POLITECNICO DI TORINO

Master’s Degree in Computer Engineering

Master’s Degree Thesis

Indoor Navigation based on the plain browser.

Supervisor
Prof. Malnati Giovanni

Candidate
Elchin Fahradov

July 2022

Abstract

Today finding any destination in an unfamiliar environment is a struggling procedure for all people. Therefore, navigation has become one of the main problems from the earliest days of human history. In the early periods, people were using different techniques for navigation. Afterward with than improvement of technology, GPS has been innovated and become an inseparable part of our daily life. However, it’s fact that today's definition of navigation is not only about the outdoor environment. With the development of architectural structures, humankind begins to create massive buildings with all the necessary facilities for humans. This change revealed the indoor navigation problem for us. Unfortunately, GPS technology cannot help us in the indoor environment. This is because GPS technology signals, when used indoors, can be very weak when passing through walls or may not pass at all. Therefore, we need a new approach to finding the destination we are looking for in a closed area. And indoor navigation technology is emerging to solve this problem. In our study, we made possible analysis and research for the use of Navigation systems for indoor infrastructures. In the initial stage, we did research about already created indoor navigation technologies with their working principles, advantages, and disadvantages. As a result, we have concluded that most of the existed indoor navigation technology require more resource and cost while configuring the building. After getting familiar with existing solutions, we introduced our methodology for the given problem.

Our methodology is a visual-based navigation system. The visual-based navigation system uses the device's camera to help scan markers or objects in the environment to determine orientation and position. The QR code was used as a visual

marker in our methodology due to its distinctive nature which helped us for solving the direction and position problem.

While doing our project the first and main struggle we have faced is to convert world point coordinates into camera coordinates. Since it's fact that objects are recognized in 3D in the real world, but in 2D on computer screens. Therefore, it is important to convert and display the 3D image on a 2D screen. This problem is known as PnP (Perspective n Point) problem. In our paper, we have analyzed this problem in a more detailed way and with possible solutions. To solve this problem, I used Rust's Lambda Twist library. It is one of the fastest and most accurate libraries of all other PnP solvers. Furthermore, as a helper library to Lambda Twist, I have used Rqrr which helps to detect QR codes with their border coordinates. These coordinates are sent to the Lambda Twist library to convert from 3D world points to the 2D coordinate system. In the end, we have applied the Dijkstra algorithm to the data we store in our database that contains the set of QR code positions with different properties. As a result, we find the shortest path, start, and destination point with its accurate direction. As an experiment, we created an imaginary map with unique coordinates and tested our application with different source and destination points. Based on the experiments and studies, we can conclude that our methodology can apply to all types of infrastructure with low cost and high performance.

Acknowledgements

Without the kind assistance and support of those who have helped me throughout the project, I would not have been able to work on this project. First of all, I would like to thank my thesis advisor, Prof. Giovanni Malnati, at Polytechnic di Torino. Prof. Malnati always assisted me with great enthusiasm and tolerance when I had questions about my research and project. Also, I would especially like to express my gratitude to my brother Tural Farhadov and my friends, Vugar Hasanov, Dorota Palma and Nijat Gurbanov. They have consistently been a great source of encouragement when things get struggling. Finally, I must express how grateful I am to my parents for their unwavering support and ongoing encouragement during my years of study and writing this thesis.

Table of Contents

1. Introduction

1.1. Background

Nowadays navigation is one of the most inseparable parts of our modern life. Every day, millions of people use GPS (Global Positioning System) systems that provide outdoor navigation to make their daily tasks easier for navigating where they want to go. It helps us in different situations and guides us to find the specific city, town, or exact location inside the city with the different directions and distances. The use of GPS technology makes this functionality easily reachable by all people using most smart gadgets. However, location-related actions such as navigation are not limited to the outdoors. It's obvious in the modern world the main tendency is to create enormous buildings that contain everything necessary for human beings. Besides, people tend to spend more time inside. According to the research done by the United States Environmental Agency (EPA, 2021) the average American spends 90% of their time indoors. The increment in the number of people and the development of architectural technology and the growing complexity and luxury of huge built environments is a sign that this trend will continue also in the future in every aspect of our life. And this brings itself the problem of finding the wanted spot easily. Positioning in places with a high potential to host communities, such as hospitals, shopping malls, and construction sites, provides advantages to people in many respects. For example, we do not want to spend our hours going around the same places while trying to reach the store where the only piece of product we want to buy is in a huge shopping mall. Additionally, it's an obvious fact that we all face difficulties while going to our appointments in hospitals or looking for the right gates for flights at airports, etc. Rather than such additional comforts, the issue of positioning in a closed area in more critical situations is a problem that we need to deal with. An example of this scenario is the disappearance of children in a large shopping mall. This is a traumatic experience for both the child and the parents. As a result of these and many other situations, positioning in closed spaces has become a need today. Therefore, this problem is one of the issues that is actively being worked on today. Unfortunately, the GNSS (Global Navigation Satellite System) signals still cannot provide an efficient solution for it. Because while using GPS technology signals indoors, it becomes very weak when passing through walls or cannot pass at all. So, what are the solutions for navigating, and managing structures with complex interiors more efficiently? All of us have noticed the list of places at the entrance of each giant building for guiding people in shopping malls, universities, hospitals, etc. However, most of the time this way of guidance causes some complexities. Therefore, we need a new approach to finding the wanted area in closed locations. And for solving this problem the Indoor Navigation technologies are coming on stage. This is a new term that appeared in recent years and a unique way of approaching navigation problems in closed areas. With the use of this technology, we could save our time in finding a wanted spot in various huge buildings in an optimal way. Also, using this technology can help people in some emergency cases such as finding an emergency exit in massive buildings or finding the necessary department in a hospital. Additionally, the way of using indoor navigation technologies not only solves the problem for people who are looking for a special spot, it also has commercial benefits for tracking and analyzing human behavior. In this way, we can improve the efficiency in a different way for different locations. For example, we could enhance the sales in the shopping mall by analyzing which location the people are interested in more. Either we could apply this technology in any museum for making improvements in design and arrangement or we can also use it in clinics for detecting which department of the hospital the patients need. There are plenty of indoor navigation technologies and in the next sections we will analyze them.

1.2. Literature Review

1.2.1. What is Indoor Positioning

Indoor positioning is when the positioning process is carried out indoors. Indoor positioning, like other technologies, has its own problems. It is necessary to choose one of the many indoor positioning techniques respective to an indoor positioning problem. In addition to the suitability of the techniques for the solution, there are also different success criteria. These criteria are considered when deciding which technology to use. These criteria are generally as follows: consistency, scalability, upgradeability, compatibility, and cost. There are cases where each criterion has advantages and disadvantages over different indoor positioning technologies. The types that stand out in indoor positioning are WIFI, RFID (Radio Frequency Identification), Bluetooth Low Energy Ultra-Wide Band.

1.2.2. Indoor positioning technologies

IoT systems are systems consisting of devices that communicate with each other in real-time. The event, which is considered the first example of IoT, was the communication of a camera system and a computer. In 1991, when a group of academics at the University of Cambridge wanted to view the coffee machine in the environment, they placed a camera in such a way that they could see the coffee machine in the environment and they sent the images they obtained of the coffee machine from this camera to their own computers, three per minute. Thus, for the first time, two devices communicate with each other in real-time (WANG, 2013)

1.2.2.1. WIFI Based Indoor Positioning

Nowadays it is now possible to see WIFI access points almost everywhere. For this reason, the number of signals received in WIFI-based indoor positioning technology is much higher than in other technologies. Maps can be obtained using these signals. When you think about it, there is now WIFI infrastructure in every building. Therefore, when it is desired to create a WIFI-based indoor positioning system, it is not necessary to create any hardware infrastructure. For this reason, there is a positive situation in terms of cost. Unlike satellite signals (GPS), WIFI signals can also pass-through walls.

In WIFI-based indoor positioning systems, what is important is the strength of the signal. The working principle of such positioning systems is generally based on RSSI (Received Signal Strength Indicator). When we consider the fact that there are access points at many points in WIFI-based systems, we get a very high number of samples in the power measurements of the signals sent to these access points. Here, the loss of power in the signals sent to the access points is inversely proportional to the distance between the access points contextually. As the distance between the access point and the signal source device decreases, RSSI (signal strength indicator) increases, and as the distance increases, RSSI decreases. In WIFI-based systems, algorithms are created based on this logic based on power loss in general. For example, Google has developed the Google maps application by adding IP address positioning as well as RSSI logic.

Apart from a large number of access points in WIFI-based systems, another important advantage is that WIFI signals can pass through the walls. However, since the walls affect the signals, it is necessary to create a unique map of each indoor space. "WIFI fingerprint" is used when creating these indoor maps. By creating a system that extracts from the structures of these maps, good work can be done on finding locations (Tunca, 2014). Machine learning technologies can also be used to make this inference.

The "fingerprinting" method, which is the most used approach in RSSI-based solutions, consists of two stages. The first stage is the training phase, and the other stage is the monitoring phase (Lee Pieh Wen). The signal data obtained during the training phase are interpolated and stored in the database. Interpolation enables the production of new data with the available data. Interpolation enables the production of new data with the available data. By interpolation, a new and unknown value is created as an estimate in the range of the available data. The margin of error is also an important factor. With this method, new signal data is obtained from the signal strength information collected in the first stage of the RSSI-based fingerprint solution in WIFI-based systems. During the monitoring phase, the data stored in the location database is agreed with this sample data. The accuracy rate of the system also depends on the number of sampling data (Lee Pieh Wen). The fingerprinting approach can also appear as "track matching".

In the studies carried out, a regression can be used as well as an interpolation method in the fingerprint and track matching approach. Regression is a method used to measure the relationship of multiple variables to each other (Tunca, 2014). Radio maps are created by using RSS (Receiver Signal Strength) obtained from WIFI signals, in other words, the received signal strength values (Zegeye, 2016). The offline signal strength fingerprint of the environment can be compared with the online signal strength fingerprint for location estimation.

In WIFI-based positioning systems, positioning accuracy can increase from centimeters to tens of meters. The accuracy rate is higher than 5 – 15 meters. The coverage area can be up to 150 meters.

1.2.2.2. Bluetooth Based Indoor Positioning

Another type of technology used in indoor positioning is Bluetooth technology. However, Bluetooth technology consumes a lot of energy while in use. Therefore, "Bluetooth Low Energy" technology is used in indoor positioning systems. The battery consumption level of Bluetooth Low Energy technology is very low. Bluetooth Low Energy offers slower connection speeds than standard Bluetooth technology, but it's faster at pairing devices. The accuracy rate of the systems is created with the architecture of the system (Tsanga, 2015). One of the most important reasons for choosing BLE (Bluetooth Low Energy) technology for indoor positioning is the low cost and low signal distortion (Tsanga, 2015). When creating an indoor positioning system with BLE technology, the first stage is to decide the reference points. This stage is crucial to finding the most accurate RSSI propagation model (Jianyong, 2014). Another stage is RSSI (Received signal strength indicator) in real-time. After a model is created with BLE, this model must be associated with the real space. Indoors, after the model is realized, the active learning process for the BLE reference points begins. In the active learning process, adjustments are made to the model. While WIFI signals are used in this WIFI-based technology, beacons are used in BLE technology. Bluetooth beacons are signal that device with BLE technology send to other devices, depending on the distance. With these signals, certain information is transmitted to other devices. When devices receive any signal, they measure the strength of that signal (Jianyong, 2014). While WIFI signals are affected by walls and objects, BLE beacons, in other words, signals are also affected by these factors. In fact, Bluetooth Low Energy technology is used for proximity detection rather than location detection. Figure 1 shows a heat map created with information obtained from BLE beacons and information about where the device emitting these beacons is potentially located (Floreani, 2015).

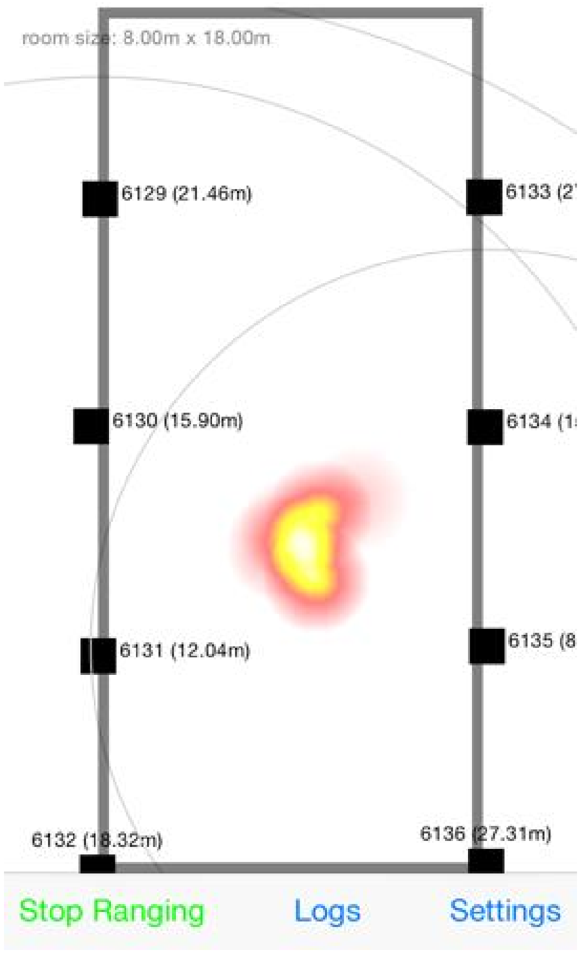


Figure 1: Bluetooth based beacons positioning (Floreani, 2015)

The most obvious example of BLE technology is Apple's IBeacon. IBeacon is a location-based device. By emitting a Bluetooth signal at regular intervals, the distance between the IBeacon and the receiving device can be measured through this signal. WIFI-based indoor positioning systems have an accuracy rate of from centimeters to tens of meters, while in BLE technology this is between 1 and 3 meters. Its capacity is less than 30 meters.

1.2.2.3. RFID Based Indoor

RFID technology is used to transport data via radio signals. RFID systems usually consist of 2 parts. These are the RFID tag and the reader part that allows the information contained in this tag to be retrieved. It is a more costly technology compared to BLE technology. RFIDs contain two types in themselves. One of them is active and the other is passive RFID. Passive RFIDs use radio waves received from the reader as energy. On the other hand, active RFIDs broadcast signals and have their own power supply. In other words, passive RFIDs become active when the RFID tag enters the reader's coverage area and transmits the read information to the transmitter of the signal. In active RFIDs, there is a separate energy source for transactions and communication. Passive RFIDs are generally used in studies with small coverage, while active RFIDs are used in studies with large coverage areas. In passive RFIDs, the coverage area is up to 4 meters, while in active RFIDs this is up to 100 meters.

RFID technology in indoor positioning is as follows: It is a system that is integrated into different zones and consists of readers that read the signals from the RFID tag and tags that will send signals indicating the location. The read signals are sent by the reader to a trader for meaningful inferences. At the same time, this information is recorded in a database. Passive RFIDs cost and weigh less than active tags. Indoor positioning algorithms are generally divided into two approaches. One of them is the model-based approach and the other is the fingerprint-based approach. In the model-based approach, a location finding process is carried out based on geometric models. In the fingerprint approach, the main issue is fingerprint matching. In the first stage, a database is created from the data obtained from the signal strength received. When a signal is received from the user, matching is made to the traces in this database, and matching location information is returned (Zou, 2014).

Different algorithms are used when creating a closed positioning system with RFID technology. These are in the form of triangulation and reference tags (Xu, 2017.). However, in indoor positioning systems, there are different algorithms besides these algorithms. The advantages of RFID Technology over other technologies are that it has an anti-interference status, its size is small and lightweight, and its integration is easy. When creating the system with RFID technology, we mentioned that it has two parts, the RFID tag, and the reader. There is also a communication network that allows these parts to communicate. This communication is carried out with a specific RF protocol. The accuracy of systems created using RFID technology varies between 1 and 2 meters. The coverage area is up to 30 meters (Zou, 2014).

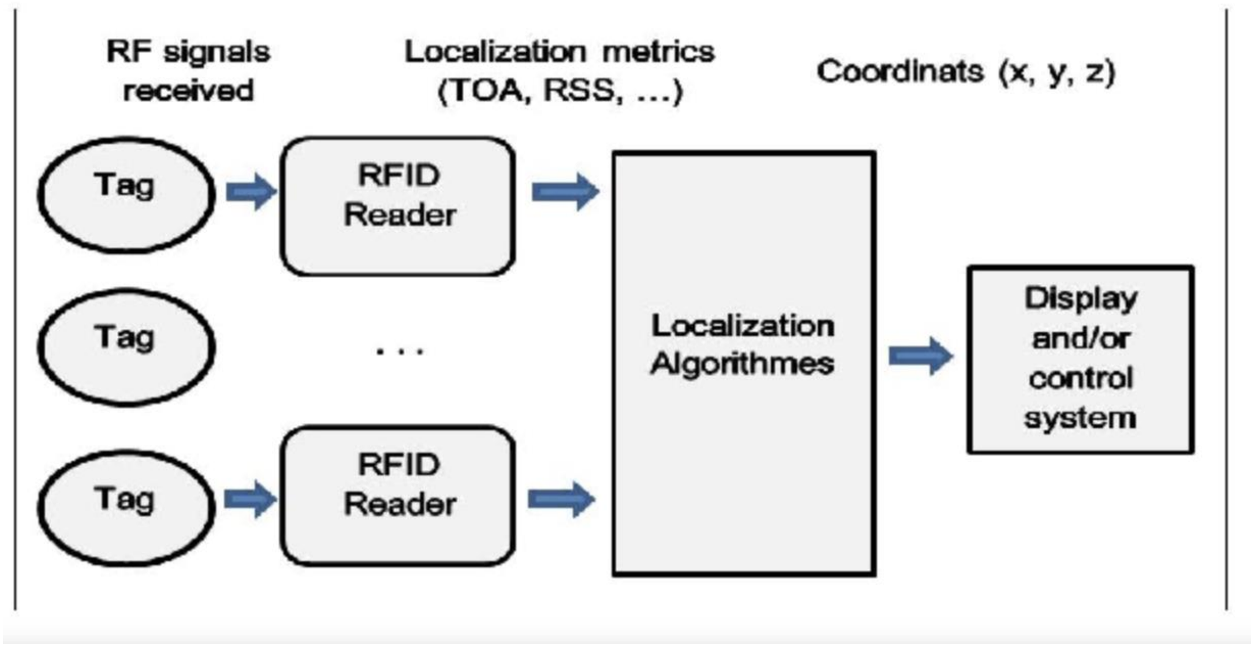


Figure 2 (Belhadi, 2014)

Figure 2 shows a block diagram example of an indoor positioning system built with RFID components. The data read from the RFID tag is read by the RFID reader. It is then sent to the localization algorithm through the reader. The location information obtained from this algorithm is given to the main system to be shown or checked (Belhadi, 2014). In RFID technology, receivers and transmitters do not need to see each other directly. ZigBee is simply a wireless network standard for data transfer over short distances. ZigBee is aimed to avoid unnecessary costs and power consumption in the transfer of small-size data. It also frees you from the complex issue of network design. RFID technology has advantages as well as disadvantages compared to other technologies. For example, the lack of RFID sensors in today's devices is one of these disadvantages. Again, one-way communication channels and unstable signal strength are also disadvantages of RFID technology (Morais, 2017). RFID systems can operate at low and high frequencies. To create a certain system, the most appropriate frequency range is used for that system. For example, the microwave frequency range is used for instant positioning systems.

1.2.2.4. UWB Based Indoor Navigation

UWB, in other words, ultra-wideband technology, uses high-bandwidth radio frequency in data transmission. With UWB technology, a small amount of energy consumption is experienced while a high amount of energy is achieved. When you want to create an indoor positioning system with this technology, go over the arrival time difference (TDOA) of radio frequency signals to find the distance between the reference and the target (Alarifi, 2016). The arrival time difference is a form of calculation algorithm used in indoor positioning. The accuracy value in UWB systems is very high compared to other technologies. The most important reason why the margin of error in UWB technology is so low is that it has a mechanism that prevents parasites. Interference is a condition that occurs when the same frequency range is shared by different applications. In order to prevent this situation in UWB systems, data transmission is carried out at different power levels even if the frequencies are the same (Kucuk, 2017). Energy consumption is low, and coverage is wide. All these advantages make this technology much more costly than other technologies. In UWB-based indoor positioning systems, the coverage area reaches up to 150 meters. The error rate varies between 0.1 – 0.3 meters. UWB systems have tags just like RFID systems. These labels act as transmitters and emit UWB signals. There are also receivers that receive these emitted signals (Kucuk, 2017). UWB signals are able to break through walls. For this reason, it can be preferred in indoor positioning systems.

1.2.2.5. Ultrason Based Indoor Positioning

Sound waves are used to perform position analysis in ultrasound-based indoor positioning systems. Ultrasound is a branch of sound that propagates at frequencies too high for the human ear to hear. Ultrasound-based systems provide much more efficient information than BLE or WIFI technology in buildings with many rooms, such as hotels and hospitals, and require less infrastructure (Gifford, 2018). In ultrasound systems, the "Arrival Time" algorithm is used as the calculation algorithm. However, since the power of ultrasonic sound waves is not enough to pass through the walls, each room in the closed space must have its own system (Ergen, 2018).

1.2.2.6. Success Criteria of Indoor Positioning Systems

The success criteria of indoor positioning systems are categorized as follows: Sensitivity: Sensitivity refers to the speed at which the instantaneous position is updated for the individual in motion being monitored. If the location of the monitored individual can be updated simultaneously, the system has been successful.

Speed: Not only for indoor positioning systems but also for all information systems, the rapid achievement of the result is a great factor for visible success. The faster a system is, the more successful it is. There is a problem with indoor positioning systems that need to be balanced in terms of speed. For a successful positioning system, the system speed must be greater than the speed at which the monitored individual moves out of the coverage range. Otherwise, the system will not work again for the individual who goes out of the coverage range. At the same time, the more often the duration of the detection pulse in the coverage range, the more likely it is that the individual will not be detected. However, if the detection process is carried out frequently, over short intervals, it means more frequency of operation, which means more power consumption. For a successful system, the size, speed, and power consumption of the region included in the coverage range must be configured in the most optimal and balanced way.

Integrability: In indoor positioning systems, various integrations can be performed only to add additional functions to the system or to strengthen existing functions. An efficient system must be compatible with integration.

Accuracy: As the margin of error between the values obtained from the system and the actual position values decreases, the accuracy of the system increases. The "Euclidean distance" of the distance between the actual location and the estimated position can be used to express the accuracy of indoor positioning systems (Alarifi, 2016). The Euclidean distance refers to the linear distance between two points.

Cost: When it comes to the concept of cost, financial situations are directly conjured up in the human mind. However, although money is an important criterion in indoor positioning systems, situations such as time, energy, and space covered are also within the cost (Alarifi, 2016). Successful systems are those that achieve maximum efficiency at minimum costs.

Coverage Area: In indoor positioning systems, the larger the area that the system can reach, the more efficiently the system works.

Data Size: When the data reaches the end-user, it should not be too large and should be transferred to the user at a minimum scale as filtered, meaningful data as much as possible.

Privacy: There is strong access control when collecting, processing, and storing users' data.

Scalability: There are two main factors that affect the size of the system: geography and the number of users. As the size of these two elements increases, the need for intervention in the system does not increase at the same rate, which is a feature that should be in a successful system (Alarifi, 2016).

1.2.2.7. Result

The most common technologies in indoor positioning systems are as follows: WIFI based, Bluetooth-based, RFID-based, and UWB-based positioning. Each of these technologies has its own advantages and disadvantages. Depending on the coverage area, the choices may differ. In addition, need-based selections can be made for error distances in positioning. Considering the cost depending on the size of the area to be studied, it is seen that BLE technology is more scalable.

In many studies on indoor positioning, it has been observed that the most common technologies are WIFI and BLE technologies. With these technologies, RSSI or fingerprint approaches are often combined. The reason for this is that the devices that require high precision in terms of hardware for approaches such as arrival angle and arrival time have high costs. Even if the technology and approaches used together are carefully chosen, the manipulative effects caused by the interaction of various causes on the signals cannot be completely avoided. In order to get rid of these effects, improvement can be realized with supporting elements such as various filtering methods and artificial intelligence.

2. Theoretic explanation of our methodology.

2.1. Theoretical explanation

The main inspiring reason for making this idea real is to help people to find the desired location in unfamiliar indoor infrastructure. Therefore, we begin to analyze different techniques which we have mentioned above and detect their pros and cons and based on these results create our own concept. Our concept will help users to detect their next destination just by scanning QR codes on the surface in different infrastructures. We could achieve these results by storing the building plan information and attaching QR codes to the necessary points. We already have mentioned different techniques such as WIFI-based, Bluetooth-assisted, and RFID tags for solving the current problem. However, with a deep comparison of these results, we can easily observe that our techniques are cost-effective and quite easy to implement in any infrastructure and use. Let's see the drawbacks of different approaches in Table 1 compared to our methodology and explain the logic of our methodology theoretically.

Table 1

Methods/ Classifications	WIFI	Bluetooth	RFID	UWB	QR code
Accuracy	Low	High	Low	High	High
Cost	Low	High	High	High	Low
Infrastructure	Low	High	High	High	Low
Algorithm	RSSI, Landmarc Approach	RSSI	RSSI	Multilateration	Lambda Twist, Dijkstra Algorithm
Coverage Area	150m	100m	30m	150m	Entire Building

In the above figure, we have compared several factors like accuracy, cost, infrastructure, algorithm, and coverage area in different techniques. After our deep analysis of all these methods, we understand that our methodology will provide us an advantage mainly in cost, infrastructure, and accuracy over the other techniques. First of all, for making an infrastructure we need to create the necessary QR code markers with the given id and URL data and attach them to the given points. Another important point is attaching all QR code markers with the same cardinal directions (North, South, East, West). In our methodology, we use a non-square border of QR code look in the southeast direction. After all these configurations, the user just needs to use a mobile phone to enter our website for choosing a destination and scan the QR code. There are several obstacles we have faced at the beginning of the project. The first problem was related to detecting the position and properly defining the next destination of the user. In the next steps, we will deeply analyze how we have solved these problems. Foremost, let's get familiar with QR code and its working principle which helped us a lot in our methodology.

2.2. QR code and how we use it in our methodology.

Our methodology is a visual-based navigation system. And we have used QR codes as a visual marker in our methodology because of their plenty of advantages. Today QR codes can be used in almost all parts of our life for different purposes. First of all, let's understand what QR code is and how we can use it. The most basic feature of the QR code, which is included in our lives at once, is that it accelerates the flow of information. Instead of remembering or taking note of the information we have seen in the newspaper, on tv, or laptop, we just can use an application that scans the image and records all necessary data we need. And this application - QR code scanner can be used by almost all kinds of android devices. From this point of view, the QR code can be considered one of the most effective visual elements of accelerating information sharing in the digital world.

QR Code is a 2D barcode system developed by Denso Wave (DENSO WAVE, n.d.) a subsidiary of Toyota, which operates in Japan. It takes its name from the initials of the words Quick Response. Its content can be any data, including a text, website address, or video link. Through QR Code reader software, a mobile phone can conveniently read the QR Code and open the relevant product or service page. It is quite simple to use. You can read all QR codes with a smart mobile phone with a camera and an app that helps to read barcodes.

2.2.1. How do QR codes work?

QR codes are two-dimensional barcodes that transmit information with the graphic code of black and white pixels. In other words, the information is transmitted by reading both the horizontal and vertical positions of the pixels rather than just the horizontal position. QR codes can be decoded with smartphones and QR scanner devices. This allows web addresses to be encoded in QR codes and automatically send the scanning device to the URL specified in the default browser.

In a visual sense, the most obvious element of the QR code is the square blocks used to route the code when scanning, ensuring that the code is scanned in any direction, including upside down, and that it sends the correct message. This functionality of the QR codes helps us in our methodology to define the direction in an optimal way. Around a total of four blocks, three of which are embedded in the corners and one in the lower right, the version format, the error checking the version, and the decoding mask are encoded. The white space that helps the reader find the edges of the code and surrounds the QR code is called the "quiet zone". QR codes can be of various sizes, and smartphones can work with QR codes of all sizes. The QR code is read from the lower right corner. Reading pixels is performed in groups of 8 parts containing one byte per 8 pixels. Byte patterns are linked to different characters depending on how the QR code is encoded. QR codes have a maximum capacity of 7089 characters numerically and a maximum of 4296 characters alphanumerically. As a binary system, its capacity is a maximum of 2953 bytes. Due to all these capacity and efficiency, the QR codes are going to use in a variety of fields.

2.2.2. What are the advantages of the QR code system?

Although QR codes are practically the same as barcode codes, the reason QR codes are so popular is that they can contain much more information. The QR code system has many advantages over the traditional barcode system. At the top of these is the ability to accommodate more characters in any language. It can be created in sizes that are 1/10 smaller than traditional barcodes. Read faster with the scanning feature from all directions. It has an error correction feature. It can be read even if there is any contamination or damage of up to 30% on it.

QR codes, which are increasingly widely used, are the most frequently used areas of business, banking sector, and business cards. Thanks to QR codes, effective solutions can be offered to many different points today. Since QR codes can be accessed using personal phones, it is a hygienic tool that allows contactless processing. Especially during the pandemic period, which causes hygiene to come to a much more important point in our lives, QR codes used in loyalty applications in different businesses such as cafes and restaurants

QR codes are now one of the indispensable elements of advertisements and promotions. In the last few years, small black and white squares have appeared in newspapers, magazine advertisements, and billboards with increasing frequency. These are QR codes that allow us to have more detailed information about the ads and promotions we are seeing. QR code creation can be done by anyone and after the QR code is created, it can be added to the desired place. This ensures that everyone who scans the QR code has access to the information added to the code. Although QR code creation is considered a complicated process, it is a process that is easily performed thanks to websites that help create QR codes.

2.2.3. How QR codes help us to detect the position and direction of the user.

In the first phase, we just use the camera of our device to take the photo with HTML canvas and save this image in base64 format. Base64 Encoding is the most widely used technique for storing or transmitting binary data by converting it into text. Afterward, we transmit this data to the rust library for detecting the QR code. For this purpose, we have used rqrr library in rust which detects and decodes QR code border coordinates and content with best performance. We will analyze the usage of this library in more detail in the following units.

After getting the border coordinates of QR code we used these coordinates to detect the position and direction of the user. Compared to the previous issue (detecting the coordinates of the QR code border) this task was fairly easy. The point is, we could generate each QR code with a different ID and content manually. Therefore, we could easily differentiate the QR codes and detect the user position based on the id on the QR code content. We keep all these QR code ids in our database with their position on the surface also with additional data which we will explain in more detail in the further steps. Thus, while the user scans the QR code we could take the id of the QR code and send it to our server for determining the position of the user, based on the QR code coordinates on the database.

2.3. Camera position estimation problem

After solving the issue related to the coordinates of a given QR code, we face with camera pose estimation problem. The problem occurs when we need to calculate the position and orientation of an object relative to the camera. Position estimation is one of the important problems in the fields of robotics, and photogrammetry. Applications such as object tracking, object recognition, and robots finding their own positions can be given as examples of the use cases of the position prediction. The first process to be done for position extraction is to choose the method to be used. These are the Single View Geometry and Multiple View Geometry methods. In both methods, camera calibration must be done first. As a result of calibration, two kinds of parameters are obtained: internal and external.

According to the selected method, a single or two-camera system is created and calibrated. Cameras are calibrated using images of a calibration object whose dimensions are known from different angles and distances. After calibration, two kinds of parameters are obtained: intrinsic and extrinsic. Intrinsic is used in the correction of lens disorders. Extrinsic parameters, on the other hand, show the relative geometric relationship between the cameras. And position estimations are made by using these parameters. In Figure 3 we can see the general diagram for position estimation.

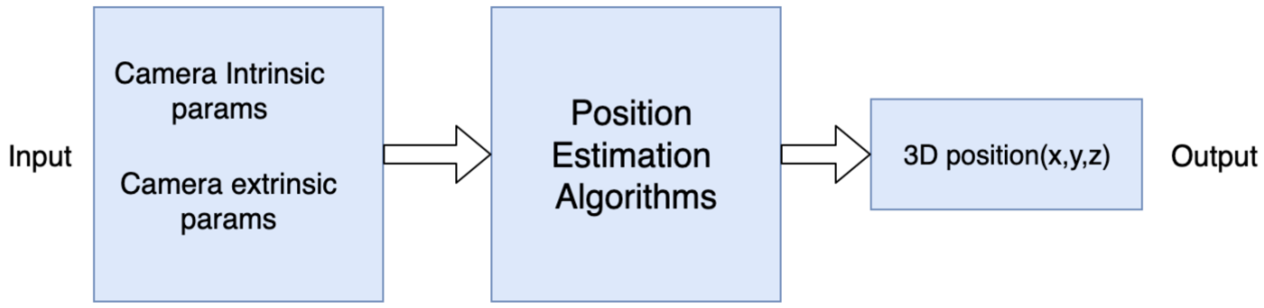


Figure 3

2.3.1. Position estimation algorithms

In this section, we will analyze different algorithms in more detail for solving the position estimation problem. The main algorithms that are used to solve this problem are Direct Linear Conversion (DLT) which makes point-based position estimation and the Perspective n-point algorithm.

Matching 3D-2D points in position estimation is a widely used method in the literature. In this method, the goal is to estimate the six degrees of camera position and camera calibration parameters: focal length, principal point, aspect ratio, and skew. This is done using the DLT (direct linear transform) algorithm. A variant of DLT is the perspective-n-point problem, which assumes that camera internal parameters are known. When multiple points are introduced to solve the system, the set of n mapping between the 3D points and their 2D projections arises from the Perspective-n-point problem that determines the given position and direction. For this reason, information about position estimation algorithms such as DLT, and PnP used in this section is given.

2.3.2. Direct linear conversion (DLT)

Direct Linear Transformation is the starting point in position recovery. First used by photogrammetry and introduced in the computer vision community, this algorithm solves a linear system of equations with unknown camera parameters, making the projection matrix predictable.

2.3.3. PnP problem

While in the real-world objects are perceived in three dimensions, they are perceived as two-dimensional on the computer screen. It is important to have the third dimension of the image on the two-dimensional screen. For example, in order for the distant appearance of the object to be realistic, it must be smaller, and the object close to the screen must be larger. This process is explained by perspective projection. In the perspective projection model, the lines are not parallel to each other. The lines exit from the projection center point. When multiple points are introduced to solve the system, the problem is called the Perspective N Point Problem (PnP), the set of n matching between the 3D points and their 2D projections becomes the problem that determines the given position and direction. The P3P issue is known as the minimal PnP situation with a finite number of solutions. It requires three (n = 3) observations in a nondegenerate configuration. The function P3P is represented in Equation 1.

Equation 1

$$[R,t] = P3P(P_{1:3}, y_{1:3})$$

Since it is a calibrated camera, the internal parameters of the camera are specific. In Figure 4, the projections of the world point P1, P2, and P3 in 3D in the plane of view are shown by y1, y2, and y3, respectively. The distance between points P1 and P2 is d_{12}, the distance between points P3 and P1 is d_{31}, the distance between points P2 and P3 is the distance between points d_{23}, P2 and P3 to the optical center O are expressed in d1, d2 and d3, respectively, the distances d1, d2, and d3 are calculated as in the Equation 2. In these equations, the angle between the lines d1 and d3 is represented as β, the angle between the lines d2 and d3 is represented as γ, and the angle between the lines d1 and d2 is represented as α.

Equation 2

$$d_{12}^2 = d_1^2 + d_2^2 + 2d_1 d_2 \cos\alpha$$

$$d_{31}^2 = d_1^2 + d_3^2 + 2d_1 d_3 \cos\beta$$

$$d_{23}^2 = d_2^2 + d_3^2 + 2d_2 d_3 \cos\gamma$$

The Equation 3 is used to find the rotation (R) and translation (t) of the camera, with the 3D points being i = 1:3, the 3D points Pi = (xi, yi, zi), and the homogeneous 2D image points corresponding to these points being yi = (ui, vi, 1).

Equation 3

$$\lambda i y i = RPi + t$$

In Equation 3, λ_i is expressing the same distance with d1, d2, and d3, which are the distances from the camera center O to each 3D point. Therefore, when the found the distances d1, d2, and d3, the scale factors λ1, λ2, and λ3 are also found automatically. The expressions that remain unknown in equation (3) are reduced from λ_i, R, t to only R and t.

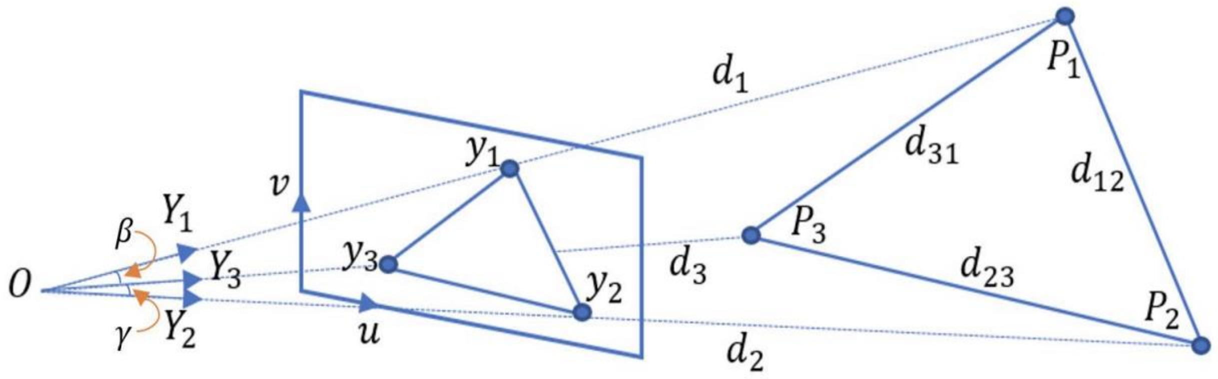


Figure 4 (OpenCV , n.d.)

2.3.4. RANSAC

RANSAC (Random Sample Consensus) is a method that predicts the parameters of a mathematical model from observed data that produces an approximate result as the number of occurrences increases. 2D image points that (OpenCV , n.d.)have a projection of the same 3D point across different image frames are called image correspondences. If there are outliers in these image counterparts, the P3P algorithm becomes prone to errors. The Ransac (Bolles., 1981) algorithm is used together with the p3p algorithm to eliminate the incorrect outliers. In the RANSAC algorithm, samples are taken from image point counterparts and the points in this sample are assumed to be inliers. Since the P3P algorithm deals with 3-point counterparts, 3-point counterparts are selected from the points in the set that are compatible points.

In the context of camera position estimation, it is very easy to implement since there is no need to predict the beginning of the parameters. The algorithm randomly extracts subsets of small dots to form what is called a hypothesis. For each hypothesis, a PnP approach is used to recover a camera position, which is then used to calculate the reprojection error. The points of reprojection that are close enough to the 2D points are called inliers. RANSAC depends on certain parameters, such as tolerance error, which decides whether to consider an inlier based on the reprojection error of a point. In Equation 4, the formula is proposed to calculate the number of iterations that the algorithm should make based on a probability p that at least one of the hypotheses wants to succeed as a coherent solution.

Equation 4

$$k = \frac{\log(1-p)}{\log(1-w^n)}$$

In this equation, w is the ratio between the inliers and the number of points. The value of k tends to increase as the size of the subsets increases.

2.3.5. Lambda Twist

In this section we will analyze the Lambda Twist: which is one of the fastest and most accurate among all other PnP solvers. In Lambda Twist rather than computing a quartic, we take advantage of the actual elliptic equations, which may be solved by accurate diagonalization quickly and efficiently. In our methodology, we applied lambda twist functionality in rust. In Equation 5 we see the general formula for calculation lambda twist which has taken from Rust official documentation

Equation 5 (Rust, n.d.)

$$\lambda_i y_i = R x_i + t, i \in 1,2,3$$

- X_i is 3D world point coordinates
- Y_i is image coordinates also called bearing vectors $Y_i \sim (u_i, v_i, 1)$
- Lambda_i is the given distance from the camera to object.

The advantage of our solution is to pretend to create a geometrically invalid and non-robust solution. The testing and analysis of this solution have proven that it has more accurate and faster results compared to other P3P solvers. We will see the several test results which have been done among the most famous P3P algorithms such as Lambda Twist, Kneip, and Ke for measuring the accuracy and robustness of data. In table 7. we can see the mean, success, failure, and uniqueness statistics of the 10^6 samples for different algorithms. As a result of this test, we could say Lambda Twist algorithms overcame other methods with less failure, more success rates, and more unique solutions (Mikael Persson, 2018).

Table 2 (Mikael Persson, 2018)

Methods	Mean	Success	Failure	Unique
Lambda Twist	278	9999968	32	16934510
Ke (2 iterations)	342	9995790	4210	16924141
Ke (10 iterations)	435	9996105	3895	16925025

Kneip	1042	9994973	5027	16909306

We can figure out the advantage of Lambda twits with 2 different test results. In Figure 5 (Mikael Persson, 2018). We observe that we can get fewer numerical errors compared to the Kneip algorithm and almost adjacent results with the Ke algorithm (Mikael Persson, 2018).

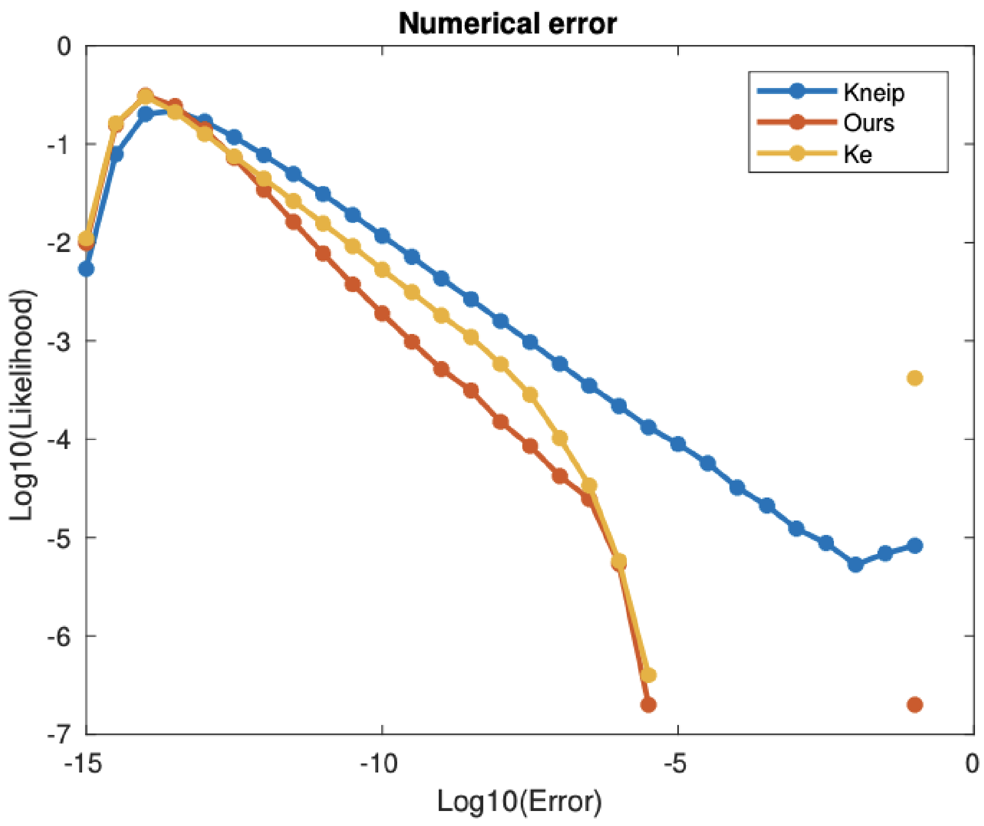


Figure 5 (Mikael Persson, 2018)

In Figure 6 (Mikael Persson, 2018) we can see the time comparison result over the 10^7 samples. In the given test Lambda Twist algorithm provides more frequency with the estimation of given data rather than the Kneip and Ke algorithms. (Mikael Persson, 2018)

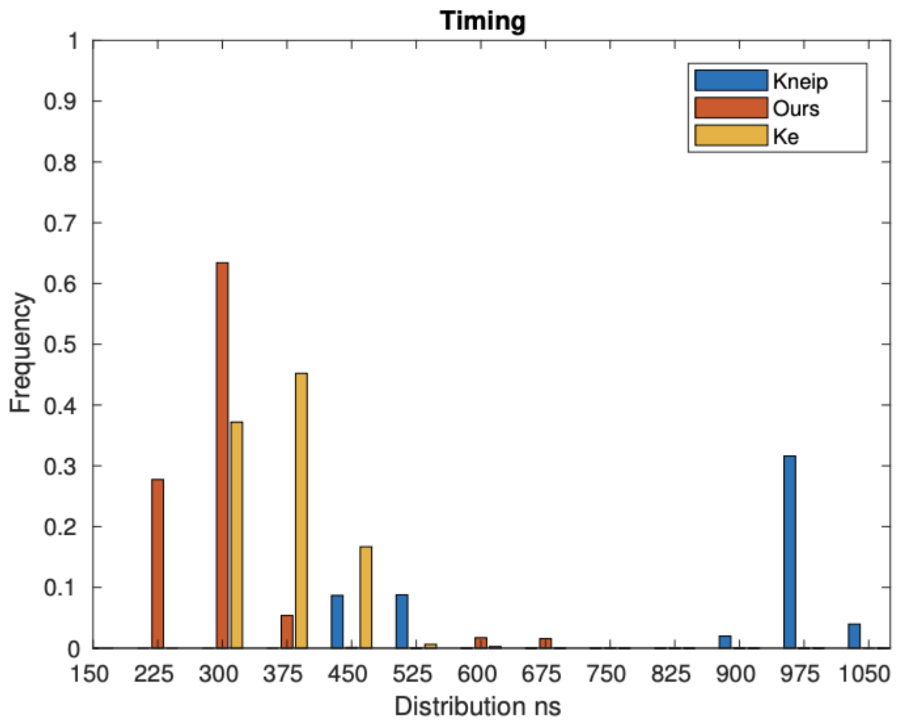


Figure 6 (Mikael Persson, 2018)

2.4. Shortest path problem and Graph theory

Finding the shortest path is the problem of determining the existence of a path that can be traveled between two nodes with the least cost. The shortest path problem can be calculated from one node to another, from each node to all nodes, or for all nodes. The shortest path algorithms are based on graph theory. Therefore, designing the logical structure with graphs and reconciling this design with the interface is one of the preliminary steps. A graph is an association-based network structure consisting of nodes and edges that connect those nodes. To put it more simply, a graph is made up of a finite set of points called vertices and line segments or curves called edges connecting points. Because we are surrounded by networks such as social networks, transport networks, or the internet, graph theory plays an important role in modern mathematics. Graphics can be used to model a wide variety of situations. To study the cities on a map, the border relationships of states or countries, friendships between people, or family relationships can be shown with graphs. For example, let's look at the floor plan below.

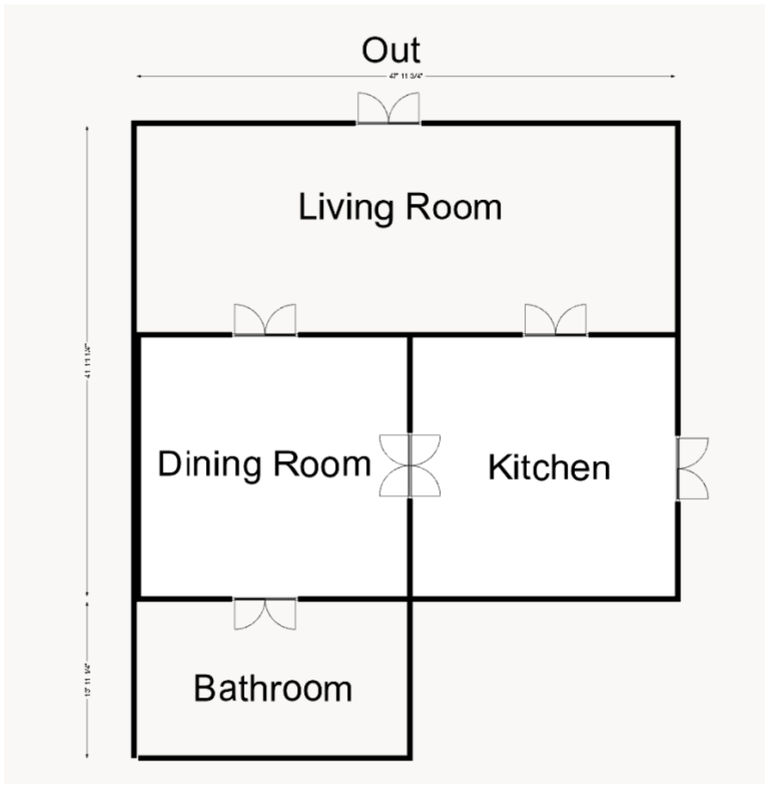


Figure 7

In this figure, we need to think about what connects objects in the floor plan. These will be the doors that connect the rooms as we observe. Therefore, we must accept the rooms as nodes and the doors as edges that connect the nodes. The living room, kitchen, and dining room all have three doors. The outside is only accessible from the living room and kitchen, and the bathroom is only connected to the dining room. In this case, our chart will be as follows as in Figure 8.

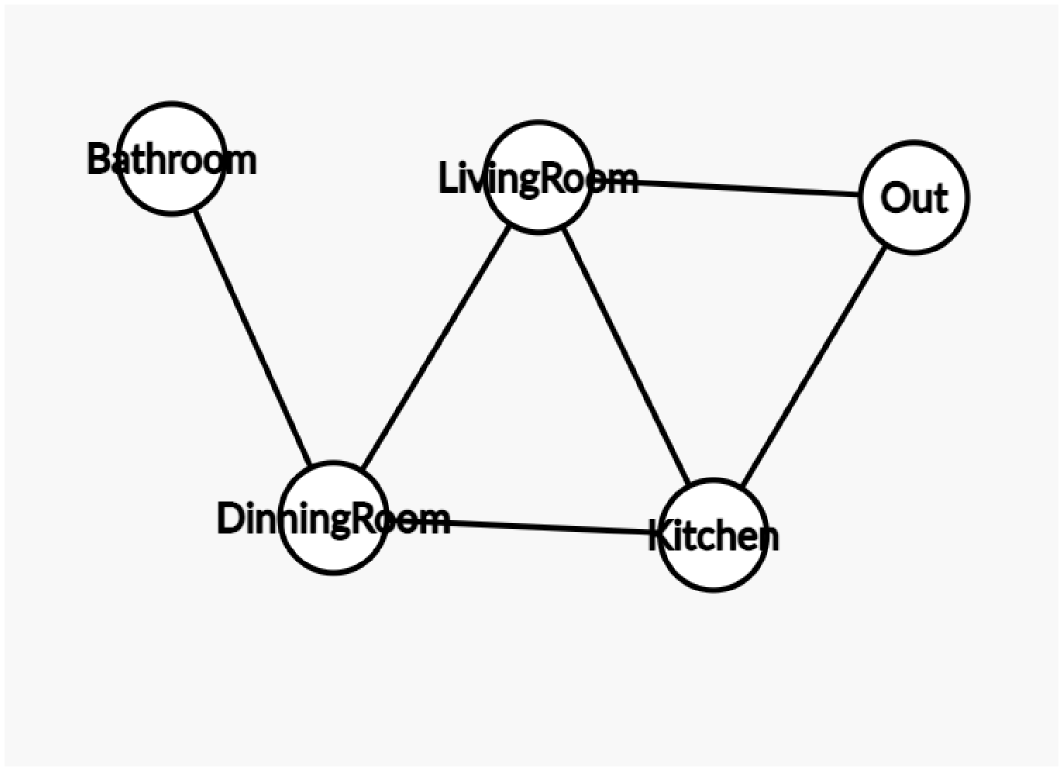


Figure 8

In this figure, we can observe the simplest graph type which consists of only nodes and edges. The general Graph theory equation is described in Equation 6.

Equation 6

$$G = (V, E)$$

This definition above means that any graph is a set of vertices and edges. Graphs are qualified based on different criteria such as directed/undirected or weighted/unweighted.

Today most of the shortest path algorithms refer to graph theory and we could apply these algorithms in different fields. For example, it helps a transportation company to build the least costly transport network. Another most famous usage area of shortest path algorithms is navigation. When using Google Maps or any navigation application, the application can take you to where you want to go in the shortest and most profitable way with the usage of shortest path algorithms. The following list contains some of the basic algorithms that solve the shortest path problem that forms the basis of graph theory.

- Dijkstra Algorithm
- Bellman-Ford Algorithm
- Floyd Algorithm

In our methodology we also have to face the shortest path problem for finding the best path from the detected source to chosen destination. For solving this problem, we have applied the Dijkstra algorithm due to the several advantages over the other algorithms. First of all, let's better understand the specification of the Dijkstra algorithm with the comparison to other algorithms and the way we applied it in our methodology.

2.4.1. Dijkstra Algorithm

The Dijkstra algorithm calculates the shortest paths from one node to all other nodes. In other words, it is the algorithm that determines the shortest path according to a certain starting point. It is developed for weighted and directional graphs and the weight value of the edges must be zero or greater than zero. The time complexity of the Dijkstra algorithm is generally calculated as O(NlogN).

The Dijkstra algorithm uses the Greedy approach to determine the shortest path. The Greedy approach is a decision-making method used to determine the next node when moving around the graph to find the best result on a particular topic. This means that when moving from one node to another, it considers the best possible local solution. Each time the progression to the next node is made according to the Greedy approach. Figure 9 (COMERT, 2015) shows the weighted graph and Figure 10 (COMERT, 2015) describes the adjacency matrix for the weighted graph (COMERT, 2015). This is done by solving the shortest path problem with the Dijkstra algorithm

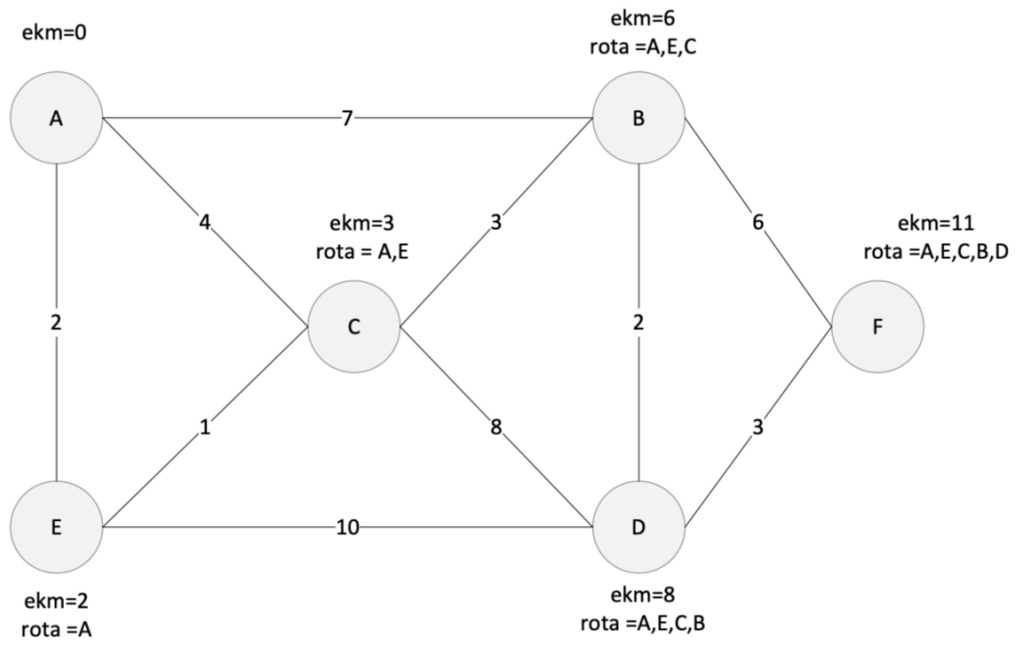


Figure 9 (COMERT, 2015)

A	A	B	C	D	E	F
A	-1	7	4	-1	2	-1
B	7	-1	3	2	-1	6
C	4	3	-1	8	1	-1
D	-1	2	8	-1	10	3
E	2	-1	1	10	-1	-1
F	-1	6	-1	3	-1	-1

Figure 10 (COMERT, 2015)

In this example we can see a weighted undirected graph and then calculate the shortest weight from node A to other nodes.

3. System Design and Implementations

System design is one of the most important factors for making successful structures for all kinds of technical products. System design is the process of identifying, developing, and designing systems that meet the specific needs of the project. A systematic approach is necessary for a consistent and well-functioning system. In our application for defining the technologies, we have analyzed the environment that our application will run, and also the algorithms that we have used for solving the different problems mentioned before.

After analyzing the users, environment, and other criteria we have decided to create a web-based application. Considering the people who enter the facility that they are not familiar with, we analyzed the different options for all kinds of users.

After all these considerations and evaluations, we decided the web-based application is the best option in our case. In Figure 11 we can see general architecture of our web application. First, let's check the system architecture and analyze the technologies we have used.

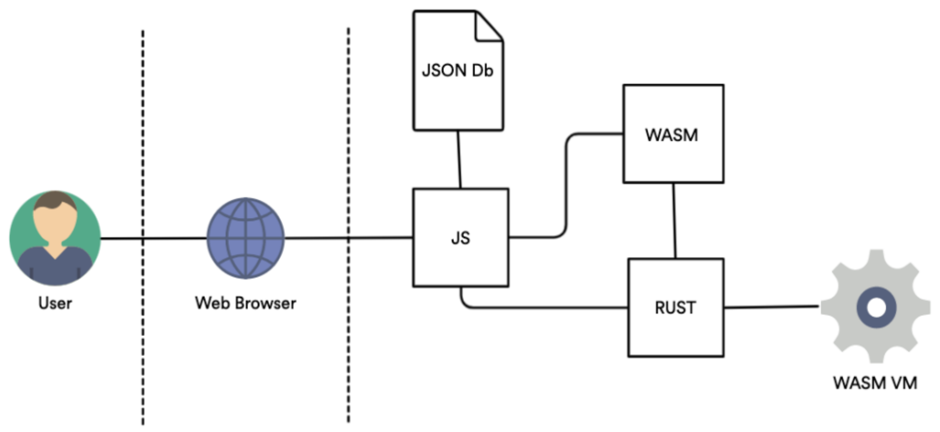


Figure 11

3.1. Technologies

3.1.1. Web Application

Web-Based Software is applications that run between the server and the client on the web browsers. The difference between web-based software compared to desktop programs and native mobile programs is that they could be accessed through a server and could be processed without the need to install programs from any computer or mobile phone. Now, thanks to the web software running on every platform, the application can be accessed smoothly. With the understanding of the various benefits that web-based software provides to users, their use is increasing rapidly and has been continuously improved. Another most important advantage of web-based software is security. In desktop software, the user should have taken precautions against security vulnerabilities. However, in web-based software, the software provider will carry out the monitoring of the security vulnerabilities. Therefore, there are fewer vulnerabilities than desktop software. Today, companies prefer web-based software for many jobs from announcements to business follow-ups. In this way, it provides fast access to employees and customers. Considering all these new trends and advantages we have decided web application environments are the best option for our application.

As we know today there are plenty of libraries and frameworks that are used for developing web-based applications. In our indoor navigation application, we have decided to use the ReactJS library.

3.1.1.1. React JS

React is the most preferred JavaScript library for creating user interfaces. Looking at its features, ReactJS has functions such as testable, server-side rendering, one-way data binding, simplicity, and a low learning curve. Another important functionality is testability in the React function which serves to determine how a component is rendered and behaved. Thanks to this function, it is very easy to test and debug applications. Server-side rendering is used to pre-create the initial states of components on the server-side. With SSR, the server has a page HTML ready to display the answer to the browser. In this way, the browser is able to load and execute JavaScript very quickly. This also allows the website to load fast. Simplicity is a feature that the library gains because of the use of JSX files simplifying the application. Learning is also facilitated by a component-based approach and different lifecycle methods. Considering all these reasons we have chosen ReactJS for the front-end development.

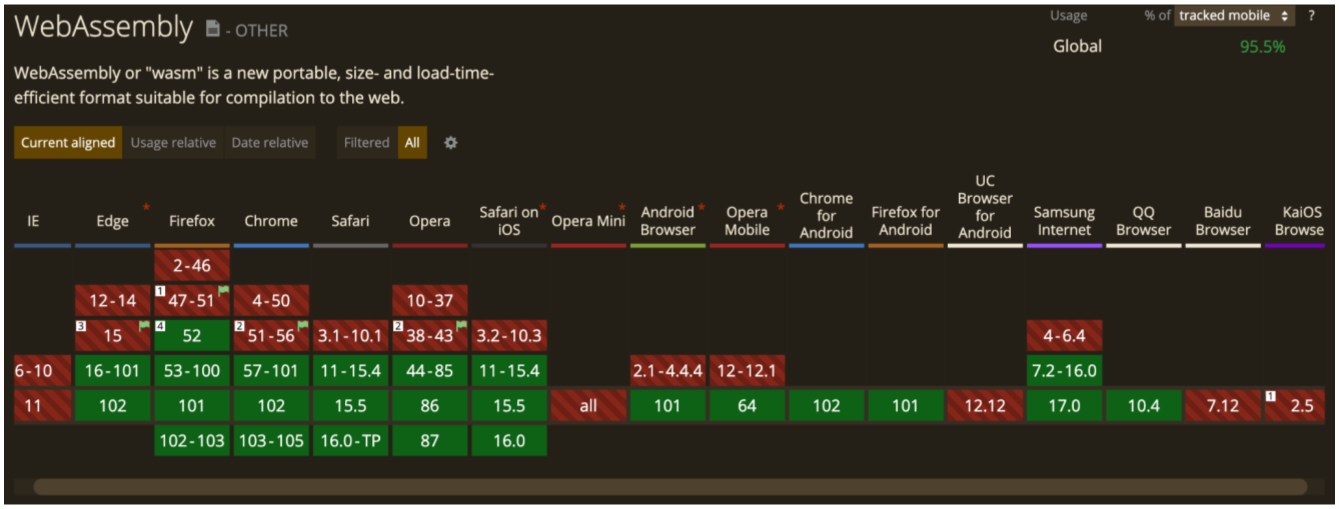
3.1.2. WebAssembly

WebAssembly is a common binary and text format for the web. WebAssembly, called wasm for short, has a binary format that is lower than JavaScript. With WebAssembly, or WASM for short, we can compile, run and migrate programs that are written in high-level languages such as C, C, and Rust in web browsers. Where JavaScript is insufficient in terms of performance, we can do 20 times more performance with WASM.

WASM can be thought of as a companion tool to JavaScript. There is no complete abandonment of JavaScript. Where JavaScript is flexible, dynamically written, and delivered with readable source code, WASM is written in a high-speed, power-save way, and is delivered through a compact binary format.

While not as good as the native performance that WASM offers, modern JavaScript engines are also quite fast. They can still perform well enough for most basic web applications and simple scripts that don't require large amounts of data compression. There is no need to use WASM for a simple operation that we can do easily with JavaScript. For example, triggering an alert when a user clicks a button can be easily done by JavaScript, it doesn't have to be as performant as it will be executed a million times per second, and it certainly doesn't need to be written in C++, C, or Rust.

However, when you want to develop a desktop application to run in a web browser, WASM will be the preferred format. However, we still need JavaScript to call WASM methods. Programs that host complex and high-CPU processes are often desktop applications. If we classify desktop applications as native, WASM is aimed to develop applications that are close to the native performance and run in the web environment. Many popular web browsers today support WASM. In Figure 12 (caniuse, n.d.) we can see the browser support version for WASM.


WebAssembly - OTHER

WebAssembly or "wasm" is a new portable, size- and load-time-efficient format suitable for compilation to the web.

Usage % of tracked mobile

Global 95.5%

IE	Edge	Firefox	Chrome	Safari	Opera	Safari on iOS	Opera Mini	Android Browser	Opera Mobile	Chrome for Android	Firefox for Android	UC Browser for Android	Samsung Internet	QQ Browser	Baidu Browser	KaiOS Browse
		2-46														
	12-14	1 47-51	4-50		10-37											
	3 15	4 52	2 51-56	3.1-10.1	2 38-43	3.2-10.3							4-6.4			
6-10	16-101	53-100	57-101	11-15.4	44-85	11-15.4		2.1-4.4.4	12-12.1				7.2-16.0			
11	102	101	102	15.5	86	15.5	all	101	64	102	101	12.12	17.0	10.4	7.12	1 2.5
		102-103	103-105	16.0-TP	87	16.0										

Figure 12 (caniuse, n.d.)

Before WASM, native performance was provided with Google's Native Client and Mozilla's asm.js technologies. Naturally, Google's focus was on Chrome and Mozilla on Firefox. First, they had developed a system that would work in their own browsers. However, both technologies have some disadvantages and obviously could not run in all environments. Therefore, in 2015, an international community World Wide Web Consortium (W3C) which is established to develop web standards, organized a WebAssembly Community Group meeting with the participation of Google, Mozilla, Apple, and Microsoft. Now, native performance is supported on all popular browsers. The implementation of WASM support in web browsers is based on asm.js and PNaCl's distributed executable concept.

3.1.2.1. *How Does WebAssembly Work?*

WASM is, in the words of its creators, a "compilation target" (Yegulalp, 2022). We don't need to write WebAssembly directly. We write in the high-level language of our choice, and it is then compiled as a WASM bytecode. The byte code is then executed on the client, where it is translated into native machine code and executed at high speed. WASM is not exactly an assembly language as it is not designed for any specific machine. It is for browsers, and when delivering code to be executed in the browser, we do not know what kind of machines your code will run on. When the scanner downloads the WASM code, it quickly converts it into a language that any machine can understand.

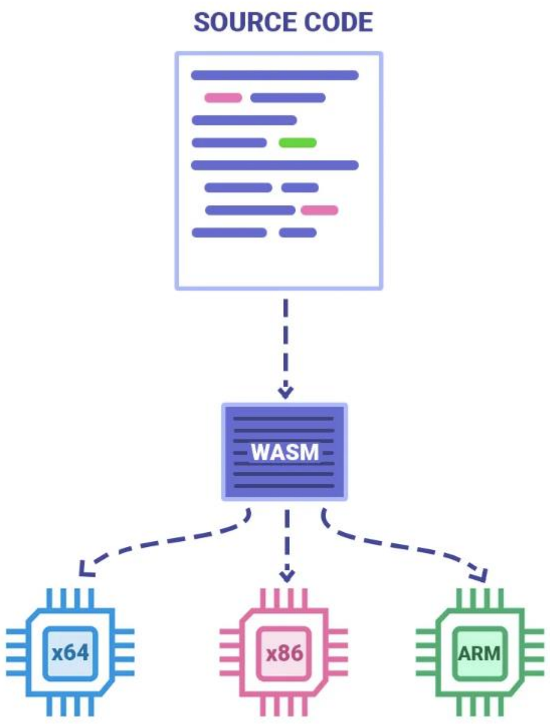


Figure 13 (Mihajlija, 2018)

The code we write in C, C++, Rust, or in any other low-level language is converted into .wat files, which are the WASM text format. The browser is presented with the .wasm file, which is the binary version of this file.

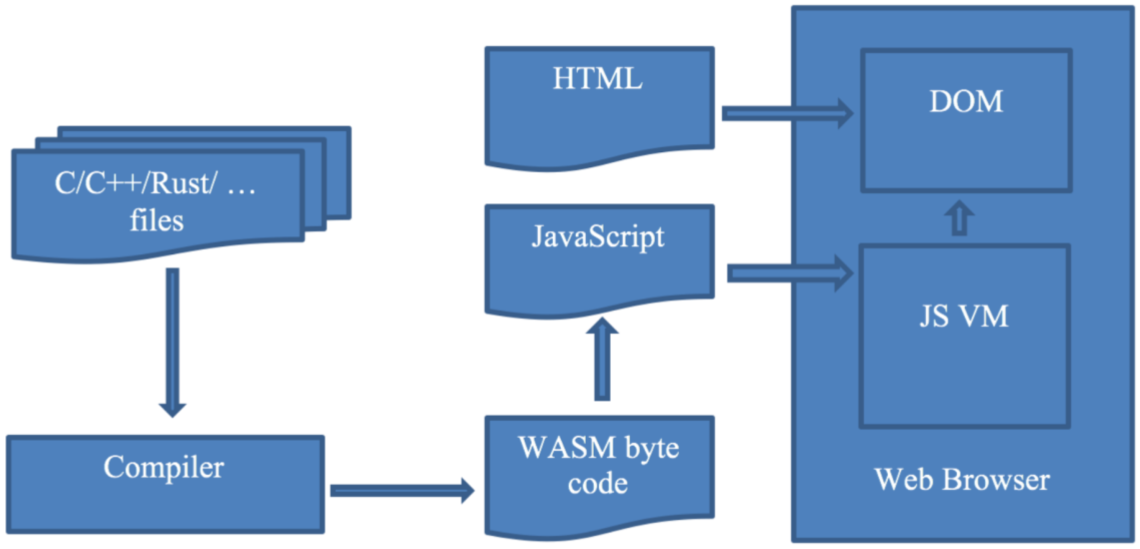


Figure 14 (Maciejak, 2017)

In Figure 1.1 we can see the general flow diagram of wasm code that runs in a web application.

3.1.2.2. Wasm application in our methodology

The one main superiority of our application is to create an application for decoding QR codes as fast as possible. Therefore, for achieving this goal we have decided to use WASM over the JS library for decoding QR codes. If we look at the research done by Efendi, we can easily observe the QR code detector which has been done by wasm the way better than the pure JS library in all criteria. (Efendi, 2020)

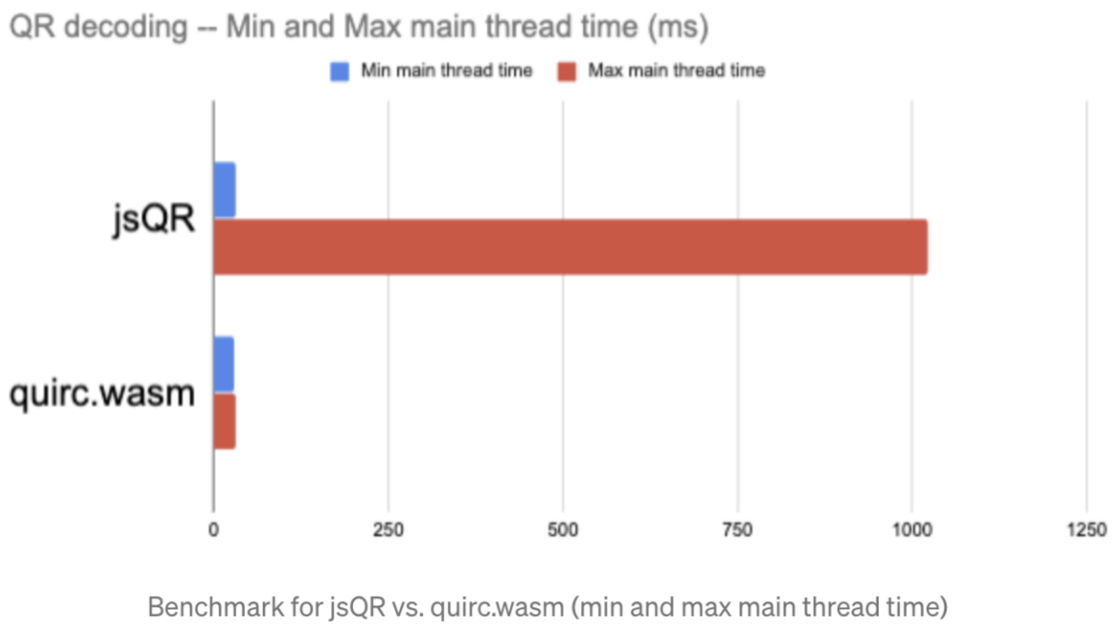


Figure 15 (Efendi, 2020)

In Figure 15 (Efendi, 2020) we can see a comparison of min and max main thread time (ms) for jsQR and quirc.wasm libraries. As we observe the max main thread for jsQR is near 1000 ms, on the other hand, quirc.wasm library shows way better performance in max time.

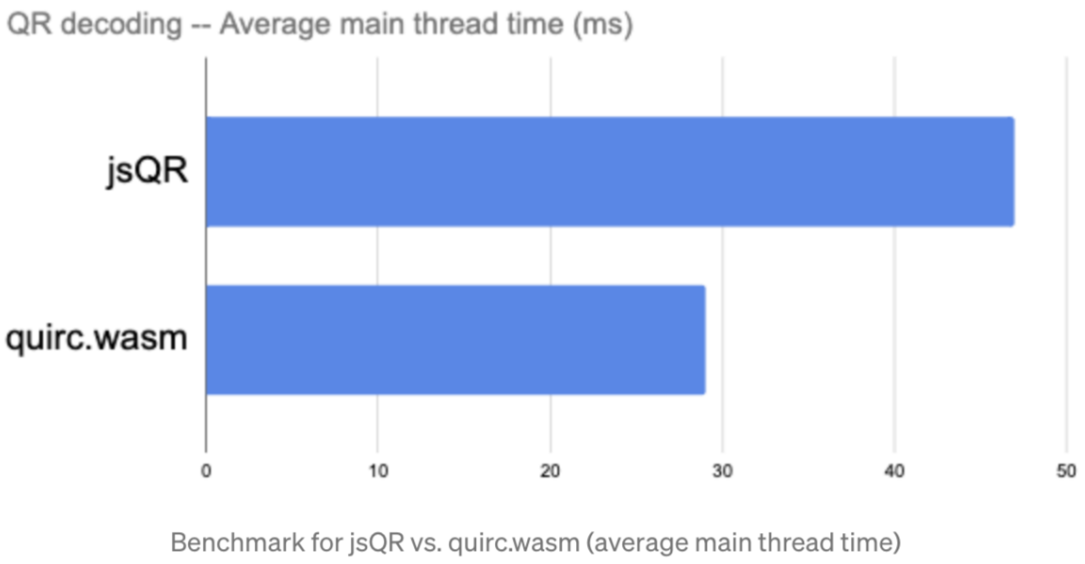


Figure 16 (Efendi, 2020)

Figure 16 (Efendi, 2020) shows the average main thread time for quirc.wasm and jsQR which is equal to 47 ms for jsQR and 28 ms for quirc.wasm. This test has been done on MacPro 2018 with 6xCPU and can display the different result with different devices.

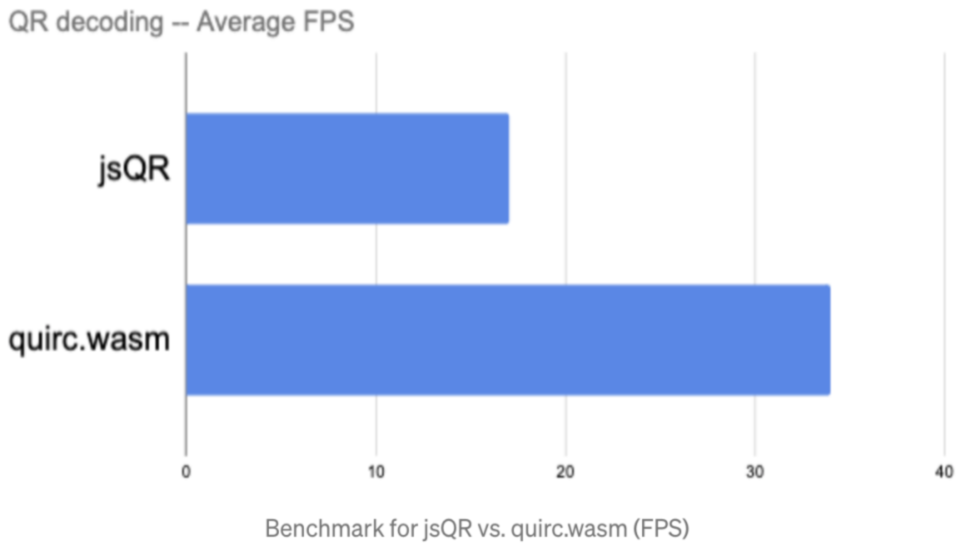


Figure 17 (Efendi, 2020)

Lastly, in Figure 17 (Efendi, 2020) there is FPS (frame per second) result for jsQR and quirc.wasm libraries. The results are 17 FPS and 34 FPS respectively for jsQR and quirc.wasm. These results can be different for different software and devices. However, it’s clearly noticeable that quirc.wasm overpowered jsQR in these parameters also.

3.1.3. Rust

As we have mentioned before WASM supported by most low-level languages also some high-level languages with the use of third-party libraries. In our application we have chosen Rust for developing WebAssembly binary. The Rust programming language is a system programming language with multiple development approaches with memory security priority, specifically designed to perform asynchronous tasks reliably. As multiple development approaches: it incorporates the capabilities of simultaneous, functional, generic, imperative, and structural programming languages. Although it is syntactically like C, it performs secure memory management with high performance. Rust is designed by Graydon Hoare who works for Mozilla Research and contributed by several famous developer such as Brendan Eich, the developer of JavaScript, and Dave Herman, the developer of the internet browser engine Servo with asm.js. The designers of the Rust language have improved and added new features to the Servo scanner engine when writing it. Rust, which has a free and open source compiler, has managed to become the "most loved software language" in the Stack Overflow developer surveys for the last 4 consecutive years.

The Rust language is designed for the development of highly reliable and large-scale systems with a predominance of asynchronous processes. Therefore, it has a set of operations that includes many features such as reliability, control of memory layout, and concurrency.

Since the Rust language is designed to be memory-safe; It does not include features such as null pointers, dangling pointers, and data races. Therefore, variables can only be assigned through given sets of forms (struct, enum, etc.). These form sets require that the input values have already been given to the form. Instead of a NULL statement, the option type is included in Rust to provide the function in data structures such as linked list and binary tree found in other languages. This type contains two testable data, either Some or None. There is also a borrowing control in Rust called borrow checker to manage the object lifecycle.

Perhaps one of the biggest features of Rust is the lack of a garbage collection system where pointers to dead objects are automatically removed, as in Java and .NET. Instead, memory and other resources are governed by the RAII (resource acquisition is initialization) principle, along with optional atomic reference counting. In this way, Rust enables the management of resources with a very small workforce. In addition, Rust simplifies the allocation of variables on the stack and does not allow the automatic conversion (boxing) from int to Integer available in other languages such as Java.

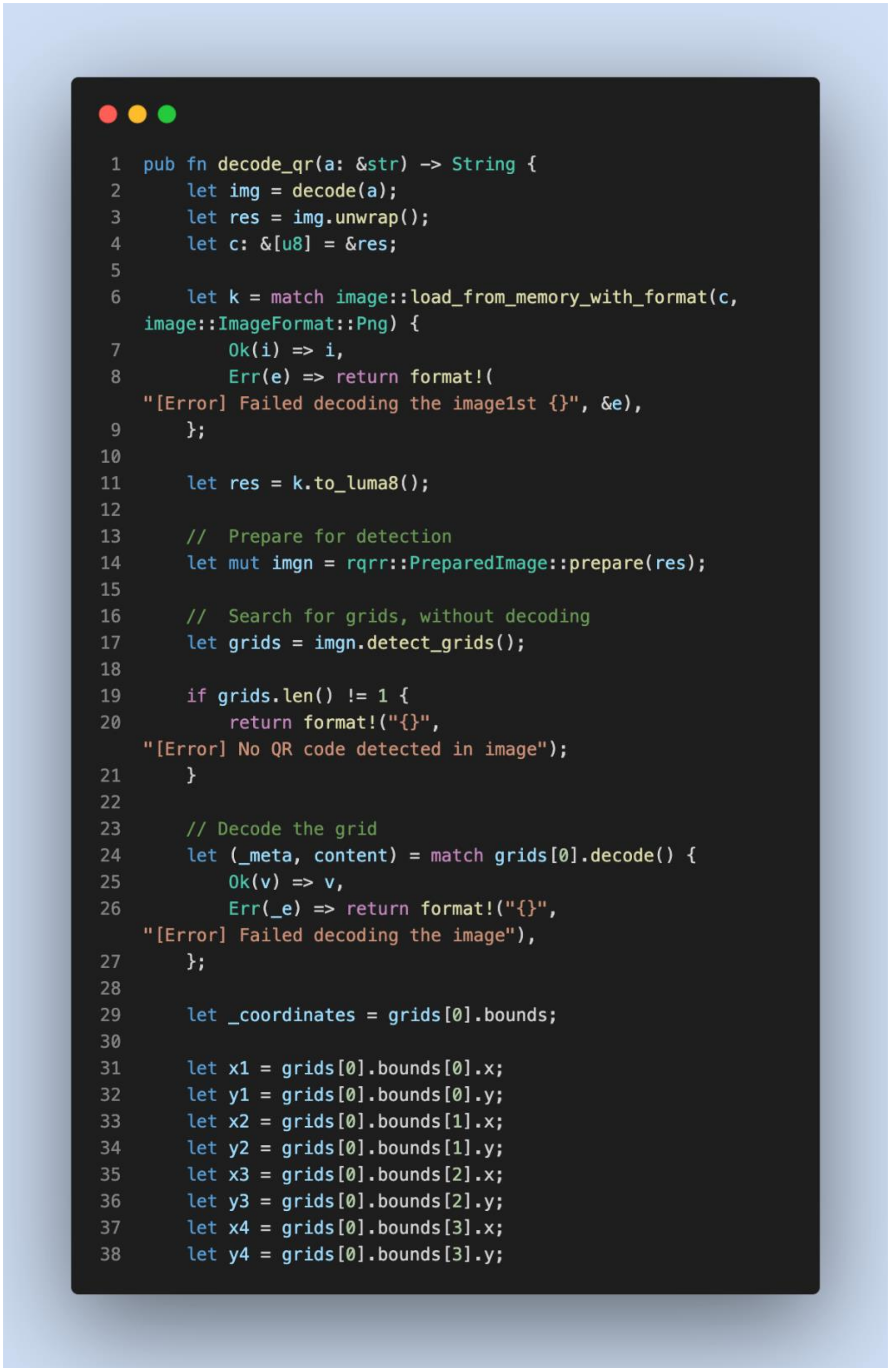
There is also the concept of reference using the & (ampersand) symbol. These references are not included in the reference count at run time. Instead, they are validated by the borrow checker at compile time. In this way, dangling pointers or other unexpected behaviors are prevented.

Rust has an ownership mechanism where each data has a unique owner. In this system, the scope (workspace) of the data is the same as the scope of the owner. Thus, when the scope of the data is completed, the owner is also wasted. Using the &T operator, values can be assigned to variables by immutable references. As a mutable referral operator, &mutT or just T can be used. There are always multiple immutable references in the app, or only one mutable reference. The Rust compiler enforces these rules at compile time by checking whether references are assigned correctly.

We have discussed the technical side of Rust language and its advantages over different languages. Rust provides so many functions and libraries for solving high-cost problems in an efficient way. We have chosen Rust languages for QR code image decode and analyzing considering it's quick running and easy development and integration with Webassembly.

3.1.3.1. Rqrr library

For detecting and decoding QR code images and their four grids we have used the rqrr library in rust. Rqrr library was made on the base of quirc library. We have already mentioned quirc library above and its comparison with the native JS library for decoding QR code. As we know from QR code analyzing, does not matter from which side of the QR code we are scanning it always returns accurate content of the QR code to us. However, we interest not only in the content of the QR code but also in the coordinates of each 4 corners of the QR code, which will help us to determine which side of the visual marker we are looking at. Therefore, we have implemented this library in our project and tested it in different environments and positions for checking if it detects correctly. After getting the QR code border coordinates we draw circles around each border with the different colors on our canvas for visually understanding and observing easily while testing. We tested our results with different devices and with various positions and we got the result we wanted thanks to rqrr library.



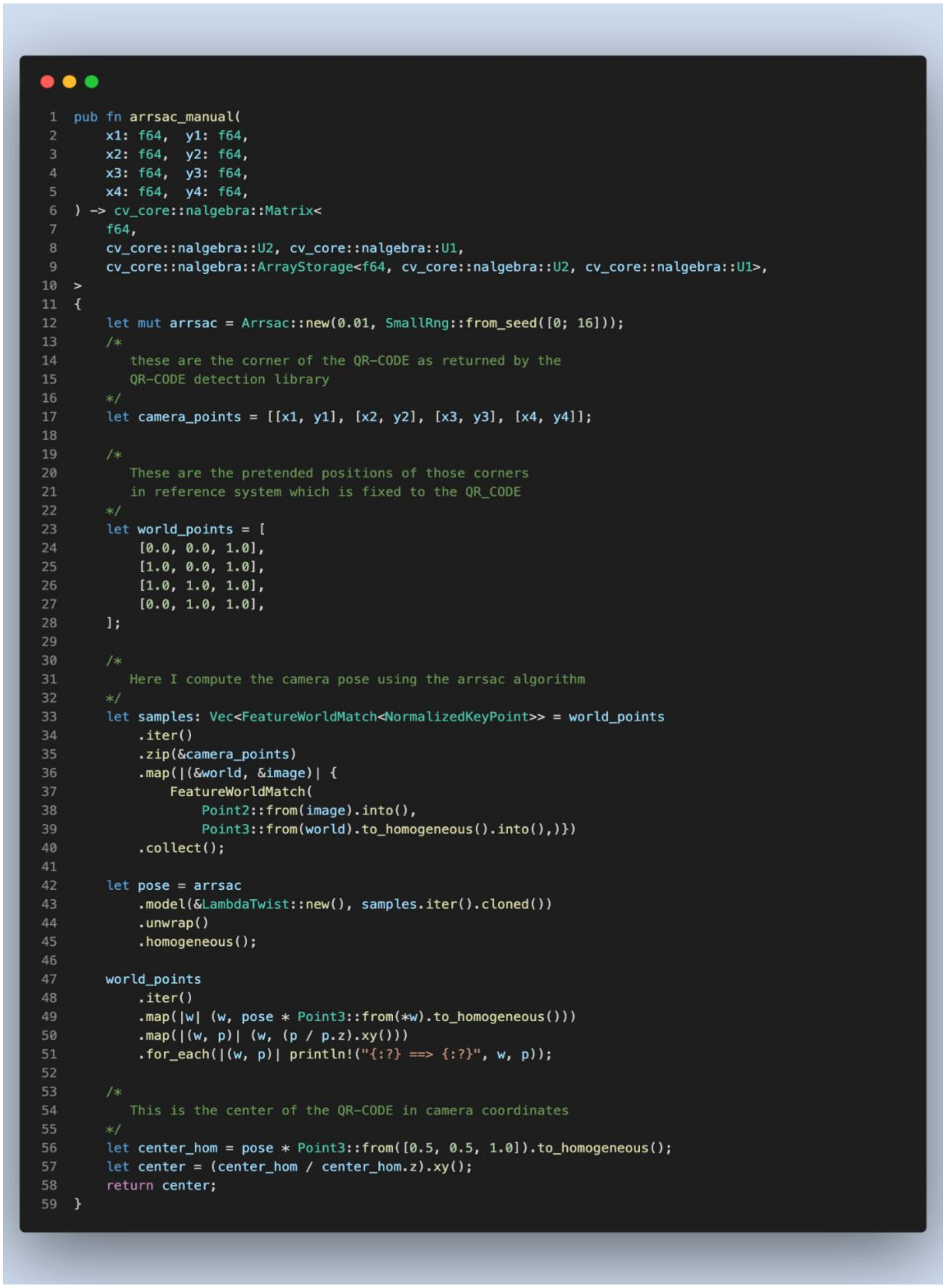
```rust
pub fn decode_qr(a: &str) -> String {
    let img = decode(a);
    let res = img.unwrap();
    let c: &[u8] = &res;

    let k = match image::load_from_memory_with_format(c,
image::ImageFormat::Png) {
        Ok(i) => i,
        Err(e) => return format!(
"[Error] Failed decoding the image1st {}", &e),
    };

    let res = k.to_luma8();

    //  Prepare for detection
    let mut imgn = rqrr::PreparedImage::prepare(res);

    //  Search for grids, without decoding
    let grids = imgn.detect_grids();

    if grids.len() != 1 {
        return format!("{}",
"[Error] No QR code detected in image");
    }

    // Decode the grid
    let (_meta, content) = match grids[0].decode() {
        Ok(v) => v,
        Err(_e) => return format!("{}",
"[Error] Failed decoding the image"),
    };

    let _coordinates = grids[0].bounds;

    let x1 = grids[0].bounds[0].x;
    let y1 = grids[0].bounds[0].y;
    let x2 = grids[0].bounds[1].x;
    let y2 = grids[0].bounds[1].y;
    let x3 = grids[0].bounds[2].x;
    let y3 = grids[0].bounds[2].y;
    let x4 = grids[0].bounds[3].x;
    let y4 = grids[0].bounds[3].y;
```

Figure 18

In Figure 18 we can see the source code we have used for detecting the border coordinates of QR code. For using this library in practice we just send our image on base64 format and then used load_from_memory_with_format() function for converting to image with the desired format. And then return the copy of the image in black and white format with the help of the to_luma() function. The purpose of this is to send images to the PreparedImage structure which works with black and white images for detecting QR codes. And then we call the detect_grids() function which helps us to detect if there is any grid in the given image or not. If we found any grid, then we take the x and y coordinates of the first four bounds of the grid which are the coordinates of the QR code borders we were looking for. By finding the coordinates and the border of the QR code we have finished one of the most difficult tasks for detecting the correct direction for the user.

3.1.3.2. Application of Lambda Twist library in Rust.

The second rust library we have used is Lambda Twist which we have already explained theoretically in the previous section. As we mentioned before Lambda Twist is one of the most accurate and fast P3P solvers. In a short form, it helps us to determine the camera position and orientation from 3D to 2D space.



```rust
pub fn arrsac_manual(
    x1: f64,  y1: f64,
    x2: f64,  y2: f64,
    x3: f64,  y3: f64,
    x4: f64,  y4: f64,
) -> cv_core::nalgebra::Matrix<
    f64,
    cv_core::nalgebra::U2, cv_core::nalgebra::U1,
    cv_core::nalgebra::ArrayStorage<f64, cv_core::nalgebra::U2, cv_core::nalgebra::U1>,
>
{
    let mut arrsac = Arrsac::new(0.01, SmallRng::from_seed([0; 16]));
    /*
       these are the corner of the QR-CODE as returned by the
       QR-CODE detection library
    */
    let camera_points = [[x1, y1], [x2, y2], [x3, y3], [x4, y4]];

    /*
       These are the pretended positions of those corners
       in reference system which is fixed to the QR_CODE
    */
    let world_points = [
        [0.0, 0.0, 1.0],
        [1.0, 0.0, 1.0],
        [1.0, 1.0, 1.0],
        [0.0, 1.0, 1.0],
    ];

    /*
       Here I compute the camera pose using the arrsac algorithm
    */
    let samples: Vec<FeatureWorldMatch<NormalizedKeyPoint>> = world_points
        .iter()
        .zip(&camera_points)
        .map(|(&world, &image)| {
            FeatureWorldMatch(
                Point2::from(image).into(),
                Point3::from(world).to_homogeneous().into(),)})
        .collect();

    let pose = arrsac
        .model(&LambdaTwist::new(), samples.iter().cloned())
        .unwrap()
        .homogeneous();

    world_points
        .iter()
        .map(|w| (w, pose * Point3::from(*w).to_homogeneous()))
        .map(|(w, p)| (w, (p / p.z).xy()))
        .for_each(|(w, p)| println!("{:?} ==> {:?}", w, p));

    /*
       This is the center of the QR-CODE in camera coordinates
    */
    let center_hom = pose * Point3::from([0.5, 0.5, 1.0]).to_homogeneous();
    let center = (center_hom / center_hom.z).xy();
    return center;
}
```

Figure 19

In Figure 19 we can see the implementation of Lambda Twist in Rust. In the first step we take coordinates of QR code which are returned by the rqrr library and create a 2-dimensional array. Afterward, we create world points array which are the pretended positions of given corners in a reference system fixed to QR code. In the next step we take a world point array and our camera points 2-dimensional array for computing camera pose using ARRSAC (Adaptive real-time random sample consensus) algorithm. ARRSAC is mainly used to find robust estimation for computer vision problems. Next, we check if the computed matrix is capable of mapping pretended corners into the same pixels of the image. Lastly, we calculate and return the center of the QR code in camera coordinates with the calculated pose and coordinates. In the interface we use this center coordinates to define accurate direction for the user.

4. Component of System

4.1. Making the system with the connection of Rust, ReactJS, and Web Assembly

After understanding the general working principles of used technologies let's see utilization and connection of these technologies in our application. As we mentioned before our project is web-based application. In the first step we installed npm and cargo package managers for making our life easy while configuring and installing necessary dependencies. Cargo is a package manager for Rust which will be used for installing dependencies and connecting and building our rust code into webassembly. And npm is a package manager for JavaScript language which is used to create and run React applications. The general roadmap for application is like as below

1. Creating React Application
2. Creating Rust Application
3. Compile the code written in Rust into Webassembly
4. Make a connection between React application to Webassembly

For creating React Application we just run command *npx create-react-app react-client* which creates new react front-end application inside folder with the name react-client. Additionally, we run *npm run eject* command. Each web application has single build dependency which all configurations done automatically with the help of npm package manager. However, we need to load and connect our webassembly to react applications. Therefore, we need extra permission for configuring external dependencies manually and this command lets us achieve this goal. In the next step we create Rust application for Webassembly. For doing this we do following steps.

- Creating Rust project with the help of cargo

We run *cargo new wasm* for creating Rust project. In our case wasm is a name of rust project.

- Create rust functions which we will call from react

After creating Rust project, we add decode_qr (Figure 18) and arrsac_manual (Figure 19) functions which we have analyzed more detailed before.

- Using wasm-bindgen library in Rust

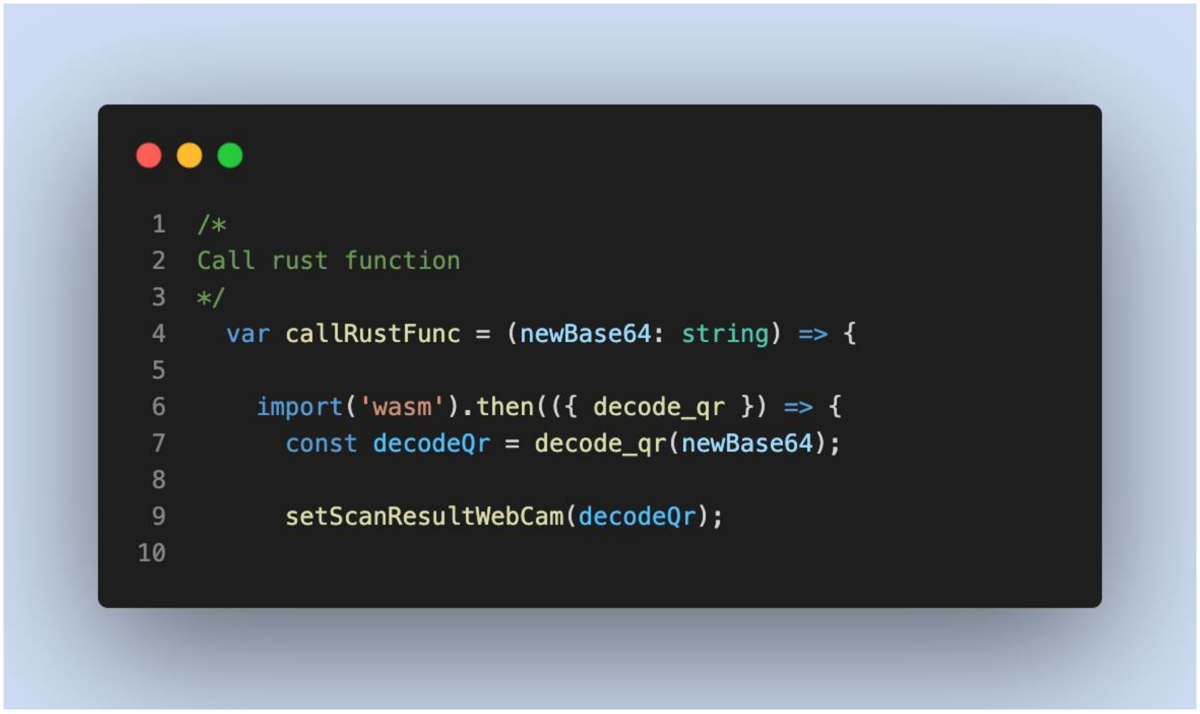
This library let us create two-way bridge between JavaScript and Rust and also between JavaScript and WebAssembly. We add this library to our cargo.toml file, which is package manager file for Rust, similar to package.json in JavaScript. For using this library, we just include #[wasm_bindgen] tag to the beginning of our functions.

- Using wasm-pack library

After finishing with Rust functions, we need to convert into wasm format for accessing from our react application. Wasm-pack is a library that allows us to create, test, and publish WebAssembly components created by Rust. Firstly, we install wasm-pack library with the use cargo package manager just running cargo install wasm-pack command. Later, we run wasm-pack build --out-dir ../wasm-build command which create wasm-build folder with the wasm and JavaScript wrapper file.

- Call Wasm from React project.

In the last step, we need just call the functions we implemented in Rust from React component. The implementation of this is shown in Figure 20



```
/*
Call rust function
*/
  var callRustFunc = (newBase64: string) => {

    import('wasm').then(({ decode_qr }) => {
      const decodeQr = decode_qr(newBase64);

      setScanResultWebCam(decodeQr);

```

Figure 20

With this last step, we finish configuring our WebAssembly project in ReactJS with Rust.

4.2. Demonstration of application

After configuring the connection of our technologies, in the next steps, we use the data which we get from Rust functions for creating a simple interface and dynamic map in React. As we know the main UI functionality of our application is detecting QR codes with a device camera. For this purpose, we use HTML5 Canvas for enabling the device camera. In each 10ms we capture the image with a camera canvas and send it to our Rust function for operating on it. However, before enabling the device camera, we create the main page with a list of destinations for the given infrastructure. We take the list of destinations from the JSON document database. We will analyze our JSON file in more detail in the next steps. In Figure 21 we can see our main page with the list of destinations for imaginary buildings.

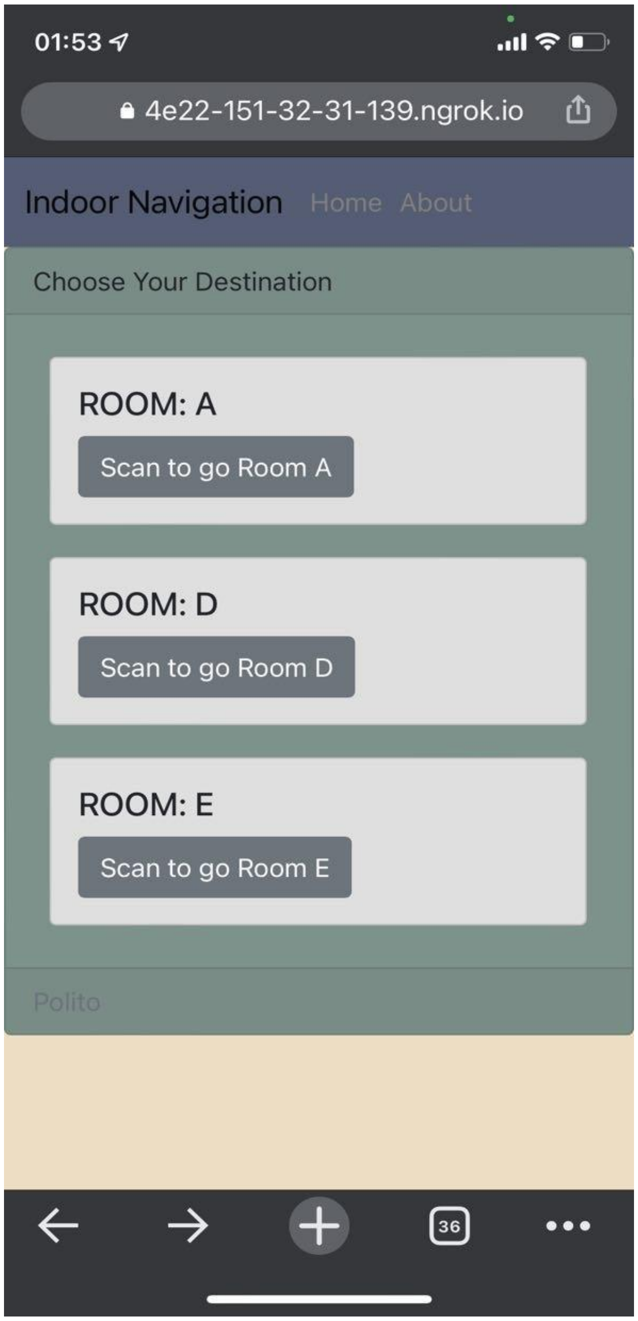

01:53
4e22-151-32-31-139.ngrok.io
Indoor Navigation
Home
About
Choose Your Destination
ROOM: A
Scan to go Room A
ROOM: D
Scan to go Room D
ROOM: E
Scan to go Room E
Polito
36

Figure 21

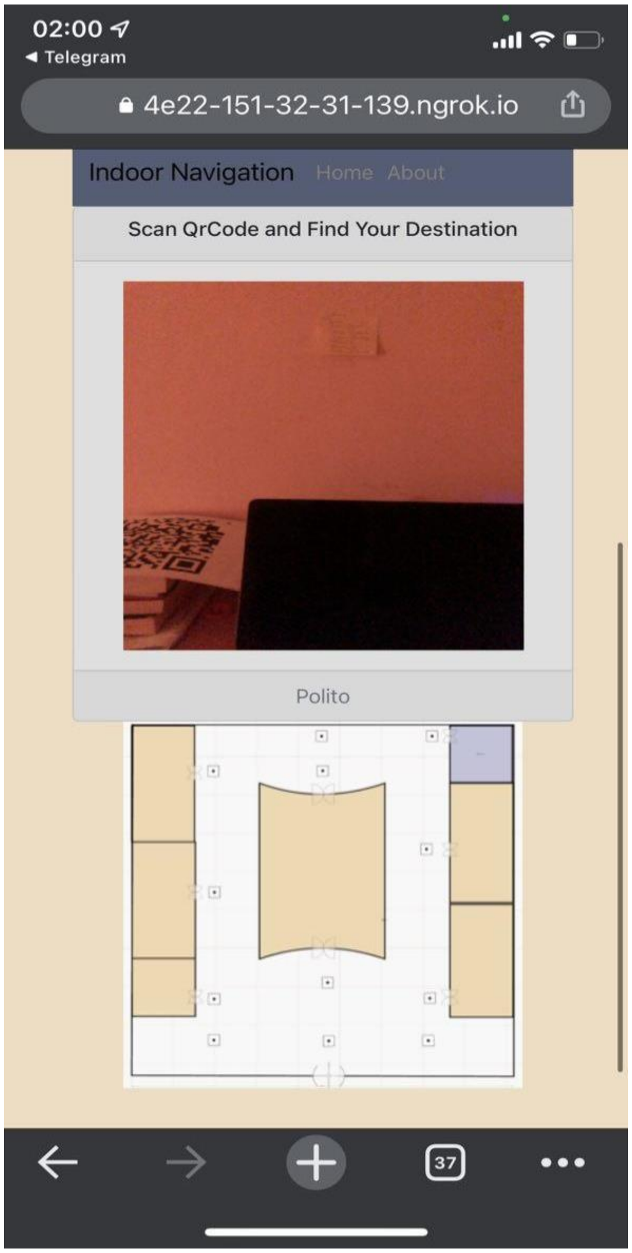

02:00
Telegram
4e22-151-32-31-139.ngrok.io
Indoor Navigation
Home
About
Scan QrCode and Find Your Destination
Polito
37

Figure 22

For choosing the desired destination from the given list, we click the button on the chosen field, and it forwards us to the next page (Figure 22). On the forwarding page, we face two main parts. On the above we see the camera and below we observe the map of the current building. On the map, we can also notice black dots in different positions which express the QR code position in a real building correspondingly on the map.

After giving camera access to the browser for using our application, the camera becomes ready to scan QR codes and detect the source, destination, path, and direction. Let’s assume we have chosen the Room D from the list as a destination. Then we find the nearest QR code (in our example it will be QR code with id 2) to us in the building and scan it. While scanning the marker on the surface we will observe an arrow which will show us the direction of next node in our path. Simultaneously we also see the update of the map below the camera. The corresponding node to the marker which we scan will appear with a green circle and destination node will appear with blue circle around it on the map. Additionally, we can see the path from the source node to the destination with red line. In the Figure 23, Figure 24, Figure 25, Figure 26 we will see the visualized result on our application from the scanned QR code position to the chosen destination (Room D) with finding shortest path.

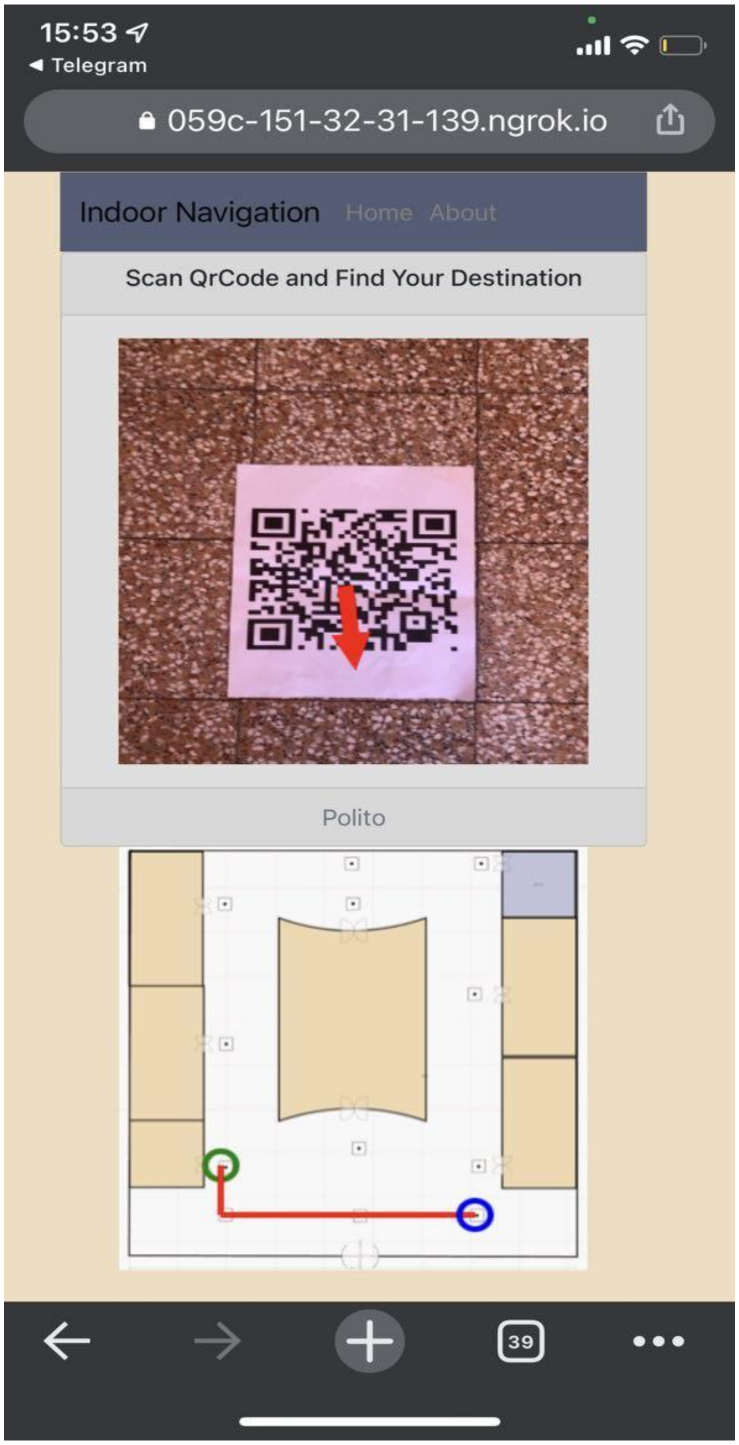

15:53
◀ Telegram
059c-151-32-31-139.ngrok.io
Indoor Navigation
Home
About
Scan QrCode and Find Your Destination
Polito
39

Figure 23

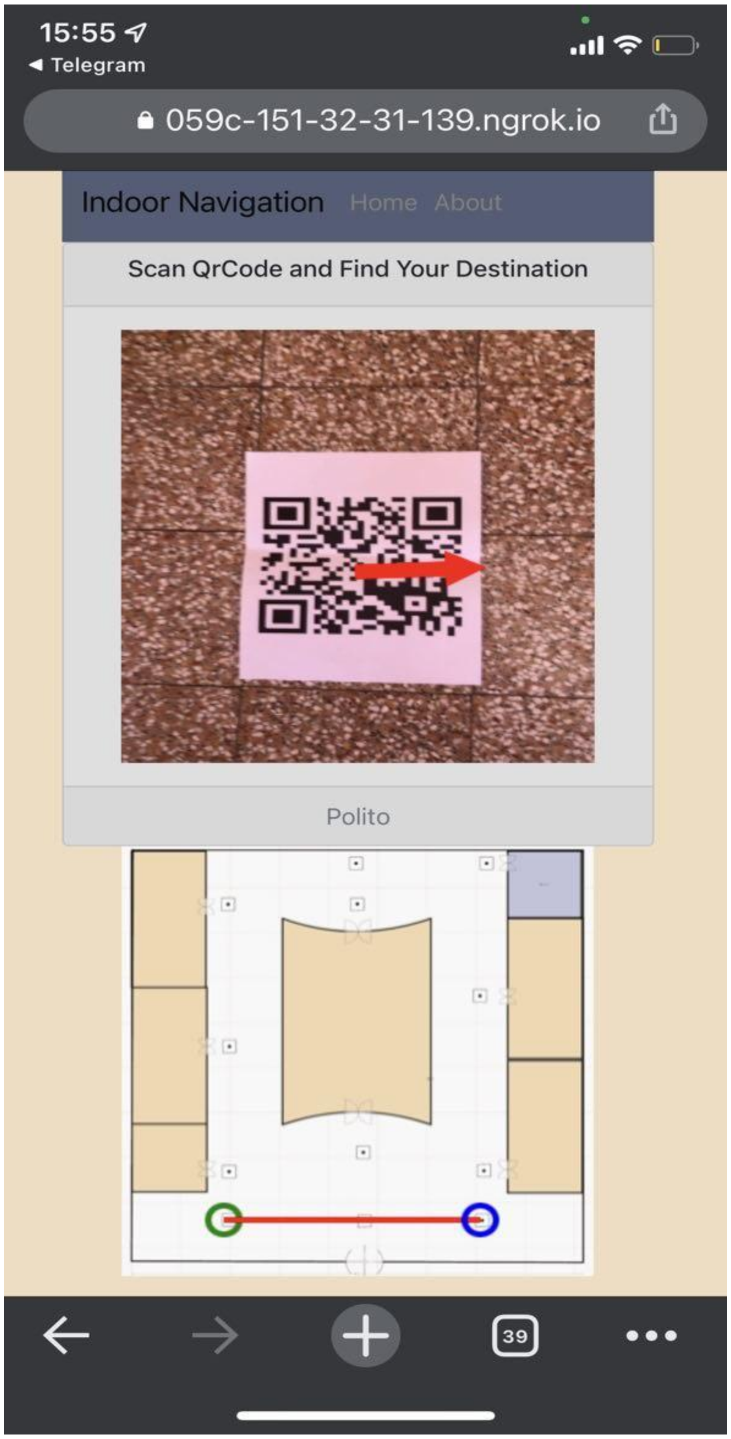

15:55
◀ Telegram
059c-151-32-31-139.ngrok.io
Indoor Navigation
Home
About
Scan QrCode and Find Your Destination
Polito
39

Figure 24

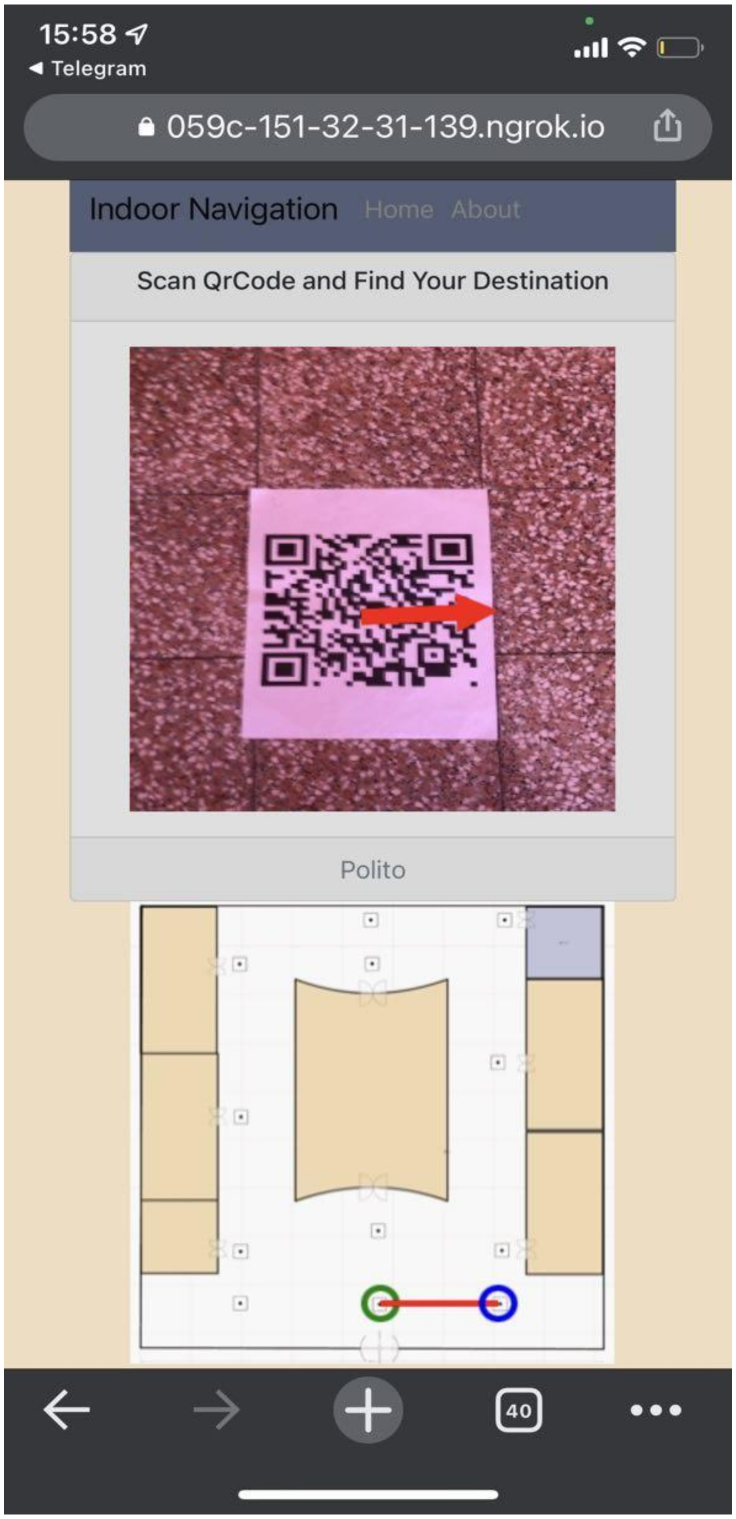

15:58
◀ Telegram
059c-151-32-31-139.ngrok.io
Indoor Navigation
Home
About
Scan QrCode and Find Your Destination
Polito
40

Figure 25

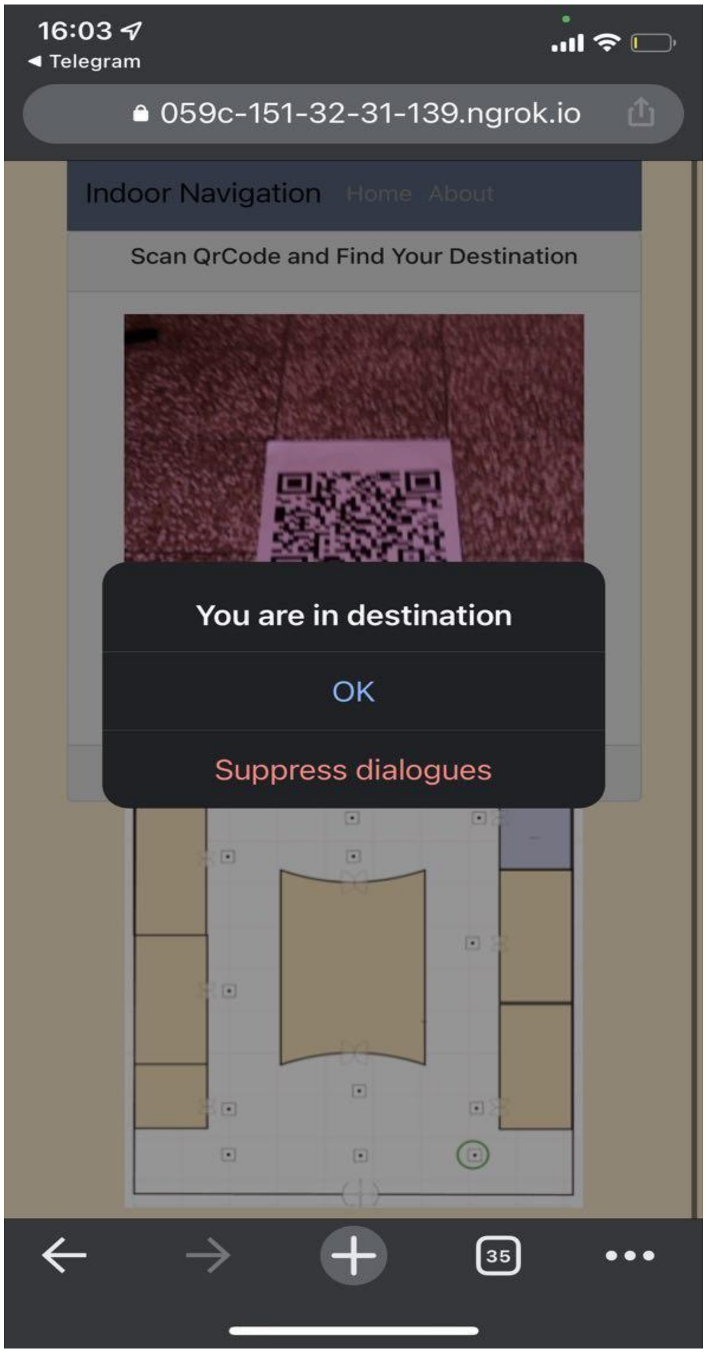


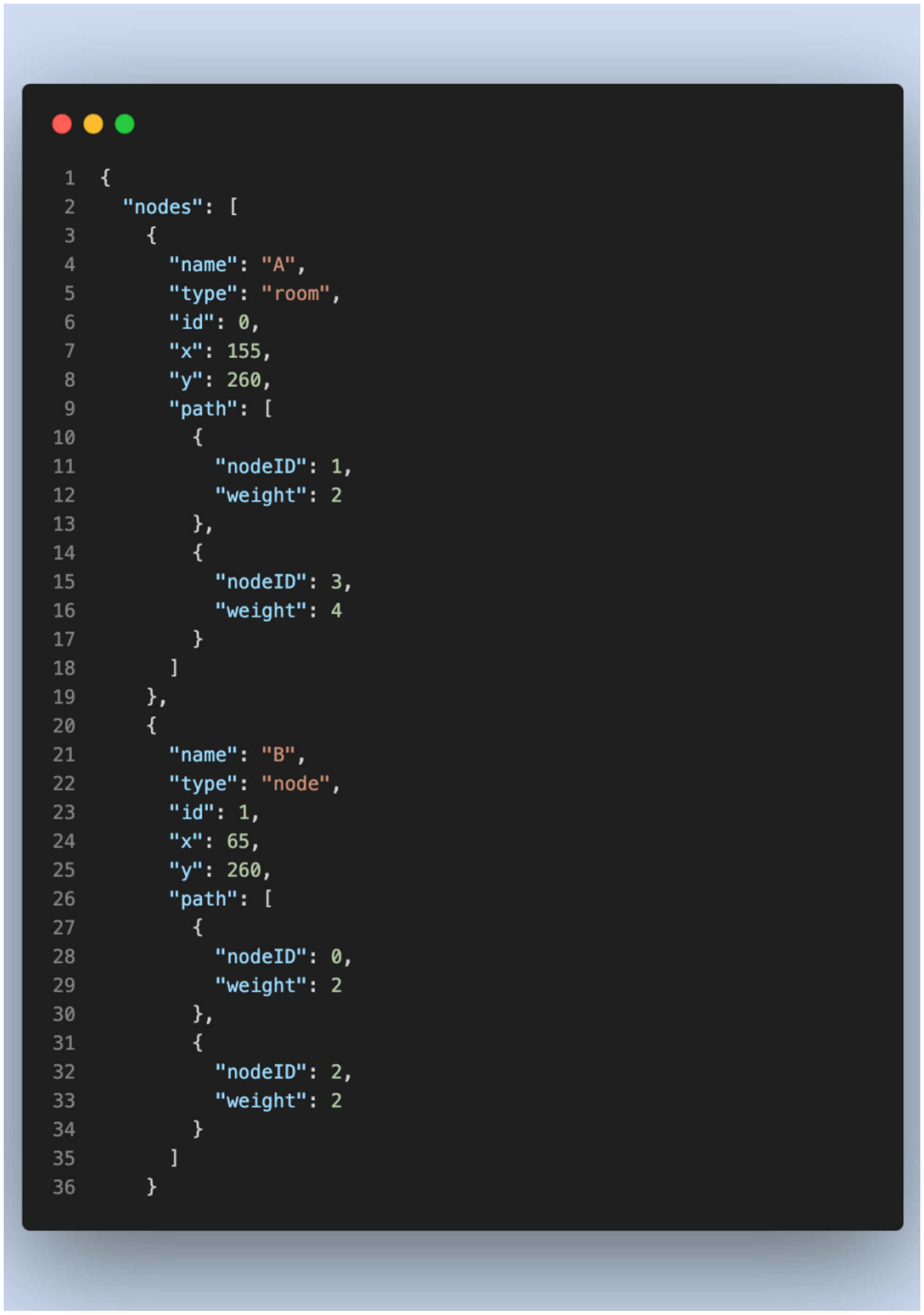
Figure 26

When we reach our chosen destination alert appears with the information that we are at our destination. Additionally, in the navbar section, there are Home and About sections. We can back to the main page when we need to choose a new destination or just check the general destination list. "About" section is just general information about the project and a simple explanation for using the application.

4.3. Design Decision and Choice of Database

Design decisions is one of them most important criteria for software development. Technical elements of the design phase can be listed as follows: Design Interface and Database choice.

While developing our application one of the main criteria is creating lightweight application as much as possible. For the first phase considering it's just a concept-based application, we just tried to use the lightweight technologies in all cases for demonstrating that our idea is applicable in the real environment. Therefore, for this stage we have created only the front-end base application for developing and demonstrating application in a quick way. On the other hand, not using backend-service on the web application has some advantages in case of cost and speed. Since we will not need to spend extra cost for running backend services and there will not lose time for data transferring from backend service to front-end. If there is no backend service, we will not run the standard database technology as it is supposed to. We have used JSON file as a database for storing the data about the coordinates of QR codes and their different parameters for the given building. JSON (JavaScript Object Notation) is a standalone data interchange format designed to represent simple data structures. It is mainly used for data exchange between two systems. JSON can be used in many applications. Its use is especially common for transferring data between servers and web applications. This is because JSON is text-based, and applications can usually retrieve data only as text. In Figure 27 we can see the general structure of our JSON file database.



```json
{
  "nodes": [
    {
      "name": "A",
      "type": "room",
      "id": 0,
      "x": 155,
      "y": 260,
      "path": [
        {
          "nodeID": 1,
          "weight": 2
        },
        {
          "nodeID": 3,
          "weight": 4
        }
      ]
    },
    {
      "name": "B",
      "type": "node",
      "id": 1,
      "x": 65,
      "y": 260,
      "path": [
        {
          "nodeID": 0,
          "weight": 2
        },
        {
          "nodeID": 2,
          "weight": 2
        }
      ]
    }
```

Figure 27

In the example given above we can see arrays object with the different properties. There is the nodes array in our object which consist set of nodes object. The node object in the array represents our QR code details with multiple property.

Each object in the nodes array has id, type, name, x, y and path properties. Let’s analyze each property separately and the usage of it in our application. Id property is unique for each QR code and helps us to differentiate and detect which QR code we are scanning. Also, we use this id while creating QR code content. Second important property is the type which the value of it can be either room or node. As we have discussed before not all the QR code represents the destination of the building. Some QR codes play the role of a bridge for making the connection between nodes, and the value of the type of property for these nodes is “node”. On the other, if the value for the type of property is “room” then we label this node as room and display the name on the destinations list. X and Y properties represent coordinates of the node on the canvas map coordinate system. Lastly, the path property is an array of connected nodes and is used to store connected node parameters such as id and weight to the current node for making the graph structure complete.

The last but not least important superiority of our application is its privacy policy. As today we know most navigation applications require to use user geolocation information. However, as we already discussed our methodology is not based on GPS navigation system or another third-party location detector technology. Therefore, the application does not require any geo location permission from the user for detecting its position. The only permission we require to open device camera for scanning the QR code. However, user is free to disable device camera after scanning the detected QR code and enable it again.

4.4. UML Use case Diagram

In Figure 28 UML Use case diagram of our application has been described for better understanding the how user will behave and interact with the system.

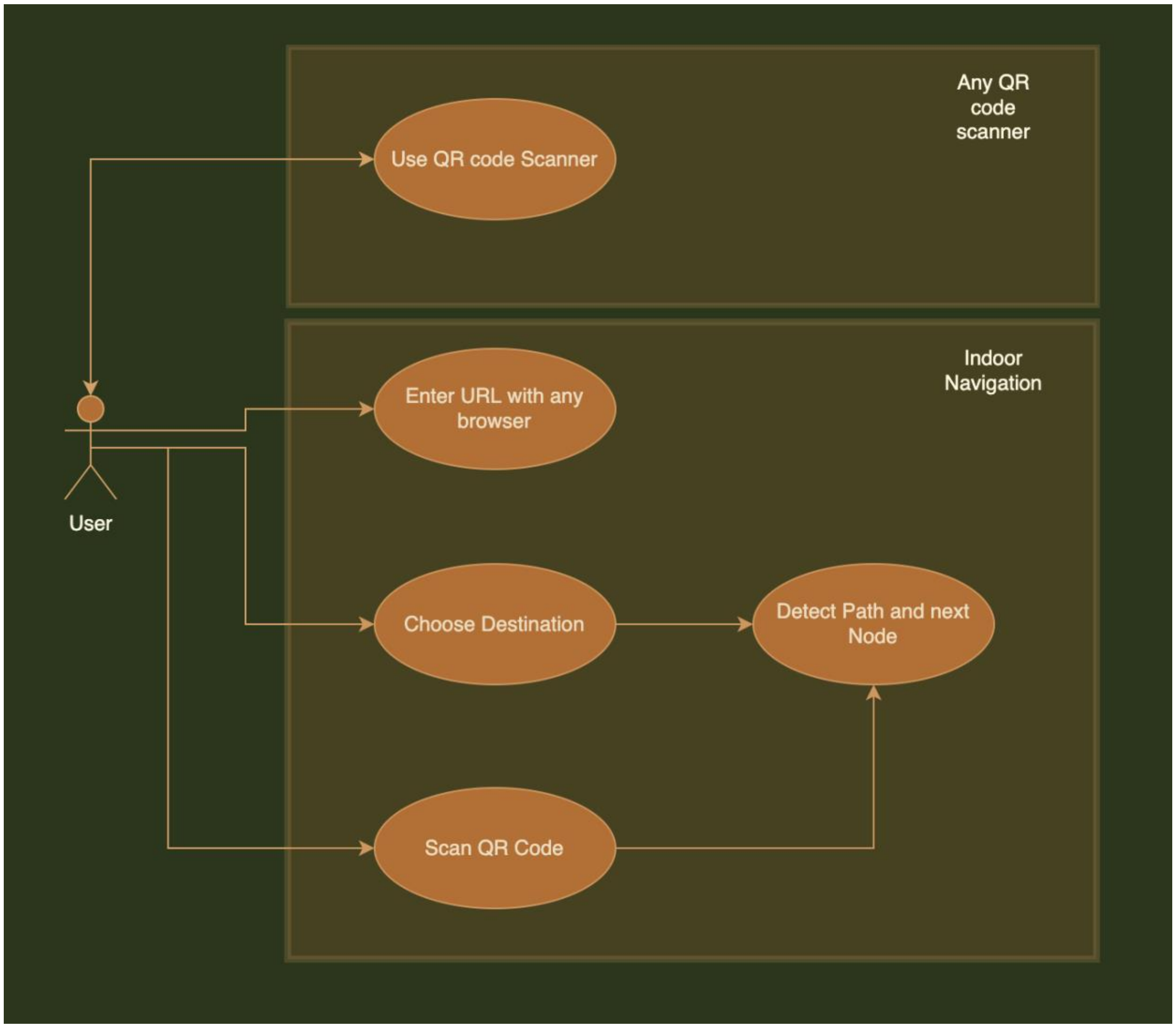


Figure 28

5. Conclusion and Future directions

5.1. Conclusion

In this work, we analyzed and studied the use of navigation systems for indoor infrastructure. Primarily we have researched already existing indoor navigation technologies and their working principles. The deep analyze and comparison of the existing solution investigated with their algorithms and advantages/disadvantages. After comparison of existing solution, we conclude that most existing indoor navigation technologies require more resources and costs while setting up a building. We introduced our approach to the problem after becoming familiar with the current solutions.

We have applied a visual-based navigation system in our methodology. The visual-based navigation system scans markers or objects in the surroundings using the camera of the device and to help identify orientation and position. Due to the distinctiveness of the QR code, which assisted us in solving the direction and location problem, it was employed as a visual marker in our research. Converting world point coordinates to camera coordinates was the first thing we did for our project. Since it is a known fact that objects are recognized in 3D in the real world but in 2D on computer screens, this issue is referred to as the PnP (Perspective n Point) problem and is one of the well-known problems in computer vision. In our thesis, we have examined this issue more deeply and offered some potential solutions. We have examined and compared various PnP solver algorithms to find a solution to this issue, and Rust's Lambda Twist Library was the result of our efforts with its great performance. Finally, we have applied the Dijkstra algorithm to the information we keep in our database, which includes a set of QR code positions with various properties. As a result, we determine the start, finish, and direction of the shortest path. To test our application, we constructed a hypothetical map with distinct coordinates and tested it with various source and destination points. Our methodology can be applied to any type of infrastructure with low cost and high performance, based on the experiments and studies that were conducted.

5.2. Future Directions

Our project is currently having some issues with scalability, the overall design of the website, and setting up augmented reality. To save time and resources, we did not develop a backend service during this early phase. Considering that this is merely design study, we have only attempted to demonstrate the applicability of our methodology. Nevertheless, one of the key future plans for our project is to develop backend services. We have used JSON files as databases due to the unavailability of backend services. However, this is not an effective method for efficiently storing large amounts of data or updating the data. Therefore, we are planning to use MongoDB (NoSQL database technology) to improve database design in the future. Additionally, applying augmented reality is another main future plan for our project. AR is used to add digital elements to the camera of user device for giving the impression that the holographic content we are viewing is a part of the real world. In our project AR functionality will help users visually better detect destinations in the surrounding environment and will help user to follow correct direction.

6. Bibliography

EPA. (2021, September 7). *EPA* . Retrieved from Environmental Protection Agency. : https://www.epa.gov/report-environment/indoor-air-quality#note1

WANG, F. (2013, December 3). *sciencedirect*. Retrieved from A comprehensive UAV indoor navigation system based on vision optical flow and laser FastSLAM: https://www.sciencedirect.com/science/article/abs/pii/S1874102913600804

Tunca, C. (2014). Retrieved from Yapay Sinir Ağları ile WiFi Tabanlı İç Mekan Konumlandırma: https://ab.org.tr/ab14/kitap/tunca_toplan_ab14.pdf

Lee Pieh Wen, C. W.-Y. (n.d.). Application of WiFi-based Indoor Positioning System in Handheld Directory System. *5th Proceedings of the European Computing Conference.*

Zegeye, W. &. (2016). WiFi RSS fingerprinting indoor localization for mobile devices. *10.1109/UEMCON.2016.7777834.* .

Tsanga, P. W. (2015). A Bluetooth-based In-door Positioning System: a Simple and Rapid Approach. *Annual Journal IIE, 35, 11-26.*

Gifford, M. (2018, October 19). *Indoor Positioning with Ultrasonic/Ultrasound*. Retrieved from https://www.leverege.com/blogpost/ultrasonic-indoor-positioning

Jianyong, Z. H. (2014). RSSI Based Bluetooth Low Energy Indoor Positioning.

Floreani, L. (2015). Retrieved from TheoremOne Blog: https://bits.theoremone.co/indoor-positioning-with-beacons/

Zou, H. X. (2014). *Platform and Algorithm Development for a RFID-Based IndoorPositioning System. Unmanned Systems.*

Belhadi, Z. (2014). *researchgate.* Retrieved from RFID Tag Indoor Localization by Fingerprinting Methods - Scientific Figure on ResearchGate:

https://www.researchgate.net/figure/RFID-localization-system-block-diagram_fig1_271909154

Xu, H. D. (2017.). *An RFID Indoor Positioning Algorithm Based on Bayesian Probability and K-Nearest Neighbor. Sensors.*

Morais, K. B. (2017). *RFID based Indoor Positioning System.*

Alarifi, A. A.-S.-H.-A.-K. (2016). *Ultra Wideband Indoor Positioning Technologies: Analysis and Recent Advances.*

Kucuk, K. (2017). Geniş Bant Konum Belirleme Sistemi Performans Analizi ve İyileştirilmesi. *SAÜ Fen Bilimleri Enstitüsü Dergisi*.

Ergen, E. I. (2018). Utılızıng Indoor Localızatıon Technologıes For Occupant Feedback Collectıon.

DENSO WAVE. (n.d.). Retrieved from QR Code development story,Technologies,DENSO WAVE: https://www.denso-wave.com/en/technology/vol1.html

OpenCV . (n.d.). Retrieved from Perspective-n-Point (PnP) pose computation : https://docs.opencv.org/4.x/d5/d1f/calib3d_solvePnP.html

Bolles., M. A. (1981). *andom sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography*. Retrieved from https://doi.org/10.1145/358669.358692

Rust. (n.d.). Retrieved from lambda_twist::LambdaTwist - Rust : https://docs.rs/lambda-twist/latest/lambda_twist/struct.LambdaTwist.html

Mikael Persson, K. N. (2018). Lambda Twist: An Accurate Fast Robust Perspective Three Point (P3P) Solver . European Conference on Computer Vision .

COMERT, Z. (2015). *zafercomert.* Retrieved from En Kısa Yol Problemi: http://www.zafercomert.com/Medya/2015_05_11_2_121_69f9a888.pdf *caniuse*. (n.d.). Retrieved from "webassembly": Can I use... Support tables for HTML5, CSS3, etc : https://caniuse.com/?search=webassembly

Yegulalp, S. (2022). *InfoWorld*. Retrieved from What is WebAssembly? The next-generation web platform explained : https://www.infoworld.com/article/3291780/what-is-webassembly-the-next-generation-web-platform-explained.html

Mihajlija, M. (2018). *WebAssembly: How and why* . Retrieved from blog.logrocket.com: https://blog.logrocket.com/webassembly-how-and-why-559b7f96cd71/

Maciejak, D. (2017). *fortinet*. Retrieved from WebAssembly 101: Bringing Bytecode to the Web: https://www.fortinet.com/blog/threat-research/webassembly-101-bringing-bytecode-to-the-web

Efendi, J. (2020). *Building 60 FPS QR Scanner for the Mobile Web* . Retrieved from https://medium.com/: https://medium.com/tokopedia-engineering/building-60-fps-qr-scanner-for-the-mobile-web-eb0deddce099