Development of a Real-Time Geofencing System for Ground Support Vehicle Monitoring in Airports

Aishwarya K, Sujith K, Pravallika M, Supraja Reddy A, Sathish P
Pavan Kalyan M, Khashyap K
Chaitanya Bharathi Institute of Technology (CBIT), Hyderabad, India – 500075
suprajareddy_ece@cbit.ac.in

Abstract

Ground support vehicles are essential for airport operations, but they come with safety risks due to their movement in high-traffic zones. These risks can lead to collisions, runway incursions, and delays, which may cause injuries and damage to equipment. This paper presents a cost-effective, real-time monitoring system designed to address these risks by using GPS modules for precise vehicle tracking, microcontrollers for data processing, and a server-side application for communication, data storage, and user interaction. The system features a web-based interface that visualizes vehicle locations on a map and offers potential geofencing capabilities. It alerts vehicle drivers as they approach restricted areas, providing notifications when they are within 2 meters of the geofence perimeter. This early warning system allows drivers to adjust their routes and avoid entering restricted zones, thereby enhancing safety. Additionally, air traffic controllers can leverage the system to prevent accidents and improve ground traffic flow. The system demonstrates its feasibility as an affordable and practical solution to enhance airport safety and operational efficiency.

Keywords: Ground support vehicle, Monitoring system, Real-time tracking, Geofencing, Airport safety

Introduction

Efficient and safe aircraft movement within airports is essential, especially in environments shared with various ground support vehicles. To minimize collisions and near-misses, this paper proposes a Ground Support Vehicle Monitoring System using GPS and geofencing technologies. Real-time GPS integration in vehicles provides situational awareness and a global view of ground operations. Geofencing creates digital barriers around aircraft zones, sending alerts when boundaries are breached, allowing air traffic or ground control to respond swiftly. This enhances safety for both aircraft and ground crews by preventing unauthorized access. to restricted areas. The system leverages real-time data for proactive decision-making, ensuring improved coordination and operational efficiency across the airport.

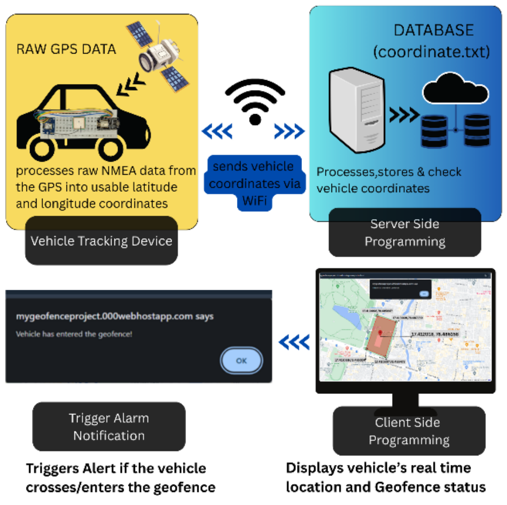
Theoretical Background

Previous research supports vehicle monitoring at airports. Smith et al. (2010) introduced neural networks for predictive maintenance [1], while Blasch et al. (2014) used optical surveillance for airport security [2]. Alomara and Tolujevs (2017) optimized vehicle movements on airfields [3], and Tang (2017) contributed to post-collision alert systems [4]. Hawas et al. (2019) demonstrated inter-vehicular communication for routing [5]. Jospine et al. (2020) applied geofencing to emergency vehicles [6], and Wang et al. (2021) focused on airport equipment tracking and collision prevention [7]. Rosayyan et al. (2021) proposed a GNSS-based geofencing model for emergency vehicles [8]. Stevens and Atkins (2021) defined unmanned aerial geofences [9], while Ding et al. (2021) analyzed historical data for airport ferry time estimations [10]. Wang et al. (2022) presented real-time tracking of logistics trains [11], and Zhou et al. (2023) worked on vehicle routing in ground handling [12]. Lofù et al. (2023) explored UAV tracking and classification [13]. Building on this, the proposed system combines GPS, microcontrollers, and server-side apps for cost-effective real-time tracking, geofencing, and optimized airport vehicle management.

Methodology

The system focuses on real-time tracking, data visualization, and geofence monitoring through efficient acquisition, user-friendly interfaces, and secure data handling. As shown in **Fig. 1**, the process begins with the in-vehicle GPS module determining location and sending it to the ESP8266 microcontroller, which decodes the coordinates and transmits them via Wi-Fi to a server. The server, using PHP, processes the data, checks for geofence violations, and stores it in a database. If a vehicle crosses a defined geofence, an alert is generated. The client-side web application displays the live vehicle position and geofence status on a map, notifying users of any breach.

To detail the operation, **Fig. 2** illustrates the methodology, showing interactions between the onboard GPS unit, server logic, and user interface. The microcontroller converts GPS data (NMEA format) into coordinates, which are then transmitted to the central server. The server logs this into a database, and the website fetches this data for display. A mapping API visualizes both vehicle position and geofences, with alerts triggered upon geofence entry or exit. The following sections will elaborate on each system block, outlining both hardware and software elements used in the prototype


RAW GPS DATA
processes raw NMEA data from the GPS into usable latitude and longitude coordinates
Vehicle Tracking Device
sends vehicle coordinates via WiFi
DATABASE (coordinate.txt)
Processes,stores & check vehicle coordinates
Server Side Programming
Trigger Alarm Notification
Triggers Alert if the vehicle crosses/enters the geofence
Client Side Programming
Displays vehicle's real time location and Geofence status

mygeofenceproject.000webhostapp.com says

Vehicle has entered the geofence!

OK

Fig 1: Pictorial representation of methodology

Vehicle Tracking

The core functionality of the vehicle tracking device relies on two essential components: a GPS module and a microcontroller. The system, which is developed by the combination of the NEO-6M GPS module with GPS ceramic antenna and the ESP8266 microcontroller, as depicted in Figure 3, provided a reliable and efficient foundation for the vehicle tracking device,The GPS module acts as a positioning system, using signals from GPS satellites to pinpoint the vehicle's location. This includes determining latitude, longitude, and often altitude data. In this paper, NEO-6M has been chosen as the GPS module. The main reason for selecting NEO-6M over other GPS modules is that it is commercially available at competitive prices, making it suitable for resource-constrained projects. A minor challenge encountered was the initial acquisition time, particularly in environments with limited sky visibility, but as the prototype is focused on airport applications, this is not a disadvantage. While the cold start time (initial satellite acquisition) averaged around 30 seconds, subsequent reacquisition times after temporary signal loss were generally faster [14]. However, it also offers high-precision GPS positioning data and power efficiency, which is indispensable for real-time tracking of ground support vehicle movements within the airport environment [15].

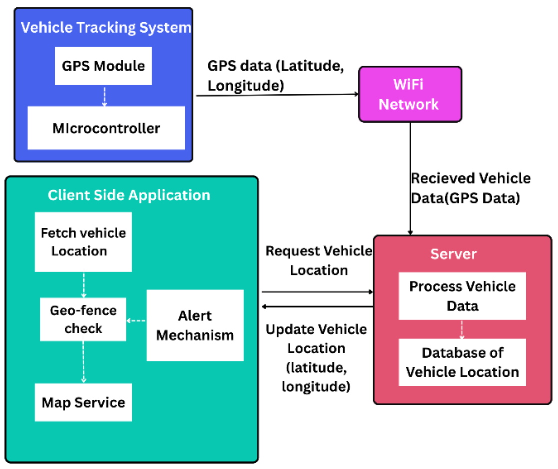


Fig 2: Schematic of the proposed system

The microcontroller acts as the central processing unit, interpreting raw GPS data and making decisions based on it. It also facilitates communication with the GPS module for configuration, data requests, and, ultimately, transmission. Cellular networks, Wi-Fi, or other protocols can be used for this transmission. For this system, considering affordability and the airport environment, the ESP8266 microcontroller was chosen. It's cost-effective, integrates Wi-Fi for wireless data transmission to a central server, and has low power consumption for extended operation on battery power within the GSE tracking units [16].The total power consumption of the system is a crucial factor for real-world deployments, especially for battery-powered applications. In this prototype, the total power consumption of the system is evaluated and estimated to be approximately 412.5 milliwatts.

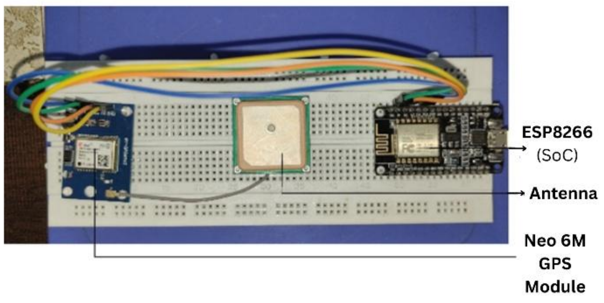


Fig 3: Prototype of the Vehicle tracking system

The emphasis on low power consumption throughout the design process resulted in a device well-suited for mobile deployments with extended battery life. These positive outcomes demonstrate the effectiveness of the chosen hardware components in meeting the objectives.

Server-Side Programming

The server-side functionality is implemented using PHP code, intended to work in conjunction with a web application. In this section working of server-side programming is explained. On the server side initially, a folder is created on a third-party web host to accommodate all the programming data and also files to append information of coordinates as shown in Figure 4 .The server-side programming begins by waiting for incoming requests. It then inspects the HTTP request method to determine the type of operation intended by the client (e.g., GET for retrieving data, POST for sending data). If the request method is POST, signifying an update for GPS coordinates, the server extracts the latitude and longitude data from the request body. This involves accessing specific fields named "lat" and "lng" within the request. To contain information on the coordinates, a file named “coordinates.txt” in the public html directory as shown in Figure 5, hosted by the third-party web hoster, is created on the back end prior to initializing the server-side programming.The server manages data processing and storage by clearing and updating the "coordinates.txt" file with newly received latitude and longitude data, as shown in Figure 6. It sends a response to the client indicating the operation's status. For non-POST requests, an error message is returned. The server-side application efficiently handles communication, data processing, storage, and geofence monitoring. It stores geofence coordinates defined through the user interface and continuously analyzes incoming location data from the vehicle tracking device.

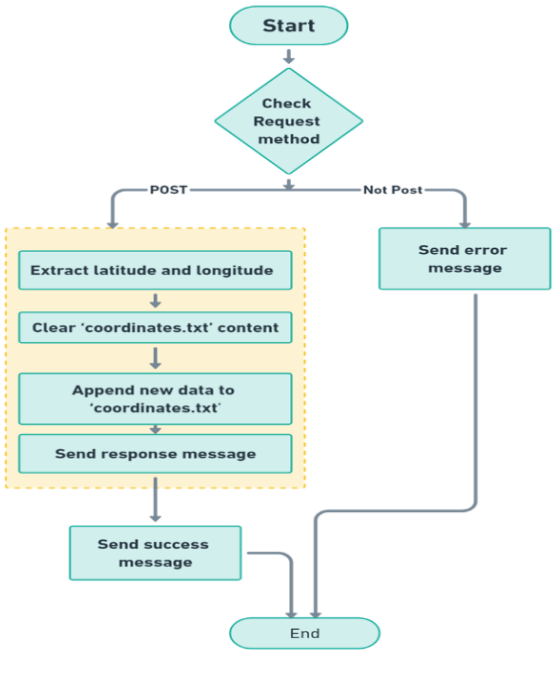


Fig 4: Flowchart of Server-side Mechanism

When a vehicle enters or exits a predefined geofence zone, the server triggers pre-programmed actions, such as sending notifications or logging events. The implementation in PHP enables reliable data management and secure information handling, effectively bridging the gap between the tracking device and user interface for real-time visualization and geofence-based alerts.

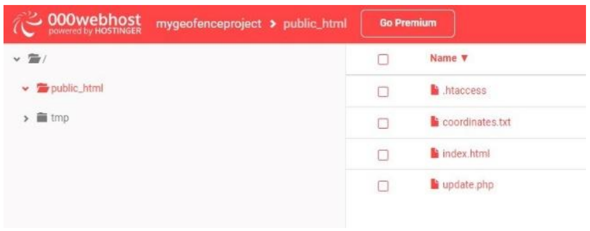
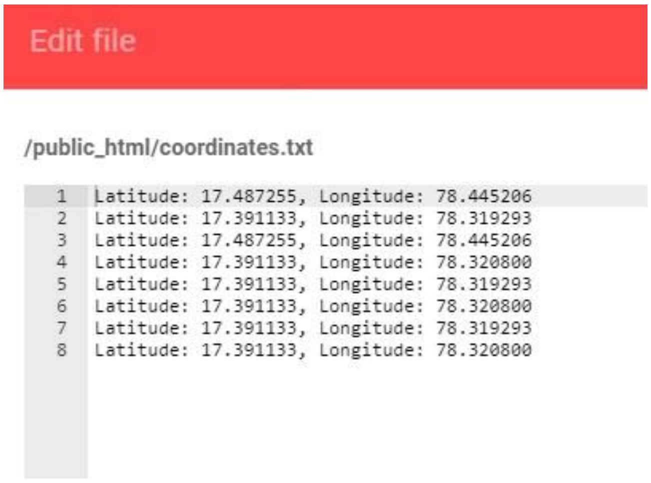


Fig 5: Display of Public HTML Directory



```
Latitude: 17.487255, Longitude: 78.445206
Latitude: 17.391133, Longitude: 78.319293
Latitude: 17.487255, Longitude: 78.445206
Latitude: 17.391133, Longitude: 78.320800
Latitude: 17.391133, Longitude: 78.319293
Latitude: 17.391133, Longitude: 78.320800
Latitude: 17.391133, Longitude: 78.319293
Latitude: 17.391133, Longitude: 78.320800
```

Fig 6: Display of GPS Data on the ‘coordinates.txt’

Client-Side Programming

The client-side application utilizes a login key for basic security and embeds a Google Map for visualization. Figure 7 signifies the algorithm of the client-side application.

Predefined geofence coordinates are stored in JavaScript to define a restricted area. A function, "updateVehicle Location", continuously retrieves the latest GPS coordinates from a server-side text file. The extracted latitude and longitude are validated, and the vehicle marker on the map is updated accordingly. Crucially, the system checks if the updated position falls within the geofence. If a breach occurs, an alert is triggered. To prevent excessive alerts due to minor location fluctuations, a 3-second delay is implemented for outside geofence detections. Additionally, an alert flag ensures only the initial geo-fence entry triggers notification. This cycle of fetching coordinates, updating marker position, and performing geofence checks continues, enabling real-time monitoring of the vehicle's location.

The below Figure 8 demonstrates the alert notification displayed in the case where, in open ground, a restricted area is considered, the vehicle is on the geofence polygon edge and the website is displaying an alert until the vehicle leaves the geofence.

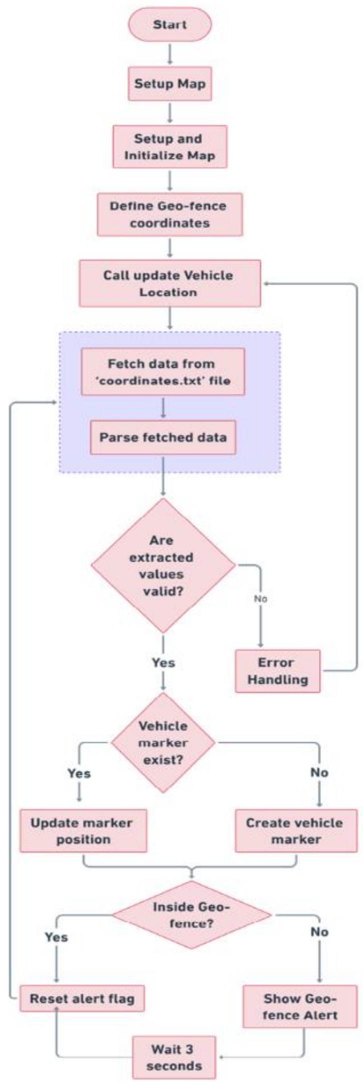
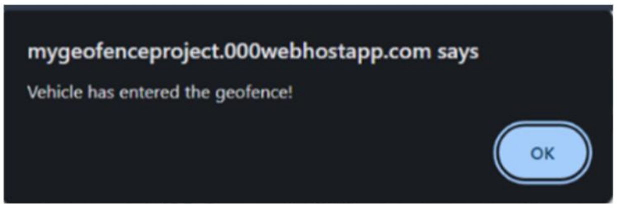


Fig 7 Flowchart on Client-side Application

Results and Discussion

To consider the airport environment, the developed system was tested in open grounds with an environment similar to an airport-restricted zone. Airports have a restricted zone with a rectangular perimeter or a square perimeter; keeping this in view rectangular and square areas were considered restricted zones in the open ground [17]. The developed geofencing system prototype was employed in a car, and test results were collected considering different use cases.


mygeofenceproject.000webhostapp.com says

Vehicle has entered the geofence!

Fig 8: Client-side application alert

The vehicle is represented in the form of a red marker and the restricted area in the open ground is shaded in red. The performance of the system is evaluated in two primary circumstances: Vehicle outside the geofence and Vehicle inside the geofence.

Case A: Vehicle outside the geofence

This section evaluates the system's performance when a runway vehicle equipped with the monitoring system is located outside the predefined geofence. To test the prototype outside the geofence, the vehicle with the prototype placed in it was put in three different locations, and in every location, a geofence polygon was considered. The system was placed at 20 m, 10 m, 5 m, and 2 m away from the geofence polygon, and the alert system was tested. Fig 9a. demonstrates the scenario of a vehicle outside of the geo-fence area and the prototype was present 20 meters away from the geofence polygon. As the vehicle is outside the geo-fence there is no alert notification being displayed. It is observed that when the prototype was 10 m and 5m, respectively, away from the geofence polygon, it didn't display any alert indicating a successful circumstance.

On the contrary, when the prototype was tested against 2 m, it displayed an alert notification (Fig 9b.). The system is designed in such a way that it can accept 6 digits after the decimal point of coordinates representation [18].

When the system has up to 6 decimals of coordinates representation, then the difference of 2 meters and below cannot be identified by it and it considers that the vehicle has entered the geofence. An advantage to this is that the driver present in the vehicle will know early on that the. vehicle is close to the geofence and can retreat before completely entering the geofence.

The successful operation in Case A and its different scenarios demonstrate the system's ability to function as intended even when the vehicle is not within the designated geofence zone. This data can be valuable for historical analysis, route planning, or monitoring overall fleet activity.

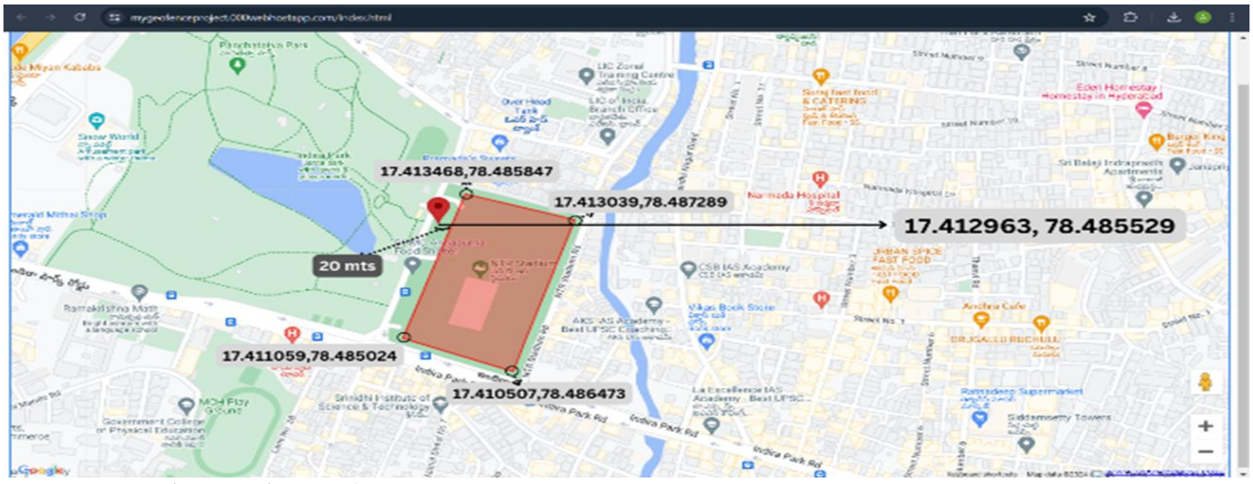


Fig 9a. 20 mts from geofence polygon

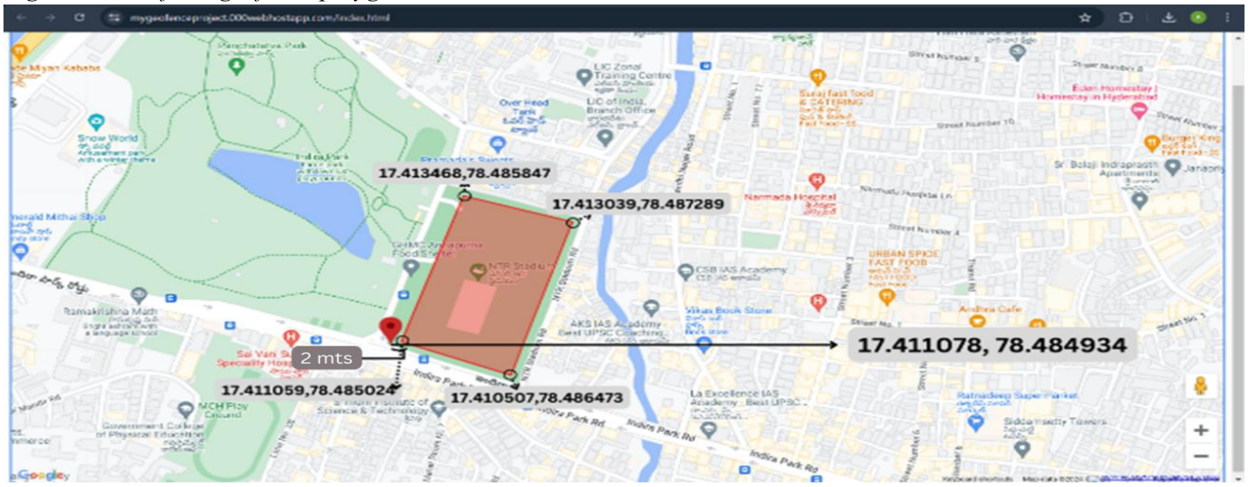
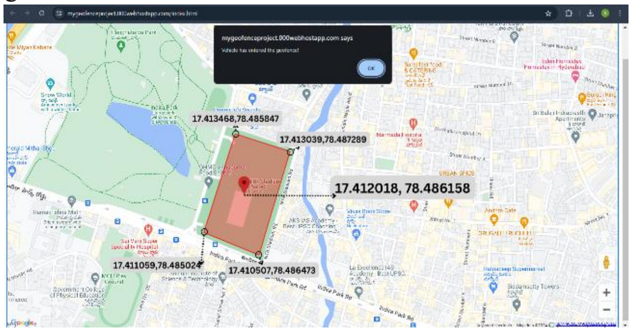


Fig 9b. 2 mts from geofence polygon

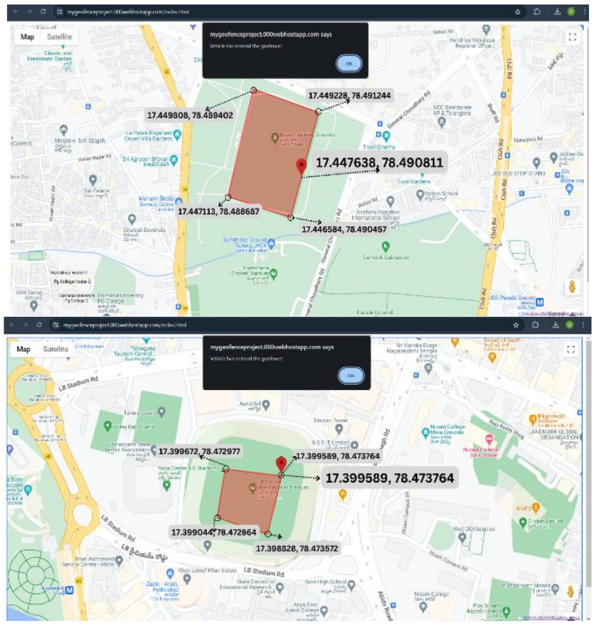
Case B: Vehicle Inside Geofence

The vehicle was tested inside the geofence in three different locations when the prototype was put inside, edge, and at the vertices of the polygon to check whether the prototype showed an alert. Fig 10 depicts that the system is showing alert notifications when the vehicle is in the geofence and Figu 11 represents the scenario of the prototype when it is on the edge and the vertices of the geofence.


mygeofenceproject.000webhostapp.com says

Vehicle has entered the geofence!

Fig 10: Alert notification when the vehicle is inside the geofence polygon


mygeofenceproject.000webhostapp.com says

Vehicle has entered the geofence!

Fig 11: Geofence Boundary Breach Warning

The successful operation in Case B demonstrates the system's core functionality of geofence monitoring. The real-time location update and the triggered alert ensure vehicle drivers and airport traffic controllers are informed when a vehicle enters a designated restricted area. This functionality is valuable for applications such as monitoring deliveries within a specific zone, tracing fleet activity in sensitive locations, or detecting unauthorized access.

Conclusion:

This paper explains a successfully built functional prototype for a vehicle monitoring system. It utilized GPS, a microcontroller, and a server application for real-time tracking, data storage, and geofence management by sending out alert notifications every 3 seconds until the vehicle leaves the restricted zone. The system displays alert notifications when the vehicle is 2 meters or less than 2 meters away from the geofence and when the vehicle is inside the geofence. This is incorporated into the system so that the driver can be alerted when the vehicle is nearing the restricted zone. A user-friendly web interface visualizes vehicle location and allows geofence control. The chosen hardware prioritized low power consumption (412.5 mW), considering the airport environment. Future development could explore advanced geofence functionalities, like route optimization for exiting restricted zones. This project demonstrated cost-effectiveness as it is just under 25 USD and can be implemented on multiple vehicles and an efficient vehicle monitoring system that can be implemented on an Air traffic control system for efficient traffic management, paving the way for further exploration of advanced features for a comprehensive tracking solution.

References

1. A. E. Smith, D. W. Coit and Y. -C. Liang, "Neural Network Models to Anticipate Failures of Airport Ground Transportation Vehicle Doors," in IEEE Transactions on Automation Science and Engineering, vol. 7, no. 1, pp. 183-188, Jan. 2010, doi: 10.1109/TASE.2009.2020508
2. E. Blasch, Z. Wang, D. Shen, H. Ling, and G. Chen, "Surveillance of Ground Vehicles for Airport Security," Proceedings of SPIE - The International Society for Optical Engineering, vol. 9089, article 90890B, June 2014, doi: 10.1117/12.2053855.
3. I.Alomara and J. Tolujevs, "Optimization of Ground Vehicles Movement on the Aerodrome," Transportation Research Procedia, vol. 24, pp. 58-64, May.2017, ISSN 2352-1465, doi:10.1016/j.trpro.2017.05.068.
4. J. Tang, "Review: Analysis and Improvement of Traffic Alert and Collision Avoidance System," in IEEE Access, vol. 5, pp. 21419-21429, Oct. 2017, doi: 10.1109/ACCESS.2017.2757598.
5. Y. E. Hawas, G. Thandavarayan, B. Basheerudeen and M. Sherif, "Testbed Evaluation of Real-Time Route Guidance in Inter-Vehicular Communication Urban Networks," in IEEE Access, vol. 7, pp. 1470-1485, Jan.2019, doi: 10.1109/ACCESS.2018.2886822.
6. A. Jospine, L. H. M. Audah, A. Hamzah, and H. Qasim, "Vehicle Monitoring System with Geofencing Capability," Journal of Electronics Voltage and Application, vol. 1, no. 2, pp. 1-13, Oct 2020, doi: 10.30880/jeva.2020.01.02.001.
7. S. Wang, Y. Che, H. Zhao and A. Lim, "Accurate Tracking, Collision Detection, and Optimal Scheduling of Airport Ground Support Equipment," in IEEE Internet of Things Journal, vol. 8, no. 1, pp. 572-584, 1 Jan.1, 2021, doi: 10.1109/JIOT.2020.3004874.
8. M. Stevens and E. Atkins, "Geofence Definition and Deconfliction for UAS Traffic Management," in IEEE Transactions on Intelligent Transportation Systems, vol. 22, no. 9, pp. 5880-5889, Sept. 2021, doi: 10.1109/TITS.2020.3040595.
9. C. Ding, J. Bi, D. Xie, X. Zhao, and Y. Liu, "Mining Travel Time of Airport Ferry Network Based on Historical Trajectory Data," Hindawi Journal of Advanced Transportation, vol. 2021, Article ID 9231451, pp. 1-11, Oct.2021, doi:10.1155/2021/9231451.
10. P. Rosayyan, S. Subramaniam and S. I. Ganesan, "Decentralized Emergency Service Vehicle Pre-Emption System Using RF Communication and GNSS-Based Geo-Fencing," in IEEE Transactions on Intelligent Transportation Systems, vol. 22, no. 12, pp. 7726-7735, Dec. 2021, doi: 10.1109/TITS.2020.3007671.
11. S. Wang, C. Li and A. Lim, "ROPHS: Determine Real-Time Status of a Multi-Carriage Logistics Train at Airport," in IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 7, pp. 6347-6356, July.2022, doi: 10.1109/TITS.2021.3055838
12. J. Zhou, Y. Wu, Z. Cao, W. Song, J. Zhang and Z. Chen, "Learning Large Neighborhood Search for Vehicle Routing in Airport Ground Handling," in IEEE Transactions on Knowledge and Data Engineering, vol. 35, no. 9, pp. 9769-9782, 1 Sept. 2023, doi: 10.1109/TKDE.2023.3249799.
13. D. Lofù, P. D. Gennaro, P. Tedeschi, T. D. Noia and E. D. Sciascio, "URANUS: Radio Frequency Tracking, Classification and Identification of Unmanned Aircraft Vehicles," in IEEE Open Journal of Vehicular Technology, vol. 4, pp. 921-935,Nov.2023, doi: 10.1109/OJVT.2023.3333676.
14. U-blox NEO-6M documentation: https://content.ublox.com/sites/default/files/products/documents/NEO-6_DataSheet_%28GPS.G6-HW-09005%29.pdf
15. U-blox NEO-6M datasheet: https://content.ublox.com/sites/default/files/products/documents/NEO-6_DataSheet_%28GPS.G6-HW-09005%29.pdf
16. ESP8266 Wi-Fi Module datasheet (refer to power consumption section): https://www.espressif.com/en/products/modules/esp8266
17. Angel, A. (n.d.). *Flight Restriction Zone (FRZ) Guidelines | Altitude Angel | Drones*. https://www.altitudeangel.com/sectors/airports/flight-restriction-zone#:~:text=Runway%20Protection%20Zones%3A%20A%20rectangle,2000%20ft%20above%20ground%20level.
18. *Why do you need 6 decimal places?* (n.d.). https://gis.maricopa.gov/GIO/HistoricalAerial/help/why_do_you_need_6_decimal_places_.htm#:~:text=Previous%20page.%20Next%20page.%20Print%20version.