INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Desenvolvimento de uma aplicação móvel de apoio à completagem de campo

FRANCISCO DA SILVA LEITE FERREIRA DE AZEVEDO
Julho de 2019

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

Desenvolvimento de uma aplicação móvel de apoio à completagem de campo

Cartografia e Realidade Aumentada

Francisco da Silva Leite Ferreira de Azevedo

Dissertação para obtenção do Grau de Mestre em Engenharia Informática, Área de Especialização em Sistemas Gráficos e Multimédia

Orientador: Prof. Doutor Filipe de Faria Pacheco

Supervisor: Eng. José Alexandre Fernando Gomes

Porto, julho 2019

Dedicatória

Aos meus pais e irmãos por depositarem toda a confiança em mim e pelo apoio incansável que me deram ao longo desta etapa.

Resumo

Uma das apostas mais recentes das organizações, em áreas como, cartografia, medicina, desporto, comércio e jogos, é a realidade aumentada. Esta tecnologia permite facilitar parte dos processos de visualização e representação gráfica de informação, devido à inserção de elementos virtuais num ambiente real.

A produção de cartografia em Portugal encontra-se numa fase de transformação, onde são estudadas novas formas de substituir metodologias tradicionais, por técnicas e tecnologias de ponta, de modo a automatizar processos de construção de mapas e diminuir ao máximo os custos gerados.

Atualmente, uma das fases da cartografia que apresenta maiores dificuldades e custos é a completagem de campo. Esta fase consiste na verificação dos objetos nas ruas de um local específico, através de "plantas" em papel, caso os objetos não se encontrem nas "plantas", estas terão de ser alteradas manualmente. Contudo, este suporte físico apresenta inúmeras dificuldades, tais como, a sua extensão, a dificuldade de manipulação de grandes quantidades de dados, a falta de enquadramento no mapa ou erros no posicionamento dos objetos.

O presente projeto consiste na realização de uma aplicação móvel com uso de realidade aumentada que projeta cenários tridimensionais e mapas, representativos dos locais de estudo. Além das projeções são também adicionadas funções de edição de cenários (adição e remoção de pontos e linhas e edição de categorias) bem como novas funcionalidades, como a representação de trajetos, filtragens e armazenamento de objetos alterados.

Para o efeito, descrever-se-á a metodologia adotada no desenvolvimento de software, onde são identificadas as possíveis soluções para os problemas, os requisitos funcionais e não funcionais da aplicação, o design arquitetural da solução, a implementação dos vários casos de uso e, por fim, os vários tipos de testes. Fatores como a usabilidade, eficiência, adaptabilidade e posicionamento são avaliados na aplicação, a partir de um público-alvo (clientes internos da organização) e um público externo.

Palavras-chave: Cartografia, Realidade aumentada, Aplicação móvel, Representação de Cenários Tridimensionais, Mapas

Abstract

One of the most recent bets of organizations, in areas such as cartography, medicine, sport, commerce and games, is the augmented reality. This technology smooths some of the visualization and graphic representation of information, due to the insertion of virtual elements in a real environment.

The production of cartography in Portugal is in a phase of transformation, where new ways of replacing traditional methodologies with techniques and cutting-edge technologies are studied, in order to automate mapping processes and minimize the costs generated.

Currently, one of the phases of cartography that presents the greatest difficulties and costs is the field completeness. This phase consists of checking the objects on the streets of a specific location through paper "plants", if these objects are not in the "plants", they must be changed manually. However, this physical support presents numerous difficulties, such as its extension, the difficulty of manipulating large amounts of data, the lack of framing on the map or errors in the positioning of objects.

The present project consists in the realization of a mobile application with augmented reality that projects three-dimensional scenes and maps, representative of the places of study. In addition to the projections, are also added editing functions (add and remove points and lines and change categories) as well as new functionalities such as path representation, filters and storage of changed objects.

To that end, the methodology adopted in software development will be described, where are shown the possible solutions to the problems, functional and non-functional requirements of the application, architectural design of the solution, implementation of the many use cases and, finally, several types of tests. Factors such as usability, efficiency, adaptability and positioning are evaluated in the application, from a public (internal customers of the organization) and an external public.

Keywords: Cartography, Augmented Reality, Mobile Application, Representation of Tridimensional Scenes, Maps

Agradecimentos

O meu primeiro, e especial agradecimento, vai para os meus pais e irmãos, pelo apoio e dedicação manifestados, que me permitiu alcançar este caminho profissional.

Agradeço ao Instituto Superior de Engenharia do Porto (ISEP), ao departamento de informática (DEI) e à InfoPortugal por todos os conhecimentos transmitidos e influências que contribuíram claramente para o sucesso deste projeto e do meu percurso académico.

Um grande agradecimento ao meu orientador Prof. Doutor Filipe de Faria Pacheco pela confiança depositada, pelo excelente acompanhamento e pelo tempo despendido durante a realização da dissertação.

Aos meus supervisores da empresa Eng. Alexandre Gomes, Eng. Diogo Marques e Eng. Nuno Lopes pelos conhecimentos transmitidos e pelo acompanhamento indispensável durante esta caminhada.

Aos restantes companheiros da InfoPortugal e amigos que me ajudaram a superar as dificuldades e que contribuíram para o sucesso deste projeto.

Por fim, deixo um forte agradecimento à minha namorada pela compreensão, motivação e carinho demonstrados ao longo do projeto.

Índice

Lista de Figuras

Lista de Tabelas

Acrónimos e Símbolos

Lista de Acrónimos

AHP	*Analytic Hierarchy Process*
AMD	Apoio à decisão multicritério
API	*Application Programming Interface*
AR	*Augmented Reality*
CAD	*Computer Aided Design*
DGT	Direção-Geral do Território
FFE	*Fuzzy Front End*
FIST	*Fast Industrial-Strength Triangulation of Polygons*
GARS	*Global Area Reference System*
GE0REF	*World Geographic Reference System*
GIS	*Geographic Information System*
GPS	*Global Positioning System*
IDC	*International Data Corporation*
ISO	*International Organization for Standardization*
JSON	*Javascript Object Notation*
MGRS	*Military Grid Reference System*
MVC	*Model-View-Controller*
MVP	*Model-View-Presenter*
MVVM	*Model-View-ViewModel*
NASA	*National Aeronautics and Space Administration*
NCD	*New Concept Development*
OOP	*Object-Oriented Programming*
QEF	*Quality Evaluation Framework*

RA	Realidade Aumentada
REST	*Representational State Transfer*
SCORM	*Sharable Content Object Reference Model*
SDK	*Software Development Kit*
UTM	*Universal Transverse Mercator*
VR	*Virtual Reality*
WGS	*World Geodetic System*

1 Introdução

O presente capítulo apresenta, inicialmente, uma breve contextualização e enquadramento do desenvolvimento das áreas da cartografia e da realidade aumentada, bem como as suas respetivas importâncias e finalidades. A área da Cartografia vem adotando novas abordagens, mais tecnológicas e eficientes, devido à necessidade de simplificar os processos cartográficos e aumentar a qualidade dos mapas produzidos.

Seguidamente, são expostas as dificuldades inerentes ao processo de cartografia, nomeadamente nas fases de completagem de campo (verificação e alteração das "plantas cartográficas) e edição cartográfica. A completagem de campo apresenta dificuldades, devido à existência singular dos métodos tradicionais cartográficos em papel e que, consequentemente, influenciam negativamente as fases seguintes.

São enumerados os objetivos do produto com o intuito de solucionar ou facilitar as atividades da cartografia, identificando o produto como uma aplicação móvel de realidade aumentada que auxiliará o levantamento e edição de campo.

Neste capítulo, são também mencionados os resultados esperados e os resultados obtidos, tendo em conta as limitações dos serviços de *software* e *hardware*, e a abordagem utilizada para atingir esses resultados.

Por fim, é resumida a análise de valor adotada, identificando o processo de inovação *Fuzzy Front End*, a proposta e criação de valor, o modelo de negócio, a cadeia de valor e a metodologia escolhida de apoio à decisão.

1.1 Contextualização

O desenvolvimento exponencial de inovações tecnológicas tem proporcionado avanços muito significativos em diversas áreas, nomeadamente, nas áreas da ciência cartográfica [1] e realidade aumentada [2]. O uso de sistemas computacionais nesta área científica tem

possibilitado aumentar o armazenamento e controlo de informação, manipular grandes quantidades de dados, representar digitalmente grandes espaços e áreas, bem como agilizar e facilitar cálculos.

Por um lado, a Cartografia tem cada vez mais um papel fundamental no que concerne à localização, identificação e estudo dos espaços [3].

A análise de localizações, a construção de estradas, a previsão do tempo, a gestão e controlo de florestas, a navegação através de mapas digitais e o GPS, são algumas das atividades que são impraticáveis na ausência de documentos cartográficos. Grande parte da cartografia é concebida com o apoio a recursos modernos: fotos aéreas, sistemas de posicionamento por satélite, programação computacional.

A maioria dos mapas elaborados atualmente, apresentam um grau elevadíssimo de exatidão e rigor, principalmente, na identificação de áreas, limites e distâncias. Para o efeito, são imprescindíveis conhecimentos específicos para a representação de aspetos naturais e artificiais, aplicação de operações de campo e laboratório, metodologia de trabalho e conhecimento técnico para a obtenção de eficácia desejada.

Neste sentido, a realidade aumentada apresenta-se como uma tecnologia que expande o mundo real através da inserção de elementos virtuais com suporte a vários dispositivos tecnológicos, onde se destacam: câmaras, sensores, giroscópio e acelerómetros. A Figura 1 exemplifica a inserção de elementos virtuais no mundo real [4].

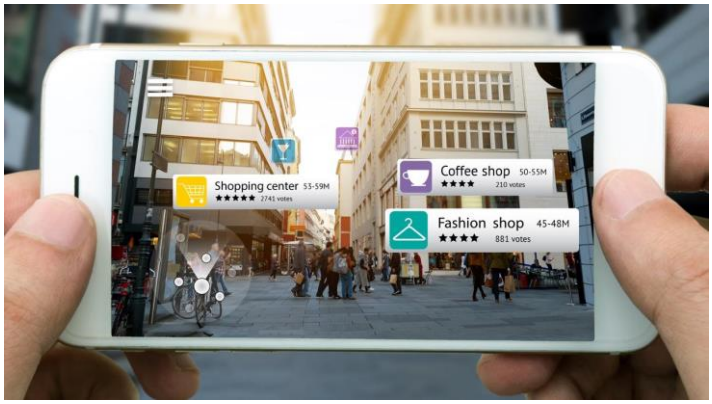


Figura 1 – Elementos virtuais no mundo real

Uma das características principais da realidade aumentada é a apresentação de modelos, imagens e vídeo dentro de um contexto real [5]. À medida que esta tecnologia se torna mais sofisticada e a redução de custos de desenvolvimento se torna uma realidade, o investimento aumenta. A Statis, portal online de previsão de mercado, prevê aproximadamente 1 bilião de utilizadores de realidade aumentada até 2020 [6].

Esta vertente, a realidade aumentada, funciona também como suporte ao desenvolvimento de novas tecnologias aplicadas em diversas áreas: medicina, cartografia, jogos, exercício físico, educação, *marketing*, entre outras [7].

A projeção de cenários cartográficos e a navegação geográfica, tendo como referência a Google Street View [8], são dois dos principais interesses e focos desta tecnologia e do próprio desenvolvimento cartográfico.

1.2 Problema

O desenvolvimento cartográfico é parcialmente efetuado de forma manual e física, podendo originar erros de imprecisão e rigor. Apesar dos contributos tecnológicos e cartográficos, existem, ainda, impasses relativos à recolha e tratamento da informação cartográfica e mapeamento.

Durante o processo de produção de mapas cartográficas é necessário realizar a fase de completagem de campo, uma vez que não é possível identificar e classificar com clareza todos os objetos cartográficos apresentados nas fotografias aéreas; objetos de maior dimensão sobrepõem-se a outros de dimensões menores [9], como o caso de postes de comunicações, tampas de saneamento, bancos, etc. Assim, este processo baseia-se na identificação e classificação de cartografia e, consequentemente, na validação das "plantas" originadas a partir das imagens aéreas. Este processo, que é executado nos próprios locais a estudar, é considerado significativamente moroso e problemático.

Ora, a InfoPortugal tem identificado, como principais problemas da verificação da informação cartográfica, a falta de eficiência da atividade, a inconsistência na identificação e categorização dos objetos, a dificuldade de enquadramento, a sobreposição de objetos, bem como, o mau posicionamento dos mesmos. Além disso, acontece com alguma frequência, que o responsável que realiza a recolha de campo, posiciona os novos elementos de forma errada (ex.: colocar um poste de iluminação no lado oposto da estrada). A manipulação de grandes quantidades de informação também são fonte de dificuldade acrescida. Os desenhos depois de corrigidos no campo, são posteriormente, reencaminhados para a fase de tratamento de dados, porém, muitas vezes, chegam com demasiada informação ou de forma incompreensível para os elementos da edição.

1.3 Objetivos

O objetivo principal da aplicação consiste no desenvolvimento de uma aplicação móvel Android de realidade aumentada que facilite os processos de completagem de campo e de edição na área de cartografia aplicada. O produto deverá mostrar cenários tridimensionais semelhantes às fotografias aéreas tiradas nas posições geográficas correspondentes. À medida que o encarregado da completagem se movimentar pelas ruas, a aplicação deverá atualizar o

ambiente cartográfico envolvido. O número de objetos apresentados na tela do dispositivo é controlado por um alcance, ou seja, um objeto que não se encontre dentro do alcance do utilizador não deverá ser mostrado. Esta medida permitirá controlar a quantidade de informação representada, diminuindo assim possíveis problemas de performance. De forma a colmatar a falta de enquadramento, é adicionado um mapa a duas dimensões com uma visão total do local e uma opção de filtragens de informação.

A aplicação poderá também adicionar e remover objetos, além de editar as respetivas informações. Estas ações serão armazenadas num ficheiro em memória, dado que para os elementos da edição cartográfica é mais prático receberem a informação através de um ficheiro, permitindo a manipulação dos dados de forma mais eficiente.

Além dos requisitos mencionados, a solução terá também como objetivos, aumentar a eficiência do processo, através de navegações no mapa, bem como, reduzir erros de deteção e de posicionamento de elementos, através de posicionamento manual dos mesmos.

A representação gráfica será desenvolvida sobre uma plataforma de desenvolvimento de *software mobile*, juntamente, com um kit de suporte à realidade aumentada. Para o efeito, é necessário avaliar o kit de realidade aumentada que mais se adequa às necessidades e que diminua, nomeadamente, os erros dos cálculos das posições. Os dados relativos às coordenadas geográficas terão de ser convertidos em diferentes tipos de coordenadas. Além disso, terão de ser efetuados cálculos de matrizes (transformações, translações, rotações, escalas e inversas) para localizar os objetos conciliando as coordenadas métricas com as coordenadas reais tendo como referência o dispositivo móvel. Parte dos cálculos poderão ser realizados manualmente ou através de bibliotecas devidamente testadas.

1.4 Resultados Esperados/Obtidos

Relativamente ao produto FieldAR espera-se obter, efetivamente, uma aplicação que represente várias cenas tridimensionais constituídas pelos elementos cartográficos recebidos através de um ficheiro com os dados geográficos (*shapefile)*. É possível que problemas externos como o GPS e a bússola possam interferir e prejudicar o desenho da cena, isto é, os objetos cartográficos poderão ficar mal posicionados no mundo real. Estes problemas poderão surgir dependendo das condições meteorológicas, da linha de vista para os satélites (completagem numa rua entre edifícios altos), interferência eletromagnética (fábricas), etc. Contudo, este tipo de erros será corrigido através de um ajuste manual realizado por parte de cada utilizador, onde este poderá movimentar a cena e rodá-la até ficar fielmente posicionada.

Outro aspeto negativo que poderá surgir será na deteção do plano de base para o desenho da cena tridimensional. Apesar de existirem bons kits de desenvolvimento de realidade aumentada, estes ainda têm efetivamente algumas limitações que poderão prejudicar o uso desta aplicação, nomeadamente, interferências com a luminosidade e deteção de diferentes superfícies horizontais.

Por outro lado, prevê-se uma diminuição no tempo de recolha de campo e, consequentemente, uma maior eficácia e aproveitamento da informação recolhida.

Relativamente às tecnologias que irão ser adotadas (kit de realidade aumentada, bibliotecas de cálculo de posições, de desenho de figuras e de conversão entre sistemas de coordenadas), espera-se que estas sejam compatíveis entre si e que satisfaçam todas as necessidades, sem comprometer a eficiência da aplicação.

Em suma, é possível assumir que apesar de existirem sempre pequenos erros, quer internos (*software* da Google, bibliotecas) como externos (cálculos internos de posições geográficas e conversões), a aplicação estará adaptada a todos os casos e terá uma elevada taxa de sucesso.

1.5 Análise de valor

Um dos principais objetivos da Cartografia em geral, e da organização em particular, é gerar valor através do aumento da eficiência de processos e da qualidade dos mapas produzidos. Neste contexto, foi elaborada uma aplicação móvel de suporte a uma das fases da cartografia e que visa a representar dados cartográficos em três dimensões num ambiente de realidade aumentada. Além dos valores referidos, este conceito de negócio pretenderá aumentar a precisão dos dados recolhidos e potenciar o aparecimento e substituição de métodos tradicionais cartográficos.

A análise de valor teve por base a identificação e utilização do processo de inovação *Fuzzy Front End* (FFE) bem como dos componentes do *New Concept Development* (NCD). Foram, também, identificados os valores para cada interveniente do negócio e criada uma cadeia de valor com o intuito de compreender as atividades a desempenhar pela organização.

Definiu-se os vários setores do negócio como as atividades chaves do negócio, a proposta de valor, custos e receitas, através de um modelo de negócio Canvas.

Por fim, suportado pelo método de apoio à decisão *Analytic Hierarchy Process* (AHP), identificou-se a tecnologia de realidade aumentada mais adequada dadas as circunstâncias.

1.6 Abordagem preconizada

O projeto foi elaborado segundo uma abordagem bem definida, que contou com um conjunto de etapas que abrangem desde a definição exata do problema até à avaliação e validação do produto.

Numa primeira fase, realizar-se-á uma contextualização por parte da organização, na qual irão ser explicados os vários setores de desenvolvimento (cartografia, *software*, comunicação e informação), os processos adotados e respetivos problemas associados e as várias metodologias de trabalho existentes.

Seguidamente, será interpretado e refinado o problema em questão e, posteriormente, serão definidos os objetivos para a resolução do problema.

Consequentemente, serão apresentadas as abordagens e soluções possíveis para o problema, bem como os produtos existentes atualmente no mercado. Além disso, serão apresentados os kits de desenvolvimento de realidade aumentada e as linguagens, bibliotecas e tecnologias a adotar.

Depois de estabelecido o estado de arte e de avaliados os procedimentos e tecnologias, identificar-se-ão os requisitos funcionais e não-funcionais da aplicação móvel, além do *design* arquitetural da solução escolhida.

Seguidamente, iniciar-se-á a implementação do produto segundo boas práticas de programação e segundo a metodologia orientada aos objetos. Paralelamente, serão efetuados testes de performance da solução, de verificação de cálculos matemáticos e geográficos, através de ferramentas de suporte, de modo a verificar a validade da solução.

A aplicação será submetida a um conjunto de testes com utilizadores reais de recolha de campo para validar os aspetos técnicos e funcionais e para aferir a usabilidade da aplicação. Por fim, serão também efetuados testes unitários para validar o código produzido e avaliada a qualidade de *software* através de um plano QEF.

2 Contexto e Estado de arte

2.1 Contexto e Problema

Atendendo ao propósito da aplicação dar suporte ao processo de cartografia da organização, é necessário contextualizar mais detalhadamente a história da cartografia descrevendo cronologicamente as descobertas de maior relevância desde os primeiros métodos rudimentares até aos processos usados atualmente.

Explicam-se todas as fases do processo de cartografia da organização, destacando a fase da completagem de campo e os problemas inerentes à criação deste conceito de negócio.

Uma vez que o produto contempla funções de posicionamento geográfico foram necessariamente realizadas conversões entre sistemas de coordenadas. Assim, irão ser esclarecidos os tipos de sistemas de coordenadas, as nomenclaturas e projeções cartográficas.

Finalmente, é retratada a história da realidade aumentada e referido o seu exponencial crescimento em diversas áreas.

2.1.1 Cartografia

Desde os tempos mais remotos que a Humanidade tem a necessidade de representar a informação geográfica. Existe controvérsia entre os arqueólogos na identificação dos primeiros mapas gerados. Alguns arqueólogos acreditam que algumas pinturas rupestres representavam pequenas paisagens que serviam de base à navegação e de localização de zonas. Os primeiros mapas eram desenhados em cavernas, argila e até pele de animais e que nada tinham de semelhança com o mundo real [10].

O mapa mais antigo do Mundo (mapa babilónico), criado em 600 a.C., representava de forma simbólica um conjunto de cidades da Babilónia sobre a Terra.

Os antigos povos greco-romanos foram os pioneiros da projeção de mapas do Mundo, na qual se destacam duas figuras enigmáticas da história da cartografia grega: Anaximandro e Ptolomeu. Anaximandro produziu o primeiro mapa documentado do Mundo e Ptolomeu criou *Geographia* um dicionário e atlas projetado do conhecimento geográfico do Império Romano [11].

No século IV a.C., a China concebe os primeiros mapas estelares mais antigos do mundo.

Na Europa, durante a época dos descobrimentos, entre os séculos XV e XVII, foram inventados instrumentos que revolucionaram a área da cartografia (bússola, sextante e telescópio), criando assim, mapas do mundo inteiro com elevado grau de detalhe. Simultaneamente, com o aperfeiçoamento e descoberta de novas técnicas de cartografia, os mapas passaram a ser cada vez mais detalhadas e precisos.

Mais tarde, os processos litográfico e fotoquímico possibilitaram o desenvolvimento de mapas sem distorção de formas e com resistência à humidade e desgaste.

Atualmente, a evolução exponencial das tecnologias digitais e de métodos mais precisos e eficientes de suporte ao mapeamento, proporcionaram um elevado impacto na conceção de cartografia. A fotografia aérea (incluindo mais recentemente *drones*) e as imagens de satélite são dois exemplos representativos destes métodos.

A cartografia tradicional em papel está a ser substituída por métodos tecnológicos avançados, por armazenamentos de grandes quantidades de informação e de novas ferramentas e aplicações para desenvolvimento e visualização de mapas cartográficos. Cada vez mais, a população tem necessidade de recorrer a aplicações móveis para se localizar, observar mapas e encontrar caminhos [12]. A maioria dos mapas é concebida através de três tipos principais de *software*: CAD, GIS e *software* que proporcionam mapas mais interativos e dinâmicos.

2.1.2 Fases da cartografia

Geralmente, a conceção de mapas da organização segue uma estrutura bem definida, constituída por seis fases distintas, mas que se complementam com o objetivo de conceber o produto final; a cobertura aérea, o apoio fotogramétrico, a triangulação aérea, a restituição fotogramétrica, a adição e a alteração em campo e a edição cartográfica são as várias fases do processo cartográfico.

Inicialmente, é estudada a área a representar, através de cartografia já existente e são identificadas as especificações técnicas do voo que permitirão estabelecer os parâmetros adequados para a execução dessa mesma cobertura. A altura do voo, os aeródromos (zonas para abastecimento por exemplo), as condições meteorológicas e os equipamentos, são fatores essenciais a considerar previamente. Esta análise detalhada pretende garantir alta qualidade das fotografias aéreas. Ainda durante esta fase é realizado o processamento das imagens e da trajetória de voo.

Seguidamente, é realizado o apoio fotogramétrico, isto é, são esclarecidos os pontos fotogramétricos que apoiam a georreferenciação das fotografias aéreas.

A fase de triangulação aérea tem como objetivo principal determinar as coordenadas dos pontos de ligação e dos parâmetros de ligação externa recorrendo a um número mínimo de pontos de apoio e de medições precisas nas fotografias.

Na fase seguinte, é executada a restituição fotogramétrica, isto é, convertida a informação topográfica a três dimensões. Esta restituição está de acordo com o catálogo de objetos definidos e respetiva categorização.

De seguida, é efetuada uma completagem de campo que permite validar e interpretar os resultados do processo de restituição fotogramétrica. Durante este processo os responsáveis por esta etapa procuram verificar se as classificações atribuídas estão de acordo com a realidade.

Finalmente, através da informação obtida do levantamento de campo é corrigida e adicionada a informação topográfica. É realizado um tratamento topológico dos dados, caracterizado por vários processos dos quais se destacam, a codificação dos dados, a edição altimétrica e a edição planimétrica.

2.1.3 Completagem de campo

Uma das principais causas da origem da completagem de campo, advém de limitações nas fotos aéreas como é o caso da ocultação de objetos por meio de objetos de maiores dimensões e na suposição incorreta das categorias de determinadas áreas. Estas limitações ocorrem quando, por exemplo, uma árvore encobre uma tampa de saneamento ou um poste ou quando é atribuída uma categoria errada a uma zona florestal (e.g. zona de eucaliptos ao invés de uma zona de pinheiros). Assim sendo e de modo a encontrar e a corrigir estas falhas são necessárias visitas ao campo.

O processo de recolha de campo é fundamental para identificar anomalias e reforçar a validade dos dados cartográficos. Durante esta fase são adicionados e removidos informações e objetos e alterado categorias e limites de áreas e propriedades.

A recolha de campo é realizada minuciosamente. Contudo, existem alguns problemas quer de tempo, quer de funcionalidade, que poderão provocar custos acrescidos e influenciar a integridade dos dados e, posteriormente, dos mapas a construir. Atualmente, este processo é relativamente moroso e pouco viável na medida em que os responsáveis por esta atividade necessitam de carregar todas as matrizes de restituição em papel.

Outra dificuldade encontrada neste processo está relacionada com a manipulação de quantidades consideráveis de informação. As “plantas” depois de corrigidas no campo são posteriormente encaminhadas para tratamento de dados. Contudo, muitas vezes, estas chegam com informação desmedida tendo em conta espaços muito pequenos e incompreensível para os outros elementos encarregues da fase de edição. A Figura 2 mostra um exemplo de uma planta corrigida.

Figura 2 – Planta corrigida

Os dados cartográficos estão organizados segundo convenções e cores com o objetivo de facilitar a interpretação dos responsáveis de campo e estão de acordo com as normas definidas pela Direção Geral do Território (DGT).

2.1.4 Sistemas de coordenadas geográficas

Um sistema de coordenadas de um mapa, bem como de outro qualquer tipo de suporte de informação geográfica pode estar de acordo com um sistema de coordenadas geográficas à superfície. Este processo de transformação é conhecido como georreferenciação [13] e é necessário para a produção cartográfica.

A georreferenciação pode ser determinada segundo quatro tipos de sistemas de coordenadas: astronómicos, cartesianos tridimensionais, elipsoidais e cartográficos. Os sistemas astronómicos são utilizados para encontrar a posição aparente dos astros na esfera celeste, os sistemas cartesianos geocêntricos são definidos segundo um sistema de eixos (X, Y e Z) com origem no centro da Terra. O sistema elipsoidal depende dos sistemas de coordenadas anteriormente mencionados, e é representado segundo um sistema de coordenadas

geodésicas (longitude, latitude e altitude), e que permite converter as coordenadas geodésicas em planas, através de projeções a duas dimensões.

Um dos sistemas de coordenadas de referência mais usados inclusive pelo GPS é o *World Geodetic System* (WGS-84) que, tal como o nome indica, trata-se de um sistema geodésico. O WGS-84 é um standard da Cartografia e da navegação de satélite.

A identificação de uma posição geográfica num mapa é dependente do tipo de projeção escolhida. Uma projeção cartográfica consiste numa forma de representar uma superfície esférica ou elítica da terra, de três dimensões, num plano de duas dimensões, cujas propriedades métricas se enumeram:

- Área
- Direção
- Distância
- Forma
- Escala

A escolha da projeção contribui para a precisão da análise GIS (sistema de informação geográfica). As projeções de mapas mais comuns são *Universal Tranverse Mercator (UTM)*, *Military Grid Reference System (MGRS), Global Area Reference System (GARS)* e *World Geographic Reference System (GEOREF).* Os resultados convertidos para cada projeção dependem de um conjunto de parâmetros e convenções.

O sistema de coordenadas da projeção UTM usa um sistema cartesiano métrico e separa a Terra em 60 zonas, cobrindo bandas de 6º graus de longitude. A posição da Terra é dada pelo número da zona onde este se insere e o par de coordenadas Este e Norte.

2.1.5 Realidade Aumentada

Surpreendentemente, a realidade aumentada e a realidade virtual têm sido trabalhadas há várias décadas apesar do seu incremento, nestes últimos anos. Estas duas áreas têm permitido faturar milhões de euros e quer os investimentos, como a procura, têm aumentado em larga escala [14]. Enquanto que a realidade virtual substitui o mundo real a realidade aumentada adiciona elementos ao mundo real.

Apesar de atualmente a área de realidade aumentada estar mais direcionada para o entretenimento, outras áreas têm vindo a explorar esta potência tecnológica: educação, treino militar, física científica, *marketing*, mapeamento, entre outras.

A realidade aumentada é um ambiente imersivo tridimensional onde são integrados elementos virtuais a partir de imagens em tempo real do mundo real, através de uma câmara e de sensores como o giroscópio, o acelerómetro e o manómetro. Além do *hardware*, algumas ferramentas de *software*, em particular a visão computorizada e o reconhecimento de objetos, permitem

manipular a informação do mundo real e melhorar a perceção do utilizador com os diversos cenários. Este ambiente pode ser visualizado de diferentes formas, quer através de óculos específicos, câmaras emparelhadas ou simplesmente, em grande parte dos telemóveis e *tablets* atuais. Esta experiência virtual permite estimular sensações visuais, auditivas, somatossensoriais e até olfativas.

Existe uma grande controvérsia sobre o ano de criação deste conceito, contudo os relatos mais antigos afirmam que, em 1957, o cinematógrafo Morton Heilig inventara o primeiro ambiente de realidade aumentada (*Sensorama*) [15]. Este sistema, apesar de não ser computacional, tinha como função principal criar experiências sensoriais aos utilizadores através da inclusão de informação adicional.

A exploração desta área foi continuada em 1965 em *Harvard* por um professor e cientista de computação, Ivan Sutherland que juntamente com o seu estudante Bob Sproull desenvolveram a conhecida "*The Sword of Damocles*". Este aparelho de realidade aumentada consistia num sistema montado na cabeça onde eram representados vários *frames* a partir de um sistema computacional muito rudimentar.

Em 1974, Myron Krueguer construiu um pequeno cenário de realidade artificial. Este projeto denominado *VideoPlace* consistia num ambiente interativo constituído por projetores e câmaras de vídeo [16].

Mais tarde, em 1990, o investigador Tom Caudell realizou uma investigação denominada, *Boeing*, que teve como objetivo substituir as grandes placas de compensado que continham instruções para cada avião, por um dispositivo também montado na cabeça que mostrava os esquemas específicos de um avião por meio de alta tecnologia e que os projetava em placas reutilizáveis geridas por um sistema computacional.

Um dos primeiros sistemas de AR verdadeiramente concebidos, suportados pelas novas tecnologias, foi o *Virtual Fixtures* por Louis Rosenberg em 1992. Este sistema robótico fora projetado para compensar as dificuldades de processamento 3D na força aérea. O exoesqueleto desta tecnologia de ponta permitia aos militares controlar maquinaria virtual para realização de determinadas tarefas.

A partir do século XX, os avanços começaram a ser mais rápidos, mais elaborados e com maior impacto no Mercado tecnológico:

- Hirokazu Kato criou o ARToolKit, biblioteca de *software* que usava *videotracking* para sobrepor gráficos computacionais com câmaras de vídeo (2000);
- Sportvision desenvolveu o primeiro sistema gráfico capaz de inserir linhas virtuais nas câmaras da NFL, durante a temporada de 2003;
- Em 2009, o Adobe Flash disponibilizou uma ferramenta de design de AR (ARToolKit);
- A Volkswagen MARTA app disponibilizava uma assistência de reparação virtual de veículos (2013);
- Em 2014, a Google anunciara os primeiros óculos de realidade virtual e aumentada;

- A Microsoft introduziu mais tarde, já em 2016 novos equipamentos de realidade aumentada, tal como o HoloLens;

2.2 Estado de arte

A funcionalidade mais relevante da aplicação, consiste na projeção de um cenário cartográfico. Deste modo, a presente secção identifica abordagens para a representação de mapas digitais. Escolheram-se quatros tipos principais de representação: representação bidimensional, tridimensional, vistas panorâmicas e tridimensional com realidade aumentada. A representação bidimensional é baseada segundo dois eixos. A tridimensional adiciona um terceiro eixo, denominado de altitude permitindo obter noções de profundidade. A visualização da cena 360º é obtida a partir de vistas (imagens) panorâmicas. Por fim, a tecnologia de realidade aumentada além de gerar um ambiente 3D real permite adicionar elementos virtuais à cena.

De seguida, exemplificam-se algumas ferramentas de software de construção de mapas e são mencionadas interfaces de programação, bem como vantagens e desvantagens de cada uma.

Relativamente ao conceito de realidade aumentada são mostradas aplicações de várias áreas incluindo na área da cartografia.

Finalmente, enumeram-se e caracterizam-se alguns dos kits de realidade aumentada existentes, bem como as vantagens e desvantagens associadas.

2.2.1 Representações cartográficas digitais

As novas tecnologias, novas técnicas e novas ferramentas transformaram por completo a conceção de mapas sendo possível não só a existência de mapas estáticos e impressos, como dispositivos digitais, automáticos e interativos.

Na globalidade, quase todas as áreas utilizam material tecnológico e o setor da Cartografia não é exceção. Apesar de grande parte do desenvolvimento de mapas ser computacional, existem ainda procedimentos manuais que para além de serem muito mais morosos, também estão suscetíveis a um maior número de erros.

Desta forma, a solução ideal para a solução deste fator, tempo, foi a criação de uma aplicação móvel. Além da sua incontestável facilidade de transporte, comparativamente com os esboços das fotos aéreas, a sua visualização poderá ser uma mais-valia comparativamente com os esboços, dado o controlo da luminosidade e a possibilidade de realizar ações de ampliação e redução das imagens. A principal desvantagem da aplicação móvel é a sua limitação de bateria.

Outra vantagem do uso do dispositivo é a possibilidade de armazenar e controlar grandes quantidades de informação. Uma das grandes desvantagens dos esboços em papel é o elevado

número de apontamentos que estes possuem provocando assim dificuldades e falhas na interpretação dos mesmos.

Representação de mapeamento bidimensional

Atualmente, devido ao avanço das vertentes cartográfica e tecnológica, são vários os meios de representação de mapas geográficos a duas e a três dimensões. O *Web Mapping*, os serviços de mapeamento para aplicações móveis e o *digital map* são os meios mais recorrentes para a exposição de mapas a duas dimensões. Enquanto que o *digital map* apenas contempla mapas estáticos, os *Web Mapping* e os serviços móveis recorrem a sistemas de informação geográficos (GIS) através de internet e, são, na maioria das vezes, interativos e em constante atualização. Estas representações não são consideradas, apenas, meios de pura cartografia, pelo que permitem aos utilizadores realizar pesquisas variadas e filtragens de informação, tal como filtragem de locais, transportes, pontos de interesse, entre outros.

Considerando os tipos de visualização de mapas existentes, esta gama abrange desde simples imagens traçadas a partir de *software* cartográfico até às imagens por satélite e foto aéreas.

A representação cartográfica bidimensional projeta os mapas apenas segundo dois eixos cartesianos, x e y, longitude e latitude, respetivamente. Esta continua a ser utilizada, para a representação de mapas estáticos e outros processos criativos de visualização de grandes quantidades de informação.

Representação de mapeamento tridimensional

Apesar de mapas cartográficos bidimensionais já estarem bem definidos, existem ainda algumas preocupações com os mapas tridimensionais. Um dos aspetos fundamentais para a criação de tridimensionalidade, é a noção de profundidade e volumes [17]. A profundidade é conseguida pela introdução de um novo eixo cartesiano o z, no caso da cartografia, a altitude e que consequentemente desencadeia muitas outras possibilidades de representação cartográfica e noções de perceção por parte dos utilizadores finais. A visualização a três dimensões pode ser segmentada ainda em visualização estática através de modelação de cenários estáticos a três dimensões e em visualizações em *run-time* como o caso da realidade aumentada. “A maioria dos mapas tridimensionais é produzida considerando os mesmos aspetos utilizados para os mapas bidimensionais, o que muitas vezes, pode causar problemas no resultado final, e consequentemente, na forma de uso” [18]. Falta de simbologia e modelos semelhantes à realidade poderão confundir os utilizadores e diminuir a taxa de sucesso de usabilidade.

Representação por vistas panorâmicas 360º

Outra forma também bastante eficaz de visualizar todos os elementos cartográficos é através de vistas panorâmicas, como é o caso da Google Street View e da Mapillary. A Google Street View disponibiliza vistas horizontais a 360º e vistas verticais a 290º permitindo assim criar um ambiente abrangente ao nível do chão com um elevado grau de detalhe.

Ambos os *softwares* permitem mostrar as fotografias segundo diferentes tamanhos e a partir de qualquer direção e diversos ângulos. A representação panorâmica apesar de ser uma abordagem interessante devido ao seu elevado nível de rigor apresenta algumas desvantagens como o caso do tempo entre atualizações, falta de noções de profundidade e possibilidade de objetos cartográficos ocultos.

Representação tridimensional com realidade aumentada

O uso de realidade aumentada em diversas áreas tem proporcionado ambientes mais interativos e dinâmicos, além de facilitar a visualização e a compreensão de vários modelos no mundo real. A adoção de sistemas de navegação e localização virtuais representam um dos fatores que mais tem contribuído para o setor de Cartografia.

A simples representação tridimensional não é suficiente para mostrar com precisão as posições e dimensões das infraestruturas, áreas e objetos cartográficos. A realidade aumentada permite tornar as navegações mais eficientes e gerar mais valor para os utilizadores através de dados mais claros e concretos [19]. Além das vantagens mencionadas, a construção de cenários de dimensões reais com base nas coordenadas de GPS contribui para melhorar a perceção dos utilizadores com o meio onde se inserem.

Outra área que tem aproveitado significativamente com o desenvolvimento deste tipo de representação, é a área do turismo. Os turistas têm usufruído de sistemas de navegação inovadores, mais abrangentes e com maior diversificação de informação, indo ao encontro dos interesses e necessidades de cada um. Exemplos do uso destas tecnologias são, a exibição de rotas virtuais no mundo real e a exposição de pontos de interesse a três dimensões com informações associadas.

2.2.2 Tecnologias de representação cartográfica

O número de aplicações que permitem construir mapas cartográficos a duas e a três dimensões tem aumentado consideravelmente. Este facto decorre da quantidade de utilidades que estes apresentam, não só na área da cartografia, mas também, em jogos, programas de educação, formação, *marketing* e redes sociais. Atualmente, MapBox*,* Google Maps API e Here Maps API são das ferramentas mais poderosas e usadas na criação e *design* de mapas, navegação e localização [20].

2.2.2.1 MapBox

Esta ferramenta fornece mapas *online* personalizados para algumas das maiores empresas mundiais, como o Facebook, Financial Times e o Snapchat. Grande parte das fontes dos dados são obtidos através da OpenStreetMap, da NASA e de bases de dados de proprietários, como a DigitalGlobe. Esta tecnologia é muito utilizada para análise espacial em tempo real, o que facilita a integração da localização nos vários dispositivos móveis. O MapBox é uma das plataformas mais usadas pelos programadores para a realização de mapeamentos a duas e a três dimensões e contempla APIs e SDKs para grande parte dos ambientes de desenvolvimento *web* e *mobile* [19].

Segundo os criadores do MapBox, "*Our tools let developers build a new world powered by location data. Real-time updates. Total customization. Developers first*.", o facto de ser direcionada para os criados de *software* permitiu que grandes aplicações como o Pokemon GO, Facebook e a NationalGeographic aderissem e utilizassem os seus serviços de navegação e localização.

Além destas possibilidades foi adicionada uma nova vertente de realidade aumentada para o mapeamento. A visualização de pontos de interesse, localizações e objetos no mundo real permitem, para além de ajudarem a localizar o utilizador no espaço, transmitem outra perceção da realidade, através da adição de elementos visuais. A plataforma Unity, o SceneKit e React *Native* são alguns dos ambientes de programação possíveis para a integração com esta vertente.

2.2.2.2 Google Maps API

A Google Maps Android *API* é a API mais usada para a criação de mapas *online* e são facilmente integrados em várias plataformas. A Google fornece tutoriais aos *developers* de *software* entre outros serviços. A Google Maps API tem também como objetivo melhorar a interação dos utilizadores com os próprios mapas. A Google Maps platform utiliza informação de mais de 200 países e territórios, e conta com milhões de atualizações por mês.

A Allianz e a Motor Harley-Davidson Company são duas das empresas que adotaram os dados desta tecnologia. Além desta possível observação dos mapas, a Google Maps API oferece outras experiências através de imagens da Street View e em 360º.

Assim como a MapBox esta disponibiliza APIs e SDKs para várias linguagens de programação (Android*,* Javascript e iOS).

2.2.2.3 Here Maps API

A Here Maps API também envia, maioritariamente, dados de localização e mapeamento. É possível através das suas APIs obter localizações de estradas, edifícios, parques entre outros objetos cartográficos. Evidentemente, a Here também disponibiliza APIs e SDKs para desenvolvimento Android, iOS, Rest e Javascript.

Apesar da integração de APIs de mapeamento e localização serem uma mais-valia para as novas aplicações, o uso de realidade aumentada tem proporcionado novas experiências para os utilizadores, através da introdução de elementos visuais no mundo real e tem sido uma das

apostas marcantes por parte das empresas, quer nas áreas de localização, *geo-tracking* e mapeamento, como também nas áreas dos jogos, mercado eletrónico e redes sociais.

2.2.3 Desenvolvimento de realidade aumentada

A potencialização da vertente de realidade aumentada tem originado aplicações cada vez mais poderosas, com finalidades distintas, bem definidas.

Os Invizimals, o Pokemon Go e o Ingress são alguns dos jogos mais icónicos e interativos pelo que inserem elementos virtuais como criaturas e objetos virtuais ao mundo real.

Com o uso do AR a aprendizagem torna-se mais fácil e interessante. Aplicações como a Anatomic permitem compreender a anatomia do corpo humano, desde os ossos, músculos e órgãos através de um scan de uma parte do corpo. Outra aplicação, também muito conhecida é o Solar System ARCore, que, tal como o nome indica, ensina e projeta o sistema solar, com as translações e rotações dos planetas, a partir de um plano escolhido pelo utilizador.

Uma das primeiras grandes empresas a implementar esta tecnologia no mercado comercial, foi a IKEA, através da aplicação IKEA Place. Aplicação semelhante ao Pottery Barn 3D Room View permite aos utilizadores observar produtos específicos e inseri-los num determinado espaço. Depois do processo de *scanning* ao chão de um determinado espaço é possível escolher certas peças do catálogo. Depois de selecionado um produto este aparecerá no ecrã e poderá ser movimentado e rodado [21].

As grandes redes sociais como o Facebook, o Snapchat e o Instagram também já adotaram esta tecnologia para reconhecimento facial. Depois de detetadas as faces é possível, não só adicionar os objetos virtuais como maquilhagem e acessórios de roupa, como também alterar partes do rosto.

No setor geográfico são destacados várias finalidades das aplicações, em particular a identificação de locais e respetivas distâncias aos utilizadores, exibição de caminhos virtuais, representação de mapas a duas e três dimensões de regiões específicas e a apresentação de modelos representativos de objetos cartográficos.

A Google tem adicionado novas funcionalidades de navegação ao sistema já existente Street View, onde para além das indicações a duas dimensões do caminho a seguir, também é possível verificar as direções no mundo real e em tempo real [22]. A Figura 3 mostra um exemplo de realidade aumentada desta ferramenta.

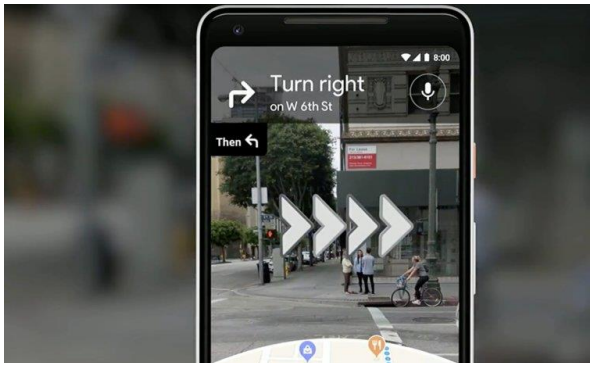


Figura 3 – Street View Google

Além de sistemas de navegação mais apelativos e eficientes, também são fornecidas informações relativas a pontos de interesse. São disponibilizados, não só informações gerais acerca de edifícios e zonas (restaurantes, hotéis, monumentos, jardins, etc), como também informações mais específicas, como preços de refeições ou estadias, distâncias entre o utilizador e o ponto de interesse, descrições e opiniões (Figura 4) [23].

Figura 4 – ARCity *app*

Além destas questões mais específicas, o grande destaque da realidade aumentada na cartografia está exatamente na projeção dos mapas 2D e 3D. Dois exemplos bastantes conhecidos de ferramentas de conceção de mapas são AReal [24] e o MapBox/ARKit. A Figura 5 e Figura 6 exemplificam as aplicações respetivamente.

Figura 5 – AReal *app*

Figura 6 – MapBox e ARKit

2.2.4 Ferramentas de realidade aumentada

O uso de realidade aumentada tem aumentado significativamente nos últimos anos, potenciando o desenvolvimento de inúmeras tecnologias e kits de desenvolvimento *software*. Os kits que mais se destacam atualmente no mercado são o Wikitude, ARCore, o ARKit e o Vuforia [25].

2.2.4.1 Wikitude

A Wikitude é uma das maiores e mais antigas ferramentas de desenvolvimento de *software* de realidade aumentada. Fundada em 2008, esta tecnologia tem como objetivo principal criar experiências interativas de realidade aumentada num ambiente real.

Esta tecnologia, apresenta um SDK poderoso com um largo leque de funcionalidades, das quais se destacam o reconhecimento de cenas, imagens e objetos, *rendering* de modelos 3D e o *tracking* de instantes. A Wikitude tem suporte para Android, iOS e Windows e *plugins* para plataformas e ambientes variados, como Unity e Cordova.

Existem algumas versões do SDK das quais se destacam a versão gratuita com parte das funcionalidades e com marca d'água e a versão Pro 3D.

2.2.4.2 ARCore

Atualmente o número de utilizadores de Android é muito superior ao número de utilizadores de iOS, proporcionando assim um aumento muito significativo dos utilizadores de ARCore comparativamente com os do ARKit. Estima-se que o número de dispositivos para 2020 será de aproximadamente de 3.6 milhares de milhão [26]. No último ano, a IDC (International Data Corporation) estima que a Google vendeu cerca de 85.1% do total de *smartphones* vendidos no mercado sendo que a Apple apenas vendeu 14.9% [27].

Este kit de desenvolvimento de *software*, criado pela Google, permite criar aplicações de realidade aumentada.

O ARCore usa três tecnologias chave para integrar a parte virtual com o mundo real a partir das câmaras dos telemóveis:

- *Motion tracking* (posição relativamente ao ambiente);
- Deteção de posições e localizações de superfícies planas horizontais e verticais;
- Estimativa e reconhecimento das condições da luz.

O *Motion tracking* é conseguido combinando as imagens da câmara e o sensor de movimento. O *hardware* que suporta esta ação é o acelerómetro que permite a medição da orientação e dos movimentos do dispositivo, o giroscópio que mede a rotação deste e, obviamente, a câmara do telemóvel, que permite mostrar a sobreposição entre o mundo real e a realidade aumentada. Existem ainda outras tecnologias que melhoram a interação e experiência do utilizador, como o uso de *machine learning* ou *computer vision*, de forma a produzir imagens e mapas espaciais de altíssima qualidade.

Um dos aspetos fortes do Android é a diversidade de dispositivos capazes de responder aos requisitos a partir do *hardware* e *design*. Além disso, o ARCore está a ser concretizado de forma a garantir a integração dos diferentes dispositivos com o *Software*.

A deteção de planos verticais e horizontais e as respetivas localizações são conseguidas através do uso do magnetómetro e do GPS. O magnetómetro determina a orientação relativa do campo magnético da Terra e o GPS permite a geolocalização e informação da localização em tempo real. O ARCore pode ser implementado em quatro ambientes de desenvolvimento, Android*,* iOS*,* Unity e Unreal Engine. Exemplos muito conhecidos de aplicações que incorporam esta tecnologia são o Pokemon Go, o Ikea Place e o AR Ruler *app*.

2.2.4.3 ARKit *e* ARKit *2*

O ARKit é uma plataforma de desenvolvimento de realidade aumentada exclusiva para dispositivos móveis iOS. Assim como no ARCore, estes ambientes permitem mostrar texto a três dimensões, objetos e personagens no mundo real. As cenas do *AR* são persistentes, isto é, um utilizador pode criar uma cena e outrem pode vê-la mais tarde.

Para além dos sensores de hardware revelados anteriormente o ARKit utiliza o reconhecimento de contexto para mapear, mediante a movimentação do telemóvel [28]. Uma das grandes vantagens desta tecnologia comparativamente com a concorrência direta é a sua altíssima eficácia na deteção de planos horizontais e verticais.

Assim como no ARCore, a câmara também é usada para determinar as fontes de luz que interagem com cada um dos elementos virtuais adicionados às cenas do mundo real. O AR dispõe de uma API muito desenvolvida e de SDKs bem definidos.

Neste momento o novo de kit de realidade aumentada ARKit pode ser experienciado por múltiplos utilizadores simultaneamente, e tem a capacidade de manter o estado depois de retomada a aplicação. Além disso, esta ferramenta tem a particularidade de permitir a inclusão de objetos reais nas experiências de realidade aumentada (efeito imersivo).

O número de *apps* que incorporam o ARKit tem vindo a aumentar progressivamente, apesar não obter o mesmo *feedback* que o ARCore. A Overstock, a Giphy World e a Porsche AR são alguns dos exemplos mais instalados pelos utilizadores de iOS.

2.2.4.4 Vuforia

O Vuforia é também um kit de desenvolvimento de realidade aumentada para aplicações móveis. A tecnologia subjacente ao reconhecimento e deteção de imagens e de simples objetos 3D é a visão computacional. Neste caso, o *developer* posiciona e orienta os objetos virtuais nas imagens do mundo real. Seguidamente, os objetos virtuais identificam a posição e a orientação da imagem em tempo real de forma ao modelo se adequar corretamente no mundo real.

Este kit disponibiliza SKDs que permitem, por sua vez, selecionar o destino de posição de imagens e modelos em tempo de execução e a capacidade de deteção de *canvas* virtuais 3D. O Vuforia contempla também APIs para C++, Java, Unity e .NET e o SDK suporta desenvolvimento para iOS e Android.

2.3 Análise de valor

Na análise de valor é descrito o processo de inovação *Fuzzy Front End* (FFE), bem como os componentes do *New Concept Development* (NCD). São, também, identificados os valores para cada interveniente do negócio e criada uma cadeia de valor de forma a analisar e compreender as atividades a desempenhar pela organização.

Este subcapítulo mostra os vários setores do negócio através de um modelo de negócio Canvas, onde são expostas as atividades do negócio, os intervenientes, os recursos necessários, os custos e receitas, etc.

Por fim, é determinada a tecnologia de realidade aumentada mais adequada para o projeto através do método de apoio à decisão *Analytic Hierarchy Process* (AHP).

2.3.1 Fuzzy Front End (FFE)

O processo característico do projeto é o *Fuzzy Front End* sendo um processo experimental, muitas vezes incerto. A comercialização é imprevisível e, portanto, há sempre uma grande resistência na introdução deste no mercado. A Figura 7 mostra o funcionamento e estrutura deste método.

O financiamento deste tipo de aplicações é bastante baixo ou até mesmo nulo, devido à sua imprevisibilidade de possível lucro. Outra das características deste processo é a necessidade de minimização de riscos e otimização do potencial, por parte dos indivíduos ou das equipas que conduzem a investigação.

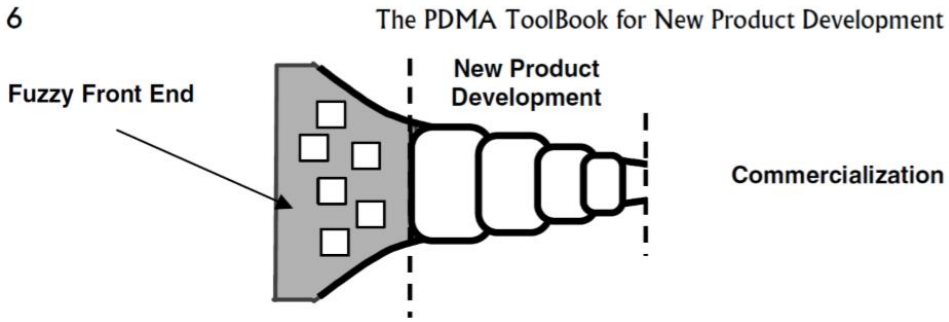


Figura 7 – Fuzzy Front End

New Concept Development Model

O modelo NCD é caracterizado por três componentes distintas:

- Os mecanismos principais que conduzem aos cinco elementos de controlo da organização são a liderança, a cultura, a estratégia de negócio;
- A área interna que define os elementos de controlo do *Fuzzy Front End* (identificação de oportunidades, análise de oportunidades, criação e seleção de ideias e enriquecimento e definição de conceitos);
- Os fatores que influenciam desde o processo de inovação até à comercialização. Estes fatores são as capacidades organizacionais, o mercado externo e as ciências internas e externas que podem estar associadas;

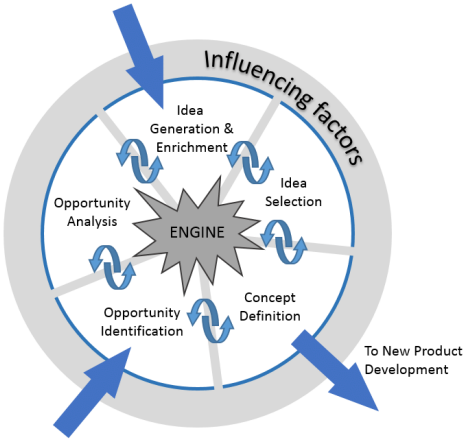


Figura 8 – *Model NCD*

São cinco o número de áreas que caracterizam o *Fuzzy Front End of Inovation* (Figura 8), identificação das oportunidades, análise das oportunidades, criação de ideias, escolha de ideias e por fim o conceito e desenvolvimento tecnológico [29].

A **identificação das oportunidades** representa as mais-valias que a organização identifica como relevantes para o negócio. Estas oportunidades poderão criar vantagens a nível de competitividade com outras empresas, desde logo, tornar a realização dos produtos/serviços mais eficiente, com diminuição de recursos e respetivos custos. Uma excelente oportunidade aproveitada poderá ser, simplesmente, o melhoramento de um produto já existente no mercado, como poderá abrir portas a novos produtos e negócios. Estas chances são exaustivamente estudas pela organização, devido às suas diversas finalidades.

Ora, a oportunidade de criação da aplicação em questão, surge da necessidade de facilitar e economizar o processo de construção de mapas, nomeadamente, na recolha de campo. A própria organização neste caso gerou uma oportunidade que poderá não só simplificar o método de cartografia em Portugal como também noutras áreas através da inclusão da realidade aumentada para a visualização de cenários reais tridimensionais.

Outro fator que também desencadeou a criação deste produto foi o facto de este ser uma das primeiras aplicações móveis relacionadas com cartografia que apresenta um vasto conjunto de funcionalidades. Existem já algumas aplicações que permitem desenhar cenários 3D e mapas, tendo em conta a localização GPS do utilizador, contudo todas as elas são ainda bastante simples e com objetivos muito específicos.

Esta possibilidade de inovação de negócio está comprometida, caso não seja realizada uma **análise das oportunidades,** juntamente com a informação vital para o negócio. Estudos do mercado, extensa investigação e experimentações são algumas das medidas tomadas para a validação dessas mesmas oportunidades. Esta extensão de negócio poderá ser influenciada positivamente através do acompanhamento do mercado tecnológico bem como das estratégias e valores da empresa.

A análise de oportunidades foi conseguida através de uma investigação da concorrência, das necessidades dos clientes (internos e externos), do mercado e o limite de exploração desta área. O facto deste serviço ser inovador e de não haver especificamente algo muito semelhante, faz com que a concorrência seja reduzida. As necessidades dos clientes na obtenção de informação gráfica do mundo real, desencadeou um estudo dos vários *softwares* de realidade aumentada (kits AR*,* APIs e SDKs).

A análise das oportunidades acompanhou o processo de desenvolvimento da aplicação móvel devido à exploração de novos procedimentos de representação virtual e de cálculos cartográficos e com vista a aumentar valor do produto e do negócio em geral.

A **criação de ideias** surge da transformação das oportunidades em ideias concretas que poderão ser construídas ou desconstruídas, reformuladas, modificadas ou atualizadas conforme o desenvolvimento do produto. A criação da ideia principal foi concebida depois de sucessivas

reuniões com os clientes internos, de uma estruturação dos requisitos, do planeamento e design da solução e da exploração prévia das tecnologias a utilizar.

Inicialmente, foi discutido o tipo de representação virtual disponibilizado para os encarregados da recolha de campo, a tecnologia de realidade aumentada mais adequada, o armazenamento da informação da cena, a possibilidade de representação de pontos, linhas, polígonos e sólidos considerando coordenadas reais de objetos cartográficos e respetivas conversões de medidas. Posteriormente, já num estado mais avançado da aplicação, debateu-se a possibilidade de adição de elementos cartográficos e de métodos de ajustes manuais das cenas.

As **ideias escolhidas** originaram a aplicação móvel de realidade aumentada para fins de representação de cenários tridimensionais segundo uma dimensão cartográfica. A tecnologia abordada para a visualização de elementos virtuais sob o mundo real foi o kit ARCore.

A representação dos objetos sob a forma de pontos e polígonos foi incluída, descartando assim os sólidos devido ao peso acrescido na memória e performance da aplicação. Os dados são armazenados em memória local do telemóvel (vantagem na recolha de campo) e os ajustes manuais da cena são conseguidos através da incorporação de joysticks e *sliders*.

Finalmente, o **conceito** consistiu no desenvolvimento do caso de negócio referido anteriormente tendo em conta as necessidades dos clientes, o mercado, os recursos disponíveis, os ricos, as oportunidades e ameaças.

2.3.2 Valor

Valor

Normalmente, o grande objetivo de qualquer empresa ou organização é a criação de valor quer para fins internos como para fins externos como os clientes finais, através da produção de produtos ou serviços.

Neste caso em particular, o valor do produto está na otimização e inovação do processo de levantamento de campo na área da cartografia e de dados cartográficos mais precisos e seguros.

Valor para o cliente

O valor tem de ir ao encontro com as necessidades e satisfações dos clientes. Cada cliente apresenta noções diferentes do valor de um produto ou serviço. O valor pode ser considerado como a utilidade e performance de um determinado produto ou serviço (benefícios) tendo em conta os custos associados.

No caso em particular, o produto permite aumentar a eficiência da realização do levantamento de campo e transmitir usabilidade e inovação através de métodos automatizados de recolha e visualização de dados cartográficos e segurança da integridade dos dados.

Valor percecionado

Numa perspetiva longitudinal de valor, a Tabela 1 revela os benefícios e sacrifícios relativos ao valor proporcionado pela organização.

Tabela 1 – Benefícios e Desvantagens para o cliente

Domínio	Produto
Benefícios	• *Software* de representação interativa de cenários cartográficos a três dimensões • Ações de adição, remoção e alteração de objetos cartográficos. • Alternativa de otimização do processo de levantamento de campo • Armazenamento de grandes quantidades de informação • Manutenção do software • Fiabilidade • Utilidade • Segurança e qualidade dos dados cartográficos • Benefícios operacionais • Diminuição de custos do processo de cartografia
Desvantagens	• Necessidade de ajuste do cenário tridimensional devido a problemas com GPS e Bússola • Necessidade dispositivos móveis e bateria • Custos de formação e aprendizagem • Custos de manutenção e suporte • Custos de tempo

2.3.3 Proposta de Valor

A proposta de valor é representada através de um produto móvel com uso de realidade aumentada que permite projetar cenários cartográficos a três dimensões com dimensões reais. Esta solução permitirá automatizar o processo de completagem de campo, diminuindo o tempo de execução do mesmo e de erros no controlo da informação.

A aplicação disponibiliza um controlo de ações de edição dos cenários, adição e remoção de objetos cartográficos e armazenamento completo de todos os dados. Os dados recolhidos são mais precisos do que segundo o método tradicional.

2.3.4 Cadeia de Valor

O conceito de cadeia de valor é a base da compreensão dos processos negócio e de maior relevância quando se trata de um negócio de cariz inovador no mercado. A cadeia de valor abrange desde a compra dos recursos até à entrega do resultado final ao cliente, e é constituída por um ou mais processos de negócio que por sua vez contêm processos e subprocessos com tarefas e atividades associadas [30]. Um processo de negócio caracteriza a atuação da organização para a concretização de um produto ou serviço de acordo com as necessidades do cliente ou do mercado. O processo é uma série de etapas que permitem através de recursos adicionar valor a um resultado. A Figura 9 exibe o modelo de cadeia de valor [31].

Figura 9 – Modelo da cadeia de valor

O modelo de Michael Porter permite compreender e analisar as atividades que efetivamente geram valor à empresa e vantagem competitiva. Estas atividades podem ser repartidas em primárias e de apoio.

As atividades primárias são conotadas como conjuntos de funções fundamentais relacionadas com o desenvolvimento dos produtos neste caso do *software* de realidade aumentada, possíveis vendas e manutenção do serviço. As atividades primárias poderão ser de origem logística de entrada, operacional, logística de saída, *marketing* e vendas e serviços.

Logística de entrada

A Logística de entrada representa os inputs ou recursos necessários para todas as atividades de negócio, neste caso em particular, o conjunto dos telemóveis para o teste da aplicação móvel, possíveis APIs para conversão de medidas e o kit de desenvolvimento de realidade aumentada (ARCore).

Operações

As operações são as atividades que convertem os recursos no produto ou serviço final, isto é, todas as etapas de desenvolvimento do software, as reuniões com os clientes para a análise dos requisitos, os testes físicos durante a recolha de campo.

Logística de saída

A logística de saída mostra todos os resultados finais que resultam das operações efetuadas pela organização representando no caso, a aplicação móvel de representação cartográfica e o armazenamento de informação num ficheiro de dados do tipo *GeoJson* na memória do dispositivo móvel.

Marketing e vendas

Este está associado ao processo de venda dos produtos e serviços e de métodos de promoção do mesmo através de plataformas de compra online, publicidade e atribuição de orçamentos.

Serviço

O serviço agrega todas as atividades que permitem aumentar o valor do software como o caso da instalação do software, possível formação, ajustes da usabilidade, adição de novos requisitos funcionais.

As atividades de apoio podem ser do tipo: aquisição, desenvolvimento de tecnologia, controlo de recursos humanos e a infraestrutura da empresa.

Infraestrutura

A infraestrutura da empresa é gerida segundo ações responsáveis por questões de qualidade e planeamento do software na completagem de campo. É também definida uma equipa de suporte e testes para continuadamente controlar e verificar fiabilidade da aplicação.

Gestão de Recursos Humanos

A gestão dos recursos humanos é feita no sentido de treinar e formar os responsáveis pela ação do levantamento de campo, bem como todos os outros intervenientes do processo de edição cartográfico que necessitam de aceder e interpretar os dados alterados. Também estão associadas atividades relativas ao desenvolvimento e suporte.

Desenvolvimento tecnológico

A recolha de campo é acompanhada por um conjunto de ferramentas e metodologias tecnológicas desde as tecnologias envolvidas na conceção e do produto em si, até os próprios métodos de manutenção, formação e suporte.

Aquisição/Compra

Este setor é responsável pela compra dos dispositivos móveis com suporte a realidade aumentada e ARCore e de *hardware* de suporte (como carregadores e *powerbanks* caso o processo de recolha campo exceda o limite da bateria dos telemóveis).

2.3.5 Modelo de Negócio Canvas

Segundo o negócio apresentado e de forma a descrever a lógica de como a organização cria e proporciona valor quer para a empresa, quer para os seus clientes, foi elaborado um modelo de negócio Canvas. O modelo de Canvas da Figura 10 é constituído pela identificação de parceiros da organização, atividades do negócio, recursos, proposta de valor, relação com os clientes, segmentação por clientes, canais de distribuição, custos e receitas.

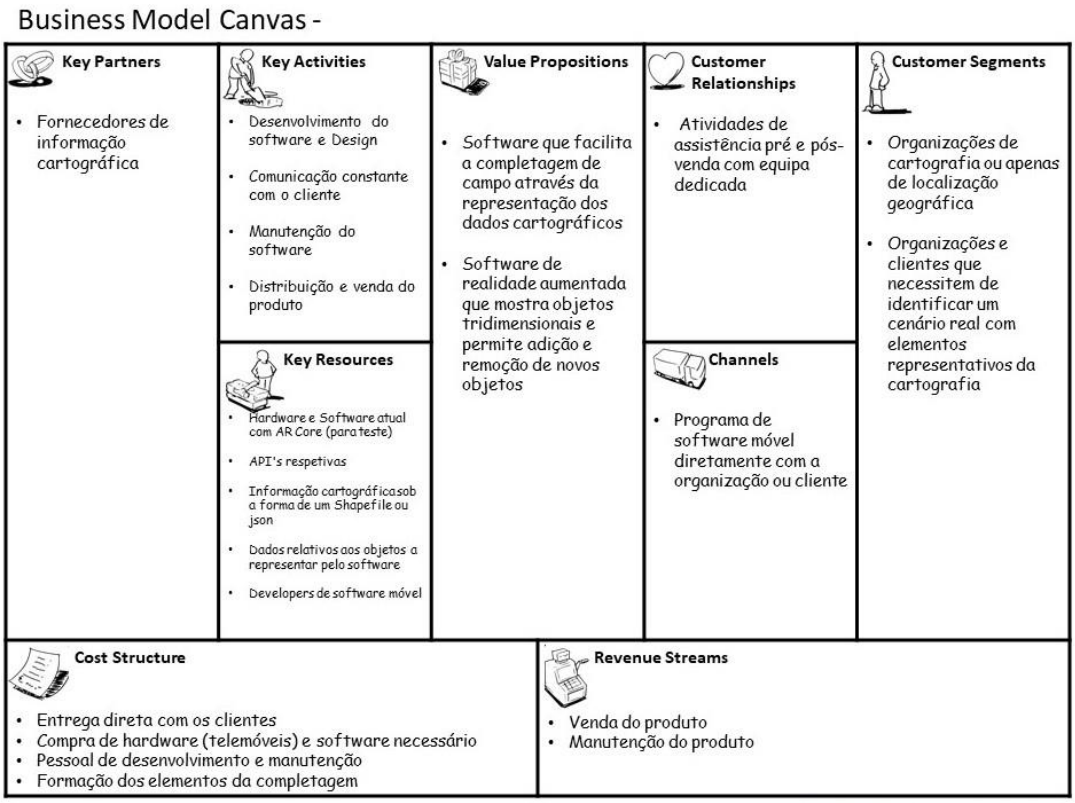


Figura 10 – *Business Model* Canvas

Parcerias

As parcerias representam a rede de fornecedores, sócios e colaboradores que contribui direta ou indiretamente com o negócio. Eventualmente, dependendo dos objetos que o cliente deseja representar nas cenas, poderá ser necessário obter dados ou informação relativas a determinados conteúdos específicos, como por exemplo, informação de comboios, edifícios, sistemas de canalização, etc.

Atividades

As atividades são as ações necessárias para a concretização eficiente e eficaz do negócio. As atividades de destaque são a produção da aplicação, comunicação com os clientes (*feedback* e dúvidas), manutenção da aplicação e distribuição e venda do produto.

Proposta de valor

A proposta de valor representa os produtos e serviços que a empresa cria e disponibiliza para os segmentos de clientes alvos. Estes trazem benefícios e valor para os clientes e, geralmente, são a solução para vários problemas.

Neste caso em particular, o valor da empresa surge do *software* que facilita o processo de completagem de campo de organizações de cartografia e/ou representação de cenários em três dimensões, de acordo com áreas predefinidas pelos clientes.

Relação com clientes

Tal como o nome indica, as relações com os clientes referem-se aos tipos de ligações estabelecidas com estes. As atividades serão de assistência pré e pós-venda com equipas dedicadas.

Canais de distribuição

Os canais de distribuição são os caminhos pela qual a empresa entrega o valor para o cliente. Além disso os canais contam também com a comunicação, vendas e compras dos produtos ou serviços e possível suporte e assistência. O *software* será distribuído fisicamente e individualmente a cada cliente e o suporte e a manutenção poderá ser via *email*, chamadas e videochamadas, ou em casos especiais, pessoalmente através de reuniões agendadas.

Segmentação

A segmentação representa os diversos grupos de pessoas ou organizações que têm interesse nos valores que a empresa vende. Neste caso, a segmentação refere-se às organizações e clientes que produzem mapas e dispõe de localização geográfica ou que necessitem de criar cenas tridimensionais específicas com elementos de áreas previamente definidas, como por exemplo, a representação de metro vias através de realidade aumentada.

Devido à fase prematura do projeto e do grau de inovação, a segmentação é relativamente restrita. Numa posterior fase de maturidade, a segmentação terá uma maior abrangência podendo usufruir de produtos e serviços de qualquer empresa que necessite de *software* de realidade aumentada ou geográfico.

Recursos

Os recursos são todos os ativos necessários para a realização de todo o negócio. É necessária uma equipa especializada no desenvolvimento e design do *software*, bem como dispositivos

móveis de teste com suporte a realidade aumentada (ARCore). *Software* de armazenamento e controlo de dados e, no caso de aplicações cartográficas, informação cartográfica sob a forma de *shapefiles* ou ficheiros *json* para a representação dos cenários cartográficos.

Receitas

As receitas são todos os recursos provenientes da possível venda de produtos e serviços aos segmentos escolhidos. As principais fontes de rendimento da organização são a venda do *software* e serviços de manutenção às organizações em particular.

Custos

Os custos do negócio são as equipas especializadas para a conceção e a consequente manutenção das aplicações, a formação necessária para o uso da aplicação na completagem de campo e o *harware* e *software* necessário para a concretização do produto.

2.3.6 Comparação de soluções

Métodos Multicritérios

Os métodos de decisão multicritério recorrem a técnicas numéricas que auxiliam os decisores a escolher uma opção de um conjunto de alternativas. Os métodos de apoio à decisão (AMD) permitem priorizar alternativas segundo critérios específicos.

O método adotado para avaliar as melhores tecnologias e soluções existentes foi o método de análise hierárquica (AHP). Este permite através da hierarquização do problema e de critérios qualitativos e quantitativos avaliar o processo de avaliação e as abordagens adotadas.

De modo a garantir a validade de comparação entre critérios foi definida uma homogeneidade entre os critérios do mesmo nível. O processo AHP é composto por sete fases distintas:

1. Construção da árvore hierárquica de decisão;
2. Comparação das alternativas e critérios;
3. Prioridade relativa de cada critério;
4. Avaliar a consistência das prioridades relativas;
5. Construção da matriz de comparação;
6. Obter prioridades;
7. Escolha das alternativas.

Este processo foi adotado a fim de identificar a melhor solução para a introdução de realidade aumentada na aplicação móvel. Os critérios revelados na Tabela 2 foram escolhidos com base nas características necessárias para a implementação do projeto e, de acordo, com os interesses da organização.

Tabela 2 - Definição dos critérios e alternativas (AHP)

Objetivo	Identificar o melhor kit de Realidade Aumentada
Critérios	Consumo, confiabilidade, ciclo de vida, funcionalidades e ambientes de desenvolvimento
Alternativas	ARCore, ARKit 2, Vuforia e Wikitude

A confiabilidade baseia-se na opinião dos consumidos relativamente ao número de falhas encontradas para cada kit; O consumo representa o número de consumidores de cada kit e o número de aplicações e bibliotecas compatíveis; As funcionalidades representam os tipos de deteção de superfícies, o tipo de processo de *rendering*, a introdução de modelos e adaptação à luminosidade; O ciclo de vida refere-se ao grau de maturidade de cada kit através da data de lançamento de cada um; Os ambientes de desenvolvimento, tal como o nome indica, representam o software onde são implementadas as soluções;

1. **Construção da árvore hierárquica de decisão**

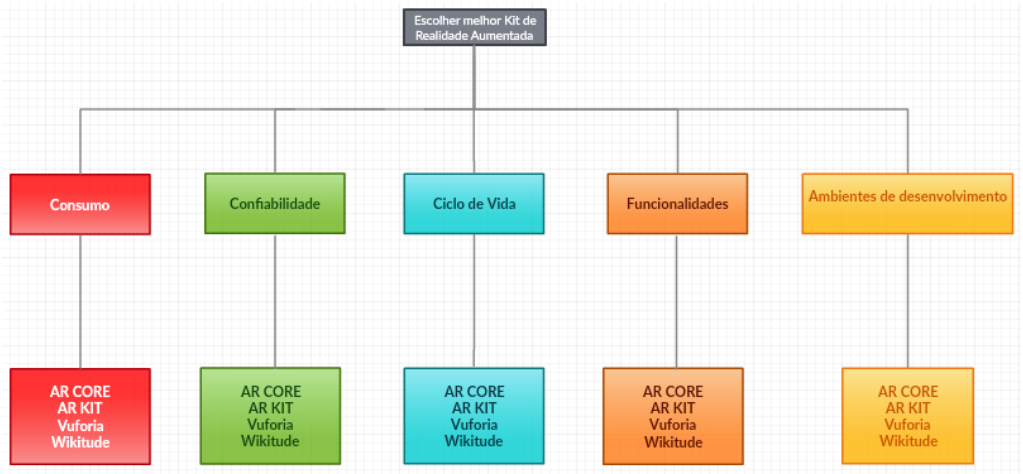


Figura 11 – Árvore hierárquica de decisão

2. **Comparação entre os elementos da hierarquia**

Na segunda fase foram estabelecidas as prioridades para os elementos de cada nível hierárquico (Tabela 3).

Tabela 3 - Prioridades de critérios

Critérios	Consumo	Confiabilidade	Ciclo de vida	Funcionalidades	Ambientes de desenvolvimento
Consumo	1	4	7	2	3
Confiabilidade	1/4	1	3	1/3	1/2
Ciclo de vida	1/7	1/3	1	1/6	1/4
Funcionalidades	1/2	3	6	1	2
Ambientes de desenvolvimento	1/3	2	4	1/2	1

3. Prioridade relativa de cada critério

Seguidamente foi normalizada a matriz comparação e calculada a prioridade relativa de cada critério (Tabela 4 e Tabela 5).

Tabela 4 - Soma de prioridades

Critérios	Consumo	Confiabilidade	Ciclo de vida	Funcionalidades	Ambientes de desenvolvimento
Consumo	1	4	7	2	3
Confiabilidade	1/4	1	3	1/3	1/2
Ciclo de vida	1/7	1/3	1	1/6	1/4
Funcionalidades	1/2	3	6	1	2
Ambientes de desenvolvimento	1/3	2	4	1/2	1
Soma	187/84	31/3	21	4	27/4

Tabela 5 - Cálculo de prioridades relativas

Critérios	Consumo	Confiabilidade	Ciclo de vida	Funcionalidades	Ambientes de desenvolvimento	Prioridade Relativa
Consumo	84/187	12/31	7/21	1/2	4/9	0.42281
Confiabilidade	21/187	3/31	3/21	1/12	2/27	0.10187
Ciclo de vida	12/187	1/31	1/21	1/24	1/27	0.04455
Funcionalidades	42/187	9/31	6/21	1/4	8/27	0.26939
Ambientes de desenvolvimento	28/187	6/31	4/21	1/8	4/27	0.16138

O peso calculado dos critérios:

- Consumo – 0.42281
- Confiabilidade – 0.10187
- Ciclo de vida – 0.04455
- Funcionalidades – 0.26939
- Ambientes de desenvolvimento – 0.16138

4. Avaliar a consistência das prioridades relativas

Seguidamente, é calculada a razão de consistência (RC) para garantir a consistência das prioridades tendo em conta grandes amostras de juízos aleatórios. Caso o RC seja superior a 0.1 é possível aferir que os julgamentos não são confiáveis, pois estão demasiado perto para o conforto de aleatoriedade.

Inicialmente, é necessário calcular o valor de λmax que representa o maior valor próprio da matriz. Considerar [Ax = λmax x] onde x é o vetor próprio.

$$\begin{bmatrix} 1 & 4 & 7 & 2 & 3 \\ 1/4 & 1 & 3 & 1/3 & 1/2 \\ 1/7 & 1/3 & 1 & 1/6 & 1/4 \\ 1/2 & 3 & 6 & 1 & 2 \\ 1/3 & 2 & 4 & 1/2 & 1 \end{bmatrix} \times \begin{bmatrix} 0.42281 \\ 0.10187 \\ 0.04455 \\ 0.26939 \\ 0.16138 \end{bmatrix} \approx \begin{bmatrix} 2.16506 \\ 0.51171 \\ 0.22415 \\ 1.37647 \\ 0.81895 \end{bmatrix}$$

$$\begin{bmatrix} 2.16506 \\ 0.51171 \\ 0.22415 \\ 1.37647 \\ 0.81895 \end{bmatrix} \approx \lambda\text{max} \begin{bmatrix} 0.42281 \\ 0.10187 \\ 0.04455 \\ 0.26939 \\ 0.16138 \end{bmatrix} \approx 5.0719$$

Uma vez calculado λmax, calcula-se o Índice de Consistência:

$$IC = \frac{\lambda max - n}{n - 1} = \frac{5.0719 - 5}{5 - 1} \approx 0.018$$

O n representa o número de critérios definidos (consumo, confiabilidade, ciclo de vida, funcionalidades e ambientes de desenvolvimento), que neste caso é 5.

O IR (índice aleatório) foi calculado para matrizes quadradas de ordem n pelo Laboratório Nacional de Oak Ridge (EUA). Os valores de IR podem ser observados na Figura 12 em função do número de critérios. Como o número de critérios é 5, o valor de IR é 1.12.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0.00	0.00	0.58	0.90	1.12	1.24	1.32	1.41	1.45	1.49	1.51	1.48	1.56	1.57	1.59

Figura 12 – Valores de IR para matrizes quadradas de ordem n

A razão de consistência é obtida da seguinte forma:

$$RC = \frac{IC}{IR} = \frac{0.018}{1.12} \approx 0.016$$

O IC representa o índice de consistência e o IR representa o índice aleatório.

Como o RC é de aproximadamente 0.016, ou seja, inferior a 0.1 é possível concluir que os julgamentos são confiáveis, logo os resultados obtidos são consistentes.

5. Construção da matriz de comparação

Na etapa seguinte foi construída uma matriz de comparação paritária para cada critério tendo em conta a importância relativa de cada uma das alternativas de realidade aumentada. A Tabela 6, Tabela 7, Tabela 8, Tabela 9 e Tabela 10 mostram os resultados obtidos.

Tabela 6 - Comparação paritária do critério Consumo

Consumo	ARKit *2*	ARCore	Vuforia	Wikitude	Prioridades relativas
ARKit *2*	1	1/5	2	1/3	0.11037
ARCore	5	1	7	2	0.42278
Vuforia	1/2	1/7	1	1/5	0.06336
Wikitude	3	1/2	5	1	0.38683
Soma	19/2	129/70	15	53/15	

Tabela 7 - Comparação paritária do critério Confiabilidade

Confiabilidade	ARKit *2*	ARCore	Vuforia	Wikitude	Prioridades relativas
ARKit *2*	1	2	5	4	0.48955
ARCore	1/2	1	4	3	0.30544
Vuforia	1/5	1/4	1	1/2	0.07862
Wikitude	1/4	1/3	2	1	0.12639
Soma	39/20	43/12	12	17/2	

Tabela 8 - Comparação paritária do critério Ciclo de vida

Ciclo de vida	ARKit *2*	ARCore	Vuforia	Wikitude	Prioridades relativas
ARKit *2*	1	2	1/2	1/3	0.16107
ARCore	1/2	1	1/3	1/4	0.09597
Vuforia	2	3	1	1/2	0.27714
Wikitude	3	4	2	1	0.46582
Soma	13/2	10	23/6	25/12	

Tabela 9 - Comparação paritária do critério Funcionalidades

Funcionalidades	ARKit *2*	ARCore	Vuforia	Wikitude	Prioridades relativas
ARKit *2*	1	2	7	4	0.50195
ARCore	1/2	1	6	3	0.31945
Vuforia	1/7	1/6	1	1/2	0.06110
Wikitude	1/4	1/3	2	1	0.11749
Soma	53/28	7/2	16	17/2	

Tabela 10 - Comparação paritária do critério Ambientes de desenvolvimento

Ambientes de desenvolvimento	ARKit *2*	ARCore	Vuforia	Wikitude	Prioridades relativas
ARKit *2*	1	1/4	2	1/5	0.10048
ARCore	4	1	6	1/2	0.32999
Vuforia	1/2	1/6	1	1/7	0.05910
Wikitude	5	2	7	1	0.51042
Soma	21/2	41/12	16	129/70	

Os Vetores de prioridade para cada um dos critérios e tecnologias pode ser observado na Tabela 11.

Tabela 11 - Vetores de prioridade (critérios/tecnologias)

Critério	Consumo	Confiabilidade	Ciclo de vida	Funcionalidades	Ambientes de desenvolvimento
ARKit *2*	0.11037	0.48955	0.16107	0.50195	0.10048
ARCore	0.42278	0.30544	0.09597	0.31945	0.32999
Vuforia	0.06336	0.07862	0.27714	0.06110	0.05910
Wikitude	0.38683	0.12639	0.46582	0.11749	0.51042

6. Obter prioridades

Nesta fase são multiplicados os valores de comparação de cada prioridade com os pesos dos critérios.

$$Vcp = \begin{bmatrix} 0.11037 & 0.48955 & 0.16107 & 0.50195 & 0.10048 \\ 0.42278 & 0.30544 & 0.09597 & 0.31945 & 0.32999 \\ 0.06336 & 0.07862 & 0.27714 & 0.06110 & 0.05910 \\ 0.38683 & 0.12639 & 0.46582 & 0.11749 & 0.51042 \end{bmatrix} \times \begin{bmatrix} 0.42281 \\ 0.10187 \\ 0.04455 \\ 0.26939 \\ 0.16138 \end{bmatrix} = \begin{bmatrix} 0.25515 \\ 0.35346 \\ 0.07314 \\ 0.31121 \end{bmatrix}$$

Prioridades compostas para cada alternativa

ARKit2 ≈ 0.25515

ARCore ≈ 0.35346

Vuforia ≈ 0.07314

Wikitude ≈ 0.31121

7. Escolha da alternativa

Com base nos resultados obtidos, dos interesses da organização e em função dos critérios e prioridades identificadas a ferramenta de realidade aumentada a adotar é o ARCore da Google com um valor de aproximadamente 0.35.

3 Avaliar soluções/tecnologias existentes

O terceiro capítulo retrata as abordagens e as tecnologias adotadas durante a realização do projeto, destacando as vantagens comparativamente com as outras soluções existentes.

Inicialmente, identificam-se as forças e fraquezas de cada tipo de representação cartográfica, tendo em consideração os interesses da organização.

No contexto das tecnologias, descrevem-se as ferramentas e serviços de apoio à realidade aumentada, geolocalização, conversão de coordenadas e criação e desenho de figuras e modelos.

Ainda neste capítulo, são descritos os dois ambientes base para o desenvolvimento da aplicação móvel, visualização e verificação dos dados cartográficos.

3.1 Abordagens adotadas

De modo a determinar as melhores abordagens para a representação de cenários cartográficos houve uma análise detalhada de cada proposta, identificando as vantagens e desvantagens de cada uma, comparando-as entre si.

A representação bidimensional, apesar das suas vantagens evidentes de custos de memória, *rendering* reduzido e facilidade de representação da cena segundo apenas dois eixos, foi descartada como ambiente principal da aplicação, devido à dificuldade de apresentação e compreensão de todos os objetos cartográficos, bem como da falta de noções de profundidade e altitude. No entanto, este tipo de representação foi também adotada, visto que os cenários tridimensionais por si, não permitem um bom enquadramento dos utilizadores no meio. A Tabela 12 mostra as vantagens e desvantagens desta solução.

Tabela 12 - Vantagens e desvantagens da representação bidimensional

Vantagens	Desvantagens
Custos de produção mais baixos	Falta de noções de altitude e profundidade
Memória de *rendering* reduzida	Dificuldade de interpretação dos objetos
Facilidade de representação segundo 2 eixos	Menor interatividade
Facilidade de enquadramento	Maior probabilidade de erros na disposição dos objetos cartográficos

O uso de modelação a duas dimensões também poderia causar falhas na interpretação do tipo de objetos, não permitindo a distinção de um jardim, uma casa, uma torre, considerando que todos eles seriam quadrados ou retângulos.

A representação segundo vistas panorâmicas apesar da sua interatividade evidente e de facilidade de enquadramento dos utilizadores no meio envolvente, não permite a introdução, edição e remoção de objetos cartográficos visto que são conjuntos de fotografias. A visualização de cenários torna-se mais simples e clara, contudo com menor poder de alteração. A Tabela 13 comprova as vantagens e desvantagens enunciadas.

Tabela 13 - Vantagens e desvantagens da representação por vistas panorâmicas

Vantagens	Desvantagens
Maior interatividade	Não permite ações de edição de cenas
Facilidade de enquadramento	Custos elevadíssimos na recolha de dados quer de tecnologia, recursos humanos e tempo
Vistas reais das cenas	Dificuldade em obter valores de localizações (latitude, longitude e altitude)

A representação tridimensional apresenta um elevado grau de viabilidade. As desvantagens desta abordagem são muito poucas comparativamente com os pontos fortes e, estão dependentes, não só dos métodos e metodologias de programação gráfica, como do *hardware* escolhido. O custo de memória é o principal ponto fraco desta solução e uma das vantagens visíveis é a facilidade de representação de todos os objetos cartográficos considerando os 3 eixos, latitude, longitude e altitude (Tabela 14).

Tabela 14 - Vantagens e desvantagens da representação tridimensional

Vantagens	Desvantagens
Representação adequada segundo três eixos	Custos de produção mais altos
Maiores noções de localização geográfica	Memória de *rendering* alta
Facilidade de alteração das cenas	Dificuldade em representar sólidos muito complexos
Mais realista	
Melhor qualidade	
Maior interação comparado com 2D	

De entre os quatro tipos possíveis de visualização, a abordagem que melhor se adequa é a representação gráfica tridimensional suportada pela realidade aumentada, pois, para além de contemplar todas as vantagens adjacentes à simples tridimensionalidade, possui uma maior interatividade com os utilizadores, representa as cenas segundo dimensões reais e apresenta localizações mais precisas relativamente à visualização 3D. A Tabela 15 mostra as vantagens que levaram à escolha deste tipo.

Tabela 15 - Vantagens e desvantagens da representação com realidade aumentada

Vantagens	Desvantagens
Vantagens das cenas 3D simples	Custos de produção mais altos
Maiores noções de localização geográfica relativamente ao simples 3D	Memória de rendering elevadíssima
Dimensões e espaços realistas	Menor número de objetos apresentados em relação ao 2D e 3D
Tecnologia em expansão	
Maior número de funcionalidades e finalidades	
Maior interação entre o sistema e os utilizadores	
Maior facilidade de introdução de sistemas de localização reais e de orientação (GPS e Bússola)	

A cena tridimensional é desenhada com base numa *shapefile* fornecida pela organização na qual contém todos os objetos cartográficos de um determinado local. Para além de coordenadas, este ficheiro tem, também informações sobre os vários objetos, as categorias (ex: construção em geral ou áreas verdes em geral) e tipos (pontos, linhas ou polígonos).

3.2 Tecnologias adotadas

3.2.1 Tecnologia de realidade aumentada

A análise multicritério de apoio à decisão, que envolveu vários critérios e graus de importância, permitiu concluir que o ARCore é, neste caso, a tecnologia de realidade aumentada mais adequada. Resumidamente, esta tecnologia foi considerada a mais adequada, na medida em que contempla grande parte das funcionalidades de *mapping* atualmente disponíveis, abrangendo um maior número de população interessada neste ramo (utilizadores e desenvolvedores) e de plataformas tecnológicas (ambientes de desenvolvimento e dispositivos móveis que suportam esta tecnologia) [32].

O ARCore possui um conjunto de características que possibilitaram o desenho de cenas cartográficas, incluindo desde pontos representativos de objetos de reduzida área: postes ou tampas de saneamento, até aos sólidos mais complexos, como prédios, monumentos e pavilhões. Esta ferramenta contempla duas APIs que foram extensivamente utilizadas para a representação gráfica: ARCore *API* e SceneForm API.

A ARCore *API* é responsável por todo o processo interno de *motion tracking* e de interpretação do ambiente através da deteção de superfícies através da câmara dos dispositivos. O ARCore gere e controla *frames* que funcionam como *mocks* (objetos usados no desenvolvimento de *software* que simulam o comportamento de objetos reais) e identifica planos horizontais e verticais. Contudo, a representação da cena baseia-se apenas segundo um plano horizontal virtual.

De modo a garantir a fiabilidade da aplicação e a componente real, o plano virtual terá de estar sob o chão. Depois de detetada e devidamente escolhida pelo utilizador, é atribuída uma âncora virtual. Esta âncora representa o ponto a partir do qual toda a cena cartográfica vai incidir. A âncora permanece exatamente na mesma posição independentemente da movimentação e orientação do telemóvel.

A SceneForm *API* foi a base para o desenvolvimento gráfico da aplicação. Os componentes principais desta interface são a estruturação da cena em nós, o *rendering* de toda a cena e a matemática associada à definição das posições e orientação de cada nó da cena. Cada objeto cartográfico possui pelo menos um nó. A estruturação da cena é baseada segundo uma hierarquia de nós, sendo o nó principal a âncora a partir do qual tudo começa. Esta tecnologia suporta o *rendering* dos modelos (modelos importados ou originados manualmente), vistas a três dimensões, materiais, luzes, texturas que tiveram especial importância na criação das

figuras e sólidos geométricos. Por fim, as funções matemáticas permitiram localizar corretamente todos os elementos da cena e respetivas orientação/rotações.

3.2.2 Conversão de coordenadas

O posicionamento dos objetos na cena só estará de acordo com a realidade caso sejam realizadas conversões entre coordenas geográficas. Segundo a metodologia e o desenvolvimento de cartografia da organização, as coordenadas da *shapefile* recebida encontram-se em WGS-84 e o sistema de coordenadas das cenas de realidade aumentada estão em metros. Este exemplo em concreto levou à necessidade de procurar uma biblioteca ou uma API que realizasse conversões para diferentes tipos de coordenadas. Considerando o GeoTools, JMapProjLib e o Proj4J das ferramentas mais usadas para conversões foram realizados testes para selecionar o melhor sistema.

Devido a problemas de incompatibilidade desta biblioteca com o Android Studio não foi possível considerar esta tecnologia.

O JMapProjLib é uma versão reduzida do Proj4J e por essa razão nem todas as projeções estão implementadas e há falta de suporte na conversão de dados. Pelo facto desta versão conter alguns erros e de não suportar uma grande variedade de projeções, concluiu-se que esta biblioteca também não é válida.

Por fim, foi testada a ferramenta Proj4J. Esta biblioteca Java contempla todos os sistemas de coordenadas necessários bem como projeções e transformações. Os sistemas de coordenadas a utilizar para as conversões são: WGS-84 4326, 3857 e 54004. O sistema 4326 é representado em graus, (latitude e longitude) e o os sistemas 3857 e 54004 encontram-se em metros com projeções *pseudo-mercator* e *Mercator* respetivamente [33]. Concluindo, a Proj4J é a biblioteca que mais se adequa às necessidades propostas.

3.2.3 Sistema de Localização

Os objetos para serem inseridos nas posições reais têm de se basear num sistema de localização do dispositivo móvel (GPS). Considerando o ambiente de desenvolvimento Android Studio e a realidade aumentada ARCore foi escolhida uma API de serviços de localização da Google (Google Location Services). Esta API disponibiliza os dados necessários para todo o processo, como a localização do telemóvel (latitude, longitude e altitude), direções (*bearing* e *azimuth*) e orientação do telemóvel (*portrait* e *landscape*). Além disso, apresenta uma estrutura bem definida, simples e bastante acessível. As informações do sistema GPS são acedidas pela API atendendo à aceitação por parte do utilizador do dispositivo.

A Google Location Services fornece dados de localização bastante precisos e atualizados. Além do *tracking*, este serviço possibilita a delimitação de áreas geográficas e o reconhecimento de atividades automatizadas.

3.2.4 Construção de polígonos a partir de triângulos

O kit ARCore disponibiliza três métodos de construção de sólidos geométricos: cubos, esferas e cilindros. Porém os objetos reais, na maioria dos casos, são irregulares, o que levou à necessidade de alterar o tipo de abordagem de modelação.

Foram realizados testes funcionais para cada abordagem/tecnologia (*openGL ES* e *graphics Android*), tendo-se verificado que ambas tinham limitações no tamanho máximo de polígonos e *views*, além de problemas consideráveis de *rendering*, com a introdução de um número médio de objetos. Assim, foi adotada a abordagem primitiva de representação de sólidos onde os sólidos geométricos seriam divididos em polígonos e os polígonos em conjuntos de triângulos. A identificação dos triângulos foi suportada pela biblioteca Earcut.

A Earcut *library* implementa um algoritmo baseado em *Fast Industrial-Strength Triangulation of Polygons* (FIST) e *Triangulation by Ear Clipping* que gera triangulação de polígonos e indica-os segundo uma ordem específica.

O método de *Ear Clipping* consiste na divisão de polígonos em triângulos a partir da identificação de dois triângulos nas duas extremidades do polígono (orelhas) [34]. À medida que são identificadas as orelhas do polígono são removidas até restar apenas um triângulo, este método pode ser observado na Figura 13 [35]. Este funcionamento só ocorre caso as figuras geométricas não possuam espaços vazios.

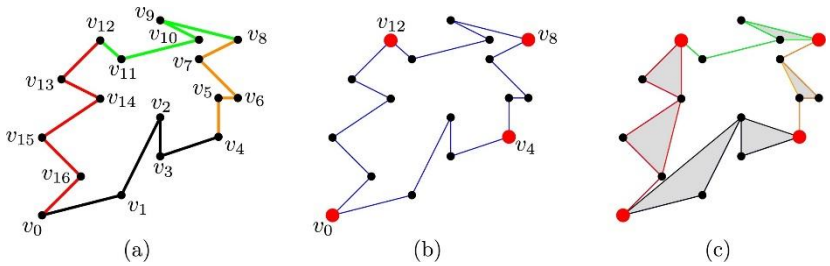


Figura 13 – Algoritmo Ear Clipping

O algoritmo FIST baseia-se no *Ear Clipping*, contudo é mais rápido e inclui polígonos com espaços vazios e a três dimensões.

3.2.5 *GeoJSON*

É um formato de representação geográfica simples baseado em *JSON*. Este formato pode ser constituído por:

- Pontos (locais e endereços);
- conjunto de linhas (ruas, estradas, entre outros);

- polígonos (para representações de edifícios, parques, até mesmo países);
- conjuntos misturados (junção dos vários tipos).

Cada objeto cartográfico é caracterizado por uma categoria mencionada anteriormente e pelas suas coordenadas respetivas.

3.2.6 Visualização de dados

No decorrer do desenvolvimento do projeto é necessário verificar os dados dos ficheiros *geojson*. O exemplo adotado para a visualização desses ficheiros é o OpenLayers.

O OpenLayers é uma biblioteca Javascript *open-source* que tem como finalidade principal representar mapas dinamicamente em qualquer página *Web*, recebendo informação geográfica de vários tipos. É constituída por uma API para construção de aplicações geográficas semelhantes ao Google Maps e ao Bing Maps. O *software* suporta vários tipos de representação de dados, a saber: *GeoRSS, Geography Markup Language* e *GeoJSON*.

Esta biblioteca mostra os vários pontos, linhas e polígonos representativos dos objetos cartográficos que irão ser guardados, alterados ou apenas apresentados a três dimensões na aplicação. A Figura 14 mostra uma visão dos dados de uma determinada zona a partir do OpenLayers.

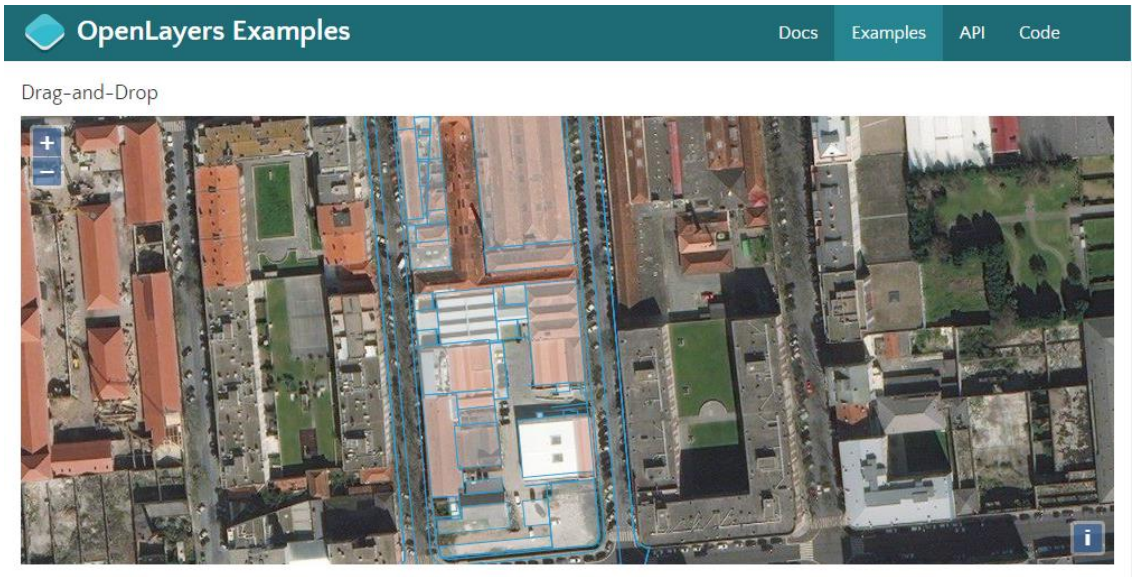


Figura 14 - Representação de dados de uma zona em OpenLayers

3.2.7 Ambiente de desenvolvimento

O ambiente escolhido para a conceção do produto foi o Android Studio, visto que é uma das plataformas mais utilizadas para desenvolvimento de software móvel com mais suporte.

O Android Studio é um ambiente de desenvolvimento de *software* exclusivo para dispositivos Android. A linguagem Java foi a escolhida de entre as duas possíveis (Java e Kotlin), dado à familiarização com a linguagem, número de *packages* existentes e maior compatibilidade com outras tecnologias.

A tecnologia contempla um SDK com bibliotecas e ferramentas de desenvolvimento que facilitam o desenvolvimento de *software*, sendo compatível com Windows, Linux e Mac OS X.

Caraterísticas do Android Studio:

- Sistema de criação flexível através do suporte para compilações em Gradle;
- Reparações rápidas e eficazes;
- Ferramentas Lint para controlo de performance, usabilidade, compatibilidade de versões entre outros;
- Capacidade de atribuição de assinaturas nas aplicações;
- Editor de *Layout* visual;
- Suporte para Emulador.

3.2.8 Linguagem R

O R é um ambiente de computação e uma linguagem de programação que permite analisar, manipular e visualizar graficamente os dados. Neste caso, esta linguagem foi adotada para representar graficamente os resultados dos inquéritos. O código implementado para obter estes resultados encontra-se nos anexos.

4 Design da Solução

O Design do projeto serve de base para estruturar o procedimento adotado bem como identificar os requisitos funcionais e não funcionais. Além disso, é esboçado um modelo de domínio, um diagrama de componentes e um fluxograma de modo a descrever os componentes do *software*, as suas dependências e interfaces. Todo o design planeado tem em consideração a solução dos problemas da fase de completagem de campo.

Além do design arquitetural da solução são também apresentados alguns exemplos de interfaces gráficas da aplicação (*mockups*).

A elaboração da arquitetura e a conceção dos modelos de design são suportados por padrões, regras e boas práticas do desenvolvimento de *software*.

4.1 Requisitos funcionais

Os requisitos funcionais contribuíram para a divisão do sistema em tarefas, caracterizadas por um conjunto de ações em resposta às necessidades dos clientes. Ao contrário dos funcionais, os requisitos não funcionais ou requisitos de qualidade não especificam resultados particulares, antes caraterísticas gerais, como o caso da usabilidade e desempenho da aplicação.

Os requisitos funcionais podem ser observados através de um diagrama de casos de uso, indicado na Figura 15. Esta ferramenta permite obter uma visão externa do sistema e mostrar graficamente os atores envolvidos e os vários requisitos funcionais.

Este diagrama tem também especial importância para definir os aspetos executáveis do produto e detetar problemas de configuração de software através das ligações e dependências.

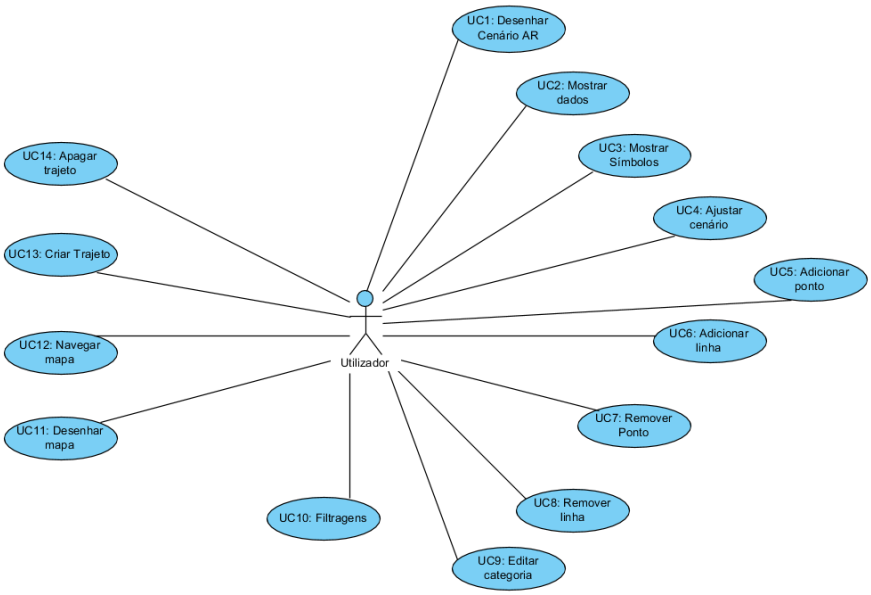


Figura 15 – Diagrama de casos de uso

Devido ao facto dos casos de uso das projeções (cena e mapa) serem demasiado extensos e complexos, criou-se a necessidade de reparti-los em pequenas tarefas de modo a facilitar a compreensão do negócio e a implementação destes casos de uso. A Tabela 16 mostra as tarefas para cada um dos casos de uso.

Tabela 16 – Repartição de casos de uso

Cenário Tridimensional	Mapa
Identificação das coordenadas geográficas do telemóvel;	Identificação do centro do mapa;
Leitura da lista de objetos cartográficos da *Datafile;*	Desenhar lista de objetos cartográficos;
Conversão de coordenadas geográficas dos objetos e da posição do telemóvel (EPSG 4326 em ESRI 3857);	Adicionar utilizador;
Transformação da matriz das posições (translação e rotação)	Adicionar orientação do utilizador;
Redução do número de objetos a desenhar (alcance);	Adicionar contorno para os objetos editados;
Inserção dos objetos no cenário;	

UC1 – Desenhar cenário AR

O primeiro caso de uso mostra para o utilizador um cenário tridimensional com os dados cartográficos disponibilizados pelo ficheiro *Datafile.* Este ficheiro é o resultado de uma conversão de uma *shapefile* (ficheiro que suporta a produção de cartografia na organização) em *geojson*. As cenas em ARCore são inseridas num sistema de eixos x,y,z sendo a unidade de medida, o metro. Dado que as unidades de medida da localização do utilizador e dos objetos cartográficos se encontram em graus é necessário fazer uma conversão entre sistemas de coordenadas. Seguidamente, as posições dos objetos são transformadas tendo em conta o novo sistema de eixos e de coordenadas. O alcance é um valor estático defino com 15 metros de raio e permite restringir o número de objetos a desenhar. Por fim o sistema mostra ao utilizador os objetos inseridos no mundo real com as devidas formas, posições e cores.

UC2 – Mostrar dados cartográficos de um objeto

O sistema mostra a categoria e o símbolo do objeto selecionado pelo utilizador. Sempre que o utilizador seleciona um objeto, a cor deste muda. A informação é apresentada dentro de uma caixa. Não é possível selecionar dois objetos ao mesmo tempo.

UC3 – Mostrar símbolo de cada objeto

O utilizador pode ver ainda no ambiente de realidade aumentada os símbolos dos vários objetos. Estes símbolos estão definidos segunda uma nomenclatura própria de cartografia e são representados por siglas a duas dimensões. A posição e a inclinação dos símbolos variam consoante a posição do utilizador e o tipo de objeto. Este sistema de posicionamento está pormenorizado no capítulo seguinte.

UC4 – Ajustar cenário

O ajuste da cena é fundamental para a visualização correta dos dados no cenário. O ajuste é constituído por uma componente de translação ou *move* e por uma componente de rotação. O utilizador poderá mover a cena através da sua seleção e poderá rodá-la através de um simples *slider* (*seekbar*).

UC5 – Adicionar ponto

O utilizador tem a possibilidade ainda de editar os objetos apresentados. Uma das funcionalidades é a adição de pontos. Estes pontos são mostrados através de cilindros de pequenas e grandes dimensões. O utilizador seleciona uma posição na cena e identifica a categoria do cilindro. As categorias são mostradas numa lista.

UC6 – Adicionar linha

A adição de linhas é semelhante ao processo de adição de pontos com a particularidade de poder adicionar mais do que um ponto na cena, formando assim os vários segmentos de linha. Estes segmentos são pequenos paralelepípedos com alturas muito pequenas. Cada conjunto de

dois pontos forma um segmento e a lista de categorias é atualizada conforme o tipo de objeto a inserir.

UC7 – Remover ponto

O caso de uso de remoção de ponto é concretizado quando o utilizador seleciona um objeto. O sistema revela ainda pequenos alertas de modo a prevenir o utilizador de remoção de pontos indesejados. Apenas objetos do tipo "Ponto" poderão ser selecionados durante o processo.

UC8 – Remoção de linhas

As linhas podem ser também apagadas seguindo o mesmo princípio do caso de uso anterior.

UC9 – Editar categoria

O utilizador tem ainda a possibilidade de alterar a categoria de determinado objeto. A lista de categorias mostra apenas as categorias do tipo do objeto selecionado. Apesar do utilizador apenas poder adicionar e remover objetos do tipo ponto e linha, é possível alterar a categoria de objetos polígonos.

UC10 - Filtragens

Este caso de uso tem como principal função filtrar a informação que é desenhada, quer na cena AR, como no mapa. À medida que são acrescentados objetos à cena, a lista das filtragens vai obviamente aumentando. O utilizador poderá selecionar as categorias que deseja visualizar, podendo ainda visualizar todos os objetos ou nenhum.

UC11 – Desenhar mapa 2D

O UC11 mostra um mapa com os mesmos objetos cartográficos do cenário, porém a duas dimensões. Os cilindros são agora substituídos por pontos e os paralelepípedos representativos das linhas e polígonos são agora linhas abertas e fechadas com ou sem preenchimento. A construção do mapa é feita a partir da identificação do seu centro. Uma vez mais, as coordenadas para cada objeto são alteradas devido ao novo sistema de coordenadas. O utilizador pode ainda ver a sua localização (ponto) no mundo real bem como a sua orientação (cone). Esta projeção tem a vantagem de exibir os objetos alterados ou adicionados durante a completagem através da inserção de um contorno.

UC12 – Navegar no mapa (*zoom* e *drag*)

A navegação do mapa pode ser feita segunda duas formas: *zoom* e *drag*. O *zoom* é detetado quando o utilizador desloca dois pontos de referência na tela. O utilizador pode ampliar ou reduzir o mapa. Além disso, este pode ainda mover o mapa através de apenas um ponto de referência.

UC13 – Criar trajeto

O mapa mostra ainda o trajeto realizado durante a completagem. O início e o término do trajeto são escolhidos pelo utilizador. Existem também as opções de pausar, parar e reiniciar o desenho do trajeto.

UC14 – Apagar trajeto

Por fim, o caso de uso "Apagar trajeto" é conseguido quando o utilizador seleciona o botão de remover. A partir deste momento o mapa deixa de apresentar o trajeto.

Os requisitos não funcionais foram identificados segundo determinados padrões, caracterizando as componentes de usabilidade, eficiência e adaptabilidade. Todas estas questões foram abordadas e avaliadas no QEF (*Quantitative Evaluation Framework*) do capítulo Experimentação e Avaliação da solução.

4.2 Modelo de Domínio

Em engenharia de software o modelo de domínio é construído segundo uma componente lógica com o intuito de organizar as entidades e mostrar o relacionamento e comportamento entre elas. Tendo em conta que a aplicação FieldAR representa uma dimensão mais gráfica, a sua estrutura de entidades é bastante simplificada e pode ser observada na Figura 16.

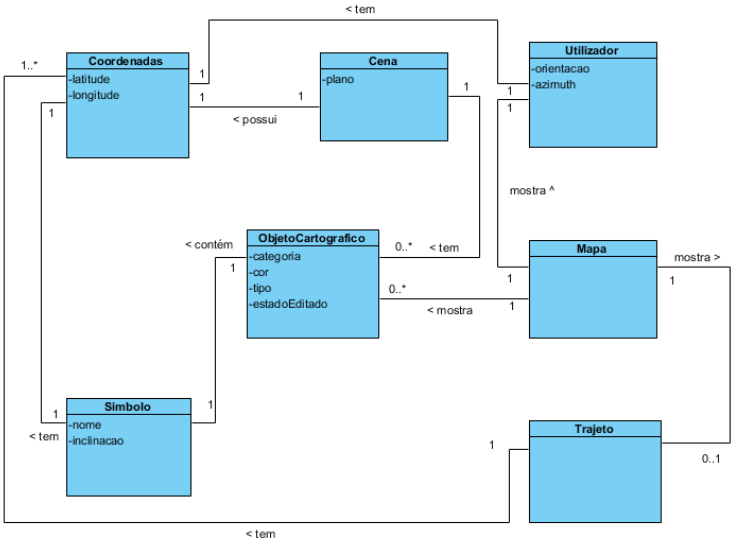


Figura 16 – Modelo de domínio

A cena é constituída por um conjunto de coordenadas (longitude e latitude), por um plano e por uma lista de objetos cartográficos.

Um objeto cartográfico é composto por uma categoria, uma cor, um tipo de objeto (ponto, linha ou polígono), um estado de edição e um símbolo. O estado de edição indica se objeto sofreu alguma alteração, se é um novo objeto ou se é um objeto inicial. Cada símbolo contém ainda um conjunto de coordenadas e uma inclinação. A inclinação pode ser apenas horizontal ou vertical dependendo do tipo de objeto.

O mapa desenha um trajeto, um utilizador e a mesma lista de objetos cartográficos do cenário. Um utilizador para além de ter um conjunto de coordenadas apresenta ainda uma orientação.

4.3 Arquitetura de componentes

Com o objetivo de especificar e documentar o sistema segundo componentes foi concebido um diagrama de componentes. Este artefacto foi dividido por vários grupos funcionais de alto nível. Cada componente do diagrama da Figura 17 apenas se relaciona com outros componentes pela necessidade de obter conhecimento para a realização de tarefas.

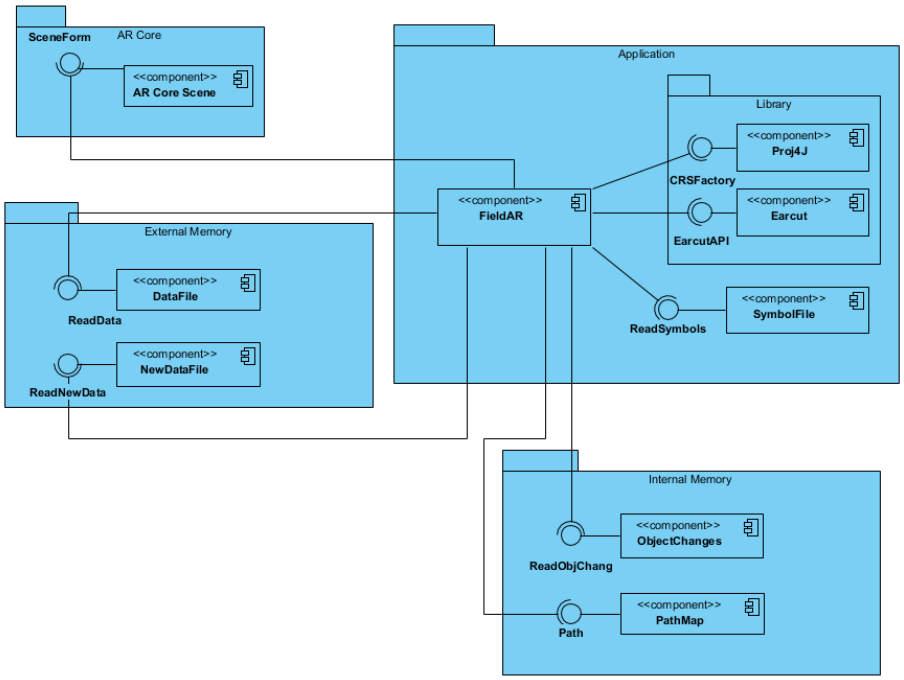


Figura 17 – Diagrama de componentes

O sistema da aplicação é constituído por quatro componentes principais: aplicação, ARCore, memória interna e externa.

Dentro da aplicação, o FieldAR representa o elemento que controla todas as atividades da aplicação, o *SymbolFile* providencia todos os dados relativos aos tipos de símbolos e categorias existentes e as bibliotecas internas Proj4J e Earcut são constituídos

respetivamente por APIs de conversão de sistemas de coordenadas geográficas e construção de polígonos.

O projeto ARCore é composto também por uma API que disponibiliza as interfaces de inserção das cenas de realidade aumentada para o FieldAR, entre outras funcionalidades.

Por fim, foram adicionadas duas componentes físicas de puro armazenamento (memória interna e externa) e são apenas apresentadas para compreensão física do negócio. Os ficheiros de dados relativos aos dados cartográficos antes e depois da edição (*Datafile* e *NewDataFile*) são guardados e acedidos na memória externa do dispositivo móvel e os documentos de armazenamento dos objetos alterados e dos trajetos são guardados na memória interna.

4.4 Diagrama de fluxo geral

Um diagrama de fluxo é uma representação visual da navegação possível do utilizador pelo sistema. Este diagrama demonstra como será a estrutura de navegação entre ações, decisões e funcionalidades da aplicação. O diagrama de fluxo, também conhecido como fluxograma ou diagrama de navegação, pode ser visto na Figura 18. Este fluxograma mostra apenas as funcionalidades gerais com um elevado grau de abstração.

Inicialmente, o utilizador necessita de dar permissões de acesso à câmara e ao ARCore antes de iniciar verdadeiramente a aplicação. Caso este não ceda a aplicação é terminada. Caso aceite, o utilizador seleciona um plano através da câmara e da realidade aumentada.

Seguidamente, é gerado todo o ambiente principal da aplicação onde é mostrado o cenário tridimensional com todos os objetos cartográficos com os devidos modelos, simbologias e dados. Além do cenário, o utilizador tem três opções de ações onde pode selecionar objetos do cenário, aceder ao menu ou ao mapa. Caso o utilizador aceda ao menu este tem a opção de filtrar os dados do cenário e do mapa ou editar o cenário através de um leque de opções de edição. Se o utilizador escolher a opção do mapa este poderá ainda ver o mapa, criar ou apagar um trajeto/percurso.

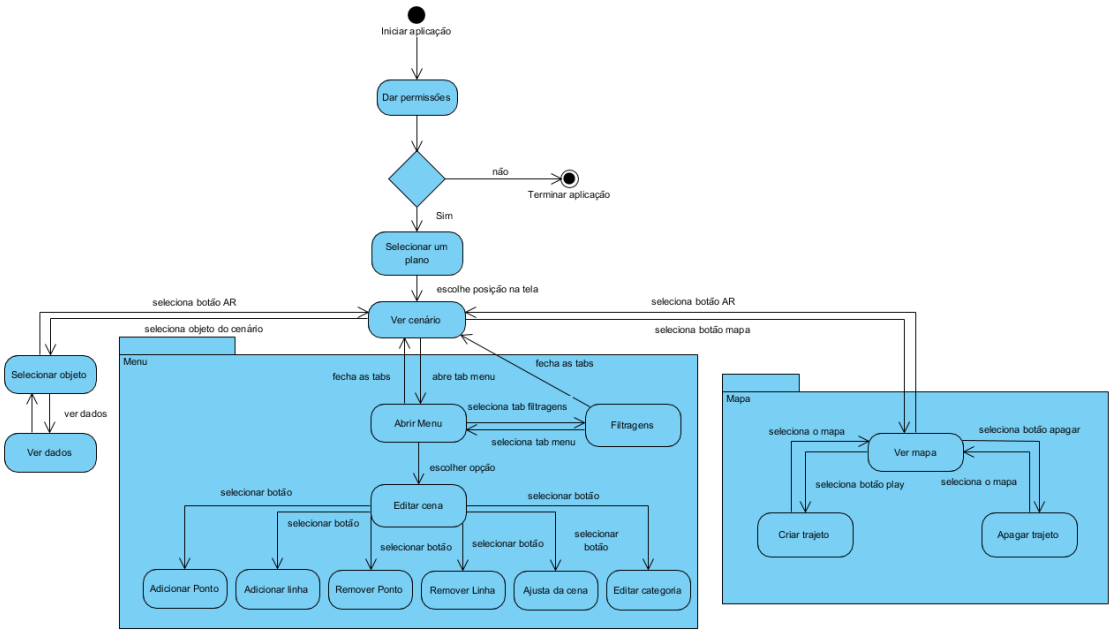


Figura 18 – Diagrama de fluxo

Além do diagrama geral foi ainda gerado mais um fluxograma muito simples (ver Figura 19) para as várias ações de edição. Tal como mencionado anteriormente, existem seis opções de edição da cena: o ajuste do cenário, adicionar ponto ou linha, remover ponto ou linha e editar categoria.

O ajuste de cena é feito quando o utilizador seleciona a cena e depois a move ou a rode conforme os seus interesses.

A adição de pontos e linhas é conseguida quando o utilizador adiciona um objeto na cena, e de seguida tem a opção de cancelar onde é direcionado novamente para a cena tridimensional, tendo a opção de mover objeto, caso este não se encontre na posição pretendida e tendo a opção de escolher logo, a categoria do objeto.

A remoção dos pontos e linhas é feito de através da seleção de um objeto. Se o utilizador pretender pode cancelar a ação e volta diretamente para a cena principal.

A edição da categoria requer que o utilizador selecione um objeto e escolha de seguida a nova categoria.

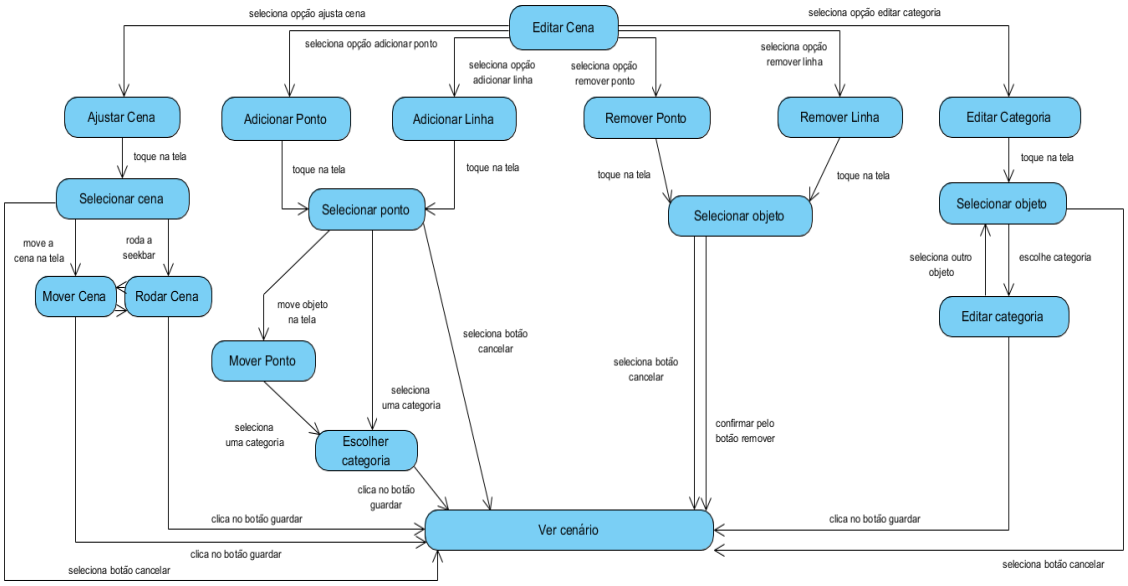


Figura 19 – Fluxograma relativo às ações de edição

4.5 *Mockups*

Além do design das várias componentes do negócio e do fluxo de navegação da solução também foi feito um esboço das interfaces gráficas principais da aplicação móvel. Estes *mockups* foram criados ao longo do projeto e basearam-se nas interfaces básicas já existentes.

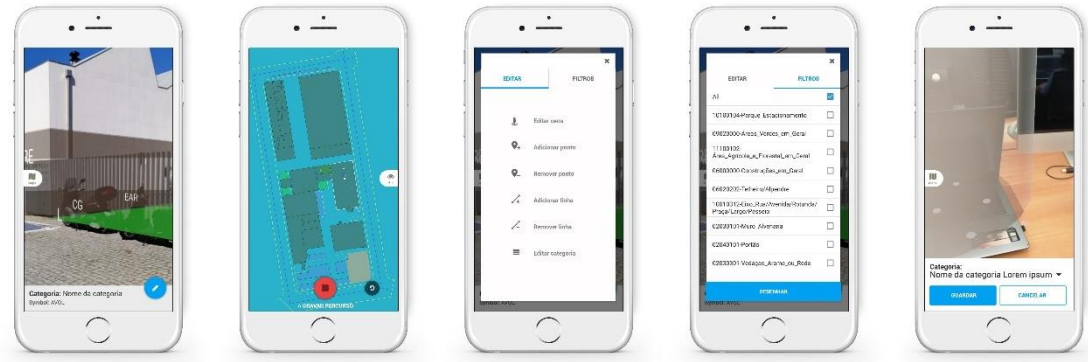


Figura 20 – Mockups da aplicação

A Figura 20 revela as interfaces gráficas do menu inicial com as cenas de realidade aumentada, do mapa, dos menus de edição e filtragens, e um exemplo de uma ação de edição.

O menu principal revela o ambiente de realidade aumentada com a informação de um objeto cartográfico, com um botão flutuante representativo do menu de edição e um botão que faz referência ao mapa.

No segundo *mockup* é mostrado o mapa sobre fundo azul com a localização e orientação do utilizador, com um botão referente ao cenário de realidade aumentada, com um sistema de controlo de trajetos. Este sistema é constituído por um botão que desenha, pausa ou para o trajeto no mapa e um botão que o remove. Mediante a ação do botão é mostrada uma pequena informação por baixo.

A terceira e a quarta imagem identificam respetivamente as *tabs* de edição da cena e filtragens de informação. Na primeira são mostradas todas as ações de edição do cenário juntamente com um ícone representativo. A *tab* das filtragens mostra uma lista com todas as categorias existentes no cenário e o botão de desenho. Além disso, o utilizador tem a opção de selecionar todas as categorias através da opção "*All*".

Por fim, é apresentado um exemplo de adição de um objeto no cenário com uma lista das categorias possíveis dependendo do tipo de objeto escolhido e com um botão para guardar o objeto e outro para cancelar a ação.

5 Construção da Solução

O presente capítulo começa por descrever a estrutura da solução com base no ambiente de desenvolvimento escolhido (Android Studio), identificando os diretórios usados, bem como os ficheiros de informação cartográfica necessários.

É também revelado o tipo de arquitetura de programação adotado no desenvolvimento da aplicação (Model-View-Controller) e o paradigma de programação (OOP), de modo a permitir a escalabilidade e a manutenibilidade da solução.

Neste capítulo é também abordada a estrutura de funcionamento da realidade aumentada através do ARCore e explicado o funcionamento e implementação de cada caso de uso.

Por fim, explica-se a persistência dos dados considerando os dois tipos de armazenamento adotados (ficheiros guardados na memória interna e externa do dispositivo móvel).

5.1 Estrutura da solução

A aplicação apresenta uma estrutura bem definida caracterizada por seis diretórios principais com funções bastante distintas. Este é o tipo de estrutura padrão adotado pelo Android Studio para cada projeto e pode ser verificado na Figura 21. Os diretórios são os seguintes:

- *sampledata* - contém os modelos 3d do cenário;
- *manifests* - integra o ficheiro *AndroidManifest* responsável por apresentar permissões de acesso a determinados sensores, ferramentas, armazenamento (câmara, giroscópio, acelerómetro, memória externa) e detalhes da aplicação para o sistema Android;
- *java* - engloba todo *java source code* incluindo testes *junit*;
- *generatedJava* - contém todo o código gerado pelo java;
- *assets* - contém um ficheiro de texto com os tipos de categorias e respetivos símbolos;
- *res* - inclui todos os recursos usados na aplicação, tal como *layouts*, *values* e *drawables*;

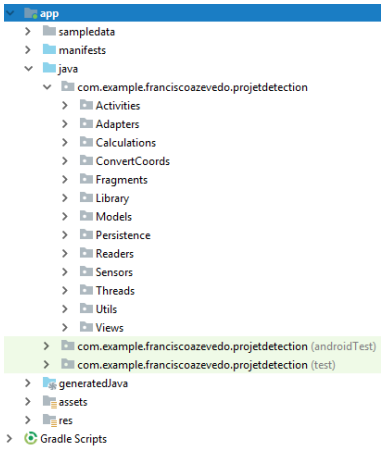


Figura 21 – Estrutura da solução

Enquanto que os dados cartográficos para completagem de campo tradicional são mostrados em papel, os dados da aplicação são guardados e apresentados num ficheiro do tipo *shapefile*, que é posteriormente convertido num ficheiro "*data.geojson*" segundo o sistema de coordenadas adequado. Este ficheiro "*data*" contempla toda a informação cartográfica necessária para a construção do cenário exceto informação de símbolos e cores. Este documento encontra-se na memória externa do telemóvel assim como o ficheiro final com os dados alterados, como será mencionado no subcapítulo da persistência.

Considerando necessária a apresentação dos símbolos no cenário tridimensional e a ausência destes no ficheiro inicial foi criado um ficheiro de texto nos *assets* da aplicação, chamado "*Catalog.txt*". O local de armazenamento foi nos *assets* devido à rapidez em aceder ao ficheiro e à necessidade de alteração dos dados. Este é composto pelos tipos de categorias e respetivos símbolos. Esta informação é proveniente do catálogo oficial da DGT (Direção-Geral do Território) denominado por "Catálogo de Objetos para cartografia topográfica à escala 1:2 000". A categoria de um objeto serve como identificador principal, ou seja, sempre é que preciso obter um determinado símbolo é procurada a categoria correspondente.

5.2 Arquitetura da aplicação

A solução foi construída com base numa arquitetura e num tipo de paradigma: *Model-View-Controller* (MVC) e Object-Oriented Programming (OOP), respetivamente.

O estilo de programação orientado a objetos foi abordado segundo um conjunto de classes, objetos e relações entre objetos. Os conceitos principais deste conceito são o polimorfismo, a abstração, o encapsulamento e a Herança. Resumidamente, o polimorfismo consiste na representação de um objeto segundo várias formas, a abstração esconde os detalhes da implementação para o utilizador, o encapsulamento agrega os atributos e métodos numa

unidade e a Herança representa a derivação entre classes. A cena, os objetos cartográficos e os sistemas de coordenadas representam vários tipos de objetos da aplicação.

A arquitetura MVC, tal como o nome indica, divide a aplicação em três componentes lógicas principais (*Controllers*, *Views* e *Models)*. A Figura 22 mostra como é processada a relação entre as várias componentes.

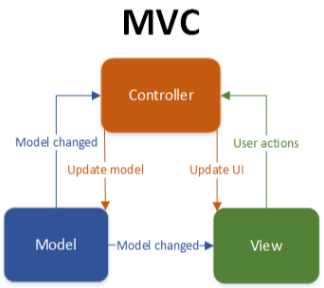


Figura 22 – Estrutura MVC

Os *Controllers* funcionam como uma interface ou uma ponte entre os *Models* e as *Views* e, portanto, processam todo o sistema lógico da aplicação e manipulam os dados recebidos dos modelos (as *activities* do projeto tem comportamentos semelhantes).

As Views são componentes de interação entre o sistema e o utilizador. Neste caso em particular, estas foram criadas a partir de vistas estáticas *xml* e de vistas dinâmicas como os casos *MapView* e *OrientationView* que são classes que estendem a *class SurfaceView*.

A componente *Model* é relativa à logica de dados e ao envio destes ao *Controller*.

Além destes três módulos principiais e de grande parte do processo lógico ser controlado por este sistema, o projeto apresenta ainda módulos para persistir os dados nos ficheiros externos, leitura dos dados, *threads*, criação de fragmentos, entre outros. Outros exemplos de arquiteturas semelhantes ao MVC são o MVP (*Model-View-Presenter*) e o MVVM (*Model-View-ViewModel*).

Estes duas componentes de desenvolvimento de *software* foram adotadas, pois permitem um rápido processo de desenvolvimento, suportam técnicas assíncronas e permitem *maintainability* e *scability*.

5.3 Realidade aumentada

5.3.1 Estrutura

O kit ARCore apresenta um conjunto de *sceneforms*, isto é, SDKs com finalidades distintas para a representação de realidade aumentada. O *Solar System* é um desses exemplos e serviu de base para a construção e estruturação dos cenários cartográficos da aplicação. Resumidamente,

esta aplicação consiste na representação do sistema solar através da realidade aumentada, onde é criada uma cena com uma hierarquia bem definida, constituída por vários nós (os planetas). Assim como o Solar *System*, o FieldAR apresenta também uma estrutura hierarquizada de nós como pode ser observado no exemplo da Figura 23. A *BaseStreet* representa o nó principal que contém todos os outros nós filhos. Os nós filhos podem ser pontos, linhas ou polígonos, que por sua vez são compostos por posições, paredes, tetos, etc.

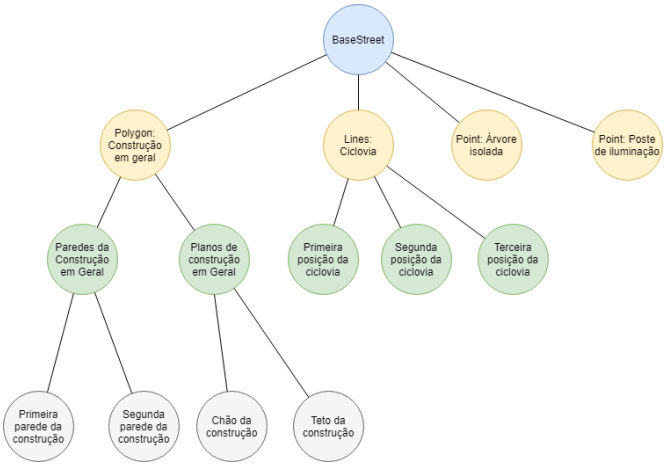


Figura 23 – Exemplo de estrutura de nós da aplicação

5.3.2 Permissões

A construção de um cenário com realidade aumentada apenas é conseguida através da conjugação de dois fatores, isto é, quando é dado acesso à câmara do telemóvel e quando o ARCore está instalado no dispositivo. Ora, a existência destes dois fatores tem de ser verificados pelo sistema através das funções requestCameraPermission(activity,requestCode*)* e *ARCoreApk.getInstance().requestInstall(activity, !installRequested)*. Caso se confirmem as duas premissas, é criado uma sessão de realidade e aumentada.

5.3.3 Modelação

A representação de modelos 3D e de vistas a duas dimensões no mundo real são feitas segundos dois tipos de renderização: *ModelRenderable* e *ViewRenderable*. Enquanto que o *ModelRenderable* renderiza três tipos de sólidos geométricos (cubos, cilindros e esferas), o *ViewRenderable* renderiza vistas a duas dimensões. Quer os modelos como as vistas possuem nós associados. De modo a poder gerar os modelos e vistas, foram inicializados e compilados os dois tipos de *renderables*, como verificado na função *renderObj* da Figura 24.

```java
/**
 * build the model and view renderable
 * In this case it creates a cylinder renderable and a normal view
 */
protected void renderObj() {
    CompletableFuture<ModelRenderable> poleStage = ModelRenderable.builder().setSource( context: this, Uri.parse("Cylinder.sfb")).build();
    viewDataObjStage = ViewRenderable.builder().setView( context: this, R.layout.view_data_obj).build();
    CompletableFuture.allOf(poleStage, viewDataObjStage)
            .handle((notUsed, throwable) -> {
                // When you build a Renderable, Sceneform loads its resources in the background while
                // returning a CompletableFuture. Call handle(), thenAccept(), or check isDone()
                // before calling get().
                if (throwable != null) {
                    DemoUtils.displayError( context: this,  errorMsg: "Unable to load renderable", throwable);
                    return null;
                }
                try {
                    poleRenderable = poleStage.get();
                    MaterialFactory.makeTransparentWithColor( context: this, new Color( r: 0.9f,  g: 0.9f,  b: 0.9f,  a: 0.5f))
                            .thenAccept(color -> poleRenderable.setMaterial(color));
                    // Everything finished loading successfully.
                    hasFinishedLoading = true;
                } catch (InterruptedException | ExecutionException ex) {
                    DemoUtils.displayError( context: this,  errorMsg: "Unable to load renderable", ex);
                }
                return null;
            });
}
```

Figura 24 – Inicialização dos modelos e vistas

5.3.4 Seleção de planos

A *arsceneView* é uma *SurfaceView* que integrado com o ARCore faz o *rendering* de uma cena constituída por *frames*. O utilizador pode criar uma cena selecionando um ponto no plano. Os toques são captados através da classe *GestureDetector*.

Considerando que o posicionamento da cena só é feito uma vez, a melhor forma de o garantir é através do gesto *onSingleTapUp()*. A seleção de um plano da cena está associada a uma única *Frame*. Consequentemente, todas as in terações serão associadas a esta. A Figura 25 mostra um exemplo de uma deteção de um plano onde as bolinhas brancas representam o plano do chão identificado.

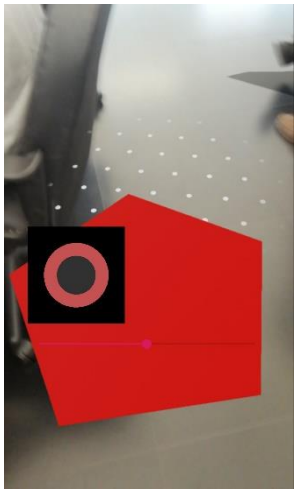


Figura 25 – Seleção de um plano e representação de um objeto

O cenário é constituído por uma âncora que representa o centro de todo o cenário e esta é posicionada mediante os interesses o utilizador. A aplicação possui uma função chamada *tryPlaceStreetSystem()* que permite detetar todos os eventos de *tap* de uma determinada *frame*, não só para o posicionamento da cena, como de novos pontos ou linhas. Os toques apenas são validados caso seja detetado um plano, caso contrário nenhum objeto é posicionado.

5.4 Casos de uso

5.4.1 Desenhar cenário cartográfico

De forma a clarificar o processo de criação de cenário, foi elaborado um diagrama de sequência (Figura 26). Este resume as interações, a complexidade e a ordem de tempo de interação entre esses os vários objetos.

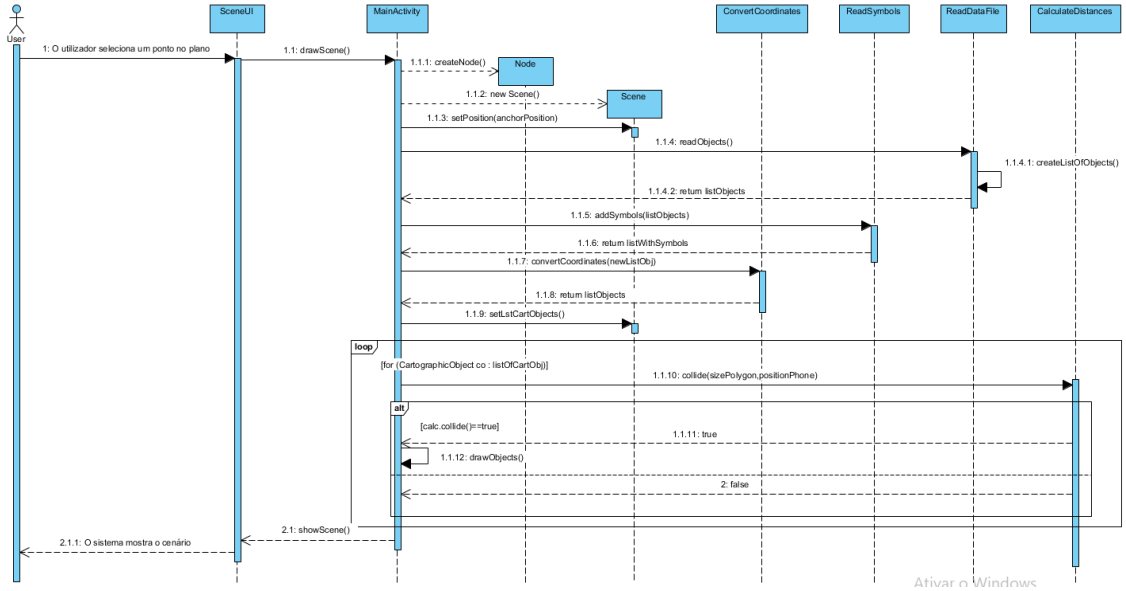


Figura 26 – Diagrama de sequência (Desenhar cenário cartográfico)

Conforme foi mencionado anteriormente, os cenários cartográficos são uma consequência da seleção de um plano. Depois de selecionado um plano é instanciado um Objeto chamado *Scene* que irá conter a sua posição, bem como a lista de todos os dados cartográficos a representar. Os dados cartográficos são lidos a partir do ficheiro *Datafile.geojson*. Estes, são convertidos em objetos do tipo *CartographicObjects* e adicionados a uma lista. Esta lista é alterada devido a conversões de sistemas de coordenadas e adições de novos atributos, tais como, cores e símbolos. Por fim, são gerados os modelos no cenário que se encontram dentro de um determinado alcance.

Ora, as coordenadas dos dados cartográficos quando lidas diretamente do ficheiro encontram-se em graus. Logo, é necessário efetuar uma conversão de sistema de coordenadas para metros (EPSG: 4326 para EPSG: 3857), uma vez que a unidade de medida dos cenários com ARCore é metros. Esta conversão é conseguida através de uma biblioteca de conversão de coordenadas Proj4J. A *class* principal que processa esta ação é chamada de *CRSFactory*.

No entanto, o processo ainda não está concluído, pois as coordenadas não se encontram no referencial cartesiano pretendido, ou seja, as coordenadas estão convertidas tendo como centro do referencial a Terra em vez do cenário de realidade aumentada como seria suposto. Assim, o algoritmo de conversão entre sistemas cartesianos passa por realizar uma translação, um ajuste de unidades e uma rotação. Este ajuste de unidades surge, uma vez que, a conversão de coordenadas de graus para metros é feita tendo como referência a linha do equador e, portanto, como Portugal se encontra a norte é necessário fazer o ajuste. A Figura 27 mostra todo o processo de conversão de coordenadas.

Por fim, a função retorna a mesma lista de objetos recebida como parâmetro, contudo já com as coordenadas corretas.

```
/**
 * converts the list of coordinates from degrees to meters
 * @param lstCartObjectsGeo list of all cartographic objects
 * @param coordPhone location of a phone
 * @param oazimuth
 * @param latPhone latitude of the person in degrees
 * @return the list of cartographic objects converted
 */
public ArrayList<CartographicObject> convertGeographicToMetric(ArrayList<CartographicObject> lstCartObjectsGeo, double [] coordPhone, double oazimuth, double latPhone) {
    double azimuth = convertAzimuth(oazimuth);
    double lat_cos = cos(Math.toRadians(latPhone));

    CRSFactory factory = new CRSFactory();
    CoordinateReferenceSystem srcCrs = factory.createFromName("EPSG:4326");
    CoordinateReferenceSystem dstCrs = factory.createFromName("EPSG:3857");
    BasicCoordinateTransform transform = new BasicCoordinateTransform(srcCrs, dstCrs);

    for (int i = 0; i < lstCartObjectsGeo.size(); i++) {
        ArrayList<Coordinate> lstCoordinates = new ArrayList<>();
        for (int j = 0; j < lstCartObjectsGeo.get(i).getLstCoordinates().size(); j++) {
            ProjCoordinate srcCoord = new ProjCoordinate(lstCartObjectsGeo.get(i).getLstCoordinates().get(j).getX(), lstCartObjectsGeo.get(i).getLstCoordinates().get(j).getY());
            ProjCoordinate dstCoord = new ProjCoordinate();

            transform.transform(srcCoord, dstCoord);

            double[] convertCoordPole = new double[2];
            convertCoordPole[0] = dstCoord.x;
            convertCoordPole[1] = dstCoord.y;

            double coordTransPoleX = convertCoordPole[0] - coordPhone[0];
            double coordTransPoleY = convertCoordPole[1] - coordPhone[1];

            double realX = -1 * coordTransPoleX * lat_cos;
            double realY = coordTransPoleY * lat_cos;

            Coordinate coordinate = new Coordinate( x: realX * cos(azimuth) + realY * sin(azimuth), y: -1 * realX * sin(azimuth) + realY * cos(azimuth));
            lstCoordinates.add(coordinate);
        }
        lstCartObjectsGeo.get(i).setLstCoordinates(lstCoordinates);
    }
    return lstCartObjectsGeo;
}
```

Figura 27 – Função que converte as coordenadas dos dados

Seguidamente, são desenhados os vários tipos de objetos (*Polygon, Point* e *LineString*) através do método *drawObjects*, onde é percorrida a lista, e desenhados os modelos em função do tipo. Os polígonos e as linhas são representados por paralelepípedos e os pontos por cilindros. Enquanto que os cilindros (pontos) de grandes dimensões representam postes, antenas, árvores, entre outros, os cilindros de pequenas dimensões simbolizam todos os objetos de pequenas dimensões, tal como tampas de saneamento, bocas de incêndio, sarjetas.

Cada objeto é constituído por pelo menos um nó e por um *ModelRenderable*. De modo a distinguir os diferentes modelos no cenário foi atribuída uma cor para cada objeto, sendo esta cor adicionada ao *Material* de cada um. O processo de identificação da cor passa primeiramente por identificar o tipo de objeto no *arrayColors* do ficheiro *arrays.xml* e seguidamente receber a cor correspondente através do ficheiro *colors.xml*.

As linhas da cena são representativas de ciclovias, limites de zonas, linhas ferroviárias. Estas são conjuntos de pequenos paralelepípedos agregados com alturas muito reduzidas. A função que cria um paralelepípedo é a *ShapeFactory.makeCube(vector3)*. Esta construção é feita de dois em dois pontos e a orientação dos segmentos de linha é calculada através dos pontos iniciais e finais através da função *Quaternion.lookRotation()*.

Por fim, a representação dos polígonos é conseguida através da função que projeta os polígonos constituintes de um determinado objeto. Existe uma função que constrói as paredes e outra que constrói chão e teto. Estes processos são independentes, pois existem objetos que não contêm paredes ou teto, como um simples jardim. Cada elemento é constituído por dois polígonos (frente e verso).

A construção dos polígonos não segue o princípio de construção dos outros objetos e, portanto, não usa as funções já existentes do ARCore de criação de modelos. Ora, isto acontece devido à impossibilidade de gerar sólidos com formas e número de vértices diferentes. Assim, optou-se por elaborar uma solução de baixo nível, na qual cada polígono é criado a partir de conjuntos de três pontos (triângulos). Para o efeito, utilizou-se biblioteca Earcut para formar e ordenar os triângulos. Basicamente, o algoritmo *Earcut.earcut(coordinates)* proveniente da biblioteca permite formar triângulos a partir dos pontos mais afastados do polígono (orelhas). À medida que estes pontos são agrupados vão sendo removidos da lista de pontos inicial e adicionados à nova lista de triângulos. Depois de definidos os triângulos e os vértices do polígono é gerado e retornado o novo *ModelRenderable.* A Figura 28 mostra o funcionamento deste processo.

```
public ModelRenderable createFloor(Material material, CartographicObject cartographicObject, int upOrDown) {
    double[] coordinates = new double[cartographicObject.getLstCoordinates().size() * 2];
    ArrayList<Vertex> vertices = new ArrayList<>(cartographicObject.getLstCoordinates().size());

    if (upOrDown == 1) {
        for (int i = 0; i < cartographicObject.getLstCoordinates().size(); i++) {
            vertices.add(Vertex.builder().setPosition(orderOfVertices(cartographicObject, coordinates, i)).build());
        }
    } else if (upOrDown == 0) {
        for (int i = cartographicObject.getLstCoordinates().size() - 1; i >= 0; i--) {
            vertices.add(Vertex.builder().setPosition( orderOfVertices(cartographicObject, coordinates, i)).build());
        }
    }


    List<Integer> triangles = EarCut.earcut(coordinates,  holeIndices: null,  dim: 2);
    Collections.reverse(triangles);

    Submesh submesh = Submesh.builder().setTriangleIndices(triangles).setMaterial(material).build();
    RenderableDefinition renderableDefinition = RenderableDefinition.builder().setVertices(vertices).setSubmeshes(Arrays.asList(submesh)).build();
    CompletableFuture future = ((Builder) ModelRenderable.builder().setSource(renderableDefinition)).build();

    ModelRenderable result;
    try {
        result = (ModelRenderable) future.get();
    } catch (InterruptedException | ExecutionException var21) {
        throw new AssertionError( message: "Error creating renderable.", var21);
    }
    if (result == null) {
        throw new AssertionError( detailMessage: "Error creating renderable.");
```

Figura 28 – Função de construção de polígonos

Os passos seguintes são semelhantes ao desenho de pontos e linhas. O resultado final desta projeção pode ver observado a partir da junção de duas *frames* de um pequeno cenário de realidade aumentada na Figura 29, contando já com os respetivos símbolos e com objetos alterados.

Figura 29 – Exemplo de cenário de realidade aumentada

5.4.2 Ver dados de um objeto

Através da aplicação é possível aceder às informações do objeto, mais concretamente, ao nome da categoria e símbolo associado. Para tal, o utilizador seleciona um objeto que contém um *setOnTapListener*() associado. Sempre que é selecionado um objeto da cena, este torna-se opaco de forma a transmitir feedback ao utilizador. Caso o utilizador selecione vários objetos apenas será mostrada a informação do último objeto e apenas este permanecerá opaco.

A função *showData(cartObj)* mostra nas duas *textViews* a categoria do objeto passado por parâmetro e o símbolo recebido do ficheiro "*Catalog*". A Figura 30 mostra a interface gráfica relativa à visualização dos dados.

Figura 30 – Visualização dos dados de um objeto

5.4.3 Mostrar Símbolos

Durante o decorrer do projeto foram realizados vários testes que permitiram identificar falhas, nomeadamente no enquadramento do utilizador. Assim, de forma a colmatar esta falha foi adicionada uma simbologia a três dimensões para cada objeto. À semelhança dos objetos, os símbolos 3D quando apresentados segundo uma componente de realidade aumentada são constituídos por um nó e por um *Renderable*. Neste caso, sendo o símbolo do tipo texto, o *Renderable* é do tipo *View*. O ViewRenderable é constituído por um objeto do tipo layout, neste caso, uma TextView que irá apresentar o símbolo do objeto correspondente.

O posicionamento dos símbolos no cenário seguiu três procedimentos distintos em função do tipo de objeto e da distância a que o utilizador se encontra deste. Ora, se o objeto fosse do tipo *Point*, o símbolo era representado com uma dada altura e na mesma posição do objeto. No caso das linhas e dos polígonos, a situação é bem mais complexa, visto que poderia haver dificuldades em visualizar símbolos devido ao tamanho de um determinado objeto. Assim, de forma a combater este obstáculo foram estabelecidos dois tipos de representação. Caso o utilizador se encontrasse em cima do objeto, o símbolo posicionar-se-ia na posição do utilizador, caso o utilizador se encontrasse fora, o símbolo estaria presente na borda mais próxima deste. A Figura 31 exemplifica os dois tipos de posicionamento.

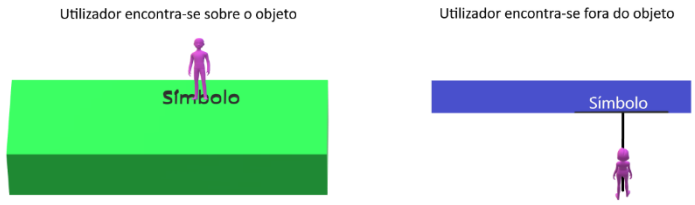


Figura 31 – Posicionamento dos símbolos

No primeiro algoritmo da Figura 32 é verificado se o utilizador se encontra em cima de um objeto. Assim, primeiramente constrói-se um Objeto do tipo *Polygon* constituído por uma lista de vértices (coordenadas dos pontos do objeto). A *class Polygon* é uma das classes já existentes da biblioteca *badlogic.gdx.math* que permite criar vários polígonos e é composta por várias funções associadas. Uma delas é a função *contains*() que permite verificar se um dado ponto está contido no polígono.

```
/**
 * check if the user is inside of polygon, if he is the position of symbol is the location of user
 * @param cartObj cartographic object
 * @param posPhone position phone
 * @return true if is inside
 */
public boolean checkIfInside(CartographicObject cartObj, Vector3 posPhone) {
    if (cartObj.getType().equalsIgnoreCase( anotherString: "Polygon")) {
        boolean isInside = false;
        float[] vertice = new float[cartObj.getLstCoordinates().size() * 2];

        for (int indice = 0; indice < cartObj.getLstCoordinates().size(); indice++) {
            vertice[2 * indice] = (float) cartObj.getLstCoordinates().get(indice).getX();
            vertice[2 * indice + 1] = (float) cartObj.getLstCoordinates().get(indice).getY();
        }

        Polygon p = new Polygon(vertice);
        if (p.contains(posPhone.x, posPhone.z)) isInside = !isInside;
        return isInside;
    }
    return false;
}
```

Figura 32 – Posição do símbolo dentro

No segundo algoritmo da Figura 33, o processo de verificação é bastante mais complexo. Primeiramente é criado um *Map* do tipo *TreeMap* com as coordenadas de cada ponto do objeto cartográfico e respetiva distância face à posição do utilizador. Consequentemente, é calculada a magnitude do vetor entre o utilizador e o ponto no polígono e o *dot* (produto escalar entre os vetores AB e AP).

Finalmente, é normalizada a distância *t* entre o utilizador e o ponto no polígono. Caso *t* seja menor que 0 o símbolo encontra-se na posição inicial. Caso *t* for maior que 1 o símbolo

encontra-se na posição final, caso contrário é adicionado o resultado do produto entre as coordenadas do vetor AB com o *t*.

```
public Vector3 calculatePosSymb(CartographicObject cartObj, Vector3 truePhone) {
    Map<Double, Coordinate> mapCoords = new HashMap<~>();
    double x, y;
    for (Coordinate c : cartObj.getLstCoordinates()) {
        x = c.getX();
        y = c.getY();
        mapCoords.put(Math.sqrt((x - truePhone.x) * (x - truePhone.x) + (y - truePhone.z) * (y - truePhone.z)), c);
    }
    Map<Double, Coordinate> treeMap = new TreeMap<~>(mapCoords);
    Map<Double, Coordinate> treeMapAux = new LinkedHashMap<>();
    treeMapAux = treeMap;

    Coordinate coord1 = (Coordinate) treeMapAux.values().toArray()[0];
    Coordinate coord2 = (Coordinate) treeMapAux.values().toArray()[1];

    double APx = truePhone.x - coord1.getX();
    double APy = truePhone.z - coord1.getY();
    double ABx = coord2.getX() - coord1.getX();
    double ABy = coord2.getY() - coord1.getY();
    double magAB2 = ABx * ABx + ABy * ABy;
    double ABdotAP = ABx * APx + ABy * APy;
    double t = ABdotAP / magAB2;
    double xret, yret;

    if (t < 0) {
        xret = coord1.getX();
        yret = coord1.getY();
    } else if (t > 1) {
        xret = coord2.getX();
        yret = coord2.getY();
    } else {
        xret = coord1.getX() + ABx * t;
        yret = coord1.getY() + ABy * t;
    }
    return new Vector3((float) xret,  v1: 0.04f, (float) yret);
}
```


Figura 33 – Função que calcula o posicionamento do símbolo nos limites

Por fim, é mostrado um exemplo (Figura 34) do posicionamento dos símbolos de dois objetos distintos, neste caso de um ponto e de um polígono, tendo em conta que o utilizador não se encontra em cima da área observada.

Figura 34 – Visualização de símbolos no cenário

5.4.4 Ajustar cenário

Uma das funcionalidades adicionais não previstas no planeamento foi o ajuste do cenário. Este requisito é bastante importante pois os cenários ficam mal posicionados no mundo real devido a erros de GPS e da bússola. Assim, o utilizador tem a possibilidade de mover a cena através de um simples *touchListener()* como referenciado na Figura 35. Outra forma de mover a cena seria através de um básico *joystick* do tipo *SurfaceView.* As grandes vantagens deste tipo de controlo, comparativamente com a solução adotada, é o facto de ser mais intuitivo e o utilizador não precisar de selecionar a cena. As vantagens da abordagem escolhida são o maior nível de usabilidade, facilidade de execução e precisão no posicionamento.

```
/**
 * this method allows the user to move the scene
 */
@SuppressLint("ClickableViewAccessibility")
public void moveScene() {
    arSceneView.getScene().getView().setOnTouchListener(new View.OnTouchListener() {
        @Override
        public boolean onTouch(View v, MotionEvent event) {
            if (adjustState) {
                switch (event.getAction() & MotionEvent.ACTION_MASK) {

                    case MotionEvent.ACTION_DOWN:
                        xDelta = event.getRawX();
                        yDelta = event.getRawY();
                        break;

                    case MotionEvent.ACTION_UP:
                        break;

                    case MotionEvent.ACTION_MOVE:
                        float addX = event.getRawX() - xDelta;
                        float addZ = event.getRawY() - yDelta;
                        xDelta = event.getRawX();
                        yDelta = event.getRawY();
                        float pointX = scene.getPosScene().x + (addX * 0.01f);
                        float pointY = scene.getPosScene().z + (addZ * 0.01f);
                        scene.setPosScene(new Vector3(pointX, scene.getPosScene().y, pointY));
                        anchorNode.setWorldPosition(scene.getPosScene());
                        break;
                }
                return false;
            }
            return false;
        }
    });
}
```

Figura 35 – Função MoveScene()

Além disso, existe ainda a possibilidade de rodar o cenário através de uma simples *seekbar* e do respetivo *listener.* Esta funcionalidade tem como o intuito combater a falha na orientação do cenário provocado pela descalibragem da bússola. O ângulo de rotação da cena é calculado pela diferença entre o valor escolhido na *seekbar* e a posição inicial. A função *Quaternion.axisAngle()* permite criar o vetor rotação da cena.

5.4.5 Ações de edição

Considerando a aplicação de suporte à completagem de campo, as ações de edição cartográfica também podem ser executadas pelo *software*. As ações possíveis são: adicionar ponto, adicionar linha, remover ponto, remover linha e editar categoria. A Figura 36 mostra a *interface* das ações possíveis.

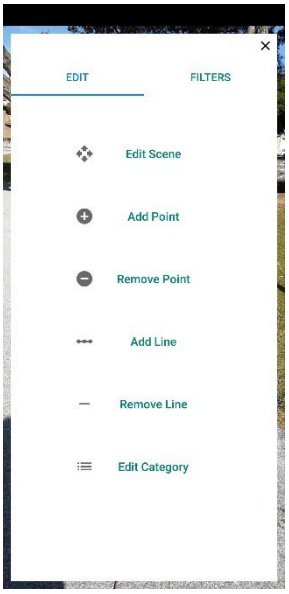


Figura 36 – *User Interface* das ações de edição

As ações de adição de pontos e linhas seguem o mesmo princípio de implementação, ou seja, ambos pertencem a uma lista de pontos e um nó principal (nó *baseStreet*) e são caracterizados pelo próprio nó, por um *ModelRenderable*, um *material* e uma posição final. Caso seja adicionado apenas um ponto à lista, é definido o objeto como do tipo ponto, caso seja adicionado mais do que um elemento, é identificado como um objeto do tipo linha.

A orientação da linha é adaptada mediante os pontos de início e fim de cada segmento. A função *addPoint(position)* da Figura 37 adiciona um ponto no plano.

```
/**
 * add a new point to the scene, if there is more than one new point in the scene,
 * the new point connects to the last point added
 * @param position position of point
 */
protected void addPoint(float[] position) {
    Node nodePoint = new Node();
    nodePoint.setParent(baseStreet);
    nodePoint.setLocalPosition(new Vector3(position[0], v1: 0f, position[1]));
    ModelRenderable rend = ShapeFactory.makeCylinder( radius: 0.3f, height: 4f, new Vector3( v: 0, v1: 2f, v2: 0), poleRenderable.getMaterial());
    nodePoint.setRenderable(rend);
    Material auxMat = nodePoint.getRenderable().getMaterial().makeCopy();
    Color color = new Color( r: 1, g: 1, b: 1, a: 0.5f);
    auxMat.setFloat4(MaterialFactory.MATERIAL_COLOR, color);
    nodePoint.getRenderable().setMaterial(auxMat);
    Node noded = new Node();
    noded.setWorldPosition(new Vector3(position[0], v1: 0f, position[1]));
    lstCoordLine.add(noded);
    moveObject(nodePoint, noded);

    if (lstCoordLine.size() > 1) {
        drawLineRealTime();
    }

}
```

Figura 37 – Função addPoint()

O sistema tem a possibilidade de reposicionar objetos movendo-os a partir de qualquer parte da tela. No caso das linhas, sempre que é alterado o posicionamento de um dos pontos o

segmento de linha correspondente é também alterado. Este princípio de movimentação é muito semelhante ao do ajuste da cena.

Depois de adicionado um ponto na cena é necessário introduzir a categoria do objeto. As categorias encontram-se numa *DropDownView* e esta interação é controlada por um *adapter* denominado *CategoryAdapter*. A *CategoryAdapter* estende a *class ArrayAdapter* e, portanto, é constituída por um construtor geral e de funções específicas de controlo da View como o *getCount()* que mostra o número de elementos e o *getView* que retorna a *View* com as categorias. A Figura 38 mostra alguns exemplos de adição de pontos e linhas.

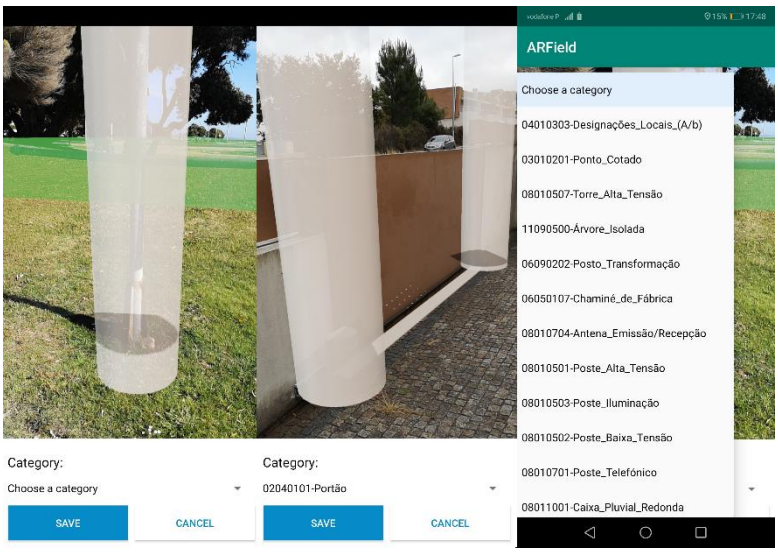


Figura 38 – Adição de pontos e linhas

Quer a remoção de pontos e linhas, como de edição de categorias, são bastante simples, tendo como ação principal comum, selecionar um objeto do cenário. Depois de selecionado, o utilizador poderá, então, concretizar uma de duas ações.

A remoção de um objeto é precedida de um alerta com o intuito de garantir a decisão correta do utilizador. Este alerta é do tipo *AlertDialog* sendo composto por uma mensagem com o nome do objeto em questão. Sempre que o utilizador seleciona um objeto este fica negro, de modo a transmitir *feedback* da ação tomada. Esta transformação de cor é realizada numa cópia do *Material* inicial. Finalmente, o objeto é eliminado da lista e do ficheiro que contém os dados cartográficos através do seu índice.

A categoria de um objeto é alterada através de uma lista de categorias. Esta lista adapta-se mediante o tipo de objeto selecionado, ou seja, quando o utilizador escolhe um objeto do tipo ponto, apenas as categorias deste tipo serão mencionadas na lista.

5.4.6 Desenhar mapa

Além do cenário tridimensional a aplicação disponibiliza um mapa com todos os dados atualizados. O mapa para além de projetar todos os objetos, mostra a localização e a orientação do utilizador, permitindo assim melhorar a perspetiva de enquadramento do utilizador.

A class *MapView* contém todas as funções relativas à construção do mapa. Esta classe estende a subclasse *SurfaceView* considerando esta como a melhor abordagem para desenhar em Android. Uma *SurfaceView* é uma vista orientada ao desenho segundo uma hierarquia de *view*.

A componente de construção escolhida para desenhar foi o *Canvas* devido à diversidade de opções de construção e de transformação, tal como, *translate, rotate, scale, drawArc, drawLine,* etc.

O método *drawMap*() define a matriz inicial que será a base de representação do mapa e é definida a escala da matriz. De seguida, o algoritmo percorre a lista de objetos usado no cenário de realidade aumentada e mediante o tipo de objeto é escolhida e desenhada a figura geométrica. A Figura 39 mostra exemplos de mapas gerados a partir de vários ficheiros de dados.

Figura 39 – Exemplos de mapas

Segundo a lógica de construção das figuras, os pontos no mapa são representados por círculos e quer as linhas como os polígonos são representados por linhas. Caso as linhas forem fechadas, estas poderão formar os polígonos.

Inicialmente, para a construção de um ponto é escolhido um raio para o círculo e é criado um objeto do tipo *Paint* que permite pintar o círculo com a cor estabelecida para cada categoria. Por fim, a função *drawCircle(positions,radius,paint)* cria o círculo tendo em conta as coordenadas do objeto, a largura e comprimento da tela, raio e o objeto *paint*.

A função do *Canvas* chamada para desenhar um objeto do tipo linha é a *drawLine*(). Este método recebe como parâmetros um ponto inicial, um ponto final e um objeto *Paint* que define a cor da linha. Normalmente, um objeto do tipo linha é constituído por vários pontos, logo é necessário criar um *loop* para interligar todos os pontos, formando assim a linha final (Figura 40).

```
/**
 * draw line representing a cartographic object
 * @param cartographicObject cartographic object
 */
@SuppressLint("ResourceAsColor")
public void drawLine2D(CartographicObject cartographicObject) {
    // Get and lock canvas object from surfaceHolder.
    Path wallpath = new Path();
    Paint wallpaint = new Paint();
    int colorCart = getColorCart(getContext(), cartographicObject);
    int color = ContextCompat.getColor(this.getContext(), colorCart);
    wallpaint.setColor(color);
    for (int i = 0; i < cartographicObject.getLstCoordinates().size() - 1; i++) {
        canvas.drawLine( startX: (float) cartographicObject.getLstCoordinates().get(i).x + getWidth() / 2,
                startY: (float) cartographicObject.getLstCoordinates().get(i).y + getHeight() / 2,
                stopX: (float) cartographicObject.getLstCoordinates().get(i + 1).x + getWidth() / 2,
                stopY: (float) cartographicObject.getLstCoordinates().get(i + 1).y + getHeight() / 2,
                wallpaint);
    }
    canvas.drawPath(wallpath, wallpaint);
    if (cartographicObject.isEdit())
        drawIconEdit(cartographicObject, wallpath);
}
```

Figura 40 – Função DrawLine() relativa ao mapa

Finalmente, são desenhados os polígonos a partir de um *Path*. Este pode ser constituído por um conjunto de segmentos de linhas ou curvas. O *path* foi a abordagem escolhida, pois é possível a partir dos vários pontos criar um polígono fechado.

No início é necessário definir o primeiro ponto do polígono, através da função *path.moveTo(coordinates)*. De seguida, à medida que é percorrida a lista de coordenadas de um determinado objeto são acrescentados os vários segmentos de linha *(path.lineTo(coordinates))*. Sendo um polígono fechado, no final do ciclo é necessário adicionar novamente a primeira posição da lista, de modo a fechar a figura geométrica.

Relativamente ao preenchimento do polígono, a *class Paint* tem a particularidade de poder definir o tipo de estilo da figura, ou seja, é possível através do *paint.setStyle()* definir um preenchimento ou contorno (*FILL*, *STROKE* e *FILL_AND_STROKE*) para a figura. A Figura 41 mostra como foi implementado o método *drawPolygon2d(cartObj)*.

```
/**
 * draws a polygon that represents a cartographic object
 * @param cartographicObject cartographic object
 */
@SuppressLint("ResourceAsColor")
public void drawPolygon2D(CartographicObject cartographicObject) {

    Paint wallpaint = new Paint();
    wallpaint.setStyle(Paint.Style.FILL);

    int colorCart = getColorCart(getContext(), cartographicObject);
    int color = ContextCompat.getColor(getContext(), colorCart);
    wallpaint.setColor(color);
    wallpaint.setAlpha(100);

    Path wallpath = new Path();
    wallpath.reset(); // only needed when reusing this path for a new build
    for (int i = 0; i < cartographicObject.getLstCoordinates().size(); i++) {
        Coordinate coordObj = cartographicObject.getLstCoordinates().get(i);
        if (i == 0) {
            wallpath.moveTo( x: (float) coordObj.x + getWidth() / 2, y: (float) coordObj.y + getHeight() / 2);
        }
        wallpath.lineTo( x: (float) coordObj.x + getWidth() / 2, y: (float) coordObj.y + getHeight() / 2);
    }
    wallpath.lineTo( x: (float) cartographicObject.getLstCoordinates().get(0).x + getWidth() / 2, y: (float) cartographicObject.getLstCoordinates().get(0).y + getHeight() / 2);
    canvas.drawPath(wallpath,wallpaint);
    Paint paint = new Paint();
    paint.setStyle(Paint.Style.STROKE);
    paint.setStrokeWidth(0.1f);
    paint.setColor(ContextCompat.getColor(getContext(),R.color.black));
    canvas.drawPath(wallpath,paint);
    if (cartographicObject.isEdit())
        drawIconEdit(cartographicObject,wallpath);
}
```

Figura 41 – Função DrawPolygon2d() relativa ao mapa

Sempre que o utilizador realiza uma ação de adição ou edição de categoria o mapa é atualizado com o intuito de informar o utilizador das alterações tomadas. A forma mais simples de representar essas alterações é através do uso de um contorno para cada objeto do mapa alterado. Esta opção é a mais viável do que, por exemplo, a adição de um símbolo, pois reduz a quantidade de informação no mapa e consequentemente, a quantidade de "poluição visual". A função *setStrokeWidth(thickness)* desenha um contorno em cada objeto cartográfico enviado por parâmetro com uma determinada espessura.

A representação do utilizador é caracterizado no mapa da Figura 42 como um círculo e um cone. O círculo representa a posição do utilizador no mapa e o cone representa a orientação no mapa e simultaneamente no mundo real.

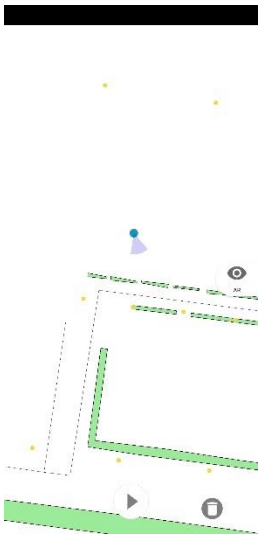


Figura 42 – Localização do utilizador

Ora, tendo em conta que o utilizador está sempre em movimento foi necessário criar uma *thread* para atualizar em *run-time* a orientação e o posicionamento do utilizador no mapa. Uma das formas de desenhar um cone a partir do *canvas* é através da função *drawArc(oval,angle,sweepAngle,useCenter,paint)*. Os parâmetros representam:

- *oval* - limites usados para definir o tamanho e a forma do arco;
- *angle* – angulo inicial em graus onde o arco começa;
- *sweepAngle* - a abertura do arco;
- *useCenter* – inclui o centro da oval no arco caso seja *true*;
- *paint* – objeto *Paint* que contém a cor do arco.

5.4.7 Navegar no mapa

A navegação no mapa pode ser feito de duas formas distintas: *zoom* e *drag*. O *zoom* é feito através de um *scale* da matriz que contém o mapa e o *drag* através de uma ação *move* do onTouchEvent().

O *zoom* é efetuado através da extensão da *class ScaleGestureDetector*. Esta classe apresenta um conjunto de funções que foram implementadas na *class MapaView*. A função *transformMatrix.postScale(factorX,factorY)* permite realizar o scale do mapa tendo como fator de aumento o valor recebido do *detector*.*getScaleFactor*(). Dado que o *scale* da matriz altera o posicionamento do mapa, é necessário realizar um *translate* para manter o mapa na posição original.

Esta abordagem é a mais indicada para a realização do zoom, pois o *ScaleGestureDetector* apenas é identificado quando o utilizador usa pelo menos dois pontos de referência e é detetado a partir do *MotionEvent*, tal como o *move* usado para mover o mapa. Uma das desvantagens desta solução, é o facto de ser necessário desenhar todo o mapa sempre que é realizada a transformação da matriz.

A funcionalidade de mover o mapa é acionada pelo evento *onTouchEvent*(). Os eventos de toque despoletados durante esta ação são:

- *Action_Down* – despoletado quando o utilizador dá o primeiro toque na tela;
- *Action_Move* – despoletado quando o utilizador move o mapa;
- *Action_Pointer_Up* – é despoletado quando o utilizador tira um dedo da tela, mas ainda existem dedos na tela.

A nova posição do mapa varia consoante o posicionamento do dedo do utilizador na tela. Esse valor é recebido segundo duas coordenadas x e y. Como os valores recebidos do *event*.*getX*() e *event*.*getY*() são valores elevados, é feita uma alteração da escala de forma a aumentar a fluidez de *drag* do mapa.

A ação *pointer_up* é mais adequada do que um simples *action_up* devido às interferências com o evento do *scale*. Além disso, este evento permite ainda identificar qual o índex do dedo que foi removido da tele. A Figura 44 e a Figura 43 mostram exemplos de navegação do mapa.

Figura 43 – Zoom in no mapa

Figura 44 – Zoom out no mapa

5.4.8 Trajeto

O trajeto no mapa descreve o caminho efetuado pelo utilizador durante a completagem de campo. Este trajeto encontra-se a preto e é construído a partir das várias posições detetadas e armazenadas num ficheiro local (Figura 45). Considerando o trajeto como um conjunto de linhas, o algoritmo de desenho é idêntico ao algoritmo usado para a desenhar os objetos do tipo Linha no mapa. Este percurso é começado a partir do momento em que o utilizador clica no botão *play* do mapa. Inicialmente, o sistema verifica se a lista de pontos está vazia e, caso não esteja, são desenhados os segmentos de linha entre cada dois pontos.

Por fim, o utilizador tem a possibilidade de recomeçar todo o trajeto, necessitando apenas apagar os dados do ficheiro e consequentemente da lista das posições.

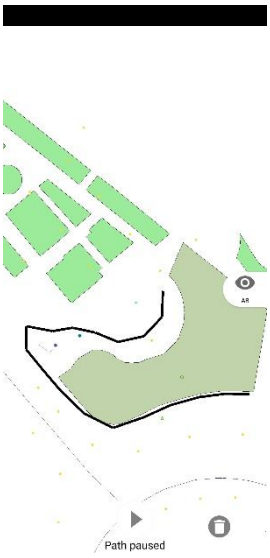


Figura 45 – Exemplo de um trajeto

5.4.9 Filtragens

As filtragens são uma componente adicional que facilitam a visualização do mapa e simultaneamente do cenário de realidade aumentada e diminuem a quantidade de informação a apresentar. Todas as categorias existentes a partir dos dados recebidos são adicionadas a uma lista e depois inseridas numa *listView* a partir de um *CategoryAdapter.* Este *adapter* permite manipular os dados da lista e obter outras informações, tal como o número de elementos da lista ou um elemento de uma posição à escolha.

Esta *View* é do tipo *CHOICE_MODE_MULTIPLE*, isto é, pode ser selecionada mais do que uma categoria para a filtragem.

Além das categorias, existe uma opção "*ALL*" para selecionar todos os dados. Sempre que é selecionada apenas uma categoria este item é desativado. Esta ação é realizada através do método *setItemChecked(position,false)*, onde o argumento *position* é 0 pois o "*ALL*" é o primeiro elemento da lista. Por outro lado, sempre que este é selecionado, todos os outros *items* são desativados. A Figura 46 mostra a *User Interface* das filtragens.

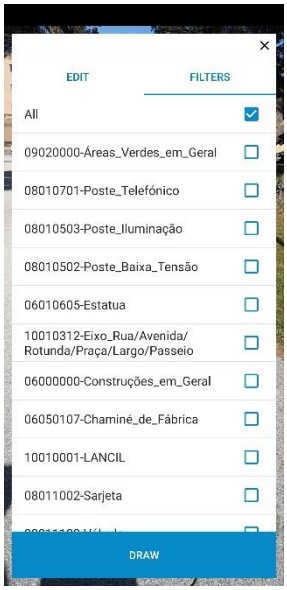


Figura 46 – *User Interface* das Filtragens

5.5 Persistência

A persistência dos dados é uma componente fundamental para as aplicações móveis. Neste caso em particular, o armazenamento dos dados é crucial, pois todas as alterações dos dados são feitos a partir do dispositivo. A escolha do tipo de persistência pode depender de vários fatores, nomeadamente do tipo de dados que se pretende armazenar, da necessidade de serem privados ou partilhados ou do espaço de armazenamento necessário.

Existem várias formas de persistir os dados em *Android*, porém existem seis formas que se destacam:

- Armazenamento interno: dados guardados em ficheiros privados no sistema de ficheiros do telemóvel;
- Armazenamento externo: dados guardados em ficheiros públicos disponíveis para partilha;
- Ficheiros partilhados: dados guardados em pares de *key-value* que não requerem uma estrutura;
- Base de dados: dados guardados num sistema de base de dados privado. Usadas para a manipulação complexa de dados relacionados;
- *Storage Access Framework* (SAF): acesso a ficheiros provenientes de vários fornecedores remotos;
- Plataforma *backEnd* (ex. Firebase): disponibiliza vários serviços remotos de base de dados em tempo real;

Ora, qualquer persistência remota é inadequada, pois implica uma ligação à rede (dados móveis) ou internet do dispositivo, o que nem sempre é possível, pois muitas vezes, a completagem de campo é executada em locais sem rede. Além disso, as fases posteriores da cartografia têm preferência em receber os dados atualizados em ficheiros *geojson*, logo a melhor forma de persistir os dados, tendo em conta estas necessidades, é através de ficheiros em memória interna e externa.

Assim foi criado um ficheiro principal do tipo *geojson* com todos os dados cartográficos atualizados (*NewDataFile*), um ficheiro do tipo texto com o trajeto realizado até ao momento e um ficheiro com os índices dos novos objetos e dos objetos alterados.

A Figura 47 mostra um exemplo de parte de um ficheiro *NewDataFile* com dados relativos a áreas verdes. Cada objeto é constituído três atributos fundamentais: nome de categoria ("TIPO"), uma forma ("*type*") e uma lista de coordenadas ("*coordinates"*).


```
{
"type": "FeatureCollection",
"crs": { "type": "name", "properties": { "name": "urn:ogc:def:crs:OGC:1.3:CRS84" } },
"features": [
{ "type": "Feature", "properties": { "NOME": null, "TIPO" : "09020000-Áreas_Verdes_em_Geral" }, "geometry": { "type": "Polygon", "coordinates": [ [ [ -8.685534859180432, 41.170859136648325, 0.0],
{ "type": "Feature", "properties": { "NOME": null, "TIPO" : "09020000-Áreas_Verdes_em_Geral" }, "geometry": { "type": "Polygon", "coordinates": [ [ [ -8.6874174374891142, 41.169855800317613, 0.0],
{ "type": "Feature", "properties": { "NOME": null, "TIPO" : "09020000-Áreas_Verdes_em_Geral" }, "geometry": { "type": "Polygon", "coordinates": [ [ [ -8.6838545523786667, 41.167596049010378, 0.0],
{ "type": "Feature", "properties": { "NOME": null, "TIPO" : "09020000-Áreas_Verdes_em_Geral" }, "geometry": { "type": "Polygon", "coordinates": [ [ [ -8.6837698094959, 41.167581496753549, 0.0],
{ "type": "Feature", "properties": { "NOME": null, "TIPO" : "09020000-Áreas_Verdes_em_Geral" }, "geometry": { "type": "Polygon", "coordinates": [ [ [ -8.6836844033330014, 41.167567325957194, 0.0],
{ "type": "Feature", "properties": { "NOME": null, "TIPO" : "09020000-Áreas_Verdes_em_Geral" }, "geometry": { "type": "Polygon", "coordinates": [ [ [ -8.683598749722782, 41.167553156096778, 0.0],
{ "type": "Feature", "properties": { "NOME": null, "TIPO" : "09020000-Áreas_Verdes_em_Geral" }, "geometry": { "type": "Polygon", "coordinates": [ [ [ -8.6835135864460561, 41.167538230121366, 0.0],
{ "type": "Feature", "properties": { "NOME": null, "TIPO" : "09020000-Áreas_Verdes_em_Geral" }, "geometry": { "type": "Polygon", "coordinates": [ [ [ -8.6834257720206251, 41.167523822103426, 0.0],
{ "type": "Feature", "properties": { "NOME": null, "TIPO" : "09020000-Áreas_Verdes_em_Geral" }, "geometry": { "type": "Polygon", "coordinates": [ [ [ -8.6833396139765761, 41.167509027502177, 0.0],
{ "type": "Feature", "properties": { "NOME": null, "TIPO" : "09020000-Áreas_Verdes_em_Geral" }, "geometry": { "type": "Polygon", "coordinates": [ [ [ -8.6832544458586708, 41.167494854388515, 0.0],
{ "type": "Feature", "properties": { "NOME": null, "TIPO" : "09020000-Áreas_Verdes_em_Geral" }, "geometry": { "type": "Polygon", "coordinates": [ [ [ -8.6831689771186809, 41.167480683207977, 0.0],
{ "type": "Feature", "properties": { "NOME": null, "TIPO" : "09020000-Áreas_Verdes_em_Geral" }, "geometry": { "type": "Polygon", "coordinates": [ [ [ -8.6830856447127721, 41.167466750001012, 0.0],
```


Figura 47 – Exemplo de um *geojson*

A leitura e a persistência dos dados para os três ficheiros é feita de forma semelhante e é baseada em duas classes complementares: *FileInputStream* e *FileOutputStream*.

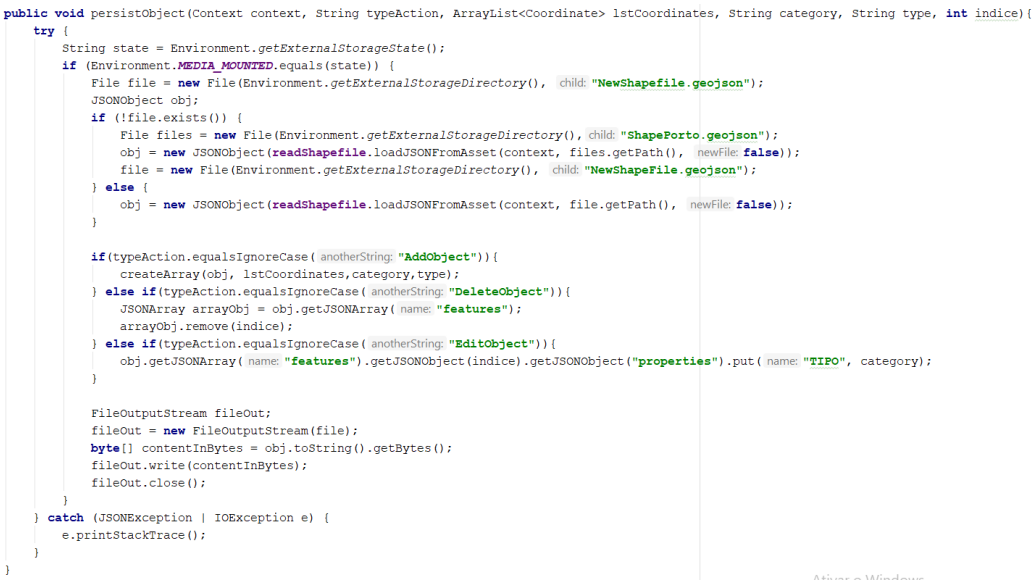
Enquanto que o processo de *FileInputStream* lê os bytes relativos ao ficheiro *geojson* através do método *read(buffer)* e converte-os numa *String*, o FileOutputStream faz respetivamente o contrário onde a função *write(buffer)* gera os bytes de saída.

Ora, depois de carregados os dados e convertidos numa *String* é necessário gerar os objetos. Considerando que esta contém os dados em formato *json* foi necessário desconstruí-la através de um *JSONArray* e de *JSONObjects*. Finalmente, os *jsonObjects* são convertidos em Objetos do tipo *CartographicObject* (Figura 48). O tipo de lista usado para representar os objetos foi o *ArrayList* dado que o tamanho da lista pode ser alterado, é orientado a objetos e os elementos podem ser adicionados ou removidos dinamicamente a partir de posições definidas.

```java
/**
 * create a list with the cartographic objects loaded from the external file
 * @param context context of main activity
 * @return an arraylist with the cartographic objects
 */
public ArrayList<CartographicObject> getObjectsFromShape(Context context) {
    ArrayList<CartographicObject> listCartographicObjects = new ArrayList<>();
    try {
        File file = new File(Environment.getExternalStorageDirectory(), child: "NewShapefile.geojson");
        JSONObject obj;
        if (file.getAbsoluteFile().exists())
            obj = new JSONObject(loadJSONFromAsset(context, file.getPath(), newFile: false));
        else {
            File files = new File(Environment.getExternalStorageDirectory(), child: "ShapePorto.geojson");
            obj = new JSONObject(loadJSONFromAsset(context, files.getPath(), newFile: false));
        }
        JSONArray m_jArry = obj.getJSONArray( name: "features");
        for (int i = 0; i < m_jArry.length(); i++) {
            JSONObject jo_inside = m_jArry.getJSONObject(i);
            JSONObject props = jo_inside.getJSONObject("geometry");
            JSONArray allCoordLines = checkType(props);
            CartographicObject p = new CartographicObject();
            p.setType(props.getString( name: "type"));
            for (int j = 0; j < allCoordLines.length(); j++) {
                if (props.getString( name: "type").equalsIgnoreCase( anotherString: "Point")) {
                    float lon = (float) (allCoordLines.getDouble( index: 0));
                    float lat = (float) (allCoordLines.getDouble( index: 1));
                    Coordinate coord = new Coordinate(lon, lat);
                    p.getLstCoordinates().add(coord);
                    break;
                }
                float lon = (float) (allCoordLines.getJSONArray(j).getDouble( index: 0));
                float lat = (float) (allCoordLines.getJSONArray(j).getDouble( index: 1));
                Coordinate coord = new Coordinate(lon, lat);
                p.getLstCoordinates().add(coord);
            }

            p.setId(i+1);
            p.setEdit(false);
            p.setCategory(getCategory(jo_inside, i, p));
            listCartographicObjects.add(i, p);
        }
    } catch (JSONException e) {
        e.printStackTrace();
        return new ArrayList<CartographicObject>();
    }
    return listCartographicObjects;
}
```

Figura 48 – Função que cria a lista com os Objetos

A abordagem contrária permite persistir os dados no ficheiro. Os objetos são convertidos em objetos do tipo *JSONObject* e, depois, através da *class FileOutputStream*, os dados são introduzidos no ficheiro. Os parâmetros são alterados conforme a ação escolhida. Adicionar ou apagar um objeto ou alterar a categoria deste são os tipos de ações permitidas. A Figura 49 apresenta o método que persiste os dados no ficheiro *NewDataFile*.



```
public void persistObject(Context context, String typeAction, ArrayList<Coordinate> lstCoordinates, String category, String type, int indice){
    try {
        String state = Environment.getExternalStorageState();
        if (Environment.MEDIA_MOUNTED.equals(state)) {
            File file = new File(Environment.getExternalStorageDirectory(), child: "NewShapefile.geojson");
            JSONObject obj;
            if (!file.exists()) {
                File files = new File(Environment.getExternalStorageDirectory(), child: "ShapePorto.geojson");
                obj = new JSONObject(readShapefile.loadJSONFromAsset(context, files.getPath(), newFile: false));
                file = new File(Environment.getExternalStorageDirectory(), child: "NewShapeFile.geojson");
            } else {
                obj = new JSONObject(readShapefile.loadJSONFromAsset(context, file.getPath(), newFile: false));
            }

            if(typeAction.equalsIgnoreCase( anotherString: "AddObject")){
                createArray(obj, lstCoordinates,category,type);
            } else if(typeAction.equalsIgnoreCase( anotherString: "DeleteObject")){
                JSONArray arrayObj = obj.getJSONArray( name: "features");
                arrayObj.remove(indice);
            } else if(typeAction.equalsIgnoreCase( anotherString: "EditObject")){
                obj.getJSONArray( name: "features").getJSONObject(indice).getJSONObject("properties").put( name: "TIPO", category);
            }

            FileOutputStream fileOut;
            fileOut = new FileOutputStream(file);
            byte[] contentInBytes = obj.toString().getBytes();
            fileOut.write(contentInBytes);
            fileOut.close();
        }
    } catch (JSONException | IOException e) {
        e.printStackTrace();
    }
}
```

Figura 49 – Função que permite a persistência dos dados

Os processos de armazenamento quer do trajeto efetuado, como da lista de novos objetos são bastante semelhantes.

No primeiro é guardada uma lista de coordenadas em formato *String* representativo das consecutivas posições do utilizador no mundo real. As coordenadas do mundo real detetadas pelo GPS são convertidas num sistema de X e Y, que representam as coordenadas no mapa.

O segundo ficheiro apenas guarda os IDs dos objetos que foram alterados ou adicionados ao *NewDataFile*. Estes IDs permitirão posteriormente identificar os objetos que terão um contorno no mapa.

6 Experimentação e avaliação da solução

De forma a validar a solução desenvolvida, foram adotadas várias metodologias de teste. A avaliação em questão é suportada por um conjunto de grandezas/critérios (tempo, memória, qualidade de *software* e precisão). Estes fatores foram decididos com base no problema e nos objetivos propostos.

A metodologia usada para avaliar a qualidade de *software* é o QEF, referenciando e qualificando as dimensões de usabilidade, eficiência, adaptabilidade e funcionalidade.

O presente capítulo mostra ainda os resultados obtidos a partir de inquéritos realizados a um público cliente da organização e a um público externo, que avaliaram e a usabilidade e a eficiência da aplicação FieldAR.

Por fim, são também mostrados alguns testes unitários e de performance que validaram o código produzido.

6.1 Problema e Objetivos

A produção de cartografia tem acompanhado a evolução das novas tecnologias. Os métodos e as ferramentas adotadas são cada vez mais dinâmicas e digitais. Apesar desta evolução existem ainda algumas fases de implementação tradicional que não recorrem à tecnologia como o caso da completagem de campo. Resumidamente, este processo consiste na verificação das plantas criadas a partir de fotos aéreas nos locais escolhidos. Quem executa este processo possui um conjunto de plantas em papel onde é possível alterar objetos cartográficos e respetivas posições, categorias, adição e remoção de novos objetos, entre outras ações. Todo este processo apresenta algumas desvantagens que podem comprometer a integridade dos mapas cartográficos desenvolvidos:

- o tempo de verificação e alteração dos dados cartográficos;

- dificuldade de enquadramento geográfico;
- erros de localização de objetos;
- adição de informação incorreta;
- suporte físico muito extenso e pouco prático;
- sobreposição de dados cartográficos;
- dificuldade em controlar e armazenar grandes quantidades de informação.

O objetivo principal da solução é o desenvolvimento de uma aplicação móvel com a introdução de realidade aumentada que facilite e suporte o processo de levantamento e edição de campo, através da representação gráfica de cenários cartográficos e ações de edição dos dados.

O produto pretenderá aumentar a eficiência deste processo e a precisão dos dados (sistemas de coordenadas) e armazenar e controlar grandes quantidades de informação.

6.2 Grandezas

A avaliação da solução é baseada segundo um conjunto de grandezas que representam os campos para a qual a aplicação está direcionada e foi desenvolvida. A eficiência da aplicação, a quantidade de dados quer da cena, quer do processo completo (memória), a qualidade do *software*, a usabilidade da aplicação e a precisão de localização de objetos cartográficos são as principais grandezas de destaque que servirão de base para toda a avaliação.

A eficiência da aplicação é fundamental para verificar a integridade do produto e confirmar a necessidade de utilização dos dois tipos de representação cartográfica.

A memória representa a quantidade de dados cartográficas que poderão ser apresentadas em simultâneo e a quantidade de dados que o ficheiro de dados poderá conter sem comprometer a execução da aplicação e o procedimento em geral. A memória de armazenamento e produção de cenários é uma mais-valia para o processo, pois caso se verifique um aumento significativo destes dois pontos, a anotação e armazenamento dos dados por parte da aplicação passa a ser mais viável.

A qualidade do *software* é avaliada de modo a analisar quais os requisitos funcionais e não funcionais que foram cumpridos e a qualidade de cada requisito atingido. É fundamental perceber a qualidade do *software* para garantir a usabilidade, eficiência e adaptabilidade do FieldAR.

A usabilidade da aplicação é testada com o intuito de validar a interação entre os utilizadores finais e o sistema e verificar a facilidade que o utilizador tem em alcançar os seus objetivos.

Por fim, será também considerada a precisão da aplicação, isto é, a precisão das localizações geográficas da cena e dos objetos que nela se encontram. A validação desta componente permitirá reforçar a necessidade da aplicação no campo visto que a localização de determinados objetos é apenas um valor estimado/aproximado.

6.3 Hipóteses a testar para suportar os resultados

Os resultados são suportados por um conjunto de hipóteses que verificam o cumprimento dos objetivos estabelecidos. A Tabela 17 mostra as hipóteses a testar para cada grandeza.

Tabela 17 - Identificação de grandezas e respetivas hipóteses

Grandeza	Hipótese
Eficiência	• A representação dos cenários tridimensionais é eficiente.
	• A navegação no mapa é eficiente.
Memória	• A aplicação suporta um elevado número de objetos na cena e no mapa
	• A aplicação armazena corretamente toda a informação
Qualidade do *software*	• A solução segundo a metodologia QEF ultrapassa os 85%
Usabilidade	• A usabilidade da aplicação não é significativamente comprometida dependendo do número de dados
Precisão	• Os novos objetos cartográficos introduzidos são registados com precisão
	• Os cenários estão bem posicionados depois do ajuste

6.4 Metodologias de avaliação

A metodologia principal usada para avaliar a qualidade de software foi o modelo *Quantitative Evaluation Framework* (QEF). Este modelo é baseado nos princípios de Engenharia de Software e de conceitos de qualidade e conta com referências de ISO 9126 e SCORM [36].

O QEF é caracterizado por um conjunto de dimensões, cada uma delas é constituída por um conjunto de fatores e requisitos com os respetivos graus de importância. As importâncias relativas de cada fator para cada dimensão podem ser representadas segundo a fórmula abaixo representada.

Dimensão i $= \sum_n (p_n \times factor_n) \; \sum_n (p_n) = 1$

p_n ∈ [0,1], o n é o número de fatores de uma dimensão.

Considerando o número de critérios, os pesos dos critérios e a percentagem de cumprimento é possível calcular o fator para cada dimensão.

Fator $f(z) = \frac{1}{\sum pr_m} \times \sum(pr_m \times pc_m)\ p_n\ \epsilon\ [0,1]$

Consequentemente a qualidade do sistema é:

q = $\left(1 - \frac{D}{\sqrt{n}}\right) * 100$ $q\epsilon[0,100]$ e D representa o desvio global do sistema

O uso do QEF é essencial para verificar a qualidade dos requisitos funcionais e não funcionais da aplicação. A qualidade total do sistema é apresentada sob a forma de percentagem.

E estrutura do QEF baseou-se em quatro dimensões fundamentais: a funcionalidade, eficiência, adaptabilidade e usabilidade. Estas dimensões foram escolhidas com base no tipo de projeto desenvolvido e respetivos objetivos. É de salientar que todos os requisitos quer funcionais como não-funcionais tiveram um peso semelhante.

Cada requisito foi avaliado segundo uma nota de 0 a 100%, porém cada cotação poderia ter apenas os valores 0, 50, 75 e 100. O valor era escolhido mediante as opções dadas para cada requisito, por exemplo, a representação dos postes no cenário apenas poderia ter o valor 0 caso não fossem representados ou 100 caso todos os postes fossem representados. Contudo, há casos em que seria injusto atribuir a cotação 0, só porque parte do objetivo não foi cumprido, daí existirem os valores de 50 e 75. Um exemplo deste tipo de abordagem surge da necessidade de qualificar a eficiência do zoom no mapa.

Estes valores foram atribuídos com base no feedback dos questionários, de questões feitas aos clientes e público externo durante as fases de teste e com a equipa de desenvolvimento. A Figura 50 exibe a pontuação para parte da componente de Funcionalidade. Esta componente é constituída por:

- um fator de realidade aumentada;
- construção de cenário;
- construção do mapa;
- trajetos;
- navegação no mapa;
- ajuste do cenário;
- edição do cenário;
- armazenamento de dados.

Funcionalidade	100	0,081	Realidade aumentada (AR Core)	10	FRA01 - Deteção de superfícies horizontais e verticais	100
				10	FRA02 - Seleção de uma superfície	100
				10	FRA03 - Criação de uma cena a partir do plano escolhido	100
	97,917	0,324	Construção do cenário	10	FCC01 - Representação de objetos a partir do ficheiro de dados	100
				10	FCC02 - Representação de objetos com transparência	100
				10	FCC03 - Representação de pontos no cenário	100
				10	FCC04 - Representação de linhas no cenário	100
				10	FCC05 - Representação de chão e tetos	100
				10	FCC06 - Representação de paredes	100
				10	FCC07 - Atualização do cenário ao fim de 10 segundos	100
				10	FCC08 - Inserção de simbologia para cada objeto cartográfico	100
				10	FCC09 - Visualização dos dados de cada objeto	100
				10	FCC10 - Representação dos objetos tendo em conta um alcance	100
				10	FCC11 - Adição de filtragens	100
				10	FCC12 - Seleção de objetos	75
	96,875	0,216	Construção do mapa	10	FCM01 - Representação dos objetos com transparência	100
				10	FCM02 - Representação de um contorno nos objetos editados	75
				10	FCM03 - Representação de pontos no mapa	100
				10	FCM04 - Representação de linhas no mapa	100
				10	FCM05 - Representação de polígonos no mapa	100
				10	FCM06 - Representação do utilizador no mapa	100
				10	FCM07 - Representação da orientação do utilizador no mapa	100
				10	FCM08 - Adição de filtragens	100
	100	0,054	Trajetos	10	FT01 - Criar Trajeto	100
				10	FT02 - Apagar Trajeto	100

Figura 50 – Resultados do QEF relativos à componente da Funcionalidade

Seguidamente foram avaliadas as componentes de adaptabilidade e eficiência. A primeira está associada ao ajuste dos elementos da aplicação dependendo do tipo de resolução, à adaptação de cenários mediante vários conjuntos de dados e à conservação de todo o sistema face às alterações realizadas pelo utilizador, mantendo a integridade da mesma. A eficiência está relacionada com a fluidez e rapidez de ajuste de cena, de navegação do mapa (*zoom* e *drag*) bem como número de erros da *app*.

Os valores são apresentados na Figura 51 onde é possível destacar as elevadas classificações.

	75	0,054	Navegação no mapa	10	FN01 - Zoom no mapa	75
				10	FN02 - Mover o mapa	75
	75	0,054	Ajuste do cenário	10	FC01 - Mover cenário	75
				10	FC02 - Rodar cenário	75
	100	0,135	Edição do cenário	10	FE01 - Edição da categoria do objeto cartográfico selecionado	100
				10	FE02 - Adição de novos pontos	100
				10	FE03 - Remover pontos da cena	100
				10	FE04 - Adição de linhas na cena	100
				10	FE05 - Remoção de linhas na cena	100
	100	0,081	Armazenamento dos dados	10	FA01 - Persistir novos objetos	100
				10	FA02 - Persistir objetos editados	100
				10	FA03 - Persistir trajetos	100
Adaptabilidade	100	0,20	Resolução	10	ARC01 - Aplicação ajusta-se ao ecrã do telemóvel	100
	100	0,20	Representação da cena	10	ARC02 - A aplicação mostra a cena corretamente dependendo do ficheiro de dados	100
	83,333	0,60	Conservação	10	AC03- O ficheiro de dados inicial não é alterado	100
				10	AC04 - O posicionamento e orientação da cena não é alterado com o tempo ou distância	50
				10	AC05 - A cena mantém se atualizada independentemente do número de execuções da aplicação	100
Eficiência	75	0,13	Modelação	10	EM01 - A cena está bem definida e identificada	75
	75	0,88	Navegação	10	EN01 - O ajuste de posições é simples e fluida	50
				10	EN02 - O ajuste da rotação é eficiente e direta	100
				10	EN03 - O zoom do mapa é eficiente	50
				10	EN04 - A movimentaçao do mapa é simples e direta	50
				10	EN03 - A modelação da cena é rapido e sem breaks	75
				10	EN04 - A aplicação não tem erros	100
				10	EN05 - A navegação entre atividades e menus é rápido e sem interrupções	100

Figura 51 – Resultados do QEF relativos à Funcionalidade, Adaptabilidade e Eficiência

A última componente Usabilidade teve como conceitos principais a interação com os menus, ações de completagem e qualidade da informação representada (verificar na Figura 52). Os valores foram atribuídos a partir dos resultados obtidos nos gráficos relativos aos inquéritos.

Usabilidade	66,667	0,27	Menus	10	UM01 - Os menus estão bem estruturados e intuitivos	75
				10	UM02 - A usabilidade não é comprometida dependendo do número de dados	50
				10	UM03 - A aplicação tem em consideração todas as exceções	75
	81,25	0,36	Edição do cenário	10	UE01 - As ações do utilizador transmitem feedback	75
				10	UE02 - O utilizador controla o posicionamento e a orientação da cena e dos objetos	50
				10	UE03 - O mapa é alterado em tempo real	100
				10	UE04 - O cenário é alterado em tempo real	100
	81,25	0,36	Qualidade do conteúdo	10	UQ01 - Indicações e textos bem escritos e intuitivos	100
				10	UQ02 - A objetos são distinguiveis e bem identificados	75
				10	UC03 - O enquadramento é explícito	75
				10	UQ03 - As posições dos objetos são precisas depois do ajuste	75

Figura 52 - Resultados do QEF relativos à Usabilidade

Depois de concluídos os cálculos, verificou-se uma qualidade total de aproximadamente 91%. Este valor, alto, resulta de todo um eficaz planeamento e desenvolvimento da aplicação. Alguns aspetos poderão servir de base para melhoria numa próxima versão destacando, assim, funcionalidades de seleção de objetos sobrepostos, informação transmitida durante o decorrer de cada ação de suporte e posicionamento das cenas no mundo real. Assim, conclui-se que o software apresenta uma qualidade muito significativa e com uma taxa de sucesso bastante elevada.

6.5 Análise dos resultados de usabilidade

Um dos aspetos principais a avaliar foi a usabilidade da aplicação. Para tal, os dois públicos foram submetidos a um processo bem estruturado de formação, experimentação e posteriormente de avaliação através de questionários. O primeiro público era constituído por três clientes internos (público-alvo) e o segundo por cinco elementos não pertencentes à organização e de faixas etárias distintas.

A formação foi realizada segundo um conjunto de três etapas. A primeira etapa consistiu numa contextualização do problema, dos objetivos e da necessidade de uso da aplicação. Esta contextualização teve maior impacto e importância no caso do público externo dado que estes não tinham conhecimento nas áreas. Durante a segunda etapa, foram mostradas as funcionalidades da aplicação bem como todas as ações de controlo e manipulação. Por fim, a aplicação foi testada tendo por base um pequeno conjunto de tarefas usadas na completagem de campo dado um percurso.

A experimentação da solução foi realizada de forma autónoma por todos os elementos de cada grupo, isto é, cada grupo teve a liberdade de experimentar a aplicação de acordo com os seus objetivos e interesses. Esta autonomia serviu para avaliar a facilidade de uso da aplicação bem como registar durações das ações.

Por fim, todos os intervenientes responderam a um questionário com o intuito principal de avaliar o estado da aplicação e o grau de usabilidade do mesmo. Este questionário foi divido em cinco componentes de avaliação:

- Experiência com cartografia e realidade aumentada;
- Performance da aplicação;
- Realidade aumentada;
- Mapas;
- Ações de completagem;

A primeira componente do inquérito teve como objetivos, caracterizar os utilizadores, perceber a experiência com tecnologias de realidade aumentada, GPS e cartografia, bem como a frequência de uso das mesmas. Os resultados desta fase podem ser verificados na Tabela 18, sendo que é possível constatar que nenhum dos utilizadores tem conhecimento ou experiência com aplicações que conjugam a cartografia com a realidade aumentada, valorizando assim a aplicação desenvolvida. Os públicos podem ser facilmente distinguíveis dada a sua experiência com cartografia.

Tabela 18 – Resultados dos inquéritos relativos à experiência das tecnologias

Perguntas	Número de respostas (Público-alvo)		Número de respostas (Público externo)	
	Sim	Não	Sim	Não
Tem alguma experiência com cartografia?	3	0	0	5
Já usou aplicações de realidade aumentada?	3	0	5	0
Se sim, estavam relacionadas com cartografia?	0	3	0	5
Costuma usar aplicações com GPS?	3	0	5	0

A aplicação FieldAR é caracterizada por um sistema de projeções de cenários de realidade aumentada e mapas com uso de GPS. Um dos fatores que permitiu avaliar a experiência dos utilizadores com este tipo de aplicações foi a frequência de uso do GPS.

Segundo o gráfico circular da Figura 53 verifica-se que 2/3 da amostra está acostumada a usar GPS algumas vezes por semana e 1/3 algumas vezes por mês. O facto do público cliente estar habituado a interagir com GPS é uma mais valia pois facilita a interação com a aplicação FieldAR e aumenta a eficácia de enquadramento e de execução das tarefas da completagem.

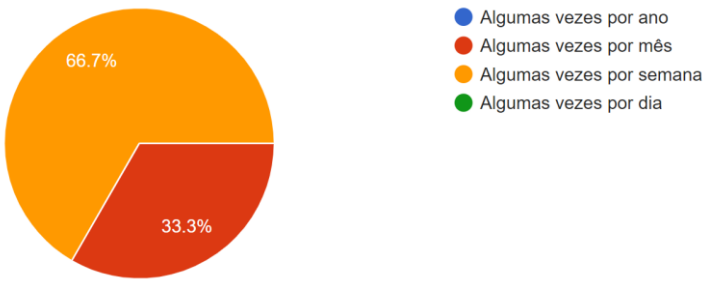


Figura 53 – Resultados de frequência de uso de GPS (público-alvo)

Relativamente, ao público externo os resultados determinaram que duas das cinco pessoas da amostra utilizam aplicações GPS algumas vezes por dia. Esta elevada frequência é derivada do uso de jogos de realidade aumentada e jogos que utilizam os mapas e cenários tridimensionais

com GPS para simular ambientes reais. Os restantes elementos usam com uma frequência de algumas vezes por mês. O gráfico circular pode ser verificado na Figura 54.

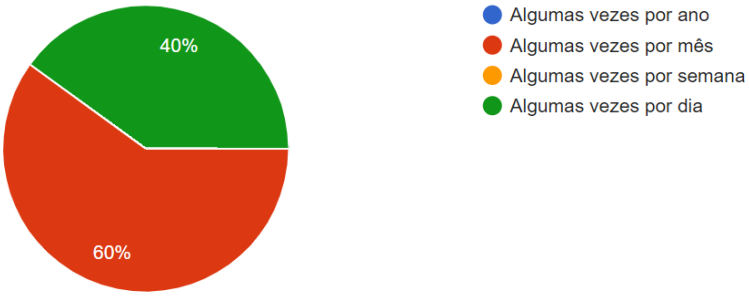


Figura 54 – Resultados de frequência de uso de GPS (público externo)

A segunda componente avalia a performance da aplicação, mais concretamente, verifica se a aplicação é intuitiva, adaptável face ao número de dados a representar, apresenta um número reduzido de bugs e disponibiliza informações de qualidade aos utilizadores, durante o decorrer das ações.

De acordo com o gráfico de barras da Figura 55 os três elementos internos consideraram a aplicação intuitiva, contudo com algumas falhas nomeadamente no processo de ajuste da cena e de informações dadas antes de cada ação. A votação dos elementos do público externo é quase homogénea sendo que 4/5 da amostra considera a aplicação bastante intuitiva. Esta componente foi avaliada segundo uma escala de 1 (Pouco intuitivo) a 5 (Muito intuitivo).

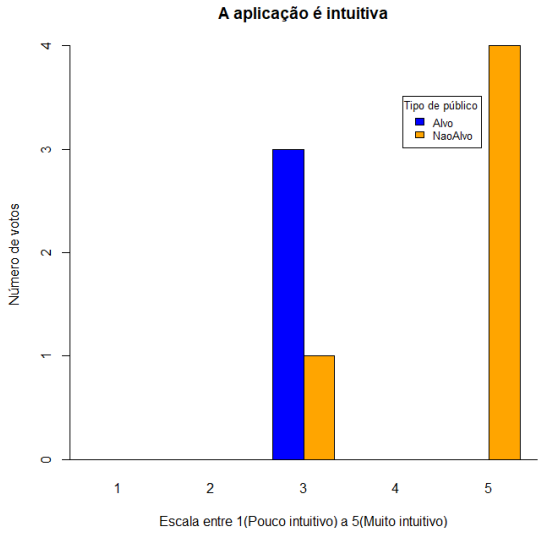


Figura 55 – Gráfico mostra se a aplicação é intuitiva

Um dos problemas principais da completagem de campo está associado à dificuldade em controlar grandes quantidades de informação. O uso do suporte tecnológico visa a combater este problema através de algoritmos de seleção de dados como acontece durante a criação dos cenários com realidade aumentada. Efetivamente, é possível destacar através do gráfico da Figura 56, que existe um grande contraste de opiniões relativamente à usabilidade da aplicação. O público-alvo considera que a aplicação é um pouco comprometida tendo em conta o número de dados (2 dos 3 elementos do grupo concordaram com a afirmação). A diminuição do processo de navegação do mapa foi o fator decisivo. Pelo contrário, o grupo externo considerou que a aplicação não era praticamente comprometida sendo que 80% discordou da afirmação.

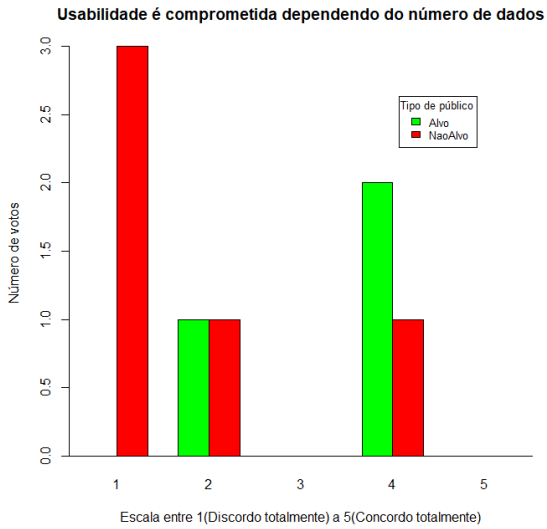


Figura 56 - Resultados da usabilidade dependendo do número de dados

Outro dos aspetos a considerar foi a quantidade e a qualidade das informações dadas pelo sistema antes e depois de cada ação. De acordo com os resultados, em ambos os grupos verificou-se que a qualidade das informações dadas antes das ações era significativamente menor do que o feedback dado depois. Apesar disso, os resultados são bastante positivos sendo que 66,7% dos clientes confirmam que a qualidade de informação fornecida é boa e bastante boa, antes e depois de cada processo respetivamente. A Tabela 19 descreve os valores obtidos para os dois grupos.

Tabela 19 – Quantidade e qualidade da informação dada

Perguntas	Número de respostas (Público-alvo)					Número de respostas (Público Externo)				
	1	2	3	4	5	1	2	3	4	5
O sistema apresenta informações relevantes antes do utilizador tomar uma ação.	-	1	-	2	-	-	-	1	1	3
O sistema apresenta feedback relevante depois do utilizador tomar uma ação.	-	-	1	1	1	-	-	-	-	5

Evidentemente, o número de erros de uma aplicação influencia a usabilidade e a interação entre o utilizador e o sistema e, portanto, foi também um dos aspetos a ter em conta no questionário. Quer o público interno como o público externo concordam que a aplicação não apresenta praticamente erros de programação, sendo que apenas um elemento de cada grupo não confirmou totalmente com a afirmação. O facto de não existirem erros permite aumentar significativamente a performance do sistema. A Figura 57 mostra os resultados dos dois grupos.

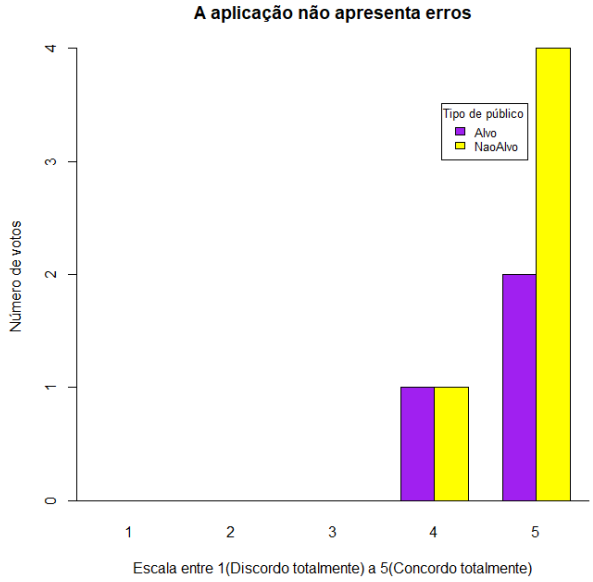


Figura 57 - Resultados relativos ao número de erros da aplicação

Finalmente, foi avaliada a performance da solução no seu âmbito geral. O público na sua globalidade considerou que a performance, conforme apresentada na Figura 58, era bastante alta avaliando-a com a pontuação 4 e 5 numa escala de 1 (Muito baixa) a 5 (Muito alta), com a particularidade de todos os intervenientes do grupo cliente terem votado na opção 4. Um dos poucos comentários mencionados relativamente à componente da performance foi a limitação de ter de reiniciar a aplicação sempre que é necessário voltar a calibrar o chão. Esta é uma das

limitações do *ARCore* em si, pois as alterações de luminosidade alteram a perceção do plano do chão. Este é um dos aspetos menos positivos da tecnologia e que não pode ser corrigido pela aplicação.

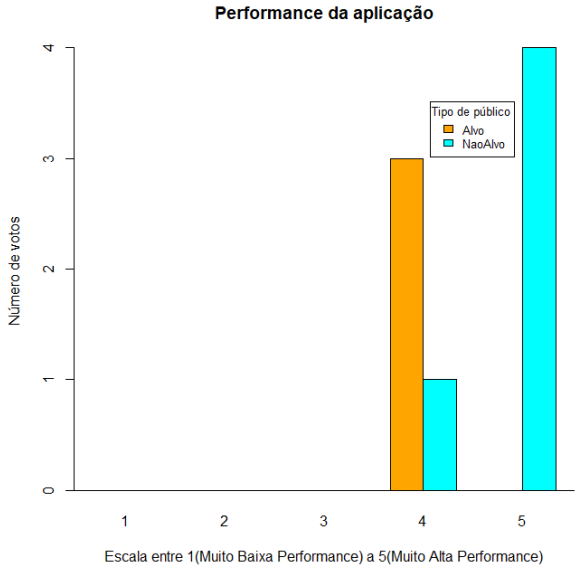


Figura 58 - Resultados da performance da aplicação

As duas componentes de avaliação seguintes são, respetivamente, os dois tipos de projeções usados para representar os dados cartográficos, cenários de realidade aumentada e mapa a duas dimensões.

O posicionamento preciso dos objetos foi um dos principais motivos para a criação do FieldAR. A Figura 59 mostra a opinião dos grupos relativamente ao posicionamento do cenário antes e depois do ajuste. Como é possível observar, os objetos antes do ajuste não se encontravam bem posicionados obtendo assim das classificações mais baixas. Cerca de 75% dos indivíduos das duas amostras avaliaram o posicionamento entre 2 e 3. Porém, depois do ajuste os resultados foram bastante positivos sendo que 100% das amostras concordaram com um ótimo posicionamento dos objetos.

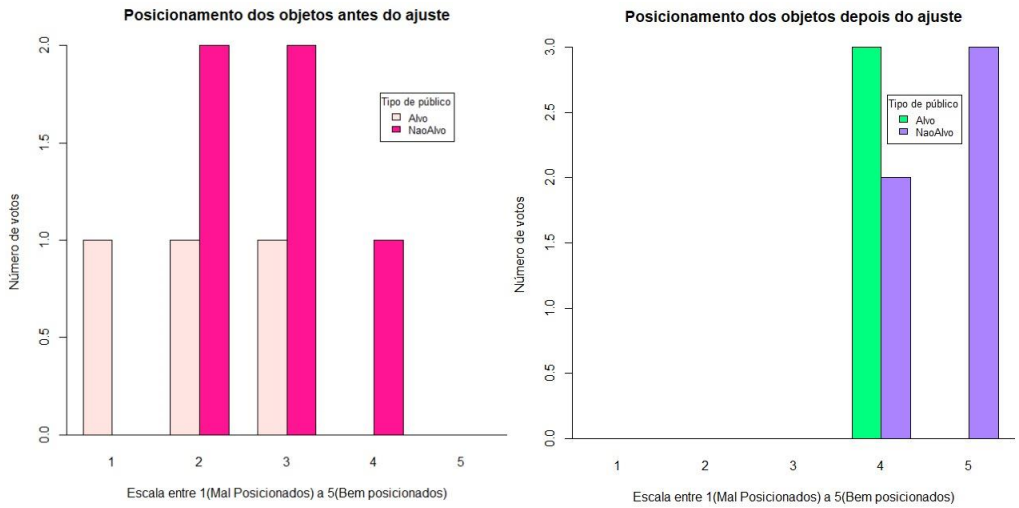


Figura 59 - Gráficos relativos ao posicionamento dos objetos

Além do posicionamento dos objetos foi necessário avaliar a visibilidade e veracidade dos símbolos e dos dados representativos de cada objeto e filtragens. No geral os resultados foram bastante positivos (Tabela 20) quer para o público-alvo como para o público externo obtendo quase sempre a pontuação máxima para cada questão. Estas classificações validam o bom enquadramento dos utilizadores no cenário e contribui para um controlo eficiente de informação cartográfica.

Tabela 20 - Resultados relativos às informações adicionais dos objetos

Perguntas	Número de respostas (Público-alvo)					Número de respostas (Público Externo)				
	1	2	3	4	5	1	2	3	4	5
Os objetos estão devidamente identificados por simbologia.	-	-	-	1	2	-	-	-	1	4
A simbologia dos objetos está bem visível.	-	-	-	-	3	-	-	-	2	3
Os dados dos objetos podem ser identificados eficazmente.	-	-	-	2	1	-	-	-	-	5
As filtragens funcionam corretamente para realidade aumentada.	-	-	-	-	3	-	-	-	-	5

Finalmente, foi averiguada a sobreposição dos objetos nos cenários de realidade aumentada. Este é dos fatores de maior relevância, dado que a validação por parte da amostra confirma o grau de importância desta aplicação, uma vez que a representação tradicional em papel apresenta grandes dificuldades em representar objetos sobrepostos. O gráfico da Figura 60 mostra as classificações obtidas para as duas amostras.

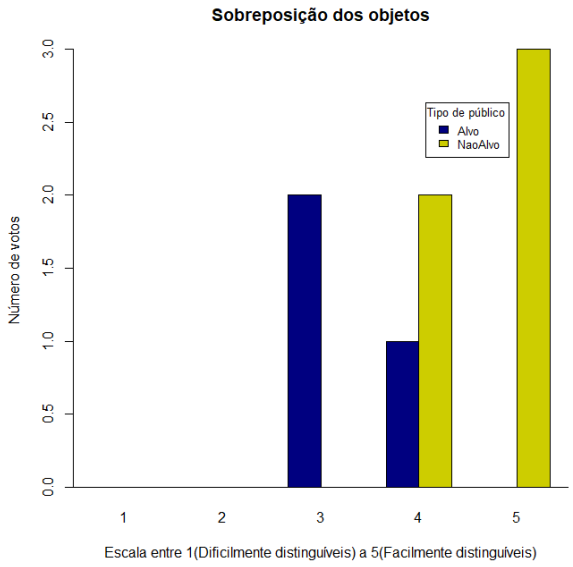


Figura 60 - Resultados da sobreposição de objetos

Cerca de 3/5 do grupo externo consideraram a sobreposição como totalmente distinguível e os restantes como bem distinguível. Em relação ao grupo cliente os resultados foram um pouco mais baixos, sendo que 2/3 da amostra classificaram a sobreposição como apenas razoavelmente distinguível. Apesar disso, os valores são bastante satisfatórios e confirmam a necessidade deste tipo de projeção.

Relativamente ao capítulo do mapa, a primeira questão abordada foi a facilidade na sua interpretação. De acordo com os votos da Figura 61, notou-se uma pequena discrepância de opiniões, sendo que, no caso dos clientes, uma das pessoas considerou a interpretação difícil, uma considerou razoável e a outra considerou fácil. Esta discrepância deve-se à influência das cores usadas no mapa, bem como da necessidade de informações adicionais. No entanto, a mediana de valores é satisfatória.

No caso do segundo público três das cinco pessoas consideraram a interpretaçao do mapa adequada e os restantes consideraram a interpretação aceitável mas com algumas falhas nomeadamente na identificação do utilizador no mapa.

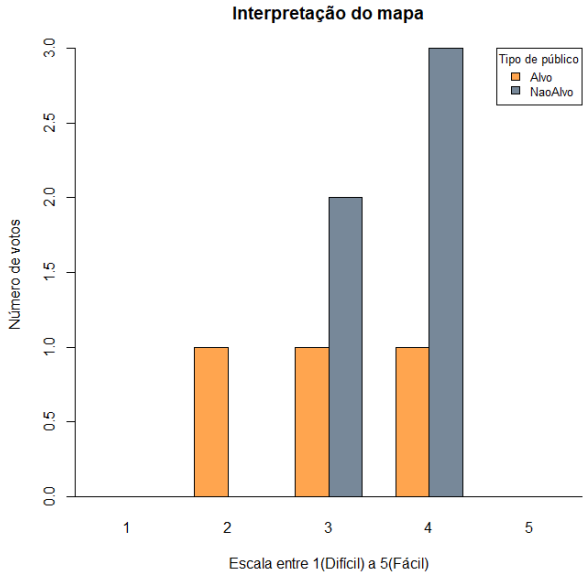


Figura 61 - Gráfico mostra a facilidade de interpretação do mapa

A navegação no mapa foi uma das poucas causas de alguma falta de eficiência da aplicação uma vez que 66.7% da amostra cliente avaliou a navegação como apenas razoavelmente eficiente e um dos clientes considerou a navegação pouco eficiente. Os valores do público externo são positivos pelo que rondam entre os 3 e 4 valores. Esta falta de eficiência afirmada pelo primeiro grupo é justificada pelas grandes quantidades de objetos no mapa. A Figura 62 mostra os resultados da navegação.

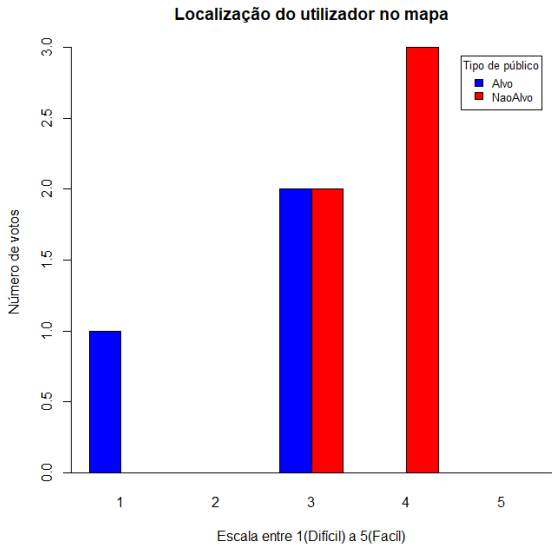


Figura 62 - Gráfico mostra a eficiência da navegação no mapa

Assim como nos cenários de realidade aumentada, as informações adicionais foram também avaliadas, ou seja, foi averiguada a quantidade de informação no mapa, a representação de trajetos e as filtragens. A Tabela 21 mostra as classificações para cada pergunta.

Tabela 21 - Classificação relativas às informações adicionais no mapa

Perguntas	Número de respostas (Público-alvo)					Número de respostas (Público Externo)				
	1	2	3	4	5	1	2	3	4	5
O mapa apresenta todos os dados cartográficos, incluindo as edições.	-	-	1	1	1	-	-	-	1	4
Os vários trajetos efetuados são registados corretamente.	-	-	-	2	1	-	-	-	2	3
As filtragens funcionam corretamente para o mapa.	-	-	-	1	2	-	-	-	-	5

Segundo as classificações obtidos foi verificado que o mapa tem um grande destaque como suporte à completagem de campo, dado que os resultados se encontram praticamente todos entre os 4 e 5 valores. A quantidade de informação pertinente no mapa e o trajeto permitem melhorar o enquadramento da pessoa no campo e as filtragens do mapa e permitem controlar a quantidade de informação que é representada reduzindo problemas de performance.

Outro dos aspetos que tinha, também, como objetivo principal combater a falta de enquadramento do utilizador, prevenindo-o assim de realizar alterações inadequadas, foi a inserção de um ícone representativo do utilizador e um cone representativo da orientação; através destes elementos e do desenho mapa foi avaliado o grau de facilidade de localização. O gráfico da Figura 63 mostra que 66.7% dos clientes internos acha aceitável e 33.3% acha fácil a localização. A classificação do público não alvo foi bastante díspar, contudo com melhores resultados.

No geral, estes resultados foram positivos, dada a quantidade de informação a representar numa tela de pequenas dimensões.

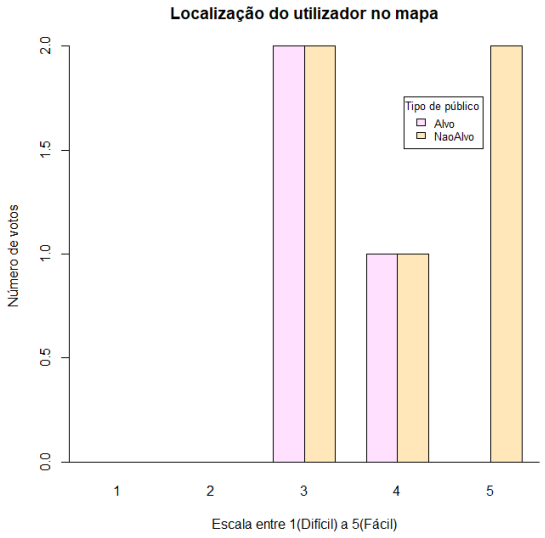


Figura 63 - Facilidade de enquadramento do utilizador

Por fim, foram avaliadas as ações de completagem como o ajuste do cenário tridimensional, adição e remoção de pontos e linhas e edição de categorias. Relativamente ao ajuste do cenário conclui-se que os resultados foram repartidos. Apesar disso, notou-se que houve dificuldade em realizar corretamente e rapidamente o ajuste. O grupo interno entendeu o ajuste relativamente acessível e o grupo externo mostrou facilidade no ajuste (verificar na Figura 64). Esta funcionalidade seria outro dos aspetos a melhorar de modo a aumentar a eficiência da atividade.

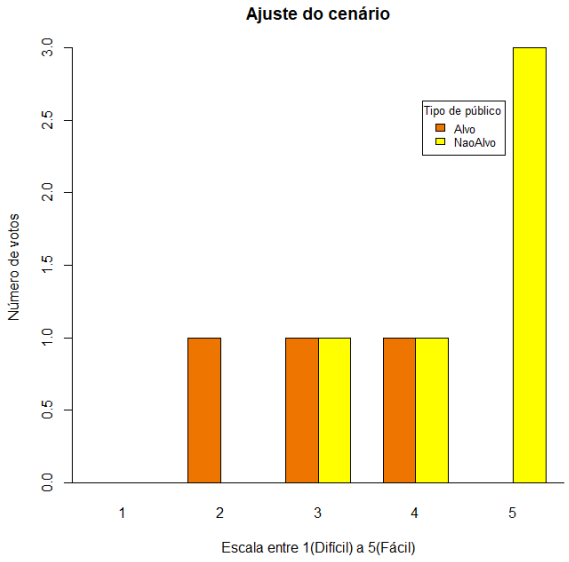


Figura 64 - Resultados da facilidade de ajuste da cena

As restantes ações de edição foram analisadas tendo em conta um fator de facilidade de execução, quer da adição e remoção de objetos, como de edição de categoria. Além disso, foi verificada a precisão de localização dos elementos adicionados (Tabela 22). Em relação à primeira componente constata-se que todas as ações foram fáceis de realizar, não havendo problemas de performance ou usabilidade. Os valores estão compreendidos entre os 4 e os 5 valores, à exceção de um voto. Relativamente à precisão dos pontos e linhas verifica-se uma pequena descida de classificação, porém mantendo valores positivos.

A edição de categoria também foi valorizada e validada obtendo dos maiores resultados do inquérito.

Tabela 22 - Valores relativos às ações de adição, remoção e alteração de objetos

Perguntas	Número de respostas (Público-alvo)					Número de respostas (Público Externo)				
	1	2	3	4	5	1	2	3	4	5
É fácil adicionar e remover linhas e pontos?	-	-	-	2	1	-	-	1	2	2
As linhas e pontos adicionados são registados com precisão.	-	-	2	1	-	-	-	-	-	5
É fácil de editar a categoria dos objetos?	-	-	-	1	2	-	-	-	-	5

Finalmente, considerando os resultados obtidos para os dois tipos de públicos conclui-se que a aplicação se encontra num estado de desenvolvimento avançado e com um grau de qualidade interessante. Alguns dos aspetos menos positivos, porém de maior relevância para a aplicação, foram corrigidos rapidamente sendo que os restantes poderão ser concluídos como trabalho futuro. Identificou-se alguns problemas de usabilidade na interpretação e navegação do mapa, tendo dos resultados mais baixos. A representação dos cenários de realidade aumentada e as ações de suporte foram bastante apreciadas pelos dois grupos afirmando mais uma vez a necessidade e a importância desta aplicação durante a produção de cartografia.

6.6 Testes unitários e de performance

Além da avaliação da usabilidade e da qualidade dos requisitos funcionais e não-funcionais da aplicação foi necessário analisar o código produzido. Para isso, foram realizados testes unitários para determinados métodos e algoritmos da aplicação.

Os testes unitários são processos que permitem testar pequenas quantidades de código, agrupadas por unidades independentes. No caso do Android Studio os testes são reunidos mediante o tipo de testes, ou seja, os testes feitos às componentes Android encontram-se separados dos testes java.

No caso da aplicação FieldAR, foram feitos testes apenas a métodos de grande importância para o negócio e que envolveram algoritmos de alguma complexidade, tal como, o posicionamento dos símbolos e construção dos polígonos a partir de triângulos. Um dos primeiros testes realizados para o cálculo da posição dos polígonos mediante o posicionamento do utilizador foi a verificação da construção dos limites de cada objeto cartográfico, ou seja, dadas as coordenadas de um objeto identificar os seus limites. Por exemplo, no caso de um ponto os limites seriam as coordenadas desse ponto, no caso de uma linha seria as coordenadas de início e fim da linha e no caso de um polígono fechado seriam as coordenadas mais externas desta figura. A Figura 65 mostra um teste feito para verificar se as coordenadas relativas dos limites de um polígono retângulo neste caso.

```
@Test
public void calculateSizePolygon(){
    Coordinate c1 = new Coordinate( x: 2, y: -1);
    Coordinate c2 = new Coordinate( x: 0, y: 0);
    Coordinate c3 = new Coordinate( x: 3, y: 4);
    Coordinate c4 = new Coordinate( x: 0, y: 6);
    ArrayList<Coordinate> listCoordinates = new ArrayList<Coordinate>();
    listCoordinates.add(c1);
    listCoordinates.add(c2);
    listCoordinates.add(c3);
    listCoordinates.add(c4);
    CartographicObject cartObject = new CartographicObject(listCoordinates,
            category: "10100104-Parque_Estacionamento", type: "Point", symbol: "PA", new Color());
    CalculateDistances calcDist = new CalculateDistances();
    double [] sizePol = calcDist.calculateSizePolygon(cartObject);
    double [] sizePoint = new double []{0,3,-1,6};
    assertArrayEquals(sizePoint,sizePol, delta: 0);
}
```

Figura 65 – Teste verifica coordenadas limites de um polígono

Inicialmente, foi criado um objeto cartográfico do tipo Polígono com uma lista de coordenadas. Seguidamente foi criado um objeto *CalculateDistance*() e foi feito o cálculo das coordenadas limites do retângulo. Por fim, foram comparadas as coordenadas calculadas com as coordenadas esperadas. Se as coordenadas fossem iguais o teste passaria com 100%. As coordenadas esperadas foram analisadas a partir de um gráfico.

Outro exemplo de teste foi o cálculo do ponto do polígono mais perto da posição do utilizador (verificar na Figura 66).

```
public void calculatePosSymb() {
    Coordinate c1 = new Coordinate( x: 2, y: -1);
    Coordinate c2 = new Coordinate( x: 0, y: 0);
    Coordinate c3 = new Coordinate( x: 3, y: 4);
    Coordinate c4 = new Coordinate( x: 0, y: 6);
    ArrayList<Coordinate> listCoordinates = new ArrayList<Coordinate>();
    listCoordinates.add(c1);
    listCoordinates.add(c2);
    listCoordinates.add(c3);
    listCoordinates.add(c4);
    CartographicObject cartObject = new CartographicObject(listCoordinates,
            category: "10100104-Parque_Estacionamento", type: "Point", symbol: "PA", new Color());
    CalculateDistances calcDist = new CalculateDistances();
    Vector3 vector3 = calcDist.calculatePosSymb(cartObject,new Vector3( v: -2, v1: 0, v2: 0));
    double [] v = new double[2];
    v[0] = vector3.x;
    v[1] = vector3.z;
    double [] vExp = {0,0};
    assertArrayEquals(vExp,v, delta: 0);
```

Figura 66 – Teste verifica o ponto do polígono relativamente ao utilizador

O processo de teste é muito semelhante ao anterior com a condicionante que é adicionada a posição do utilizador. As coordenadas esperadas são iguais às coordenadas calculadas, o que se conclui que para este caso o algoritmo de posicionamento do símbolo está correto.

Alguns exemplos de testes passados relativos à posição da simbologia podem ser apresentados na Figura 67.

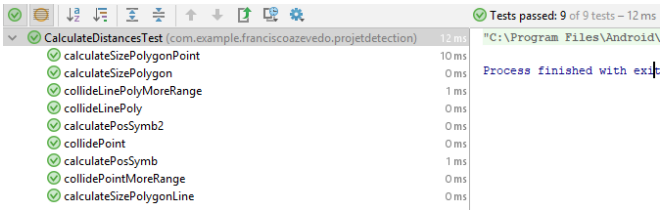


Figura 67 – Exemplo de testes passados corretamente

Devido à quantidade de informação manipulada e representada através das projeções e pelo facto de ser uma aplicação de cariz gráfico foi necessário realizar ao longo do projeto determinados testes de performance que consistiam na verificação das alterações de memória, CPU e energia ao longo do tempo. Este tipo de testes teve o suporte da ferramenta Android Profiler (ferramenta do Android Studio).

A Figura 68 mostra o desempenho do dispositivo num dado momento. É possível observar que quer o CPU, como a memória e a energia se mantêm relativamente constantes ao longo do tempo.

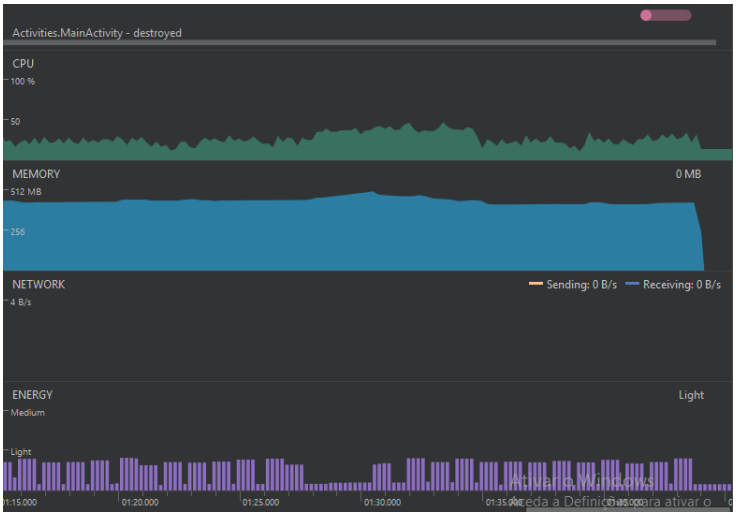


Figura 68 – Performance do dispositivo durante a execução

À medida que iam sendo inseridos os algoritmos de desenho de figuras e modelos era testado o comportamento da aplicação. Caso a aplicação sofresse alterações inesperadas e bastante significativas o algoritmo era alterado de modo a manter a performance do dispositivo. Um exemplo destas transformações surgiu devido a falhas na inserção dos símbolos nos cenários de realidade aumentada. A Figura 69 revela o aumento de memória despendida pelo dispositivo ao longo do tempo já com a funcionalidade de exibição dos símbolos.

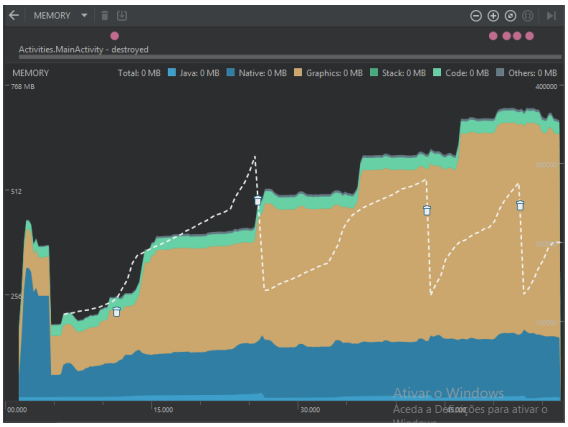


Figura 69 – Alterações de memória do dispositivo

Esta alteração dos dados permitiu verificar que o número de símbolos para cada objeto se multiplicava de x em x segundos, o que levou à reformulação do algoritmo. A Figura 70 revela a memória do dispositivo depois das alterações feitas, verificando que a memória usada agora foi linear ao longo da execução.

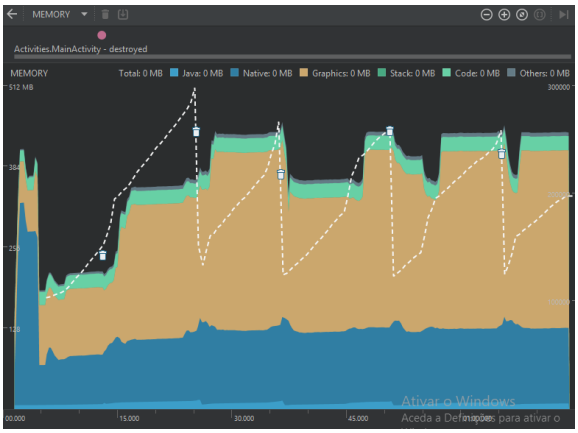


Figura 70 – Memória do dispositivo depois da reformulação do algoritmo

7 Conclusões

O capítulo das conclusões apresenta os objetivos concretizados, bem como as limitações da aplicação e trabalho futuro. Além disso, é também referida a experiência de trabalho adquirida na completagem de campo, através do qual foi possível identificar e analisar as dificuldades desta fase cartográfica.

Finalmente, é feita uma apreciação global, onde é expressada uma retrospetiva geral do projeto e onde são enumeradas as experiências obtidas de enriquecimento pessoal e profissional.

7.1 Objetivos realizados

O objetivo principal do projeto proposto consistiu no desenvolvimento de uma aplicação móvel de realidade aumentada que facilitasse e melhorasse os processos de completagem de campo e de edição na área de cartografia aplicada.

Dado como encerrado o desenvolvimento do projeto constata-se que a aplicação é capaz de mostrar cenários tridimensionais com realidade aumentada semelhantes às fotos aéreas tiradas nas posições geográficas correspondentes, bem como representar os dados através de um mapa a duas dimensões. Devido a uma falta de enquadramento por parte dos utilizadores, foram adicionadas componentes no cenário, como cores e simbologia. A visualização dos dados da aplicação é melhorada dada a possibilidade de navegação no mapa e de filtragens de informação.

Além das projeções, é também possível realizar as ações de edição da completagem como adição e remoção de objetos e visualização e alteração dos dados. Verificou-se ainda que estas ações são bastante práticas, rápidas e eficazes comparativamente com a metodologia tradicional.

Resumidamente, todos os objetivos foram cumpridos com distinção e rigor e, dada a eficiência no desenvolvimento das tarefas, foi possível adicionar novos requisitos funcionais e alterar o design da solução. A distinção visual entre os objetos iniciais e os objetos editados, a adição da orientação do utilizador em tempo real e a apresentação do trajeto feito durante a completação são alguns dos requisitos adicionados durante o processo.

Os objetivos não funcionais de usabilidade, adaptabilidade, eficiência e desempenho foram praticamente todos cumpridos destacando também a qualidade das *User Interfaces*.

7.2 Outros trabalhos realizados

A par do desenvolvimento do projeto realizou-se uma completagem de campo oficial, isto é, trabalho cartográfico para um dado cliente. Esta consistiu na identificação determinados tipos de objetos (tampas pluviais e de saneamento, postes de alta e baixa tensão, postes de iluminação, sarjetas, armários elétricos, entre outros) na zona industrial de Aveiro e alteração das plantas criadas a partir das fotos aéreas.

Esta oportunidade surgiu da necessidade de identificar e analisar todos os problemas/dificuldades que existem durante o processo.

Além da compreensão do negócio e das etapas adjacentes ao processo de completagem, foram também realizados de testes de campo pela aplicação a fim de encontrar possíveis erros de programação, falhas na usabilidade e representação dos dados.

Durante a atividade foram encontradas dificuldades por parte do utilizador a enquadrar-se no ambiente de realidade aumentada e falhas de representação de determinados objetos o que levou à necessidade de adicionar novos requisitos.

7.3 Limitações e trabalho futuro

O produto FieldAR apresenta, no entanto, algumas limitações internas e externas. As limitações externas não dependem do *developer,* mas sim de determinada tecnologia, algoritmo, biblioteca, etc.

Neste caso em particular, a tecnologia ARCore apresenta algumas limitações. Parte delas foram mencionadas anteriormente:

- interferência com diferentes tipos de luminosidade;
- dificuldade em detetar superfícies brilhantes e claras;
- perda do plano do chão dependendo da distância entre o utilizador e o plano inicial;
- dificuldade em representar grandes quantidades de modelos tridimensionais;

- aumento do erro de posicionamento de objetos face à posição inicial.

Outras desvantagens existentes a nível externo são os erros de calibração das bússolas e GPS. Apesar dos erros de GPS serem cada vez mais pequenos, devido à utilização de algoritmos mais avançados, o erro ainda é significativo, considerando a precisão necessária para aplicações cartográficas. Esta limitação apenas acontece antes do utilizador realizar o ajuste do cenário, contudo é tempo gasto e memória desperdiçada.

Por fim, a bateria do telemóvel é também uma limitação, dado que o processo de completagem pode demorar várias horas seguidas.

A nível interno, a aplicação demonstra alguma falta de eficiência em controlar os dois tipos de projeção em simultâneo, verificando-se algumas dificuldades em navegar eficazmente no mapa. Além disso, foram encontrados alguns aspetos menos positivos relativos à usabilidade, tais como, dificuldade em ajustar a cena, tempo despendido durante o ajuste e falta de informação antes da concretização das ações.

Relativamente a trabalho futuro, além da melhoria dos requisitos funcionais já existentes, a aplicação deverá ser adaptável aos vários tipos de ficheiros *geojson*, uma vez que existem estruturas diferentes dependendo das versões da *shapefile* e do tipo de conversão de ficheiros.

Finalmente, o produto numa fase de maior maturidade deverá consumir menor processador e memória durante a execução da aplicação. Além disso, a aplicação deverá ser submetida a um conjunto de testes a fim de determinar o grau de precisão dos objetos posicionados, bem como o tempo gasto para cada atividade.

7.4 Apreciação final

O projeto atual foi o resultado de todo um trabalho e investigação implementados durante o estágio. Os conhecimentos de engenharia de software adquiridos na licenciatura e mestrado de engenharia informática serviram de suporte para o desenvolvimento do produto, desde o planeamento até à execução dos testes.

Efetuando uma retrospetiva sob o ponto de vista final da aplicação, esta obteve um resultado bastante positivo tendo sido aprovada pelo QEF com uma percentagem aproximada de 91%. Todos os requisitos funcionais e não funcionais foram aprovados pelos dois públicos de teste através das respostas aos questionários. A organização cliente demonstrou também grande satisfação com o grau de qualidade do produto.

Este percurso proporcionou-me muitos momentos de enriquecimento, quer a nível profissional, como pessoal. A nível profissional o uso de tecnologias de ponta como o ARCore permitiram ter uma maior noção do mercado atual e das apostas das organizações neste ramo; e o estudo da cartografia permitiu não só aumentar o conhecimento geral, como também expandir novas

perspetivas. A nível pessoal o contacto profissional com uma organização de grande prestígio nacional permitiu-me melhorar competências sociais e comunicativas.

A aplicação FieldAR foi uma experiência muito gratificante que certamente terá impacto no desenvolvimento de novos produtos e serviços de suporte à cartografia e servirá de influência para a substituição das metodologias tradicionais adotadas na produção de cartografia em Portugal.

Referências

[1] G. Gartner e H. Huang, «Recent research developments in modern cartography in Europe», *International Journal of Cartography*, vol. 2, n. 1, pp. 1–5, Jan. 2016.

[2] Brandon Gaille, «13 Augmented Reality Industry Statistics, Trends & Analysis», *BrandonGaille.com*, 19-Jul-2018. [Online]. Disponível em: https://brandongaille.com/13-augmented-reality-industry-statistics-trends-analysis/. [Acedido: 12-Fev-2019].

[3] S. D. R. Santos, L. S. Delazari, e M. C. Brandalize, «Projeto cartográfico para representação tridimensional de redes de energia», *Boletim de Ciências Geodésicas*, vol. 19, n. 2, pp. 247–267, Jun. 2013.

[4] «How augmented reality is changing the way we shop», 17-Dez-2018. [Online]. Disponível em: https://www.verizon.com/about/our-company/fourth-industrial-revolution/how-augmented-reality-changing-way-we-shop. [Acedido: 20-Fev-2019].

[5] «2018 Update: A Timeline and History of Augmented Reality», *Colocation America*, 09-Mai-2018. [Online]. Disponível em: https://www.colocationamerica.com/blog/history-of-augmented-reality. [Acedido: 09-Fev-2019].

[6] Sales, «6 VR and AR Statistics: Shaping the Future of Augmented Reality with Data». [Online]. Disponível em: https://www.newgenapps.com/blog/6-vr-and-ar-statistics-shaping-the-future-of-augmented-reality-with-data. [Acedido: 09-Fev-2019].

[7] «Applications of Augmented Reality». [Online]. Disponível em: https://www.lifewire.com/applications-of-augmented-reality-2495561. [Acedido: 02-Jul-2019].

[8] J. Clover, «Google Maps Gaining AR Street View, "For You" Recommendations, and More». [Online]. Disponível em: https://www.macrumors.com/2018/05/08/google-maps-ar-street-view/. [Acedido: 21-Fev-2019].

[9] C. M. de Estarreja, «Informação à população: Atualização da Cartografia». [Online]. Disponível em: https://www.cm-estarreja.pt/noticias/5780. [Acedido: 13-Fev-2019].

[10] Amanda Briney, «How Did Map-Making Begin?», *ThoughtCo*. [Online]. Disponível em: https://www.thoughtco.com/the-history-of-cartography-1435696. [Acedido: 09-Fev-2019].

[11] environmentalscience, «Cartography: Introduction, History, and Uses | EnvironmentalScience.org». .

[12] J. Bobrich e S. Otto, «Search Exit Table des matières Index des auteurs Recherches Sortir», p. 4.

[13] «Importance of Geo referencing for the Mapping Data», *GIS Consortium (India) Pvt. Ltd*, 13-Mar-2018. .

[14] «TOP 10 STATISTICS...ON THE IMPACT OF DIGITAL TECHNOLOGIES», *Irosoft*, 04-Mai-2018. .

[15] J. Flanagan, «A Brief History of Augmented Reality», *Data Driven Investor*, 14-Nov-2018. .

[16] Augment, «Infographic: The History of Augmented Reality», *Augment News*, 12-Mai-2016. .

[17] L. Emgård e S. Zlatanova, «Design of an integrated 3D information model», p. 16.

[18] S. D. R. Santos, L. S. Delazari, e M. C. Brandalize, «Projeto cartográfico para representação tridimensional de redes de energia», *Boletim de Ciências Geodésicas*, vol. 19, n. 2, pp. 247–267, Jun. 2013.

[19] D. Schmalstieg e G. Reitmayr, «Augmented Reality as a Medium for Cartography», em *Multimedia Cartography*, W. Cartwright, M. P. Peterson, e G. Gartner, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 267–281.

[20] B. Lang, «6 Free Tools For Creating Your Own Maps», *Mapme*, 24-Dez-2015. .

[21] Pocket-lint, «10 best ARKit-enabled apps: The best augmented reality apps for iOS», *Pocket-lint*, 19-Mai-2018. [Online]. Disponível em: https://www.pocket-lint.com/apps/news/142315-best-arkit-enabled-augmented-reality-apps-for-ios-and-iphone. [Acedido: 09-Fev-2019].

[22] C. Gartenberg, «Google Maps is getting augmented reality directions and recommendation features», *The Verge*, 08-Mai-2018. [Online]. Disponível em: https://www.theverge.com/2018/5/8/17332480/google-maps-augmented-reality-directions-walking-ar-street-view-personalized-recommendations-voting. [Acedido: 09-Fev-2019].

[23] Blippar, «Welcome to AR City: Augmented Reality Maps and Navigation», *Blippar*. [Online]. Disponível em: https://www.blippar.com/blog/2017/11/06/welcome-ar-city-future-maps-and-navigation. [Acedido: 17-Fev-2019].

[24] Urbica, «How to Make Maps in Augmented Reality», *Urbica*, 14-Set-2018. .

[25] «Best Augmented Reality SDK for AR development in 2018 - 2019», *Thinkmobiles*, 20-Jan-2017. .

[26] «Newzoo: Smartphone Users Will Reach 3.6 Billion By 2020». [Online]. Disponível em: https://www.alistdaily.com/digital/newzoo-smartphone-users-3-6-billion-by-2020/. [Acedido: 02-Jul-2019].

[27] «IDC - Smartphone Market Share - OS». [Online]. Disponível em: https://www.idc.com/promo/smartphone-market-share/os. [Acedido: 02-Jul-2019].

[28] Andrew Williams, «What is Apple ARKit: Everything you need to know about iPhone AR», *Trusted Reviews*, 04-Jun-2018. [Online]. Disponível em: https://www.trustedreviews.com/news/what-is-apple-arkit-3286676. [Acedido: 09-Fev-2019].

[29] P. A. Koen *et al.*, «PROVIDING CLARITY AND A COMMON LANGUAGE TO THE “FUZZY FRONT END”».

[30] A. Serafim, «O Modelo de Cadeia de Valor de Michael Porter - Portal Gestão». [Online]. Disponível em: https://www.portal-gestao.com/artigos/6991-o-modelo-de-cadeia-de-valor-de-michael-porter.html. [Acedido: 09-Fev-2019].

[31] M. Amaral, «Cadeia de valor de Michael Porter: Como funciona?», *Casa da Consultoria*, 27-Out-2016. .

[32] M. Miesnieks, «How is ARCore better than ARKit?», *6D.ai*, 01-Set-2017. .

[33] «Mercator projection», *Wikipedia*. 04-Fev-2019.

[34] «Polygon triangulation», *Wikipedia*. 11-Jun-2018.

[35] G. Eder, M. Held, e P. Palfrader, «Parallelized ear clipping for the triangulation and constrained Delaunay triangulation of polygons», *Computational Geometry*, vol. 73, pp. 15–23, Ago. 2018.

[36] P. Escudeiro, *Avaliação da Qualidade em Conteúdos Digitais*. 2007.

Anexos

1. Cálculo das prioridades relativas para cada tecnologia de realidade aumentada segundo os critérios estabelecidos

Consumo	ARKit 2	ARCore	Vuforia	Wikitude	Prioridades relativas
ARKit 2	2/19	14/129	2/15	5/53	0,11037
ARCore	6/19	70/129	4/15	30/53	0,42278
Vuforia	1/19	10/129	1/15	3/53	0,06336
Wikitude	10/19	35/129	7/15	15/53	0,38683
Soma	19/2	129/70	15	53/15	

Confiabilidade	ARKit 2	ARCore	Vuforia	Wikitude	Prioridades relativas
ARKit 2	20/39	24/43	5/12	8/17	0,48955
ARCore	10/39	12/43	4/12	6/17	0,30544
Vuforia	4/39	3/43	1/12	1/17	0,07862
Wikitude	5/39	4/43	2/12	2/17	0,12639
Soma	39/20	43/12	12	17/2	

Ciclo de vida	ARKit 2	ARCore	Vuforia	Wikitude	Prioridades relativas
ARKit 2	2/13	2/10	3/23	4/25	0,16107
ARCore	1/13	1/10	2/23	3/25	0,09597
Vuforia	4/13	3/10	6/23	6/25	0,27714
Wikitude	6/13	4/10	12/23	12/25	0,46582
Soma	13/2	10	23/6	25/12	

Funcionalidades	ARKit 2	ARCore	Vuforia	Wikitude	Prioridades relativas
ARKit 2	28/53	4/7	7/16	8/17	0,50195
ARCore	14/53	2/7	6/16	6/17	0,31945
Vuforia	4/53	1/21	1/16	1/17	0,06110
Wikitude	7/53	2/21	2/16	2/17	0,11749
Soma	53/28	7/2	16	17/2	

Ambientes de desenvolvimento	ARKit 2	ARCore	Vuforia	Wikitude	Prioridades relativas
ARKit 2	2/21	3/41	2/16	14/129	0,10048
ARCore	8/21	12/41	6/16	35/129	0,32999
Vuforia	1/21	2/41	1/16	10/129	0,05910
Wikitude	10/21	24/41	7/16	70/129	0,51042
Soma	21/2	41/12	16	129/70	

2. QEF

q	D	Qi	Dimension	Qj	Wij (Factor Weight j in Dim i) [0,1]	Factor	rwjk (requirement weight k in Factor j) {2, 4, 6, 8, 10}	Requirement	wfk % requirement fulfillment k) [0,100]
				100	0,081	Realidade aumentada (AR Core)	10	FRA01 - Deteção de superfícies horizontais e verticais	100
							10	FRA02 - Seleção de uma superficie	100
							10	FRA03 - Criação de uma cena a partir do plano escolhido	100
				97,917	0,324	Construção do cenário	10	FCC01 - Representação de objetos a partir do ficheiro de dados	100
							10	FCC02 - Representação de objetos com transparência	100
							10	FCC03 - Representação de pontos no cenário	100
							10	FCC04 - Representação de linhas no cenário	100
							10	FCC05 - Representação de chão e tetos	100
							10	FCC06 - Representação de paredes	100
							10	FCC07 - Atualização do cenário ao fim de 10 segundos	100
							10	FCC08 - Inserção de simbologia para cada objeto cartográfico	100
							10	FCC09 - Visualização dos dados de cada objeto	100
							10	FCC10 - Representação dos objetos tendo em conta um alcance	100
							10	FCC11 - Adição de filtragens	100
							10	FCC12 - Seleção de objetos	75
							10	FCM01 - Representação dos objetos com transparência	100
							10	FCM02 - Representação de um contorno nos objetos editados	75
		95,95	Funcionalidade	96,875	0,216	Construção do mapa	10	FCM03 - Representação de pontos no mapa	100
							10	FCM04 - Representação de linhas no mapa	100
							10	FCM05 - Representação de polígonos no mapa	100
							10	FCM06 - Representação do utilizador no mapa	100
							10	FCM07 - Representação da orientação do utilizador no mapa	100
							10	FCM08 - Adição de filtragens	100
				100	0,054	Trajetos	10	FT01 - Criar Trajeto	100
							10	FT02 - Apagar Trajeto	100
91%	0,35			75	0,054	Navegação no mapa	10	FN01 - Zoom no mapa	75
							10	FN02 - Mover o mapa	75
				75	0,054	Ajuste do cenário	10	FC01 - Mover cenário	75
							10	FC02 - Rodar cenário	75
				100	0,135	Edição do cenário	10	FE01 - Edição da categoria do objeto cartográfico selecionado	100
							10	FE02 - Adição de novos pontos	100
							10	FE03 - Remover pontos da cena	100
							10	FE04 - Adição de linhas na cena	100
							10	FE05 - Remoção de linhas na cena	100
				100	0,081	Armazenamento dos dados	10	FA01 - Persistir novos objetos	100
							10	FA02 - Persistir objetos editados	100
							10	FA03 - Persistir trajetos	100
		90	Adaptabilidade	100	0,20	Resolução	10	AR01 - Aplicação ajusta-se ao ecrã do telemóvel	100
				100	0,20	Representação da cena	10	ARC01 - A aplicação mostra a cena corretamente dependendo do ficheiro de dados	100
				83,333	0,60	Conservação	10	AC01- O ficheiro de dados inicial não é alterado	100
							10	AC02 - O posicionamento e orientação da cena não é alterado com o tempo ou distância	50
							10	AC03 - A cena mantém se atualizada independentemente do número de execuções da aplicação	100
		75	Eficiência	75	0,13	Modelação	10	EM01 - A cena está bem definida e identificada	75
				75	0,88	Navegação	10	EN01 - O ajuste de posições é simples e fluida	50
							10	EN02 - O ajuste da rotação é eficiente e direta	100
							10	EN03 - O zoom do mapa é eficiente	50
							10	EN04 - A movimentaçao do mapa é simples e direta	50
							10	EN03 - A modelação da cena é rapido e sem breaks	75
							10	EN04 - A aplicação não tem erros	100
							10	EN05 - A navegação entre atividades e menus é rápido e sem interrupções	100

		77,27	Usabilidade	66,667	0,27	Menus	10	UM01 - Os menus estão bem estruturados e intuitivos	75
							10	UM02 - A usabilidade não é comprometida dependendo do número de dados	50
							10	UM03 - A aplicação tem em consideração todas as exceções	75
				91,25	0,36	Edição do cenário	10	UE01 - As ações do utilizador transmitem feedback	75
							10	UE02 - O utilizador controla o posicionamento e a orientação da cena e dos objetos	50
							10	UE03 - O mapa é alterado em tempo real	100
							10	UE04 - O cenário é alterado em tempo real	100
				81,25	0,36	Qualidade do conteúdo	10	UQ01 - Indicações e textos bem escritos e intuitivos	100
							10	UQ02 - A objetos são distinguiveis e bem identificados	75
							10	UC03 - O enquadramento é explícito	75
							10	UQ03 - As posições dos objetos são precisas depois do ajuste	75

3. Cálculos dos gráficos em R

```
#Tratamento dos dados

install.packages("readxl")

library("readxl")


Dados <-read_excel("C:/Users/FranciscoAzevedo/Desktop/InqueritoVirtualAdventure.xlsx")

View(Dados)

names(Dados)<-
c("TipoPublico","1.","2.","3.","4.","5.","6.","7.","8.","9.","10.","11.","12.","13.","14.","15.","16.
","17.","18.","19.","20.","21.","22.","23.","24.","25.","26.","27.","28.")


attach(Dados)

str(Dados)

#-----------------------------------------------------------------------------------------------------------

#Gráfico de barras

#Verifica se a aplicação é intuitiva

dadosIntuitiva<-factor(Dados$`6.`,levels=1:5)

barplot(table(Dados$TipoPublico,dadosIntuitiva),beside = TRUE,ylab = "Número de votos",

      xlab = "Escala entre 1(Pouco intuitivo) a 5(Muito intuitivo)", main="A aplicação é
intuitiva", col = c("blue","orange"))

legend("topright", inset=0.12, title="Tipo de público",

   c("Alvo","NaoAlvo"), fill=c("blue","orange"), cex=0.8)

abline(h=0)


#Gráfico de barras

#Verifica se a aplicação é intuitiva

dadosIntuitiva<-factor(Dados$`7.`,levels=1:5)
```

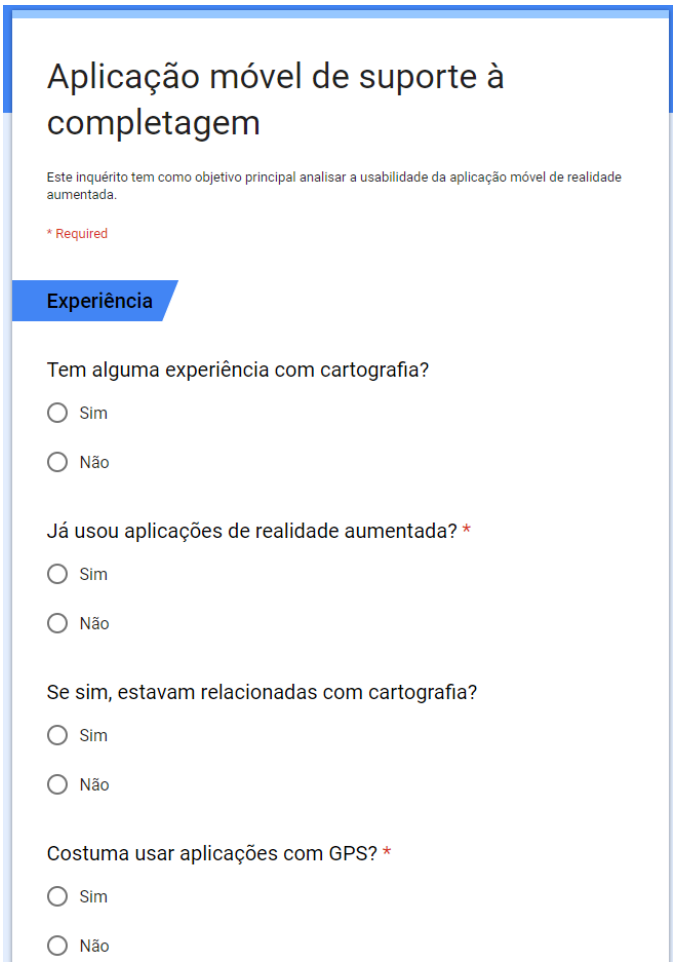
```
barplot(table(Dados$TipoPublico,dadosIntuitiva),beside = TRUE,ylab = "Número de votos",

      xlab = "Escala entre 1(Discordo totalmente) a 5(Concordo totalmente)",
main="Usabilidade é comprometida dependendo do número de dados", col =
c("green","red"))

legend("topright", inset=0.12, title="Tipo de público",

   c("Alvo","NaoAlvo"), fill=c("green","red"), cex=0.8)

abline(h=0)


#Gráfico de barras

#Verifica se a aplicação é intuitiva

dadosIntuitiva<-factor(Dados$`10.`,levels=1:5)

barplot(table(Dados$TipoPublico,dadosIntuitiva),beside = TRUE,ylab = "Número de votos",

      xlab = "Escala entre 1(Discordo totalmente) a 5(Concordo totalmente)", main="A
aplicação não apresenta erros", col = c("purple","yellow"))

legend("topright", inset=0.12, title="Tipo de público",

   c("Alvo","NaoAlvo"), fill=c("purple","yellow"), cex=0.8)

abline(h=0)


#Gráfico de barras

#Verifica se a aplicação é intuitiva

dadosIntuitiva<-factor(Dados$`11.`,levels=1:5)

barplot(table(Dados$TipoPublico,dadosIntuitiva),beside = TRUE,ylab = "Número de votos",

      xlab = "Escala entre 1(Muito Baixa Performance) a 5(Muito Alta Performance)",
main="Performance da aplicação", col = c("orange","cyan"))

legend("topright", inset=0.12, title="Tipo de público",

   c("Alvo","NaoAlvo"), fill=c("orange","cyan"), cex=0.8)

abline(h=0)

#Gráfico de barras
```

```
#Verifica se a aplicação é intuitiva

dadosIntuitiva<-factor(Dados$`12.`,levels=1:5)

barplot(table(Dados$TipoPublico,dadosIntuitiva),beside = TRUE,ylab = "Número de votos",

      xlab = "Escala entre 1(Mal Posicionados) a 5(Bem posicionados)", main="Posicionamento dos objetos antes do ajuste", col = c("mistyrose1","deeppink"))

legend("topright", inset=0.12, title="Tipo de público",

   c("Alvo","NaoAlvo"), fill=c("mistyrose1","deeppink"), cex=0.8)

abline(h=0)


#Gráfico de barras

#Verifica se a aplicação é intuitiva

dadosIntuitiva<-factor(Dados$`13.`,levels=1:5)

barplot(table(Dados$TipoPublico,dadosIntuitiva),beside = TRUE,ylab = "Número de votos",

      xlab = "Escala entre 1(Mal Posicionados) a 5(Bem posicionados)", main="Posicionamento dos objetos depois do ajuste", col = c("springgreen","mediumpurple1"))

legend("topright", inset=0.12, title="Tipo de público",

   c("Alvo","NaoAlvo"), fill=c("springgreen","mediumpurple1"), cex=0.8)

abline(h=0)


#Gráfico de barras

#Verifica se a aplicação é intuitiva

dadosIntuitiva<-factor(Dados$`16.`,levels=1:5)

barplot(table(Dados$TipoPublico,dadosIntuitiva),beside = TRUE,ylab = "Número de votos",

      xlab = "Escala entre 1(Dificilmente distinguiveis) a 5(Facilmente distinguiveis)", main="Sobreposição dos objetos", col = c("navy","yellow3"))

legend("topright", inset=0.12, title="Tipo de público",

   c("Alvo","NaoAlvo"), fill=c("navy","yellow3"), cex=0.8)

abline(h=0)
```

```
#Gráfico de barras

#Verifica se a aplicação é intuitiva

dadosIntuitiva<-factor(Dados$`19.`,levels=1:5)

barplot(table(Dados$TipoPublico,dadosIntuitiva),beside = TRUE,ylab = "Número de votos",

      xlab = "Escala entre 1(Dificil) a 5(Facil)", main="Interpretação do mapa", col =
c("tan1","lightslategrey"))

legend("topright", inset=0.02, title="Tipo de público",

   c("Alvo","NaoAlvo"), fill=c("tan1","lightslategrey"), cex=0.8)

abline(h=0)


#Gráfico de barras

#Navegação no mapa

dadosIntuitiva<-factor(Dados$`21.`,levels=1:5)

barplot(table(Dados$TipoPublico,dadosIntuitiva),beside = TRUE,ylab = "Número de votos",

      xlab = "Escala entre 1(Difícil) a 5(Facíl)", main="Localização do utilizador no mapa", col =
c("blue","red"))

abline(h=0)

legend("topright", inset=0.02, title="Tipo de público",

   c("Alvo","NaoAlvo"), fill=c("blue","red"), cex=0.8)


#Gráfico de barras

#Verifica se a aplicação é intuitiva

dadosIntuitiva<-factor(Dados$`22.`,levels=1:5)

barplot(table(Dados$TipoPublico,dadosIntuitiva),beside = TRUE,ylab = "Número de votos",

      xlab = "Escala entre 1(Dificil) a 5(Facil)", main="Localização do utilizador no mapa", col =
c("thistle1","wheat1"))
```

```
legend("topright", inset=0.12, title="Tipo de público",
   c("Alvo","NaoAlvo"), fill=c("thistle1","wheat1"), cex=0.8)
abline(h=0)

#Gráfico de barras
#Verifica se a aplicação é intuitiva
dadosIntuitiva<-factor(Dados$`25.`,levels=1:5)
barplot(table(Dados$TipoPublico,dadosIntuitiva),beside = TRUE,ylab = "Número de votos",
      xlab = "Escala entre 1(Dificil) a 5(Facil)", main="Ajuste do cenário", col =
c("darkorange2","yellow"))
legend("topright", inset=0.12, title="Tipo de público",
   c("Alvo","NaoAlvo"), fill=c("darkorange2","yellow"), cex=0.8)
abline(h=0)
```

4. Inquérito



Aplicação móvel de suporte à completagem

Este inquérito tem como objetivo principal analisar a usabilidade da aplicação móvel de realidade aumentada.

* Required

Experiência

Tem alguma experiência com cartografia?

- Sim
- Não

Já usou aplicações de realidade aumentada? *

- Sim
- Não

Se sim, estavam relacionadas com cartografia?

- Sim
- Não

Costuma usar aplicações com GPS? *

- Sim
- Não

Se sim com que frequência?

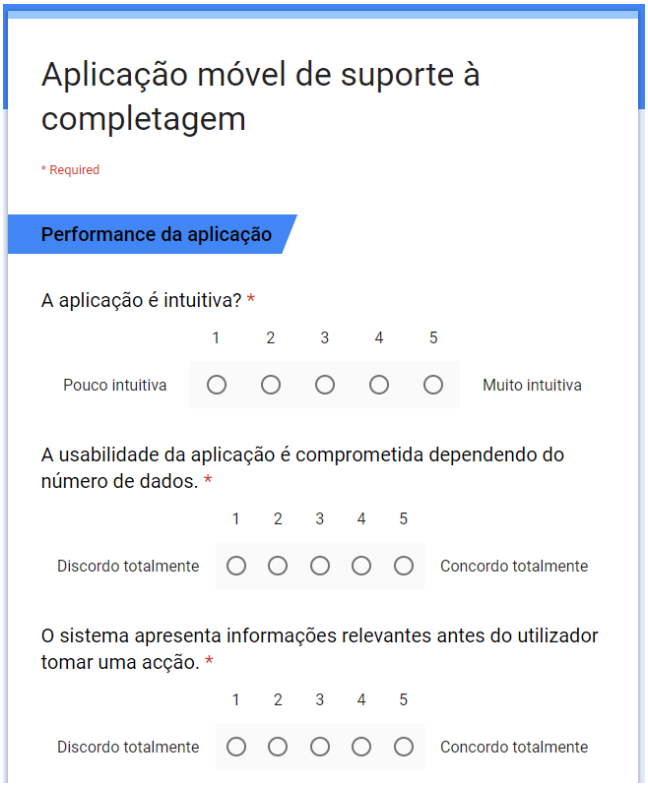
- Algumas vezes por ano
- Algumas vezes por mês
- Algumas vezes por semana
- Algumas vezes por dia

Tem algum comentário relativo às aplicações de cartografia e de realidade aumentada que conhece?

Your answer

NEXT

Never submit passwords through Google Forms.



Aplicação móvel de suporte à completagem

* Required

Performance da aplicação

A aplicação é intuitiva? *

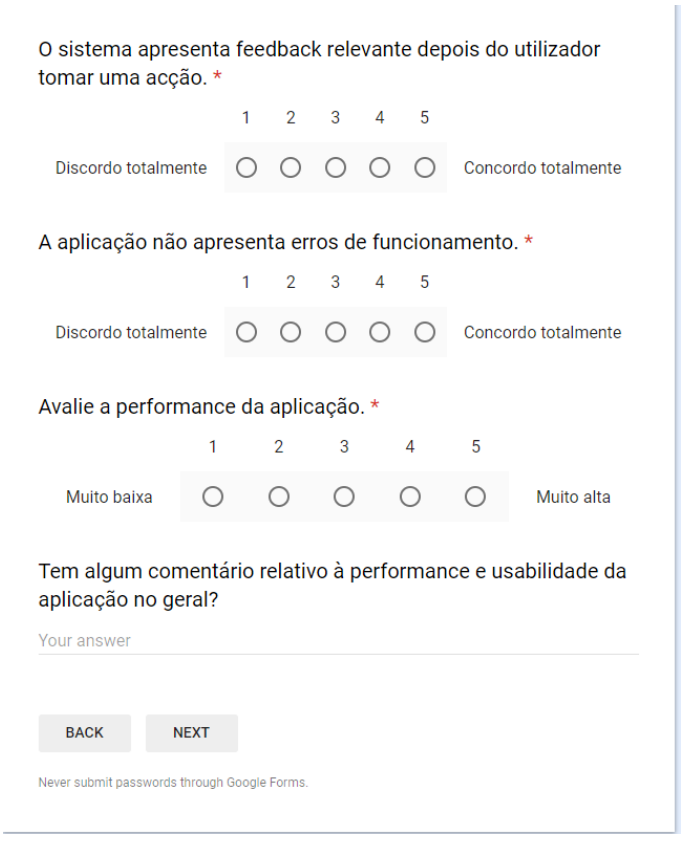
	1	2	3	4	5	
Pouco intuitiva	○	○	○	○	○	Muito intuitiva

A usabilidade da aplicação é comprometida dependendo do número de dados. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

O sistema apresenta informações relevantes antes do utilizador tomar uma acção. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente



O sistema apresenta feedback relevante depois do utilizador tomar uma acção. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

A aplicação não apresenta erros de funcionamento. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

Avalie a performance da aplicação. *

	1	2	3	4	5	
Muito baixa	○	○	○	○	○	Muito alta

Tem algum comentário relativo à performance e usabilidade da aplicação no geral?

Your answer

BACK NEXT

Never submit passwords through Google Forms.

Aplicação móvel de suporte à completagem

* Required

Realidade aumentada

Antes do ajuste do cenário, os objetos estão bem posicionados? *

	1	2	3	4	5	
Mal posicionados	○	○	○	○	○	Bem posicionados

Depois do ajuste do cenário, os objetos estão bem posicionados? *

	1	2	3	4	5	
Mal posicionados	○	○	○	○	○	Bem posicionados

Os objetos estão devidamente identificados por simbologia. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

A simbologia dos objectos está bem visível. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

Os objetos sobrepostos são facilmente distinguíveis. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

Os dados dos objetos podem ser identificados eficazmente. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

As filtragens funcionam corretamente para realidade aumentada. *

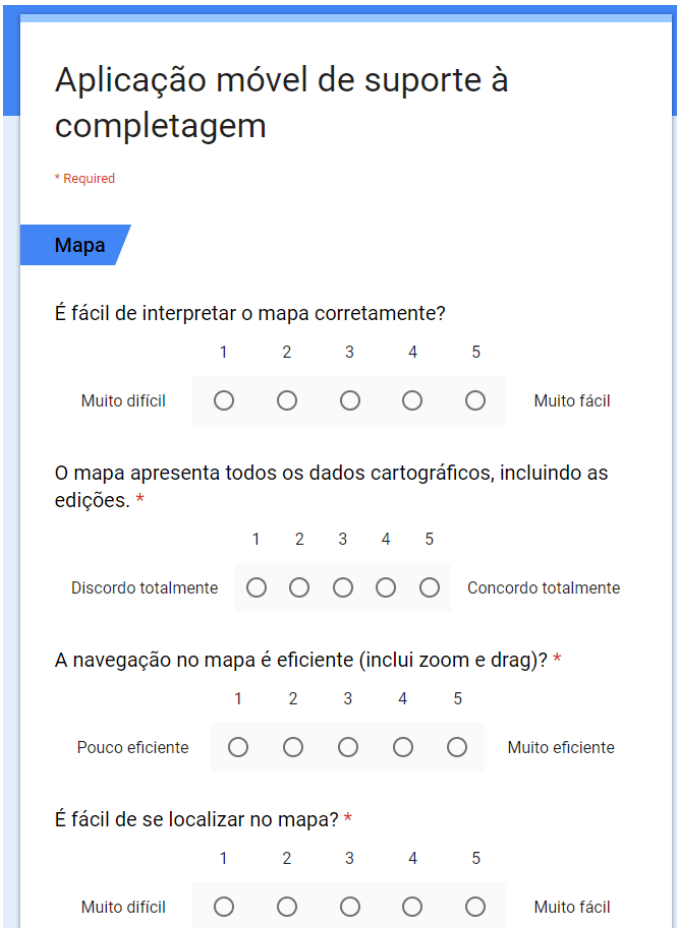
	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

Tem algum comentário sobre alguma questão relativa aos cenários de realidade aumentada e respetivas ações?

Your answer

BACK NEXT

Never submit passwords through Google Forms.



Aplicação móvel de suporte à completagem

* Required

Mapa

É fácil de interpretar o mapa corretamente?

	1	2	3	4	5	
Muito difícil	○	○	○	○	○	Muito fácil

O mapa apresenta todos os dados cartográficos, incluindo as edições. *

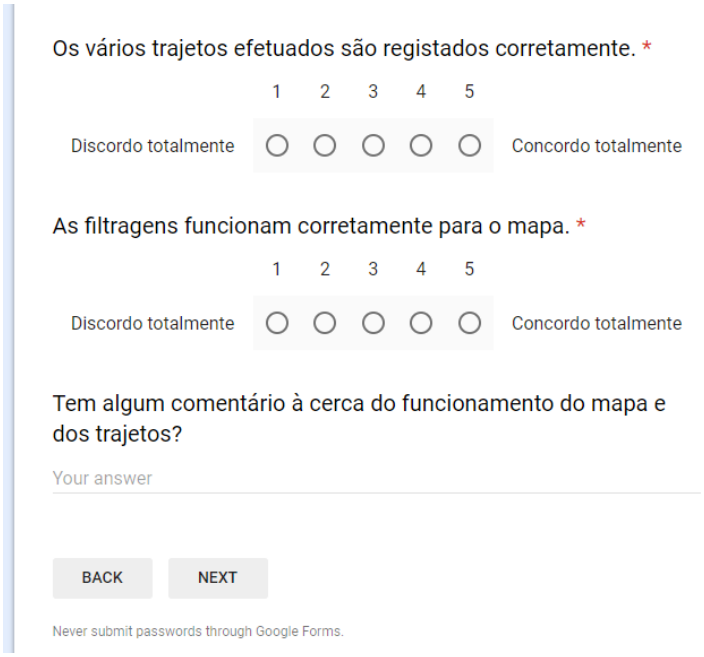
	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

A navegação no mapa é eficiente (inclui zoom e drag)? *

	1	2	3	4	5	
Pouco eficiente	○	○	○	○	○	Muito eficiente

É fácil de se localizar no mapa? *

	1	2	3	4	5	
Muito difícil	○	○	○	○	○	Muito fácil



Os vários trajetos efetuados são registados corretamente. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

As filtragens funcionam corretamente para o mapa. *

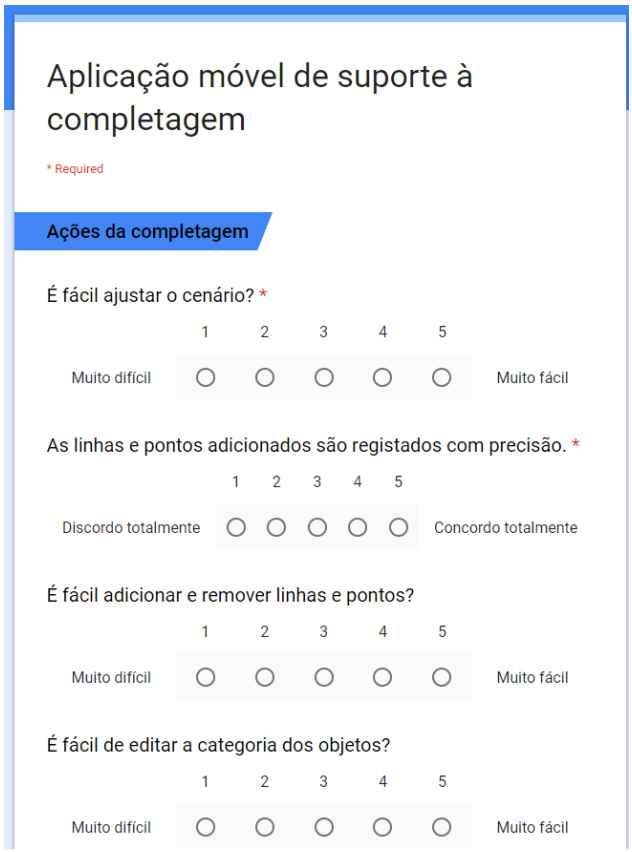
	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

Tem algum comentário à cerca do funcionamento do mapa e dos trajetos?

Your answer

BACK NEXT

Never submit passwords through Google Forms.



Aplicação móvel de suporte à completagem

* Required

Ações da completagem

É fácil ajustar o cenário? *

	1	2	3	4	5	
Muito difícil	○	○	○	○	○	Muito fácil

As linhas e pontos adicionados são registados com precisão. *

	1	2	3	4	5	
Discordo totalmente	○	○	○	○	○	Concordo totalmente

É fácil adicionar e remover linhas e pontos?

	1	2	3	4	5	
Muito difícil	○	○	○	○	○	Muito fácil

É fácil de editar a categoria dos objetos?

	1	2	3	4	5	
Muito difícil	○	○	○	○	○	Muito fácil

Tem algum comentário à cerca do tempo despendido nas ações de completagem (adição e remoção de pontos e linhas, editar categoria e cena)?

Your answer

BACK NEXT

Never submit passwords through Google Forms.

Aplicação móvel de suporte à completagem

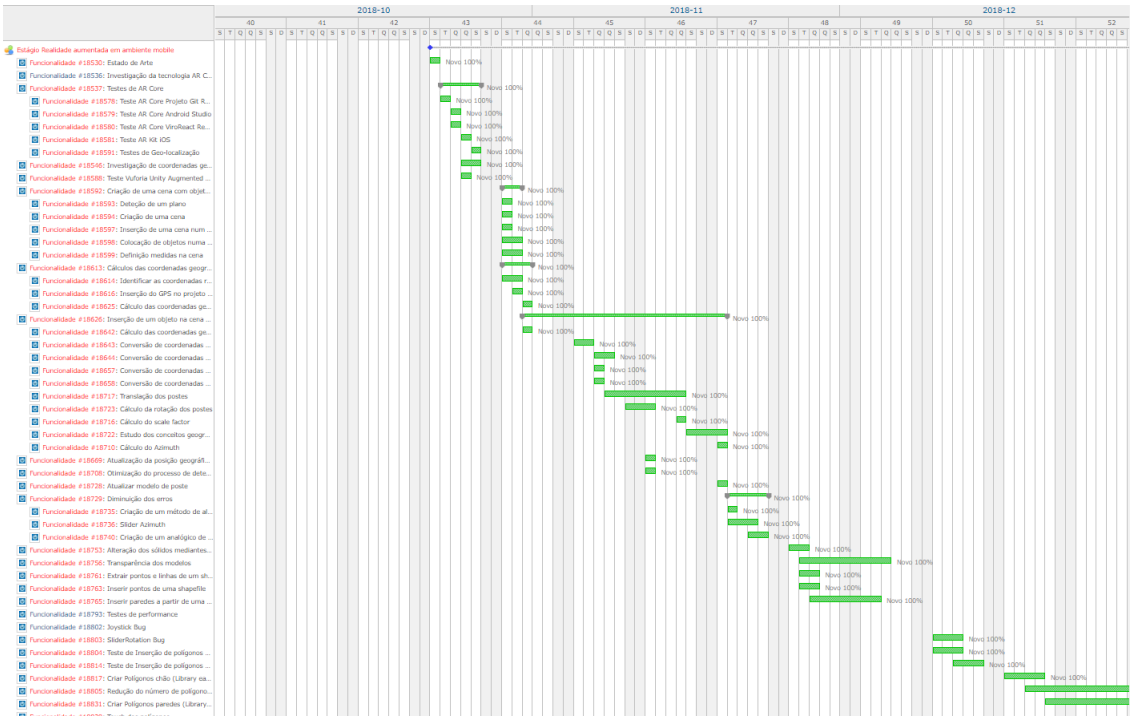
Tem algum comentário sobre alguma questão que não tenha sido abordada neste inquérito? Partilhe aqui connosco:

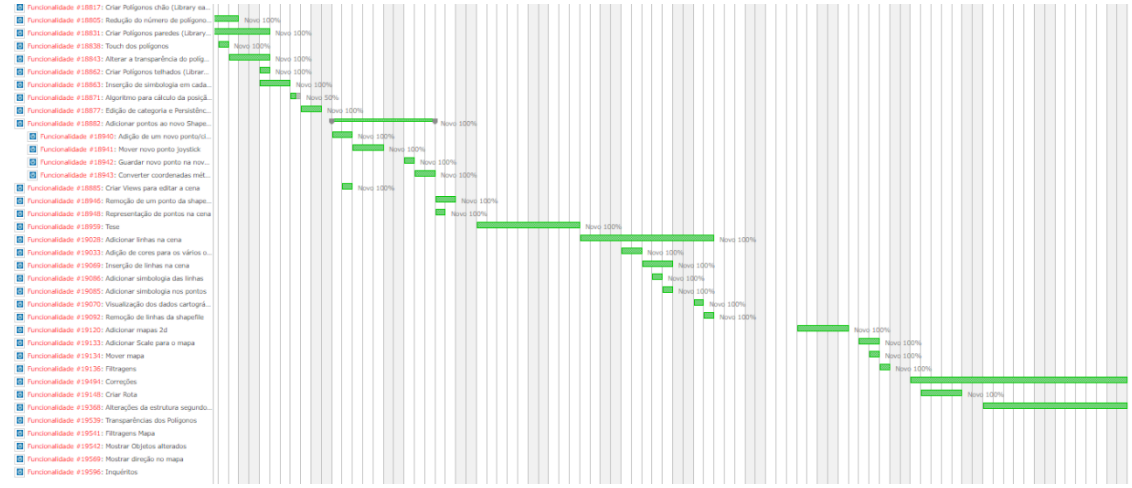
Your answer

BACK SUBMIT

Never submit passwords through Google Forms.

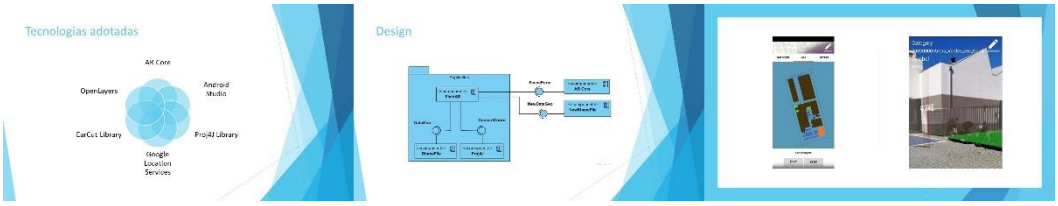
5. Diagrama de Gantt com planeamento





6. Apresentação intermédia PowerPoint


Sobreposição de dados cartográficos
A sobreposição de objetos prejudica a visualização e interpretação das plantas (ex.: uma árvore isolada e uma tampa de saneamento sobrepostas).
Cria a necessidade de utilização de apontamentos, provocando assim maior poluição visual e complexidade.
Dificuldade em controlar grandes quantidades de informação
Enorme densidade de informação provocada pelas alterações nos dados em ambientes reduzidos prejudicam posteriormente a sua leitura e utilização.
Objetivos
Aplicação móvel
Realidade aumentada
Automatização de processos da completagem
Representação gráfica de cenários
Ações de edição de objetos
Controlo de dados
Abordagem 1/2
Aplicação móvel otimiza ações de enquadramento e edição de objetos cartográficos através de realidade aumentada.
Auxílio de mapa 2D em permanente atualização e representação de trajetos.
Uso de realidade aumentada e conversões entre sistemas de coordenadas.
Sistema preparado para qualquer eventualidade, rápido feedback e informação detalhada de cada objeto cartográfico.
Abordagem 2/2
Aplicação móvel caracterizada pelas representações (AR e 2D) e ficheiros em memória.
Distinção dos dados através de modelos (realidade aumentada) e informação relevante.
Armazenamento dos dados na memória externa do dispositivo móvel:


Tecnologias adotadas
AR Core
OpenLayers
Android Studio
CarCut Library
ProjAU Library
Google Location Services
Design