IoT-Based Railway Rail Fitting Detection Using Infrared Algorithm

Vinodhini S.[1], Sundhar Raj R.[2], Jeeva S.[3]

[1]Assistant Professor, [2,3]Student, Department of Information Technology, Velammal Engineering College, Surapet, Chennai, India.

E-mail: [1]s.vinodhini1991@gmail.com, [2]sundharraj332006@gmail.com, [3]jeevajeevasekar228@gmail.com

Abstract

The reliability of railway lines relies heavily on rail fittings such as clips and fasteners that keep the rails firmly attached to the sleepers. Failure in rail fittings such as clips leads to an increased risk of derailments. This paper highlights the development of a monitoring system based on infrared sensing technique, microcontroller and cellular network to detect rail faults caused by fittings failure. When the readings in the reflected signal from the fittings become too low or erratic, an infrared sensor signals the Arduino Nano microcontroller to identify a fault. On confirmation of the fault, Raspberry Pi microcomputer triggers the camera to read the Quick Response code from the railway and locate the fault's location before the information is sent to maintenance crews through GSM module. In testing this technology using the model rail track, the proposed system successfully detected faults in eight test cases where it achieved 100% accuracy in detecting missing fittings and 92% for loose fittings. The entire system operates automatically and uses low-cost parts.

Keywords: Rail Fitting Detection, Infrared Sensing, IoT, Railway Safety, GSM Alert, QR Code Localization.

1. Introduction

Rail fitting malfunctions like broken elastic fasteners, loose bolt fasteners, and misaligned base plates are one of the major reasons for derailments around the world. Although there is no question about the seriousness of this threat, the current approach to detecting problems with rail fittings is mostly based on routine visual inspections, and does not take into

DOI: https://doi.org/10.36548/rrrj.2026.1.002
Received: 25.02.2026, received in revised form: 27.03.2026, accepted: 15.04.2026, published: 27.04.2026

account problems arising in between inspection passes. It is possible for a rail fitting to come loose within a few hours after completion of an inspection pass and remain unnoticed for days. Clearly, there is a strong need for an automated solution capable of constant monitoring, fault detection and accurate localization.

The current solutions like machine vision inspection trains, accelerometer sensors for vibration measurement and acoustic emission sensors require either dedicated train runs for checking all rails, complex calibration routines or substantial processing power and, therefore, are unsuitable for wide scale deployment across the network. Here we propose a decentralized solution with one low power infrared sensor unit located at each rail fitting. Every unit will monitor the fitting and signal only upon detection of malfunction. Affordable commercial hardware combined with QR code location marking makes the cost-per-node reasonable.

1.1 Research Objectives

The specific objectives of this research are,

1. Designing a cost-effective IoT sensor node that continuously tracks the presence of any missing or loose rail fastenings through infrared reflectance measurements.
2. Development of an algorithm for detecting loose fastenings based on threshold value with variance analysis and debounce feature to eliminate false alarms.
3. Concurrent integration of QR code geolocation with the Arduino-to-Raspberry Pi data pipeline to match each detected anomaly with the corresponding rail section location.
4. Delivering fault notifications in the form of structured messages to the authorities in charge of maintenance, using GSM, within 15 seconds of detection.
5. Conducting field testing of the prototype on a test rail track, determining its detection accuracy, reaction time, and percentage of false alerts.

2. Literature Survey

A wide variety of methods used for railway fault detection are indicated by the literature survey provided in Table 1, ranging from machine learning and signal processing techniques to computer vision and IoT-based systems. Initial work concentrated on methods like recurrent

neural network-based fault diagnosis, classifier fusion, and threshold-based condition monitoring methods that enhanced the efficiency of diagnosing problems with railway tracks. Thanks to advancements in computational techniques, deep learning, and computer vision methods have been extensively studied for use in railway inspection, making it possible to automatically detect cracks and other defects through visual inspection. Although computationally intensive, they can only be implemented on inspection cars rather than in a continuous manner.

Table 1. Summary of Related Works in Railway Fault Detection

Reference	Method / Technique	Findings / Contribution
De Bruin et al. [1]	RNN / LSTM	LSTM outperforms CNN for fault detection
Oukhellou et al. [2]	Classifier Fusion	Improved diagnosis robustness
García Márquez & Roberts [3]	Thresholding	Reduced unplanned outages
Vileiniskis & Remenyte-Prescott [4]	OCSVM	Reliable classification
Karakose et al. [5]	Image Processing	Detects faults at speed
James et al. [6]	CNN	Accurate crack localisation
Shafique et al. [7]	CNN	High detection accuracy
Jo et al. [8]	IoT Framework	Architecture for smart railways
Fraga-Lamas et al. [9]	IIoT Review	Identified challenges
Shah et al. [10]	IoT / Acceleration	Real-time damage analysis
Ghosh et al. [11]	Signal Processing	Real-time detection
Siddiqui et al. [12]	IoT	Localized faults
Chellaswamy et al. [13]	Cloud Monitoring	Remote monitoring
Iyer et al. [14]	LEACH Protocol	Detects multiple defects

Minguell & Pandit [15]	Deep Learning	Compared detection models
Padhi et al. [16]	IoT Sensors	Identifies faults
Saritas et al. [17]	Deep Learning	Strong classification
Krishna et al. [18]	IR Sensor	Efficient crack detection

On the other hand, IoT systems have proven popular for real-time, distributed railway monitoring purposes. Some researches suggested sensor-based methods of faults detection utilizing acoustic analysis and vibration measurement techniques, in addition to cloud-based architectures that could collect and process data. Even though they have a great potential for scalability, they can require additional calibration or expensive hardware installations. Finally, some researchers applied infrared sensor-based systems to monitor physical defects in railroads.

Even with all these developments, many of the current solutions do not achieve a perfect balance among low cost, scalability, and real-time monitoring coupled with accurate localization. This is what drives the development of the proposed system which combines infrared technology, QR code localization, and GSM communications.

3. Proposed Methodology

The proposed system that involves the use of Internet of Things technology for fitting monitoring on rails and uses infrared detection along with QR code-based position tagging. In general, the idea behind the system involves fixing a small-size sensor near each rail fitting and continually monitoring it through an infrared detector. This is done using the use of a cellular module which alerts the locality once a malfunctioning occurs in the fitting area. It is designed to be very simple and relatively cheap in nature to allow the deployment of such sensors in large numbers on tracks spanning over large distances. The processing part of the circuit is divided into two parts; the Arduino Nano board, which deals with real-time sensor data and threshold values, and the Raspberry pi, which controls the camera and QR codes. Weatherproof labels with QR codes that indicate fitting numbers are placed near each rail fitting location. In other words, it allows fixed-positioning without the need for any positioning hardware.

3.1 The Infrared Detection Algorithm

The infrared detector algorithm runs using an Arduino Nano through a continuous polling technique. Infrared beams are released by the infrared sensor that aims for the rail fitting, usually metallic. Where the rail fitting is placed, it causes the reflection of the beam to bounce back into the infrared sensor. By analyzing the analog value and comparing it against a predetermined threshold, if the values remain higher than the set threshold, then the fitting is considered detected. However, when there is a missing rail fitting, it causes the infrared beam to get scattered, and since the infrared sensor gets values lower than the threshold, a fault is detected. The algorithm includes a debouncing window of 200 to 500 milliseconds to detect any transient interferences, and if the values remain less than the threshold during this time, the fault is recorded. If the rail fitting is partially loosened, this causes inconsistency in the signals received, and through this, the algorithm calculates the variance of the values taken over ten successive readings, where if the values exceed a set threshold, a loose fitting is recorded.

Once the presence of a fault is established, the Arduino triggers a command to the Raspberry Pi, which subsequently triggers the camera to scan a QR code using the OpenCV and Pyzbar modules. The QR code contains the information related to the track segment and fitting number, after which a message is formed that incorporates the fault details and sent out through a cellular network to the maintenance department. The complete procedure takes about eight to twelve seconds to accomplish.

3.2 Pseudocode

3.2.1 Arduino Nano Process

```
PROCEDURE Arduino_Main
   INITIALISE IR_sensor on analog pin A0
   SET T_th = 500
   SET T_var = 40000
   SET N = 10
   SET debounce_period = 300 milliseconds
   INITIALISE reading_buffer as empty array of size N
   SET debounce_timer = 0

   LOOP continuously
      READ V from IR_sensor analog input

      IF V < T_th THEN
         INCREMENT debounce_timer by polling interval
```

```
        IF debounce_timer >= debounce_period THEN
          SET fault_type = MISSING
          SET GPIO trigger pin HIGH
          SET GPIO fault type pin to MISSING code
          RESET debounce_timer to 0
        END IF

      ELSE
        RESET debounce_timer to 0
        APPEND V to reading_buffer

        IF reading_buffer length == N THEN
          COMPUTE mu = mean of reading_buffer
          COMPUTE sigma_sq = variance of reading_buffer

          IF sigma_sq > T_var THEN
            SET fault_type = LOOSE
            SET GPIO trigger pin HIGH
            SET GPIO fault type pin to LOOSE code
          END IF

          REMOVE oldest value from reading_buffer
        END IF
      END IF

      DELAY 50 milliseconds
    END LOOP

END PROCEDURE
```

3.2.2 Raspberry Pi Process

```
PROCEDURE RPi_Main
  INITIALISE GPIO input pins
  INITIALISE camera module
  INITIALISE GSM serial port at 9600 baud
  SET hold_period = 60 seconds
  LOOP continuously
    IF GPIO trigger pin == HIGH THEN
      READ fault_type from GPIO fault type pin
      SET GPIO trigger pin LOW
      ACTIVATE camera module
      CAPTURE frame from camera
      PREPROCESS frame using OpenCV grayscale conversion
      DECODE frame using pyzbar QR decoder
      IF decoded QR result is not empty THEN
        SET location_id = decoded QR string
        SET timestamp = current system clock time
        COMPOSE message as:
```

```
            FAULT, fault_type, LOCATION, location_id, TIME, timestamp
          SEND AT command to GSM module
          SEND message string to GSM module
          SEND end-of-message character to GSM module
        END IF
        DEACTIVATE camera module
        SLEEP for hold_period
      END IF
      DELAY 100 milliseconds
    END LOOP
END PROCEDURE
```

3.3 System Workflow and Operational Sequence

The process illustrated in Figure 1 below provides a detailed flow chart that shows the complete process of operation from system start up until an alarm is generated by the system. This is done through the following stages of operation:

Stage 1 - Start: The system starts up as the power is connected and the hardware components comprising the Arduino Nano, IR sensor, Raspberry Pi, camera module, and GSM modules become active and initialize the system.

Stage 2 - Monitoring Rail Fitting for Faults using IR Sensor: An IR sensor beam is sent towards the rail fitting by the sensor at specific intervals. The system continues to read the analog data continuously.

Stage 3 - Decision Point: IR Sensor Output Stable and Greater than 500 Units?: When the IR sensor reading is stable and greater than 500 units, the fitting is safe. The system takes the Yes path and continues monitoring.

Stage 4 – Debounce Timer Initialization: If the level of the IR signal falls below the set threshold or fluctuates irregularly, a debounce timer ranging from 200 milliseconds to 500 milliseconds is initiated to distinguish between actual faults and temporary interferences caused by vibration due to passing trains or debris.

Stage 5 – Decision: Persisting Anomaly Within Set Time? If the signal returns to normal levels before the debounce time ends, then the NO path is followed and the system proceeds back to Stage 3. If, on the other hand, the signal remains below the threshold or continues to fluctuate during the whole debounce time, the YES path confirms that the anomaly is an actual fault.

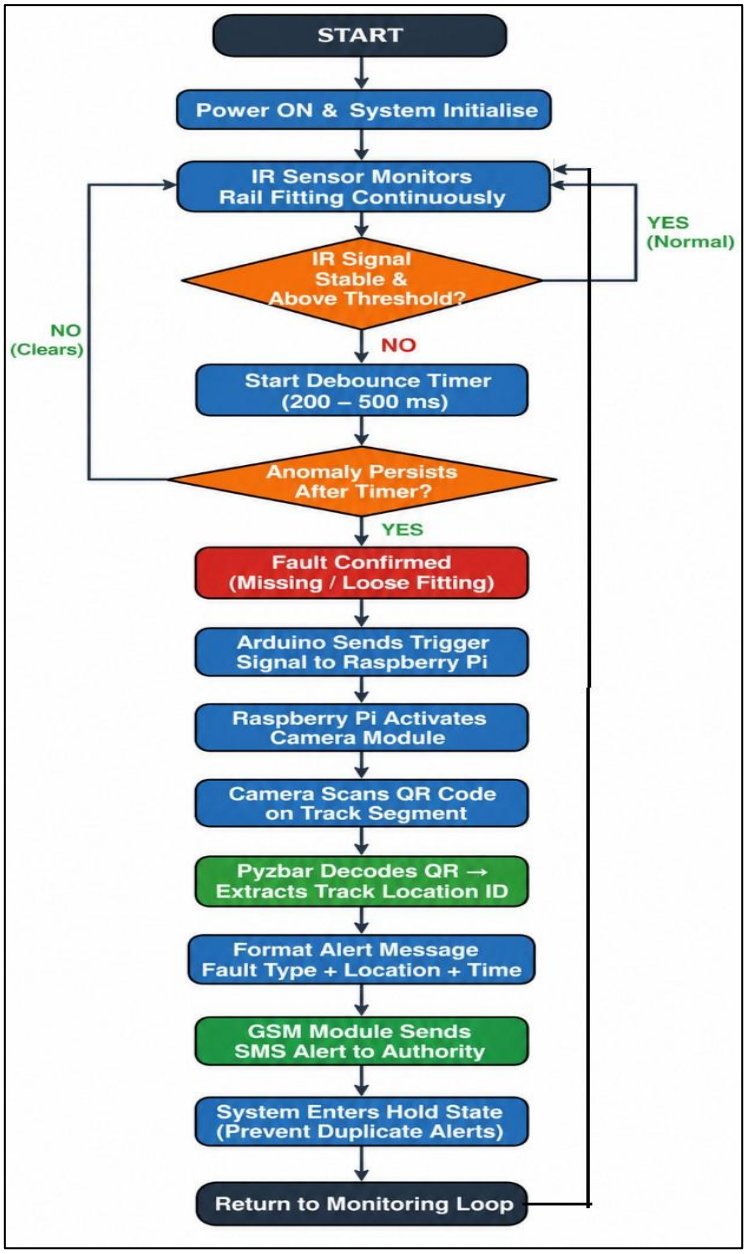

Figure 1. System Workflow of the IoT-Based Railway Rail Fitting Detection

Stage 6 – Fault Confirmation: The Arduino Nano identifies the condition as an actual fault and categorizes it as a missing or loose coupling according to whether the signal is continuously low or if its fluctuations exceed the instability threshold, respectively.

Stage 7 – Digital Signal Sent by Arduino to Raspberry Pi: A digital signal at the HIGH level is then sent from the Arduino Nano to the Raspberry Pi using one of its designated GPIO pins.

Stage 8 - Activate the Camera Module: Once the signal is received by the Raspberry Pi, it starts the process of activating the camera module attached to it.

Stage 9 - Scan the QR Code on the Track Section: After activation, the camera scans the QR code that is pasted on the rail section near the fitting, which contains information about the rail section along with its fitting number.

Stage 10 - Decode the QR Code Using Pyzbar: Once the code is scanned using the camera, the Raspberry Pi decodes the QR image and identifies the location string of the track section along with the fitting.

Stage 11 - Compose the Alert Message: Finally, the Raspberry Pi composes an alert message containing details related to the malfunction and the location string obtained after decoding the QR code.

Stage 12 – SMS Notification by GSM Module to Authorities: The message is then sent from the Raspberry Pi to the GSM module through serial communication and subsequently sent as an SMS notification to the railway authorities who are stored in the system.

Stage 13 – Hold Period Initiated by System: Following the transmission of the message, the system enters into a short holding period in order to avoid generating multiple notifications for a single fault detection.

Stage 14 – Reverting to Infrared Monitoring Cycle: Once the holding period is over, the system will continue to monitor the infrared signals in an effort to identify new faults in the nearby fitting position.

4. Results and Discussion

Implementation of the proposed prototype (Figure 2) was done on a model railway track of 3 meters having six fitting locations. Implementation details involved the use of the following hardware components – Arduino Nano, IR Sensor FC-51, Raspberry Pi 3B+, Pi Camera v2 and SIM800L GSM module – all connected to a regulated power supply source of 12V/2A. IR sensor was used with a sampling rate of 50 milliseconds. Environmental conditions varied during testing of this system. Temperature values for experiments were between 22°C and 28°C while relative humidity remained in the range of 45% to 65%. Lighting conditions taken into consideration were – normal lighting (between 500 to 800 lux), dim lighting (between 20 to 50 lux) and darkness (0 lux). Fitting positions were marked with QR codes of size 25mm x 25mm at 10cm distance.

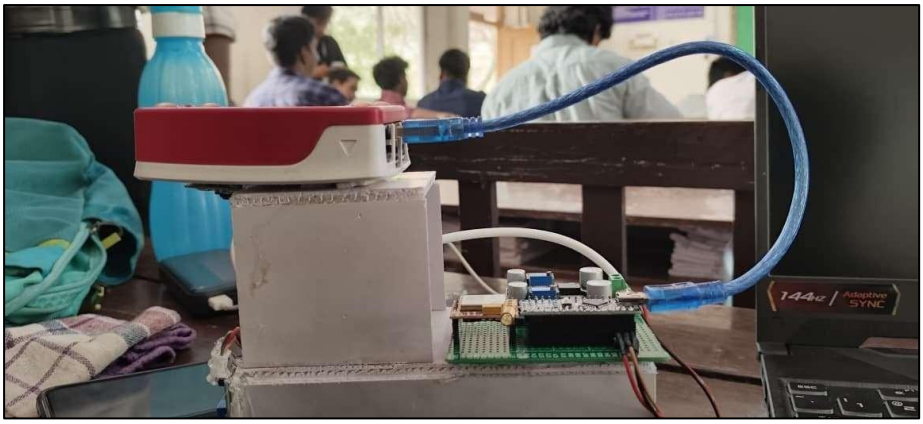

Figure 2. Prototype Implementation of the Proposed System

The system was tested using eight test cases (as in Table 2) that covered different rail fitting states such as present, absent, loose, and dirty. The readings of the infrared sensor showed a difference between these states. Stable readings of high value implied presence of fitting, whereas low readings suggested absence. Loose rail fittings generated fluctuating readings owing to their unstable position. The debounce circuit efficiently filtered out temporary fluctuations such as vibration and noise. The fitting state could be accurately detected without any erroneous identification for all test cases. The QR code could be decoded when there was actually a fault. The message alerts were sent through the GSM module without fail.

Table 2. Summary of Experimental Evaluation Under Multiple Rail Fitting Conditions

Test Case	Fitting Status	IR Output	QR Decoded	Alert Sent
TC-01	Present	HIGH (Stable)	Not Triggered	No (Normal)
TC-02	Missing	LOW (Drop)	Yes	Yes
TC-03	Loose	Fluctuating	Yes	Yes
TC-04	Present (Dirty)	HIGH (Stable)	Not Triggered	No (Normal)
TC-05	Missing	LOW (Drop)	Yes	Yes
TC-06	Loose	Fluctuating	Yes	Yes
TC-07	Present	HIGH (Stable)	Not Triggered	No (Normal)
TC-08	Missing (Night)	LOW (Drop)	Yes	Yes

The quantitative analysis of sensor data (Figure 3) clearly demonstrates the differences between the states of fitting. Perfectly fitted sensors showed high levels of readings with very small variance, while the absence of fitting generated low and steady readings that were far

from the threshold value. Unsteady readings caused by the vibrations of sensors with looser fitting generated higher variances under simulated loads. This variance-oriented approach was highly accurate at determining loose fittings versus other stable states; however, when little displacement occurred, variance sometimes reached the threshold level without exceeding it, slightly lowering detection performance.

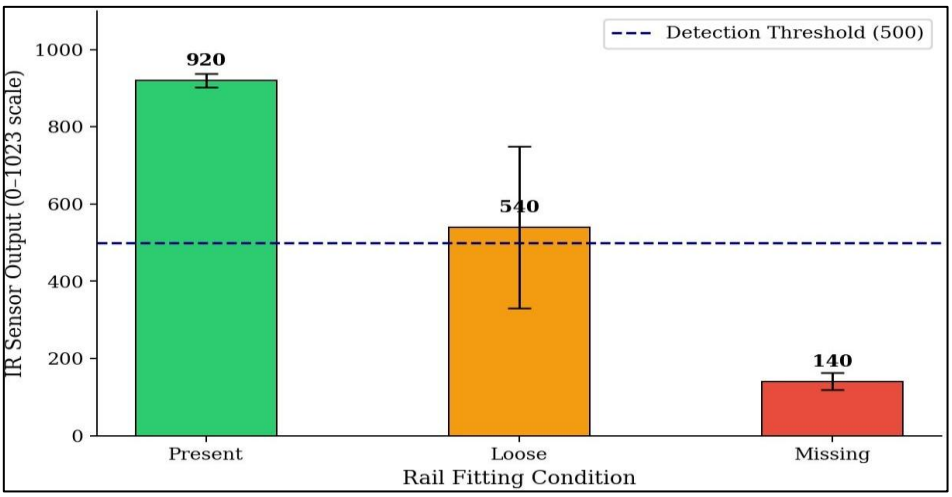

Figure 3. Average Infrared Sensor Output Under Different Rail Fitting Conditions

The response time of the system (in Figure 4) was between 8.2 seconds and 14.6 seconds in 30 iterations, with a median response time of 10.8 seconds. About 80% of the messages were sent in 12 seconds. The response time depends mainly on the GSM network delay, whereas the total time of processing the IR sensor signal, the QR code, and the message was quite small (about 2–3 seconds). It proves that the system works in real-time.

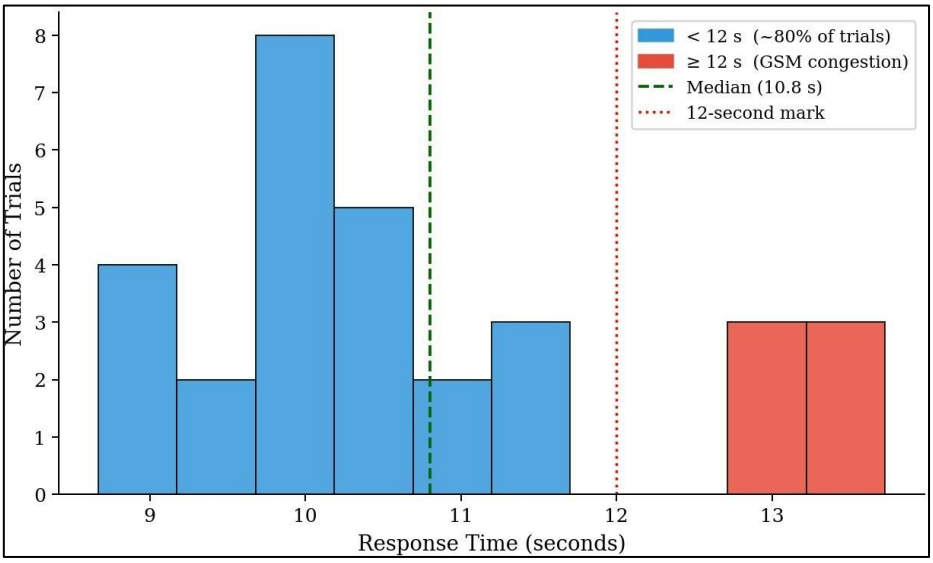

Figure 4. End-to-End System Response Time Distribution Measured from Fault Detection to GSM Alert Delivery Across Multiple Trials

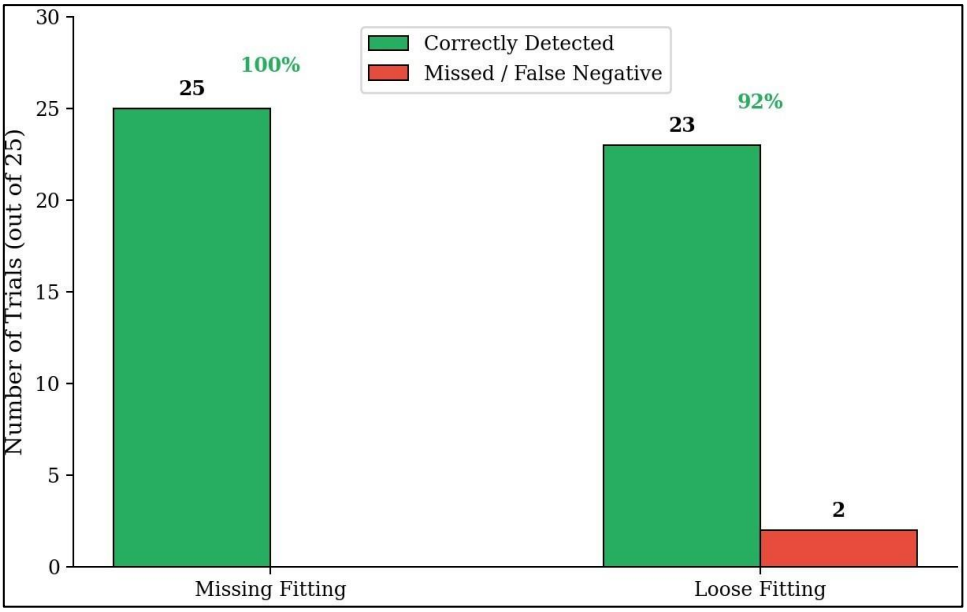

Figure 5. Fault Detection Accuracy for Missing and Loose Rail Fittings Based on Controlled Experimental Trials

As shown in Figure 5, the system yielded 100% accuracy for missing fittings and 92% accuracy for loose fittings during 50 experiments conducted under controlled conditions. The lower accuracy of loose fitting detection is due to marginal displacement values that are close to the sensitivity limit of the variance algorithm. However, this study presents a highly reliable system with regard to its simplicity, efficiency, and performance. In comparison with the vision- or computationally-intensive algorithms, the infrared sensing technique allows continuous monitoring with minimal resource consumption. Additionally, QR code localization makes GPS module unnecessary and hence simplifies the design and decreases costs. Overall, the findings confirm the applicability of the suggested system to monitor the safety of railways in real time. However, it can be further enhanced by employing multiple sensors in the future.

5. Conclusion

In this work, an IoT-based rail fitting detection system has been proposed based on the use of infrared sensor technology, QR code technology and GSM communication protocol that could accurately detect missing or loose rail fittings. The system can operate continuously without any need for human intervention with relatively inexpensive and easily available hardware. Experimental analysis conducted on eight scenarios with fifty controlled trials has shown promising performance metrics of the system. Detection accuracy reached 100% for

identifying the presence of missing rail fittings and 92% accuracy rate for detecting loose rail fittings. However, slight difficulties arose in detecting marginal displacement of rail fittings due to the sensitivity threshold of the infrared sensor employed. QR code technology worked effectively under various light conditions, while GSM technology enabled the prompt transmission of information about any defects in a reasonable time period. Nevertheless, the limitation of the infrared technology is the lack of ability to recognize early stage of loosening process. Therefore, future research will concentrate on incorporating the use of vibration sensors, such as piezoelectric and MEMS accelerometers. Furthermore, GPS system can be integrated in the system as well as cloud and mobile applications.

References

[1] De Bruin, Tim, Kim Verbert, and Robert Babuška. "Railway Track Circuit Fault Diagnosis Using Recurrent Neural Networks." IEEE transactions on neural networks and learning systems 28, no. 3 (2016): 523-533.

[2] Oukhellou, Latifa, Alexandra Debiolles, Thierry Denœux, and Patrice Aknin. "Fault Diagnosis in Railway Track Circuits Using Dempster–Shafer Classifier Fusion." Engineering Applications of Artificial Intelligence 23, no. 1 (2010): 117-128.

[3] García Márquez, Fausto Pedro, Clive Roberts, and Andrew M. Tobias. "Railway Point Mechanisms: Condition Monitoring and Fault Detection." Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit 224, no. 1 (2010): 35-44.

[4] Vileiniskis, Marius, Rasa Remenyte-Prescott, and Dovile Rama. "A Fault Detection Method for Railway Point Systems." Proceedings of the Institution of Mechanical Engineers, Part F: Journal of Rail and Rapid Transit 230, no. 3 (2016): 852-865.

[5] Karakose, Mehmet, Orhan Yaman, Mehmet Baygin, Kagan Murat, and Erhan Akin. "A New Computer Vision Based Method for Rail Track Detection and Fault Diagnosis in Railways." International Journal of Mechanical Engineering and Robotics Research 6, no. 1 (2017): 22-17.

[6] James, Ashish, Wang Jie, Yang Xulei, Ye Chenghao, Nguyen Bao Ngan, Lou Yuxin, Su Yi, Vijay Chandrasekhar, and Zeng Zeng. "Tracknet-a Deep Learning Based Fault

Detection for Railway Track Inspection." In 2018 International Conference on Intelligent Rail Transportation (ICIRT), IEEE, 2018, 1-5.

[7] Shafique, Rahman, Hafeez-Ur-Rehman Siddiqui, Furqan Rustam, Saleem Ullah, Muhammad Abubakar Siddique, Ernesto Lee, Imran Ashraf, and Sandra Dudley. "A Novel Approach to Railway Track Faults Detection Using Acoustic Analysis." Sensors 21, no. 18 (2021): 6221.

[8] Jo, Ohyun, Yong-Kyu Kim, and Juyeop Kim. "Internet of Things for Smart Railway: Feasibility and Applications." IEEE Internet of Things Journal 5, no. 2 (2017): 482-490.

[9] Fraga-Lamas, Paula, Tiago M. Fernández-Caramés, and Luis Castedo. "Towards the Internet of Smart Trains: A Review on Industrial IoT-Connected Railways." Sensors 17, no. 6 (2017): 1457.

[10] Shah, Ali Akbar, Naveed Anwar Bhatti, Kapal Dev, and Bhawani Shankar Chowdhry. "MUHAFIZ: IoT-Based Track Recording Vehicle for the Damage Analysis of the Railway Track." IEEE Internet of Things Journal 8, no. 11 (2021): 9397-9406.

[11] Ghosh, Chayan, Anshul Verma, and Pradeepika Verma. "Real Time Fault Detection in Railway Tracks Using Fast Fourier Transformation and Discrete Wavelet Transformation." International Journal of Information Technology 14, no. 1 (2022): 31-40.

[12] Siddiqui, Hafeez Ur Rehman, Adil Ali Saleem, Muhammad Amjad Raza, Kainat Zafar, Kashif Munir, and Sandra Dudley. "IoT Based Railway Track Faults Detection and Localization Using Acoustic Analysis." IEEE Access 10 (2022): 106520-106533.

[13] Chellaswamy, C., T. S. Geetha, A. Vanathi, and K. Venkatachalam. "An IoT Based Rail Track Condition Monitoring and Derailment Prevention System." International Journal of RF Technologies 11, no. 2 (2020): 81-107.

[14] Iyer, Srikrishna, T. Velmurugan, Amir Hossein Gandomi, V. Noor Mohammed, K. Saravanan, and S. Nandakumar. "Structural Health Monitoring of Railway Tracks Using IoT-Based Multi-Robot System." Neural Computing and Applications 33, no. 11 (2021): 5897-5915.

[15] Minguell, Marta Garcia, and Ravi Pandit. "TrackSafe: A Comparative Study of Data-Driven Techniques for Automated Railway Track Fault Detection Using Image Datasets." Engineering Applications of Artificial Intelligence 125 (2023): 106622.

[16] Padhi, Shridhar, Mansi Subhedar, Saikiran Behra, and Tejesh Patil. "IoT Based Condition Monitoring for Railway Track Fault Detection in Smart Cities." IETE Journal of Research 69, no. 9 (2023): 5794-5803.

[17] Saritas, M. Mustafa, Yavuz Selim Taspinar, Ilkay Cinar, and Murat Koklu. "Railway Track Fault Detection with ResNet Deep Learning Models." In 2023 International Conference on Intelligent Systems and New Applications (ICISNA'23). 2023.

[18] Krishna, B. R., D. Seshendra, G. Raja, T. Sudharshan, and K. Srikanth. "Railway Track Fault Detection System by Using IR Sensors and Bluetooth Technology." Asian Journal of Applied Science and Technology (AJAST) 1, no. 6 (2017): 82-84.