Abstract— Indoor positioning systems help to localize objects and spaces inside a building, where Global Positioning System (GPS) don’t work effectively including cellular networks. But since the localization technologies incur great costs, a need for simulating localization indoors becomes essential. With this application we attempt to demonstrate, one such use case where we simulate indoor positioning system by rendering relevant 3D graphics and relevant location information on the mobile display. Localization is achieved by creating a location API simulator and using the orientation sensor in phone and tablet. Relevant graphical information based on the users context and selection, is rendered using OpenGLES on top of the live camera stream. Our attempt is to demonstrate through our application the strong untapped potential in augmented reality techniques in the context of indoor. The application will be a generic solution and can be easily composed to be use in any indoor space such as museums, hospitals, malls, etc.

Keywords— Augmented Reality; Android OS; OpenGLES; Rendering virtual objects; Occlusion Culling

I. INTRODUCTION

The Android application for Indoor Positioning using Augmented Reality is system that leverages augmented reality to render relevant information on the user interface. The application overlays relevant location information on the live camera stream based on user’s context. Our application has a wide range of functionalities based on the user’s preferences and it provides a rich user experience by enhancing interactivity with virtual objects on screen [9]. The application would provide a generic framework that can be easily tailored and applied to almost any indoor space. We simulate indoor positioning by developing a location API simulator because we do not use any tracking technologies like GPS, Wi-Fi, etc. We have hardcoded all the locations with which we have simulated the location API. We ultimately try to demonstrate how augmented reality can enhance the utility of a simulated indoor positioning system and we aim to achieve the following:-

• To leverage augmented reality for rendering relevant information on the user interface.
• To overlay location information on the live camera stream based on user’s context-a wide range of functionalities based on the user’s preferences.
• To provide a rich user experience by enhancing interactivity with virtual objects on screen.

II. REVIEW OF LITERATURE

Augmented Reality (AR), also known as Mixed Reality, aims to combine virtual and real scene together to achieve that virtual ones are belong to the real world [7]
We have surveyed the potential of augmented reality based applications in order to gain more insight of the features that could be expected of our application through the following papers and web documents:

2.1 Research articles

2.1.1 Survey of Wireless Indoor Positioning Techniques and Systems by Hui Liu, Darabi, H., Banerjee, P., Jing Liu[1]
It gave us an overview of some of the most popular technologies used in Indoor Positioning Systems -Global Positioning System (GPS), Wi-Fi, Bluetooth and Infrared.

2.1.2 Place lab: device positioning using radio beacons in the wild [5]
This paper described that Place Lab provides location, based on known positions of the access points which are provided by a database cached on the detecting device. Place Lab is entirely dependent on the availability of beacon locations, without which it cannot estimate anything about the current location which seemed to be a drawback.

2.1.3 Indoor positioning in Bluetooth networks using fingerprinting and lateration approach
In this paper [8], they use Bluetooth devices for indoor positioning and use signal based parameters such as received power level for position estimation. This paper presented an experimental relationship between the received power level and distance using the standard radio propagation model. Localization systems based on Bluetooth were explained but due to its short range we decided not to develop the system based on Bluetooth.

2.2 Web documents

2.2.1 Qualcomm Indoor Location [2]
Since Qualcomm’s indoor location technology (IZat) which is a chip-based platform that facilitates delivery of location-aware networks is proprietary, acquiring its licenses was difficult and hence not considered in the development of our project.

2.2.2 Google Indoor Maps [3]
Google has provided their Google Indoor Maps API for Android Smartphones and Tablets for location detection based on Indoor Environments.

2.2.3 InfSoft indoor navigation [6]
InfSoft makes use of multiple wireless technologies – Wi-Fi, Bluetooth and GPS to localize an object indoors. It further implements augmented reality to overlay the device’s camera view with relevant navigational information and is used as a reference.

III. DEVELOPMENT AND METHODOLOGY

3.1 Resources

The Eclipse IDE [12] provides an environment for development of android application on any operating system (Windows, Linux or Mac). Eclipse IDE is a freeware and thus helps in reducing the overall cost of the project. Hence, Eclipse is used. Eclipse Mars version has been used. Android SDK [12] (Software Development Kit) contains the APIs and compiler required for android application development. This is integrated with Eclipse IDE for development purpose. Android SDK API Level 19 has been used for the current project. We use OpenGL ES 2.0 [4] [13] to render graphics in Android. The Android application for Indoor Positioning using Augmented Reality has been developed under the Android operating systems using the Java JDK [12] (Java Development Kit) and it has all the APIs and compiler required for development and deployment of the application.
3.2 User Interface of RealARCampus and Site plan of campus

The application is meant for three types of users i.e. Student Mode, Staff mode, Visitor Mode. So the first activity is the registration or login activity where we can identify the user type and determine whether the user is authorized to be using the respective mode.

Once the user is logged in the user would then see a list of floors out of which he can select one of them. As one can see the information that the user needs to have to use the application is very little because of which its interface is intuitive.

Then after selecting the floor the user is then given two options i.e. P1 or P2. Now these points would be physically laid out throughout the college on the floor. The following figure shows the third floor plan of Shree L.R.Tiwari College of engineering where you can see the two points i.e. P1 and P2 are marked in red on it. The site plan is almost the same for all the floors. These points are used so that when the user stands on that point (For E.g. Say P1), then he knows exactly which button to select in order to get the augmented view of the relevant location information.
In order to make this application more useful to the students and the staff of the college, we have provided the option of uploading and downloading notices. The feature of uploading the notices is provided in the staff mode only. But the option of downloading the notices is provided in both student and staff mode. These features are not included in the visitor mode as this information doesn’t concern this type of user.

IV. PROPOSED SYSTEM ARCHITECTURE

Our application employs a basic client-server architecture where each client and server consists of the following modules and the flow of the modules is also shown below:

4.1 Mobile Client:

It consists of 5 modules:

4.1.1 Authentication module:

The authentication module prompts the user to login through the internet. If the user is not logged in then the user can register by entering an already provided user ID and password that would differentiate the types of the users i.e. Student, Staff or Visitor. This module interacts with the database application module by sending HTTP post requests for validating the username and password.
4.1.2 Image Capturing Module:

This is basically implementing the first layer i.e. the camera preview layer of our application. It takes the live video feed captured by the camera in real time and passes this input to the rendering module along with the spatial parameters like camera orientation.

4.1.3 Rendering Module:

Rendering Module is responsible for processing the returned virtual information and superposes that with the real image collected by camera and then outputs to the screen.

4.1.4 Location API simulator:

This module intends to simulate a location API and is consisting of all the locations relevant to every reference point set on every floor. These locations were identified, documented and stored in this module to simulate the location API’s like Google Maps API and so on. So we don’t need any tracking technologies like GPS, Wi-Fi, etc. as all the locations are found by us after a survey of the campus and manually finding the relevant position of an object in real world with respect to the reference point i.e. say P1.

4.1.5 Upload/Download module:

This module enables the user i.e. only the staff and student to download notices. But only the staff is given the privilege to upload important notices and no one else.
4.2 **Server:**

It consists of 2 modules:

**4.2.1 Database application module:**

The database application module helps the android client application to communicate with the remote MySQL [17] database. It has been implemented using PHP [16] which provides the necessary web service. It fetches the data from the database processing module and returns the JSON [14] file to either the authentication module or the upload or download module of the mobile client.

**4.2.2 Database processing module:**

This module retrieves data from the database created using MySQL and passes this data to the database application module. When the Mobile client sends a request, it gets response as a JSON file which it then parses to display the information received.

The different layers of our augmented reality application with respect to the rendering module are composed of:

A. **Camera Preview:** This is the first layer which will cause the image preview from the camera.

B. **OpenGLES Renderer:** Above the camera preview layer there is the OpenGLES renderer which is a transparent overlay and is developed using OpenGLES. Its background is made transparent so that you can still see the camera preview on the background.

C. **Android UI Overlay:** Above the OpenGLES layer is the default Android overlay, so that we could use the standard android buttons and other UI elements provided by the Android SDK.

The three major threads in the application:

D. **UI thread:** We have the UI thread which handles all the UI overlay above the OpenGLES overlay. Here the UI is presented to the user so that the user can interact with the application.

E. **Render thread:** We have the render thread which renders all the virtual objects. Here OpenGLES always draws 20 frames per second in the OpenGL virtual world and steps through all the virtual objects.

F. **Update thread:** We have the Update thread which updates all the virtual objects. This update mechanism is called every 20ms.

![Fig 4.2: The Three Layers of RealARCampus application](image1)

![Fig 4.3: The Three Layers of RealARCampus application](image2)

The key features of any AR application and its realization through our application are:

A. **Rendering virtual objects:**

In our application we are able to display AR content and the process employed for rendering virtual objects is represented in step form as follows:
Step 0: Begin
Step 1: Register Exception Handler
Step 2: Load device dependent
Step 3: Creating OpenGL overlay
Step 4: Initializing system values
Step 5: Creating OpenGL content
Step 6: Creating camera overlay
Step 7: Enabling User input
Step 8: Creating virtual world
Step 9: Creating GUI overlay settings
Step 10: Adding all overlays
Step 11: Entering full screen mode
Step 12: Done

B. Provide a perspective view and achieve occlusion culling

To give feelings of an augmented world to the user we need to implement Occlusion culling provides the user with a sense of perspective similar to the real world. Occlusion culling increases rendering performance simply by not rendering geometry that is outside the view frustum or hidden by objects closer to the camera [11]. A perspective view is created to cause objects further away to appear smaller. But to achieve this we should know the position of the user and also need to fix the position of the virtual objects.

So we have considered for our application that the user’s location is the reference point i.e. (0, 0, 0) in the virtual world and any object is added to this virtual world with respect to this reference point. Also as we mentioned that the user will be standing at a fixed point say P1, the reference point is also fixed. This enables us to provide relevant location information without the use of any tracking technologies like Wi-Fi, GPS, etc. To determine the position of a virtual object in the virtual world with respect to the reference point, we determine the direction towards which the camera is pointing i.e. whether is it pointing in the North, South, and East or West direction.

First our application determines the virtual position of virtual object by triangulation of positions. An auxiliary intersection point is needed between the reference point i.e. say P1 (latitude_1, longitude_1) and the position of the virtual object (latitude_2, longitude_2), as shown in Fig. 3.

We calculate the distance from the reference point to the auxiliary point (distance A), associated with a relative position in function of the longitude of the virtual position of the virtual object (longitude_2). [10] This is how our application indicates if the real world objects are in the “East” or “West” from our position and then we place the virtual objects to provide relevant information about the location. We then calculate the distance from the auxiliary point to the virtual position of the object (distance B), which is associated with a position in function of the latitude of the position of the virtual object (latitude_2), thereby providing the position of the object relative to “North” or “South” according to the situation. It should be kept in mind that to determine these relative positions the higher latitude involves that this point is further north of the “Equator” and if the point has higher longitude, this is located further east of the central meridian. By means of two
conditional sentences, the latitudes and longitudes of the reference point and virtual object location have been compared [10]:

<table>
<thead>
<tr>
<th>Reference_point_lat = North, if ( \text{Latitude}_2 &gt; \text{Latitude}_1 )</th>
<th>South, otherwise</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference_point_lon = East, if ( \text{Longitude}_2 &gt; \text{Longitude}_1 )</td>
<td>West, otherwise</td>
</tr>
</tbody>
</table>

These relative positions then determine what virtual object to render based on the direction where the camera is pointing and thereby achieves occlusion culling i.e. when the user turn the camera North, then all the virtual objects that were in the North direction would be rendered and not the others thereby improving the efficiency of our application. Now the possible directions are limited: north, south, east and west. But we also need to set a certain range of angles provided by the device compass for every possible direction, as seen below [10]:

<table>
<thead>
<tr>
<th>Location (Heading) = North, if ( \text{previousHead} &lt; 45.0 ) &amp; ( \text{previousHead} &gt;= 315 )</th>
<th>East, if ( \text{previousHead} &gt;= 45.0 ) &amp; ( \text{previousHead} &lt; 135 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>South, if ( \text{previousHead} &gt;= 135 ) &amp; ( \text{previousHead} &lt; 225 )</td>
<td>West, if ( \text{previousHead} &gt;= 225 ) &amp; ( \text{previousHead} &lt; 315 )</td>
</tr>
</tbody>
</table>

Once the application knows where the user is pointing, the application renders the appropriate virtual objects providing the relevant location information.

**V. RESULTS**

The features mentioned in the previous section can be observed in the screenshots taken from a laboratory where we were testing the application and have successfully rendered the virtual objects (For Example Animated Arrows) and have also displayed relative location information. We have provided the check for notice in the augmented view so that users view is not cluttered with irrelevant information in the beginning.

*Fig 5.1: Virtual objects rendered and displaying the relevant location information*

Here the perspective view which makes farther objects appears smaller than the objects that are closer to the users view. The objects not in the user view have been occluded as it can be seen in
figure 4.2. We have provided the check for notice in the augmented view so that users view is not cluttered with irrelevant information in the beginning and is shown in figure 4.3

![Fig 5.2: Perspective view achieved with occlusion culling](image1)

![Fig 5.3: Clutter free view for viewing the notices](image2)

We have also been able to successfully upload and download notices in the database and inform the user about it as well.

![Fig 5.4: Successful upload and download of Notice Images](image3)

**VI. CONCLUSION**

RealARCampus, the Android application for Indoor Positioning using Augmented Reality is a system that leverages augmented reality to render relevant information on the user interface. The application overlays points of interest on the live camera stream based on user’s context. Our
application has a wide range of functionalities based on the user’s preferences and it provides a rich user experience by enhancing interactivity with virtual objects on screen. The application would provide a generic framework that can be easily tailored and applied to almost any indoor space.

VII. FUTURE WORK

Future developments include the following:

- The application would be adapted to the Android tablets.
- We would include an indoor map view of the college for which we would use the JOSM [15] (Java OpenStreetMap Editor) tool to generate .osm files that would provide background imagery of the campus site (where site plan of college is uploaded by us) and locations of each point on the site thereby providing an indoor map view of our campus.

REFERENCES