Chapter 1

Introduction

1.1 Problem Statement

To develop an android application for smartphones which helps the user to navigate through an unfamiliar indoor environment using Augmented Reality as an interface.

1.2 Introduction to ARIN
(Augmented Reality Based Indoor Navigation System)

ARIN (Augmented Reality based Indoor Navigation system) is a product which is aimed at making indoor navigation in an unfamiliar indoor environment easier and more interactive with the use of AR. This system can navigate a user without continuous network connectivity.

Augmented Reality is a technology that superimposes a computer-generated image on a user's view of the real world, thus providing a composite through mobile devices, tablets, or smart glasses. It enhances experiences by adding virtual components such as digital images, graphics, or sensations as a new layer of interaction with the real world. AR studies in maintenance show promising results in enhancing human performance in carrying out technical maintenance tasks.

By using this trending technology we aim to develop an interactive indoor navigation system. Indoor positioning has gained popularity recently due to its potential to be used in the increasing complexity of indoor environment and because GPS signals are restricted to outdoor purposes. The main objective of this work is to design a new method to develop indoor positioning navigation system through image processing.

The idea of this work can be broadly applied to mobile devices such as mobile phones or tablets which is easily available for a common user.

Text recognition with OCR

OCR stands for Optical Character Recognition. It is a widespread technology to recognize text inside images, such as scanned documents and photos. OCR technology is used to convert virtually any kind of images containing written text (typed, handwritten or printed) into machine-readable text data. [6] Probably the most well-known use case for OCR is converting printed paper documents into machine-readable text documents. Once a scanned paper document went through OCR processing, the text of the document can be edited with word processors like Microsoft Word or Google Docs.

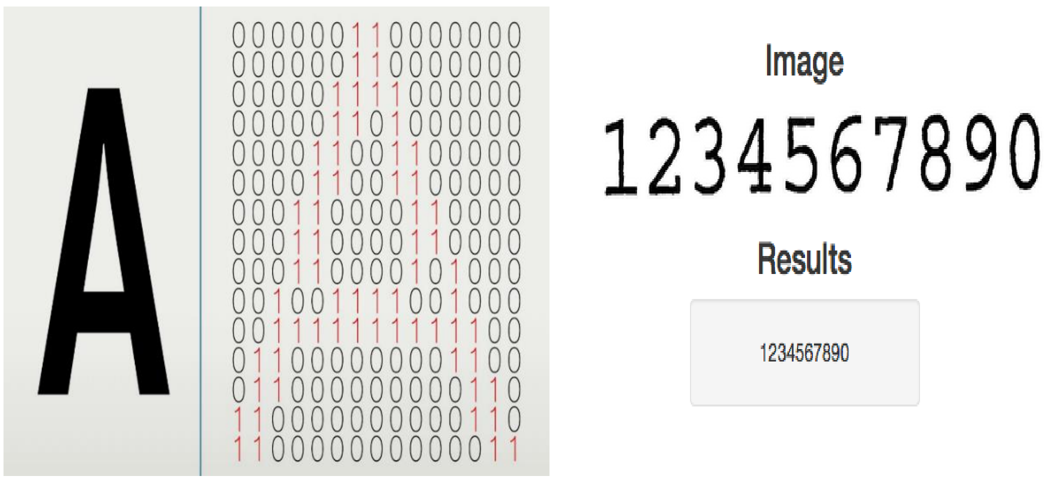


fig 1.1 Optical Character Recognition

Augmented Reality

"The origin of the word augmented is augment, which means to add or enhance something. In the case of Augmented Reality (also called AR), graphics, sounds, and touch feedback are added into our natural world to create an enhanced user experience." [1] In AR, we increase the usability of an interface by adding sensory information like computer generated images, sounds and in some cases, touch feedback over user's view of the real world. This enhances user's current perception of reality.

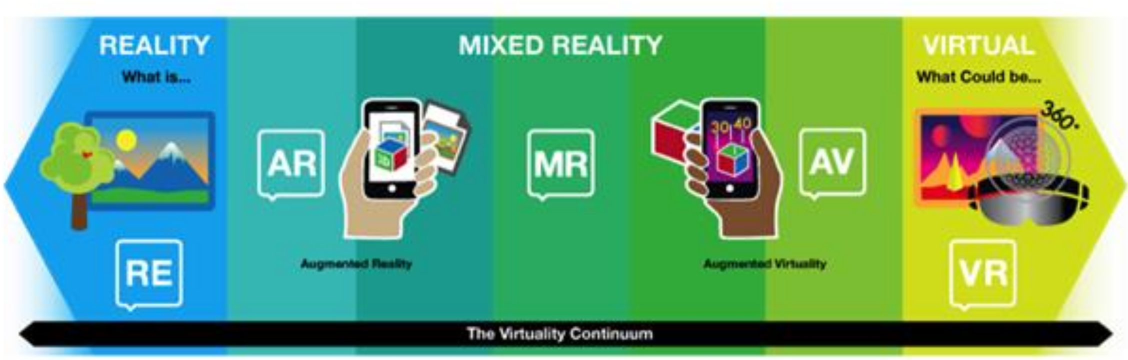


fig 1.2 Reality Virtuality continuum

Marker-based AR:

Marker-based augmented reality (also called Image Recognition) is one of the easiest type of AR for implementation. Based on image recognition, marker based AR works by recognising a visual marker in the environment and generating a digital image only when a known marker is sensed. The markers are simple and unique, such as the QR codes.[7] It finds its applications in the manufacturing and construction industry.

fig 1.3 Marker-based AR

Marker-less AR:

In markerless augmented reality, sensors in devices are used to detect the real world environment. With the emergence of smart devices, elements such as GPS, accelerometers, velocity meter, digital compass are pre included in the device which make the existence of Markerless AR possible. A strong force behind markerless augmented reality technology is the wide availability of smartphones and location detection features they provide. It is most commonly used for mapping directions, finding nearby businesses, and other location-centric mobile applications.[7]

fig 1.4 Marker-less AR

Projection based AR:

This is a relatively newer trend in AR, which works on the principle of advance projection technology that forecasts light onto real world surfaces and senses human interaction with the light. This process is carried out by distinguishing between the expected and altered projection.[7] This type of AR is usually used in manufacturing companies to assist in manufacturing, assembly, sequencing and training operations.

fig 1.5 Projection based AR

Superimposition based AR:

Superimposition based augmented reality works with object detection and recognition. It recognises an object and then replaces it by superimposing a digital image over it.[7] Some consumer-facing examples of superimposition based augmented reality can be organising furniture in a room or virtual trial rooms for clothing.

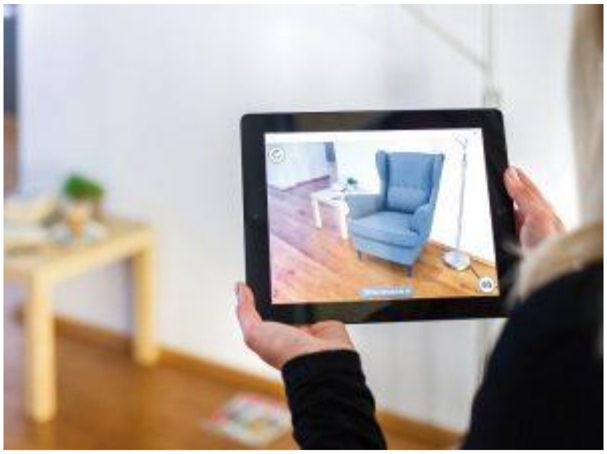


fig 1.6 Superimposition based AR

1.3 Motivation

Our motivation for this project stems from the fact that people are increasingly relying upon their smartphones to solve some of their common daily problems. One such problem that smartphones have not yet completely solved is indoor navigation. Navigation systems have been widely used in outdoor environments, but indoor navigation systems are still in early development stages. An indoor navigation app would certainly benefit users who are unfamiliar with a place. Tourists, for instance, would have a better experience if they could navigate confidently inside a tourist attraction without any assistance. In places such as museums and art galleries, the application could be extended to plan for the most optimal or ‘popular’ routes.

Furthermore, even though augmented reality is a topic having research more than 20 years in timeline, it started trending recently when Google announced their work on Google Glasses or when AR applications, photo filters and games went a hit. This void in research in this area along with interest in developing user-friendly applications further motivated us to take up this project.

1.4 Aim and Objectives

The aim of the project is to develop an AR based indoor navigation system for handheld devices like smartphones, tablets etc. using image comparison to identify location with an intuitive user interface.

- To make indoor navigation seamless and interactive through a hand-held mobile device using AR technology.

- To detect of current user location relative to nearby landmarks by vision-based feature extraction or visual positioning.

- To determine of shortest path from current location to desired destination and navigate user by augmenting directions in user's real-world view captured.

- To explore the intuitive depiction of information as observed in augmented reality systems.

Chapter 2

Literature Review

2.1 Augmented Reality (Base Paper)

Cognitive Augmented Reality. Nils Petersen, Didier Stricker (22 August 2015)

In the early 1990s, researchers Paul Milgram, Haruo Takemura, Akira Utsumi, and Fumio Kishino introduced a concept called the reality-virtuality (RV) continuum (Milgram, 1994). [3] While the researchers originally designed the reality-virtuality continuum to address mixed reality and the display technologies of the era, the original framework is still quite useful.

They defined mixed reality environments as those in which "real world and virtual world objects are presented together". Their definition of mixed reality served as an umbrella term that encompassed both virtual and augmented reality technologies.[5]

It explores and explains their Cognitive AR system for getting an AR manual, showing step-by-step instructions to a user wearing a Head-Mounted Display (HMD). It also presents a complete approach for creating augmented reality content for procedural tasks from video examples and give the details about the presentation of such content at runtime. According to the paper, today, applications face problems in giving a good demonstration due to some reasons like difficulty in content creation, ergonomic and hardware limitations and lack of "System Intelligence".[1]

In this paper they've shown that an effective approach for procedural workflows can be developed using visual observation only. An illustration of the simplified authoring and implementation process is shown. The main steps can be summarized as shown in fig 2.1.

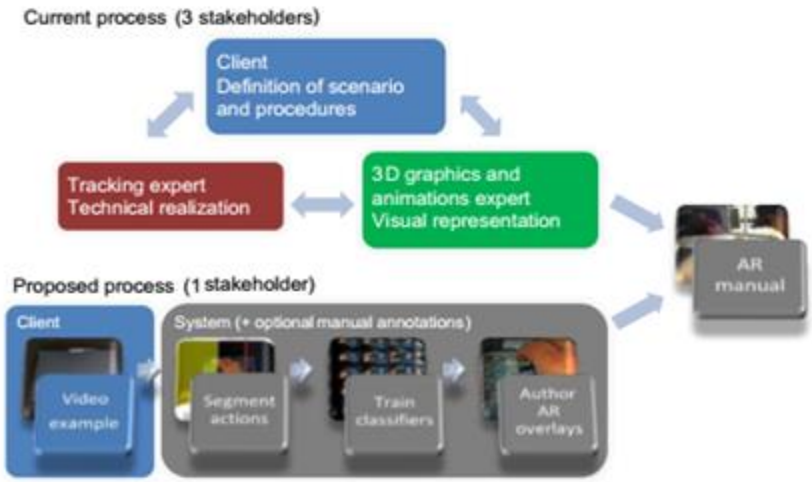


fig 2.1 Effective approach for procedural workflows[1]

The paper presents an overview of a novel learning-based authoring approach towards procedural task assistance using Augmented Reality. The resulting system is comprehensive and allows the fully automatic creation of Augmented Reality manuals from video examples as well as their context-driven presentation in AR. [1] The recording of a single reference recording of a workflow is sufficient for a practical system.

With availability of additional reference recordings, the system not only improves in precision and recall but is also able to estimate certain task-specific properties, like required level of accuracy and distinction of erratic and intended actions. The presented approach is the first to combine classical AR with machine learning, classification and basic reasoning methods, leading towards a cognitive system, aware about scene state and user actions. To underscore the extension of the merely spatial paradigm of Augmented Reality with the cognitive components, we call this combination Cognitive Augmented Reality.

Beside the presented live augmentation with Head-Mounted Display, the technology and methodology proposed in this paper opens many additional fields of application, such as the automated generation of written task documentation, support for documenting error indications, or analysis of maintenance procedures on a process level. [1]

2.2 Indoor Navigation (Base Paper)

BookMark: Appropriating Existing Infrastructure to Facilitate Scalable Indoor Navigation -Jennifer Pearson, Simon Robinson, Matt Jones. UK (2 February 2017)

BookMark provides visitors to the library with a detailed map to any desired book by simply scanning the barcode on the back of any other book in the library. Taking advantage of these pervasive codes by using them to determine where a user is standing in the physical space. By scanning the barcode on the back of any nearby book, BookMark can determine which book the user is holding, which shelf it was on, and, consequently, where they are positioned in the library. Using this information, and the details of the book they wish to locate, BookMark is then able to suggest a route from one location to the other, guiding the user to the correct shelf. [2]

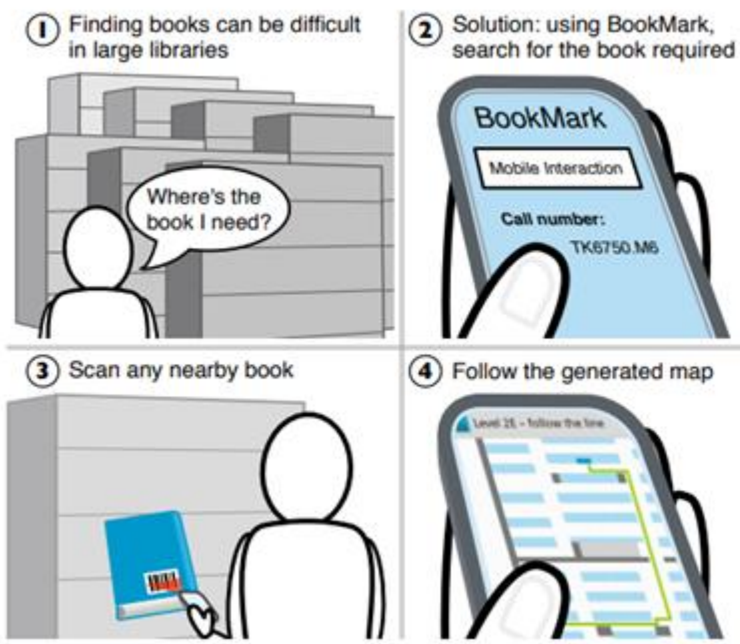


fig.2.2 Using BookMark system in libraries for locating required book

<u>The BookMark system</u>

BookMark was developed as an Android application, compatible with any phone that has a camera. The app allowed users to search the library catalogue to build up a list of items to find. To locate their desired books, users simply scan any nearby item, which produces a personalized map to the position of the book they are looking for, or indeed any item in the library's collection. [2]

Mapping the library

In order to provide fine-grained directions to specific bookcases within the library, maps were created for each floor. These maps included the locations of stairwells, doors, exits, elevators, obstacles such as pillars or interior walls and, of course, every bookcase within the physical space.

Second step was to convert existing library maps to SVG formatted graphics, and color coded each specific element to assist with navigation. For example, to distinguish between entrance and staircase types, as well as different sized and shaped bookcases and obstacles. [2]

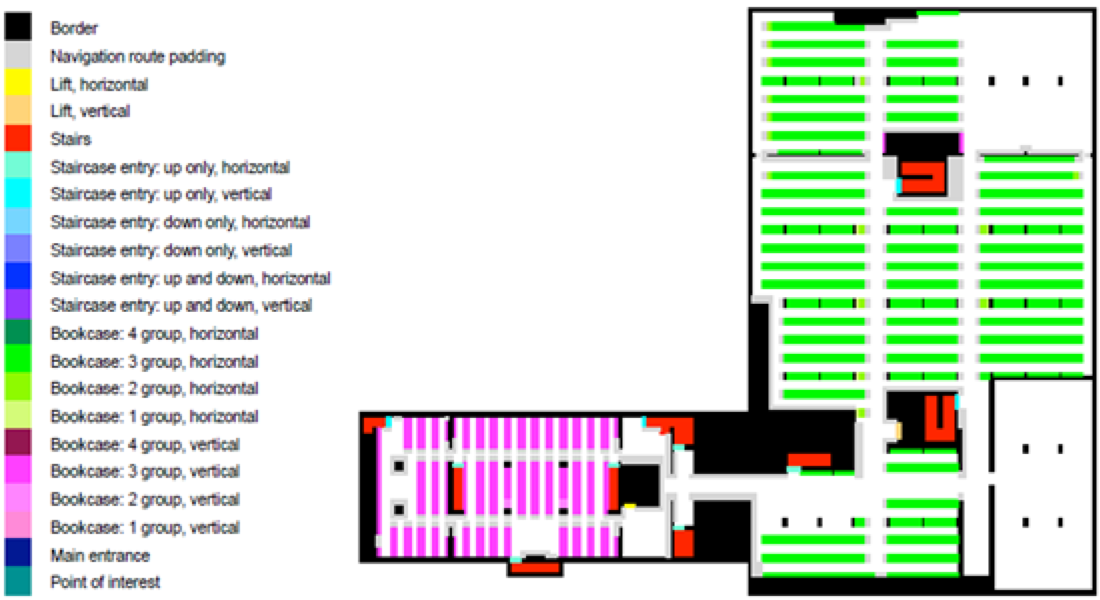


fig.2.3.The colour coded SVG map used to generate navigation instructions within the library

The next step in the process of mapping the library was to create a database of call number ranges to allow the system to determine the location of any item within the collection. This provides the core mapping capability of the system, taking advantage of the fact that the ordering of the items is known, and therefore allowing us to calculate the location of any item within the collection in relation to the locations of key items. [2]

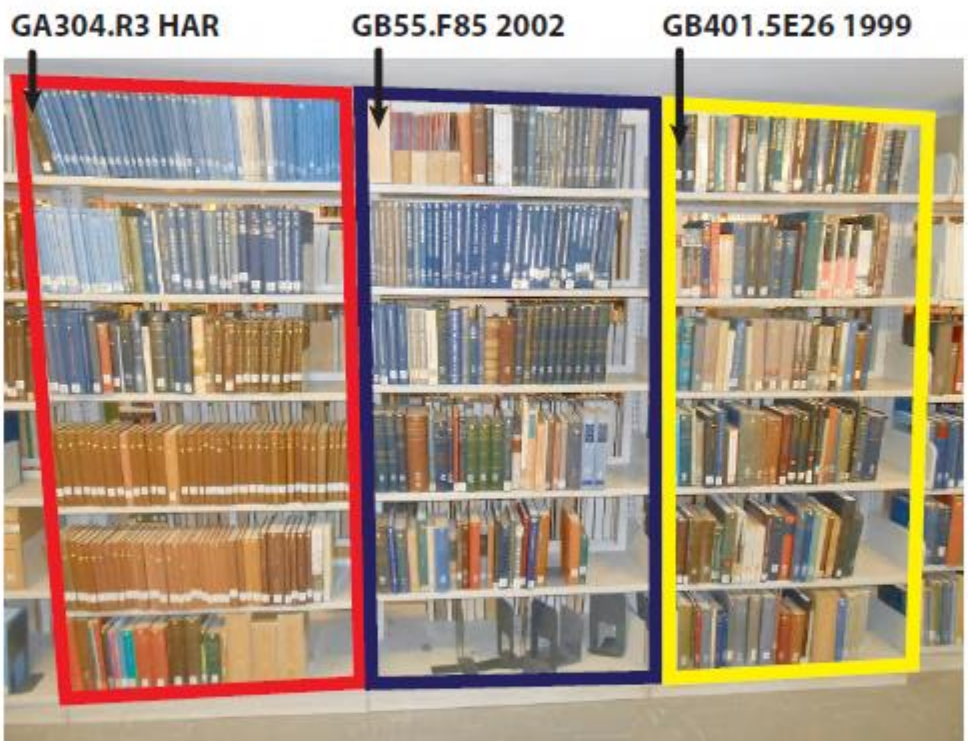


Fig.2.4 Arrangement of books in different stacks

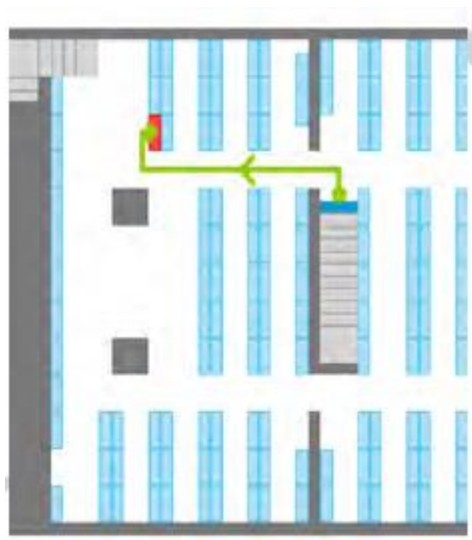
In the example shown, we can deduce that any books with call numbers between GA304.R3 HAR and GB55.F85 2002 will be in the stack highlighted in red, and any books with call numbers between GB55.F85 2002 and GB401.5E26 1999 will be in the bookcase highlighted in blue. Any items with call numbers greater than GB401.5E26 1999 will be located in the yellow stack (or adjacent bookcases), and so on. In this way, by recording the call number of the top-leftmost book of each bookcase in the library, we are able to locate any item. [2]

<u>Client Application</u>

Now users first select the items they wish to locate. This can be done within the application by searching the existing online library catalogue adding desired books to a list of items. After selecting a book from the list, the user must then inform the app of their current location in order for navigation to begin. This step can be done in one of two ways. The primary method is to scan the barcode on the back of any nearby book and application is able to pinpoint the closest bookshelf to where the user is standing, and hence draw a map from here to the desired item's location. If the user is not near to a bookshelf, they can choose to start navigation from the library's main entrance. If the desired book is on a different floor to where the user is standing, then the application will first navigate the user to the nearest stairwell or lift, and direct them to tap the screen once they arrive. Upon this action, the app gives directions from the stairwell to the next stage. [2]

Step 1. Scan the barcode

Step 2. Choose the required book



Step 3. Follow the route

fig. 2.5. Steps for BookMark Navigation

Advantages

1) Simple and cheap: Other than a smartphone this approach does not require any special hardware on user's side.

2) Digital barcodes: Use of an easily recognizable marker makes the use of existing infrastructure to facilitate precise navigation to specific books within a library.

Disadvantages

1) Books in the wrong shelf location: If a user scans a book that is in the wrong physical location (or searches for a book that has been placed in the wrong position), they will be shown an incorrect map.

2) Reorganization of books in the library: Libraries are constantly updating their collections, and will sometimes need to reorganize their shelves. If not managed, this process will result in inconsistencies between book locations in the app and physical book locations on library shelves.

Chapter 3

Project Requirements

3.1 Hardware Requirements:

- ARCore supported smartphone
- Camera: Minimum 4Mp resolution

3.2 Software Requirements:

- Android version: API 28 and above
- Cross platform support

3.3 Technology Stack:

- Android Studio (version 3.0)
- Database: XML
- Google ARcore SDK
- Unity Platform
- Google Poly for 3D objects
- MLkit on Firebase Backend as a Service

3.4 Developer System Requirements:

- RAM: 8 GB or above
- Processor: Intel core I5 or above

Chapter 4

Proposed System Architecture

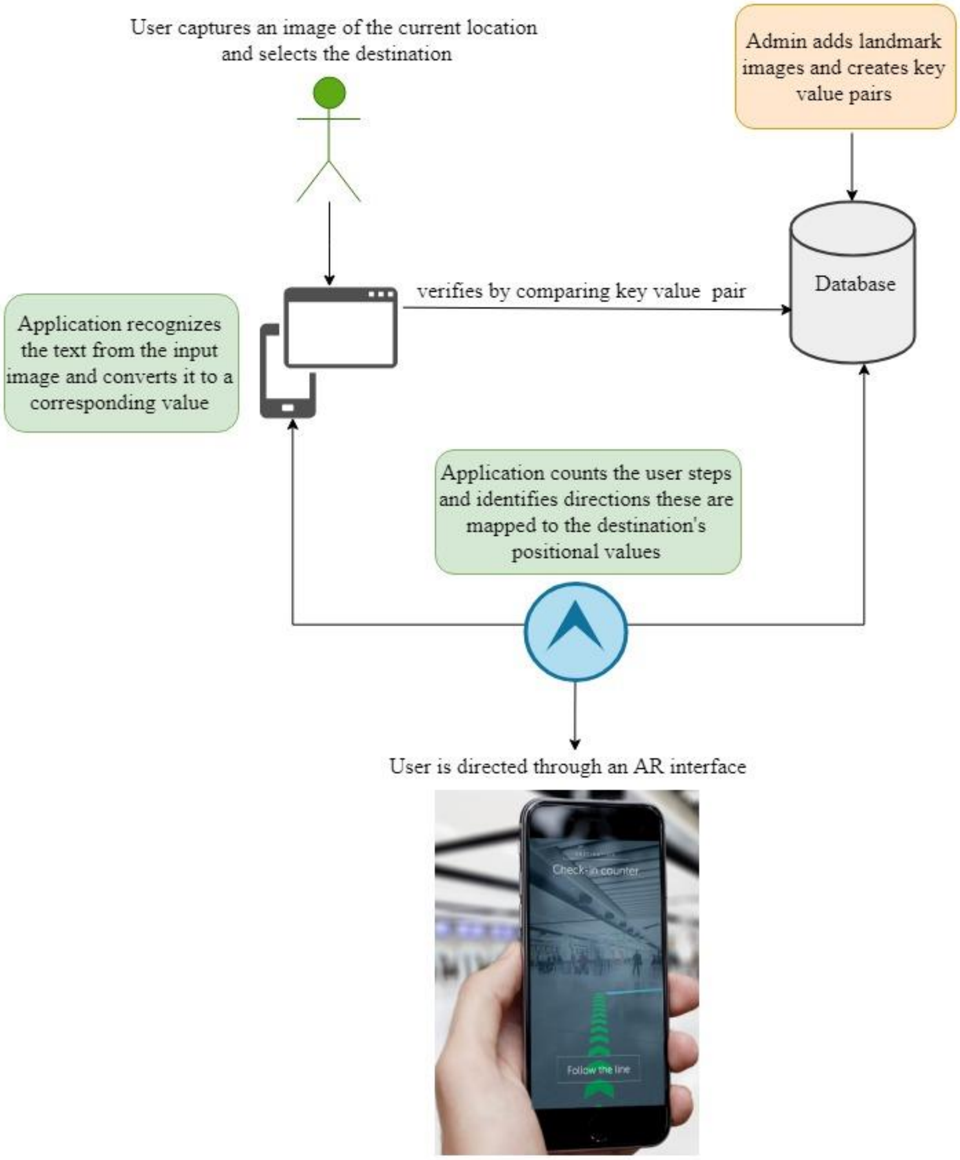


fig. 4.1.Proposed System Architecture

Procedure

Step 1:
- Admin adds landmark images of the intended area.
- Images are considered as key and corresponding value is assigned to them.
- They are added to the database.

Step 2:
- Source identification is done by recognizing text from the image captured by the user.
- Destination is selected from the list of available options.
- Text is compared to each key value pair from the database.
- Source is verified.

Step 3:
- Application counts the user steps and determine directions.
- These steps are mapped to the destination's positional values.

Step 4:
- User is guided based on the counting steps to the destination through an Augmented Reality interface

Chapter 5

Project Design

5.1 Use case diagram

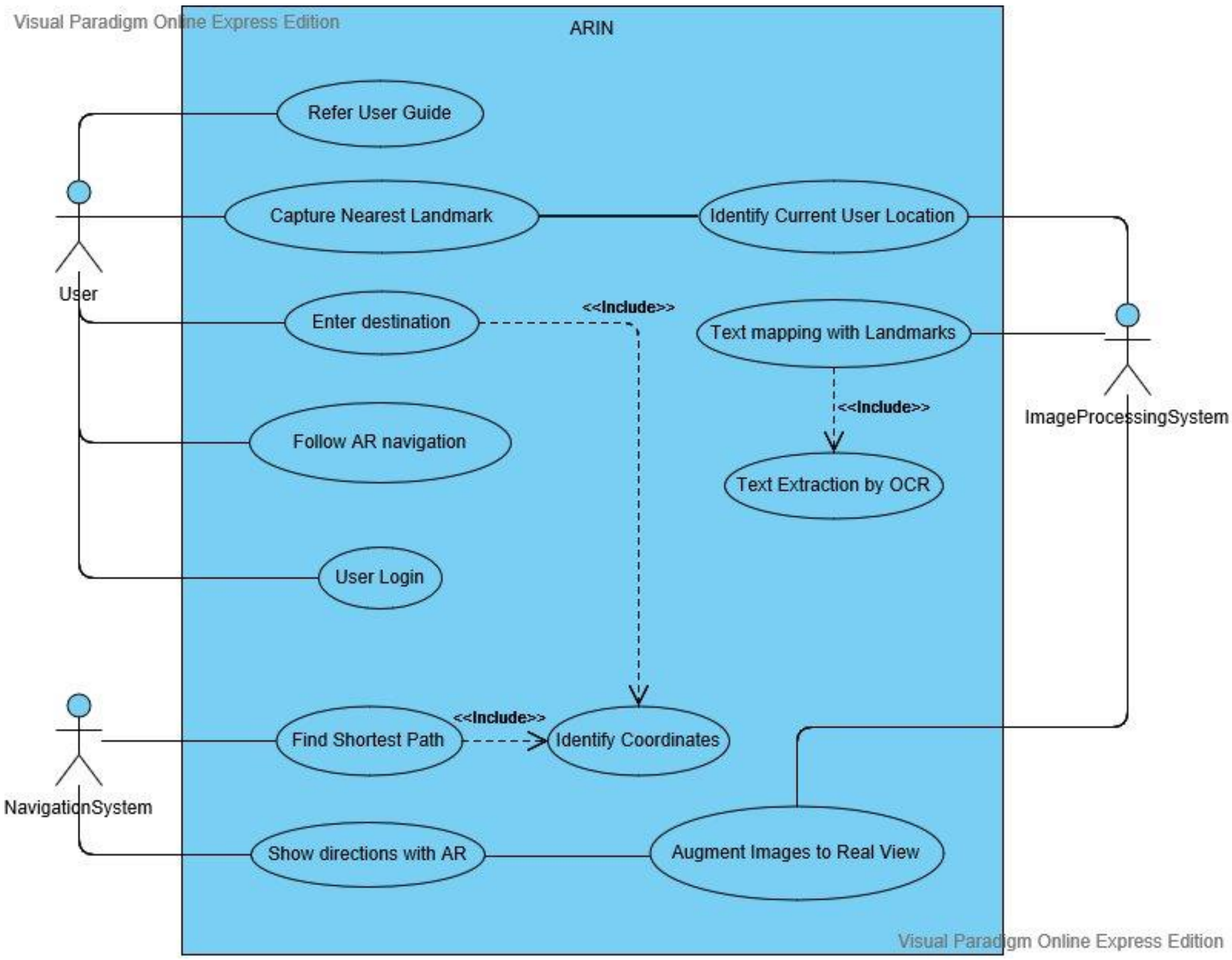


fig 5.1 Use case diagram

Description:

Actor	Goal	Brief
User	Use the application to reach desired destination	1.User login 2.Refer the user guide 3.Capture the nearest landmark 4.Enter the destination 5.Follow AR navigation
Image Processing System	Perform image processing and text extraction	1.Text extraction by OCR 2. Identifies the current location of the user. 3. Text mapping with landmarks 4. Augment images to real view
Navigation System	Guide the user through an AR interface	1. Identify the coordinates 2. Find shortest path 3. Show directions using AR

Table 5.1 Use case diagram description

5.2 Activity Diagram

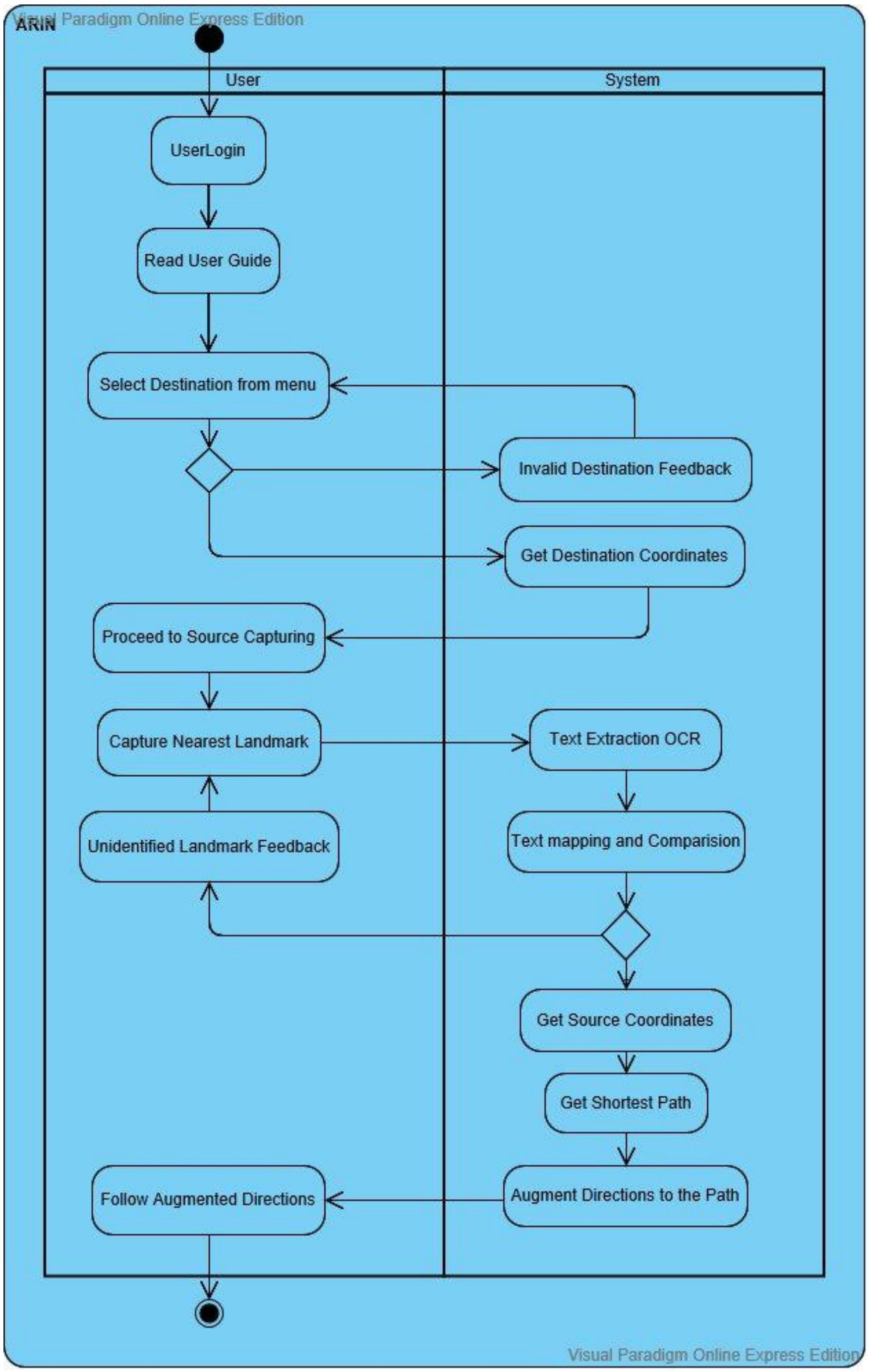


fig 5.2 Activity diagram

Description

1. The user enters login credentials
2. Guide is displayed by the application, the user reads the guide and follows the instructions
3. The user selects the destination from the menu
4. The system guide verifies the selected destination
5. The system obtains destination coordinates
6. If valid, the user proceeds to capture the source landmark
7. Text is extracted from the image by OCR
8. The text is mapped and compared with the database landmarks
9. If the source is successfully identified the coordinates are obtained and the shortest path is recognized
10. The user follows the augmented directions directed by the application

Chapter 6

System Implementation

fig 6.1 Implementation flow

6.1 Source Detection

- Source Detection here refers to determine the user’s current location relative to the intended region.
- The Admin identifies the prime locations of the navigation area, images of the same are stored in the database.
- Text recognizer is implemented in backend to detect text from the captured image.
- The text is stored as a key and corresponding value of its location is set and saved as a key value pair in the database.
- The user captures an image by smartphone camera of the nearest landmark, it is verified by the application by extracting the text from the image and matching it with the database keys.

Technology used (MLKit):

ML Kit is a mobile SDK that brings Google's machine learning expertise to Android and iOS apps in a powerful yet easy-to-use package.

ML Kit makes it easy to apply ML techniques in your apps by bringing Google's ML technologies, such as the Google Cloud Vision API, Mobile Vision, and TensorFlow Lite, together in a single SDK.

MLKit, an SDK that offers ready-to-use APIs such as:

1. Barcode scanning
2. Text recognition
3. Landmark detection
4. Image labelling

to build Android and iOS mobile apps [8]

Text recognition using ML Kit

1. Add Firebase to the Android project
2. Add the dependencies for the ML Kit Android libraries to your module (app-level) Gradle file (usually `app/build.gradle`):

   dependencies {
   //…
   implementation 'com.google.firebase:firebase-ml-vision:19.0.3'}

3. Recognise text in images

   - To recognize text in an image, create a FirebaseVisionImage object from either a Bitmap, media.Image, ByteBuffer, byte array, or a file on the device. Then, pass the FirebaseVisionImage object to theFirebaseVisionTextRecognizer's processImage method.

     FirebaseVisionImage image = FirebaseVisionImage.fromBitmap(bitmap);

   - To create a FirebaseVisionImage object from a media.Image object, such as when capturing an image from a device's camera, first determine the angle the image must be rotated to compensate for both the device's rotation and the orientation of camera sensor in the device:

   - Then, pass the media.Image object and the rotation value to FirebaseVisionImage.fromMediaImage():
     FirebaseVisionImage image =
     FirebaseVisionImage.fromMediaImage(mediaImage, rotation);

4. Get an instance of FirebaseVisionDocumentTextRecognizer:

   FirebaseVisionDocumentTextRecognizer detector =
   FirebaseVision.getInstance() .getCloudDocumentTextRecognizer();

5. Pass the image to the processImage method:

   detector.processImage(myImage)

```
.addOnSuccessListener(new OnSuccessListener<FirebaseVisionDocumentText>() {
   @Override
       public void onSuccess(FirebaseVisionDocumentText result) {
          // Task completed successfully
          // ...
       } })
    .addOnFailureListener(new OnFailureListener() {
       @Override
       public void onFailure(@NonNull Exception e) {
          // Task failed with an exception
          // ...
       }       });
```

6. Extract text from blocks of recognized text:

   If the text recognition operation succeeds, it will return a FirebaseVisionDocumentText object. AFirebaseVisionDocumentText object contains the full text recognized in the image and a hierarchy of objects that reflect the structure of the recognized document:

   - FirebaseVisionDocumentText.Block
   - FirebaseVisionDocumentText.Paragraph
   - FirebaseVisionDocumentText.Word
   - FirebaseVisionDocumentText.Symbol

MainActivity.java

- The code above configures the text recognition detector and calls the function (processTextRecognitionResult) with the response.

```
private void runTextRecognition() {FirebaseVisionImage image =
FirebaseVisionImage.fromBitmap(mSelectedImage);

FirebaseVisionTextRecognizer recognizer =
FirebaseVision.getInstance()            .getOnDeviceTextRecognizer();

mTextButton.setEnabled(false);

recognizer.processImage(image)

.addOnSuccessListener

new OnSuccessListener<FirebaseVisionText>() {

@Override

public void onSuccess(FirebaseVisionText texts) {

mTextButton.setEnabled(true);
```

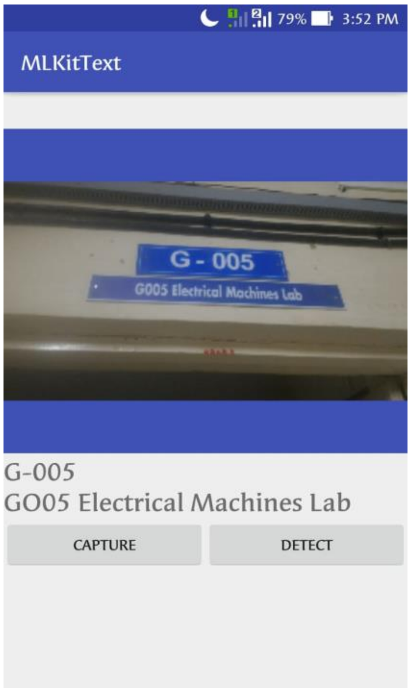
```
processTextRecognitionResult(texts);       }   })
.addOnFailureListener(
new OnFailureListener() {
@Override
public void onFailure(@NonNull Exception e) {
mTextButton.setEnabled(true);
e.printStackTrace();                       } }); }
```

MainActivity.java

- The function (processTextRecognitionResult) in the MainActivity class to parse the results and display them in your app.

```
 private void processTextRecognitionResult(FirebaseVisionText texts) {
List<FirebaseVisionText.TextBlock> blocks = texts.getTextBlocks();
if (blocks.size() == 0) {
showToast("No text found");  return;       }
mGraphicOverlay.clear();
for (int i = 0; i < blocks.size(); i++) {
List<FirebaseVisionText.Line> lines = blocks.get(i).getLines();
for (int j = 0; j < lines.size(); j++) {
List<FirebaseVisionText.Element> elements = lines.get(j).getElements();
for (int k = 0; k < elements.size(); k++) {
Graphic textGraphic = new TextGraphic(mGraphicOverlay, elements.get(k));
 mGraphicOverlay.add(textGraphic);            }         }      }    }
```

Results:



By creating key-value pairs of the location give the following results -

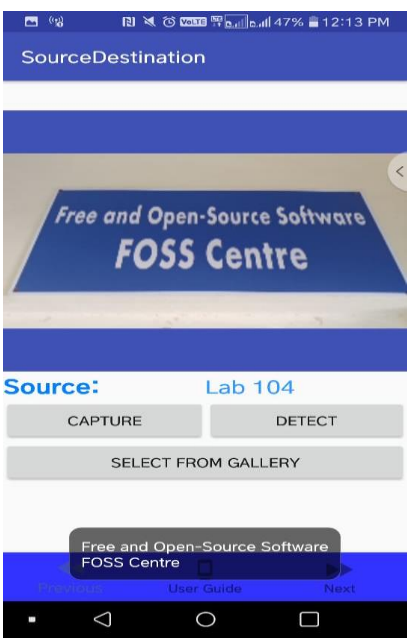


fig 6.2 Source identification results

6.2 Navigation

It is the process or activity of guiding the user to the destination following an appropriate route.

Technology used (Pedometer):

"Pedometer" records the number of steps you have walked and displays them again along with the number of calories that you have burned, distance, walking time and speed per hour.

It is easy to use. Once you push the Start button, all you have to do is hold your smartphone as you always do and walk.[9]

<u>MainActivity.java</u>

```
public class MainActivity extends AppCompatActivity implements SensorEventListener, StepListener
{private StepDetector simpleStepDetector;SensorManager sensorManager; Sensor accelerometer;

private static final String TEXT_NUM_STEPS = "Number of Steps: ";

private int numSteps;    private TextView TvSteps;

 protected void onCreate(Bundle savedInstanceState) {

 super.onCreate(savedInstanceState);

 setContentView(R.layout.activity_main);

 sensorManager = (SensorManager) getSystemService(SENSOR_SERVICE);

 accelerometer = sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);

 simpleStepDetector = new StepDetector(); simpleStepDetector.registerListener(this);

 TvSteps = findViewById(R.id.tv_steps);

 Button btnStart = findViewById(R.id.btn_start); Button btnStop = findViewById(R.id.btn_stop);

 btnStart.setOnClickListener(new View.OnClickListener()

 public void onClick(View arg0) {            numSteps = 0;
sensorManager.registerListener(MainActivity.this,    accelerometer,
SensorManager.SENSOR_DELAY_FASTEST);     }       });

 btnStop.setOnClickListener(new View.OnClickListener() }

 public void onClick(View arg0) {sensorManager.unregisterListener(MainActivity.this);          }       });

 public void onSensorChanged(SensorEvent event) {if (event.sensor.getType() ==
sensor.TYPE_ACCELEROMETER) {
simpleStepDetector.updateAccelerometer(                event.timestamp, event.values[0],
event.values[1], event.values[2]);       }    ;

 public void step(long timeNs) {       numSteps++;       TvSteps.setText(TEXT_NUM_STEPS +
numSteps);    }
```

Results:

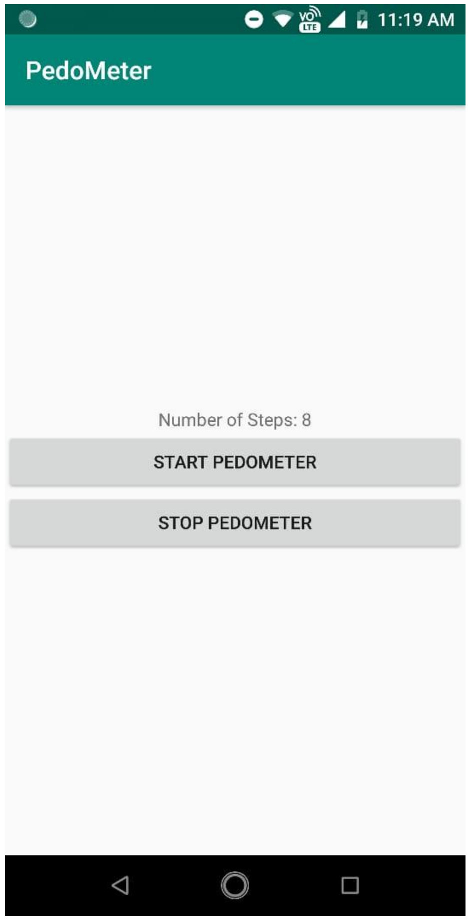


fig 6.3 Pedometer

6.3 Augmented Realty

Augmented reality (AR) is a type of interactive, reality-based display environment that takes the capabilities of computer generated display, sound, text and effects to enhance the user's real-world experience.

Augmented reality combines real and computer-based scenes and images to deliver a unified but enhanced view of the world.

Technology used (ARCore):

ARCore is Google's platform for building augmented reality experiences that seamlessly blend the digital and physical worlds. [10]

ARCore uses three key technologies to integrate virtual content with the real world as seen through your phone's camera:

1. Motion tracking allows the phone to understand and track its position relative to the world.
2. Environmental understanding allows the phone to detect the size and location of flat horizontal surfaces like the ground or a coffee table.
3. Light estimation allows the phone to estimate the environment's current lighting conditions.

<u>MainActivity.java</u>

```
public class MainActivity extends AppCompatActivity {
private ArFragment fragment;    private PointerDrawable pointer = new PointerDrawable();
private boolean isTracking;    private boolean isHitting;
public void onClick(View view) {
fragment = (ArFragment)
private void onUpdate() {boolean trackingChanged = updateTracking();
View contentView = findViewById(android.R.id.content);
if (isTracking) { boolean hitTestChanged = updateHitTest();
if (hitTestChanged) {pointer.setEnabled(isHitting);contentView.invalidate();   }   }
private boolean updateTracking() {Frame frame = fragment.getArSceneView().getArFrame();
boolean wasTracking = isTracking;
isTracking = frame != null && frame.getCamera().getTrackingState() == TrackingState.TRACKING;
return isTracking != wasTracking;}private boolean updateHitTest() {
Frame frame = fragment.getArSceneView().getArFrame();
android.graphics.Point pt = getScreenCenter();
```

```
List<HitResult> hits; boolean wasHitting = isHitting; isHitting = false;
if (frame != null) { hits = frame.hitTest(pt.x, pt.y); for (HitResult hit : hits) {Trackable trackable = hit.getTrackable();
if (trackable instanceof Plane &&((Plane) trackable).isPoseInPolygon(hit.getHitPose())) {
isHitting = true; break;}}}return wasHitting != isHitting; }
private android.graphics.Point getScreenCenter() {
View vw = findViewById(android.R.id.content);
return new android.graphics.Point(vw.getWidth()/2, vw.getHeight()/2);
} private void initializeGallery() {
  LinearLayout gallery = findViewById(R.id.gallery_layout);
 ImageView andy = new ImageView(this);
  andy.setImageResource(R.drawable.droid_thumb);
  andy.setContentDescription("andy");
  andy.setOnClickListener(view ->{addObject(Uri.parse("andy.sfb"));});
 gallery.addView(andy);
  }private void addNodeToScene(ArFragment fragment, Anchor anchor, Renderable renderable) {
  AnchorNode anchorNode = new AnchorNode(anchor);
   TransformableNode node = new TransformableNode(fragment.getTransformationSystem());
   node.setRenderable(renderable);  node.setParent(anchorNode);
  fragment.getArSceneView().getScene().addChild(anchorNode);
   node.select();    }
```

Results:

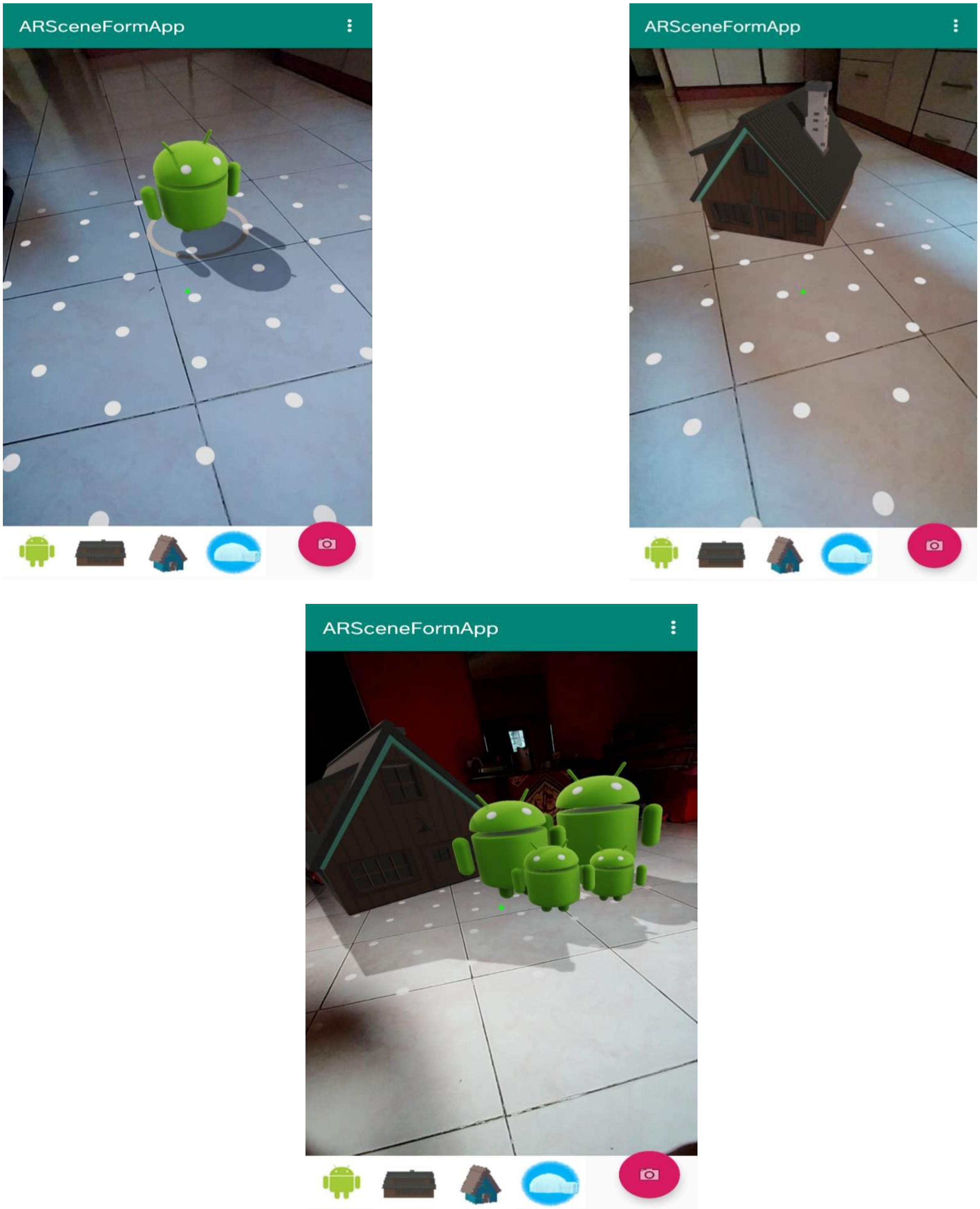


fig 6.4 AR Implementation results

Chapter 7

Testing

1. Capture image as a landmark

Subject : Functional **Status :** Design

Designer : admin **Type :** Manual

Objective : Give image as an input for searching an optimal path

Pre Condition : The user must have an android phone with a decent camera and our application installed and running

Test Data: Image with prominent landmark

Steps	*Description*	*Expected Result*
Step 1	Open camera of mobile from our application	Camera opens successfully
Step 2	Click picture of a landmark	Picture of landmark is saved
Step 3	Click on proceed button	Location identified. Display message “Source location captured”

Table 7.1 Capture image as landmark (valid)

2. Capture image as a landmark[invalid]

Subject : Functional **Status :** Design

Designer : admin **Type :** Manual

Objective : Give image as an input for searching an optimal path

Pre Condition : The user must have an android phone with a decent camera

Test Data: Image without distinguishable landmark

Steps	*Description*	*Expected Result*
Step 1	Open camera of mobile from our application	Camera opens successfully
Step 2	Click picture of a landmark	User clicks the picture of his/her current landmark
Step 3	Click on proceed button	Location unidentifiable. Display message “Failed to identify source. Capture new image”

Table 7.2 Capture image as landmark (invalid)

3. Enter Destination Location (Valid)

Subject : Functional **Status :** Design

Designer : admin **Type :** Manual

Objective : Give image as an input for searching an optimal path

Pre Condition : The user must have an android phone with a decent camera

Test Data: Room 101

Steps	*Description*	*Expected Result*
Step 1	Enter destination location	Display selected destination
Step 2	Click on proceed button	Destination identified. Display “Capture source”

Table 7.3 Enter Destination Location (Valid)

4. Enter Destination Location (Invalid)

Subject : Functional **Status :** Design

Designer : admin **Type :** Manual

Objective : Give image as an input for searching an optimal path

Pre Condition : The user must have an android phone with a decent camera

Test Data: Room 701

Steps	*Description*	*Expected Result*
Step 1	Enter destination location	Display selected destination
Step 2	Click on proceed button	Destination match not found. Display message “Enter valid destination”

Table 7.4 Enter Destination Location (Invalid)

5. Image Comparison[match found]

Subject : Performance **Status :** Design

Designer : admin **Type :** Manual

Objective : Good quality image as an input by the user

Pre Condition : The user must have an android phone with a decent camera

Test Data

Steps	*Description*	*Expected Result*
Step 1	Input an image	Compare with database images
Step 2	Compare image	Error value less than threshold. Image identified as same. Return message “Images are same”

Table 7.5 Image Comparison (match found)

6. Image Comparison[match not found]

Subject : Performance **Status :** Design

Designer : admin **Type :** Manual

Objective : Good quality image as an input by the user

Pre Condition : The user must have an android phone with a decent camera

Test Data

Steps	*Description*	*Expected Result*
Step 1	Input an image	Compare with database images
Step 2	Compare image	Error value more than threshold. Image identified as different. Return message “Images not same”

Table 7.6 Image Comparison (match not found)

7. Navigation[path found]

Subject : Functional **Status :** Design

Designer : admin **Type :** Manual

Objective : Good quality image as an input by the user

Pre Condition : The user must have an android phone with a decent camera

Test Data: User follows directions correctly

Steps	*Description*	*Expected Result*
Step 1	Enter source	Source location gets identified
Step 2	Enter destination	Destination record is identified
Step 3	Click proceed	Optimum path is calculated. Navigate user to destination

Table 7.7 Navigation(path found)

8. Navigation[path not found]

Subject : Functional **Status :** Design

Designer : admin **Type :** Manual

Objective : Good quality image as an input by the user

Pre Condition : The user must have an android phone with a decent camera

Test Data: User follows directions correctly

Steps	*Description*	*Expected Result*
Step 1	Enter source	Source location gets identified
Step 2	Enter destination	Destination record is identified
Step 3	Click proceed	Path not found. Display “Destination not reachable”

Table 7.8 Navigation(path not found)

9. Augmented Reality

Subject : Functional **Status :** Design

Designer : admin **Type :** Manual

Objective : Good quality image as an input by the user

Pre Condition : The user must have an android phone with a decent camera

Steps	*Description*	*Expected Result*
Step 1	Proceed to navigation	User gets navigated to the destination using Augmented Reality. Easy guidance for the user with the help of AR

Table 7.9 Augmented Reality

Chapter 8

Project Plan

2018-19 Semester-1	
Duration	**Plan**
June 2018	
First week	Discuss about the project flow
Second week	System Architecture designing
Third Week	Finalization of technology stack to be used
Fourth Week	Exploring the finalized technology stack
July 2018	
First Week	Studying of image comparison techniques
Second Week	Comparison between the studied techniques of image comparison
Third Week	Wireframing
Fourth Week	OpenCV installation and study
August 2018	
First Week	Integration of OpenCV and Android
Second Week	Discussion about the application design
Third Week	Exploring android studio in detail
Fourth Week	Making some demo apps in android and learning to integrate them with OpenCV
September 2018	
First Week	Studying the results of image comparison techniques
Second Week	Designing the android application for the undertaken project
Third Week	Report preparation and reviewing

2018-19 Semester-II	
Duration	**Plan**
December 2018	
Second Week	Paper Submission for Springer Publication
Third Week	Study of navigation techniques
Fourth Week	Study of visualization techniques
January 2019	
First Week	Study of Unity and ARCore
Second Week	Springer Conference paper discussion, SLAM research
Third Week	Udemy courses discussion, Sample AR models, Conference PPT planning
Fourth Week	Application for departmental floor plan, Conference PPT discussion
February 2019	
First Week	Experimenting with various navigation techniques, Springer conference
Second Week	Preparing the departmental floor map that includes the distances between landmarks
Third Week	Implementation of Pedometer to count the steps taken by user while walking from one landmark to the other
Fourth Week	Implementation of navigational compass to detect directions
March 2019	
First Week	Updation of source detection application
Second Week	Implementation of Augmented Reality
Third Week	Integration of different modules into one
Fourth Week	Preparation for Infovision(Idea Presentation Competition)
April 2019	
First Week	Reviewing of entire project, presentation and report preparation

Table 8.1 Project Plan

Chapter 9

Applications

Airports
Directions directly to the gate, a specific shop or to a restaurant. Improve the traveler's experience and reduce the number of people missing their flights.

Universities

Offer students, employees and visitors indoor navigation and guide them to lecture rooms, canteens and printers.

Shopping centres

With indoor wayfinding shoppers are guided to specific shops and restaurants, which makes the shopping experience more fun and satisfying

Hospitals

Indoor navigation can improve your staff's efficiency and reduce patient and visitor stress by providing a familiar and easy-to-use wayfinding tool.

Other large venues

Indoor navigation can be applied to all large venues. Corporate headquarters, convention centres, theme parks and many more.

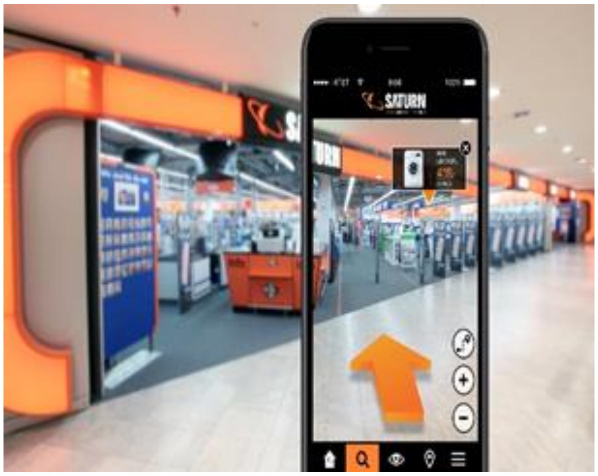


Table 9.1 Applications

Chapter 10

Conclusion

The various navigation techniques that exist currently such as Wi-Fi fingerprinting, Bluetooth Low Energy etc. have an external hardware requirement such as Wi-Fi access points or beacons respectively. If we are trying to implement the indoor navigation system on smartphones, these are additional requirements that our application eliminates. The accuracy can be improved with the help of computer vision.

The various augmented reality types, and implementation methods studied show the different possible implementations of this technology. AR can be implemented with head mounted displays as well as hand held devices like smartphones. AR can be marker based or markerless. A markerless AR system running on smartphones instead of HMDs has better usability and also does not require any distinct images or codes to be stored additionally.

Chapter 11

Future Scope

1. It is possible to extend the scope of the application to multiple floors (using barometer). Currently, it is devised only for a single floor.

2. Redirecting the user while navigation, in case the user takes an incorrect step can be handled in the future scope.

3. This application has been customized only for a particular indoor space, but can be later modeled dynamically for various indoor spaces.

4. The application currently works on Android devices but can be extended to iOS as well.

Chapter 12

References

[1] Journal article - Nils Petersen, Didier Stricker . Cognitive Augmented Reality . 22 August 2015

[2] Jennifer Pearson, Simon Robinson, Matt Jones. BookMark: Appropriating Existing Infrastructure to Facilitate Scalable Indoor Navigation FIT Lab, Computer Science Department, Swansea University, Swansea, SA2 8PP, UK, 2 February 2017

[3] Journal article – Paul Milgram, Haruo Takemura, Akira Utsumi, Fumio Kishino . Augmented Reality: A class of displays on the reality-virtuality continuum .SPiE . 1994

[4] Journal article – Syed Waqar Jaffry, Huma Shoaib . A Survey of Augmented Reality. International Conference on Virtual and Augmented Reality. Singapore · January 2015

[5] ARSC: Augmented Reality Student Card An Augmented Reality Soultion for the Education field. El Sayed, N. A. M., Zayed, H. H., & Sharawy, M. I. (2010). ARSC: Augmented Reality Student Card. 2010 International Computer Engineering Conference (ICENCO). doi:10.1109/icenco.2010.5720437

[6] https://docparser.com/blog/what-is-ocr/

[7] Reality Technologies. Augmented Reality . https://www.realitytechnologies.com/augmented-reality/

[8] Google Developers. (2018). MLKit Documentation [Online].https://firebase.google.com/docs/ml-kit/

[9] https://play.google.com/store/apps/details?id=com.tayu.tau.pedometer&hl=en

[10] https://www.techopedia.com/definition/4776/augmented-reality-ar