K. L. V. R. Saraswathi, et. al. International Journal of Engineering Research and Applications
www.ijera.com
ISSN: 2248-9622, Vol. 13, Issue 10, October 2023, pp 08-13

RESEARCH ARTICLE OPEN ACCESS

ARCore Geospatial API

K. L. V. R. Saraswathi*, G. Laakshini**, K.Shivani***, Dr Shaik Subhani****
**(Department of Information Technology, Sreenidhi Institute of Science and Technology, Ghatkesar*
Email: ramyakummari786@gmail.com)
***(Department of Information Technology, Sreenidhi Institute of Science and Technology, Ghatkesar*
Email: gaddameedhilaakshini@gmail.com)
****(Department of Information Technology, Sreenidhi Institute of Science and Technology, Ghatkesar*
Email: shivanikomma2020@gmail.com)
*****(Department of Information Technology, Sreenidhi Institute of Science and Technology, Ghatkesar*
Email: shaiksubhani@sreenidhi.edu.in)

ABSTRACT
This project focuses on creating an augmented reality (AR) application using the latest ARCore Geospatial API for Android devices, with the objective of seamlessly integrating virtual objects into the real world based on geospatial data. Key components encompass managing the AR session, rendering realistic visuals, and designing user-friendly controls, all facilitated by the Geospatial API for precise positioning and alignment with real-world locations. Through this app, users can explore their surroundings, accurately place virtual markers, and interact with geospatial information overlaid on a map, providing an immersive AR experience enriched with geospatial context. Considerations are made for device compatibility, performance optimization, and seamless integration with external geolocation services, while future enhancements may include features such as geospatial data visualization and geolocation-driven interactions. In summary, this project serves as a foundation for developing geospatially aware AR applications, offering users an immersive and geospatially accurate AR experience.

Keywords - ARCore Geospatial API, Augmented Reality (AR), Geospatial Integration, Immersive AR Experience, Precise Positioning

---

Date of Submission: 22-09-2023 Date of acceptance: 05-10-2023

I. INTRODUCTION

The ARCore Geospatial API, developed by Google, represents a groundbreaking advancement in the realm of augmented reality (AR) applications. Its primary mission is to empower developers to craft immersive AR experiences that seamlessly intertwine digital content with the physical world. This innovative tool leverages geospatial information, including GPS coordinates and mapping data, to anchor virtual objects and annotations to precise locations on Earth's surface. In doing so, it overcomes the traditional limitations of AR, which were primarily confined to indoor environments or marker-based tracking systems. By offering accurate positioning and alignment of virtual objects in outdoor settings, the ARCore Geospatial API opens up a wealth of possibilities for location-based AR experiences.

The scope of this API encompasses a wide range of features, including outdoor AR experiences, geospatial annotation, real-time collaboration, geolocation integration, environmental understanding, and cross-platform compatibility. This comprehensive approach empowers developers to create AR applications that not only place virtual content seamlessly in outdoor environments but also support real-time collaboration and integrate seamlessly with maps and geolocation services.

The proposed system builds upon the ARCore Geospatial API to create an AR app that enhances users' real-world experiences. It allows for the overlay of virtual objects and annotations in outdoor settings, providing geospatially-aware features such as geospatial annotation, map integration, real-time rendering, AR session lifecycle management, interactive capabilities, and the potential for real-time collaboration. This system offers an engaging and immersive way for users to explore and interact with their outdoor surroundings, making it a versatile platform for outdoor navigation, tourism, gaming, education, and more.

The merits of this system are significant. It enhances users' real-world experiences by seamlessly blending digital and physical elements. Users can place geospatial markers at specific locations to provide context or information, all while interacting with the virtual content through real-time

rendering. The AR session lifecycle management ensures smooth operation, and visually appealing visualizations enhance realism. This system promotes exploration and learning, encouraging users to discover and engage with their outdoor environment. Furthermore, its modularized code structure allows for customization and future enhancements, making it adaptable to various use cases and ensuring its longevity in the ever-evolving world of AR applications.

II. LITERATURE REVIEW

The evolution of Augmented Reality (AR) has deep roots in computer vision and computer graphics, dating back to the 1960s. Early AR applications explored overlaying digital information onto the real world through head-mounted displays and marker-based systems.

The introduction of ARCore and ARKit transformed mobile AR by providing frameworks for indoor and marker-based AR experiences. However, these platforms set the stage for the ARCore Geospatial API's emergence, which extends AR capabilities into geospatially-aware outdoor environments.

Geospatial data, including GPS coordinates and mapping information, plays a crucial role in anchoring virtual objects to specific real-world locations. The integration of geospatial data and AR technology opened the door to outdoor navigation and location-based AR. Early outdoor AR applications, such as tourism apps offering historical information at landmarks or educational apps for geological exploration, demonstrated the potential of geospatially-aware AR.

Challenges like device compatibility and performance optimization existed, but the industry anticipated specialized geospatial AR tools to realize outdoor AR's full potential.

The ARCore Geospatial API builds upon a history of AR development, incorporating geospatial data for precise outdoor positioning. This evolution reflects a growing demand for geospatially-aware AR applications, marking a significant milestone in AR's convergence with geospatial data.

III. METHODOLOGY

III.1. Technology Stack

III.1.1. ARCore: The application relies heavily on Google's ARCore platform. ARCore provides the foundation for augmented reality experiences, offering features such as motion tracking, environmental understanding, and light estimation.

III.1.2. OpenGLES: For rendering 3D graphics, utilizes OpenGL ES (Embedded Systems), which is a cross-platform API for mobile and embedded devices. This technology allows for efficient rendering of virtual objects within the real-world camera feed.

III.1.3. Google Maps API: The integration with the Google Maps API enables the app to display maps, geolocation data, and interact with Google Maps services. It leverages Google Maps' vast database of geographic information.

III.1.4. Android Studio: The development environment is Android Studio, the official integrated development environment (IDE) for Android app development. It provides tools for coding, debugging, and testing Android applications.

III.2. Algorithms and Techniques

3.2.1. Geospatial Tracking: The app leverages multiple sensor inputs, including GPS and accelerometer data, to perform geospatial tracking. These sensors provide information about the device's movement and location. ARCore's geospatial mode further enhances tracking accuracy by fusing sensor data with visual data from the device's camera. The combination of sensor fusion and ARCore's tracking capabilities enables the app to determine the precise geographical location (latitude and longitude) and altitude of the device.

3.2.2. Pose Estimation: Pose estimation algorithms are crucial for determining the device's orientation and position in 3D space. These algorithms work by analyzing the visual data from the device's camera and identifying key features in the environment. By tracking the movement of these features over time, the app can accurately estimate the device's pose (position and orientation) relative to the real world. This information is essential for aligning virtual objects with the physical environment and ensuring they appear anchored in the correct locations.

3.2.3. Map Rendering: The rendering of maps on the device's screen involves a series of sophisticated algorithms:

- Map Projection: The app uses map projection algorithms to convert geographic coordinates (latitude and longitude) into screen coordinates. This allows the app to accurately position and render map elements on the screen based on the user's location.
- Tile Fetching: To display detailed maps, the app fetches map data in the form of tiles from online map services. Tile fetching algorithms manage the retrieval and caching of these tiles efficiently.
- Rendering Engine: The app employs a rendering engine to draw map elements, including roads, landmarks, and geographical features. This engine handles the rendering of map layers and ensures a smooth and responsive user

experience, even during map interactions like panning and zooming.

3.2.4. 3D Object Rendering: To render 3D objects in the AR view, the app utilizes a combination of technologies and algorithms:

- Shaders: Shaders are used to define how virtual objects are lit, textured, and shaded. They determine the appearance of 3D objects in the AR scene, taking into account lighting conditions and material properties.
- Mesh Rendering: Mesh rendering techniques construct the 3D geometry of virtual objects. This involves defining the vertices, faces, and textures that make up a 3D model. The app uses pre-defined 3D models or dynamically generated meshes to represent AR objects.

3.2.5. Interaction Algorithms: The app implements algorithms to facilitate user interaction with AR objects and the map:

- Gesture Recognition: Gesture recognition algorithms identify user gestures, such as tapping on the screen to place AR anchors. These algorithms interpret user input and trigger corresponding actions, enabling users to interact with virtual content seamlessly.
- Touch Input Handling: Algorithms for touch input handling process user touches and gestures on the screen. This includes detecting swipe, pinch-to-zoom, and drag interactions, ensuring a responsive and intuitive user interface.

Overall, the app's methodology combines sensor data, ARCore's tracking capabilities, advanced rendering techniques, and user interaction algorithms to create an immersive geospatial AR experience. It seamlessly integrates virtual content with the real world while providing precise geospatial information and interactive map functionality. The result is an engaging and informative augmented reality application.

3.3. Implementation Details

3.3.1. ARCore Session Management: The ARCoreSessionLifecycleHelper is a crucial component responsible for managing the lifecycle of the ARCore session. It performs the following tasks:

- Initialization: It initializes the ARCore session, ensuring that the session is ready for augmented reality experiences.
- Configuration Handling: The helper class manages the configuration of the ARCore session. This includes setting options specific to geospatial tracking, such as enabling Geospatial Mode. Configuring the session correctly is essential for precise AR tracking and rendering.
- Exception Handling: The ARCoreSessionLifecycleHelper gracefully handles exceptions that may occur during session initialization or resume. It provides informative error messages to users, helping them understand and resolve issues. This is crucial for a user-friendly experience.

3.3.2. AR Rendering and Tracking: The app relies on the HelloGeoRenderer class to render AR content within the device's camera feed. This class is responsible for:

- AR Object Rendering: It renders AR objects, such as geospatial markers, in the camera view. This involves using shaders and mesh rendering techniques to create realistic virtual objects.
- Tracking Integration: HelloGeoRenderer incorporates ARCore's tracking data to position virtual objects accurately in the real world. This integration ensures that AR content aligns correctly with the physical environment.

3.3.3. Geospatial Mode Configuration: The app leverages Geospatial Mode within ARCore. This mode enhances geospatial awareness by allowing for precise placement of AR content based on geographical coordinates. Configuring Geospatial Mode is essential for location-based augmented reality experiences.

3.3.4. Permission Handling: GeoPermissionsHelper is responsible for requesting and managing permissions for essential device resources:

- Camera Permission: The app requests camera permissions to access the device's camera for AR experiences.
- Location Permission: Location permissions are necessary to access GPS and other location services, enabling geospatial tracking and map functionalities.
- Proper permission handling ensures that the app can access the required resources without interruptions.

3.3.5. Google Maps Integration: The app integrates with the Google Maps API to provide map-related features:

- API Key Configuration: It includes the setup of API keys for both ARCore and Google Maps, enabling access to these services.
- Map Display: The app displays maps on the device's screen, allowing users to view geographical information.
- Map Position Update: Based on the device's geospatial pose, the app updates the map's position, ensuring that the displayed map aligns with the real-world location.
- Interactive Map Features: Users can interact with the map, including placing anchors and exploring geographical data. This interactive map functionality enhances the user experience.

3.3.6. Anchor Placement: Users can place anchors on the map by tapping specific locations. The app calculates the altitude relative to the device's geospatial pose, ensuring that the anchors are positioned accurately. These anchors serve as reference points for rendering AR content.

3.3.7. Error Handling: The app implements robust error handling to address potential issues:

- Camera Unavailability: If the device's camera is not available, the app handles this gracefully and provides a user-friendly error message.
- Outdated ARCore Versions: If ARCore is outdated, the app guides users on how to update it, ensuring compatibility and optimal performance.

By meticulously combining these technologies, algorithms, and detailed implementation strategies, The ARCore geospatial app provides users with a rich and immersive augmented reality experience, seamlessly merging digital content with the physical world while offering robust performance and precise geospatial accuracy.

IV. RESULTS AND ANALYSIS

IV.1. Features

IV.1.1. 3D Object Rendering: The app renders a 3D object in the real world using ARCore. This 3D object is accurately placed in the user's physical environment, aligning with real-world objects and surfaces.

IV.1.2. Interactive 3D Object: Users can interact with the 3D object. It's possible to change the position or orientation of the 3D object, providing an engaging and interactive AR experience.

IV.1.3. Map Display: The app features an integrated map view that displays a map of the user's surroundings. This map allows users to explore their environment and interact with geographic features.

IV.1.4. User Location Tracking: The app tracks the user's current location in real-time using GPS or other location services. The user's location is displayed on the map as a marker or symbol, indicating their precise position.

IV.1.5. AR Anchors on Map: When users interact with the AR component by placing an anchor or marker in the real world, this action is reflected on the map. An anchor symbol on the map corresponds to the physical location where the user placed the AR anchor.

IV.1.6. Geospatial Data: The app provides additional geospatial information, including latitude, longitude, and altitude. This data is displayed on the map, allowing users to see their exact geographic coordinates.

IV.2. Output

When a user launches the app and grants the necessary permissions, they can expect the following output:

4.2.1. The app's camera view will activate, and ARCore will initialize, allowing the user to view their physical surroundings through the device's camera.

4.2.2. A 3D object will be rendered in the real world, accurately aligned with the user's environment. The user can interact with this 3D object, such as moving it to different locations.

4.2.3. The app's map view will display a map of the user's current surroundings. The user's precise location (latitude, longitude) will be marked on the map with a symbol, indicating where they are in the real world.(fig 1.1)

4.2.4. If the user interacts with the AR component by tapping on the screen to place an anchor, an anchor symbol will appear on the map at the corresponding location.(fig 1.2)

4.2.5. Additionally, the app will display the user's current altitude, providing a comprehensive set of geospatial information.

Overall, the output is an immersive AR experience that seamlessly combines the real world with virtual 3D objects, all while providing users with detailed geospatial data and an interactive map interface. This combination of features offers a rich and engaging augmented reality experience with real-world context.

fig 1.1

fig 1.2

V. CONCLUSION AND FUTURESCOPE

V.1. Conclusion

The geospatial app based on the provided code allows users to experience augmented reality (AR) using the ARCore framework. The app leverages the new ARCore Geospatial API to enable the placement of virtual objects in real-world locations based on geospatial data.

The app's core features include rendering virtual objects, such as geospatial markers, in the AR scene, integrating with a map for user interaction and anchor placement, and providing a seamless AR experience through the use of shaders, textures, and meshes.

The code is organized and modularized, with components such as the HelloGeoRenderer and SampleRender classes responsible for rendering and managing the ARCore session. The code demonstrates the utilization of ARCore, Android platform APIs, and external resources for AR functionality. The app has certain hardware and software requirements, including ARCore compatibility, a supported Android version, and sufficient device specifications. While implementing the code, certain limitations and challenges may arise, and performance or usability issues should be considered.

In conclusion, the geospatial app built using the provided code offers an engaging AR experience by combining real-world geospatial data with virtual objects. It showcases the capabilities of the ARCore Geospatial API and provides a foundation for further enhancements and customizations in AR applications.

V.2. Future scope

5.2.1. Enhanced Geospatial Interactions: Support geofencing geolocation-based notifications, and real-time collaboration.

5.2.2. Additional AR Features: Explore markerless object recognition, surface detection, and advanced visual effects.

5.2.3. Integration with External APIs: Integrate mapping services, weather data providers, and geospatial databases.

5.2.4. Customization and Personalization: Allow users to customize virtual objects and share their own markers.

5.2.5. Performance Optimization: Optimize rendering, reduce battery consumption, and improve tracking accuracy.

5.2.6. Integration with Other Technologies: Explore integration with machine learning, computer vision, and IoT devices.

5.2.7. Cross-Platform Support: Adapt the app for iOS and other platforms for wider accessibility.

The future scope of the geospatial app includes expanding geospatial interactions, incorporating advanced AR features, integrating with external APIs, offering customization options, optimizing performance, exploring new technologies, and supporting multiple platforms.

REFERENCES

[1] https://developers.google.com/ar
[2] https://developer.android.com/docs
[3] https://developers.google.com/maps/documentation
[4] https://docs.opencv.org/
[5] https://developers.google.com/ar
[6] https://github.com/google-ar/arcore-android-sdk
[7] https://developers.google.com/ar/develop/java/geospatial
[8] https://developers.google.com/ar/develop/unity
[9] https://developers.google.com/ar/develop/c
[10] https://github.com/google-ar/arcore-unity-sdk
[11] https://developers.google.com/ar/develop/java/augmented-images
[12] https://developers.google.com/ar/develop/java/quickstart
[13] https://github.com/google-ar/arcore-unity-extensions
[14] https://developers.google.com/ar/develop/java/augmented-reality

[15] https://developers.google.com/ar/develop/java/augmented-faces
[16] https://developers.google.com/ar/develop/java/augmented-image
[17] https://github.com/google-ar/arcore-android-sdk/tree/main/samples
[18] https://developers.google.com/ar/develop/java/augmented-reality/cloud- anchors/overview-android
[19] https://developers.google.com/ar/develop/java/sceneform
[20] https://developers.google.com/ar/develop/java/gestures
[21] https://developers.google.com/ar/develop/java/scene-viewer
[22] https://developers.google.com/ar/develop/java/occlusion
[23] https://developers.google.com/ar/develop/java/scene-viewer/android-quickstart
[24] https://developers.google.com/ar/develop/java/image-tracking