Mestrado em Engenharia Informática
Estágio
Relatório Final

Aplicação de Realidade Virtual para Conteúdo Multimédia

Junho, 2016

Bruno Madureira
brunojm@student.dei.uc.pt

Orientador do DEI:
Prof. Tiago Baptista
baptista@dei.uc.pt

FCTUC DEPARTAMENTO
DE ENGENHARIA INFORMÁTICA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Mestrado em Engenharia Informática
Estágio
Relatório Final

Aplicação de Realidade Virtual para Conteúdo Multimédia

Junho, 2016

Bruno Madureira
brunojm@student.dei.uc.pt

Orientador do DEI:
Prof. Tiago Baptista
baptista@dei.uc.pt

Orientador da empresa:
Eng. Filipe Brunido
filipe.brunido@wit-software.com

Júri Arguente:
Prof. Luis Macedo
macedo@dei.uc.pt

Júri Vogal:
Prof. Pedro Abreu
pha@dei.uc.pt

FCTUC DEPARTAMENTO
DE ENGENHARIA INFORMÁTICA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Resumo

A realidade virtual é uma tecnologia já bastante antiga que, no entanto, nunca atingiu a maturidade nem o sucesso comercial. Presentemente esta tecnologia encontra-se a atravessar um momento de vários avanços tecnológicos e cada vez mais surgem previsões que vai mudar drasticamente a vida das pessoas num futuro muito próximo. A realidade virtual pretende imergir o seu utilizador num mundo virtual que lhe pareça real. Este processo é conseguido através da simulação de sensações reais através do uso de *software* e *hardware* construídos para o efeito.

O presente documento foi realizado no âmbito da disciplina de Dissertação/Estágio do Mestrado em Engenharia Informática da Universidade de Coimbra. O estágio foi realizado na WIT-Software e consistiu na construção de uma prova de conceito, envolvendo a aplicação de realidade virtual, em conteúdo multimédia.

O objetivo deste estágio é o desenvolvimento de uma prova de conceito, constituída por uma plataforma *mobile* para consumo de conteúdo multimédia em realidade virtual. Esta plataforma foi desenvolvida para dispositivos iOS de 4.7 polegadas e suportada por dispositivos de realidade virtual do tipo "Google Cardboard".

Com o desenvolvimento desta plataforma espera-se compreender melhor como aplicar a realidade virtual num contexto móvel, ajudando a empresa a estar preparada para lidar com o mercado emergente de *V.R.*

Palavras-Chave

Action Camera, iOS, Mobile V.R., OpenGL, PowerPoint, Virtual Reality, V.R. Engine, Video Player, Music Player, GLSL, OOXML, Objective-C, C++.

Abstract

Virtual Reality is a technology with some years that, however has never reached neither maturity neither commercial sucess. Nowadays this technology is going through a moment of technological advances and it's predicted that it will drasticly change the life of people in a near future. Virtual reality aims to immerge its user in a virtual world that it appears to be real to him. This process is reached by the simulation of real sensations by the use of software and hardware built for the effect.

This document has been produced within the subject of Dissertation / Intership in Computer Science from the University of Coimbra. The intership was held in WIT-Software and consisted in building a proof of concept, concerning the application of virtual reality application in multimedia content.

The aim of this internship is the development of a proof of concept, constituted by a mobile platform for the consumption of multimedia content in virtual reality. This platform was developed for 4.7 inch iOS devices and supported by virtual reality devices of the type "Google Cardboard".

With the development of this platform, it's expected to be learned how to apply virtual reality in a mobile context, helping the company to be prepared to deal with the emergent V.R. market.

Keywords

Action Camera, iOS, Mobile V.R., OpenGL, PowerPoint, Virtual Reality, V.R. Engine, Video Player, Music Player, GLSL, OOXML, Objective-C, C++.

Índice

Lista de Figuras

Lista de Tabelas

Anexos

Anexo A- Arquitetura do motor V.R.

Anexo B-Software Licenses.

Anexo C- Testes realizados.

Anexo D- Diagrama de Gant 1º Semestre.

Anexo E- Diagrama de Gant 2º Semestre.

Lista de Acrónimos

V.R.	Virtual Reality
API	Aplication Programming Interface
SDK	System Devolopment Kit
OLED	Organic Light Emiting Diode
HDMI	High Definition Multimedia Interface
GLSL	OpenGl Shading Language
XML	eXtensible Markup Language
OOXML	Office Open XML
XPATH	XML Path Language
UI	User Interface
UX	User Experience

Capítulo 1
Introdução

O desenvolvimento deste projeto está incluído no âmbito da disciplina de estágio do Mestrado em Engenharia Informática da Faculdade de Ciências e Tecnologia da Universidade de Coimbra. O trabalho ocorreu na empresa WIT-Software, sob a orientação do Professor Doutor Tiago Baptista, do DEI e do Engenheiro Filipe Brunido, da WIT-Software.

1.1. A instituição

A WIT-Software (1) é uma empresa especializada no desenvolvimento de aplicações para o mundo das aplicações móveis.

Fundada em 2001, como *spin-off* da Universidade de Coimbra, a WIT-Software é hoje uma empresa em franca expansão que tem vindo a consolidar uma posição robusta no mercado mundial. A empresa conta já com escritórios em Coimbra, Lisboa, Porto, Leiria, Reading (UK) e San José (EUA).

1.2. Contextualização

A realidade Virtual (*V.R.*) (2) é uma tecnologia bastante antiga, que ao longo dos tempos passou quer por períodos de forte aposta, quer por períodos de esquecimento. Há vários anos, especialmente desde o início da década de 50, que existem vários esforços para implementar uma plataforma de realidade virtual, no entanto nenhuma delas vingou no mercado. Mais recentemente tem-se assistido a um reaparecimento do entusiasmo por esta tecnologia, esperando-se que o público geral adquira bastante interesse, tornando-se uma tecnologia comum no dia-a-dia.

A realidade virtual permite proporcionar sensações reais através de meios eletrónicos. Um dos sentidos mais estimulados é a visão, já que o ser humano obtém parte da percepção que tem do mundo a partir deste sentido. Para estimular a visão usando *V.R.* é gerada uma imagem 3D, que vai dar a impressão do real, através do uso de equipamento eletrónico. A realidade virtual tem aplicação em vários campos como a saúde, ensino e entretenimento.

1.3. O projeto

Este estágio visa desenvolver uma aplicação de realidade virtual, em ambiente móvel na plataforma iPhone com ajuda de um dispositivo de realidade virtual do tipo "Google CardBoard", mais concretamente ZEISS VR One. Esta aplicação permite ao utilizador consumir conteúdo multimédia de forma mais imersiva do que conseguiria sem o uso de *V.R,* sendo capaz de reproduzir fotos, música, vídeos e PowerPoint em realidade virtual. Grande parte do esforço deste estágio, será explorar as potencialidades desta tecnologia.

1.4. Estrutura do documento

O restante deste documento foi estruturado da seguinte forma:

O segundo capítulo apresenta o estudo do estado da arte efectuado bem como as suas conclusões. Esta análise incidiu em *hardware* de realidade virtual, *SDKs* e aplicações de realidade virtual existentes.

O terceiro capítulo apresenta o *software* a desenvolver, dividindo-o em várias componentes e detalhando cada uma delas.

O quarto capítulo apresenta o planeamento efectuado bem como o que foi feito e como este se relaciona com o planeado.

O capítulo cinco lista os requisitos recolhidos com o fim de guiar a aplicação. Estes requisitos são do tipo funcional e não-funcional. Também foi apresentada uma análise de riscos.

O capítulo seis apresenta as decisões de design de interface e de design experiência efetuadas. É apresentada a filosofia geral escolhida, bem como a interface de cada componente.

O sétimo capítulo apresenta a arquitetura do *software* desenvolvida. Esta arquitetura lista as várias decisões e desafios que foram fundamentais no desenvolvimento deste *software*.

O capítulo oito apresenta os requisitos não implementados bem como justifica o motivo da sua exclusão do âmbito da implementação.

O capítulo nove apresenta o tipo de testes realizados, os seus resultados, as conclusões e um plano de mitigação desenvolvido a partir das conclusões elucidadas. Foi ainda apresentado o estado da execução do plano

O capítulo dez apresenta o estado do projeto, assim como o seu futuro.

No capítulo onze são descritas as principais conclusões, é feita uma reflexão do projeto, bem como discuto o que foi aprendido durante o estágio.

Capítulo 2
Estado da Arte

Apesar da tecnologia de realidade virtual (3) já existir há algumas décadas, esta tecnologia nunca se encontrou acessível ao público geral. Neste momento há um ressurgimento do interesse por estas tecnologias bem como um aumento da acessibilidade destas ao público geral (4).

Este estágio lida com tecnologia emergente, por esse motivo foi necessária uma pesquisa para perceber o estado atual da tecnologia: o que existe no mercado, que plataformas existem e que *SDKs* se encontram disponíveis. Esta parte do estágio foi extremamente importante, já que ajudou a perceber as potencialidades e os limites destas tecnologias.

Inicialmente, a empresa já tinha definido que o desenvolvimento ocorreria para *iOS*. No entanto, foi me pedido para validar esta decisão e analisar alternativas.

No presente capítulo, será apresentada a análise de algumas tecnologias de *V.R.* que existem no mercado, tanto de *hardware* como *SDKs* e alguns produtos de software. Esta análise, incidirá na avaliação dos pontos fortes e fracos de cada produto/*SDK*/aplicação, com o objetivo de validar a decisão de *hardware* realizada inicialmente.

Este capítulo será divido em três grandes grupos: Análise de *hardware* de realidade virtual, Análise de *SDKs* e Análise de Aplicações.

2.1. Análise de Hardware de Realidade Virtual

Esta subsecção tem como objetivo analisar as ofertas existentes no campo da realidade virtual, à data da escrita deste texto. Serão apresentados e comparados alguns dos produtos mais relevantes no mercado atual.

2.1.1. Oculus Rift

Figura 1-Oculus Rift versão para o consumidor

O Oculus Rift (5) é um headset de realidade virtual desenvolvido pela Oculus V.R. Este produto foi lançado em março de 2016. Este headset tem uma resolução de 2160x1200

usando dois ecrãs OLED (1080x1200 por olho), headphones embutidos e um comando Xbox One, o acabamento do produto é bastante cuidado e bastante confortável.

O aparelho é capaz de realizar tracking aos movimentos da cabeça a 360º, usando sensores embutidos e uma câmara. O Oculus Rift usa drivers proprietárias, capazes de interagir com o *hardware* a um nível bastante baixo, fazendo *bypass* ao sistema operativo. Estas *drivers* permitem um tempo de resposta muito baixo, resultando numa experiência imersiva e livre de *motion-sickness*.

Para funcionar, este aparelho requer um computador com o sistema operativo Windows e com um *hardware* tão ou mais poderoso que o seguinte:

- NVIDIA GTX 970 / AMD 290 equivalent or greater
- Intel i5-4590 equivalent or greater
- 8GB+ RAM
- Compatible HDMI 1.3 video output
- 2x USB 3.0 ports
- Windows 7 SP1 or newer

2.1.2. HTC Vive

Figura 2-HTC Vive

O HTC Vive (6) é um aparelho de realidade virtual desenvolvido no âmbito de uma colaboração entre a Valve e a HTC. Este aparelho tem como ponto diferenciador oferecer uma experiencia "ao nível da sala" e permite que o utilizador se desloque dentro dos limites da sala. Isto é conseguido através de dois sensores que são colocados em pontos opostos da sala em vez de apenas um sensor embutido no aparelho. Este é suportado por dois ecrãs LCD, cada um com uma resolução 1080x1200 e uma taxa de *refresh* de 90 hz. Este dispositivo é bastante orientado a *gaming*, tendo o suporte da plataforma Steam.

2.1.3. Playstation V.R

Figura 3-Playstation V.R

O Playstation V.R. (7) é um aparelho de realidade virtual desenvolvido pela Sony. É um aparelho orientado a *gaming* desenvolvido para ser usado em conjunto com a Playstation 4. Este aparelho possui um ecrã 1920x1080 com tecnologia OLED. O ponto diferenciador deste aparelho é a sua taxa de *refresh*: 120hz, superior a qualquer competidor no momento. A latência é 18 ms. Este é um valor muito baixo, contribuindo para uma experiência fluida.

O lançamento deste produto é esperado em Setembro de 2016.

2.1.4. StarVR

Figura 4-Star V.R

O Star V.R (8) é um *headset* de realidade virtual produzido no âmbito de uma colaboração entre a Starbreeze e a InfinitEye's. Este aparelho possui dois ecrãs OLED cada um com uma resolução de 2560x1440, fazendo da resolução total uns impressionantes 5K panorâmicos, uma resolução muito maior que os seus concorrentes. O facto de este

aparelho possuir 2 ecrãs com uma disposição panorâmica leva a que o *field-of-view* seja enorme: 210ºx130º. O *tracking* da cabeça é realizado através do uso de giroscópios, acelerómetros e magnetómetros. Para funcionar este aparelho necessitará de um computador, no entanto as especificações ainda não foram reveladas.

2.1.5. Oculus Gear VR

Figura 5-Oculus Gear V.R

O Oculus Gear VR (9) é um aparelho de realidade virtual produzido no âmbito de uma colaboração entre a Samsung e a Oculus VR. Este aparelho funciona com certos telemóveis Samsung (Galaxy S7, Galaxy S7 EDGE, Galaxy S6, Galaxy S6 EDGE, Galaxy Note 4) para produzir uma experiência de realidade virtual num formato móvel. Este aparelho tem sensores de movimento integrados, um painel táctil para *input* e liga-se ao telemóvel através de uma ligação USB. Foi pensado para uso prolongado e por isso é feito de materiais confortáveis.

2.1.6. Google Cardboard

Figura 6-Google Cardboard

O Google Cardboard (10) é um aparelho de realidade virtual desenvolvido pela Google, que tem como objetivo proporcionar uma experiência de *V.R.* simples e barata. Este aparelho necessita de um telemóvel para conseguir proporcionar a experiência a que se propõe. O Google Cardboard, tal como o nome indica, é constituído por uma caixa de cartão que tem a responsabilidade de encapsular o telemóvel, 2 lentes bifocais de 40 mm e um botão. Na primeira geração o botão era um gatilho que funcionava em cooperação com o giroscópio do telemóvel, na segunda versão o gatilho foi substituído por um simples botão que toca na superfície do ecrã. A primeira versão suporta aparelhos até 5.2 polegadas, a segunda suporta aparelhos até 6 polegadas.

Apesar de este aparelho ser desenhado e as especificações serem mantidas pela Google, a sua produção não é realizada pelos mesmos. As especificações são públicas, qualquer pessoa pode fabricar o seu próprio Google Cardboard já que a sua montagem é bastante simples.

2.1.7. Comparativo

É possível ver que estamos perante dois diferentes grupos de hardware: os aparelhos *mobile* e os que requerem ligação a um computador.

Aparelhos não *mobile*: analisando os dispositivos inseridos nesta categoria, constatei que a qualidade gráfica apresentada é bastante semelhante para todos os produtos. No entanto, o StarVR possui uma resolução superior aos concorrentes e o PlayStation VR uma taxa de *refresh* superior. Em termos de imersão todos estes produtos apresentam experiências semelhantes, já que todos têm resolução e taxa de *refresh* suficientes para proporcionar uma experiência imersiva livre de enjoos. Em relação aos preços: o Oculus Rift custa 599 dólares e o HTC VIVE custa 900 dólares e o PlayStation VR 400 dólares. Em termos do preço do *hardware* da máquina, que irá suportar estes aparelhos, será de esperar no mínimo um computador na ordem dos 1000 euros, excepto o PlayStation VR, que será suportado por uma PS4 que tem um custo de 400 euros. Em relação ao número de aplicações todas as plataformas já têm bastantes demos técnicas e experiências.

Aparelhos *mobile*: em relação a esta categoria podemos afirmar que os Oculus Gear são superiores aos kits Google Carboard, em termos gráficos e imersão, no entanto o seu preço é muito superior: é possível construir um kit Google Carboard por menos de 5 euros, enquanto o preço do Oculus Gear é de 100 euros. Em relação ao aparelho que vai suportar o kit, o Oculus Gear obriga ao uso de modelos específicos bastante caros, no caso do Google CardBoard, qualquer telemóvel com giroscópio, ecrã de no mínimo de 4.7 polegadas e razoável potencia gráfica é capaz de proporcionar uma experiência aceitável. Em termos do número de apps o Google Cardboard ganha claramente já que é uma plataforma aberta e compatível com muitos mais dispositivos.

Aparelhos não *mobile* vs *mobile*: é possível notar que estamos perante duas abordagens e filosofias completamente distintas. As abordagens não-*mobile* são orientadas a *gaming* e a alta qualidade de experiência. Em contrapartida, as abordagens *mobile* primam pela acessibilidade de aquisição e pela portabilidade, sendo por isso mais aliciantes para a produção de conteúdos para as massas.

Aparelho	Qualidade Gráfica	Imersão	Preço	Preço de aparelho que este depende	Número de aplicações
Oculus Rift	Alta	Alta	Alta	Alta	Alta
HTC VIVE	Alta	Alta	N/A	Alta	Alta
PlayStation VR	Alta	Alta	N/A	Média	N/A
StarVR	Alta	Alta	N/A	Alta	N/A
Oculus Gear	Média	Alta	Média	Alta	Média
Google Cardboard	Média	Média	Baixa	Média	Alta

Tabela 1-Tabela comparativa

2.2. Análise de SDKs

Nesta secção irei analisar várias *SDKs* disponíveis para o desenvolvimento de aplicações de realidade virtual. Algumas delas funcionam com dispositivos específicos enquanto outras funcionam com várias famílias deles.

2.2.1. Oculus Mobile SDK

Figura 7- Oculus mobile SDK

Este *SDK* (11) foi desenvolvido pela Oculus com o objetivo de suportar o desenvolvimento para a plataforma Oculus Gear V.R. O Oculus Mobile *SDK* oferece suporte para correção da distorção das lentes, *head tracking*, rendering lado-a-lado, acesso ao *touchpad*, e tratamento do *input* do utilizador.

2.2.2. CardBoard SDK for Android

Figura 8- Android

O Cardboard SDK (12) é um *SDK* para Android desenvolvido pela Google para o desenvolvimento de aplicações de realidade virtual para Android. Este *SDK* tem como objetivo proporcionar uma experiência de realidade virtual natural e barata. Este *SDK* simplifica tarefas como:

- Correção da distorção das lentes.
- *Head tracking*.
- Calibração 3D.
- Rendering lado-a-lado.
- Stereo geometry configuration.
- Tratamento do *input do utilizador*.

Este *SDK* é open-source.

De notar que existe uma *port* não oficial para iOS, este port possui as mesmas funcionalidades da versão oficial de Android.

2.2.3. CardBoard SDK for Unity

Figura 9- Unity

Carboard SDK (13) for Unity é um *kit* de desenvolvimento, elaborado pela Google para o desenvolvimento de Aplicações *V.R.* para Android e iOS usando Unity. O Unity é conhecido por permitir desenvolver jogos multiplataforma facilmente. Este *SDK* usa as propriedades inerentes do Unity para permitir um desenvolvimento simples para Android e iOS.

Os *scripts* e *prefabs* deste SDK permitem tudo que é possível no *SDK* nativo da Google para Android e:

- Começar um novo projeto *V.R.* em Unity.
- Adaptar um projeto de Unity existente para *V.R.*
- Criar uma aplicação que mude entre modo de *V.R.* e normal facilmente.

2.2.4. Comparativo

Analisando estes *SDKs* é patente que o que oferecem é bastante semelhante entre eles, não existindo muitas conclusões para serem tiradas. De notar que apenas as *SDK* da Oculus não são *open-source* e apenas a SDK Cardboard para Unity tem suporte para *gazing detection*.

API	Open Source	Lens Distortion correction	Head tracking	3d Calibration	Side-by-side rendering	Stereo geometry configuration	User input event handling	Adapt previous projects to V.R	Gazing detection	Simply change from V.R. to non V.R. mode
Oculus Mobile SDK	X	√	√	√	√	√	√	√	X	√
Cardboard SDK for Android	√	√	√	√	√	√	√	√	X	√
Cardboard SDK for Unity	√	√	√	√	√	√	√	√	√	√

Tabela 2-Comparativo de SDKs

2.3. Análise de Aplicações

Esta secção irá analisar algumas aplicações de realidade virtual que existem no mercado. O objetivo desta análise é identificar bons aspectos em cada aplicação com o objetivo de encontrar ideias para o desenvolvimento desta prova de conceito. Esta análise é resultante da minha experiência pessoal, ao utilizar as aplicações. Tive em atenção a pontos como a qualidade gráfica, a tendência da aplicação a provocar enjoos e a qualidade geral como experiência de entretenimento. Não existiu a necessidade de usar mais pessoas além de mim para testar estas aplicações já que esta aplicação se trata de uma prova de conceito exploratória, não requerendo esse nível tão elevado de recursos.

2.3.1. Roller Coaster VR

Figura 10- Roller Coaster VR

Esta aplicação (14) proporciona a experiência de andar numa montanha russa num cenário tropical. A experiencia é totalmente passiva, sendo a única interação que o utilizador tem é a de puxar a alavanca para começar a viagem. Durante esta, o utilizador vê o mundo à sua volta com um ângulo de visão de 360º. A viagem é feita a um ritmo bastante rápido, dando ao utilizador uma boa sensação de velocidade. No teste efectuado por mim considerei os gráficos bastante bons e não me deparei com *motion-sickness*.

2.3.2. VR One Cinema

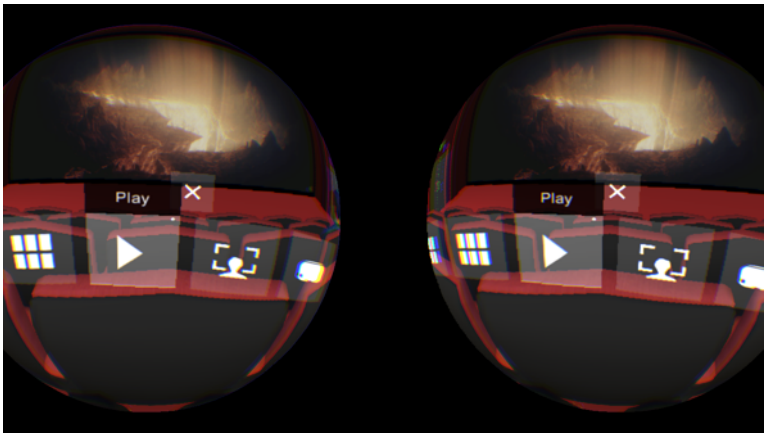

Figura 11- VR One Cinema

VR One Cinema (15) é uma aplicação especialmente desenvolvida para o Zeiss VR ONE kit, no entanto funciona com outros kits semelhantes ao Google Cardboard. Esta aplicação permite ao utilizador ver vídeos que estão gravados no telemóvel do utilizador enquanto está imerso numa sala de cinema virtual. O utilizador tem opções tais como mexer à volta da sala, pausar o vídeo ou popular a sala com espectadores. Considerei a qualidade gráfica razoável e não me deparei com *motion-sickness*.

2.3.3. A time in space VR

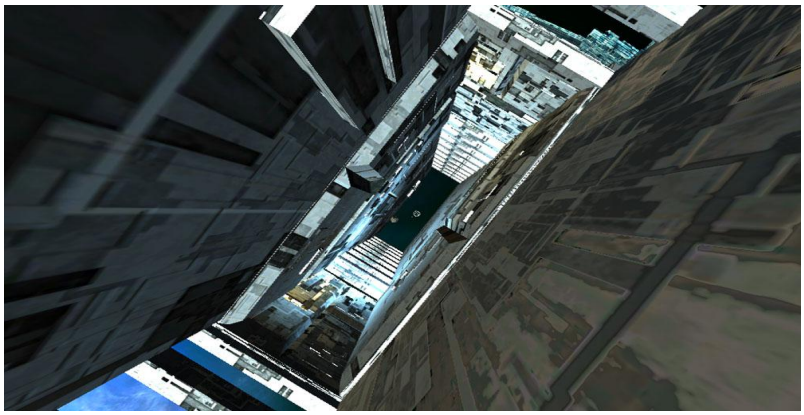

Figura 12- A time in space VR

Esta aplicação (16) simula uma viagem pelo espaço, viagem que é extremamente colorida, com cores bastante brilhantes e um ambiente complexo, resultando numa experiência psicadélica. Contudo, no meu teste deparei-me com vários momentos de baixos *F.P.S.* e senti bastante *motion-sickness*.

2.3.4. A Chair in a Room

Figura 13- A chair in a Room

Esta aplicação (17) tem como objetivo imergir o utilizador em um ambiente sinistro e claustrofóbico. O utilizador é fechado numa sala escura e misteriosa na qual está uma misteriosa cadeira e objetos estranhos. De acordo com a minha experiência com a aplicação posso afirmar que a qualidade gráfica me pareceu bastante elevada, das melhores que vi e não senti *motion-sickness*.

2.3.5. Sisters

Figura 14- Sisters

Esta aplicação (18) é uma experiencia de terror, o utilizador é imerso numa sala com dois bonecos, coisas bastantes assustadoras acontecem na sala enquanto o utilizador olha a volta. Esta experiência apresentou os melhores gráficos que encontrei na aplicação que testei e não apresentou *motion-sickness*.

2.3.6. High Speed Stunt Bike

Figura 15- High Speed Stunt Race

Esta aplicação (19) é um jogo em que o utilizador conduz uma mota a alta velocidade no meio do trânsito. O jogador tem de se desviar para não embater contra carros e tem de mandar os outros motoqueiros para fora da estrada. A principal caraterística diferenciadora desta aplicação é o seu controlo ser efectuado a partir do movimento da cabeça do utilizador, isto é, a mota vira para o lado que o utilizador virar a cabeça. A aplicação apresentou uma qualidade gráfica elevada e ausência de *motion-sickness*.

2.3.7. VR Speed Stunt Race

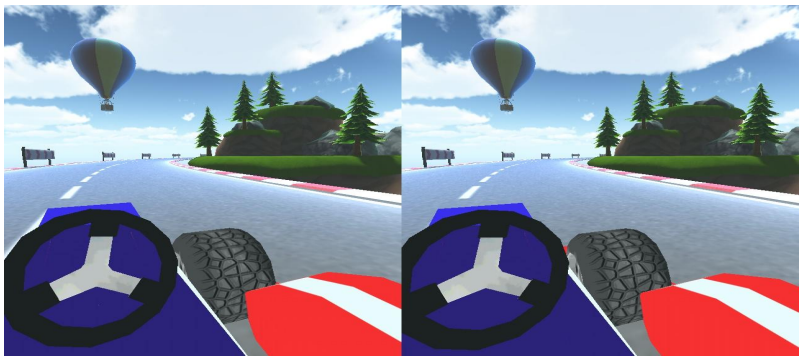

Figura 16- VR Speed Stunt Race

Esta aplicação (20) proporciona a experiência de estar dentro de um carro de *stunts* enquanto este percorre uma pista de corrida. Esta pista tem imensos saltos e acrobacias, fazendo recordar jogos como Mario Kart. É uma experiência passiva em que o utilizador apenas olha à volta, não interagindo com o mundo. O facto desta experiência me parecer bastante bem conseguida fez como que me sentisse extremamente imerso no ambiente.

2.3.8. Nighttime Terror

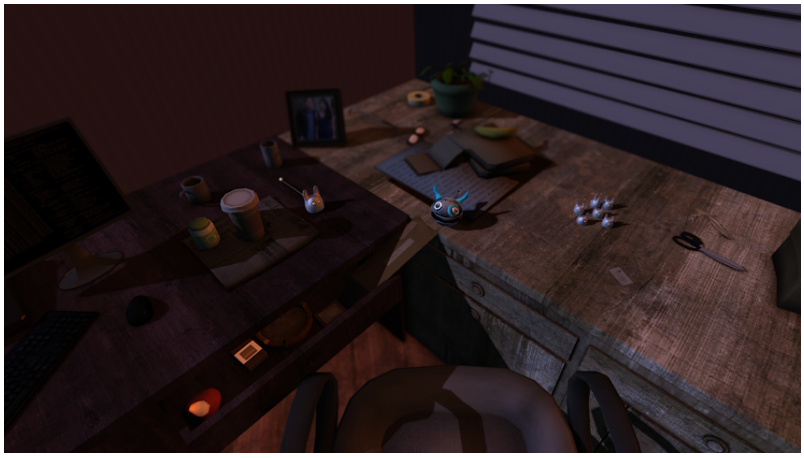

Figura 17- Nighttime Terror

Esta aplicação (21) é uma experiência de survival horror em que o objetivo é sobreviver o máximo de tempo possível ao ataque de uma horda de fantasmas. Ao contrário da maior parte dos jogos *V.R.* em que o jogador vê o mundo na primeira pessoa, o utilizador vê a ação num plano aéreo.

2.3.9. Hint Mint VR

Figura 18- Hint Mint VR

Esta aplicação (22) simula a visita a um museu, permitindo ao utilizador ver vários quadros em várias salas. A experiência é bastante interessante, no entanto achei que os gráficos apresentados são insuficientes para conseguir apreciar a arte.

2.3.10. Blockbuster VR

Figura 19- Blookbuster VR

Esta aplicação (23) proporciona a experiência de estar dentro de uma montanha russa num filme de ação, cheio de explosões e saltos. Esta aplicação apresenta um ambiente virtual bastante complexo e dinâmico. Achei que, graficamente, a aplicação é excelente e não me deparei com *motion-sickness*.

2.3.11. V.R Mac-Pan

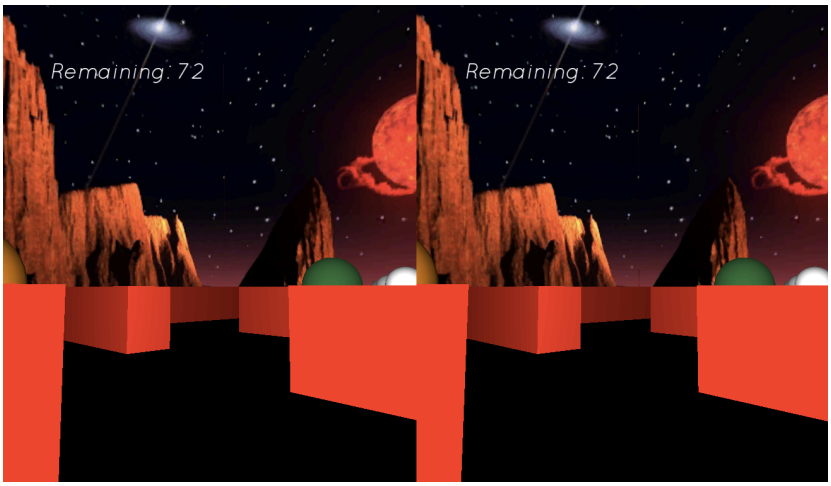

Figura 20- VR Mac-Pan

V.R Mac-Pan (24) é uma adaptação na primeira pessoa do popular jogo Pac-Man. O objetivo do jogo é similar ao do jogo original. O utilizador está imerso num labirinto, tendo como objetivo comer bolas e fugir dos fantasmas. A perspetiva na primeira pessoa e os controlos típicos da realidade virtual (*Auto-walk*, olhar para mover) tornam o clássico jogo numa experiência extremamente imersiva.

2.3.12. VRFly

Figura 21- VRFly

Esta aplicação (25) propõe ao utilizador controlar uma mosca livremente dentro de uma casa. Esta aplicação está extremamente bem conseguida, gráficos bons, ausência de *motion-sickness* e um ambiente grande para explorar, além disso é possível interagir com alguns objetos. Achei que graficamente esta aplicação é razoável e esta não me provocou *motion-sickness*.

2.3.13. VR Asteroids

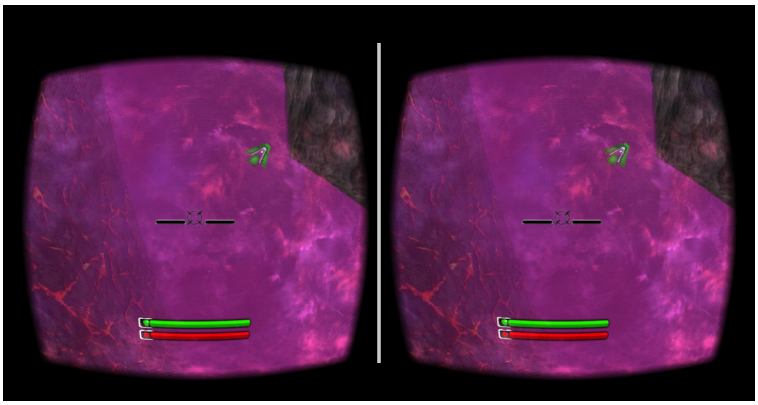

Figura 22- VR Asteroids

Este jogo (26) coloca o jogador dentro de uma nave espacial e dá ao utilizador a missão de destruir ou desviar dos asteroides que vêm na sua direção. A experiência que eu me deparei foi marcada por satisfação no campo gráfico e não me deparei com *motion-sickness*.

2.3. Conclusões do teste de aplicações

Após o teste de várias aplicações, foi possível perceber que a maior parte das aplicações não passam de demos técnicas que demonstram o potencial da tecnologia. Nenhuma aplicação que experimentei me faria adquirir qualquer um dos produtos, fica a faltar substância no entretenimento proporcionado. Ficou patente a falta de uma *killer-app*.

2.4. Conclusões do estado da Arte

Após a pesquisa que foi requerida para a escrita desta secção foi possível tirar várias conclusões. A realidade virtual é uma tecnologia extraordinária já que permite um nível de imersão nunca antes visto. As suas aplicações são extensas: entretenimento, ensino, medicina, pelo que há muito por onde explorar.

Tanto a nível de *hardware* tanto como de *software* é possível dizer que esta tecnologia está a crescer. Várias barreiras tecnológicas têm sido derrubadas nos últimos anos e as empresas têm-se esforçado para apresentar um ambiente de desenvolvimento confortável aos *developers*. Também já começam a ser lançados vários produtos de entretenimento de qualidade, marcando o começo da entrada desta tecnologia para o *mainstream*.

Cada vez mais a realidade virtual móvel (suportada por um telemóvel) mostra-se como um sector a apostar, já que a qualidade de experiência é razoável e promete melhorar rapidamente já que a evolução do *hardware* móvel continua a um ritmo estonteante. Esta forma de realidade virtual apesar de não ser tão graficamente impressionante, tem como grande vantagem a ausência de fios, permitindo um uso mais cómodo e casual, tendo por isso grande potencial para atrair as massas.

Tendo em conta tudo o que foi mencionado em cima, acho que este é o momento ideal para investir nesta tecnologia. Uma vez que prevejo que esta irá afetar vários mercados tecnológicos de forma disruptiva, é prudente ter conhecimento na área já que poderá levar a bastantes ganhos.

Capítulo 3
Trabalho a realizar e abordagem

O objetivo deste estágio é o desenvolvimento de uma prova de conceito que consiste numa aplicação de realidade virtual para dispositivos iOS compatíveis com aparelhos de realidade virtual do tipo Google CardBoard (iPhone 6, IPhone 6S). É do interesse da empresa ganhar conhecimento na área da realidade virtual, mais concretamente sobre o uso desta tecnologia na plataforma iOS. Este estágio visa descobrir as potencialidades desta tecnologia nesta plataforma.

Inicialmente foi decidido pela empresa que o que foi proposto no estágio não era suficientemente inovador, não ajudando a alcançar os objetivos definidos internamente.

Durante o primeiro semestre foi me proposto encontrar formas inovadoras de explorar esta tecnologia na plataforma escolhida. Após várias reuniões e reformulações do âmbito do estágio foi decidido que a aplicação a desenvolver iria ser constituída por uma aplicação multimédia com quatro tipos de conteúdos distintos:

- Galeria de fotos.
- Reprodução de música.
- Reprodução de vídeos.
- Reprodução de PowerPoint.

3.1. Galeria de Fotos

A aplicação suporta uma galeria de fotos em realidade virtual, em que o utilizador poderá visualizar fotos guardadas no telemóvel. O objetivo da exploração deste conteúdo é proporcionar ao utilizador uma forma mais imersiva de consumir a sua galeria de fotos.

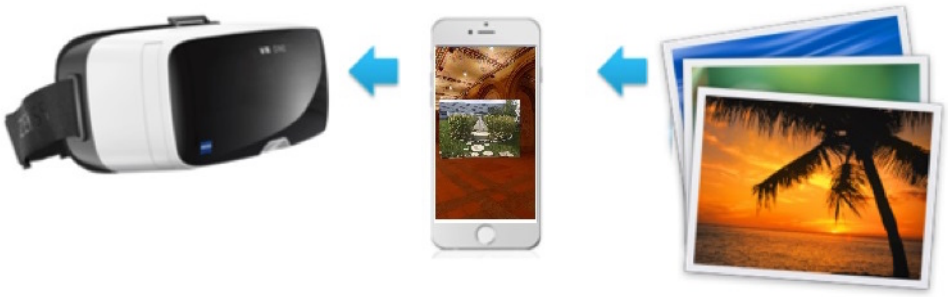

Figura 23- Galeria fotográfica em V.R.

3.2. Reprodução de música

A exploração desta componente multimédia pretende proporcionar aos fãs de música uma nova forma de a apreciar. Esta componente da aplicação é direcionada a pessoas que procuram uma maneira diferente de consumir a sua música. Com esta aplicação, o utilizador poderá consumir e controlar a sua música num ambiente virtual.

Figura 24- Reprodução de música em V.R.

3.3. Reprodução de vídeos

Esta aplicação suporta reprodução de vídeos num ambiente de realidade virtual, mais concretamente de vídeos gerados por action cameras (vídeos com elevado *field-of-view*).

Uma action camera carateriza-se por conseguir capturar imagens e vídeos com um *field-of-view* superior ao capturado pela maior parte das câmaras. Estas câmaras divergem da maioria no formato das lentes: estas são circulares, provocando distorção que é responsável pela captura de imagem com um *field-of-view* elevado.

A proposta desta componente é, usando a tecnologia de *V.R*, remover a distorção, mantendo o *field-of-view* elevado. Isto irá permitir uma experiência de visualização mais imersiva.

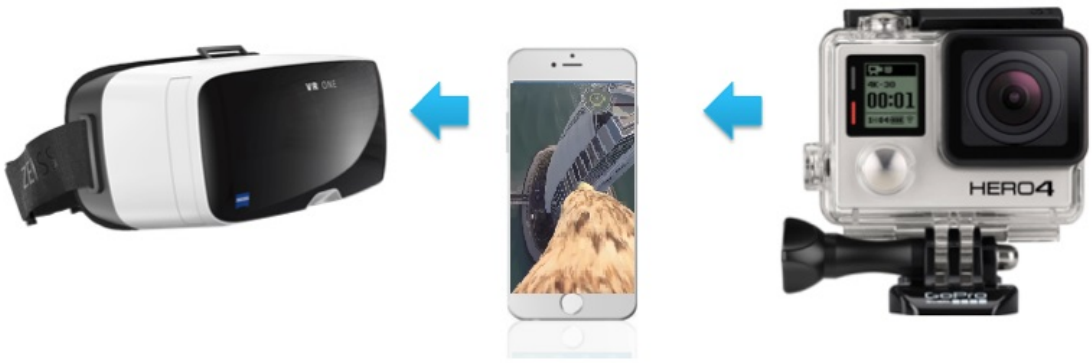

Figura 25- Reprodução de vídeos originados de uma Action Camera em V.R.

3.4. Reprodução de PowerPoint

As apresentações tendem a ser aborrecidas e facilmente a audiência perde interesse, já que quem está a assistir à mesma não se sente envolvido, a não ser que a apresentação seja de excelente qualidade. É objetivo deste estágio reproduzir apresentações num dispositivo do tipo *Cardboard*, tornando uma apresentação banal numa experiência muito mais imersiva e interessante, ajudando a transmitir a mensagem que a apresentação pretende. Esta aplicação permitirá reproduzir um qualquer ficheiro .pptx em realidade virtual. Neste formato, o utilizador terá o seu campo de visão totalmente obstruído pela apresentação e será possível apresentar conteúdo num ambiente muito mais imersivo.

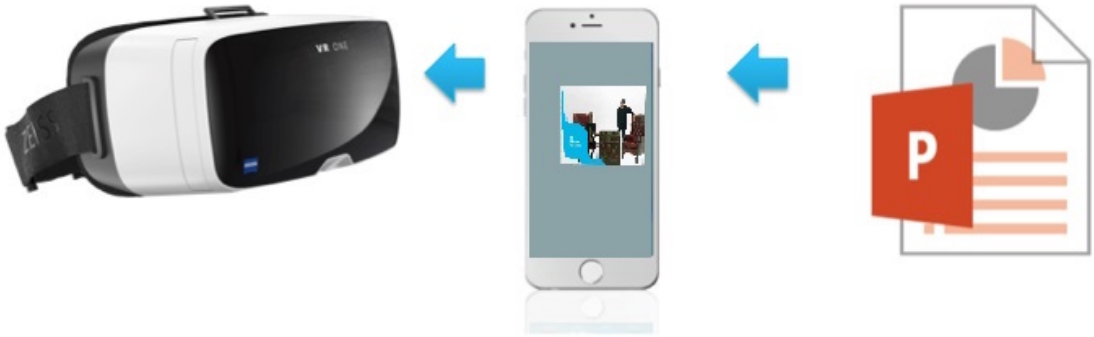

Figura 26- Reprodução de PowerPoint em V.R.

3.5 Metodologia de trabalho

Para desenvolver este trabalho foi usada uma metodologia ágil (27). O processo envolveu reuniões diárias informais com o orientador da empresa. Nestas reuniões era revisto o meu progresso no dia anterior e definidos objetivos para o dia atual.

No primeiro semestre não foi realizada escrita de código, foi-me pedido para estudar as várias alternativas tecnológicas, bem como formular a prova de conceito que iria ser produzida no segundo semestre, já que, como referido anteriormente, é da opinião da empresa que a proposta inicial é pouco ambiciosa e não apresenta inovação suficiente.

No segundo semestre deu-se a implementação da prova de conceito. É de notar que houve uma abordagem de constante re-avaliação dos requisitos e abordagem seguida, tendo alguns requisitos acabado por ser ignorados ou reformulados.

Capítulo 4
Planeamento

Nesta secção vai ser apresentado o planeamento (28) do projeto bem como uma visão detalhada do que foi feito em cada semestre.

4.1 1 º Semestre

Este estágio é de natureza bastante aberta e exploratória, logo foi necessário bastante estudo a nível do estado da arte, bem como das tecnologias. Este trabalho foi essencial para definir o foco da aplicação, bem como para famialirizar o estagiário com as tecnologias, preparando-me para efetuar decisões correctas, durante o desenvolvimento.

No primeiro semestre foi pedida a redefinição da prova de conceito a ser construída, para isso produzi vários documentos que foram sujeitos a aprovação com o fim de procurar um conceito original e que permitisse aumentar o conhecimento da empresa nesta área. Foi também objetivo deste semestre validar a escolha da plataforma de realidade virtual.

Este projeto começou com uma fase inicial de pesquisa sobre o estado da arte em realidade virtual, iniciando-se por um estudo sobre os dispositivos de realidade virtual, disponíveis no mercado. De seguida, foram estudados os vários *SDKs* associados a cada dispositivo, permitindo perceber as capacidades que estes oferecem. Foram testadas várias aplicações com o fim de perceber o estado do mercado, limitações tecnológicas, bem como tirar ideias para definir melhor o foco da prova de conceito a realizar. Por fim, para ter melhor noção do que podia ou não usar no projeto, foi elaborado um documento sobre licenças de *software*. Na fase seguinte, escrevi três documentos a detalhar ideias para três provas de conceito, no entanto, ficou definido que o que iria ser desenvolvido seria constituído por uma aplicação unificada com características das várias provas de conceito documentadas. De seguida, escrevi um documento a descrever a arquitetura do motor de *V.R.* que tem a responsabilidade de suportar a aplicação desenvolvida. Por fim, foi realizado o *porting* de uma aplicação de *V.R.* Android para iOS, com o fim de adquirir conhecimento na área.

4.2. 2º semestre

O segundo semestre começou com o desenvolvimento do motor de realidade virtual. Para isso, estudei como funcionava o C++ e o Objective-C (apesar de já ter lidado um pouco no semestre anterior foi necessário aprofundar o estudo), bem como aprendi a interligar o uso das duas linguagens. Para me preparar para lidar com o requisito de criação de uma aplicação facilmente portável, estudei como construir código independe de plataforma em C++ e como criar uma interface para comunicar com uma plataforma abstrata.

No desenvolvimento do *engine* comecei por tratar de desenhar texturas no ecrã em *V.R.*, de seguida desenvolvi classes que representassem objetos gráficos e encapsulassem propriedades que permitissem fácil manipulação. Por fim, tratei do sistema de carregamento de *Assets* bem como de eventos.

A primeira componente desenvolvida após o desenvolvimento do *engine* foi a Galeria de fotos já que esta era a mais simples de implementar. Para ter conhecimento suficiente para implementar esta componente foi necessário que se fizesse uma pesquisa por *APIs* que permitissem acesso ao conteúdo fotográfico contido no telemóvel. De seguida, desenhei um sistema que permitisse interligar essa *API* com o processo de carregamento de fotos. Para assegurar uma interface agradável ao utilizador desenvolvi várias animações, usando as capacidades do motor bem como de *shaders* usando a linguagem GLSL.

A seguinte componente a ser implementada foi o reprodutor de música. Para implementar esta componente estudei formas de reproduzir som e formas de aceder à biblioteca musical de um dispositivo iOS. Apesar de ter conseguido reproduzir sons, não consegui aceder à biblioteca. Para contornar o problema deste primeiro método não conseguir reproduzir a biblioteca musical do utilizador, mudei a abordagem a ser seguida. Encontrei uma *API* capaz de controlar o *player* nativo do iOS e criei uma interface gráfica para o utilizador. Desta forma, o utilizador consegue controlar a sua música em *V.R.* Foi também apresentada a capa do álbum da música a ser reproduzida atualmente.

A terceira componente a ser implementada foi a componente de vídeo. Procurei *APIs* que permitissem reproduzir vídeo num contexto OpenGL em iOS, não tendo sucesso. Para contrariar este problema desenvolvi um *player* de vídeo. Depois de ter conseguido implementar a reprodução de vídeo, procurei uma forma de tornar a reprodução de vídeos provenientes de uma Action Camera mais imersiva, a solução passou pela reprodução da textura num plano curvo.

A última componente a ser desenvolvida foi a componente de reprodução de PowerPoint. Esta componente foi sem dúvida a mais desafiante e trabalhosa. O primeiro passo foi um estudo do formato OOXML bem como uma análise de como seria possível explorá-lo o interesse deste *software*. Estudado o formato parti para o estudo de tecnologias que facilitassem o *parsing* destes documentos.

As tecnologias estudadas para o *parsing* de documentos PowerPoint foram: tecnologias de desencriptação, vários tipos de *parsing* de XML, XPATH. Estas tecnologias foram essenciais para extração de dados de um ficheiro PowerPoint, com estas foi construída uma árvore que representa um documento PowerPoint. Por fim, foi criado um processo que convertia os dados da árvore anteriormente mencionada em dados que podiam ser usados para o desenho de slides. Para tornar esta componente mais *user-friendly* desenhei uma interface, que permite ao utilizador controlar o fluxo da apresentação. Para finalizar, foi adicionada uma música em fundo para tornar a experiência mais imersiva.

Após implementar as componentes estive ocupado com o tratamento da UI da aplicação, adicionei animações bem como alterações nas texturas e *shaders* utilizados.

Depois do tratamento da UI deu-se a um processo de realização testes e correção de erros.

4.3. Desvios no calendário na implementação

O **Anexo D** é constituído por um diagrama de Gant que ilustra o trabalho realizado bem como o planeamento para o segundo semestre. O **Anexo E** apresenta o diagrama de Gant contendo a calendarização final após o fim do projeto.

Capítulo 5
Análise de Requisitos

Esta secção irá analisar os requisitos de cada uma das provas de conceito a ser desenvolvidas assim como do motor de realidade virtual. Para este fim, serão usadas User Stories (29) para ilustrar os requisitos funcionais, serão também descritos os requisitos não-funcionais (30) encontrados, bem como riscos (31) identificados e as suas respetivas estratégias de mitigação (32).

5.1. Requisitos do motor de realidade virtual

User Stories

ID	História	Sistemas afetados	Prioridade
USE001	Como utilizador eu quero ver vídeos em *V.R.* de forma a sentir me imerso.	Motor, Aplicação suportada	3
USE002	Como utilizador quero visualizar fotos em *V.R.* para me sentir imerso nas mesmas.	Motor, Aplicação suportada	3
USE003	Como utilizador quero ouvir som num ambiente em *V.R.*	Motor, Aplicação suportada	3
USE004	Como utilizador quero navegar menus em *V.R.* de forma imersiva.	Motor, Aplicação suportada	3
USE005	Como utilizador eu quero entrar e sair do modo *V.R.* apenas tirando e colocando o device no kit *V.R.*	Motor, Aplicação suportada	1
USE006	Como utilizador eu quero enviar *inputs* ao programa apenas olhando objetos.	Motor, Aplicação suportada	3
USE007	Como utilizador eu quero enviar *inputs* ao programa usando interface táctil.	Motor, Aplicação suportada	2
USE008	Como utilizador eu quero enviar *inputs* para o programa através do botão de *trigger*.	Motor, Aplicação suportada	2

Tabela 3- Engine User Stories

Requisitos não-funcionais

ID	Requerimentos não-funcionais	Derivado de	Métricas de sucesso
NFE001	A experiência deve ser imersiva.	Todos	Todos os testers devem reportar bom nível de imersão
NFE002	O *delay* entre as ações do utilizador e o resultado correspondente não deve ser perceptível.	Todos	Não deve ser possível detetar *delay*.
NFE003	Vídeo em *V.R.* deve ser reproduzido com *framerate* suficiente para evitar *motion-sickness*.	USE001	Testers não devem reportar *motion-sickness*.

NFE004	As fotos apresentadas não devem aparecer distorcidas.	USE002	Os testers não devem reportar distorção nas fotos apresentadas.
NFE005	O som deve ser apresentado da mesma forma em *V.R.* e no modo não *V.R.*	USE003	Testers não devem reportar falta de qualidade de som no modo *V.R.*
NFE006	Navegação nos menus deve parecer natural.	USE004	Os testers não devem reportar nenhuma dificuldade na navegação pelo menu.
NFE007	Navegação pelo Menu deve parecer semelhante ao modo não *V.R.*, apenas mais imersiva.	USE004	Os testers não devem reportar nenhuma dificuldade na navegação pelo menu em nenhum dos modos.
NFE008	O programa não deve parar quando muda de modo *V.R.* para não *V.R.* e o oposto.	USE005	Testers não devem reportar "*lag*"
NFE009	Seleção através de olhar deve ser precisa	USE006	Os testers não devem reportar dificuldades a selecionar objetos
NFE010	O *Heads Up Display* que suporta a aplicação não deve ser intrusivo.	USE006	Os testers não devem reportar insatisfação com o *Heads Up Display*.
NFE011	O evento de *trigger* deve ser reconhecido em contexto	USE008	Os tester devem reportar que o uso do *trigger* lhes pareceu natural.

Tabela 4- Requisistos não-funcionais do motor

Riscos

ID	Riscos	Histórias associadas	Estratégia de mitigação
RISKE001	A resolução do vídeo/telemóvel pode ser insuficiente para conseguir uma qualidade de experiência insatisfatória.	USE001	Recomendar uma resolução mínima para o telemóvel.
RISKE002	Poderão aparecer *tradeoffs* entre qualidade de experiência e uso de bateria.	Todos	Optimizar para performance
RISKE003	Apresentar texto em *V.R.* é difícil.	USE001, USE004	Implementar todas as boas práticas recomendadas a *V.R.* developers

Tabela 5- Riscos do Motor

5.2. Requisitos específicos da componente Galeria de fotos

User Stories

ID	Categoria	História	Sistemas afetados	Prioridade (min=1, max=3)
USF001	Utilizador	Como utilizador quero visualizar a minha galeria fotográfica guardada no meu device em realidade virtual.	Galeria de fotos, Telemóvel	3
USF002	Utilizador	Quero um sistema que me permita navegar pela minha biblioteca e selecionar as fotos que pretendo visualizar.	Galeria de fotos, Telemóvel	3
USF003	Utilizador	Como utilizador quero visualizar a minha biblioteca fotográfica guardada num serviço de *cloud*.	Galeria de fotos, Telemóvel	2
USF004	Utilizador	Como utilizador quero apagar o meu conteúdo utilizando a mesma aplicação que usei para o consumir.	Galeria de fotos, Telemóvel	1

Tabela 6-User Stories da componente Galeria de fotos

Requisitos não-funcionais

ID	Requisitos não-funcionais	Derivado de	Métricas de sucesso
NFF001	O utilizador deve poder visualizar imagens num conjunto de imagens ou individualmente.	USF001, USF002, USF003	Estes dois modos devem estar presentes.
NFF002	As imagens devem possuir detalhe suficiente para satisfazer o utilizador	USF001, USF002, USF003	Os utilizadores testados não devem reportar falta de detalhe nas fotografias.
NFF003	A interface deve ser simples e imersiva.	USF002	Os utilizadores devem reportar satisfação com a interface.

Tabela 7- Requisitos não-funcionais da componente Galeria de fotos

Riscos

ID	Riscos	Histórias Associadas	Estratégia de mitigação
RISKF001	O processo de conversão ou a performance do telemóvel podem não ser suficientes para uma experiência com um *frame-rate* suficiente para a experiência ser divertida.	USF001, USF002, USF003	Minimizar a qualidade da resolução das fotos ou o número de fotos no ecrã.
RISKF002	A resolução do vídeo ou telemóvel podem não ser suficientes.	USF00,USF002,USF003	Recomendar ao utilizador modelos que sabemos que a experiência seja satisfatória.
RISKF003	Restrições de acesso ao sistema de ficheiros podem causar problemas.	USF001, USF004	Estudar as capacidades de acesso ao *file system* do *SDK*.

Tabela 8- Riscos da componente Galeria de fotos

5.3. Requisitos específicos da Componente de Reprodução de Música

User Stories

ID	Categoria	História	Sistemas afetados	Prioridade (min=1, max=3)
USM001	Utilizador	Como utilizador quero ouvir a minha música guardada no telemóvel enquanto estou inserido num ambiente de realidade virtual.	Componente de reprodução de música, Telemóvel	3
USM002	Utilizador	Como utilizador quero selecionar a minha música com uma interface em realidade virtual.	Componente de reprodução de música, Telemóvel	3
USM003	Utilizador	Como utilizador quero ouvir a minha música em *streaming* enquanto estou inserido num ambiente de realidade virtual.	Componente de reprodução de música, Telemóvel	2
USM004	Utilizador	Como utilizado quero controlar o fluxo da reprodução da música atual	Componente de reprodução de música, Telemóvel	1

Tabela 9- Requisitos funcionais da Componente de reprodução de Música

Requisitos não-funcionais

ID	Requisitos não-funcionais	Derivado de	Métricas de sucesso
NFM001	O utilizador deve puder selecionar a próxima música enquanto ouve a atual.	Todos	Estes dois modos devem estar presentes.
NFM002	A animação deverá reagir consoante a música a ser tocada	Todos	Os utilizadores testados não devem reportar falta de detalhe nas fotografias.
NFM003	A capa do álbum deve ser apresentada se possível.	Todos	Os utilizadores devem reportar satisfação com a interface.

Tabela 10- Requisitos não-funcionais da Componente de reprodução de Música

Riscos

ID	Riscos	Histórias Associadas	Estratégia de mitigação
RISKM001	Restrições de acesso ao sistema de ficheiros podem causar problemas.	US001, US004	Estudar as capacidades de acesso ao *file system* do *SDK* antes de começar a implementação.

Tabela 11-Riscos Componente de reprodução de Música

5.4. Requisitos específicos da componente de vídeo

User Stories

ID	Categoria	História	Sistemas afetados	Prioridade (min=1, max=3)
USP001	Utilizador	Como utilizador quero visualizar os meus vídeos provenientes de uma *action camera* com um grande *field-of-view* para me sentir mais imerso no conteúdo.	Componente de vídeo, Telemóvel	3
USP002	Utilizador	Como utilizador quero guardar o meu conteúdo proveniente de uma *action camera* no meu telemóvel de forma a poder vê-lo em qualquer lugar.	Componente de vídeo, Telemóvel	1
USP003	Utilizador	Como utilizador quero assistir a conteúdo proveniente de uma action camera no meu device de *V.R.* através de *streaming*.	Componente de vídeo, Telemóvel	2
USP004	Utilizador	Como utilizador quero apagar o meu conteúdo utilizando a mesma aplicação que usei para o consumir.	Componente de vídeo, Telemóvel	1

USP005	Utilizador	Como utilizador quero controlar o fluxo do vídeo.	Componente de vídeo,	2
USP006	Utilizador	Como utilizador quero selecionar o conteúdo que quero reproduzir em *V.R.* de forma a sentir-me mais imerso.	Componente de vídeo, Telemóvel	3
USP007	Utilizador	Como utilizador quero que a aplicação se adapte automaticamente ao *V.R.* device que estou a utilizar para conseguir a máxima qualidade de experiência.	Componente de vídeo, Telemóvel, *Cardboard kit*	1

Tabela 12- VR User Stories da componente de Vídeo

Requisitos não-funcionais

ID	Requisitos não-funcionais	Derivado de	Métricas de sucesso
NFP001	O vídeo deve ser desenhado numa superfície curva de forma ao utilizador puder olhar 170º a sua frente e ser deparado com parte do vídeo.	USP001	Os utilizadores sujeitos a teste devem reportar boa imersão.
NFP002	O vídeo e o som devem ser sincronizados	USP001	Ambiente muda em relação ao slide atual.

Tabela 13- Requisitos não-funcionais da componente de Vídeo

Riscos

ID	Riscos	Histórias Associadas	Estratégia de mitigação
RISKP001	As soluções que já existem podem ter restrições de licenças que podem ser incompatíveis com a proposta deste *software* ou simplesmente não terem a performance esperada.	Todas as funcionais.	Deve ser realizado um estudo sobre que tipo de licenças são possíveis ser usadas e que tipo de licenças as *APIs* existentes oferecem.
RISKP002	A performance do telemóvel pode não ser suficiente para uma experiência com um *frame-rate* suficiente para a experiência ser divertida.	USP001	Procurar *APIs* e métodos que permitam reproduzir vídeo da forma mais eficiente computacionalmente possível.

Tabela 14- Riscos da componente de Vídeo

5.5. Requisitos específicos da componente de reprodução PowerPoint

Requisitos funcionais

ID	Categoria	Historia	Sistemas afetados	Prioridade (min=1, max=3)
USP001	Utilizador	Como utilizador, quero ver uma apresentação em realidade virtual.	PowerPoint *V.R.*, Telemóvel, Host's computer	3
USP002	Utilizador	Como utilizador, eu quero ver conteúdo espalhado à sua volta de forma a sentir-me imerso.	PowerPoint *V.R.*, Telemóvel, Host's computer	2
USP003	Apresentador/Utilizador	Como criador da apresentação, quero controlar o fluxo do que aparece de forma a passar a mensagem / Como utilizador quero controlar o fluxo do que aparece para receber a mensagem.	PowerPoint *V.R.*, Telemóvel, Host's computer	2
USP004	Utilizador	Como utilizador quero ter a habilidade de navegar para o slide anterior de forma a poder corrigir um input errado ou perceber algo melhor.	PowerPoint *V.R.*, Telemóvel, Host's computer	2
USP005	Utilizador	Como utilizador quero puder visualizar conteúdo multimédia como vídeos na apresentação.	PowerPoint *V.R.*, Telemóvel, Host's computer	2

Tabela 15- Requisitos funcionais da reprodução PowerPoint

Requisitos não-funcionais

ID	Requisitos não-funcionais	Derivado de	Métricas de sucesso
NFP001	A experiência deve ser imersiva.	USP001	Os utilizadores sujeitos a teste devem reportar boa imersão.
NFP002	O ambiente envolvente deve reagir ao slide atual.	USP001	Ambiente muda em relação ao slide atual.
NFP003	O texto deve ser legível.	USP001	Os sujeitos de teste não devem reportar dificuldades com os testes.

Tabela 16- Requisitos não-funcionais da reprodução PowerPoint

Riscos

ID	Riscos	Historias associadas	Estratégias de mitigação
RISKP001	Informação extraída não permite criar efeitos interessantes.	NFP002	Usar efeitos simples
RISKP002	Apresentar texto em *V.R.* é conhecido por ser complicado.	NFP003	Utilizar imagens com texto em vez de o renderizar.

Tabela 17- Riscos da reprodução PowerPoint

Capítulo 6
User Interface e User Experience

Esta secção visa apresentar as decisões efetuadas no âmbito da UI (*user interface*) e da UX (*user experience*).

A UI (33) e a UX (34) são duas disciplinas de design que muitas vezes se confundem (35). A UX lida com a qualidade de experiência percepcionada ao utilizador resultante da interação com a aplicação como um todo. A UI preocupa-se com o *layout* da aplicação, bem como com as texturas e botões utilizados.

Como fui o único responsável por todo o desenvolvimento do projeto, foi da minha responsabilidade criar/recolher os *assets* que foram usados na aplicação, bem como definir o *layout* das várias componentes do projeto bem como do projeto como um todo (36).

6.1. Assets

Para garantir a qualidade de experiência do utilizador foram utilizados vários tipos de *assets*. Nesta subseção são apresentados os vários tipos de *assets* utilizados:

- Texturas com fim de seleção: estas texturas representam botões. Normalmente estas estão acompanhadas por *shaders* animados ou animações.
- Texturas de ambiente: estas texturas representam vários objetos não interativos. Estas texturas têm como fim decorar o ambiente em que o utilizador está imerso.
- Texturas de fundo: estas texturas são imagens panorâmicas 360º. Estas texturas têm como fim criar um ambiente envolvente imersivo, transmitindo a sensação do utilizador estar na localização representada pela textura.
- Efeitos sonoros: estes *assets* são constituídos por pequenos clipes de som. A sua finalidade é fornecer ao utilizador um *output* sonoro que confirme a realização de um *input* pelo mesmo.

6.2. User Experience

Filosofia de design

No inicio do projeto foi pedido ao estagiário para desenhar um *look and feel* comum para a aplicação. A solução para este desafio passou pelo desenho de uma abordagem minimalista. Esta abordagem apresenta as seguintes vantagens:

- Necessidade de menos *assets*.
- Este tipo de abordagens são cada vez mais usadas, desta forma a aplicação está de acordo com a corrente de design atual.
- Ambientes mais vazios e simples, tornando a interação por parte do utilizador mais simples.
- O fato desta abordagem usar poucos assets e se basear em ambientes simples permite poupar recursos computacionais e consequentemente tornar a experiência mais fluida.

Organização da Aplicação

A aplicação é dividida em várias componentes, cada componente é responsável por fornecer um conjunto de serviços englobados numa só temática. As componentes presentes são:

- Componente de galeria de fotos.
- Componente de reprodução de música.
- Componente de reprodução de vídeos.
- Componente de reprodução de PowerPoint.

O programa segue uma estrutura de cenas, tendo cada componente uma ou mais cenas. Quando o utilizador entra no programa depara-se com uma interface que permite selecionar a componente a aceder (Home Scene).

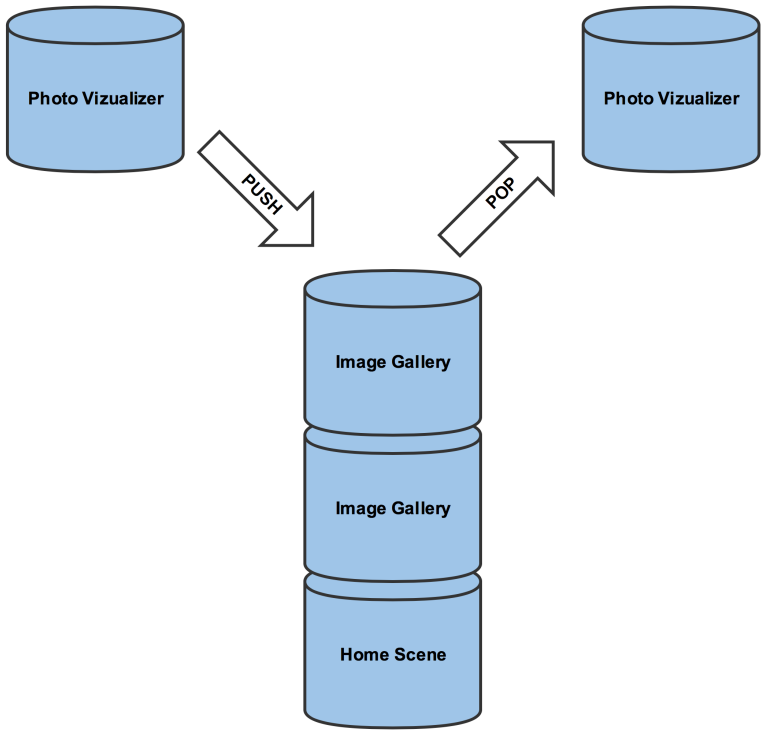

Figura 27- Pilha de cenas

O sistema de cenas baseia-se numa pilha, quando o utilizador faz um *input* que leva a entrada em uma cena, esta é adicionada à pilha. Quando o utilizador sai da cena atual esta é retirada da mesma e é carregada a cena que lhe antecede na pilha.

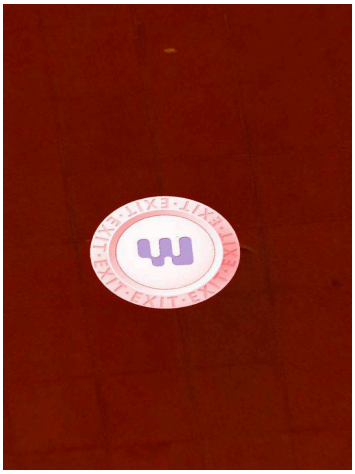

Figura 28- Ícone de saída da cena atual

Métodos de input

A aplicação usa o giroscópio do telemóvel para mover o conteúdo mostrado ao utilizador. Desta forma o utilizador consegue visualizar um ambiente virtual em 360º.O utilizador consegue alterar a porção deste ambiente que está a visualizar movendo a cabeça.

Para a interação com o ambiente virtual optou-se pela não utilização de dispositivos extra. Tendo este requisito em conta tomou-se a decisão de utilizar o olhar como método de seleção, isto é, quando o utilizador olha para um objeto interativo do ambiente virtual durante um período de tempo definido, este reage como um botão causando uma resposta por parte do programa. Para facilitar este processo, é sempre desenhado um ponteiro verde no ecrã, representando para onde o utilizador está a olhar. Para mostrar ao utilizador que está a interagir com um objecto, este é animado durante o intervalo do tempo que o utilizador está a olhar para ele.

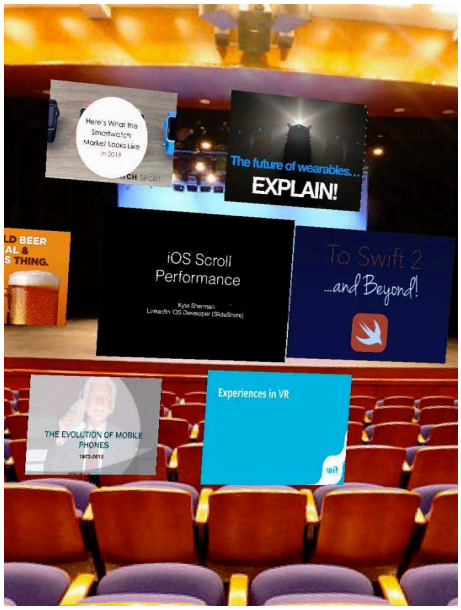

Figura 29- Exemplo de interação com o ambiente

Mudança de modo de visualização:

Esta aplicação, além de contar com um modo de realidade virtual, conta com um modo em que o conteúdo é apresentado de forma "tradicional" ao utilizador.

A mudança entre estes dois modos é realizada com o utilizador virando o dispositivo, de forma a alterar a disposição do mesmo. A presença do dispositivo no modo *portrait* desliga a apresentação de conteúdo de realidade virtual, pelo contrário a presença do dispositivo no modo *landscape* ativa o modo de realidade virtual.

6.3. UI e UX de cada componente

Nesta subseção será exposto o *design* de cada componente do *software* desenvolvido (37).

Photo Gallery

Esta componente é responsável por apresentar uma galeria fotográfica com o conteúdo guardado no dispositivo.

A interface com o utilizador é constituída por várias cenas, estas cenas podem ser de dois tipos:

- **Image Gallery**: este tipo de cena mostra várias fotos do device em 360º graus à volta do user, permitindo navegar para outras cenas que mostrem mais fotos. É possível também selecionar uma foto e observa-la na cena Photo Visualizer.

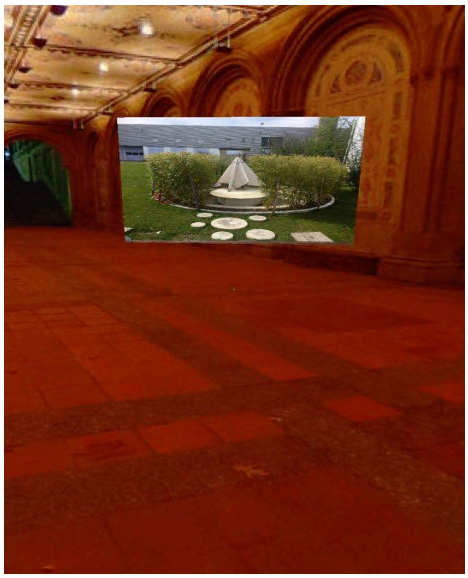

Figura 30- Image Gallery

- **Photo Visualizer**: esta cena é responsável por apresentar uma foto anteriormente selecionada na cena Image Gallery. É também possível navegar entre fotos nesta cena utilizando botões localizados ao lado de cada foto.

Figura 31- Photo Vizualiser

Music Player

Esta componente é responsável por reproduzir as músicas do utilizador num ambiente de realidade virtual.

Esta componente apenas possui uma cena, esta apresenta as seguintes funcionalidades:

- Reprodução de música.
- Apresentação da capa do álbum da música atual.
- Controlo do fluxo da reprodução de música.
- Apresenta o progresso da música atual.

Figura 32- Music Player

Video Gallery

Esta componente é responsável pela reprodução de vídeos. Apenas se considerou a reprodução de vídeos com elevado *field-of-view* (170º) provenientes de action-cameras. A reprodução de outro tipo de vídeos foi ignorada já que o processo para reproduzir vídeos de action camera é um *"super-set"* da reprodução de vídeos "normais". Os vídeos reproduzidos são reproduzidos num plano curvo, permitindo ao utilizador ter uma experiência muito mais imersiva, devido ao fato do conteúdo lhe ser apresentado num ângulo de 170º.

Interface com o utilizador:

Esta componente possui 2 cenas:

Vídeo Selector: esta cena é responsável pela seleção do vídeo a ser reproduzido.

Figura 33- Vídeo Selector

Vídeo Player: é responsabilidade desta cena a reprodução de vídeos.

Figura 34- Vídeo Player

PowerPoint Gallery

Esta componente adiciona ao programa a capacidade de reproduzir PowerPoint em realidade virtual.

Esta componente está dividida em duas cenas:

PowerPoint Selector: é responsabilidade desta cena apresentar ao utilizador uma interface para selecionar o PowerPoint que deseja visualizar.

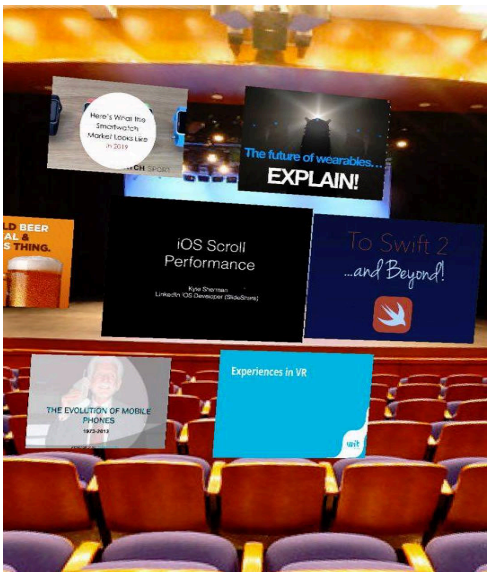

Figura 35- PowerPoint Selector

PowerPoint Player: esta cena tem como objetivo a reprodução do PowerPoint selecionado na cena PowerPoint Selector.

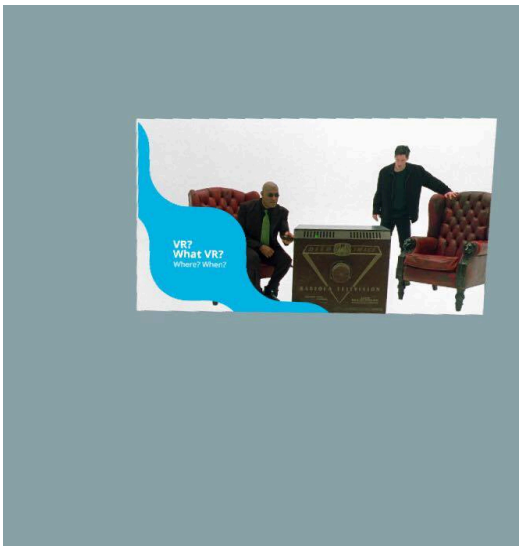

Figura 36- PowerPoint Player

Os slides são substituídos em cada sete segundos, no entanto o utilizador pode controlar o fluxo da apresentação utilizando botões presentes na cena.

Home Screen

Figura 37- Ícone de seleção da PhotoGallery na Home Scene

O Home Screen serve de interface às várias componentes. O utilizador seleciona a componente que deseja simplesmente olhando durante três segundos. Esta componente apenas possui uma cena e esta é sempre a primeira da pilha, sendo a única que nunca sofre a operação de "*pop*".

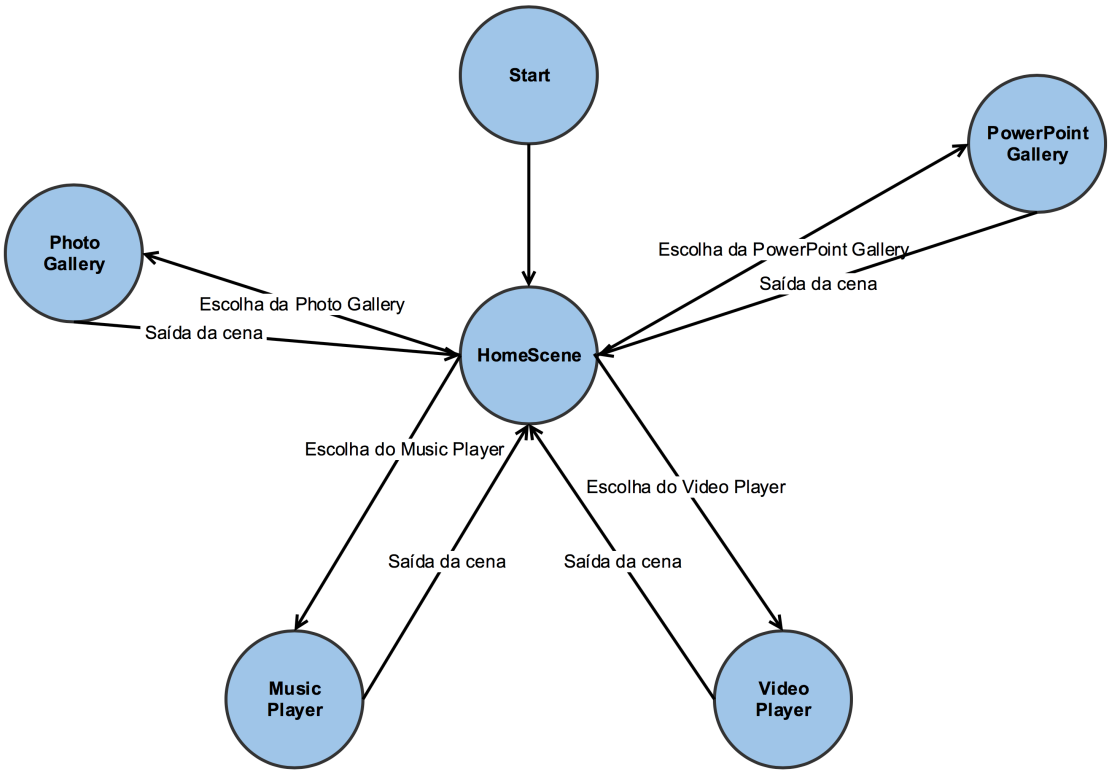

Figura 38- Opções de navegação na Home Scene

Capítulo 7
Arquitetura da solução de software

Nesta secção será exposta a arquitetura do *software* produzido. O foco desta secção será expor as componentes e decisões chave que foram a base do desenvolvimento deste *software*.

7.1. Arquitetura Motor Realidade Virtual

Nesta secção vai ser feita uma breve descrição da arquitetura proposta, para o Motor de realidade virtual a ser desenvolvido. Para mais detalhe consultar o Anexo A.

7.1.1. System View

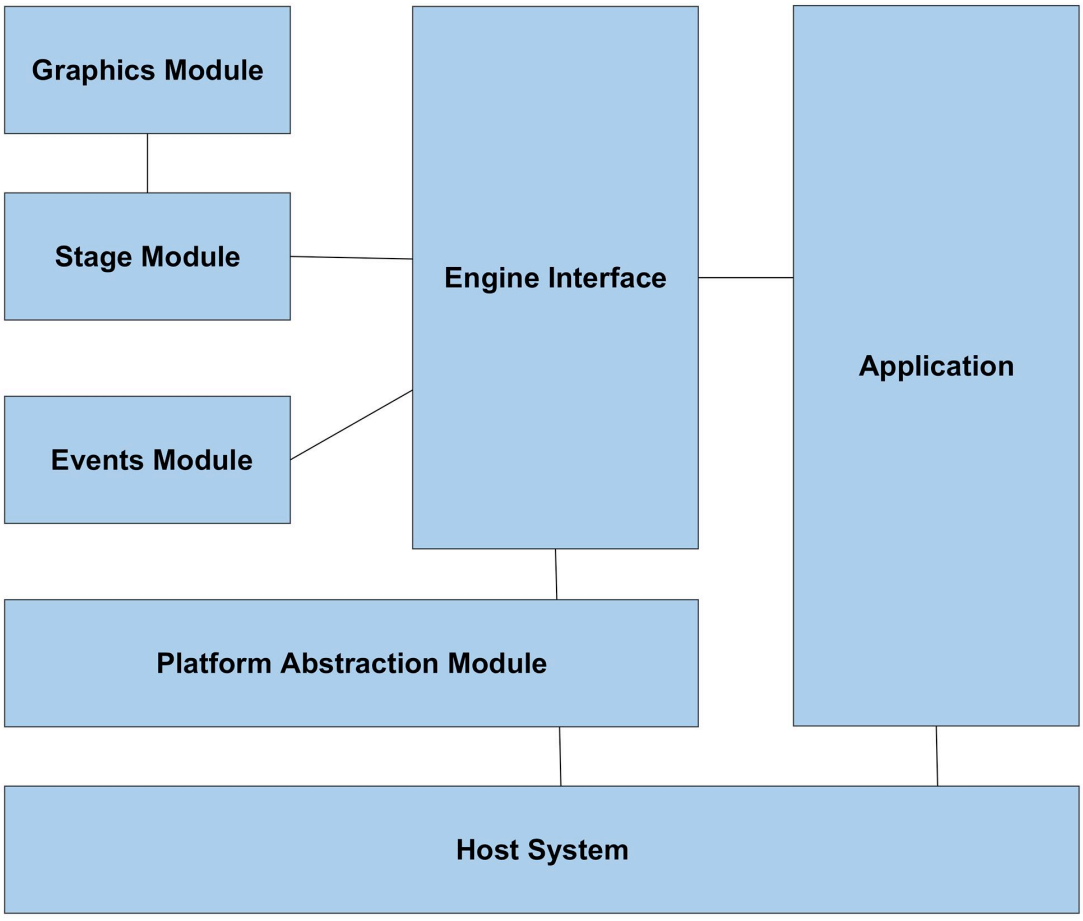

Figura 39- Engine System View

Este sistema tem a responsabilidade de realizar tarefas de forma a oferecer capacidades de *V.R.* à aplicação suportada. O sistema é composto por cinco componentes, esta modularidade oferece a possibilidade de mudar um componente sem ter de mudar outros. Decidi dividir o sistema em cinco componentes porque me pareceu o equilíbrio perfeito

entre simplicidade arquitetural e modularidade. Cada componente é responsável por uma só responsabilidade de alto nível, sendo esta divisão natural. De seguida serão explicitadas as responsabilidades e funções de cada módulo.

Graphics Module: Este componente é responsável pelo desenho de toda a parte gráfica da aplicação. É este componente que é responsável pelo efeito *V.R.* ser criado com sucesso.

Responsabilidades:

- Fornecer modo *V.R.*
- Fornecer modo *Landscape.*
- Fornecer modo *Portrait.*

Comunica com:

- Stage Component Recebe chamadas de desenho e modo atual (*V.R*/ não *V.R*).

Stage Module: A responsabilidade deste componente é controlar o *array* de cenas, as cenas e ordenar à componente gráfica para desenhar.

Responsabilidades:

- Gerir cenas.
- Gerir objetos das cenas.

Comunica com:

- Stage Component: Envia chamadas de desenho e informação sobre o modo atual (*V.R*/ no *V.R*).
- Motor interface: Recebe requests de atualização de cena e informação sobre o modo atual.

Events Modulde: Este componente é responsável pelo *handling* de eventos levantados pelas aplicações suportadas e gerar novos eventos passando-os para a Engine interface.

Responsabilidades:

- Tratamento de eventos.
- Geração de eventos.

Comunica com:

- Motor interface: recebe e envia eventos.

Platform Abstration Module: Neste componente ocorrem tarefas como identificar o modelo do telefone e as configurações que o motor tem de ter para se adaptar. Se já existir uma configuração é feito o loading.

Responsabilidades:

- Configurações.
- Descobrir o modelo do device.
- Guardar configurações e informação relevante.

Comunica com:

- Motor interface: Troca configurações e informação relevante.
- Host System: Pede acesso a capacidades de harware, acesso às capacidades de *hardware*, suporte para *File I/O*, suporte para *input* táctil.

Motor Interface: Este componente é central ao motor já que está envolvido em muitos processos de comunicação indiretos. Esta componente oferece uma interface para a aplicação comunicar com as componentes do motor. É da responsabilidade deste componente permitir comunicação indireta entre o Stage Component, Event Component e Platform Component.

Responsabilidades:

- Fornecer uma interface pela qual a aplicação possa aceder aos componentes do motor.

Comunica com:

- Stage Component: Envia pedidos de atualização de ecrã e modo atual (*V.R*/não-*V.R*).
- Events Component: Trades events.
- Platform Component: Troca configurações e informação relevante
- Aplicação: Troca eventos, erros e informação contextual. Recebe requests para mudar de modo *V.R.* para normal e vice-versa, mudar de cena e outras capacidades

Application: A aplicação suportada tem o uso de realidade virtual como o seu foco. Para usar realidade virtual a aplicação depende inteiramente do uso do motor de *V.R.* A aplicação comunica com o motor por uma interface providenciada pelo mesmo.

Responsabilidades:

- Instanciar o motor de *V.R.*, trocar eventos com a interface do motor, ligar e desligar modo *V.R.*, lógica de aplicação.

Comunica com:

- Motor interface: Troca eventos, erros e informação sobre contexto. Envia pedidos para mudança de modo (*V.R*/*non-V.R*), mudar cena e outras capacidades
- Host System: Acede espaço de memória do programa.

Host System: Tanto o motor como a aplicação são quase independentes de plataforma, apenas precisando de pequenos ajustes para serem portados para outra plataforma. O motor é especificado para suportar um sistema operativo genérico.

Responsabilidades:

- Providenciar acesso a capacidades de *hardwar*e e *file support*.

Comunica com:

- Platform abstration module: Fornece suporte para eventos, erros e informação de contexto. Recebe requests para ligar modo *V.R.*, mudar de cena e outras capacidades.

7.1.2 Physical View

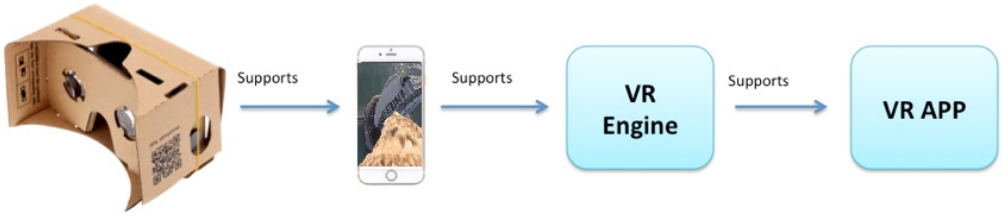

Figura 40- Physical View

Esta view mostra a relação física entre os componentes do sistema, permitindo perceber a posição do motor de realidade virtual em relação ao sistema a desenvolver. O *VR Engine* suporta a app prestando todos os serviços que envolvem reprodução de conteúdo *V.R.*, eventos *V.R.* e acesso ao sistema operativo inerente. O sistema operativo permite acesso ao file system e às capacidades do hardware do dispositivo. O *kit VR* através das suas lentes permite ao sistema reproduzir conteúdo em *V.R.* Para mais detalhe consultar o anexo:

Anexo A- Arquitetura do motor *V.R.*

7.2. Implementação abstrata à plataforma:

Foi pedido pela empresa que a implementação fosse o mais independente da plataforma possível, para o fazer foram tomadas as seguintes decisões.

- Usar OpenGL ES 2 (38) como plataforma gráfica, já que esta plataforma é compatível com Android bem como Windows Phone. A alternativa seria usar uma *API* gráfica da Apple como o Metal ou o *SceneKit*.
- Utilizar o C++ como linguagem principal, foi da responsabilidade do código produzido com esta linguagem toda a lógica do programa e interação com a componente gráfica. Certas *APIs* de acesso ao hardware do device requeriam uso de Objective-C. Para criar um bom nível de abstração mesmo usando uma linguagem especifica a plataforma foram criadas várias classes *bridge*, estas classes escondiam a complexidade do Objective-C, tornando as chamadas desses métodos independentes de plataforma.

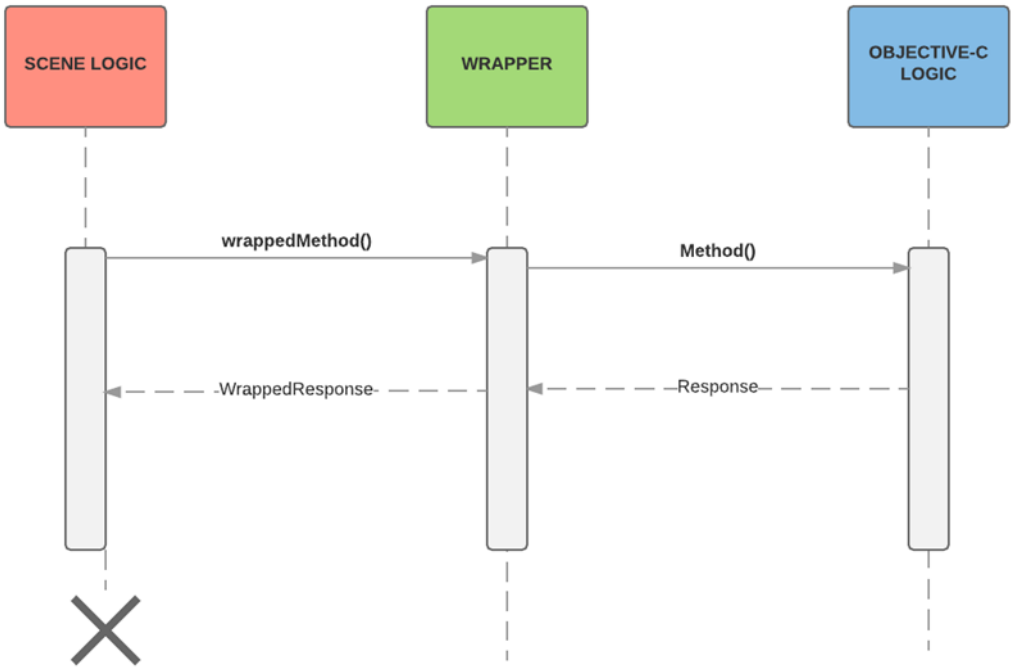

Figura 54- Diagrama de sequência representando a chamada de um método Objective-C encapsulado por uma interface C++

7.3. Tecnologias

Nesta subsecção serão descritas as tecnologias que vão ser usadas no decorrer do projeto. Foram necessárias várias linguagens de programação e algumas tecnologias para interagir com conteúdo multimédia.

7.3.1. Objective-C

O Objective-C (38) é a mais antiga e madura linguagem de programação nativa do iOS (39). Esta linguagem permite o uso de C/C++ simultaneamente de forma transparente. Este fator e o facto de ser uma linguagem madura tornou-a uma escolha óbvia. A alternativa era usar Swift, esta linguagem é recente (lançada em junho de 2014) e não integra C/C++ de forma tão simples como o Objective-C, por estes motivos o Objective-C apresenta vantagens suficientes para justificar a sua escolha.

7.3.2. C++14

O C++ (40) será usado para ajudar a garantir a portabilidade do código. Todo o código que não interaja diretamente com o device vai ser escrito nesta linguagem. Será escolhida a versão mais atual devido a várias melhorias em relação às mais antigas, nomeadamente devido à adição de shared pointers já que estes facilitam o processo de gestão de memória.

7.3.3. OpenGL ES 2

A componente gráfica será construída através do uso do OpenGL ES 2 (38), já que é uma API gráfica bastante usada e documentada, lightweight e compatível com várias plataformas móveis além de iOS como Android e Windows Phone. Não foram escolhidas versões mais recentes já que estas não apresentam melhorias significativas em relação à mesma e o número de devices compatíveis é drasticamente menor.

7.3.4. Libav

Libav (41) é uma biblioteca cross-platform que permite converter, manipular e reproduzir vários tipos de formatos multimédia. Esta biblioteca será usada na componente de reprodução de vídeo do programa a desenvolver.

7.3.5. OOXML

OOXML (42) é uma especificação *open-source* de documentos de escritório. Foi proposta e desenvolvida por várias empresas, tal como a Microsoft. Este formato é usado no Microsoft Office. Esta especificação será estudada e aplicada na construção do parser responsável pela componente de PowerPoint da Aplicação.

7.3.6. XPath

XPath (43) é uma *querry language* que permite aceder a conteúdos de um ficheiro XML através de querrys. Esta linguagem será usada para fazer *parsing* aos ficheiros PowerPoint. O XPath tem como vantagem evitar a necessidade da escrita de um *parser* de raiz. Isto poderia levar à existência de *bugs* ou uma performance inferior.

7.4. Hierarquia de objetos:

Para criar uma interface simples para desenvolver cada um dos componentes foram criadas classes responsáveis pelos objetos presentes no ambiente gráfico. Estas classes possuem uma relação de derivação entre elas, desta forma é fácil criar um conjunto de classes com propriedades partilhadas.

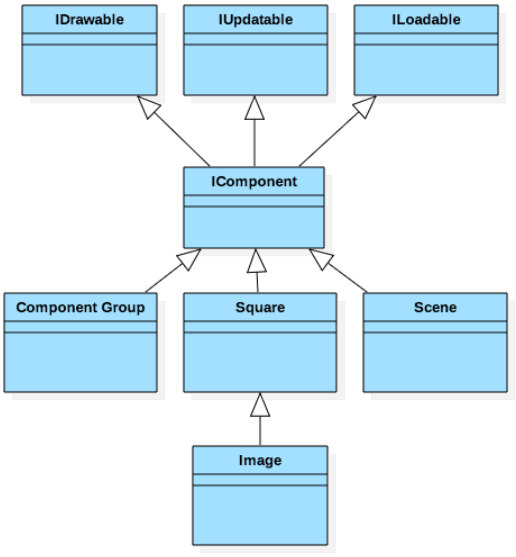

Figura 41- Hierarquia de Classes

Este diagrama demonstra a relação hierárquica entre as várias classes. É possível ver que por exemplo, um objecto tipo Image e tipo Scene partilham as propriedades associadas ao objeto IComponent. Por sua vez o objeto IComponent herda todas as propriedades de um objeto IUpdatable, IDrawable e ILoadable.

IUpdatable: Esta classe possui a capacidade identificar se o utilizador está a olhar para um objeto teste tipo, permitindo lançar eventos. Esta classe é essencial para permitir que o utilizador interaja com o software.

IDrawable: Esta classe possui propriedades que permitem desenhar um objeto no contexto OpenGL actual.

ILoadable: Esta classe permite carregar objetos de vários tipos de forma assíncrona registando-os numa lista associada ao objecto ILoadble.

IComponent: Esta classe oferece aos seus objetos capacidades de exibirem animações bem como de manipular a sua posição no espaço permitindo operações como: translação, rotação e manipulação do tamanho do objeto.

Square: Esta classe oferece aos seus objetos capacidade de realizar rotações em relação a um ponto definido.

Scene: Esta classe funciona como container de objetos. Cada scene contém 0 ou mais objetos dos diversos tipos desenhados bem como uma cor de fundo.

ComponentGroup: Esta classe é um container de vários objetos, no entanto pode ser incluída dentro de uma cena, podendo existir vários ao mesmo tempo e não possuindo uma cor de fundo. Esta classe é útil na medida que permite instanciar a mesma operação a todos os objetos contidos um objeto deste tipo.

Image: Esta classe permite associar uma textura a um objeto. Esta textura pode ser manipulada associando diferentes shaders aos objetos deste tipo. É também possível personalizar a resolução da imagem.

7.5. Sistema de eventos:

As mudanças do estado do programa são suportadas por um sistema de eventos, cada evento é constituído por dois elementos:

- Nome
- *Payload*

O programa está preparado para tratar vários tipos de eventos, estes são identificados pelo seu nome. A payload serve para personalizar a resposta do motor daquela instância em particular.

Quando uma ação causa um evento, este é criado e inserido numa fila de eventos. A cada ciclo de criação de nova frame é da responsabilidade da função de "*update*" verificar se existem eventos na fila de eventos. Se existirem eventos pendentes estes são tratados de acordo com o seu tipo e *payload*.

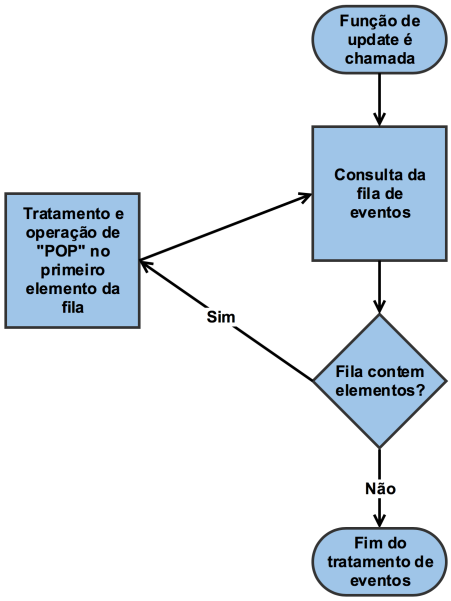

Figura 42- Diagrama de fluxo representando o processo de tratamento de eventos.

Tipos de eventos mais relevantes:

EVENT_GAZE_START- Este evento é lançado quando o utilizador olha para um objeto. É extremamente útil para indicar o momento em que certas animações devem ser capturadas.

EVENT_GAZE_END- Este evento é lançado quando o utilizador olha para um objeto. É extremamente útil para indicar o momento em que certas animações devem ser desativadas.

EVENT_GAZE_LONG_STARE- Este evento é lançado quando o utilizador olha para um objeto por um período de tempo definido, este intervalo de tempo é definido para cada objeto. Este evento é usado para encapsular o *input* do utilizador.

EVENT_SCENE_PUSH-Este evento é lançado quando um *input* leva à necessidade da colocação de uma nova cena na pilha de cenas.

EVENT_SCENE_PUSH_STARTED- Este evento é lançado antes da colocação de uma nova cena na *stack*.

EVENT_SCENE_PUSH_COMPLETE- Este evento é lançado depois da colocação de uma nova cena na *stack*. Este evento é extremamente útil porque permite que os inputs sejam bloqueados até este evento seja lançado ou pode servir para identificar quando certos efeitos sonoros são inicializados.

EVENT_SCENE_POP- Este evento é lançado quando um input leva à necessidade de retirar a cena atual do topo da pilha de cenas.

EVENT_SCENE_POP_STARTED- Este evento é lançado antes de uma cena ser retirada da *stack*. É útil para limpar recursos e parar a reprodução de efeitos sonoros.

EVENT_SCENE_POP_COMPLETE- Este evento é lançado depois da cena do topo da stack ser retirada.

EVENT_STOP_SONG- Este evento é lançado quando o *payload* de um evento EVENT_SCENE_POP_STARTED é referente ao módulo de reprodução de música, permitindo parar a reprodução da mesma.

EVENT_CHANGE_COVER- Este evento é lançado quando o *payload* de um evento EVENT_SCENE_LONG_STARE é referente a mudança da música a ser reproduzida no contexto do módulo de reprodução música. Este evento causa o carregamento da capa do álbum da próxima música a ser reproduzida.

EVENT_SLIDE_END- Este evento é lançado no contexto do módulo de reprodução de PowerPoint. É útil já que permite anunciar ao Motor o momento em que deve libertar recursos do slide atual e começar o carregamento do próximo slide.

EVENT_NEXT_SLIDE- Este evento é lançado no contexto do módulo de reprodução de PowerPoint. É útil já que permite anunciar o momento em que o próximo slide deve ser desenhado.

7.6. Recursos extra para Realidade Virtual:

Para lidar com o processo de gerar uma experiência de realidade virtual usou-se um *port* da biblioteca oficial do Google Cardboard. Esta biblioteca oferece os seguintes serviços:

- **Headtracking**- Este SDK garante uma interface para oferecer à aplicação tracking preciso do ângulo da cabeça do utilizador.
- **Distortion correction**- Este SDK permite corrigir a distorção em cada uma das duas imagens geradas pela aplicação.

Estas duas funcionalidades foram integradas com o motor desenvolvido, poupando tempo de implementação. De notar que a este SDK apenas possui estas capacidades, tudo o resto foi da minha responsabilidade.

7.7. Paralelismo

Assegurar o uso correcto e eficiente de paralelismo (44) foi uma das maiores preocupações do design desta aplicação. Estes problemas levaram à necessidade de o design desta aplicação ter como base paralelismo:

- O Open GL não pode correr na mesma *thread* que a lógica, já que desta forma a actualização do contexto gráfico teria de esperar pelo processamento lógico causando uma frame-rate não aceitável.
- O Open GL e lógica não podem ficar parados à espera do carregamento de resources. Muitos resources são bastante grandes, isto levaria a que sempre que uma textura fosse carregada a aplicação seja forçada a parar até o processo estar completo.
- O iOS restringe o uso do Open Gl na main thread.

Para combater estes problemas mencionados foram tomadas as seguintes decisões:

- A main thread é responsável por toda a lógica do programa.
- Uma thread é responsável pelas chamadas Open GL.
- Outra thread é responsável pelo carregamento de recursos.

Esta abordagem permite dividir responsabilidades, tornando a aplicação mais responsiva. Este diagrama exemplifica o uso de paralelismo na construção e desenho de uma imagem.

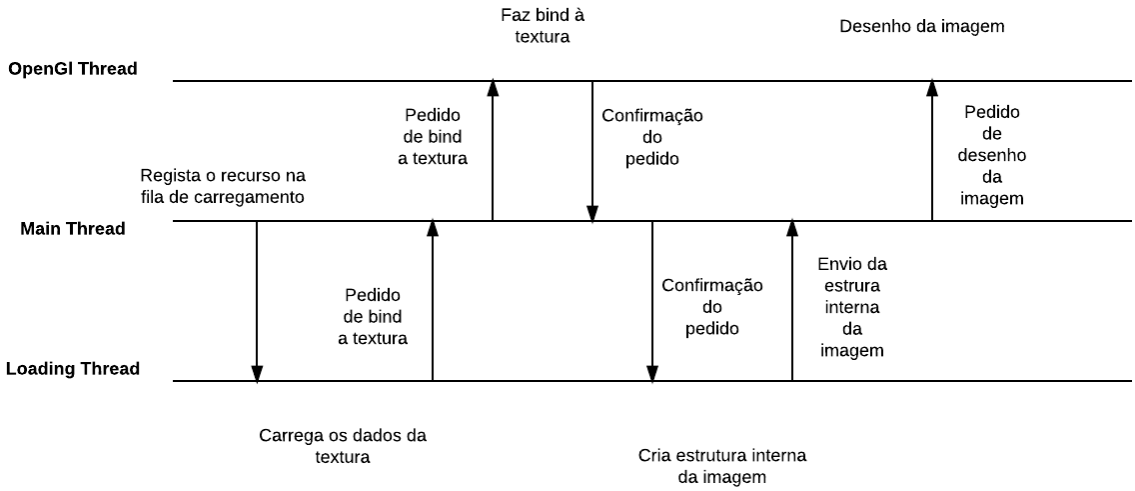

Figura 43- Loading a Image Process

7.8. Desenho:

Esta secção vai explicar o processo de desenho neste *software*. Esta explicação será dividida em várias secções já que existe independência entre as várias componentes deste processo, a alteração de uma não implica a alteração das outras.

Contexto Gráfico

Para a construção do ambiente gráfico foi usado o Open Gl ES 2. Quando o programa é iniciado este requer as dimensões do dispositivo ao sistema operativo que serão essenciais na construção do contexto Open GL. De seguida este instancia o contexto Open Gl, o render buffer, o frame buffer e o display link. Antes de desenhar alguma coisa é necessário existirem shaders carregados, por isso alguns shaders essenciais são carregados neste momento.

O processo de render é dividido em dois tipos diferentes, o render em modo *V.R.* e o render normal. O render normal utiliza a posição da cabeça bem como a perspetiva definida na inicialização do programa. O render em *V.R.* utiliza as mesmas propriedades que o render normal, no entanto estes render é realizado duas vezes em dois viewports diferentes. O ecrã é assim dividido em dois viewports. No entanto para o efeito *V.R.* ser realizado correctamente cada viewport tem de mostrar uma imagem ligeiramente diferente do outro, sendo isto da responsabilidade do *SDK* do Google Cardboard.

Objetos gráficos

Todos objetos gráficos herdam da classe IComponent, devido a isto todos partilham as seguintes propriedades bem como funções com a capacidade manipulação das propriedades mencionadas:

- Posição
- Tamanho
- Rotação
- Visibilidade

- Gaze Detection
- Dados personalizados

Transformações gráficas

As classes IComponentGroup e Scene permitem encapsular vários objetos IComponent. As alterações a estes objetos são depois transmitidas aos objetos encapsulados permitindo operações como rodar um conjunto de objetos da mesma forma ou realizar a mesma translação a todos os objetos da cena.

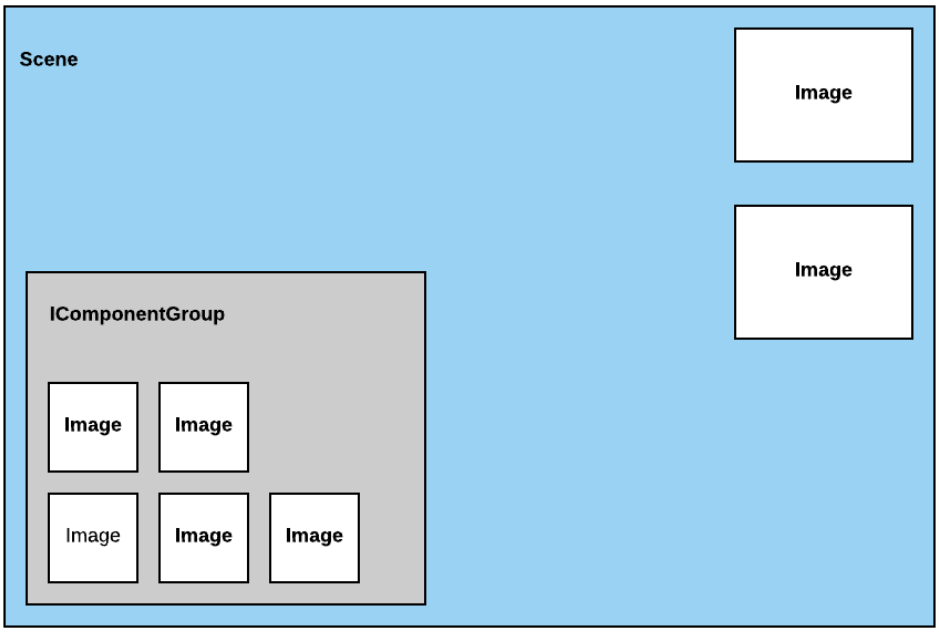

Figura 44- Container diagram contento os objetos encapsulados em uma scene.

7.9. HeadTracking:

Para realizar o *headtracking* recorri ao port para iOS do *SDK* do Google Cardboard, este fornece leituras da posição da cabeça. A cada iteração do loop de desenho é consultada esta posição e feita uma translação a todo o contexto Open Gl.

7.10. Carregamento de recursos:

Desenvolvi um sistema de carregamento para vários tipos de recursos como texturas e shaders. Este processo foi levemente ilustrado na secção de paralelismo. Este processo começa com o registo do pedido do carregamento numa fila de pedidos. A cada interação do ciclo de desenho, uma *thread* vai verificar se existem pedidos, se existirem inicia-se o processo de carregamento, todo este processo é efetuado na mesma *thread* excetuando o caso do *bind* das imagens, já que o pedido tem de ser feito pela main *thread* à *thread* responsável pelo Open Gl. Depois de carregada a textura o recurso é registado numa lista de *Assets* pertencendo a cena que requisitou o recurso.

7.11. Animações:

Um pouco por toda a aplicação vários objetos encontram-se animados, estas animações têm três objetivos: embelezar e tornar o programa mais atrativo, simbolizar que o utilizador está a iniciar uma ação ou simbolizar que o utilizador acabou de realizar uma ação.

Foram implementados vários tipos de animações, a maior parte das animações leva como parâmetro o objeto afetado e a duração da animação. As mais relevantes são:

- **Fade in**- Esta animação consiste em inicialmente desenhar um objeto com elevada transparência e progressivamente reduzi-la até um valor nulo.
- **Fade out**- Esta animação consiste em aumentar a transparência de um objeto até ele desaparecer.
- **Fade between**- Esta animação consiste em variar a transparência de uma imagem entre dois valores estipulados.
- **Move to**- Esta animação move o objeto da sua posição atual para a posição definida num movimento com a duração do intervalo de tempo estipulado como parâmetro.
- **Move between**- Esta animação move o objeto entre duas posições definidas num movimento com a duração do intervalo de tempo estipulado como parâmetro.
- **Scale to**- Esta animação altera o tamanho de um objeto entre o tamanho inicial e uma escala definida.
- **Scale between**- Esta animação altera o tamanho de um objeto com uma escala definida.
- **Rotation**- Esta animação roda um objeto num determinado ângulo.
- **Rotation between**- Esta animação roda um objeto entre um intervalo de ângulos.
- **Fade Background**- Esta animação altera o fundo da cena entre o valor atual e o valor especificado.

As animações funcionam da seguinte forma: quando é inicializada uma animação um "relógio" é inicializado, a cada frame é alterada a propriedade que é suposto ser afetada pela animação, o quanto é alterada essa propriedade por frame é dependente do tempo da animação. Quando o relógio atinge o período especificado para o fim da animação esta acaba.

7.12. Player de vídeo:

Um dos requisitos desta aplicação foi a necessidade de reproduzir vídeo, para isto é necessário um *player*. Procurei vários *SDKs* que reproduzissem vídeo e oferecessem a hipótese de "desenhar o vídeo" num contexto Open Gl, no entanto não encontrei nenhum. Para resolver este problema resolvi construir um motor de reprodução de vídeo.

Para resolver o problema do motor de vídeo funcionar inserido num contexto OpenGl, tomei a decisão que o motor construído teria como base o principio de que cada frame seria desenhado como textura.

A recolha de frames foi realizada com a biblioteca Libav (41) como base. Esta é uma biblioteca *open-source* que auxilia na reprodução de vários formatos de vídeo. Foi usada para extrair cada frame do vídeo. Cada frame depois foi convertido numa textura e desenhado. O

diagrama seguinte exemplifica o processo de reprodução de vídeo, com foco na extração de cada frame.
Este diagrama explicita o algoritmo usado na sincronização dos frames.

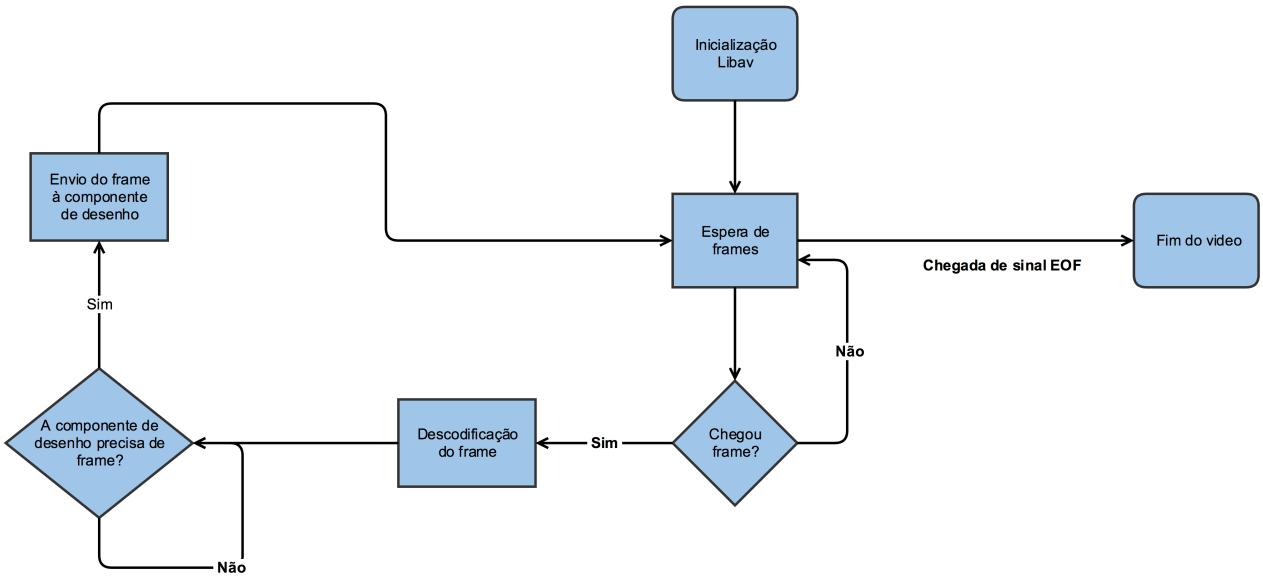

Figura 45- Captura e sincronização de frames

Não foi implementada a reprodução do som, já que este iria acrescentar pouco valor em relação ao trabalho acrescido que iria provocar. A reprodução de som iria exigir uma forma mais complicada de sincronizar frames, já que o som também teria de ser sincronizado.

7.13. Fundo Dinâmico

Tanto no módulo PowerPoint, como no de reprodução de música foi usada esta ferramenta. Para extrair esta cor criei uma CGImage e calculei a média de cada componente *RGB* de todos os pixéis da imagem. Foi usada uma optimização: a imagem foi transformada numa versão Jpeg de má qualidade e com resolução inferior, permitindo um cálculo muito mais rápido e muito menos exigente em memória. De seguida a cor do fundo é mudada para a cor calculada. Quando a imagem alvo muda, este processo é repetido.

7.14. Fotos 360º

Foram usadas várias fotos 360º como fundo de várias cenas, estas foram com o intuito de aumentar a imersão do utilizador. Foram escolhidas algumas fotos de uso livre recolhidas na internet. Cada foto foi cortada a meio, sendo o corte ao longo do eixo dos "y". Cada uma das imagens foi mapeada numa superfície curva definida por uma malha de pontos. O número de pontos desta malha é proporcional à resolução da imagem bem como ao tamanho da esfera criada pela união destas duas superfícies.

7.15. Shaders

O OpenGL ES 2 obriga ao uso de *shaders* para qualquer operação de desenho. Além de um *shader* básico escrevi vários *shaders* que adicionaram efeitos interessantes ao programa, tendo

a mesma utilidade que as animações. Os *shaders* utilizados foram escritos usando a linguagem GLSL.

Os shaders utilizados foram os seguintes (de notar que cada *shader* é dividido em um vertex shader e um fragment shader):

- **Basic shader**- *Shader* responsável pelo desenho da maior parte dos objetos, não implementa nenhum efeito.
- **Pointer shader**-*Shader* responsável pelo desenho do ponteiro.
- **Progress bar shader**- *Shader* responsável pelo desenho da barra de progresso da reprodução de música.
- **XRay shader**- *Shader* responsável por um efeito de seleção. Este efeito consiste na alteração da cor de um objeto ao longo do eixo dos "x".
- **Selection shader**- *Shader* responsável por um efeito de seleção. Este efeito consiste na alteração da cor de um objeto ao longo do eixo dos "x" da cor original para uma cor sólida.
- **Emboss Shader** - *Shader* responsável por um efeito de seleção, este efeito consistiu em alterar a cor do objeto desenhado da cor original até apresentar um efeito de "*emboss*", este efeito consiste na alteração da cor original até uma versão preto e branco com os relevos bastantes salientados. Esta transição é continua e é controlada pelo tempo que o utilizador olha para o objeto

7.16. Photo Gallery

O grande desafio no desenvolvimento desta componente do programa foi o carregamento das fotos do device. Para atingir esse fim usei a biblioteca nativa da Apple para este fim: PHimage Manager.

A biblioteca PHimage Manager é parte da Framework Photos, esta Framework foi adicionada na versão 8 do iOS e é responsável por aceder aos conteúdos fotográficos guardados no device.

Com a biblioteca mencionada é possível construir uma lista com objetos do tipo *Asset*, estes objetos contêm informação sobre cada uma das fotos na biblioteca. No caso desta aplicação, esta lista foi ordenada por ordem cronológica.

Utilizando os *Assets* é possível extrair o diretório correspondente a cada foto, assim é possível construir uma lista de diretórios.

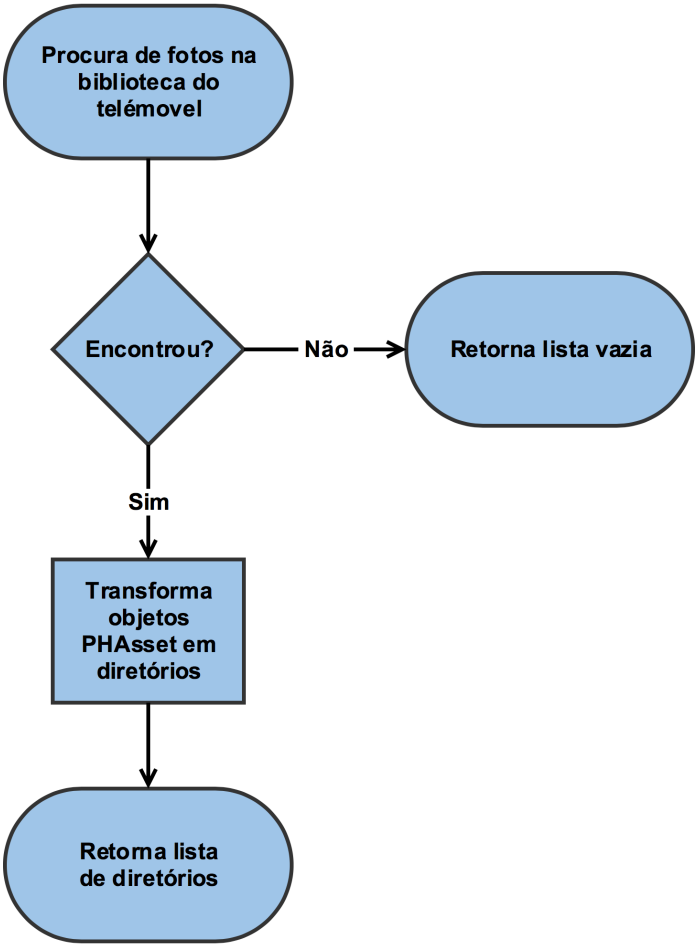

Figura 46- Processo de carregamento de procura dos diretórios das fotos

O carregamento das fotos é feito sempre que a interface o exigir. Assim sempre que seja necessário o carregamento de uma foto é repetido o seguinte processo: através de um diretório é criada uma UImage (tipo de imagem nativo do iOS) como não é possível passar diretamente uma UImage ao OpenGL extrai-se a *raw data* deste objeto resultando numa textura que será referenciada ao OpenGL.

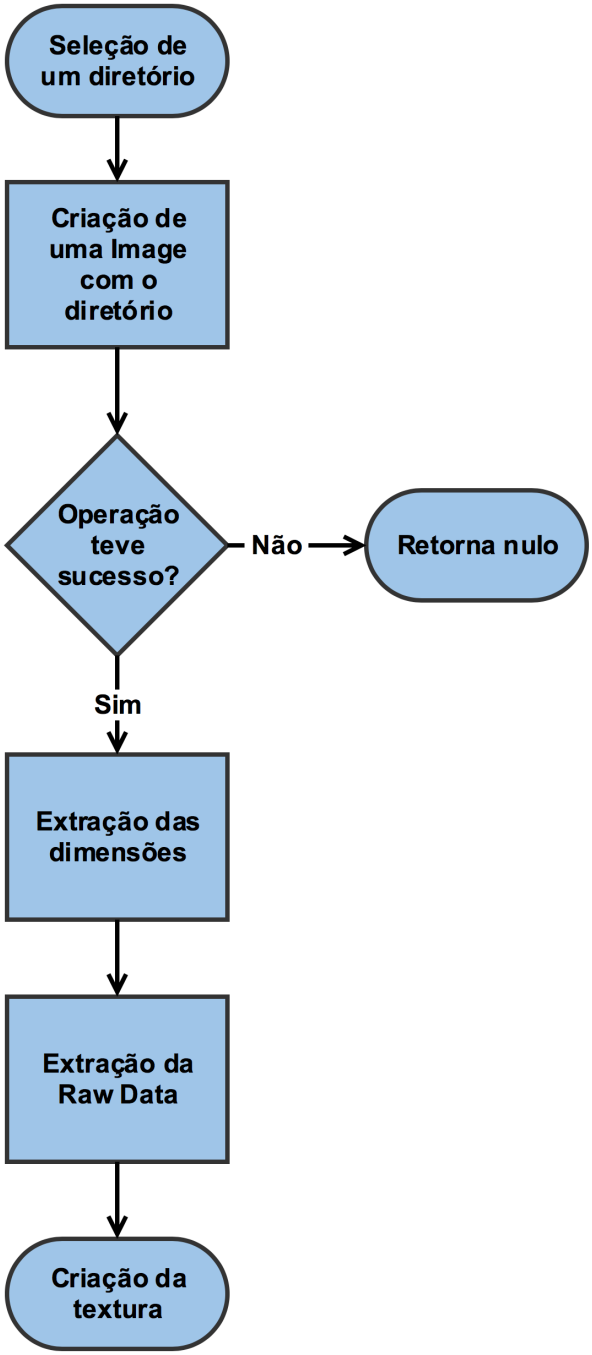

Figura 47- Processo de criação de uma textura a partir de um diretório

7.17. Music player

Na construção deste componente usei a biblioteca nativa do iOS para o efeito, MPMusicPlayerController. Esta biblioteca permite controlar o player nativo para reproduzir as músicas em *background*. Este diagrama ilustra o processo.

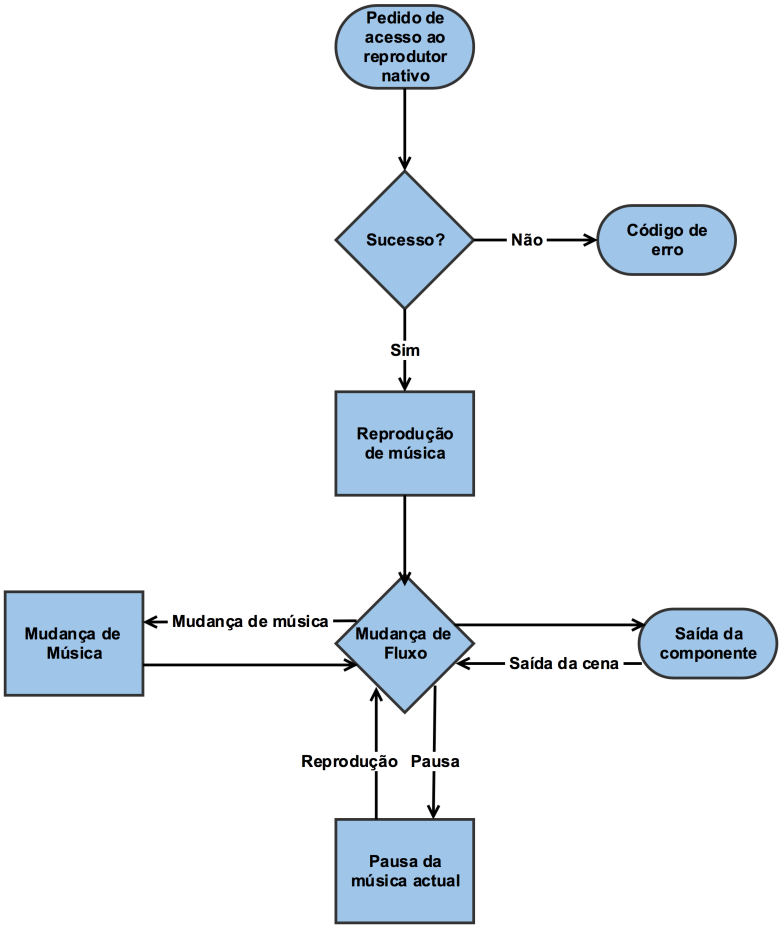

Figura 48- Fluxo de reprodução de música

Para aceder ao *player* nativo era necessário instanciar um objeto MPMusicPlayerController. Este possui métodos que permitem aceder e manipular o estado atual do *player*. Foi criada uma interface entre os botões da UI e estes métodos, permitindo ao utilizador controlar o fluxo de reprodução da sua música. Foi também carregada a foto de capa do álbum atual, o processo do seu carregamento é semelhante ao do carregamento de fotos, no entanto neste caso o diretório é enviado pela biblioteca MPMusicPlayerController fornece um diretório correspondente ao à capa do álbum da música atual. A capa do álbum é útil também no processo de construção da cor de fundo, esta varia consoante a capa do álbum, num processo que envolve o cálculo da cor média da capa do álbum, mais à frente esse processo será explicado.

De notar o fundo da cena é a cor média da capa do álbum, o processo para a construção desta cor será explicado na subsecção outros.

7.18. Video Gallery

Reprodução de Vídeo:

Para a reprodução de vídeo foi usado o *player* construído, no entanto foi necessário ter cuidados extra para garantir que a experiência fosse imersiva. Para garantir imersão no conteúdo (vídeos 170º) o vídeo foi desenhado numa tela virtual curva.

Efeito Curvo:

Este efeito foi criado mapeando a textura numa superfície curva, nesta textura o ponto mais próximo, o mais distante e o ponto central (onde está o utilizador) formam um ângulo de 170º. Conseguindo assim um efeito indicado para a reprodução destes vídeos de forma *widescreen*.

7.19. PowerPoint Gallery

Esta componente adiciona ao programa a capacidade de reproduzir PowerPoint. Para o fazer estudei a especificação OOXML (42). O OOXML é a especificação pela qual os documentos .docx se regem, sejam eles originários do Microsoft PowerPoint ou outra alternativa. Depois de estudada a especificação, desenhei um processo que levou à reprodução de PowerPoint num ambiente *V.R.* imersivo.

OOXML e a sua estrutura interna

OOXML (office open xml document) é uma especificação desenvolvida pela Microsoft para suportar os vários diferentes tipos de documentos integrados no Microsoft Office. Esta especificação visa facilitar a manipulação destes ficheiros, criando compatibilidade entre a solução de escritório da Microsoft e de outras companhias, bem como facilitando manipulação de dados.

Um ficheiro Docx é um ficheiro zip. Depois de descompactado revela uma estrutura interna constituída por várias pastas, conteúdos multimédia e ficheiros xml.

Para este estágio era necessário explorar apenas documentos de apresentação, estudei mais especificamente o subformato presentationML. Um típico ficheiro do formato em cima mencionado tem a seguinte estrutura:

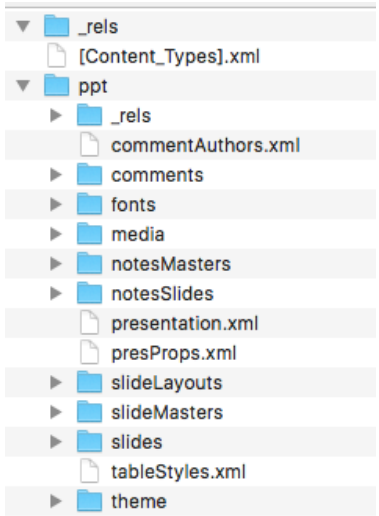

Figura 49- Estrutura interna OOXML

Este tipo de documentos descreve uma apresentação através de um sistema de relações, é possível estabelecer relações entre diferentes componentes no mesmo ou em diferentes ficheiros.

Abordagem

A abordagem para reproduzir uma apresentação de PowerPoint consistiu em extrair todas as imagens da apresentação e de seguida na construção de uma árvore contendo as imagens de cada slide bem como informação do modo como devem ser apresentadas. Ignorei a presença de texto porque a sua ausência pode ser facilmente colmatada usando imagens com texto.

A árvore contruída tem a seguinte estrutura:

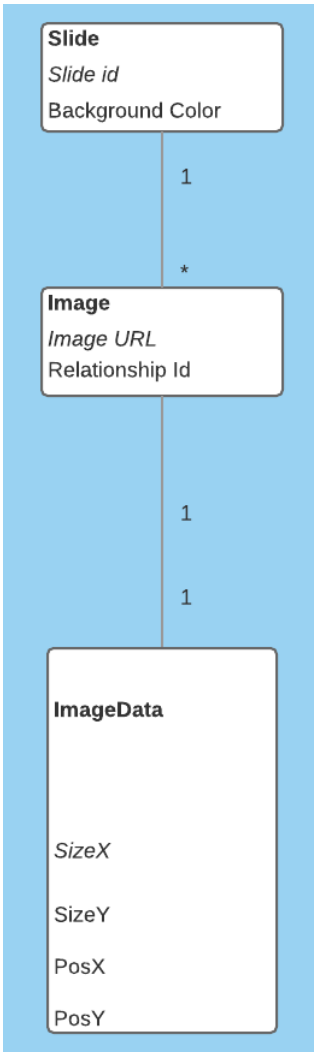

Figura 50- Estrutura interna de cada elemento da árvore de Parsing

Parsing

O seguinte diagrama de fluxo explica o processo de *parsing*:

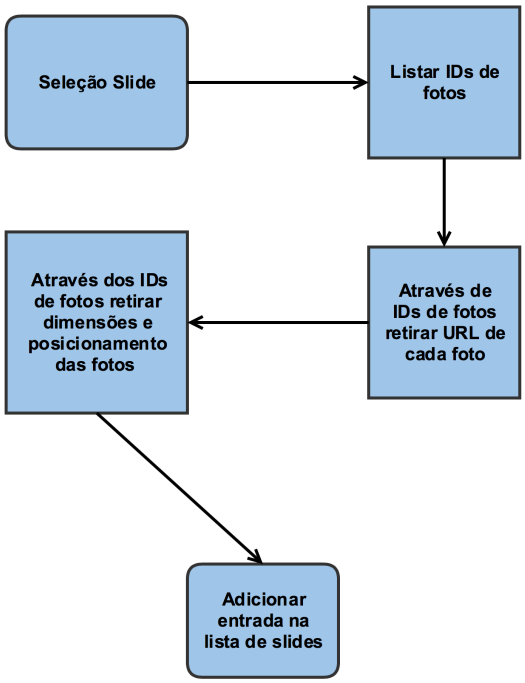

Figura 51- Fluxo de parsing de um slide

Utilizando como "chave" o id de cada imagem e as suas várias relações foram retiradas as informações necessárias de vários ficheiros xml, este processo realizado com a ajuda da biblioteca Libxml2. O libxml2 é uma biblioteca que fornece capacidades de parsing de ficheiros xml usando Xpath (43).

Mapping das Coordenadas

As coordenadas e dimensões presentes na especificação OOXML estão em *english metric units* (EMUs), estas para puderem ser usadas, tiveram de ser convertidas em pixéis. Para realizar a conversão usei a seguinte formula:

$$PixelData = EmuData/9525$$

Reprodução de PowerPoint

Para reproduzir PowerPoint foi criada uma cena em que é desenhada cada imagem do slide a ser reproduzido. Cada imagem é extraída da árvore mencionada anteriormente e com as coordenadas mapeadas de forma a serem desenhadas.

De notar que o a cor de fundo da cena é a average color do PowerPoint atual, o processo para a construção desta cor será explicado na subsecção outros.

Capítulo 8
Trabalho efetuado

Foram implementados todos os *user stories* e respeitados todos os requisitos exceto os mencionados na secção seguinte.

8.1. User Stories não implementadas

As seguintes user stories não foram implementadas:

Motor VR:

- **USE007**- "Como utilizador eu quero enviar inputs ao programa usando interface táctil."
  Este requisito foi ignorado já que para colocar uma interface tátil teria de ser desenvolvido um outro ambiente que não simulasse um ambiente virtual em 360º e sim um ambiente estático.
- **USE008**- "Como utilizador eu quero enviar inputs para o programa através do botão de *trigger*."
  Foi decidido que o *input* do utilizador seria feito apenas pelo olhar, por esse motivo este requisito foi descartado.

Componente de galeria fotográfica:

- **USF003**- "Como utilizador quero visualizar a minha biblioteca fotográfica guardada num serviço de *cloud*."
  Este requisito foi ignorado porque foi decidido em colaboração com o orientador da empresa que este requisito traria pouco beneficio para a empresa em relação ao tempo de implementação.
- **USF004**- "Como utilizador quero apagar o meu conteúdo utilizando a mesma aplicação que usei para o consumir."
  Este requisito foi ignorado porque foi decidido em colaboração com o orientador da empresa que este requisito traria pouco beneficio para a empresa em relação ao tempo de implementação.

Componente de reprodução de música:

- **USM004**-"Como utilizador quero controlar o fluxo da reprodução da música atual ."
  Não foi implementada nenhuma forma do utilizador controlar a velocidade com que a faixa é reproduzida bem como avançar e recuar na mesma.

Componente de reprodução de vídeo:

- **USV003** - "Como utilizador quero assistir a conteúdo proveniente de uma action camera no meu device de *V.R.* através de streaming."
  Este requisito foi ignorado porque foi decidido em colaboração com o orientador da empresa que este requisito traria pouco beneficio para a empresa em relação ao tempo de implementação.

- **USV004** - "Como utilizador quero apagar o meu conteúdo utilizando a mesma app que usei para o consumir"
  Este requisito foi ignorado porque foi decidido em colaboração com o orientador da empresa que este requisito traria pouco beneficio para a empresa em relação ao tempo de implementação.
- **USV005**- "Como utilizador quero controlar o fluxo do vídeo."
  Este requisito foi ignorado porque foi decidido em colaboração com o orientador da empresa que este requisito traria pouco beneficio para a empresa em relação ao tempo de implementação.

- **USV006**-"Como utilizador quero que a aplicação se adapte automaticamente ao dispositivo de realidade virtual que estou a utilizar para conseguir a máxima qualidade de experiência."
  Este requisito foi ignorado devido ao port da *API* oficial do Google Carboard não suportar esta feature.

Componente reprodução Powerpoint:

- **USP002** - "Como utilizador, eu quero ver conteúdo espalhado à sua volta de forma a sentir-me imerso."
  Este requisito ficou parcialmente desenvolvido, no entanto vários bugs estavam a atrasar o projeto. Na *build* final acabou por ser ignorado.
- **USP003** - "Como utilizador quero puder visualizar conteúdo multimédia como vídeos na apresentação."
  Este requisito acabou por ser ignorado e o tempo que lhe foi alocado acabou por ser usado para fins que o orientador da empresa achou mais urgente, nomeadamente tratamento mais cuidado da *UI*.

8.2 Requisitos não-funcionais ignorados

Os seguintes requisitos não foram cumpridos:

Motor VR:

- **NFE011** - "O evento de trigger deve ser reconhecido em contexto."
  Este requisito não foi cumprido já que não foi usado o *trigger* como input.

Componente de reprodução de vídeo:

- **NFV002** - "O som e o vídeo devem ser sincronizados."
  Não foi tido em conta a reprodução de vídeo, dessa forma não se deu a sincronização aqui falada.

8.3 Mitigações efetuadas

Para conseguir responder a estes requisitos as suas estratégias de mitigação foram realizadas as estratégias de mitigação definidas.

Motor VR:

- **RISKF002** - "A resolução do vídeo ou telemóvel podem não ser suficientes."

  Estratégia: "Recomendar ao utilizador modelos que sabemos que a experiência seja satisfatória."

  A experiência apenas tem qualidade suficiente em iPhones com ecrã com

resolução igual ou superior ao do iPhone 6.

Music VR:

- **RISKM001**- "Restrições de acesso ao sistema de ficheiro podem causar problemas."

  Estratégia: "Estudar as capacidades de acesso ao file system do *SDK*."

  Foram encontradas várias limitações no acesso ao *file system*, tendo sido necessário optar por usar o *player* nativo em *background*.

PowerPoint VR:

- **RISKP002**- "Apresentar texto em *V.R.* é conhecido por ser complicado. "

  Estratégia: "Utilizar imagens em texto em vez de o renderizar."

  Renderizar texto apresentou-se uma tarefa extremamente complicada, por esse motivo foi utilizada a estratégia de mitigação definida.

Capítulo 9
Testes

Esta secção apresenta e analisa os testes realizados. A abordagem utilizada seguiu levemente os princípios de uma Avaliação Heurística (45).

Esta análise tinha como objetivo encontrar problemas de interação, bem como problemas gráficos. Foram escolhidas 5 pessoas que trabalhavam na WIT Software e foi-lhes pedido que testassem a aplicação e respondessem a algumas perguntas. Estas perguntas foram desenhadas com o objetivo de reunir o máximo de informação com o mínimo de aborrecimento para a pessoa que participou no teste, já que se o teste se alongasse as respostas das pessoas já aborrecidas perderiam qualidade.

Perguntas realizadas

Q1: Apresente dois pontos negativos e um positivo sobre cada componente.

Q2: Quais são os aspetos gráficos que mudava?

Q3: Que funcionalidade extra adicionava ao programa?

Q4: A experiência de realidade virtual causou-lhe algum incómodo?

Q5: Qual é a componente que mais gostou e a que menos gostou?

9.1. Método de análise de resultados

Para cada questão foram analisadas as várias respostas, destas foram isoladas as mais frequentes e relevantes. Analisando os resultados foram tiradas conclusões e foi delineado um plano de mitigação dos problemas encontrados.

9.2. Resultados

Aspetos positivos apontados: nesta secção serão expostos os aspetos positivos mais relevantes apontados pelas pessoas entrevistadas. Estes aspetos vão ser divididos em subseções, cada uma representando uma componente do programa:

Photo Gallery:

- Interface simples e natural.
- Fundo com foto 360º torna a experiência imersiva.

Music Player:

- Interface simples
- Animações presentes apresentam alguma qualidade

Video Player:

- Efeito da reprodução de vídeo em *V.R.*
- Efeito da reprodução de vídeo GoPro em 170º.

PowerPoint Player:

- Efeito de seleção de PowerPoint
- Experiência imersiva.

Aspetos gráficos a mudar:

Os aspetos gráficos que foram sugeridos para serem mudados foram:

- Ambiente de escolha de música devia ser mais agradável.
- Interface da cena inicial devia ser mais limpa.
- Botão de voltar atrás devia estar mais próximo.
- Ambientes de fundo deviam ter mais haver com a cena atual.
- Aplicação deveria correr a uma resolução superior.

Aspetos da interface a mudar:

- Devia existir uma forma de voltar à cena inicial diretamente em qualquer ponto do programa.
- O ambiente devia ser mais interativo.

Funcionalidades extra a implementar:

Questionei os entrevistados sobre que funcionalidades acham que seria interessante o programa apresentar, e as respostas mais relevantes foram:

- Som no vídeo.
- Fotos e vídeo 360º
- Ambientes com fundo relativo à cena em questão.
- Implementação de um browser.
- Adição de efeitos extra na componente de reprodução de música.

Problemas com motion-sickness

Apenas um entrevistando reportou problemas de *motion-sickness*, no entanto este apresenta vertigens como condição clínica, este facto pode ter contribuído para a experiência reportada.

Componente mais apreciada

As componentes mais apreciadas pelas pessoas entrevistadas foram a componente de PowerPoint e a componente de vídeo.

Componente menos apreciada:

Foi me impossível concluir quais foram as cenas menos apreciadas com esta amostra, para conseguir concluir algo deveria proceder a mais testes.

9.3. Conclusões

Analisando os problemas apresentados posso concluir que a aplicação se encontra com alguma qualidade necessitando, no entanto de alguns ajustes. Foi delineado um plano de como melhorar com a ajuda das críticas recebidas. De notar que o plano não tem em conta um possível desenvolvimento depois do estágio ter sido finalizado, apenas alguns pontos vão ser tratados no ambiente do estágio.

9.4. Plano de mitigação dos problemas encontrados

Analisando os resultados dos testes foi tomado o seguinte "plano de ação":

Realizado no estágio:

- Fixar a imagem no modo de visualização de foto individual.
- Alteração da posição dos botões.
- Criação de um botão extra na galeria de fotos que leve o utilizador diretamente à cena inicial.
- Colocar fotos 360º de fundo com temática da cena atual.

A realizar posteriormente:

- Adição de som ao vídeo.

Capítulo 10
Trabalho futuro

A prova de conceito foi desenvolvida com sucesso. Esta já foi usada nas apresentações do dia aberto da empresa e provavelmente será usada em eventos semelhantes.

Foi discutida a possibilidade de usar a componente de reprodução de PowerPoint para apresentações de produtos.

A continuação do projeto ainda não foi discutida, essa discussão decorrerá num futuro próximo.

Capítulo 11
Conclusões

Analisando o produto final deste estágio bem como o processo de desenvolvimento é seguro dizer que este contribuiu grandemente para o conhecimento da empresa sobre as potencialidades da realidade virtual em ambiente móvel, especialmente quando esta é aplicada na plataforma iOS.

Como resultado do estágio conclui-se que entre as componentes desenvolvidas viu-se bastante potencial na componente de vídeo e na a componente de PowerPoint já que estas são bastante originais e suscitaram interesse tanto internamente como em algumas pessoas que participaram nos testes. Também foi possível concluir que o ecrã do iPhone não possui resolução suficiente para uma experiência satisfatória, sendo por isso uma plataforma a evitar de momento.

Este conhecimento adquirido permite à empresa ter mais informação sobre esta temática, isto vai lhe permitir estar preparada para uma possível entrada da realidade virtual na tecnologia corrente.

Em termos de competências, o estágio foi um grande desafio já que me obrigou a lidar com várias tecnologias ao mesmo tempo, muitas delas que nunca tinha tido contacto e obrigou-me a adquirir sentido estético e algumas capacidades de design já este foi da minha responsabilidade.

Este estágio contribuiu grandemente para o meu desenvolvimento como profissional em termos de aquisição de competências bem como na aquisição de variadas *soft skills* uma vez que me obrigou a integrar num ambiente profissional.

Referências

1. WIT-Software. [Online] http://www.wit-software.com.

2. *Virtual Reality*. [Online] https://en.wikipedia.org/wiki/Virtual_reality.

3. Virtual Reality . [Online] http://www.vrs.org.uk/.

4. Virtual Reality History. [Online] http://www.vrs.org.uk/virtual-reality/history.html.

5. *Oculus Rift*. [Online] https://www.oculus.com/en-us/.

6. Vive, HTC. [Online] http://www.htcvive.com/us/.

7. *PlayStation V.R*. [Online] https://www.playstation.com/en-us/explore/playstation-vr/.

8. *StarVR*. [Online] http://www.starvr.com.

9. Samsung V .R. [Online] http://www.samsung.com/global/galaxy/wearables/gear-vr/.

10. *Google cardboard* . [Online] https://www.google.com/get/cardboard/.

11. Oculus Rift SDK. [Online] https://developer.oculus.com/.

12. Cardboard SDK iOS. [Online] https://github.com/rsanchezsaez/CardboardSDK-iOS.

13. Cardboard SDK Unity. [Online] https://developers.google.com/vr/unity/.

14. *Roller Coaster V.R*. [Online]
https://play.google.com/store/apps/details?id=com.fibrum.roallercoastervr&hl=pt _PT.

15. V.R One Cinema. [Online] https://itunes.apple.com/pt/app/vr-one- cinema/id945065060?mt=8.

16. A Time in Space V.R. [Online]
https://play.google.com/store/apps/details?id=com.Creanet3d.com&hl=pt_PT.

17. A Chair in a Room. [Online]
https://play.google.com/store/apps/details?id=com.RyanBousfield.AChairInARoom.

18. Sisters. [Online] https://play.google.com/store/apps/details?id=com.otherworld.Sisters&hl=pt_PT.

19. High speed stunt bike V.R . [Online]
https://play.google.com/store/apps/details?id=com.at.highway.stunt.bike.riders.vr.

20. V.R speed stunt race. [Online] https://play.google.com/store/apps/details?id=com.tulip.vr.stunt.race.

21. Nighttime Terror. [Online] http://nighttimeterror.com.

22. Hint Mint V.R. [Online]
https://play.google.com/store/apps/details?id=com.Reverge.Hintmint&hl=pt_PT.

23. Blockbuster V.R. [Online]
https://play.google.com/store/apps/details?id=com.fibrum.blockbustervr&hl=pt_ PT.

24. V.R Mac Pan . [Online] https://play.google.com/store/apps/details?id=com.VRcade.MacPan&hl=pt_PT.

25. VRFly. [Online] https://play.google.com/store/apps/details?id=com.JanRaacke.VRFly.

26. V.R asteroids. [Online]
https://play.google.com/store/apps/details?id=com.oriongames.spaceshooter&hl= pt_PT.

27. Agile. [Online] http://agilemethodology.org.

28. Project Planning. [Online] http://www.apastyle.org/learn/quick-guide-on-references.aspx - Websites.

29. User Stories. [Online] https://www .mountaingoatsoftware.com/agile/user-stories.

30. Non-funtional Requeriments. [Online] https://en.wikipedia.org/wiki/Non-functional_requirement.

31. Software Risks. [Online] http://www .itproportal.com/2010/06/14/top-ten-software-development-risks/.

32. Risk Management. [Online] https://en.wikipedia.org/wiki/Risk_management.

33. User interface. [Online] http://www.usability.gov/what-and-why/user-interface-design.html.

34. User interface experience. [Online] https://www .nngroup.com/articles/definition-user-experience/.

35. UI vs UX. [Online] http://www .fastcodesign.com/3032719/ui-ux-who-does-what-a-designers-guide-to-the-tech-industry.

36. User interface design. [Online] https://en.wikipedia.org/wiki/User_interface_design.

37. How to design a good UI. [Online] http://www .goodui.org/.

38. Aaftab Munshi, Dan Ginsburg, Dave Shreiner. *OpenGL® ES 2.0 Programming Guide.* s.l. : Addison-Wesley Professional, 2009.

39. Neuburg, Matt. *iOS 7 Programming Fundamentals.* s.l. : O'Reilly Media, 2013.

40. C++ Inheritance. [Online] http://www.cplusplus.com/doc/tutorial/inheritance/.

41. LibAV . [Online] https://libav.org/.

42. OOXML. [Online] http://officeopenxml.com/.

43. XPATH. [Online] http://www.w3schools.com/xsl/xpath_intro.asp.

44. Introduction to parallel computing. [Online] https://computing.llnl.gov/tutorials/parallel_comp/.

45. Heuristic Evaluation. [Online] https://en.wikipedia.org/wiki/Heuristic_evaluation.