

# Traffic Sign Recognition Using Artificial Intelligence Technique

Urvish Bhavsar<sup>1</sup>, Prof and Head (Dr) Manish Thakker<sup>2</sup>, Assistant Prof (Dr) Manisha Patel<sup>3</sup>

<sup>1</sup>Urvish Bhavsar, Department of Applied instrumentation(Instrumentation and control), LD college of engineering, Gujarat, Ahmedabad, India

<sup>2</sup>Professor and Head (Dr) Manish Thakker, Department of Applied instrumentation(Instrumentation and control), LD college of engineering, Gujarat, Ahmedabad, India

<sup>3</sup>Assistant Prof (Dr) Manisha Patel, Department of Applied instrumentation(Instrumentation and control), LD college of engineering, Gujarat, Ahmedabad, India

\*\*\*

**Abstract** - Traffic signs are one of the most primary components in any transportation system. As modern transportation gradually shifts towards automated and semi-automated driving systems, the need for accurate and robust traffic sign recognition is increasing. Traditional techniques struggle with real-world complexities, such as shadows and harsh environmental conditions. To address these challenges, the present research combined a classical edge-detection method with a Polar coordinate-based Convolutional Neural Network(PC-CNN). Edge detection stage reduces noise and redundant background details, ensuring that the features remain consistent even under variable lighting or environmental disturbances. The polar-CNN is better suited to recognize signs from multiple orientations. The combination of both methods improved accuracy, and precision as compared to the conventional CNN-based models. This hybrid technique provides a solution for traffic sign recognition.

**Key Words:** Image Processing, Pre-processing, Edge detection, Rotation-invariant, Polar Convolutional Neural Network, Polar coordinates, Traffic Sign Recognition, Intelligent Transportation Systems

## 1. INTRODUCTION

Traffic-sign recognition is a technology by which a vehicle is able to recognize the traffic signs put on the road. e.g. "Speed limit" or "turn ahead" or "left ahead". Traffic signs provide valuable information to drivers and other road users. By keeping drivers informed of traffic signs. TSR helps prevent accidents caused by missed or ignored signs, such as speed limits and stop signs. There are three main types of road signs : regulatory signs , which dictate rules and commands (Speed limits) . warning signs , which alert drivers to potential hazards (Curve ahead). inforatory signs , which provide directional or facility informations (Hospital, Town). CNNs lack the ability to learn fully rotation-invariant features because. Rotation-invariant feature learning is critically important in real-world applications. Therefore I propose a new approach, using polar coordinate transformation to convert rotation variations into translation variations, which standard CNNs handle naturally. This provides the theoretical motivation for designing the PC-CNN model.

To develop a hybrid traffic sign prediction (e.g, Canny , Sobel, Prewitt) with Polar CNN for enhanced feature extraction. To increase the recognition accuracy and resistance towards illumination, orientation , and partial occlusion of traffic signs. The project aims at traffic sign prediction via computer vision and AIML based deep learning approaches , with preprocessing via traditional edge detection methods. Benchmark datasets like GTSRB will be utilized mainly for training and validation in experiments. Can be expanded with embedded systems or automative hardware or as a traffic optimization.

Road sign prediction is critical for autonomous vehicles and intelligent transportation systems. Here I integrates edge detection method and Polar CNN method to achieve enhanced recognition accuracy and robustness. Edge detection emphasizes boundaries of signs , and the Polar CNN method achieves rotation and scale invariance appropriate for real-world scenarios.

## 1.1 Literature Review

This section describes the present system's literature review, as well as the existing system's problem statements. Harikesh Kumar Sharma and Anupama Jamwa[1] suggest a reliable and accurate traffic sign recognition system that can help enhance road safety and reduce accidents caused by driver error. To achieve detection accuracy so that the system is viable for practical use. Their Classification accuracy is nearly 94.5%. The model can run in various lighting conditions, including shadows and daylight brightness. Ugur Yuzgec, and Irfan OKten[2] demonstrated performance degradation under adverse conditions, but still acceptable. Comparison showing adding augmentation weakens improved performance. Future improvements include addressing class imbalance, expanding data augmentation, testing in real-world driving conditions, and exploring advanced architectures. Bharath Kumar, and Anupama Rani[3] try to improve over existing methods by leveraging deep learning and reduce reliance on human observations. Their research identifies detection performance can degrade with increased distance, lower quality , or environmental noise. The use of CNNs further enhances prediction, precision, and reduces false

detection. Compared to existing systems like K-means clustering, the proposed model shows superior performance in terms of accuracy, computation speed, and cost-effectiveness. Bhogadi Sreeja, and Sruthila Bokka [4] evaluate and compare various available methods for traffic sign detection under similar conditions to see which perform better. They Measure accuracy, possibly detection rate, precision, recall etc . for different techniques. The results show that deep learning techniques significantly better than the traditional deep learning techniques. The CNN model achieves an accuracy of 95%, establishing a strong foundational performance. Ruoqiao Jiang, and Shaohui Mei[5] uses Rotation-Invariant Feature Learning, which allow feature extraction to be invariant to image rotations, without needing overly complex architectures or heavy augmentation. This suggest better performance. To outperform or be competitive with other rotation-invariant CNN designs, but without introducing lots of extra parameters or complexity. The research outcomes show that the proposed PC-CNN significantly improves rotation-invariant feature learning compared to traditional CNNs models. PC-CNN again achieves superior performance with an accuracy of 97.80%, highlighting its ability to maintain in-variance without relying on multiple branches. Senthilnayaki B, and Rajeswary C[6] successfully resulted in the development of complete end-to-end traffic sign detection. The developed model shows strong potential for deployment in Advanced Driver Assistance system. Further, it reduced traffic accidents rats, more stable and efficient traffic management system. Dr. Vijaykumar S. Bidve, ,and Sneha Wagh[7] provides a strong foundation for future improvements, such as hardware implementation, voice alerts, and adaptive learning for new traffic sign patterns. Ahmed J. Abougarair, and Mohammed Elmaryul[8] finding also open up future scope and research direction for the hardware implementation, edge-device optimization for newly introduced traffic signs. G jithin, Yanamala Umesh[9] proposed system detects the traffic signal and recognizes using machine learning algorithms. The proposed system is also scalable for detecting and recognizing the traffic sign by image processing. The system is not having complex process to detect and recognize that the data like the existing system. Proposed system gives genuine and fast result than existing system.

## 2 . WORKING OF PROPOSED MODEL

### 2.1 Datasets

German Traffic Sign Recognition Benchmark(GTSRB) dataset used, which is widely recognized as a dataset for traffic signs-related research. Here it provides 51,893 real-world images, which are further categorized into 43 different traffic-sign classes. Figure 1 shows some sample dataset images.



Fig - 1 : Some sample images from the GTSRB dataset

## 2.2 HARDWARE AND SOFTWARE REQUIREMENT

### 2.2.1 HARDWARE REQUIREMENTS

**Device :** Laptop (LENOVO Ideapad 320) **Input Devices :** Keyboard, Mouse **RAM :** 2 GB , **Storage :** At least 10 GB free disk space is recommended for installing Python, VS Code, and storing datasets, libraries, and other project files. **Arduino Uno :** The Arduino Uno is an open-source microcontroller board. It handles and controls input and output, managing communication protocols. **RGB color module:** It is an electronic component that combines red, green, and blue LEDs. It typically features 4 pins (R,G,B, and common), allowing Arduino to control light, Defines colors by specifying red, green, and blue values from 0-255.

### 2.2.2 SOFTWARE REQUIREMENTS

**Operating system :** Windows 10 : compatible with Python, drivers, various libraries and package used **VS Code :** VS Code provides several features, including: Syntax highlighting ,Git integration ,User friendly interface , Debugging tools **Arduino IDE :** For programming in Microcontroller like Arduino Uno. **Coding Language :** Python

### 2.2.3 LIBRARIES REQUIREMENTS

**NumPy :** NumPy stands for Numerical Python, which is an open-source Python library designed for efficient numerical calculation and data processing. **OpenCV (Open Source Computer Vision) :** widely used open-source library developed for real-time computer vision and image processing applications. **Keras :** Keras is a high-level deep learning API in Python and designed to enable fast development with neural networks. It is the top of the low-level deep learning frameworks such as TensorFlow and provides a user friend interface for building, training, and evaluating deep learning models. **Matplotlib :** Matplotlib is widely used Python visualization library designed for the interactive and animation type of plots. Matplotlib is used in deep learning to analyze visual metrics such as accuracy, loss, precision, recall, and confusion metrics. **Pygame :** Pygame is an open-source

Python library designed for developing multimedia applications. It provides handling of graphics, sound, and timing control. With minimal complexity, developers can create real-time interactive applications. **pyttsx3** : pyttsx3 is an open-source text-to-speech library in Python that enables applications to convert textual information into natural-sounding speech. It works offline, making it suitable for real-time embedded intelligent systems where an internet connection is limited or unavailable. **Scikit-Learn** : it is used for fast model building, training and evaluation..

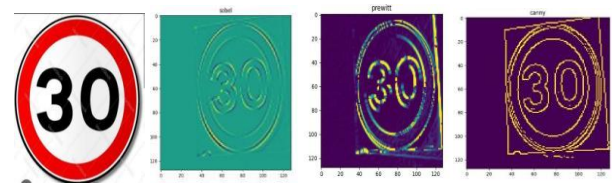


Fig - 2 : Original image , Sobel , Prewitt , Canny edge detection result

### 2.3 PREPROCESSING

**1 Image Resizing and Spatial Normalization** To ensure the model's optimum performance of neural network models, initially all raw images were converted into NumPy arrays. Then images were re-sized to a resolution, typically 32x32 or 64x64 pixels, depending on the model architecture. **Iresized = R(Ioriginal, H, W)** , where H and W denote the target height and width.

**2 Color Space Conversion** Originally, traffic signs are stored in RGB format. So, this type of color information is less critical compared to the shape and edge when polar transformation is using. Hence, images are converted to grayscale: **Igray = 0.299R + 0.587G + 0.114B**

**3 Pixel Intensity Normalization** To overcome this, pixel intensities are normalized as mentioned : **Inorm = Igray/255**. This results in values in the range of [0,1] or [-1,1].

**4 Polar Coordinate Transformation** After normalization is completed, the grayscale images are converted from Cartesian coordinates to polar coordinates. **Polar Coordinate Transformation Algorithm** Find height and width :  $h, w = \text{img.shape}$  Find centre :  $\text{centre} = (w//2, h//2)$  Find radius :  $\text{radius} = \text{np.sqrt}((w/2)**2 + (h/2)**2)$  Convert into Polar transformation :  $\text{polar\_img} = \text{cv2.wrapPolar}(\text{gray}, (360, \text{radius}), \text{centre}, \text{radius}, \text{cv2.WRAP\_FILL\_OUTLIERS})$

### 5 Edge detection comparison

Some traditional edge-detection techniques such as Canny, Sobel, and Prewitt algorithms used and applied some basic edge detection algorithms for an image to find out which one is the most suitable.

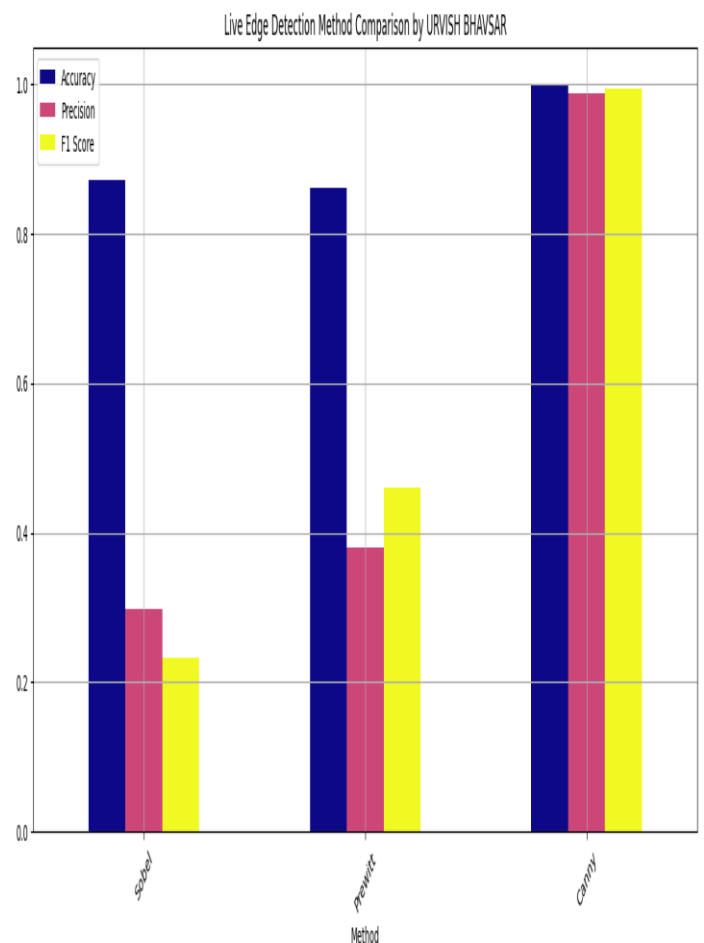


Fig - 3 : Result of three edge detection method

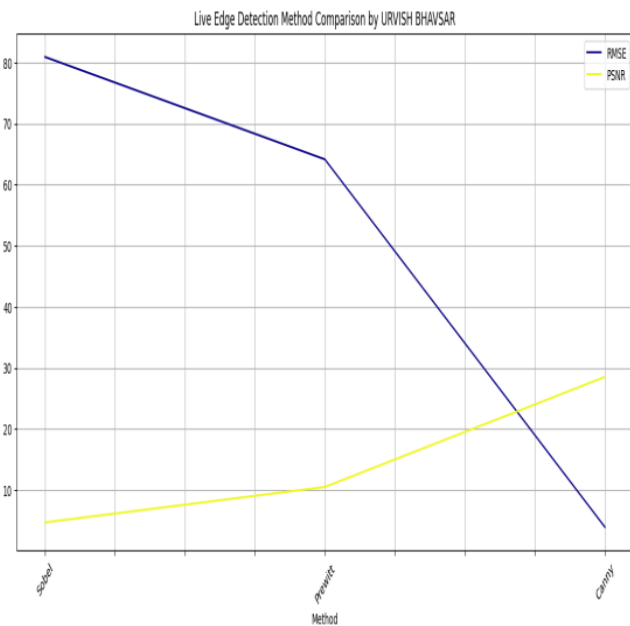


Fig - 4 : Comparison of Edge detection Method in terms of RMSE and PSNR

```
PS C:\Users\URVISH> python .\Edge_detection_comp.py
Capturing image... Press SPACE to capture.
Method Accuracy Precision F1 Score
0 Sobel 0.870728 0.298507 0.232052
1 Prewitt 0.859558 0.380397 0.459479
2 Canny 0.998596 0.986542 0.993225
Method RMSE PSNR
0 Sobel 80.903176 4.711964
1 Prewitt 64.133855 10.511633
2 Canny 3.946833 28.526921
```

Fig - 5 : Edge detection comparison score

Canny edge detection significantly improves traffic sign recognition by reducing noise, improving high-precision, and enabling robust, real-time identification in complex, low-contrast, or poor-lighting conditions. It simplifies images for faster classification.

## 2. POLAR COORDINATE CONVOLUTIONAL NEURAL NETWORK

Figure 9 shows the architecture of proposed system. Input layer : Takes the standard Cartesian image. The input layer of the Polar CNN receives traffic images, which are resized and preprocessed. Similar to a basic CNN, normalization is applied. The input layer plays a crucial role in providing consistent images for transformation into polar space.

Polar Coordinate Transformation layer: Maps the image pixels to a polar grid. This layer converts the Cartesian(x, y) coordinate pixel grid onto polar coordinates by radius (r) and angle (θ). Through this, circular, triangular or octagonal sign shapes become linearized patterns, making them easier for the CNN to detect.

Convolutional and Polling layers : Extracts features from the polar grid.

Flatten Layer : Converts the final feature map into a one-dimensional vector.

Fully Connected Layers : Classifies the extracted features and passed from Rectified Liner Unit activation function.

Output Layer : Provides the final prediction.

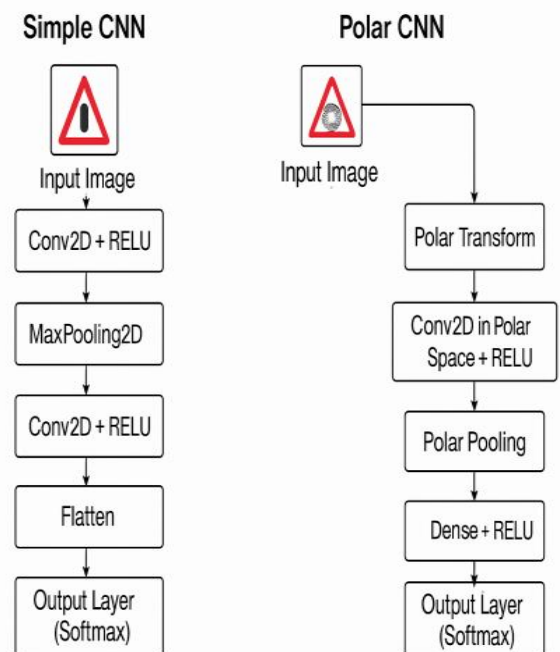


Fig - 6 : System Architecture

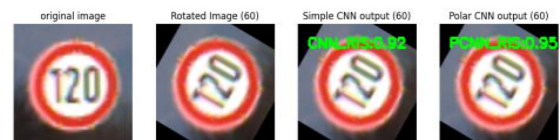


Fig - 7 : Practically comparison of RIS score for different angles

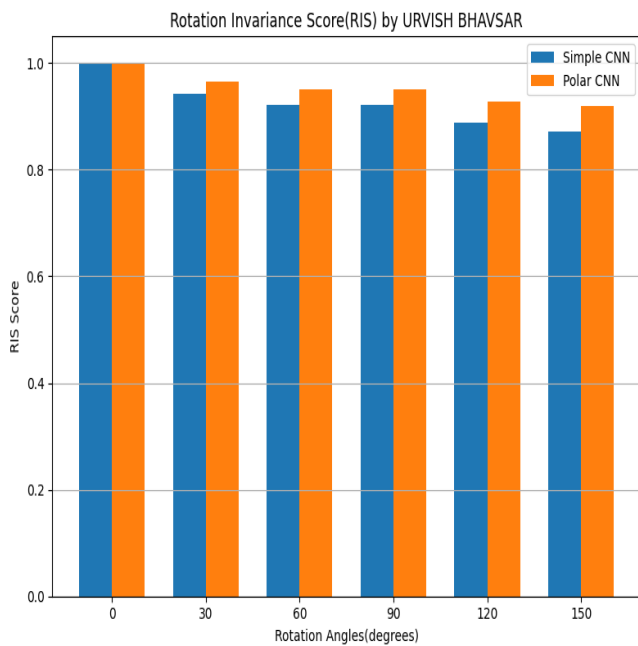


Fig -8 : RIS score for different angles

The complete preprocessing pipeline contributed significantly to the reduction of noise, improvement of the reduction of noise, and enhancement of the feature, which strengthens the performance of the traffic sign recognition system. It clearly shows polar CNN better perform when image is rotated.

### 3.1 MODEL TRAINING

**1 Data Loading and Preprocessing :** Images are loaded, resized and normalized. For PolarCNN, images are converted to polar coordinates before being fed to the network.

**2 Data Augmentation :** In this step, augmentation techniques such as rotation, scaling, translation, zooming, and brightness adjustments are applied which reduce overfitting. It also helpful for the improving generalization.

**3 Initialize CNN weights Randomly :** weights

**4 Forward Propagation :** Input images pass through different layers such as convolution, pooling, and fully connected layers to generate predicted class probabilities using the Softmax function.

**5 Loss Computation :** The predicted output is compared with the labels using the cross-entropy loss function for prediction error.

**6 Backpropagation :** It helps to update weights to minimize error.

**7 Weight Update :** The Adam optimizer updates the weights using adaptive learning rates, which helps to minimize loss efficiently.

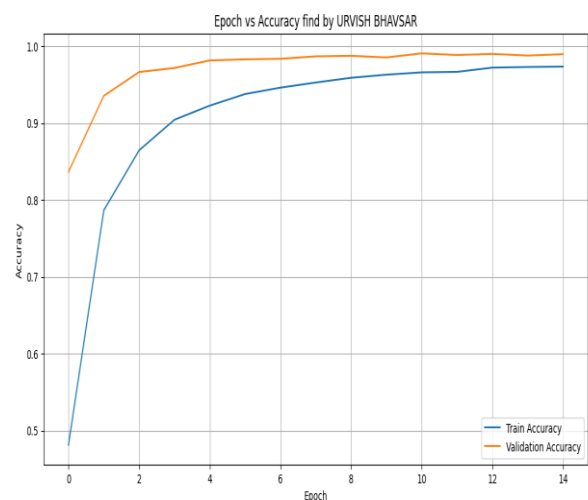
**8 Validation Evaluation :** This process is repeated from Step 2 to Steps 5 iteratively over all the epochs, which measure validation accuracy and loss.

**9 Model Check pointing :** The best performing model is saved for deployment as follow : model.save("model.h5"), which is further used with real-time applications.

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
801/801  ██████████ 86s 87ms/step - accuracy: 0.9666
Epoch 13/15
801/801  ██████████ 94s 102ms/step - accuracy: 0.9728
Epoch 14/15
801/801  ██████████ 71s 88ms/step - accuracy: 0.9729
Epoch 15/15
801/801  ██████████ 72s 90ms/step - accuracy: 0.9729
223/223  ██████████ 125s 533ms/step

PolarCNN Traffic Sign Detection Results by URVISH BHAVSAR:
Accuracy: 98.7079%
F1 Score: 0.9871
MSE: 1.6114
PSNR: -2.07 dB
    
```



### 3.5 MODEL TESTING RESULTS

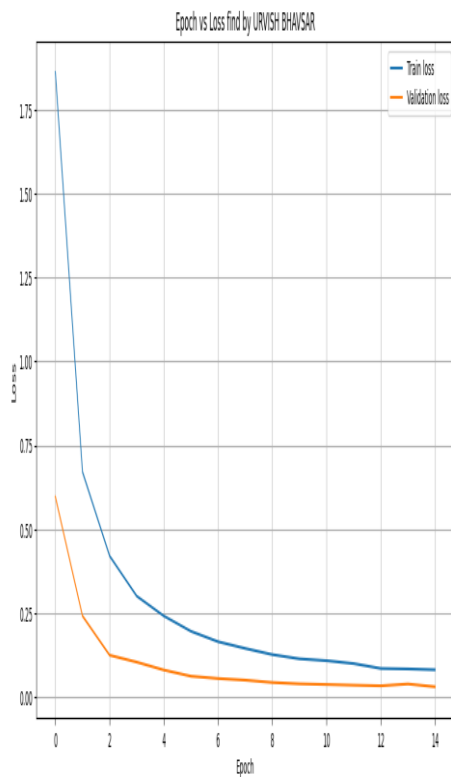


Fig - 9 : Trainig result

- 3.2 Model Testing**
- 1 Load the Trained Model :** First load the trained polar-CNN model(.h5 file), this was saved after training. `Model = keras.model("Polar_cnn.h5")`.
- 2 Load the Test Dataset :** The next step is loading test images from dataset, which you have to test.
- 3 Preprocess Test Images :** Apply preprocessing steps such as Resize images( 32 x 32 or 64 x 64) , Normalize pixel values(0-1) and Convert to array.
- 4 Run Model Prediction :** `Prediction = model.predict(img)` , `Class_id = prediction.argmax()` This model outputs the predicted traffic sign class
- 5 Calculate performance metrics :** After testing many images, performance metrics such as accuracy, precision, recall, and F1-score are calculated to evaluate the model performance.
- 6 Visualize Results :** Display Bounding box detection results and prediction labels with light and sound indication.

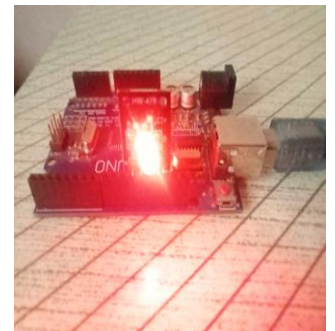


Fig - 10 : Detection sign : Speed limit(30km/h) + bounding box + Red light indication + Sound



Fig - 11 : Detection sign : Children crossing + bounding box + Blue light indication + sound



Fig - 12 : Detection sign : Keep right + bounding box + Green light indication + sound

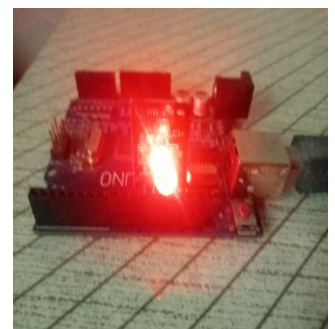


Fig - 13 : Detection sign : No passing for vehicles over 3.5 metric tons + bounding box + Red light indication + sound



Fig - 14 : Detection sign : Bumpy road + bounding box + Blue light indication + sound

### 3.3 COMPARISON AND RESULTS

Table -1: Comparison and result

Feature	Simple CNN	Polar CNN
Input Format	Cartesian(x,y)	Polar(r, $\theta$ )(radius, angle)
Rotation Invariance	No	Yes(inherent)
Implementation	Easier	Slightly complex
Robustness to Rotation	Low	High
Model Performance in terms of Classification Accuracy	nearly 94.5%	nearly 98.7 %
Use case Example	Digit Recognition	Traffic Sign detection

### 4 CONCLUSIONS AND FUTURE SCOPE

Developed an AI-based Traffic Sign Recognition system using Polar CNN for accurate detection of traffic signs. The system displays bounding box around the sign, sign name as text, and voice alerts(audio feedback) to inform the user about the detected traffic sign. The Polar CNN model improves recognition performance by effectively capturing rotational features of traffic signs. To enhance the practical usability of the system, hardware integration was implemented using an Arduino and RGB LED module. Where different colors indicates different conditions. Red LED : indicates prohibitory signs , Blue LED : indicates warning signs , Green LED : indicates mandatory signs. This system demonstrates how AI combined with hardware can improve driver awareness and road safety.

The existing system can be improves using larger and more diverse traffic sign datasets.

Additional sensors such as cameras, and IOT modules can be integrated for smart traffic monitoring systems integration with autonomous vehicle and Advanced Driver Assistance System (ADAS) can enhance road safety.

### REFERENCES

- [1] Harikesh Kumar Sharma Anupama Jamwa, "Traffic Sign Recognition System using CNN" International Journal of Innovative Science and Research Technology (IJSRT),11 November 2023.
- [2] Ugur Yuzgec Irfan OKten, "Traffic Signs Recognition using Deep learning Model", ResearchGate, May 2025.
- [3] Bharath Kumar Anupama Rani, "Traffic Sign Detection using Convolution Neural Network", International Journal of Creative research thought (IJCRT), 5 May 2020.
- [4] Ruoqiao Jiang, Shaohui Mei, "Polar Coordinate Convolution Neural Network : From Rotation Invariance to Translation Invariance", IEEE 2019.
- [5] Bhogadi Sreeja Sruthila Bokka Giddi Shravya Katari Sri Vidya Vardini , "Traffic Sign Detection using Transfer learning and a Comparison Between Different Techniques", IEEE 2022.
- [6] Senthilnayaki B, Rajeswary C, Nivetha G, Dharanyadevi P, Mahalakshmi G, A.Devi "Traffic Sign Prediction and Classification Using Image Processing Techniques"2022 International Conference on Smart Technologies and Systems for Next Generation Computing (ICSTSN).
- [7] Dr. Vijaykumar S. Bidve, Anula Bhole, Mrunalini Temgire, Sneha Wagh, Bhakti Toraskar, " Traffic sign detection using CNN", IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 7 Issue 7, July 2020.
- [8] Ahmed J. Abougarair, Mohammed Elmaryul, Mohamed KI Aburakhis , "Real time traffic Sign detection and recognition for autonomous vehicle", International Robotics & Automation Journal , Volume 8 Issue 3 - 2022
- [9] G jithin, Yanamala Umesh, "Traffic sign detection and recognition using deep learning", Institute of Science and technology, May-2022 .
- [10] Kaggle GTSRB - German Traffic Sign[Data set]. Kaggle. Retrieved from <https://www.kaggle.com/gtsrb-german-traffic-sign>.