UNIVERSITAT DE BARCELONA

Treball final de grau

GRAU D'ENGINYERIA INFORMÀTICA

Facultat de Matemàtiques i Informàtica

Universitat de Barcelona

Estudi i Disseny d'Interaccions d'un Joc Seriós en Realitat Virtual

Autor: Astor Prieto DehghanPour

Directora: Dra. Inmaculada Rodríguez Santiago

Realitzat a: Departament de Matemàtiques i Informàtica

Barcelona, June 27, 2018

Abstract

Fracsland is an educative game (developed in the University of Barcelona) for children in Primary Education. It is based in teaching basic notions on Fractions and how to operate with them. Making use of Unity3D's game engine, the game leads the player through different zones of the Fracsland island in order to complete missions using different objects or fractions.

The main idea of this project is to make a port of the *Fracsland* game that uses the strong points and the newest HCI schemes that Virtual Reality (or VR from now on) introduces into the interactive media and interactive arts (we use the Oculus Rift platform). Then, we try to re-imagine the different elements found in a traditional 'Flat-Screen' game (UI Experience, Locomotion systems...) and bring them into VR, emphasizing in the extra inmersiveness (also called 'presence' in the VR circles) of having your hands and head tracked in a 3D environment.

This project also explores the possibilities of including Conversational Agents (implemented with Machine-Learning services) into the VR game. These agents are developed in another final year project and offer the player a more natural and familiar interface when interacting with the different elements of the scenes. Thus, we explore the effectiveness of having *Natural Language* (from now on NL) conversations within the game.

Resum

Fracsland és un joc de caràcter educatiu (desenvolupat a la UB) per a nens del Cicle Superior de Primària. El joc intenta ensenyar les nocions bàsiques i conceptes rellevants a l'hora d'estudiar Fraccions i operacions amb Fraccions. Fent ús del motor Unity3D, el joc condueix al jugador per diferents zones de l'illa de Fracsland per tal resoldre missions fen ús d'objectes i fraccions trobades arreu.

La principal idea d'aquest projecte és fer un 'Port' del joc Fracsland que intenti aprofitar les bondats i els nous esquemes d'interacció entre humà-maquina en Realitat Virtual (en concret, nosaltres usem Oculus Rift). Usant la immersió afegida o 'presencia' que ens dóna el fet de poder tenir tant les mans com el cap localitzats en el món real, intentem reimaginar com haurien de ser els diferents elements que podem trobar en un videojoc tradicional (*Flat-Screen Gaming*) per a acomodar-se a una experiència fluida i plaent.

Per altra banda, aquest projecte també explora les opcions d'incloure Agents Conversacionals al joc Fracsland (implementats fent ús de *Machine-Learning Agents*). Aquests agents haurien d'oferir una interfície de control més natural i còmode per a l'usuari en interactuar amb diferents elements del joc. D'aquesta forma explorem l'efectivitat d'usar conversacions en llenguatge natural al nostre projecte.

Contents

List of Figures

1 Introducció

Aquest projecte se situa a l'àmbit tecnològic de la *Realitat Virtual* (o per abreviar, VR). Mitjançant l'ús d'aquesta nova tecnologia es pretén estudiar i dissenyar com han de ser les interaccions amb els jocs en VR per a aconseguir una major immersió i compromís per part dels jugadors.

Aquesta tecnologia va sorgir en 2012 gràcies als avanços fets en el camp de les targetes gràfiques en l'última dècada. Van veure la llum una nova onada de productes i tecnologies relacionades amb la realitat virtual per a ordinadors equipats amb prou potència gràfica. Palmer Luckey va fundar *Oculus VR* el Juny del 2012 amb una campanya de *Kickstarter* i dos anys després s'hi van sumar HTC en conjunt amb Valve (empresa propietària de la plataforma *SteamVR*) mostrant el seu *HTC Vive* (veure *Figure 1*).

Figure 1: Oculus Rift (a la esquerra) i HTC Vive (a la dreta).

Fruit d'aquesta barreja de l'interès del públic i la competitivitat entre Oculus (comprada poc després de 2015 per Facebook) i SteamVR (la plataforma que dóna suport al visor HTC Vive), l'any 2016 van sortir al mercat massiu els seus productes *Consumer Ready Rift* i *Vive* per a usar directament en qualsevol ordinador que fes ús d'una targeta gràfica amb prou potència (*mínim recomanat pels fabricants dels visors: nVidia GTX 970 o superior*) i tingui el sistema operatiu *Windows 10*.

Un sistema actual de VR consta d'un seguit de dispositius que treballen en conjunt per a poder situar a l'espai físic un visor de VR (també conegut com a HMD o *Head-Mounted Display*). Per a poder fer això, comptaran amb un sistema de *tracking* [1]i uns dispositius per a veure i controlar les experiències de joc [2] sempre connectats a un PC que farà tot el treball computacional. Els visors de Oculus i SteamVR, tot i que tenir diferències tècniques en l'ús de tecnologies de *tracking* [3], aconsegueixen un resultat molt semblant que resulta en el que s'anomena "presencia". Aquest terme es refereix al fenomen que succeeix quan, mitjançant un sistema de VR, aconseguim enganyar als nostres sentits fent sentir que estem immersos en l'experiència que es presenta al davant nostre.

Concretament, aquest projecte anomenat *FracslandVR* és, per una banda, un projecte de caràcter educatiu que es basa en un projecte anterior [1] que tenia com a objectiu

[1]Sistema de càmeres o sensors que situen a l'espai físic els visors i comandaments de VR.

[2]Normalment són el mateix visor o HMD i dos comandaments extres, un per a cada ma.

[3]*Rift* usa càmeres infraroges que localitzen uns leds situats al visor i als comandaments, mentre que *Vive* usa uns sensors que projecten una matriu infraroja que és "llegida" pel visor i els comandaments per a posicionar-los

ajudar als alumnes del Cicle Superior de Primària consolidar el concepte de Fracció per a poder relacionar-lo amb facilitat amb coses del món o la vida quotidiana [1]. Per altra banda, *FracslandVR* és un projecte que vol aconseguir integrar les noves tecnologies de realitat virtual introduïdes aquests últims dos anys per a investigar el nou paradigma d'interacció que genera estar dins l'experiència d'un joc. Usem la plataforma d'Oculus per al seu visor *Rift* al motor de joc Unity a Windows 10. En aquesta integració es busca aconseguir una experiència de joc més rica i immersiva per a aconseguir un major compromís per part dels usuaris amb els jocs seriosos i de caràcter educatiu.

Figure 2: Imatge de la escena principal de *FracslandVR*, *Fracstown*.

El joc, com a la seva versió original, presenta al jugador com a un nàufrag que arriba en vaixell a una illa anomenada Fracsland (veure *Figure 2*) on, amb l'ajut de diferents objectes i fraccions que trobaran arreu, hauran de resoldre una sèrie de tasques per a trobar les peces que ha de reparar per a sortir de l'illa. Aquestes tasques les obtindran del governador de l'illa, en *Lord Barus*. En aquestes tasques, el jugador haurà de derrotar enemics, reconstruir ponts o fer que les plantes creixin més ràpid a canvi de les noves peces del seu vaixell. A diferència de l'illa original, en la versió de VR el jugador és immers en aquesta, disposant de les seves mans (representades pels controls Oculus Touch) com a eina per a comandar a l'heroi en les seves missions per a reparar el seu vaixell. O si ho prefereix, el jugador pot entrar dins el cos de l'heroi i desplaçar-se en primera persona dins el món de Fracsland. El jugador trobarà a les seves mans totes les eines que necessita per a comprar nous articles de la tenda del venedor del joc (amb l'ajut de la veu), interactuar amb els diferents personatges que es presentin, desplaçar-se pel món de Fracsland (veure *Figure 3*) o resoldre les diferents tasques encomanades pel governador. Per altra banda, també comptarà amb l'ajut d'un amic mussol que el guiarà en el cas que el jugador es trobi perdut a l'illa (també amb l'ajut de veu).

Figure 3: Imatges de les diferents localitzacions de *FracslandVR*.

FracslandVR és un projecte amb finalitats educatives que sorgeix de l'oportunitat que ofereixen les noves tecnologies de la VR per a donar un entorn més immersiu i empàtic per a que els alumnes consolidin conceptes matemàtics apresos a classe.

El projecte de *FracslandVR*, com ja s'ha esmentat, intenta servir com a complement a l'educació en fraccions, ajudant a consolidar el concepte mentre estan immersos en una experiència immersiva. En concret, aquest projecte es centra en l'àmbit d'estudi de HCI (*Human-Computer Interaction*) i en concret centrat en les interaccions amb sistemes de VR. Per altra banda, comprèn tot el camp de desenvolupament de videojocs (coneixements matemàtics i físics per a representar un món pseudo-realista, coneixements de programació i disseny de software, coneixements de gràfics per a poder donar una experiència fluida i de bon rendiment, etc). Però requereix molt compte a l'hora de seguir les pautes i bases del desenvolupament de jocs tradicionals, ja que molts mecanismes estudiats i aplicats als jocs de pantalla plana no serveixen en el paradigma de disseny d'experiències en VR (limitacions dels visors, nou sistema d'interacció, problema de la locomoció, etc).

Paral·lelament a aquest projecte, s'ha treballat en un projecte per a incloure Agents Conversacionals a la versió de VR del joc *Fracsland* [12]. En la inclusió d'aquests agents en el joc ens endinsem en terreny del ML (*Machine Learning*) per a poder oferir nous sistemes d'interacció que imitin a la interacció humana natural dins un sistema de VR amb el servei ofert per Google *DialogFlow*. Per a poder incloure un agent al nostre projecte que pugui mantenir una conversa mínima per a un propòsit específic, necessitem també coneixements web i de xarxes per a comunicar-nos correctament amb el servei DialogFlow.

1.1 Conceptes Relacionats

En aquesta secció trobarem els principals termes a explicar o definir per a saber a què ens referim nosaltres en usar-los en el nostre projecte. També s'explicaran alguns termes específics relacionats amb tecnologies de VR i *Game Design*.

- **Virtual Reality (o VR)**: si seguim les definicions ja fetes als anys vuitanta sobre la realitat virtual, podem dir que nosaltres ens cenyirem a la mes *device-driven*[4][2]. La realitat virtual són un conjunt de tecnologies i dispositius que permeten reproduir experiències immersives o jocs a un ordinador modern i potent. Això inclou el visor de VR, qualsevol interfície de control extra (Oculus *Touch* o Vive *Wands* com a comandaments) i els dispositius necessaris per a posicionar-lo a l'espai físic i connectar-lo amb el PC i fins i tot la targeta gràfica d'aquest.

- **Virtual Reality Experience (o VRX)**: quan parlem de productes de *software* a la VR, usar el terme "videojoc" molts cops genera unes expectatives en relació a com es mostraran aquests que poden no ser coherents amb la seva experiència real. Per a aquest motiu, la comunitat de desenvolupadors de VR i els fabricants de visors han intentat popularitzat el terme "experiència immersiva" per a parlar de qualsevol joc o eina que faci ús d'aquesta tecnologia en comptes dels coneguts "videojoc" o "aplicació". En el cas del nostre projecte, parlar d'experiència o videojoc és el mateix.

[4]Amb això ens referim que la nostra definició es basa en els components i dispositius tecnològics per a usar-la primàriament.

- **VR *Setup***: conjunt de dispositius (HMD, comandaments i sistema de *tracking*) necessaris per a usar experiències de VR. En el nostre cas, usem el terme per a referir-nos als diferents sistemes o productes que trobem al mercat actualment. Un usuari de VR pot tenir un *setup* amb un visor Vive, els seus comandaments *Wands* (o qualssevol suportats per l'estàndard *OpenVR*) i els seus sensors de projecció infraroja, mentre que un altre (com nosaltres) pot comptar amb el visor *Rift*, els comandaments d'Oculus *Touch* i les seves càmeres pel *tracking*.

- **Head Mounted Display (o HMD)**: és una expressió anglesa per a parlar dels visors de VR. En el nostre cas, es refereix exclusivament al dispositiu visor amb una pantalla per ull de VR.

- ***Tracking* System**: sistema de dispositius, tecnologies i algorismes que permeten a un sistema de VR ser posicionat a l'espai físic i virtual. En les tecnologies actuals tots els sistemes de *tracking* amb els que comptem són *outside-in* i només uns pocs visors nous incorporen *inside-out tracking*[5]. En el nostre cas, ens referim exclusivament al sistema de dispositius de *tracking* amb el que comptarem, dos càmeres infraroges d'Oculus posicionades mirant cap endavant (*Front-Facing Setup*).

- **Software Developement Kit (o SDK)**: al nostre projecte, conjunt de *scripts* que usem al nostre motor de joc per a integrar totes les funcions ofertes per Oculus en el seu *plugin*.

- **Tassa de Refresc (FPS o *Framerate*)**: nombre de *frames* a mostrar en un segon en un *render* gràfic. Per a la VR es recomana als desenvolupadors d'experiències apuntar als 90 FPS per a aconseguir la màxima fluïdesa i immersió possibles.

- **Asynchronous Time Warp (o ATW)**: és una nova tecnologia desenvolupada per NVIDIA i Oculus per a targetes gràfiques d'ordinador. A grans trets, les targetes gràfiques, a l'hora de renderitzar, generen uns *frames* intermedis per poder mostrar algun *output* als visors en cas de no arribar a temps de generar el següent *frame*. Aquesta tècnica però, només funciona sobre el *viewport* que es veu al visor, no guarda cap informació posicional del *frame*. Per exemple, si la targeta gràfica salta molts *frames* en un *span* curt de temps, saltaria el ATW i tot i girar el cap estaríem fixats en una imatge al visor [5].

- **Asynchronous Space Warp (o ASW)**: aquesta és una nova tecnologia desenvolupada també per NVIDIA i Oculus que va un pas més enllà i, usant el ATW, genera una capa de *frames* intermedis que generen un "*frame* escena" que guarda informació posicional i rotacional del visor. Com a resultat, salts en el *framerate* es transformen en quasi imperceptibles dins del visor[7].

- **User Interface (o UI)**: per nosaltres, són tots aquells elements d'un producte de *software* que donen informació ordenada de forma gràfica (o no) al seu usuari. En els videojocs en concret, aquests elements han d'intentar integrar-se a la narrativa de joc per a poder ser més immersius, i si són per VR encara mes. N'hi ha de quatre tipus segons la seva naturalesa al món del joc (veure *Figure 4*). En el nostre

[5]Els sistemes *outside-in* com el nostre necessiten de dispositius que serveixin de marc de referència per a posicionar i entendre el moviment de l'usuari amb el visor de VR. Els nous sistemes *inside-out* als visors de VR usen algorismes i càmeres per posicionar-se sol, sense dispositius de referència externs.

cas, ens referim a tots els elements de *Fracsland* que donen informació i permeten al jugador entendre l'experiència de joc.

- **World Space**: en termes de UI, aquest terme es refereix als elements que es posicionen en l'espai virtual del joc i per tant existeixen a la geometria d'aquest.

- **Screen Space**: en termes de UI, aquest terme es refereix als elements que existeixen únicament a la càmera del jugador, i que per tant es veuen fixats al *viewport* d'aquesta.

- **UI no-Diegètica**: aquest tipus de UI és el més comú als videojocs tradicionals. Són elements fixats al *viewport* de la càmera de joc (com els HUD) que no existeixen a l'espai virtual del joc ni tenen perquè ser integrats amb la narrativa d'aquest. Un exemple en el nostre cas és el minimapa i la vista de vida i nivell de *Fracsland*, que només existeixen a les cantonades superiors de la càmera i no a les escenes de joc.

- **UI Espacial**: aquest tipus d'elements de UI existeixen a l'espai virtual però no tenen perquè estar integrats o tenir motiu de ser a la narrativa del joc. Per a donar un exemple, a la última versió de *FracslandVR*, la tenda de l'illa es considera espacial, ja que és una interfície plana que no usa la ficció del joc per a mostrar els objectes a la taula del venedor.

- **UI Meta**: tots aquells elements de UI que tenen sentit i s'integren en la narrativa de joc però que només existeixen a l'espai de càmera. Com a millor exemple, la sang a la visió de la càmera als jocs bèl·lics per a representar la salut del jugador.

- **UI Diegètica**: aquests tipus d'elements de UI existeixen a l'espai virtual de joc i són integrats a la narrativa d'aquest. Són objectes de UI amb posició al món virtual i amb motiu de ser en aquest. Pot ser un exemple els sistemes de visió nocturna d'un joc bèl·lic, que mostra les siluetes dels personatges a l'espai de joc.

- **Canvas**: objecte o classe de Unity emprada al nostre projecte que ens permet mostrar elements de UI tradicionals gràfics. Pot ser *World Space* o *Screen Space* segons es vulgui un UI espacial i diegètic o no-diegètic i tradicional.

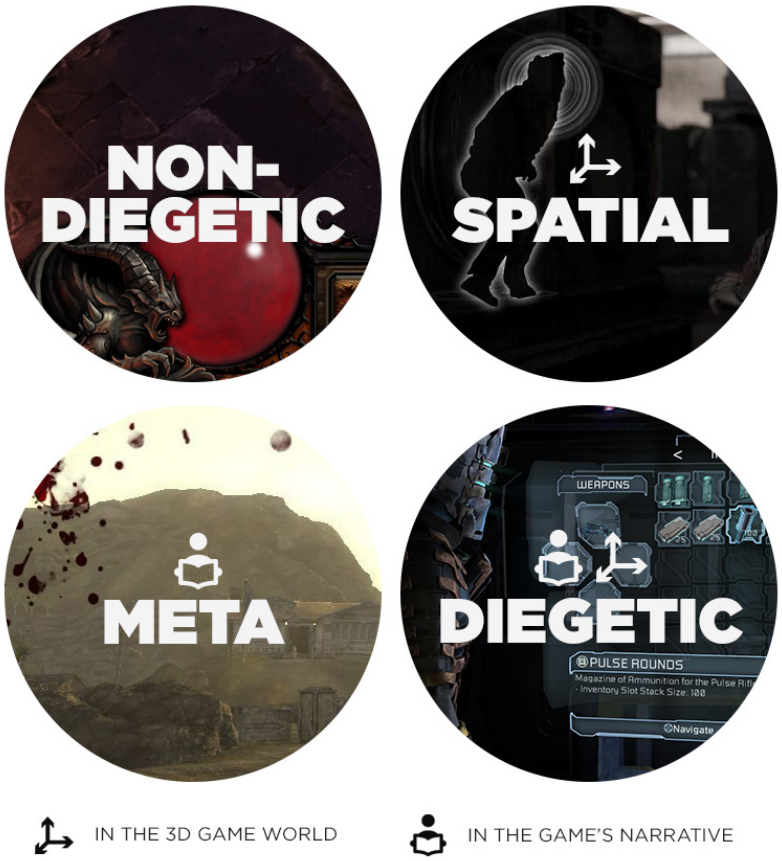

Figure 4: Diferents tipus de UI segons la seva natura al mon de joc, segons *Anthony Stonehouse*. [3]

2 Objectius

L'objectiu principal d'aquest projecte és estudiar la portabilitat d'un joc tradicional a la VR. Com a cas d'estudi, nosaltres agafem el joc seriós de fraccions *Fracsland* i el portem a la realitat virtual, tot fent un seguit d'adaptacions i modificacions que permetin gaudir de la mateixa experiència que ja oferia però amb totes les virtuts que aporta aquesta nova tecnologia.

Aquest projecte s'ha desenvolupat en paral·lel amb un altre localitzat en l'estudi d'agents conversacionals en jocs VR. Mentre que aquest projecte desenvolupa la versió base de Fracsland en VR, l'altre projecte genera un seguit d'agents conversacionals que s'integren al joc per a oferir alternatives als problemes que sorgeixen a la VR en usar elements tradicionals de *Game Design*.

Ja que el meu company és l'autor dels agents conversacionals (fent ús del servei en línia DialogFlow de Google), en aquesta secció només es mencionen els objectius principals desglossats a afrontar en la generació d'una portabilitat per a VR d'un joc seriós (FracslandVR) així com els objectius que sorgeixen d'integrar aquests agents a un joc VR en Unity [12].

2.1 "Portejar" Fracsland a VR

A l'hora de portar un joc tradicional d'ordinador com Fracsland (fet amb Unity) a la realitat virtual, ens sorgeixen una sèrie de problemes o dificultats a afrontar per a que les mecàniques de funcionament del joc original siguin cohesives amb la nova interfície de control que ofereixen tant el visor de realitat virtual com els controladors de les mans (Oculus Rift + Oculus Touch). Com podem adaptar els sistemes d'interacció basats en UI (*User Interface*) no-diegètica (és a dir, vinculada a la visió d'una càmera) per a que el jugador pugui tenir una bona experiència d'usuari? Com podem canviar els sistemes de moviment del jugador per a que siguin immersius i coherents amb la resta de mecàniques de l'experiència de joc?

Es plantegen els següents objectius específics partint de les qüestions observades:

- **Introducció de la VR a Unity:**

  Realitzar els passos mínims necessaris per a que el motor de joc Unity pugui interpretar correctament el visor de realitat virtual d'Oculus. Això vol dir integrar el SDK d'Oculus per a que Unity pugui oferir-nos informació posicional i rotacional del sistema Rift i Touch i pugui accedir a la plataforma d'Oculus des d'un projecte qualsevol. Un cop preparat l'entorn, podem usar totes les funcionalitats ofertes per aquest SDK i començar a visualitzar el projecte original a la VR. Posteriorment, s'afegeixen els elements claus i indispensables per a la VR al joc original de Fracsland com ho són el nou UI o els diferents sistemes de locomoció.

- **Interfície d'Usuari en VR:**

  Canviar els sistemes d'interacció per UI clàssica del joc original Fracsland. Això inclou eliminar o redissenyar els elements tradicionals del UI del joc com poden ser elements contextuals de UI no-diegètics, HUD o sistema d'interacció per ratolí per a que funcionin en un entorn de joc que compta amb un usuari que està dins l'espai virtual. Es plantejaran nous mètodes d'interacció i de presentació de la informació del joc en interfícies situades a l'espai virtual que tenen en compte al jugador dins

d'aquest espai i els punts de *tracking* amb els que compta el sistema Oculus. És a dir, intentarem usar aquests punts (com les mans) que sempre es troben disponibles a un sistema de VR per a vincular informació (que es troba normalment a un HUD) que és sempre rellevant en un joc.

- **Locomoció i punt de vista en VR:**

  Canviar la forma en la qual el joc es presenta a l'usuari. Presentarem diferents formes d'entendre aquest nou punt de vista ofert per la VR respecte als conceptes tradicionals *First-Person* o *Third-Person* (veure *Figure 5*), quins tipus de control alternatius es poden oferir a més de l'obvi "usuari és l'heroi" (*First-Person*). Donarem una sèrie de solucions al problema del desplaçament dins la VR en funció del "punt de vista" en el que es trobi l'usuari, és a dir mecanismes que permetin a l'usuari moure's lliurement per l'espai virtual de joc sense haver de trencar immersió a causa de limitacions de l'espai físic real.

Figure 5: Exemples de primera persona (esquerra) i tercera persona (dreta) al joc *GTA V* de *Rockstar Games*.

2.2 Integració de Agents Conversacionals

Un cop oferta la interacció "equivalent" a la que tindria el joc original, aquí es plantegen objectius relatius a quines funcions es poden fer encara més naturals a nivell interacció amb l'usuari fent ús d'agents conversacionals que entenguin una conversació bàsica amb un humà.

- **Facilitar la integració dels Agents a FracslandVR:**

  Intentarem oferir una forma fàcil d'integrar aquests agents al joc en VR tot fent ús de tècniques de disseny de *software*. A l'afegir els components necessaris del projecte dels Agents Conversacionals volem que no hi hagin conflictes entre els *scripts* del SDK d'Oculus o el de DialogFlow.

3 Treballs Relacionats

En el disseny i implementació d'experiències en VR, i a causa de la novetat d'aquesta tecnologia implicada en els HMD's, trobem que no hi ha una sèrie de pautes o patrons de disseny basades en HCI per a poder interactuar de forma sòlida amb aquestes. Actualment, com a desenvolupadors, no comptem ni amb eines estandarditzades per a poder desenvolupar per a tot el mercat del VR [6], ni amb una massa crítica d'usuaris o desenvolupadors per a poder comptar amb un flux de *feedback* prou bo a l'hora d'enfocar el disseny o solució concreta dels jocs en VR.

Per tant, ens hem de centrar primàriament en jocs i eines fetes per altres desenvolupadors que ja s'han topat amb aquests problemes per a poder adaptar sistemes i mecàniques tradicionals en els videojocs, com per exemple interacció amb una interfície tipus HUD (UI no diegètica, veure *Figure 6*), o crear nous sistemes d'interacció dissenyats per a VR, com per exemple el problema de la locomoció en un espai virtual.

Figure 6: Captura d'un joc *Flat-Screen* o tradicional que fa us de HUD.

En aquesta secció, veurem diferents productes (jocs i eines serioses) que ja estan al mercat i que ens han ajudat a inspirar-nos a l'hora de trobar solucions usades en FracslandVR; poden ser solucions prou creatives, és a dir, que usen d'un nou paradigma de disseny o bé solucions prou robustes, és a dir, que "tradueixen" prou bé funcionalitats clàssiques d'una eina o joc d'ordinador. Ens referim a solucions que tot no ser pensades des de zero per a VR, implementen molt bé interaccions amb elements coneguts tradicionalment, com poden ser teclats virtuals. A la VR, construïm un món virtual on l'espai virtual es mesura com al món real [7]. Hi interactuarem de la forma més familiar possible, usant les mans, usant gestos, etc.

[6] Actualment, cada visor de realitat virtual té el seu propi SDK i no tots els visors estan integrats a tots els motors gràfics moderns

[7] Veure *Figure 7*, diferents espais de *tracking* maxims admesos per als principals sistemes de VR.

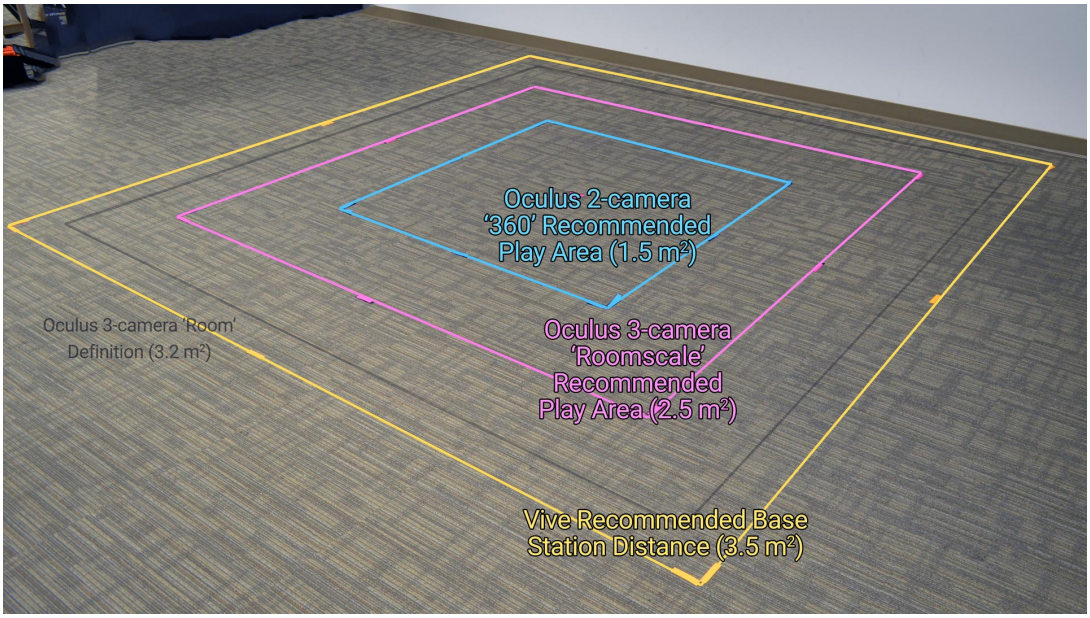

Figure 7: Fotografia que mostra els diferents espais màxims admesos pels diferents sistemes de VR.

3.1 *State-of-the-Art* en jocs VR

Aquí trobem els primers jocs o experiències que s'han fet lloc al mercat d'Oculus i SteamVR. Són jocs de diferents gèneres i diferents desenvolupadors que comparteixen filosofia de disseny i aporten un seguit de pràctiques interessants a la VR. Alguns d'aquests jocs probablement es convertiran en les bases de com implementar interaccions en futurs productes de VR.

3.1.1 Brass Tactics

Brass Tactics és un joc d'estratègia en temps real (*Real-Time Strategy*) desenvolupat per *Hidden Path Entertainment*, creador del clàssic *Age Of Empires* per a PC. Compatible amb Oculus Rift i amb HTC Vive (amb un port no-oficial), sempre fent ús dels controls per a les mans (els *Touch* en cas de l'Oculus Rift, les *Wands* en el cas de HTC Vive). Podríem considerar que *Brass Tactics* és el successor a la realitat virtual dels RTS clàssics d'ordinador.

L'experiència es basa en partides d'entre dos o tres usuaris (o Reis) que lluiten per conquerir el terreny de joc. Al joc, el terreny és presentat al jugador com a un taulell semblant al *Risk* de taula pel que fa a mida, proporcions i interacció. Cada rei (o Regne) desplega un seguit de fortificacions que poden generar unitats amb habilitats i característiques diferents per tal d'ocupar espai del regne contrari i així acabar amb el castell del rei o reis enemics. Per a poder construir les fortificacions es fa ús de diferents recursos que genera cada regne amb el temps.

La part interessant i que ens ha inspirat d'aquesta experiència es troba en el seu UI i en el seu sistema de Locomoció. Pràcticament tota la UIX se centra en les mans del jugador (veure *Figure 8*). Trobem al revers de cada mà l'indicador d'or i diamants, les dues monedes de canvi del joc, així com les fortificacions a construir de forma contextual, si girem la mà i l'orientem amb la palma cap amunt.

Figure 8: Imatges d'una partida i del UI de *Brass Tactics*.

Respecte al sistema de locomoció, a l'hora de controlar el taulell de joc observem molt *feedback* visual quan fem ús de les mans per a moure'ns. Podem desplaçar-nos pel terreny amb una mà si la tanquem [8] i controlar l'alçada del jugador respecte la taula si en tanquem les dues [9]. Per altra banda, podem seleccionar una fortificació, unitat o conjunt d'aquests si apuntem amb l'índex i indiquem la nova posició o acció a fer. L'experiència d'usuari amb el UI i amb el taulell són prou polides per a intentar extrapolar-les i usar-les en quasi qualsevol joc de tercera persona en VR.

3.1.2 Echo Arena

Aquest joc explora una barreja d'acció i esports en gravetat zero. Desenvolupat per l'estudi *Ready At Dawn* i publicat per Oculus, només disponible de forma oficial a l'Oculus Store. Aquest joc té influencies i és fortament inspirat pel llibre i la pel·lícula *The Ender's Game*. El joc pren com a base un camp d'un esport "futurista" i enfronta a dos equips de cinc jugadors reals que cooperen i es coordinen per a tenir possessió del disc de joc i fer-lo passar per l'arc de gol enemic (veure *Figure 9*). L'equip amb més gols guanya el partit.

Figure 9: Imatge del camp de joc i els avatars dels jugadors de *Echo Arena*.

Trobem en aquesta experiència una sèrie de decisions de disseny molt bones a l'hora de treballar en VR. Ens hem inspirat en el seu HUD, on podem veure una interfície molt minimalista que indica únicament si has sigut impactat per algun jugador contrari i el temps d'espera per a poder generar una barrera i cobrir-te dels pròxims cops. Com a conseqüència, tenim uns pocs elements molt útils i de freqüent consulta fixats sempre al

[8] Moviment en els eixos X i Z de l'espai virtual (*panning*).
[9] Moviment en els eixos X, Y i Z de l'espai virtual (moviment *6 Degrees Of Freedom*).

nostre camp de visió (quasi no intrusius i no-diegètics) i la resta d'indicadors es troben situats arreu del camp de joc (elements diegètics i espacials), com si és tractes d'un marcador de futbol real. D'aquesta forma, el jugador pot centrar-se en el joc i en la profunditat afegida del VR sense haver de preocupar-se d'estar contínuament pendent d'elements propers a la cara (on el text no es llegeix bé) com hi estaríem tradicionalment en un joc així en *Flat-Screen*.

Presenten una solució molt creativa al problema de la locomoció en realitat virtual. Ja que és un joc basat en gravetat zero, els jugadors tenen la possibilitat d'agafar qualsevol part del terreny de joc per a impulsar-se en la direcció desitjada així com uns petits propulsors als canells per a corregir i canviar la trajectòria. A l'espai físic no desplacem més que els braços realment. Com a resultat, movem l'espai virtual al voltant nostre i no el físic, donant un efecte de gravetat zero molt real i que dóna una sensació d'immersió extrema tot i no moure'ns tal com percebem al joc.

3.1.3 Rec Room

Rec Room és un joc *MMO-RPG-Free-To-Play* (*Massive Multiplayer Online - Role-Play Game*) social desenvolupat per *Against Gravity* on tots els jugadors es troben en un mateix lobby on poden parlar, interactuar i jugar a diferents activitats com si d'un institut en estiu és tractes.

Aquest joc fa un molt bon ús de UI contextuals, on barreja interaccions amb interfícies clàssiques passades a l'espai virtual (pantalles que s'obren a l'aire i pots controlar amb els dits) i elements completament integrats a l'espai (espacials, veure *Figure 10*). Ens ha servit com a inspiració per a trobar bones formes de presentar elements complexos de UI de forma espacial.

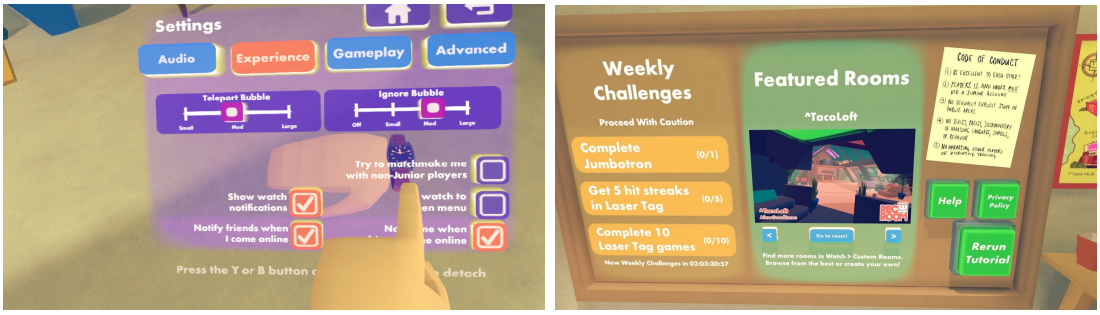

Figure 10: Imatges de diferents elements espacials de UI per a modificar les preferències d'usuari de *Rec Room*.

3.1.4 A Township Tale

Desenvolupat per *AltaVR*, aquest joc és una de les primeres adaptacions d'un MMO-RPG de mon obert a la realitat virtual des de zero. Basat en missions, equipament, lluita, construcció i creació d'eines, aquest joc es basa en l'experiència clàssica del multijugador massiu en línia (semblant al funcionament del *World of Warcraft*) barrejat amb un joc d'exploració i rol amb elements tipus *Minecraft*. El que fa destacar a aquesta experiència és que ha sigut dissenyada des del principi tenint en compte les limitacions de l'actual tecnologia, i implementa en conseqüència un nombre de sistemes i mecàniques bastant innovadores. Tot el seu UI ens pot servir d'inspiració per a poder trobar bones formes de presentar elements de UI diegètics al joc.

Si observem la *Figure 11*, trobem que com a jugadors, tenim el nostre inventari separat en "esferes" al voltant nostre, on cada esfera representa un *slot* o espai d'aquest inventari. Cada espai pot guardar fins a 256 unitats d'un element.

Aquest sistema d'emmagatzematge es pot trobar en diferents bases que el jugador habita; armaris, prestatgeries, taules...

Figure 11: Imatges dels diferents elements diegètics (motxilla, prestatgeria i butxaques) per a guardar objectes de *A Township Tale*.

3.2 Eines i productivitat en VR

En aquí trobarem les eines i experiències més centrades en creació, rendiment i productivitat, així com les aplicacions que et permeten usar el teu ordinador al complet des d'un visor de realitat virtual. Observarem en aquest apartat que en l'àmbit de la productivitat en VR no tenim productes tan ben acabats com podem trobar del costat dels jocs, però que fan un bon intent i troben uns bons compromisos entre usabilitat, immersió i complexitat d'ús.

3.2.1 Virtual Desktop i BigScreen

Virtual Desktop i *BigScreen* són eines bàsiques i més esteses en les principals plataformes de realitat virtual per a fer ús complet de les funcionalitats d'un ordinador des de dins d'un visor de VR. Aquestes eines solen compartir els mateixos principis de funcionament, tot i diferir en petits aspectes de rendiment. *Virtual Desktop* és la millor eina per a controlar un ordinador; et llença en una habitació customitzable per l'usuari a on pots invocar i crear tants escriptoris o pantalles virtuals com el teu equip permeti (per potència).

Mitjançant l'ús de les mans l'usuari és capaç d'alternar control entre diferents pantalles, ja que el ratolí passa a ser un punter que surt del dit índex de la mà dominant, de forma immediata i intuïtiva. Et permet canviar d'aplicacions ràpidament o bé paral·lelitzar-les en diferents finestres virtuals. Per altra banda no trobem encara una bona solució al teclat virtual, ja que fins ara, aquests tipus d'eina no donen millor alternativa que usar un teclat virtual amb dos dits (altament complicat, molt poc rendiment) o bé usar el teu teclat real sense poder-lo veure.

BigScreen, l'alternativa gratis i que funciona de la mateixa manera, afegeix una capa de connectivitat a aquesta sèrie d'eines i permet fins a tres persones (amb els seus respectius "PC's virtuals") en un mateix espai, compartint pantalles, àudio i contingut entre ells (veure *Figure 12*).

Figure 12: Imatges de l'entorn de treball d'escriptori de *Virtual Desktop* (a la esquerra) i una sala amb dos usuaris compartint pantalles i escriptoris dels seus ordinadors a *BigScreen*.

Com a conclusió, aquests productes són unes eines molt potents i poderoses que ofereixen una immersió molt més gran a l'hora d'usar un ordinador com una màquina multimèdia, però que encara no donen una interfície de control prou bona per a substituir les aplicacions de productivitat convencionals. Ens poden servir d'inspiració a l'hora de trobar mètodes intuïtius per a interactuar amb elements de UI espacials molt complexes en el nostre projecte.

3.2.2 Google Earth VR

L'experiència oficial de Google que permet accedir a tot el servei de mapes i modelat del món de Google Earth en realitat virtual.

En aquesta eina trobem una sèrie d'elements nous que s'incorporen a l'esquema de control clàssic que ofereix Earth a la web, com per exemple un globus terraqüi amb una xinxeta que ens permet moure'ns instantàniament entre zones del globus (veure *Figure 13*), a part de les clàssiques manipulacions també disponibles a la versió *Flat-Screen* amb el ratolí o dits en cas d'interfície tàctil (arrossegar per desplaçar-se, dos dits per rotar perspectiva, etc).

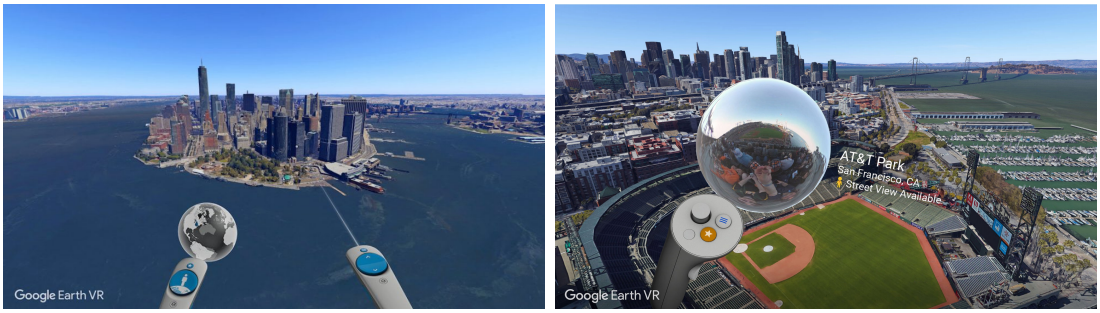

Figure 13: Imatges de *Google Earth VR* amb la UI dels comandaments *Touch*.

També trobem integració amb el servei de *Street View*, que en realitat virtual encara té més sentit, ja que els *mipmaps* empleats pels vehicles capturadors de Google són en 360 graus, cosa que permet posar l'usuari en qualsevol carrer de la terra (que tingui aquests *mipmaps* fets per part de Google).

Aquesta eina ens serveix d'inspiració en la seva UI diegètica mostrada al voltant de les mans. Podem observar un ús molt enginyós d'aquest tipus de UI, que aprofita les posicions de les mans per a poder mostrar informació de forma constant sense que sempre obstrueixi la visió de l'usuari com passaria normalment amb la UI de l'aplicació web.

3.2.3 Medium, Quill i Tilt Brush

En aquest grup de tres experiències, les dues primeres han sigut desenvolupades per propi Oculus i la última per Google. Són diferents eines que permeten esculpir, modelar i dibuixar en tres dimensions fent ús de diferents pinzells i estris per a donar forma i color.

Totes tres usen les mans com a eina multi usos que, combinant elements de UI contextuals per escollir diferents opcions, permet canviar de forma i color al material a afegir o bé permet fer formes i extrudir material existent de la nostra creació, però cadascuna se centra en un tipus d'acabat artístic. *Medium* és l'eina més semblant a un escultor d'argila i permet fer textures molt realistes polint superfícies; Quill se centra més en dibuixar amb pinzells i llapis en un espai tridimensional per a donar una sensació de dibuix viu i finalment *Tilt Brush* és una barreja entre les possibilitats de Quill amb un potent editor d'efectes i partícules (*Figure 14*).

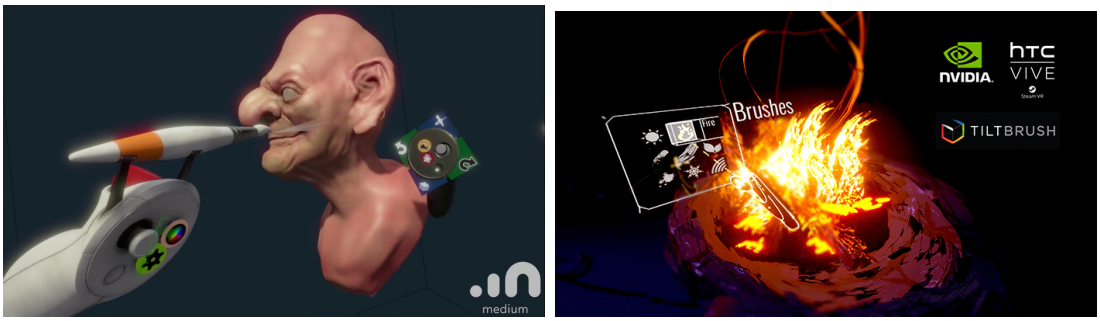

Figure 14: Imatges de les experiències de dibuix *Medium* i *Tilt Brush*.

Aquestes eines ens inspiren en dues coses: per una banda, la seva UI espacial i diegètica vinculada a les mans (semblant a la de *Google Earth*, *Rec Room* o *Brass Tactics*) ens ensenya una bona forma de mostrar informació a l'usuari de forma molt immersiva. Per l'altra, el seu *look-and-feel low-poly* en mostra que usar aquest tipus de "visuals" que són menys costoses de renderitzar pot donar una bona solidesa estètica a qualsevol joc en VR sense requerir molts recursos gràfics.

4 Anàlisi

En el nostre projecte dividim l'anàlisi dels problemes a tractar en tipus de requeriments. Primer, requeriments funcionals, és a dir, coses que s'han de dissenyar i implementar directament al joc perquè funcioni a la VR. Després, requeriments no-funcionals, és a dir, coses a tenir en compte quan dissenyem sistemes per a jocs de VR. Finalment requeriments tecnològics, coses a tenir en compte tecnològicament quan usem sistemes actuals de VR com Oculus Rift.

4.1 Requeriments funcionals

Fracsland, com la majoria de jocs tradicionals d'ordinador per a nens, centra la major part de la seva interacció en el ratolí i un seguit d'elements d'interfície no-diegètics: el HUD, que conté tota la informació amb la qual comptarà el jugador per a avançar en la història (inventari, minimapa, missions...), les interfícies contextuals (la tenda del joc, les missions de *Lord Barus*, la selecció de plantes a la granja...) i les bombolles de text que comuniquin els diferents personatges de l'illa.

El ratolí forma el paper protagonista a l'hora de desplaçar-se i explorar l'entorn. El jugador es desplaçarà fent clic al terreny, seleccionarà els elements del món fent clic en ells o interactuarà amb el HUD (veure *Figure 15*) per obtenir informació fent clic en els respectius elements. Tota interacció és intuïtiva per a coneixedors de la interfície ratolí (actualment, gairebé tothom).

Figure 15: Imatge del HUD original de *Fracsland*.

Aquest paradigma amb ratolí canvia radicalment a la realitat virtual, però si volem mantenir una experiència de joc semblant a l'original i alhora intuïtiva i immersiva haurem de buscar solució a un seguit de problemes que sorgeixen en integrar cap i mans a l'entorn virtual:

- **Locomoció a la VR**: és el primer problema que sorgeix a l'atacar un projecte tradicional en *Third Person*. Fracsland posa al jugador en una càmera externa al personatge que encarna i hi actua com a un "deu" que controla els moviments i accions de l'heroi del joc. Aquest *PoV* (*Point of View*) pot semblar no ser l'opció més intuïtiva, ja que es podria esperar en primera instància que les experiències en VR haurien de ser totes encarnant al protagonista de la història (o *First Person*) per a aconseguir immersió total. Per a un joc com el nostre, en què l'escala a la que s'ha realitzat és petita (els personatges i escenes com a joguines dins el visor),

podria també tenir sentit incloure una mena de traducció d'aquesta *Third Person* del joc original o *God Mode*. Així doncs, contemplarem ambdós *PoV* en el projecte i donarem l'opció de canviar-lo dins del joc, donant un mecanisme de transició entre aquests.

Un cop plantejada la qüestió del *PoV* a l'illa en VR, hem de pensar també en quin serà el sistema amb el qual comptarà l'usuari per a desplaçar-se per l'entorn. Aquest punt és especialment delicat, ja que qualsevol sistema que no sigui suficientment natural i intuïtiu podria provocar marejos i malestar a l'usuari. Interfícies com els *joysticks* per a controlar el moviment a la VR donen un resultat molt pràctic per a moure's per l'espai virtual (potencialment infinit) però són molt més propenses a trencar la il·lusió d'immersió o fins i tot marejar i desorientar, ja que simulen un moviment en un món virtual desconnectat dels moviments reals que l'usuari està fent.

Les solucions més usades en productes de VR són dues: teletransportar l'usuari en primera persona d'un punt a un altre fent ús d'un *fade-out-fade-in* a negre per a evitar moviments simulats, o bé usar el món com a un taulell o objecte per a agafar i "tirar" d'aquest i desplaçar-se. Explorarem ambdues opcions per a determinar quina s'adequaria al cas de Fracsland.

- **Experiència d'Usuari a la VR (UIX)**: com hem dit abans, Fracsland és un joc que centra la seva interacció en elements no-diegètics de UI, primàriament, un HUD.

  Els coneguts HUD's sofreixen d'uns problemes molt concrets a la VR a causa de limitacions de la tecnologia; el text molt petit o les icones molt complexes, si es vinculen com es fa tradicionalment a un HUD, a la perifèria de la vista, serà il·legible o no serà clar perquè els visors de VR renderitzen amb claredat al centre dels panels i baixa la fidelitat als vèrtexs. Aquest recurs, anomenat *Fixed-Foveated Rendering* aprofita el fet que en general mirem endavant i al centre per a estalviar potència gràfica i evitar artefactes visuals (deguts a les lents del visor), però fa amb les densitats de panel dels visors actuals, els HUD hagin de ser minimalistes i senzills o millor encara, inexistents. Aquest problema desapareixerà als anys vinents amb l'arribada de millors algoritmes de *Eye-Tracking* i la introducció del *Eye Gaze Based-Foveated Rendering*, que sempre cercarà on mira l'usuari per a saber quin punt ha de ser mostrat amb més qualitat.

  S'haurà de pensar en un sistema alternatiu al HUD per a mostrar al jugador en tot moment la informació indispensable que ha de conèixer. També s'haurà de pensar com es traduirà a la VR tota UI contextual, és a dir on es posicionaran els elements del joc (com la tenda o el taulell de missions) al mon de *Fracsland* perquè l'usuari pugui seguir-ne fent ús. Aquests canvis generaran un seguit d'elements de UI diegètics o espacials per a que el jugador pugui interactuar de forma natural amb el joc.

4.2 Requeriments no-funcionals

Tot i tenir plantejats els requeriments funcionals a dissenyar i implementar a FracslandVR, encara hem de tenir en compte satisfer les següents condicions per a que la nostra experiència de joc pugui ser fluida. Els paràmetres de rendiment que s'exigeixen

a continuació són marcats pels *guidelines* d'Oculus i són fruit de la seva exploració i investigació sobre immersió i confort a la VR.

- **Bon rendiment / *Framerate* estable**: per a una bona experiència en VR, es recomana que el software permeti un *output* estable de 90 FPS (*Frames per second*, o almenys 60). Qualsevol experiència sota 45 FPS activarà el sistema ASW (*Asynchronous Space Warp*, veure secció 1.1) per a "virtualitzar" i introduir en el *pipeline* gràfic *frames* duplicats per a doblar el *framerate*. Tot i poder usar aquest recurs, se segueix recomanant encaridament apuntar a un rendiment de 90 FPS per a evitar possibles desconforts en els usuaris. És molt important evitar problemes de *performance* per a no trencar mai la immersió a l'experiència de l'usuari.

- **Elevada claredat *in-game***: si aconseguim uns mínims de rendiment estables a la nostra experiència però no aconseguim fer el joc clar i accessible per al nou usuari, podem fer perdre l'interès, esglaiar-lo o fins i tot generar mal de cap. L'excés d'informació textual per exemple, sense fer ús de recursos visuals pot fer que una persona amb un lleuger problema de visió hagi de fer un esforç sobrehumà per a poder enfocar i llegir tota la informació en la seva totalitat. És important aconseguir sistemes que facin ús de recursos molt visuals i autoexplicatius. És molt important donar un clar *feedback* visual de les accions fetes per l'usuari.

4.3 Requeriments tecnològics

En aquest projecte, i en qualsevol que faci ús de la VR, s'han de tenir en compte tots els requeriments o problemes inherents a l'usar aquesta tecnologia:

- **Targetes Gràfiques d'última generació**: en l'ús dels sistemes de VR actuals, sempre hem de comptar amb una connexió cablejada (un port HDMI 1.3 i un port USB 3.0) amb un ordinador bastant potent per a poder renderitzar experiències prou fluides. Per la natura d'aquests visors i com que contenen dues pantalles (una per cadascun dels ulls), l'ordinador responsable de córrer aquestes experiències ha de tenir incorporada una targeta gràfica amb prou potència. Segons els *guidelines* d'Oculus, la targeta ha de donar *output* a les dues pantalles de 1800 x 1200 *píxels per ull* [10] amb una tassa de refresc o *framerate* de 90Hz (90 FPS). Això suposa un increment de cost del 108.34% en el nombre de píxels a renderitzar *per frame* respecte a un videojoc per a una única pantalla de resolució 1080p (2.073.600 píxels vs. 4.320.000 píxels) a més de mantenir estable l'estàndard de 90Hz en comptes dels tradicionals 30 o 60Hz del medi tradicional.

  Fins fa poc aquests costos feien que poques targetes poguessin donar suport a la VR, únicament les targetes de nVidia GTX series 900 (GTX 970 i 980) i series 10 (GTX 1060, 1070, i 1080) oferien compatibilitat completa amb tots els visors disponibles a Windows 10, i algunes poques d'AMD (abans ATI) començaven a oferir suport nadiu (Radeon RX 470 i 480).

  Aquestes targetes incorporen tecnologies desenvolupades (definides al glossari) específicament per al renderitzat de VR com són el ATW o ASW, que ofereixen

[10] Tot i que en aquests tipus de visors de VR té més sentit parlar de PPD (*pixels-per-degree*) o PPI (*pixels-per-inch*) per a descriure la fidelitat visual.

sistemes de recuperació en cas d'error, sistemes de *fallback* i sistemes d'estalvi de rendiment.

Aquest últim any i segons els últims productes disponibles al mercat, les targetes gràfiques que suporten la VR incorporen el segell *VR Ready* i cada cop es poden trobar més fabricants a més dels ja coneguts nVidia o AMD.

Amb els propers avanços en el camp d'aquestes targetes, cada cop serà més accessible aquesta potència de renderització i s'abaratiran els costos del mercat massiu, donant un preu d'entrada a la VR relativament més baix i una tassa d'adopció més alta, ja que molts ordinadors començaran a portar prou potència de sèrie per a fer VR.

- **Visors amb dos panels**: hem de comptar amb la visió estereoscòpica (en 3D) des de l'inici del desenvolupament, ja que en aquests visors comptem amb dos panels i lents independents per als ulls. Per a donar lloc a bons productes per a VR, hem de comptar amb el sentit de la profunditat i l'escala dins les experiències de joc; no podem llençar a l'usuari en una escena immensa i molt complexa sense cap indicació i que no se senti perdut, com tampoc podem donar una escena diminuta i amb moltes explicacions escrites i esperar que l'usuari pugui entendre amb claredat tot el text.

  Actualment, els visors disponibles al mercat estan limitats a una resolució mitjana de 12-15 PPD (l'equivalent a la visió humana en un visor es calcula prop els 60 PPD) i un *Field of View* de 110 graus (el *FoV* humà es calcula sobre els 210 graus) cosa que comporta una sèrie de limitacions tècniques que no poden ser oblidades quan desenvolupem per a la VR. Ja que comptem amb un quart aproximadament de la resolució optima, haurem de fer compromisos a l'hora de presentar elements sobrecarregats o text en UI del joc, ja que podrien veure's amb baixa fidelitat visual al visor.

  Amb les properes generacions de visors de VR, i amb l'augment de la densitat dels panels, probablement podrem situar-nos prop dels 30 PPD i 140 graus de *FoV* i comptar amb molta més fidelitat visual, donant lloc a la possibilitat d'oferir entorns més complexos i interfícies més tradicionals (com escriptoris d'ordinador) dins la VR còmodament.

- **Tecnologies de *tracking* dels visors**: cada visor disponible ofereix una solució semblant però diferent del problema de la localització (o *tracking*) del visor a l'espai en VR. A causa d'això, els desenvolupadors poden escollir oferir diferents tipus d'experiències: les anomenades *Standing* (quiet al lloc, que poden ser pensades per 180 *Front Facing* o 360 graus) compten amb un sistema com el d'Oculus, unes càmeres amb un rang limitat que poden estar al davant de l'usuari, o al davant i al darrera. Les anomenades *Room-Scale* que solen usar de més espai físic i d'un sistema de *tracking* més complet com el de HTC Vive i les bases de projecció infraroja. Un sistema com el d'Oculus requeriria des de 4 càmeres per a oferir una experiència *Room-Scale*.

  L'espai físic amb el qual el desenvolupador compta pot variar molt amb el *setup* del sistema de VR de l'usuari. Pot ser que els nostres usuaris comptin amb diferents sistemes de VR i diferent espai físic disponible i moltes vegades resulta millor

dissenyar per a un espai reduït i *Front Facing* tot i comptar com a desenvolupador amb un sistema *Room-Scale*.

Aquestes tecnologies quedaran pròximament obsoletes, amb la quantitat de treball que s'està duent a terme en el camp dels algorismes de *inside-out tracking*. És a dir, en futures revisions dels visors de VR, s'incorporaran sistemes amb IMUs (*Inertial Measurement Unit*) que permetin al mateix visor i als controls saber on són a l'espai per si mateixos sense fer ús de càmeres o bases externes i donant lloc a possibles espais virtuals il·limitats. Google ja ha mostrat la capacitat i potència dels seus algorismes de SLAM (*Simultaneous Localization And Mapping*) en els seus *smartphones* del projecte *Tango*. Aquesta tecnologia ja s'està implementant en prototips d'Oculus (prototip anomenat *Santa Cruz*).

Per altra banda, en la realització d'aquest projecte haurem de comptar amb accés a *Internet* en tot moment per a poder accedir a la "plataforma en línia" docent que ofereix Fracsland i per a que que Unity pugui comunicar-se amb el servei de conversació de Google *DialogFlow*. Així el joc podrà interpretar les frases dites pel jugador, enviar-les al servidor on es troba l'agent conversacional i que aquest enviï la seva resposta processada, que serà "llegida" pel servei de veu de Windows.

Els *specs* o requeriments mínims que ha de tenir un equip amb **Windows 10** per a córrer la VR i en concret el projecte *FracslandVR* queden doncs així (*Figure 16*):

Chipset	**Graphics Card**	**RAM**	**Drive**	**Oculus Ver.**
i5-4590 o superior	NVIDIA GTX 970/1060	8GB/16GB	3GB	v1.24

Figure 16: Especificacions mínimes per a un PC que hagi de córrer *FracslandVR*.

4.4 Comandament d'Oculus *Touch*

En el nostre projecte usem el sistema de VR d'Oculus, inclosos els comandaments *Touch*, que ens serveixen per a tenir les mans *tracked* a l'espai. Aquests comandaments però, també ens ofereixen una interfície amb botons i superfícies tàctils molt semblants als d'un comandament de consola tradicional (com el de *Microsoft* amb la seva XBOX One, veure *Figure 17*) que ens permet fer comparacions per a conèixer millor les possibilitats de *Touch*. En aquesta secció comentarem la terminologia dels comandaments de consola (XBOX) tradicionals per tal de fer paral·lelismes i familiaritzar-nos amb els nostres:

Figure 17: Imatge del comandament de la consola XBOX One de *Microsoft*, inclos amb el visor *Rift*.

En aquests comandaments tradicionals, i per tal de poder trobar parts comunes amb els *Touch*, trobem tres seccions ben diferenciades:

- ***Triggers***: són polsadors analògics que normalment s'utilitzen per a fer accions molt repetitives (com disparar a un joc *Shooter*). En aquí solem trobar els *triggers* `LB` (o `L1` a *PlayStation*), `RB` (o `R1`), `LT` (o `L2`) i `RT` (o `R2`).

  Als *Touch*, i de forma similar als botons i als *joysticks*, aquests es divideixen dos a cada comandament, els podem veure al punt 1 de la *Figure 18*. En aquests *triggers* a VR se solen vincular les funcions d'agafar (*gripper*) i acció o disparar (*trigger*).

- **Botons**: on normalment trobem els polsadors anomenats `A` (o `Creu`, en el cas de *PlayStation*), `B` (o `Cercle`), `X` (o `Quadrat`) i `Y` (o `Triangle`). Aquests botons usualment donen accés a funcionalitats d'ús immediat (com obrir menús, o activar i seleccionar elements al joc).

  En els *Touch* aquests botons es divideixen en dues seccions separades (dos botons en cada comandament, veure punt 2 de la *Figure 18*).

- ***Joysticks***: finalment trobem dues palanques analògiques que usualment s'usen per a controlar desplaçament i rotació de la càmera del jugador. Al comandament de *Microsoft* normalment parlem de `L` *joystick* i `R` *joystick* i inclouen un polsador extra prement-los (coneguts com a `L3` i `R3`).

  Als *Touch* aquests elements solen oferir una forma alternativa de locomoció per als jugadors avançats de VR (moviment totalment artificial) o bé s'utilitzen per a manipular punters en elements de UI. Els podem veure a la part 3 de la *Figure 18*.

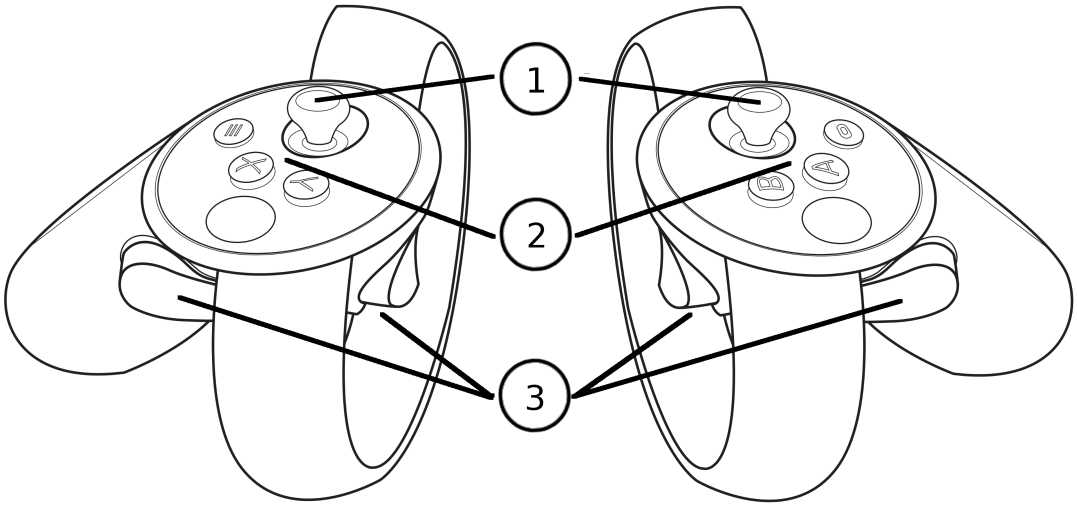

Figure 18: Imatge dels comandaments d'Oculus *Touch*, amb les seves parts diferenciades.

Per acabar, cal comentar que al contrari que els comandaments de consola normals, i per a donar una major sensació d'immersió, els comandaments usats al nostre sistema de VR també ofereixen una superfície tàctil a cada element usant un seguit de sensors capacitius. Això vol dir que cada botó, *trigger* o *joystick* podran saber si esta sent "polsat" (*touched*), "tocat" (*pressed*) o "sense contacte" (*released*) segons la proximitat de les mans de l'usuari a aquests.

4.5 Diagrama de Casos d'Us

En aquest diagrama de casos d'ús (*Figure 19*), trobem tant els casos originals plantejats pel seu autor com els nous (en groc al diagrama) que han sigut afegits per a acomodar l'experiència en VR.

El nostre treball, al ser un *port* del projecte original *Fracsland*, recaurà en modificar els casos d'us originals per a adaptar-los a la VR sense canviar l'essència d'aquests. Observant el UCD (*Use Case Diagram*) del projecte original trobem els següents grans grups sota els quals podem classificar els casos d'ús:

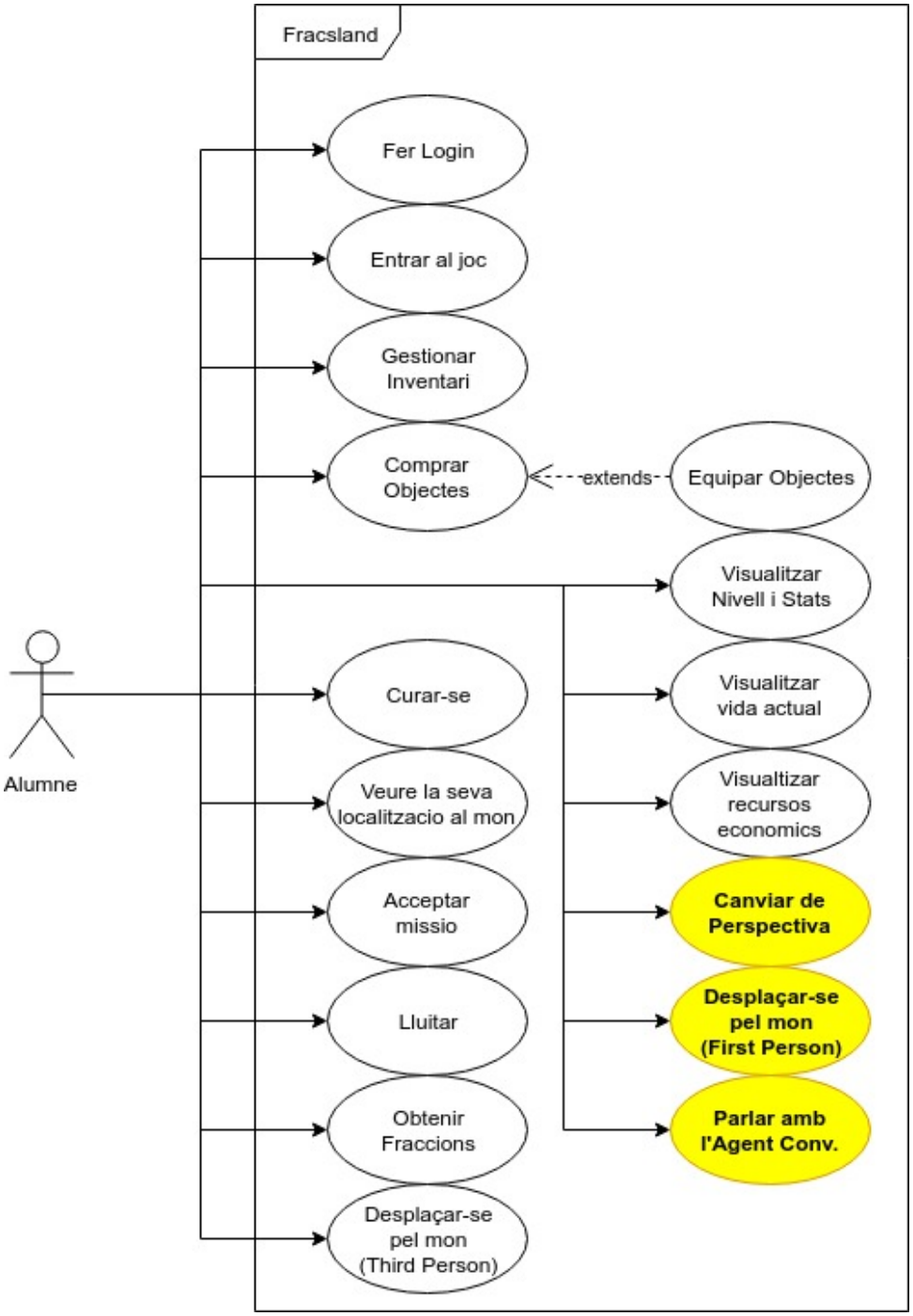

Figure 19: Diagrama de casos d'us original de *Fracsland*. [1]

- **Lleugerament modificats**: aquests casos d'ús han sofert una traducció prou immediata en el projecte de VR. Tot conservant la filosofia *point-and-click* del joc original, s'ha modificat l'UC per a permetre interaccions amb els comandaments d'Oculus sense arribar a modificar el seu flux principal. Interaccions a on parlàvem de "fer clic" o "seleccionar" (tot amb ratolí) ara admeten vectors per a apuntar dins l'escena virtual amb les mans com a referència, parlarem més aviat de "apuntar amb l'índex". Aquí podem trobar els UC "Fer Login", "Entrar al joc", "Curar-se", "Acceptar missió", "Lluitar" i "Obtenir Fraccions".

  Com podem observar a la *Figure 20*, el flux bàsic de l'UC no ha sigut canviat en essència però si han calgut fer algunes modificacions per a que s'entengui que tot i parlar de **seleccionar**, per exemple, no volem dir amb el ratolí i des de fora del joc, sinó que ens referim a fer el gest amb una de les mans i apuntar físicament als elements de UI del joc. Aquesta mateixa situació (o molt similar) es repeteix per la resta de casos d'ús d'aquest grup.

Nom:	**UC1** - Fer Login
Actor:	Alumne i "Servidor" de joc
Descripció:	L'usuari fa *login* al sistema.
Precondicions:	L'usuari acaba d'obrir el joc, no ha fet *login*.
Flux Bàsic: 1. L'usuari obre el joc i s'obre l'escena inicial de *login*. 2. L'usuari **apunta amb la mà i selecciona** el botó de *login*. 3. El sistema "verifica" les dades de sessió i entra al joc.	
Flux Alternatiu: 3a. El sistema detecta un error (de connexió o credencials) i mostra un missatge d'error a l'usuari.	
Postcondicions:	L'usuari ha entrat al sistema i al joc.

Figure 20: Cas d'us literal "Fer Login" modificat per a FracslandVR.

- **Flux principal modificat**: aquests casos d'ús han sigut modificats en el seu flux d'execució bàsic per a acomodar un millor sistema d'interacció a la VR. Principalment, aquests UC acomoden en la seva descripció el flux bàsic en interactuar amb un sistema de UI pensat pel nou paradigma de la VR (tot element ha d'intentar ser diegètic, o com a mínim espacial). Normalment, accions descrites com "seleccionar al UI" ara implicaran trobar aquests elements al món virtual, o bé situats a l'abast del jugador (com pot ser UI vinculada a les mans) o bé vinculats a un lloc de l'escena (com pot ser la tenda i el seu UI en parlar amb el venedor).

  Aquí trobem els casos d'ús "Gestionar inventari", "Comprar objectes", "Equipar objectes", "Visualitzar Nivell i Stats (estadístiques)", "Visualitzar vida actual", "Visualitzar recursos econòmics", "Veure la seva localització al món" i "Desplaçar-se pel mon (*Third Person*)".

  Per exemple, aquí veiem diferents exemples de canvis en casos d'ús d'aquest tipus:

Nom:	**UC4** - Comprar Objectes
Actor:	Alumne
Descripció:	L'usuari compra objectes amb els recursos que posseeix.
Precondicions:	L'usuari està jugant a l'escena "*Fracstown*".
Flux Bàsic: 1. El jugador se situa davant de la tenda del venedor de l'illa. 2. El jugador **apunta i selecciona** al venedor amb la mà. 3. El venedor obrirà sobre la seva tenda el panel de UI a seleccionar objectes. 4. El jugador **apuntarà i escollirà** l'objecte que li interessi, d'entre armes, cascos i abillaments. 5. El venedor comprovarà que el jugador tingui suficients recursos econòmics. 6. En cas positiu, el venedor confirmarà la compra i donarà accés al **UC5** (Equipar Objecte) per a aquest objecte en el seu UI.	
Flux Alternatiu: 6a. En cas negatiu, el venedor informarà de la situació a l'usuari amb un missatge d'error.	
Postcondicions:	L'usuari ha adquirit nova equipació.

Figure 21: Cas d'us literal "Comprar Objectes" modificat per a FracslandVR.

Nom:	**UC6** - Visualitzar Nivell i Stats
Actor:	Alumne
Descripció:	L'usuari consulta el seu nivell els seus *stats* de joc.
Precondicions:	L'usuari ha entrat i està jugant al joc.
Flux Bàsic: 1. El jugador s'emporta la seva **mà esquerra** al camp de visió, al revers d'aquesta troba la seva vida i nivell (**UC8**). 2. El jugador gira la mà esquerra fins que queda orientada **amb la palma cap amunt**. 3. El sistema al detectar la **posició de la mà**, mostrarà sobre ella el *stats manager* amb les dades rellevants.	
Flux Alternatiu:	
Postcondicions:	L'usuari sap el seu nivell i els seus *stats*.

Figure 22: Cas d'us literal "Visualitzar Nivell i Stats" modificat per a FracslandVR.

Nom:	**UC9** - Veure la seva localització al món
Actor:	Alumne
Descripció:	L'usuari consulta la seva posició actual al minimapa.
Precondicions:	L'usuari ha entrat i està jugant al joc.
Flux Bàsic: 1. L'usuari vol saber on es troba a l'illa i prem el **botó de menú** del **comandament esquerre** (pressiona el *joystick* del comandament esquerre). 2. El sistema obre un panel de UI al davant del jugador amb informació extra que no es troba vinculada a les mans. 3. L'usuari disposa del minimapa en aquest menú contextual, fins a tornar a prémer el botó inicial de menú.	
Flux Alternatiu:	
Postcondicions:	L'usuari coneix la seva posició al món.

Figure 23: Cas d'us literal "Veure localització al món" modificat per a FracslandVR.

Com podem veure a les *Figure 21, 22 i 23*, el flux bàsic ha sigut lleugerament modificat tot respectant els objectius i funcions primàries d'aquests. En molts d'aquests els canvis es deuen a la relocalització dels elements que existien al HUD original i ara es troben a les mans (*Figure 24*), relocalització d'elements a l'espai de joc (tenda, panel de missions, etc) o bé a la necessitat d'usar els comandaments *Touch* d'Oculus en comptes de ratolí/teclat.

Figure 24: Imatge dels nous elements de UI de *FracslandVR* per al **UC6**.

- **Nous casos d'ús**: finalment aquí trobem els casos d'ús nous inclosos a la versió de FracslandVR per a acomodar els nous sistemes d'interacció pensats per a la VR. Aquí oferim opcions exclusives a aquesta nova versió del joc, com per exemple canviar el *PoV* del jugador per posar-lo als ulls de l'heroi del joc (*First Person*), desplaçar-se en aquesta nova perspectiva o bé parlar (fent ús d'agents conversacionals) amb l'ajudant de l'heroi per a demanar-li pistes o amb el venedor del joc per a comprar objectes.

Nom:	**UC12** - Canviar de Perspectiva
Actor:	Alumne
Descripció:	L'usuari canvia de *PoV* al joc.
Precondicions:	L'usuari ha entrat i està jugant al joc.
Flux Bàsic: 1. L'usuari es troba a un *PoV* (*Third Person* o *First Person*) concret i vol canviar-lo. 2. L'usuari prem el botó de canvi de *PoV* del **comandament dret**. 3. El sistema del joc fa una transició *fade-to-black* i col·loca a l'usuari al seu nou *PoV*.	
Flux Alternatiu:	
Postcondicions:	L'usuari ha canviat la seva perspectiva de joc.

Figure 25: Nou cas d'us literal "Canviar de Perspectiva" per a FracslandVR.

Nom:	**UC14** - Desplaçar-se pel món (*First Person*)
Actor:	Alumne
Descripció:	L'usuari es desplaçarà per l'escena de joc.
Precondicions:	L'usuari està jugant i es troba en el mode *First Person*.
Flux Bàsic: 1. L'usuari mou un dels *joysticks* dels comandaments *Touch* per a activar el sistema de locomoció (sense deixar-lo anar). 2. El sistema mostra un indicador de UI amb la nova posició del jugador. 3. L'usuari pot moure el comandament amb el que ha començat la interacció per a desplaçar l'indicador. 4. L'usuari també pot canviar la rotació de sortida amb la rotació del *joystick* emprat per al moviment. 5. Un cop l'usuari troba la nova posició i rotació desitjades amb l'indicador el sistema comprovarà la nova posició destí i canviarà el color d'aquest en funció de la validesa del destí. 6. Si la posició es valida i l'usuari deixa anar el *joystick*, el sistema s'encarregarà de fer la transició *fade-out-fade-in* a la nova localització.	
Flux Alternatiu: 6a. Si la posició no es valida, el sistema descartarà aquest destí i no desplaçarà a l'usuari.	
Postcondicions:	L'usuari s'ha desplaçat pel mon en primera persona.

Figure 26: Nou cas d'us literal "Desplaçar-se pel mon (*First Person*)" per a FracslandVR.

Nom:	**UC17** - Desplaçar-se pel mon (*Third Person*)
Actor:	Alumne
Descripció:	L'usuari es desplaçarà per l'escena de joc.
Precondicions:	L'usuari està jugant i es troba en el mode *Third Person*.
Flux Bàsic: 1. L'usuari prem els *triggers* dels dits índex i cor d'un dels **comandaments**. 2. L'usuari mou la seva mà en la direcció contrària a la que vol anar, com si **agafes el món**. 3. El sistema desplaçarà l'avatar del jugador, que es podrà moure sobre el seu **mateix pla** (alçada).	
Flux Alternatiu: 1a. L'usuari prem els *triggers* dels dits índex i cor **dels dos comandaments**. 2a. L'usuari mou les seves mans en la direcció contrària a la que vol anar, com si **agafes el món**. 3a. El sistema desplaçarà l'avatar del jugador, que tindrà total llibertat per moure's **amb sis graus de llibertat** (eixos X, Y i Z).	
Postcondicions:	L'usuari s'ha desplaçat pel mon en tercera persona.

Figure 27: Nou cas d'us literal "Desplaçar-se pel mon (*Third Person*)" per a FracslandVR.

Podem observar (a les *Figure 25, 26 i 27*) que aquests nous casos d'ús no tenen un impacte real en les mecàniques de joc, però si modifiquen l'experiència base de joc per a poder donar opcions d'interacció més adients a la VR (mes naturals, intuïtives...).

5 Disseny

En aquesta secció podem trobar les principals decisions de disseny preses a l'hora d'implementar *FracslandVR*. També podem trobar els principals diagrames de disseny (de classes i de seqüència) de cada nou UC i com s'adeqüen i integren amb la resta de sistemes ja oferts pel projecte original.

Parlarem de les noves classes introduïdes pel SDK d'Oculus i les noves classes fetes per nosaltres, tot esmentant els canvis més grans soferts per part de les classes originals del joc (no VR). Primerament però, farem èmfasi en explicar la situació actual de la VR als motors de joc moderns.

5.1 Estat de la VR als motors de joc moderns

A causa de la natura del mercat de la VR actualment, no trobem uns estàndards o uns protocols per a que els fabricants de visors donin un accés més universalitzat al *runtime* dels seus productes. Aquesta situació fa que cada fabricant (com pot ser Oculus, HTC o Sony) tingui la seva pròpia plataforma i SDK's (*Software Developement Kit*) per als seus visors, no sempre donant suport en les principals eines per a fer videojocs. Això vol dir que els desenvolupadors hauran de tenir molt de compte a l'hora d'escollir un motor de joc que suporti tots els visors principals. També serà feina seva integrar cadascuna de les plataformes (o SDK's) dels visors de VR al seu projecte.

Tot i trobar extensa documentació i exemples per part dels principals fabricants, Oculus i Valve ofereixen un blog de referència amb totes les explicacions per a usar els seus *plugins*. Hem observat que aquesta mateixa manca d'estàndards fa que cadascun canviï els seus SDK's entre versions de forma dràstica si així ho desitgen i en aquests casos no sempre queden tots els canvis documentats a un *changelog* [11] prou acurat.

Recau doncs en el desenvolupador d'experiències per a VR el fet d'integrar les diferents plataformes en les quals vol donar-se exposició i això suposa un increment inevitable en la complexitat del projecte en què treballi. El desenvolupador haurà de pensar en quins sistemes de VR posseiran els seus usuaris [12] i a quins voldrà donar suport en el seu joc i dissenyar amb un nivell d'abstracció que permeti generar sistemes el més homòlegs possibles entre plataformes de VR.

5.1.1 Diferents integracions per a cada plataforma

A grans trets i per a resumir l'estat dels estàndards actuals a la VR, podem parlar dels dos més estesos entre els usuaris:

SteamVR (en concret Valve com a desenvolupador) ofereix el seu estàndard *OpenVR* que volent ser de filosofia *open source* permet donar suport a qualsevol fabricant que vulgui el seu visor a la plataforma (com HTC amb Vive o el mateix Rift d'Oculus) amb *plugins* que suporten els motors de joc més coneguts, Unity i Unreal Engine. Aquest tipus de suport obert "a qui vulgui apuntar-se" segueix una filosofia que ens és familiar de Valve i de les plataformes tradicionals de videojocs d'ordinador.

[11] Oculus SDK ver. *reference*: developer.oculus.com/documentation/unity/latest/concepts/unity-reference-scripting,
Oculus SDK ver. *release notes*: developer.oculus.com/documentation/unity/latest/concepts/release-archive

[12] Quins controls (com els *Touch* d'Oculus o les *Wands* de SteamVR) i quins visors tenen instal·lats al seu ordinador.

Per altra banda, Oculus i la seva plataforma només donen suport als visors Rift de la mateixa companyia, creant un ecosistema de software més tancat i semblant al que podria trobar-se a una consola moderna com PS4 i la seva *PlayStation Network* o a XBOX i el seu *XBOX Live*. Tot i que aquesta filosofia privativa per part d'Oculus sembla intuïtivament pitjor, això també permet als desenvolupadors d'experiències per a la plataforma comptar amb un sistema mes acotat; sempre comptaran que l'usuari estarà usant el visor Rift i els controls Touch i no com a SteamVR, que no poden fer assumpcions d'aquest tipus respecte al sistema de VR que usa el seu usuari. Permet crear experiències de joc més afinades a les capacitats que ofereix aquest visor. La plataforma d'Oculus també ofereix un seguit de funcionalitats al seu *plugin* per a connectar usuaris en *lobbies* per a experiències multijugador, generar i confirmar microtransaccions per a compres dins l'experiència, demanar i confirmar les identitats dels usuaris dins l'experiència, etc.

Per donar un exemple, si en un projecte volguéssim mostrar el nostre producte a la tenda virtual d'Oculus i de Steam [13] i volguéssim també suportar diferents tipus de visors (visors de SteamVR per una banda i Oculus Rift per l'altra), hauríem de fer una versió per a la tenda de Steam amb el suport dels visors d'aquesta i una altra per a la tenda d'Oculus i el visor Rift. Una altra opció seria donar suport total en l'estàndard *OpenVR*, inclòs el visor d'Oculus, així amb una única versió i integració podríem fer funcionar la nostra experiència amb qualsevol visor, tot perdent la possibilitat d'exposar-nos a la tenda d'Oculus o fer servir les funcionalitats de la plataforma.

Oculus compta amb un seguit d'eines per a integrar funcionalitats en VR a qualsevol producte. Podem trobar integració nativa per a aplicacions de Windows 10, integració amb *web* mitjançant l'extensió *WebVR*, integració per a Android (fent ús dels visors GearVR o el recent Oculus Go) amb una extensió d'*Android Studio* i finalment amb els motors de joc Unity i Unreal Engine.

5.1.2 OVRPlugin per a Unity i Unreal Engine

Ja que com a desenvolupadors comptem amb una unitat d'Oculus Rift i Oculus Touch, el projecte original està fet en Unity i tenint en compte l'antiguitat d'Oculus al mercat, cerquem a la documentació oficial com funciona el seu SDK en relació amb el motor de joc.

El SDK d'Oculus es divideix en tres components principals, instal·lats independentment, que integren parts diferents de la plataforma (veure *Figure 28*):

- **Oculus Utilities**: on està contingut *OVRPlugin*, la part central del SDK, els recursos i *prefabs* necessaris per a córrer la VR. Tot i que aquest *OVRPlugin* ja està integrat de forma nadiua des de la versió 2017.1 de Unity, es recomana instal·lar aquesta part de la plataforma per a assegurar-nos de tenir la última versió del *plugin*, ja que la de sèrie podria estar desactualitzada. Aquest component s'encarrega de les funcions principals del visor: control de posició, rotació, input, etc. Aquest component permet accés *runtime* al visor i controls.

- **Avatar SDK**: que conté tots els components necessaris per a mostrar l'avatar per defecte d'Oculus (mans i cap). Tot i que no és estrictament necessari usar aquesta part de la plataforma (sobretot en cas de fer ús d'un avatar fet a mesura per l'experiència), aquesta ofereix totes les animacions i transicions per a fer gestos amb les mans i usar-les a l'escena de Unity i permet usar-les en qualsevol avatar.

[13] Oculus Store: *oculus.com/experiences/rift*, SteamVR Store: *store.steampowered.com/vr*

- **Platform SDK**: que ofereix la part més social de la plataforma. Dona suport a la creació *in-game* de sales multijugador, permet posar passarel·la de pagament per a microtransaccions, confirma i usa els perfils dels usuaris d'Oculus, etc. Aquesta part del SDK és la més opcional del joc; per a una experiència d'un sol jugador, sense fer ús de capacitats socials, microtransaccions o *leaderboards* i centrada en la narració d'una història no tindria gaire sentit incloure-la, ja que un cop confirmada i llençada a la *Oculus Store*, no se'n faria ús.

Figure 28: Diagrama simplificat de relacions entre els diferents components del SDK de Oculus per a *Unity*.

5.2 *Oculus Utilities for Unity (OVRPlugin)*

En primera instància, necessitem conèixer el funcionament intern del SDK d'Oculus per a saber que ens ofereix el seu *plugin* per a Unity. La plataforma d'Oculus, com hem comentat a la secció anterior, es divideix en tres grans components que treballen junts per a aconseguir cadascun diferents tasques.

En el diagrama de la *Figure 29* podem veure les relacions entre els principals components (o *scripts*) del SDK en la realització del nostre projecte. L'entitat en blau representa un *prefab* [14] de Unity, `OVRPlayerController`, que conté tota la funcionalitat del visor i els comandaments d'Oculus per a usar a l'escena. Sol ser el substitut de les càmeres tradicionals de Unity. Les entitats en taronja son diferents classes opcionals que ofereixen interfícies per a accedir o bé a punters en l'espai virtual (`OVRRaycaster` i `OVRPhysicsRaycaster`) o a botons i superfícies tàctils dels comandaments (`OVRInput`).

[14] Els *prefab* son el sistema que te Unity per a guardar un *GameObject* concret amb les seves propietats, a mode de *template*.

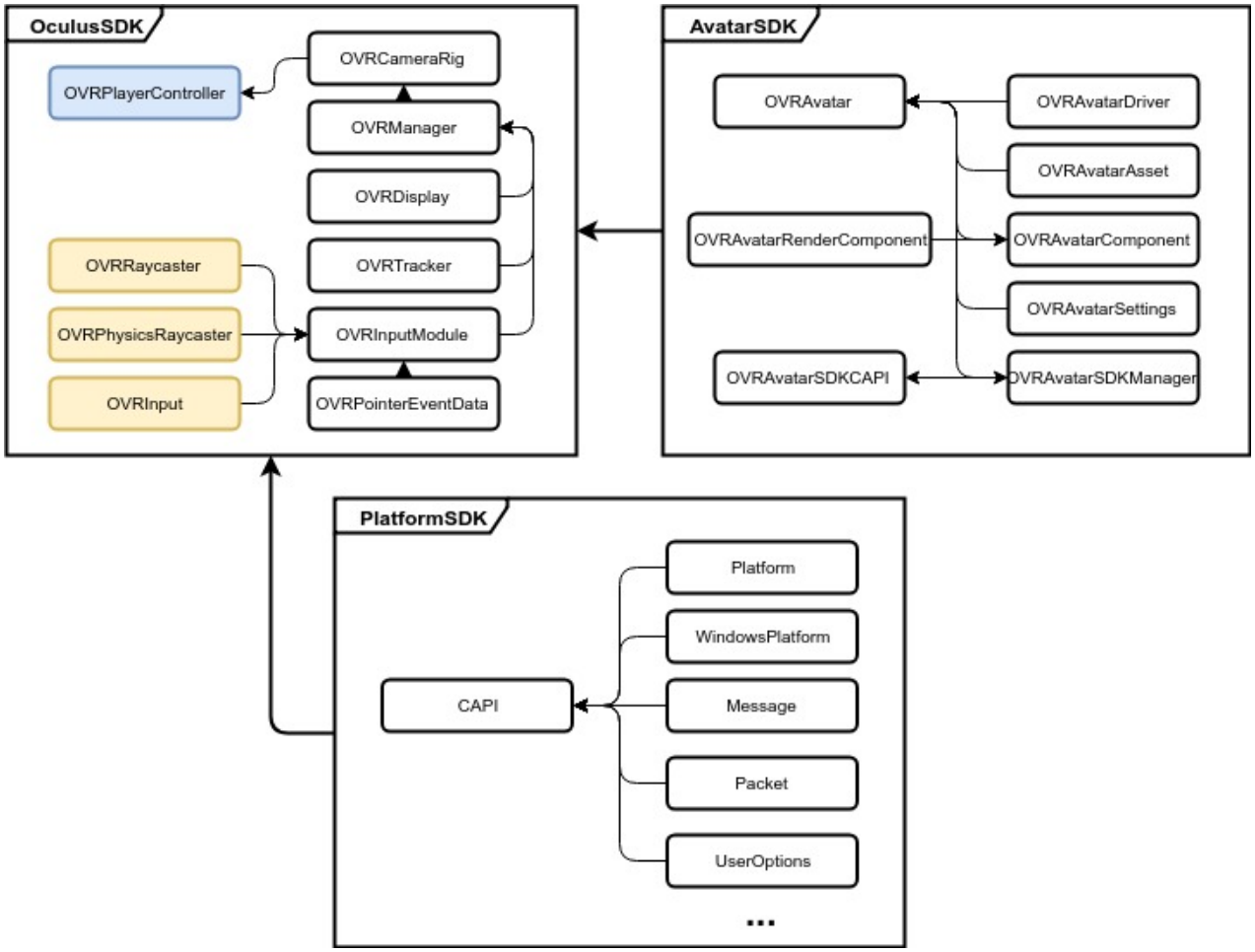

Figure 29: Relació entre els tres components i les classes emprades en el nostre projecte de la plataforma de Oculus per a *Unity*.

Les entitats en blanc son tot un conjunt de components que (cadascun en el seu domini) cooperen per a dur a terme la seva tasca. Al OculusSDK trobem que per a fer funcionar aquest component clau `OVRPlayerController` es fa ús del nou sistema d'interacció que admet vectors `OVRInputModule` (que substitueix el *Standalone Input Module* dels *EventSystems* tradicionals a Unity) i diferents *scripts* per a la funcionalitat bàsica dels panels del visor i el sistema de *tracking* (`OVRDisplay` i `OVRTracker`) tot coordinats per les classes `OVRManager` i `OVRCameraRig`.

Al AvatarSDK trobem tot un seguit de classes que es coordinen amb `OVRAvatar` per a renderitzar i animar les mans i cap de l'avatar de l'usuari d'Oculus. En aquest component trobem que els seus *managers* `OVRAvatarSDKCAPI` i `OVRAvatarSDKManager` tenen un paper més secundari, oferint una porta per a comunicar-se amb els controladors dels altres components (el ja conegut `OVRManager` o el del PlatformSDK `CAPI`, *Core API*).

5.3 Diagrama de classes de FracslandVR

Per a generar la versió en VR de *Fracsland* inevitablement s'ha aprofitat el disseny original del joc, donant lloc a les següents classes principals en el nostre projecte:

Com podem veure a la *Figure 30*, les principals cinc classes de la versió original de *Fracsland* han sigut conservades tot sofrint petites modificacions per a acomodar les entitats en vermell, els components del SDK d'Oculus. Per altra banda, s'ha afegit una

nova classe anomenada `WorldGUIController` (en groc) que comparteix responsabilitats amb l'original `GUIController` per a mostrar diferents elements de UI durant l'experiència de joc. La nova s'encarregarà de situar i controlar els elements espacials del joc (vinculats a les mans o a diferents punts de les escenes, veure *Figure 38* i *Figure 49*) mentre que l'antiga s'usarà per mostrar informació addicional a mode de menú contextual *overlay* (minimapa i objectius).

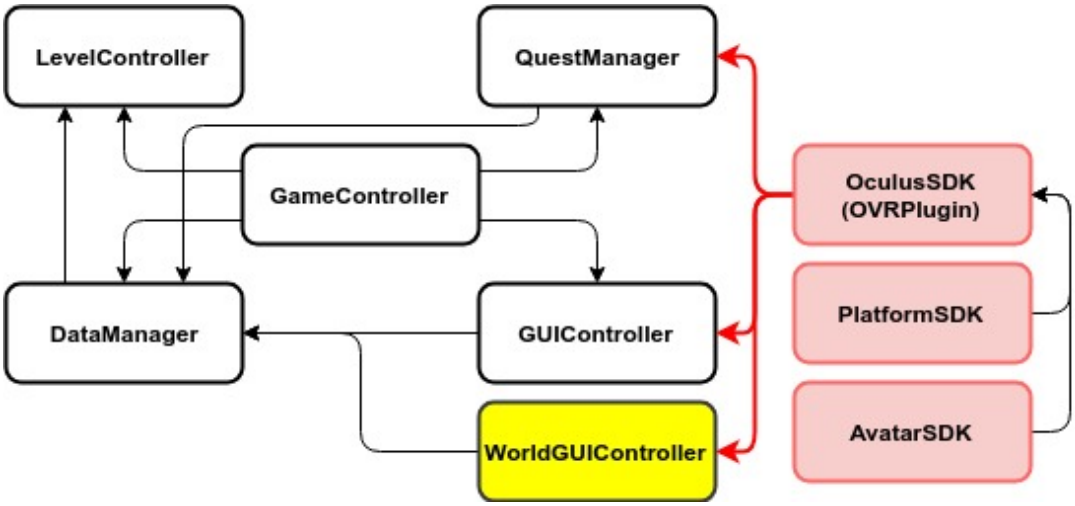

Figure 30: Diagrama de relacions de les principals classes de *FracslandVR*.

En aquesta nova versió, el control de la interfície d'usuari ha sigut dividit segons el tipus de *Canvas* requerit. `WorldGUIController` conte els objectes que s'han de posar en *World Space* a Unity i que per tant tindran una posició real a l'escena. Aquí trobarem tant els elements que es posicionen un sol cop a l'escena (com la tenda o el taulell de missions de *Lord Barus*) com els que segueixen la posició de les mans a mode de *Reworked HUD* [15] (indicador de vida i nivell, inventari i *stats*, etc). A l'altra banda, `GUIController` funcionarà a mode de "HUD flotant" situat davant l'usuari i contindrà elements "pseudo-espacials" (situats sempre davant la visió de l'usuari de forma artificial *overlay*) que donaran informació secundària al jugador com el minimapa o els objectius desglossats.

Ambdues classes, `WorldGUIController` i `GUIController`, es comunicaran constantment amb `OVRInput` i l'objecte d'escena `OVRPlayerController` per a consultar la posició del visor i les mans del jugador i saber com ha de posicionar la UI en conseqüència (veure *Figure 36*).

5.4 Locomoció a FracslandVR

Per a solucionar el problema de la locomoció al nostre projecte, *FracslandVR* ofereix dos sistemes independents i complementaris que prenen els conceptes de primera i tercera persona als videojocs tradicionals com a base filosòfica en el seu desenvolupament.

[15] Parlem d'aquesta UI com *Reworked HUD*, ja que és el millor símil per a aquest tipus d'interfície permanent a les mans.

5.4.1 *First-Person*

Aquest *PoV* és el punt de vista més obvi per a la VR. L'usuari se situa als ulls de l'heroi i mitjançant un sistema d'apuntat i teletransport, pot resoldre les missions de l'illa com si ell fos el protagonista.

En la nostra versió del *First Person* per VR, fem ús d'un dels sistemes de locomoció més usats i coneguts per a experiències d'aquest tipus (*Township Tale* i *Rec Room* per exemple). Consisteix en usar els comandaments *Touch* de les mans per a apuntar amb una paràbola-làser un nou punt del món i el sistema s'encarrega de fer un fos a negre per a desplaçar a l'usuari sense crear desconnexió entre els moviments reals i els virtuals. Aquest sistema és ofert al SDK d'Oculus i pot ser modificat i adaptat per a qualsevol projecte de VR en primera persona en Unity. Està format per una màquina d'estats que controla les "intencions de teletransport" (usuari apuntant, usuari quiet i usuari teletransportant-se) i comença sempre a l'estat *Ready* (veure *Figure 31*). Actua usant un seguit de *scripts* que permeten personalitzar el seu funcionament en cada estat.

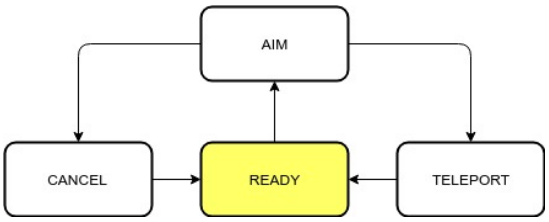

Figure 31: Estats del sistema de locomoció en primera persona de *FracslandVR*.

En aquest punt de vista, l'objecte `OVRPlayerController` se situa a on es troba l'heroi a l'escena i aquest últim es fa invisible per a l'usuari. Es desactiva també la possibilitat de moure's amb les mans com a la tercera persona.

A la *Figure 32* podem veure com es relacionen les classes per al funcionament de la locomoció en primera persona. La classe `LocomotionTeleport`, continguda al *GameObject* `LocomotionController`, s'encarrega de gestionar l'estat segons el *input* donat per l'usuari. El sistema usa el component `TeleportDestination` per a generar una possible localització a la qual es desplaçarà l'usuari cada cop que passa de l'estat *Ready* a l'estat *Aim*. Tot el sistema connecta `LocomotionTeleport` amb l'objecte `OVRPlayerController` mitjançant el seu *script* `MainCameraController`, que controla quin sistema (tercera o primera persona) ha de ser actiu en cada moment.

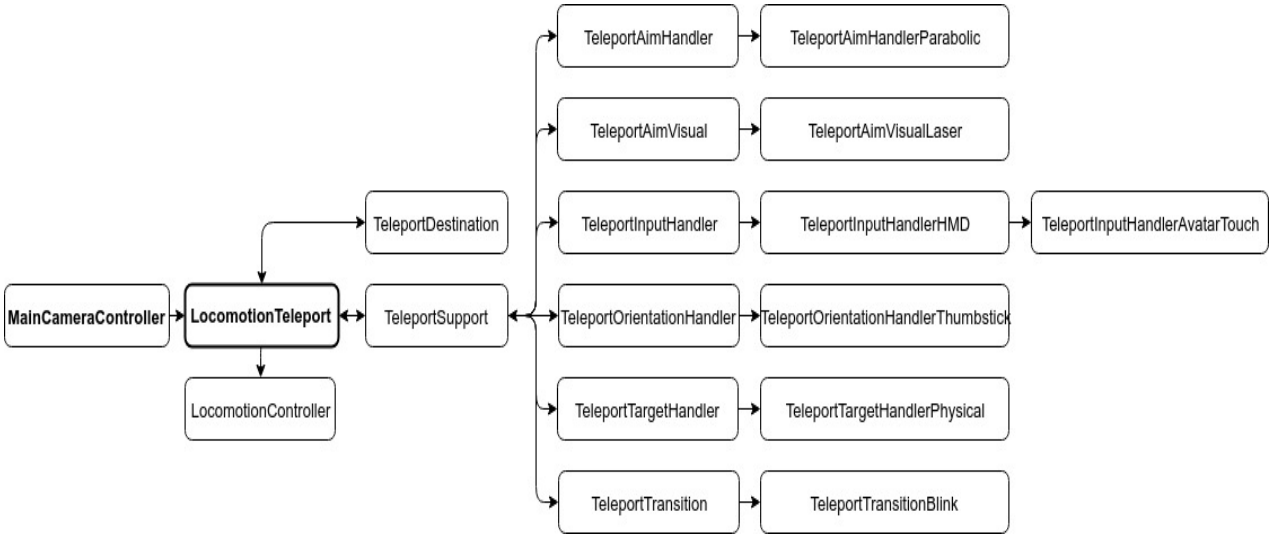

Figure 32: Relacions entre components del sistema de locomoció *First Person*.

En el cas d'aquest diagrama, les classes "pares" com `TeleportAimHandler` o `TeleportTargetHandler` es relacionen únicament amb les classes "filles" emprades al nostre projecte per a aconseguir la nostra configuració del sistema de locomoció en primera persona. Si en comptes d'usar les classes `TeleportAimHandlerParabolic` i `TeleportTargetHandlerPhysical` (filles de les abans esmentades) al nostre sistema de locomoció haguéssim usat, per exemple, `TeleportAimHandlerLaser` i `TeleportTargetHandlerNavMesh` podríem fer que el sistema mostri un làser recte per a apuntar en comptes d'una paràbola i fer que la validació dels `TeleportDestination` respongui a un *NavMesh* de Unity en comptes de simples *Colliders* al terreny de l'escena.

Podem veure al següent diagrama (veure *Figure 33*) de seqüència el flux típic d'aquest sistema un cop l'usuari passa a l'estat d'apuntat usant el *joystick* dels seus comandaments estant en primera persona fins que torna a l'estat preparat:

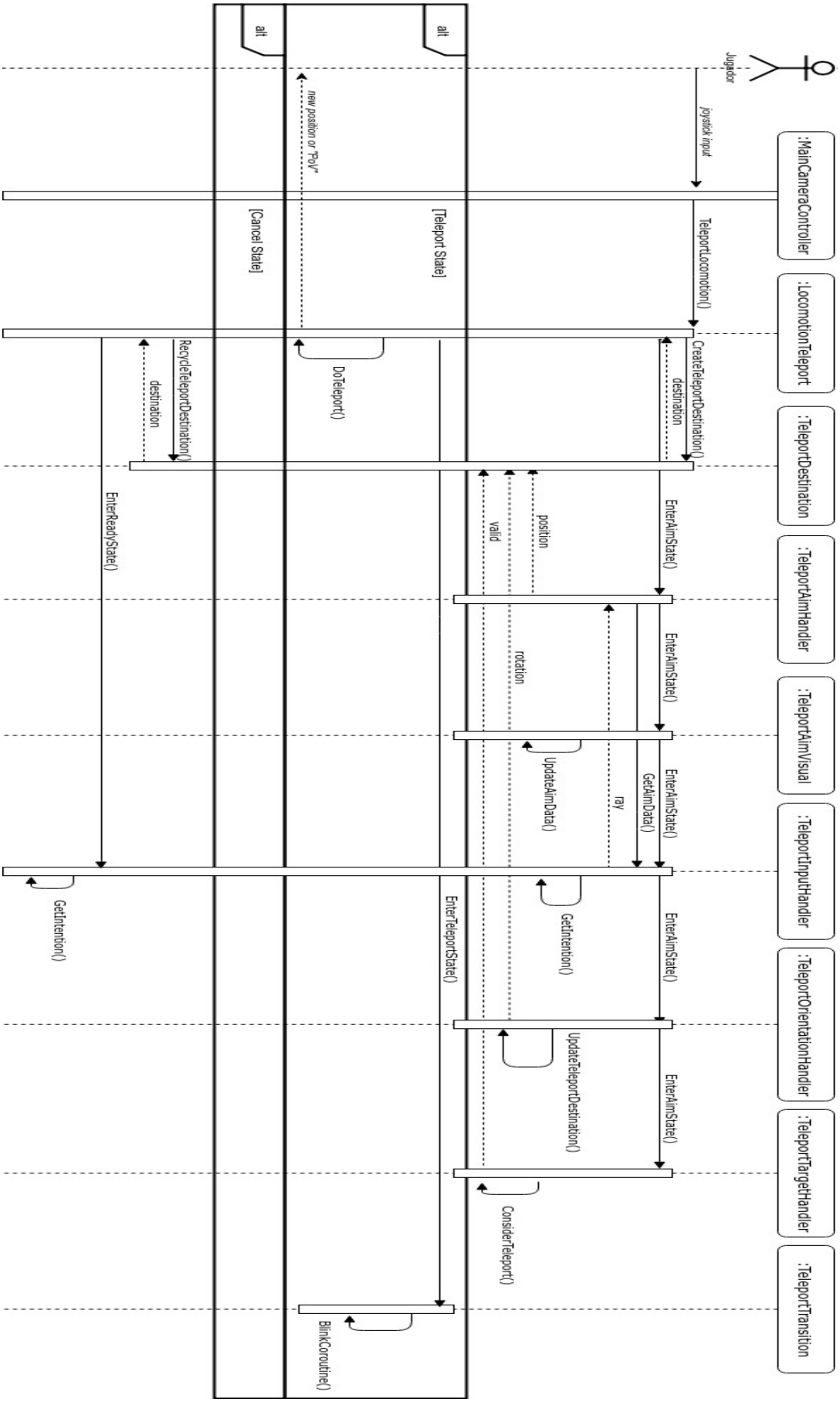

Figure 33: Diagrama de Seqüencia en desplaçar-se pel mon en primera persona.

Veiem que cada component comparteix responsabilitat per a posicionar i validar la nova possible localització a desplaçar-se. Són `TeleportAimHandler` i `TeleportOrientationHandler` els que guarden la posició i rotació marcada per l'indicador visual parabòlic. Aquest indicador és generat per `TeleportAimVisual` i canvia de color depenent de la validesa de la posició destí (ha de ser terreny navegable per a que sigui valida) que és comprovada per `TeleportTargetHandler`. Finalment l'*input* de l'usuari, és a dir la intenció, és comprovat per `TeleportInputHandler`. Un cop confirmada, són `LocomotionTeleport` i `TeleportTransition` els que fan el moviment efectiu. Cada cop que s'usa aquesta teleportació, `LocomotionTeleport` recicla els objectes destins `TeleportDestination` que ja han sigut usats i genera un de nou per al següent moviment.

5.4.2 *God-Mode* (o *Third-Person*)

Aquest sistema és una reinterpretació del punt de vista en tercera persona tradicional als videojocs. Prenem com a base el fet que hem de poder veure l'heroi des de fora dels seus ulls i usant com a inspiració els sistemes que podem trobar a *Brass Tactics* o *Echo Arena*, s'ha desenvolupat un sistema de locomoció en tercera persona també anomenat *God Mode* que usa les mans com a eines per a "agafar" l'espai virtual i "flotar" per ell.

Aquest sistema és molt més senzill que el de primera persona, ja que fent ús de les funcionalitats de simulació de físiques oferides per Unity amb els seus *Colliders* i *Rigidbodies*, podem prendre la velocitat lineal dels comandaments de les mans per a afegir una forca a l'objecte `OVRPlayerController` i així desplaçar-se en la direcció marcada per les mans sempre que no col·lideixi amb altres objectes.

En aquest cas (*Figure 34*), no comptem amb una complexa màquina d'estats per a saber i controlar a on anirà l'usuari en cada moment, ja que el moviment de l'heroi i el seu propi seran independents completament. En aquest punt de vista, l'heroi passa a ser controlat d'una forma similar, *point-and-click*, a l'original de *Fracsland*:

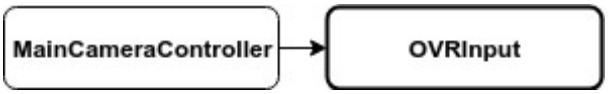

Figure 34: Relacions entre classes implicades en la locomoció en tercera persona.

- **Desplaçar l'heroi**: l'usuari aixeca el seu dit índex del *trigger* i tancarà el del seu dit cor per a mostrar un làser, que servirà per a apuntar al mapa i desplaçar l'heroi.
- **Desplaçar el seu avatar**: l'usuari tancara el dit índex i el cor, fent un gest de "tancar" la mà (o mans), per a controlar el moviment del seu avatar i poder-se apropar o allunyar de l'escena com si estigués manipulant un diorama o una taula amb joguines.

En el següent diagrama de seqüència podem veure el flux típic en desplaçar-se en tercera persona:

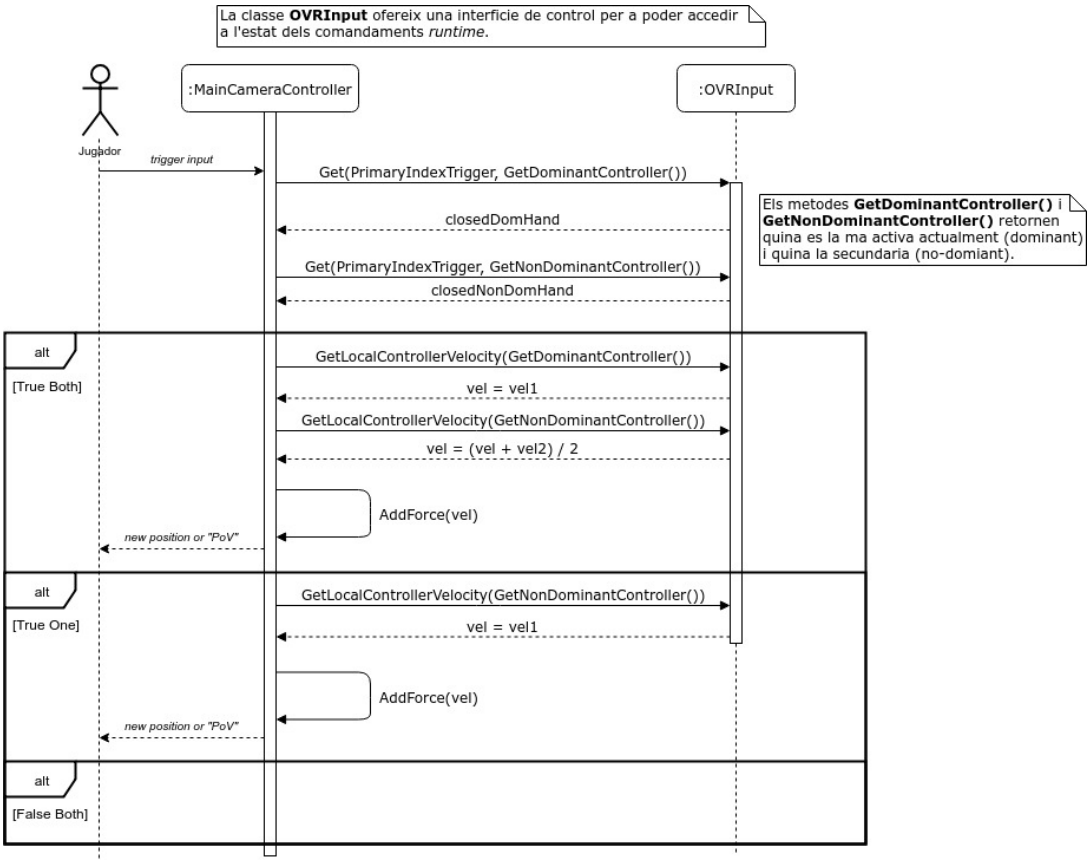

Figure 35: Diagrama de Seqüencia en desplaçar-se pel mon en tercera persona.

Aquesta implementació (veure *Figure 35*) intenta donar més control sobre els sis graus de llibertat oferts en aquest mode diferenciant el moviment amb una sola mà del de dues mans alhora. "Agafar" el món amb una sola ma permet a l'usuari desplaçar-se sobre l'eix X i Z de l'espai com si estigués en condicions de gravetat zero, però amb una resistència o "fregament" que el frenarà. Agafar-lo amb les dues permet moure's sobre els tres eixos i controlar així també l'elevació respecte al mapa tot amb un fregament menor.

5.5 Interfícies a FracslandVR

A causa d'un seguit de limitacions tecnològiques la forma de fer interfícies d'usuari a VR canvia molt respecte a un videojoc tradicional, això causa que en un projecte com *Fracsland*, que basa les seves interaccions centrant-les en el ratolí i el seu HUD, s'hagin de prendre un seguit de decisions que canvien per complet com l'usuari consulta la informació indispensable de joc.

Basant-nos en lliçons apreses en els exemples i escenes de referència ofertes al SDK d'Oculus i diferents jocs amb bons dissenys de UI diegètics com *Brass Tactics*, *Rec Room* o *A Township Tale*, en aquest projecte finalment s'ha optat per mostrar la informació de joc que usualment trobaríem al HUD del joc 2'5D aquesta vegada vinculada a les mans. Per exemple, en comptes de trobar la vida o el seu inventari a una finestra flotant davant seu, l'usuari mirarà la seva mà esquerra i trobarà la seva barra de vida al dors de la mà;

o girarà la mà dreta orientant la palma cap amunt i el seu inventari apareixerà sobre d'aquesta.

Al diagrama de relacions del `WorldGUIController` (*Figure 36*) podem veure les principals classes modificades per a adequar el projecte a la VR i la nova distribució d'elements entre els controladors. Podem veure com la nova classe controladora, `WorldGUIController`, passa a assumir quasi tota la responsabilitat de mostrar la UI a l'usuari i els elements que abans eren del HUD ara tenen la seva posició a l'espai de l'escena. L'antic controlador `GUIController` per altra banda, mostra la informació massa complexa o gran en forma d'un panel flotant davant de l'avatar de l'usuari en forma de menú contextual activat amb un botó de les mans.

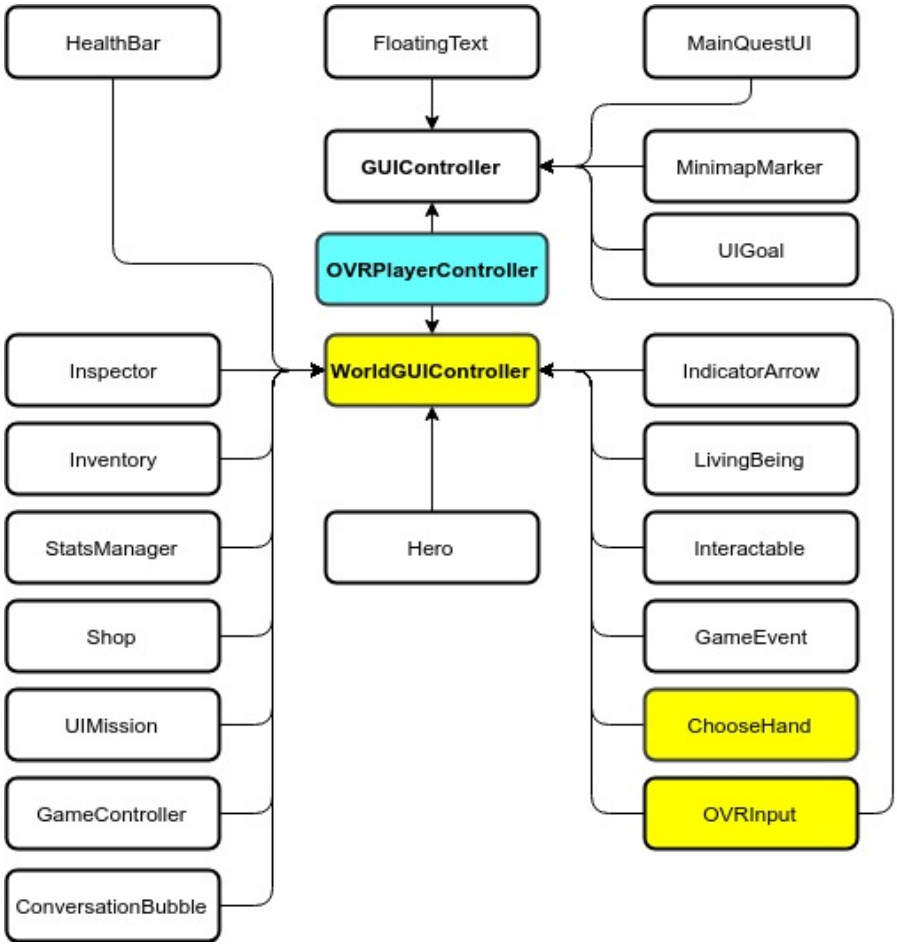

Figure 36: Diagrama de relacions de les classes modificades `GUIController` i la nova `WorldGUIController`.

En blanc podem veure els components que ja existien a la versió original, tot modificats per a adequar-se al *Canvas* espacial. En groc podem trobar classes noves implicades en la gestió del UI. `ChooseHand` per exemple gestiona quin comandament està sent usat en cada moment, per saber quina és la "mà activa". El nou controlador `WorldGUIController` consulta constantment a la classe `OVRInput` per a poder posicionar correctament el UI de les mans i accedeix a l'objecte `OVRPlayerController` (en blau) per a tenir les posicions de referència de l'avatar.

5.5.1 Conversió del HUD i elements del mon (diegètics i espacials)

Com ja hem mencionat prèviament, al nostre projecte (i a la VR en general) les mans passen a ser el principal recurs constant al joc, donant així la possibilitat de mostrar informació immediata o senzilla vinculada a aquests elements amb els quals sempre comptarem.

A *FracslandVR* l'usuari comptarà amb diversos elements de UI diegètics i espacials, alguns pocs contextuals distribuïts arreu del mapa i molts altres al voltant del model de les seves mans.

La tenda del joc, el taulell de missions (*Figure 37*) o el panel d'elecció de vegetal a plantar en la granja són exemples de UI espacial contextual que apareixerà davant l'usuari vinculada a un punt proper del seu element "invocador". Si el jugador interactua amb el venedor o en *Lord Barus* per exemple, es mostraran uns panels que li permetran escollir les diferents opcions amb el làser del dit índex (veure *Figure 38*).

Figure 37: Exemples diferents d'elements espacials a *FracslandVR*.

Aquests elements són mostrats i amagats pel controlador `WorldGUIController` gràcies al *script* del SDK d'Oculus `OVRInputModule`, que actua com a *EventSystem* de Unity i transforma els clics en *Screen Space* en vectors d'apuntat (*raycasts*) en *World Space* amb el seu `OVRPointerEventData`. Els `GameEvents` juntament amb els objectes de joc i NPCs amb el component `Interactable` han sigut modificats també per a ser usats amb *raycasts* en l'espai virtual. Són els responsables de parlar amb el controlador i fer les crides pertinents a aquest.

Figure 38: Imatges amb l'*Inspector* i els *stats* del jugador a *FracslandVR*.

Les mans del jugador contindran tots els elements que normalment trobaria immediatament al HUD: la barra de vida, el seu nivell o l'inspector del joc (a l'apuntar a un element `Interactable`); i els elements contextuals del HUD més comuns i usats com l'inventari del jugador o la seva aparença i estadístiques.

Aquests elements són controlats i invocats directament des de `WorldGUIController`, que manté actualitzada la posició del UI a les mans i mostra o amaga els elements contextuals de les mans segons l'orientació d'aquestes. El conjunt d'un producte vectorial i un angle entre vectors en aquest controlador comprova si alguna de les mans està orientada palma amunt i mostra l'inventari en el cas de la mà dreta i les estadístiques del jugador a la mà esquerra. D'aquesta forma s'ofereixen de forma immediata però sense ser intrusiva.

Combinant el producte vectorial i l'angle format entre els vectors *Up* de l'escena de Unity amb els del comandament de les mans (veure *Equacions 1, 2, 3* i *4*), trobem una mètrica que ens permet saber quan la ma passa a estar en l'orientació desitjada (palma amunt). Si els valors (en combinació) estan dins una finestra concreta, es mostra el UI pertinent, en cas contrari s'amaga.

$$lHand.up \times World.up < 0.5 \tag{1}$$

$$rHand.up \times World.up < 0.5 \tag{2}$$

$$\frac{\overline{lHand.up} \cdot \overline{World.up}}{|\overline{lHand.up}| \cdot |\overline{World.up}|} < 50^\circ \tag{3}$$

$$\frac{\overline{rHand.up} \cdot \overline{World.up}}{|\overline{rHand.up}| \cdot |\overline{World.up}|} > 100^\circ \tag{4}$$

El disseny visual dels elements de *FracslandVR* però és quasi idèntic a l'original i podria adequar-se encara mes a la VR si se'n fes ús de textures o imatges *low-poly* per al seu UI, que permeten una major claredat visual, millor rendiment i una sorprenent solidesa visual dins els visors actuals.

5.5.2 Menú i elements no-diegètics (o "pseudo-espacials")

A la versió de VR de *Fracsland* hi ha però també uns elements prou complexos gràficament que no permeten seguir aquest principi de disseny de mostrar-los a les mans.

Figure 39: Imatge del menú secundari no-diegètic de *FracslandVR*.

El minimapa o la llista d'objectius desglossats són objectes que s'han deixat com a responsabilitat de l'antic controlador de UI i aquest ara fa la funció de *canvas* flotant davant la visió de l'usuari, invocat pressionant el *joystick* del comandament esquerre (*Figure 39*). Aquest menú contextual no el considerem espacial del tot, ja que no té una posició real al món virtual, sinó que sempre quedarà fixat en una molt propera a la visió de la càmera lleugerament davant seu. Aquesta és una de les tècniques per a presentar HUDs o *overlays* a la VR. Es pot interactuar amb aquesta sense major modificació, ja que fa ús del *EventSystem* personalitzat d'Oculus.

Aquest tipus d'elements també poden ser redissenyats per a estar encara més integrats al joc i ser més immersius. En pròxims treballs, per a evitar fer ús de tècniques poc optimes per a la VR actual, es podria plantejar la generació d'un objecte tipus "llibre" que mostres la informacio que ara es troba en el menu secundari (*overlay*) del joc a les seves pàgines, semblant al llibre d'instrucció trobat a l'esquena de l'avatar de joc de *OrbusVR* (veure *Figure 40*). Així tota la informació es trobaria en diferents llocs de l'avatar del jugador, podent abandonar completament la interacció clàssica "prem botó, obre UI" per a fer una de més natural basada en gestos del cos o postures de les mans.

Figure 40: Captura del Llibre d'instrucció del joc *OrbusVR*.

6 Implementació

Actualment podem trobar un ampli espectre de visors disponibles al mercat introduïts aquest últim any per part d'altres empreses (entre elles incloses Microsoft) i desenvolupadors de hardware (Lenovo, LG, Acer...) a sobre dels ja coneguts Rift i Vive que ofereixen pràcticament la mateixa experiència o amb algunes petites millores introduïdes en relació a resolució interna del visor (veure *Figure 41*). Tots ells són productes amb relativa poca adopció d'usuaris en respecte a l'interès invertit per part del sector a causa de l'alt preu d'adquisició i als requisits mínims requerits per a que un ordinador domèstic pugui suportar-los. S'espera però que aquestes condicions canviïn amb el temps quan el cost d'entrada per la realitat virtual baixi i els ordinadors suficientment potents siguin més accessibles.

Figure 41: Visors de PlayStation (a la dreta) i Microsoft (Samsung Oddisey i Acer MR).

La situació tan fragmentada en el mercat de la realitat virtual (mercat molt nou i no hi ha suficient massa crítica com per a invertir en bases i estàndards) fa que la integració amb les eines més usades per part dels desenvolupadors de software, com poden ser els motors de joc Unity3D o Unreal Engine, no sigui trivial, ja que no existeix un estàndard comú proposat per a integrar les funcionalitats d'aquests visors amb els sistemes existents. Això provoca també una fragmentació en termes de plataformes de software; ja que cada visor té la seva integració en cada motor, si els desenvolupadors no integren totes les plataformes en les quals vulguin tenir suport no podran usar el seu software en diferents visors.

Per a aconseguir assolir els objectius plantejats en aquest projecte i complir amb el disseny establert, la implementació de *FracslandVR* ha sigut dividida en *builds* o versions separades, segons s'afegien funcionalitats o es convertien elements per a la VR, que veurem a continuació.

Aquí podem trobar des de la primera versió, on únicament es tracta d'integrar el SDK d'Oculus al projecte original fent el mínim de modificacions possibles, fins a l'última que inclou la nova interfície i ambdós sistemes de Locomoció per a l'illa en VR.

6.1 Versió preliminar de *FracslandVR*

En la primera versió realitzada de *FracslandVR* podem trobar la adaptació més ingènua a l'agafar *Fracsland* i sumar-li el SDK d'Oculus; una mena de tercera persona per VR sense adaptar cap sistema de joc per a aquest tipus d'interacció. Per a aconseguir-la, únicament canviem el *GameObject* de la càmera a Unity per `OVRPlayerController` i passem el *Canvas* de joc a *Screen Space - Camera* tot vinculant-lo amb aquest nou objecte càmera.

En aquesta versió comptem amb el nou objecte base a les escenes de joc `OVRPlayerController` (*Figure 42*), que substitueix les càmeres tradicionals de Unity i també comptem amb el làser d'apuntat del dit índex de la mà, obligatòriament necessari si volem seleccionar coses a l'espai virtual i que funciona gràcies a la classe `ChooseHand`. Com que aquesta versió manté tota mecànica feta al projecte original, l'usuari no tindrà cap tipus de control sobre la posició de la càmera, que seguirà els moviments de l'heroi, i únicament tindrà la possibilitat de rotar-la usant aquest com a pivot. Aquesta solució per al PoV ràpidament ens ensenya que les càmeres que es mouen seguint els desplaçaments d'un personatge solen generar marejos i dificultats per a seguir aquests moviments, tot i provar d'implementar tècniques de *lerping* (també conegut com a *Smooth Follow*) en les seves translacions. Per altra banda, aprenem també que mai hauríem de tocar la rotació de l'objecte `OVRPlayerController`, ja que aquesta sensació de desconfort i desconnexió amb el moviment encara és més pronunciada.

Figure 42: Imatges de l'objecte `OVRPlayerController` al inspector i escena de Unity.

En referència a la interfície d'usuari, el UI del joc es mostra exactament igual que la seva versió *Flat Screen* (*Figure 43*), donant la mateixa informació en un *overlay* davant la visió de la càmera a mode de HUD molt complet. Aquest objecte HUD (anomenat `UI` a l'escena de Unity) aprofita el nou component `OVRInputModule` per a que s'hi pugui interactuar amb ell usant el làser de la mà. Fa ús del mode de *Canvas* de Unity *Screen Space - Camera* per a fer aquesta traducció d'espai de pantalla a espai "virtual" de càmera. A més de poder controlar-lo apuntant amb la mà, els botons dels comandaments Touch ofereixen *shortcuts* per a obrir o invocar certes parts d'aquest (com l'inventari, el *stats*, etc).

Figure 43: Imatge del HUD emprat a la versió preliminar de *FracslandVR*.

A les imatges de la *Figure 44* podem observar a l'inspector de Unity els paràmetres emprats en el *Canvas* que compon aquest HUD "traduit" a l'espai virtual.

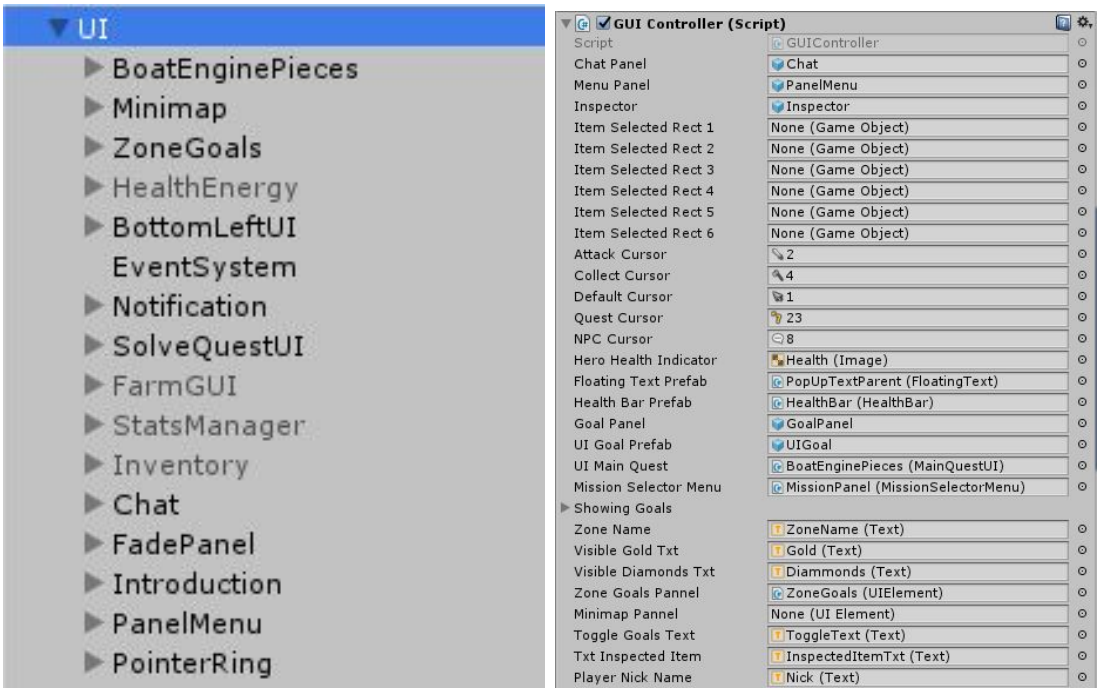

Figure 44: Imatges de l'objecte `GUIController` usat a *FracslandVR*.

Aquesta versió del HUD ens ensenya ràpidament que no hem d'atapeir la visió de l'usuari amb molts elements complexos de UI, ja que actuaran com a distracció, no seran perfectament clars i comprensibles, oclouran la visió i no permetran interpretar tot el que passa a l'espai de joc sense perdre el focus d'atenció.

6.2 *FracslandVR* amb Locomoció en tercera persona

En aquesta versió es comencen a adaptar els sistemes coneguts a *Fracsland* per al bon funcionament en VR. Aquí comptem amb un sistema per a moure l'avatar del jugador (el protagonista del joc) i l'heroi (el personatge del joc) independentment. Aquest sistema, autoanomenada locomoció *God Mode* (o *Third Person*), permet a l'usuari "agafar" el món virtual i impulsar-se en la direcció contrària al moviment de les mans, tancant els dits índex i cor (prement ambdos *triggers* del comandament Touch) com si fes el gest de tancar les mans.

Aquest tipus de locomoció dona més control a l'usuari sobre on vol situar-se per a veure l'escena de joc i dóna lloc a una experiència de joc que no provoca tant desconfort en els seus moviments i no trenquem tant la immersió. En aquesta *build* fem responsable a la càmera principal (l'objecte ja conegut `OVRPlayerController` amb el component `MainCameraController`) de controlar els comandaments de les mans i les seves velocitats locals tot parlant amb la classe `OVRInput`. Per altra banda, s'ofereix per primer cop l'opció d'ocultar el HUD prement el *joystick* de la mà esquerra, netejant així la visió de l'usuari d'elements complexos i permetent veure l'escena amb claredat, com a solució temporal.

Figure 45: Imatge de la versió amb HUD des-activable (desactivat en aqui) de *FracslandVR* i la seva tenda de joc.

Tot i començar a adaptar els sistemes coneguts per a la VR, tota la interfície segueix encara funcionant com *overlay* de la càmera (*Figure 43*) i si l'usuari (tot tenint el HUD desactivat) interactua amb el venedor de la tenda per exemple, aquesta s'obrirà encara davant la seva visió, tapant tota l'escena que quedi darrere (*Figure 45*). Per altra banda, aquesta nova forma de desplaçar-se per món causa que a velocitats o acceleracions molt altes, el motor de joc no sigui capaç de mantenir la posició del HUD constant, donant lloc a uns artefactes visuals molestos o *jittering* que fan que aquest es mostri entre rafegues.

6.3 *FracslandVR* amb nova UI

Per a solucionar els problemes causats amb els components originals de UI i els introduïts al HUD amb la integració de la locomoció en tercera persona, plantegem un seguit de transformacions a realitzar a aquests elements, sobretot al HUD de *Fracsland*, i un seguit de redistribucions de responsabilitats entre classes controladores per a que puguin haver-hi dos *Canvas* de Unity a les escenes del joc.

Segons el disseny plantejat, el controlador `GUIController` s'encarregarà de gestionar el *overlay* ja conegut (*Screen Space - Camera*) i el nou `WorldGUIController` de mostrar els elements diegètics/espacials nous i posicionar-los a cada *frame* si cal (*World Space*, veure *Figure 46*).

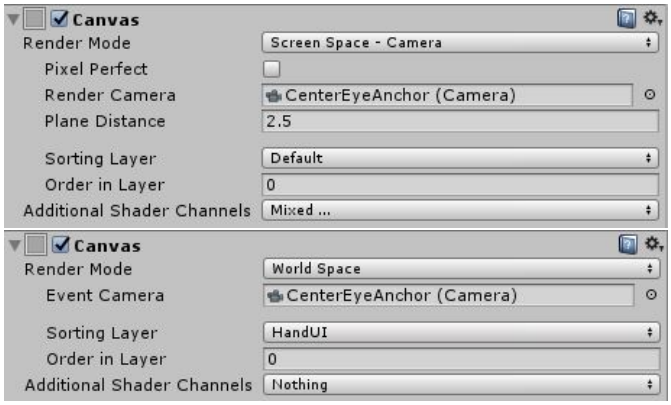

Figure 46: Imatges amb els *Canvas* emprats a *FracslandVR* a l'inspector de Unity.

Aquests nous elements espacials a `WorldGUIController` es troben de dues maneres: o bé de forma contextual arreu del món, sent activats a l'interactuar amb el seu *parent* (*Figure 47*), o bé vinculats a la superfície del model de les mans de l'avatar.

Figure 47: Imatges d'exemples de UI al mon de *FracslandVR*.

- **Elements del món**: els elements del primer grup són aquells com la tenda, el taulell per escollir que verdura plantar o la fletxa indicadora de missió, que tot i poder ser prou complexos per a mostrar-los a les mans, té sentit que se situïn a l'espai virtual de l'escena prop de l'objecte de joc `Interactable` que els invoca.

- **Elements de les mans**: els del segon grup són elements que es consideren prou indispensables (que formarien part dels elements immediatament visibles d'un HUD), prou clars i senzills per a poder situar-los com a recurs visual constant a les mans, com la barra de vida o el nivell del jugador. També trobem elements contextuals vinculats a les mans, però que s'activen i desactiven de forma diferent als del primer grup: a aquests elements s'hi podrà accedir amb un gest de la mà, per exemple, orientar la palma de la mà cap amunt mostrarà elements de UI depenent de la mà en qüestió. La mà dreta mostra l'inventari en ella, i l'esquerra el component `StatsManager`.

Molts d'aquests elements que han sofert aquesta conversió a UI diegètica/espacial han passat per un seguit de transformacions per a que el seu funcionament sigui més consistent amb l'ús de l'espai virtual.

Els elements de UI al *Fracsland* original (no VR) compten amb un seguit de funcions que els posicionen correctament a pantalla (p. ex. les funcions de Unity `WorldToScreenPoint()`

o `WorldToViewportPoint()`). Aquestes funcions fan que la seva posició a l'escena no importi, ja que en temps d'execució es mostraran sempre a la mateixa part de la pantalla i deixaran d'existir a l'espai virtual (UI no-Diegetica, veure secció 1.1). Les funcions que passen la UI de *World Space* a *Screen Space* han de desaparèixer del codi, ja que no faran més que moure aquests elements a l'hora de posicionar-los a l'espai virtual. En el nostre cas, hem de fer especial èmfasi a les instanciacions d'elements de UI per codi, per assegurar-nos de poder-los situar a on volem un cop passats al *Canvas World Space* en *runtime*. Per exemple, al nostre cas no només haurem d'eliminar aquests tipus de funcions del *Canvas* vestigial (`GUIController`), sinó que també ens hem d'assegurar que les `localPosition`, `localRotation` i `localScale` d'aquests elements de UI instanciats per codi siguin *resetted* per a que els *scripts* que els posicionen ho facin correctament.

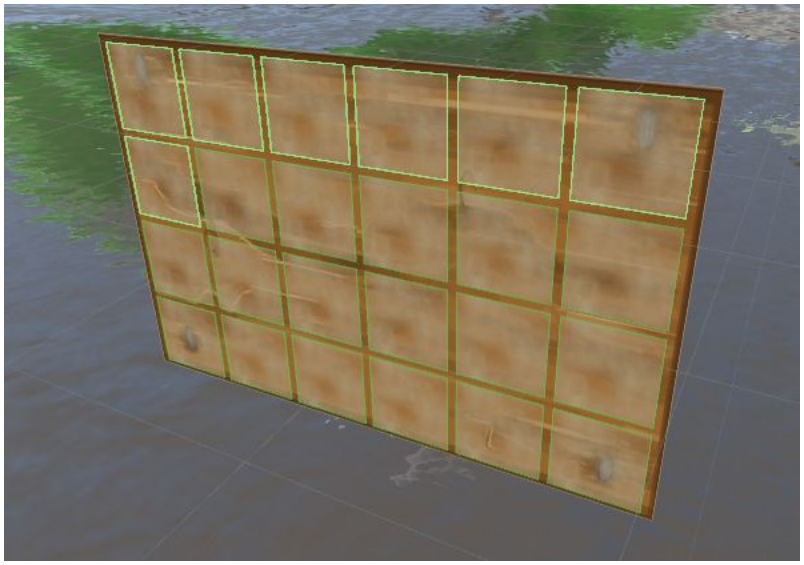

Figure 48: Imatge amb els *Colliders* afegits al inventari (en verd) per al seu funcionament a *FracslandVR*.

Altres elements, per altra banda, tenen ara *Colliders* per a poder permetre un bon funcionament del *raycast* dut a terme pel làser de selecció de la mà (veure *Figure 48*). L'inventari o els botons espacials de la tenda i el panel de missions són exemples d'aquests elements que ara compten amb aquests components per a detectar i bloquejar les col·lisions amb punters a l'espai virtual. Les imatges o icones en components de UI a Unity no bloquegen els *raycast* en les interaccions amb aquests punters en VR (per culpa del nou `OVRInputModule` presumiblement). Com a conseqüència, trobem un estil d'interacció *drag-and-drop* semblant a l'original. L'inventari respondrà al *raycast* del làser de la mà i permetrà seleccionar objectes del mapa o portar-los del seu inventari al món de joc per resoldre les missions (veure *Figure 49*).

Figure 49: Imatges de interaccions amb objectes de l'inventari per a resoldre la missio de la granja de *FracslandVR*.

Per altra banda, s'han inclòs els components `Inspector` (*Figure 49*) i `ConversationBubble` (*Figure 50*) al *Canvas* espacial per a poder mostrar el nom dels elements `Interactable` en ser apuntats pel làser de la mà i poder visualitzar les seves respostes predefinides a l'usuari en forma bombolla de diàleg. L'`Inspector` es mostra molt prop del dit on neix el làser d'apuntat, mentre que les bombolles de diàleg es mostren molt a prop dels personatges que les emeten.

Figure 50: Imatge de la bombolla de text de *FracslandVR*.

Finalment els elements conservats a `GUIController` ara juguen un paper més secundari, ja que les principals fonts d'informació passen a ser les mans en comptes del HUD. Ara en aquest *Canvas-overlay* podrem trobar aquells elements que són gràficament massa complexos per a posar-los a les mans o no té sentit que siguin a un lloc concret del mapa. Mostrats en un taulell a mode de "menú secundari" ocult per defecte, al contrari del funcionament de la funció d'ocultar HUD. S'afegeix també un indicador visual per a saber si aquest "menú" està obert o no i amb quin botó del comandament l'invocarem (boto L3 a les plataformes tradicionals). Per exemple, l'usuari trobarà a aquest nou menú el minimapa amb la seva posició i la llista d'objectius a complir a la missió, informació complexa però prou útil per a conservar-la i que pot ser prescindible gràcies a altres elements de UI.

6.4 *FracslandVR* amb Locomoció en primera persona i canvi de perspectiva

Aquesta última versió vol integrar la nova locomoció en tercera persona amb la màquina d'estats `LocomotionController` donada per Oculus, que ofereix un sistema en primera persona per a teleportar-se sense gran desconfort visual.

En el nostre projecte hem implementat un *flag* (anomenat `firstPersonEnabled`) a la classe *Singleton* `GameController` per a gestionar en quin PoV es troba l'usuari en cada moment (veure *Figure 51*). Aquest *flag* és controlat pel botó del *joystick* dret (conegut tradicionalment com a R3) i és comprovat en tot moment per `MainCameraController` que activa o desactiva els sistemes de locomoció necessaris en cada moment. Per a permetre un canvi de perspectiva còmode i fluid, s'ha afegit un nou estat d'entrada i sortida a la màquina del SDK d'Oculus *Transition* (veure *Figure 31* a la secció 5.4.1) per a poder fer una transició semblant a la teleportació de `LocomotionTeleport` en canviar de *PoV*, *fade-out-fade-in* que resulta ser molt còmode visualment. Ara, sempre que canviem de PoV, la màquina d'estats de la primera persona entrarà a l'estat *Ready* passant abans per *Transition* i sortirà també per aquest.

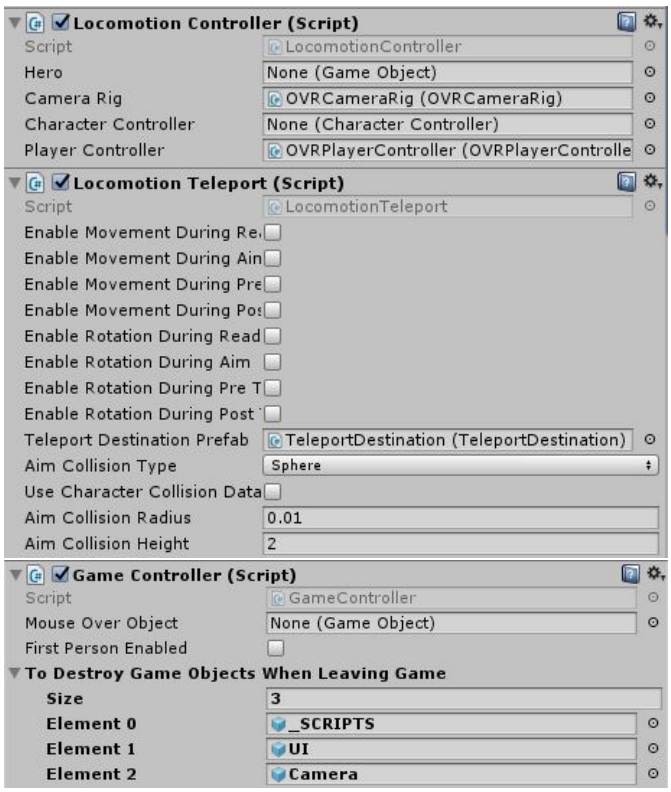

Figure 51: L'objecte `LocomotionController` i *flag* de PoV de `GameController` a l'inspector de Unity.

Aquest nou sistema per a la primera persona funciona amb els *joysticks* dels comandaments de les mans i permet usar una corba parabòlica per a apuntar a destins llunyans i marcar una nova posició i rotació de sortida com a nova localització de l'usuari. En usar la transició de primera a tercera persona o viceversa, el sistema portarà amb un fos a negre l'usuari fins a la posició del nou punt de vista. Si en acabar la transició (estat *Transition*) el punt de vista és la primera persona, la màquina d'estats salta a *Ready* per a permetre futurs moviments. Un cop dins aquesta i a l'estat *Ready*, si l'usuari usa el

joystick, apareixerà la paràbola per a apuntar (*Figure 52*), tot amb un indicador al final per a saber la nova posició del jugador i la nova orientació (controlada amb l'orientació del *joystick* emprat), que estarà activa fins a deixar anar la palanca analògica (estat *Aim*). Un cop es deixa anar, el sistema comprova que el destí sigui una posició valida, és a dir que l'heroi pot moure's fins allà, i farà la teleportació amb aquest fos a negre (estat *Teleport* i estat *Cancel*).

Figure 52: Imatges de l'indicador parabòlic-làser en verd (destí valid) i vermell (destí no valid) i de destí del sistema de primera persona de *FracslandVR*.

Finalment, en acabar la interacció, la màquina torna a l'estat *Ready* per permetre més moviments o tornar a l'estat *Transition* per a canviar de punt de vista. Trobem al menú secundari *overlay* un indicador visual que ens diu amb una figura molt senzilla si estem en primera o tercera persona en cada moment, donant així *feedback* visual immediat a l'usuari (veure *Figure 55*).

Per al correcte funcionament d'aquest sistema, s'han afegit *Mesh Colliders* a tots els objectes de les escenes els quals no són "navegables" per l'heroi (*Figure 53*). Aquests components permeten al *script* `TeleportTargetHandler` de la màquina d'estats comprovar si la posició destí està dins d'un d'aquests elements.

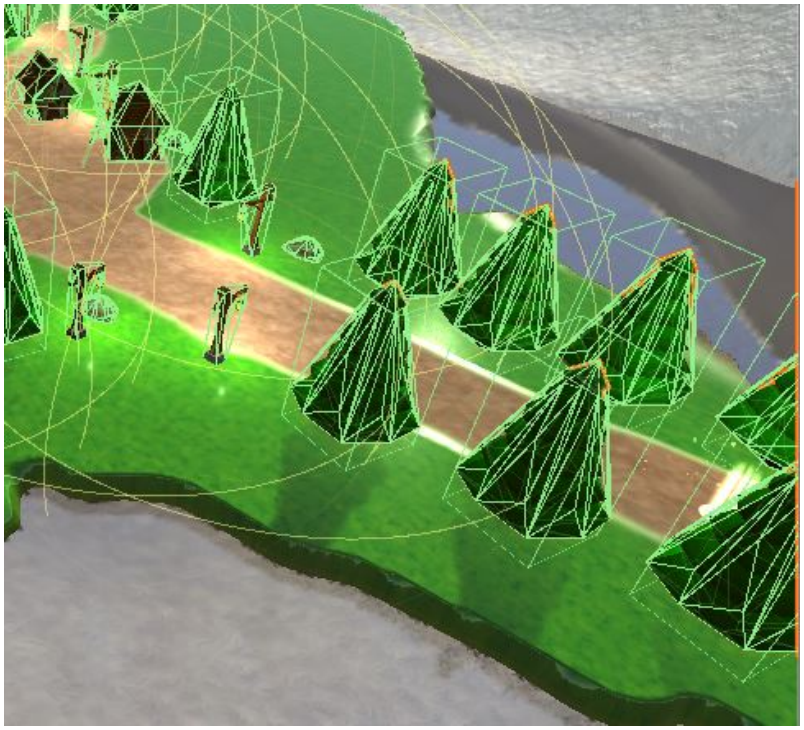

Figure 53: Imatges dels *Mesh Colliders* (en verd) de l'escena `TownVR` a Unity.

Finalment, en aquesta versió s'han afegit també un seguit d'elements de UI que ajuden a l'usuari amb aquestes funcionalitats específiques de la VR. Com podem veure a la *Figure 54* ara tenim en tot moment un indicador de "centre de l'espai físic de joc" i un marcador amb la posició actual del jugador als nostres peus, amb les seves respectives fletxes que indiquen la direcció *forward* de cada un. Aquest indicador marca cap on esta mirant l'usuari (fletxa "interior") i cap on estan orientats els sensors del sistema (fletxa "exterior"). També podem trobar un *chatlog* a la mà dreta, que guarda i mostra les frases dites pels usuaris als agents conversacionals integrats al projecte paralel [12], i finalment, indicadors visuals al menú secundari que ens diran en tot moment en què *PoV* ens trobem (*Figure 55*).

Figure 54: Imatges de l'indicador de centre d'espai (sota la vista del jugador) i del *chatlog* dels agents conversacionals.

Figure 55: Imatge dels indicadors de UI de menu i *PoV* (el negre vol dir menú secundari tancat o *God-Mode* actiu, i el blanc vol dir menú secundari obert o activada la *First Person*, segons l'indicador que sigui).

7 Conclusions i Resultats

Per a acabar, dividirem aquesta secció de resultats en tres apartats diferents. Primerament, un apartat en el que comentarem com ha quedat el projecte en relació amb els objectius plantejats, un segon en el qual parlarem dels resultats trobats en treballar en aquest projecte, tot esmenant els problemes trobats al llarg de la seva realització, i un tercer en el qual farem menció del possible treball futur i les possibles millores que es poden dur a terme a *FracslandVR*.

7.1 Objectius de *FracslandVR*

Primer de tot, avaluarem com ha acabat el projecte de *FracslandVR* respecte als objectius plantejats per a aquest en un començament. També parlarem de *bugs* o comportaments inconsistents que podem trobar al nostre projecte en usar les noves funcionalitats o elements implementats i esmentarem possibles solucions a aquests.

- **Interfície d'Usuari a *FracslandVR***:

  Com hem vist, la nostra nova interfície es pot dividir en dos: els elements de UI de les mans i els elements de UI del món. Aquestes solucions de UI intenten aprofitar al màxim la immersió a l'espai virtual de joc que ofereix la VR i donen lloc a elements de UI espacials que intenten situar-se de forma lògica i intuïtiva seguint la narrativa i *look* original de *Fracsland*.

  La nova interfície de *FracslandVR* ha comportat una sèrie de canvis en el disseny original que fa que molts dels objectes de UI hagin augmentat en complexitat (pel que fa a *scripts*, no visualment parlant). Això implicarà que han de ser tractats amb més compte dins de la lògica del joc. Per exemple, l'addició de *Colliders* a molts elements del UI del joc fa que hagin de ser desactivats quan no es mostrin i activats quan si (com pot ser l'inventari del joc quan la palma esta orientada cap amunt o cap abaix), ja que les classes originals de *Fracsland* esperen que aquests elements no siguin desactivats mai per complet [16]. Si no controléssim correctament aquests nous *Colliders* dels elements espacials i diegètics del UI de *FracslandVR* podríem, per exemple, bloquejar el *raycast* del làser d'apuntat de la mà sense veure cap element de UI o fins i tot fer que el *raycast* del làser travessi els elements de UI interactius.

  Per altra banda, l'ús dels recursos gràfics originals de UI de *Fracsland* ha causat que, en molts casos, ens hàgim limitat a "traduir espacialment" [17] les interaccions del joc, cosa que provoca alguns resultats que poden ser millorables per a la immersió en VR. El *Field of View* o la fidelitat visual del visor *Rift* [18] per exemple, ens pot limitar a l'hora de veure text a les interfícies una mica llunyanes del món de joc a *FracslandVR*.

- **Locomoció i punt de vista a *FracslandVR***:

[16] Parlem d'activar/desactivar els *Colliders* dins els *GameObject* vs. activar/desactivar els *GameObject* per complet.

[17] Aquesta traducció a la qual ens referim és la que implica passar els elements no-diegètics de UI originals a la nostra versió espacial (o en el millor dels casos, diegètica).

[18] FoV de l'Oculus *Rift*: 110 graus.

En el nostre projecte hem explorat dos sistemes de locomoció diferents i complementaris que ofereixen diferents nivells de confort i control a l'usuari a l'experiència de joc de *FracslandVR*. Aquests sistemes i punts de vista encunyats *First Person* i *Third Person* conviuen al joc donant un sistema més segur i confortable en el primer cas i un sistema més ràpid i "omnipresent" però amb un temps d'adaptació lleugerament més alt en el segon. Tot i això, ambdós sistemes de locomoció tenen la seva mínima corba d'aprenentatge i cada un introdueix la seva complexitat a la lògica de *FracslandVR* i un seguit de problemes.

El sistema de tercera persona del projecte, al contrari del seu oposat (*First Person*) que valida les posicions abans de teleportar-se i no deixa sortir del "terreny de joc", permet entrar dins de certes construccions de les escenes de joc tot i tenir un sistema de detecció de col·lisió (que si funciona en cas del terra de les escenes, per exemple) a causa de problemes amb els *Mesh Colliders* a Unity. Les escenes també han de controlar d'alguna forma els límits o *boundaries* a les que l'usuari pot anar, ja que si no l'usuari podria sortir del terreny de joc i perdre's.

Per altra banda, el sistema de primera persona del projecte introdueix un problema inexistent a l'anterior esmentat: els objectes del món de joc interactius (com ho són els animals de la granja, els *Non Playable Characters* o les palmeres de la platja de *Fracsland*) no respondran a l'heroi si no augmentem el seu "radi de detecció" d'aquests. Aquest radi és el sistema que té el joc per saber si l'objecte que representa l'heroi ha arribat a on es troba l'objecte interactiu. Com el sistema de tercera persona de *FracslandVR* usa un paradigma d'interacció prou semblant que el joc original (*point-and-click* amb el làser de la mà) el seu radi de detecció no cal que es modifiqui, però en el cas del sistema de primera persona cal modificarlo per poder tenir una "àrea d'influència" als elements interactius perquè l'usuari pugui seleccionar-los i interactuar amb ells. Amb "àrea d'influència" ens referim a l'espai en què respondran al jugador els elements interactius del món de joc de *FracslandVR*.

7.2 Pautes i problemes trobats treballant en *FracslandVR*

Durant el desenvolupament de *FracslandVR* hem pogut conèixer i aprendre la situació actual de la VR per a desenvolupadors de software. A causa de la novetat de la tecnologia trobem una situació complicada en relació al *Game Design* en VR.

Primerament, trobem un munt de *plugins* diferents per a integrar diferents visors a un joc en VR i no trobem un estàndard per a desenvolupar-los de forma que ofereixin fàcil accés entre visors en un sol SDK. A sobre d'això, s'ha de comptar amb què aquestes plataformes de suport de VR són altament volàtils, és a dir, que poden canviar molt ràpidament i dràsticament entre versions per a acomodar noves funcionalitats. Moltes vegades no tenim uns documents de referència prou actualitzats a les pàgines dels fabricants de visors i això pot provocar que els desenvolupadors puguin preferir usar versions antigues de diversos SDKs per a comptar amb més documentació o "temps de *testing*". Si no es pensa amb deteniment a priori a quins visors i plataformes es vol donar suport, podem trobar-nos amb complicacions pel que fa a compatibilitat entre SDKs i versions de Unity o *bugs* indocumentats de noves versions amb bastant facilitat.

La pàgina d'Oculus conte referencies a cada versió de les diferents integracions que ofereix, tot explicant els canvis fets entre versions (*changelog*) i les noves funcionalitats. Trobem en les nostres versions usades del *OVRPlugin* per a Unity (*ver. 1.24*) alguns

errors en la documentació referent als *mappings* dels botons dels comandaments *Touch*, que exemplifiquen a la perfecció aquesta situació complicada per als desenvolupadors de VR.

Per escenificar una situació que podria ser real: un petit estudi de jocs que just comença en el desenvolupament de VR pot acabar gastant un munt de recursos (temps i personal) en trobar aquest tipus d'errors en documentacions entre versions i repercutir en el temps de desenvolupament. A això s'hi suma la complexitat exponencial de voler donar suport a més visors de VR, ja que aquests problemes es poden repetir per a cada integració diferent. No ajuda a aquest estudi el fet que hagi de tenir a la seva disposició un visor i sistema de VR de cada tipus per a fer *testing* en cada plataforma. Tampoc ajuda el fet que no trobem gairebé res d'informació a les plataformes *online* o fòrums a on es presenten solucions de la comunitat com les conegudes *stackoverflow* i *UnityForums* a causa de la falta de projectes en VR. El fet que no comptem amb un llibre de pautes o nocions bàsiques per a desenvolupar experiències en VR obliga als desenvolupadors a treballar sota un paradigma mes de "prova i error". Tots aquests fets obliguen als desenvolupadors a tirar d'inventiva i creativitat per a implementar noves experiències intuïtives, amb bon rendiment i amb suport entre plataformes.

En el nostre cas a sobre, estem treballant amb un projecte ja completament desenvolupat per a adaptar-lo a la VR. Aquesta situació concreta causa un seguit de problemes extra en la realització de la nova versió de *Fracsland*. Intuïtivament, portar un joc existent a la VR sembla una tasca més senzilla que dissenyar un joc des de zero per a aquesta, però és probablement igual de complexa. En ambdós casos hem de tenir en compte tots aquests factors abans esmenats per a generar una bona experiència de joc, però en el nostre també hem de tenir en compte el disseny original del joc per a poder adequar-lo al nou paradigma de la VR tot respectant-lo. El que ens estalviem en dissenyar els *assets* i fer la implementació del joc, s'ha hagut d'invertir en temps de recerca per a entendre les decisions de disseny i el *codebase* de la plataforma original de *Fracsland*, en crear i provar sistemes alternatius per a una bona interacció a la VR i en investigar jocs de VR i exemples d'Oculus per a saber usar el seu SDK al complet.

Per a facilitar futurs desenvolupaments de nous jocs i experiències en VR o bé *ports* de jocs existents, presentem una sèrie de pautes o "bones pràctiques" per a poder dissenyar-los més fàcilment i de la forma més immersiva i intuïtiva possible:

- **Ni rotacions ni moviments artificials**:

  En les primeres versions de *FracslandVR*, vàrem observar que les tècniques tradicionals de seguiment i manipulació de càmera en els videojocs en VR no funcionaven en general, ja que generen una desconnexió entre els moviments artificials del seguiment de la càmera amb la posició quieta de l'usuari. Qualsevol desplaçament i sobretot rotació que es vulgui aplicar a la càmera ha de fer-se mitjançant una transició en la qual l'usuari no percebi aquestes translacions, ja que de forma contrària el podríem marejar. Es recomana que en la implementació d'interaccions amb càmeres de VR sempre s'ofereixi una situació de joc que permeti a l'usuari entendre el moviment artificial que està passant immediatament per tal de no trencar immersió.

  Per exemple, si volem fer ús de càmeres fixades, ho podrem fer sempre que puguem justificar aquest *setup* a l'escena de joc. Un joc de carreres en VR en el que el jugador és el conductor d'un cotxe o fins i tot una nau (6DOF) no genera cap tipus de desconnexió amb el moviment del vehicle, perquè normalment ja esperem estar

quiets dins d'aquest. Si volem fer ús de càmeres en tercera persona (com el nostre *God Mode* de *FracslandVR*), ho farem de forma que l'usuari pugui entendre el seu moviment independentment del personatge a seguir per l'escena. Si volem usar la més immersiva primera persona, tractarem d'oferir un sistema de transicions visualment plaents per a no fer pensar a l'usuari que el món es mou sense cap moviment seu real. De moment, per a aquest últim paradigma s'ha trobat que la teleportació, tot usant diferents tècniques visuals per a la transició segons l'experiència, és la millor forma de no trencar immersió en aquests moviments des de dins del joc. Molts jocs amb ritmes més moderats o lents usen una tècnica semblant a l'emprada a *FracslandVR*, el fos a negre, però molts altres de ritme més intens (com el conegut *Gunheart* de l'Oculus Store) usen de transicions més impactants visualment parlant, com un efecte de "salt de colors" entre la posició original i la final.

- **Màxima claredat i màxim *feedback***:

  En tota experiència per a VR, hem de tenir en compte que l'usuari tindrà quasi tots els sentits bloquejats pel visor que vesteix. Ni la vista ni l'oïda estan disponibles mentre jugues en VR, així que especialment en aquest cas hem de comptar amb un sistema de *feedback* excepcional que doni a l'usuari la forma de saber de forma molt senzilla on es troba ubicat al món real. Recomanem encaridament l'ús de sistemes que van més enllà dels indicadors visuals (tot i ser els més populars i efectius) per a reforçar el *feedback* que rep el jugador com sons o vibració (*feedback* hàptic dels comandaments); a la VR moltes vegades l'usuari es pot veure aclaparat per un munt d'estímuls visuals que facin que ignori certes fites visuals que avisen de coses importants del món real, en aquests casos pot ser d'ajuda un petit extra per a captar l'atenció del jugador. Per altra banda, recomanem també que l'ús de la interfície gràfica d'usuari "tradicional" es redueixi al mínim per a donar lloc a interaccions més naturals (properes a com es farien al món real) amb els elements de joc pertinents; només voldrem mostrar la informació realment indispensable tota l'estona, tota la resta que pugui ser contextual serà preferible veure-la només en cas de necessitat.

  Per exemple, podem comptar amb elements visuals com els implementats a *FracslandVR* per a saber on està el centre de joc o com els elements trobats a experiències més populars com *Robo Recall* de *Epic Games* que, a mes, indiquen en els *setups* de VR *Front Facing* si l'usuari mira cap a la direcció dels sensors o la contraria (per evitar problemes d'oclusió dels comandaments). També podem comptar amb elements visuals, sonors i hàptics per a donar avisos forts de coses molt concretes que passen a l'experiència de joc, com la vibració, el filtre *grayscale* i el so que s'emet quan ets eliminat d'una partida d'alguna activitat de *Rec Room* tot i no estar mirant que t'ha eliminat.

- **El UI ha de ser diegètic, o com a mínim espacial**:

  En els videojocs tradicionals normalment trobem als seus UI pocs elements diegètics, alguns d'espacials i un munt de no-diegètics, és a dir, simples objectes que es mostren fixats a la perifèria de la càmera. Tots aquests components en pantalla solen conformar el que coneixem com a HUD, que ja sabem que no funcionen gaire bé a VR a causa de limitacions tecnològiques. Per a poder donar una interfície en la qual oferir informació de forma homòloga al HUD (de forma immediata i en qualsevol moment) s'aconsella que tots els elements de UI d'un joc en VR tinguin un motiu i

lloc de ser al món virtual. S'aconsella també seguir el punt anterior i només comptar amb elements de UI indispensables. Les millors implementacions resulten ser les que aprofiten molt els espais de les escenes per a mostrar la majoria d'elements disposats arreu d'un lloc i les que usen pocs elements ben escollits per a controlar coses molt usades a les mans (com en *Brass Tactics* o bé *Tilt Brush*).

- **El Disseny de *software* importa molt**:

  En la creació d'un joc en VR però sobretot en la generació de la versió en VR d'un joc existent hem de tenir molt en compte els principis de disseny de software. Justament per culpa de la situació fragmentada de les plataformes de VR als motors de joc, trobem que no seguir uns bons principis de disseny pot comportar el desenvolupament d'un projecte poc escalable o que no admeti millores i modificacions sense canviar el seu *core* o fins i tot un fort impacte negatiu al rendiment del joc en integrar SDKs de VR. Per a aquest motiu, es recomana estudiar i seguir molt bé les pautes de disseny de software per a tenir una *codebase* prou modular i senzilla. Sobretot en fer *ports* de jocs existents com *Fracsland*, es recomana fortament conèixer els principis de disseny originals per a poder entendre les decisions preses pels desenvolupadors originals i així no "embrutar" o complicar l'estructura del projecte innecessàriament.

  Per exemple, en el nostre cas a *FracslandVR*, ens hem trobat amb un seguit de decisions de disseny preses a la versió original que compliquen el desenvolupament de la versió per a VR a causa de problemes amb l'accés a les dades de Unity per part dels SDK d'Oculus. Els *scripts* principals del joc original són objectes de tipus *Singleton*, que facilita molt l'accés a les dades que guarda des de qualsevol altre *script*. Aquestes classes o objectes pero, no són visibles en canviar d'escena pels components del SDK d'Oculus. En el nostre cas, això resulta en no poder vincular l'objecte contenidor de `GUIController` de forma fàcil a la càmera del visor entre escenes. El nostre *workaround* és usar la classe `WorldGUIController` per al nou *Canvas* espacial sense seguir aquest paradigma de disseny seguit en les originals (classes *Singleton*). D'aquesta forma aconseguim que `WorldGUIController` actuï com una "façana" que es pot comunicar amb la resta d'aquests components a cada escena.

7.3 Treball futur a *FracslandVR*

El desenvolupament de software i en concret de videojocs, en aquesta època en què es venen seguint la filosofia *Game as a Service*, és sempre cíclic. En acabar una iteració de desenvolupament completa i rebre el *feedback* adient, podem pensar en possibles millores a fer en el nostre producte per a donar un millor acabat o experiència.

En el cas de *FracslandVR*, volem oferir un seguit de coses que hem plantejat i pensem poden millorar l'experiència de joc. Podem observar diferents coses a millorar o treballar en el futur:

- **Canvis per a la *QoL* (*Quality Of Life*) de joc**:

  En la versió de VR de *Fracsland*, es poden fer diverses coses per a millorar-la sense canviar cap *asset* del projecte. Es poden netejar totes les carpetes de recursos gràfics no usats, les carpetes de *scripts* vestigials (els relacionats amb la plataforma *online* original de *Fracsland* i els no usats del SDK d'Oculus, per exemple) i netejar les

escenes de joc d'objectes o components que no s'usin (com poden ser els relacionats amb el *chat*, les notificacions o la UI original) per a aconseguir de forma fàcil un major rendiment i un *output* més estable de FPS.

Per altra banda, recursos com els rajos de sol emesos sobre el UI o els efectes de reflexió de l'aigua que en una pantalla resulten molt convincents, impacten molt negativament al rendiment del joc i no tenen un resultat visual gens bonic en els panels dels visors de VR. Aquests elements podrien ser eliminats o modificats per a mostrar-se únicament a la pantalla plana i no a la VR.

- **Avaluacions amb usuaris**:

  Per tal d'aconseguir *feedback* d'usuaris reals abans de seguir iterant sobre el disseny plantejat a *FracslandVR*, es pot realitzar una prova o sessió d'avaluació amb nens i nenes. Aquesta prova ens pot donar l'oportunitat d'extreure informació valuosa sobre com funcionen els nostres sistemes implementats.

  Es pot dissenyar una escena de *training* especial per a veure com els usuaris s'habituen a l'ús de la UI del joc i als nous sistemes de moviment en VR. En aquesta escena es podria preparar un *script* que enregistres tota activitat amb UI o comandaments dels jugadors en forma d'un *log* de text. Amb aquestes dades podem fer un anàlisi estadístic de les funcions afegides a la VR més usades o de les funcions menys clares.

  Finalment, fent ús d'una enquesta preparada per als jugadors de la prova també podríem trobar suggeriments o reflexions per a trobar millores en la comunicació amb la UI de *FracslandVR*. Els usuaris reals (nens i nenes en el nostre cas) podrien trobar que falta *feedback* visual al joc tot i que nosaltres pensem que tot es veu molt clar.

- **Integracions amb mes plataformes de VR**:

  L'addició de suport per a més plataformes i visors de VR a *FracslandVR* sembla ser una altra opció de millora per al joc en VR molt clara. Afegir al projecte integració amb el *plugin OpenVR* per a donar suport al visor Vive de *SteamVR* per exemple, donaria suport a les dos majors plataformes del mercat actual sense necessitat de moltes modificacions al projecte actual. Únicament, i ja que el sistema de càmeres és molt similar, s'hauria de canviar les funcions que accedeixen a la interfície de control dels comandaments i el visor (en el cas d'Oculus la classe `OVRInput`).

  Les noves eines recents com VRTK (*Virtual Reality Tool Kit*) o *NewtonVR*, que faciliten enormement aquest procés actuant a mode de llibreria *wrapper* per a tenir accés universal i transparent a les plataformes dels visors més comuns del mercat. En les seves *webs* trobem els *plugins* per a usar amb Unity i Unreal Engine juntament amb un extens apartat de documentació amb el nombre de SDK's suportats i funcionalitats disponibles i un canal de *Slack* [19] per a comunicar-se amb els desenvolupadors en cas de necessitat.

  L'ús d'entorns com *VRTK* o *NewtonVR* pot facilitar molt la integració de diversos SDKs alhora (actualment suporten el d'Oculus *OVRPlugin*, *OpenVR* i el de *Daydream* de Google). Aquestes eines poden oferir una interfície comú per a que els

[19] invitació al canal: vrtk-slack-invite.herokuapp.com

futurs treballs realitzats al projecte puguin visualitzar-se en més visors que el *Rift* d'Oculus.

De totes maneres, finalment es recomana esperar fins al final d'aquest any en curs per a intentar integrar més plataformes de VR a *FracslandVR*, ja que s'espera que durant l'estiu i fins a nadal, *Khronos Group* (creadors de *OpenGL*) millori molt el seu estàndard *OpenXR* i aconsegueixi més suport i tracció per part de la indústria. Això podria canviar completament la forma en què els fabricants dissenyen els SDKs dels visors i facilitar el treball futur dels desenvolupadors per a donar suport a molts dispositius diferents en un projecte com *FracslandVR*.

8 Referencies

[1] Muriel Ordoñez, Cristian: *Fracsland, joc seriós per aprendre fraccions*, Barcelona, Espanya: Universitat de Barcelona, 2014.

[2] Steuer, J: *Defining virtual reality: Dimensions determining telepresence. Journal of communication*, 1992.

[3] Fagerholt, E.; Lorentzon, M.: *Beyond the HUD - user interfaces for increased player immersion in FPS games*, 2009.

[4] Gamasutra: *User Interface design in Videogames*, gamasutra.com/blogs/AnthonyStonehouse/20140227/211823/User_interface_design_in_video_games, visitat: 20 Juny 2018.

[5] Oculus Developers: *Asynchronous Timewarp on Oculus Rift*, developer.oculus.com/blog/asynchronous-timewarp-on-oculus-rift, visitat: 10 Juny 2018.

[6] NVIDIA VRWorks: *Context Priority*, developer.nvidia.com/vrworks/headset/contextpriority, visitat: 10 Juny 2018.

[7] Oculus Developers: *Asynchronous Spacewarp*, developer.oculus.com/blog/asynchronous-spacewarp, visitat: 10 Juny 2018.

[8] Unity Manual: *Input for Oculus*, docs.unity3d.com/Manual/OculusControllers, visitat: 20 Juny 2018.

[9] RGB Schemes: *Dev Diaries - Oculus Touch And Finger Stuff (Part 1, 2 & Updates)*, rgbschemes.com/blog/oculus-touch-and-finger-stuff-part-1, visitat: 20 Juny 2018.

[10] VRTK: *Documentation*, vrtoolkit.readme.io/docs/summary, visitat: 20 Juny 2018.

[11] NewtonVR: *Readme*, newtonvr.readme.io, visitat: 20 Juny 2018.

[12] Toujouse, Carles: *Interaccions Conversacionals a un Joc Seriós*, Barcelona, Espanya: Universitat de Barcelona, 2018.

9 Annex

Annex A: Manual del Desenvolupador

En aquest document farem una breu explicació dels passos a seguir i l'equip mínim necessari per a executar i treballar amb el projecte de Unity *FracslandVR*. Tot el treball realitzat en aquest projecte es troba en un *subset* d'escenes del joc que s'anomenen igual que les originals amb el sufix "VR" a la carpeta `Assets/Resources/Scenes/`. Els *scripts* modificats i afegits de la versió VR i amb agents conversacionals de *Fracsland* es troben a la carpeta `Assets/Scripts/` i a la carpeta `Assets/ConversationalAgents/CustomScripts/`.

A.1 Requeriments mínims del projecte

Per a poder treballar amb el projecte de *FracslandVR*, primer ens hem d'assegurar que el nostre equip (*software* i *hardware*) siguin adients per a la VR, ja que els dispositius de VR no funcionaran correctament en cas de no complir-los. A part d'això, obligatòriament haurem de comptar amb el sistema de VR d'Oculus: el visor *Rift*, els seus comandaments *Touch* i un parell o fins i tot tres sensors.

En concret, hem de mirar de complir els requeriments d'Oculus i el seu sistema de VR:

Minimum System Requirements			
Chipset	**Graphics Card**	**RAM**	**Drive**
i5-4590	NVIDIA GTX 970/1060	8GB/16GB	3GB

Figure 56: Requeriments mínims del visor *Rift* d'Oculus.

Per altra banda, aquest projecte ha sigut desenvolupat i *tested* en les següents versions de *software*:

- **Unity**: versió 2017.3.1f1[20].
- **Oculus Utilities**: versio 1.23[21].
- **Avatar SDK**: versio 1.23.
- **Platform SDK**: versio 1.23.
- **ApiAi (DialogFlow)**: versió 1[22].
- **Windows Voice**: *wrapper* no-oficial de la funció de veu de Windows (`Microsoft Speech API`) per a Unity[23].

S'ha de tenir especial compte al canviar de versió de Unity, ja que cada versió del SDK d'Oculus pot introduir errors o *bugs* concrets a cada una d'aquestes. En els *links* de referència inclosos podem trobar informació sobre que versions de Unity són les provades per Oculus en cada *Release*.

Per altra banda, també caldrà tenir molta cura si es vol actualitzar la versió del SDK d'Oculus. Es recomana usar la mateixa versió de Unity i del SDK d'Oculus que la

[20] unity3d.com/es/unity/whats-new/unity-2017.3.1
[21] developer.oculus.com/documentation/unity/latest/concepts/release-1.23
[22] github.com/dialogflow/dialogflow-unity-client
[23] chadweisshaar.com/blog/2015/07/02/microsoft-speech-for-unity

referenciada aquí. Tot i això, si es vol actualitzar el *plugin* d'Oculus, s'anima a fer una neteja del SDK antic[24].

A.2 Passos a seguir per executar *FracslandVR*

En aquí desglossem els passos a seguir per a córrer i desenvolupar el projecte en Unity:

1. Assegura-nos de tenir l'equip *Oculus Rift + Oculus Touch* connectat a l'ordinador i amb el client d'Oculus obert.

2. Assegurar-nos també de disposar de connexió a *Internet*, ja que alguns d'aquests passos o alguns components del joc (com els **Agents Conversacionals** del joc) en fan ús.

3. Comprovar que no hi hagi *updates* del client d'Oculus o dels ***drivers* de la targeta gràfica**. Si n'hi ha, fer-los abans de començar.

4. Per a poder usar el servei de veu de Windows al projecte (l'usen els **agents conversacionals**), cal anar a la configuració del sistema i habilitar un seguit d'opcions. Primer, entrarem a l'apartat de `Privadesa` i sota el menú *Veu, inserció d'entrades manuscrites i escriptura* hem d'activar el servei de veu de Windows. Després, entrem a l'apartat de `Hora i Llengua` i un cop aquí anem al menú *Veu*. Aquí dins trobarem l'opció de canviar el llenguatge de detecció de Windows si desitgem (*Angles* per al nostre projecte) i marcarem la casella `Reconèixer accents no natius d'aquest idioma` per a facilitar la comprensió del nostre anglès.

5. Executar amb *Netbeans* el projecte inclòs al directori `Fracsland/FracslandServerFake/`. És el servidor fals fet per proveir al joc amb els *jsons* de dades que necessita en cada moment. Ha de quedar en execució per a poder accedir al joc. Hem d'assegurar-nos que els ports i *paths* d'aquest *server* en Java coincideixin amb els que hi ha al *script* `Constants.cs` al projecte de *Fracsland*[25].

6. Un cop està el *server* en Java corrent i el projecte de Unity (directori `Fracsland/Game/`) obert, hem de situar-nos a l'escena `Login` (**sense canviar les credencials d'inici que hi ha *per defecte***) i ja podrem començar a jugar a *FracslandVR*.

Observacions:

- Sembla haver-hi un error tant en Mac com en Windows 10 al mètode `ReloadEquippedItems()` (relacionat amb l'antiga plataforma web de *Fracsland*) que salta sempre al començament del joc (`InvalidCastException`). Comentar aquesta part del codi al *script* `StatsManager.cs` soluciona el problema i es pot jugar amb normalitat.

- Al carregar qualsevol escena de la versió VR de *Fracsland* (a partir de l'escena `TownVR`) salta un error (`NullReferenceException`) al *script* `OVRInputModule.cs`. Aquest error es deu al fet que aquest component està sempre buscant a l'escena un objecte (`OVRGazePointer`) per a dibuixar als elements de UI. En el nostre cas usem un làser a la mà per a apuntar i no necessitem un punter al UI, així que podem ignorar aquest error de moment.

[24] S'hauran d'eliminar els directoris `Assets/OVR`, `Assets/OVRInspector`, `Assets/OVRAvatar`, `Assets/OVRPlatform` per complet abans d'afegir la nova versió al projecte.

[25] Aquestes constants s'anomenen `API_PORT` i `API_BASE_URL` al *script* `Constants.cs`.

Annex B: Manual del Jugador (Com jugar a *FracslandVR*)

En aquest document expliquem com jugar i que ha de fer l'usuari a *FracslandVR*.

FracslandVR és un joc d'aventures en VR que et posa en la pell d'un nàufrag que acaba d'arribar a una misteriosa illa després de trencar el seu vaixell en una tempesta. Però no a una illa qualsevol, aquesta illa funciona amb la màgia de les **Fraccions**! Hauràs de parlar amb els diferents habitants de *Fracsland*, que et demanaran ajuda amb diferents tasques. A canvi, t'oferiran peces perquè arreglis el teu vaixell i així escapar de l'illa.

Per a poder jugar a *FracslandVR* són necessaris els comandaments d'Oculus *Touch*. Aquests comandaments t'ofereixen l'opció de controlar l'illa amb les teves mans! Per a explicar els controls amb facilitat, els dividim en dues categories:

- **Controls del joc**:

  Les possibilitats que ofereixen els mateixos botons dels comandaments dins del joc.

  - `A` o `X`: botó de selecció, equivalent al clic esquerre al ratolí de *Fracsland*. Punt 1 del diagrama.
  - `B` o `Y`: boto per a parlar amb l'ajudant de *FracslandVR*. Si et trobes prop de la tenda de l'illa, serveix per parlar amb el venedor. Punt 2 del diagrama.
  - *triggers* i *grippers*: aquests botons serveixen per a canviar de postura la mà al joc. Hi ha diferents postures que activen coses a *FracslandVR*: premer el *gripper* i no tocar el *trigger* d'una mà activarà el làser d'apuntat d'aquesta. Per altra banda, prémer els dos alhora (i si ens trobem en *God-Mode*) ens permet "agafar" el món i desplaçar-nos per aquest. Punt 3 del diagrama.
  - `L3` o *joystick* esquerre: boto que mostra o amaga el menú secundari amb el minimapa i els objectius (*overlay*). Dins la *First Person*, la palanca analògica ens servirà per a fer aparèixer la paràbola d'apuntat. Punt 4 del diagrama.
  - `R3` o *joystick* dret: boto que canvia de punt de vista al jugador. Si es troba a *God-Mode* canviarà a *First Person* i viceversa. Dins la *First Person*, la palanca analògica ens servirà per a fer aparèixer la paràbola d'apuntat. Punt 5 del diagrama.

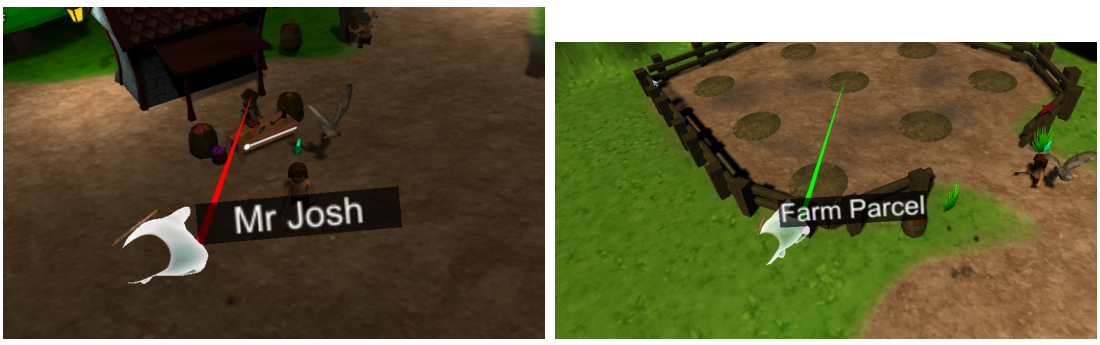

Figure 57: Imatge del làser d'apuntat de la mà al joc.

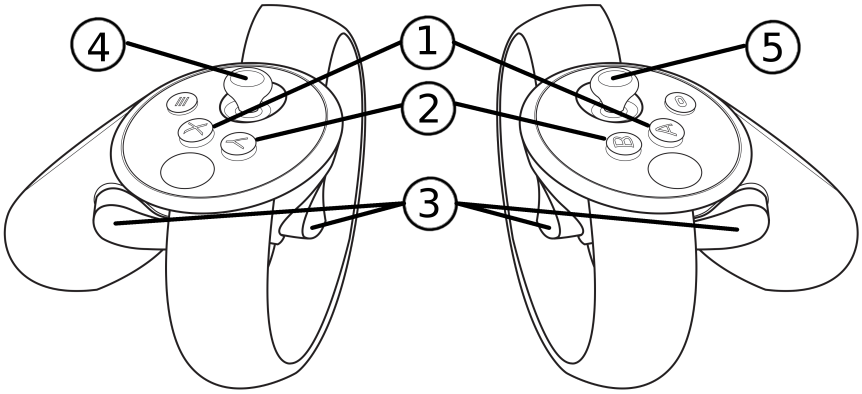

Figure 58: Imatge dels comandaments *Touch*, usats per jugar a *FracslandVR*.

- **Accions del joc**:

  Gestos o moviments dels comandaments que permeten interactuar amb coses dins del joc.

  - `Obrir Inventari`: es troba a la teva mà dreta! Solament has d'apropar-te la mà al camp de visió i orientar-la amb la palma cap amunt, el veuràs aparèixer a sobre d'aquesta!
  - `Obrir stats`: es troba a la teva mà esquerra! Solament has d'apropar-te la mà al camp de visió i orientar-la amb la palma cap amunt, el veuràs aparèixer a sobre d'aquesta!

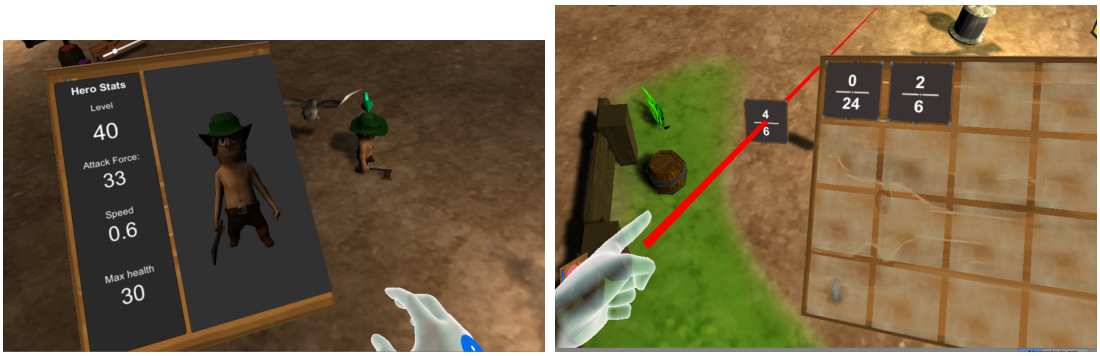

Figure 59: Imatge amb els *stats* del jugador a la mà (esquerra) i el *drag-and-drop* de l'Inventari (dreta).

  - *`Drag and Drop`*: combinant el gest per a fer aparèixer el làser (*gripper* pressionat i *trigger* sense contacte) amb el botó de selecció (`A` o `X`) podem seleccionar coses que hem recollit del món del nostre Inventari. Aquestes coses ens poden servir per resoldre missions i ajudar als personatges de l'illa.