information

Article

INSUS: Indoor Navigation System Using Unity and Smartphone for User Ambulation Assistance

Evianita Dewi Fajrianti [1], Nobuo Funabiki [1,*], Sritrusta Sukaridhoto [2], Yohanes Yohanie Fridelin Panduman [1], Kong Dezheng [1], Fang Shihao [1] and Anak Agung Surya Pradhana [3]

[1] Graduate School of Natural Science and Technology, Okayama University, Okayama 700-8530, Japan; p2mu1tom@s.okayama-u.ac.jp (E.D.F.)
[2] Department of Informatic and Computer, Politeknik Elektronika Negeri Surabaya, Surabaya 60111, Indonesia
[3] Indonesian Institute of Business and Technology (INSTIKI), Bali 80225, Indonesia
* Correspondence: funabiki@okayama-u.ac.jp

Abstract: Currently, outdoor navigation systems have widely been used around the world on smartphones. They rely on GPS (Global Positioning System). However, indoor navigation systems are still under development due to the complex structure of indoor environments, including multiple floors, many rooms, steps, and elevators. In this paper, we present the design and implementation of the *Indoor Navigation System using Unity and Smartphone (INSUS)*. *INSUS* shows the arrow of the moving direction on the camera view based on a smartphone's augmented reality (AR) technology. To trace the user location, it utilizes the Simultaneous Localization and Mapping (SLAM) technique with a gyroscope and a camera in a smartphone to track users' movements inside a building after initializing the current location by the QR code. *Unity* is introduced to obtain the 3D information of the target indoor environment for *Visual SLAM*. The data are stored in the IoT application server called *SEMAR* for visualizations. We implement a prototype system of *INSUS* inside buildings in two universities. We found that scanning QR codes with the smartphone perpendicular in angle between 60° and 100° achieves the highest QR code detection accuracy. We also found that the phone's tilt angles influence the navigation success rate, with 90° to 100° tilt angles giving better navigation success compared to lower tilt angles. INSUS also proved to be a robust navigation system, evidenced by near identical navigation success rate results in navigation scenarios with or without disturbance. Furthermore, based on the questionnaire responses from the respondents, it was generally found that INSUS received positive feedback and there is support to improve the system.

Keywords: indoor navigation system; INSUS; unity; QR code; smartphone; SEMAR

check for updates

Citation: Fajrianti, E.D.; Funabiki, N.; Sukaridhoto, S.; Panduman, Y.Y.F.; Dezheng, K.; Shihao, F.; Surya Pradhana, A.A. INSUS: Indoor Navigation System Using Unity and Smartphone for User Ambulation Assistance. *Information* **2023**, *14*, 359. https://doi.org/10.3390/info14070359

Academic Editor: Weisong Wen

Received: 20 May 2023
Revised: 21 June 2023
Accepted: 22 June 2023
Published: 24 June 2023

1. Introduction

Navigation is the process of guiding a user from a starting point to a destination by following a path [1]. The first step in navigation is to determine the initial position within the surrounding environment. Navigation is divided into two categories based on the environment, namely outdoor and indoor [2]. Outdoor navigation uses GPS (Global Positioning System) technology in conjunction with other localization techniques such as multilateration and multiangulation [3]. GPS technology has yielded positive outcomes and has been widely implemented in various fields, owing to its high degree of precision. The situation is changing when it comes to indoor navigation, as GPS technology is unusable for this purpose. As a result, different methods are used for indoor navigation. There are several common methods of indoor navigation, including fingerprinting, image matching, and multilateration [4–7].

Among the many indoor navigation methods that require expensive computations and additional devices, there is an existing method called virtual location [8]. This method is composed of two components, namely virtual environment and virtual coordinate.

The virtual environment represents the physical environment. It allows for the mapping of physical locations to a corresponding location in the virtual environment. The virtual coordinates are utilized to estimate the user's position in the physical environment using various techniques.

To create a virtual environment, traditional 3D modelling techniques are commonly used. Manual modelling employs classic approaches that use correct proportions and buildings as references. The building's floor plan is utilized to construct the virtual environment in 3D; the walls, concrete columns, entrances, and floors are modeled [9]. With this technology, the three-dimensional shape of the physical environment's layout is produced in fine detail.

In addition to the virtual environment, virtual coordinates are defined as a set of numbers that represent a user's positions in the virtual environment, based on data from sensors such as the Inertia Measurement Unit (IMU) and the camera on the user's smartphone. Virtual coordinates estimate user position in indoor space and provide navigational instructions [10].

In order to establish the initial location within the virtual environment, pre-defined location coordinates within the environment are used. These coordinates then act as the starting point for the user's navigation. The technique used to acquire this pre-defined coordinate utilizes a Quick Response (QR) Code scanned by the camera of the user's smartphone. After the initial location is established, the process of estimating the user's location is continued using Simultaneous Localization and Mapping (SLAM).

SLAM employs sensors such as cameras and gyroscope sensors. The data from both of these sensors are gathered from the user's movements and surroundings. As the user moves through the environment, SLAM continually updates the estimated position and orientation, providing feedback for navigation. This approach allows for accurate and efficient indoor navigation of complex building environments.

Nonetheless, the use of SLAM is prone drift, which occurs as a result of improper use, such as excessive tilt angles that obstruct the camera's ability to recognize the environment when the user ambulates. This inaccuracy negatively affects the destination location [11]. The inaccuracies in user location estimation caused by drift lead to confusing navigation information for the user.

To address this issue, the rescan procedure was introduced as a solution. The rescan procedure instructs the user to rescan the QR code in the environment to update the system's estimation of their position and orientation. This corrects errors that accumulated in the SLAM process due to drift or other factors and improves navigation instructions. The rescan procedure ensures that the system stays calibrated with the user's actual position and movements, thus leading to a reliable and consistent navigation performance.

During the navigation process, a path is needed to reach a destination. A common and efficient path planning algorithm, the A* algorithm, which utilizes artificial intelligence to find the fastest path with low cost, is introduced [12]. As the algorithm works, it builds a path from the start node to the destination node by connecting the nodes with the lowest total cost.

Taking advantage of the use of cameras in SLAM, an Augmented Reality (AR)-based user interface is the proper choice. The ability to seamlessly integrate the real environment with virtual objects, such as path guidance objects, is one of the benefits of using AR. Another important point for indoor navigation systems is that it is essential for users to be able to view the path to their destination while maintaining awareness of their real-world surroundings [13]. Augmented Reality (AR) technology, which allows for the simultaneous visualization of virtual objects and the actual environment, represents a potentially effective solution.

This paper presents a smartphone-based indoor navigation system that employs virtual locations, SLAM localization, and AR technology for navigation application user interfaces. The use of SLAM enables users to ambulate indoors without the need for additional devices. These indoor navigation systems use a predefined 3D environment.

To establish the user's initial position, a QR code is scanned. To mitigate location drift that occurs while using the system, a rescan procedure is introduced. A database provider, called the SEMAR server, is employed to store the user's ID, authentication, and navigation data. Lastly, the Unity Game Engine software is used to combine these strategies, integrated and exported into an Android-based mobile application.

In this study, we evaluated and compared our application-based indoor navigation system prototype in two different buildings: the #2 Engineering Building at Okayama University and the Graduate Building at PENS. The #2 Engineering Building has four levels, while the Graduate Building only utilizes two levels of its eleven stories, with each floor having varying numbers of rooms that serve as destinations. User testing was conducted in both buildings and upon completing the experiment, participants were provided with a questionnaire to evaluate the performance and functionality of the prototype application.

The continuation of this paper is as follows. Section 2 reviews the previous studies and each used method. Section 3 compares various indoor navigation techniques. Section 4 presents the proposed system. Section 5 evaluates the proposed system through experiments. Finally, Section 6 concludes the paper's findings.

2. Preliminaries

This section provides a summary of studies on indoor navigation systems. Additionally, the application's user interface, path planning algorithm, and communication protocol used to store data in the server are introduced.

2.1. Indoor Localization by Wireless

Utilizing wireless signals inside buildings is a common method for indoor localization. Multiple techniques have been implemented for various applications, and the advantages and drawbacks are described in this section.

2.1.1. Multilateration Method

Unlike GPS and GLONASS, which use satellite signals, indoor navigation systems commonly use wireless signals from access points (APs) or beacons [14]. The location of the receiver is determined by triangulating the differences in signal strengths from several APs or beacons [15–17].

In [18], De Oliveira et al. focused on indoor localization systems for tracking objects in various indoor environments such as retail and logistics. They presented an implementation of a *Bluetooth Low Energy (BLE)*-based localization system using the multilateration and Kalman filter techniques.

In [19], Guidara et al. highlighted expense and flexibility challenges of wired power supplies in dynamic indoor environments. This approach utilizes a battery-powered wireless node to stay in the sleep mode as long as possible to extend the battery life in the indoor localization platform. They employ RSSI data for the multilateration technique and eliminate the need for the time synchronization to lower the implementation cost. However, they need additional devices and preconfigured receivers/APs in the use of multilateration techniques.

The scalability of implementing the wireless signal triangulation approach is challenging, as developers must ensure that any access point (AP) in the building is compatible with the system [14]. The limitation of this approach is that users need to carry additional items with them. Some implementations allow users to only use their smartphones as receivers [20]. The method can be limited in the areas that are covered by APs or beacon signals, as mentioned in [21].

2.1.2. Fingerprinting Method

This approach employs wireless signals to generate a signal strength dictionary of each AP or beacon. This dictionary is then compared with a real-time reading of signal strength. In [22], Pradini et al. used the fingerprinting method for the indoor localization with a small, inexpensive device for the receiver.

In [23], Yuanzhi et al. implemented wireless signal fingerprinting to improve upon previous methods affected by a fluctuation due to human movement. The optimization method increased accuracy and minimized link quality indicator (LQI) reading fluctuation by increasing the fingerprinting section when accuracy was insufficient.

In [24], Polak et al. explored the use of machine learning algorithms for localizations by fingerprinting approaches. They employ *Bluetooth Low Energy (BLE)* for the RSSI-based localization. Their results indicate that BLE causes signal fluctuations. Among machine learning techniques, the *random forest classifier* demonstrates the best performance even with the largest model size.

In [25], Sinha et al. introduced an indoor localization approach using a *Convolutional Neural Network (CNN)* model designed for Wi-Fi-based fingerprinting localization. In the offline stage, the CNN structure is trained to extract features from Wi-Fi signals and create fingerprints. During the online positioning stage, the CNN-based localizer estimates the target's position. However, due to the CNN approach, the real-time localization on a smartphone is difficult.

In [26], Chen et al. proposed an indoor localization method that utilizes magnetic fields for fingerprinting. They paper employs a *Recurrent Probabilistic Neural Network (RPNN)* model to accurately sense magnetic fingerprints during positioning. However, due to the susceptibility of magnetic fields to interferences from other magnetic fields or magnetic storms, the process of creating a fingerprinting map poses a significant challenge.

2.2. *Localization by SLAM Technique*

The *SLAM technique* has recently been used in indoor navigation systems [27,28]. It enables the generation of a map of the surrounding environment in real time and detects the current location in the map. This process usually requires the integration of *LIDAR* to improve the accuracy of the outputs [29]. However, the use of *LIDAR* in smartphones leads to increased battery consumption, reducing battery life and affecting overall device performance.

In the implementation of INSUS, we use *Visual SLAM*. Instead of *LIDAR*, it employs a camera and a gyroscope sensor in a smartphone as the input devices to estimate the user's location. The camera captures a sequence of images of the environment where the user is moving. From each image, feature points such as corners, edges, and horizontal or vertical planes of the environment are extracted to be used as reference points to estimate the current location in the environment through comparing them with their estimated positions from the 3D map at various locations with the specific camera angle that is obtained by the gyroscope. The rotation and orientation data from the gyroscope refer to the pitch (upward or downward rotation around the x-axis), the yaw (leftward or rightward rotation around the z-axis), and the roll (leftward or rightward rotation around the y-axis).

2.3. *3D Modelling Techniques*

The creation of 3D objects is mainly divided into automatic and traditional methods. Automatic methods leverage advanced technologies and algorithms to generate 3D models with minimal human interventions.

In [30], Barhoumi et al. highlighted the significance of 3D object placements in AR applications, particularly for architecture purposes. They utilize an automatic modelling method using a 3D scanner to reconstruct the entire interior and exterior of a building. However, the scanned 3D objects require complex preprocessing, such as cleaning, resampling, filtering, and point cloud registration, before being utilized in AR applications.

Several 3D modelling techniques, including *base modelling*, which produces 3D objects resembling boxes, have been widely used to generate 3D objects with low polygon counts [31]. INSUS can generate the precise 3D virtual environment efficiently using the original object as the reference, where it employs base modelling to produce the 3D virtual environment with low polygon counts.

2.4. Indoor Map Path Planning Techniques

Indoor map path planning is a critical task in indoor navigation systems. Balado et al. introduced a method for path planning using indoor point clouds instead of semantic BIM data [32]. The path planning algorithm used in this method is Dijkstra's algorithm, which takes longer to determine the fastest path to the destination than the A* algorithm [33]. The proposed algorithm is similar to our approach for planning a path to reach the destination location. INSUS employs the A* path-planning algorithm for fast and effective path generation toward the destination.

2.5. User Interface

An interface is essential for promoting effective communication between the users and the application. To display indoor navigation paths without obstructing the user's view of the environment, the interface supports both of these features. AR interfaces accomplish these requirements. Gan et al. developed an AR-based navigation application for emergency evacuation purposes [34]. It utilizes a museum's permanent features as markers for an AR application that activates a virtual agent to guide the user out of the building. The results demonstrate that the system can successfully guide users to safety during an emergency. The proposed interface is similar to our approach of guiding users to reach their destination in indoor spaces. INSUS utilizes AR to display visual guidance in the form of arrays of 3D arrows that direct the users to reach their destination.

2.6. REST API

The Representational State Transfer (REST) architecture is commonly used to provide services for interoperability across multiple contexts and platforms, including the World Wide Web [35]. These architecture's two important characteristics are statelessness and support for cross-platform consumption. It is now a widely accepted standard procedure for providing services online. The micro-service design generally includes REST application programming interfaces (APIs). The REST architecture has been extended through research to accommodate distributed systems. INSUS employs REST API communication protocol to distribute the user's ID, authentication, and navigation data to a database for visualization.

3. Comparing INSUS Techniques with Others Systems

Several implementations of indoor navigation systems have been developed prior to our work. In this section, we aim to compare our implementation with other systems regarding techniques and characteristics.

3.1. Comparison of Techniques Used in Indoor Navigation System

This section provides a comparative analysis of the commonly employed techniques for indoor navigation. This section discusses the benefits and drawbacks of integrating this method into an indoor navigation system.

Table 1 compares the commonly used techniques for indoor navigation systems. The technique includes virtual location, fingerprinting, image matching, and multilateration in terms of various characteristics such as accuracy, consistency, number of devices, complexity, scalability, and implementation cost.

Table 1. Comparison of indoor navigation systems techniques.

	Virtual Location [28]	**Fingerprinting** [22]	**Image Matching** [36]	**Multilateration** [14]
Accuracy	HIGH	HIGH	MODERATE	HIGH
Consistency	MODERATE	HIGH	LOW	MODERATE
Number of Device	1	>1	>1	>1
Complexity	MODERATE	MODERATE	HIGH	HIGH
Scalability	LOW	HIGH	HIGH	HIGH
Implementation Cost	LOW	MODERATE	HIGH	HIGH

Accuracy refers to the level of precision and correctness of each method's obtained location. Fingerprinting, virtual location, and multilateration have high accuracy, while image matching has moderate accuracy.

Consistency is the ability of a technique to provide similar results repeatedly. Fingerprinting has the highest consistency, followed by multilateration, virtual location, and image matching.

Number of devices used in each technique also differs. The virtual location uses one device, while the other techniques require more than one device.

Complexity refers to the level of technical difficulty required to implement each technique. Image matching and multilateration have high complexity, and virtual location and fingerprinting have moderate complexity.

Scalability is a crucial factor in determining the suitability of a technique for indoor navigation systems. It refers to the ability of a technique to accommodate larger indoor environments without compromising its performance. Therefore, ensuring scalability is a significant challenge in the development of indoor navigation systems. Fingerprinting and image matching have high scalabilities, while virtual location and multilateration have low scalability.

Implementation cost refers to the expenses of implementing each technique. The virtual location has the lowest implementation cost, followed by fingerprinting, while image matching and multilateration have high implementation costs.

To conclude, the implementation of virtual location has advantages and disadvantages over earlier indoor navigation methods. One of the advantages of this technique is that the number of devices used is small as well as the ability to apply to new locations or environments easily. However, for the level of consistency in tracking the user's location for navigational purposes, this technique is still inferior compared to other techniques.

3.2. Comparing of Characteristics in Indoor Navigation Products

The following characteristics are presented to sufficiently characterize each approach in comparison with recent related works in the literature:

1. **Adaptive UI**: indicates the capability of the system to use suitable UI for viewing navigation as well as the path.
2. **Walk-in Navigation**: indicates the capability to navigate while also walking.
3. **Zero Additional Devices**: indicates the system operates without any additional devices.
4. **Path Guidelines**: indicates the capability of identifying and displaying the most efficient path from the user's current location to the intended destination.
5. **Multistory Navigation**: indicates the capability of providing multi-floor navigation.

Table 2 compares free and commercialized indoor navigation systems based on characteristics. These systems are compared on five characteristics: Adaptive UI, Walk-in Navigation, Zero Additional Devices, Path Guidelines, and Multistory Navigation. The systems included in the comparison are Navin, IndoorAtlas, InMapz, Situm, Google Maps, and the proposed INSUS.

Navin and InMapz lack the support of adaptive UI and need additional devices. Navin enables multistory navigations, while InMapz lacks walk-in navigations. IndoorAtlas and Situm offer walk-in and multistory navigations. Only IndoorAtlas provides path guidelines.

Google Maps supports multistory navigations and requires no additional devices, but does not provide walk-in navigations or path guidelines.

Table 2. Similar free and commercialized indoor navigation systems.

Products Names	Adaptive UI	Walk-In Navigation	Zero Additional Device	Path Guidelines	Multistory Navigation
Navin [37]	✗	✓	✗	✗	✓
IndoorAtlas [38]	✗	✓	✗	✓	✓
InMapz [39]	✗	✗	✗	✓	✓
Situm [40]	✓	✓	✗	✓	✓
Google Maps [41]	✓	✗	✓	✗	✓
INSUS	✓	✓	✓	✓	✓

✓ denotes the availability of characteristics. ✗ denotes the unavailability of characteristics.

4. System Design

This section provides an overview of the proposed system design.

4.1. Overview

Figure 1 shows an overview of the INSUS system, designed for indoor navigation. The system is implemented in two buildings: the #2 Engineering Building at Okayama University in Japan and the Graduate Building at Politeknik Elektronika Negeri Surabaya. It consists of four essential parts: Input, Unity Game Engine, SEMAR Server, and Output.

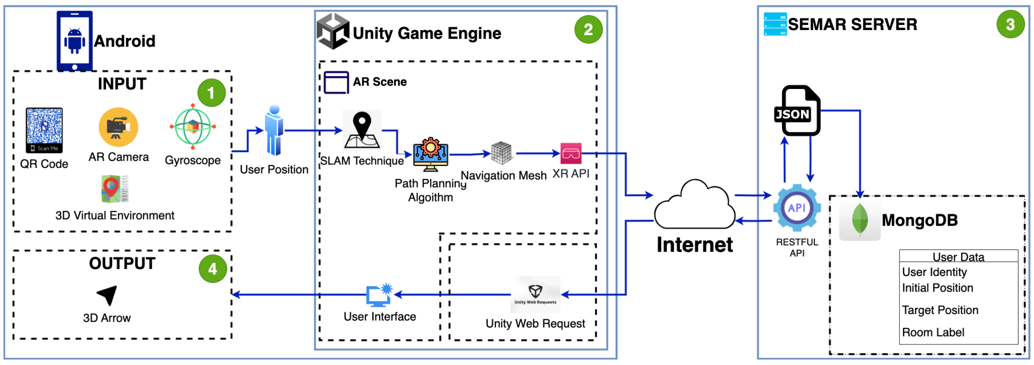


Figure 1. INSUS design.

INSUS consists of four inputs, a gyroscope, a 3D virtual environment, an AR camera, and a QR code. These components operate in parallel to allow the system to function. The gyroscope measures the orientation and rotation of the device, a 3D virtual environment provides a digital map of the surrounding area, an AR camera captures real-time images of the environment, and a QR code allows the system to identify the location of the user. By using these components, the system accurately determines the user's starting position and guides them to their destination.

The indoor navigation system continuously estimates the user's location in real time using a SLAM technique after the initial location is defined. Predefined 3D virtual environments are used by SLAM to localize within the environment. This estimation is carried out by combining information from various sources, a camera, and gyroscopes.

The Unity game engine is a software commonly used for immersive technology research due to its adaptability and the availability of frameworks, libraries, and SDKs that accelerate development. One such tool is the navigation mesh, created by combining a path-planning algorithm with a 3D virtual environment. It allows for effective navigation using the navigation mesh and the 3D virtual environment, with the graphical user interface guiding the user along the path calculated by the algorithm.

The Unity game engine employs the AR core to create augmented reality technologies. This Android-based framework enhances the immersive experience by enabling floor detection and displaying a 3D arrow. The Unity web request is another essential component, enabling systems to connect the internet and process requested data on the server. In this study, we integrate the SEMAR server as a database provider for navigation implementation using REST API communication protocols.

The SEMAR server is connected to the system through the REST API. Within the API, JavaScript Object Notation (JSON) is used to group different data effectively. The data sent to the SEMAR server are logged and recorded in MongoDB for user activity tracking. The system retrieves the stored data through the REST API to display it on the UI of the AR Scene.

The system shows a 3D arrow heading from the user's starting point to their destination. This arrow is part of the user interface and acts as a path guide.

4.2. Input

This section focuses on the input used in the system and how the user ambulates. The input considered is the gyroscope, 3D virtual environment, AR camera, and QR code. Each component's function in the system is explained in the following subsections.

4.2.1. Gyroscope

The gyroscope measures the angle and speed at which an object rotates along an axis, such as the rotation of a camera. The Inertial Measurement Unit (IMU) of INSUS includes a gyroscope that measures the rotation of the camera on the roll, pitch, and yaw angles [42].

The axes of roll, pitch, and yaw of smartphones are depicted in Figure 2; this order of axes follows the Android gyroscope axis rule documentation. However, the device needs to be held in a fixed position when using INSUS to ensure the orientation stays relatively the same as the user's orientation. INSUS utilizes this process to control the rotation of the virtual coordinate in the virtual location. The rotation data of virtual coordinates follow the axes rules of Unity, as the horizontal plane is on the x and z-axis, and the vertical plane is located on the y-axis.

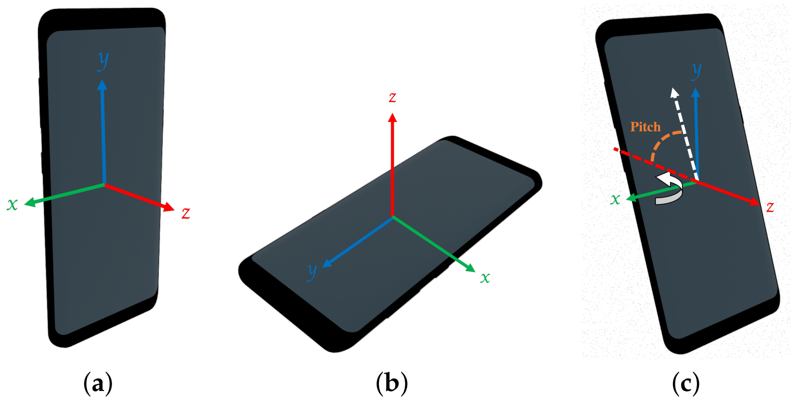


Figure 2. Gyroscope orientation on the smartphone. (**a**) position parallel to y-axis; (**b**) position parallel to x-axis; (**c**) position of the pitch.

4.2.2. 3D Virtual Environment

A 3D virtual environment is a digital representation of a real-world environment that is created through various modelling techniques. The traditional approach to creating a 3D virtual environment involves using a real-world object as a reference. The model is built to accurately reflect the object's shapes, details, and measurements. One advantage of traditional modelling techniques is the lighter 3D virtual environment compared to automated modelling techniques. INSUS utilizes the virtual location to estimate the user's position while traversing the physical environment. Therefore, INSUS needs to configure the 3D virtual environment that represents the map of a building. The 3D virtual environment is created for each floor using Blender software and is configured in the Unity game engine. The 3D virtual environment in this paper consists of four floors, with each room composition shown in Tables 3 and 4.

Table 3. #2 Engineering Building composition.

	1F	**2F**	**3F**	**4F**
Room Target	D101	D201	D301	D401
	D102	D202	D302	D402
	D103	D203	D303	D403
	D104	D204	D304	D404
	D105	D205	D305	D405
	D106	D206	D306	D406
	D107	D207	D307	-
	D108	D208	D308	-
	Toilet	Toilet	Toilet	Toilet
	Restroom	Restroom	Restroom	Restroom
	Elevator	Elevator	Elevator	Elevator
Room as Initial Position	Lobby	D207	D306	D406
Poly count	999	2151	2145	614

Table 4. PENS Graduate Building composition.

	9F	**10F**
Room Target	0901	1001
	0902	1002
	0903	1003
	0904	1004
	0905	1005
	0906	1006
	0907	1007
	0908	1008
	0909	1009
Rooms as Initial Position	0902	1002
	0903	
	0904	
	0908	
	0909	
Poly count	3916	4209

Each room in the building serves as a target goal in experiments. The 3D virtual environment of the building is imported into the Unity Game Engine to facilitate the navigation setting, the initial position, and the target position for each room. Four initial position points are selected. A low-polygon 3D virtual environment is essential in the augmented reality (AR) system, to reduce the GPU usage and optimize the rendering process on limited computing devices such as smartphones. Such low-poly 3D virtual environments achieve real-time rendering of the environment. The 3D virtual environment, with a total tris value of 5909 in the paper, is well below the optimal rendering limit value for the 3D virtual environment.

In the PENS Graduate Building, which consists of 11 floors, INSUS is only implemented for 9th and 10th floors. They have the same number of rooms but with different room order and room size. The initial location that is set in the PENS Graduate Building offers much more variety. Despite the fact that the 3D virtual environment of the PENS Graduate Building is much larger, the number of polygons was successfully kept to a minimum. The total number of polygons is 8125, which is far below the limit value of the tris and vertices stated above.

4.2.3. AR Camera

An augmented reality camera is a device that overlays digital information to the real environment while incorporating types of algorithms to track real-life objects and detect corners and horizontal or vertical planes [43]. This allows virtual elements such as text, images, and 3D models to be seamlessly integrated into the real world, as explained in [44].

INSUS employs an AR camera to perform three primary functions: detecting a floor surface, scanning QR codes, and estimating the user's position using the SLAM Equation (1). The Simultaneous Localization and Mapping (SLAM) technique in the AR camera processes the device's position relative to the surroundings of the environment. It utilizes a control function u_t and a camera as a vision sensor o_t with time-lapse t to estimate the user's state x_t and map of the environment m_t, resulting in the SLAM Equation (1).

$$P(m_{t+1}, x_{t+1} | o_{1:t+1}, u_{1:t}) \tag{1}$$

An AR camera is able to detect floor surfaces or horizontal planes by finding a cluster of feature points that is coplanar and forming a common horizontal plane. This horizontal plane is employed to position the visual guideline of the 3D arrow to reach the destination. To scan the QR code, the AR camera utilizes a computer vision algorithm to decode the black-and-white pattern of the QR code. This information is quickly interpreted by the AR camera system with high accuracy, enabling it to perform actions based on the decoded data. The AR camera employed the SLAM technique, a combination of visual data from the device's camera and inertial data from the device's sensors to estimate the user's position and orientation in the real world. Specifically, the system utilizes feature points in the camera image to estimate the device's position relative to the surroundings, while the device's sensors provide data on the device's motion and orientation.

4.2.4. QR Code

A QR code is a type of barcode that holds data such as URLs, text, and other information. It consists of a grid of black and white squares arranged in a precise pattern recognized by devices with cameras or scanners [45]. QR codes have a higher storage capacity than traditional barcodes to store more complex information patterns [46,47].

In the INSUS system, QR codes are used as labels for rooms and markers for the user's starting position. When a QR code is scanned, the encoded data are sent to the SEMAR server for storage, and the AR scene uses the data to determine the user's initial location by utilizing an AR camera and gyroscope data to generate a path to the destination. To reduce the risk of scanning errors due to similar patterns with almost identical information, the system recommends a specific distance and height for scanning the QR code, as shown in Figure 3.

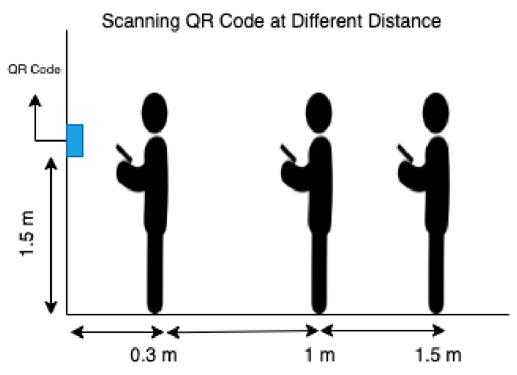


Figure 3. Illustration of user distance to QR code scanning.

To determine the user's starting position, INSUS places several QR codes on each floor. Scanning the QR code requires adjusting the camera's distance from the QR code image. The QR code scanning scenario has two conditions: unrecognized QR code, labeled as "unrecognized room", and recognized QR code, labeled as "n room detected". Figure 3 illustrates that the optimal distance for scanning QR codes is about 0.3 m from the device. Scanning from a distance of less than 1 m is not recommended, as it takes additional time for the camera to focus and increases the risk of incorrectly reading the QR code. Algorithm 1 provides a pseudocode of how the QR code process affects the initial navigation system in INSUS.

Algorithm 1 Pseudocode for Scanning QR code

```
Ensure: $InitCoordinate = 0$
  while $InitCoordinate = 0$ do
    $StartScan()$
    if $Scanned$ then
      print "n room detected"
      $InitCoordinate = 1$
    else
      print "unrecognized room label"
  $SendInitCoordinateToServer()$
  $SendInitCoordinateToARScene()$
```

The QR code process begins with an initial condition where *InitCoordinate* is set to 0 to indicate that there is no saved or existing *InitCoordinate*. The system then enters a *while* loop while *InitCoordinate* is empty to search for the initial location by scanning the QR code using the $StartScan()$ function. If the room label is not recognized, the system prints "unrecognized room label", and the *while* loop continues to run. When the room label is recognized from the QR code, the value of *InitCoordinate* is set to 1, and the system prints "n room detected", breaking the *while* loop. The $SendInitCoordinateToServer()$ and $SendInitCoordinateToARScene()$ functions are then executed outside the *while* loop to send the initial location data to the server and AR Scene via a REST API.

4.3. User Position

Determining the user's position accurately is essential for an indoor navigation system. In INSUS, the user's position is determined using various inputs, including a QR code, AR camera, gyroscope, and 3D virtual environment. The QR code is employed as a starting point for the user's position. When the user scans the QR code using their smartphone's camera, the INSUS recognizes the code and uses it to establish the user's initial position.

Once the initial position is determined, the AR camera detects and tracks the user's surroundings in real time. It is used to track the user's movements and adjust their position accordingly. The gyroscope is utilized to track the user's orientation and rotation, then combined with the position data obtained from the AR camera to provide a more accurate user position.

Thus, the 3D virtual environment is used to provide a reference point for the user's position. By comparing the user's position relative to the virtual environment, INSUS adjusts the user's position and orientation as necessary to ensure accurate navigation.

4.4. Unity

Unity is a cross-platform game engine developed by Unity Technologies. It is used to create video games and other interactive content such as architectural visualizations, real-time 3D animations, and simulations. Unity provides imitations of real-life physical principles, including artificial intelligence. Unity provides simple functionality and extensive documentation for designing and researching AR mobile applications. The AR scene, which includes pathfinding, a 3D virtual environment setup, and a user interface, is integrated within Unity to build an AR application. To connect with the SEMAR server, which serves as the database of the INSUS, Unity provides the Unity Web Request library to establish a connection with the SEMAR server via the REST API and transmit data in

JSON format. Therefore, the use of Unity web request enables the transmission of data between the mobile application and the server.

4.4.1. Pathfinding

Pathfinding is the process of finding a route between two points in a game or simulation environment. It is commonly used to enable objects to move around and navigate the environment in a realistic and efficient manner. In INSUS, the indoor navigation settings are configured using a Navigation Mesh (NavMesh) data structure. NavMesh is a navigation library in the Unity game engine that provides path planning and pathfinding for games. It is created by defining the walkable area from a 3D virtual environment. In order to use NavMesh for navigation inside the building, a 3D virtual environment of the building is created with precise measurements.

Figure 4 shows a virtual environment processed using the NavMesh tool. The surface covered with blue mesh is a walkable area, while the walls not covered with mesh constitute an inaccessible area. Each floor has various rooms that are chosen as destinations through the INSUS user interface. To determine the optimal path between rooms, INSUS employs the A* algorithm, a widely used Best First Search (BFS) method that efficiently calculates the shortest route while minimizing cost. The A* algorithm relies on an evaluation function that takes into account factors such as distance and obstacles to determine the best path. The A* algorithm uses an evaluation function:

$$f(n) = g(n) + h(n) \tag{2}$$

where

$g(n)$ is the cost to reach the goal from n;
$h(n)$ is the estimated cost of the heuristic from the goal to n;
$f(n)$ is the estimated total cost of the path from n to the goal.

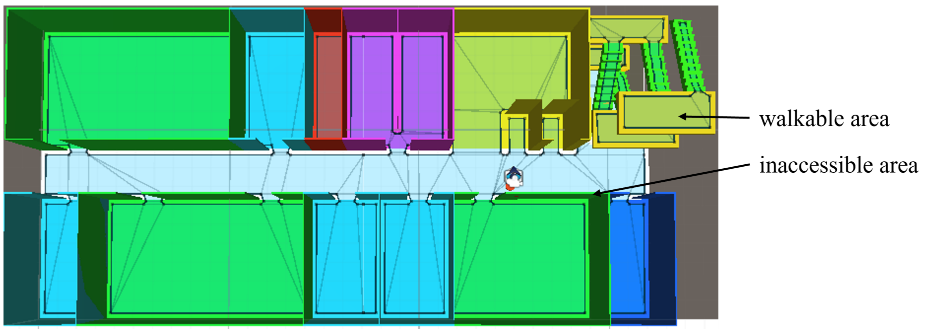


Figure 4. Navigation mesh in 3D virtual environment.

In the A* algorithm, the $g(n)$ function is influential in obtaining a faster path than other BFS methods. For example, Table 5 is the value of $h(n)$ from the initial position of room D207 to D202.

Table 5. Straight-line distance to D202.

Straight Distance to Room D202									
Rooms	D207	D206	D205	D203	D201	D202	D204	EPS	Toilet
$h(n)$	21.55	20.50	16.70	12.70	2.50	0	10.70	13.28	16.90

Based on calculations using the A* evaluation function, three possible paths were taken: D206, Toilet, and D202. From the analysis of the A* algorithm, the closest path from node A to H is obtained, with an estimated total cost of the path from D207 to D202 reaching a $f(n)$ value of 21.55 m. Figure 5 illustrates finding the shortest path from the initial room "D207" to the target room "D202". The relation between nodes D207 and D202 has the shortest path with a lower cost when compared to the path selection in Figure 6.

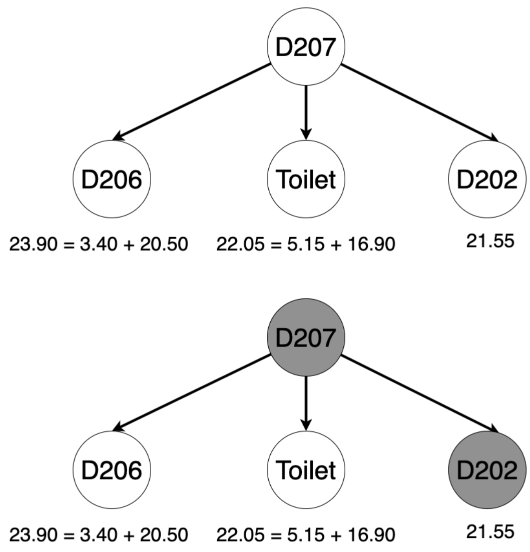


Figure 5. A* algorithm shortest path calculation.

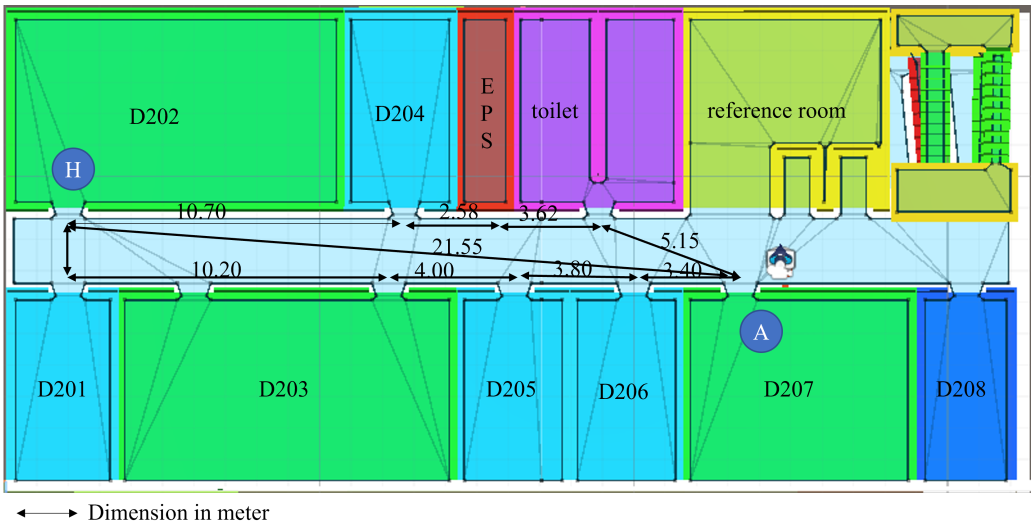


Figure 6. A* algorithm finding shortest path from node A to node H.

4.4.2. AR Scene

The AR Scene in this study involves the use of augmented reality (AR) technology to display virtual elements such as messages, photos, and 3D models in the real world through devices such as smartphones, tablets, or AR headgear. To implement AR scenarios, the Unity Game Engine and ARCore SDK for Android phones are introduced. ARCore is an SDK with predefined functions and libraries for object tracking, surface detection, lighting, and shadow effects [48]. ARCore is capable of detecting and tracking the user's surroundings in real time by employing the SLAM technique.

4.4.3. User Interface

A user interface (UI) is part of a computer system that allows users to interact with the system. It includes elements such as graphical elements (e.g., icons, buttons, menus), texts, and input fields, as well as visual and audio feedback, which enable the user to perform tasks and access information [49]. The design and layout of a user interface are critical to the usability and effectiveness of a system. The user interface of INSUS has been designed to be user-friendly, visually attractive, and intuitive, facilitating users in reaching their destination effortlessly and promptly.

INSUS is a complex system that utilizes various components and interfaces to enable indoor navigation. The user interface is divided into three components, including the initial position scene and the navigation scene. These UIs are displayed in Figure 7.

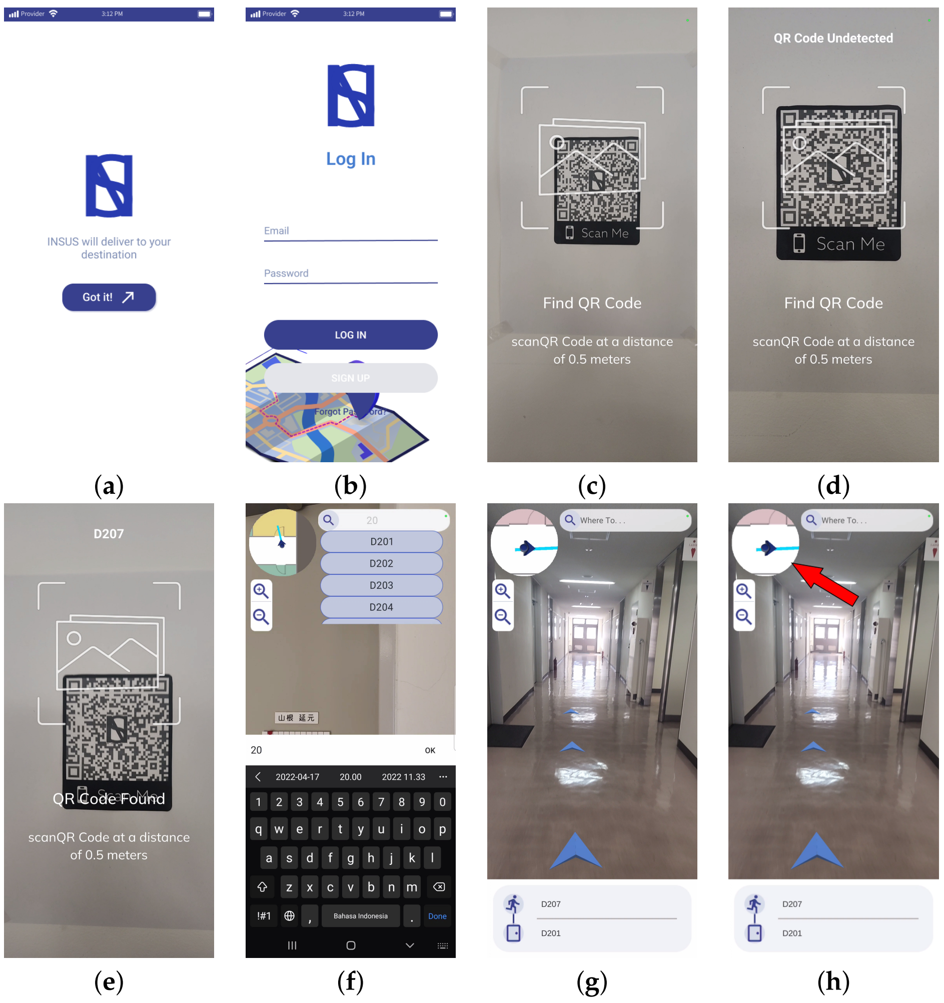


Figure 7. INSUS user interface. (**a**) initial UI; (**b**) log in menu UI; (**c**) scanning QR code UI; (**d**) failed to read QR code message; (**e**) QR code successfully scanned; (**f**) entering destination; (**g**) 3D arrow shows path to destination; (**h**) minimap of the INSUS pointed by the red arrow.

The UI scene is the first scene that users encounter when they open the INSUS application. It consists of a 2D interface that facilitates user introduction, registration, sign-in, and password reset, which are necessary steps that the user must complete before running the application. Figure 7a,b display the UIs for this scene.

The initial position scene is where users determine their initial position by scanning the QR code displayed on the building's walls. When the user fails to follow the scanning protocol, an incorrect reading or no reading is displayed, as shown in Figure 7d; successful QR code scanning results of the recognized room label and the user's initial location are shown in Figure 7e.

In the navigation scene, users can enter the desired target rooms in a search column, as shown in Figure 7f. The system uses an A* algorithm to determine the shortest distance between the user's current position and the target room. The corresponding path is displayed as a 3D arrow, which guides the user to their destination, as shown in Figure 7g. The system imports a 2D map that serves as a mini-map, displaying the user's current position in the building, as shown in Figure 7h.

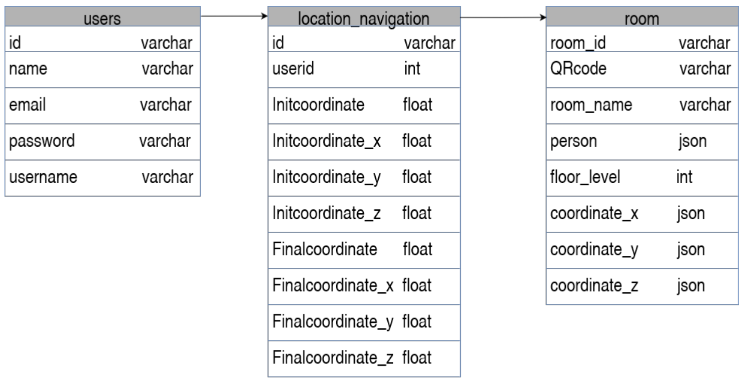
4.4.4. Unity Web Request

Unity Web Request is a feature of the Unity Game Engine that allows applications to send and receive HTTP requests and responses from web servers. It is commonly used to retrieve or send data to a server. Unity Web Request supports both GET and POST requests, which is a suitable tool for sending and receiving data through a REST API, such as the user's initial location in INSUS. With minimal setup required in the Unity game engine, Unity Web Request is a simple and efficient way to connect INSUS to a server, such as a SEMAR server.

4.5. SEMAR Server

The SEMAR server employs the MQTT or REST API communication protocol [50]. The SEMAR server's function is utilized to expedite and simplify the development process. The SEMAR server stores the current location data, the initial location, and the target location with the user's ID.

The API enables two or more platforms to communicate with each other. This system employs GET/POST operations to manage the data from the server in the application individually. In the database design, there is a relationship between each entity; as shown in Figure 8, each user may have multiple target rooms (1:N). Each position has x, y, and z coordinate values, where the x and z values represent horizontal plane displacements, and the y value represents vertical plane displacement. Each room has a different coordinate value. The data are sent to the SEMAR server in the JSON format.



users	
id	varchar
name	varchar
email	varchar
password	varchar
username	varchar

location_navigation	
id	varchar
userid	int
Initcoordinate	float
Initcoordinate_x	float
Initcoordinate_y	float
Initcoordinate_z	float
Finalcoordinate	float
Finalcoordinate_x	float
Finalcoordinate_y	float
Finalcoordinate_z	float

room	
room_id	varchar
QRcode	varchar
room_name	varchar
person	json
floor_level	int
coordinate_x	json
coordinate_y	json
coordinate_z	json

Figure 8. relationship of data sent to the SEMAR server.

4.6. Output

In order for INSUS to function properly as an AR application for assisting users ambulation, after determining the shortest path from the initial location to the target location using the A* algorithm, a 3D arrow is displayed to guide users to reach their destination. The 3D arrow appears once the floor is detected by an AR camera and localized by the SLAM technique. The distance between each 3D arrow is defined by the target location and the current location, as shown in Figure 7g.

4.7. User Experience Measurement

To evaluate the user experience of INSUS, we utilized the PIECES framework and a Likert scale. PIECES stands for Performance, Information and Data, Economics, Control and Security, Efficiency, and Service. These six dimensions are commonly used in evaluating the usability and effectiveness of interactive systems.

We designed a survey based on the PIECES framework and asked participants to rate their experience with INSUS on a Likert scale. The Likert scale is a widely used rating scale in social science research that measures attitudes or perceptions on a continuum from strongly agree to strongly disagree. We used a five-point Likert scale, where 1 indicated "strongly disagree" and 5 indicated "strongly agree".

The results of the survey were analyzed to determine the strengths and weaknesses of INSUS in each dimension of the PIECES framework. This information is used to improve the system and enhance the user experience.

5. Evaluation and Discussion

The system's performance was evaluated and compared by conducting tests in two different buildings: the #2 Engineering Building at Okayama University and the PENS Graduate Building.

5.1. Device Requirements and Scenario

The devices were selected based on the specifications recommended by Google's ARCore. Table 6 shows the specifications of the devices used for the experiment. Our experiments followed the predefined scenarios to find the best practice for scanning the QR code and to evaluate the navigation accuracy. During each experiment, to consider the conventional use of a smartphone, disturbances such as random smartphone shaking or walking speed changes were introduced to evaluate the robustness of the system.

Table 6. Device secification.

Component	**Specification**			
	PENS Graduate Building		**#2 Engineering Building**	
Device	Samsung S22	Samsung S9+	Samsung S22 Ultra	Realme 9 Pro+
OS	Android	Android	Android	Android
Chipset	Snapdragon 8 Gen 1	Exynos 9810	Snapdragon 8 Gen 1	Mediatek Dimensity 920
GPU	Adreno 730	Mali-G72 MP18	Adreno 730	Mali-G68
RAM	8 GB	6 GB	12 GB	8 GB
LCD	2340 × 1080 pixels	2960 × 1440 pixels	1440 × 3088 pixels	2400 × 1080
Refresh Rate	120 Hz	60 Hz	120 Hz	90 Hz

5.2. QR Code Scanning Practice

First, we conducted experiments by scanning the QR code with different relative angles of the smartphone to the QR code to examine the best practice. The five cases of 30°, 45°, 60°, 90°, and 100° were considered, as shown in Figure 9. For each case, the QR code detection accuracy was evaluated in the two buildings with the two smartphones.

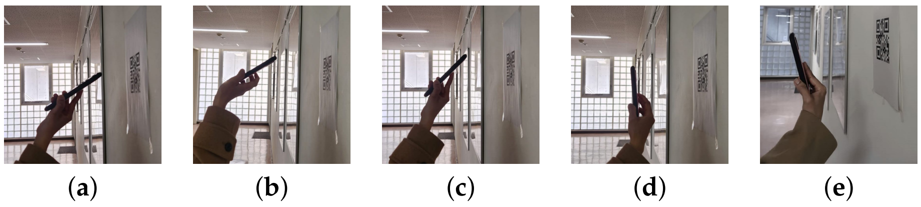


Figure 9. Smartphone angles in QR code detection experiments. (**a**) QR test on viewing angle on 30°; (**b**) QR test on viewing angle on 45°; (**c**) QR test on viewing angle on 60°; (**d**) QR test on viewing angle on 90°; (**e**) QR test on viewing angle on 100°.

Table 7 shows the results of QR code detection experiments. The table indicates that when the smartphone angle is between 60° to 100°, it achieves an accuracy of more than 80% for both smartphones tested. The vertical position produces a more predictable and consistent orientation of the QR code detection; the camera sensor captures the QR code and processes it to extract the information. Excessive tilting of the smartphone (i.e., 30° and 45°) distorts or obscures the QR code image and makes it difficult to recognize. Thus, we recommend that users position their smartphones at an angle between 60° and 100° to achieve the highest QR code detection accuracy. This optimal range for the smartphone camera captures a clear and undistorted QR code image.

Table 7. QR code detection accuracy under different angles.

Scenario (Angle)	PENS Graduate Building		#2 Engineering Building	
	Samsung S22	Samsung S9+	Samsung S22 Ultra	Realme 9 Pro+
30°	82.8%	74.2%	88.5%	77.1%
45°	88.5%	77.1%	91.4%	82.8%
60°	91.4%	82.8%	94.2%	88.5%
90°	100%	94.2%	100%	91.4%
100°	100%	100%	100%	100%

5.3. Navigation Accuracy

Second, we conducted an experiment to assess the navigation success rate across multiple floors using camera and gyroscope data to estimate the user's current location. To evaluate the navigation success rate, the navigation of 6 cases of different angles, 0, 30, 45, 60, 90, and 100 °, was considered, as shown in Figure 10. The experimental process introduces disturbance in the form of smartphone shaking and walking speed.

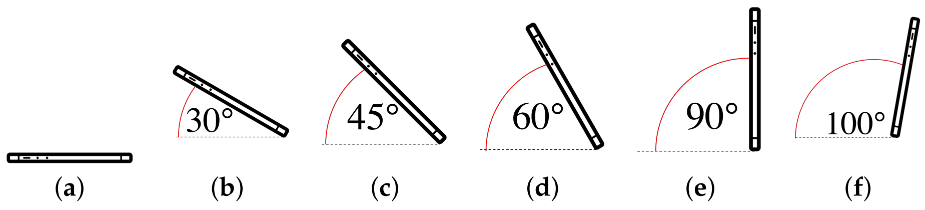


Figure 10. Smartphone angles in navigation experiments. (**a**) 0°; (**b**) 30°; (**c**) 45°; (**d**) 60°; (**e**) 90°; (**f**) 100°.

Table 8 shows the results of the navigation experiments where the SLAM technique was used for indoor navigation. The table indicates that the average navigation success rate for indoor navigation in both scenarios, with disturbances and without disturbances, is higher than 80% when the smartphone is held upright between an angle of 60° and 100° degrees. This range enables the camera to understand the surrounding environment clearly. On the other hand, angles in the range of 0° to 45° resulted in poor navigation accuracy. This is due to the SLAM's failure to detect any changes or features in the surrounding environment, as it is facing downward towards the floor. As a result, the SLAM technique is not able to estimate the user's position accurately at this angle. Thus, we recommend that the users orient their smartphones at an angle between 60° and 100° to achieve the highest navigation success rate. This preferable range enables the smartphone camera to capture the surrounding environment's feature points and measures the gyroscope device rotation and orientation data.

Table 8. Navigation success rates with disturbances and without disturbances.

Success Rate		with Disturbance						without Disturbance					
		Angles											
		0°	30°	45°	60°	90°	100°	0°	30°	45°	60°	90°	100°
PENS Graduate Building	9F	57%	64%	79%	92%	100%	100%	79%	86%	86%	92%	100%	100%
	10F	57%	71%	79%	86%	100%	100%	71%	79%	86%	100%	100%	100%
#2 Engineering Building	1F	57%	64%	79%	86%	100%	100%	79%	86%	86%	100%	100%	100%
	2F	57%	64%	79%	86%	100%	100%	71%	79%	79%	100%	100%	100%
	3F	50%	57%	71%	86%	100%	100%	71%	71%	79%	86%	100%	100%
	4F	57%	64%	71%	79%	100%	100%	71%	79%	79%	92%	100%	100%
Average		55.8%	64%	76.3%	85.8%	100%	100%	73.6%	80%	82.5%	95%	100%	100%

5.4. User Experience

To evaluate the user experience after the implementation of INSUS, measurements were conducted to determine the user response. The study participants were students who were in the #2 Engineering Building at Okayama University and PENS Graduate Building. Several questions were asked to the users, including the system's accuracy, usability, simplicity of use, intention to use it, and suggestions for improvement. The obtained results are presented in Table 9.

Table 9. User experience results.

Questions	PENS Graduate Building		#2 Engineering Building	
	Number of Students Use This System			
	Agree	Disagree	Agree	Disagree
This system is accurate	30	0	20	0
This system is useful	22	8	18	2
Want to use more	23	7	15	5
This system easy to use	26	4	17	3
Improve systems	28	2	20	0

Table 9 presents the user experience results, showing that most participants agreed that INSUS is accurate, useful, and easy to use. However, some participants disagreed and did not intend to use the system again. Additionally, the participants provided suggestions for system's improvement.

In the PENS Graduate Building, 30 participants agreed that INSUS is accurate, while 22 found it useful. On the other hand, seven participants disagreed with its usefulness and did not intend to use it again. In the #2 Engineering Building, 20 participants agreed that INSUS is accurate, and 18 found it useful. However, five participants disagreed with the system's usefulness and did not intend to use it again.

Overall, 100% of the total respondents from both buildings agreed that INSUS is accurate, while 80% agreed that it is useful, 76% agreed that they want to use it again, 86% agreed that it is easy to use, and 96% agreed that it can still be improved. Therefore, this study found that INSUS received positive responses from users in both buildings.

6. Conclusions

This paper presented the design and implementation of the *Indoor Navigation System using Unity and Smartphone (INSUS)*. It utilizes the SLAM technique with a gyroscope and a camera in a smartphone to track users' movements inside a building after initializing the current location by the QR code. *Unity* is introduced to obtain the 3D information of the target indoor environment for *Visual SLAM*. The data are stored in the IoT application server called *SEMAR* for visualizations.

The experiments were conducted in two buildings to determine the optimal QR code scanning practice, the navigation accuracy, and the user experience. The results showed that a smartphone angle in the range of 60° to 100° provides the highest QR code detection accuracy and the range of 90° to 100° provides the highest navigation accuracy. As for the user experience, positive feedback was obtained from questionnaire responses.

In future works, we will improve the navigation accuracy by using other information, such as GPS position and landmark object recognition in the environment, and will integrate INSUS with outdoor navigation systems.

Author Contributions: Conceptualization, E.D.F., N.F. and S.S.; Methodology, E.D.F., N.F. and S.S.; Software, E.D.F., Y.Y.F.P., K.D., F.S. and A.A.S.P.; Investigation, K.D. and F.S.; Writing—original draft, E.D.F.; Writing—review & editing, N.F.; Supervision, N.F. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sáez, Y.; Montes, H.; Garcia, A.; Muñoz, J.; Collado, E.; Mendoza, R. Indoor Navigation Technologies Based on RFID Systems to Assist Visually Impaired People: A Review and a Proposal. *IEEE Lat. Am. Trans.* **2021**, *19*, 1286–1298. [CrossRef]
2. Li, N.; Guan, L.; Gao, Y.; Du, S.; Wu, M.; Guang, X.; Cong, X. Indoor and outdoor low-cost seamless integrated navigation system based on the integration of INS/GNSS/Lidar System. *Remote Sens.* **2020**, *12*, 3271. [CrossRef]
3. Anjum, M.; Abdullah Khan, M.; Hassan, S.A.; Jung, H.; Dev, K. Analysis of time-weighted Lora-based positioning using machine learning. *Comput. Commun.* **2022**, *193*, 266–278. [CrossRef]
4. Li, Y.; Zhuang, Y.; Hu, X.; Gao, Z.; Hu, J.; Chen, L.; He, Z.; Pei, L.; Chen, K.; Wang, M.; et al. Location-Enabled IoT (LE-IoT): A Survey of Positioning Techniques, Error Sources, and Mitigation. *arXiv* **2020**, arXiv:2004.03738.
5. Nessa, A.; Adhikari, B.; Hussain, F.; Fernando, X.N. A Survey of Machine Learning for Indoor Positioning. *IEEE Access* **2020**, *8*, 214945–214965. [CrossRef]
6. Zhang, Z.; Du, H.; Choi, S.; Cho, S.H. TIPS: Transformer Based Indoor Positioning System Using Both CSI and DoA of WiFi Signal. *IEEE Access* **2022**, *10*, 111363–111376. [CrossRef]
7. Gadhgadhi, A.; Hachaichi, Y.; Zairi, H. A machine learning based indoor localization. In Proceedings of the 2020 4th International Conference on Advanced Systems and Emergent Technologies (IC_ASET), Hammamet, Tunisia, 15–18 December 2020. [CrossRef]
8. Rustagi, T.; Yoo, K. Indoor AR navigation using tilesets. In Proceedings of the 24th ACM Symposium on Virtual Reality Software and Technology, Tokyo, Japan, 28 November–1 December 2018; pp. 1–2.
9. Selin, E. 10 Different Types of 3D Modeling Techniques. 2021. Available online: https://artisticrender.com/10-different-types-of-3d-modeling-techniques/ (accessed on 2 February 2023).
10. Asraf, O.; Shama, F.; Klein, I. PDRNet: A Deep-Learning Pedestrian Dead Reckoning Framework. *IEEE Sensors J.* **2022**, *22*, 4932–4939. [CrossRef]
11. Zhou, Z.; Feng, W.; Li, P.; Liu, Z.; Xu, X.; Yao, Y. A fusion method of pedestrian dead reckoning and pseudo indoor plan based on conditional random field. *Measurement* **2023**, *207*, 112417. [CrossRef]
12. Zhao, J.; Xu, Q.; Zlatanova, S.; Liu, L.; Ye, C.; Feng, T. Weighted octree-based 3D indoor pathfinding for multiple locomotion types. *Int. J. Appl. Earth Obs. Geoinf.* **2022**, *112*, 102900. [CrossRef]
13. Chidsin, W.; Gu, Y.; Goncharenko, I. AR-based navigation using RGB-D camera and hybrid map. *Sustainability* **2021**, *13*, 5585. [CrossRef]
14. Chu, E.T.H.; Wang, S.C.; Chang, C.C.; Liu, J.W.S.; Hsu, J.; Wu, H.M. WPIN: A waypoint-based indoor navigation system. In Proceedings of the IPIN (Short Papers/Work-in-Progress Papers), Pisa, Italy, 30 September–3 October 2019; pp. 187–194.
15. Wichmann, J. Indoor positioning systems in hospitals: A scoping review. *Digit. Health* **2022**, *8*, 20552076221081696. [CrossRef]
16. Mackey, A.; Spachos, P.; Plataniotis, K.N. Enhanced Indoor Navigation System with beacons and Kalman filters. In Proceedings of the 2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP), Anaheim, CA, USA, 26–29 November 2018. [CrossRef]
17. Sangthong, J. The indoor navigation using mapping technique based on signal strength difference. In Proceedings of the 2018 21st International Symposium on Wireless Personal Multimedia Communications (WPMC), Chiang Rai, Thailand, 25–28 November 2018. [CrossRef]
18. De Oliveira, L.S.; Rayel, O.K.; Leitao, P. Low-cost indoor localization system combining multilateration and kalman filter. In Proceedings of the 2021 IEEE 30th International Symposium on Industrial Electronics (ISIE), Kyoto, Japan, 20–23 June 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 1–6.
19. Guidara, A.; Derbel, F.; Fersi, G.; Bdiri, S.; Jemaa, M.B. Energy-efficient on-demand indoor localization platform based on wireless sensor networks using low power wake up receiver. *Ad Hoc Netw.* **2019**, *93*, 101902. [CrossRef]
20. Ashraf, I.; Hur, S.; Park, Y. Smartphone sensor based indoor positioning: Current status, opportunities, and future challenges. *Electronics* **2020**, *9*, 891. [CrossRef]
21. Huang, H.; Zeng, Q.; Chen, R.; Meng, Q.; Wang, J.; Zeng, S. Seamless navigation methodology optimized for indoor/outdoor detection based on WIFI. In Proceedings of the 2018 Ubiquitous Positioning, Indoor Navigation and Location-Based Services (UPINLBS), Wuhan, China, 22–23 March 2018. [CrossRef]
22. Puspitaningayu, P.; Funabiki, N.; Huo, Y.; Hamazaki, K.; Kuribayashi, M.; Kao, W.C. Application of fingerprint-based indoor localization system using IEEE 802.15.4 to two-floors environment. In Proceedings of the 2022 IEEE 4th Global Conference on Life Sciences and Technologies (LifeTech), Osaka, Japan, 7–9 March 2022. [CrossRef]
23. Huo, Y.; Puspitaningayu, P.; Funabiki, N.; Hamazaki, K.; Kuribayashi, M.; Kojima, K. A proposal of the fingerprint optimization method for the fingerprint-based indoor localization system with IEEE 802.15.4 devices. *Information* **2022**, *13*, 211. [CrossRef]
24. Polak, L.; Rozum, S.; Slanina, M.; Bravenec, T.; Fryza, T.; Pikrakis, A. Received signal strength fingerprinting-based indoor location estimation employing machine learning. *Sensors* **2021**, *21*, 4605. [CrossRef] [PubMed]
25. Sinha, R.S.; Hwang, S.H. Comparison of CNN applications for RSSI-based fingerprint indoor localization. *Electronics* **2019**, *8*, 989. [CrossRef]

26. Chen, C.H.; Chen, P.W.; Chen, P.J.; Liu, T.H. Indoor Positioning Using Magnetic Fingerprint Map Captured by Magnetic Sensor Array. *Sensors* **2021**, *21*, 5707. [CrossRef]
27. Deng, Y.; Ai, H.; Deng, Z.; Gao, W.; Shang, J. An overview of indoor positioning and mapping technology standards. *Standards* **2022**, *2*, 157–183. [CrossRef]
28. Gerstweiler, G.; Vonach, E.; Kaufmann, H. Hymotrack: A Mobile AR navigation system for complex indoor environments. *Sensors* **2015**, *16*, 17. [CrossRef]
29. Zou, Q.; Sun, Q.; Chen, L.; Nie, B.; Li, Q. A Comparative Analysis of LiDAR SLAM-Based Indoor Navigation for Autonomous Vehicles. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 6907–6921. [CrossRef]
30. El Barhoumi, N.; Hajji, R.; Bouali, Z.; Ben Brahim, Y.; Kharroubi, A. Assessment of 3D Models Placement Methods in Augmented Reality. *Appl. Sci.* **2022**, *12*, 10620. [CrossRef]
31. Prithal. Different Types of 3D Modelling. 2021. Available online: https://xo3d.co.uk/different-types-of-3d-modelling/ (accessed on 23 January 2023).
32. Balado, J.; Díaz-Vilariño, L.; Arias, P.; Frías, E. Point clouds to direct indoor pedestrian pathfinding. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2019**, *XLII-2/W13*, 753–759. [CrossRef]
33. Candra, A.; Budiman, M.A.; Hartanto, K. Dijkstra's and A-Star in Finding the Shortest Path: A Tutorial. In Proceedings of the 2020 International Conference on Data Science, Artificial Intelligence, and Business Analytics (DATABIA), Medan, Indonesia, 16–17 July 2020; pp. 28–32. [CrossRef]
34. Gan, Q.; Liu, Z.; Liu, T.; Chai, Y. An indoor evacuation guidance system with an AR virtual agent. *Procedia Comput. Sci.* **2022**, *213*, 636–642. [CrossRef]
35. Ehsan, A.; Abuhaliqa, M.A.; Catal, C.; Mishra, D. RESTful API testing methodologies: Rationale, challenges, and solution directions. *Appl. Sci.* **2022**, *12*, 4369. [CrossRef]
36. Wang, X.; Qin, D.; Guo, R.; Zhao, M.; Ma, L.; Berhane, T.M. The technology of crowd-sourcing landmarks-assisted smartphone in indoor localization. *IEEE Access* **2020**, *8*, 57036–57048. [CrossRef]
37. Navin. Indoor and Outdoor Navigation. 2013. Available online: https://nav-in.com/ (accessed on 5 February 2023).
38. IndoorAtlas. Indooratlas API Documentation. 2012. Available online: https://docs.indooratlas.com/apidocs/ (accessed on 5 February 2023).
39. InMapz. Inmapz Home. 2016. Available online: https://inmapz.com/ (accessed on 5 February 2023).
40. Situm. 01—Introduction Archives. 2014. Available online: https://situm.com/docs-category/changelogs/ (accessed on 7 February 2023).
41. Google. Google Indoor Map. 2018. Available online: https://www.google.com/maps/about/partners/indoormaps (accessed on 5 February 2023).
42. Lemoyne, R.; Mastroianni, T. Implementation of a Smartphone as a Wearable and Wireless Gyroscope Platform for Machine Learning Classification of Hemiplegic Gait Through a Multilayer Perceptron Neural Network. In Proceedings of the 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), Orlando, FL, USA, 17–20 December 2018; pp. 946–950. [CrossRef]
43. Taira, G.M.N.; Sementille, A.C.; Sanches, S.R.R. Influence of the Camera Viewpoint on Augmented Reality Interaction. *IEEE Lat. Am. Trans.* **2018**, *16*, 260–264. [CrossRef]
44. Tang, F.; Wu, Y.; Hou, X.; Ling, H. 3D Mapping and 6D Pose Computation for Real Time Augmented Reality on Cylindrical Objects. *IEEE Trans. Circuits Syst. Video Technol.* **2020**, *30*, 2887–2899. [CrossRef]
45. Chou, G.J.; Wang, R.Z. The Nested QR Code. *IEEE Signal Process. Lett.* **2020**, *27*, 1230–1234. [CrossRef]
46. Eugênio Gonçalves, H.; Xavier Medeiros, L.; Coutinho Mateus, A. Algorithm for Locating the Vertices of a QR Code and Removing Perspective. *IEEE Lat. Am. Trans.* **2021**, *19*, 1933–1940. [CrossRef]
47. Huang, P.C.; Chang, C.C.; Li, Y.H.; Liu, Y. Efficient QR Code Secret Embedding Mechanism Based on Hamming Code. *IEEE Access* **2020**, *8*, 86706–86714. [CrossRef]
48. GoogleDeveloper. Fundamental Concepts/ARcore/GoogleDeveloper. Available online: https://developers.google.com/ar/develop/fundamentals (accessed on 5 February 2023).
49. Iqbal, M.W.; Ch, N.A.; Shahzad, S.K.; Naqvi, M.R.; Khan, B.A.; Ali, Z. User Context Ontology for Adaptive Mobile-Phone Interfaces. *IEEE Access* **2021**, *9*, 96751–96762. [CrossRef]
50. Panduman, Y.Y.; Funabiki, N.; Puspitaningayu, P.; Kuribayashi, M.; Sukaridhoto, S.; Kao, W.C. Design and implementation of SEMAR IOT server platform with applications. *Sensors* **2022**, *22*, 6436. [CrossRef] [PubMed]