Cooperative Perception of Connected Vehicles for Safety

April 2023 | Final Report

SAFE-D
SAFETY THROUGH DISRUPTION

Texas A&M Transportation Institute

San Diego State University
Leadership Starts Here

VT | Virginia Tech Transportation Institute
Virginia Tech

Disclaimer

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated in the interest of information exchange. The report is funded, partially or entirely, by a grant from the U.S. Department of Transportation's University Transportation Centers Program. However, the U.S. Government assumes no liability for the contents or use thereof.

TECHNICAL REPORT DOCUMENTATION PAGE

1. Report No. 05-115	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle Cooperative Perception of Connected Vehicles for Safety		5. Report Date April 2023 6. Performing Organization Code:
7. Author(s) Azim Eskandarian Dr. Prasenjit Ghorai Anshul Nayak		8. Performing Organization Report No. 05-115
9. Performing Organization Name and Address: Virginia Polytechnic Institute and State University Virginia Tech Transportation Institute 3500 Transportation Research Plaza Blacksburg, Virginia 24061 USA		10. Work Unit No. 11. Contract or Grant No. 69A3551747115/ Project 05-115
12. Sponsoring Agency Name and Address Office of the Secretary of Transportation (OST) U.S. Department of Transportation (US DOT) Department of Transportation, Washington, D.C., USA-20590		13. Type of Report and Period Final Research Report 14. Sponsoring Agency Code

15. Supplementary Notes
This project was funded by the Safety through Disruption (Safe-D) National University Transportation Center, a grant from the U.S. Department of Transportation – Office of the Assistant Secretary for Research and Technology, University Transportation Centers Program.

16. Abstract
In cooperative perception, reliably detecting surrounding objects and communicating the information between vehicles is necessary for safety. However, vehicle-to-vehicle transmission of huge datasets or images can be computationally expensive and often not feasible in real time. A robust approach to ensure cooperation involves relative pose estimation between two vehicles sharing a common field of view. Detecting the object and transferring its location information in real time is necessary when the object is not in the ego vehicle's field of view. In such scenarios, reliable and robust pose recovery of the object at each instant ensures the ego vehicle accurately estimates its trajectory. Once pose recovery is established, the object's location information can be obtained for future trajectory prediction. Deterministic predictions provide only point estimates of future states which is not trustworthy under dynamic traffic scenarios. Estimating the uncertainty associated with the predicted states with a certain level of confidence can lead to robust path planning. This study proposed quantifying this uncertainty during forecasting using stochastic approximation, which deterministic approaches fail to capture. The current method is simple and applies Bayesian approximation during inference to standard neural network architectures for estimating uncertainty. The predictions between the probabilistic neural network models were compared with the standard deterministic models. The results indicate that the mean predicted path of probabilistic models was closer to the ground truth when compared with the deterministic prediction. The study has been extended to multiple datasets, providing a comprehensive comparison for each model.

17. Key Words Cooperative perception, uncertainty quantification, trajectory prediction, Bayesian inference, relative pose	18. Distribution Statement No restrictions. This document is available to the public through the Safe-D National UTC website, as well as the following repositories: VTechWorks, The National Transportation Library, The Transportation Library, Volpe National Transportation Systems Center, Federal Highway Administration Research Library, and the National Technical Reports Library.

19. Security Classif. (Of this report) Unclassified	20. Security Classif. (Of this page) Unclassified	21. No. of Pages 20	22. Price $0

Form DOT F 1700.7 (8-72) Reproduction of completed page authorized

Abstract

In cooperative perception, reliably detecting surrounding objects and communicating the information between vehicles is necessary for safety. However, vehicle-to-vehicle transmission of huge datasets or images can be computationally expensive and often not feasible in real time. A robust approach to ensure such cooperation involves relative pose estimation between two vehicles sharing a common field of view. Detecting the object and transferring its location information in real time is necessary when the object is not in the ego vehicle's field of view. In such scenarios, reliable and robust pose recovery of the object at each instant ensures the ego vehicle accurately estimates its trajectory. Once pose recovery is established, the object's location information can be obtained for future trajectory prediction. Deterministic predictions provide only point estimates of future states, which is not trustworthy under dynamic traffic scenarios. Estimating the uncertainty associated with the predicted states with a certain level of confidence can lead to robust path planning. This study proposed quantifying this uncertainty during forecasting using stochastic approximation, which deterministic approaches fail to capture. The current method is simple and applies Bayesian approximation during inference to standard neural network architectures for estimating uncertainty. The predictions between the probabilistic neural network models were compared with the standard deterministic models. The results indicate that the mean predicted path of probabilistic models was closer to the ground truth when compared with the deterministic prediction. The study has been extended to multiple datasets, providing a comprehensive comparison for each model.

Acknowledgements

This project was funded by the Safety through Disruption (Safe-D) National University Transportation Center, a grant from the U.S. Department of Transportation – Office of the Assistant Secretary for Research and Technology, University Transportation Centers Program. Special thanks to Dr. Montasir Abbas, who agreed to serve as an external subject matter expert to review this report. We would also like to extend our gratitude to Dr. Zachary Doerzaph for his input and feedback on the work.

Table of Contents

List of Figures

List of Tables

Introduction

In connected vehicles, cooperative perception enables the ego vehicle to estimate its own pose (relative position and orientation) with another vehicle based on common visual features (Figure 1a). Cooperative perception involves map merging after pose estimation using sensors and is critical for object detection, trajectory prediction, and subsequent path planning in connected vehicles [1]. Range-based sensors such as lidar and radar can be used to generate spatial maps for pose estimation, but these sensors present significant drawbacks in terms of high cost and are often prone to interference. Meanwhile, vision-based sensors like monocular or stereo vision cameras have been used to test reliability and robustness in cooperative perception [2]. However, object localization and transfer of high-resolution images using vehicle-to-vehicle communication can be expensive and not feasible in real time. Moreover, in some scenarios, there might be a sudden loss of communication between two vehicles, which will affect the overall trajectory estimation in the ego vehicle coordinate frame. Hence, to ensure robust cooperative perception in short range, we first studied the relative pose estimation between two vehicles sharing a common field of view.

Estimation of relative pose involves feature detection and description, feature matching, outlier rejection, determination of fundamental and essential matrices, and pose recovery [3]. The most popular feature descriptor algorithm is SIFT, introduced by D.G. Lowe [4], which uses Difference of Gaussian (DoG) to detect features and is invariant to transformations like rotation and scale. A more robust and computationally inexpensive feature descriptor, the SURF [5] approach, relies on the determinant of Hessian matrix and has been used in relative pose estimation and localization of static objects [6]. However, binary feature descriptors are deemed faster and have recently seen significant applications in embedded robotics for real-time applications. ORB [7] is the most popular binary descriptor and is a blend of the FAST and BRIEF algorithms, where feature points are evaluated using the Harris corner method. Meanwhile, AKAZE [8], another binary detection description algorithm, is based on modified local binary difference and is computationally efficient. Comparative studies in the past have analyzed the efficacy of such methods with applications related to image stitching or image reconstruction [9] as well as feature-based ego motion estimation in visual odometry [10]. In the current study, however, we have compared the traditional descriptors like SIFT, SURF, and KAZE with binary descriptors like ORB, AKAZE, and BRISK to estimate the efficacy and robustness of each algorithm in relative pose estimation. In addition, we tested the efficacy of each algorithm based on varying levels of camera exposure and relative angular orientation between two vehicles. Relative pose estimation is crucial for estimating the detected object in the ego vehicle reference frame (Figure 1a).

Once the relative pose between vehicles is established, the ego vehicle can obtain the estimated trajectory of vulnerable road users (VRUs) like pedestrians and bicyclists. This will enable a self-driving car to use historical trajectory information to predict the future states of detected objects. For a self-driving car, awareness of the surrounding environment is crucial for correct and safe maneuvering [11] [12]. However, complex maneuvers require a trustworthy estimate of the future

states of VRUs [13]. Continuous progress has been made toward predicting the motion of VRUs with a certain degree of effectiveness [14] [15]. However, most prediction models are deterministic and provide only point estimates of the future states [16]. Such assumptions may be helpful for specific scenarios but, in a dynamic environment with multiple interactions, deterministic predictions can be overconfident and lack robustness. Since humans tend to change directions swiftly, deterministic predictions may fail to capture this randomness in pedestrian trajectory and thus ignore the uncertainty associated with the motion. A more robust approach will be to provide a probability distribution based on the likelihood of a pedestrian's location for each predicted state rather than a single point estimate. The uncertainty associated with predicted states can enable autonomous vehicles to achieve uncertainty-aware motion planning that will be more robust and trustworthy compared to planning based on deterministic prediction.

For instance, deterministic prediction outputs point estimates of future states. However, during long-term forecasts or multiple actor interactions, deterministic predictions deviate from the ground truth. In such a scenario, the planning algorithm may be uncertain about the future states, and the risk of collision will be substantially higher. Therefore, a robust and trustworthy planning algorithm requires improvement in confidence of the predicted states. This can be achieved through probabilistic prediction of states with associated uncertainty (Figure 1b).

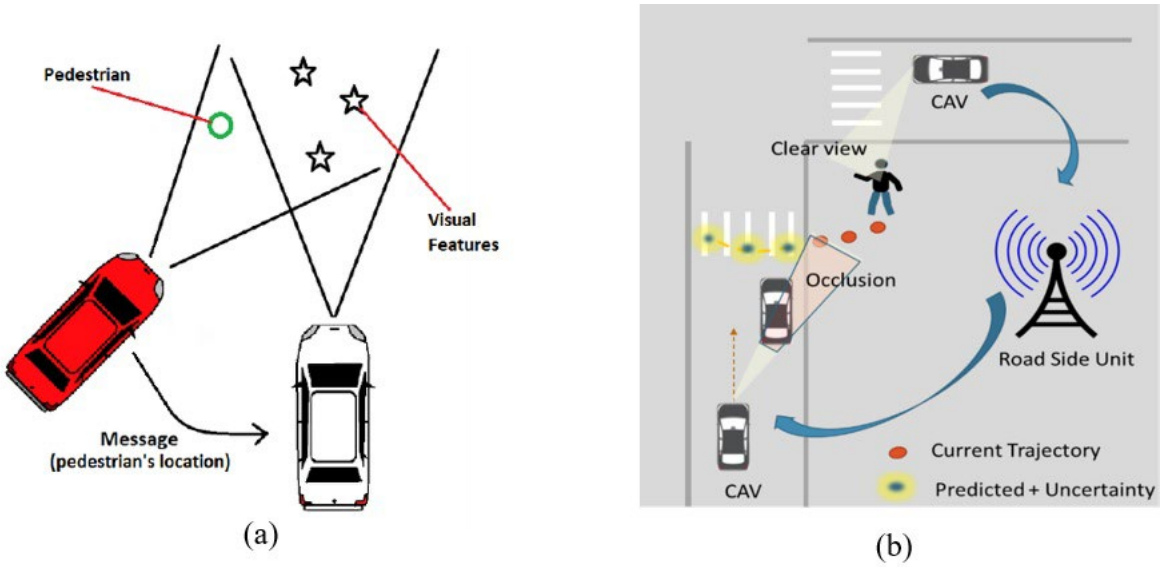


Figure 1. Diagrams. (a) Cooperative perception with relative pose estimation; (b) Trajectory prediction of road users with uncertainty quantification.

The risk-aware region around the future state captures all the probable locations the pedestrian can be present at a future time with certain confidence. It enables planning algorithms to either completely avoid the region or plan intelligently to execute uncertainty-aware motion planning that is more robust and less prone to collision [17]. Hence, we proposed the second study to investigate a risk-aware motion prediction model for pedestrians which can be further integrated with intelligent planning algorithms in future. The current model can perform long-term motion

prediction with high positional accuracy. The uncertainty in motion is estimated in the form of a distribution of trajectories with mean and variance such that the ground truth mostly lies within 95% of the confidence interval of the predictions.

Related Work

Cooperative Prediction of VRUs with Uncertainty Estimates

In recent years, deep neural networks (DNNs) have been used extensively for pedestrian trajectory forecasting. Most of the networks are based on recurrent neural network (RNN) architecture, which captures the temporal dynamics of sequential data [18]. Although RNNs should retain complete information of a temporal sequence, they fail practically to propagate long-term dependencies. Hence, long short-term memory (LSTM) [19] [20] approaches have been used for sequential prediction due to their improved capability in back propagating long-term error. Seminal works like scene-LSTM [21] and social-LSTM [22] have used LSTM network architecture to incorporate either scene information or social pooling between multiple pedestrians for enhanced trajectory forecasting. However, most prior work on LSTM focused on improving prediction accuracy and did not stress quantifying uncertainty. Recently, convolutional neural networks (CNNs) have been used for trajectory prediction. Specifically, a fast CNN-based model compared to a 1D convolutional model with LSTM showed improvement in temporal representation of trajectory [23]. Further, Simone et al. [24] elaborated on previous work by introducing novel preprocessing and data augmentation techniques to outperform other complex models. However, the tested models output deterministic estimates of future state and did not quantify uncertainty. Deterministic predictions may not be trustworthy in complex traffic scenarios. Hence, probabilistic inference of predictions can be useful in safety-critical tasks like collision avoidance [25] and uncertainty-aware motion planning.

Traditionally, the Kalman filter has been used for uncertainty estimation [26], but it fails to capture nonlinearities during long-term forecasts. Deep learning methods like Gaussian processes (GP) [27] and the Gaussian mixture model [28] have been used for probabilistic forecasting, too. For instance, a GP model with an unscented Kalman filter (UKF) was used for long-term obstacle prediction for collision avoidance [29]. However, GP kernels are infinite-dimensional and require extensive parameter-tuning for accurate prediction. Further, methods like stochastic reachability analysis have been performed by assigning probabilities to future states in the reachable set [30]. However, reachability analysis [31] often estimates all possible future states, resulting in a large infeasible set. Recently, nonlinear architectures like neural networks have been used for probabilistic trajectory forecasting. A CNN-based architecture was used to capture uncertainty, but for short-term forecasts only [32]. Further, an LSTM architecture could predict long-term probabilistic estimates of future states using an occupancy grid [33]. The problem was formulated as a multi-class classification problem using SoftMax to quantify probabilistic distribution of future states over the occupancy grid. However, SoftMax often leads to an overconfident prediction, and the model can be uncertain with high SoftMax output [34]. More sophisticated

models like Bayesian neural networks (BNNs) [35] have been used recently in an attempt to capture the uncertainty in time series data.

Typically, a BNN uses a prior distribution to formulate a posterior distribution, which is used to quantify uncertainty during prediction. The BNN can capture three types of uncertainties: (1) model uncertainty, or *epistemic uncertainty*, (2) inherent noise in data, also known as *aleatoric uncertainty*, and (3) model misspecification that occurs when testing data is different from the training dataset. In this report, our focus is to predict epistemic uncertainty associated with future trajectories. Although BNNs accurately capture uncertainty, the inference becomes challenging due to many model parameters. This often requires incorporating variational methods for Bayesian inference that can reduce computational costs. Recently, Monte Carlo (MC) dropout has been used as a variational method for uncertainty estimation in time series forecasting [36] without any significant change to the network architecture.

In this report, we elaborate upon this Bayesian approximation of using MC dropout for pedestrian uncertainty estimation. Our work primarily focuses on comparing the prediction performance of deterministic and probabilistic models. We quantified and compared the uncertainty during trajectory forecasting using three popular neural network architectures for time-series forecasting: LSTM, 1D CNN, and CNN-LSTM. Our novelty lies in showing the importance of probabilistic forecasting of future states over deterministic predictions and providing a detailed performance comparison between each probabilistic model and its associated deterministic model. Moreover, this work also provides a comprehensive performance comparison of both probabilistic and deterministic models on popular pedestrian datasets.

Method

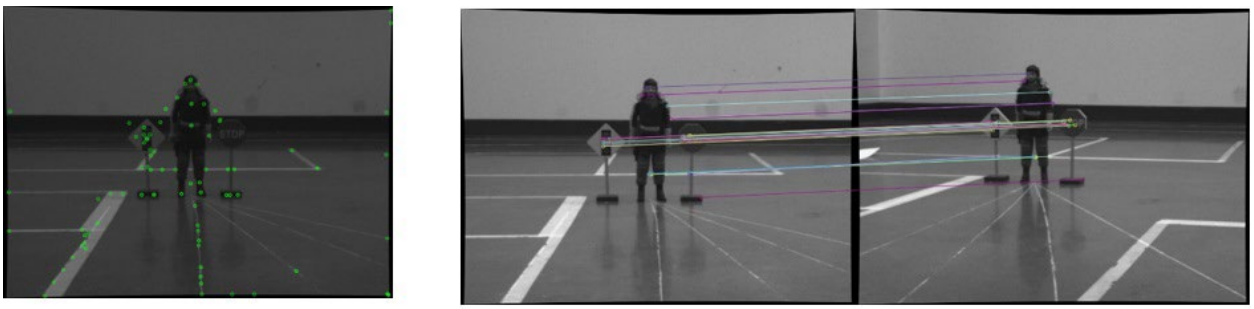
We have broadly classified our project into two aspects: cooperative perception and relative pose estimation between a pair of vehicles having vehicle-to-vehicle capability. The relative pose estimation allows data transfer from another vehicle's frame of reference to the ego vehicle's reference frame. The methods for pose estimation are listed below. Further, we compared the efficacy of popular feature description algorithms like SIFT, ORB, and KAZE in establishing accurate pose at different orientation angles and lighting conditions. Apart from perception, we also explored the effect of connected autonomy in the prediction pipeline. In cooperative prediction of future trajectory for VRUs, we developed a novel Bayesian approach to incorporate uncertainty estimation during trajectory prediction.

Cooperative Perception and Relative Pose Estimation

Overall estimation of relative pose between two vehicles can be broadly classified into five stages: (1) feature detection and description, (2) feature matching, (3) outlier rejection, (4) determination of fundamental and essential matrix, and (5) pose recovery through rotation (R) and translation (t). At each instant, the rotation and translation vector can be updated to establish cooperative

perception. In such a scenario, the location of vulnerable objects can be made available at the ego vehicle's coordinate frame through the established pose via matrix transformation.

Feature Detection and Description: During feature detection, we search for key points in an image, usually a corner, edge, blob, or flat surface (Figure 2a). Patches or points with large gradient intensity like corners of an image act as good descriptors, while a flat region is textureless and does not represent a good feature point. The classic Harris detector and the DoG approaches are some of the techniques we used for feature detection. Once the features are obtained, pixel information about a small patch around each feature is extracted to acquire the feature descriptor vector. Some popular feature descriptors are classified into string-based (e.g., SIFT, SURF, KAZE) or binary descriptors (e.g., ORB, BRISK). The string-based feature descriptor vector contains information about orientation of gradient histograms. Meanwhile, in binary descriptors, a filter with each cell having varying grayscale is convolved with the image ,and a logical operation is performed to obtain a binary key specific for that feature.



(a) (b)

Figure 2. Photos. (a) Feature descriptors identify the key points in an image; (b) Feature matching between key points for a pair of images as seen from different perspectives.

Feature Matching: Once the features and their descriptors have been established between a pair of images, feature matching is applied based on the vectorial distance between the descriptors (Figure 2b). Popular matching methods include the brute force method and the FLANN-based matcher. In the brute force method, L1 or L2 norm is used for calculating the vectorial distance for descriptors like SIFT, SURF, and KAZE, while Hamming distance is used for binary descriptors. However, such methods match both the true positive and the false negative features. Hence, we used the nearest neighbor distance ratio in SIFT to filter out preferred matches [4]. For our current study, we used a threshold ratio of 0.7.

Reject Outliers: We applied a RANSAC algorithm with 1,000 iterations and 99% confidence to the matches to reject any outliers. The final list of features represents the good descriptors and are inliers. The ratio of inliers to total detected features gives an estimation of the efficacy of the feature descriptors. Moreover, the absence of outliers is essential to determine the fundamental (F) and essential matrix (E).

Fundamental and Essential Matrix: Any point in the 3D world gets registered in the image plane through a simple transform: $x = PX$. Here, x is a homogeneous image coordinate and X represents the point in the 3D world. $P = K [R \mid t]$ stores the information about camera parameters using both the intrinsic camera calibration matrix (K) and extrinsic parameters like rotation (R) and translation (t), signifying the pose between a pair of cameras while sharing a common field of view.

When two cameras look at the same image from different perspective such that the 3D coordinate of the object is stored as homogeneous image coordinates x and x_0 in the two cameras, then the image points must satisfy:

$$x_0^T F x = 0 \tag{1}$$

where F is a rank deficient square matrix known as the fundamental matrix. The properties of the fundamental matrix can be obtained by direct linear transform based on the set of n matches. There are eight utmost correct matches necessary for determining F [37] in a calibrated camera. In turn, the intrinsic parameters of a camera are usually known, and the essential matrix (E) can be computed from the fundamental matrix as: $E = K_0^T F K$, where K is the upper triangular camera calibration matrix.

$$K = \begin{pmatrix} f_x & \beta & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2}$$

f_x and f_y represent the lens scaling factors along x and y directions, respectively, and β is the skewness factor, while x_0 and y_0 represent the principal coordinates on the image plane.

Relative Pose Estimation: E can be decomposed to obtain the pose between a pair of images and is primarily a product of the skew symmetric translation matrix and the rotation matrix.

$$E = [t]_x R \tag{3}$$

E has only five degrees of freedom: three for rotation and two for translation. The translation vector only shows the relative direction of one camera with respect to the other rather than absolute distance. Finally, the rotation matrix can be converted to corresponding Euler angles $[\psi, \theta, \varphi]$, which show the relative orientation without any ambiguity.

Cooperative Trajectory Prediction with Uncertainty Estimation

Deep learning architectures have been used extensively for prediction tasks. However, most networks are deterministic, generating point estimates without any confidence interval (Figure 3a). Conversely, a BNN produces uncertainty-aware predictions based on a stochastic network (Figure 3b). Although stochastic networks might provide better point estimates than other standard neural networks, their main aim is to provide trustworthy uncertainty estimates for predictions.

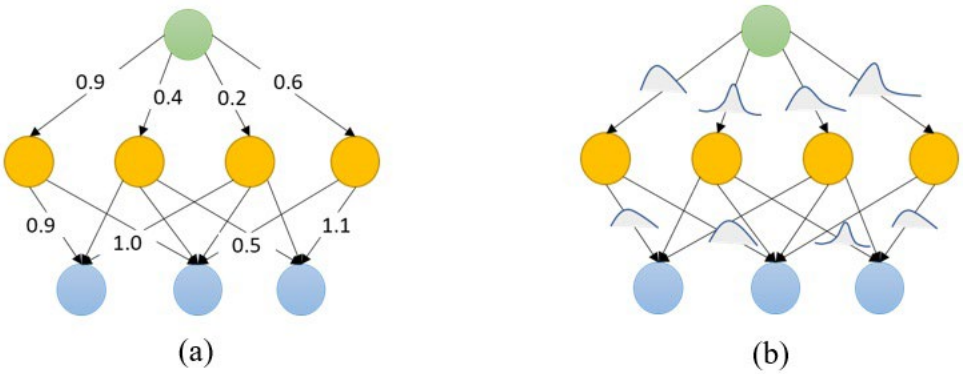


Figure 3. Diagrams. (a) Standard neural network with deterministic weights; (b) BNN with prior distribution over weights.

Typically, a BNN has probabilistic distribution over parameters like weights or activation functions. The parameters are learned using Bayesian inference as

$$P(\theta|D) = \frac{P(D|\theta)\,P(\theta)}{P(D)} \tag{4}$$

where $P(\theta|D)$ is the posterior (which implies the hypothesis) and θ is statistically updated based on data, D. $P(\theta|D)$ is the main aspect of a stochastic network and captures the learning of weights. The probability $P(D|\theta)$ is called likelihood, which is a marginal distribution of data provided by the weights. Meanwhile, $P(\theta)$ is the prior distribution over weights, while $P(D)$ is the evidence. The prior distribution samples from a stochastic distribution of weight with probability $P(\theta)$, unlike standard NNs, which have deterministic weights (Figure 3). As a result, the posterior distribution also becomes stochastic, and the metrics can be estimated by computing the mean of predicted distribution along with its associated variance. A BNN has several advantages over traditional neural networks, such as estimating uncertainty, decomposing the uncertainty into epistemic and aleatoric components, and integrating the knowledge of the prior distribution into the model. For instance, provided a set of training states $X = x_1, x_2, \ldots, x_T$ and output labels $Y = y_1, y_2, \ldots, y_T$ for a pedestrian, the distribution of its future predicted states, $y*$, can be computed by marginalizing the posterior, $p(\theta|\ D = X,Y)$, over some new input data point, $x*$:

$$p(y^*|x^*, X, Y) = \int_{\theta} p(y^*|x^*, \theta')\; p(\theta'|X, Y)\; d\theta' \tag{5}$$

θ' represents weights and $p(\theta')$ refers to the probability of sampling from prior weight distribution. During Bayesian prediction, computing the posterior $p(\theta'|\ X,Y)$ can be quite challenging, and many sampling algorithms like MC Markov Chain and variational inference [38] have been used to approximate the posterior. However, most sampling methods are either computationally expensive or introduce many parameters into the model, which stems into a complex problem. Further, a BNN incurs additional computation costs due to its nonlinearity. Hence, we used the MC dropout

method [34], which has been used to accurately approximate a BNN without significantly changing the model. The MC dropout method assumes that weights are stochastically dropped during inference. This process is then repeated for N forward passes to generate a random distribution of predicted values. Therefore, we applied MC dropout to different neural network architectures for uncertainty quantification during predictions.

Encoder-Decoder Model

We developed an encoder-decoder based simple LSTM architecture (Figure 4) to predict future trajectory of pedestrians in multiple time horizons. An encoder creates an embedding of essential features as a series of encoded space vectors, which the decoder uses to estimate outputs. Suppose $\{x_t\}$ represents the x-position up to T time steps as $\{x_1, x_2, \ldots, x_T\}$; then, the encoder embeds the data into an encoded space using a nonlinear function as $e = g(x)$. The decoder uses the encoded features to construct F forward time steps, $\{x_{T+1}, x_{T+2}, \ldots, x_{T+F}\}$. The current architecture has two LSTM cells for the encoder and one for the decoder (Figure 4). Through an ablation study, we found encoding both position and velocity of the pedestrian was beneficial for accurate prediction rather than encoding only position information. Hence, the input data for LSTM layers is a multivariate time series with four features $\{x, y, u, v\}$ corresponding to x and y position and velocity, respectively. The output of the neural network has similar features, and the predicted $\hat{x}$ and $\hat{y}$ show the pedestrian's future position. For regularization, we implemented dropout of weights with a certain probability, p, followed by tanh activation. The steps are repeated for each LSTM layer to create the encoded vector space, which carries the essential features of the training data. The decoder then uses the latent encoded vector to predict the future motion. The decoder architecture has a single LSTM layer that takes the encoded vector space as input. The LSTM layer is then followed by a dropout layer with linear activation to estimate the output.

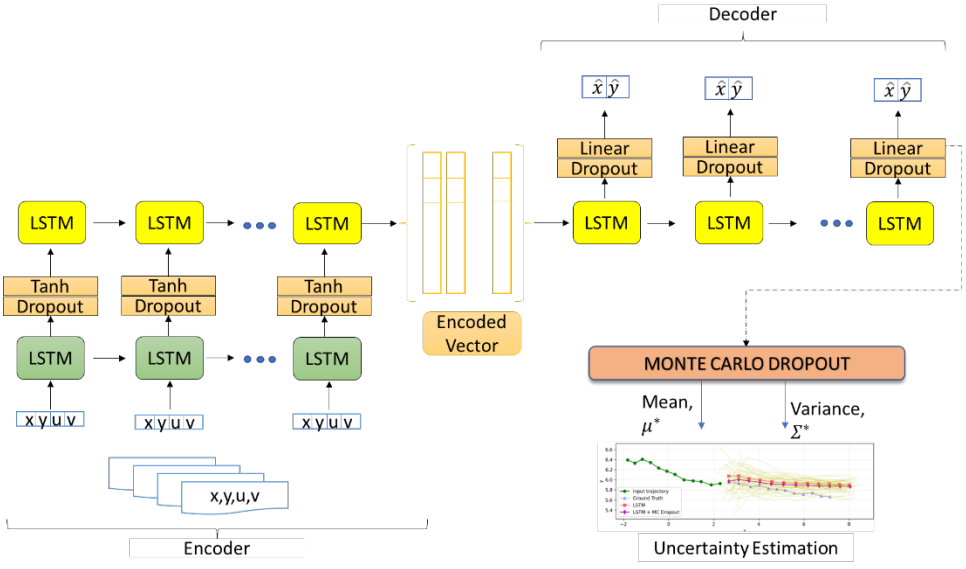


Figure 4. Flowchart. LSTM neural network architecture with dropout during inference for uncertainty quantification.

Convolutional Model

Recently, convolutional networks like 1D CNNs have been used for time series analysis [23]. The CNN model is a simple sequence-to-sequence architecture that uses convolutional layers to handle temporal representations. In the current 1D CNN model, we represent the past trajectory as a one-dimensional channel with four features, $\{x, y, u, v\}$. We constantly pad the input at each convolutional layer so that the number of features in both input and output remains the same. We have used "causal" padding for modeling temporal data, which eliminates the problem of auto correlation such that the output[t] does not depend upon input[t+1]. In total, three 1D convolutional layers with 128, 64, and 64 filters, respectively, have been used to extract features (Figure 5). We used a kernel size of 5, which showed better root mean-squared error in an ablation study with other odd kernel sizes {3,7}. A single dropout layer with probability p is applied after the first two convolutional layers to prevent overfitting. Further, a ReLU activation function has been applied to all the convolutional layers. We observed that a 1D Max Pool layer, followed by upsampling, performed better than global average pooling. Finally, for many-to-many predictions, a time-distributed dense layer has been used to predict multiple time steps while simultaneously generating a single trajectory. Further, MC dropout can be applied during inference to generate a distribution of trajectories.

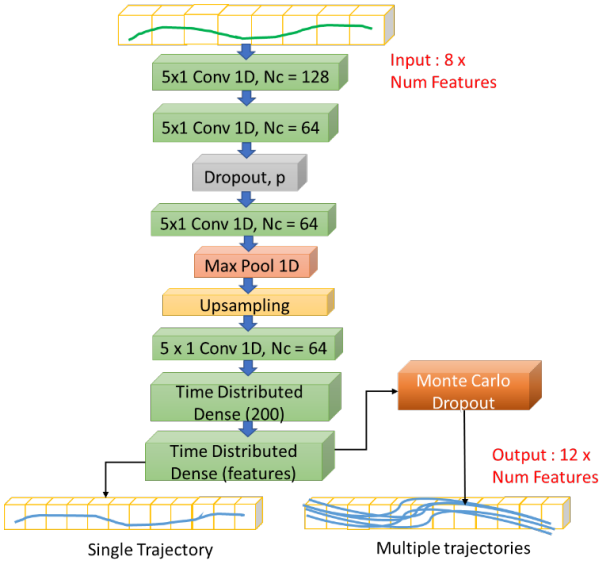


Figure 5. Flowchart. CNN architecture with dropout.

CNN-LSTM

A CNN-LSTM architecture is a hybrid network that uses both convolution and LSTM layers as an end-to-end model for time-series forecasting. Unlike the LSTM encoder-decoder model, one-dimensional convolutional layers are used for feature extraction to create an embedding rather than LSTM layers. The current model has two one-dimensional convolution layers with filters 128 and 64, respectively. Each convolution layer is followed by a dropout layer where weights are dropped with a certain probability, p, to prevent overfitting. In order to process the data into the format required by the LSTM, a Flatten layer is connected after convolution. Further, the LSTM layer

acts as a decoder and utilizes the features for prediction. Like previous models, a time-distributed layer at the end enables multi-step forecasting.

MC Dropout

Epistemic uncertainty during forecasting problems can be estimated from the mean and variance of the marginal distribution, $p(y^*|x^*, X,Y)$. Recall that X and Y represent the training input and output samples while y^* represents the predicted states for some new test sample, x^*. The marginal distribution is sampled from posterior $p(\theta| X,Y)$ based on test input x^*. The sampling process can be challenging and computationally expensive due to nonlinearity in BNNs. Therefore, MC dropout can be used as a variational approximation to such BNNs [34] without significantly modifying the existing network architecture. Hence, model uncertainty, known as epistemic uncertainty, can be estimated with ease without any additional computational cost, unlike other inference methods.

As discussed, MC dropout has been applied to each network architecture during inference to quantify uncertainty (Figure 3 and Figure 4). The MC dropout model applies stochastic dropout during test time to the neural network. Thus, for any new set of input x^*, we compute the inference by random dropout at each layer of the model. The probability of dropout is set as p and the inference model is run N times to obtain a set of outputs $\{y_1^*, y_2^*, \ldots, y_N^*\}$. We can then estimate the mean, $\overline{y^*}$, and variance, Σ_{y^*}, of the marginal distribution where the variance indicates model uncertainty (Equation 6).

$$\overline{y^*} = \frac{1}{N}\sum_{n=1}^{N} y_{(n)}^* \tag{6}$$

$$\Sigma_{y^*} = \frac{1}{N}\sum_{n=1}^{N}\left(y_{(n)}^* - \overline{y^*}\right)^2$$

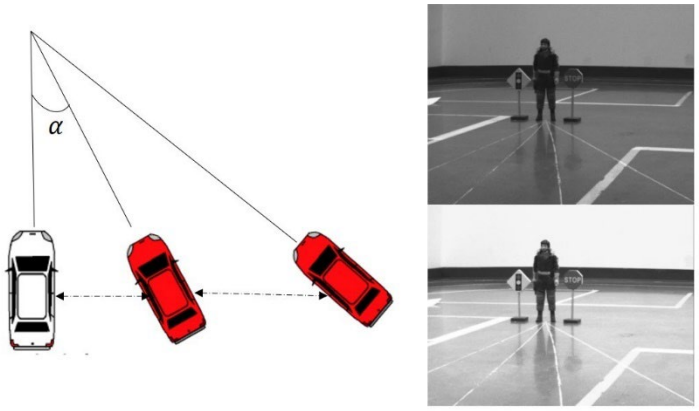
Results and Discussion

Cooperative Perception and Relative Pose Estimation

In relative localization, information obtained in another vehicle's frame of reference can be transferred to the receiver's reference frame by accurately establishing relative pose between the two when sharing a common field of view. For the current study, we have considered a vision-based monocular camera mounted on Wi-Fi bots to capture images with varying degrees of angular orientation and exposure as parameters for comparing different feature matching algorithms.

In the current study, three angles of relative orientation between the bots signifying yaw were selected. Figure 6 shows the schematic for relative angular orientation where the other vehicle (white) was kept fixed at 0° while the orientation of the second bot was subsequently varied from

10° to 30° such that the second bot translated unidirectionally away from the first bot while maintaining the relative orientation (Figure 6a). Moreover, pose estimation and feature matching performance under different illumination is critical in determining the efficacy and robustness of various feature descriptors. Hence, the current study considered three levels of exposure settings—500, 1,000, and 2,000 lumens. Apart from understanding the effect of relative angle and exposure on feature description and matching for a pair of images, our focus was to compare the efficacy and robustness of various feature descriptor algorithms on the above parameters. We used the PyCharm platform with OpenCV 3.3 for the current study, and Table 1 shows feature descriptors with their corresponding parameters. To have a fair comparison for computational cost, the parameters for each algorithm were selected so the size of descriptors is approximately the same.



(a) Relative Angle (b) Exposure

Figure 6. Diagram and Photos. Comparison of feature descriptor algorithms with varying (a) relative angle (10°, 20°, 30°) between Wi-Fi bots and (b) exposure level: top -1,000, bottom -2,000 lumens.

Table 1. Parameters For Proposed Study

Feature Descriptor	OpenCV Parameters
SIFT	cv2.SIFT (CT = 0.04, σ = 1.6)
KAZE	cv2.KAZE (Octave Layers = 3)
ORB (1,000)	Cv2.ORB (*n* features = 1,000)
BRISK	cv2.BRISK()
AKAZE	cv2.AKAZE (Octave Layers = 3)

For the current study on relative orientation, we procured a vision-based monocular camera mounted on Wi-Fi bots to capture images. A pair of images must share a common field of view for pose recovery. Once the key points are detected and features are matched, outliers are rejected through RANSAC to retain only a few good matches necessary for estimating fundamental and essential matrices. Eventually, the method must show repeatability while recovering pose and be robust to image transformations like scale, rotation, lighting, and affine transforms. The performance of considered feature descriptor algorithms was gauged per the indicators below:

- *Total Features*: Individual key points detected on each image.
- *Feature Matches*: Total matches between two images including outliers.
- *Rejection Ratio* $= \frac{Total\ Matches}{Outliers}$
- *Matching Time*: Total matching time between a pair of images signifying computational cost.

Table 2 provides comparative performance among various feature descriptors at a specific orientation of 20° and an exposure setting of 1,000 lumens. ORB detects almost double the maximum features detected by any other method. ORB is closely followed by SIFT and KAZE, while other binary descriptors like BRISK and AKAZE detected fewer features. Although ORB detects the maximum features, the total matches between the pair of images for SIFT and ORB (1,000) are comparable, closely followed by AKAZE. However, total matches do not fully signify the efficacy of pose recovery, as some of the matches can be inaccurate and act as outliers. Hence, an approximate estimate of robustness of feature matching can be related to rejection ratio, signifying good matches or inliers. Although SIFT has the highest total matches, the rejection ratio is also high, around 16%, meaning one sixth of total matches are outliers. However, the binary descriptors like ORB and AKAZE have a very low rejection rate compared to SIFT and KAZE, signifying most of the found matches are good matches. The presence of outliers is important because it can significantly alter the fundamental and essential matrices necessary for correct pose recovery. Finally, the feature matching time signifies the computational time each method needs to detect, match, and reject outliers for computation of rotation and translation matrices. It is observed that KAZE takes maximum time while BRISK takes the least time and is computationally inexpensive. The current results show only a preliminary performance estimation of all the feature descriptors for a specific exposure and angular orientation.

Table 2. Performance Comparison for Feature Matching Algorithms at Exposure = 1,000 and Relative Orientation = 20°

Feature Descriptor Algorithm	Total Features Detection – Image 1	Total Features Detection – Image 2	Total Matched Features	Outliers Rejected	Outlier Total	Feature Matching Time (s)
SIFT	294	366	73	12	0.165	0.232
KAZE	244	259	43	7	0.163	0.552
ORB	492	500	43	6	0.1395	0.304
ORB (1,000)	746	795	71	4	0.056	0.285
BRISK	195	224	39	3	0.0769	0.071
AKAZE	196	200	68	7	0.103	0.324

Cooperative Prediction of Future Trajectory with Uncertainty Quantification

In this section, we discuss the uncertainty estimation and performance metrics of pedestrian trajectory prediction. Initially, we quantified uncertainty in prediction using the probabilistic models based on the ETH dataset [39]. We also provide a confident estimation of our predictions with respect to the ground truth. Finally, we provide a comprehensive comparison of performance metrics among all the models across multiple datasets.

We have discussed the datasets, data augmentation, implementation details for each network, and the performance metrics. Following common practice from literature [22], we trained our models on publicly available pedestrian datasets. The two most popular datasets are the ETH dataset [39], which contains the ETH and HOTEL scene, and the UCY dataset [40], which contains the UNIV, ZARA1, and ZARA2 scenes. In order to draw parallels with past works [23], we studied 8 (3.2 secs) historical steps to predict 12 (4.8 secs) steps into the future. Further, we extended our study to predict multiple time horizons with long-term forecasts up to 8 seconds into the future.

Data Augmentation

Initially, we trained our model on the ETH dataset only, which contains approximately 420 pedestrian trajectories under varied crowd settings. However, a small number of trajectories is insufficient for training. Therefore, we performed data augmentation using the Takens Embedding theorem [41]. We used a sliding window of $T = 1$ step to generate multiple small trajectories out of a single large trajectory. For instance, a pedestrian's trajectory of 29 steps will result in 10 small $\{x, y\}$ trajectory pairs of 20 steps each if past trajectory information of 8 steps is used for predicting 12 steps into the future. In total, we constructed 1,597 multivariate time series sequences that we split into 1,260 training and 337 testing sequences for the ETH HOTEL dataset. Further, we will also show results for the ZARA1 and ZARA2 datasets to provide a comprehensive comparison of uncertainty quantification across all the models.

Implementation Details

All the neural networks are trained end-to-end using TensorFlow. Adam optimizer with a learning rate of $1e-3$ was used to compute the mean-squared error loss. Each model was trained for 100 epochs with a batch size of 32. The LSTM encoder-decoder model was trained with tanh activation while the other two models had ReLU activation. Ten percent of the training data was used for validation. The mean-squared error was monitored on the validation loss with callback functions like EarlyStopping and ReduceLROnPlateau. The model was compiled and fitted using training and test data.

Performance Metrics

The trained model was then used to predict the future position of the pedestrian. By default, the model predicts deterministic future states. However, probabilistic predictions can be inferred using MC dropout. We can run the stochastic inference using MC dropout repeatedly to generate a distribution of trajectories (Figure 7a). The mean of the distribution represents the predicted path,

while the associated variance quantifies uncertainty. We adopted widely used performance metrics [22], namely average displacement error (ADE) and final displacement error (FDE), for prediction comparison between the deterministic and probabilistic models.

- Average Displacement Error, ADE $= \frac{1}{T}\sum_{t=t_0}^{t_f}\left\|\hat{Y}_{(t)} - Y_{(t)}\right\|$. The mean of Euclidean distance over all estimated points of a trajectory with the ground truth. Here, $\hat{Y}_{(t)}$ is the predicted location at timestamp t and $Y_{(t)}$ is the ground truth position.
- Final Displacement Error, FDE $= \left\|\hat{Y}_{(t)} - Y_{(t)}\right\|$. The mean of Euclidean distance between the predicted destination and true destination across all trajectories.

For predicting uncertainty and evaluating its trustworthiness, we generated a distribution of trajectories using MC dropout. We have shown the uncertainty estimation of a single pedestrian trajectory from the ETH dataset using the 1D CNN network (Figure 7). For the current scenario, we predicted 12 steps or 4.8 seconds into the future based on 3.2 seconds of past trajectory data. A single trajectory (red) is generated by deterministic prediction and provides point estimates of future states. However, we generated probabilistic predictions by applying MC dropout to each neural network model. For instance, the 1D CNN with MC dropout model was sampled $N = 30$ times during inference with a stochastic dropout, $p = 0.2$ (20% of the weights are randomly dropped) to generate a distribution of N different trajectories (Figure 7a). The mean and variance of the distribution quantifies the uncertainty during trajectory prediction.

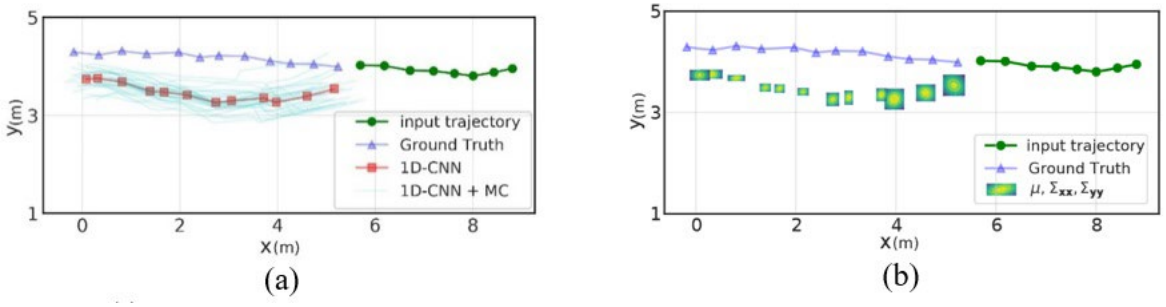


Figure 7. Graphs. (a) Trajectory prediction using MC dropout with the 1D CNN network; (b) Bivariate Gaussian distribution of future state uncertainty. Pedestrians move from left to right.

The predicted trajectory distribution (Figure 7a) shows the pedestrian's motion along both x and y directions, with predominant motion along x. Therefore, we needed to quantify the mean and variance along both the directions and treat the predicted trajectory cluster as a bivariate distribution. At each prediction step, the trajectories can be represented as a cluster of N points distributed on the x-y domain. Assuming Gaussian distribution for each point cluster, we can then estimate the mean and covariance showing the associated uncertainty at each step (Figure 7b). The Gaussian representation of the predicted states thus shows the future states with associated uncertainty. Further, we can compute the mean and covariance of this bivariate Gaussian distribution as:

$$\begin{bmatrix} X \\ y \end{bmatrix} \sim N\left(\begin{bmatrix} X \\ y \end{bmatrix}, \begin{bmatrix} \Sigma_{XX} & \Sigma_{XY} \\ \Sigma_{YX} & \Sigma_{YY} \end{bmatrix}\right) \quad (7)$$

Here, $\{\mu_x, \mu_y\}$ and $\{\Sigma_{xx}, \Sigma_{xy}, \Sigma_{yy}\}$ represent the mean and covariance of the pedestrian's movement along the *x*-*y* domain, respectively. Each covariance term shows the correlation of motion along one direction with respect to another. However, instead of variance Σ, only the standard deviations $\sigma = \sqrt{\Sigma}$ along *x* and *y* were considered to quantify the uncertainty during trajectory prediction.

We compared the estimated uncertainty during prediction of each probabilistic model with the ground truth, ▲ (Figure 8). We have shown the mean predicted path with two standard deviations (2σ) along *x* and *y* directions to quantify uncertainty. The model takes 8 input states (•, green dot) to predict 12 states into the future. One can visualize the mean of the predicted distribution (•, blue dot) along with the standard deviations (+) along *x* and *y* for a single trajectory chosen from the ETH dataset (Figure 8). On inspection, it seems the uncertainty predicted by the CNN-LSTM model grows with the prediction horizon while both the LSTM and 1D CNN models provide conservative probabilistic estimates that neither decrease nor increase with time. Further, it appears that the FDE is minimum for 1D CNN while it is maximum for LSTM. However, to obtain conclusive evidence on predictive accuracy of each model, we need to consider the performance metrics of all possible trajectories and estimate whether the ground truth lies within the 95% (2σ) confidence interval of our predictions.

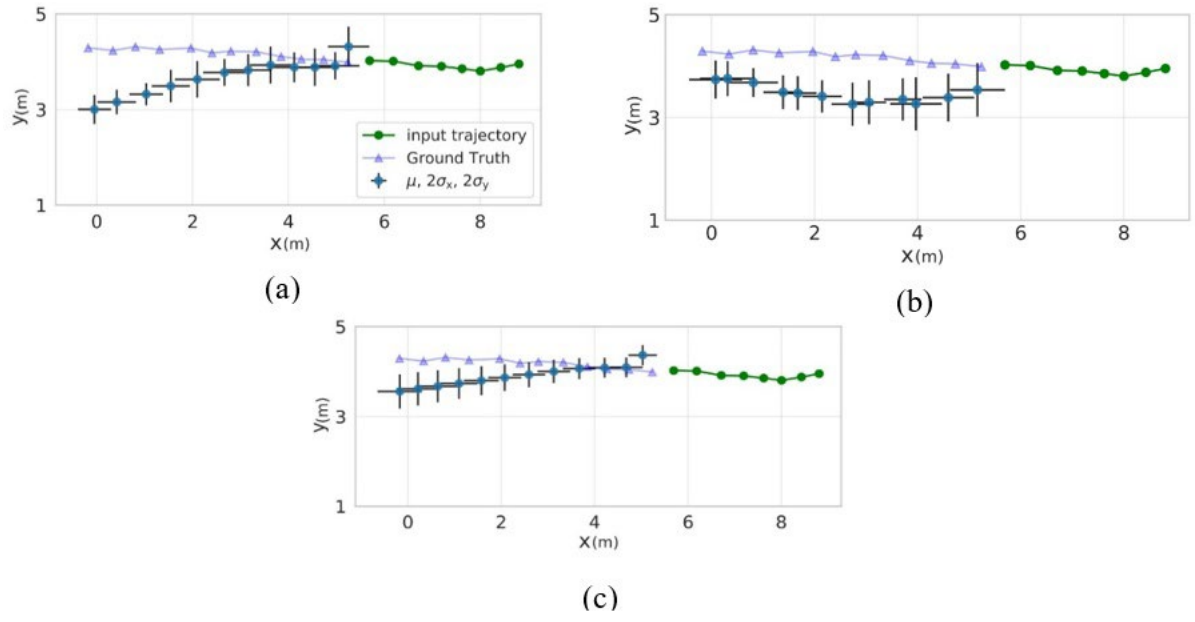


Figure 8. Graphs. Pedestrian future trajectory prediction (12 steps, black) with 95% confidence interval based on past input trajectory (8 steps, green) for (a) LSTM, (b) 1D CNN, and (c) CNN-LSTM.

Confidence Score

We defined a parameter known as confidence score (*CS*) to check whether the ground truth $\{x_{true}, y_{true}\}$ lies within two standard deviations (2σ) of our predicted distribution.

$$CS_x = \frac{|x_{true} - \mu_x|_{i=1,..F} < 2\sigma_x}{F} \text{ x } 100$$
$$CS_y = \frac{|y_{true} - \mu_y|_{i=1,..F} < 2\sigma_y}{F} \text{ x } 100 \quad (8)$$

Here, F represents the number of predicted states in the future. We predict the confidence score along x and y for each test trajectory and then take the mean across all trajectories to obtain a single confidence score for that prediction horizon. Further, we also show the variation of confidence score with prediction time horizon T_F = 3.2, 4.8, 6.4, and 8 seconds into the future.

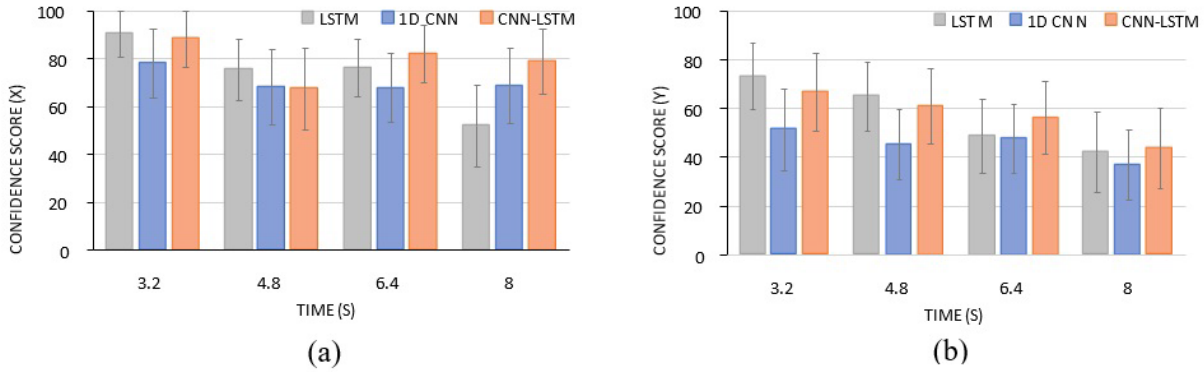


Figure 9. Bar Charts. Variation of confidence score with prediction horizon T_F along (a) x and (b) y.

The LSTM model with MC dropout (gray) provides a high mean confidence score over other models for forecasts up to 4.8 seconds. For instance, at T_F = 3.2 seconds, the LSTM model has a mean $CS_x \approx 90\%$, which signifies the percentage of ground truth that lies within the 2σ confidence interval of the predictions (Figure 9a). However, for long-term forecasts beyond 4.8 seconds, the CNN-LSTM model (orange) outperforms the other two models with a mean $CS_x \approx 80\%$. A similar trend can be observed for CS_y for both short-term and long-term forecasts (Figure 9b). However, the confidence score along y is less than x for each model. This shows the model is more confident and captures uncertainty effectively along the direction of predominant motion and is suitable for long-term linear motions. However, when the motion is significantly small along any direction, the model is less confident, and uncertainty is not captured accurately. Among all models, the 1D CNN probabilistic model (blue) has the lowest confidence score along both x and y across all time horizons. Our results thus indicate that the LSTM model (gray) performs better for short-term probabilistic predictions, while the CNN-LSTM model (light maroon) has better long-term probabilistic accuracy. Further, the confidence score along y gradually decreased with the prediction horizon for each model. This shows the positional accuracy along y for predicted states becomes more uncertain with the increase in prediction horizon. To understand how prediction horizon affects uncertainty estimates, we studied its effect on performance metrics like ADE and FDE. Further, we compared each probabilistic model against its deterministic prediction. We also studied the effect of stochastic dropout probability p on performance metrics.

Quantitative Evaluation

In this section, we evaluate the current probabilistic methods and compare them to state-of-the-art pedestrian forecasting models across the ETH [28] and UCY [29] datasets. Evaluation of error metrics ADE and FDE for the state-of-the-art models is based on the best-of-20 protocol from a list of 20 trajectories. As a preferred practice in literature, we observed 8 steps to predict 12 steps into the future. Table 3 summarizes the performance metrics, ADE/FDE, based on our predictions for the current models. In total, there are three probabilistic models trained with MC dropout and three standard neural network architectures that generate single deterministic predictions. All probabilistic models with MC dropout are inferred using dropout probability $p = 0.2$.

Table 3. Quantitative ADE/FDE For Predicting 12 Future Steps Given 8 Previous Time Steps

Models	ETH	HOTEL	ZARA1	ZARA2	UNIV	AVERAGE
S-LSTM [12]	1.09/2.35	0.79/1.76	0.47/1.00	0.56/1.17	0.67/1.40	0.72/1.54
SGAN [32]	0.87/1.62	0.67/1.37	0.35/0.68	0.42/0.84	0.76/1.52	0.61/1.21
Sophie [33]	0.70/1.43	0.76/1.67	0.30/0.63	0.38/0.78	0.54/1.24	0.54/1.15
Social-BiGAT [34]	0.69/1.29	0.49/1.01	**0.30/0.62**	**0.36/0.75**	0.55/1.32	0.48/1.00
LSTM	0.54/0.94	0.33/0.46	0.51/0.96	0.53/0.96	0.75/0.93	0.53/0.85
1D CNN	0.71/0.90	0.71/1.04	0.75/1.02	0.86/1.16	0.95/1.24	0.79/1.07
CNN-LSTM	0.68/1.11	0.98/1.29	0.73/0.99	0.95/1.27	0.87/1.11	0.84/1.15
LSTM + MC	0.55/0.94	**0.32/0.45**	0.51/0.96	0.54/0.96	0.59/0.84	**0.50/0.83**
1D CNN + MC	0.69/0.84	0.58/0.79	0.73/0.99	0.85/1.15	0.71/0.85	0.71/0.92
CNN-LSTM + MC	**0.48/0.82**	**0.3/0.48**	0.50/0.83	0.77/1.12	**0.53/0.86**	**0.51/0.82**

Lower ADE/FDE is better and represented in bold.

We compared our model against the following state-of-the-art-models:

- S-LSTM [12]: Each person is modeled using LSTM with a social pooling layer.
- SGAN [32]: A generative model that uses social information.
- Sophie [33]: A generative model that uses both social and image information.
- Social-BiGAT [34]: Attention-based generative network for multi-modal forecasting.

The current results indicate that the novel CNN-LSTM model with MC dropout has the lowest error among the considered probabilistic and deterministic networks for the ETH and HOTEL scenes. The mean predicted path has a minimum ADE/FDE (0.3/0.48) for the HOTEL scene. We speculate that the CNN captures features more efficiently than a standard LSTM encoder.

Moreover, the probabilistic CNN-LSTM model outperforms every other generative baseline model across the ETH and HOTEL scenes, which shows inclusion of social information like agent interaction may not necessarily produce better performance metrics. As suggested in the literature [14], the simple linear velocity model [32] outperforms many state-of-the-art architectures. Moreover, the consideration of pedestrian velocity information during training improved the performance of current models without the consideration of social information.

Although probabilistic CNN-LSTM achieves state-of-the art performance for the ETH and HOTEL scenes, the same is not true across ZARA scenes. In the ETH and HOTEL scenes, the pedestrian density is low, and the scenario is simple. In these situations, the inclusion of only position and velocity information produced better predictions. Meanwhile, the pedestrian density is higher and includes significant pedestrian interactions in the ZARA scenes. For those scenes, the current proposed model performs worse than generative models. Nonetheless, the overall average error for probabilistic LSTM and CNN-LSTM networks was minimum across all datasets. Although the social-BiGAT had a comparable ADE, the final FDE was considerably larger compared to current probabilistic models. Interestingly, the probabilistic models had a smaller FDE over all state-of-the-art generative models. This observation may be due to the averaging of multiple trajectories generated from probabilistic models. Overall, the current probabilistic models outperform the state-of-the-art generative models across the ETH dataset while performing worse for the ZARA scenes. This finding shows that it is not always necessary to consider contextual or social information to accurately predict trajectories. Furthermore, the current model is much faster and easier for real-time implementation, as it does not rely on social interaction or contextual cues.

Conclusions and Recommendations

This report presents a two-fold approach to integrating cooperative perception and cooperative prediction. In cooperative perception, we compared the efficacy of different feature descriptors like SIFT, ORB, and KAZE in relative pose estimation. We found that SIFT and ORB outperformed other descriptors in total matched features and matching time. Pose recovery enables the ego vehicle to estimate VRU trajectory in its own frame of reference. Next, with the available information, we can predict the future trajectory of VRUs. We used a Bayesian approach with MC dropout to quantify the uncertainty in pedestrian trajectory prediction. The method was evaluated on a real-world pedestrian dataset to generate a distribution of trajectories instead of a single trajectory. The results indicate that the mean predicted path of the probabilistic models is better and closer to the ground truth than the predictions from deterministic models [42]. The probabilistic models also had a lower average error than state-of-the-art methods for the ETH dataset.

In the future, we plan to improve the probabilistic predictions by incorporating perception uncertainty into the prediction module [43]. Usually, the sensors that track an object are noisy, and deterministic estimates cannot be trusted. Therefore, our research inclination lies in capturing the

sensing uncertainty and propagating the information into the prediction architecture. The prediction module can then take the upstream uncertainty into consideration to make more robust future predictions. Overall, this method will help make robust future state prediction directly from noisy raw sensor information.

Additional Products

Education and Workforce Development Products

1. One Ph.D. student and one post-doctoral associate have been involved in the project throughout its duration. The student developed a state-of-the-art algorithm for robust trajectory prediction of vehicles and VRUs that predicts both future states and associated uncertainty. This work will help uncertainty-aware cooperative planning in the future. The project greatly contributed to the student's Ph.D. dissertation. Further, the student is currently working on a journal manuscript that incorporates the sensing uncertainty elements in the current work and will be submitted for publication in the future.

2. As a part of knowledge transfer, the research on "Uncertainty estimation of pedestrian future trajectory" using Bayesian approximation was presented at the Annual Mechanical Student Research Symposium held at Virginia Tech.

3. Further, the perception and prediction algorithms developed from this project have been implemented in a 1/10 scaled autonomous car [44], which will help other graduate and undergraduate students in their own research projects.

Technology Transfer Products

1. Nayak, A., A. Eskandarian, and Z. Doerzaph, Uncertainty Estimation of Pedestrian Future Trajectory Using Bayesian Approximation. *IEEE Open Journal of Intelligent Transportation Systems*, Vol. 3, 2022, pp. 617-630. https://doi.org/10.1109/OJITS.2022.3205504

2. Ghorai, P., A. Eskandarian, Y.-K. Kim, and G. Mehr. State Estimation and Motion Prediction of Vehicles and Vulnerable Road Users for Cooperative Autonomous Driving: A Survey. *IEEE Transactions on Intelligent Transportation Systems,* Vol. 23, No. 10, 2022. pp. 16983-17002.

3. Nayak, A., A. Eskandarian, P. Ghorai, and Z. Doerzaph. A Comparative Study on Feature Descriptors for Relative Pose Estimation in Connected Vehicles. *ASME International Mechanical Engineering Congress and Exposition,* Vol. 85628, American Society of Mechanical Engineers, 2021, p. V07BT07A022. https://doi.org/10.1115/IMECE2021-70693

4. Ghorai, P., A. Eskandarian, A. Nayak, and Z. Doerzaph. Relative Pose Estimation for Cooperative Vehicles and Tracking of Multiple Dynamic Objects. In *ASME International Mechanical Engineering Congress and Exposition*, Vol. 85628, American Society of Mechanical Engineers, 2021, p. V07BT07A017. https://doi.org/10.1115/IMECE2021-69651

5. The manuscript "Pedestrian Trajectory Forecasting Using Deep Ensembles Under Sensing Uncertainty" is complete and has been submitted soon to *IEEE Transactions on Intelligent Transportation Systems*.

Data Products

- Link to Dataset – https://doi.org/10.15787/VTT1/BTEWCS
- Project Description – The goal of the project was to establish cooperative perception with relative pose recovery to estimate the position of objects and predict the future trajectory of objects with associated uncertainty.
- Data Scope – Five publicly available pedestrian datasets ETH, HOTEL, ZARA1, ZARA2, and UNIV were processed to generate more than 9,000 training trajectories. Each trajectory contains information on pedestrian ID, position, and velocity along x, y, and z.
- Data Specification – A detailed description of each variable in the dataset is provided in the Appendix.

References

1. Kim, S.-W., Z. J. Chong, B. Qin, X. Shen, Z. Cheng, W. Liu, and M. H. Ang. Cooperative Perception for Autonomous Vehicle Control on the Road: Motivation and Experimental Results. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE*, 2013, pp. 5059–5066.

2. Bensrhair, A., M. Bertozzi, A. Broggi, P. Miche, S. Mousset, and G. Toulminet. A Cooperative Approach to Vision-based Vehicle Detection. In *ITSC 2001. 2001 IEEE Intelligent Transportation Systems. Proceedings (Cat. No. 01TH8585), IEEE*, 2001, pp. 207–212.

3. Szeliski, R. *Computer Vision: Algorithms and Applications*. Springer Science & Business Media, 2010.

4. Lowe, D. G.. Distinctive Image Features from Scaleinvariant Keypoints. *International Journal of Computer Vision*, Vol. 60, No. 2, 2004, pp. 91–110.

5. Bay, H., T. Tuytelaars, and L. Gool. Surf: Speeded Up Robust Features. In *European Conference on Computer Vision*, Springer, 2006, pp. 404–417.

6. Sridhar, S., and A. Eskandarian. Cooperative Perception in Autonomous Ground Vehicles Using a Mobilerobot Testbed. *IET Intelligent Transport Systems*, Vol. 13, No. 10, 2019, pp. 1545–1556.

7. Rublee, E., V. Rabaud, K. Konolige, and G. Bradski. ORB: An Efficient Alternative to SIFT or SURF. In *International Conference on Computer Vision*, *IEEE*, 2011, pp. 2564–2571.

8. Alcantarilla, P. F., J. Nuevo, and A. Bartoli. Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 34, No. 7, 2011, pp. 1281–1298.

9. Tareen, S. A. K., and Z. Saleem. A Comparative Analysis of SIFT, SURF, KAZE, AKAZE, ORB, and BRISK. In *International Conference on Computing, Mathematics and Engineering Technologies (iCoMET), IEEE*, 2018, pp. 1–10.

10. Chien, H.-J., C.-C. Chuang, C.-Y. Chen, and R. Klette. When to Use What Feature? SIFT, SURF, ORB, or A-KAZE Features for Monocular Visual Odometry. In *International Conference on Image and Vision Computing New Zealand (IVCNZ)*, Palmerston North, New Zealand, 2016, pp. 1-6, https://doi.org/10.1109/IVCNZ.2016.7804434

11. Mehr, G., P. Ghorai, C. Zhang, A. Nayak, D. Patel, S. Sivashangaran, and A. Eskandarian. X-CAR: An Experimental Vehicle Platform for Connected Autonomy Research. *IEEE Intelligent Transportation Systems Magazine,* 2022.

12. Nayak, A., A. Eskandarian, P. Ghorai, and Z. Doerzaph. A Comparative Study on Feature Descriptors for Relative Pose Estimation in Connected Vehicles. In *ASME International*

Mechanical Engineering Congress and Exposition, Vol. 85628, American Society of Mechanical Engineers, 2021, p. V07BT07A022.

13. A. Eskandarian, C. Wu, and C. Sun. Research Advances and Challenges of Autonomous and Connected Ground Vehicles. *IEEE Transactions on Intelligent Transportation Systems,* Vol. 22, No. 2, Feb. 2021, pp. 683-711. https://doi.org/10.1109/TITS.2019.2958352

14. Xue, H., D. Q. Huynh, and M. Reynolds. Bi-prediction: Pedestrian Trajectory Prediction Based on Bidirectional LSTM Classification. In *International Conference on Digital Image Computing: Techniques and Applications (DICTA),* IEEE, 2017, pp. 1-8.

15. Xue, H., D. Q. Huynh, and M. Reynolds. SS-LSTM: A Hierarchical LSTM Model for Pedestrian Trajectory Prediction. In *IEEE Winter Conference on Applications of Computer Vision (WACV),* IEEE, 2018, pp. 11861194.

16. A. Houenou, P. Bonnifait, V. Cherfaoui, and W. Yao. Vehicle Trajectory Prediction Based on Motion Model and Maneuver Recognition. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 4363-4369. https://doi.org/10.1109/IROS.2013.6696982

17. Kahn, G., A. Villaflor, V. Pong, P. Abbeel, and S. Levine. Uncertainty-aware Reinforcement Learning for Collision Avoidance. *arXiv preprint arXiv*:1702.01182, 2017.

18. Suo, Y., W. Chen, C. Claramunt, and S. Yang. A Ship Trajectory Prediction Framework Based on a Recurrent Neural Network. *Sensors,* Vol. 20, No. 18, 2020, pp. 5133.

19. Park, S. H., B.-D. Kim, C. M. Kang, C. C. Chung, and J. W. Choi. Sequence-to-sequence Prediction of Vehicle Trajectory via LSTM Encoder-Decoder Architecture." In *IEEE Intelligent Vehicles Symposium (IV),* IEEE, 2018, pp. 1672-1678.

20. Altche, F., and A. de La Fortelle. An LSTM Network for Highway Trajectory Prediction. In *IEEE 20th International Conference on Intelligent Transportation Systems (ITSC),* IEEE, 2017, pp. 353-359.

21. Manh, H., and G. Alaghband. Scene-LSTM: A Model for Human Trajectory Prediction. *arXiv preprint arXiv*:1808.04018, 2018.

22. Alahi, A., K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese. Social LSTM: Human Trajectory Prediction in Crowded Spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 961-971.

23. Nikhil, N., and B. Tran Morris. Convolutional Neural Network for Trajectory Prediction. In *Proceedings of the European Conference on Computer Vision (ECCV)* Workshop, Anticipating Human Behavior, arXiv:1809.00696, 2018.

24. Zamboni, S., Z. T. Kefato, S. Girdzijauskas, C. Noren, and L. Dal Col. Pedestrian Trajectory Prediction with Convolutional Neural Networks. *Pattern Recognition*, Vol. 121, 2022, pp. 108252.

25. Wu, X., A. Nayak, and A. Eskandarian. Motion Planning of Autonomous Vehicles under Dynamic Traffic Environment in Intersections Using Probabilistic Rapidly Exploring Random Tree. *SAE International Journal of Connected and Automated Vehicles,* Vol. 4, No. 4, 2021, pp. 383-399.

26. Ammoun, S., and F. Nashashibi. Real Time Trajectory Prediction for Collision Risk Estimation Between Vehicles." In *IEEE 5th International Conference on Intelligent Computer Communication and Processing*, IEEE, 2009, pp. 417-422.

27. Ellis, D., E. Sommerlade, and I. Reid. Modelling Pedestrian Trajectory Patterns with Gaussian Processes. In *IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops,* IEEE, 2009, pp. 1229-1234.

28. Wiest, J., M. Höffken, U. Kressel, and K. Dietmayer. Probabilistic Trajectory Prediction with Gaussian Mixture Models. In *IEEE Intelligent Vehicles Symposium*, 2012, pp. 141-146.

29. Sun, C., and A. Eskandarian. A Predictive Frontal and Oblique Collision Mitigation System for Autonomous Vehicles. *ASME Letters in Dynamic Systems and Control*, Vol. 1, No. 4, 2021, p. 041012.

30. Khattar, V., and A. Eskandarian. Stochastic Predictive Control for Crash Avoidance in Autonomous Vehicles Based on Stochastic Reachable Set Threat Assessment. In *ASME International Mechanical Engineering Congress and Exposition,* Vol. 85628, American Society of Mechanical Engineers, 2021, p. V07BT07A026.

31. Althoff, M.. Reachability *Analysis and Its Application to the Safety Assessment of Autonomous Cars*. PhD dissertaion. Technische Universität München, Munich, Germany, 2010.

32. Zernetsch, S., H. Reichert, V. Kress, K. Doll, and B. Sick. Trajectory Forecasts with Uncertainties of Vulnerable Road Users by Means of Neural Networks. In *IEEE Intelligent Vehicles Symposium (IV),* IEEE, 2019, pp. 810-815.

33. Kim, B.-D., C. M. Kang, J. Kim, S. H. Lee, C. C. Chung, and J. W. Choi. Probabilistic Vehicle Trajectory Prediction over Occupancy Grid Map via Recurrent Neural Network. In *IEEE 20th International Conference on Intelligent Transportation Systems (ITSC),* IEEE, 2017, pp. 399-404.

34. Gal, Y., and Z. Ghahramani. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *International Conference on Machine Learning,* PMLR, 2016, pp. 1050-1059.

35. Zhang, X., and S. Mahadevan. Bayesian Neural Networks for Flight Trajectory Prediction and Safety Assessment. *Decision Support Systems,* Vol. 131, No. 2020, p. 113246.

36. Zhu, L., and N. Laptev. Deep and Confident Prediction for Time Series at Uber. In *IEEE International Conference on Data Mining Workshops (ICDMW),* IEEE, 2017, pp. 103-110.

37. Longuet-Higgins, H. C. A Computer Algorithm for Reconstructing a Scene from Two Projections. *Nature*, Vol. 293, No. 5828, 1981, pp. 133–135.

38. Jospin, L. V., W. Buntine, F. Boussaid, H. Laga, and M. Bennamoun. Hands-on Bayesian Neural Networks: A Tutorial for Deep Learning Users. *arXiv preprint arXiv*:2007.06823, 2020.

39. Pellegrini, S., A. Ess, and L. Van Gool. Improving Data Association by Joint Modeling of Pedestrian Trajectories and Groupings. In *European Conference on Computer Vision*, 2010, pp. 452-465.

40. Leal-Taixe, L., M. Fenzi, A. Kuznetsova, B. Rosenhahn, and S. Savarese, Learning an Image-Based Motion Context for Multiple People Tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 3542-3549. https://doi.org/10.1109/CVPR.2014.453

41. Takens, F. Detecting Strange Attractors in Turbulence. In *Dynamical Systems and Turbulence* (D. Rand and L. S. Young, eds., pp. 366-381), Springer, Berlin, Heidelberg, 1981.

42. Nayak, A., Eskandarian, A., & Doerzaph, Z. (2022). Uncertainty estimation of pedestrian future trajectory using Bayesian approximation. IEEE Open Journal of Intelligent Transportation Systems, 3, 617-630.

43. Ghorai, P., Eskandarian, A., Kim, Y. K., & Mehr, G. (2022). State estimation and motion prediction of vehicles and vulnerable road users for cooperative autonomous driving: A survey. IEEE Transactions on Intelligent Transportation Systems, 23(10), 16983-17002.

44. Sivashangaran, S., & Eskandarian, A. (2022). XTENTH-CAR: A Proportionally Scaled Experimental Vehicle Platform for Connected Autonomy and All-Terrain Research. arXiv preprint arXiv:2212.01691.

Appendix: Data Description

#	Variable	Description
1	ID	Pedestrian ID
2	Time	Time in Frames
3	x	Pedestrian *x* coordinate
4	y	Pedestrian *y* coordinate
5	z	Pedestrian *z* coordinate
6	v_x	Velocity of pedestrian along *x*
7	v_y	Velocity of pedestrian along *y*
8	v_z	Velocity of pedestrian along *z*