PROJETO ARL: UMA BIBLIOTECA PARA COMPARTILHAMENTO DE OBJETOS UTILIZANDO REALIDADE AUMENTADA

Aluno(a): Maicon Santos da Silva

Orientador: Dalton Solano dos Reis

FURB
UNIVERSIDADE DE BLUMENAU

Roteiro

- Introdução
- Objetivos
- Fundamentação teórica
- Trabalhos correlatos
- Requisitos funcionais
- Requisitos não funcionais
- Especificação
- Implementação
- Análise dos resultados
- Conclusões e sugestões
- Apresentação prática

Introdução

- A importação da computação gráfica;
- A popularização da realidade aumentada;
- Aplicação da realidade aumentada;
- O uso da realidade aumentada por desenvolvedores que não possuem conhecimento/vivência com RA.

Objetivos

- O objetivo geral deste trabalho é desenvolver uma biblioteca para a criação de aplicativos móveis que permitam o compartilhamento de objetos 3D entre os usuários;
- Objetivos específicos:
  - disponibilizar um processo para importar objetos 3D modelados por programas de terceiros para dentro de um repositório no dispositivo móvel;
  - permitir posicionar os objetos importados usando a câmera e o sistema de GPS;
  - desenvolver uma rotina que analisa coordenadas do GPS e procura por objetos posicionados nas proximidades;
  - permitir visualizar, usando realidade aumentada, os objetos localizados nas proximidades.

Fundamentação Teórica

Realidade aumentada

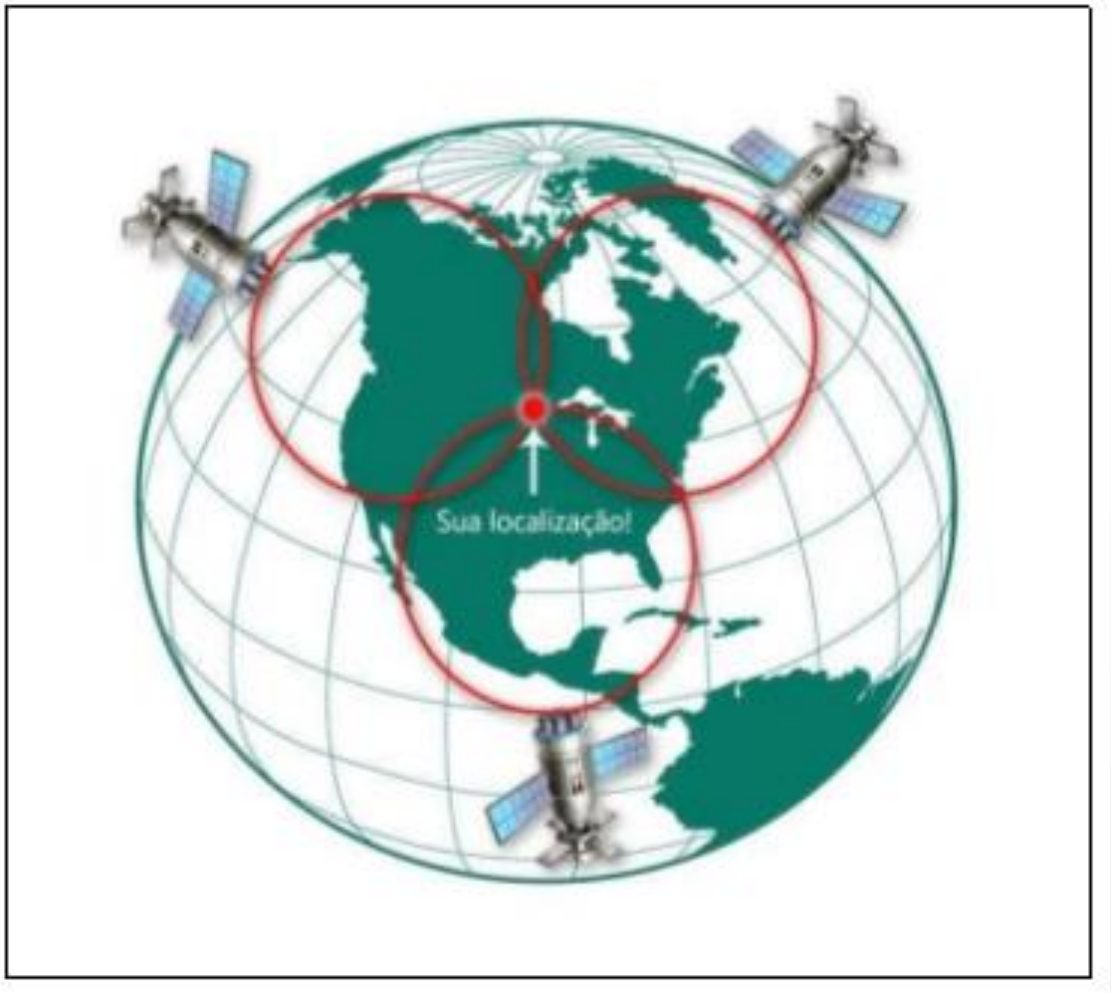
- A realidade aumentada precisa seguir três propriedades: combinar objetos reais e virtuais num ambiente real; rodar interativamente, e em tempo real; e registrar objetos reais e virtuais uns com os outros (AZUMA, 2001).

Fundamentação Teórica

Sistema de localização global (GPS)

- Funciona com um conjunto de três satélites;
- Sinais de rádio e cálculo da distância através de triangulação.

Fundamentação Teórica



Fundamentação Teórica

Object File Format (wavefront obj)

- Os formato consiste em linhas, onde cada uma possui uma chave e vários valores;
- Armazenamento de cores.

Chave	Descrição
#	Comentário
v	Vértice
l	Linha
f	Superfície
vt	Coordenada de textura
vn	Normal
g	Grupo
...	...

Trabalhos Correlatos

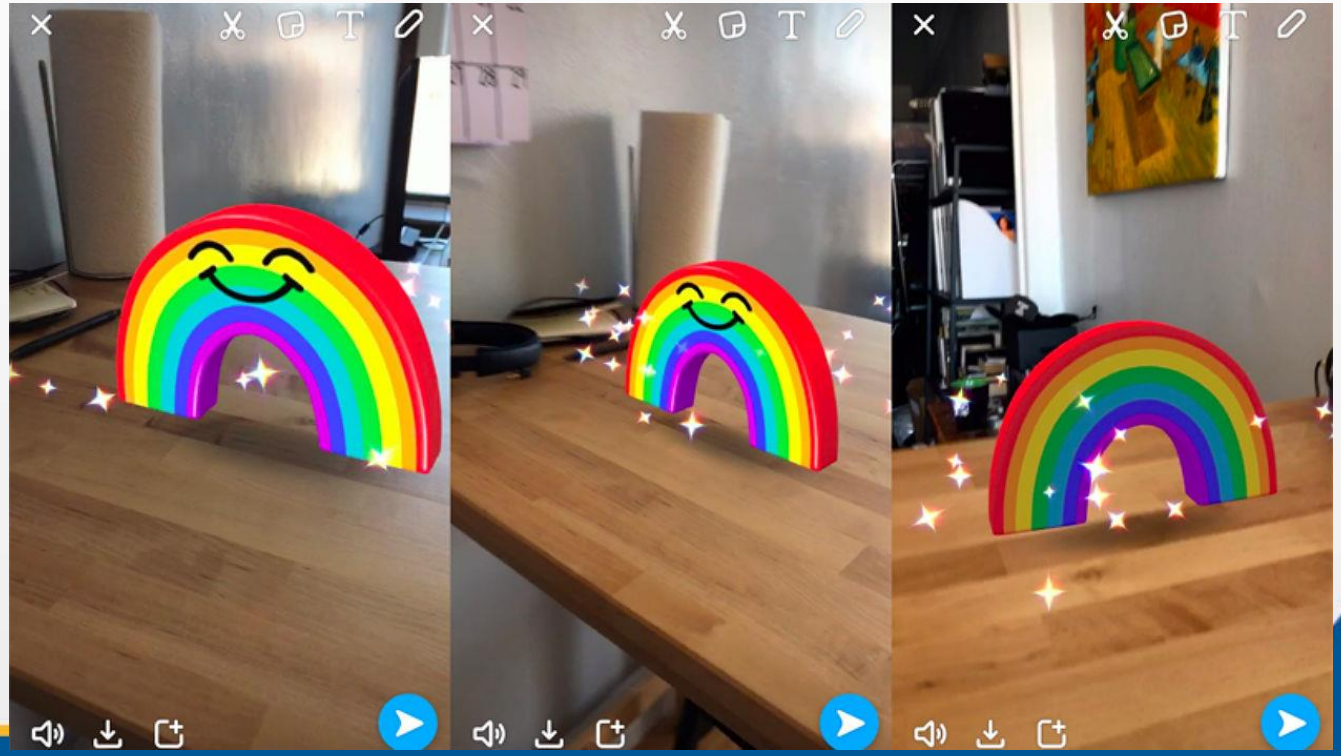
- Augment:
  - Permitir que o usuário importe modelos 3D personalizados e o posicione em cena para visualização com realidade aumentada.

Trabalhos Correlatos

- Pokémon GO:
  - Localizar Pokémons espalhados pelo mapa utilizando GPS para identificar e então visualizá-los utilizando realidade aumentada.

Trabalhos Correlatos

- Snapchat:
  - Tirar fotos utilizando diversos filtros que utilizam realidade aumentada através de rastreamento de marcadores (rosto do usuário).



Requisitos funcionais

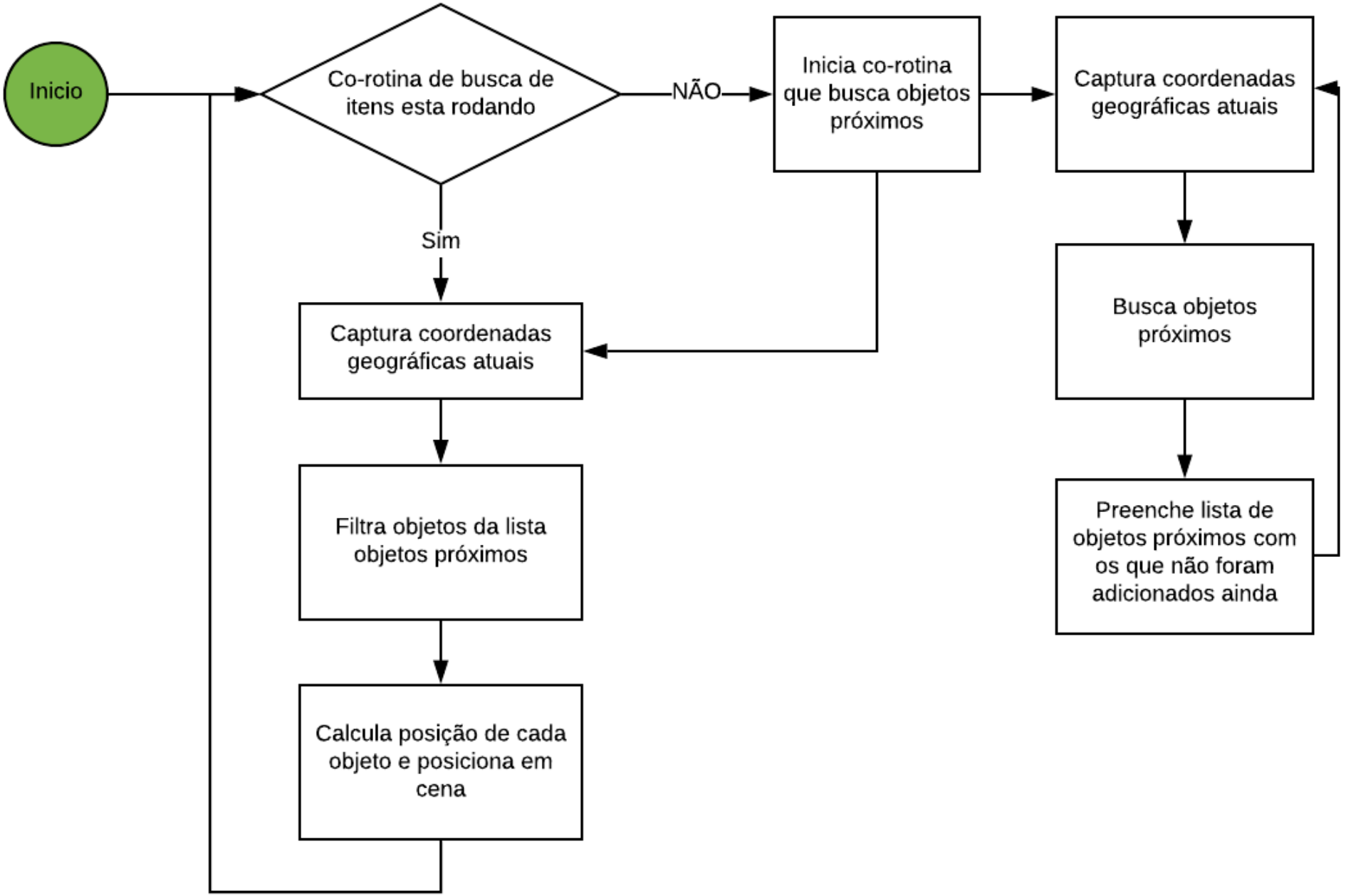
- Possuir um repositório na nuvem e uma interface para cadastrar objetos e suas devidas coordenadas;
- Possibilitar vincular objetos do repositório a coordenadas no mapa;
- Permitir detectar objetos cadastrados nas proximidades;
- Permitir que o usuário posso ver os objetos nas proximidades através da câmera usando realidade aumentada.

Requisitos não funcionais

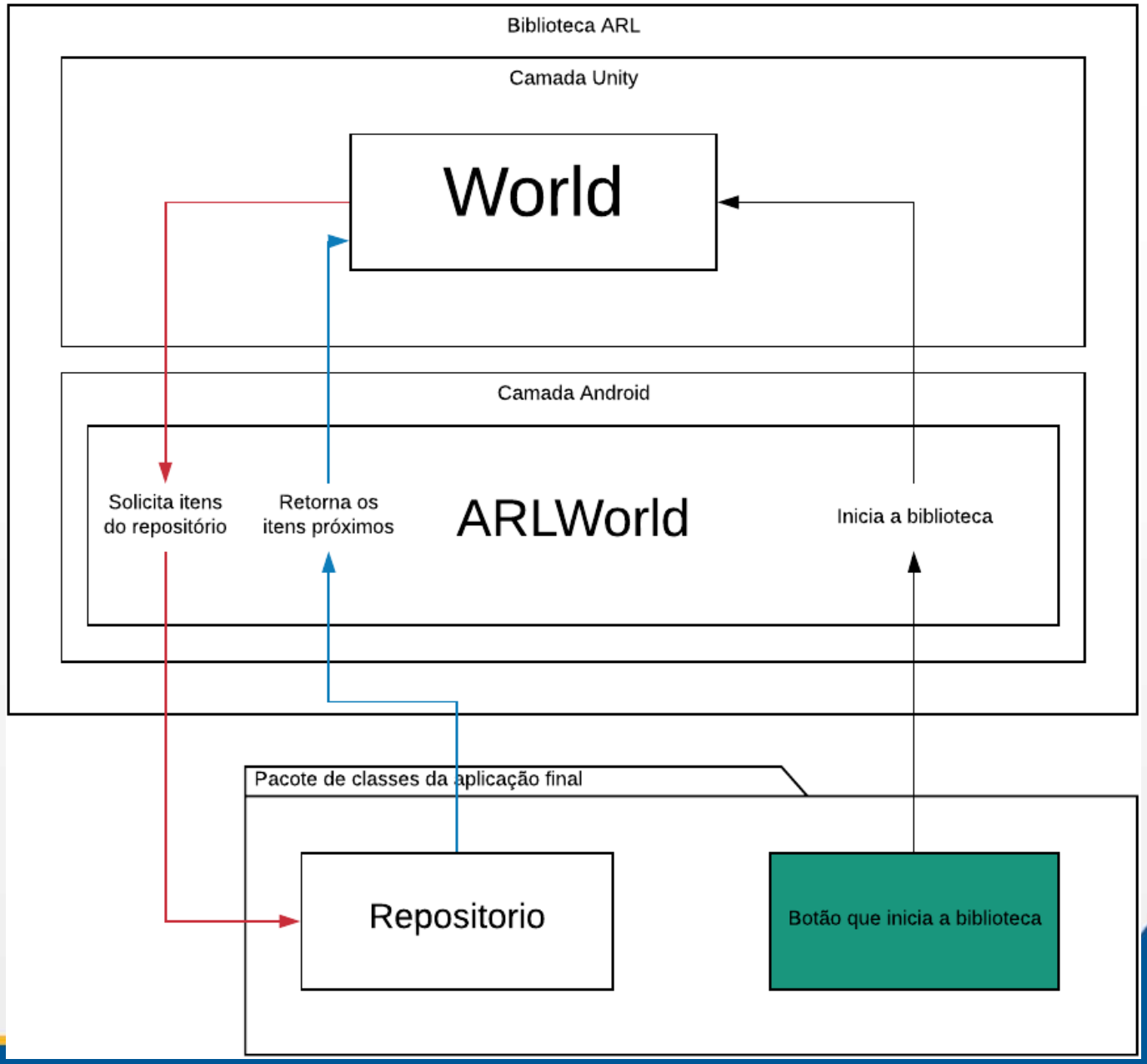
- Ser desenvolvido como uma biblioteca;
- Disponibilizar um projeto demonstração junto a documentação;
- Ser desenvolvido para dispositivos móveis e inicialmente para rodar na plataforma Android.

Especificação

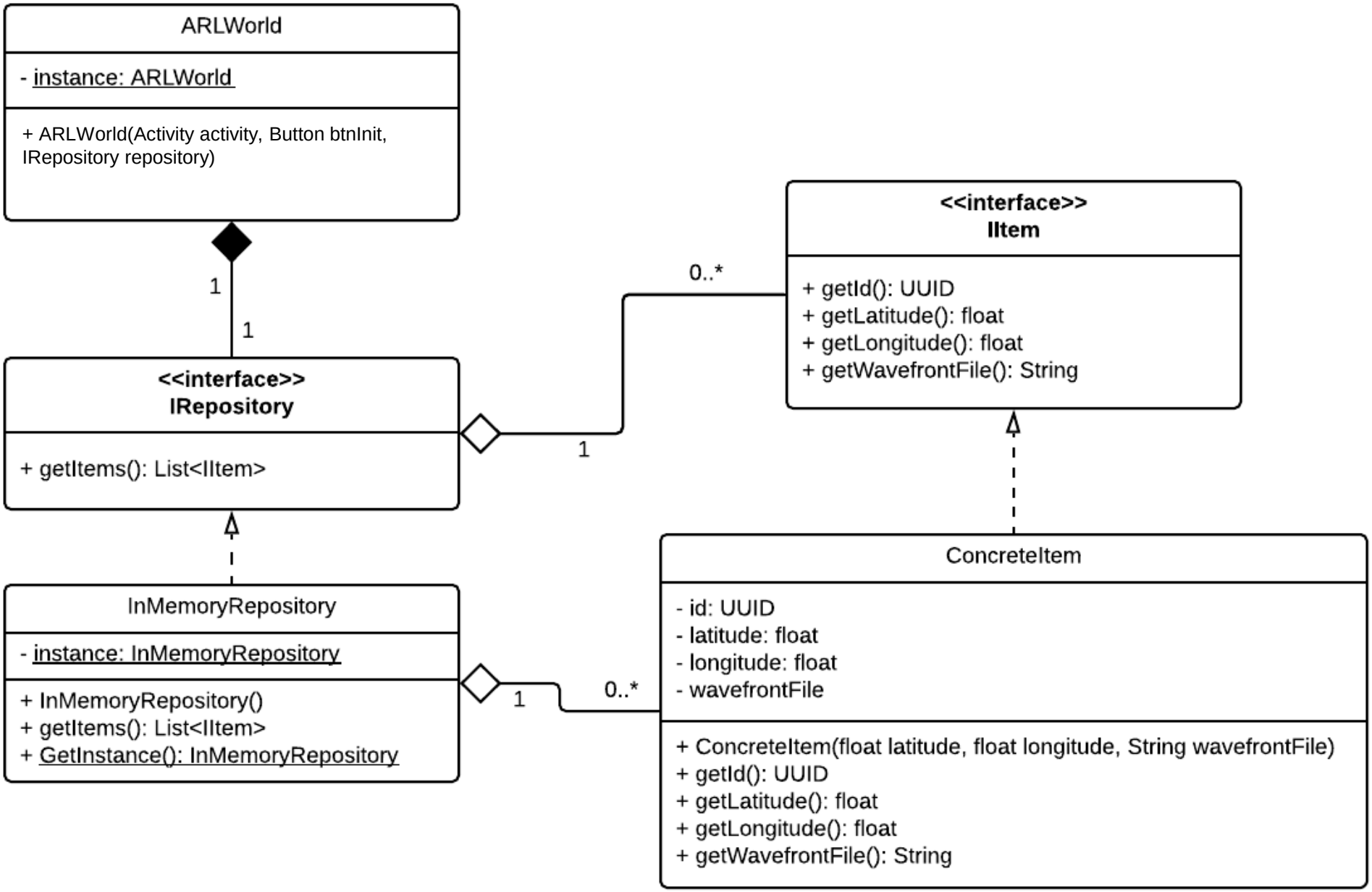
Fluxograma de posicionamento



Visão geral

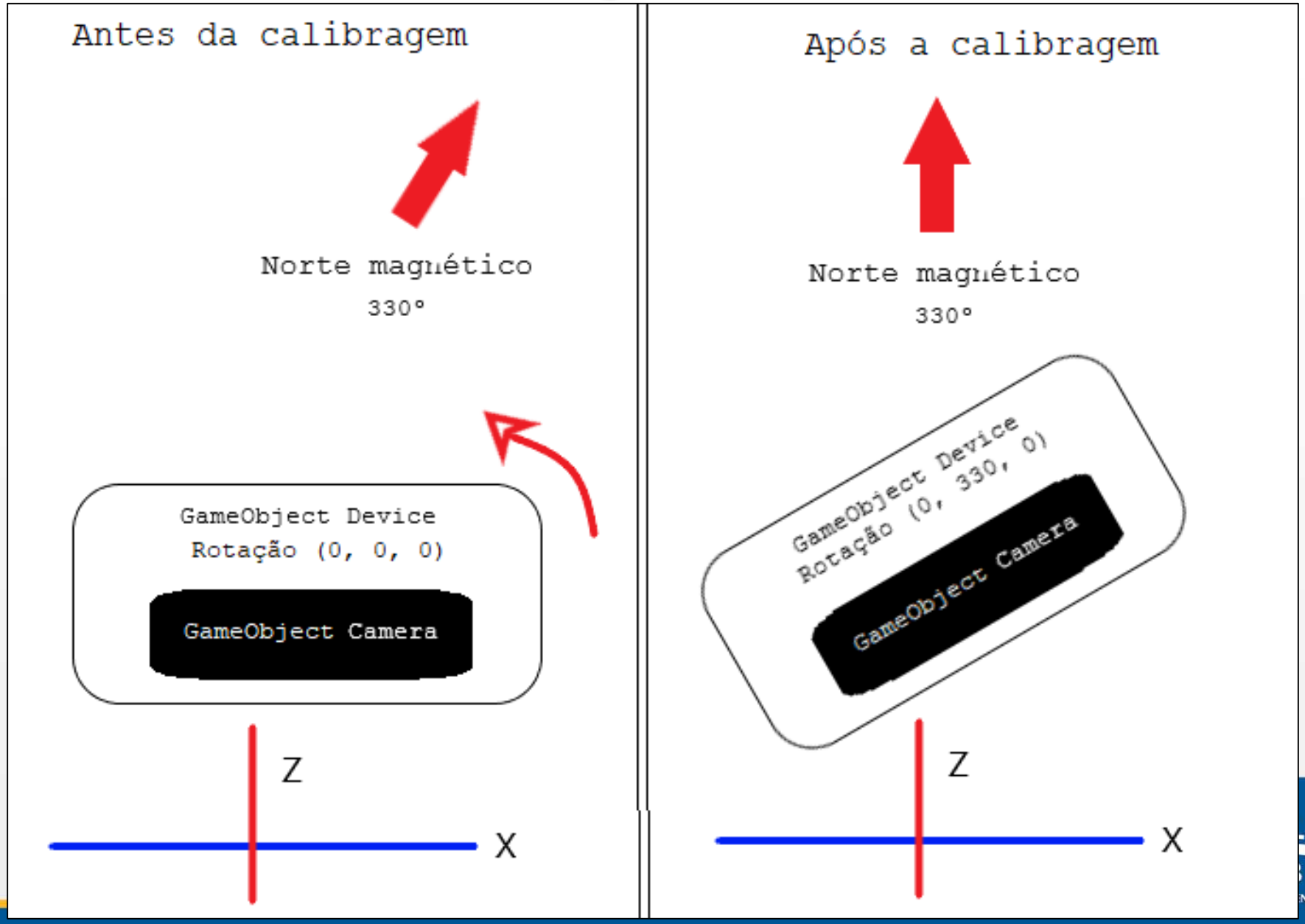

Biblioteca ARL
Camada Unity
World
Camada Android
Solicita itens do repositório
Retorna os itens próximos
ARLWorld
Inicia a biblioteca
Pacote de classes da aplicação final
Repositorio
Botão que inicia a biblioteca

Aplicação e biblioteca (integração)



Implementação

Calibragem inicial



Sensores do dispositivo

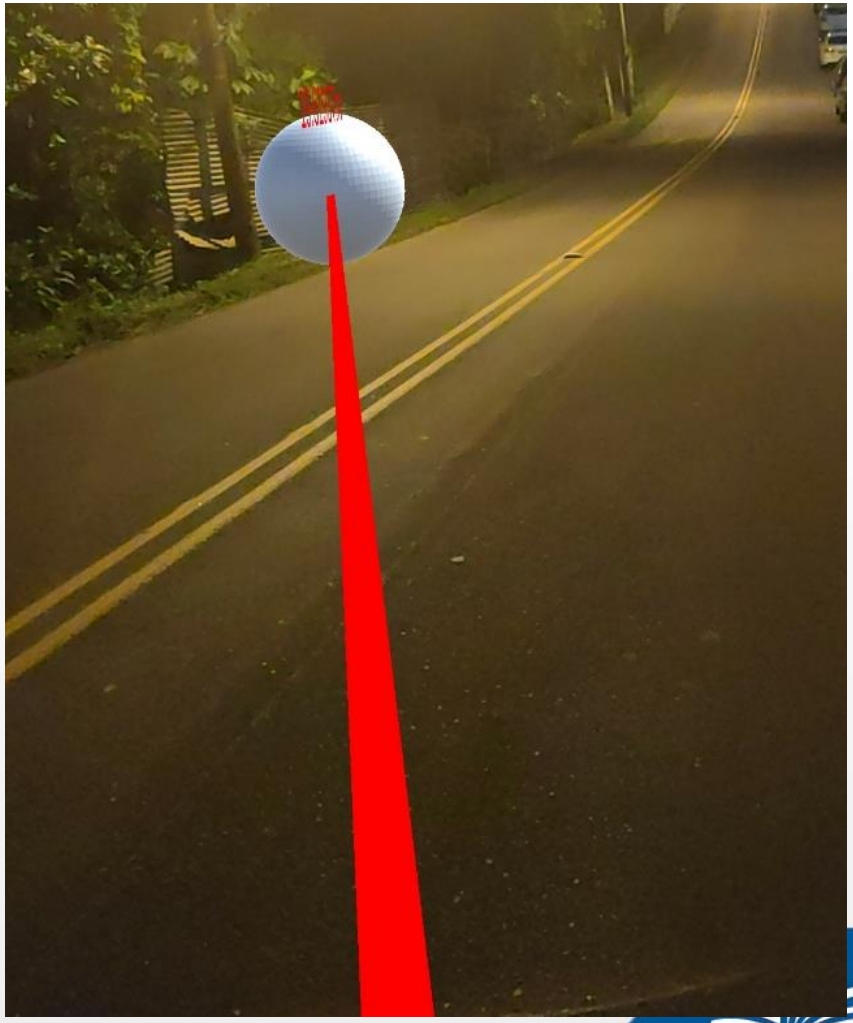
```
20      void Start()
21      {
22          Inicialização e verificação de permissões
34
35          locationService = UnityEngine.Input.location;
36          locationService.Start(10, 0.01f);
37
38          nextTime = Time.time;
39      }
40
41      void Update()
42      {
43          lastNorth = compassService.magneticHeading;
44
45          if (nextTime < Time.time)
46          {
47              if (locationService.status == LocationServiceStatus.Running)
48                  lastLocationInformation = locationService.lastData;
49
50              nextTime = Time.time + 1f;
51          }
52
53          if (mainCamera.transform.rotation.eulerAngles.x > 180)
54          {
55              lastNorth += 180;
56
57              if (lastNorth > 360)
58                  lastNorth -= 360;
59          }
60
61          Calibragem
73      }
```

Posicionamento de objetos

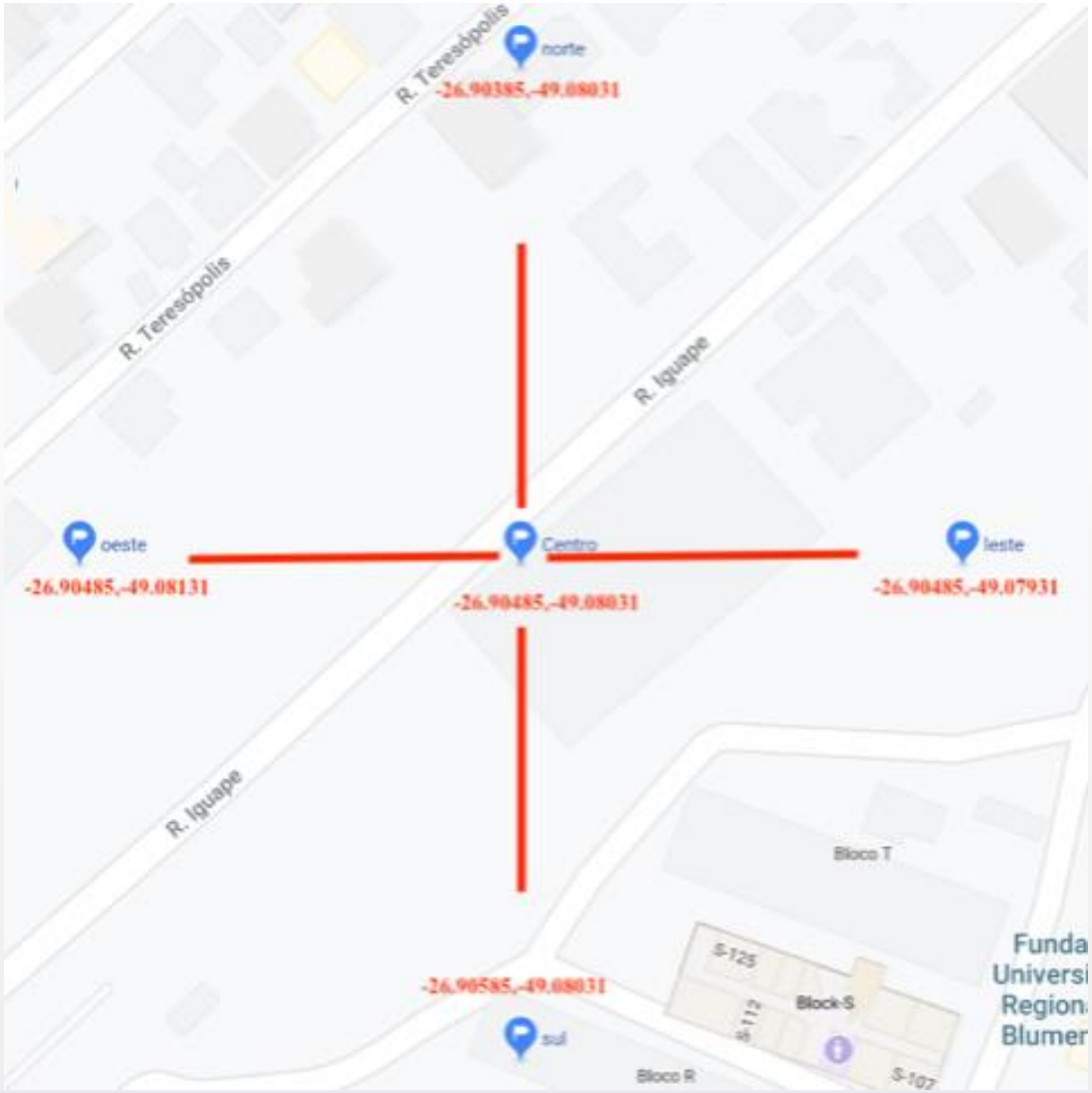
```
85          foreach (var itemInfo in items)
86          {
87              float relativeAngle = Functions.AngleBetween(latitude, longitude, itemInfo.Latitude, itemInfo.Longitude);
88              float distance = Functions.HaversineDistance(latitude, longitude, itemInfo.Latitude, itemInfo.Longitude);
89              var position = Quaternion.AngleAxis(relativeAngle, Vector3.up) * Vector3.forward * distance;
90
91              UiInstance.AddLog($"Calculated position {position}");
92
93              Pose pose = new Pose(position, Quaternion.identity);
94              Anchor anchor = Session.CreateAnchor(pose);
95
96              var newItem = Instantiate(DefaultItemModel);
97              newItem.transform.position = position;
98              newItem.transform.rotation = Quaternion.identity;
99              newItem.transform.parent = anchor.transform ;
100
101             var itemScript = newItem.AddComponent<ItemScript>();
102
103             itemScript.PlaceItem(itemInfo, position);
104             itemInfo.Placed = true;
105         }
```

Análise dos Resultados

- Correlatos;
- Imprecisão do GPS.



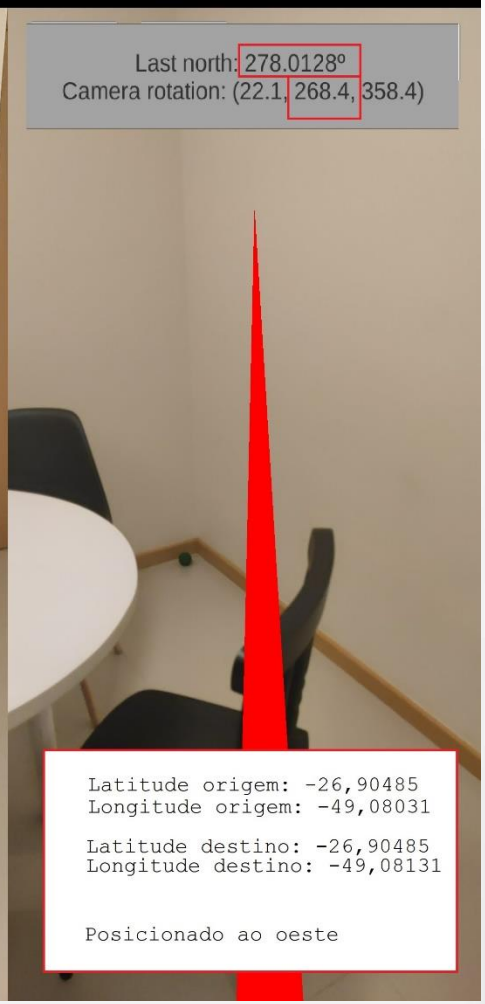
- Variação com a bússola em grandes distâncias.




Last north: 0.4903234º
Camera rotation: (13.6, 359.8, 358.9)
Latitude origem: -26,90485
Longitude origem: -49,08031
Latitude destino: -26,90385
Longitude destino: -49,08031
Posicionado ao norte


Last north: 96.18663º
Camera rotation: (24.0, 87.6, 358.5)
Latitude origem: -26,90485
Longitude origem: -49,08031
Latitude destino: -26,90485
Longitude destino: -49,07931
Posicionado ao leste


Last north: 187.8826º
Camera rotation: (19.2, 181.4, 358.5)
Latitude origem: -26,90485
Longitude origem: -49,08031
Latitude destino: -26,90585
Longitude destino: -49,08031
Posicionado ao sul


Last north: 278.0128º
Camera rotation: (22.1, 268.4, 358.4)
Latitude origem: -26,90485
Longitude origem: -49,08031
Latitude destino: -26,90485
Longitude destino: -49,08131
Posicionado ao oeste

- Usabilidade da biblioteca.

```
10  public class MainActivity extends Activity {
11      @Override
12      protected void onCreate(Bundle savedInstanceState) {
13          super.onCreate(savedInstanceState);
14          setContentView(R.layout.main_activity);
15
16          ItemsRepository repository = new ItemsRepository(); // repositório implementado
17
18          // binding de botões
19          Button btnCamera = (Button) findViewById(R.id.btnCamera);
20          Button btnNewItem = (Button) findViewById(R.id.btnNewItem);
21          Button btnGetItems = (Button) findViewById(R.id.btnGetItems);
22
23          btnNewItem.setOnClickListener(v -> { // configurando tela de cadastro
24              Intent itemsActivity = new Intent( packageContext: this, NewItemActivity.class);
25              startActivity(itemsActivity);
26          });
27
28          btnGetItems.setOnClickListener(v -> { // configurando tela de consulta
29              Intent itemsActivity = new Intent( packageContext: this, ItemsActivity.class);
30              startActivity(itemsActivity);
31          });
32
33          ARLWorld arlWorld = new ARLWorld( activity: this, btnCamera, repository); // instanciando a aplicação
34      }
35  }
```

```java
public class ItemsRepository implements IRepository {
    private static ItemsRepository instance;
    private List<IItem> itemList;

    public ItemsRepository(){
        instance = this;
        this.itemList = new ArrayList();
    }

    public List<IItem> getItems() { return itemList; }

    public static ItemsRepository GetInstance() { return instance; }

}
```

```java
public class Item implements IItem {
    private UUID id;
    private float latitude;
    private float longitude;
    private String wavefrontFile;

    public Item(float latitude, float longitude, String wavefrontFile)
    {
        this.id = UUID.randomUUID();
        this.latitude = latitude;
        this.longitude = longitude;
        this.wavefrontFile = wavefrontFile;
    }
    @Override
    public UUID getId() { return id; }
    @Override
    public float getLatitude() { return latitude; }
    @Override
    public float getLongigute() { return longitude; }
    @Override
    public String getWavefrontFile() { return wavefrontFile; }
}
```

Conclusões e Sugestões

- Objetivos alcançados:
  - Importação de objetos;
  - Visualização em realidade aumentada;
  - Compartilhamento entre usuários;
  - Trabalho no formato de biblioteca;
- Objetivos alcançados com algumas falhas:
  - Posicionamento dos objetos (GPS e bússola).

Conclusões e Sugestões

- Sugestões para extensão:
  - Importar outros formatos de arquivos;
  - Importar materiais e animações para os modelos;
  - Posicionar os modelos em cena através da câmera;
  - Calibrar a rotação da câmera com o norte de forma periódicas.

Apresentação prática