iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 11 **Issue:** X **Month of publication:** October 2023

DOI: https://doi.org/10.22214/ijraset.2023.53664

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue X Oct 2023- Available at www.ijraset.com

Augmented Reality Based Indoor Navigation System

Isha Deshmukh[1], Siddhi Chavan[2], Abhishek Kamtekar[3], Mrs. Vandana Dixit[4]
[1, 2, 3]*Information Technology*
[4]*Information Technology PESMCOE, Shivajinagar, Pune 07*

Abstract: ***Every people, at some point in their life, may get lost inside a large auditorium, struggled to find their exit point at the airport, or may get late to a lecture because they could not find the right lecture hall at the university. This paper addresses the development of an Augmented Reality Based Indoor Navigation Application. It is based on four main modules, 3D model building, shortest path calculation, QR code scanning and path visualization. The application is developed in Unity using AR Core XR plugin management and AR Foundation plugin. ZXing library is used for scanning and decoding the QR codes. We aim at developing the front end in the simplest way possible so that the users can easily reach their destination by just opening the camera where the directions are shown as animations in their surroundings.***
Keywords: Augmented Reality, ZXing, Unity, ARCore, QR codes, 3D model, navigation, path visualization

I. INTRODUCTION

Indoor Navigation or Mapping is a revolutionary concept that visualizes indoor venue and spatial data on a digital 2-Dimensional or 3-Dimensional map. Unlike in Outdoor Navigation because of low signal strength we cannot use GPS which makes getting the current location of the user all the way more difficult. In our application we have proposed the usage of QR codes to get the current location of the user. On scanning a QR code the user's position will be reset in the 3D model with respect to his position in the real environment.AR based Indoor Navigation provides users with an enhanced navigation experience by overlaying virtual information onto the real-world environment. It also offers greater accuracy and precision in location tracking, as well as the ability to provide real-time information about the user's surroundings. The objective of this research paper is to explore the possibilities and advancements in AR-based indoor navigation systems. The proposed application will focus on providing a robust solution to Indoor Navigation challenge. It will provide users with the shortest and most accurate path from source to destination. In case the user gets lost or goes in a wrong direction than what the AR arrows are pointing at the path will be rerouted accordingly.

II. EXISTING SYSTEMS

There are quite a few existing approaches towards Indoor Navigation systems. All these different techniques have approached the problem with different perspectives and hence have very varied implementation. All of them have their own set of advantages and limitations.

A. Wi-Fi Based Systems

Wi-Fi-based indoor navigation systems leverage the existing Wi-Fi infrastructure to provide positioning and navigation information to users. These systems utilize signal strength measurements from Wi-Fi access points to estimate the user's location. However, the accuracy of Wi-Fi-based systems can be affected by signal interference, multipath propagation, and limited coverage.

B. Bluetooth Beacon Systems

Bluetooth beacon systems utilize low-power Bluetooth beacons placed throughout an indoor environment to enable navigation. These beacons transmit signals that can be received by users' devices, allowing for proximity-based positioning. However, Bluetooth beacon systems require the installation and maintenance of a network of beacons, which can be costly and time-consuming.

C. Inertial Navigation Systems

Inertial navigation systems rely on sensors such as accelerometers and gyroscopes to track the user's movement and estimate their position. By measuring changes in velocity and direction, these systems can provide real-time positioning information. However, inertial navigation systems suffer from cumulative errors over time, leading to decreased accuracy.

D. Ultrasound-Based Systems

Ultrasound-based indoor navigation systems utilize ultrasound signals and receivers to estimate the user's location. These systems emit ultrasonic waves and measure the time it takes for the waves to reach the receiver, enabling distance calculations. However, ultrasound-based systems require a dense network of transmitters and receivers, which can be costly to install and maintain.

III. SYSTEM ARCHITECTURE

The proposed solution is going to a mobile Application. There are four main modules whose interdependent functionality sums up the architecture.

A. 3D Model Building

Here, we generate a 3D model of the indoor space from the blueprints of the area. This model is the 3D visual representation of the real environment in which the user will navigate. The model is constructed in the Unity editor by creating Game Objects of different shapes as required by the model structure. The approach is to do all the required calculations by taking this 3D model as reference then overlay the result onto the real environment. Hence, accuracy of the model is very important to get the best results.

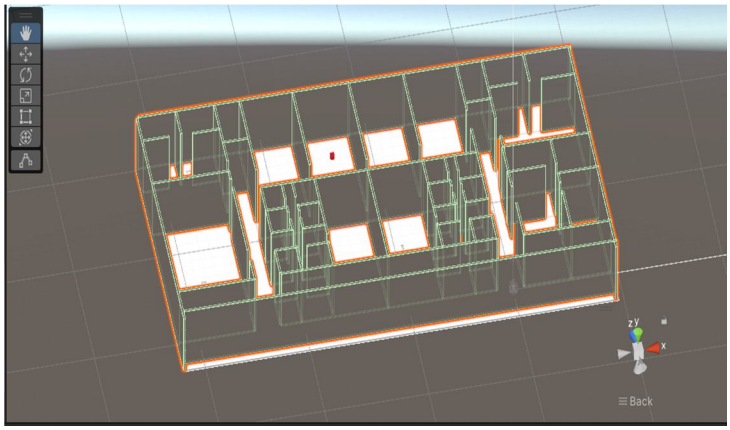


Figure 1: 3D model

B. NavMesh Generation

Unity is the core for making gaming applications. In games there are mapped areas where the game character is allowed to move freely. We are using the same concept in our application to decide walkable area for the user. After the 3D model generation, we will bake the model with appropriate setting of user height and user radius. This will generate the NavMesh which is visible as a blue overlay on the model. The walls are deselected and the floor is overlayed by the mesh determining walkable spaces.

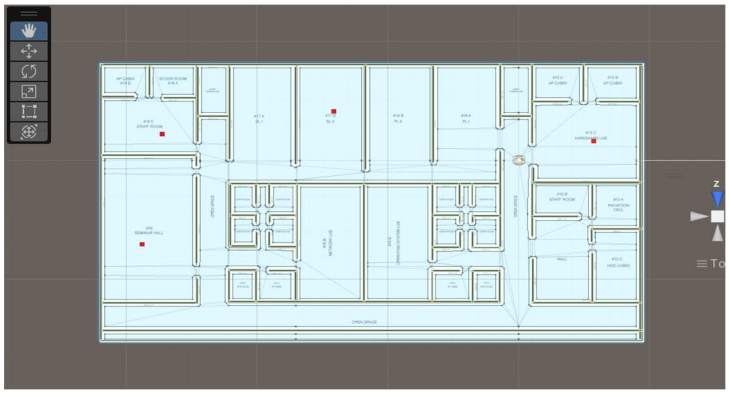


Figure 2: NavMesh

C. Mini Map

The UI interface whenever loaded will show a 2d map called Mini Map of the real environment.

D. Indicator

This is a Game Object which points to user's current location in the environment. It appears as a blue dot and will be placed on the Mini Map. As user's move in the real environment, it will move accordingly in the 3D space, which the user will be able to see in the Mini Map. It helps user keep a track his movement in the environment.

E. Start Point

When the user opens the application on the UI the mini map and other components will be shown. Initially irrespective of user's position in the real environment, in the virtual space the indicator will be at a fixed starting point. This point is pre-defined during map generation.

F. QRCode Scanning

Once the user scans a QRcode using the scanQR button, the indicator position on the mini map will reset to user's current location in the real environment.

G. Destination Targets.

After setting the source location, user will now choose the destination target from the drop-down menu.

H. Pathfinding and Visualization

Once source and destination are set A* algorithm will calculate the shortest and most accurate path. The path will then be overlayed in the form on 3D virtual line in the real environment.

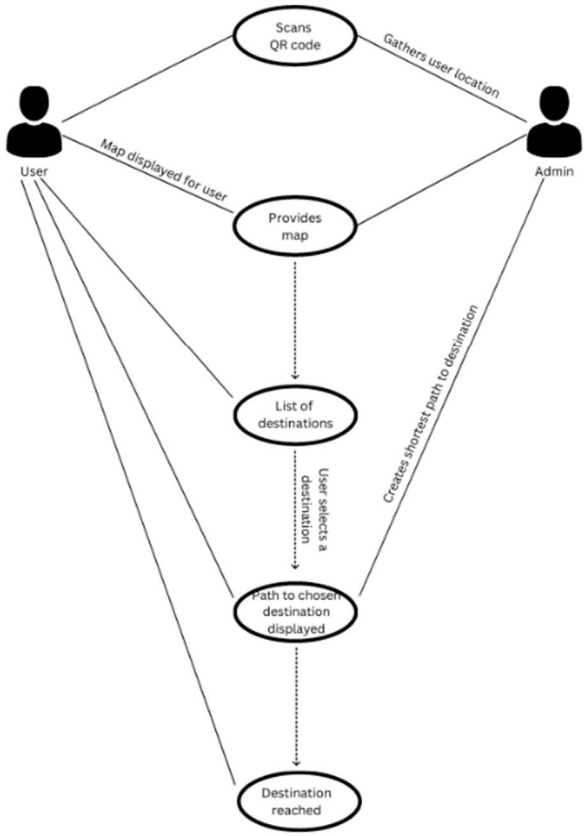


Figure 3: Functional Requirement

IV. TECHNOLOGY

A. Unity

It is a popular game development engine that can also be used for creating augmented reality (AR) applications, including AR indoor navigation systems. It provides several features and capabilities that are beneficial for building AR navigation experiences. It offers powerful tools for creating and manipulating 3D content, which can be used to build realistic indoor environments. You can model and import 3D assets, apply textures and materials, and arrange objects to create the indoor spaces where the navigation takes place. It provides a wide range of user interaction options, including touch, gestures, and input from sensors. These interactions can be utilized to enable users to interact with the AR indoor navigation system, such as selecting destinations, zooming in/out, or rotating the view

B. AR Foundation

Unity's AR Foundation is a framework that allows developers to create AR applications that can run on multiple AR platforms, such as ARKit (iOS) and ARCore (Android). It provides a unified API for interacting with AR features, including plane detection, object tracking, and spatial mapping, which are essential for indoor navigation.

C. ARCore

Unity supports AR tracking technologies like ARKit and ARCore, which enable accurate tracking of the device's position and orientation in the real world. This tracking information is crucial for aligning the virtual navigation elements with the physical environment.

D. NavMesh

NavMesh is a key component in Unity that enables efficient and realistic navigation for characters or objects within a game or simulation environment. It is particularly useful for implementing pathfinding and movement in a 3D world. It refers to a specialized data structure that represents the walkable surfaces or areas within a game or simulation scene. It consists of interconnected triangles or polygons that define where characters or objects can navigate. Unity's NavMesh system provides built-in tools for generating and utilizing NavMesh within your game or simulation.

E. A* Algorithm

The A* algorithm combines elements of both Dijkstra's algorithm and greedy best-first search. It uses a heuristic function to estimate the cost of reaching the goal from each node and selects the most promising nodes to explore. The algorithm is guided by both the actual cost of reaching a node (g-value) and the estimated cost from the node to the goal (h-value).

F. ZXing Library

It is a popular open-source library for barcode and QR code processing. It provides functionality for decoding and encoding various types of barcodes, including QR codes, in different formats. Here are some key features and uses of the ZXing library.

V. AR POSITIONING TECHNIQUES

Augmented Reality (AR) positioning techniques are used to accurately determine the position and orientation of virtual objects in the real world. These techniques enable AR applications to align virtual content with the user's physical environment.

1) *Marker-Based Tracking:* Marker-based tracking involves using predefined markers or visual cues placed in the real world to track the user's position and orientation. These markers are typically recognized by computer vision algorithms, and their positions are used to align virtual objects. Marker-based tracking can be robust and accurate, but it requires markers to be present in the user's environment.
2) *Natural Feature Tracking:* Natural feature tracking techniques use the features present in the real-world environment, such as distinctive landmarks or objects, for position tracking. Computer vision algorithms analyse the visual features and match them to a reference image or map to estimate the user's position. Natural feature tracking can provide good accuracy without the need for markers, but it relies on the availability of recognizable features.
3) *SLAM (Simultaneous Localization and Mapping):* SLAM is a technique that combines real-time localization with the construction of a map of the environment. It uses sensor data, such as camera images or depth information, to estimate the user's position while simultaneously creating a map of the surroundings. SLAM is particularly useful for AR applications that require real-time tracking and mapping in dynamic environments.

Here for our proposed system, we are using combination of Marker-Based Tracking and SLAM. Using QR codes we will be getting user's current position which is a marker-based approach. Once the indicator position is set, we will place navigation script on the indicator Game Object which will track the user movement using SLAM algorithm.

VI. MAPPING THE ENVIRONMENT

When it comes to creating 3D models for AR applications, there are several approaches available.

A. Point Clouds

Point clouds involve capturing many 3D points in the real-world environment using depth sensors, such as LiDAR or depth cameras. These points represent the geometry and spatial information of the physical space. Point cloud data can be processed and used to create a 3D model of the environment. The advantage of point clouds is that they provide a detailed representation of the real world, allowing virtual objects to be accurately aligned with the physical environment. Point cloud-based approaches often require sophisticated sensor hardware and computational resources for processing and rendering.

B. Manual Model Building in Unity

In this approach, the 3D model of the environment is manually created in Unity, the game engine commonly used for AR development. Developers use modelling tools within Unity or import pre-made 3D models to build the virtual representation of the physical space. The manual model building process involves designing and placing virtual objects, defining their positions, orientations, and interactions with the environment. This approach allows developers to have full control over the virtual scene and customize it according to the specific needs of the AR application. It is suitable for situations where precise control over the virtual content is required, such as interactive AR experiences or simulations.

C. Anchors:

Anchors are predefined points or markers in the physical environment that act as reference points for aligning virtual objects. They can be visual markers like QR codes or physical landmarks like specific objects or features in the environment. By recognizing and tracking these anchors using computer vision or sensor-based techniques, the AR system can position and anchor virtual content in the correct locations. Anchors provide a reliable and consistent way to align virtual objects, making them appear stable and fixed in the real world. The choice of approach depends on factors such as the level of accuracy required, available hardware and sensors, complexity of the environment, and the desired user experience. Developers can also combine multiple approaches to achieve the desired results.

VII. PROPOSED METHODOLOGY

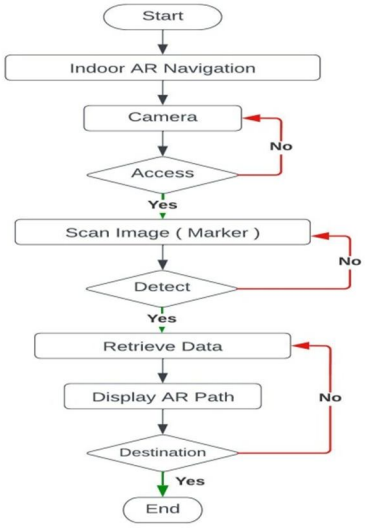


Figure 4: Control Flow Diagram

A. Scanning QR Code

On opening the application, the UI will have a Mini Map with indicator at the default location. The user can begin with clicking on scan qr code button. This will open another activity which will ask for camera access. On granting camera access the the camera frame will be received having the qr code image. On receive camera frame a method is triggered in the backend to receive the captured frame.

1) *Processing the QR Code:* The QR code is received as a XRCpuImage. The image is converted and stored in an array format. For decoding the method takes the pixel data, along with the width and height of the image, and returns a Result object containing the decoded QR code information.

B. Reset the Indicator Position

The result obtained from the qr code is the name of one of the destinations. As here the approach is all the destinations can be source points too. So qr codes are placed strategically at all possible destinations which will also act as source points. Once decoded, if it is a valid source point then the indicator position is reset on the mini map.

C. Path calculation

Unity has an inbuilt method called NavMesh.CalculatePath which we are using to calculate the shortest path between source and destination.

1) *Start and End Point Validation:* The method first checks if the provided start and end points are valid and lie within the bounds of the NavMesh. If either point is outside the NavMesh or not reachable, the method returns false, indicating that no path can be found.
2) *NavMesh Query Initialization:* The method initializes a NavMeshQuery object, which serves as the main interface for querying the NavMesh data.
3) *Query Status and Polygons:* The method uses the NavMeshQuery object to perform a series of queries to identify the polygons that contain the start and end points. It retrieves the corresponding polygon IDs for both points.
4) *Path Calculation:* Using the start and end polygon IDs, the method calls the internal path calculation algorithm, which is typically based on the A* (A-star) algorithm. This algorithm searches the NavMesh graph, evaluating potential paths based on their cost and heuristic estimates.
5) *Path Optimization:* The algorithm explores the connected polygons, considering factors such as polygon costs, obstacles, and any constraints defined in the NavMesh. It gradually expands the search, comparing the estimated costs of different paths and selecting the most optimal route.
6) *Waypoint Extraction:* Once the path calculation is complete, the method retrieves the calculated path as a series of waypoints or vertices. These waypoints represent the optimal route between the start and end points on the NavMesh. The waypoints are typically stored in an array or list.
7) *Path Smoothing:* Depending on the specific implementation, the method may apply path smoothing or optimization techniques to the calculated path. This can involve removing unnecessary waypoints, adjusting the curvature of the path, or considering other factors to ensure smooth and natural navigation.
8) *Result Reporting:* The method returns the calculated path, typically as an array or list of waypoints, allowing the caller to utilize the path for further processing or navigation.

D. Line Rendering

After calculating the shortest path using a pathfinding algorithm, the Line Renderer component in Unity can be used to visually display the path in the real environment. The Line Renderer allows you to draw lines or curves in 3D space, which can be useful for visualizing the path on the screen or in augmented reality (AR) environments.

1) *Obtain the Calculated Path:* Once you have the shortest path, typically represented as a series of waypoints or vertices, you can pass this data to the Line Renderer component.
2) *Set up the Line Renderer:* Attach a Line Renderer component to a game object in your scene. You can customize various properties of the Line Renderer, such as the colour, width, and material used for rendering the line.
3) *Set the Positions:* Assign the positions of the Line Renderer using the calculated path. Iterate through the waypoints of the path and set the positions of the Line Renderer to match those waypoints. Each waypoint represents a point in 3D space along the desired path.

4) *Display the Line:* Enable or activate the Line Renderer component to start displaying the line in the real environment. The line will appear as a visual representation of the calculated path, connecting the waypoints or vertices.
5) *Update the Line:* If the path changes dynamically or needs to be updated in real-time, you can recompute the path using the pathfinding algorithm as needed and update the positions of the Line Renderer accordingly. This allows the line to adjust and reflect any changes in the path.

E. Device positioning and movement

One the line has been rendered and is visible to the user he can hold the mobile device in front of him and follow the path. To gauge the device movement along with the line we use Visual Inertial Odometry. It is a common technique used in AR to estimate the device's position and movement by combining visual data from the camera with the motion sensor data. VIO algorithms analyse the camera feed and track visual features in the environment to determine the camera's relative position and motion.

VIII. USER INTERFACE

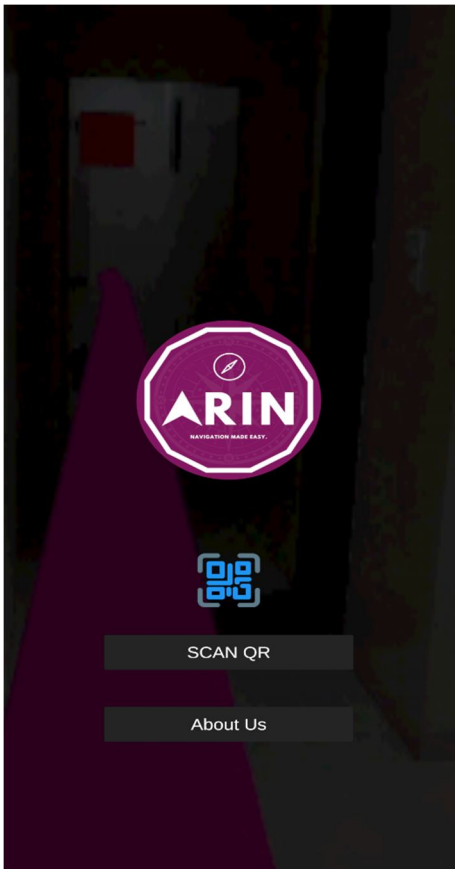


Figure 5: Home Page

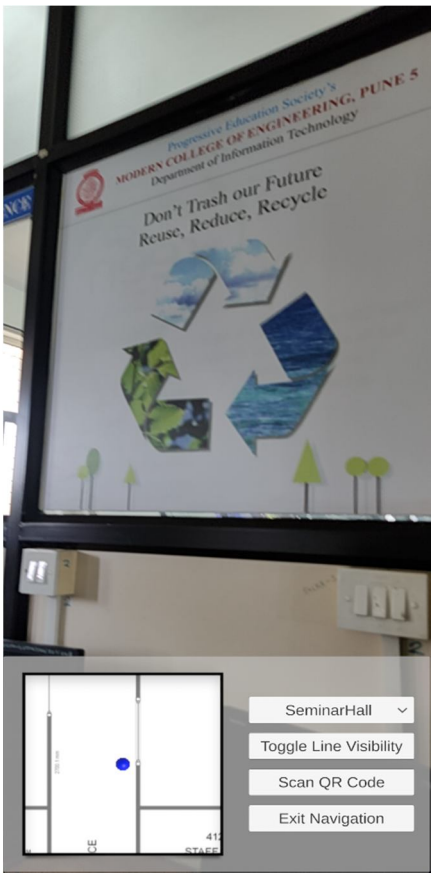


Figure 6: User Interface

The above two figures represent the UI of the mobile application. The first figure shows what the user will see when he will open the application. After clicking on SCAN QR and on successful scanning figure 5 will load.
In figure 6 the bottom left corner is the Mini Map with the blue dot as the indicator. On the bottom right the user has various options to choose from.
First from dropdown list of destination targets he will select the target, followed by toggling the line visibility to see the navigation path.
During navigation if he loses focus, he can again scan a nearby qr code and continue with the navigation. After reaching the destination the user can exit the navigation, otherwise the application will continue accessing his camera in the background and drain the mobile phone battery.

IX. TEST CASES AND RESULTS

To test our application for robustness and accuracy we implemented it under different scenarios.

A. Scanning Valid QR code

Expected behaviour: The application will successfully scan the QR code, decode the source target and accordingly reset the indicator position.

Figure 7: Scan QR

B. Scanning Multiple QR Codes

Expected Behaviour: Simultaneous scanning of QR codes should work without the need to refresh or restart the application

C. Scanning Invalid QR Code

Expected Result: The application should handle the invalid QR code gracefully and provide appropriate feedback or error messages

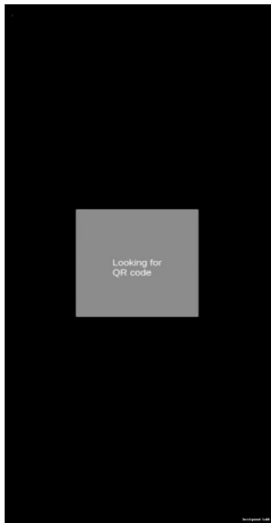


Figure 8: Invalid QR

D. Rerouting

Expected Behaviour: In case of losing camera, focus or going in the wrong direction, when the rendered line is displaying, the path should dynamically reroute rather than the need to scan another QR code.

E. Path Visualization

Expected Behaviour: After setting the source and destination target, the rendered line should be overlayed in the real environment, and will be visible to the user through his mobile device.

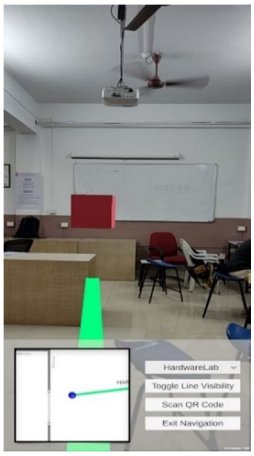


Figure 9: Path display

F. Toggle Line visibility

Expected Behaviour: After the navigation line is visible on toggling the line visibility button the line display should show correct display.

G. Navigation Target selection

Expected Result: The application should set the target position correctly based on the selected navigation target.

Figure 10: target selection

X. RESULTS DISCUSSION

After successfully testing all the relevant cases, it is important to perform a thorough result analysis and have a discussion on the findings. From the figures shown in the result section we can see that, the QR code scanning functionality is robust and capable of handling multiple codes in succession. This allows for seamless location identification and navigation initiation. The rerouting mechanism demonstrates the system's ability to adapt to real-time changes and provide dynamic navigation instructions. This enhances user experience and ensures efficient navigation even in unexpected scenarios.

The system accurately calculated the shortest paths between the user's location and the destination, and the visualization of the paths in the AR interface was clear and aligned with the real environment. The path calculation algorithm utilized by the system is reliable, providing optimal routes for navigation. The visualization of paths in the AR environment enhances user understanding and facilitates seamless navigation within the indoor space.

The system's ability to handle user movements, provide real-time updates, and respond swiftly contributes to a smooth and enjoyable navigation experience. Users can rely on the system for accurate guidance throughout their journey. The user interface design considerations were effective in creating an intuitive and user-friendly AR indoor navigation system. Usability testing helped identify areas for improvement, resulting in an interface that users find straightforward and comfortable to interact with.

XI. CHALLENGES

1) Tracking and Alignment Issues: Challenges in accurately tracking the user's position and aligning the virtual overlays with the real-world environment can lead to misalignment and inaccurate navigation instructions.
2) Environmental Interference: Environmental factors such as poor lighting conditions, reflective surfaces, or occlusions can impact the performance of AR tracking systems and affect the reliability of the navigation system.
3) Battery Life and Device Limitations: Augmented reality applications can be resource-intensive and consume significant battery power. Insufficient battery life or device limitations can impact the usability and longevity of the application during prolonged navigation sessions.
4) User Orientation and Spatial Awareness: Users may face challenges in understanding and interpreting the augmented reality overlays, resulting in confusion, disorientation, or difficulty in following the navigation instructions.

XII.FUTURE RESEARCH

Extend the system to support multi-floor navigation within complex indoor environments. This would enable users to seamlessly navigate between different levels of a building or facility. Explore additional AR features to enhance the user experience, such as 3D object recognition, interactive visual cues, or augmented reality annotations that provide contextual information about points of interest along the navigation path.

Implement voice-guided instructions to provide auditory cues and directions, enhancing accessibility and allowing users to navigate hands-free. Incorporate social features into the application, such as the ability to share routes or locations with friends, real-time location sharing, or collaborative navigation in group settings. Develop tools or integrations to facilitate the creation and management of indoor maps, allowing administrators or users to update map data, add new points of interest, and customize the navigation experience.

Integrate the application with smart home systems to enable seamless integration between indoor navigation and smart devices, such as voice assistants or home automation systems, for an enhanced and connected user experience. Explore the application of machine learning and AI techniques to improve path planning algorithms, optimize navigation routes based on user preferences or historical data, and provide personalized recommendations for points of interest.

Extend the compatibility of the application to support multiple platforms and devices, including smartphones, tablets, AR glasses, or wearable devices, to cater to a wider range of users. Incorporate user feedback mechanisms and analytics to gather insights on user behaviour, preferences, and pain points, allowing continuous improvement of the application based on real-world usage data.

XIII. CONCLUSION

In conclusion, the development of an augmented reality (AR) indoor navigation system presents a significant advancement in the field of indoor navigation. Through the integration of AR technologies, such as QR code scanning, path generation, and AR visualization, the application provides users with an intuitive and interactive way to navigate complex indoor environments.

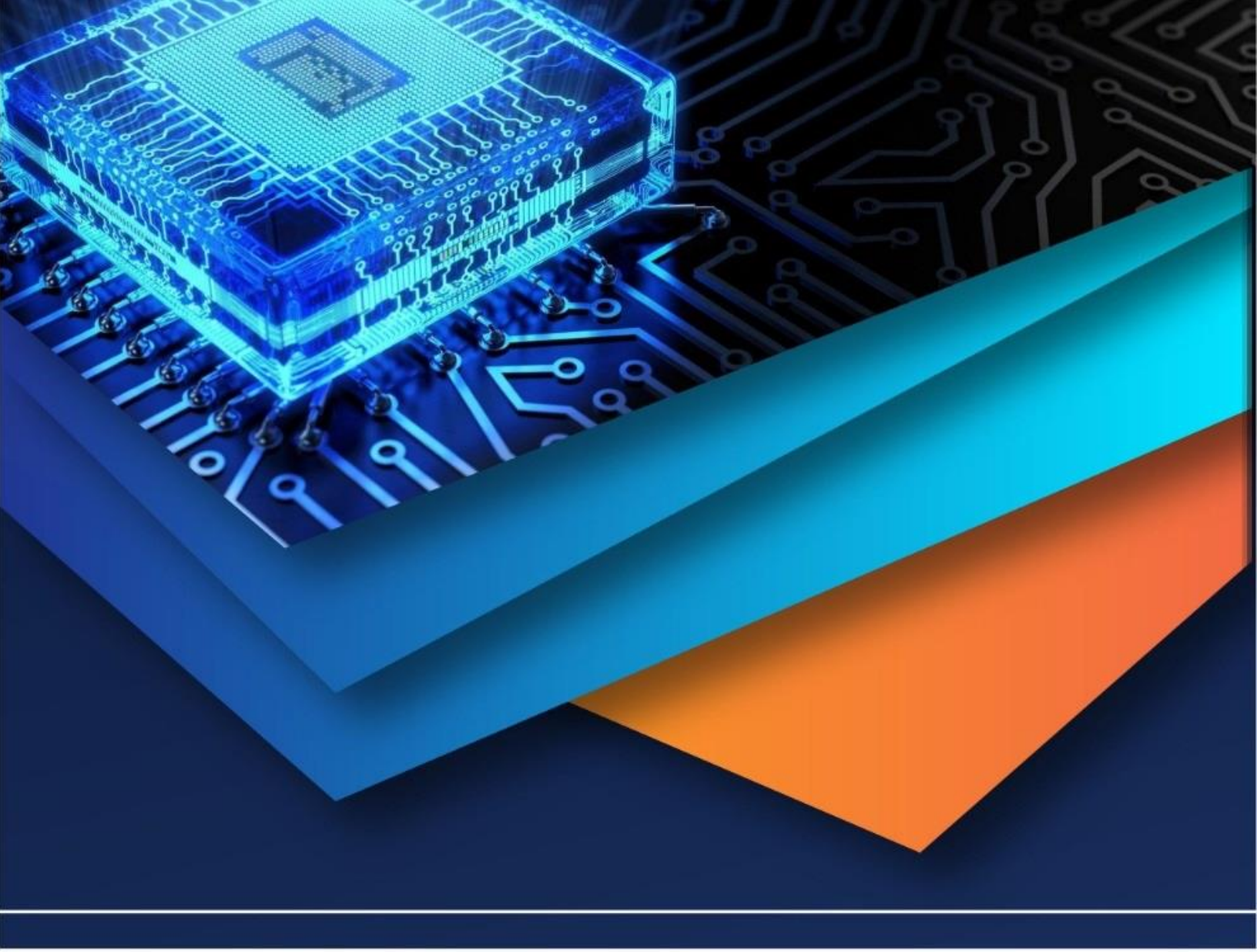
Throughout the research and development process, several key components were implemented and tested successfully. These include the 3D map building using techniques like cloud points or manual modelling, QR code scanning using the ZXing library, path calculation using the A* algorithm and NavMesh, and AR visualization of the calculated path using Unity's Line Renderer.

Extensive testing of the application, including scanning single and multiple QR codes, rerouting, path calculation, and visualization, demonstrated its robustness and accuracy. The successful execution of the test cases validates the effectiveness of the implemented functionalities, ensuring a seamless navigation experience for users.

REFERENCES

[1] Yadav, R., Chugh, H., Jain, V., & Baneriee, P. (2018, October). Indoor Navigation System Using Visual Positioning System with Augmented Reality. In 2018 International Conference on Automation and Computational Engineering (ICACE) (pp. 52-56). IEEE
[2] Mamaeva, D., Afanasev, M., Bakshaev, V., & Kliachin, M. (2019, November). A multi-functional method of QR code used during the process of indoor navigation. In 2019 International Conference on Engineering and Telecommunication (EnT) (pp. 1-4). IEEE.

[3] Sato, F. (2018, July). Indoor Navigation System Based on Augmented Reality Markers. In 2018 International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing.
[4] Umair Rehman and Shi Cao "Augmented-Reality Based Indoor Navigation: A Comparative Analysis of Handheld Devices Versus Google Glass" in IEEE Transactions on Human-Machine Systems, vol. 47,no. 1, pp. 140-15, Feb 2017.
[5] Jang, S.H., 2012, September. A qr code-based indoor navigation system using augmented reality. In GIScience–Seventh International Conference on Geographic Information Science, USA.
[6] Wang, C.S., Chiang, D.J. and Ho, Y.Y., 2012, July. 3D augmented reality mobile navigation system supporting indoor positioning function. In 2012 IEEE International Conference on Computational Intelligence and Cybernetics (CyberneticsCom)(pp. 64-68). IEEE.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY