16] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *CVPR*, 2015, pp. 3431–3440.

[17] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *ICRL*, 2015, pp. 1–14.

[18] J. Schlechtriemen, A. Wedel, J. Hillenbrand, G. Breuel, and K.-D. Kuhnert, "A lane change detection approach using feature ranking with maximized predictive power," in *Intelligent Vehicles Symposium*. IEEE, 2014, pp. 108–114.

[19] B. Sen, J. D. Smith, and W. G. Najm, "Analysis of lane change crashes," Tech. Rep., 2003.

[20] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[21] K. R. Driggs-Campbell and R. Bajcsy, "Identifying modes of intent from driver behaviors in dynamic environments," *CoRR*, vol. abs/1505.05921, 2015.

[22] J. Schlechtriemen, A. Wedel, G. Breuel, and K.-D. Kuhnert, "A probabilistic long term prediction approach for highway scenarios," in *Intelligent Transportation Systems*. IEEE, 2014, pp. 732–738.

[23] I. Gurobi Optimization, "Gurobi optimizer reference manual," 2016. [Online]. Available: http://www.gurobi.com

[24] D. M. Powers, "Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation," 2011.

[25] D. Lee, A. Hansen, and J. K. Hedrick, "Probabilistic inference of traffic participants' lane change intention for enhancing adaptive cruise control," in *Intelligent Vehicles Symposium*. IEEE, 2017.