2017 IEEE Intelligent Vehicles Symposium (IV)
June 11-14, 2017, Redondo Beach, CA, USA

Automated Driving by Monocular Camera Using Deep Mixture of Experts

V. John[1], S. Mita[1], H. Tehrani[2] and K. Ishimaru[3]

***Abstract*— In this paper, we propose a real-time vision-based filtering algorithm for steering angle estimation in autonomous driving. A novel scene-based particle filtering algorithm is used to estimate and track the steering angle using images obtained from a monocular camera. Highly accurate proposal distributions and likelihood are modeled for the second order particle filter, at the scene-level, using deep learning. For every road scene, an individual proposal distribution and likelihood model is learnt for the corresponding particle filter. The proposal distribution is modeled using a novel long short term memory network-mixture-of-expert-based regression framework. To facilitate the learning of highly accurate proposal distributions, each road scene is partitioned into straight driving, left turning and right turning sub-partitions. Subsequently, each expert in the regression framework accurately model the expert driver's behavior within a specific partition of the given road scene. Owing to the accuracy of the modelled proposal distributions, the steering angle is robustly tracked, even with a limited number of sampled particles. The sampled particles are assigned importance weights using a deep learning-based likelihood. The likelihood is modeled with a convolutional neural network and extra trees-based regression framework, which predicts the steering angle for a given image. We validate our proposed algorithm using multiple sequences. We perform a detailed parameter analysis and a comparative analysis of our proposed algorithm with different baseline algorithms. Experimental results show that the proposed algorithm can robustly track the steering angles with few particles in real-time even for challenging scenes.**

I. Introduction

Vision-based steering angle tracking is an important problem in automated driving and advanced driver assistance systems (ADASs) [1]. Typically, ADASs assist drivers and reduce the number of driver fatigue related accidents. Vision-based steering angle estimation is the task of predicting the vehicle's steering angle using images obtained from the on-board camera system. The main challenges in solving the problem occur from the appearance, environmental and illumination variations in the road scene and non-linear vehicle dynamics. Additionally, the proposed solution should be computationally efficient.

State-of-the art vision-based steering angle prediction algorithms [2], [3] estimate the steering angle from images using the end-to-end deep learning framework. The deep learning framework predicts the steering angles from image frames. In this work, we extend the state-of-the-art by proposing a filtering algorithm to estimate the steering angle.

[1] V. John and Seiichi Mita are with the Toyota Technological Institute, Japan {vijayjohn, smita}@toyota-ti.ac.jp.
[2] Hossein Tehrani is with Denso, Japan hossein_tehrani,@denso.co.jp
[3] Kazuhisa Ishimaru is with Nippon Soken, Japan kazuhisa_ishimaru@soken1.denso.co.jp

Tracking steering angles to perform vision-based automated driving is not straightforward. Non-linear filtering algorithms like the particle filter [4] with first order Markov chain can be used to track the steering angles. However, a high number of particles have to be sampled from wide distributions, for example, Gaussian distribution with large standard deviation, to account for the non-linear vehicle dynamics, sudden trajectory changes and road scene variations, limiting the computational efficiency [4].

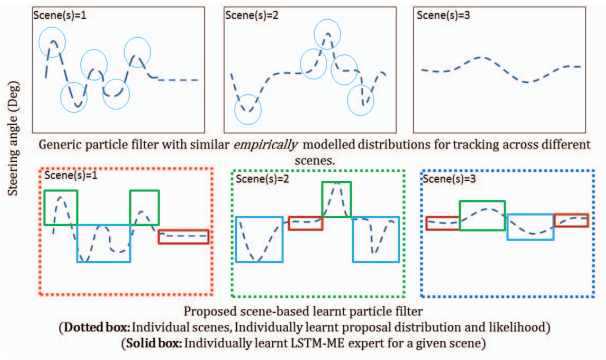


Fig. 1. Comparison between the standard particle filter and the scene-based deep filtering framework. In the first row, the trajectory change regions are highlighted. In the second row, for the scene-based particle filtering framework, each image corresponds to a specific scene. Additionally, we highlight the straight driving, left turning and right turning partition within each road scene.

In this paper, we first experimentally validate these limitations of the standard particle filter (Sec V). Subsequently, we propose a novel scene-based higher order particle filter to estimate the steering angles. In the proposed filtering framework, we firstly, adopt the second order particle filter to track the non-linear steering angle dynamics. Secondly, we accurately model the proposal distributions and the likelihoods for the particle filters at the scene level using deep learning. More specifically, an individual proposal and likelihood distribution is modeled for an individual road scene. The proposal distribution is modelled using a novel long short term-mixture-of-expert-based regression framework. The novel *deep* mixture-of-expert (DME) framework accurately models the regression data with a set of experts. Each individual expert models an expert driver's behavior in a specific partition of a given road scene. The road scene partition correspond to straight driving, right turning and left turning. The proposed DME framework accurately models the proposal distribution compared to the a single deep learning framework as shown in the experimental section (Sec V). Owing to the highly accurate proposal distributions, we can

robustly track the steering angles with few particles (200), sampled from the modelled, narrow, proposal distribution. A comparison between the standard particle filter and the proposed DME framework is shown in Fig 1.

The samples generated from the proposal distribution are assigned importance weights and normalized to represent the posterior distribution. The importance weights are computed using a deep learning-based likelihood. More specifically, we adopt the convolutional neural network (CNN) and extra trees-based regression framework to predict the steering angle from an individual road scene image. The predicted steering angle is then utilized to model the likelihood. We evaluate our proposed algorithm using multiple datasets. We compare the performance of our algorithm with baseline algorithms, and report improved performance. Additionally, we perform a detailed parameter analysis of our algorithm, and present a discussion of the results.

Our main contribution to literature are as follows: 1) Real-time steering angle tracking using learnt scene-based second order particle filtering; 2) Novel long short term memory-based mixture of experts to formulate the proposal distribution; 3) Convolutional neural network and extra trees-based prediction of the steering angle from the images. 4) Experimental validation of the vision-based steering angle tracking using the standard particle filter. The rest of the paper is structured as follows. In Section II, we review the literature. We present a brief overview of the proposed algorithm along with the research problem in Section III. The details of the algorithm are presented in Section IV and the results obtained are presented in Section V. We present our discussion and the direction of future work in Section VI.

II. Literature Review

Tracking has been studied extensively, and has been used for a number of applications [1], [5]. In ADASs, filtering algorithms like the Kalman filter and non-linear particle filter [4] have been utilized for problems such as vehicle tracking [1] and pedestrian tracking [5]. The generic particle filter is a non-linear filtering approach formulated using first order Markov chain, where the current state depends on the previous state alone [4]. The generic particle filter with first order motion model is widely used in literature. However, in the case of objects with highly non-linear motion or high dimension search space, the generic particle filter is not sufficient [6]. Researchers address this limitation by either incorporating machine learning techniques such as Gaussian process [7] or by subspace modeling [8] within the particle filter framework. For example, in the work by Ko et al. [7], learnt Gaussian process-based regression models are used to represent the first order motion and observation models within the particle filter. Alternatively, in case of subspace modeling, subspace modeling techniques such as motion PCA [8] or Gaussian process dynamic model [9] are used to learn the reduced subspace of the high dimensional search space. Subsequently, particle filtering is employed in the reduced subspace. Apart from utilizing learning techniques, researchers also address the limitations of the generic particle filter by adopting higher order motion models [6]. Such models enhance the robustness of the particle filter in tracking non-linear models. Pan et al. [6] apply Markovian properties to a moral graph to obtain closed form solutions of higher order particle filters, which are then used for human tracking. Similarly, in the work by Liu et al. [10] probabilistic tracking is performed utilizing second order dynamics.

In this work, we propose to combine deep learning within a second order particle filter to estimate steering angles from road scene images. By this approach, we integrate the advantages of learning and Bayesian filtering. Previously researchers have investigated neural network-based approaches to perform vision-based driving [2], [11], [12]. Pomerleau et al. [12] developed the Autonomous Land Vehicle in a Neural Network (ALVINN) system, where an end-to-end neural network was used to steer the vehicles. Recently, Bojarski et al. [2] extend this system by utilizing the CNN to perform vision-based steering angle prediction. This approach is further extended by Chen et al. [3], where a decision module is integrated with the CNN to perform vision-based driving, where the authors not only predict the steering angle but also make driving decisions. Compared to existing literature, we propose to perform vision-based deep filtering for autonomous driving.

III. Brief Overview of the Algorithm

We first provide a brief overview of the research problem and the algorithm components. Subsequently, we describe the learning and testing phases of the algorithm in detail.

Research Problem. For a given road scene s with k images, $I^s_{1:K}$, and corresponding image-based observations, $\mathbf{z}^s_{1:k}$, obtained using an expert driver, the vision-based filtering framework estimates the steering angles $\mathbf{x}^s_{1:k}$. The posterior distribution is represented as $P(\mathbf{x}^s_{1:k}|\mathbf{z}^s_{1:k})$. The label s for a road scene is represented using the vehicle's GPS data (latitude and longitude) and heading angle on the given scene.

In this paper, we propose a novel scene-based particle filtering framework to estimate the posterior distribution. The deep learning framework is integrated within the second order particle filter to generate the proposed filtering framework. Compared to the standard particle filter, the second order particle filter accurately models the non-linear dynamics as demonstrated in the experimental section (Sec V). Given the second order particle filter, we model highly accurate proposal distributions and likelihoods for the particle filter, at the scene-level, using deep learning. Essentially, an individual proposal distribution and likelihood is modeled for every s-th road scene. The proposal distribution is modeled using a novel long short term memory (LSTM) mixture-of-expert framework. Each individual expert models an expert driver's behavior in a specific partition of a given road scene. The road scene partition correspond to straight driving, right turning and left turning. The output of the multiple experts are weighted by the gating network, and refined by a neural network-based refinement network. An overview of the LSTM-based expert framework is presented

in Fig 2. The samples drawn from the learnt proposal distribution are assigned an importance weight using the likelihood. The likelihood is modeled using steering angle observations,($\mathbf{z}^s_{1:k}$), that are predicted from image ($I^s_{1:k}$). The observations are predicted using a CNN and extra trees-based regression framework. We next review the algorithm components before presenting the algorithm.

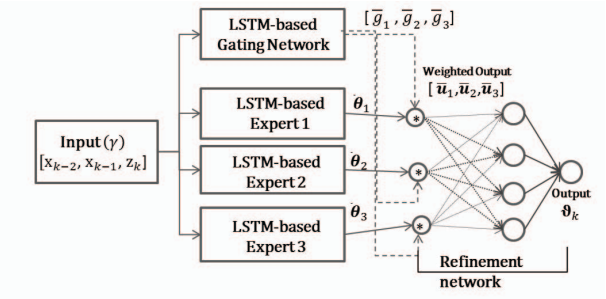


Fig. 2. An overview of the proposed LSTM mixture-of-experts based regression framework

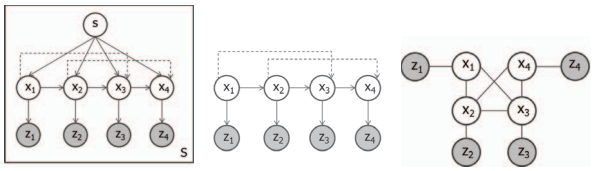
Long Short Term Memory. The Long Short Term Memory (LSTM) an extension of the recurrent neural network (RNN) is used to model the temporal dynamics [13]. The LSTM utilizes multiple memory units and gates to transmit the states x between the different time instants. Additionally, the LSTM "selectively" forgets the previous memory and only transfers some selected memory cells. Owing to this methodology, the LSTM address the vanishing gradient problem associated with the RNN.

Convolutional Neural Network. In recent years, the convolutional neural network (CNN)framework has produced state-of-the-art detection accuracy [14]. The CNN contains multiple learnable layers that simultaneously perform the feature representation and feature classification. The CNN layers extract different levels of representation from the input image using learnable filters. The multiple filters at each layer are trained using the back-propagation algorithm.

Extra trees. Extra trees is an ensemble learning method [15], where multiple models are trained on a single training data. These models are then used to make the predictions for the test data. Typically, the predictions from the individual models are combined to estimate the ensemble's final output.

A. Second Order Particle Filtering

We utilise a generic particle filter with a scene-based second order Markov chain model to perform vision-based steering angle tracking. For the s-th road scene, the current state $\mathbf{x}_k$ in the second order Markov chain depends on the previous two states $\mathbf{x}_{k-2}$ and $\mathbf{x}_{k-1}$. The proposed scene-based state space model with second order Markov chain is shown in Fig 3-a. The latent variable s corresponds to the scene label associated with the latent state or steering variables $[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_2]$. If the latent variable s is observed or known, the state space model and the corresponding moral graph is given in Fig 3-b,c.



(a) (b) (c)

Fig. 3. (a) Scene-based state space model with second order Markov chain. (b) Second order Markov chain, if the road scene is known. (c) Moral graph of the second order Markov chain.

Utilizing Markovian properties on the moral graph, the following conditional independence are obtained: $\mathbf{z}_k \perp \mathbf{z}_{1:k-1} | \mathbf{x}_{1:k}$; $\mathbf{z}_k \perp \mathbf{x}_{1:k-1} | \mathbf{x}_k$; $\mathbf{x}_k \perp \mathbf{z}_{1:k-1} | \mathbf{x}_{1:k-1}$; $\mathbf{x}_k \perp \mathbf{x}_{1:k-3} | \mathbf{x}_{k-2:k-1}$. $x \perp y | z$ denotes that x and y are conditionally independent given z. Subsequently, the following distributions are obtained,

$$\begin{aligned} p(\mathbf{z}_k|\mathbf{x}_{1:k}, \mathbf{z}_{1:k-1}) &= p(\mathbf{z}_k|\mathbf{x}_k) \\ p(\mathbf{x}_k|\mathbf{x}_{1:k-1}, \mathbf{z}_{1:k-1}) &= p(\mathbf{x}_k|\mathbf{x}_{k-2:k-1}) \end{aligned} \tag{1}$$

The above equations are used to determine the posterior distribution for the second-order particle filter, given as,

$$p(\mathbf{x}_{k-1:k}|z_{1:k}) \approx \sum_{n=1}^{N} w^i_k \delta(\mathbf{x}_{k-1:k} - \mathbf{x}^i_{k-1:k}) \tag{2}$$

where $\{\mathbf{x}^i_{k-1:k}, w^i_{1:k}\}_{n=1}^N$ represents the set of N particles and weights used to approximates the posterior distribution.

For second-order filtering, the state are initialised by sampling from a prior distribution $p([\mathbf{x}_1, \mathbf{x}_2])$. Particles at the subsequent k-th instant "alone", $p(\mathbf{x}_k)$, are sampled from the proposal distribution, $q(\mathbf{x}^i_k | \mathbf{x}^i_{k-2:k-1}, \mathbf{z}_k)$.

The sampled particles are then assigned importance weights. Following the derivations in Pan et al. [6], the conditional independence formulations and the distributions in Eqns 1 are used to assign the weights for the particles as,

$$w^i_k \approx w^i_{k-1} \frac{p(\mathbf{z}_k|\mathbf{x}_k) p(\mathbf{x}^i_k | \mathbf{x}^i_{k-2:k-1})}{q(\mathbf{x}^i_k | \mathbf{x}^i_{k-2:k-1}, \mathbf{z}_k)} \tag{3}$$

where $p(\mathbf{z}_k|\mathbf{x}_k)$ is the likelihood, $p(\mathbf{x}^i_k | \mathbf{x}^i_{k-2:k-1})$ is the transition probability and $q(\mathbf{x}^i_k | \mathbf{x}^i_{k-2:k-1}, \mathbf{z}_k)$ is the proposal distribution for importance sampling. The assigned weights are then normalized and used to estimate the state.

Note that the state, $\mathbf{x}_{k-1:k}$, can be recursively estimated at the k-1 and k-th instants together. In this joint estimation framework, state at frame k and k+1 would be estimated together, followed by joint estimation of the states at frames k+2 and k+3. However, by following this framework, the error could be accumulated during the frame propagation leading to degeneracy. The resampling in such circumstances might not address this limitation [6]. To avoid this issue, we adopt the algorithm proposed by Pan et al. [6]. Essentially, for a given state, $\mathbf{x}_{k-1:k}$, we only estimate the k-instant component in the state, which is given as, $\hat{\mathbf{x}}_k = \sum_{n=1}^{N} w^i_k \mathbf{x}^i_k$.

The k-th estimate is then used to update the k-th component in the joint state, given as $[\mathbf{x}_{k-1}, \hat{\mathbf{x}}_k]$. This updated state is then used within the proposal distribution to generate the particles for the k+1-th frame "alone". To account for particle impoverishment, we perform resampling if the effective sample size $N_f = 1/\sum_{n=1}^{N}(w_k^i)^2$ is less than a threshold. For detailed derivations we refer the readers to Pan et al. [6].

IV. Detailed Overview of the Algorithm

We next present a detailed overview of the learning and testing phases of the algorithm

A. Learning Phase

In the learning phase, we model the proposal distribution and the likelihood, individually, for different road scenes. Every s-th scene contains an training dataset with K_s training images $\mathbf{I}^s = \{I_k\}_{k=1}^{K_s}$ with corresponding human steering angle data $\mathbf{X}^s = \{\mathbf{x}_k\}_{k=1}^{K_s}$ in degrees obtained from the CANBUS. Gaussian random noise are added to the human steering angle data to account for the measurement uncertainty, generating $\mathbf{Z}^s$. We refer to this dataset as the *primary* training dataset, which is used to model the likelihood and the proposal distribution.

1) Likelihood: We model the likelihood using a deep learning-based regression framework. The regression framework utilizes the features extracted from the image to predict the image-based steering angle. This is given as,

$$\mathbf{z} = f_l^s(\lambda) \tag{4}$$

where λ represents the road features. We utilize the pre-trained *road* CNN model [16] to extract the features from the road scene image I. The road CNN architecture is similar to the Alexnet architecture [17]. The input is a RGB image with size $256 \times 256 \times 3$. There are five convolutional layers with 96 filters of size 11×11 ($C1$), 256 filters of size 5×5 ($C2$), 384 filters of size 3×3 ($C3$), 384 filters of size 3×3 ($C4$) and 256 filters of size 3×3 ($C5$), respectively. The first, second and fifth convolutional layer are followed by maximum pooling with filter size 3×3, ($P1, P2, P5$). Following the convolutional layers, there are two fully connected layers with 4096 neurons followed by a drop-out layer with drop-out ratio of 0.5. The final output layer contain 2 neurons with softmax function.

The road CNN network is a binary road scene classifier, which is pre-trained with the *Places* dataset [18] and road scene images. In this paper, λ, for a given image, are represented using the output feature maps of the pooling layer after the fifth convolutional layer ($P5$). The road scene feature is a 12544 dim feature vector corresponding to the output feature map of $P5$ contains 256 feature maps with size 7×7.

The extracted road scene features are then used to train the s-th extra trees regression model to predict the steering angle. The training pair consists of the input road features, $\Lambda^s = \{\lambda_k\}_{k=1}^{K_s}$, and output steering angle data with measurement noise, $\mathbf{Z}^s = \{\mathbf{z}_k\}_{k=1}^{K_s}$. Λ^s is computed from all the images in the *primary* training dataset. Following the training of the s-th extra trees regressor, we formulate the likelihood distribution as,

$$p(f_l^s(\lambda)|\mathbf{x}^s) = \mathbb{N}(\mathbf{x}_k^s; f_l(\lambda_k^s), \sigma_l) \tag{5}$$

where $f_l^s(\lambda_k)$ is the mean of the Gaussian distribution and σ_l represents the standard deviation. An overview of the likelihood-based regression framework is shown in Fig 4.

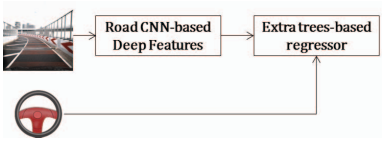


Fig. 4. CNN and extra trees-based regression framework

2) Proposal Distribution: A novel deep learning-based regression framework is used to model the proposal distribution (Eqn 3). This regression framework for the s-th road scene is given as,

$$\mathbf{x}_k^s = f_p^s(\mathbf{x}_{k-2:k-1}^s, \mathbf{z}_k^s) \tag{6}$$

where $[\mathbf{x}_{k-2:k-1}^s, \mathbf{z}_k^s]$ represent the regression input, γ^s, while $\mathbf{x}_k^s$ represents the regression output, θ^s. The output, θ^s, represents the human driving-based steering angle data at the k-th frame, $\mathbf{x}_k^s$. While, the corresponding input,γ, represents the following two components: 1) The human driving-based steering angle data, $\mathbf{x}_{k-2:k-1}^s$; 2) The image-based steering angle data obtained at the k-th frame, $\mathbf{z}_k^s$, using the CNN and extra trees-based regression framework. The proposed regression framework is trained using training pairs of inputs Γ^s and outputs Θ^s generated from primary training dataset. These training pairs are termed as the *primary* training pairs.

To learn the regression function $f_p^s()$ in Eqn 6, we adopt a LSTM-based mixture of experts (ME) framework [19]. In this work the ME framework is adopted and extended using deep learning. The proposed deep learning-based ME framework contains the following components: LSTM-based experts; LSTM-based gating network; Neural network-based refinement. An illustration of the proposed framework is given in Fig 2.

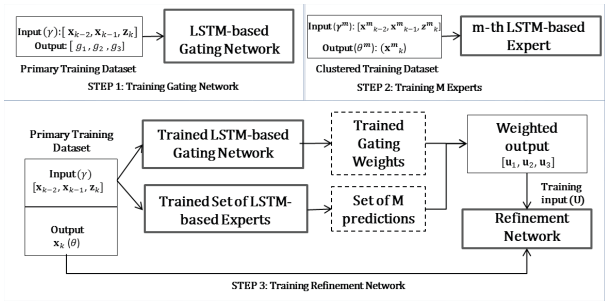


Fig. 5. An overview of the three training steps for the LSTM-based regression framework.

To train the proposed ME framework, the training pairs of inputs Γ^s and outputs Θ^s, are partitioned into M non-overlapping clusters. The cluster indices of the *primary* training pairs are represented using the cluster label m. Following clustering, each m-th cluster contains training pairs with regression inputs Γ^s_m and regression outputs Θ^s_m. These training pairs are termed as the *clustered* training pairs. We next explain the different components in the algorithm. An overview of the training steps are given in Fig 5.

a) LSTM-based experts: The LSTM experts, formulated as regression frameworks, are trained on the clustered dataset to predict the *preliminary* steering angle, $\bar{\theta}$. Each m-th LSTM-based expert contains 32 hidden units, followed by 1 output neuron. Each LSTM expert is trained using the corresponding *clustered* training pairs. The LSTM is trained individually with mean square error loss function and ADAM optimisation algorithm [20]. In this work, we manually partition each road scene into three clusters corresponding to *driving straight*, *turning right* and *turning left* scenario. Thus, three LSTM experts are employed to estimate the *preliminary* steering angle for every road scene. These three expert regions are illustrated in Fig 1.

b) LSTM-based gating network: The LSTM gating network generates a set of weights $\bar{\mathbf{g}}^s = \{\bar{g}^{s,m}\}_{m=1}^M$, which weights the *preliminary* steering angle. The set of weights satisfy the following assumption, $\sum_m \bar{g}^m = 1$. The LSTM-based gating network contains 32 hidden unit, followed by 3 output neurons with softmax activation function. The gating network is formulated as a multi-label classifier. The input to the gating network corresponds to γ^s in the *primary* training pair. The gating network's output is represented by a multi-label vector $\mathbf{g}^s = \{g^s_m\}_{m=1}^M$. The multi-label vectors are generated using the cluster indices of the *primary* training pairs. More specifically, for a given γ^s, we first retrieve their cluster index m. Subsequently, the m-th index g^s_m is set to 1, and the remaining indices are set to 0. The complete set of training pairs of the LSTM gating network are given as Γ^s and $\mathbf{G}^s$. The gating network is trained with the rmsprop optimisation algorithm and the categorical cross entropy loss function.

c) Refinement network: The gating network weights the *preliminary* steering angle, to generate the weighted outputs. The output of the m-th LSTM expert weighted by the gating network is given as $\{\mathbf{u}^{s,m} = \bar{g}^{s,m} \times \bar{\theta}^{s,m}\}_{m=1}^M$. In the standard ME framework, the weighted output are summed to generate the final output. However, in our proposed framework, the weighted output is given as an input to the refinement network. The refinement network is trained to *refine* the weighted output. The refinement network contains two layers, a fully connected layer with 32 neurons and an output layer with 1 neuron, which predicts the final steering angle ϑ.

The training data for the refinement network are generated from the *primary* training pairs. More specifically, Γ^s is given as an input to the gating network and experts to generate the refinement network training input, $\mathbf{U}^s$. The training output corresponds to Θ^s in the *primary* training pairs. The refinement network is trained with mean square error loss function and ADAM optimisation algorithm [20]. Following the training of our proposed regression framework, we formulate the proposal distribution as,

$$q(\mathbf{x}^s_k|\mathbf{x}^s_{k-2:k-1}, \mathbf{z}_k) = \mathbb{N}(\mathbf{x}^s_k; f^s_p(\mathbf{x}^s_{k-2:k-1}, \mathbf{z}^s_k), \sigma_p) \quad (7)$$

where $f^s_p(\mathbf{x}^s_{k-2:k-1}, \mathbf{z}^s_k)$, the mean of the Gaussian distribution, represents the output of the proposed regression framework, and σ_p represents the standard deviation. The different training steps are shown in Fig 2-b. The proposed ME framework accurately models the expert driver behavior for each road scene, compared to a single LSTM-based proposal distribution for each road scene. We demonstrate these observations in the experimental section (Sec V).

B. Testing Phase

Utilizing the modeled distributions, particle filtering is performed in the testing phase. We first retrieve the vehicle's heading angle and GPS latitude and longitude information, to estimate the road scene label s. Subsequently, the proposal distribution and likelihood formulation corresponding to the s-th scene is retrieved. The initial particle filtering steering angle states $[\mathbf{x}^s_1, \mathbf{x}^s_2]$ are sampled from a prior distribution $p([\mathbf{x_1}, \mathbf{x_2}])$. The prior distribution is given as,

$$p([\mathbf{x^s_1}, \mathbf{x^s_2}]) = \mathbb{N}([f^s_l(\lambda_1), f^s_l(\lambda_2)], \Sigma_i) \quad (8)$$

where $f^s_l(\lambda)$, the mean of the Gaussian distribution, represents the output of the s-th scene's extra trees-based regression framework. Σ_i represents the covariance matrix. Following the initialization, in the subsequent k+1 instants, samples are drawn from the s-th scene's proposal distribution (Eqn 7). The samples drawn are assigned importance weights using Eqn 3. The likelihood is calculated using Eqn 5, while the transition probability in Eqn 3 is represented by an online second order motion model, which is given as,

$$p(\mathbf{x}^s_k|\mathbf{x}^s_{k-2:k-1}) = \mathbb{N}(\mathbf{x}^s_k; \mathbf{x}^s_{k-1} + (\mathbf{x}^s_{k-2} - \mathbf{x}^s_{k-1}), \sigma_t) \quad (9)$$

where $\mathbf{x}^s_k = \mathbf{x}^s_{k-1} + (\mathbf{x}^s_{k-2} - \mathbf{x}^s_{k-1})$, the Gaussian mean, represents the online second order motion model, while σ_t represents the standard deviation. Given the weight assignment, the weights are normalized before estimating and updating the state $\hat{\mathbf{x}}^s_k$ in $[\mathbf{x}^{s,i}_{k-1}, \hat{\mathbf{x}}^s_k]$. Finally, we perform re-sampling, if the effective sample size N_f is below a predefined threshold.

V. Experimental Result

We validate our proposed algorithm using four different road scenes that are acquired using our experimental vehicle. We perform a comparative analysis with baseline algorithms and show that the performance of our proposed algorithm is better. We also perform a detailed parameter analysis of our algorithm. We implement our algorithm on a Linux machine with Nvidia Titan X graphics card using Python with Keras [21], Caffe [22] and scikit-library. We report an average computational time 80 ms per frame to estimate the steering angle.

A. Dataset and Algorithm Parameters

Dataset: The four different road scenes correspond to "highway" ($s1$), "urban road" ($s2$), "parking lot" ($s3$) and "highway ramp" ($s4$). The data from these road scenes contain video sequences with the human driving steering angle. Each scene is acquired from the same route indexed by the GPS data (latitude and longitude) and the vehicle's heading angle. The video sequences are partitioned into distinct training sequences and unseen test sequences. The $s1$ scene contains 13961 training data and 13911 testing data. The $s2$ contains 1801 training and 1600 testing data. The $s3$ scene contains 200 training and 150 testing data. The $s4$ scene contains 1200 training and 900 testing data. Amongst these road scenes, the $s3$, $s4$ and $s2$ are challenging road scenes with high non-linear or sudden changes in the steering angle motion and the corresponding image sequence. Amongst these the $s3$ is the most challenging. On the other hand, the $s1$ mainly contains linear motion with few turns. This can be observed in the results shown below, where $s1$ reports the least error. Example images from the dataset are shown in Fig 6.

Algorithm Parameters: In the learning phase for the likelihood formulation, 100 trees are used to train the extra trees regression model. In the likelihood formulation in Eqn 5, σ_l is set to 2.5. In the proposal distribution formulation, the LSTM expert, gating network and refinement network are trained with batch size 10 and 10000 iterations. The σ_p in Eqn 7 is set to 2.5. In the testing phase, we utilize 200 particles within the particle filtering framework. In the prior distribution in Eqn 8, Σ_i is defined as a diagonal matrix with value 0.1. While, the σ_t in the transition probability in Eqn 9 is also set to 2.5. All these parameters values are empirically set. In our experiments, we report the filtering accuracy by computing the Euclidean distance measure between the estimated steering angles and the ground truth steering angles in the test data over three trials. The ground truth steering angles correspond to the actual driving steering angle.

Fig. 6. Sample scenes from the four road scenes

B. Comparative Analysis

We perform a comparison of our proposed algorithm with the standard particle filter with first order motion model and adaptive velocity motion model. In the first order motion model (PF-0), we assume no motion, where $\mathbf{x}_k = \mathbf{x}_{k-1}$. In the adaptive velocity motion model based generic particle filter (PF-AV) [23], the motion model is given as $\mathbf{x}_k = \mathbf{x}_{k-1} + (\mathbf{x}_{k-2} - \mathbf{x}_{k-1})$. The predictions in both the particle filters are diffused using Gaussian distribution-based random noise. We also compare our algorithm with neural network-based steering angle prediction (FCN). In this approach, the neural network is used to model the regression-based motion prediction $f()$ in $q(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{x}_{k-2}, \mathbf{z}_k) = \mathbb{N}(\mathbf{x}_k; f(\mathbf{x}_{k-1}, \mathbf{x}_{k-2}, \mathbf{z}_k), \sigma)$. For all these baseline algorithms, we utilise our CNN and extra trees-based likelihood formulation to weight the particles and obtain the estimates. Finally, we also show the results of the steering angles predicted directly from the image using the CNN and extra trees-based regression framework (CNN-ET). This prediction approach is most similar to the comparative approaches in literature [2], [12]. The results are reported over the different datasets in Table I. The ground truth steering angle data and the estimated steering angle data over the different sequences are shown in Fig 7- 9.

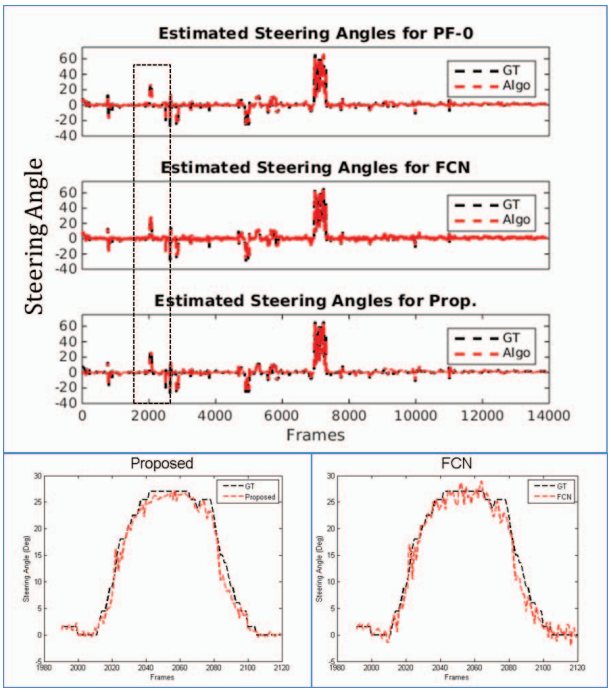


Fig. 7. Graph plotting the estimated steering algorithms for the ($s1$) sequence. In the bottom row, we highlight the errors in the highlighted region.

TABLE I

THE MEAN AND STD.DEV OF THE ESTIMATION ACCURACIES.

Alg.	s1	s2	s3	s4
Prop.	0.56 ± 0.08	1.55 ± 0.09	5.05 ± 0.29	2.19 ± 0.07
PF-0	0.74 ± 0.20	2.26 ± 0.23	13.54 ± 3.28	3.86 ± 0.39
PF-AV	0.87 ± 0.26	3.06 ± 0.29	10.46 ± 1.80	3.80 ± 0.77
FCN.	0.69 ± 0.32	1.7 ± 0.36	11.08 ± 0.37	3.13 ± 1.24
CNN-ET.	0.66 ± 0.10	2.05 ± 0.4	11.39 ± 1.23	3.62 ± 0.35

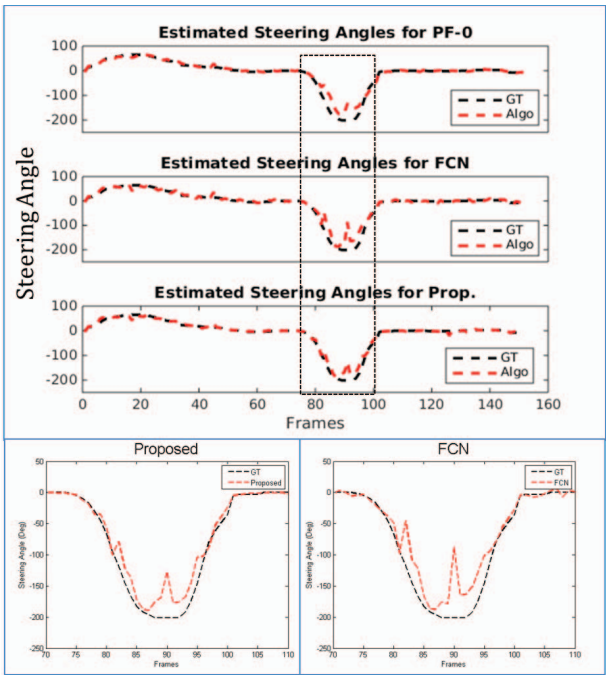


Fig. 8. Graph plotting the estimated steering algorithms for the $s3$ sequence. In the bottom row, we highlight the errors in the highlighted region.

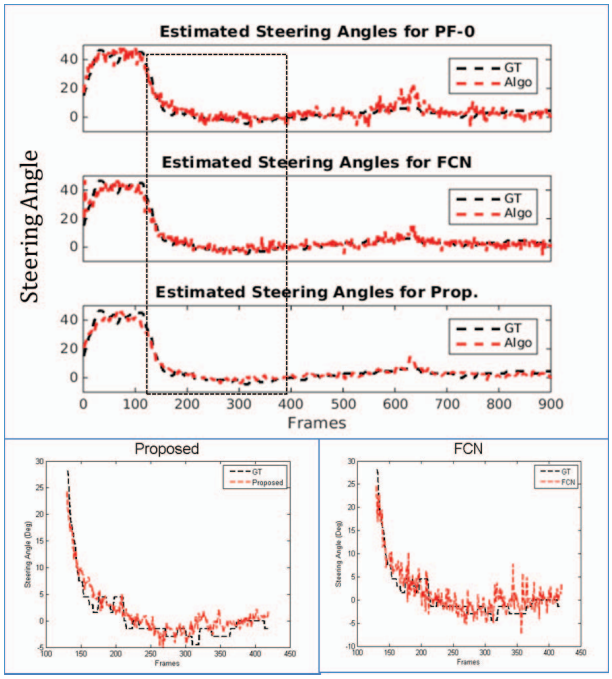


Fig. 9. Graph plotting the estimated steering algorithms for the $s4$ sequence. In the bottom row, we highlight the errors in the highlighted region.

C. Parameter Analysis

Motion Model: We evaluate the proposed generic particle filter by replacing the second order proposal and transition distribution (Eqn 3) with a first order LSTM-based proposal distribution (LSTM-1) and first order motion model-based transition distribution. The proposal distribution for LSTM-1 is given as $q(\mathbf{x}_k|\mathbf{x}_{k-1},\mathbf{z}_k) = \mathbb{N}(\mathbf{x}_k; f(\mathbf{x}_{k-1},\mathbf{z}_k),\sigma)$, while the transition distribution is given as, $p(\mathbf{x}_k|\mathbf{x}_{k-1}) = \mathbb{N}(\mathbf{x}_k;\mathbf{x}_{k-1},\sigma_t)$. As shown in Table II, the proposal algorithm performs better than the first order LSTM across the different scenes.

TABLE II

THE MEAN AND STD.DEV OF THE ESTIMATION ACCURACIES.

Alg.	s1	s2	s3	s4
Prop.	0.56 ± 0.08	1.55 ± 0.09	5.05 ± 0.29	2.19 ± 0.07
LSTM-1	0.61 ± 0.11	1.86 ± 0.15	8.55 ± 1.92	3.68 ± 0.25

LSTM-based Mixture of Experts: We evaluate the proposed LSTM-based mixture of experts framework, by replacing the proposal distribution in our algorithm with a single LSTM-based proposal distribution (LSTM-S). More specifically, a single second order LSTM is trained with the *primary* training pairs to generate function $f_p()$ used to formulate the proposal distribution in Eqn 7. This results obtained in Table III show the advantages of our proposed scheme.

TABLE III

THE MEAN AND STD.DEV OF THE ESTIMATION ACCURACIES.

Algo.	s1	s2	s3	s4
Prop.	0.56 ± 0.08	1.55 ± 0.09	5.05 ± 0.29	2.19 ± 0.07
LSTM-S	0.69 ± 0.11	2.9 ± 0.16	10.84 ± 1.74	3.01 ± 0.96

Refinement Network: We further evaluate the proposed LSTM-based regression framework. In this experiment, we remove the refinement network from the LSTM architecture in Sec IV-A.2 and directly sum the weighted output similar to the standard mixture-of-experts framework (LSTM-MOE). This results tabulated in Table IV show that refinement network improves the performance accuracy

TABLE IV

THE MEAN AND STD.DEV OF THE ESTIMATION ACCURACIES.

Algo.	s1	s2	s3	s4
Prop.	0.56 ± 0.08	1.55 ± 0.09	5.05 ± 0.29	2.19 ± 0.07
LSTM-MOE	0.89 ± 0.16	1.73 ± 0.17	11.88 ± 1.49	3.33 ± 0.26

Number of particles: We finally evaluate the particle filter with varying particles. As shown in Table V, the performance marginally improves with an increase in the computational complexity.

TABLE V

THE MEAN AND STD.DEV OF THE ESTIMATION ACCURACIES ALONG WITH AVERAGE TIME PER FRAME (MS).

Part.	s1	s2	s3	s4
200 (80)	0.56 ± 0.08	1.55 ± 0.09	5.05 ± 0.29	2.19 ± 0.07
500 (130)	0.54 ± 0.03	1.54 ± 0.04	4.27 ± 0.14	2.19 ± 0.06
1000 (250)	0.53 ± 0.03	1.53 ± 0.04	4.15 ± 0.11	2.18 ± 0.03
5000 (750)	0.52 ± 0.02	1.51 ± 0.04	4.14 ± 0.13	2.17 ± 0.02

Discussion: Based on our experimental results, we can see that the proposed algorithm performs better than the baseline

algorithm. Additionally, the parametric evaluation demonstrates the advantage of our algorithm. In this paper, owing to the limitations in computational memory, we limited the proposed framework to the second order particle. However, the proposed algorithm can be easily extended to incorporate higher order Markov chain. In the likelihood formulation, we utilize the extra trees regression framework for predicting the steering angles, as we obtained better results than the random forest regression framework (CNN-RF) as shown in Table VI. We modified the road CNN model classifier to function as an end-to-end regressor, and performed the fine-tuning with the *primary* training dataset. However, owing to the limited number of training samples, the steering angles were not accurately predicted from the image. Consequently, we do not use an end-to-end deep learning framework to predict the steering angles. A similar observation is reported in the work of John et al. [16].

We illustrate the differences in the proposal distribution formulation between the standard particle filter and our proposed framework in Fig 10. As discussed in Section I, the standard particle requires a wider distribution and higher number of particles to account for non-linear steering angle dynamics and sudden trajectory changes across different road scenes. The particles are sampled from a similar proposal distribution across the different scenes. On the other hand, the proposed particle filtering framework with scene-based models can track robustly with few particles, sampled from narrow accurate proposal distributions.

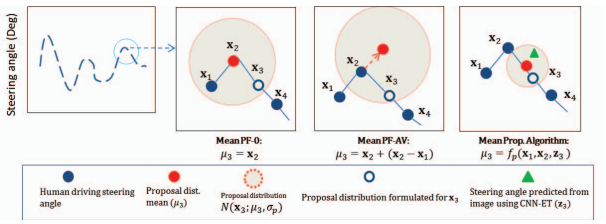


Fig. 10. Comparison of the proposal distribution formulation between the standard particle filter and the proposed learnt particle filter.

TABLE VI

THE MEAN AND STD.DEV OF THE ESTIMATION ACCURACIES.

Algo.	s1	s2	s3	s4
CNN-ET.	0.66 ± 0.10	2.05 ± 0.4	11.39 ± 1.23	3.62 ± 0.35
CNN-RF.	0.74 ± 0.12	2.07 ± 0.3	11.41 ± 1.32	3.68 ± 0.4

VI. Conclusion and Future Work

In this paper, we propose a novel scene-based particle filtering algorithm to estimate the steering angle for vision-based autonomous driving. Highly accurate proposal distributions and likelihoods are learnt, at the scene-level, for the particle filter using deep learning. A novel long short term memory network-mixture-of-expert learning framework is used to formulate the proposal distribution. We evaluate the sampled particles using a likelihood formulated with the convolutional neural network and extra trees regression. We validate our proposed algorithm with multiple driving sequences, and perform a comparative and parametric analysis. We show that the performance of our proposed algorithm is better, especially in challenging scenarios. In our future work, we will automatically cluster the training dataset and evaluate the algorithm with third and fourth-order models.

References

[1] A. Petrovskaya and S. Thrun, "Model based vehicle detection and tracking for autonomous urban driving," *Autonomous Robots*, vol. 26, no. 2, pp. 123–139, 2009.

[2] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to end learning for self-driving cars." *CoRR*, vol. abs/1604.07316, 2016.

[3] C. Chen, A. Seff, A. L. Kornhauser, and J. Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving." in *International Conference on Computer Vision*, 2015.

[4] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," *Transactions in Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.

[5] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. V. Gool, "Robust tracking-by-detection using a detector confidence particle filter," in *International Conference on Computer Vision*, 2009.

[6] P. Pan and D. Schonfeld, "Visual tracking using high-order particle filtering," *IEEE Signal Processing Letters*, vol. 18, no. 1, pp. 51–54, 2011.

[7] J. Ko and D. Fox, "Gp-bayesfilters: Bayesian filtering using gaussian process prediction and observation models," *Autonomous Robots*, vol. 27, no. 1, pp. 75–90, 2009.

[8] D. J. Fleet, "Motion models for people tracking," *Visual Analysis of Humans: Looking at People*, pp. 171–198, 2011.

[9] R. Urtasun, D. J. Fleet, and P. Fua, "3d people tracking with gaussian process dynamical models," in *Computer Vision and Pattern Recognition*, 2006.

[10] F. Liu, X. Lin, S. Li, and Y. Shi, "Multi-modal face tracking using bayesian network." in *Int. Workshop on Analysis and Modeling of Faces and Gestures*, 2003.

[11] Y. Lecun, U. Muller, E. Cosatto, and B. Flepp, "Off-road obstacle avoidance through end-to-end learning," in *Advances in Neural Information Processing Systems*, 2006.

[12] D. A. Pomerleau, "Neural network vision for robot driving," in *The Handbook of Brain Theory and Neural Networks*, 1996.

[13] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Computer Vision and Pattern Recognition*, 2016.

[15] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees." *Machine Learning.*, vol. 63, no. 1, pp. 3–42, 2006.

[16] V. John, Z. Liu, C. Guo, S. Mita, and K. Kidono, "Real-time lane estimation using deep features and extra trees regression." in *Pacific-Rim Symposium of Image and Video Technology*, 2015.

[17] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Neural Information Processing Systems*, 2012.

[18] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database." in *Advances in Neural Information Processing Systems.*, 2014.

[19] S. Masoudnia and R. Ebrahimpour, "Mixture of experts: a literature survey," *Artificial Intelligence Review*, vol. 42, no. 2, pp. 275–293, 2014.

[20] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[21] F. Chollet, "Keras," https://github.com/fchollet/keras, 2015.

[22] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guaddarrame, and T. Darrel, "Caffe: Convolutional architecture for fast feature embedding," in *arXiv preprint arXiv:1408.5093*, 2014.

[23] A. O. Balan, L. Sigal, and M. J. Black, "A quantitative evaluation of video-based 3D person tracking," in *International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, VS-PETS*, 2005.