IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511144

IoT-Based Real-Time Accident Detection and Emergency Alert System

Ashish Takankar[1], Vaishnavi Shukla[2], Manish Nagare[3], Pragati Aiwale[4]

Department of Computer Science and Engineering, SKNSCOE Korti, Pandharpur, Maharashtra, India[1-4]

Abstract: Road crashes continue to be a major safety problem worldwide, and many deaths can be linked to the time lost before help reaches the victims. This paper presents an IoT-Based Real-Time Accident Detection and Emergency Alert System that automatically identifies unusual vehicle events and sends out an emergency alert without any manual action. The design brings together an ESP32 microcontroller, an MPU6050 six-axis accelerometer/gyroscope, a supplementary vibration sensor and a NEO-6M GPS receiver. Unlike conventional systems that rely solely on vibration sensors or SMS messages, the proposed unit validates events through sensor fusion, connects to the cloud via the Blynk platform and triggers emergency actions through a relay. The ESP32 constantly examines motion data to recognize severe events such as collisions, rollovers and sudden stops. When an accident is confirmed, the unit obtains the vehicle's coordinates and forwards the event status and position to preset emergency contacts, while a buzzer, LED indicators, a 16×2 LCD and a relay give local feedback. The approach seeks to shorten the gap between the occurrence of an accident and the delivery of a notification, and the prototype is inexpensive, compact and appropriate for vehicle safety use. It is meant to serve as an early-warning and notification aid; the final emergency response and the verification of the accident remain the responsibility of people and authorized services.

Keywords: Internet of Things, Accident Detection, ESP32, GPS, Emergency Alert, Sensors, Real-Time Monitoring.

I. INTRODUCTION

A. Background

Road traffic accidents pose a serious public-safety problem, since the interval between a crash and the arrival of help has a strong bearing on how severe the outcome turns out to be. Often an injured person cannot place a call or describe where the crash occurred, and accidents on deserted roads or during the night can go unnoticed for a long time. A system that automatically senses a probable accident and sends out its location can therefore enable quicker notification and help rescue services reach the site sooner.

The Internet of Things (IoT) allows physical devices to sense their surroundings, process the collected data and exchange information over a network. In the vehicle-safety domain, IoT technology can link motion sensors, embedded controllers, GPS receivers and communication modules into an automatic accident-notification unit that works without any action from the driver.

B. Evolution of Accident Detection Systems

Accident detection has been studied for more than ten years, and the underlying technologies have changed substantially. Early devices mostly depended on vibration or shock sensors to identify impacts. These were cheap and easy to fit, but they often raised false alarms, because everyday road disturbances such as bumps, potholes and engine vibration were misread as collisions.

Subsequent work turned to accelerometer-based GSM systems to improve accuracy. These sent an SMS whenever the measured acceleration went beyond a preset limit. Detection was more reliable than with vibration sensors, but such systems still relied on one measured quantity, found it hard to separate crash types such as rollovers, and depended on SMS delivery, which can be slow and gives the recipient only limited real-time visibility.

Recent IoT-based designs add wireless connectivity and cloud services, yet many of them still employ a single kind of sensor and automate only part of the process. The system proposed here combines sensor fusion, cloud connectivity through the Blynk platform and relay-triggered emergency actions in a single compact unit. Table I outlines how the proposed system differs from earlier approaches.

C. Contributions of This Work

The principal contributions of this paper are as follows. First, an inexpensive ESP32-based architecture is described that fuses accelerometer, gyroscope and impact data to identify collisions, rollovers and sudden stops. Second, a validation stage is introduced so that an alarm is not raised on the basis of one sensor reading alone. Third, GPS positioning is combined with cloud connectivity, making the accident status and coordinates available to emergency contacts and on a

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511144

monitoring dashboard. Finally, relay-based actuation, a manual SOS switch, a 16×2 LCD and a reset key are included to make the unit practical to use and to let it return to normal operation after an incident.

TABLE I.
COMPARATIVE OVERVIEW OF EXISTING AND PROPOSED SYSTEMS

Feature	Vibration only	Accel. + GSM	Existing IoT	Proposed system
Primary sensor	Vibration sensor	Accelerometer	Accelerometer / single-type sensor	Sensor fusion (MPU6050 + additional sensors)
Communication method	Local alarm	SMS	SMS / partial IoT	Wi-Fi cloud + automated emergency activation
Real-time monitoring	Basic	Basic	Partial	Fully enabled
False alarm rate	High	Medium	Medium	Low, owing to sensor fusion
Crash condition handling	Single type	Single type	Limited	Multidirectional impact + rollover detection
Human input required	Yes	Yes	Sometimes	No (fully automatic)
Approx. response time	10–60 s (local alarm only)	30–180 s (SMS delays)	10–40 s (unstable IoT)	1–5 s (real-time cloud alert)
Advancement level	Basic	Improved	Moderate	High (advanced automation)

The response times are indicative figures for each class of system and do not represent measured results from this prototype.

II. PROBLEM STATEMENT

Traditional accident reporting relies largely on the driver, passengers or bystanders to notice the crash and call for emergency help. Such manual reporting can cause delays, supply incomplete location details, or fail altogether when the occupants are unable to respond. On rural or lightly used roads, a long time may pass before a passer-by notices a crash, and even a caller who gets through may be unable to explain exactly where the vehicle is.

Current low-cost options solve this problem only in part. Vibration-based units trigger many false alarms, while SMS-based accelerometer units suffer from delayed messages and offer no continuous monitoring. The proposed system fills this gap by combining sensing, processing, location tracking and wireless notification in a single IoT-enabled unit that needs no human input to send an alert.

III. OBJECTIVES

- To sense sudden impact or abnormal motion that may point to a vehicle accident.
- To obtain the geographic coordinates of the vehicle through GPS.
- To generate an emergency alert automatically once an accident is suspected.
- To send the accident status and location details to predefined contacts.
- To offer a low-cost, compact prototype suitable for real-time vehicle monitoring.
- To limit false alarms by validating events through sensor fusion instead of a single sensor reading.
- To provide local audible, visual and relay-based feedback together with a manual SOS option for passengers.

IV. PROPOSED SYSTEM

A. System Architecture

The proposed architecture comprises sensing, processing, location, communication and alert elements. A six-axis MPU6050 accelerometer/gyroscope, along with an impact-sensing unit, continuously monitors the motion of the vehicle. The ESP32 microcontroller takes in the sensor readings, interprets them and applies threshold-based event detection with validation. Once a possible accident is recognized, the NEO-6M GPS module provides the latitude and longitude. The

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511144

communication interface then sends an emergency message to the configured recipients and uploads the event data to the Blynk cloud platform over Wi-Fi.

The ESP32 serves as the main controller and handles data interpretation, cloud transmission and relay control. The MPU6050 gives continuous information about motion dynamics, and the NEO-6M receiver supplies real-time position data. A 16×2 LCD presents system status and diagnostic details during operation, including sensor status, GPS status, accident alerts and live coordinates. A buzzer and a relay deliver immediate audible and physical feedback in an emergency. A regulated power supply lets the unit run safely from the vehicle battery, and a reset key returns the system to normal operation after a critical event. Fig. 1 shows the overall conceptual flow of the system.

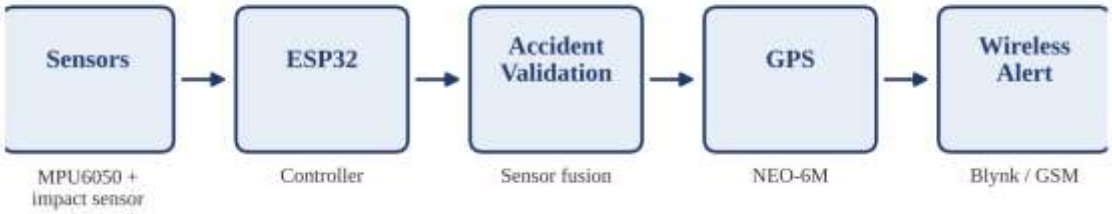


Fig. 1. Conceptual flow of the proposed accident detection and emergency alert system.

B. Major Components

The prototype is built from an ESP32 development board, an MPU6050 six-axis accelerometer/gyroscope, a vibration or impact sensor, a GPS module such as the NEO-6M, a GSM module such as the SIM800L for use where cellular messaging is needed, a buzzer, LED indicators, a 16×2 LCD, a relay module and a regulated power supply. Components may be substituted depending on the communication network and the needs of the prototype. Table II gives the role of each major component.

TABLE II.
MAJOR SYSTEM COMPONENTS

Component	Function
ESP32	Handles sensor processing, cloud communication and overall system control
MPU6050 accelerometer/gyroscope	Measures acceleration and changes in orientation
Impact/vibration sensor	Gives an additional indication of an accident event
GPS module (NEO-6M)	Provides latitude, longitude and timestamp
GSM/wireless module	Sends the emergency notification
16×2 LCD	Shows system status and diagnostic details
Buzzer/LED	Gives a local accident alert indication
Relay module	Switches on warning lights, the buzzer or engine cut-off
Power supply	Supplies regulated power from the vehicle battery

C. Hardware Requirements and Specifications

Table III lists the hardware used in the prototype along with its specification and purpose. The ESP32 was chosen because its dual-core design and built-in Wi-Fi and Bluetooth make it a good fit for cloud-based accident detection. The MPU6050 combines a 3-axis accelerometer and a 3-axis gyroscope in one package, so deceleration, impact force, tilt variation and rollover movement can all be monitored with a single device. The SW-420 module gives an independent impact indication that is used during validation.

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511144

TABLE III.
HARDWARE REQUIREMENTS AND SPECIFICATIONS

Component	Specification	Purpose
ESP32 board	32-bit, dual-core, Wi-Fi and Bluetooth enabled	Processes sensor data and controls the system
MPU6050	3-axis accelerometer + 3-axis gyroscope	Detects sudden acceleration, impact and abnormal vehicle movement
SW-420 sensor	Vibration/impact detection module	Detects sudden vibration during an accident
NEO-6M GPS	GPS receiver	Provides the accident location coordinates
SIM800L GSM	GSM/GPRS communication	Sends emergency alerts and accident information
Li-ion battery	Rechargeable battery	Supplies power to the system
TP4056 module	Li-ion battery charging module	Charges and protects the battery
PCB and wires	Circuit board and connectors	Connects and holds all hardware components together
Buzzer	5 V/3.3 V buzzer	Gives a local accident/emergency indication

D. Software Requirements

The firmware is written and debugged in the Arduino IDE. ESP32 software support reads and processes the sensor data and implements the accident-detection logic, while a GPS software interface retrieves and processes the location of the vehicle. GSM communication software sends the emergency alerts and accident details where cellular messaging is used. A cloud platform stores the accident data and enables real-time monitoring, and the Blynk IoT platform shows the accident status, alerts and location to the user. Table IV lists the software components and their roles.

TABLE IV.
SOFTWARE REQUIREMENTS

Software	Role
Arduino IDE	Used to program and debug the ESP32 firmware
ESP32 software support	Processes sensor data and runs the accident-detection system
GPS software interface	Retrieves and processes the accident location of the vehicle
GSM communication software	Sends emergency alerts and accident details
Firebase / cloud platform	Stores accident data and supports real-time monitoring
Blynk IoT platform	Shows accident status, alerts and location

E. Accident Detection Logic

The controller continuously samples motion-related sensor values. From the three accelerometer axes, the resultant acceleration is calculated as

$$a_{total} = \sqrt{(a_x^2 + a_y^2 + a_z^2)} \quad (1)$$

where a_x, a_y and a_z are the acceleration components along the three axes. A sudden collision or an abrupt stop shows up as a sharp rise or drop in a_{total}. Using the gravity components, the tilt of the vehicle relative to the vertical is given by

$$\theta = \text{atan2}(\sqrt{(a_x^2 + a_y^2)}, a_z) \quad (2)$$

and the magnitude of the angular rate measured by the gyroscope is

$$\omega_{total} = \sqrt{(g_x^2 + g_y^2 + g_z^2)} \quad (3)$$

where g_x, g_y and g_z are the gyroscope readings. A suspected accident is declared when the measured acceleration or impact signal exceeds a configured threshold and the event satisfies the validation logic, which can be written as



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026
DOI: 10.17148/IJARCCE.2026.1511144

$$(a_{total} \geq T_a) \text{ AND } [(\omega_{total} \geq T_\omega) \text{ OR } (\theta \geq T_\theta) \text{ OR } \textit{impact flag}] \text{ for } \Delta t \qquad (4)$$

where T_a, T_ω and T_θ are the calibrated acceleration, angular-rate and tilt thresholds, and Δt is a short validation period. The threshold values are set by calibration for the particular vehicle and mounting position. Using several indicators together is the basis of the sensor-fusion approach: a pothole or a hard brake may raise the acceleration by itself, whereas a genuine collision or rollover comes with an impact indication, a rapid rotation or a lasting change in orientation. Table V summarizes the principal events considered and their indicators.

TABLE V.
ACCIDENT EVENTS AND SENSING INDICATORS

Event	Main indicators	Sensing source
Frontal/side collision	Sharp rise in a_{total} along one or more axes, together with an impact signal	MPU6050 accelerometer, SW-420
Rollover	Large tilt θ that persists after the event, high angular rate	MPU6050 gyroscope and accelerometer
Abrupt stop	Sudden deceleration followed by loss of motion	MPU6050 accelerometer, GPS speed
Pothole/rough road	Brief vibration without sustained tilt or rotation	Rejected by validation

A short validation period is applied to reduce false alerts caused by ordinary vibration or sudden braking. After confirmation, the controller requests the GPS coordinates and prepares the emergency message.

F. GPS Location Acquisition

After an accident has been confirmed, the NEO-6M GPS module obtains the latitude, longitude and time of the crash. The module passes the position data to the ESP32 through a serial interface, and the controller extracts the coordinates from the sentences it receives. A GPS fix may be missing for a short while, for instance after a cold start or where satellite visibility is poor, so the controller falls back on the most recent available coordinates and repeats the acquisition attempt if no valid fix exists. The coordinates are then added to the emergency message, allowing the recipients to open the location directly on a map.

G. Cloud Connectivity and Emergency Response

Once validation and location acquisition are complete, the accident details and coordinates are sent over Wi-Fi to the Blynk cloud platform, which promptly notifies the designated emergency contacts and refreshes the dashboard. Event records may also be saved on a Firebase/cloud platform to support monitoring and later analysis. Where only cellular coverage is available, the SIM800L GSM module can send the same information as a message.

Depending on implementation choices, the relay module can switch on physical safety features such as buzzers, warning lights or engine shutdown. A manual SOS switch is also included so that passengers can call for help when an accident has happened but was not detected automatically. The 16×2 LCD shows the system status and live coordinates, and the reset key returns the unit to normal operation once the incident has been dealt with.

H. Design Considerations

Several practical factors shape the design. The first is sensor mounting: the MPU6050 must be fixed firmly and aligned with the vehicle axes, since a loosely attached sensor records its own movement instead of that of the vehicle. The second is power: the unit draws energy from the vehicle battery through a regulated supply so that voltage fluctuations while the engine starts do not reset the controller, and a rechargeable Li-ion cell with a TP4056 charging module can keep the unit running if the main supply is cut by the crash. The third is communication redundancy: the Wi-Fi link to the Blynk platform gives real-time cloud alerts when a network is available, while the SIM800L GSM module offers a cellular fallback for messaging where Wi-Fi coverage is missing. The fourth is local feedback: because network delivery cannot be guaranteed, the buzzer, LED indicators, LCD and relay work independently of the network, so that people close to the vehicle are still warned.

V. METHODOLOGY

An ESP32 microcontroller forms the computing core of the proposed system; its dual-core design and integrated Wi-Fi make it well suited to cloud-based accident detection. Motion analysis relies on the MPU6050, which combines an

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511144

accelerometer and a gyroscope to track rapid deceleration, impact force, tilt changes and rollover movement. The NEO-6M GPS module is included to supply accurate position coordinates during emergencies.

When the ESP32 finds motion characteristics beyond the calibrated safety thresholds, it starts a three-phase emergency procedure. In the first phase, sensor fusion analysis checks that an accident has actually taken place. In the second phase, the GPS module determines the position of the vehicle. In the third phase, the accident details and the coordinates are sent to the Blynk cloud platform, which alerts the designated emergency contacts and updates the dashboard. The complete procedure is as follows.

- Initialize the ESP32, motion sensors, GPS, communication module and alert indicators.
- Read the acceleration, orientation and impact values continuously.
- Compare the readings against the configured accident-detection thresholds.
- Confirm the event over a short validation interval to limit false triggering.
- Obtain the latest available GPS coordinates.
- Compose an emergency message containing the accident status and location.
- Send the notification to the predefined emergency contacts and the cloud dashboard.
- Switch on the local indicators and, if configured, the relay output.
- Save or display the event information for monitoring and subsequent analysis.

VI. ALGORITHM

Let the system state at time t be described by the sensor tuple $S(t) = \langle A(t), G(t), I(t) \rangle$, where A(t) and G(t) are the 3-axis acceleration and gyroscope vectors and I(t) is the impact signal. The system operates according to the following steps:

1. **Initialization:** Initialize all hardware modules at t = 0. Set the accident flag $F_{acc} \leftarrow$ False and the validation counter C $\leftarrow$ 0.

2. **Data Acquisition:** Read the sensor values continuously at interval Δt: A(t), G(t), I(t) ← read_sensors().

3. **Feature Extraction and Threshold Check:** Compute the resultant acceleration $a_{total}(t) = \sqrt{(a_x^2 + a_y^2 + a_z^2)}$ and the angular rate $\omega(t) = \sqrt{(g_x^2 + g_y^2 + g_z^2)}$. A preliminary event P(t) is flagged when any parameter crosses its threshold (T_a, T_ω, T_I):

$$P(t) = 1, \text{ if } (a_{total} \geq T_a) \vee (\omega \geq T_\omega) \vee (I \geq T_I)$$

$$P(t) = 0, \text{ otherwise}$$

4. **Event Validation (Sensor Fusion):** To eliminate false alarms, the event is checked over a confirmation interval. The counter C increases only while the event persists:

$$C(t) = C(t - \Delta t) + 1, \text{ if } P(t) = 1$$

$$C(t) = 0, \text{ if } P(t) = 0$$

   The accident is confirmed only when $C(t) \geq N_{val}$ (the minimum validation count), that is, F_{acc} = True if $C(t) \geq N_{val}$.

5. **Alert Generation and Transmission:** When F_{acc} = True, obtain the GPS coordinates (Lat, Lon) and build the message M_{alert} = {Status, Timestamp, Lat, Lon, Request}. Send M_{alert} over the available network $\mathcal{N}$ and activate the local indicators:

$$Transmit(M_{alert}, \mathcal{N}) \rightarrow Contacts, Buzzer \leftarrow ON, Relay \leftarrow ON$$

6. **Reset:** Once the event has been handled, set $F_{acc} \leftarrow$ False and C(t) ← 0, then return to Step 2.

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511144

VII. IMPLEMENTATION AND WORKING

A. Hardware Integration

The sensors, GPS receiver, display, buzzer and relay are wired to the ESP32 and mounted on a PCB. The unit takes its power from the vehicle battery through a regulated supply so that the voltage reaching the ESP32 and the peripheral modules stays within safe limits in the vehicle environment. A rechargeable Li-ion battery with a TP4056 charging module may be used to power the system and to protect the battery while it charges. The MPU6050 should be fixed rigidly to the vehicle body with its axes aligned to those of the vehicle; a loose mounting would introduce vibration that is unrelated to the vehicle's motion.

B. Operation

While the vehicle is in motion, the sensing unit stays active and the ESP32 samples the sensor values at regular intervals. An abrupt abnormal event is assessed by the accident-detection logic instead of depending on a single sensor reading. After the event is confirmed, the GPS coordinates are collected and an emergency message is sent. A typical notification may include an accident warning, the date and time, the latitude and longitude, and a request for help. The recipient can use the location details to take further action. Meanwhile, the LCD shows the system status and the accident alert, and the buzzer sounds to warn people near the vehicle.

C. Recovery After an Event

Once a critical event has been handled, the reset key returns the system to normal operation. This stops the same incident from producing repeated alerts and lets the unit resume monitoring without being powered off. If an alert was raised by mistake, the same key can be used to clear the condition.

D. Alert Message Format

The ESP32 produces the emergency message as soon as the GPS coordinates are available. The message is kept brief so that it can be delivered quickly over either the cloud link or a cellular message. Table VI lists the fields of the notification. The date and time let the recipient judge how recent the alert is, and the coordinates allow the location to be opened straight in a map application.

TABLE VI.
FIELDS OF THE EMERGENCY ALERT MESSAGE

Field	Description
Status	Accident warning (automatic) or SOS request (manual)
Date and time	Timestamp taken from the GPS module
Latitude, longitude	Vehicle coordinates at the moment of the event
Request	Call for assistance sent to the predefined contacts

E. Calibration and Test Procedure

Before use, the thresholds are calibrated for the specific vehicle and mounting position. With the vehicle at rest, the resting sensor values are recorded so that the offset of each axis can be removed and the gravity component is correctly identified. The thresholds are then adjusted through controlled tests, such as ordinary driving over rough surfaces, hard braking, and simulated impact and tilting of the unit on a test bench, so that the settings ignore normal disturbances yet still react to crash-like motion. The influence of the validation period is also studied: a longer period filters out more spurious triggers but delays the alert, so the final value is chosen as a compromise between the two. These tests should be repeated whenever the sensor position or the vehicle type changes.

VIII. RESULTS AND DISCUSSION

The proposed prototype offers an integrated chain from sensing to notification: accident-related sensor event → event validation → GPS location acquisition → emergency message generation → alert transmission. The architecture is appropriate for demonstrating real-time detection and notification with widely available IoT hardware. Table VII lists the expected behavior of the system under the main operating conditions.

 This work is licensed under a Creative Commons Attribution 4.0 International License

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940



International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026

DOI: 10.17148/IJARCCE.2026.1511144

TABLE VII.
EXPECTED SYSTEM OUTPUTS

Condition	System response
Normal vehicle motion	Keep monitoring
Sudden non-validated motion	Keep monitoring after the validation check
Confirmed accident-like event	Obtain the GPS position and trigger the alert
GPS temporarily unavailable	Retry location acquisition / report the location as unavailable
Communication unavailable	Retry transmission and keep the local event indication
Manual SOS pressed	Send the alert with the latest available location

In comparison with vibration-only and SMS-based accelerometer systems, using several sensing indicators is expected to lower the number of false alarms, and combining Wi-Fi cloud connectivity with automatic notification is expected to shorten the time to alert, as indicated in Table I. Including gyroscope data makes it possible to identify rollover situations in addition to frontal and lateral impacts, which a single-axis threshold cannot do reliably.

The time needed to raise an alert is the sum of the sensing and validation interval, the GPS retrieval time and the network transmission time. Among these, the validation interval is chosen by the designer, while the transmission time depends on network quality, so the overall response varies with the operating conditions. The design therefore keeps the post-detection steps to a minimum and carries out continuous monitoring with only simple comparisons, so that extra processing takes place only after a validated event.

Actual detection accuracy depends on sensor placement, threshold calibration, vehicle type, road conditions, GPS availability, communication coverage and the validation algorithm. Practical deployment therefore calls for extensive testing in controlled experiments together with a safety review. The performance figures in this paper are design targets and indicative values for the compared classes of system, and quantitative evaluation under controlled test conditions is left for future work.

IX. ADVANTAGES AND APPLICATIONS

The system can lessen the reliance on manual reporting, supply useful GPS location data, allow remote monitoring and be built from relatively inexpensive embedded hardware. Sensor fusion reduces the chance of false alarms, and the relay output lets the unit act physically, for example by sounding a buzzer, turning on warning lights or shutting down the engine. The manual SOS switch gives passengers another way to ask for help.

The concept can be adapted to cars, buses, commercial vehicles, fleet monitoring and other connected transport applications. In a fleet, the cloud dashboard lets an operator follow the status of several vehicles and send assistance quickly when an alert arrives.

X. LIMITATIONS AND FUTURE SCOPE

Sensor thresholds may need recalibration for different vehicles and road conditions. GPS performance can suffer from buildings, tunnels and poor satellite visibility. Wireless alerts rely on network availability: Wi-Fi-based cloud alerts need a connected network, and cellular alerts need sufficient signal coverage. A single threshold-based method may produce false positives or overlook complex accident patterns.

Future work may include machine-learning models trained on multi-sensor time-series data, cloud dashboards, automatic control-room integration, stronger event verification, and more reliable location and communication. Systematic field tests with various vehicles and road surfaces would also allow the thresholds and the validation period to be tuned from measured data.

Security and privacy also deserve attention. Because the alert carries the location of the vehicle, access to the cloud dashboard and the contact list should be limited to authorized users, and the stored event records should be protected. The relay-based engine shutdown should be enabled only after a careful safety review, as stopping the engine in an unsuitable situation could itself create a hazard.

 This work is licensed under a Creative Commons Attribution 4.0 International License



International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 11, November 2026
DOI: 10.17148/IJARCCE.2026.1511144

XI. CONCLUSION

This paper described an IoT-Based Real-Time Accident Detection and Emergency Alert System that brings together sensor-based detection of accident events, GPS location acquisition, embedded processing, cloud connectivity and wireless emergency notification. The proposed architecture, built around the ESP32, the MPU6050 and the NEO-6M GPS module, offers a practical way to automatically report a possible accident and its location. With proper calibration, dependable communication and real-world validation, the system can act as an early-warning and notification layer for vehicle safety applications.

REFERENCES

[1]. Espressif Systems, “ESP32 Series Datasheet,” Espressif Systems, technical documentation.
[2]. TDK InvenSense, “MPU-6000 and MPU-6050 Product Specification,” technical documentation.
[3]. u-blox, “NEO-6 u-blox 6 GPS Modules Data Sheet,” u-blox AG, technical documentation.
[4]. SIMCom Wireless Solutions, “SIM800 Series AT Command Manual,” technical documentation.
[5]. M. Swan, “Sensor Mania! The Internet of Things, Wearable Computing, Objective Metrics, and the Quantified Self 2.0,” Journal of Sensor and Actuator Networks, vol. 1, no. 3, pp. 217–253, 2012.
[6]. A. Whitmore, A. Agarwal, and L. Da Xu, “The Internet of Things—A survey of topics and trends,” Information Systems Frontiers, vol. 17, pp. 261–274, 2015.
[7]. J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, “Internet of Things (IoT): A vision, architectural elements, and future directions,” Future Generation Computer Systems, vol. 29, no. 7, pp. 1645–1660, 2013.

 This work is licensed under a Creative Commons Attribution 4.0 International License