Framework for Remote Environmental Sensing in Augmented Reality: Real-time Temperature Data Integration with AR Foundation and IoT

Yugesh Padmanaban .V   
*Department of Computer Science and Engineering [cybersecurity & IoT].*  
*Sri Ramachandra Institute Of Higher Education and Research.*Chennai, India.  
yugee03@gmail.com

*Abstract*—Remote monitoring of environmental conditions in hazardous or inaccessible areas, such as chemical plants, remote ecological sites, or disaster zones, is essential for safety and efficiency but often challenging to achieve in real time. This paper presents a novel augmented reality (AR) application that provides interactive, real-time temperature monitoring by visualizing data from remotely placed IoT-based sensors. Leveraging AR Foundation, AR Core, and the XR Interaction Toolkit in Unity, this mobile AR interface allows users to scan a physical surface, place a 3D model, and observe live temperature data from IoT sensors connected to an Arduino, transmitted continuously through an API link. The model dynamically responds to temperature changes, with visual indicators—such as color shifts—when readings exceed critical thresholds, offering an intuitive solution for real-time environmental monitoring where human presence is limited or unsafe. The application is particularly useful in real-life scenarios such as monitoring temperature-sensitive industrial equipment, ensuring safety in hazardous factory zones, or tracking equipment conditions in intensive care units (ICUs), providing immediate and actionable insights. This approach combines the advantages of IoT and AR to enhance situational awareness, providing a low-latency, accessible framework for responsive remote sensing in mobile AR environments.

Keywords—Augmented Reality, Internet of Things (IoT) Integration, Remote Hazardous Area Monitoring, Cloud-based IoT Data Streaming, Mobile AR Interfaces, Real-time Sensor Data Processing.

# Introduction

Augmented Reality (AR) combined with the Internet of Things (IoT) offers innovative solutions for real-time environmental monitoring in hazardous or remote locations. Traditional systems often lack interactivity and real-time visual feedback, limiting their effectiveness in critical scenarios. This project focuses on developing an AR-based system for remote temperature monitoring, utilizing IoT-enabled temperature sensors connected to an Arduino platform that transmits live data via an API from Arduino Cloud. The AR interface dynamically visualizes this data through 3D models, providing immediate feedback to users based on real-time conditions.

The system employs Unity's AR Foundation, XR Interaction Toolkit, and Lean Touch framework to ensure an interactive, scalable, and mobile-compatible experience. Users can interact with 3D models using touch gestures such as scaling, rotation, and movement, enabling seamless and precise data visualization even on devices with limited processing power. Real-time responsiveness is achieved as the 3D model adapts to temperature changes, such as altering its colour when thresholds are exceeded, enhancing situational awareness and decision-making.

This AR-IoT integration has broad applications in fields like industrial safety, ecological monitoring, and disaster management. By delivering immersive visualizations and immediate feedback, the system improves the efficiency of monitoring critical conditions in inaccessible environments. With cross-platform compatibility supported by AR-Core and ARKit, the solution ensures adaptability and scalability, setting the foundation for enhanced monitoring in high-risk environments.

# Background

# Methodology

The suggested system combines mobile technology, Internet of Things (IoT) sensors, and augmented reality (AR) to enable real-time temperature monitoring in dangerous or remote areas. Sensor setup, data gathering, augmented reality visualisation, and user engagement with the mobile application are the many phases of the process.

## Requirement Analysis

First, The project begins with identifying key features of the AR application, including real-time sensor data visualization, 3D model interaction, and dynamic environmental monitoring. The hardware requirements include an ArduinoNano33IoT board and a DHT22sensor for precise temperature measurement. The Arduino IoT Cloud provides the API link to transmit real-time data, ensuring the system can operate effectively in hazardous or inaccessible environments.

## AR Integration

The application employs Unity’s AR Foundation to overlay virtual 3D models on the physical world. AR capabilities allow users to scan surfaces and place a 3D model dynamically in their surroundings. Utilizing Lean Touch enhances interactivity, enabling gestures like scaling, rotation, and repositioning of the 3D model. Additionally, the AR interface visually alerts users with dynamic colour changes in the 3D model based on live temperature data, ensuring a seamless interaction between the virtual and real-world environments.

## 3D Model Integration

The integration of 3D models involves importing models into Unity, refining them with textures and materials for a realistic appearance. C# scripts are implemented to allow user interactions, including model scaling, rotation, and colour changes based on temperature readings received via the API. Performance optimization ensures smooth operation in AR environments.

## Sensor Data Integration

The DHT22 sensor, connected to the Arduino Nano 33 IoT, captures temperature readings. These readings are transmitted to the Arduino IoT Cloud, generating a REST API link that the Unity application fetches in real time. This integration ensures the app reflects dynamic temperature changes instantly. When temperature thresholds are exceeded, the 3D model provides immediate visual feedback, such as colour shifts, offering an intuitive mechanism for environmental monitoring.

## Testing and Debugging

The application undergoes rigorous testing phases, including functional, usability, and performance testing. These phases ensure the system operates reliably under various scenarios. Issues identified are resolved through iterative debugging. Beta testing with user groups evaluates real-world functionality, and feedback is incorporated to improve application stability and refine features for a better user experience.

## Continuous Monitoring and Updates

Post-launch, the application will be monitored continuously to identify and address emerging issues or vulnerabilities. Regular maintenance cycles will ensure the system remains secure and operational. User feedback will guide future updates and feature enhancements, ensuring the application adapts to evolving requirements and maintains relevance in real-world scenarios.

# Implementation

## System Configuration and Setup

The implementation began with configuring Unity 2022.3, selected for its compatibility with AR Foundation and XR Interaction Toolkit. Essential packages like AR-Core, AR Foundation, and XR Interaction Toolkit were integrated to enable augmented reality functionality. The Arduino Nano 33 IoT board and DHT22 sensor were configured to collect temperature data, which was then linked to the Arduino Cloud for real-time API access. Prepared 3D models, optimized for mobile rendering, were imported into Unity to represent the IoT setup visually.

## Augmented Reality Functionality

The implementation of augmented reality functionalities was achieved through the utilization of Unity's XR Origin and AR Ray-cast Manager, facilitating the precise detection of planar surfaces such as floors, tables, or other horizontal substrates within the physical environment. These components enabled the seamless overlay of virtual elements onto the real world, establishing a robust AR interface.

Through intuitive interaction mechanisms, users were empowered to deploy a singular, dynamically-responsive 3D representation of the DHT22 sensor onto the identified surface via a simple tap. A sophisticated instantiation control logic ensured the presence of only one model at any given moment. Subsequent user inputs initiated the relocation of the pre-existing model to newly designated positions, obviating redundant instantiations and enhancing spatial coherence.

## Integration of Sensor Data

The Arduino Nano 33 IoT board, equipped with the DHT22 sensor, was configured to collect live temperature readings. These readings were transmitted to the Arduino Cloud, which generated an API endpoint for real-time data access. Unity retrieved this sensor data using RESTful API calls, ensuring accurate and continuous communication between the hardware and the application.

In the augmented reality environment, this real-time data dynamically altered the visual properties of the 3D model. For example, if the temperature exceeded 50°C, the 3D model's colour shifted to red, indicating a critical condition. This visual cue provided users with immediate feedback on the sensor's status, bridging the gap between IoT data and AR visualization. This integration was optimized to ensure responsiveness, enabling users to perceive and react to environmental changes effectively.

Thorough testing validated the accuracy of the data transmission, ensuring that sensor readings were reflected correctly in the AR environment. The result was a cohesive system where live sensor data drove real-time visual updates, enhancing situational awareness and interactivity.

## Interactive Model Design

The interactive capabilities of the 3D model were meticulously crafted to facilitate intuitive user engagement within the augmented reality environment. Leveraging advanced touch-based gesture recognition, features such as rotation, scaling, and zooming were seamlessly integrated through the utilization of the Lean Touch asset. This asset provided a versatile and robust framework, enabling precise detection and execution of multi-touch inputs.

These functionalities empowered users to manipulate the virtual representation of the DHT22 sensor dynamically, granting them the ability to explore the model from various perspectives with fluidity and precision. The implementation ensured that interaction remained responsive and ergonomic, fostering an immersive and user-centric interface for AR visualization.

## Validation and Deployment

The comprehensive AR application underwent extensive validation to ensure robust functionality and seamless user experience. Rigorous testing was conducted to evaluate critical aspects, including the accuracy of surface detection, real-time sensor data visualization, and interactive capabilities of the 3D model. The API integration was meticulously assessed to verify flawless communication between the Arduino Nano 33 IoT, the DHT22 sensor, and the Unity AR environment.

Following successful validation, the application was deployed on Android devices, where multi-touch gestures were rigorously examined for precision and responsiveness. Post-deployment, real-world testing was carried out in dynamic environments to confirm the application's operational stability and adaptability. This iterative testing and deployment process ensured that the application met its intended objectives and delivered a seamless and reliable AR experience.

##### References

The template will number citations consecutively within brackets [1]. The sentence punctuation follows the bracket [2]. Refer simply to the reference number, as in [3]—do not use “Ref. [3]” or “reference [3]” except at the beginning of a sentence: “Reference [3] was the first ...”

Number footnotes separately in superscripts. Place the actual footnote at the bottom of the column in which it was cited. Do not put footnotes in the abstract or reference list. Use letters for table footnotes.

Unless there are six authors or more give all authors’ names; do not use “et al.”. Papers that have not been published, even if they have been submitted for publication, should be cited as “unpublished” [4]. Papers that have been accepted for publication should be cited as “in press” [5]. Capitalize only the first word in a paper title, except for proper nouns and element symbols.

For papers published in translation journals, please give the English citation first, followed by the original foreign-language citation [6].

1. G. Eason, B. Noble, and I. N. Sneddon, “On certain integrals of Lipschitz-Hankel type involving products of Bessel functions,” Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955. *(references)*
2. J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.
3. I. S. Jacobs and C. P. Bean, “Fine particles, thin films and exchange anisotropy,” in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.
4. K. Elissa, “Title of paper if known,” unpublished.
5. R. Nicole, “Title of paper with only first word capitalized,” J. Name Stand. Abbrev., in press.
6. Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, “Electron spectroscopy studies on magneto-optical media and plastic substrate interface,” IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].
7. M. Young, The Technical Writer’s Handbook. Mill Valley, CA: University Science, 1989.
8. K. Eves and J. Valasek, “Adaptive control for singularly perturbed systems examples,” Code Ocean, Aug. 2023. [Online]. Available: <https://codeocean.com/capsule/4989235/tree>
9. D. P. Kingma and M. Welling, “Auto-encoding variational Bayes,” 2013, arXiv:1312.6114. [Online]. Available: <https://arxiv.org/abs/1312.6114>
10. S. Liu, “Wi-Fi Energy Detection Testbed (12MTC),” 2023, gitHub repository. [Online]. Available: https://github.com/liustone99/Wi-Fi-Energy-Detection-Testbed-12MTC
11. “Treatment episode data set: discharges (TEDS-D): concatenated, 2006 to 2009.” U.S. Department of Health and Human Services, Substance Abuse and Mental Health Services Administration, Office of Applied Studies, August, 2013, DOI:10.3886/ICPSR30122.v2

**IEEE conference templates contain guidance text for composing and formatting conference papers. Please ensure that all template text is removed from your conference paper prior to submission to the conference. Failure to remove template text from your paper may result in your paper not being published.**

We suggest that you use a text box to insert a graphic (which is ideally a 300 dpi TIFF or EPS file, with all fonts embedded) because, in an MSW document, this method is somewhat more stable than directly inserting a picture.

To have non-visible rules on your frame, use the MSWord “Format” pull-down menu, select Text Box > Colors and Lines to choose No Fill and No Line.