Advanced Emergency Braking with Sensor Fusion

Yunuo Zhang, Jacob Buckelew

I. Abstract

In this paper we implemented Autonomous Emergency Braking(AEB) using a sensor funsion algorithm. The AEB system is designed to help drivers avoid or mitigate collisions with other vehicles. The final product is a model of an AEB controller that incorporates the sensors, environment, and vehicle dynamics of the system. We simulate the system with different scenarios and show the results for two different driving scenarios to assess safety properties. We also compare the results of different scenarios such as TTC calculation time vs stoppint time, changing of FCW status vs changing of AEB status.

II. Introduction

Around 94% of accidents were caused by human errors according to the survey of the American National Highway Traffic Safety Administration (NHTSA) [1]. There are many practical reasons that may cause accidents from human perspective: jay-walking pedestrians, limited view of traffic conditions, violation of the traffic rules, lack of attention, impaired driving, driver distraction etc. Nowadays, researchers begin to give more attention on ensuring safety of autonomous vehicles (AV). Autonomous Emergency Braking (AEB) system is designed to ensure the safety of drivers by preventing collisions and road accidents.

An AEB system (AEBS) is a type of advanced-driver assistance system (ADAS) that is designed to aid the driver in identifying critical scenarios so that accidents can be avoided. Essentially, AEBS detects the presence of traffic ahead of the driver and will automatically apply brakes in emergency scenarios if the driver is too slow or unable to react [2]. Often, such systems rely on sensor technologies such as vision, RADAR, and LIDAR sensors to measure distances between the vehicle and other objects [3]. In order to reduce uncertainty, multiple sensors can be integrated into a sensor fusion model so that better informed decisions can be made in response to changes in the driver's environment.

For example, Fig. 1 integrates two different types of sensors into a sensor fusion model, namely RADAR and video sensors. These independent sensors will collect input data such as the distance between the ego vehicle and a vehicle ahead of it. All of the data is sent over a switch to a component that controls the sensor fusion algorithm. From there, the controller can decide what amount of deceleration is needed and then send this output to a system that controls the braking in the vehicle.

There have been many examples demonstrating the success of AEBS in reducing collisions such as in the case of pedestrian collisions [4]–[6], rear-end collisions [6]–[9], and cyclist collisions [10]. In [9], a study was conducted to evaluate the effectiveness of AEB and forward collision warning (FCW)

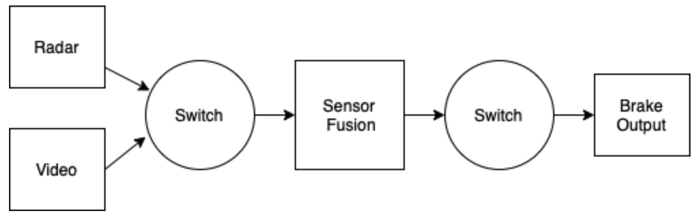

Fig. 1. A simplified sensor fusion model

and found a 50% reduction in rear-end crash involvement rates when combining the two techniques. Furthermore, on its own, AEB was able to reduce rear-end crash involvement rates by 43%. Regarding pedestrian collisions, A study in [5] found a reduction of 25%-27% in pedestrian collision risk and a 29%-27% reduction in pedestrian injury collision risk. The literature has generally been positive with respect to the potential for AEBS in preventing pedestrian injuries and rear-end collisions. Thus, we would like to construct an AEBS model to further explore this potential and contribute to the well-being and safety of drivers.

AEBS can be modeled as autonomous cyber-physical systems since it is not necessary for drivers to operate the system. In AEBS, the 3 key subsystems are those dedicated to sensor fusion, decision logic, and controlling the brakes of the vehicle. The sensor fusion component must handle the input into the AEBS and be able to collect data such as the distance from the ego vehicle to the vehicle ahead of it. Using a combination of sensors, each sensor in this component must pre-process the data it collects and then send it to a central control system that houses the decision logic of the AEBS. Because sensor fusion reduces the uncertainty, the controller can then make an informed decision as to whether or not the brakes should be applied. If a critical event is detected, the central controller will take action to determine the amount of deceleration that is needed to avoid collision and the vehicle will then apply the braking action.

This paper proposes an AEBS model that integrates sensor fusion technology using the Simulink modeling environment. In our design, we rely on sensor fusion to provide the inputs to our AEB controller. We also model the decision logic of the AEB controller and its interactions with the braking system of the car in order to initiate deceleration and prevent collisions. In order to evaluate the performance of the controller, we will define safety requirements and assess the satisfaction of these requirements using test scenarios.

The rest of the paper can be outlined as follows. In Section III, we describe background related to the use of sensor fusion in autonomous vehicles and provide examples of previous works that have modeled AEBS. Next, Section IV and Section V cover the design of our proposed AEB controller and the sensor and environment systems, respectively. Section VI discusses the safety verification of our AEB controller. Finally, Section VII covers the simulations and analysis of our proposed AEB controller and in Section VIII we conclude our study.

III. Background

A. Sensor Fusion

In AVs, the most prominent sensors used in detecting vehicle surroundings are GPS, RADAR, vision, and LIDAR sensors [11]. A sensor fusion model aims to merge the data generated by various sensors in order to calculate a more accurate state estimate about objects in the environment. By leveraging the complementary capabilities of the various sensors, the sensor fusion techniques reduces the overall uncertainty in forming objection detection hypotheses [11]. Besides increasing the confidence rate of the data, sensor fusion can also allow for a higher resolution on the data, reduce errors in data collection, and bring completeness to the data [12]. In an AEBS, accurate state estimation is integral to ensuring the safety of the driver and therefore implementing sensor fusion will be a key part of our proposed model.

In order to implement sensor fusion, there are usually 3 important tasks that must be considered: estimation methods, classification methods, and inference methods [13]. Using these three methods, one can then build a general structure of how raw data helps determine the choices made in the decision logic of a controller. At the lowest level, a fusion occurs among signals and it is at this level where recursive estimation methods such as Kalman filtering and nonrecursive estimations methods such as the least-squares method can be employed [12], [13]. After estimation methods are applied, fusion is conducted on the features that are collected from the multiple sensors and merged into single features that achieve a high level of discrimination [14]. At this level, it is common to implement clustering algorithms such as *k*-means clustering or classification methods such as support-vector machines (SVMs) to solve classification tasks for objects that are detected by the sensors. Finally, at the highest level there is the decision fusion process that essentially considers the decisions made by each of the sensors and forms one joint decision. Among the most commonly used inference methods at this stage are bayesian inference methods, particle filters, and fuzzy logic [12], [13].

Although there are many methods that can be implemented for the different levels of fusion, selecting the most appropriate methods in the context of the application domain is often the most difficult task for solving sensor fusion problems. Historically, statistical methods have been relied on to solve sensor fusion problems, but recently there has been a shift toward implementing deep learning sensor fusion algorithms [15]. Many of the studies that have explored this frontier have relied on Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN), such as in the case of the YOLO CNN [15], [16]. YOLO introduced a fusion model that merged RGB camera data with LIDAR data, and the merged data could then be used an input into a CNN for feature extraction and object detection. The weighted mean YOLO generated promising results, outcompeting another YOLO algorithm and a Faster R-CNN [17] model which had previously set state-of-the-art performance in object detection [16].

An alternative approach with deep learning was taken in [18] and [19] with the creation of VoxelNet. Essentially, VoxelNet relied on input from 3D sensors that would be fed into an object detection framework composed of 3 major parts: a feature learning network that learns complex features of objects using voxel feature encoding layers, convolutional layers that aggregate voxel features, and finally a region proposal network that uses the aggregated feature map to output a detection result [18]. Unlike previous works, VoxelNet explored the powerful applications of 3D sensors and how they could be useful for object detection problems. Compared to other methods, VoxelNet's novel approach proved to be very powerful and outperformed previous state-of-the-art methods in detection scenarios involving pedestrians, cyclists, and vehicles [18]. Ultimately, the success of VoxelNet would usher in a new wave of research in 3D object detection and continue to fuel the growth of deep learning in the development of novel sensor fusion methods.

B. Related Work

Since the growth of the AV industry in the early 2000s, there has been extensive research done in building and optimizing AEB controllers. One of the notable early examples of building AEBS can be found in [20] which involved an obstacle detection component and a controller component that was further divided into an upper-level controller and lower-level controller. Using obstacle detection information generated from the fusion of RADAR and vision sensor data, the upper-level controller would decide the control mode using safety indexes such as time-to-collision (TTC) and warning index. If collision was a threat, the TTC would begin to rise and the safety index would drop, and therefore the upper-level controller enforced a threshold method to detect when the indexes rose above or dropped below a critical value. The upper-level controller would then send a mode to a lower-level controller corresponding to various states such as "Warning Region" or "Safe Region", and at this stage the controller would decide if a warning signal needed to be displayed to the driver and if braking action needed to be applied. This framework can be easily visualized in Fig. 2 below.

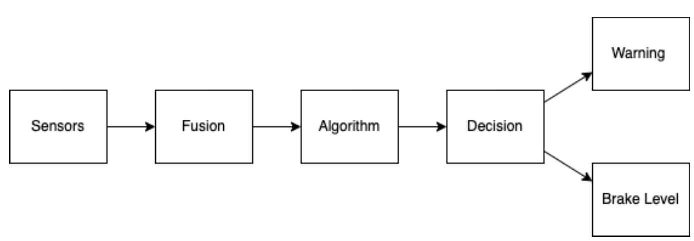

Fig. 2. Structure of a simple AEBS

There are many other examples of AEBS models in the literature that have explored how to best implement sensor fusion [21]–[31], and many of these examples have relied environments like Simulink [21]–[23] because of its Automated System Driving Toolbox and Driving Scenario Designer app, which is useful for generating test scenarios. The model that is most similar to our proposed model can be found in [21], which relies on the AEBTestBench in Matlab's ADAS toolkit to validate the safety and performance of the the proposed AEB model. Using the Automated System Driving Toolbox, separate components representing the calculation of TTC and time to forward collision warning (FCW) are composed together with sensor blocks and an AEB controller to build the full AEBS. Once the model has been composed, the Driving Scenario Designer App can be used to perform simulations on scenarios such as those proposed by the European New Car Assessment Programme (Euro NCAP). Thus, Simulink provides a helpful all-in-one environment that streamlines the design phase and testing phase of AEBS models.

Regarding sensor fusion, the model in [21] relied specifically on Kalman filters to process the incoming data and obtain value-based state estimations based on observations of the lead vehicle. Using the sensor fusion information, the AEB controller can calculate the time needed for FCW and if it greater than the TTC, it will send an alert to the driver. If the driver is unable to apply braking action, the AEBS steps in and minimizes the speed of the vehicle on its own. Other studies have demonstrated progress in improving sensor fusion to handle vehicle recognition [32], make more accurate driving path estimations [29], and have stronger perception ability [31]. In our proposed model however, we will focus primarily on exploring the Kalman filtering technique for sensor fusion.

After designing an AEBS model, it is necessary to satisfy the safety and liveness properties of the system due to its safety-critical nature. One of the most essential safety requirements is to ensure that the distance between the ego vehicle and an obstacle ahead of it never reaches zero. For example, a case study in [33] constructed a Simulink-based AEBS that integrated an image-based classifier for object detection. A signal temporal logic (STL) specification was defined where the longitudinal distance between the the ego and an obstacle should always be greater than zero. Then, a validity domain graph was constructed to identify possible counterexamples that could occur if the classifier failed to classify objects ahead of the ego vehicle. By validating specific and unambiguous requirements for an AEBS model, safety properties can be assessed more easily and efficiently to ensure the safety of drivers.

IV. Modeling: AEB Controller

A. Braking Controller

As figures 3 and 4 show, or as the model braking controller inside the ”$AEB_controller.slx$” in AEB controller directory shows, the AEB controller calculates time-to-collision (TTC), stopping time for different braking situations, and controls the decision logic of the AEB controller implemented by Stateflow.

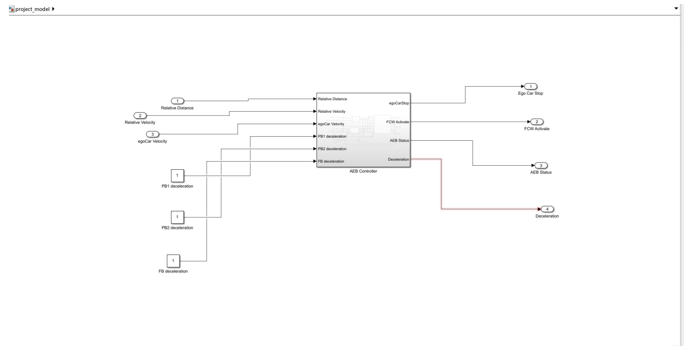

Fig. 3. AEB braking controller

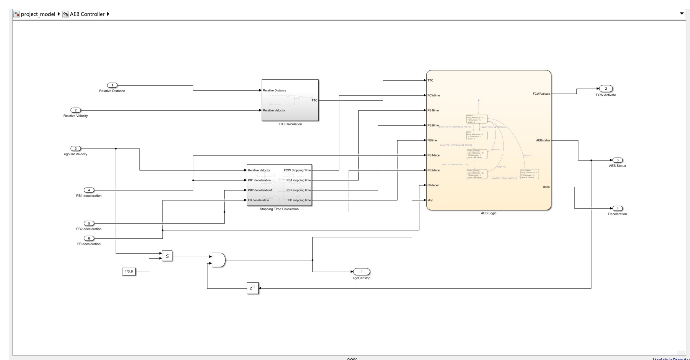

Fig. 4. Inside the AEB controller

B. TTC calculation and stopping time calculation

Time-to-collision(TTC) is defined as the time estimated that will collide with the vehicle ahead. As figure 5 shows, we define the time needed for Forward Collision Warning (FCW) begin to alert the driver as the sum of time needed for the driver to react when he/she hear the warning and the time needed for the car to completely stop when the driver begins applying the brake. We denote the former time as T_{react} and the latter as T_{stop}, then $T_{FCW} = T_{react} + T_{stop}$. We define the controller logic in a way that the FCW alert activates if TTC with the lead vehicle is less than T_{FCW}. If the driver fails to apply the brake in time, such as due to distraction, the AEB system acts independently of the driver to avoid or mitigate the collision. The AEB system is designed to apply cascaded braking, i.e it will receive the signal from the AEB decision logic to decide whether it should apply for 1st stage partial braking or 2nd stage partial braking or full braking. We implemented the automatically braking in such a cascaded way so that we can not only ensure safety by avoiding collision but also provide a best braking experience to avoid "sudden stops" as much as possible. Figure 4 shows the design model to calculate stopping time for different types of braking and figure 4 shows the Stateflow Chart that inputs these information, and the chart includes the logic for when the system should apply which type of braking.

C. Nonlinear MPC Controller

As figure 6 shows, there is a Nonlinear MPC (NLMPC) Controller Subsystem contained in the AEB controller.

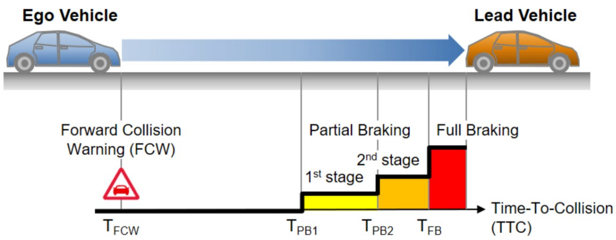

Fig. 5. FCW alert system

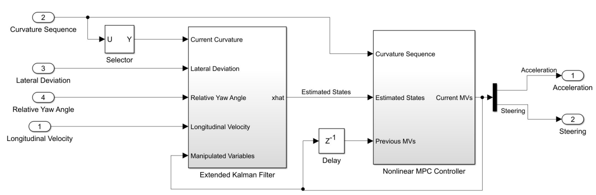

Fig. 6. Nonlinear MPC Controller

The NLMPC controller reads the ego longitudinal velocity, curvature sequence, relative yaw angle, and lateral deviation, and then outputs the steering angle and acceleration for the ego vehicle. The product of the road curvature and the longitudinal velocity is modeled as a measured disturbance. The NLMPC controller is a prediction model with seven state variables, three output variables, and two manipulated variables. The seven state variables are: Lateral Velocity, Yaw Rate, Longitudinal velocity, Longitudinal acceleration, Lateral deviation, Relative yaw angle, Output disturbance of relative yaw angle. The three output variables are Longitudinal velocity, Lateral deviation, Sum of the yaw angle and yaw angle output disturbance. The manipulated variables are acceleration and steering. The extended Kalman filter (EKF) provides estimations for these seven state variables. The estimated values are then sent as output into the NLMPC controller. Then, the NLMPC outputs acceleration, and steering angle.

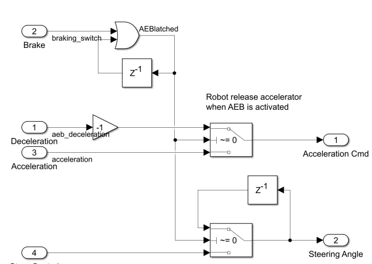

Fig. 7. Controller Mode Selector

In addition, the AEB controller system also contains a controller mode selector which releases the vehicle accelerator when AEB is activated, as figure 7 shows.

V. Modeliing: Sensors and Environment

There is also a Sensors and Environment system which configures the road network, defines target actor trajectories, and synthesizes sensors.

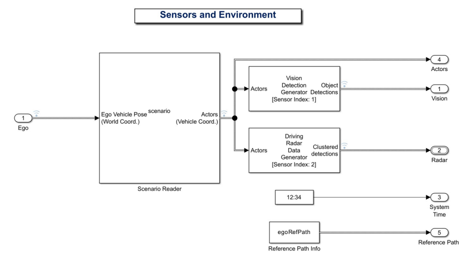

Fig. 8. Sensors and Environment

As figure 8 shows, the Scenario Reader reads the drivingScenario object from the workspace and then reads the actor data from that object. There are several available driving scenarios that we test for our experiments, the list of driving scenarios are avialble in Matlab are: 'scenario 01 AEB Bicyclist Longitudinal 25width', 'scenario 02 AEB Bicyclist Longitudinal 50width', 'scenario 03 AEB Bicyclist Longitudinal 75width', 'scenario 20 AEB Pedestrian Longitudinal 50width', 'scenario 23 AEB PedestrianChild Nearside 50width', 'scenario 24 AEB PedestrianTurning Farside 10kph', 'scenario 25 AEB PedestrianTurning Nearside 10kph'.

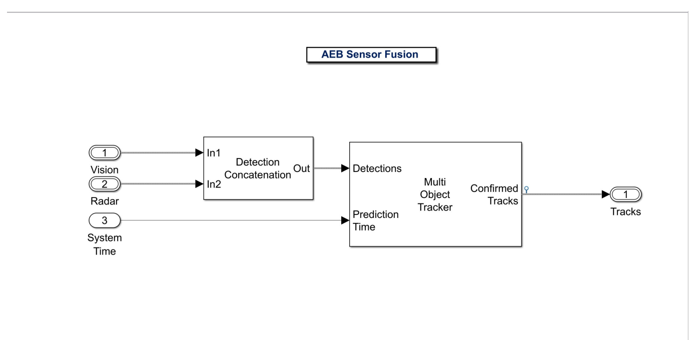

Fig. 9. AEB sensor fusion

As figure 9 shows, the system processes vision and radar detections and generates the position and velocity of the tracks relative to the ego vehicle. The detection concatenation combines the vision and radar detections onto a single output bus and the multi-object tracker performs sensor fusion and outputs the tracks of stationary and moving objects.

VI. Safety Verification

The safety monitor inspects the entire system behavior and ensures that the safety requirements are always satisfied. The safety requirements are defined by using helperAEBSetup post-load callback function. For our case, the safety

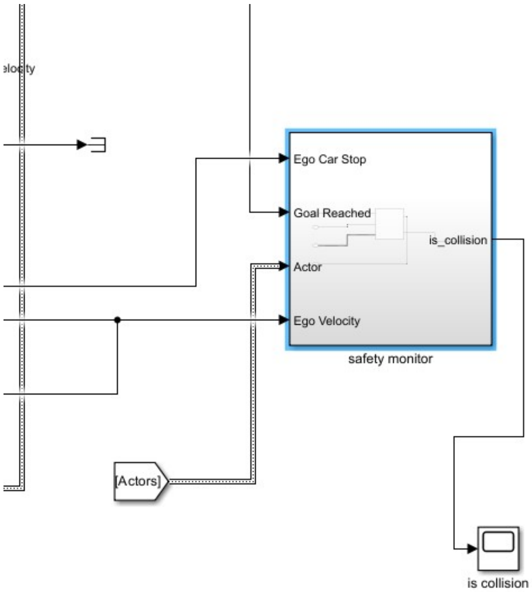

Fig. 10. Safety Monitor

requirement is defined as the relative distance (i.e the distance between the ego car and the lead car) should be always larger than a threshold value. If it is smaller than the threshold, then the safety monitor can detect such violation of safety requirement. The other safety requirement is that collision should never happen. Inside the safety monitor, it verifies whether the ego vehicle collides with the target actor at any point during the simulation.

VII. Simulations and Analysis

In our work, we tested the model for different driving scenarios. We will show the simulation results for both "scenario 25 AEB PedestrianTurning Nearside 10kph" and "scenario 23 AEB PedestrianChild Nearside 50width".

A. Scenario 25 AEB PedestrianTurning Nearside 10kph

This scenario simulates the case that the ego vehicle is driving on a cross turning road and suddenly pedestrian rushes to the middle of the road as soon as the ego vehicle is about to turn right. We tested this scenario and showed that it can safely stop before reaching the pedestrian using a pre-defined safety distance, and thus avoid hitting the pedestrian.

As figure 11 and 12 shows, it simulates the scenario that the ego car is about to turn right and detects a pedestrian suddenly appearing on the right side of a crossroad. Figure 12 shows that the ego vehicle can safely stop at a safe distance and avoid hitting the pedestrian.

As figure 13 show, when running the scenario, the scope of "is collision" is always 0 which demonstrates that it always satisfies the safety requirement. The scope of "is collision" is connected with the output of the safety monitor of the system. Whenever there is a violation happening during running the

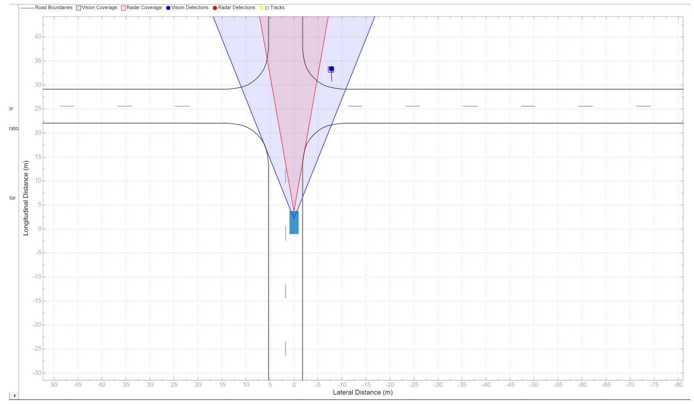

Fig. 11. Scenario 25 simulates on crossroad

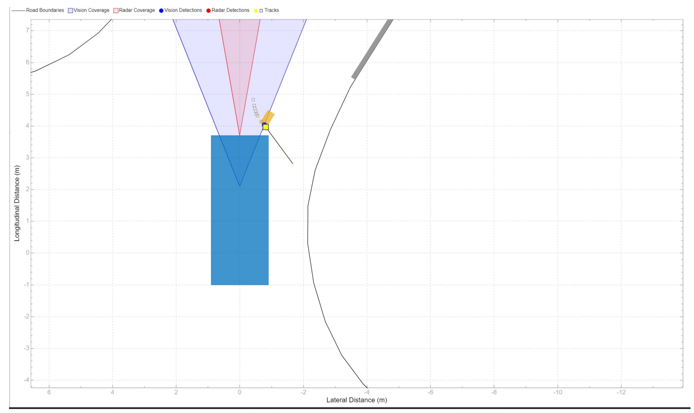

Fig. 12. Scenario 25 simulation ego car stop at safety distance

scenario, there will be a non-zero value appearing in the "is collision" scope.

The figure 14 shows the results plots of scenario 25. For example, it compares the TTC calculation time for this scenario with stopping time for this scenario. The plots also records when FCW warning activates, and the exact time when AEB status becomes active. The plots also record information about ego car acceleration, ego car yaw, yaw rate, ego car velocity, and headway.

Finally, as figure 15 shows, there is also a metric assessment that can help better visualize the status of the AEB system and FCW system. Initially, the light of FCW and the light of AEB are gray, meaning that they are both inactive. When the FCW warning activates, the light of FCW will turn red. When the 1st partial braking is activated, the light of AEB will turn yellow. And if the 2nd partial braking is activated, the light of AEB will turn orange. Finally, if the full braking is activated, the light of AEB will turn red.

B. scenario 23 AEB PedestrianChild Nearside 50width

Scenario 23 simulates that the ego vehicle is running on a straight road and there will be a pedestrian rushing into the middle of the road. Again, the ego vehicle should stop at a safe distance and avoid hitting the pedestrian.

Figure 16 shows the simulation of driving scenario 16. As shown, the ego vehicle needs to detect and stop in time before

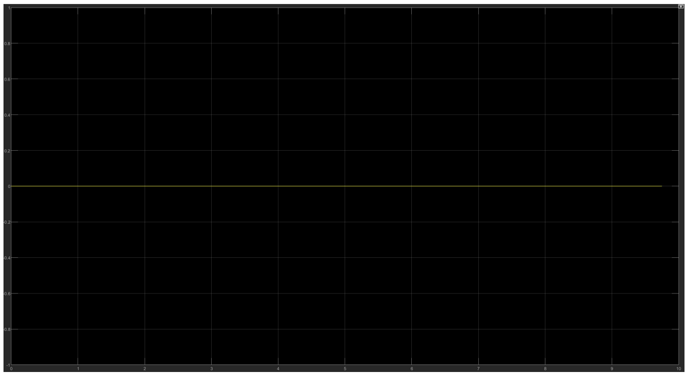

Fig. 13. verify always satisfy safety requirement for scenario 25

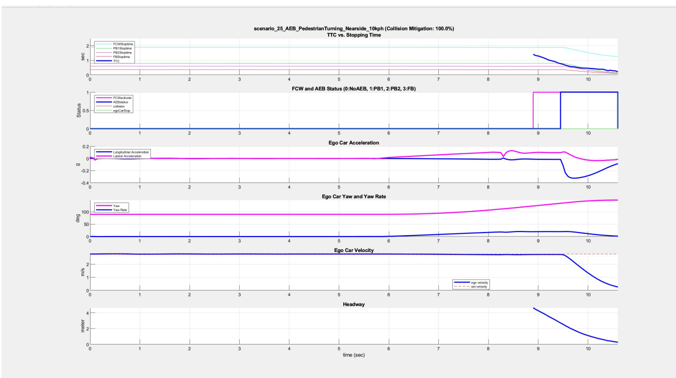

Fig. 14. scenario 25 results plots

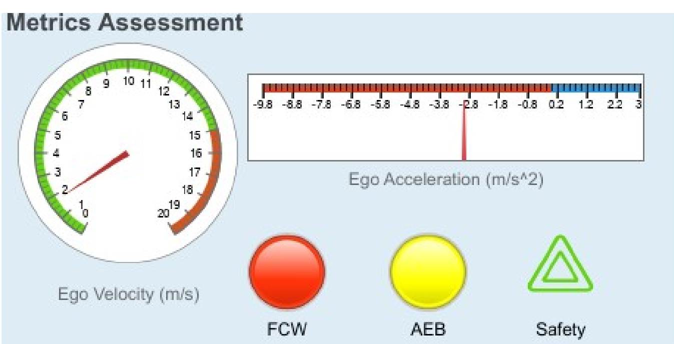

Fig. 15. metric assessment

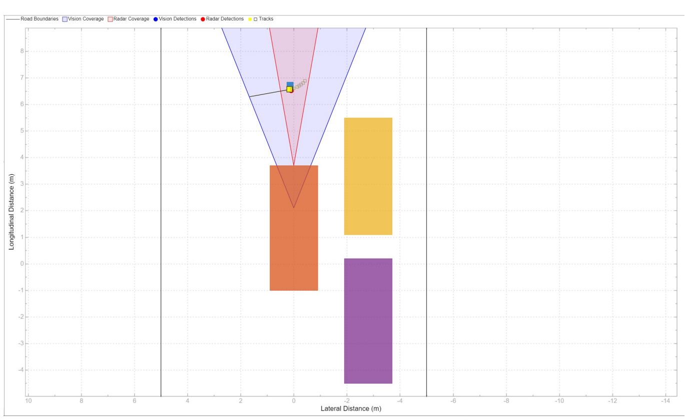

Fig. 16. Scenarios 23 Simulation

hitting the pedestrian. During the first phase of running the simulation, since there are other cars stopping at right side of the road, it is even more difficult for the ego vehicle to detect the suddenly appearing pedestrian.

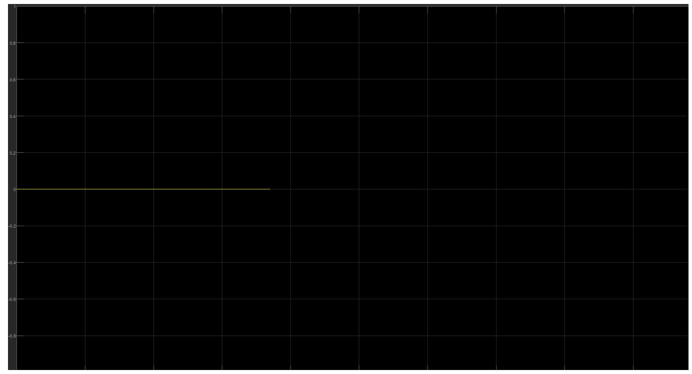

Fig. 17. Scenario 23 is collision

Figure 17 shows the "is collision" scope value for Scenario 23. As represented, the value for the "is collision" scope is still always zero which means that for scenario 23, the model also always satisfies the safety requirements.

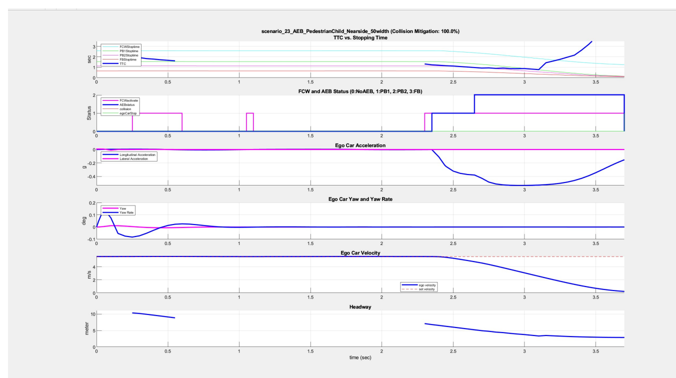

Fig. 18. scenario 23 results plots

Figure 18 shows the results plots for Scenario 23. As the same structure of results plots for Scenario 25, the plots include comparison between TTC calculation time vs. stopping time, FCW status vs. AEB status, information about ego car acceleration, ego car yaw, yaw rate, ego velocity, and headway. There are some observed interesting differences between results for Scenario 23 and results for Scenario 25 though. The first interesting observed difference is that unlike results in Scenario 25 which has decreasing trend of TTC calculation time, it turns out that TTC calculation time actually increases in Scenario 23, but for stopping time for various braking levels, the trend is still decreasing. The other interesting observed difference is that in results of Scenario 25, both FCW and AEB status changes only once at whole simulation, and for AEB status, there is either level 0 or level 1. However, for Scenario 23, with the same simulation, the FCW status changes three times and the AEB status changes two times. Unlike results in Scenario 25 which only has AEB status 0 or 1, in results of Scenario 23, there are AEB status 0, 1, and 2. Applying a higher level of braking means that Scenario 23 is actually more "dangerous" than Scenario 25. This is probably due to the fact that there are some "obstacle" cars stopping at right side of the road which making detecting pedestrian more difficultly.

VIII. Conclusion

In this final project, we implemented the project model Advanced Emergency Braking System with Sensor Fusion Technique. We tested our system in different driving scenarios, plotted the results, and verified that the model's safety requirements were being satisfied. We found that for different scenarios, there are different results for TTC calculation time, AEB status, FCW status, and stopping time. These results show that the model can verify and handle the dangerous situations efficiently.

References

[1] X. Wang, Q. Liu, F. Guo, S. Fang, X. Xu, and X. Chen, "Causation analysis of crashes and near crashes using naturalistic driving data," *Accident Analysis Prevention*, vol. 177, p. 106821, 11 2022.

[2] S. Rajendar and V. K. Kaliappan, "Recent advancements in autonomous emergency braking: A survey," in *2021 5th International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, 2021, pp. 1027–1032.

[3] A. Dixit, P. D. Devangbhai, and C. R. Kumar, "Modelling and testing of emergency braking in autonomous vehicles," in *2021 Innovations in Power and Advanced Computing Technologies (i-PACT)*, 2021, pp. 1–6.

[4] S. H. Haus, R. Sherony, and H. C. Gabler, "Estimated benefit of automated emergency braking systems for vehicleâĂŞpedestrian crashes in the united states," *Traffic Injury Prevention*, vol. 20, pp. 171–176, 11 2019.

[5] J. B. Cicchino, "Effects of automatic emergency braking systems on pedestrian crash risk," *Accident Analysis Prevention*, vol. 172, p. 106686, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0001457522001221

[6] H. Tan, F. Zhao, H. Hao, Z. Liu, A. Amer, and H. Babiker, "Automatic emergency braking (aeb) system impact on fatality and injury reduction in china," *International Journal of Environmental Research and Public Health*, vol. 17, p. 917, 02 2020.

[7] B. Fildes, M. Keall, N. Bos, A. Lie, Y. Page, C. Pastor, L. Pennisi, M. Rizzi, P. Thomas, and C. Tingvall, "Effectiveness of low speed autonomous emergency braking in real-world rear-end crashes," *Accident; analysis and prevention*, vol. 81, pp. 24–29, 04 2015.

[8] U. Z. A. Hamid, F. R. A. Zakuan, K. A. Zulkepli, M. Z. Azmi, H. Zamzuri, M. A. A. Rahman, and M. A. Zakaria, "Autonomous emergency braking system with potential field risk assessment for frontal collision mitigation," in *2017 IEEE Conference on Systems, Process and Control (ICSPC)*, 2017, pp. 71–76.

[9] J. B. Cicchino, "Effectiveness of forward collision warning and autonomous emergency braking systems in reducing front-to-rear crash rates," *Accident Analysis Prevention*, vol. 99, pp. 142–152, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0001457516304006

[10] J. Duan, R. Li, L. Hou, W. Wang, G. Li, S. E. Li, B. Cheng, and H. Gao, "Driver braking behavior analysis to improve autonomous emergency braking systems in typical chinese vehicle-bicycle conflicts," *Accident Analysis Prevention*, vol. 108, pp. 74–82, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0001457517303019

[11] S.-J. Babak, S. A. Hussain, B. Karakas, and S. Cetin, "Control of autonomous ground vehicles: a brief technical review," *IOP Conference Series: Materials Science and Engineering*, vol. 224, no. 1, p. 012029, jul 2017. [Online]. Available: https://dx.doi.org/10.1088/1757-899X/224/1/012029

[12] M. L. Fung, M. Z. Q. Chen, and Y. H. Chen, "Sensor fusion: A review of methods and applications," in *2017 29th Chinese Control And Decision Conference (CCDC)*, 2017, pp. 3853–3860.

[13] R. C. Luo, C. C. Chang, and C. C. Lai, "Multisensor fusion and integration: Theories, applications, and its perspectives," *IEEE Sensors Journal*, vol. 11, no. 12, pp. 3122–3138, 2011.

[14] A. Rattani, D. R. Kisku, M. Bicego, and M. Tistarelli, "Feature level fusion of face and fingerprint biometrics," in *2007 First IEEE International Conference on Biometrics: Theory, Applications, and Systems*, 2007, pp. 1–6.

[15] D. J. Yeong, G. Velasco-Hernandez, J. Barry, and J. Walsh, "Sensor and sensor fusion technology in autonomous vehicles: A review," *Sensors*, vol. 21, no. 6, 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/6/2140

[16] J. Kim, J. Kim, and J. Cho, "An advanced object classification strategy using yolo through camera and lidar sensor fusion," in *2019 13th International Conference on Signal Processing and Communication Systems (ICSPCS)*, 2019, pp. 1–5.

[17] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," 2015. [Online]. Available: https://arxiv.org/abs/1506.01497

[18] Y. Zhou and O. Tuzel, "Voxelnet: End-to-end learning for point cloud based 3d object detection," 2017. [Online]. Available: https://arxiv.org/abs/1711.06396

[19] V. A. Sindagi, Y. Zhou, and O. Tuzel, "Mvx-net: Multimodal voxelnet for 3d object detection," 2019. [Online]. Available: https://arxiv.org/abs/1904.01649

[20] L. Taeyoung, K. Kyongsu, Y. andJangseop, and L. Jaewan, "Development and evaluations of advanced emergency braking system algorithm for the commercial vehicle," in *Enhanced Safety of Vehicles Conference, ESV*, 2011, pp. 11–0290.

[21] A. Dixit, P. D. Devangbhai, and C. R. Kumar, "Modelling and testing of emergency braking in autonomous vehicles," in *2021 Innovations in Power and Advanced Computing Technologies (i-PACT)*, 2021, pp. 1–6.

[22] N. P. Babu M, M. Gawande, and P. Rajalakshmi, "Autonomous emergency breaking (aeb) evaluation for indian traffic scenarios using gps and lidar data," in *2022 IEEE IAS Global Conference on Emerging Technologies (GlobConET)*, 2022, pp. 655–660.

[23] M. Ariyanto, G. D. Haryadi, M. Munadi, R. Ismail, and Z. Hendra, "Development of low-cost autonomous emergency braking system (aebs) for an electric car," in *2018 5th International Conference on Electric Vehicular Technology (ICEVT)*, 2018, pp. 167–171.

[24] H.-t. Kim and B. Song, "Vehicle recognition based on radar and vision sensor fusion for automatic emergency braking," in *2013 13th International Conference on Control, Automation and Systems (ICCAS 2013)*, 2013, pp. 1342–1346.

[25] X. He, X. Ji, K. Yang, and J. Wu, "Autonomous emergency braking based on radial basis function neural network variable structure control for collision avoidance," in *2017 IEEE 2nd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, 2017, pp. 378–383.

[26] L. Zhuo, "Research on reliability of autonomous emergency braking system based on prescan," in *2022 IEEE International Conference on Advances in Electrical Engineering and Computer Applications (AEECA)*, 2022, pp. 681–684.

[27] Y. Fu, C. Li, F. R. Yu, T. H. Luan, and Y. Zhang, "A decision-making strategy for vehicle autonomous braking in emergency via deep reinforcement learning," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 5876–5888, 2020.

[28] U. Z. A. Hamid, F. R. A. Zakuan, K. A. Zulkepli, M. Z. Azmi, H. Zamzuri, M. A. A. Rahman, and M. A. Zakaria, "Autonomous emergency braking system with potential field risk assessment for frontal collision mitigation," in *2017 IEEE Conference on Systems, Process and Control (ICSPC)*, 2017, pp. 71–76.

[29] D. Lee, B. Kim, K. Yi, and J. Lee, "Development of an integrated driving path estimation algorithm for acc and aebs using multi-sensor fusion," in *2012 IEEE 75th Vehicular Technology Conference (VTC Spring)*, 2012, pp. 1–5.

[30] Y. Zhang, J. Li, and X. Liu, "Research on integrated navigation in aeb system based on multi-source fusion," in *IET International Radar Conference (IET IRC 2020)*, vol. 2020, 2020, pp. 37–42.

[31] Y. Zhu, R. Xu, H. An, A. Zhang, and K. Lu, "Research on automatic emergency braking system development and test platform," in *2022 Fifth International Conference on Connected and Autonomous Driving (MetroCAD)*, 2022, pp. 1–6.

[32] H.-t. Kim and B. Song, "Vehicle recognition based on radar and vision sensor fusion for automatic emergency braking," in *2013 13th International Conference on Control, Automation and Systems (ICCAS 2013)*, 2013, pp. 1342–1346.

[33] T. Dreossi, A. DonzÃl', and S. A. Seshia, "Compositional falsification of cyber-physical systems with machine learning components," 2017. [Online]. Available: https://arxiv.org/abs/1703.00978