2018 21st International Conference on Intelligent Transportation Systems (ITSC)
Maui, Hawaii, USA, November 4-7, 2018

Towards Modeling of Perception Errors in Autonomous Vehicles

Pallavi Mitra*, Apratim Choudhury†, Vimal Rau Aparow†, Giridharan Kulandaivelu† and Justin Dauwels§
*Interdisciplinary Graduate School, Nanyang Technological University
†Centre of Excellence for Testing & Research of AVs - NTU
§School of EEE, Nanyang Technological University

***Abstract*—Detection and tracking of dynamic traffic objects such as pedestrians, cyclists, and surrounding ground vehicles is an important part of the perception of Autonomous Vehicle (AV). In practice, the presence of noise corrupts sensors' ideal performance, causing detection and state estimation of moving objects to be erroneous. These detection errors propagate through the overall system and potentially compromise the reliability and safety of the AV. To get an assurance that the vehicle will operate safely, any simulation platform for an AV must include a realistic representation of the fallacies in vehicle's perception. In this study, the perception error for a vision based detection algorithm of the camera sensor is modeled by applying auto-regressive moving average (ARMA) and nonlinear auto-regressive (NAR) method. It will enable statistical error values to be injected into ideal values obtained from simulation models. The proposed approach is evaluated based on several test case scenarios using various environmental and traffic information. A comparative analysis of the behavior of the AV with and without perception error model for the imperfection of camera sensor has been undertaken using the CarMaker platform. The investigation of the impact on the behavior of the AV by the variation of the state (distance, brake-torque) clearly depict the effectiveness of incorporating the error model at detection level in CarMaker.**

Keywords: Autonomous Vehicle; Auto-regressive Moving Average; Bounding box offset; CarMaker; Error modeling; Perception error; Sensor imperfection

I. INTRODUCTION

AVs have the potential to overcome a majority of traffic accidents caused due to the lack of human drivers' attention [1]. Simulation is being progressively incorporated for the development of AV systems because it permits the testing of a larger number of scenarios than would be viable with real-world testing, including risky situations which involve humans [2]. Hence, testing in virtual worlds is more secure, more proficient, and less expensive than real world testing [3].

A series of sensors such as lidar, radar, cameras are required to comprehend the surrounding environment in which the vehicle is traveling in real time and behave accordingly [4]. To predict and plan accurately, segmentation of the scene, exact localization of each traffic object, and their state estimation are essential features [5]. Due to imperfections of the sensors, several challenges in segmentation and motion inference might arise. For example, the estimations of depth by lidar experience the effects of noise, incorporate wrong reflections, and contain missing information. Camera images are rich and dense but, lack depth information [6]. In addition, external noise, lighting conditions such as sunny, cloudy, dark etc. have an adverse impact on them. In adverse weather conditions, some of the sensors might become untrustworthy and create visibility problems [7]. Thus, moving traffic objects such as cars, pedestrians, cyclists will not be detected properly and it may produce a hazardous as well as fatal situation. During environmental perception, detection and tracking of dynamic traffic objects can lead to significant issues. As real-world sensors are corrupted by noise, detection of moving obstacles are erroneous and produce uncertainty in the prediction of the correct states of the detected object. Some current robotics simulators such as GAZEBO, V-REP incorporate a simplified first order Gauss Markov model to add noise the the simulated sensor data [8]. However, this assumption produces uncertainty under various environmental condition which is vital in AV simulation. Additionally, vehicle dynamic simulation software providers use perfect sensor modeling (no physical models) to assess the vehicle functional performance [9]. Thus, high-fidelity simulation of an AV must incorporate an abstraction of the sensor error that captures the resultant misinterpretation by the vehicle's perception system. Hence, the perception-error model is the key point of the simulation process as this is where errors are generated during detection and can cause variation in decision making.

II. PERCEPTION ERROR CALCULATION

Here, we only consider perception errors associated with the camera sensor. For object detection, Faster RCNN has been chosen as state-of-the-art object detector [10]. Inaccurate bounding box is one type of perception errors. When an object is recognized with a misaligned bounding box, it leads to an error in localizing the object [11]. In order to model the bounding box offset, the main inputs required are ground truth and domain and measurement domain. The ground truth domain deals with the actual set up, in which object size is calculated in terms of bounding box's pixel area. The measurement domain pertains to the measurement results obtained from the sensors. Therefore, noise and sensor errors are part of the measurement domain. The error component is calculated by subtracting the measured data from the ground truth as shown in Fig 1.

978-1-7281-0323-5/18/$31.00 ©2018 IEEE

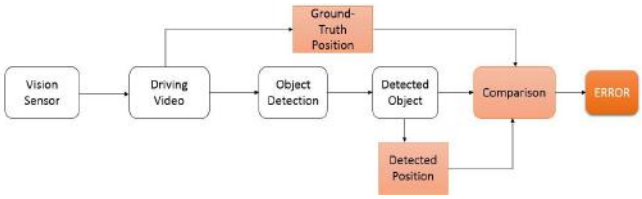

Fig. 1. Bounding Box Offset Calculation.

The ground-truth bounding boxes are labeled by top-left (tl) and bottom-right (br) corner coordinates. These coordinates are measured in terms of pixel with respect to the camera local frame. So, the shift in pixel coordinates for left-top and bottom-right corner gives the measure of error in detected bounding box, which is defined as:

$$E_{tl} = [x_{tl}, y_{tl}]_d - [x_{tl}, y_{tl}]_{gt}, \tag{1}$$

$$E_{br} = [x_{br}, y_{br}]_d - [x_{br}, y_{br}]_{gt}, \tag{2}$$

where E_{tl} and E_{br} are errors in pixel coordinates of top-left and bottom-right corner respectively. d and gt stand for detected and ground-truth bounding boxes respectively.

III. METHODS USED TO MODEL BOUNDING BOX OFFSET

A. Error Modeling Using Linear ARMA model

An Auto-Regressive Moving Average (ARMA) of orders p and q [12] is defined as:

$$X_t = C + \varepsilon_t + \sum_{i=1}^{p} \varphi_i X_{t-i} + \sum_{j=1}^{q} \theta_j \varepsilon_{t-j}, \tag{3}$$

where ε_t is zero-mean white noise with variance σ^2, p; $q \geq 0$ are integers, φ is set of parameters of auto regressive model, and θ is set of parameters of moving average model. The generated time series X_t from this model is known as ARMA(p,q) process, where p corresponds to the order of the Auto-Regressive (AR) component and q corresponds to the order of Moving Average (MA) component. ARMA model is used to predict the future values of a given time series X_t. In our case, X_t represents the time series of bounding box offset obtained from (1) & (2).

In time series analysis, an iterative three-stage modeling approach has been used to find the best fitted time-series ARMA Model [13]:

(i) Model identification & model selection
(ii) Parameter estimation of ARMA (p,q)
(iii) Model checking

1) Model Identification & Model Selection: This step determines the stationarity and seasonality of the time series, which needs to be modeled. In this step, we have to find the order of the auto-regressive and moving average part. To identify the order p and q, auto-correlation function (ACF) and partial auto-correlation function (PACF) are plotted. Generally, for an auto-regressive process of order 1 i.e. AR(1), the ACF plot is an exponential decay function. ACF plot for higher-order auto-regressive processes consists of a mixture of damped sinusoidal and exponentially decay components. The ARMA lags cannot be selected solely by looking at the ACF and PACF. Thus, to choose the best model from a set of possible models, we can take the help of a model selection method. The most common model selection method is the Bayesian Information Criteria (BIC) [14]. The BIC value of a model is given by -

$$\text{BIC} = -2\ln(\widehat{L}) + K\ln(n), \tag{4}$$

where $\widehat{L}$ = maximum likelihood function of the model, K = number of estimated parameters in the model and n = sample size (observation). During model selection, the model with the smallest value of BIC will be chosen.

2) Parameter estimation of ARMA (p,q): This step estimates the $p+q+2$ parameters (φ, θ, C, σ^2) by the method of maximum likelihood estimation (MLE).

3) Model Checking: The next step would be to carry out a test by checking the BIC values to verify whether the specifications are obeyed by the estimated model. If the estimation is insufficient, we must build a better model by returning to step one.

B. Error Modeling Using Nonlinear Auto-Regressive (NAR) Model

To model a time series data, a linear ARMA model is a good stepping stone. However, in a real life scenario, a time series will possess many nonlinear characteristics. Hence, to deal with such nonlinearities, we have to utilize a nonlinear auto regressive (NAR) model [15], which can be defined as follows:

$$y_t = G(y_{t-1}, y_{t-2}, y_{t-3}, ...) + \varepsilon_t, \tag{5}$$

where y is the variable of interest and ε_t is the error term or noise term. The function G corresponds to nonlinear function, such as neural network, sigmoid network, wavelet network, etc. To select the best model from a set of possible models, we can take the help of Bayesian Information Criteria (BIC) model selection method. The BIC value of a model [16] is given by:

$$\text{BIC} = (n)\ln(\text{SSE}/n) + (p)\ln(n), \tag{6}$$

where SSE = sum squared error, n = number of training samples and p = number of parameters (weights and biases). The model with smallest BIC is favored when picking from a few models.

IV. APPLICATION OF THE BOUNDING BOX OFFSET MODEL IN A SIMULATION PLATFORM

In order to illustrate the application of the bounding box offset model for camera sensor, we decided to integrate it with CarMaker [17], for adding noise to its camera object sensor data. CarMaker is used extensively in the automotive

industry as a virtual test platform, to develop applications for longitudinal, lateral and vertical dynamics of the vehicle. This can be done because CarMaker offers tools to create highly detailed virtual prototypes of vehicle models along with very detailed representation of both the static (road network, buildings, traffic signals, etc.) and dynamic environment (surrounding traffic, pedestrians, etc.).

However, before a vehicle and sensor model in CarMaker can be used for testing, we need to ensure that we are able to control it using an Automated Driving System (ADS). For this purpose, we have chosen the self driving software, Autoware [18], developed by Nagoya University.

The basic architecture for simulating autonomous behaviour of a CarMaker vehicle, with the ADS of Autoware, has been shown in Fig 2. Sensor Object data is extracted from CarMaker using it's APO (Application Online) interface. It is then packaged into a string and transferred to a ROS node via TCP/IP socket. Object information, in case of a pedestrian, consists of absolute distance and angular displacement of the pedestrian, relative to the vehicle. Since the error model is designed for bounding boxes we convert the object information, which are coordinates relative to the vehicle frame, into a two-dimensional bounding box relative to the camera image frame. The error is then added to the bounding box coordinates, and the object data is recomputed w.r.t. the vehicle frame. This data is then passed onto Autoware's motion planning module to compute the desired velocity and steering angle commands. These control inputs are then passed back to CarMaker, via TCP/IP and the APO.

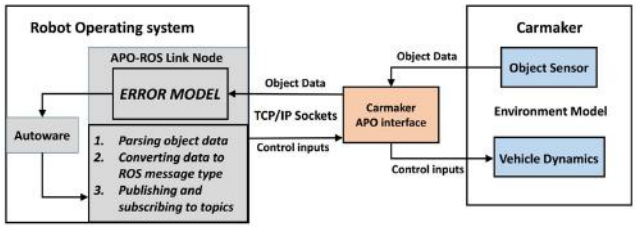

Fig. 2. Simulation Architecture

V. SIMULATION AND RESULTS

A. Dataset

In order to develop the error model we have utilized the KITTI object tracking evaluation 2012 [19]. The dataset has been recorded by driving on highways and urban area around the mid-size city of Karlsruhe. The 2D bounding boxes for each images have been evaluated for 'car'& 'pedestrian'classes. The resolution of the images is 1238×374 pixels and the frame rate is 10 fps. Two subsets of this dataset consisting of 314 and 1059 image frames are chosen for this study. These videos are taken in different condition like the sunny road with shadow, the urban road with traffic and highway respectively. These two dataset are denoted as 'Dataset'1 & 'Dataset'2 in the following sections.

B. Modeling of bounding box offset using linear ARMA

With the help of the error data generated, as described in Section II, and the iterative three stage modeling approach, as mentioned in Section IIIA, the parameters (C, φ, θ, σ^2) and the orders (p, q) of the ARMA model are estimated.

1) Dataset 1: ARMA(3,3), ARMA(3,3), ARMA(4,4), ARMA(3,3) models are obtained as best fitted time-series models, as they match the displacement errors of x, y coordinates of top-left (x_{tl}, y_{tl}) and bottom-right (x_{br}, y_{br}) corner of the bounding box respectively. The estimated parameters of these ARMA models are listed in Table I. The combined time series plots of errors predicted by ARMA models and actual errors are shown in Fig.3.

TABLE I
PARAMETERS OF ARMA MODELS

Coordinates	Constant	AR Coeffi-cient	MA Coeffi-cient	Variance	BIC Value
x_{tl}	-1.3010	0.8440 -0.8018 0.6583	-0.4384 0.6405 -0.2880	16.9413	1752.1
y_{tl}	-3.5303	-0.6465 0.4427 0.3870	1.2081 0.2775 -0.1113	53.3757	2033.8
x_{br}	0.8219	-0.2248 0.0200 0.3443 0.6796	0.4904 0.2582 -0.2564 -0.7707	31.8455	1919.1
y_{br}	0.0874	1.8263 -1.0801 0.2419	-1.3421 0.6023 -0.1993	64.1949	2106.6

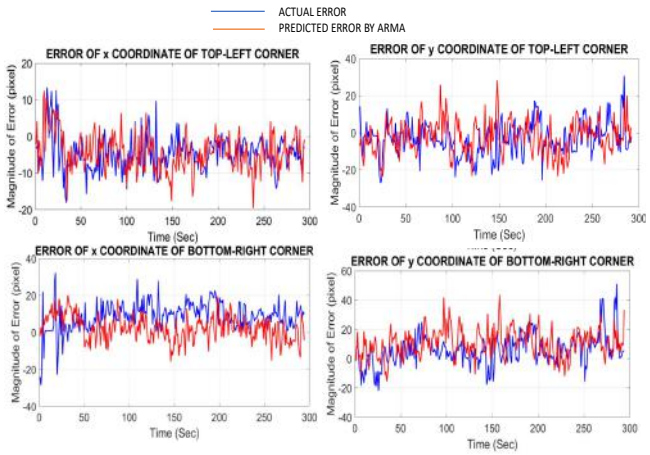

Fig. 3. Combined time series plot of errors of top-left and bottom-right corner coordinates.

2) Dataset 2: ARMA(1,1), ARMA(3,3), ARMA(3,3), ARMA(3,1) are obtained as best fitted time-series models, which match the models that generated the error of of x, y coordinates of top-left (x_{tl}, y_{tl}) and bottom-right (x_{br}, y_{br}) corner of the bounding box respectively. The estimated parameters of these ARMA models are listed in Table II. The combined time series plot of errors predicted by ARMA models and actual errors are shown in Fig 4.

TABLE II
PARAMETERS OF ARMA MODELS

Coordinates	Constant	AR Co-efficient	MA Co-efficient	Variance	BIC Value
x_{tl}	-0.1567	0.9238	-0.3276	78.5534	3134.3
y_{tl}	2.0447	-0.3308 0.0774 0.6737 0.3192	0.7045 0.4325 -0.2644 -0.2834	59.6750	3028.5
x_{br}	-0.5506	0.9161 -0.6704 0.6189	-0.3631 0.5172 -0.2818	115.7480	3569.2
y_{br}	-1.1622	0.850371 -0.0014 0.0041	-0.39681	71.1296	3082.8

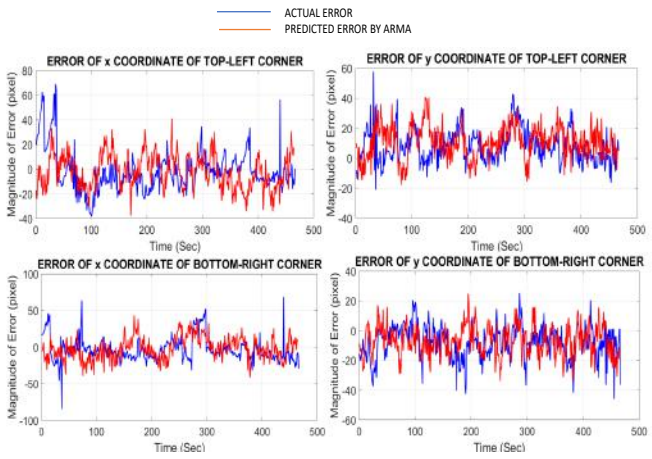

Fig. 4. Combined time series plot of errors of top-left and bottom-right corner coordinates.

C. Modeling of bounding box offset using nonlinear auto-regressive (NAR)

Here, feed-forward artificial neural network (ANN) has been used to approximate the nonlinear function for nonlinear autoregressive function. The model predicts the value of the observation y_t (bounding box offset) using past two values of y with a noise term. The feed-forward ANN with one hidden layer and bias in hidden and output layer is used to model the bounding box offset as per the given description.

(i) *Input Layer* : The bounding box offset vector (X), obtained in section II is fed into the input layer of the neural network.

(ii) *Hidden Layer* : The input layer passes on the value X to the hidden layer. The number of neurons in hidden layer can be varied. The output of the hidden layer is given by:

$$z = a\Big(W^1 * X + b^1\Big), \tag{7}$$

where a is activation function (hyperbolic tangent), X is the input vector, and W^1 and b^1 are the weight and bias vectors of the hidden layer respectively. In our case, both W^1 and b^1 are $h \times 1$ vectors (h = size of hidden layer).

(iii) *Output Layer* : The output vector is defined as:

$$y = a\Big(W^2 * z + b^2\Big), \tag{8}$$

where a is the activation function (linear function), z is the output of hidden layer, and W^2 and b^2 are the weight and bias vectors of the output layer. Here the dimensions of W^2 and b^2 are $1 \times h$ and 1×1 respectively.

We consider different ANN architectures with hidden layer sizes of 10, 15, 20 and 25 respectively.

Dataset 1: The BIC values obtained from the different architectures of NARNET are presented in Table III.

TABLE III
PARAMETERS OF NARNET MODELS

Coordinates	NARNET (10)	NARNET (15)	NARNET (20)	NARNET (25)
x_{tl}	827.4	851.57	**811.1**	876.6
y_{tl}	1202.6	**1148.7**	1166.2	1208.1
x_{br}	1050.7	1032.4	1100.2	**1006.1**
y_{br}	1216.2	**1178.1**	1211.0	1207.6

BIC values are smallest for the NARNET with 20, 15, 25 and 15 hidden units respectively, as shown in Table III. These NARNET models are chosen so that they predict the models that generate the error data of top-left and bottom-right corner coordinates. The combined time series plot of the errors predicted by NARNET models and actual errors is shown in Fig 5.

Dataset 2: The BIC values for different architectures of NARNET are listed in Table IV.

BIC values for NARNET with 20 hidden units, 15 hidden units, 20 hidden units and 10 hidden units are the smallest ones for the bounding box offsets. These NARNET models are chosen which predict the models that generated the error data of top-left and bottom-right corner coordinates. The combined time series plot of errors predicted by NARNET models and actual errors is shown in Fig 6.

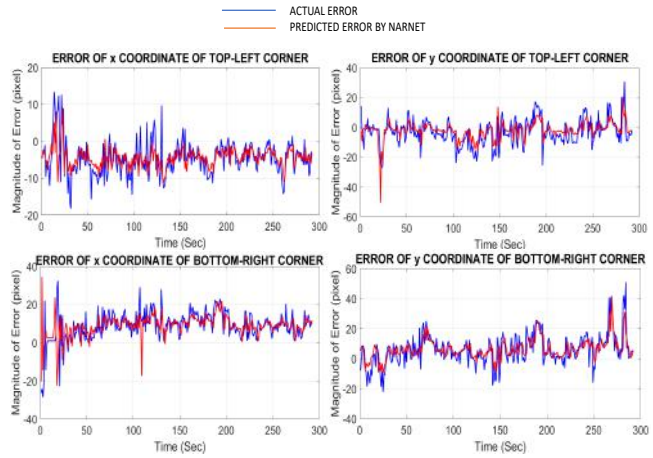

Fig. 5. Combined time series plot of errors of top-left and bottom-right corner coordinates.

TABLE IV
PARAMETERS OF NARNET MODELS

Coordinates	NARNET (10)	NARNET (15)	NARNET (20)	NARNET (25)
x_{tl}	1971.9	1952	**1904.6**	1930.7
y_{tl}	1854.2	**1841.5**	1964.0	1887.3
x_{br}	2165.7	2173.4	**2164.3**	2243.1
y_{br}	**1973.9**	2004.6	2056.7	2004.1

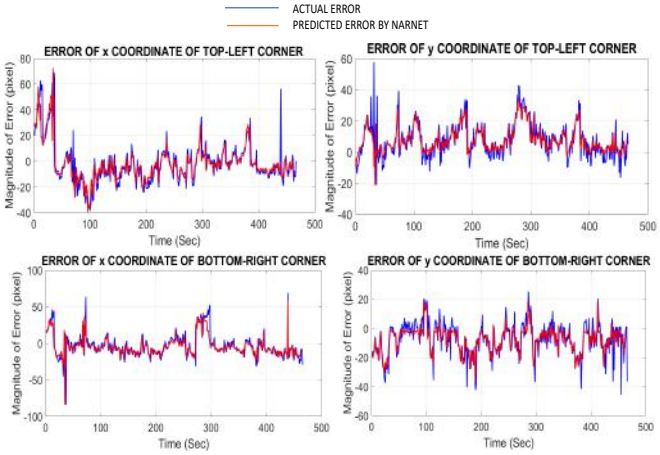

Fig. 6. Combined time series plot of errors of top-left and bottom-right corner coordinates.

D. Comparison of BIC Values

The BIC values of both the linear ARMA model and NARNET model are listed in Table V.

For both datasets, the BIC values for nonlinear auto-regressive models are smaller than that of the linear ARMA models. This seems to suggest that nonlinear auto-regressive models provide a better representation of the bounding box offset than the linear ARMA model.

TABLE V
BIC VALUES COMPARISON

	Coordinates	linear ARMA	NARNET
Dataset 1	x_{tl}	1752.1	811.1
	y_{tl}	2033.8	1148.7
	x_{br}	1919.1	1006.1
	y_{br}	2106.6	1178.1
Dataset 2	x_{tl}	3134.3	1904.6
	y_{tl}	3028.5	1841.5
	x_{br}	3569.2	2164.3
	y_{br}	3082.8	1973.9

From the obtained NARNET models, we can generate instances of bounding box errors, which in turn can be integrated in simulations of AVs in realistic environments (see subsection E).

E. Simulation configuration

In order for the error model to be applicable to the simulated camera images in CarMaker, the resolution of the CarMaker animation was chosen to be default, i.e., 1960×966. Horizontal field of view of the camera sensor is $20°$. The error-free bounding box coordinates for the objects within the simulated camera frame have been calculated using the object's actual distance, the azimuth angle, and the camera matrix. Different samples drawn from the NARNET models (obtained in Section D) have been incorporated into the simulation platform in order to simulate bounding box imperfections. Fig 7 represents the bounding box for a detected pedestrian, in CarMaker, without (left) and with (right) stochastic errors generated from the NARNET model.

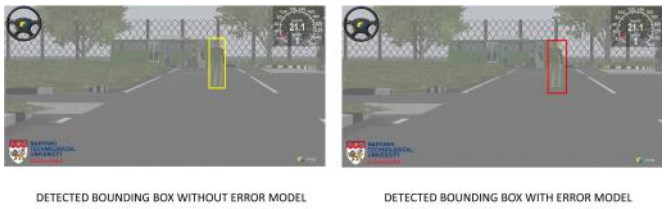

Fig. 7. Detected bounding box without and with error.

The distance of the detected object from vehicle is calculated using the pinhole projection formula, as follows:

$$d(mm) = \frac{K_f \times H(mm)}{O_h(pixel)} \tag{9}$$

where K_f is the focal length (mm) * pixel density, H is the actual height of the object in mm, and O_h is the object height in image in pixel. The distance calculation is carried out assuming that projection of the 3-D object onto a 2-D plane is based on the pinhole camera model [20]. This

is a simplified approach for calculating distances of objects in images, and requires knowledge of the actual height of the object. Even though this formula may not be appropriate for certain real-life applications, our aim is to illustrate how the vehicle behavior is affected by perception errors. Therefore, if distance estimation done based on bounding box dimensions, a more sophisticated formula will also be subject to similar predicaments if there are errors in the bounding box generation.

Due to the changes in the bounding box's position of the detected object, the perceived distance of the object from the vehicle and subsequently, the brake-torque of the vehicle is different from the actual state. This may lead to erroneous behavior of the vehicle, potentially with dramatic consequences. In a more complex traffic environment, involving multiple pedestrians and vehicles, we can expect a significant effect of the error in distance profile. To visualize the effect of the errors on distance and brake-torque of the vehicle in presence of one pedestrian, 4 samples drawn from NARNET models have been considered (see Figs. 8 and 9). We generated the bounding box errors by the NARNET model, since this models is more accurate than the linear ARMA model (see Section V-D).

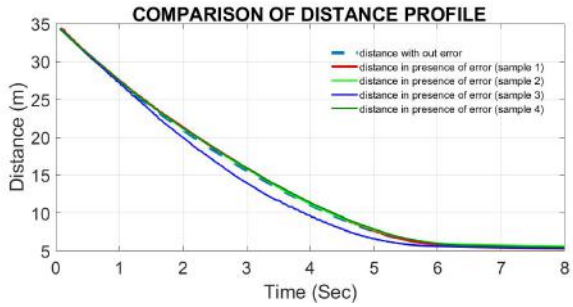

Fig. 8. Distance profile of vehicle in presence of error model.

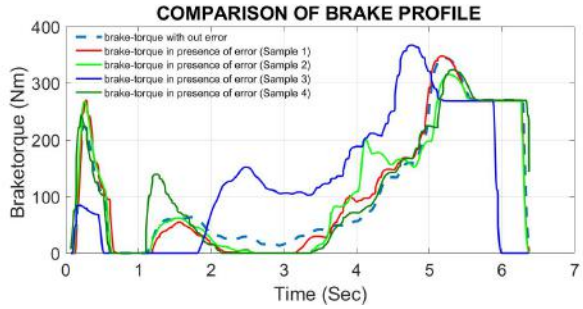

Fig. 9. Brake profile of vehicle in presence of error model.

VI. CONCLUSION

In this paper, we have modeled imperfections in the bounding box detection using a linear ARMA and a NARNET model. The NARNET model performs better than linear ARMA as far as modeling bounding box offset is concerned. This work shows a novel approach to integrate the error model into the object detection module of CarMaker, which helps in generating erroneous bounding boxes. The control action, i.e., brake-torque applied at the wheel of the vehicle will vary according to the detected position of the object. The additional parameters that influence the errors such as various weather conditions, complex environment, or multiple traffic objects may lead to significant deviations of the perceived states from the ground truth. These additional effects will help us to check the behavior of AV in different test case scenarios during simulation. A more in-depth study on this will be carried out in the future, where we will compare the results from simulations to field measurements on test circuits.

REFERENCES

[1] Broggi, Alberto, et al. "Intelligent vehicles." Springer Handbook of Robotics. Springer International Publishing, 2016. 1627-1656.

[2] W.-K. Chen, Linear Networks and Systems (Book style). Belmont, CA: Wadsworth, 1993, pp. 123135.

[3] Donikian, Stephane. "VUEMS: a virtual urban environment modeling system." Computer Graphics International, 1997. Proceedings. IEEE, 1997.

[4] Liu, Shaoshan, et al. "CAAD: Computer Architecture for Autonomous Driving." arXiv preprint arXiv:1702.01894 (2017).

[5] Katrakazas, Christos, et al. "Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions." Transportation Research Part C: Emerging Technologies 60 (2015): 416-442.

[6] Ummenhofer, Benjamin, et al. "DeMoN: Depth and motion network for learning monocular stereo." arXiv preprint arXiv:1612.02401 (2016).

[7] West, Darrell M. "Moving forward: Self-driving vehicles in China, Europe, Japan, Korea, and the United States." (2016).

[8] Shah, Shital, et al. "Airsim: High-fidelity visual and physical simulation for autonomous vehicles." Field and Service Robotics. Springer, Cham, 2018.

[9] IPG Automotive GmbH, http://www.ipg.de

[10] Ren, Shaoqing, et al. "Faster R-CNN: Towards real-time object detection with region proposal networks." Advances in neural information processing systems. 2015.

[11] Hoiem, Derek, Yodsawalai Chodpathumwan, and Qieyun Dai. "Diagnosing error in object detectors." Computer Vision?ECCV 2012 (2012): 340-353.

[12] Box, George EP, et al. Time series analysis: forecasting and control. John Wiley & Sons, 2015.

[13] Yule, G. Udny. "On a method of investigating periodicities in disturbed series, with special reference to Wolfer's sunspot numbers." Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character 226 (1927): 267-298.

[14] Findley, David F. "Counterexamples to parsimony and BIC." Annals of the Institute of Statistical Mathematics 43.3 (1991): 505-514.

[15] Billings, Stephen A. Nonlinear system identification: NARMAX methods in the time, frequency, and spatio-temporal domains. John Wiley & Sons, 2013.

[16] Allende, Hctor, Claudio Moraga, and Rodrigo Salas. "Artificial neural networks in time series forecasting: A comparative analysis." Kybernetika 38.6 (2002): 685-707.

[17] https://ipg-automotive.com/

[18] Kato, Shinpei, et al. ?An open approach to autonomous vehicles.? IEEE Micro 35.6 (2015): 60-68.

[19] Geiger, Andreas, Philip Lenz, and Raquel Urtasun. "Are we ready for autonomous driving? the kitti vision benchmark suite." Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on. IEEE, 2012.

[20] Sturm, Peter. "Pinhole camera model." Computer Vision. Springer US, 2014. 610-613.