International Journal of Computer Applications (0975 – 8887)
Volume 186 – No.34, August 2024

Adaptive Headlight Control and Real-Time Pedestrian Detection

Mohammed Zakaria
Electronics and Communication Dept.
Faculty of Engineering
Helwan University

Esraa Yaser
Electronics and Communication Dept.
Faculty of Engineering
Helwan University

Mohammed Rooby
Electronics and Communication Dept.
Faculty of Engineering
Helwan University

Mohammed Hamed
Electronics and Communication Dept.
Faculty of Engineering
Helwan University

Mohammed Mahmoud
Electronics and Communication Dept.
Faculty of Engineering
Helwan University

Azza M. Anis
Electronics and Communication Dept.
Faculty of Engineering
Helwan University

ABSTRACT

In rural areas, illumination is weak due to a lack of ambient light, and the risk of being in a fatal crash is higher at night compared to the day. Besides, headlamps are not properly utilized because drivers ignore oncoming vehicles and keep on utilizing their high beams that induce glare for approaching cars, resulting in temporary blindness. Therefore, having a clear view of the road and nighttime object recognition are critical in this situation. This paper introduces the design of an adaptive headlight control system for automobiles with an artificial intelligence object detection module to detect objects in front of a car. The system includes a camera-radar sensor fusion model to evaluate the sensor's readings and fuse them into one readout for a microcontroller unit (MCU). Then, the MCU controls servo motors mounted to light emitting diode (LED) arrays and rotates them right or left according to the steering wheel sensor angle. Hence, the intensity of the headlight LEDs is adapted to reduce the glare on oncoming vehicles and enhance vision at night. The reliability of the designed system is verified by Proteus. Then, a prototype for the proposed design is implemented and tested at different road scenarios.

Keywords

Adaptive Headlight System, CARLA, CNN, Deep Learning, Object Detection, Roboflow Dataset, YOLO v8, Sensor Fusion, Raspberry Pi.

1. INTRODUCTION

The safe development of road transportation facilitates the movement of people and goods and provides easy access to various social and economic services. With the quick expansion of the road network and motorization, the problem of road accidents appeared. According to the world health organization (WHO) reports, road traffic accidents result in approximately 1.19 million people dying annually and 20–50 million injuries. These accidents frequently happen at night despite having less traffic. This is due to blurred vision and using conventional headlighting [1].

Therefore, numerous researchers have developed headlight systems that can be controlled to give the best visibility when driving circumstances vary. One of the reviewed designs used steering angle sensor technology for adaptive lighting installations in cars [2]. These sensors use the steering wheel's location to determine how to reposition the lights. Then, the adaptive lighting system uses the sensor data to pivot the headlamps in the direction of the turn when the steering wheel is turned. This is a very useful choice if your route winds through rural or mountainous areas and is winding. However, this tactic might not work as well if the motorist travels in a straight line. In another study, the global positioning system (GPS) technology was used to produce adaptive lighting [3]. In this technique, the position of a car and the direction of the road ahead are determined by GPS sensors. Then, the adaptive headlight technology modifies the direction of the beam accordingly. This approach can be particularly useful in rural or mountainous areas with narrow or steep roadways. But in cities, where roads are more likely to be straight than curved, this tactic might not work as well. In [4], an adaptable headlight system was made with light sensors. These sensors utilize the surrounding light measurements to adjust the brightness of the headlights. By reducing glare, this may enhance eyesight in low-light conditions. This tactic works well only while driving in an environment where ambient light levels fluctuate, like at sunrise or sunset.

In this paper, we propose a system for adaptive headlight control for vehicles to make nighttime driving safer and to decrease road accidents at night. The proposed system automatically adjusts the vertical and horizontal light axis by using a camera module and sensors. In addition to that, we utilize you-only-look-once (YOLO) deep learning object detection algorithm to detect the objects in the frames from the camera module. We also incorporate a sensor fusion model to fuse sensors' readings and filter errors. Then, a microcontroller processes all of this data to control the headlights in different road scenarios.

2. PROPOSED SYSTEM

The block diagram for the developed adaptive headlight control system is indicated in Figure 1. The microcontroller unit

(MCU), which interfaces with input and output devices, is the primary part. The system includes a camera to record frames, which are then processed and analyzed by an object detection algorithm to identify the presence of vehicles, pedestrians, and traffic signs. A radar sensor is also used to measure the separation between a vehicle and oncoming traffic as well as to calculate the vehicle's speed and direction of motion. Additionally, a sensor fusion model is implemented by the Kalman filter procedure. It takes the readings from the camera and radar sensor frames and fuses them to remove errors and noise, produce clean readings, and deliver one input source reading to the MCU. As well, the MCU takes data from a light-dependent resistor (LDR) sensor to determine whether to turn on the high beam or the low beam based on the amount of ambient light detected. Besides, a steering wheel angle sensor measures the steering wheel's angle and sends the readings to the MCU so it can determine the angle and direction at which to rotate the light beam. Finally, the MCU controls servo motors, which are utilized to spin the light emitting diodes (LEDs) inside the headlight system. The operation of the designed system is presented in the flowchart given in Figure 2.

3. METHODOLOGY

This section provides the theory of operation, design, and implementation of each part in the proposed system.

3.1 Object Detection

A computer recognition technique called object detection enables a software program to identify, find, and track an object in an image or video. Object detection has revolutionized because of deep learning algorithms, which have greatly increased accuracy and speed [5, 6]. Therefore, this paper uses the deep convolutional neural network (CNN) technique in the well-known YOLO family of object identification algorithms for real-time detection [7, 8]. The YOLO algorithm locates objects by splitting the input snapshot into grids and calculating the sophistication probability and bounding boxes for each grid square, as shown in Figure 3. YOLO version 8 (v8) is selected with the Roboflow dataset [9] for the real-time detection of vehicles, traffic signs, and pedestrians.

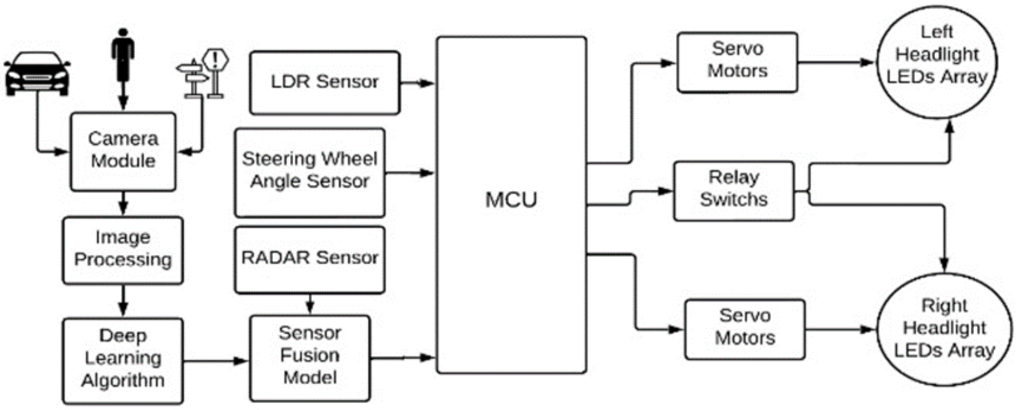

Fig 1: Block diagram of the proposed design

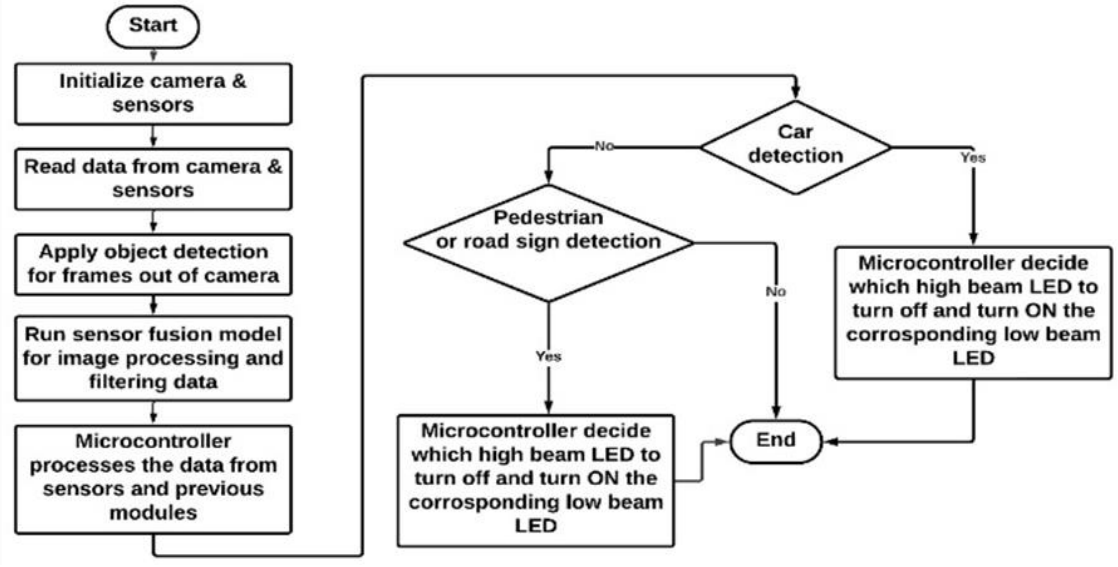

Fig 2: Operation flowchart of the system

This algorithm is used for high-speed detection and classification in real-time, as well as its ability to deal with many objects in an image with less effect from time transitions between frames. In the proposed system, the object detection algorithm receives a video from the camera then its frames are divided into three sections, each of which influences a specific

LED fitted in the headlight to limit the intensity of the light provided to a specific area in the front of a car when the model detects another automobile in the frame.

Then, the dataset is divided into three classes: vehicles, traffic signs, and pedestrians to train the model. Each class has a different attribute for the MCU when detected in a frame. When a vehicle is detected, the MCU controls the intensity of the LEDs. When a traffic sign is detected, the MCU sends a notification to the driver with instructions on the sign. When pedestrians are detected, the MCU notifies the driver of their presence and reduces the intensity of the headlight LED that is connected to its specific area in the frame.

3.2 Camera-Radar Sensor Fusion

A process of combining data from several sensors to create a comprehensive, reliable, and accurate understanding of a system is known as a sensor fusion [10].

Numerous industries, such as robotics, self-driving cars, aerospace, healthcare, and consumer electronics, are based on this technology [11-16].

Several strategies are used to successfully combine data from several sensors. One of the most popular methods is the Kalman filter [17]. Kalman filters are recursive algorithms that estimate the state of a dynamic system based on a set of data [18]. As seen in Figure 4, the algorithm operates in two stages: prediction and updating. During the prediction phase, the Kalman filter generates estimates of the current state variables and their uncertainty. These estimates are updated using a weighted average. Then, the estimates of higher confidence gain more weight when the next measurement is taken [19].

The process is iterative. With just the current input measurements, the state that has already been established, and the uncertainty matrix, it can operate in real-time without the need for further past data. The major purpose for using the Kalman filter in the proposed system is to eliminate the sensor reading errors and to combine camera frames with radar sensor data to offer one decision to the MCU.

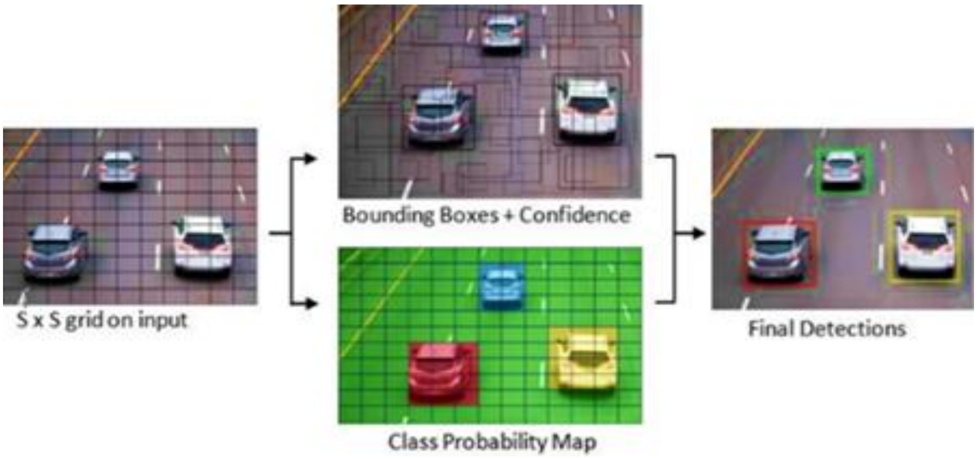

Fig 3: YOLO methodology [8]

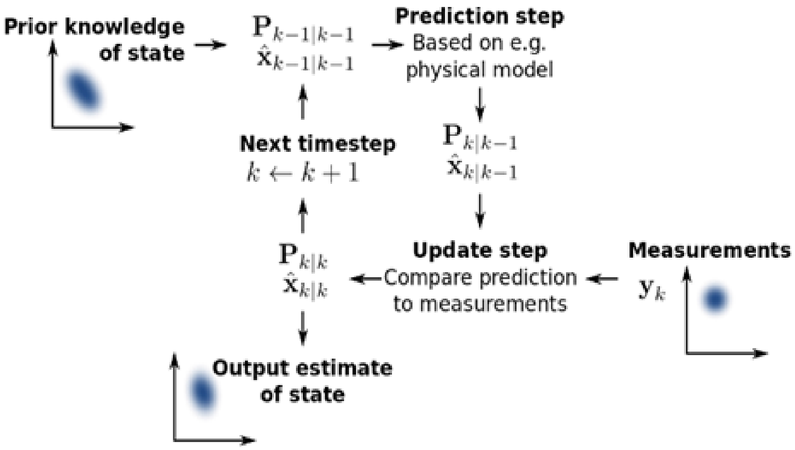

Fig 4: Phases of Kalman filter [19]

3.3 CARLA

An open-source platform called car-learning-to-act (CARLA) is used in applications of autonomous vehicles [20]. CARLA features freely used open digital assets (such as buildings, cars, and urban layouts) to create driving scenarios. The simulation platform makes it possible to define ambient conditions flexibly. It is designed as an unreal engine open-source layer [21]. The CARLA Python API [22, 23] sends commands to the server to modify the environment, like the number of vehicles, pedestrians, and weather conditions.

4. RESULTS AND ANALYSIS

In this design, CARLA generates different nighttime road scenarios such as presented in Figure 5. Then, the generated scenarios are fed into the object detection model.

The results of the detection model are implemented using YOLO v8 with a medium size after testing different sizes for the model as listed in Table 1. The table shows that the medium size of YOLO v8 provides acceptable accuracy and high speed in real-time processing.

Table 1. Sizes and accuracies for YOLO v8 model

Size	No. of Epochs	Data Size	Accuracy	Splitting Ratio
Nano	25	5671	67.25%	Train: 70% Validation: 20% Test: 10%
Small			69.03%	
Medium			71.25%	
Large			74.06%	

Then, the Roboflow dataset is used to train the model for different road conditions. After the training process is done the confusion matrix is generated to check the validation of the model and the performance of the classification. The generated matrix compares the true and the predicted values for each class as illustrated in Figure 6. As shown in the figure, the matrix gives the prediction summary for 14 trained classes: no turn, bicycle, forbidden to turn left, motorcycle, no parking, person, speed limit 30, speed limit 40, speed limit 60, speed limit 90, stop sign, traffic light, trunk, vehicle.

In addition, the loss curves are represented in Figure 7 to estimate how far off the model's predictions are from the true values. The loss value measures the model’s effectiveness during the training process, where the lower loss values mean better model performance and the larger values indicate worse performance. The curves display the effect of different types of loss on the training and the validation process versus the number of epochs. The “box_loss” describes the bounding-box regression-loss, which indicates the difference between the predicted bounding-box dimensions and the actual values. The “cls_loss” stands for the classification-loss and shows the difference between the expected and the actual class probabilities for each object in the frame. The "dfl_loss” refers to the distribution-focal-loss, which calculates the model’s capability to detect objects of various features. As seen in the figure, all loss curves converge to a low and stable value during training and validation. After that, the bounding boxes of the detection process with the detected classes are presented on CARLA images in Figure 8. As shown, the vehicles, persons, speed limit 60, traffic light, and forbidden to turn left are precisely defined by the proposed Roboflow-based YOLO-trained model.

Fig 5: Different nighttime road scenarios: a vehicle on two-way road, a vehicle is turning right, a vehicle on one-way road (right-lane), and a vehicle is changing lane are presented clockwise from the top left

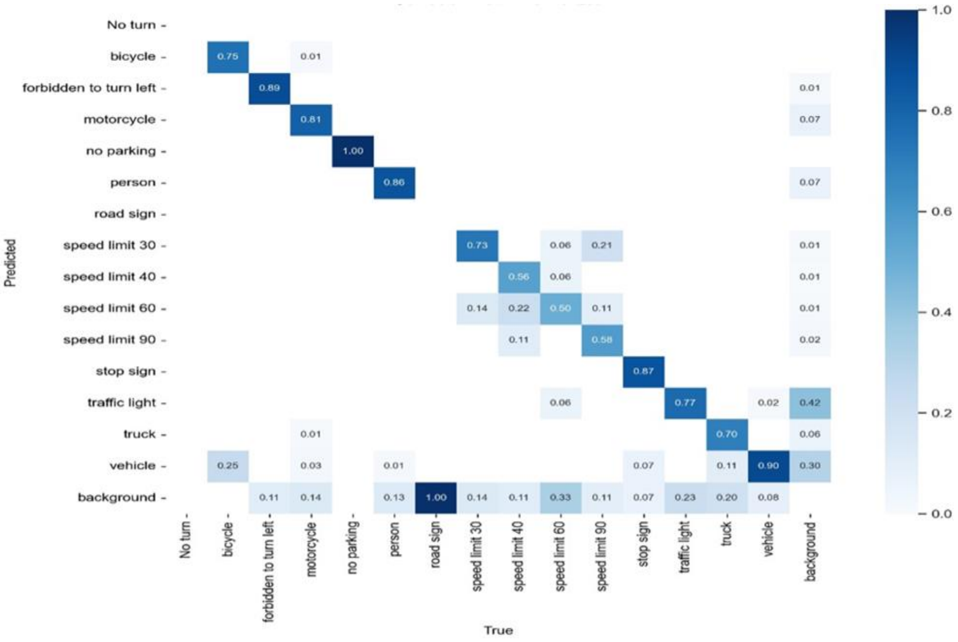

Fig 6: Confusion matrix normalized

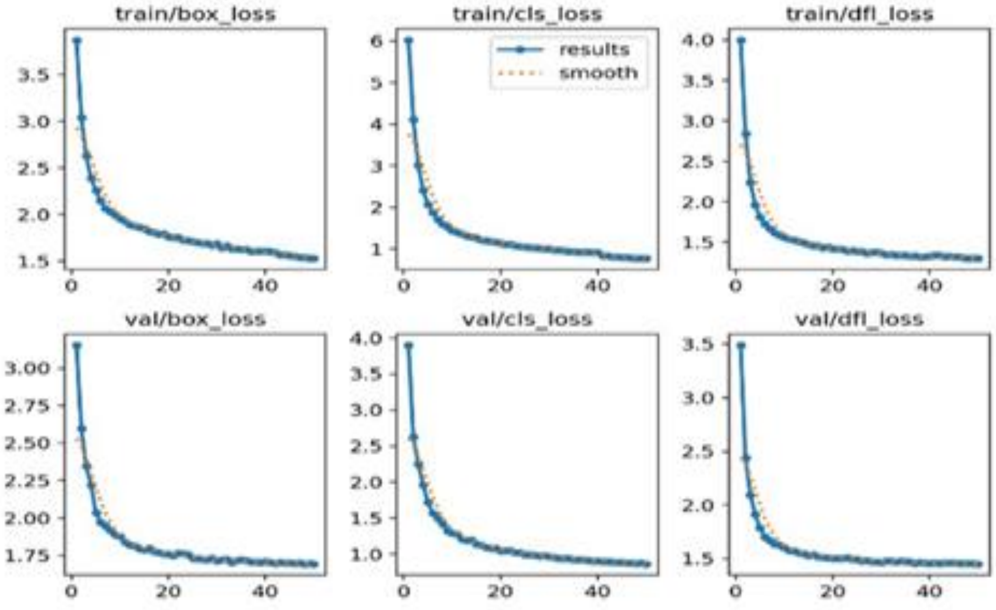

Fig 7: Training and validation loss curves

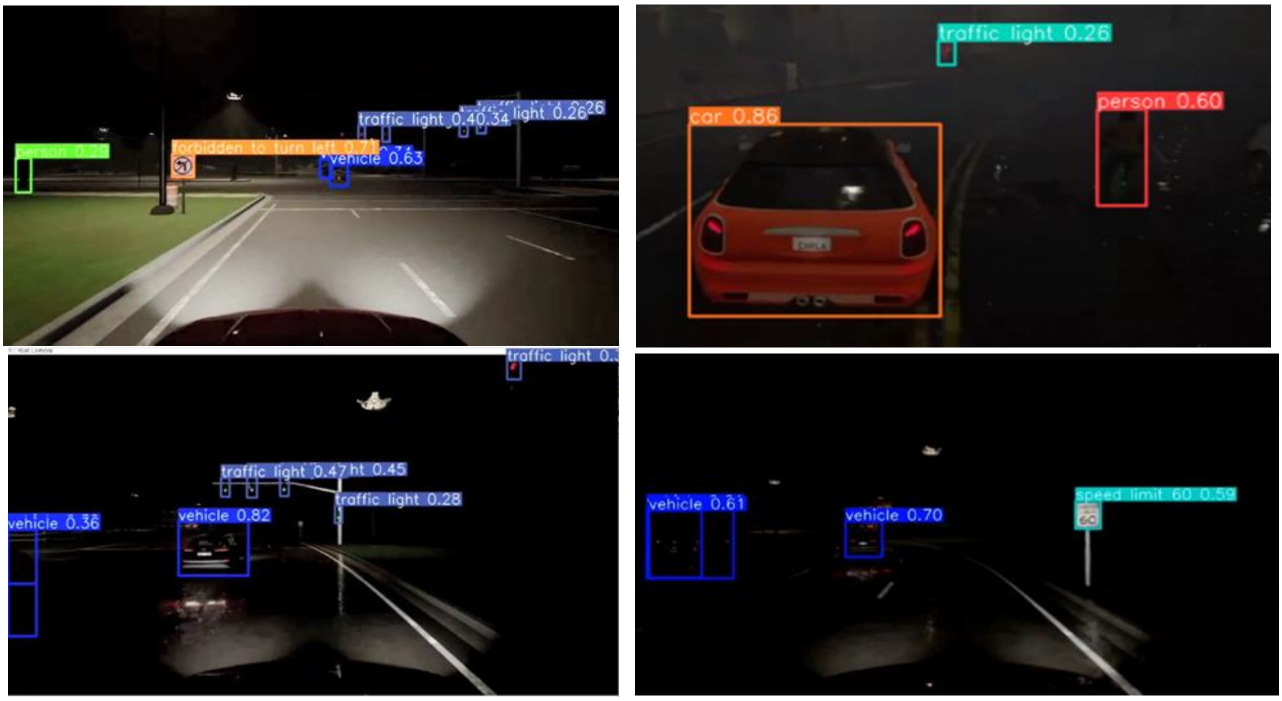

Fig 8: Detection of the objects in different nighttime weather conditions: clear nighttime, foggy weather, rainy weather, and heavy rain weather are presented clockwise from the top left

Then, the whole system is implemented on Proteus [24] as displayed in Figure 9. The results show how the MCU (Raspberry Pi [25]) interfaces with the LEDs and the servo motor. The MCU operates according to CARLA sensors and the sensor fusion model. The MCU changes the ON/OFF status of the LEDs according to the road-light conditions and moves the servo motor left (Figure 9(a)) or right (Figure 9(b)) according to the steering wheel angle position. The angle can be positive, negative, or zero. The effect of the steering wheel angle on the servo motor movement is displayed in the hardware implementation in Figure 10. In the proposed design, the servo motor is turned off when the steering wheel angle is from 0 to 15° as shown in Figure 10(a). The servo motor moves right with an angle of 45° and 80° when the steering wheel angle ranges from 16° to 30° (Figure 10(b)) and greater than 30° (Figure 10(c)), respectively.

Subsequently, Blender software [26] is used to create a model for the introduced system. Finally, a prototype for the system is designed and tested in different nighttime scenarios as illustrated in Figure 11. The scenario presented in Figure 11(a), shows that the adaptive system turns ON all high-beam LEDs because no objects are detected. After that in Figure 11(b), the headlight system turns off the second left LED, to reduce the effect of the glare on the detected vehicle. In Figure 11(c), the system turns off two LEDs for more glare reduction on the detected vehicles and enhances seeing ability.

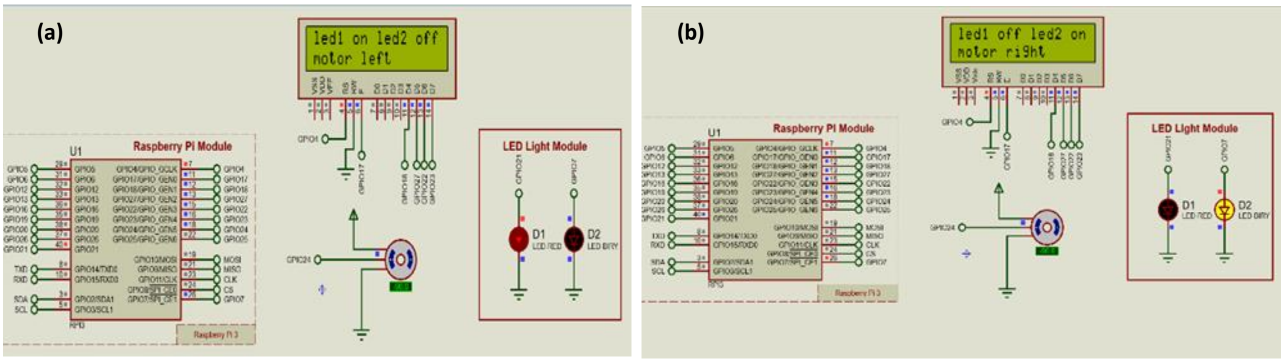

Fig 9: Proteus results when the road scenario decides: (a) turns ON the left LED and rotates left the servo motor, (b) turns ON the right LED and rotates right the servo motor

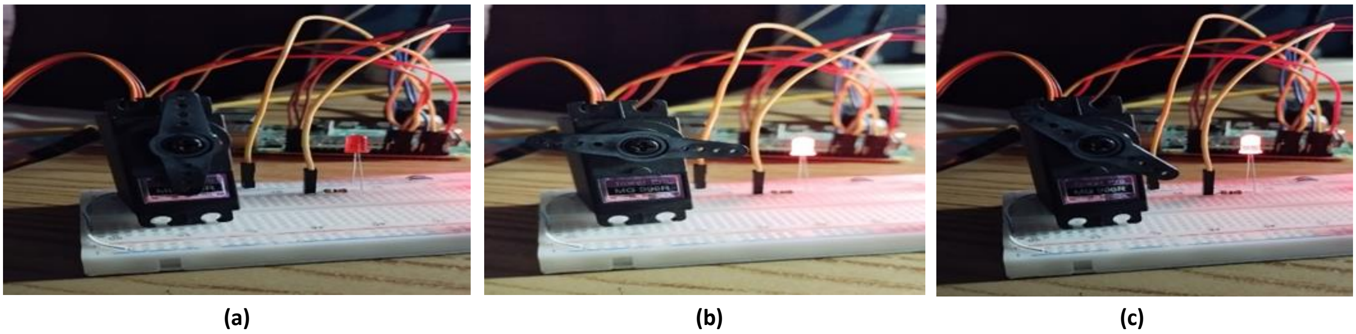

Fig 10: Different conditions for the servo motor when: (a) turned off, (b) moved right at an angle of 45°, and (c) moved right at an angle of 80°

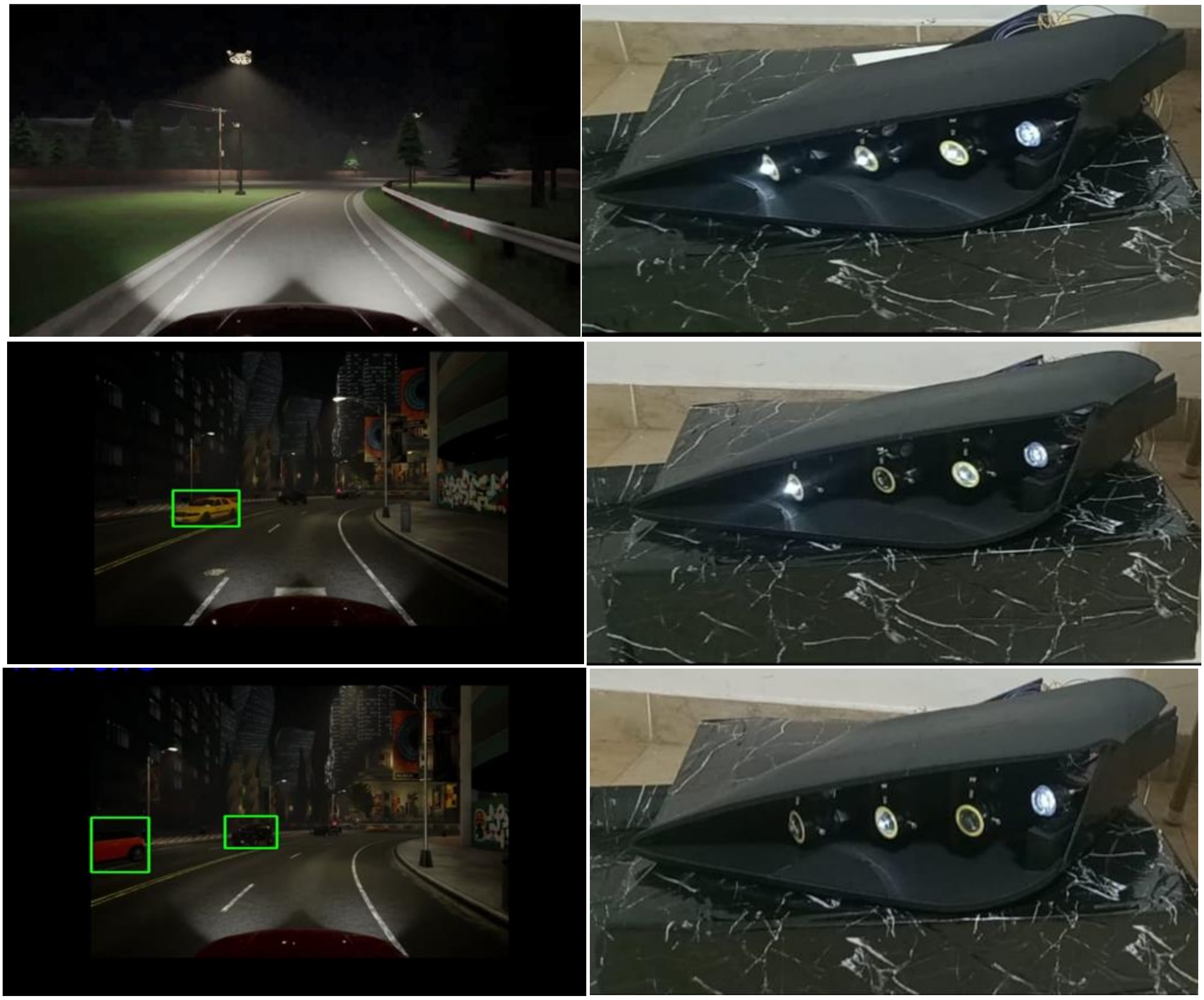

Fig 11: Testing the prototype of the adaptive headlight control system when: (a) turning ON all LEDs when no objects detected, (b) turning off one LED, and (c) turning off two LEDs to reduce the effect of the glare on the detected vehicles

5. CONCLUSION

In this paper, an adaptive headlight system is designed for drivers who care about road safety at night. Based on CARLA nighttime scenarios, different road conditions have been tested with the aid of the YOLO v8 object detection algorithm. The obtained results have demonstrated that the YOLO v8 algorithm is an efficient object detection platform for improving speed and accuracy in comparison to its predecessors. The Kalman filter camera-radar fusion model has been utilized to reduce the error of sensors' readings. The detected vehicles, pedestrians, and traffic signs have been passed to the MCU, and the intensity of the headlight LEDs has been adapted accordingly. The designed adaptive headlight system has all the advantages of the more expensive, high-end adaptive lighting systems provided by luxury automobile manufacturers. The presented system is more affordable, needs fewer parts, and is easier to install, making it more user-friendly for a larger variety of applications.

6. REFERENCES

[1] World health organization: road traffic injuries. https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries

[2] Youn, J., Di-Yin, M., Cho, J., and Park, D. 2015. Steering wheel-based adaptive headlight controller with symmetric angle sensor compensator for functional safety requirement. In 2015 IEEE 4th Global Conference on Consumer Electronics (GCCE). IEEE. 619-620).

[3] Xiong, S., Tan, G., Guo, X., and Xiao, L. 2017. The research of the adaptive front lighting system based on GIS and GPS. SAE Technical Paper.

[4] Ahmed, I. 2020. Adaptive light control systems. Technical Report.

[5] Li, G., Yang, Y., and Qu, X. 2019. Deep learning approaches on pedestrian detection in hazy weather. IEEE Transactions on Industrial Electronics. 67(10), 8889-8899.

[6] Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., and Farhan, L. 2021. Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. Journal of Big Data, 8, 1-74.

[7] Saraf, P., Watve, S., and Kulkarni, A. 2022. Object detection: literature review. In International Conference on Soft Computing and Pattern Recognition. Cham: Springer Nature Switzerland. 485-494.

[8] Terven, J., Córdova-Esparza, D. M., and Romero-González, J. A. 2023. A comprehensive review of yolo architectures in computer vision: from yolov1 to yolov8 and yolo-nas. Machine Learning and Knowledge Extraction. 5(4), 1680-1716.

[9] Roboflow dataset. https://roboflow.com

[10] Yeong, D. J., Velasco-Hernandez, G., Barry, J., and Walsh, J. 2021. Sensor and sensor fusion technology in autonomous vehicles: a review. Sensors. 21(6), 2140.

[11] Novak, D., and Riener, R. 2015. A survey of sensor fusion methods in wearable robotics. Robotics and Autonomous Systems. 73, 155-170.

[12] Mendes-Jr, J. J. A., Vieira, M. E. M., Pires, M. B., and Stevan-Jr, S. L. 2016. Sensor fusion and smart sensor in sports and biomedical applications. Sensors. 16(10), 1569.

[13] Fung, M. L., Chen, M. Z., and Chen, Y. H. 2017. Sensor fusion: a review of methods and applications. In 2017 29th Chinese Control and Decision Conference (CCDC). IEEE. 3853-3860.

[14] Sengupta, A., Yoshizawa, A., and Cao, S. 2022. Automatic radar-camera dataset generation for sensor-fusion applications. IEEE Robotics and Automation Letters. 7(2), 2875-2882.

[15] Tyagi, N., Chavan, S. S., and Gangadharan, S. M. P. 2023. Towards intelligent industrial systems: a comprehensive survey of sensor fusion techniques in IIoT. Measurement: Sensors. 100944.

[16] Abdelkader, M. R., Abdullah, E. T., Mohamed, R. A., Salam, R. K., Mohamed, O. Y., and Anis, A. M. 2023. AStar-algorithm based voice-controlled wheelchair for quadriplegic patients. International Journal of Computer Applications. 185(34), 31-35.

[17] Khodarahmi, M., and Maihami, V. 2023. A review on Kalman filter models. Archives of Computational Methods in Engineering. 30(1), 727-747.

[18] Liu, H., Hu, F., Su, J., Wei, X., and Qin, R. 2020. Comparisons on Kalman-filter-based dynamic state estimation algorithms of power systems. IEEE Access. 8, 51035-51043.

[19] Zhong, C., Darbandi, M., Nassr, M., Latifian, A., Hosseinzadeh, M., and Navimipour, N. J. 2024. A new cloud-based method for composition of healthcare services using deep reinforcement learning and Kalman filtering. Computers in Biology and Medicine. 172, 108152.

[20] Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., and Koltun, V. 2017. CARLA: an open urban driving simulator. In Conference on Robot Learning. PMLR. 1-16.

[21] Unreal engine. https://www.unrealengine.com

[22] Python API tutorial - CARLA simulator.

https://carla.readthedocs.io/en/0.9.2/python_api_tutorial

[23] Gao, W., Tang, J., and Wang, T. 2021. An object detection research method based on CARLA simulation. In Journal of Physics: Conference Series. IOP Publishing. 1948(1), 012163.

[24] Proteus: PCB design and circuit simulator software. https://www.labcenter.com

[25] Raspberry Pi foundation. https://www.raspberrypi.org

[26] Blender software. https://www.blender.org