Nigerian Research Journal of Engineering and Environmental Sciences 10(2) 2025 pp. 594-607
p ISSN: 2635-3342; e ISSN: 2635-3350

Nigerian Research Journal of Engineering and Environmental Sciences
Journal homepage: www.rjees.com

Original Research Article

Development of an Intelligent Road Condition Monitoring System using Citizen Sensing Technique

***[1]Musa, A., [2]Bala, J.A., [2]Folorunso, T.A., [3]Abdulrahman, H.S. and [2]Oloyede, M.**

[1]Department of Surveying and Geoinformatics, Federal University of Technology, Minna, Nigeria.
[2]Department of Mechatronics Engineering, Federal University of Technology Minna, Minna, Nigeria.
[3]Department of Civil Engineering, Federal University of Technology Minna, Minna, Nigeria.
*ahmed.musa@futminna.edu.ng; amusa22@gmail.com

http://doi.org/10.5281/zenodo.18062038

ARTICLE INFORMATION

Article history:
Received 24 Oct. 2025
Revised 28 Nov. 2025
Accepted 11 Dec. 2025
Available online 30 Dec. 2025

Keywords:
Citizen sensing
Deep learning
Pothole detection
Road condition monitoring
YOLOv4

ABSTRACT

Road transportation is Nigeria's most important mode of transportation, due to increase in vehicle ownership and the role of roads in economic activities. Road anomalies, which include potholes and speed bumps, impede the movement of traffic on roads. Therefore, there is a need for an intelligent system to detect these road anomalies and document them. Besides providing real-time assistance to road users and future driverless vehicles, an intelligent road information system can be used to build databases of road conditions across the country and to create road remediation strategies and schemes for costing road repairs. This paper presents the development of an intelligent, end-to-end road condition monitoring system using a citizen sensing approach. The approach was employed because it allows road users to gather road data with dedicated devices rather than installing sensors along the roadway. The system detects potholes using a deep learning-based object detection model (Tiny YOLOv4). The dataset used in validating the approach consisted of 1,265 images for training, 401 images for testing and 118 images for validation. The model had an overall precision of 77.00%, recall of 66.00%, F1-score of 71.00%, average IoU of 58.65% and mean average precision (mAP@0.50) of 69.99%. Based on the obtained results, the system demonstrated the ability to detect potholes on the road surface. Furthermore, this system establishes a critical first step towards a large-scale, cost-effective remote road monitoring infrastructure. Additionally, the use of citizen data collected by road users significantly reduces the cost of deploying the technique.

1. INTRODUCTION

Road transportation is one of the major and most important modes of transportation (Rodrigue, 2020). Road transportation has become popular due to the increased number of vehicle owners (Ukonze et al, 2020). Farmers and traders mainly transport their goods and farm produce to the market by road (Yusuf,

2020). However, over the years, road transportation has been plagued with a series of road accidents, especially in the developing world.

The World Health Organization (WHO) reported in 2022 that every year around 1.3 million individuals die due to road accidents, while 20-50 million people experience injuries and traumas that could potentially result in disabilities or serious health issues. Additionally, road accidents also have adverse socio-economic consequences for nations. According to the WHO's 2022 data, road accidents have inflicted a financial toll of approximately 3% of the Gross Domestic Product (GDP) in numerous countries. The United Nations' Sustainable Development Goal 3, Target 3.6 aims to reduce the global number of deaths caused by road accidents by 50% by the year 2030. Deme (2019) highlights that in Africa, road accidents are predominantly caused by factors such as excessive speeding, driving under the influence of alcohol, and poor road conditions.

In attempts to prevent road crashes, various techniques for road accident prevention have been introduced. One of the techniques is the alcohol detection system that will prevent automobiles from starting if it detects the presence of ethanol in the air (Ahmad et al, 2019), which might mean that the driver is drunk. Other techniques include an over-speeding detection system (Kotagi et al, 2020), and improved traffic sign detection algorithms (Cao et al, 2019). Bad road conditions have also been a major cause of road accidents (Bello-Salau et al, 2019).

Bad road conditions refer to the existence of road irregularities like potholes, speed bumps, rutting, and cracks on the road surface (Bello-Salau et al, 2019). The problem is that most government agencies in charge of the maintenance of roads and also road users do not know where these anomalies are, and presently these anomalies are regularly found and reported manually by qualified inspectors which can be time-consuming and stressful (Fan et al, 2020). Therefore, there is a need for intelligent road condition monitoring systems to automatically detect and report these anomalies. Additionally, the use of intelligent road fault detections methods provides an economically cheap and cost-effective method for road maintenance and road data acquisition.

Road condition monitoring systems are systems that detect the presence of road anomalies (Bello-Salau et al, 2019). Some of the popular road condition monitoring techniques use accelerometers and smartphones acceleration sensors (Bidgoli et al, 2018; Edwan et al, 2019; van Khang and Renault, 2019; Bello-Salau et al, 2020; Du et al, 2020) to detect anomalies due to the vibration of vehicles when these anomalies are encountered. However, most of these techniques suffer from limitations such as inaccuracy due to irregularly measured acceleration signals from the accelerometer and also the variation in sensitivity of acceleration sensors from different phone platforms. Moreover, the existing systems do not incorporate position tagging in their images. Due to these shortcomings, an intelligent road condition monitoring system using citizen sensing technique is proposed in this study. This system utilizes deep learning, specifically Tiny YOLOv4 object detection, to identify road defects.

Most of the reviewed literature (Anaissi et al, 2019; Kyriakou et al, 2019; Bansal et al, 2020; Chen et al, 2020; Kumar et al, 2020; Ping et al, 2020; Varona et al, 2020; Al-Shaghouri et al, 2021; Kamalesh et al, 2021) used image processing for the detection of road anomalies and recorded reasonably high efficiency. However, most of these works were unable to detect other road anomalies, while others require a large number of images to train the model for optimal performance. Other road anomaly detection techniques used accelerometers or smartphones acceleration sensors. However, these methods suffer from low accuracy due to noise from the sensors and also due to reasons like driver behaviour and pothole size. Crucially, many of the reviewed systems are designed as isolated detection tools rather than integrated components of a larger road infrastructure management ecosystem. They often lack the emphasis on scalable data collection, precise geotagging for actionable maintenance reports, or the architecture for long-term data aggregation necessary for predictive analysis. The proposed system seeks to address these limitations by developing an intelligent road condition monitoring system using citizen sensing technique that accurately monitors road conditions with a lesser number of input images.

Citizen sensing is when citizens own and use low-cost sensors to collect data about their environment, share those data with others, and also use those data to offer solutions to environmental issues (Coulson and Woods, 2021). In the prototype developed, an electronic device was mounted on the dashboard of

a vehicle which provided live feeds of the road. Anomalies on the road were then detected from the feeds using image processing. Images of the anomalies and their Global Positioning System (GPS) locations were logged on a database and were viewed and analyzed on a desktop application. This system can be extended to simultaneously accommodate many motorists.

While detecting potholes is a crucial and immediate need, the ultimate objective of this research was to develop a comprehensive, intelligent system for the remote monitoring of road conditions. Such a system would serve a dual purpose: (1) providing real-time driver assistance and warnings, and (2) generating actionable, data-driven insights for government agencies to plan and prioritize maintenance operations. By aggregating geotagged defect data over time, it becomes possible to not only identify problem areas but also to monitor the deterioration of road surfaces, estimate the scope and potential cost of repairs, and optimize resource allocation. The citizen sensing technique adopted here is the most viable and scalable method to populate this system with the vast amount of data required, dramatically reducing the capital expenditure associated with traditional sensor networks.

The images used in the study vary with camera height, vehicle speed, weather, lighting, road type and pothole size. The sample images used in the study were selected with a view to permitting representativeness of the samples across these dimensions.

2. MATERIALS AND METHODS

2.1. System Description

The development of an intelligent road condition monitoring system using citizen sensing technique involves numerous components. These components include a Raspberry Pi microcomputer, Pi camera, power supply, wire and case. The designed system makes use of the camera as the input source to the Raspberry Pi microcomputer when acquiring the road surface images. The Raspberry Pi then performs the anomaly detection process on the input stream from the camera, generates a stream with bounding boxes around the anomalies and sends the generated stream to the desktop application along with the GPS location of the anomalies which are stored in a database and displayed for the user. Figures 1 and 2 show the block diagram and the system operation flowchart respectively.

Figure 1 shows the block diagram of how the various components interact with one another. The input stream is supplied to the Raspberry Pi which is a microcontroller that serves as the brain of the system. The generated stream after the detection process and the GPS location of the detected anomalies from the input stream, are then sent to the Desktop application to be displayed and the relevant information is stored in the database for later retrieval and analysis.

Figure 2 shows the whole process involved in acquiring, gathering, processing and transmitting information from one part of the system to others. Road anomalies were detected from the images of the road surface using Tiny YOLOv4 object detection algorithm. The algorithm generates a video stream with bounding boxes around detected anomalies. The generated stream is then sent to the desktop application. If anomalies are detected, their GPS locations are captured, and the anomaly information is then sent to the database and also displayed on the desktop application.

2.2. Tiny YOLOv4 Object Detection Model Development

This section describes the methods employed in the detection of anomalies from the video feed of the road. The anomalies are detected using a deep learning model trained using the Tiny YOLOv4 object detection algorithm. Figure 3 shows the processes employed in training the model and finally using the model for real-time detection of the anomalies. The data acquisition process involved the collection of pothole images to train the model. The dataset used consisted of 1,784 images collected both manually and online. The dataset was then split into 1265 images for training, 401 for testing and 118 for validation. Figure 4 shows a pothole image from the dataset.

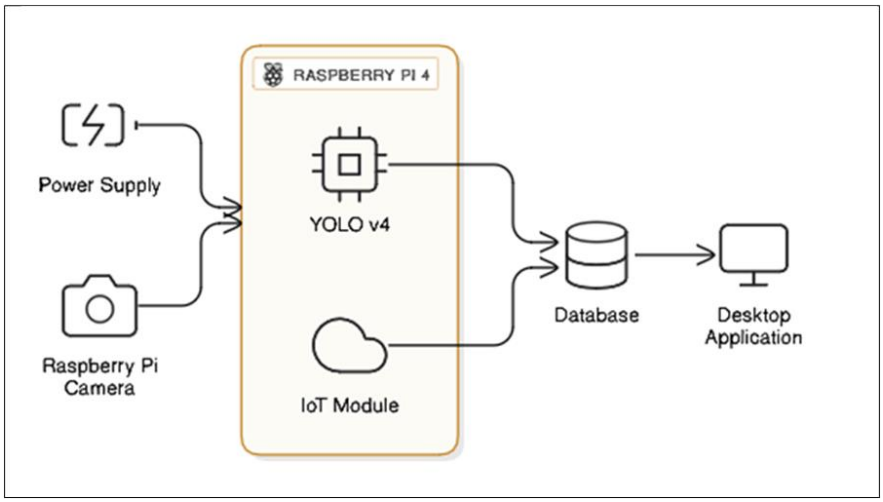

Figure 1: Architecture of intelligent road condition monitoring system

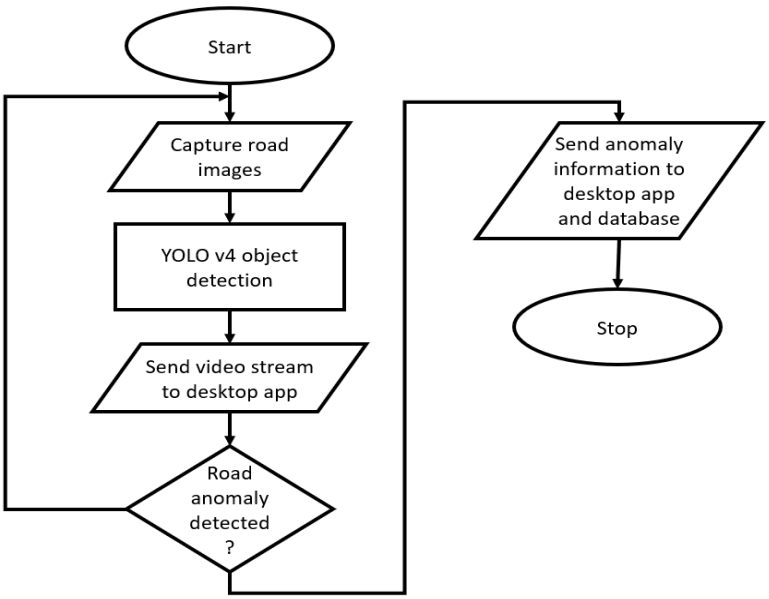

Figure 2: System flowchart

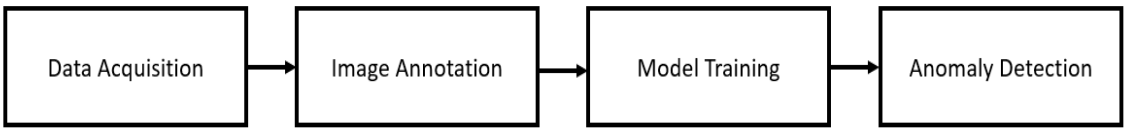

Figure 3: Model training and anomaly detection steps

After data acquisition, the next step was image annotation. This process involved drawing bounding boxes and generating the coordinates of the potholes in the training, testing and validation images from the dataset. The annotation process was carried out using LabelImg, an open-source image annotation tool. The dataset was labeled by one annotator to identify potholes according to predefined criteria. A pothole was defined as a surface depression on the road pavement with visible edges caused by material loss. Images could contain multiple potholes, and bounding boxes were drawn tightly around visible regions of damage. A quality check was conducted after annotation to remove ambiguous or inconsistent labels. These annotations give

information to the deep learning model about the images that it is being fed. Figure 4 shows the image after annotation.

Figure 4: Pothole image with annotation

After image annotation, the next step was training the model. The model was trained through transfer learning using a pre-trained Tiny YOLOV4 weight model, which was trained on the COCO (Common Objects in Context) dataset for detecting different objects. This enabled the training process to be faster as it was not training from scratch, and also because the number of images in the project's own dataset were not sufficient for training from scratch. The model was trained using the Tiny YOLOV4, which is suitable for edge devices such as Raspberry Pi. The selection of Tiny YOLOv4 was a deliberate compromise that prioritized high-speed inference suitable for edge devices like the Raspberry Pi over the marginal accuracy gains of larger models. Furthermore, initializing the network with pre-trained weights via transfer learning was essential to circumvent the constraints of a limited dataset, enabling efficient and effective training for the specific object detection task.

The training process was configured to enable training for 8000 steps with a batch size of 32. The learning rates were also scheduled to be reduced at steps 6400 and 7200. The training was done on Google Colaboratory using Darknet, an open-source neural network framework. The training process took approximately 7 hours to complete. After the training, the trained weight model files were obtained. Table 1 shows the model hyperparameters implemented in the model development.

After the model training process, the obtained model files were deployed on the Raspberry Pi for anomaly detection. Real-time video stream was collected from the pi camera using the Open-CV (Open-Source Computer Vision) Python Library. The anomaly detection process was then carried out on the live video stream, generating a stream with bounding boxes around the potholes and the confidence level above the bounding boxes. Figure 5 shows a frame from a stream generated from the anomaly detection process, showing bounding boxes around potholes and their confidence levels.

2.3. Desktop Application and IoT Platform Development

The Desktop application is used to retrieve the generated stream and GPS locations of the anomalies from the Raspberry Pi in real time. The application was designed using PyQt5, a Python framework for designing cross-platform GUI applications. After the stream and their GPS locations are retrieved, they will appear in the application for viewing by the user.

After the anomaly detection process, the generated stream and the GPS locations of the anomalies are sent from the Raspberry Pi to the desktop application through network sockets. The Raspberry Pi and the desktop have to be connected to the same network for the stream and the GPS location to be sent and received successfully. If the anomalies are detected, the relevant information is stored on a database locally on the desktop. Figure 6 shows the interaction between the Raspberry Pi and the desktop through the internet.

Table 1: Selected model hyperparameters

Parameter	Value	Role
Network input size	416x416	Defines the resolution of input images fed into the CNN.
Total images	1784	The dataset size used for training.
Number of classes	1	Specifies the target categories to be detected. In this case, only one class (pothole) is being classified.
Batch size	32	Determines how many samples are processed before updating model weights. Affects training stability and speed.
Subdivision	1	Splits the batch into smaller mini-batches for memory efficiency. Lower values require more GPU memory but speed up training.
Maximum batches	8000	Defines the total number of training iterations. Higher values allow more learning but may risk overfitting.
Learning rate	0.00261	Controls the step size during weight updates. Value ensures efficient convergence without overshooting.
Activation	Leaky ReLU	Defines how neurons transform inputs into outputs in the network. The choice of activation function introduces non-linearity, enabling the model to learn complex patterns.
Burn in	1000	A phase where the learning rate gradually increases from a small value to the specified maximum. Helps stabilize training and avoid large weight updates at the beginning.
Training-validation testing Split	71:7:22	Determines how data is divided: training for learning patterns, validation for tuning hyperparameters, and testing for evaluating final model performance. Ensures generalization and prevents overfitting.
Weight decay	0.0005	Prevents overfitting by penalizing large weights, encouraging the model to learn simpler patterns for object detection. Regularizes the network by penalizing large weights, reducing overfitting.
Momentum	0.9	Helps accelerate gradient descent in the right direction, improving convergence speed and stability during training.
Data augmentation	Angle = 0, Saturation = 1.5, exposure = 1.5, hue = 0.1	Increases dataset diversity by applying transformations, enhancing the model's robustness to variations in object appearance and orientation.
Input normalization	1	Ensures consistent data scale, leading to faster convergence and more stable object detection performance.
Learning rate schedule policy	Steps = 6400 and 7200	Adjusts the learning rate over time to balance fast learning early on and fine-tuning later for optimal object detection accuracy.

Figure 5: Frame from generated stream

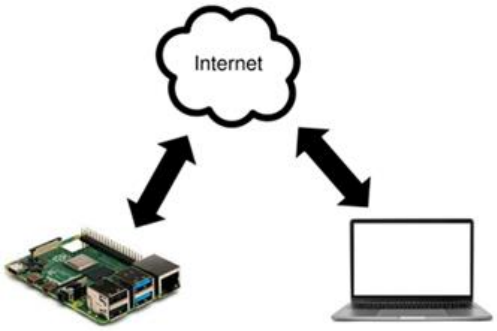

Figure 6: Interaction between Raspberry Pi and desktop application

2.4. Prototype Development of Intelligent Road Condition Monitoring

2.4.1. Hardware design

The hardware components consist of a Raspberry Pi camera module to get and send images of the road to the processing unit of the system, a Raspberry Pi microcomputer for the processing of images and control of the system, a GPS module, a battery, and a case to house all the components. The Raspberry Pi camera module is a compact and lightweight camera designed to be compatible with Raspberry Pi boards. It utilizes the MIPI camera serial interface protocol to establish communication with the Raspberry Pi. The camera module is commonly employed in various applications such as image processing, machine learning, and surveillance projects. In the context of road-related projects, the Raspberry Pi camera can be utilized to capture images of the road, which can then be transmitted to the Raspberry Pi for further analysis and processing.

The Raspberry Pi is an affordable and compact computer that can be connected to a monitor or TV and operated with a standard keyboard and mouse. Despite its small size, the Raspberry Pi is a powerful device that can be utilized for various projects, particularly in the fields of Internet of Things (IoT) and general computing. In the context of detecting road anomalies in images captured by the camera, the Raspberry Pi can play a crucial role. It can process and analyse the captured images using various algorithms and techniques, allowing for the detection of road anomalies such as potholes or speed bumps. The Raspberry Pi's capabilities and flexibility make it a suitable platform for implementing road anomaly detection systems. A 5 V battery was used to power the whole system, which is suitable for the Raspberry Pi.

2.4.2. Software design

The software components of the system comprise a Python program to implement the road anomaly detection algorithms and a MATLAB program that was used for system modelling simulation. The components are described below:

i. Python: Python is a popular high-level programming language created by Guido Van Rossum in 1991. Python was used in retrieving the live stream from the Pi camera and for carrying out the object detection process on it. It was also used in the desktop application development.

ii. MATLAB: MATLAB, short for Matrix Laboratory, is a high-level programming language focused on numerical computation and simulation. In this project, MATLAB was used for model simulation of the system to test how the system would work.

iii. Darknet: Darknet is an open-source neural network framework written in C and CUDA and used to train YOLO models. It is known for its speed and efficiency in implementing deep neural networks for computer vision tasks. In this project, the deep learning model was trained using Darknet.

iv. Google Colaboratory: Google Colaboratory (Colab) is a free service created by Google Research. It is a Linux machine that has an interface based on the Jupyter Notebook service. This virtual computer features a 2-core CPU, 12GB of RAM, and is upgradeable for free to 25GB if additional memory is required. When using the notebook, the GPU was selected at random. Available GPUs

include Nvidia Tesla K80s, T4s, P4s, and P100s. In this project, the deep learning model was trained on Colab as it has more computational resources than a regular PC.

2.5. Performance Evaluation

The system was tested for its precision, recall, F1-score, average precision, and mean average precision in detecting the road anomalies. Precision is a performance metric used to evaluate the quality of predictions made by a deep learning model. The mathematical expression for precision is given in Equation (1). Recall is used to evaluate the completeness or ability of a deep learning model to identify all relevant instances of a particular class. It is represented mathematically in Equation (2). The F1-score is a performance metric commonly used in deep learning to assess the balance between precision and recall. The mathematical expression for F1-score is given in Equation (3). Average Precision (AP) is a performance metric commonly used in object detection tasks to evaluate the quality of ranked lists or detection results. It is represented mathematically in Equation (4). The Average Precision measures a model's accuracy by summarizing its performance across different recall levels, essentially calculating the area under the Precision-Recall (PR) curve for a single object class, indicating how well it balances finding objects (recall) with being correct (precision). Mean average precision (mAP) is a performance metric commonly used in object detection tasks to evaluate the average precision across multiple classes or categories. The mathematical expression for mean average precision is given in Equation (5).

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{1}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2}$$

$$\text{F1 Score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Presicion} + \text{Recall}} \tag{3}$$

$$\text{Average Precision(AP)} = \int_0^1 p(r)dr \tag{4}$$

$$\text{Mean Average Precision (mAP)} = \frac{1}{\text{N}} \sum_{\text{i=1}}^{\text{N}} \text{AP}_\text{i} \tag{5}$$

Here True Positive (TP) indicates a positive instance that is correctly predicted as positive; False Positive (FP) refers to a negative instance that is incorrectly predicted as positive; True Negative (TN) represents a negative instance that is correctly predicted as negative; and False Negative (FN) denotes a positive instance that is incorrectly predicted as negative. N is the number of object classes or categories.

3. RESULTS AND DISCUSSION

In this section, results obtained from the design, development and testing of the road condition monitoring system are presented and discussed. The section comprises of the result achieved from training the deep learning model, the hardware and software implementation of the system and the performance of the system obtained from the result of the performance evaluation metrics.

3.1. Tiny YOLOv4 Object Detection Algorithm

During the training of the Tiny YOLOv4 object detection model, the performance of the network is often monitored through the use of a loss plot. This plot provides a visual representation of how well the model is learning over time by tracking changes in the loss function across training iterations. Figure 7 presents the loss plot obtained throughout the training process.

A loss plot illustrates the evolution of the loss function, which measures the difference between predicted outputs and the actual ground truth labels. In the case of Tiny YOLOv4, a deep neural network-based algorithm, the model's parameters are updated during training to minimize this difference. A steadily

decreasing loss indicates effective learning and improved predictions, while a plateau or rising loss may suggest challenges such as overfitting or stagnation in the learning process. As shown in Figure 7, the loss value decreases progressively as training iterations increase. This downward trend confirms that the Tiny YOLOv4 model was successfully learning from the training data and improving its prediction accuracy. The observed pattern highlights the effectiveness of the training process and indicates that the model was converging toward optimal performance.

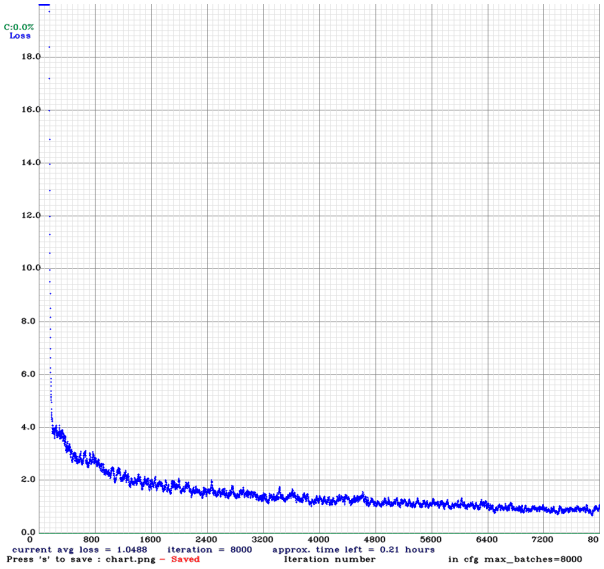

Figure 7: Loss plot during training

3.2. Intelligent Road Condition Monitoring Prototype

For the practical implementation of the road condition monitoring system, all hardware components were assembled and connected to form a complete unit. This integration ensured that each component worked together to support the system's functionality. Figure 8(a) illustrates the assembled system with all the hardware components properly connected. Once assembled, the system was enclosed within a casing to protect the internal components, improve durability, and enhance the overall appearance. Figure 8(b) presents the prototype after casing, showing a more compact and portable design.

(a) (b)

Figure 8: Developed prototype (a) before casing, and (b) after casing

The Raspberry Pi, embedded within the device, serves as the central processing unit, while an 8500 mAh power bank provides power for standalone operation. When powered on, the system automatically connects to a dedicated internet service shared with the desktop application. It continuously captures image frames of the road environment and employs the Tiny YOLOv4 algorithm to detect potholes in real time. The identified potholes, along with their GPS coordinates, are transmitted to the desktop application for monitoring.

Essentially, the system demonstrates a robust, portable, and efficient solution for road condition monitoring. By placing the device on a vehicle dashboard, real-time pothole detection and reporting become possible, contributing to safer and smarter road infrastructure management.

3.3. Intelligent Road Condition Monitoring Desktop Application

To complement the hardware system, a desktop application was developed to receive and display data from the Raspberry Pi in real time. This application not only displayed the video stream but also overlaid bounding boxes around detected potholes, as generated by the Tiny YOLOv4 model. In addition, the GPS coordinates of each detected pothole were transmitted from the Raspberry Pi and displayed within the application interface. The desktop application was implemented using PyQt, a Python library for creating cross-platform graphical user interfaces. Figure 9 illustrates the application in operation, receiving and displaying a live video stream from the Raspberry Pi.

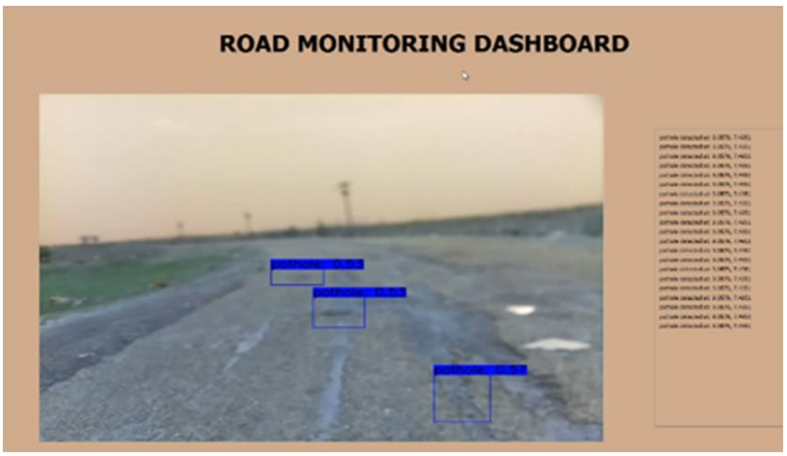

Figure 9: Desktop application with online detection

Communication between the Raspberry Pi and the application was established using network sockets, which serve as endpoints for sending and receiving data across computer networks. Beyond real-time streaming, the application also supports offline analysis by allowing users to load stored video files and perform pothole detection. In this mode, the application processes the video and generates bounding boxes around the identified potholes, as demonstrated in Figure 10.

In summary, the desktop application serves as an essential interface for both real-time and offline road condition monitoring. Its ability to integrate video streaming, pothole detection, and GPS location tracking provides a comprehensive monitoring tool. This dual functionality enhances system flexibility, making the application suitable for live deployments as well as retrospective road infrastructure analysis.

Figure 10: Desktop application with offline detection

3.4. Performance Evaluation

To assess the effectiveness of the developed pothole detection system, a set of widely adopted performance evaluation metrics was applied. These metrics provide insight into the model's accuracy, robustness, and reliability in detecting potholes under different conditions. The results obtained are summarized in Table 2, which presents the values of precision, recall, F1-score, average Intersection over Union (IoU), average precision, and mean average precision (mAP@0.50).

Table 2: Performance evaluation results

Performance metric	Value (%)
Precision	77.00
Recall	66.00
F1-Score	71.00
Average IoU	58.65
Average Precision	69.99
Mean Average Precision (mAP@0.50)	69.99

The system achieved a precision of 77%, indicating that 77% of the potholes predicted by the model were correct. The recall of 66% shows that the model successfully identified 66% of all potholes present in the dataset. Combining these metrics, the F1-score of 71% reflects a balanced trade-off between precision and recall, representing moderate overall performance. Furthermore, the average IoU of 58.65% demonstrates the model's capability in generating bounding boxes that moderately overlap with the ground truth. In terms of object detection accuracy, the system attained an average precision of 69.99%, while the mAP@0.50 was also recorded at 69.99%, confirming consistent performance at the set threshold.

To contextualize the performance of the proposed pothole detection model, a comparative analysis was conducted against recent studies that have also focused on efficient, edge-compatible object detection frameworks. The selection of comparable works was prioritized based on the use of public datasets and lightweight architectures suitable for potential real-world deployment. Table 3 summarizes the key performance metrics (including Precision, Recall, F1-Score, and mAP@0.5), providing a baseline for evaluating the efficacy of our Tiny YOLOv4 implementation.

Table 3: Comparative analysis with other models

Ref	Dataset type	Anomaly types	Technique	Precision (%)	Recall (%)	F1-Score	mAP@0.5 (%)
Asad et al. (2022)	Public	Potholes	MobileNet v2	42.0	56.0	47.9	47.4
Asad et al. (2022)	Public	Potholes	Tiny YOLO v4	76.0	75.0	76.0	80.04
Chen et al. (2019)	Public	Potholes	SSD-MobileNet	32.86	76.28	45.92	-
Ours	Public and Private	Potholes	Tiny YOLO v4	77.0	66.0	71.0	69.99

The comparative analysis reveals critical trade-offs and performance alignments within the domain of lightweight pothole detection. Our model achieved the highest precision (77.0%) among the compared works, indicating a superior ability to minimize false positives. This is a valuable trait in scenarios where unnecessary alerts must be avoided. However, this high precision comes with a corresponding decrease in recall (66.0%) compared to some counterparts, such as Chen et al. (2019), who achieved a recall of 76.28% but with significantly lower precision (32.86%). This inverse relationship highlights the classic precision-recall trade-off. Our model strikes a balance, resulting in a robust F1-Score of 71.0, which is substantially higher than that of Chen et al. (2019) but lower than the 76.0 reported by Asad et al. (2022) using the same Tiny YOLOv4 architecture. The discrepancy with the results of Asad et al. (2022) can be attributed to differences in the dataset composition and the specific anomalies present, particularly as our model was trained on a combined public and private dataset, which may introduce greater variability and complexity.

Ultimately, the results confirm that our developed system provides a competitive and well-balanced performance profile, effectively leveraging the Tiny YOLOv4 architecture for reliable pothole detection. The evaluation results highlight that the proposed pothole detection system delivers a moderate yet reliable performance across multiple key metrics. While there is room for significant improvement, especially in recall and bounding box precision, the system offers a promising foundation for further research and enhancements in the future. The research team plans to pursue the most pressing of the deficiencies identified in the current system.

4. CONCLUSION

An intelligent road condition monitoring system based on citizen sensing technique and Tiny YOLOv4 deep learning model was successfully built and tested. The system was developed in accordance with the specified requirements for accurately detecting potholes on road surfaces. The evaluation of the trained model yielded a precision of 77%, a recall of 66%, an F1-score of 71%, and a mean Average Precision (mAP@0.50) of 69.99%. These metrics indicate the model's capability to accurately detect potholes. Furthermore, the system demonstrated practical utility by seamlessly processing real-time video streams, successfully detecting and annotating potholes, while the accompanying desktop application effectively displayed these results alongside their GPS coordinates. This research establishes a critical first step towards our ultimate goal of a large-scale, remote road monitoring infrastructure for driver assistance and proactive road maintenance. The use of citizen-sourced data presents a transformative, cost-effective method for gathering the vast geotagged dataset required to monitor road network health. The successful prototype demonstrates that this approach is technically viable, although many challenges still need to be addressed.

However, the reliability of any system built on crowd-sourced data is contingent on the quality and accuracy of its inputs. Therefore, the most immediate and critical focus of our future work is the development and implementation of a robust data validation framework. The current system logs detected anomalies and locations; the next stage of research will focus on designing algorithms to automatically verify citizen-reported data. This will involve techniques such as cross-validating reports from multiple independent users at the same location. Establishing this validation mechanism is the essential next step to ensure the data is trustworthy enough for maintenance authorities to base budgetary and operational decisions upon, thereby transforming raw data into actionable and reliable intelligence for smart infrastructure management.

The other challenges the research hopes to address subsequently include the following:

i. The validation set used in this paper is only 118 images. That may seem small for hyperparameter selection when the test set is 401 images. Ongoing research uses k-fold cross-validation and repeated random splits where feasible, and also expands the validation set size. Furthermore, it plans to report the mean and standard deviation of key metrics over multiple runs to capture training variability.

ii. This paper reports only mAP at a single IoU threshold of 0.50. The current community standard is to report mAP averaged across IoU thresholds from 0.50 to 0.95 in steps of 0.05. Ongoing extension of this research includes mAP@[0.50:0.95], AP at different IoU thresholds, and per size AP for small medium and large potholes. It also plans to provide precision recall (PR) curves and PR area.

iii. Ongoing research plans to provide confusion analysis and a breakdown of false positives and false negatives, including typical failure modes: for example, false detections on shadows, water patches, markings, or certain textures. It plans to show representative true positive and false positive examples.

iv. The core target of this paper was real-time detection on Raspberry Pi. Yet, the paper gives no inference benchmark on the Pi. Ongoing extension of this research plans to report frames per second, CPU and memory usage, thermal throttling behaviour and latency for the detection pipeline, including camera capture, preprocessing, inference and post-processing. The team is also investigating issues that must be considered for real deployments, such as model optimization steps by means of, for example, quantization, pruning, and using a TFLite or TensorRT build. Also, future publications will report trade-offs between speed and accuracy, sampling frequency, expected accuracy, and synchronization between image frames and GPS timestamps. The

research recognizes that for real applications in road maintenance, the user must quantify geolocation error and show how multiple readings at different vehicle speeds affect the location estimate.

v. The prototype uses local network sockets and requires Pi and desktop to be on the same network. This is not fully scalable for citizen sensing. Alternative architectures such as local storage and batch upload over mobile networks, MQTT (Message Queuing Telemetry Transport) with brokers, or server-side ingestion is being considered. Also, considerations are being made for intermittent connectivity, which is a reality in countries like Nigeria.

vi. The ongoing extensions to the research include head-to-head comparisons to other detectors on the same dataset. It hopes to compare Tiny YOLOv4 to at least one other lightweight model, such as MobileNet SSD, Tiny YOLOv3 or EfficientDet lite. They also include ablation on input resolution, augmentation types and different training schedules. This is needed to show that the choices made are always justified and not arbitrary.

vii. The reported metrics are moderate: precision 77%, recall 66%, F1 71%, average IoU 58.65% and mAP@0.50 at 69.99%. These values reflect reasonable baseline performance. With many real-world conditions and noise, the recall in deployment may drop further. It is, therefore, important to provide more targeted analysis (such as performance by lighting conditions and pothole size) and present confidence intervals for each scenario.

viii. Citizen sensing raises privacy and legal questions. Addressing privacy of video capture in public spaces, consent, and potential misuse is very important. Any system designed for field and public use must explain sufficiently well how personally identifiable information is handled and how data governance is implemented. These issues ae being addressed in the ongoing extensions of this research.

5. ACKNOWLEDGMENT

The authors acknowledge the Tertiary Education Trust Fund (TETFUND) of Nigeria that funded this work through the Institution-Based Research Initiative (IBRI) with grant No. TETFund/FUTMINNA/2024/092.

6. CONFLICT OF INTEREST

There is no conflict of interest associated with this work.

REFERENCES

Ahmad, I., Suhaimi, M. F. and Yusri, N. A. N. (2019). Development of alcohol sensor detector with engine locking system for accident prevention. In *AIP conference proceedings*, Vol. 2129, No. 1, pp. 020196, *5th International Conference on Green Design and Manufacture 2019*, Jawa Barat, Indonesia, 29-30 April 2019. AIP Publishing LLC. https://doi.org/10.1063/1.5118204.

Asad, M. H., Khaliq, S., Yousaf, M. H., Ullah, M. O. and Ahmad, A. (2022). Pothole Detection Using Deep Learning : A Real-Time and AI-on-the-Edge Perspective. *Advances in Civil Engineering*, *2022*, 1–13.

Al-Shaghouri, A., Alkhatib, R. and Berjaoui, S. (2021). Real-time pothole detection using deep learning. Available electronically at https://doi.org/10.48550/arXiv.2107.06356. Last accessed on October 23, 2025.

Anaissi, A., Khoa, N. L. D., Rakotoarivelo, T., Alamdari, M. M. and Wang, Y. (2019). Smart pothole detection system using vehicle-mounted sensors and machine learning. *Journal of Civil Structural Health Monitoring*, 9(1), pp. 91-102. https://doi.org/10.1007/s13349-019-00323-0.

Bansal, K., Mittal, K., Ahuja, G., Singh, A. and Gill, S. S. (2020). DeepBus: machine learning based real time pothole detection system for smart transportation using IoT. *Internet Technology Letters*, 3(3), pp. 2-7. https://doi.org/10.1002/itl2.156.

Bello-Salau, H., Aibinu, A. M., Onumanyi, A. J., Onwuka, E. N., Dukiya, J. J. and Ohize, H. (2020). New road anomaly detection and characterization algorithm for autonomous vehicles. *Applied Computing and Informatics*, 16(1-2), pp. 223-239. https://doi.org/10.1016/j.aci.2018.05.002.

Bello-Salau, H., Onumanyi, A. J., Aibinu, A. M., Onwuka, E. N., Dukiya, J. J. and Ohize, H. (2019). A survey of accelerometer-based techniques for road anomalies detection and characterization. *International Journal of Engineering Science and Application*, *3*(1), pp. 8-20.

Bidgoli, M. A., Golroo, A., Nadjar, H. S., Rashidabad, A. G. and Ganji, M. R. (2019). Road roughness measurement using a cost-effective sensor-based monitoring system. *Automation in Construction*, 104, pp. 140-152. https://doi.org/10.1016/j.autcon.2019.04.007.

Cao, J., Song, C., Peng, S., Xiao, F. and Song, S. (2019). Improved traffic sign detection and recognition algorithm for intelligent vehicles. *Sensors*, 19(18), pp. 4021. https://doi.org/10.3390/s19184021.

Chen, H., Yao, M. and Gu, Q. (2020). Pothole detection using location-aware convolutional neural networks. *International Journal of Machine Learning and Cybernetics*, 11(4), pp. 899-911. https://doi.org/10.1007/s13042-020-01078-7.

Chen, S. Y., Zhang, Y., Zhang, Y. H., Yu, J. J. and Zhu, Y. X. (2019). Embedded system for road damage detection by deep convolutional neural network. Mathematical biosciences and engineering: MBE, 16(6), 7982-7994.

Coulson, S. and Woods, M. (2021). Citizen sensing: an action-orientated framework for citizen science. *Frontiers in Communications*, 6, pp. 629700. https://doi.org/10.3389/fcomm.2021.629700.

Deme, D. (2019). Review on factors causing road traffic accident in Africa. *Journal of Architecture and Construction*, 2(3), pp. 41-49. https://doi.org/10.22259/2637-5796.0203004.

Du, R., Qiu, G., Gao, K., Hu, L. and Liu, L. (2020). Abnormal road surface recognition based on smartphone acceleration sensor. *Sensors*, 20(2), pp. 451. https://doi.org/10.3390/s20020451.

Edwan, E., Sarsour, N. and Alatrash, M. (2019). Mobile application for bumps detection and warning utilizing smartphone sensors. In: Proceedings of 2019 International Conference on Promising Electronic Technologies (ICPET), Gaza, Palestine, October 23-24, 2019, pp. 50-54. IEEE. https://doi.org/10.1109/ICPET.2019.00017.

Fan, R., Ozgunalp, U., Hosking, B., Liu, M., and Pitas, I. (2020). Pothole detection based on disparity transformation and road surface modelling. *IEEE Transactions on Image Processing*, 29(11210017), pp. 897-908. https://doi.org/10.1109/TIP.2019.2933750.

Kamalesh, M. S., Chokkalingam, B., Arumugam, J., Sengottaiyan, G., Subramani, S. and Shah, M. A. (2021). An intelligent real time pothole detection and warning system for automobile applications based on IoT technology. *Journal of Applied Science and Engineering*, 24(1), pp. 77-81. https://doi.org/10.6180/jase.202102_24(1).0010.

Kotagi, P. B., PK, N., Madakari, P., OV, H. and Vishwanath, N. (2020). Overspeed Detection using Image Processing for Indian Highways. *International Journal of Engineering and Advanced Technology*, 9(5), pp. 1202-1207. https://doi.org/10.35940/ijeat.E1123.069520.

Kumar, A., Chakrapani, C., Kalita, D. J. and Singh, V. P. (2020). A modern pothole detection technique using deep learning. In: Proceedings of 2nd International Conference on Data, Engineering and Applications (IDEA), Bhopal, India, 28-29 February 2020, https://doi.org/10.1109/IDEA49133.2020.9170705.

Kyriakou, C., Christodoulou, S. E. and Dimitriou, L. (2019). Smartphone-based pothole detection utilizing artificial neural networks. *Journal of Infrastructure Systems*, 25(3), p. 04019019. https://doi.org/10.1061/(asce)is.1943-555x.0000489.

Ping, P., Yang, X. and Gao, Z. (2020). A deep learning approach for street pothole detection. In: Proceedings of 2020 IEEE Sixth International Conference on Big Data Computing Service and Applications (BigDataService), Oxford, United Kingdom, 3-6 August 2020, pp. 198-204. IEEE. https://doi.org/10.1109/BigDataService49289.2020.00039.

Rodrigue, J. P. (2020). *The Geography of Transport Systems*. Routledge. https://doi.org/10.4324/9780429346323.

Ukonze, F. I., Nwachukwu, M. U., Mba, H. C., Okeke, D. C. and Jiburum, U. (2020). Determinants of vehicle ownership in Nigeria. *Sage Open*, 10(2), pp. 1-13. https://doi.org/10.1177/2158244020922970.

Van Khang, N. and Renault, É. (2019). Cooperative sensing and analysis for a smart pothole detection. In: Proceedings of 15th International Wireless Communications & Mobile Computing Conference (IWCMC), Tangier, Morocco, June 24-28, 2019, pp. 1785-1790. IEEE. https://doi.org/10.1109/IWCMC.2019.8766600.

Varona, B., Monteserin, A. and Teyseyre, A. (2020). A deep learning approach to automatic road surface monitoring and pothole detection. *Personal and Ubiquitous Computing*, 24(4), pp.519-534. https://doi.org/10.1007/s00779-019-01234-z.

Yusuf, I. E. (2020). The impact of road transport on tomato production and marketing in Nigeria. *Journal of Nigeria Transport History*, 1(2), pp. 1-18.