Augmented Reality based Navigation

CS461 Project:
Aryan Agrawal (170101013)
Udbhav Chugh (170101081)
Mayank Baranwal (170101084)

Augmented Reality (AR)

Definition of AR

- Interactive experience of a real-world environment where objects enhanced by computer generated images
- Can be across multiple senses: visual, auditory, somatosensory, haptic, and olfactory → Our focus towards visual
- Famous example: Pokémon Go

Challenges with AR

- Computationally expensive
- Need for multiple sensors to create immersive experience

Brief Timeline of Popular Navigation Apps

Google Maps features:

- Online version of maps
- Introduction of satellite images to public domain
- Algorithms to chart optimal routes based on time

Disadvantages:

- No live street classification; difficult to differentiate streets

Google Street View (Quasi - AR)

- Multiple photos taken by 360 degree cameras
- Images stitched together that provide basic navigation

Disadvantages:

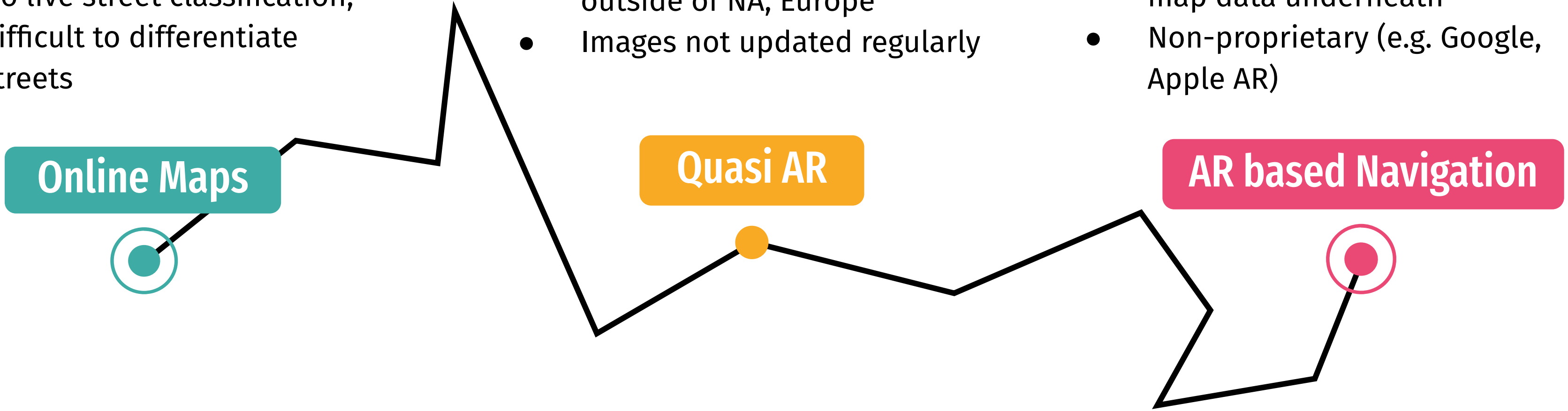
- Google Street View banned outside of NA, Europe
- Images not updated regularly

Integrating AR with Maps

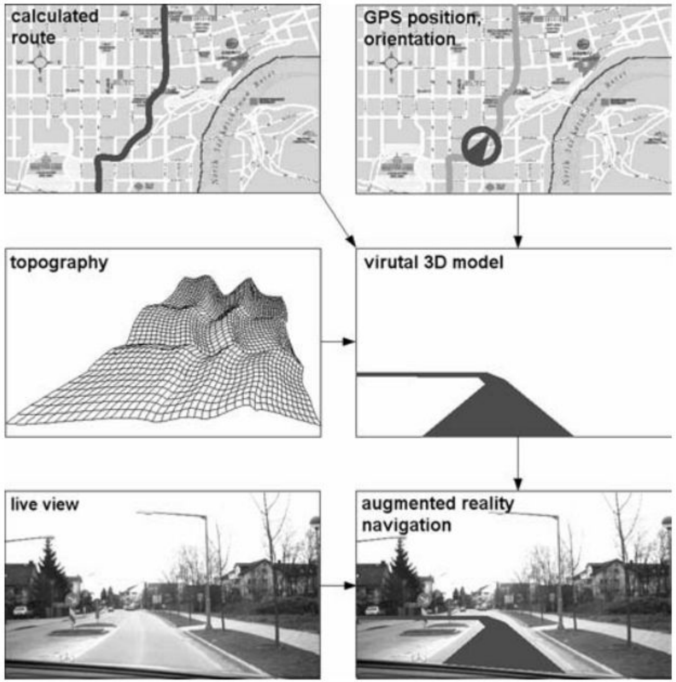
- Live user's camera feed
- Navigation aids overlayed on top

Disadvantages:

- Only testing versions available
- Requires heavily annotated map data underneath
- Non-proprietary (e.g. Google, Apple AR)



Recent AR based Navigation Works



Augmented reality navigation systems
(Wolfgang Narzt, Gustav Pomberger, Alois Ferscha)

1. Work out route with online maps
2. Have a pre-processed topography map of route
3. Develop virtual 3D model
4. Project 3D route on top on map

Disadvantages

- Targeted towards automobile based navigation
- Requirement of existing topography map → Inapplicable to remote locations
- Cannot run on a low-end smartphone

Recent AR based Navigation Works (Contd)

Based	Sensor	Advantage	Weakness
Sensor-based	GPS Gyro Accelerometer	Easy to develop	uncomfortable in indoor movement
Vision-based	Computer Vision Open GL Marker based Marker-less based	High immersion level	self technology development
Hybrid	Vision + Sensor	If tracking is lost, it will be compensated through the sensor. The weight of the vision is compensated by the sensor	Hard to develop, Unavailability of contents

Augmented Reality Navigation System on Android

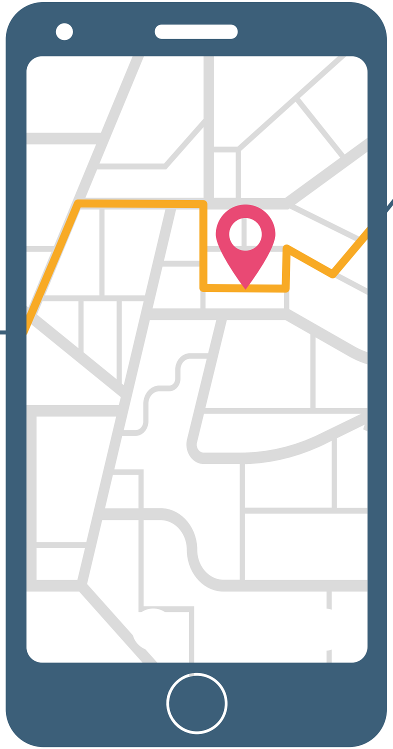
(Chee Oh Chung, Yilun He, Hoe Kyung Jung)

1. Utilize Google Maps API to plot route
2. Displace waypoints at large distances with hybrid (sensor + vision) based approach to plot markers

Disadvantages

- Waypoints spaced too far apart for street level differentiation
- Markers are static → Do not take into account that they may be out of line-of-sight

End-to-End Solution



Input

Geocoordinates of destination are provided.

Creating AR World

Finding path from source to destination and create location markers.

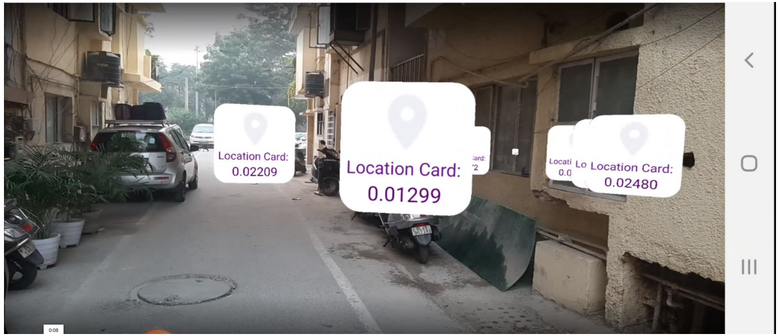
Placing markers

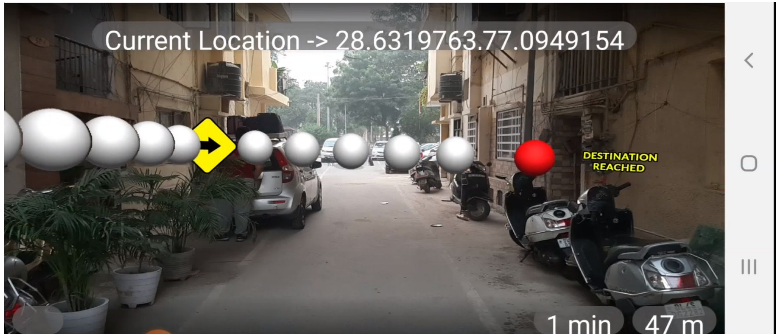
Placing equidistant markers based on haversine distance between two checkpoints.

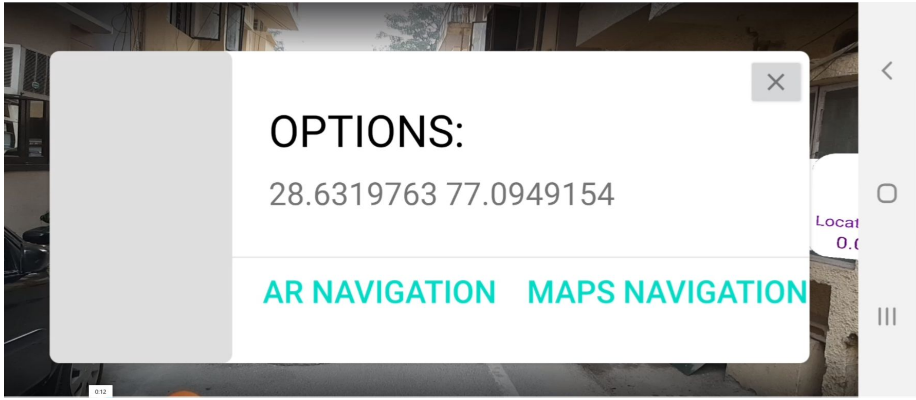
Real Time Rendering

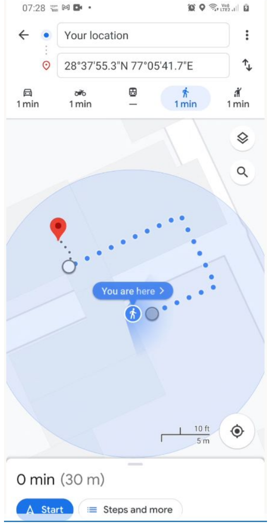
Identifying geo coordinates by GPS, recognizing appropriate markers from AR world and accurately position them in video stream.

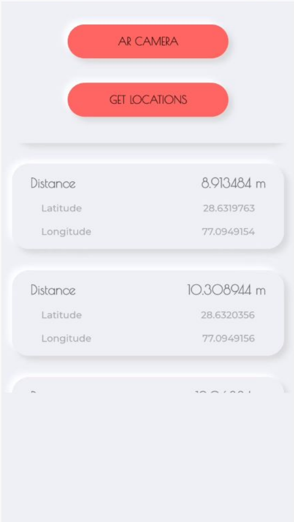
Screenshots











Library Utilised: BeyondAR

About

Open source framework for creating AR applications for Android devices.

Geolocalization

Uses GPS to find the geo coordinates and insert AR information annotations using appropriate geo coordinates and altitude.

AR object Rendering

Uses OpenGL to create the graphical object with required specification.

Sensors

BeyondAR uses Accelerometer and Magnetic field sensors to provide the AR experience.

BeyondAR (Cont.)

BeyondAR Features

- Create the AR world with current geo coordinates of user.
- Convert the geobjects to gl points representation relative to geo-coordinates of the AR world.
- Works by taking snapshot of view per frame, placing the gl objects and returning the rendered frame.
- Sensitive to the smartphone/User motion.

Advantages:

- Computationally Inexpensive.
- Quick and efficient to run unlike other AR engines

Our Approach

AR Rendering

Based on the final coordinates of the destination, we will render objects in augmented reality (with aid of BeyondAR), which will act as (guiding points) for the user to navigate

Marker Positioning

Our app will place the AR guiding points adjusting to the changes in the speed and direction of the smartphone using Haversine distance.

Integration with Maps

Existing Maps application will be integrated to find the path, which we will then use to display the AR-based guiding points along the path for easy navigation.

Special Features

Different Marker Features

Markers of different colours and sizes to depict checkpoints, changing direction (going left or right), etc. and reflect the user's perspective.

Controlling Sparsity of Markers

Allow the user to set how close they want these guiding points to be depending on the surrounding lighting and weather conditions.

Haversine Distance and Heading Angle

Haversine Algorithm

- Calculates the shortest distance between two points on a sphere using their latitude and longitude values. Haversine is expressed as:
- haversine of the central angle (which is d/r) is calculated by the following formula:
- Solving it gives the following value of distance:

$$haversine(\theta) = sin^2\left(\frac{\theta}{2}\right)$$

$$\left(\frac{d}{r}\right) = haversine(\Phi_2 - \Phi_1) + cos(\Phi_1)cos(\Phi_2)haversine(\lambda_2 - \lambda_1)$$

$$d = rhav^{-1}(h) = 2rsin^{-1}(\sqrt{h})$$

$$d = 2rsin^{-1}\left(\sqrt{sin^2\left(\frac{\Phi_2-\Phi_1}{2}\right) + cos(\Phi_1)cos(\Phi_2)sin^2\left(\frac{\lambda_2-\lambda_1}{2}\right)}\right)$$

Heading or Bearing Angle

- A heading is the angle of a direction from a fixed reference point(true north).
- Used to compute direction to traverse given two coordinates on Earth.
- Using the bearing angle, we can calculate the intermediate geo coordinates (taking marker sparsity into account)

$$\theta = \text{atan2}(\sin\Delta\lambda \cdot \cos\varphi_2 , \cos\varphi_1 \cdot \sin\varphi_2 - \sin\varphi_1 \cdot \cos\varphi_2 \cdot \cos\Delta\lambda)$$

φ_1, λ_1 is the start point, φ_2, λ_2 the end point ($\Delta\lambda$ is the difference in longitude)

- latitude of second point = la2 = asin(sin la1 * cos Ad + cos la1 * sin Ad * cos θ), and
- longitude of second point = lo2 = lo1 + atan2(sin θ * sin Ad * cos la1 , cos Ad – sin la1 * sin la2)

Demo and Code (Demo Link)

Non AR Features

Integrated Google Maps to navigate to destination without the use of AR

AR Rendering and Navigation

Rendering destination locations in AR space and navigating to chosen location using AR markers.

Controlling Sparsity of Markers

Allow the user to set the proximity of AR markers during navigation

Difficulties Encountered

Seamless Integration

Needed to integrate Google Maps API, Haversine distance calculation and BeyondAR

Computational Cost

Choice of algorithm must work on low-end smartphones and only require GPS, camera and Internet connection

Android Studio

Relatively new environment and framework → API support and structure different to standard applications

Limitations and Future Work

Occlusion
Use a periodic background thread to continuously detect location and not display occluded markers.

Offline Navigation
Application requires active Internet connection

Marker Aesthetics
Markers' lighting direction may not match sunlight direction → Requires additional hardware to add

Haptic Feedback
Alternate means to provide user navigational feedback (to turn and ensure correct path is followed)

Thank You!