(Established under Karnataka Act No. 16 of 2013)
100-ft Ring Road, Bengaluru – 560 085, Karnataka, India

Project Phase 2 Report

3D Mapping and Navigation Bot using VIO SLAM

Submitted by

Harshith N Srivatsa (PES1201701183)

Karan Vikyath V R (PES1201701747)

Preksha N (PES1201700827)

JAN – MAY 2021

Under the Guidance of

Dr. M. J. Venkatarangan

Professor

Department of Electrical and Electronics

PES University

Bengaluru

FACULTY OF ENGINEERING
DEPARTMENT OF ELECTRICAL AND ELECTRONICS
B.TECH IN EEE

FACULTY OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND ELECTRONICS BACHELOR OF TECHNOLOGY

CERTIFICATE

This is to certify that the Dissertation entitled

"3D Mapping and Navigation Bot using VIO SLAM"

Is a Bonafide work carried out by

Harshith N Srivatsa (PES1201701183)
Karan Vikyath V R (PES1201701747)
Preksha N (PES1201700827)

In partial fulfilment for the completion of the course work in the Program of Study B.Tech in Electrical and Electronics Engineering under rules and regulations of PES University, Bengaluru during the period August 2020 – December 2020. It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report.

(Signaturewithdate&Seal) *InternalGuide*	*(Signaturewithdate&Seal)*	*(Signaturewithdate&Seal)*
Dr. M. J.Venkataranagan	**Dr. Keshavan. B. K**	**Dr. Keshavan. B. K**
Professor	Chairperson	Dean Faculty of
Electrical and Electronics Engineering	Electrical and Electronics Engineering	Engineering and Technology
PES University	PES University	PES University

NameofExaminer:*Signaturewithdate*

1.

2.

3.

DECLARATION

We Harshith N Srivatsa, Karan Vikyath V R, and Preksha N, hereby declare that the project entitled, ***"3D Mapping and Navigation Bot using VIO SLAM"***, is an original work done by me under the guidance of Dr. M. J. Venkatarangan, Professor, Dept. of EEE, and is being submitted in fulfilment of the requirements for completion of the course work in the Program of Study B.Tech in Electrical and Electronics Engineering.

PLACE: BENGALURU
DATE :

Harshith N Srivatsa
(PES1201701183)
Electrical and Electronics Engineering

Karan Vikyath V R
(PES1201701747)
Electrical and Electronics Engineering

Preksha N
(PES1201700827)
Electronics and Communication Engineering

ACKNOWLEDGMENT

We extend our deep sense of gratitude and sincere thanks to our chairman **Dr.M.R.Doreswamy** (Founder, Chancellor –PES University), **Prof. Jawahar Doreswamy,** Pro-Chancellor of PES University and **Dr. J Surya Prasad**, the Vice Chancellor of PES University for giving us an opportunity to be a student of this reputed institution.

We extend our respect to our registrar **Dr. K.S. Sridhar** for his valuable support to conduct this Project under PES institution.

It is our Privilege to thank Chairperson of the Department and Dean of Faculty **Dr.Keshavan.B.K** Department of Electrical and Electronics Engineering for his support and guidance for doing our Project.

We express our sincere gratitude to our guide **Dr. M. J. Venkatarangan** for his valuable guidance and suggestion technically and logically for doing our Project work.

We also express our gratitude to all our faculty members, parents and fellow mates who have helped us to carry out this work. Last but not the least, we thank almighty God for his blessings showered on us during this Project period.

ABSTRACT

The developments in technology over the past decade and the increasing use of robots has helped humans accomplish tasks which were nearly impossible. One such issue that got the attention of people was mapping through unknown terrains that were either very dangerous or not accessible to humans. Areas such as mines, chemical and radioactive hazard zones and military zones which need regular surveillance and monitoring is a risky task for humans to carry out. Thus, the importance of robots for surveillance and mapping comes into foreground here. The mobile robots can replace individuals in these tasks of mapping the unknown areas and this map can be later used for various applications. This would mean growth in a lot of areas related to military, mining and even extra-terrestrial expeditions.

Simultaneous mapping and localization (SLAM), is the computational problem of constructing or updating a map of an unknown environment while simultaneously keeping track of an agent's location within it.

This Project aims to build a self-designed Robot which can be manually controlled from the user/client end. The mobile robot housing the smartphone can be maneuvered with the help of live RGB image feed to map its surroundings and navigate in the real time environment. The Robot is capable of generating a 3D map of an unknown environment, with details of different surfaces (both horizontal and vertical) of the area and navigating through it while avoiding obstacles. The 3D map that is generated can be made use for different purposes like Indoor navigation, Architectural planning, Path planning for robots, Augmented Reality applications, better and more intelligent SLAM algorithms are used even on the Mars Rover to 3D map the surface of Mars to extract detailed information of the terrain.

This robot runs on an application built using Google's ARCore and Depth API on the Unity Platform. The details of the environment are obtained from an RGB Camera of the smartphone and the corresponding information from the images is sent to Unity 3D environment. As an end result, a 3D map of the environment is generated. Using the Visual Inertial odometry SLAM in Google's ARCore SDK, the movement and location of the robot is captured by the application.

The final prototype consists of a mobile robot which carries a smartphone and moves around, 3D mapping the surrounding and rendering the 3D map onto another device on the client/user end for viewing.

Table of Contents

1. INTRODUCTION

1.1 MOTIVATION

The human sense organ system is one of the most evolved and co-ordinated in all of the animal kingdom. It helps us perceive our surroundings and understand it better. But what about robots? How do they perceive the environment? How do they identify obstacles and move around? All these questions kept us thinking and led us towards SLAM and its application in robotics. Thus, we decided to pursue our final year project in this domain.

Being final year engineering students, we were always on the outlook for new age technologies and what kind of work is being done by the big establishments like Alphabet, Amazon, Microsoft and so on. One such technology that caught our attention was Google's Project Tango. The first step of making AR technology reachable to the common man. It worked on the basis of RGBD Camera and was applicable only for smartphones which housed an RGBD camera. This project was later scrapped, a newer and better version of it Google's ARCore was released in 2018 which worked on any smartphone that consisted of just an RGB camera.

Thus, the curiosity in robotics computer vision and fascination in AR technology made us combine the aspects of both and come up with this project called 3D Mapping and navigation bot using SLAM techniques.

Figure 1

Figure 2

1.2 PROBLEM STATEMENT

In this era of ever-growing technology, it is very easy for us to navigate through known spaces. But how is one able to navigate in unknown spaces and terrains? How can one locate themselves if they have no prior knowledge of it? How is one able to travel if they do not have a map?

There are few possible solutions to these questions and we will be addressing these questions in our project and our final prototype will showcase a possible solution to these real-world problems.

Figure 3

Figure 4

The possible solutions might be:

1. A person manually observing his surroundings to generate a map.
2. Use satellite imaging to generate a map.
3. Use a mobile robot to navigate through the environment and generate a map.

The first approach to solving this problem is manual surveillance and based on the information and features observed a map can be generated. But this method is very cumbersome and there is a possibility of errors to creep in. With all the technological advances in this modern era, such an approach would be obsolete.

The second approach is using satellite imaging and GPS for navigation and localisation. This approach is being used in many states of the art applications such as the Google Maps for outdoor navigation. But in such situations where GPS or satellite imaging is used, it helps only in outdoor spaces but not indoors. What about areas which are underground such as mines, subways and inside building complexes?

Hence there is a great need for a solution in such scenarios in order to find paths and to locate ourselves in that environment. So this is where SLAM is needed the most. It helps in generating a map even indoors while simultaneously localising itself within this map as shown in figures 5 and 6.

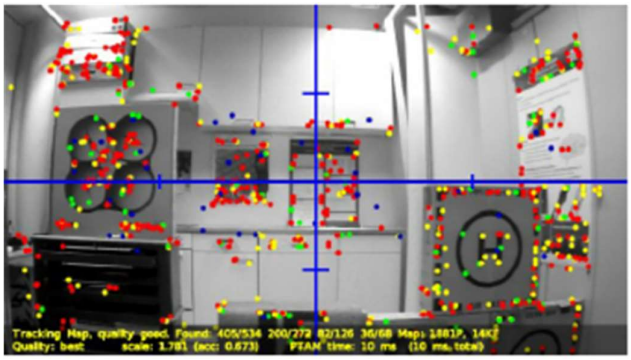

Figure 5

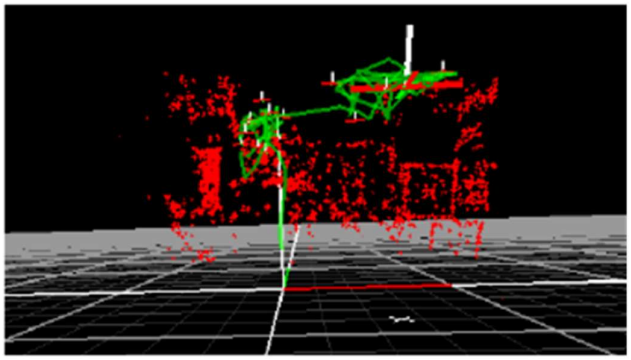

Figure 6

1.3 OBJECTIVES

This project aims to solve real world problems such as mapping and surveillance of areas which are not accessible or are hazardous to humans. The final goal is to create a medium sized, self-designed robot that can cater to various human needs, along with an application which controls the mobile robot manually, while also mapping the environment in real time. The Robot hence developed is able to create a 3D map of an area, with details on the obstacles and walls, features of an area and navigating through them based on the live RGB image feed rendering directly LIVE onto the user/client-side PC. This not only gives a 3D map of the surrounding which can be used for various purposes later on but also gives a clear RGB image feed that can be used to assess and get a better understanding of the surrounding.

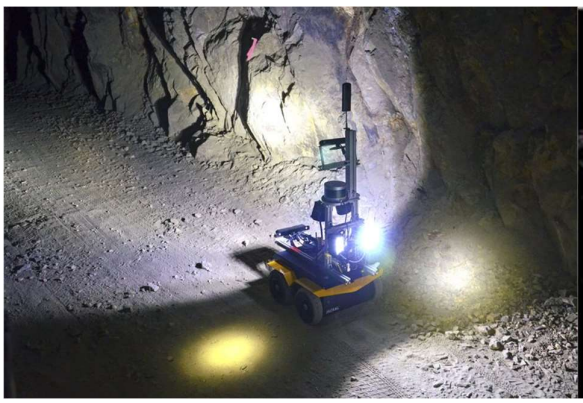

Figure 7

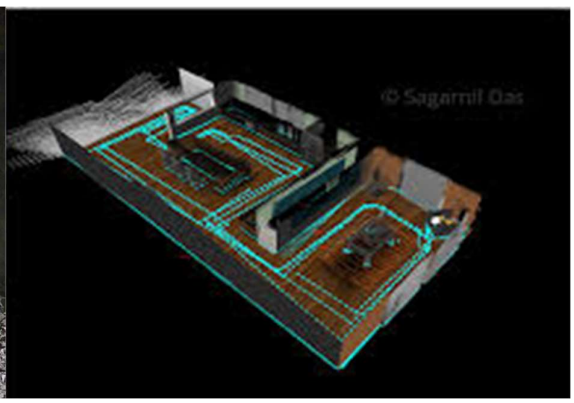

Figure 8

1.4 BACKGROUND

Simultaneous localization and mapping, or SLAM for short, is a concept of generation of a map and localising oneself in the map based on different feature points and landmarks. This technology is implemented onto a robot or unmanned vehicle to navigate through the environment making use of the map it generates. A combined process of environmental feature extraction and map generation, while navigating through the same. SLAM technique is used in robot mapping or robotic cartography. SLAM is one of the main basic requirements that allows the robot to continually localize within the same environment without accumulating drift.

1.4.1 TYPES OF SLAM

SLAM implementation is done in many ways with help of various sensors for different purposes. SLAM is largely based on mathematical and statistical algorithms such as the Kalman filter. The Kalman filter works on the principle of taking into account continuous measurements over the entire duration of movement rather than just a single one. Considering all the accumulated data the algorithm then gives a probabilistic position value based on the maximum likelihood technique.

A few of the prominent SLAM implementations are:

- EKF Slam
- Graph SLAM
- Parallel Tracking and Mapping (PTAM) SLAM
- ORB SLAM
- RGB - D SLAM
- Visual Inertial Odometry SLAM

In our project we will be incorporating the latest version of SLAM implementation, that is the Visual Inertial Odometry SLAM or in short the VIO SLAM.

The earlier versions of slam relied on the probabilistic approach were based on landmarks and sensor readings of both previous and current position the actual position of the bot was determined.

In figure 9, the stars represent the landmarks in the environment and the sensor values for distance of these from the mobile robot are taken.

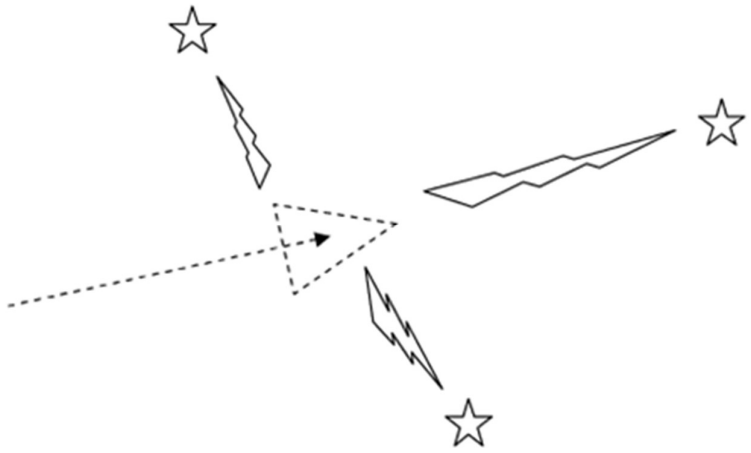

Figure 9

Once the bot moves the probabilistic value of the position is calculated based on previous and current sensor values. This position so calculated has some error attached to it as seen in the figure 10.

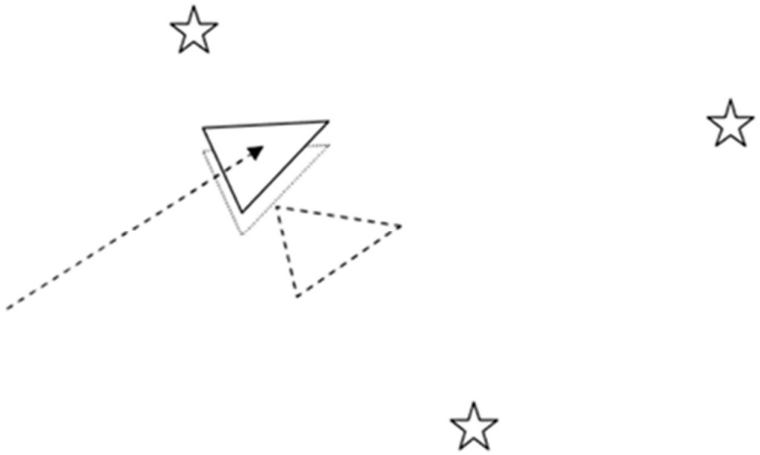

Figure 10

This picture shows the previous position (dotted triangle), probabilistic position (grey triangle) and the actual position (black triangle) of the mobile robot.

This error which creeps in with every position estimation adds up resulting in less accurate results which might lead to crashing of the robot and so on. Thus the Visual Inertial Odometry SLAM tries to reduce this error by using the results of the RGB camera along with the inertial sensor value to give a more accurate position of the robot.

This method is known as the prediction and correction technique were based on the RGB image feed the feature points in the image are extracted using which the current position is calculated. Along with this estimation the inertial sensor values along with the odometry is considered and the prediction is done. The difference of these two is used to calculate the error in position and thus come up with the corresponding correction.

This sequence happens for every position estimation thus reducing the individual errors which results in reduction in overall error in prediction.

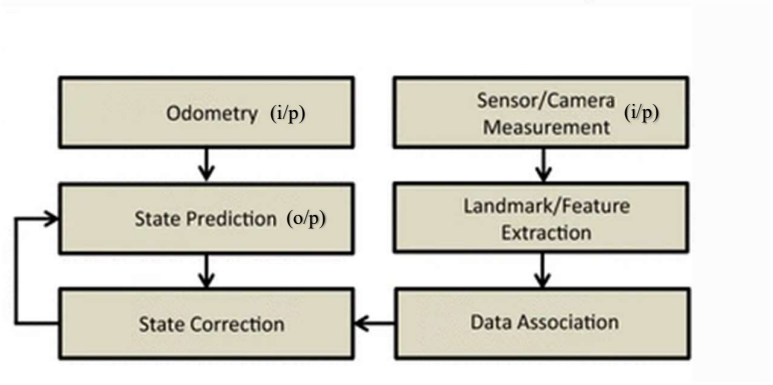

Figure 11

Figure 11 shows the flow of information in position estimation.

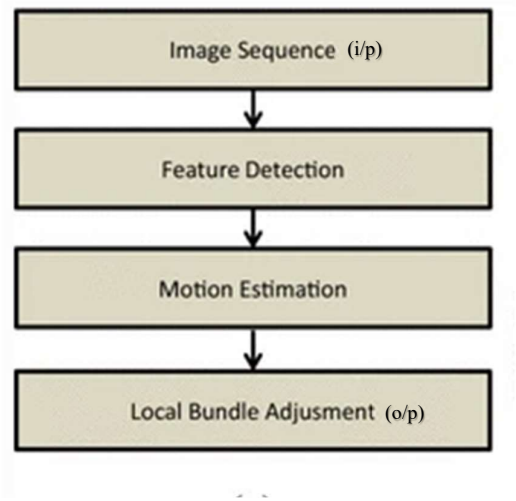

Figure 12

Thus, figure 12 summarises the entire VIO SLAM workflow.

In our project we will be incorporating Google's AR Core SDK, which works on the principle of VIO SLAM, using which an android application is built which will run on the smartphone. It will interface with the inbuilt camera of the smartphone for the RGB image feed and also use the inertial sensors available in the smartphone to reduce the errors to as minimal as possible and hence enhance the performance of the robot.

1.4.2 APPLICATIONS OF SLAM

1. Autonomous vehicles: Autonomous cars require localization to navigate in a particular environment. This is where SLAM is needed the most. SLAM is used in these using LiDAR's and RADAR. Google's self-driving car is the best example of autonomous vehicles which uses SLAM techniques. The car takes measurements from LiDAR's which can create a 3D map of its surroundings. This helps it navigate through the traffic and evade collision with other vehicles.

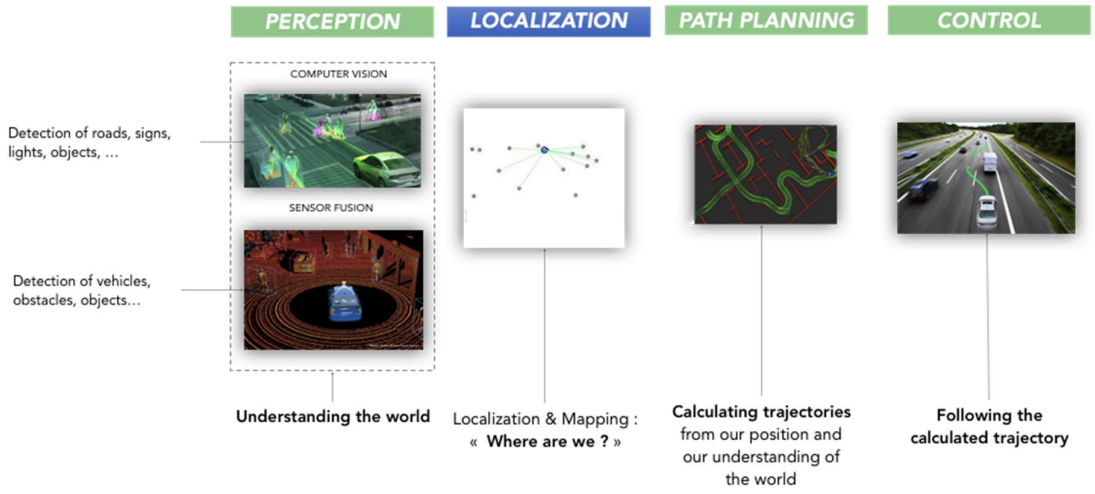

Figure 13

2. Augmented Reality: Since AR is used in most of the mobile devices, SLAM works with cameras on mobile devices using monocular cameras. Wikitude SDK6 is one of the examples of this. SDK6 combines image recognition and tracking, improved geo-location AR and it has a new Wikitude 3D tracking technology which is SLAM based becoming the best augmented reality SDK for many of the electronic gadgets. It uses SLAM to track the surroundings of the user to localise itself and also allows for placements of objects in that particular environment. As and when the user moves, a 3D map of the scene will be rendered and this allows to augment 3D objects without fiducial markers.

Figure 14

3. At home: One of the best examples of this is Roomba vacuums. It uses vSLAM or the visual simultaneous localization and mapping. It enables the roving vacuum to create a map of its surroundings and chart out "where it is, where it's been and where it needs to clean". Inclusion of such new age technology into home appliances have revolutionized the products enabling them to do much more with different kinds of features enabled within them.

Figure 15

4. Autonomous robot: Human beings are very proficient in interacting with their environments and adapting to any situations. Humans are very good at achieving locomotion and manipulation by applying forces at contact points between the body and temperature. This impressive ability has been a source of inspiration in the development of humanoid robots. Humans, being unable to access certain areas like in disaster situations, the work had to be attempted by robots instead, in such complex environments designed for humans. SLAM is used in Humanoid robots to recognise the environment they are in and locate themselves in it.

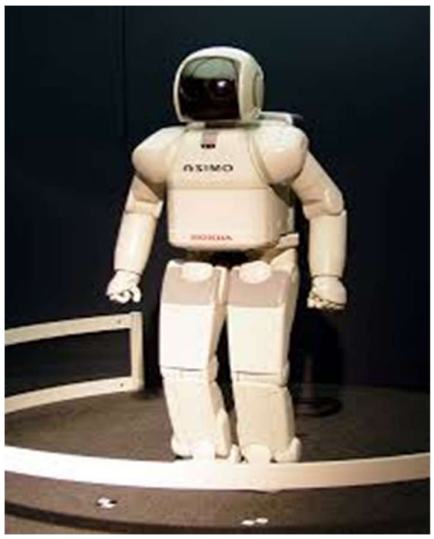

Figure 16

5. Space: SLAM is used for terrain mapping for localisation of space rovers. The Mars Exploration Rover used a Visual Odometry system which helped in improving the navigation accuracy. The sandy and rocky terrains make it slippery for the rovers to drive over them and they get slipped by unpredictable amounts. Visual odometry system helped them in giving a better idea of how far it has actually travelled. SLAM is designed in these vehicles to maintain an accurate knowledge of its position when absolute positioning information is not available.

Figure 17

1.5 PROBLEMS FACED

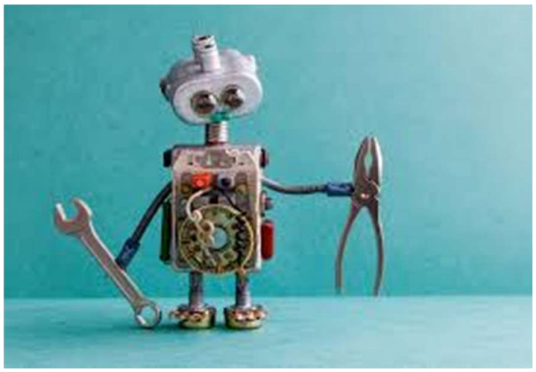

Figure 18

Some of the challenges faced in the modern age in designing a mobile robot using SLAM are as follows:

1. High computational power required by the on-board circuitry for making it totally autonomous and also to cover larger areas.
2. Dealing with dynamic obstacles while manoeuvring. When a camera frame has dynamic obstacles like a moving person, animals or inanimate objects, the 3D mapping of this frame becomes quite difficult.
3. There exists a noticeable latency between the on-board circuitry which does the 3D mapping and the circuit which is responsible for movement of the robot.
4. The inaccuracy in the location of the robot. The position which the robot comprehends through odometry and SLAM is not the exact location of the robot. This inaccuracy in the exact location is quite difficult to eliminate completely with the current technology available in the market.
5. The 3D mapping of similar looking objects in an area can cause inaccuracies. For example, two similar looking tables at different distances from the camera may end up overlapping or cause redundancies in the 3D map.

Thus in this project we look to address these issues and minimise the errors while we build the final prototype of the mobile robot along with the application.

2. SURVEYS: Literature Survey, System Survey and Technology Survey

2.1 LITERATURE SURVEY

Paper [1]: Tobias Feigl, Andreas Porada1, Steve Steiner1, Christoffer L¨offler, Christopher Mutschler and Michael Philippsen: "**Localization Limitations of ARCore, ARKit, and Hololens in Dynamic Large-scale Industry Environments**"

- This paper was used to study the applicability of today's popular AR systems(Apple ARkit, Google ARcore, and Microsoft Hololens) in such industrial contexts.
- Takeaway: The takeaway from this project would be to keep in mind about the challenges faced when there's a dynamic environment for mapping.

Paper [2]: Songmin Jia, Ke Wang, and Xiuzhi Li:" **Mobile Robot Simultaneous Localization and Mapping Based on a Monocular Camera"**

- This paper was used to understand the novel monocular vision-based SLAM algorithms for mobile robots.
- Takeaway: Instead of using a monocular camera, RGB cameras can provide a rich 2D visual and 3D depth information that are well suited to the motion estimation of indoor mobile robots.

Paper [3]:Xiaochen Zhang, Xiaoyu Yao,Yi Zhu,Fei Hu :" **An ARCore Based User Centric Assistive Navigation System for Visually Impaired People"**

- This paper briefs us about an assistive navigation system for visually impaired people that takes advantage of ARcore to acquire robust computer vision.
- Takeaway: We can implement this navigation system in an outdoor environment and also the human-machine interaction delay can be improved.

The instruction delivery is not instantaneous, and the latency becomes a bottleneck when dealing with urgent interaction requests. It is vital when dealing with urgency in navigation. The user may not be able to access multiple instructions simultaneously, and environmental sounds may cause interference.

Paper [4]: Jonas Halvarsson
"**Using SLAM-based technology to improve directional navigation in an Augmented Reality game"**

- This paper was used to understand the background and the process of SLAM, its recent developments, features and issues faced.
- Takeaway: We can implement the augmented reality and localization features of the project for our robot for augmented reality navigation and the robot to be self-aware of its local positioning.
-

Paper [5]: Alif Ridzuan Khairuddin, Mohamad Shukor Talib, Habibollah Haron Faculty of Computing Universiti Teknologi Malaysia Johor Bahru, Malaysia: **"Review on Simultaneous Localization and Mapping (SLAM)"**

- This paper gives an overview of what SLAM is and various parameters to me considered for implementing the same. It sheds light on the background and process of SLAM, its recent developments, features and issues faced.
- Takeaway: Parameters to take into account while implementing SLAM like the estimated and true positioning, relative positioning, overlap, environment surfaces and colours, etc

Paper [6]: Austin Corotan College of Science and Engineering Western Washington University Bellingham, Washington; Jianna Jian Zhang Irgen-Gioro College of Science and Engineering Western Washington University Bellingham, Washington : **"An Indoor Navigation Robot Using Augmented Reality"**

- This paper addresses the issue of autonomous robotic navigation in an indoor environment. An Augmented Reality (AR) based navigation system is created using the JA.

- Takeaway: Introduced to use of AR Core for mobile robot navigation. Talks about using cloud-based data storage and retrieval of previously available data to locate the bot. Tests used to compare and evaluate the model. Data extraction and routing using JAQL.

2.2 SYSTEM SURVEY:

The possible softwares that can be used for a mobile robot is:

1. Robotic Operating System with Gazebo
2. V-REP
3. Unity

Robot Operating System (ROS) is robotics middleware (i.e. collection of software frameworks for robot software development). It is a set of software libraries and tools that help you build robot applications. ROS is one of the oldest and most popular platforms used for slam robots. We wanted to work with a newer and less implemented platform, hence, the reason for choosing Unity.

Gazebo: Gazebo is a simulation software that is used to simulate and test bots and train them to predict their outputs.

V-REP: V-REP stands for virtual robot experimentation platform which provides 3D mapping and simulations software for mechanical and robotic projects.

Unity is a cross-platform game engine developed by Unity Technologies,
The engine can be used to create three-dimensional, two-dimensional, virtual reality, and augmented reality applications, as well as simulations and other experiences. We are integrating Unity with ARCore to implement SLAM onto the robot. It provides a platform for 3D simulation. Figure 19 shows the game scene of Unity platform.

Given that there are multiple platforms for simulation we decided to go ahead with the Unity platform along with Google's ARCore SDK as it was relatively a newer technology and provided AR based features and mobile application building platform both in one software itself. This enables using available equipment such as your smartphone and turns it into a central processing unit for a bot. Thus replacing higher end processors like the Jetson Nano with the already available smartphone.

Figure 19.Game Scene inUnity

2.3 TECHNOLOGY SURVEY:

The possible technologies that can be used for a mobile robot is:

1. Using LIDAR
2. Using IMU
3. Using camera and computer vision

Lidar, which stands for *Light Detection and Ranging,* is a remote sensing method that uses light in the form of a pulsed laser to measure different ranges (variable distances) to the Earth. A lidar instrument principally consists of a laser, a scanner, and a specialized GPS receiver. Lidar systems allow scientists and mapping professionals to examine both natural and manmade environments with accuracy, precision, and flexibility.

An **inertial measurement unit** (**IMU**) is an electronic device that measures and reports a body's specific force, angular rate, and sometimes the orientation of the body, using a combination of accelerometers, gyroscopes, and sometimes magnetometers. IMUs are often incorporated into Inertial Navigation Systems which utilize the raw IMU measurements to calculate attitude, angular rates, linear velocity and position relative to a global reference frame.

A major disadvantage of IMU (Inertial Measurement Units) is that they suffer from accumulated error. Lidars (Light detection and ranging) performance degrades in bad weather conditions and it is difficult to track. It is also an expensive technology. Due to these reasons, we are using the camera and computer vision approach which has more accuracy and is cheaper.

Thus, with these drawbacks in mind we decided to use a smartphone as a replacement for various kinds of sensors and camera along with the central processing unit. Our Main goal was to replace all these components with a single device which is widely available and can work as a plug and play device.

Major reasons for choosing smartphone over other traditional methods:

1. Harness the potential of smartphones: To make maximum usage of already existing technology which is easily accessible to the common man. Most of the smartphones come with state-of-the-art processors and we look to harness its power by using it as the central processing unit as well as a sensor unit on the mobile robot.
2. Compact and lightweight: Most smart phones come with inbuilt cameras and sensors all packed into a compact device. This aids in making the bot smaller for various use cases.
3. Cost: Most people working on robots built it using various microcontrollers. They will have to spend more money on various sensors such as IMU and RGB camera

as well. Thus, the smartphone acts as a cost-effective alternative which houses various sensors along with a powerful CPU within.

4. Accessibility: Smartphone is a device which almost everyone possesses. It is easily available and can be used as plug and play device for scenarios where there needs to be immediate action taken.
5. Software update and maintenance: The application built using the Unity platform will be used as the server-side application which runs on the smartphone. The updation and maintenance of this software and rolling out further feature will be relatively easier for smartphone apps then compared to the traditional microcontroller approach

3. DEVELOPMENT MODEL AND METHODOLOGY USED:

3.1 Study and selection phase:

To begin with, we reviewed some of the literature papers in order to familiarise ourselves with all the technologies available in the market. We also learnt the background and process of SLAM, its developments and future applications.
Robotic Operating System (ROS) has been used for implementing SLAM robotics for a while, hence we did research on the latest technology which could be used.

We came across Google's ARCore which is an Augmented reality platform put forth by Google in 2018. The Google AR Core platform can be integrated with platforms like Unity, Unreal Engine, AR Foundation to create AR applications for both iOS and Android. Hence, we decided to try implementing SLAM using this new technology in our project

3.2 Architecture and Design phase

There are a number of design choices which are available for the SLAM robot. The SLAM robot design in our project was inspired by the TurtleBot structure. This structure is designed to be easy to build and assemble and it uses components and standard materials that are readily available. Another reason for the selection of a circular structure was that it's hard for those robots to find a corner that it's unable to get out of.

Design Specifications:
The robot chassis can be designed in many ways depending on the user's needs. Two main options we came across of the chassis are

1. 3D printing
2. Laser cutting of acrylic sheets

3D printing of a chassis is quite expensive and requires a 3D model. As this is a time intensive process to learn 3D modelling from scratch, we decided on acrylic sheet cutting.

Acrylic sheet cutting is a much cheaper and reliable option, and it also gave us exposure to designing.

- Designing an android application using Unity platform integrated with Google ARCore SDK. This application uses the inbuilt RGB camera of an android smartphone, which extracts the images and detects different horizontal and vertical surfaces with the help of point-cloud generation.

3.3 Implementation phase:

The implementation of SLAM and surface detection for 3D Mapping can be done in a number of ways such as simulation through Gazebo and hardware implementation using ROS, but we chose to work with Unity as it is a newer technology and it has the capability of building applications for Android and iOS smartphones which help us use the power of already available smartphone processors for multiple use cases.
The 3D mapping is done with the help of Google's ARCore which was released in 2018. Here is the detailed working and implementation of Google's ARCore and Depth API:

3.3.1 Google's ARCore

In the past couple of years technological advancement in newer fields such as Augmented Reality has been very astounding. A more interactive and visually immersive experience of the Augmented Reality technology has drawn attention towards itself. Augmented reality (AR) gives a combination of the real-world environment along with computer generated information such as animated images, graphics, audio effects and other perceptual information.

Augmented Reality brings digital components to life in our perception of the world, thus enhancing our natural environment and providing more powerful and intuitive insight on the world around us. It has made environmental information digitally manipulable which can be used for a variety of tasks such as object detection or recognition, motion tracking, and environmental understanding.

The use of AR has been growing rapidly in recent years. Several AR development kits, including AR Core, Wikitude, Vuforia, and ARkit have made AR much more accessible to developers which opens up the horizon for application of AR technology to be applied across many fields. From the entertainment and gaming industry to surveillance motion tracking and image recognition and even autonomous navigation, AR applications keep increasing by the day.

Google's AR Core aims to use real world surrounds with software occlusions to enhance the overall experience and interaction with the environment. It uses the inbuilt RGB camera of the smartphone to extract live feed and can perform multiple tasks such as

motion detection, environment understanding, light estimation, etc. This also includes surface detection, 3D mapping and self-localisation of the mobile in the environment it is in.

AR Core is supported by any Android smartphone running Android 7 (Nougat) and later. The fundamental 2 features of the SDK is to identify its own location and understanding the environment. To do this AR Core implements Visual Inertial Odometry SLAM or VIO SLAM. Using the inbuilt camera feed for RGB images and inbuilt odometry sensor values from the smartphone for identification and estimation of its position in the environment.

Google's ARCore consists of a Depth API which is used for computer vision. It processes the RGB camera feed and gives an understanding of Depth and other features of the environment.

3.3.2 Depth API

Figure 20 shows the entire flowchart of AR Core. The AR Core consists of Depth API which is used for depth perception in the RGB camera feed. The Depth API takes in the RGB values from the image and converts into a lower-level Depth map. This lower-level depth map is easier to understand and interpret using which other modules and functionalities can be built.

For each image having the R, G and B values in the form of matrices the Depth API converts it into a Depth Map consisting of Depth Array, Depth Mesh and Depth Texture data structures for different use cases. Thus based on the R, G and B values of the pixels in the image the corresponding depth parameters are calculated.

Further to this a set of conversion utilities are provided to improve the work flow of development of softwares and switch between screen space and world space representation.

The values so generated are further realised into Localised Depth, Surface Depth and Dense Depth.

Localised Depth uses the Depth Array to work on a small number of points directly on the CPU. Tasks such as computing physical measurements, checking for collision of rendered AR objects with real world objects, identification of geometry of the physical world are done here.

Surface Depth leverages the CPU/GPU to create an update depth mesh in real time. Thus, this allows us to identify physics of the physical environment, the texture, surfaces at different depths in the image.

Dense Depth requires GPU processing to render screen space effect with fragment shaders, computation photography tasks like re-lightening the surroundings, aperture effects like depth and field and occlusion of virtual objects into the physical world

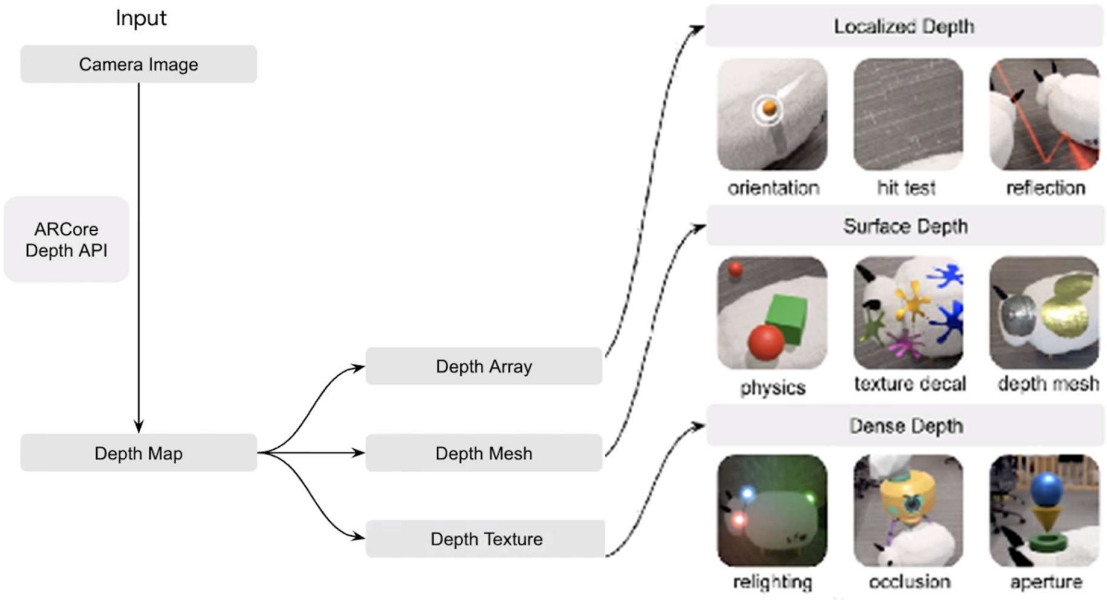

Figure 20. Flowchart of ARCore

The final output of the Depth API generates a point cloud from the available information from the image. The point cloud indicates different feature points which are considered for generation of the image landmarks. Based on these feature points various surfaces and objects are detected and these distinct feature points are used to estimate change in position of the camera.

Based on the previous frame and the position of the feature points in that image and the feature points in the new frame the change in position and angle is estimated. AR Core uses this visual information along with inertial measurements from the device's IMU to estimate the position and orientation of the camera relative to the world over time as shown in the figure 20.

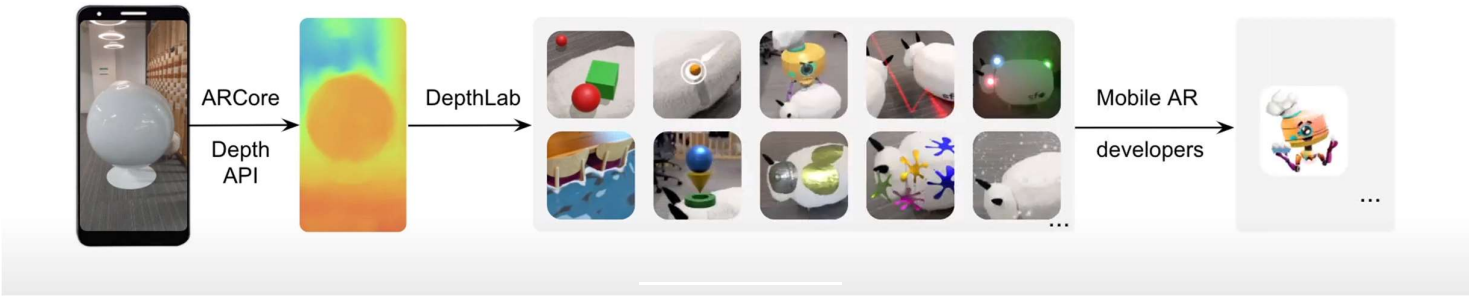

Figure 21

4. SYSTEM ARCHITECTURE:

4.1 Mechanical Architecture:

There are many types of designs available for Slam robots. The slam robot in our design has been inspired by the turtle-bot structure. It will be a two layered structure with a clamp on the 2nd layer in order to hold the mobile phone. The chassis was chosen to be circular in shape as shown in the figure 22.

Figure 22. Chassis Model

4.1.1 CHASSIS DESIGN

Slam robots can be designed in different ways according to the user specifications. In our project, we initially used TinkerCad which is a software used to design 3D models of the chassis.
Material used: Acrylic
Dimensions: Circular chassis with 14cm in diameter.

It is a 2 layered chassis in which the bottom plate contains the motor drivers and the Lithium polymer battery to power the microcontrollers and the motors. This plate is supported by wheels and castor wheels for the free movement of the bot.
The upper layer has been separated from the bottom layer with the help of separating rods. This layer houses the mobile phone which has been supported by the clamps.

The designing and procurement of the chassis was made in phase 1 of the project. Further continuing from that in the 2nd phase we came up with 2 possible mounts for the mobile on the bot and based on logical reasoning we chose one particular mount.

The two different designs for the bot namely the vertical mount and horizontal mount.

1. Vertical mount: In this, the mobile phone will be placed vertical in position for capturing the live feed. The mobile phone will be supported by a U-shaped clamp structure in the base of the chassis and the upper part of the mobile will be held in place by the upper layer of the chassis.

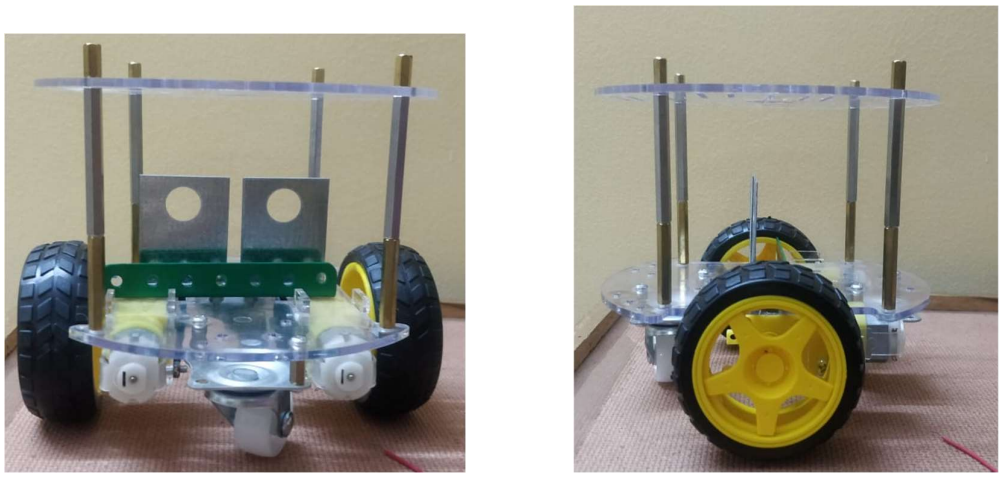

Figure 23. Vertical Mount of Smartphone

2. Horizontal mount: In this, the mobile phone is placed horizontally. The mobile will be supported by a U-shaped clamp structure which is placed on the upper layer of the chassis. The lower layer will house the battery, microcontroller, etc.

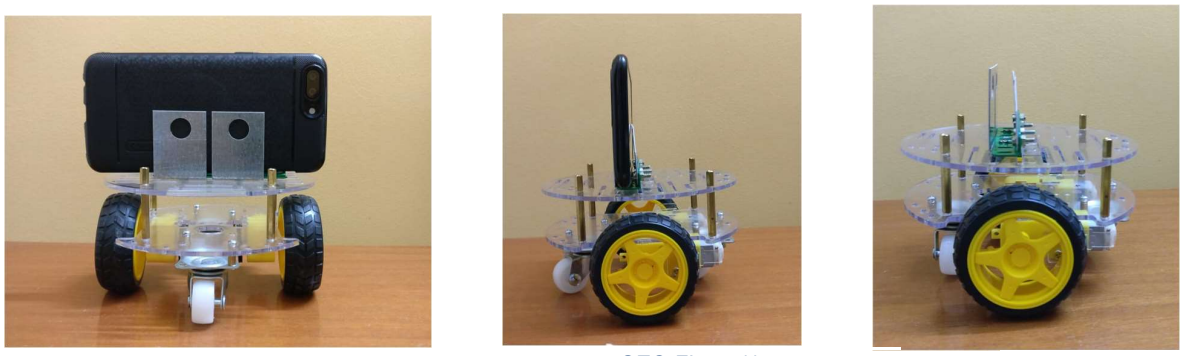

Figure 24. Horizontal Mount of the Smartphone

Based on various factors such as the centre of gravity of the robot, the accessibility and usability of the smartphone and field of view of the cameras we finally decided to go ahead with the horizontal mount as it suited our needs.

4.2 Hardware Architecture

4.2.1 Power Details:

NodeMCU = 3.3V x 70mA =2.31mW
HBridge = 12V x 600mA =7200mW
Motors = 1000rpm–12V centre shaft – Geared DC motor
Torque = 0.5kg/cm
=12V x 300mA x 2hrs = 7200mW
Total P = 14.631W

Assuming 2000mAh battery:
Battery: 12V x 2000 x 10^(-3) X (efficiency of components)
= 12 x 2 x 0.8
=20
Load = 14.631 x X
Load/Battery cap = Number of batteries
(14.631 x X)/20 = 1
X = 1.5Hrs

Thus based on the power calculations we derived that to run the robot for approximately 90 mins we need a 2000 mAh battery.

4.2.2 Torque calculations:

This torque value represents the total torque required to accelerate the robot up an incline.
T=100/e* ((a+gsinθ)*M*R)/N = 0.0559Nm or 0.570kgf-cm
Where
efficiency = 70%
a=0.075m/s^2, g=9.8m/s^2
angle of inclination=15degrees
mass of the robot=1kg
Radius of wheel=0.03m
N=total number of drive wheels excluding castor wheels=2
Conversion factor of Nm to kgf-cm:
1Nm=10.2kgf-cm

Based on the weight of the components put together and the dimensions of the bot the final torque calculations were made which resulted in 10.2 kgf-cm.

4.2.3 Component specifications:

1. NodeMCU-ESp8266: It is a microcontroller board based on ESP 8266. It has 16 general purpose input-output pins on its board. It takes an input voltage of 7-12V and operates at around 3.3V. It supports UART, SPI and I2C interface.

Figure 25. NodeMCU ESP8266

Specifications from the datasheet

Microcontroller	ESP8266 32bit
NodeMCU size	49mmX26mm
Pin spacing	0.9``(22.86mm)
Clock speed	80MHz
USB connector	micro USB
Operating voltage	3.3V
Input voltage	4.5V-10V
Digital I/O pins	11
Analog in pins	1
ADC range	0.3.3V

Table 1

2. ARCore supported smartphone: The mobile has an app through which the map generated can be seen and the bot can be moved accordingly.

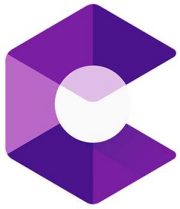

Figure 26.Google ARCore Logo

3. Motor driver L298: It is a high-power motor driver module for driving DC and stepper motors. It operates on an input voltage of 12V and gives an output of 5V. This helps in the movement of the bot.

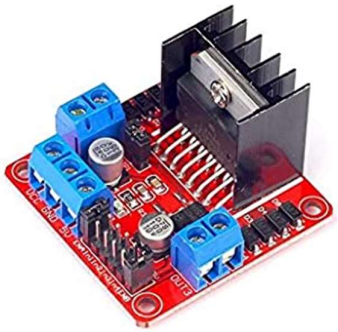

Figure 27. Motor Driver L298

Specifications from the datasheet:

Operating voltage	12-16V
Peak current	14A
Continuous current	30A
No. of channels	2
Battery protection mode	Yes
Maximum PWM frequency	20kHz
Dimensions	60mm x 54mm x 12mm
Weight	19gm
LED indicator	Yes
Overcurrent protection	Yes

Table 2

4. DC motor+wheels: The DC motors aid in the motion of the robot as well as serve as actuators in the mechanical design of the robot. 2 dc motors have been used here for a total of 2 wheels and 2 castor wheels.

Specifications for the motor from the datasheet:

Input voltage	12V
No load current	800mA
Speed	100rpm
Counts per revolution	29250
motor diameter	28.5mm
length	93mm
weight	170gm
holding torque	27kgcm

Table 3

5. Lithium Polymer battery: We are using a 12V lithium polymer battery to power the motor drivers as well as the NodeMCU as a standalone device. It takes a maximum charging current of 1.5A and the total capacity of the battery is 2000mAh.

Figure 28. LiPo Battery

6. Unity v2019: Unity is a cross platform engine that is used to create 2D,3D, virtual reality and augmented reality games, as well as simulations and other experiences. It also has a set of software libraries and tools that help build robot applications in a modular manner.

Figure 29.Unity Platform Logo

4.3 Software Architecture:

4.3.1 Block diagram and working principle:

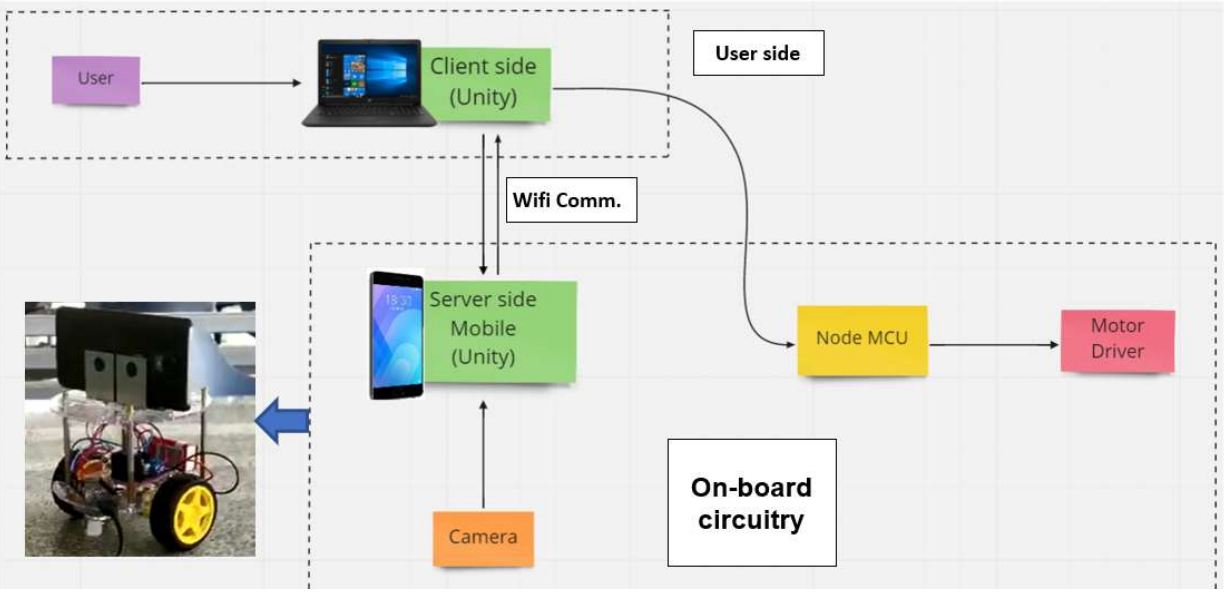

Figure 30. System and Communication Block Diagram

Figure 30 shows the entire working and communication between various parts and components of the projects. The project aims to generate a 3D map of the environment using a robot with a smartphone mounted on it which works as CPU for processing and uses inbuilt cameras for video feed. Most of the software is built using the Unity platform along with ARCore SDK and Depth API.

On the whole the project can be divided into 2 main categories:

1. On-board circuitry and softwares
2. User side software

The on-board circuitry consists of a nodeMCU, Motor drivers, Batteries, and a smartphone with an integrated camera. All of these are mounted on the chassis of the bot. The Unity application built will be running on the mobile which interfaces with the inbuilt camera of the smartphone to obtain live RGB camera image feed for surface detection. This camera feed is then processed by the application which identifies both horizontal and vertical surfaces dynamically.

On identification of a surface the app calculated the depth at which the surface is using the Depth API and correspondingly send packets of information to the NodeMCU to control the motors respectively

The Augmented Reality Platform Unity can be divided into 2 parts: server side and the client side.

On the server side, the server basically fetches the data feed from the RGB camera of the mobile and initiates a TCP connection with the NodeMCU and sends the data through it. All the components are connected to the same WIFI network.

NodeMCU is connected to the motor drivers and makes the bot move in the direction according to the map generated. NodeMCU in turn communicates with the server.
The client fetches the data from the server and transfers it to 2 or more devices connected and this can be accessed by the user simultaneously (eg: laptop and mobile).

As part of deliverables of Phase 1 of the project, the server side of the application that is interfacing the NodeMCU with the motors for manual control of the bot was done. Along with that on the software side of things, the surface detection application was built on Unity and tested.

Thus, in phase 2 we looked at building on what was achieved in phase 1. On the software front we had successfully achieved dynamic surface detection, i.e. both horizontal as well as vertical surface detection. So, in phase 2 we looked into 3D mapping technology and how information can be extracted from the surfaces detected to generate a 3D map and render it onto the client side of the application.

On the client side an application was built which could manually control the robot to maneuver it around along with live RGB image feed from the server-side application to the client side of the application.

4.3.2 Representation of an object in 3D space.

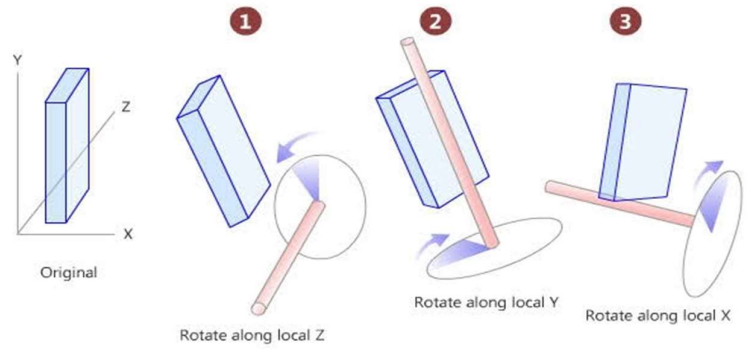

Figure 31

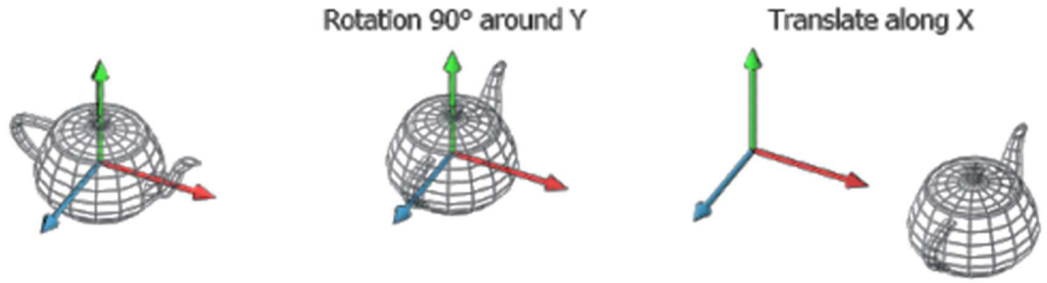

Figure 32

To represent an object in the real world onto a virtual 3D environment, 3 sets of coordinates are required. They are the Position, Rotation and Scaling.
The position determines the exact location of the object with respect to x,y,z 3D axes. As it is shown in the figure 31 the object is placed in the coordinate system and its exact position is determined.

Rotation coordinates determine the orientation or inclination of the object in the 3D base. As it is seen in the figure 31, an object can be rotated along any axes and the orientation can be determined. For example, an object is rotated 90 degrees around the Y axis and then translated along the X axis. Even if it has been translated initially and then rotated, the orientation and position of the object remains the same.
So, in this way, an object's location and inclination can be found.

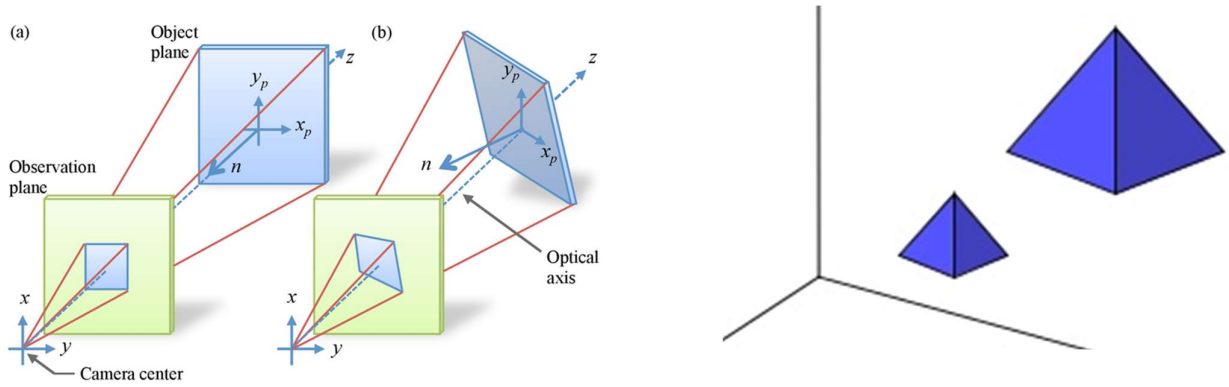

Figure 33

Finally, the Scaling coordinates are used to recreate the object with the actual dimensions. As it is shown in the figure 33, based on the distance or depth of the object from the camera, a scaling factor is generated which can be used to represent the 3D object in the real-life environment.

4.3.3 3D Mapping block diagram and working principle:

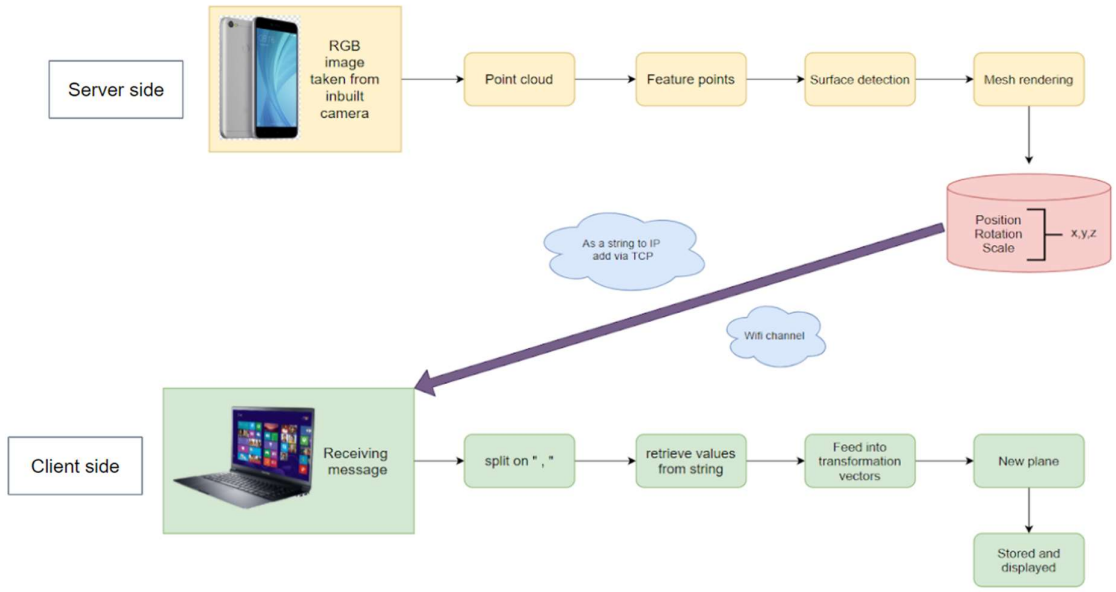

Figure 34. 3D Mapping and Rendering block diagram

Figure 34 shows the entire working of how a 3D map can be generated.

1. The working can be divided into two parts: the Client side and the Server side.

2. The Smartphone with an inbuilt camera is placed on the mobile robot and acts as the Server side of Unity and our personal computer, the Client side of Unity.
3. The Smartphone uses its inbuilt camera to capture the RGB images from the surroundings.
4. From the image thus captured we generate the point cloud. These point clouds are used to perceive different depths in the image frame captured.
4. These point clouds that are generated are further used to generate feature points of similar depths and discard unwanted point clouds.
5. A mesh is rendered over all the feature points of similar depths to detect different horizontal and vertical surfaces dynamically.
6. To represent an object in 3D space, we need 3 sets of coordinates which are position, rotation and scaling. These 3 sets of coordinates are extracted as and when a surface is detected.
7. Each surface generated has a unique name i.e, the nth number of surface detected.

```
Sending

message  = "name,position x,positiony,position z,rotation x,rotation y,rotation z,scale x,scale y,scale z"
message1 = "1,10,0,1,10,20,31,11,1,1,1"
message2 = "2,0.5,0,10,1,2.5,3.1,3.3,1,1,1"
message3 = "3,1.5,0,1,3,0.25,0.6,1.23,1,1,1"

One message for each Surface
```

Figure 35. Example of String Message Sent to Client Side

Figure 35 shows the representation of a message which is sent to the client side.

8. The message is a concatenated string with the name, position rotation and scaling values converted into a string which is separated by a comma(,).
9.This message is generated for each surface that is detected on the smartphone application.
10.The message is sent to the client side i.e., the unity application running on the pc through the IPaddress of the pc, where the smartphone and the laptop is connected to the same WIFI network.
11.The client-side Unity receives the message and splits it using “,” which acts as a delimiter as shown, which is stored in a string array called ‘packet’.

```
Receiving

Creating a dictionary 'planes'
'Planes' stores the name and gameobject of each plane
The GameObject stores the position,rotation and scale of each surface

String[] packet = message is split using ',' as demilitter

String[] packet = ["1","10","0","10","20","31","11","1","1","1"]

plane name = packet[0]
plane position = [packet[1],packet[2],packet[3]];
plane rotation = [packet[4],packet[5],packet[6]];
plane scale = [packet[7],packet[8],packet[9]];
```

Figure 36. Example of Message Received

12. A dictionary has been used in order to store the name and the game object of each plane as a key-value pair. The game object is the main part of the process which has the ability to store the position, rotation and scaling values.

13. Initially when a surface is detected, we create a name and game-object pair for the surface. This is added into the dictionary called ‘planes’. Hence, the game-object with all its values is responsible for rendering the surface in the 3d environment of Unity.

14. Every time a message is received for a surface, we search the dictionary based on the key i.e., the number of the surface as shown in the figure 36. If it already exists, we just update the values of the already existing game object. If the encountered surface is a new surface, we create a new name, object pair which is appended to the dictionary ‘planes’ .

15. When all the values of surfaces are stored, in this particular process a surface will be rendered into the unity game environment i.e., the 3D environment. Similarly, when all the surfaces detected are stored in the 3D space, a 3D map of the environment is generated with all the surfaces. The 3D map obtained shows only the floor plan which shows all the horizontal and vertical surfaces ignoring any noise or unwanted objects.

Thus, this is the entire workflow of the final prototype starting from the server-side application, which runs on the smartphone, detecting a surface to the client side application where the information is received, processed and a 3D map is reconstructed which is rendered onto the screen on the client side application

Manual Control of the Bot from Unity

1. On the Client Side of Unity, we have a game object which is responsible for the manual control of the robot while accessing the live RGB feed.
2. The control is done using the arrow keys of the laptop.

3. The object has two scripts, one for sensing the arrow key pressed and the other is for sending the string message to NodeMCU.
4. Whenever an arrow key is pressed, a string message is sent to NodeMCU
   a. When up arrow is pressed, “f” as a string is sent.
   b. When down arrow is pressed, “b” as a string is sent.
   c. When the right arrow is pressed, “r” as a string is sent.
   d. When the left arrow is pressed, “l” as a string is sent.
   e. When nothing is pressed, “s” for stop is sent.
5. The Script on NodeMcu receives the data and checks the string in a case statement.
6. Based on the string received, forward, backward,right ,left and stop functions are executed respectively
7. In this way the robot is controlled manually using the arrow keys of the PC.

During this entire process the live RGB camera feed is rendered on client end application via Wifi Channel which is established between the smartphone and the user PC. Based on this Live feed the user can maneuver the robot through various obstacles in the unknown environment.

5. SYSTEM TESTS AND RESULTS

As deliverables of Phase 2 of the Capstone project we have built a mobile robot using acrylic sheets and inspired by TurtleBot architecture. The Bot houses all the components along with a smartphone on it. The onboard circuitry is powered by a 2000 mAh battery. The application for the smartphone was built using Unity platform along with AR Core SDK and Depth API for surface detection and 3D map generation. A client-side application used to manually control the robot, get live RGB image feed and rendering of the 3D map was also built in the process. The further sections illustrate the test cases and results obtained in then.

5.1 TEST SCENARIO

The application for surface detection and Live RGB image rendering was built using Unity Platform. The applications are tested on real world test cases such as floors, walls, doors, etc. The application was expected to generate a depth map from the RGB image from the camera, using which a point cloud is generated. The various points from the point cloud would then be integrated to identify feature points in the image frame and recognise landmarks and surfaces in the image.

On identification of a surface, a mesh should be rendered over the surface to show the identified surface. The information thus acquired is then sent to the user end for rendering Similarly the live RGB image feed also is rendered onto the user end via a WIFI channel.

With respect to the hardware, a fully functional mobile robot was built and put to test. The manual control of the robot was tested with haptic feedback and reduced latency. The mobile robot was controlled remotely by the user and was made to move through an obstacle course.

The test results of the prototype have been elaborated in the further section.

5.2 ACTUAL RESULTS

Based on the results obtained from the prototype, it is observed that all the features are in accordance with the expected results. The application is successfully identifying the surfaces, both horizontal and vertical, including depth and elevation differentiation. This information is then being processed and rendered to the client side of the application where the 3D map is generated.

Server-side application

The server-side application will be running on the smartphone which is mounted on the bot using a U clamp structure. The application is built using Unity Platform and is Android and iOS compatible. For the User Interface of the app a simplistic UI was designed which

gave the user the option to choose between only RGB image feed and 3D mapping. Figure 37 shows the server-side application running on the smartphone.

Figure 37. Screenshot of Launch Screen of the application

Both of these options had the manual control of the robot built into them

1. Live Feed - This option starts interfacing with the inbuilt camera of the smartphone and the RGB images frames thus recorded is sent to the user side application running on the PC via WIFI communication. Figure 38 is a screenshot of the user side PC with the live feed from the bot.

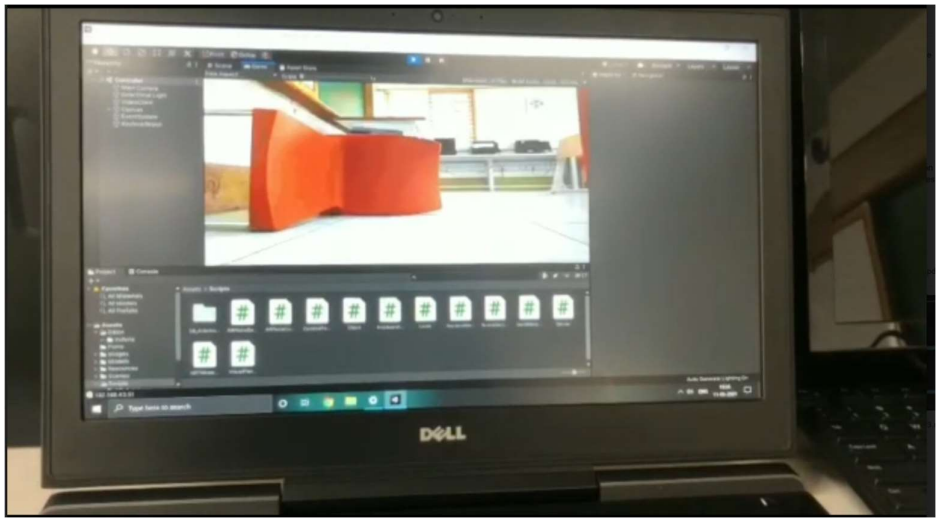

Figure 38. Live Feed Rendered on Game Scene in Unity

2. 3D Mapping - Dynamic Horizontal and vertical surface detection:

The features of both horizontal and vertical surface detection from phase 1 were combined to build a single app to detect both horizontal and vertical surfaces together and dynamically. Figure 39 shows the detection of the door (vertical) and the floor (horizontal) surface being detected as 2 separate surfaces.

The point cloud which is generated from the RGB image feed is used to extract the feature points and identify different kinds of surfaces and lays a mesh on it.

The application is able to dynamically detect both surfaces. In other words, we can walk around, with the application running on the smartphone and the application will be able to detect the surfaces on the go.

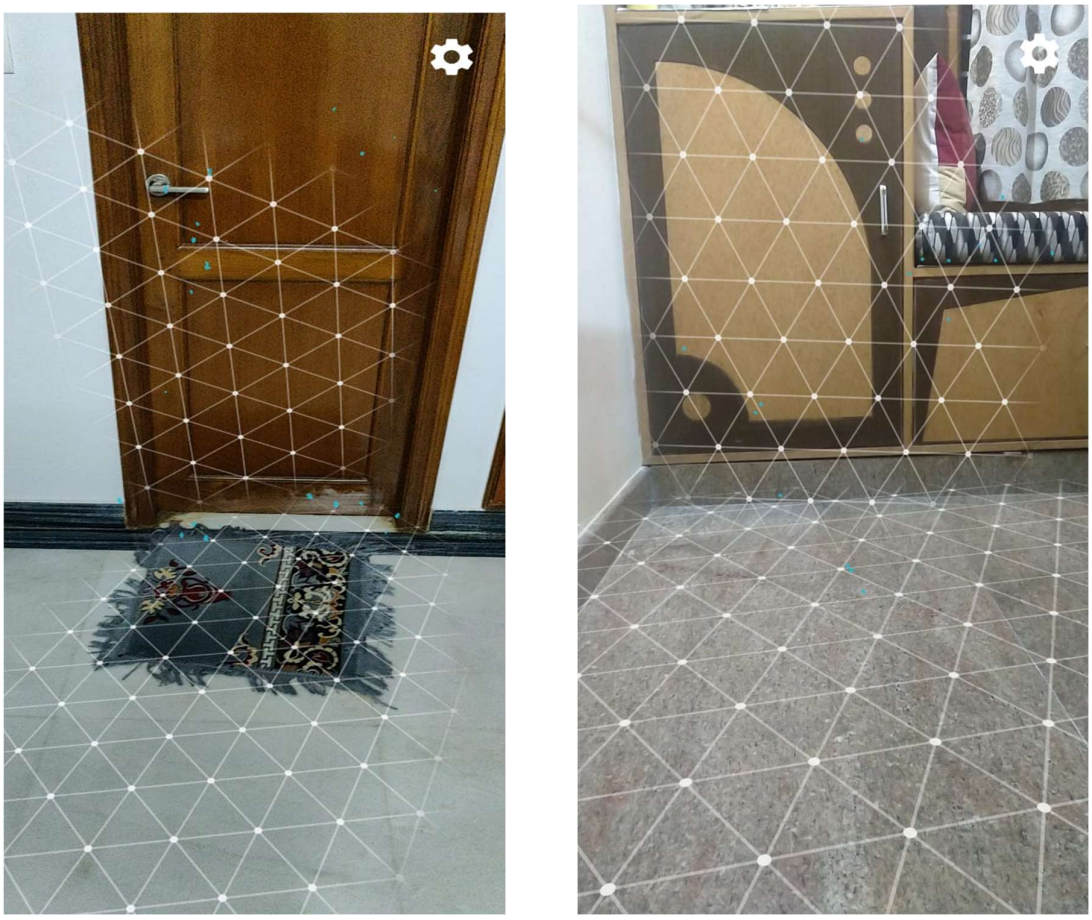

Figure 39. Dynamic Horizontal and Vertical Detection

Client side - 3D Map rendering:

The information of the detected surfaces is encoded as a string and then sent to the client end of the application where the message is processed and information is retrieved as explained in the methodology. This 3X3 matrix thus generated is then fed to the Unity engine which converts the matrix into a surface in the Game Object and renders it onto the screen.

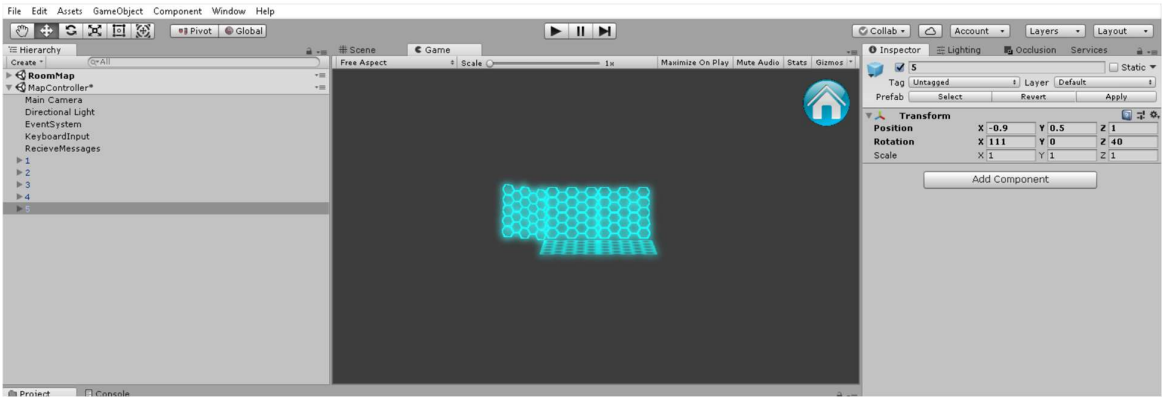

Figure 40. Plane Rendering in Unity

With every new surface detected a new set of values ar degenerated and sent which is deciphered and corresponding surface is rendered. The values of the 3X3 matrix (position, rotation and scaling) are fed into the transformation matrix on the right top corner of the screen. This creates a new surface in the unity engine and renders it onto the screen as seen in the below images.

Figure 41 shows 4 surfaces which are already detected and a corresponding 3D map is rendered on the screen. On identification of a new surface a new mesh is generated and as seen in the figure 42, it is appended to the already existing 3D map. Thus in this way the entire 3D map of the surrounding is generated.

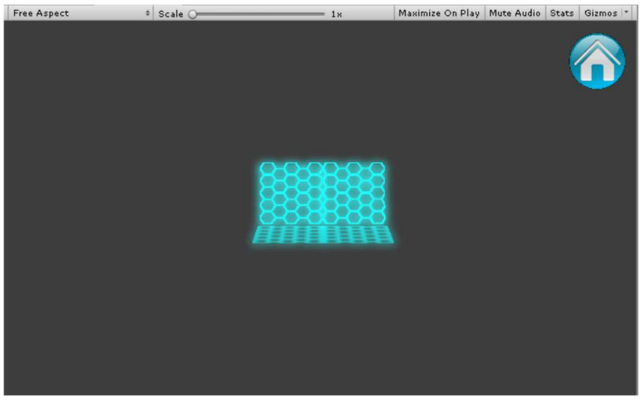

Figure 41. Before detection of the new surface

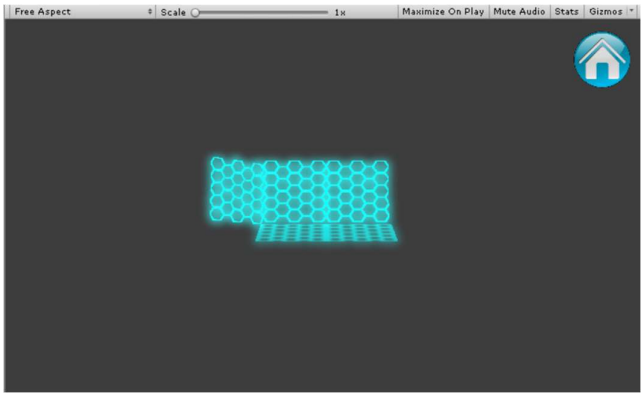

Figure 42. After detection of the new surfcae

These screenshots show the 3D map of an entire floor and illustrates how the 3D map will look once the process of scanning is done. The 3D map hence generated can be exported as a .fbx file which can be useful to solve various real-world problems.

The following screenshots represent the expected 3D map of a building, consisting of corridors and a room, from various angles. This 3D map is rendered on the Game Object for the user to view

Figure 43 represents the top view of the 3D map

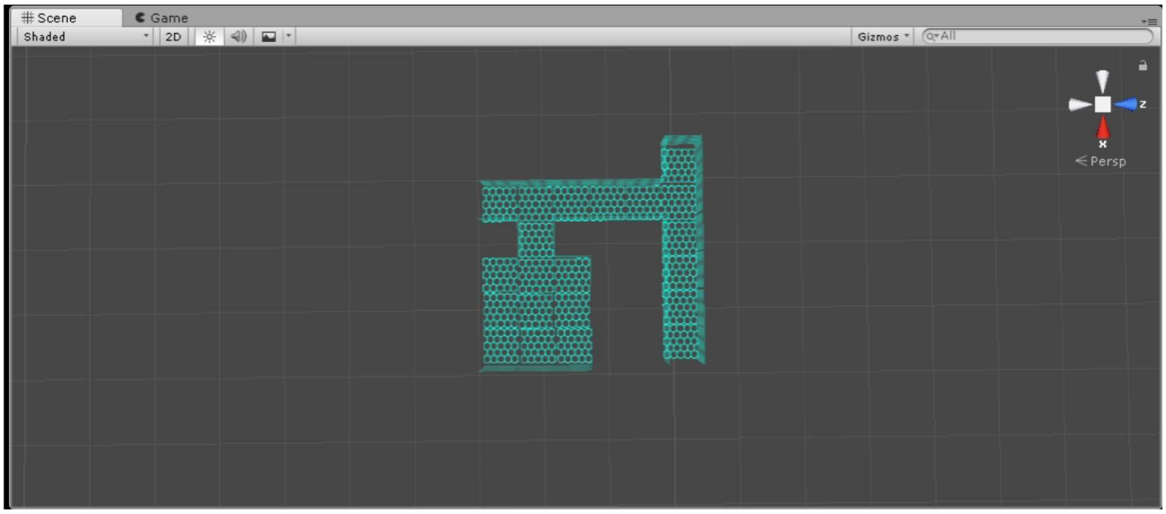

Figure 43. Top View of the 3D map

These are the various side angles to view the 3D map generated.

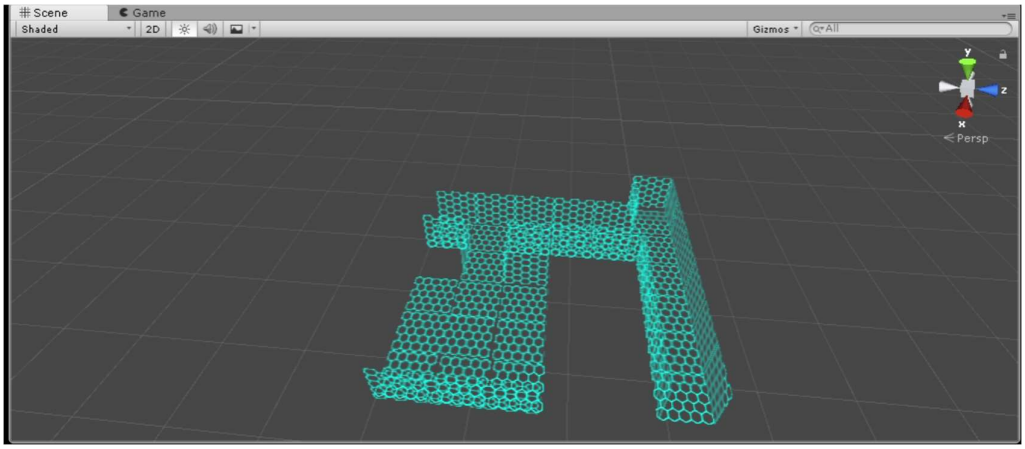

Figure 44

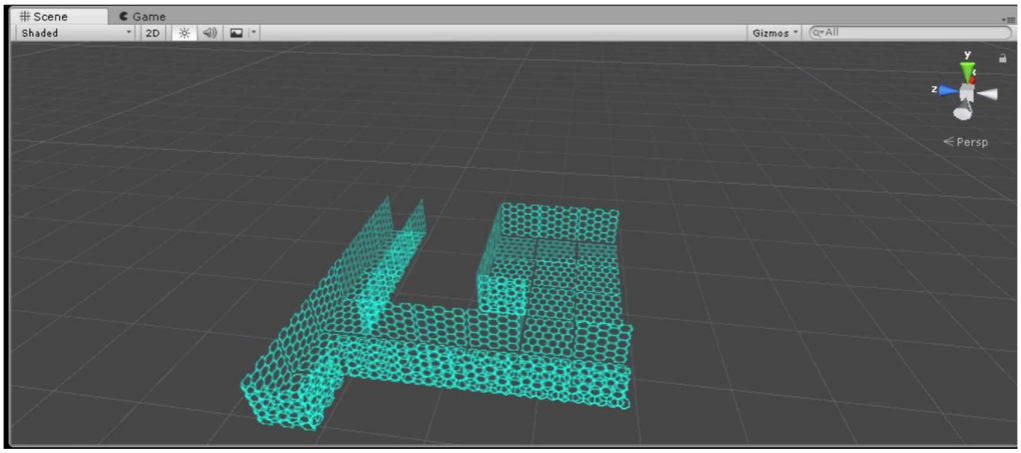

Figure 45

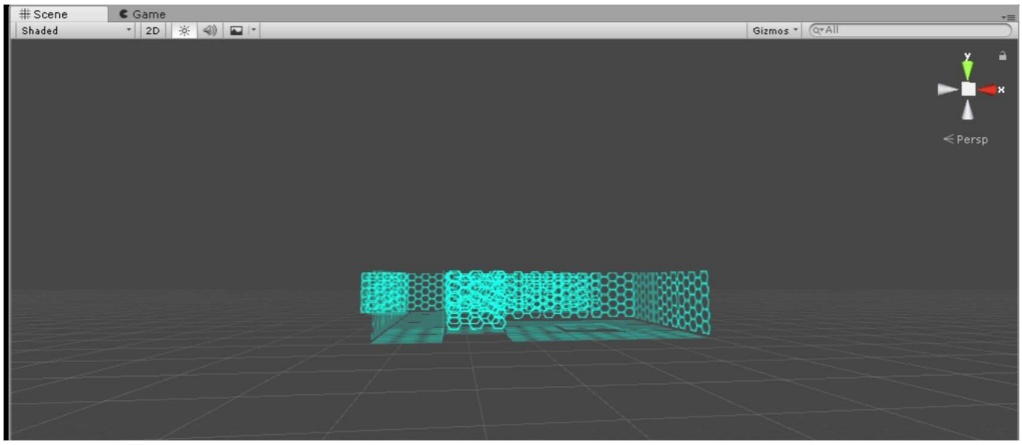

Figure 46

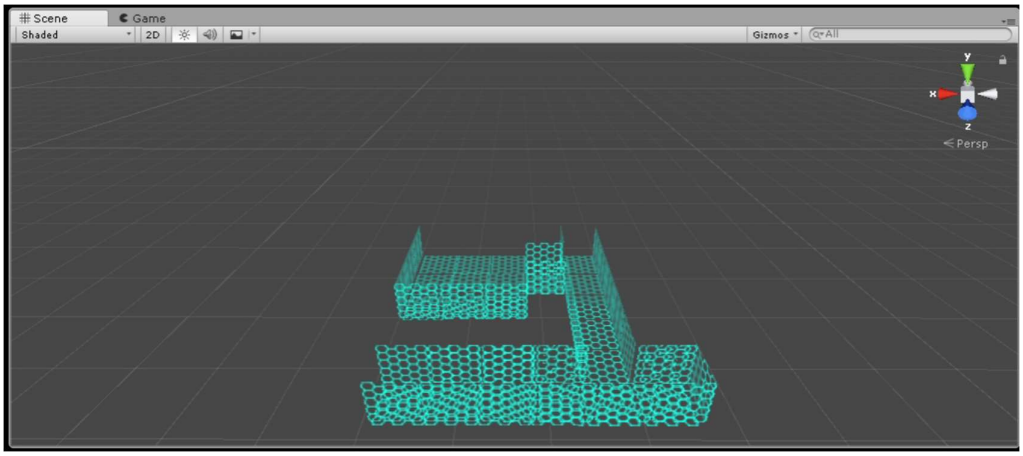

Figure 47

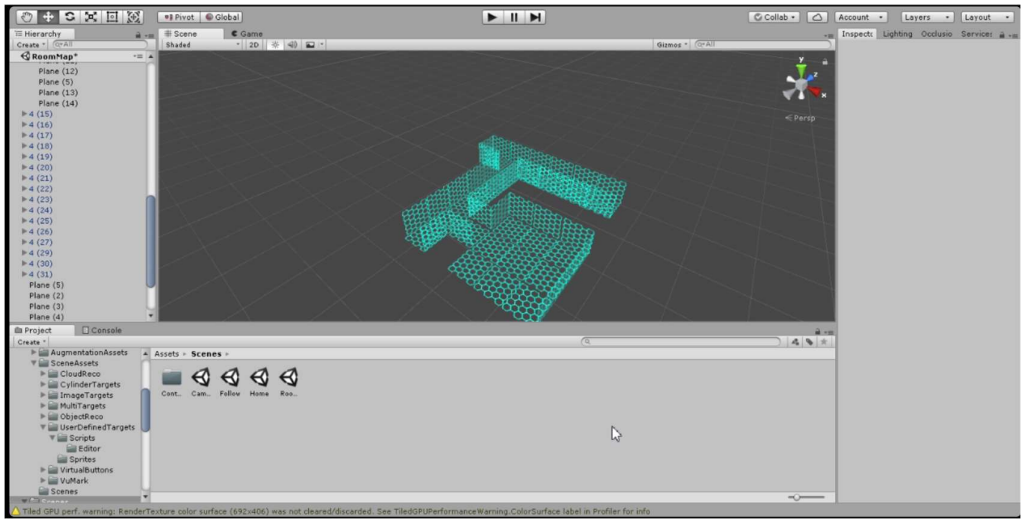

Figure 48. Screenshot of Map Rendering as seen in Unity

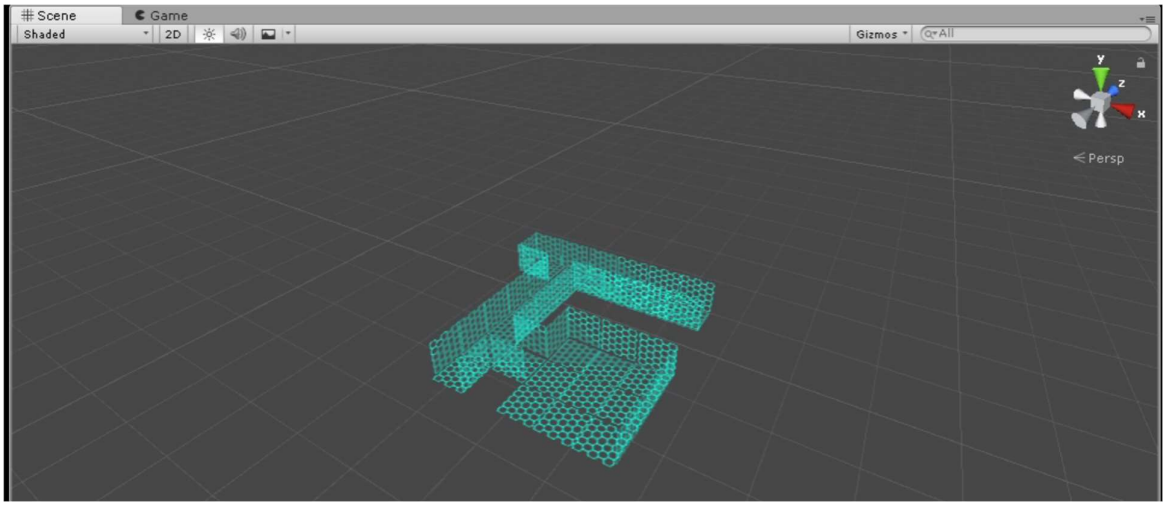

Figure 49

This is how it will look on the Unity Game Scene and this is the corresponding .fbx 3D image generated

5.3 QUANTIZATION OF THE RESULTS

The above section explained the results obtained from various test scenarios. This section will illustrate the data from the tests which helps in understanding the accuracy of the model.

Object	Parameter	Actual	Detected	Error	Error %	Accuracy %
Curtain	**Length**	1.2 m	1.13 m	0.08 m	6.66 %	94.17 %
	Width	0.5 m	0.46 m	0.04 m	8 %	92 %
	Area	0.6 sq. m	0.519 sq. m	0.08 sq. m	13.5 %	86.5 %
Door	**Length**	2 m	1.94 m	0.06 m	3 %	97 %
	Width	0.8 m	0.77 m	0.03 m	3.75 %	96.25 %
	Area	1.6 sq. m	1.49 sq. m	0.11 sq. m	6.88 %	93.12 %
Wall	**Length**	3 m	2.77 m	0.22 m	7.33 %	92.33 %
	Width	2.75 m	2.62 m	0.13	4.72 %	95.28
	Area	8.25 sq. m	7.26 sq m	1 sq. m	12 %	88 %
Floor	**Length**	4.57 m	4.9 m	- 0.33 m	7.22 %	92.77 %
	Width	2.75 m	2.98 m	- 0.23 m	8.36 %	91.63 %
	Area	12.57 sq. m	14.602 sq. m	2.032 sq. m	16.16 %	83.83 %

From the above data retrieved from different test cases, The application gives us an accuracy of **92 %.**

6. APPLICATION

This project aims to solve many real world problems such as mapping of indoor environments such as buildings and malls which have a plethora of applications. Some of the main applications of the project are:

a. Architectural planning and Indoor navigation:

The 3D map that this project generates can be used as a 3D architectural floor plan for various interior design purposes as well as construction planning. It is more advantageous than the regular 2D map as it adds an extra dimension to the map which helps the architects plan and design better. The 3D map of the floor provides different viewing angles which will help the workers visualise the building better. It helps provide a better perspective and understanding of the structure for the architects and the designers as well.

The product is designed in such a way that it is easy to use and accessible to the common man. The application has to be downloaded onto the smartphone and the architect is good to go. By just walking around with the smartphone in his hand, he will be able to effortlessly generate a 3D map.

This map can be used for navigation purposes in areas like shopping malls, museums and office complexes to navigate from one place to another. An indoor navigation system can be built based on the 3D map. This will make navigation within closed environments more reliable and easier.

b. Augmented Reality Applications:

Augmented reality and Mixed reality are the new age technologies that are revolutionising the world. Such visualisation and creating a virtual copy of the real world will aid in the development of many Augmented Reality applications.

The 3D map generated from this application can be imported into the Unity software to design other applications. Real world environments can be scanned and the 3D map thus generated can be used in building augmented reality based Games and applications thus creating a more realistic experience.

The scope of possibilities and use cases of VIO SLAM in robotics and Augmented Reality are endless and with every passing day dawns a new opportunity where such technologies can be put to better use.

7. CONCLUSION

This report consolidates the work done in building a prototype from scratch of the project 3D Mapping and Navigation Bot Using SLAM Techniques. The final prototype consists of a mobile robot which is custom built using acrylic sheets based on the required features. On the mobile robot is a provision to mount a smartphone which acts as the CPU of the bot. Along with this is the other onboard circuitry. The bot is made totally wireless and is controlled by the user via WIFI.

On the software front the application (both client side and server side) was built using Unity Platform along with Google's AR Core SDK and Depth API. The software is working as intended in detecting surfaces and rendering it onto the client side. The RGB image live feed is also working seamlessly with min lag in latency. Thus the deliverables of Phase 2 of the capstone project has been successfully achieved with proof of concept and final prototype.

8. FUTURE SCOPE

This project can he further be worked upon to solve more real-world problems. As of now the application renders a mesh onto the user end to create a 3D map. It can further enhance this into rendering the point cloud itself onto the user end to generate a more realistic 3D map. This can also be done with colour imaging and image processing to try and retain the colours and textures of the real-world objects as well.

The mobile robot in our case was manually controlled by the user. This feature can be extended so that the bot is purely autonomous and is independent of any human interference, thus allowing the bot to maneuver and map the unknown environment autonomously.

9. REFERENCES

9.1 Literature survey papers:

1. Tobias Feigl, Andreas Porada1, Steve Steiner1, Christoffer L¨offler, Christopher Mutschler and Michael Philippsen: "**Localization Limitations of ARCore, ARKit, and Hololens in Dynamic Large-scale Industry Environments**"

   https://www.researchgate.net/publication/340047854_Localization_Limitations_of_ARCore_ARKit_and_Hololens_in_Dynamic_Large-scale_Industry_Environments

2. Songmin Jia, Ke Wang, and Xiuzhi Li:" **Mobile Robot Simultaneous Localization and Mapping Based on a Monocular Camera"**

   https://www.hindawi.com/journals/jr/2016/7630340/

3. Xiaochen Zhang, Xiaoyu Yao,Yi Zhu,Fei Hu :" **An ARCore Based User Centric Assistive Navigation System for Visually Impaired People"**

   https://www.mdpi.com/2076-3417/9/5/989/htm

4. Jonas Halvarsson: "**Using SLAM-based technology to improve directional navigation in an Augmented Reality game"**

   https://www.semanticscholar.org/paper/Using-SLAM-based-technology-to-improve-directional-Halvarsson/00f441387f04f40aad6491ce23bdeb0ece17d12e?p2df

5. Alif Ridzuan Khairuddin, Mohamad Shukor Talib, Habibollah Haron Faculty of Computing Universiti Teknologi Malaysia Johor Bahru, Malaysia: **"Review on Simultaneous Localization and Mapping (SLAM)"**

   https://ieeexplore.ieee.org/document/7482163

6. Austin Corotan College of Science and Engineering Western Washington University Bellingham, Washington; Jianna Jian Zhang Irgen-Gioro College of Science and Engineering Western Washington University Bellingham, Washington: **"An Indoor Navigation Robot Using Augmented Reality"**

   https://ieeexplore.ieee.org/document/8813348

9.2 Other Reference links:

1. https://developers.google.com/ar/develop/java/depth/overview
2. https://en.wikipedia.org/wiki/Simultaneous_localization_and_mapping
3. https://arxiv.org/pdf/1910.01122.pdf
4. https://ipsjcva.springeropen.com/articles/10.1186/s41074-017-0027-2
5. https://courses.ece.cornell.edu/ece6930/ECE6930_Spring16_Final_MEng_Reports/SLAM/Real-time%20ROSberryPi%20SLAM%20Robot.pdf
6. https://blog.rabimba.com/2018/10/arcore-and-arkit-SLAM.html
7. https://developers.google.com/ar/discover/concepts
8. https://link.springer.com/article/10.1007/s40903-015-0032-7
9. https://www.arway.app/#mapping
10. https://www.coursera.org/lecture/ar/multi-plane-detection-and-spatial-mapping-bWhND
11. https://library.vuforia.com/features/environments/area-targets/area-targets-in-unity.html

Simultaneous mapping and localization (SLAM)

ORIGINALITY REPORT

5%	5%	3%	4%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	vinsonias.com Internet Source	1%
2	en.wikipedia.org Internet Source	1%
3	Submitted to Anglia Ruskin University Student Paper	<1%
4	Submitted to Southampton Solent University Student Paper	<1%
5	wikimili.com Internet Source	<1%
6	www.engineersgarage.com Internet Source	<1%
7	Submitted to University of Birmingham Student Paper	<1%
8	Submitted to University of Sunderland Student Paper	<1%
9	ijsrset.com Internet Source	<1%

10	www.iceefest.com Internet Source	<1%
11	Submitted to University of Wolverhampton Student Paper	<1%
12	Jon Peddie. "Augmented Reality", Springer Science and Business Media LLC, 2017 Publication	<1%
13	Submitted to Universiti Teknikal Malaysia Melaka Student Paper	<1%
14	Submitted to University of Queensland Student Paper	<1%
15	Frank Hoisl, Kristina Shea. "An interactive, visual approach to developing and applying parametric three-dimensional spatial grammars", Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 2011 Publication	<1%
16	www.bcut.ro Internet Source	<1%
17	www.yash-sharma.com Internet Source	<1%
18	www2.imm.dtu.dk Internet Source	<1%

19	ednex.me Internet Source	<1%
20	Submitted to Oregon State University Student Paper	<1%
21	www.routledge.com Internet Source	<1%

Exclude quotes	On	Exclude matches	< 5 words
Exclude bibliography	On		