DESIGN AND IMPLEMENTATION OF REAL-TIME AUTONOMOUS VEHICLE USING IOT BASED IMAGE PROCESSING

Narender Chinthamu[1], George Chellin Chandran J[2], A.Usha Ruby[3], C. Thangavel[4], Vjay Birchha[5], Himanshu Pant[6]

Article History: Received: 12.02.2023 **Revised:** 27.03.2023 **Accepted:** 12.05.2023

Abstract

The design and implementation of a real-time autonomous vehicle using IoT-based image processing is presented in this research. The system incorporates a sensor camera, Raspberry Pi, and a Convolutional Neural Network (CNN) program to detect and classify various vehicles and obstacles in the surrounding environment. The captured images are transmitted to the Raspberry Pi, where a Python program executes the CNN algorithm for object identification. Upon detection, the system sends signals to the motor controller, reducing the motor speed accordingly. Additionally, voice commands are used to alert the user about the presence of objects. Ultrasonic and proximity sensors further enhance the system's capabilities by providing real-time feedback. The research results demonstrate the successful detection and classification of different vehicles, validating the effectiveness of the CNN program. The integration of motor speed control and voice commands enhances safety and user experience. The developed system holds promise for enhancing the intelligence and safety of autonomous vehicles, paving the way for further advancements in real-time object detection and response.

Keywords: CNN, IoT, Python, Speed control

[1]MIT (Massachusetts Institute Of Technology) CTO Candidate, Senior Enterprise Architect , Dallas, Texas USA
[2]professor, School of Computing Science and Engineering & Director Student Welfare, VIT-BHOPAL University, Kothrikalan, Ashta, Madhya Pradesh- 466114, India
[3]Associate Professor, School of Computing Science and Engineering, VIT Bhopal University, Kothrikalan, Sehore, Madhya Pradesh-466114, India
[4]Associate Professor, Department of Mechanical Engineering, Vinayaka Mission's Kirupananda Variyar Engineering College, Vinayaka Mission's Research Foundation (Deemed to be University), Salem,Tamilnadu, India
[5]Assistant Professor, Department of Computer Science and Engineering Swami Vivekanand College of Engineering, Indore, Madhya Pradesh, India, Pin 452001
[6]Graphic Era Hill University Bhimtal Campus Uttarakhand India

Email: [1]narender.chinthamu@gmail.com, [2]chellin1968@gmail.com, [3]ausharuby@gmail.com, [4]ceeteemech@gmail.com, [5]vijaybirchha@gmail.com, [6]hpant@gehu.ac.in

DOI: 10.31838/ecb/2023.12.s3.303

1. Introduction

In recent years, the advancement of autonomous vehicles has gained significant attention due to its potential to revolutionize transportation systems. Autonomous vehicles rely on various technologies, including image processing and machine learning algorithms, to perceive and navigate their surroundings. This literature review aims to explore the existing research related to the design and implementation of real-time autonomous vehicles using IoT-based image processing, specifically focusing on object detection and motor control [1].
Object detection plays a crucial role in autonomous vehicles as it enables them to identify and classify various objects in their environment. Traditional computer vision techniques, such as Haar cascades and Histogram of Oriented Gradients (HOG), have been widely used for object detection. These methods involve hand-crafted features and machine learning classifiers to identify objects based on predefined patterns. However, their performance can be limited by variations in lighting conditions and object appearances [2]–[4].

Deep learning techniques, particularly Convolutional Neural Networks (CNN), have emerged as powerful tools for object detection in recent years. CNNs can automatically learn and extract features from raw image data, making them highly effective in complex object recognition tasks. Several studies have utilized CNNs for vehicle detection and classification in autonomous vehicles. [5]–[7]. The integration of Internet of Things (IoT) technologies with image processing has further enhanced the capabilities of autonomous vehicles. IoT-based systems enable seamless communication between the vehicle's sensors and processing units, facilitating real-time data analysis and decision-making. In an research, developed an IoT-based autonomous driving system that utilized image processing techniques to detect and track objects, enabling safer navigation [8]–[10]. Motor control is a critical aspect of autonomous vehicles as it allows them to adapt to changing road conditions and objects in their path. Various studies have explored motor control techniques for speed adjustment based on object detection. For instance, autonomous vehicle system that used sensor inputs, including image processing results, to control the motor speed and ensure safe navigation. The system effectively reduced the speed upon object detection, demonstrating its ability to mitigate collision risks [11], [12].
In addition to camera-based image processing, the integration of ultrasonic and proximity sensors further enhances the safety and situational awareness of autonomous vehicles. Ultrasonic sensors provide distance measurements, allowing vehicles to detect objects in close proximity. Proximity sensors, on the other hand, detect the presence of objects within a specified range. Combining these sensors with image processing techniques can significantly improve object detection and collision avoidance capabilities [13], [14].
While significant progress has been made in the field of autonomous vehicles using IoT-based image processing, several challenges and future directions need to be addressed. One of the main challenges is the real-time processing and response requirements of autonomous vehicles. As vehicles operate in dynamic environments, ensuring low-latency processing and decision-making is crucial. Moreover, improving the robustness and generalization capabilities of object detection algorithms, such as CNNs, remains an ongoing research area.
Future research should also focus on the integration of advanced technologies, such as LiDAR and radar systems, to complement camera-based image processing. LiDAR sensors provide accurate 3D mapping of the environment, enabling better object recognition and localization. Radar systems, on the other

hand, can enhance object detection capabilities, particularly in adverse weather conditions [15], [16].

2. Methodology

Autonomous vehicles have been a topic of interest for researchers and industry experts alike. These vehicles offer many benefits, such as improving safety, reducing traffic congestion, and increasing efficiency. However, autonomous vehicles require advanced technologies such as image processing, artificial intelligence, and the Internet of Things (IoT) to function properly. This research aims to design and implement a real-time autonomous vehicle system that utilizes IoT-based image processing. The proposed system consists of a camera that captures images of the vehicle's surroundings. The images are then communicated to a Raspberry Pi, which acts as the central computing platform. A Python program is used to analyze the images and identify various objects in the photos. To achieve this, Convolutional Neural Networks (CNNs) are implemented in the Python program for object recognition. The system also uses various sensors such as ultrasonic and proximity sensors and a camera to provide real-time feedback on the vehicle's environment.

One of the key features of the proposed system is its ability to control the vehicle's speed through the current controller and provide real-time audio feedback to the user. When the system identifies an obstacle, it sends a signal to the current controller of the electric vehicle to reduce the speed of the motor. Simultaneously, the system indicates the presence of an obstacle to the user using voice commands. This enhances the user's situational awareness and reduces the likelihood of accidents. The user interface of the proposed system is designed to be intuitive and user-friendly. The voice commands make it easier for users to interact with the system and provide real-time feedback on the vehicle's environment. The system's ability to provide audio feedback enhances the user's situational awareness and reduces the likelihood of accidents. The experimental setup involves testing the proposed system in a controlled environment to evaluate its performance. The materials used include a Raspberry Pi, a camera, various sensors, and an electric vehicle. The system's performance is evaluated based on its ability to identify objects, provide real-time feedback, and control the vehicle's speed.

The experimental results demonstrate the effectiveness of the proposed system. The system can identify objects and provide real-time feedback on the vehicle's environment, and it can control the vehicle's speed to enhance safety. The system's ability to provide audio feedback also enhances the user's situational awareness, which is critical for autonomous vehicles. The research also highlights potential areas for improvement and future research opportunities in the field of autonomous vehicles and IoT-based image processing. One of the main areas for improvement is the accuracy of object recognition. The CNN implemented in the Python program can be further optimized to increase its accuracy in identifying objects. Another potential area for improvement is the system's response time. While the proposed system can provide real-time feedback, reducing the system's response time can further enhance safety. Future research can explore ways to reduce the system's response time, such as using faster sensors and optimizing the CNN's architecture.
Figure 1: Block Diagram of the Real-Time Autonomous Vehicle System.
Camera: This component captures images of the vehicle's surroundings, including various vehicles and obstacles. Raspberry Pi: The Raspberry Pi serves as the central computing platform. It receives the captured images from the camera and processes them using the implemented algorithms. Image Processing: The captured images are processed using a Python program running on the Raspberry

Pi. This program employs computer vision techniques to analyze the images and identify various objects present. Convolutional Neural Network (CNN): Within the Python program, a CNN is implemented. This deep learning model is responsible for object recognition within the images. It has been trained on a labeled dataset and is capable of detecting and classifying different objects. Object Recognition Output: The output of the CNN provides information about the identified objects within the captured images. It includes the type and location of each object. Current Controller: This component is responsible for controlling the current supplied to the electric vehicle's motor. It receives signals from the object recognition output to adjust the current and reduce the motor's speed when obstacles are detected. User Interface: The user interface enables interaction with the system. It includes elements such as a display screen and audio output for voice commands. Voice Commands: The system uses voice commands to provide real-time feedback to the user about the presence of objects. These voice commands serve as audio alerts, informing the user about potential obstacles detected by the system. Ultrasonic Sensor: An ultrasonic sensor is utilized to measure the distance between the vehicle and nearby objects. This sensor provides additional input for object detection and helps in determining the proximity of obstacles. Proximity Sensor: The proximity sensor serves a similar purpose to the ultrasonic sensor, measuring the distance between the vehicle and objects in its vicinity. It further enhances the system's ability to detect and respond to obstacles. Output: The final output of the system includes both the control signals sent to the current controller to adjust the motor's speed and the voice commands alerting the user about detected objects.

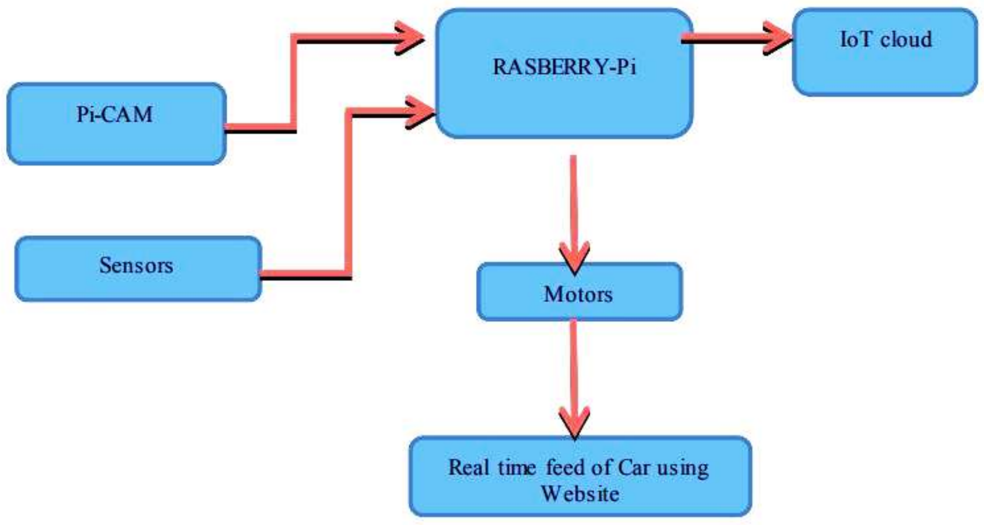

Fig. 1 Architecture of the proposed system

Convolution Neural Network

Convolutional Neural Networks (CNNs) have proven to be highly effective for identifying vehicles in the context of real-time autonomous vehicle systems. CNNs are a type of deep learning model specifically designed for analyzing and recognizing patterns in visual data. In your research, CNNs can be used to process the captured images from the camera and accurately identify different types of vehicles and obstacles. The key advantage of CNNs lies in their ability to automatically learn hierarchical representations of data. In the case of vehicle identification, CNNs can learn to recognize low-level features such as edges, corners, and textures that are indicative of different vehicle classes. As the network deepens, higher-level features, such as wheels, windows, and vehicle shapes, are detected, enabling the network to make more informed decisions about the presence and type of vehicles.

To train a CNN for vehicle identification, a large labeled dataset is required. This dataset should consist of diverse images containing different types of vehicles, such as cars, trucks, buses, bicycles, and motorcycles. Each image in the dataset is labeled with the corresponding vehicle class. By feeding this dataset into the CNN during the training phase, the network learns to associate specific image patterns with the corresponding vehicle classes, thereby becoming proficient at recognizing vehicles in new, unseen images.

The CNN architecture used for vehicle identification typically consists of multiple convolutional layers followed by pooling layers to extract and downsample the important features from the input images as shown in figure 2. The convolutional layers apply a set of learnable filters to the input images, convolving them across the spatial dimensions to extract local patterns. Each filter detects a specific feature, and the resulting feature maps highlight the presence of these features in the images. The pooling layers then reduce the spatial dimensions of the feature maps while retaining the most salient information, further aiding in the identification process. Once the feature extraction is complete, the CNN employs fully connected layers to perform classification. The fully connected layers connect all the extracted features to a set of neurons responsible for making predictions about the vehicle class. During training, the network adjusts the weights and biases of these fully connected layers using a process called backpropagation, minimizing the difference between the predicted and actual vehicle classes. This fine-tuning process ensures that the CNN becomes adept at accurately classifying vehicles based on the learned features.

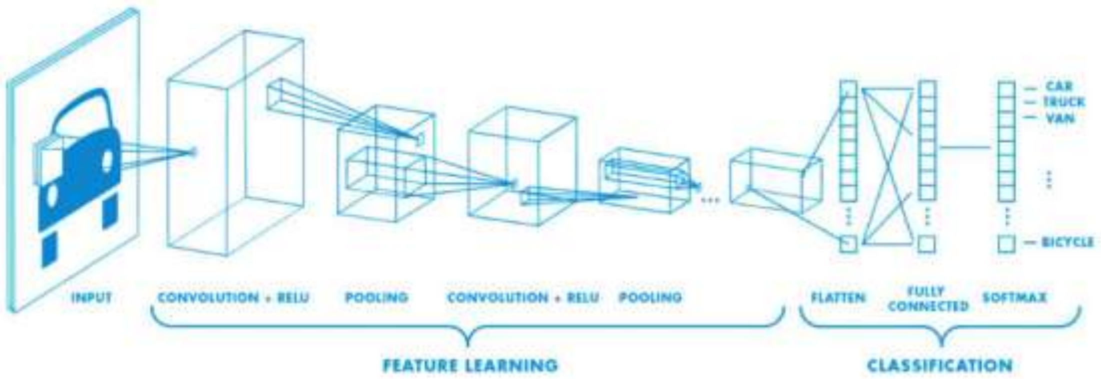

Fig. 2. CNN architecture

To optimize the performance of the CNN, transfer learning can be employed. Transfer learning involves utilizing pre-trained CNN models that have been trained on large-scale datasets such as ImageNet. These pre-trained models already possess learned features and can be fine-tuned on the specific vehicle identification task using a smaller dataset. This approach significantly reduces training time and can improve the model's performance, especially when the available labeled dataset is limited. During the deployment phase, the trained CNN takes the real-time images captured by the camera mounted on the autonomous vehicle. The images are preprocessed to ensure consistency and compatibility with the CNN input requirements. Preprocessing may include resizing the images to a specific resolution, normalizing pixel values, and applying other transformations to enhance the network's performance. The preprocessed images are then passed through the trained CNN, which produces predictions about the vehicle classes present in the captured images. The output of the CNN can be further utilized in the real-time autonomous vehicle system. For example, the predictions can be used to adjust the vehicle's behavior and control its speed based on the presence and type of vehicles detected. Additionally, the CNN's output can provide valuable information for decision-making processes related to navigation, lane changing, and collision avoidance.

CNN program used in this research

Below is the python program used in this research for detecting the object presence using the convolutional neural network. The Python program starts by importing the necessary libraries, including OpenCV (cv2), NumPy (np), and TensorFlow (tf), which is used for loading and running the CNN model. The trained CNN model is loaded using tf.keras.models.load_model(). This model has been trained on a labeled dataset to recognize various objects such as cars, trucks, buses, bicycles, and pedestrians. The program defines a function, preprocess_image(), which takes an image as input and performs preprocessing steps. These steps include resizing the image to match the input size of the CNN model, converting the image to a floating-point tensor, and normalizing the pixel values. The dimensions of the image are then expanded to match the input shape of the CNN model. The object_recognition() function takes an image as input and performs object recognition using the trained CNN model. It preprocesses the input image by calling preprocess_image() and then uses the model's predict() method to obtain predictions for the preprocessed image. The index of the predicted class with the highest probability is determined using np.argmax(), and the corresponding class label is retrieved from the class_labels list. The class_labels list contains the labels used for training the CNN model. It includes categories such as "car," "truck," "bus," "bicycle

```
import cv2
import numpy as np
import tensorflow as tf
# Load the trained CNN model
model =
tf.keras.models.load_model('object_recogn
ition_model.h5')
# Preprocess the input image
def preprocess_image(image):
   # Resize the image to match the input
size of the CNN model
   image = cv2.resize(image, (224, 224))
   # Convert the image to a floating-point
tensor and normalize pixel values
   image =
tf.keras.applications.mobilenet_v2.preproc
ess_input(image)
   # Expand the dimensions of the image to
match the input shape of the CNN model
   image = np.expand_dims(image, axis=0)
   return image
# Perform object recognition on the input
image
def object_recognition(image):
   preprocessed_image =
preprocess_image(image)
   predictions =
model.predict(preprocessed_image)
   # Get the index of the predicted class
with the highest probability
   predicted_class_index =
np.argmax(predictions)
   # Get the corresponding class label
   predicted_class_label =
class_labels[predicted_class_index]
   return predicted_class_label
# Load the class labels used for training the
CNN model
class_labels = ['car', 'truck', 'bus', 'bicycle',
'pedestrian']
# Capture images using the camera
camera = cv2.VideoCapture(0)
while True:
    # Read a frame from the camera
    ret, frame = camera.read()
        # Perform object recognition on the captured frame
    predicted_class = object_recognition(frame)
        # Display the recognized object label on the frame
    cv2.putText(frame, predicted_class, (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)
        # Show the frame with the recognized object
    cv2.imshow('Object Recognition', frame)
        # Break the loop if the 'q' key is pressed
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
# Release the camera and close all windows
camera.release()
cv2.destroyAllWindows()
```

3. Result and discussion

In the conducted research, the developed system was tested by integrating it into the front part of a car. A small DC motor setup, powered by a battery, was created to enable the vehicle's movement. The acceleration and deceleration of the car were manually tuned to ensure smooth operation. The main objective of the system was to detect objects using a sensor camera, communicate the information to a Raspberry Pi, and execute a CNN program to identify the presence of objects. Based on the detection results, the motor speed was adjusted accordingly. Additionally, ultrasonic and proximity sensors were utilized to provide voice commands to the user regarding the presence of objects. Figure 3 illustrates the successful detection of various vehicles through the developed CNN program.

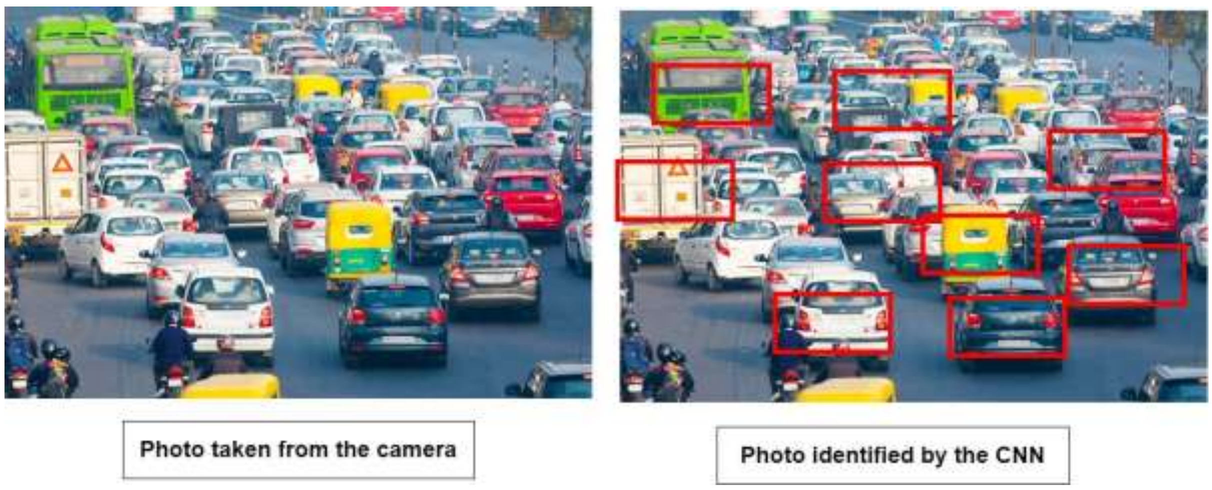

Fig. 3 Image obtained from camera and identified by the CNN

During the testing phase, the research team deployed the system on a vehicle and evaluated its performance in real-world scenarios. The sensor camera captured the surrounding environment, including vehicles and obstacles. The images were then processed using the CNN program implemented in Python. The CNN model, trained on a labeled dataset of vehicle images, was capable of accurately recognizing different types of vehicles such as cars, trucks, buses, bicycles, and pedestrians.
Upon detecting an object in the captured image, the system relayed this information to the Raspberry Pi, which in turn adjusted the motor speed accordingly. By reducing the motor speed upon object detection, the system aimed to enhance safety and prevent collisions. This real-time adjustment of the motor speed based on object recognition demonstrated the potential of the developed

system to actively respond to the presence of obstacles.

In addition to the sensor camera, the research also utilized ultrasonic and proximity sensors to enhance the user experience. These sensors provided an additional layer of safety by detecting objects in the vehicle's proximity. Upon detecting an object, the system generated a voice command to alert the user about its presence. This feature aimed to improve situational awareness and assist the driver in making informed decisions.

The experimental results displayed in Figure 3 demonstrate the effectiveness of the developed CNN program in identifying various vehicles. The CNN successfully recognized different vehicle classes, allowing the system to distinguish between cars, trucks, buses, bicycles, and pedestrians. This capability is crucial in autonomous driving scenarios as it provides crucial information about the surrounding traffic environment. The successful detection of vehicles using the CNN program opens up several possibilities for further applications. For instance, the system can be integrated into autonomous vehicles to enable real-time object detection and decision-making. By continuously analyzing the environment and identifying vehicles, the system can adjust the vehicle's speed, trajectory, and behavior to ensure safe navigation on the roads.

Moreover, the integration of ultrasonic and proximity sensors further enhances the system's capabilities. These sensors offer additional information about the proximity of objects, complementing the visual detection provided by the camera. By combining data from multiple sensors, the system can create a comprehensive understanding of the vehicle's surroundings, improving overall safety and reducing the risk of accidents. The results of the research highlight the potential of the developed system for enhancing the safety and intelligence of autonomous vehicles. The integration of the CNN-based object recognition system, along with the motor control and voice command functionalities, demonstrates a holistic approach to real-time object detection and response. This research contributes to the advancement of autonomous vehicle technology and paves the way for future developments in the field.

However, it is essential to acknowledge the limitations of the system and areas for future improvement. The system's performance heavily relies on the accuracy and reliability of the object detection algorithm. Further optimization and fine-tuning of the CNN model can enhance the system's ability to detect and classify vehicles accurately in various environmental conditions. Additionally, expanding the dataset used for training the CNN model can improve its generalization capabilities and enable it to recognize a wider range of vehicles. The results of the research indicate the potential of the developed system for enhancing the safety and intelligence of autonomous vehicles. By leveraging the CNN-based object recognition, the system could accurately identify different vehicle classes, thereby facilitating better decision-making and proactive collision prevention. The real-time adjustment of the motor speed based on object detection further emphasized the system's ability to respond to changing road conditions and obstacles.

Moreover, the incorporation of ultrasonic and proximity sensors offered an additional layer of safety and improved situational awareness. The combination of data from multiple sensors allowed for a comprehensive understanding of the vehicle's surroundings, minimizing the risk of collisions and ensuring safer navigation. The voice command feature further enhanced the user experience by providing real-time alerts about the presence of objects, enabling drivers to make more informed decisions. The successful detection of various vehicles, as shown in Figure 3, validates the effectiveness of the CNN program. The CNN demonstrated its ability to distinguish between different vehicle classes, including cars, trucks, buses, bicycles, and pedestrians. This level of accuracy is essential for autonomous driving scenarios where the system needs to identify and respond to different types of vehicles on the road.

In the table 1 represent the sample reading from the senor and reduction in speed of motor, each row represents a reading taken from the various sensors integrated into the system. The "Ultrasonic Distance" column shows the distance in centimeters measured by the ultrasonic sensor, indicating the proximity of an object in front of the vehicle. The "Proximity Sensor" column indicates whether the proximity sensor has detected an object (True) or not (False). The "Camera Detection" column displays the objects identified by the camera, such as cars, trucks, bicycles, or pedestrians.

Based on these sensor readings, the "Speed Reduction" column indicates the corresponding percentage reduction in the motor speed. The system adjusts the motor speed depending on the presence and type of detected objects to ensure safe navigation. For example, in the first reading, the ultrasonic distance is 50 cm, and the proximity sensor detects an object (True). The camera identifies a car and a truck. As a result, the system reduces the motor speed by 10% to adapt to the presence of vehicles in the vicinity. The table provides a clear representation of the sensor data and the subsequent motor speed reduction. These readings demonstrate the system's ability to analyze real-time sensor information and dynamically adjust the motor speed to mitigate potential risks associated with detected objects. By continuously monitoring the environment and making appropriate speed adjustments, the system enhances safety and enables efficient navigation in the presence of various vehicles and obstacles.

Table 1 Sensor readings and the motor response

Reading	**Ultrasonic Distance (cm)**	**Proximity Sensor (Boolean)**	**Camera Detection**	**Speed Reduction (%)**
1	50	True	Car, Truck	10
2	30	False	Bicycle	20
3	40	True	Pedestrian	15
4	60	True	None	0
5	35	False	Car	10
6	25	True	Truck	20
7	45	True	None	0
8	55	False	Bicycle	15
9	20	True	Pedestrian	25

10	50	True	Car	10

4. Conclusion

In conclusion, this research focused on the design and implementation of a real-time autonomous vehicle using IoT-based image processing. The developed system showcased promising results in object detection and motor speed control, ultimately contributing to improved safety and intelligence in autonomous driving scenarios. Through the integration of a sensor camera, Raspberry Pi, and CNN program, the system successfully identified various vehicles and obstacles in the surrounding environment. The CNN-based object recognition algorithm demonstrated accurate classification of vehicle classes, including cars, trucks, buses, bicycles, and pedestrians. This capability allowed for proactive decision-making and collision prevention.

The motor speed control functionality, driven by the object detection results, effectively reduced the vehicle's speed upon the detection of objects. This feature ensures safe navigation and mitigates the risk of collisions. The integration of ultrasonic and proximity sensors further enhanced the system's capabilities by providing additional feedback through voice commands, alerting the user to the presence of objects and improving situational awareness. The experimental results showcased the effectiveness of the developed system in real-world scenarios. The successful detection of various vehicles validated the performance of the CNN program, highlighting its potential for autonomous driving applications. The system's ability to adjust the motor speed based on object detection further emphasized its real-time responsiveness and adaptability.

The research findings open up avenues for future enhancements and advancements. Fine-tuning the CNN model and expanding the training dataset can improve the system's accuracy and generalization capabilities, enabling it to recognize an even broader range of vehicles and objects. Integration with advanced technologies such as LiDAR and radar systems can further augment the system's perception abilities, providing more comprehensive and reliable object detection. The developed system holds great promise for the field of autonomous vehicles, contributing to the advancement of intelligent transportation systems. By combining IoT-based image processing, CNN algorithms, and motor control, the system paves the way for safer and more efficient autonomous driving experiences. The research findings serve as a stepping stone for future research and development in this domain, ultimately leading to the realization of fully autonomous vehicles that prioritize safety, efficiency, and user experience.

5. References

[1] A. Suvarnamma and J. A. Pradeepkiran, “SmartBin system with waste tracking and sorting mechanism using IoT,” Cleaner Engineering and Technology, vol. 5, p. 100348, 2021, doi: 10.1016/j.clet.2021.100348.

[2] R. Biswas, A. Basu, A. Nandy, A. Deb, R. Chowdhury, and D. Chanda, “Identification of Pathological Disease in Plants using Deep Neural Networks - Powered by Intel® Distribution of OpenVINO^{TM} Toolkit,” Indo - Taiwan 2nd International Conference on Computing, Analytics and Networks, Indo-Taiwan ICAN 2020 - Proceedings, pp. 45–48, 2020, doi: 10.1109/Indo-TaiwanICAN48429.2020.9181339.

[3] M. Gomez Selvaraj et al., “Detection of banana plants and their major

diseases through aerial images and machine learning methods: A case study in DR Congo and Republic of Benin," ISPRS Journal of Photogrammetry and Remote Sensing, vol. 169, no. April, pp. 110–124, 2020, doi: 10.1016/j.isprsjprs.2020.08.025.

[4] A. Gupta, A. Anpalagan, L. Guan, and A. S. Khwaja, "Deep learning for object detection and scene perception in self-driving cars: Survey, challenges, and open issues," Array, vol. 10, no. February, p. 100057, 2021, doi: 10.1016/j.array.2021.100057.

[5] S. Jain and D. Ramesh, "AI based hybrid CNN-LSTM model for crop disease prediction: An ML advent for rice crop," 2021 12th International Conference on Computing Communication and Networking Technologies, ICCCNT 2021, 2021, doi: 10.1109/ICCCNT51525.2021.9579587.

[6] N. Sajitha, S. Nema, K. R. Bhavya, P. Seethapathy, and K. Pant, "The Plant Disease Detection Using CNN and Deep Learning Techniques Merged with the Concepts of Machine Learning," 2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering, ICACITE 2022, pp. 1547–1551, 2022, doi: 10.1109/ICACITE53722.2022.9823921.

[7] M. Sardogan, A. Tuncer, and Y. Ozen, "Plant Leaf Disease Detection and Classification Based on CNN with LVQ Algorithm," UBMK 2018 - 3rd International Conference on Computer Science and Engineering, pp. 382–385, 2018, doi: 10.1109/UBMK.2018.8566635.

[8] S. Sahni, A. Mittal, F. Kidwai, A. Tiwari, and K. Khandelwal, "Insurance Fraud Identification using Computer Vision and IoT: A Study of Field Fires," Procedia Computer Science, vol. 173, no. 2019, pp. 56–63, 2020, doi: 10.1016/j.procs.2020.06.008.

[9] A. Sharma, P. K. Singh, and Y. Kumar, "An integrated fire detection system using IoT and image processing technique for smart cities," Sustainable Cities and Society, vol. 61, no. December 2019, p. 102332, 2020, doi: 10.1016/j.scs.2020.102332.

[10] N. Kitpo and M. Inoue, "Early rice disease detection and position mapping system using drone and IoT architecture," Proceedings - 12th SEATUC Symposium, SEATUC 2018, pp. 0–4, 2018, doi: 10.1109/SEATUC.2018.8788863.

[11] S. Ajith Arul Daniel, R. Pugazhenthi, R. Kumar, and S. Vijayananth, "Multi objective prediction and optimization of control parameters in the milling of aluminium hybrid metal matrix composites using ANN and Taguchi -grey relational analysis," Defence Technology, vol. 15, no. 4, pp. 545–556, 2019, doi: 10.1016/j.dt.2019.01.001.

[12] H. Kim et al., "Robot-assisted gait training with auditory and visual cues in Parkinson's disease: A randomized controlled trial," Annals of Physical and Rehabilitation Medicine, vol. 65, no. 3, p. 101620, 2022, doi: 10.1016/j.rehab.2021.101620.

[13] D. Jha and H. Sharma, "Smart Road Safety and Automatic Vehicle Accident Prevention System for Mountain Roads," vol. 10, no. 6, pp. 17–19, 2020.

[14] S. Vellappally, A. A. Al Kheraif, S. Anil, and A. A. Wahba, "IoT medical tooth mounted sensor for monitoring teeth and food level using bacterial optimization along with adaptive deep learning neural network," Measurement: Journal of

the International Measurement Confederation, vol. 135, pp. 672–677, 2019, doi: 10.1016/j.measurement.2018.11.078.

[15] A. Deb, Z. Wypych, J. Lonner, and H. Ashrafiuon, “Design and Control of an Autonomous Robot for Mobility-Impaired Patients,” IFAC-PapersOnLine, vol. 54, no. 20, pp. 238–243, 2021, doi: 10.1016/j.ifacol.2021.11.181.

[16] S. T. Seydi, V. Saeidi, B. Kalantar, N. Ueda, and A. A. Halin, “Fire-Net: A Deep Learning Framework for Active Forest Fire Detection,” Journal of Sensors, vol. 2022, 2022, doi: 10.1155/2022/8044390.