ALMA MATER STUDIORUM – UNIVERSITÀ DI BOLOGNA
CAMPUS DI CESENA

DIPARTIMENTO DI INFORMATICA – SCIENZA E INGEGNERIA
Corso di Laurea in Ingegneria e Scienze Informatiche

REALTÀ MISTA MULTI-PIATTAFORMA: STUDIO E IMPLEMENTAZIONE DI UN LAYER DI INTEROPERABILITÀ NEL FRAMEWORK PER MONDI AUMENTATI MIRAGE

Elaborato in
SISTEMI EMBEDDED E INTERNET OF THINGS

Relatore
Prof. ALESSANDRO RICCI

Corelatore
Dott. Ing. ANGELO CROATTI

Presentata da
SAMUELE BURATTINI

Anno Accademico 2018 – 2019

alla mia Famiglia

Indice

Introduzione

Sin dalla loro invenzione, i computer hanno radicalmente cambiato il nostro stile di vita, entrando a far parte di una dimensione sempre più pervasiva della nostra realtà ogni qual volta che le capacità computazionali miglioravano e le dimensioni diminuivano.

Dai primi grandi calcolatori, al Personal Computer, la tecnologia è entrata nelle nostre case dandoci accesso, con la nascita di Internet, ad un mondo di informazioni impossibili da ottenere altrimenti. Poi, dagli anni 2000, con la rivoluzione innescata dagli Smartphone abbiamo ricevuto la possibilità di portare queste informazioni sempre con noi, sempre a portata di mano.

Ora i dispositivi mobili che stanno nelle nostre tasche ricordano solo esteriormente quelli introdotti circa dieci anni fa. Sono, infatti, molto più potenti e "smart" di quanto non fossero mai stati prima.

Tutto questo ci porta a concludere che forse si stia davvero avvicinando il tempo di una nuova rivoluzione.

La possibilità di avere quelli che sono sostanzialmente mini-computer, sufficientemente potenti per gesitre grandi quantità di dati e comunicare tramite diverse tecnologie di rete, ha infatti aperto le porte al mondo dell'Internet of Things (IoT) che si fonda sull'idea di rendere "intelligenti" gli oggetti che ci circondano permettendo loro di comunicare informazioni che li riguardano attraverso Internet.

In un mondo in cui la tecnologia diventa sempre più pervasiva, entrando in contatto diretto con il mondo fisico, molti si interrogano su come questa enorme quantità di dati possa essere effettivamente utilizzata in modo efficace.

Per alcuni la risposta sta nella realizzazione di un' interfaccia per l'accesso a questo nuovo tipo di conoscenza digitale del mondo fisico: la Mixed Reality.

La Mixed Reality è una tecnologia che permette, attraverso l'utilizzo di appositi dispositivi hardware, di visualizzare un layer digitale sovrapposto al mondo fisico. Per questo motivo è ideale per poter integrare la conoscenza del mondo reale, fornita da dispositivi embedded, in applicazioni di diversa natura.

Le sue radici affondano negli anni '70 del novecento, ma solo adesso siamo in grado di avere la potenza computazionale necessaria in dispositivi che siano

anche comodi da utilizzare per portare questa tecnologia fuori dai laboratori e permettergli di esprimere il suo reale potenziale.

Questo, unito al fatto che ora è possibile ottenere informazioni contestuali all'ambiente con relativa semplicità, è il presupposto che ha rilanciato la ricerca per la realizzazione di una Realtà Mista efficace ed intuitiva.

Tuttavia, per ottenere un risultato che porti effettivamente ad un miglioramento sperimentabile nella vita quotidiana, la strada è ancora lunga e sono molte le sfide aperte sia per realizzare le tecnologie abilitanti, sia per il design delle interazioni con il mondo virtuale, ed è ancora difficile prevedere quando queste problematiche saranno completamente risolte e potremmo beneficiare di una Mixed Reality davvero immersiva e coinvolgente.

Una cosa certa, però, è che se la Realtà Mista entrerà a far parte della nostra quotidianità, allora dovrà tenere conto del grandissimo numero di utenti che popoleranno il mondo virtuale. Non solo, gli utenti dovranno avere la possibilità di interagire tra loro e con gli oggetti virtuali in modo collaborativo. Dovranno avere, sostanzialmente, la possibilità di condividere, così come condividono il mondo fisico, anche quello virtuale.

Ma se è vero che saranno molti gli utenti di questa nuova realtà, allora è più che probabile che diverse persone utilizzeranno diversi dispositivi. Questo potrebbe portare ad una esplosione di piattaforme, come avvenuto per gli smartphone, con la differenza che queste differenze potrebbero essere ancora più forti dal momento che esistono molte categorie di prodotti, da quelli indossabili, per esempio, a quelli portabili.

Questo ci porta a concludere che lo sviluppo di un'infrastruttura software in grado di permettere la condivisione della Realtà Mista debba necessariamente prevedere un supporto multi piattaforma per poter effettivamente raggiungere tutti i futuri utenti.

In questa tesi si esplora questa prospettiva, prendendo in analisi l'idea degli Augmented Worlds[8] come modello di un possibile scenario di Realtà Mista condivisa ed aggiungendo all'infrastruttura un layer di interoperabilità che abiliti allo sviluppo di applicazioni per diverse piattaforme che condividano lo stesso mondo aumentato.

Dopo aver approfondito i concetti base della Realtà Mista, e analizzato lo stato dell'arte delle tecnologie esistenti, con particolare attenzione agli aspetti di condivisione, il lavoro pratico si è concentrato sull'integrazione nel modello di un supporto multi-piattaforma prendendo come esempi un Head Mounted Display e una piattaforma per Realtà Aumentata su smartphone.

Questo ha portato alla realizzazione di un'applicazione, basata su un caso di studio reale, che permette agli utenti di visualizzare e leggere cooperativamente la copia digitale di un libro storico.

Capitolo 1

Realtà Mista: una panoramica

In questo capitolo viene fatta una panoramica sulla Realtà Aumentata, dalle premesse teoriche che hanno portato allo sviluppo di questa tecnologia, agli aspetti tecnici di come viene realizzata allo stato attuale dello sviluppo. Inoltre vengono definite le caratteristiche che contraddistinguono le applicazioni in Realtà Mista, fulcro di questa tesi. Vengono infine introdotti e illustrati alcuni concetti chiave che sono utili per la trattazione degli argomenti a seguire.

1.1 Aumentare la realtà

L'idea di base che sta dietro tutte le tecnologie di questo tipo è quella di far percepire la realtà in modo differente, sfruttare quindi dispositivi hardware in grado di *aumentare* ciò che viene visto dai nostri occhi con oggetti e spazi virtuali generati grazie alla computer-grapichs.

Questo consente sostanzialmente di arricchire il mondo reale in cui siamo immersi quotidianamente con un'insieme di altre informazioni che possono descrivere in maniera migliore ciò che stiamo osservando, oppure che possono introdurre elementi nuovi e creare una dimensione alternativa indipendente, ma allo stesso tempo strettamente legata con quella fisica che ci circonda.

A seconda dell'uso che ne viene fatta la Realtà Aumentata diventa, quindi, un'interfaccia per l'accesso alle informazioni o un'esperienza immersiva in un mondo che è arricchito da elementi in grado di superare i limiti del nostro mondo, pur manifestandosi ai nostri occhi come realistici.

Oltre all'aspetto più ludico della tecnologia, la Realtà Aumentata si pone quindi come una forma avanzata di accesso alla conoscenza [10]: un'interfaccia diversa da quelle a cui ci siamo abituati nel corso della nostra esperienza con i computer perché incredibilmente più naturale e pratica e soprattutto priva del fattore limitante dello schermo che appiattisce e simula in due dimensioni il complesso mondo tridimensionale in cui siamo abituati a muoverci.

Questo apre le porte a diversi scenari, dai più semplici, come ad esempio gli Head Up Display (HUD) per poter visualizzare informazioni a margine della nostra visuale, come nel caso di molti videogiochi in prima persona, a scenari ben più articolati in cui la conoscenza viene mostrata in modo contestuale a quello che stiamo facendo ad esempio indicandoci come posizionarci e muoverci per completare un determinato compito.

Non dobbiamo dimenticare, infatti, che la Realtà Aumentata offre molto più che una semplice visualizzazione delle informazioni sul mondo. La potenza di questa tecnologia risiede proprio nella sua versatilità e nella capacità di creare esperienze che siano, prima di tutto, immersive per chi le vive.

Nella creazione di queste esperienze è importante identificare il grado di virtualizzazione dell'ambiente che viene messo in gioco. È possibile, infatti, immaginare un *virtuality continuum*[9] così come definito da P. Milgram e F. Kishino, ai cui estremi troviamo il mondo fisico ed il mondo completamente virtuale, ovvero una realtà completamente generata dal computer che non ha necessariamente legami con il mondo che conosciamo (Figura 1.1).

Sfruttando l'astrazione del continuum virtuale si può dare una definizione di Realtà Mista come tutto ciò che si trova tra gli estremi del continuum stesso. Di fatto quindi con il termine generico Mixed Reality si identifica ogni esperienza in cui vengono percepiti sia il mondo fisico che il mondo virtuale con diversi gradi di sovrapposizione.

Andiamo a vedere perché questa definizione, sebbene accettata da molti, è piuttosto generica e quindi imprecisa.

Figura 1.1: La rappresentazione del continuum virtuale come presentato da P. Milgram e F. Kishino

1.2 Realtà Mista: una definizione

Nell'uso comune, i termini Realtà Mista e Realtà Aumentata vengono utilizzati in modo intercambiabile. È bene quindi approfondire la differenza tra queste due definizioni e chiarire l'interpretazione che ne viene data in questa trattazione.

Con il termine Realtà Aumentata vengono normalmente identificate tutte le applicazioni che in qualche modo sovrappongono alla visualizzazione del mondo reale elementi digitali.

La Realtà Mista è invece definita come un sottoinsieme delle applicazioni in Realtà Aumentata in cui si ha una comprensione specifica dell'ambiente fisico e della posizione degli oggetti sia virtuali che non. In una applicazione di Realtà Mista l'utente può navigare in modo fluido simultaneamente all'interno del mondo fisico e di quello virtuale che sono, non solo sovrapposti, ma mescolati in modo da comportarsi coerentemente con quello che ci aspettiamo in termini di prospettiva ed interazione [10].

Riferendoci sempre al continuum (Figura 1.1), vediamo quindi come, secondo la definizione che viene adottata per questa trattazione, la Realtà Mista invece che comprendere tutto ciò che si trova tra gli estremi è meglio identificabile come il punto medio di questo spettro in cui le due realtà sono mescolate (mixed) al punto tale che diventa difficile distinguere gli elementi di una e l'altra.

Per ulteriore chiarezza è bene delineare anche la differenza tra Realtà Aumentata e realtà virtuale che spesso vengono erroneamente accumunate negli articoli divulgativi.

La confusione nasce dal fatto che queste tecnologie sono vicine per il tipo di dispositivi hardware utilizzati, dal momento che necessitano di attrezzature simili quali visori, accelerometri, giroscopi, anche se, per realizzare la Realtà Aumentata, è necessaria una parte di strumentazione ottica più complessa che la rende più costosa e difficile.

La sostanziale differenza, però, sta nel tipo di esperienza che viene vissuta dall'utente, in quanto, nel caso della realtà virtuale, viene perso completamente il contatto con la realtà fisica e ci si immerge in un mondo virtuale generato completamente dalla computer-graphics al punto che alcuni utenti lamentano una sensazione di nausea per la perdita di riferimenti reali. (Figura 1.2)

1.3 Visualizzare i contenuti digitali: gli ologrammi

Avendo spiegato il modo in cui la Realtà Aumentata va a sovrapporre un contenuto digitale sul mondo fisico, andiamo a definire l'astrazione che viene usata per rappresentare questo contenuto: gli ologrammi.

Il termine che è stato scelto per identificare gli elementi costituenti del mondo virtuale è *ologrammi*, probabilmente a richiamo delle iconiche tecnologie in grado di proiettare immagini tridimensionali nello spazio viste in noti film di fantascienza come ad esempio Star Wars.

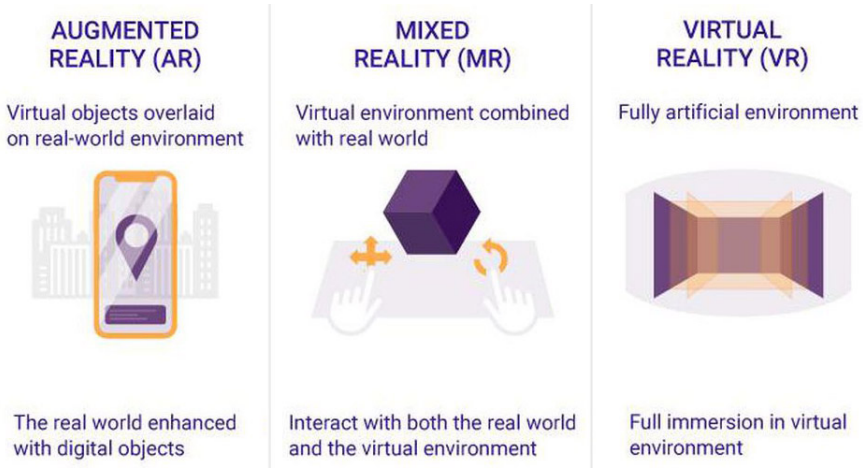

Figura 1.2: Le differenze tra Realtà Aumentata, mista e virtuale così come intese in questa trattazione.

Di certo, la differenza principale con gli ologrammi di quel tipo è che nelle applicazioni in Realtà Aumentata per visualizzare gli elementi virtuali è necessario avere un supporto dal momento che questi non possono essere percepiti ad occhio nudo. Sono infatti generati grazie alla computer-graphics e sovrapposti in modo digitale, o ottico nel caso di alcuni visori, al mondo fisico.

Manca quindi una definizione chiara e precisa di cosa sia effettivamente un ologramma. Per semplicità possiamo definirli come un'entità virtuale statica o dinamica che viene immersa nel mondo fisico tramite una rappresentazione tridimensionale.

Gli ologrammi possono quindi avere un proprio comportamento, e possono essere manipolati, a seconda della tecnologia abilitante, in modo più o meno realistico.

1.4 Tecniche software per la Realtà Mista

Ora che abbiamo definito in modo chiaro che cosa ci aspettiamo da una applicazione in Realtà Mista, vediamo come è possibile realizzarla con la strumentazione Hardware e Software disponibile attualmente.

Possiamo suddividere il ciclo di elaborazione di un'applicazione in Realtà Aumentata in due macro-compiti[5]:

1. L'applicazione deve determinare lo stato attuale del mondo fisico e determinare lo stato attuale del mondo virtuale.

2. L'applicazione deve visualizzare il mondo virtuale in registrazione con il mondo reale in un modo da far percepire ai partecipanti gli elementi del mondo virtuale come parte del mondo fisico.

Per aumentare la granularità delle operazioni, e rendere più chiaro tecnicamente che cosa accade ad ogni update dell'applicazione, possiamo raffinare gli step precedenti suddividendo ognuno dei due in altri due passi, ottenendo così il seguente ciclo di esecuzione:

1. Rilevazione e comprensione della geometria dello spazio
2. Identificazione e risposta all'interazione da parte dell'utente
3. Comprensione della posizione dell'osservatore in relazione allo spazio e conseguente calcolo della porzione degli ologrammi da visualizzare
4. Rendering grafico della scena per sovrapporre gli ologrammi al mondo reale

Negli ultimi anni, grazie alla maggiore potenza computazionale portabile su dispositivi di dimensioni ridotte e alle tecniche di elaborazione delle immagini sempre più avanzate, si sono ottenuti grandi progressi nelle fasi di identificazione della posizione degli ologrammi e della comprensione dello spazio in cui l'utente si muove.

Esistono due approcci differenti per tentare di risolvere il problema della registrazione spaziale. Andiamo quindi ad analizzarli nel dettaglio.

Riconoscimento marker-based

Si parla di Realtà Aumentata *marker-based* quando vengono applicati nello spazio degli appositi elementi per, appunto, marcare la posizione di un oggetto virtuale nel mondo fisico.

Questi *marker* non sono altro che immagini 2D asimmetriche, solitamente in bianco e nero in modo da renderne più semplice il riconoscimento e l'elaborazione da parte degli algoritmi di computer-vision.

Il sistema, in questo modo, ottiene una comprensione dello spazio in base all'orientamento e posizione del marker visualizzato e riesce a ricostruire un sistema di coordinate tridimensionale con cui lavorare e in base al quale posizionare gli ologrammi. Il sistema di tracking si basa quindi su che cosa "vede"[10].

Questo significa anche che, a meno di non adottare approcci ibridi, una volta che il marker scompare dall'inquadratura o non viene più riconosciuto

anche gli ologrammi ad esso legati scompaiono di conseguenza dal momento che il sistema non è più in grado di posizionarli correttamente.

Framework moderni permettono di inserire nel database di marker riconosciuti dall'applicazione anche immagini più complesse, trasformando il concetto di marker in un'*augmented image*, così come viene definita da Google nel suo framework ARCore, che permette così di operare con immagini più gradevoli che nascondano la funzione di riferimento spaziale che svolgono (Figura 1.3).

Tecniche più avanzate di riconoscimento prevedono la possibilità di utilizzare come marker anche oggetti tridimensionali: un esempio commerciale è quello realizzato recentemente da LEGO® con i loro set *Hidden Side* che utilizzano come marker proprio una costruzione tridimensionale creata con i celebri mattoncini per aumentare l'esperienza di un giocatore attraverso lo schermo del proprio smartphone.

Con questi metodi sempre più naturali l'approccio marker-based resta una delle opzioni più solide in termini di posizionamento e aderenza tra elementi digitali e reali. È chiaro, però, che non è possibile adattare questo sistema a tutti i casi d'uso, specie se consideriamo contesti ampi come ad esempio spazi outdoor in cui potrebbe risultare difficile posizionare dei marker di qualsiasi natura.

Figura 1.3: Augmented Images utilizzate come marker per mini-giochi in Realtà Aumentata su Nintendo3DS (2011)

Marker-less Mixed Reality

La tecnica sviluppata per risolvere il problema del tracking della posizione del dispositivo in un ambiente sconosciuto è la Simultaneous Localization and Mapping (SLAM).

Questa tencnica è sicuramente costosa in termini computazionali perché il dispositivo deve riuscire, interpretando i dati dei sensori inerziali, accelerometri e giroscopi, a tracciare una mappa tridimensionale dell'ambiente e, allo stesso tempo, localizzare se stesso all'interno di quella mappa. Tuttavia, si riescono ad ottenere risultati sufficientemente precisi in ambienti completamente sconosciuti quindi la SLAM è attualmente la tecnica utilizzata in quasi tutti i moderni dispositivi di supporto alla Realtà Aumentata, dal momento che non richiede nessuna conoscenza pregressa dell'ambiente, cosa che la rende estremamente versatile.

Basandosi interamente sulla fotocamera e i sensori del dispositivo il tracking è purtroppo molto sensibile a movimenti bruschi e alla perdita di nitidezza nelle immagini, azioni che necessitano una relocalizzazione approssimativa del dispositivo dal momento che viene perso il riferimento alla mappa generata, tuttavia i sistemi moderni riescono a recuperare in modo rapido il tracciamento iniziale quindi questo non è un problema particolarmente grave se l'utente utilizza i dispositivi nel modo corretto.

Nel creare una mappatura dell'ambiente la SLAM si occupa anche di restituire all'applicazione una rappresentazione tridimensionale dell'ambiente (detta in gergo tecnico *mesh*) formata da un'insieme di poligoni in modo da permettere all'applicazione di utilizzarla per inserire gli ologrammi nella scena e rendere il loro comportamento coerente con gli oggetti reali presenti, come ad esempio pavimento e pareti. Questo è fondamentale per riuscire a creare applicazioni in Realtà Mista che, come abbiamo visto, necessitano di una comprensione dello spazio molto precisa.

Per essere efficace la SLAM si affida ad una tecnica denominata Natural Feature Tracking (NFT) che consiste nell'individuare dei "punti di interesse" all'interno del mondo fisico e sfruttare la geometria di quei punti per identificare superfici e forme e tracciarne la posizione nello spazio.

Solitamente per identificare questi punti vengono utilizzate tecniche di riconoscimento degli angoli derivate dalla computer-vision.

È importante tenere a mente che la luminosità degli ambienti assume un ruolo fondamentale per il corretto funzionamento degli algoritmi di tracking. Questo rende le applicazioni suscettibili a variazioni di luce naturale in contesti outdoor o anche ad ambienti illuminati in modo troppo omogeneo in contesti indoor come ad esempio una stanza completamente bianca che non offre riferimenti sufficienti all'algoritmo per comprendere la geometria dello spazio.

Per la questione delle performance algoritmi di SLAM efficaci sono stati elaborati per essere supportati in tempi rapidi anche dagli smartphone, la presenza di un hardware dedicato permette di ridurre sicuramente il tempo di computazione e di avere un tracking più accurato al prezzo di un maggior costo del dispositivo, come nel caso dei visori.

1.5 Tecnologie per la Realtà Mista

In questa sezione vengono mostrate le principali tecnologie che supportano lo sviluppo di esperienze in Realtà Mista in modo da dare un'idea del panorama esistente, delle funzionalità principali e degli strumenti necessari per sviluppare queste applicazioni.

Google ARCore e Apple ARKit

ARCore e ARKit sono le tecnologie sviluppate rispettivamente da Google e Apple Inc. per supportare applicazioni in Realtà Aumentata sui dispositivi mobili Android e iOS.

Il funzionamento di entrambi è basato sullo sfruttamento della sensoristica presente nei dispositivi per fare tracking dell'ambiente e stimare la posizione nello spazio (SLAM - Sezione 1.4). Lo spazio viene ricostruito come un insieme di piani orizzontali o verticali identificando dei raggruppamenti di feature point che vengono riconosciuti come allineati su una superfice comune.

L'interazione che l'utente ha nei confronti dell'applicazione è gestita tramite l'operazione di raycasting, ovvero la proiezione di un "raggio" a partire dal dispositivo che va ad individuare un punto su uno dei piani riconosciuti in base all'orientamento del dispositivo stesso. Questo permette di simulare un'interazione tramite il touchscreen, proiettando il punto toccato nello schermo nello spazio tridimensionale.

L'elenco delle feature principali include il riconoscimento di immagini da utilizzare come marker per il posizionamento degli oggetti e la stima della luminosità media dell'ambiente per uniformare gli oggetti alla realtà in cui vengono immersi.

Recentemente ARKit ha rilasciato la sua ultima versione integrando alcuni elementi di intelligenza artificiale per il riconoscimento delle persone nella scena in modo da gestire in modo più efficace l'occlusione degli elementi virtuali e per riconoscere il movimento delle persone inquadrate.

Microsoft Hololens 2

Dal punto di vista tecnologico Microsoft al momento rappresenta il top di gamma nella produzione di Head Mounted Display (HMD) ovvero visori per Realtà Mista indossabili. (Figura 1.4 a)

Inoltre lato software è stato sviluppato lo stack Windows Mixed Reality che supporta applicazioni di Realtà Aumentata per visori anche di altri produttori.

Il principale punto di forza è Hololens 2: un HMD con hardware dedicato per permettergli di svolgere sia le operazioni di ricostruzione dinamica dell'am-

biente, che il rendering, senza la necessità di essere connesso ad un computer. Questa caratteristica, unita ad un dispositivo relativamente leggero ed al supporto per il riconoscimento vocale, della posizione degli occhi e delle gesture delle mani rende Hololens il display più vicino a quelli che potrebbero popolare la nostra quotidianità: allo stesso tempo potente ed intuitivo.

Ovviamente, per poter essere utilizzabile senza cavi il campo visivo supportato dal processore grafico è di circa 52°, indubbiamente una finestra minore rispetto alla normale vista umana, ma comunque un risultato più che accettabile per un dispositivo che mette la sua forza nella comodità di utilizzo libero e nell'ergonomia della manipolazione degli ologrammi.

Meta Developement Kit 2

Il visore Meta2 è un HMD sviluppato dalla compagnia americana indipendente Meta(Figura 1.4 b). Il visore è cablato e deve essere collegato ad un PC con un hardware adatto per supportare le complesse operazioni di interpretazione dei sensori e rendering grafico degli ologrammi da proiettare sulle lenti semi-trasparenti.

Oltre ad avere un set di sensori di tutto rispetto, che gli permettono di effettuare il tracking dell'ambiente tramite SLAM, Meta2 offre un campo visivo di circa 90°, molto più grande della maggior parte dei competitor, permettendo così di avere una visione incredibilmente naturale per l'utente che non percepisce la limitazione se non in particolari situazioni.

Questo è un aspetto molto forte che permette di pagare il prezzo di una limitata mobilità e rende il visore Meta2 un ottimo laboratorio per applicazioni in Realtà Mista che non necessitano una libertà di movimento in grandi spazi.

Sarà infatti il dispositivo che verrà utilizzato per la realizzazione della parte pratica di questa tesi sull'integrazione delle diverse piattaforme come esempio di HMD.

Magic Leap One

Degno di nota nella scena dei visori per Realtà Mista, Magic Leap One è una realtà indipendente che mira a realizzare un HMD dal design decisamente non convenzionale (Figura 1.4 c). Il dispositivo è composto da due apparecchi, un visore realizzato a forma di occhiale con due lenti separate, cablato ad una centrale di controllo leggera e indossabile.

Magic Leap supporta una forma basilare di riconoscimento delle gesture e anche un *eye-tracking*. Viene fornito con un telecomando per rendere più precisa l'interazione in certi scenari a patto di perdere in immersività.

Il design di Magic Leap è tutto orientato ad una comodità di utilizzo prolungato, cercando di rmuovere peso dalla parte indossabile sulla testa che, nel caso degli altri visori, può essere stancante per i muscoli del collo. Il campo visivo risultante da questo particolare design è simile a quello di Hololens 2.

La nota negativa che alcuni utenti lamentano è il fatto che le lenti di Magic Leap sono oscurate sensibilmente per migliorare la visibilità degli ologrammi, non particolarmente luminosi a causa del setup leggero, Questo genera a volte una separazione troppo forte dal mondo fisico e una perdita in immersività.

(a) Hololens 2 (b) Meta 2 (c) Magic Leap One

Figura 1.4: Gli Head Mounted Display a confronto: da sinistra, Hololens (a), Meta2 (b) e Magic Leap One (c)

Unity3D

Unity è un Game Engine sviluppato per realizzare videogiochi in 2D e 3D. Offre agli sviluppatori un ambiente di sviluppo integrato (IDE) molto intuitivo da utilizzare e supporta numerose piattaforme, motivo per cui ha preso piede facilmente.

All'interno dello stesso ambiente viene integrata la parte grafica con la gestione del comportamento implementato tramite Scripting principalmente in C# .

In particolare per quel che riguarda Unity3D la semplicità di racchiudere la manipolazione dei modelli tridimensionali con tanto di personalizzazione di texture e gestione dell'illuminazione in un ambiente di sviluppo unico, che integri anche la logica applicativa, lo ha reso uno strumento potente per lo sviluppo rapido di applicazioni in Realtà Aumentata e mista pur non essendo inizialmente nato in questa prospettiva.

Infatti, quasi tutte le tecnologie elencate forniscono un pacchetto di *Assets* utilizzabili all'interno dell'engine.

Dal momento che sarà lo strumento utilizzato nello sviluppo di questa tesi approfondiremo alcuni concetti nella sezione 4.2.

1.6 Aspetti critici e sfide aperte

Uno dei requisiti fondamentali per far sì che le esperienze in Realtà Mista siano accettate completamente dagli utenti è l'integrazione fluida degli ologrammi nel mondo reale e, per poter ottenere questo risultato, è necessario che gli oggetti reali e virtuali interagiscano tra loro in modo realistico.

Le interazioni tra gli oggetti nella Realtà Mista possono essere suddivise in due categorie: visuali e fisiche. Gli aspetti visuali che vediamo nel mondo fisico e ci aspettiamo anche in quello virtuale coinvolgono la riflessione della luce, la creazione di ombre, l'occlusione ecc. Gli aspetti fisici riguardano invece il rispetto delle leggi della cinematica come ad esempio ed una adeguata risposta alle collisioni. [4]

Queste interazioni sono ancora per la maggior parte in via di sviluppo, in quanto gli algoritmi necessari per calcolare il comportamento fisico in modo preciso sono indubbiamente costosi in termini computazionali e richiedono una comprensione del mondo fisico molto più accurata di quanto attualmente siamo in grado di estrapolare grazie alle tecniche viste nella sezione 1.4.

Questi aspetti restano attualmente la principale sfida aperta per quel che riguarda la creazione di esperienze che siano davvero immersive per l'utente e per raggiungere una fusione realistica dei due mondi virtuale e reale.

La loro realizzazione è a dir poco fondamentale per riuscire a raggiungere risultati credibili e sufficientemente immersivi e per dare all'utente la sensazione che gli oggetti virtuali non siano semplicemente "incollati" sulla realtà fisica.

La ricerca si sta muovendo, infatti, in questa direzione. Sia ARCore e ARKit supportano ad esempio un meccanismo di *light estimation* che si occupa di illuminare coerentemente gli oggetti in base alla luminosità media rilevata dal sensore della fotocamera ma non supportano ad esempio la proiezione delle ombre che a livello cognitivo è indispensabile per rendere più precisa la collocazione nello spazio di un oggetto.

Un altro aspetto chiave ancora quasi del tutto assente nei supporti allo sviluppo di Realtà Mista è la gestione dell'occlusione tra oggetti reali e virtuali. Nonostante in letteratura siano presenti degli studi anche datati a riguardo, [4] siamo ancora ben lontani dall'ottenere risultati soddisfacenti per problemi legati alle performance di calcolo ed al risultato visivo. Come è facile intuire se si vuole ottenere un risultato visivo migliore c'è bisogno di più tempo di calcolo e la latenza è il parametro da tenere maggiormente sotto controllo nello sviluppo di una tecnologia di questo tipo.

Ottenere una mappa della profondità da un'immagine è un compito estremamente complesso e farlo su ogni frame in tempo quasi reale è sicuramente un problema di difficile risoluzione.

La nuova versione di ARKit 3 è stata lanciata con la grande novità di essere in grado di rilevare le persone inquadrate e occludere in modo corretto gli oggetti virtuali quando altre persone sono presenti utilizzando degli algoritmi di machine learning per rilevarle nella scena e stimarne la profondità.

Questo è indubbiamente un passo avanti ed un segno che presto si potranno raggiungere risultati di realismo ben diversi da quelli a cui siamo abituati ora, avvicinandosi sempre più ad una vera e propria Realtà Mista in cui virtuale e reale sono quasi indistinguibili.

Capitolo 2

Realtà Mista multi-utente ed interoperabilità

In questo capitolo si esplorano le prospettive in termini di condivisione multi utente della Realtà Mista e si cerca di dimostrare perché questo aspetto, che è stato fin'ora messo da parte, dovrebbe invece assumere un ruolo centrale per lo sviluppo futuro di questa tecnologia. Attraverso l'esplorazione delle ricerche in merito e dello stato dell'arte delle tecnologie abilitanti, l'obiettivo è quello di rendere evidente come, al momento, l'approccio a questo problema sia stato indirizzato come una mera estensione degli strumenti di sviluppo per le varie piattaforme, mentre dovrebbe essere completamente ripensato su un diverso livello di astrazione in modo da creare soluzioni più robuste e versatili.

2.1 Motivazioni

La maggior parte delle applicazioni attuali in Realtà Aumentata e mista sono create attorno ad un singolo utente che visualizza gli ologrammi ed interagisce con essi.

La scelta di creare applicazioni in questo modo deriva sicuramente da una facilità di realizzazione perché nel progettare e realizzare esperienze che siano condivise subentrano problemi, sia di design, che tecnici, non banali.

È chiaro però che, con il progredire di questa tecnologia, la Realtà Mista non potrà rimanere slegata dal fatto che, come il mondo fisico, anche il mondo virtuale sarà popolato da più persone contemporaneamente.

Già allo stato attuale delle tecnologie la Realtà Mista si sta avvicinando ad alcuni ambiti lavorativi quali l'architettura, il design e la medicina. In un ambiente lavorativo di per sè collaborativo, anche la tecnologia si deve adattare per mostrare a tutti la stessa visione e permettere a tutti di partecipare atti-

vamente. È importante che svolga quindi un ruolo che sia quanto più inclusivo possibile.

Un altro concetto importante sul quale si sta concentrando la ricerca è quello della telepresenza, la possibilità di essere virtualmente presenti in uno spazio molto lontano da quello in cui ci troviamo. Di fatto il corrispettivo in Realtà Mista di una videochiamata, la telepresenza apre le porte a scenari molto più ricchi in quanto attraverso la Realtà Mista l'utente virtuale sarebbe in grado non solo di parlare e mostrarsi agli altri ma anche di interagire con gli oggetti virtuali. Aziende specializzate come Spatial[1] stanno iniziando a presentare delle soluzioni per creare spazi di condivisione in cui collaborare anche in telepresenza(Figura 2.1).

Figura 2.1: Una immagine dimostrativa degli spazi creativi realizzabili attraverso Spatial. È possibile notare un membro in telepresenza attraverso un avatar.

Scenari di questo tipo stanno già emergendo ed è bene focalizzare la ricerca sugli aspetti abilitanti per la creazione di esperienze condivise. D'altronde se il futuro verso il quale ci stiamo muovendo comprende la Realtà Aumentata come tecnologia pervasiva nel nostro stile di vita è bene iniziare da subito a muoversi nella direzione migliore.

Non dobbiamo infatti tralasciare l'impatto sociale che una tecnologia potente quanto la Realtà Mista può avere nella nostra quotidianità.

[1] `https://spatial.is/`

Nel tentativo di riassumere questi problemi e regolamentare la Realtà Mista, in un articolo del 2016 John Rousseau, designer presso Artefact [2], ha proposto tre leggi [11] richiamando le famose leggi di Asimov sulla robotica[1] che, in modo conciso e più filosofico che pratico, cercano di regolamentare i limiti etici della tecnologia aumentante.

Le "leggi" proposte da Rousseau sono le seguenti:

1. *Mixed reality must enhance our capacity for mindful attention.*
2. *Mixed reality must embody a shared human experience.*
3. *Mixed reality must respect boundaries between commerce and data.*

La seconda, in particolare, definisce l'esigenza intrinseca di una condivisione all'interno della nuova realtà. Per spiegarla con le parole di Rousseau stesso, la Realtà Mista può permetterci di vedere mondi personalizzati per ciascuno di noi e questo potrebbe far perdere il contatto tra gli abitanti della realtà fisica. Per contrastare ciò, dovremo creare nuovi modelli ed esperienze progettati per facilitare la scoperta, la condivisione e la connessione umana, sia virtualmente che nel mondo reale.[11]

Messe in luce le motivazioni che giustificano l'interesse dell'esplorazione di una tecnologia di Realtà Mista condivisa, questa tesi si concentra sugli aspetti puramente tecnici della realizzazione di una condivisione multi-piattaforma, tenendo bene a mente che la dimensione condivisa porta con sè una serie di altri problemi legati al design delle esperienze in termini di effettiva collaborazione e ad aspetti di sicurezza, privacy ed etica che non verranno trattati, ma che stanno già iniziando ad essere studiati in altre ricerche.[12]

2.2 Condivisione ed interoperabilità

Abbiamo parlato di come la condivisione tra più utenti sia determinante per il successo e per migliorare l'impatto sulla nostra vita di queste tecnologie, ma, attualmente, le esperienze in Realtà Aumentata che ci vengono proposte, sulla base delle tecnologie esistenti, sono quasi tutte per un singolo dispositivo ed un singolo utente.

Questo è dovuto al fatto che, per realizzare esperienze multi utente è necessario condividere il mondo digitale e fare in modo che ogni utente possa osservarlo dalla propria prospettiva ed, eventualmente, interagire con esso rendendo le proprie azioni efficaci anche per tutti gli altri utenti.

Per fare un esempio intuitivo, se volessimo realizzare una condivisone basilare tra due utenti, potremmo far loro inquadrare lo stesso marker in modo

[2] `https://www.artefactgroup.com`

da permettere a ciascuno di vedere l'ologramma dalla propria prospettiva. In realtà, però vedrebbero due copie dello stesso ologramma e la manipolazione di uno non verrebbe visualizzata dall'altro creando una situazione di inconsistenza.

Come affermato nello studio volto alla realizzazione di Studierstube[13], un sistema in grado di visualizzare dati scientifici tridimensionali per più spettatori simultanei all'interno di una stanza attraverso dei visori per Realtà Aumentata, un approccio client-server può risolvere alcuni dei principali problemi di condivisione.

Infatti, nella soluzione adottata nel lavoro iniziato nel 1998, un server mantiene aggiornato un database contenente tutti i dati da visualizzare e ogni utente ne riceve una copia tramite un'applicazione client che poi si occupa di renderizzare il contenuto. Per mantenere la consistenza della visualizzazione tra gli utenti nel corso dell'evoluzione degli oggetti, dal momento che esistono più copie locali dello stesso oggetto ogni variazione deve essere segnalata al server che si occupa di propagare il messaggio contente il cambiamento di stato agli altri utenti.

In questo prototipo gli aspetti di tracking dell'ambiente erano, volutamente, ridotti al minimo e lo scenario ipotizzato era estremamente statico in modo da potersi concentrare unicamente sugli aspetti propagazione dei cambiamenti in modo consistente tra gli utenti.

Tuttavia questa idea, sebbene possa sembrare estremamente basilare, è di fatto il minimo indispensabile per poter creare un'esperienza che sia realmente condivisa.

Avendo definito nella sezione 1.4 un ciclo di operazioni che una applicazione in Realtà Aumentata deve svolgere per funzionare correttamente, vediamo come è possibile estenderlo ad uno scenario condiviso alla luce di queste considerazioni:

1. Rilevazione e comprensione della geometria dello spazio
2. **Allineamento del mondo digitale condiviso con il mondo fisico rilevato**
3. Identificazione e risposta all'interazione da parte dell'utente
4. **Aggiornamento del mondo digitale condiviso in base all'interazione ricevuta**
5. Comprensione della posizione dell'osservatore in relazione allo spazio e conseguente calcolo della porzione degli ologrammi da visualizzare
6. Rendering grafico della scena per sovrapporre gli ologrammi al mondo reale

I punti in grassetto sono stati evidenziati in quanto racchiudono l'essenza di ciò che è necessario realizzare a livello condiviso, mentre le altre azioni possono essere effettuate dal dispositivo dell'utente.

Se sul piano teorico il raggiungimento di questi obiettivi sembra relativamente semplice, dal punto di vista pratico emergono una serie di problematiche che ostacolano la realizzazione di uno scenario che sia effettivamente condiviso.

Non esistono infatti protocolli ad hoc per il trasferimento delle grandi quantità di dati necessarie ad esprimere la geometria tridimensionale del mondo rilevato tramite i sensori e degli ologrammi da rappresentare coerentemente in esso.

Inoltre, nelle applicazioni in Realtà Aumentata è stato già ampiamente certificato quanto la latenza sia critica [2] per mantenere l'immersività dell'esperienza e la corretta registrazione spazio-temporale del contenuto virtuale con il mondo fisico. Aggiungere un livello di comunicazione via rete comporta indubbiamente un ulteriore fattore di ritardo, difficile da gestire se gli aggiornamenti sono molto frequenti e pesanti, in particolare se si considera anche di mantenere aggiornato il tracking dinamico dell'ambiente.

Oltre alle difficoltà tecniche del trasferimento dati rapido, i dispositivi di accesso utilizzati potrebbero anche essere eterogenei e, quindi, le fasi da svolgere localmente dovrebbero in qualche modo adattarsi alle differenze tra le piattaforme.

Sarebbe quindi necessario avere una descrizione del mondo comprensibile da dispositivi differenti, così come una descrizione della geometria degli oggetti da renderizzare sulle singole piattaforme.

Il problema della condivisione si scontra quindi con quello dell'interoperabilità ed è chiaro che, data l'attuale frammentazione del mercato che non può che peggiorare con lo sviluppo di nuove tecnologie abilitanti, devono essere affrontati insieme.

Un supporto multi-piattaforma, inoltre, potrebbe giocare un ruolo chiave anche per una maggiore fruibilità del contenuto virtuale da parte dei singoli utenti. È possibile immaginare uno scenario futuro in cui non avremo solamente un dispositivo personale di accesso, ma più di uno diversi a seconda del loro utilizzo.

In uno scenario di questo tipo, la possibilità di accedere allo stesso layer digitale condiviso con dispositivi diversi sarebbe fondamentale e si potrebbe, in qualsiasi momento, visualizzarne il contenuto senza preoccuparsi di dover prendere ed utilizzare il "dispositivo corretto", ma utilizzando quello che si ha a portata di mano.

L'obiettivo di questa tesi è per l'appunto estendere un prototipo di architettura condivisa per renderla accessibile da più piattaforme, non cercando di fornire uno standard per lo scambio delle informazioni, bensì mostrando

proprio come, una volta adottato un linguaggio comune (nel nostro caso le astrazioni fornite da Unity3D supportato dalla maggior parte delle tecnologie) è possibile realizzare una forma di condivisione efficace mantenendo una istanza del mondo in esecuzione su un server ed interfacciandosi ad essa dai singoli dispositivi anche se diversi tra loro.

Andiamo ora a vedere quali sono le diverse forme di condivisione possibili nel mondo della Realtà Mista e quali sono i requisiti tecnici che li accomunano.

2.3 Forme di condivisione

Parlando di condivisione è giusto chiarire alcuni importanti fattori che caratterizzano il tipo di esperienza condivisa in modo da valutare tutte le diverse possibilità quando si va a relizzare una nuova applicazione.

La documentazione Microsoft sulla creazione di esperienze condivise[3] articola la propria sezione sulle diverse forme di condivisione ponendo sei domande allo sviluppatore che aiutano a chiarire il tipo di esperienza che si vuole realizzare. Questi punti sono utili per andare ad elencare i diversi scenari possibili e permettono di dare una definizione operativa più precisa di condivisione, tenendo in considerazione le sue complesse sfaccettature.

- **Come avviene la condivisione**

  La maggior parte degli scenari di condivisione ricade nelle modalità di presentazione o collaborazione. Nella prima un utente gestisce gli ologrammi e gli altri li osservano, nella seconda tutti gli utenti interagiscono per ottenere un risultato comune. A questi, Microsoft aggiunge lo scenario di tutorato tra utenti in cui uno guida l'altro verso il raggiungimento di un obiettivo.

- **Quanti utenti partecipano allo scenario condiviso**

  La dimensione del gruppo di utenti è necessaria per capire quali strategie attuare sia in termini di design che anche, soprattutto, dal punto di vista tecnico per realizzare una comunicazione affidabile. Gruppi oltre le 6 persone co-locate, caricano la rete di un traffico intenso e bisogna esserne ben consapevoli per scegliere quale architettura utilizzare.

- **Dove si trovano gli utenti che condividono**

  Gli utenti che condividono il contenuto possono essere *co-locati*, oppure in luoghi fisici diversi. È anche possibile che entrambi gli scenari si intersezionino con due gruppi di utenti co-locati che condividono tra loro

[3] `https://docs.microsoft.com/en-us/windows/mixed-reality/shared-experiences-in-mixed-reality`

la stessa applicazione. In questi scenari, gli aspetti di privacy per la distinzione tra quali contenuti devono essere condivisi e quali no giocano un ruolo fondamentale e complicano il modello logico del mondo da condividere.

- **Quanto sono simili gli ambienti degli utenti**

  Dal momento che la Realtà Mista è strettamente legata al mondo fisico, è fondamentale saper progettare un'applicazione che sia efficace in ambienti anche diversi tra loro, ad esempio nel caso di utenti dislocati che, tuttavia, devono condividere lo stesso contenuto. Per risolvere problemi di questa natura si stanno sviluppando delle tecniche di riconoscimento semantico dell'ambiente in modo da poter appoggiare un ologramma su un tavolo riconosciuto in quanto tale indipendentemente dalla dimensione della stanza o dalla forma del tavolo stesso.

- **Quando avviene la condivisione**

  La condivisione può essere, nello scenario più tipico, sincrona con una forma di collaborazione che vede tutti gli utenti impegnati nello stesso compito contemporaneamente, ma anche asincrona come nel caso di una "stanza aumentata" in cui chiunque può manipolare il contenuto digitale in momenti diversi mantenendo le modifiche nel tempo. Scenari di questo tipo richiedono l'utilizzo di un concetto di sessione permanente per recuperare lo stato degli ologrammi oltre che ovviamente una forma di autenticazione in modo da tracciare chi ha effettuato quali modifiche.

- **Che dispositivi utilizzano**

  La differenza dei dispositivi di accesso è fondamentale nella progettazione delle esperienze condivise, dal momento che, con diverse piattaforme, si ottengono diversi gradi di immersione. Un'applicazione ben progettata deve permettere a tutti gli utenti di interagire in modo semplice ed efficace, indipendentemente dal dispositivo utilizzato.

In generale, i concetti chiave che emergono dalle combinazioni di questi scenari sono: la necessità di avere un concetto di sessione a cui gli utenti possono partecipare, la possibilità di avere un riferimento spaziale per indicare un preciso punto in un ambiente fisico (ancora), la necessità di una forma di connessione su cui condividere le interazioni e gli aggiornamenti di stato degli ologrammi e la possibilità di salvare lo stato del mondo in modo da poterlo ripristinare in un diverso momento nel tempo.

Questi elementi tecnici sono tutti importanti, alcuni più altri meno, a seconda del tipo di esperienza, per lo sviluppo di una reale forma di condivisione.

2.4 Stato dell'arte delle tecnologie abilitanti

Prima di procedere con la presentazione della soluzione proposta, analizziamo lo stato dell'arte delle tecnologie con un focus particolare sulle opzioni di condivisione offerte nativamente e le prospettive di interoperabilità.

Google ARCore

A livello cooperativo ARCore mette a disposizione il supporto delle `Cloud Anchors`. Un'ancora in ARCore è un oggetto che viene istanziato in un'applicazione a partire da un oggetto tracciato dal sistema, ovvero un `Trackable` che può rappresentare una superficie o un'immagine (marker). L'ancora, come è possibile capire intuitivamente, serve a mantenere il riferimento relativo all'oggetto su cui è stata creata.

Questo è necessario perché la comprensione del mondo che ARCore ottiene tramite i sensori potrebbe subire variazioni nel corso del tempo, alterando la posizione o rotazione di alcune superfici. Fissando gli oggetti con delle ancore queste variazioni vengono applicate anche agli oggetti permettendo di avere un risultato sempre più realistico man mano che il framework riconosce l'ambiente in modo più preciso.

Le `Cloud Anchor` hanno funzionamento in tutto e per tutto simile alle ancore locali e vengono hostate su dei server Google per un periodo massimo di 24 ore.

Un utente funge da host, una volta tracciato l'ambiente, può posizionare un'ancora e caricarla sul server. Gli altri utenti possono a quel punto accedere alla sessione condivisa, ottenre l'ancora dal server e ARCore si occupa di riconoscere l'ambiente e posizionare l'ancora in modo coerente. Ovviamente sul server viene caricato un'intorno di feature-points in modo da permettere di riconoscere l'ambiente circostante all'ancora stessa.

Questo meccanismo di condivizione è efficace per mostrare in simultanea a due utenti uno stesso oggetto inserito all'interno di una porzione di spazio che però deve essere stato tracciato.

Purtroppo non c'è nessuna consistenza all'infuori della posizione: se l'oggetto avesse un comportamento dinamico non ci sarebbe modo di mantenerlo aggiornato sfruttando il supporto nativo di ARCore, dal momento che non è possibile condividere altro che l'ancora che ne fissa il riferimento allo spazio.

Uno dei principali vantaggi di ARCore è che è disponibile anche per dispositivi iOS rendendola di fatto l'unica tecnologia per Realtà Aumentata disponibile per tutte le piattaforme mobili.

Apple ARKit

La gestione in ARKit della condivisione è simile a quella di ARCore dal momento che un utente deve necessariamente tracciare per primo l'ambiente, per poter poi condividere la `ARWorldMap`. La mappa non viene caricata su nessun server, ma, sfruttando l'astrazione della `MultiPeerConnectivity` già esistente tra dispositivi iOS, viene inviata direttamente a tutti i peer disponibili sulla stessa rete locale.

Sempre grazie alle capacità della rete peer-to-peer, ARKit offre un meccanismo nativo di update coerente delle entità aumentate tramite scambio di messaggi. Dal momento che inviare la mappa completa del mondo è un'operazione costosa in termini di banda e tempo solitamente viene fatto solo all'avvio dell'applicazione. Tutti gli update futuri possono essere inviati, invece, come dati tra i peer. È possibile ad esempio inviare nuove `ARAnchor` per specificare orientamento e posizione di una nuova entità aggiunta da uno qualunque degli utenti.

Oltre alle informazioni statiche, è possibile, nell'ottica di realizzare un'esperienza collaborativa, condividere azioni seguendo un pattern di action queue. Ogni peer ha un'istanza dell'applicazione che ad ogni update verifica se ci sono azioni da eseguire. Ogni volta che un peer registra una nuova azione questa viene inviata a tutti gli altri ed aggiunta alla coda.

Le simulazioni della fisica vengono eseguite localmente, quindi, per risolvere il problema della consistenza, si può operare in questo modo: viene assegnato ad un peer il ruolo di server sulla base del quale sincronizzare tutti gli altri, il server quindi invia oltre alle proprie azioni anche le informazioni rilevanti per la fisica che fungono da controllo dei calcoli locali, ovviamente vanno scelte delle strategie per limitare al minimo il consumo di banda per garantire un update fluido.

ARKit permette quindi di realizzare una forma di condivisione più solida, locale, e con funzionalità di update più strutturate rispetto alle `CloudAnchors` di ARCore. La principale limitazione resta il tracking preliminare dell'ambiente e l'interoperabilità limitata ai dispositivi Apple con requisiti hardware abbastanza stringenti (processore Apple A9 per le vecchie versioni e A12 per l'ultima versione con l'occlusione dinamica delle persone).

Windows Mixed Reality

Per quel che riguarda l'ambiente Microsoft, la soluzione offerta è Windows Mixed Reality: uno stack multipiattaforma per supportare applicazioni in Realtà Mista.

Dal punto di vista collaborativo, Microsoft offre una soluzione di ancore condivise tramite server Azure che permette a dispositivi anche con hardware differenti (es. HoloLens e uno Smartphone) di sincronizzare il posizionamento degli ologrammi in un sistema di riferimento condiviso. Questo permette ad esempio di creare una "modalità spettatore" in modo da permettere anche a chi non sta indossando il casco di visualizzare gli ologrammi in modo coerente sfruttando il supporto delle tecnologie ARCore e ARKit che abbiamo visto in precedenza.

Le Azure Spatial Anchors non pongono limitazioni di tempo anche se richiedono una fase iniziale di set-up del server Azure che può scoraggiarne l'utilizzo per un'applicazione relativamente semplice, rispetto alle soluzioni precedenti che sono più veloci da realizzare. La feature principale è l'accesso multiplatform a livello di presentazione, che pone le basi per importanti sviluppi di applicazioni a diversi gradi di immersione.

Come nelle altre tecnologie, le ancore possono essere piazzate dopo aver fatto uno scanning dell'ambiente. Il vantaggio è che, una volta che questa operazione è stata fatta la prima volta da un dispositivo, poi, potenzialmente per sempre, altri possono accedere allo stesso scenario aumentato.

Un meccanismo particolare implementato dalle Spatial Anchors è quello della connessione tra diverse ancore. Questo permette ad ancore appartenenti alla stessa sessione di mantenere relazioni spaziali e utilizzare queste relazioni in diverse situazioni a beneficio dell'utente ad esempio indicando in tempo reale la direzione da seguire per la prossima ancora: feature molto utile in diversi scenari di grandi dimensioni come la visita di un museo aumentato.

Nel caso particolare di HoloLens è possibile esportare un'ancora utilizzando la serializzazione ed inviarla ad un altro dispositivo localmente ad esempio utilizzando una connessione TCP. Questo meccanismo, meno stabile e versatile di quello presentato precedentemente, permette di creare esperienze condivise localmente tra dispositivi simili senza doversi preoccupare di gestire l'infrastruttura server.

Per condividere anche l'update del mondo non è presente un servizio nativo e lo sviluppatore può scegliere se implementarlo ad hoc o affidarsi ad un servizio di terze parti per il multiplayer come ad esempio Photon che viene consigliato in alcuni tutorial ufficiali.

Altre tecnologie

Oltre ai principali produttori che abbiamo citato, nel panorama delle tecnologie di Realtà Aumentata, sono moltissime le aziende indipendenti che si dedicano allo sviluppo di applicazioni software con obiettivi più mirati.

Per quel che riguarda l'aspetto di condivisione, tra le più note e promettenti c'è la già citata Spatial[4] che propone ambienti di sviluppo collaborativi basati sull'idea della telepresenza e delle "lavagne virtuali" in cui poter appendere post-it con contenuti eterogenei. Spatial è stata lanciata inizialmente come applicazione per Hololens, ma viene offerto un sistema di condivisione anche per utenti Desktop.

Un'altra azienda promettente che sta lavorando su un supporto per gli sviluppatori per creare applicazioni con condivisione è Ubiquity6[5] che come feature principali mira ad offrire un riconoscimento dell'ambiente non solo geometrico, ma semantico, in modo da distinguere gli oggetti presenti nelle stanze. Propone anche una sessione permanente nel tempo in grado di mantenere lo stato e farlo progredire in relazione al tempo trascorso nel mondo reale.

Entrambe le aziende rivelano molto poco delle loro effettive capacità e delle infrastrutture proposte oltre alle immagini dimostrative. Indubbiamente, però, sono il segno che il tema della condivisione è uno dei più caldi da esplorare al momento ed è possibile iniziare a realizzare esperienze che siano effettivamente condivise e utili per gli utenti.

2.5 Interoperabilità nel mercato attuale

Dall'analisi delle tecnologie esistenti emerge sostanzialmente che quasi tutti i produttori si stanno muovendo nella direzione di uno scenario aperto alla condivisione, probabilmente motivati dalle stesse ragioni che abbiamo citato all'inizio di questo capitolo.

Quasi tutti i servizi nativi abilitanti alla condivisione, tuttavia, sono forniti come estensione delle funzionalità principali creando l'astrazione di un'ancora in grado di essere riconosciuta dai sistemi di tracking e non ponendo l'accento su una reale dimensione condivisa in grado di mantenere non solo i riferimenti ma anche lo stato del mondo.

Inoltre, a parte per la soluzione offerta da Microsoft che offre un supporto da spettatori per gli utenti dotati di smartphone, le altre tecnologie non considerano minimamente la possibilità di interagire con dispositivi diversi e restano completamente isolate rendendo l'applicazione dipendente dalla piattaforma.

Tuttavia, anche nel caso di Microsoft, si ha una forma di condivisione estremamente limitata, e non viene accennato praticamente nulla sull'idea di una sessione persistente a cui poter accedere nel corso del tempo come invece

[4] `https://spatial.is`
[5] `https://ubiquity6.com/`

viene proposto dalle nuove tecnologie emergenti dedicate interamente agli spazi condivisi come Spatial e Ubiquity6.

In generale, il tema della condivisione è attualmente trattato come puramente operativo, per poter fornire agli sviluppatori una qualche forma di supporto per iniziare a creare esperienze più complesse. È chiaro che questo genere di modello cambierà in futuro in quanto, come abbiamo detto, non è sufficiente a supportare tutti i tipi di applicazioni, ma solamente alcuni scenari particolarmente limitati e semplici da gestire.

Per la creazione di un vero e proprio layer aumentato condiviso è necessario, quindi, definire un nuovo livello di astrazione da porre più in alto delle singole tecnologie abilitanti. Questo potrebbe portare enormi benefici per risolvere allo stesso tempo il problema della condivisione e quello dell'interoperabilità.

Non solo, la creazione di un'infrastruttura più complessa, ma in grado di essere supportata dalle diverse piattaforme, permetterebbe di migliorare le qualità del software prodotto quali portabilità, riusabilità, interoperabilità, e facilità di manutenzione permettendo agli sviluppatori di lavorare in modo più efficiente e aprendo le porte ad una maggiore diffusione delle applicazioni in Realtà Aumentata che in questa fase sono ancora estremamente difficili da sviluppare e soprattutto richiedono enormi quantità di tempo per le fasi di debug e testing che potrebbero invece essere simulate se separate dalla parte responsabile del rendering.

La proposta in letteratura che viene valutata in questa tesi come possibile soluzione a questi problemi è quella degli Augmented Worlds che verranno approfonditi nel capitolo successivo.

Capitolo 3

La proposta degli Augmented Worlds

Chiarita la necessità della definizione di un modello indipendente dalle tecnologie ed i benefici che esso potrebbe portare nello sviluppo di applicazioni in Realtà Mista con focus collaborativo, in questo capitolo viene presentata l'infrastruttura degli Augmented Worlds[8]: un modello ad agenti per la creazione di ambienti pervasivi che integrino elementi di Realtà Mista. Sebbene ancora in fase prototipale, l'utilizzo del framework MiRAgE, basato sul modello, ha portato alla realizzazione di applicazioni in contesti reali, evidenziandone le potenzialità e candidandolo come soluzione per alcuni dei problemi evidenziati sul tema della condivisione tra cui l'interoperabilità tra diverse piattaforme.

3.1 Il modello concettuale degli Augmented Worlds

L'idea degli Augmented Worlds nasce come ponte per integrare diverse tecnologie d'avanguardia in un unico scenario che sia strettamente legato alla realtà in cui ci muoviamo quotidianamente.

In particolare, gli sviluppi nei settori di Pervasive Computing, Mixed Reality e Web Of Things pongono le basi per la creazione di ambienti intelligenti, aumentati dalla presenza di una rete di sensori per rilevare le caratteristiche del mondo fisico e popolati da agenti in grado di interagire tra loro e con gli utenti umani tramite l'interfaccia della Realtà Mista.

Fortemente influenzato dalla ricerca sui Mirror Worlds[14] che mappano un livello digitale sul mondo fisico creando, appunto, uno specchio virtuale della realtà in cui svolgere la computazione, il modello degli Augmented World ne

presenta un'implementazione in cui gli agenti assumono un ruolo chiave nella gestione degli ologrammi per interfacciarsi con il mondo fisico.

I principi secondo i quali è stato progettato il modello degli Augmented Worlds sono i seguenti:

- Supporto per un contesto multi-utente e cooperazione tra utenti umani. Gli agenti godono delle stesse capacità di cooperazione degli umani.
- *Bidirectional Augmentation* come forma estesa rispetto al concetto di Realtà Aumentata o Mista che coinvolge anche la prospettiva del pervasive computing di arricchire l'ambiente con capacità computazionali
- Creazione di un sistema aperto nel quale sia possibile entrare e uscire dinamicamente ed agire in real-time.
- Supporto per diversi paradigmi di sviluppo ad agenti senza vincolare gli sviluppatori ad una specifica tecnologia.
- Supporto a diverse forme di tecnologie per la Realtà Aumentata e Mista separando il layer di visualizzazione dalla logica applicativa del sistema.
- Generalità per permettere di sviluppare contesti applicativi differenti sia in termini di dimensioni (indoor/outdoor) sia in termini di dominio.

L'astrazione che viene data agli elementi presenti nel layer digitale è quella di Entità Aumentate (AE) che mantengono una posizione ben specifica nello spazio e possono essere osservate o modificate dagli agenti che popolano il mondo aumentato. Le entità possono essere accoppiate a degli ologrammi che rendono le entità visibili agli utenti del sistema, permettendo loro di interagire con esse attraverso la Realtà Mista.

Il diagramma UML in figura 3.1 illustra le principali componenti del modello degli Augmented Worlds. È possibile notare come un `Augmented World` è composto da una collezione di `Augmented Entities` e può essere partizionato in `Region`, in modo da definire in modo più stringente alcune zone spaziali dalle quali ottenere gli identificatori delle entità che vi entrano o escono.

Gli `Agents` devono in primo luogo entrare all'interno dell'`Augmented World` per poter creare, modificare e osservare le `Augmented Entities`.

Le entità aumentate possono avere un `Hologram` per essere percepite dall' `Human User` che modella un utente del sistema che mette in esecuzione un'applicazione in grado di accedere all'interno del mondo. L'utente viene a sua volta modellato da uno `User Avatar` digitale così come gli oggetti reali e vengono di fatto anch'essi rappresentati come `Augmented Entity` in modo da non avere distinzioni ad alto livello e poter operare su tutti gli elementi del mondo aumentato allo stesso modo.

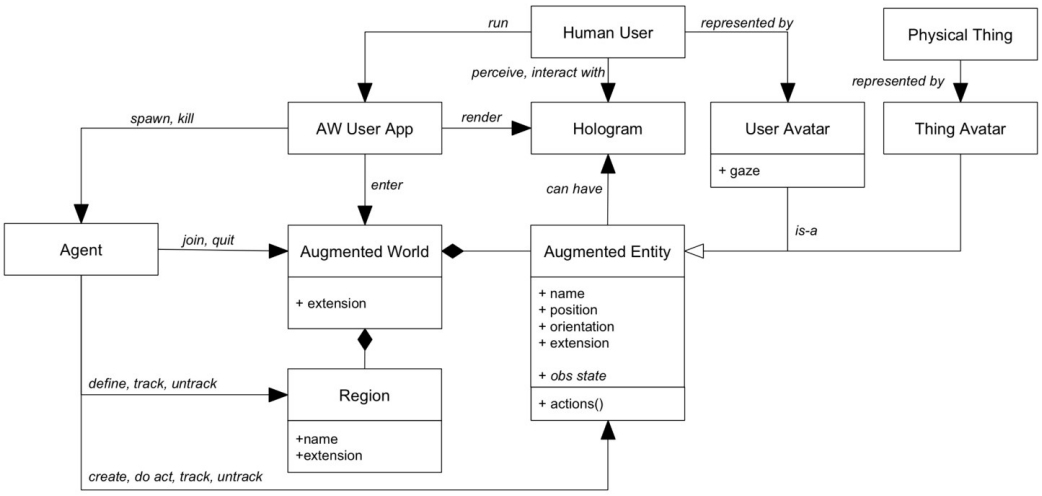

Figura 3.1: Modello concettuale degli Augmented Worlds

3.2 Un framework per Augmented Worlds: MiRAgE

Sulla base del modello concettuale degli Augmented Worlds è stato realizzato un prototipo: MiRAgE (**Mi**xed **R**eality based **A**u**g**mented **E**nvironments) [7] si pone come esempio di una implementazione come framework per agevolare lo sviluppo futuro delle applicazioni e del modello stesso.

MiRAgE è stato progettato con l'intento di fornire agli sviluppatori un livello di astrazione che permetta di sviluppare Augmented Worlds senza doversi curare degli aspetti di gestione dei sistemi distribuiti e cercando di isolare il più possibile le tecnologie di Realtà Aumentata in modo da permettere di concentrarsi quasi esclusivamente sul design delle entità aumentate, come oggetti dotati di uno stato osservabile ed azioni da poter svolgere su di esse, sulla programmazione del comportamento degli agenti che popolano il sistema e sulla creazione dei modelli tridimensionali degli ologrammi che rappresentino le entità.

3.2.1 Panoramica infrastrutturale

I principali requisiti per cui MiRAgE è stato progettato sono:

- Supporto per scenari indoor e outdoor di mondi aumentati
- Supporto per diverse tecnologie ad agenti

- Supporto per dispositivi di natura diversa

Dai requisiti emerge come il principio fondante che è stato scelto per la creazione del framework è la *generalità* ovvero la capacità di essere uno strumento allo stesso tempo potente, ma non vincolante nelle mani degli sviluppatori.

Altri aspetti importanti sul quale è stata basata la progettazione sono l'affidabilità del sistema, in quanto gli sviluppatori devono poter dipendere dal framework per lo sviluppo delle proprie applicazioni, e la scalabilità per permettere configurazioni di mondi aumentati anche potenzialmente molto grandi e popolati, fornendo una astrazione logica per gli utenti e gli agenti in modo da celare la natura distribuita del sistema.

A livello infrastrutturale è possibile individuare tre componenti principali (Figura 3.2)

- L' **AW-Runtime** che fornisce l'infrastruttura per mettere in esecuzione istanze di Augmented World fornendo un'interfaccia comune agli agenti per interagirci ed osservarli. Il runtime è pensato per essere eseguito su un architettura server, possibilmente distribuita sul cloud, pur mantenendo un livello di astrazione logico che nasconda la struttura sottostante. È di fatto il centro di comando di tutto il mondo ed è compito del Runtime di propagare gli eventi scaturiti dalle entità aumentate a tutti gli osservatori ed in particolar modo all'Hologram Engine.

- L' **Hologram Engine** supporta la visualizzazione degli ologrammi e mantiene aggiornata la loro geometria in tutti i dispositivi degli utenti umani che osservano il mondo, considerando le interazioni fisiche e gli input degli utenti. È in esecuzione sia sull'AW-Runtime che sui dispositivi degli utenti, comunicando tramite un canale di rete (es. una connessione TCP). Dal punto di vista dell'Augmented World mantiene il supporto per mantenere la geometria di ogni ologramma e ne mantiene aggiornata la visualizzazione in accordo con le proprietà dell'AE associata. Dal punto di vista degli utenti mette a disposizione le funzionalità per la visualizzazione, la gestione della fisica e la ricezione degli input.

- Il **WoAT** (Web of Augmented Things)[6] è un livello di comunicazione che permette massima libertà ed interoperabilità a livello applicativo, incoraggiando comunque un design orientato agli agenti, ed è progettato per avere una buona scalabilità per gestire anche grandi numeri di entità aumentate. Permette anche una buona integrazione con il mondo dell'IoT (Internet of Things) nell'ottica di realizzare reti di sensori che comunichino al mondo aumentato lo stato del mondo fisico. La soluzione del Web of Things permette di raggiungere le risorse disponibile tramite API di tipo REST per interagire con esse, in questo modo le interazioni

tra entità aumentate sono gesitite tramite operazioni HTTP raggiungibili tramite indirizzi URL a partire dall'indirizzo *root* dell'Augmented World.

Attualmente l'implementazione del prototipo di MiRAgE è realizzata utilizzando Java per le componenti lato server dell'applicazione (AW-Runtime, WoAT Service) e C# nella forma di Script Unity per il lato client.

A livello operativo invece, MiRAgE offre un elenco di API delle azioni di base per mezzo delle quali è possibile entrare nel mondo e creare, distruggere e osservare entità aumentate e regioni. Tutte le altre azioni possibili vengono invece direttamente esposte dalle entità aumentate come interfacce specifiche, in questo modo il modello si mantiene generale in quanto non determina in nessun modo il comportamento delle entità aumentate permettendo di realizzare scenari con domini applicativi molto diversi tra loro.

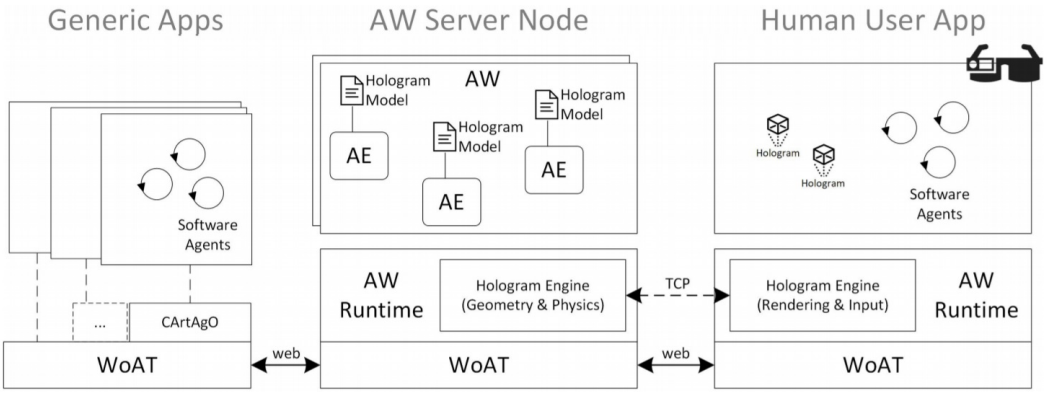

Figura 3.2: Architettura logica del framework MiRAgE

3.2.2 Hologram Engine Bridge

Analizzando l'infrastruttura di MiRAgE abbiamo osservato come il componente responsabile della visualizzazione del mondo aumentato, l'Hologram Engine, sia effettivamente suddiviso in due componenti, una lato server e l'altra lato client.

La connessione tra ogni Hologram Engine locale e l'istanza centrale in esecuzione nell'AW-Runtime è mantenuta da un Hologram Engine Bridge (HEB) il cui ruolo all'interno del sistema è possibile osservare nel diagramma delle classi nella figura 3.3.

La comunicazione dell'Hologram Engine Bridge è attualmente implementata basandosi sul protocollo TCP dal momento che è essenziale che le informazioni relative all'update del mondo arrivino nell'ordine corretto per evitare problemi di inconsistenza tra le visualizzazioni degli ologrammi e lo stato effettivo

del mondo aumentato. La responsabilità di connettersi al server è delegata agli Hologram Engine locali dal momento che le applicazioni client devono necessariamente come prima cosa entrare nel mondo aumentato e quindi conoscono l'IP del server a cui connettersi.

Il compito principale del bridge è quello di aggiornare tutti gli utenti del sistema propagando gli eventi relativi alla creazione o alterazione degli ologrammi legati alle varie entità aumentate. Questo significa, per esempio, che, nel momento in cui un agente modifica la posizione di una entità aumentata, se questa entità ha un ologramma associato ad essa, l'informazione relativa alla nuova posizione viene incapsulata in un oggetto JSON ed inviata a tutti gli Hologram Engine.

Allo stesso tempo L'HEB permette agli Hologram Engine locali di aggiornare l'AW-Runtime con le posizioni degli UserAvatar e sulle interazioni che gli utenti hanno con gli ologrammi, in modo da propagare correttamente gli esiti delle azioni.

Questo componente è il fulcro che abilita una corretta condivisione del mondo aumentato e rappresenta quindi il cuore del supporto multi utente in MiRAgE.

3.3 Multi-piattaforma in MiRAgE

Analizzando il comportamento dell'Hologram Engine abbiamo appurato che l'infrastruttura di MiRAgE è di propria natura adatta ad ospitare più utenti.

Questi utenti possono essere reali o agenti, per gli agenti è sufficiente il layer del Web of Augmented Things per interfacciarsi correttamente al mondo, in quanto non hanno la necessità di renderizzare il contenuto e di conseguenza non hanno problemi in termini di latenza rispetto agli utenti umani che potrebbero trovare fastidioso l'utilizzo del sistema se lo scambio di informazioni non fosse sufficientemente rapido, dal momento che la visualizzazione degli ologrammi risulterebbe poco fluida.

Come abbiamo visto la soluzione a questo problema viene risolta dall'Hologram Engine Bridge che si occupa di avere una comunicazione diretta con i motori grafici in esecuzione sui dispositivi.

Dal momento che l'obiettivo di questa tesi è quello di integrare un supporto multi-piattaforma, è in questo contesto che si andrà ad inserire il contributo pratico realizzato cercando di estendere l'infrastruttura multi-utente già presente in MiRAgE.

Non essendo stato definito uno standard per inviare informazioni relative alla geometria tridimensionale necessaria per scambiare informazioni sugli olo-

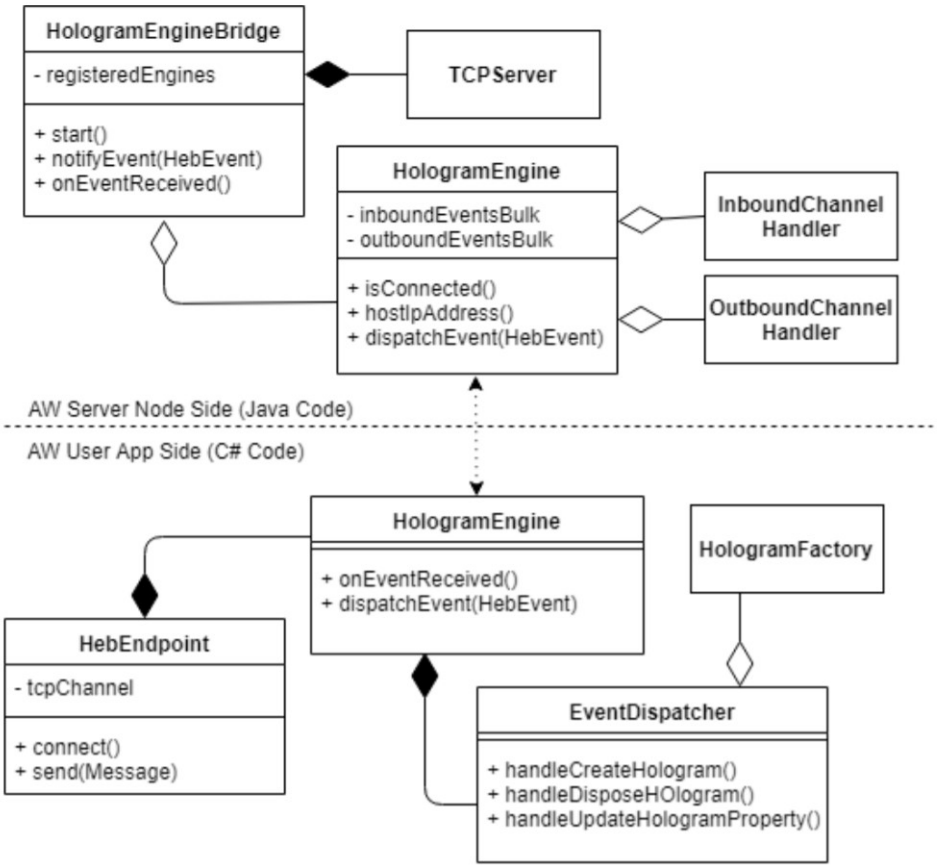

Figura 3.3: Diagramma delle classi del dettaglio riguardo la comunicazione tra Hologram Engine

grammi e dal momento che la standardizzazione non è tra gli obiettivi di questa tesi, si sceglie di utilizzare il concetto dei GameObject di Unity come astrazione comune per permettere agli Hologram Engine di dialogare, supponendo che tutti i dispositivi connessi supportino l'engine grafico che è attualmente lo standard *de facto* per lo sviluppo di applicazioni in Realtà Aumentata.

Questa assunzione, tuttavia, non priva di generalità la soluzione proposta dal momento che per estendere ulteriormente il supporto non solo a diverse piattaforme, ma anche a diversi engine grafici, sarà sufficiente rimpiazzare il layer di comunicazione con un protocollo ad hoc.

Capitolo 4

Integrazione di una condivisione multi-dispositivo su MiRAgE

In questo capitolo si vanno a descrivere gli step intrapresi nella progettazione di un livello di astrazione per permettere a diverse tecnologie di entrare all'interno dei mondi aumentati realizzati con MiRAgE e della successiva integrazione del HMD Meta2 utilizzato come esempio in questa tesi.

4.1 Stato attuale e obiettivi dell'integrazione

Dall'analisi dell'infrastruttura di MiRAgE fatta nel capitolo precedente emerge che è stato sviluppato nell'ottica di poter essere esteso per supportare non solo più utenti ma anche diverse piattaforme.

Nella sua fase prototipale, MiRAgE è stato sviluppato utilizzando la libreria per Realtà Aumentata marker-based Vuforia attraverso la quale si possono gestire gli ologrammi renderizzati dalla UserApp a patto di mantenere inquadrato un marker per stabilire un riferimento spaziale.

Successivamente, dal momento che le tecnologie puramente marker-based sono diventate obsolete per lo sviluppo di un mondo aumentato di certe dimensioni, è stata integrata la piattaforma ARCore all'interno del framework in modo da supportare le piattaforme mobili e abilitare gli sviluppatori, attraverso MiRAgE, all'utilizzo di alcune feature moderne come il motion tracking.

L'estensione per il supporto di ARCore è stata realizzata con l'ottica di separare i compiti dell'Hologram Engine da quelli della tecnologia abilitante, favorendo la modularità per l'inserimento di altre piattaforme all'interno di MiRAgE.

Il compito di questa tesi è raffinare ulteriormente questa modularità e realizzare una integrazione dell'SDK in Unity fornito dal visore Meta2 in mo-

do da esplorare il comportamento del mondo aumentato osservandolo da due piattaforme molto diverse tra loro.

In particolare gli obiettivi dell'integrazione sono i seguenti:

- **Registrazione dei dispositivi per l'ingresso nel mondo**
  Gli Augmented Worlds sono pensati come ambienti popolati sincronizzati sia temporalmente che spazialmente con il mondo fisico. Come abbiamo osservato dalla struttura del framework gli utenti hanno l'obbligo, per prima cosa, di entrare all'interno del mondo per poter essere riconosciuti come nuove entità aumentate e ricevere le informazioni sulle altre entità che ne fanno parte.

  In questa fase di "ingresso" si deve, oltre che notificare l'AW-Runtime, sincronizzare il sistema di riferimento del dispositivo dell'utente con quello del mondo aumentato, in modo che ci sia coerenza tra le informazioni relative alle posizioni mantenute sul server.

  Per integrare correttamente Meta2 per lavorare all'interno di un mondo aumentato è quindi necessario elaborare una strategia, sufficientemente precisa, per svolgere questa operazione.

- **Visione coerente delle entità aumentate**
  L'obiettivo principlae dell'integrazione è permettere una visualizzazione coerente delle entità in termini sia di registrazione spaziale, che di stato aggiornato. Sfruttando le potenzialità del framework si mira a fare in modo che gli utenti percepiscano gli ologrammi in modo sempre coerente.

- **Possibilità di interazione con diversi livelli di immersione**
  Oltre a poter visualizzare gli ologrammi gli utenti devono poter essere in grado di interagire in modo basilare con essi e le loro azioni devono alterare correttamente lo stato delle corrispondenti entità aumentate, facendo in modo che anche gli altri utenti vedano il risultato correttamente. L'idea è quella di sfruttare le capacità dei diversi dispositivi in modo da creare interazioni che permettano di svolgere determinate azioni anche a livelli di immersione differenti a seconda della piattaforma utilizzata.

  Su questo punto si potrebbe lavorare in modo intensivo, ad esempio elaborando un sistema di interfacce dipendente dalle capacità dei dispositivi, in questa tesi però, l'obiettivo è semplicemente quello di mostrare che l'integrazione permette di svolgere azioni sugli ologrammi e non soffermarsi sul dettaglio di come queste azioni devono essere progettate per renderle generalmente applicabili a tutti i contesti, cosa che potrà essere argomento di espansioni future del framework.

4.2 Unity: concetti base

Come accennato in precedenza, lo strumento di sviluppo utilizzato per integrare le diverse piattaforme è Unity per cui è stato realizzato un insieme di script che abilitano lo sviluppatore all'adozione di MiRAgE per la propria applicazione.

Per semplificare la trattazione vengono elecati alcune delle astrazioni principali introdotte nell'engine che sono utili per una comprensione più profonda del lavoro svolto.

- **Scene**
  La Scene in Unity rappresenta una "vista" dell'applicazione, può essere un menu, un livello di un gioco. In generale rappresenta lo spazio in cui si muove il player e dove vengono posizionati gli oggetti. Possono essere salvate e riaperte in qualsiasi momento.

- **GameObject**
  Tutti gli oggetti all'interno di una Scene sono rappresentati come GameObjects (GO). Questa astrazione permette di identificare in modo comune tutti gli elementi del gioco e su ognuno di essere svolgere operazioni basilari come l'Update che determina l'evoluzione del gioco.

  Ogni GameObject possiede di default un `Transform` ovvero un componente che specifica dimensioni, posizione e rotazione dell'oggetto relativamente allo spazio di coordinate della Scena.

  Per aggiungere altre capacità ad un GO è necessario aggiungervi dei Component.

- **Component**
  Un Component è una funzionalità generica che può essere aggiunta a qualsiasi GameObject per aggiungergli un comportamento o descrivere il suo aspetto.

  Ne esistono di molti pre-esistenti che permettono di specializzare un GameObject, ad esempio aggiungendogli una Mesh per renderlo effettivamente tridimensionale, specializzare il materiale con cui deve essere renderizzato, oppure per renderlo soggetto alle leggi della fisica.

  Ogni Componente ha diversi paramteri da impostare per modificare leggermente il comportamento e questo può essere fatto comodamente dall'editor che espone un'interfaccia grafica per inserire i valori.

- **Script** Per estendere il comportamento di un oggetto con funzionalità personalizzate è necessario creare uno Script, solitamente utilizzando

C# (ma sono disponibili anche altri linguaggi di scripting), che interagisca con i vari Components e renda l'oggetto "vivo".

Ogni Script di comportamento estende la classe `MonoBehaviour` ed eredita le due funzioni principali `Start()` chiamata quando l'oggetto viene istanziato e `Update()` chiamata ad ogni frame del gioco.

- **Prefab**
  È possibile salvare dei GameObject sotto forma di Prefab in modo da conservare lo stato di tutti i Component ad essi associati. I Prefab possono essere riutilizzati in altri progetti a patto di importare tutti gli oggetti necessari e permettono di istanziare oggetti complessi.

4.3 Layer di interoperabilità per Hologram Engine

Per integrare correttamente un supporto multi-piattaforma si è scelto di progettare un livello intermedio che fosse sufficientemente generale per contenere le funzionalità necessarie al funzionamento del mondo aumentato lasciando la massima libertà di estensione per le diverse piattaforme.

La progettazione di questo livello è stata guidata dalla suddivisione delle responsabilità da affidare interamente ai dispositivi da quelle generali che accomunano il comportamento di un qualsiasi device all'interno del mondo aumentato.

In questo modo, la struttura risultante è centrata su un intermediario tra l'HologramEngine e la piattaforma di destinazione che si occupa di gestire la visualizzazione degli ologrammi in modo coerente con lo stato corrente del dispositivo. (Figura 4.2)

Nella struttura complessiva del framework abbiamo quindi un blocco che funge da raccordo tra la logica applicativa presente nella `UserApp` e la parte dedicata alla ricezione degli eventi relativi al mondo aumentato con cui l'applicazione entra in contatto delegata all'HologramEngine.

Questa mediazione è fondamentale per rendere il comportamento dell'HologramEngine completamente indipendente dalla piattaforma sottostante, deputando il rilevamento dell'input dell'utente alla logica applicativa e la registrazione spaziale con il mondo aumentato a componenti specifici relativi al dispositivo, poichè diversi dispositivi necessitano metodi di allineamento diversi a seconda delle caratteristiche tecniche.

Il collocamento logico del nuovo layer è evidenziato nella figura 4.1.

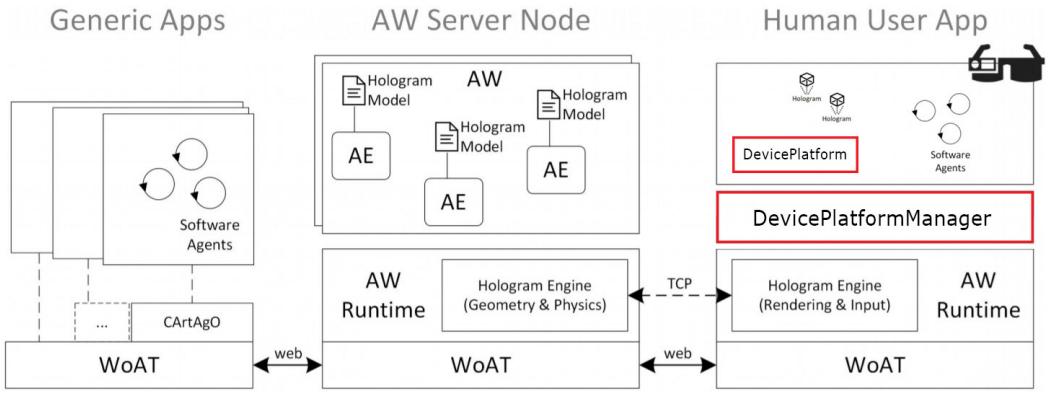

Figura 4.1: Il collocamento logico all'interno dell'infrastruttura di MiRAgE del layer di interoperabilità per l'HologramEngine.

Si nota come questa soluzione si compone di due componenti principali, uno generico per collegare il mondo aumentato all'applicazione e uno specifico da implementare per ogni nuova piattaforma da supportare.

Il `DevicePlatformManager` è il componente individuato per svolgere il ruolo di intermediario. Si occupa di ricevere la lista degli ologrammi attivi dall'HologramManager di MiRAgE e di mostrarli o nasconderli a seconda della stato della registrazione del dispositivo con il mondo aumentato.

```
public class DevicePlatformManager : MonoBehaviour,
    IRegistrationListener {

    public bool DeviceRegistered { get { return
        _device.DeviceRegistered; } }


    public DevicePlatform _device;
    private GameObject _currentParent;

    [...]

    // Update is called once per frame
    void Update()
    {
        if (HologramEngine.Instance.AwServiceInfo == null)
        {
            HideHolograms();
            return;
        }
```

```
        if (DeviceRegistered)
        {
            ShowHolograms(_currentParent);
        } else
        {
            HideHolograms();
        }
    [...]
     }

}
```

Listato 4.1: Il metodo di update del DevicePlatformManager

Per fare ciò implementa l'interfaccia `IRegistrationListener` che viene usata dal `DevicePlatform` per realizzare il pattern Observer e notificare gli osservatori dello stato del dispositivo.

Questa scelta è stata fatta nell'ottica di semplificare il lavoro futuro per l'abilitazione di nuove piattaforme, in quanto, per supportare un nuovo dispositivo, sarà sufficiente estendere da `DevicePlatform` ed implementare i metodi `CheckRegistration()` che viene chiamato ad ogni update e restituisce lo stato attuale della registrazione del dispositivo ed il metodo `GetCurrentParent()` che restituisce il GameObject da utilizzare come parent per gli ologrammi in modo che mantengano la posizione corretta con il sistema di riferimento individuato.

```
public abstract class DevicePlatform : MonoBehaviour {

    public bool DeviceRegistered { get; private set; }

    private GameObject _currentParent
    private List<IRegistrationListener> _listeners;

    [...]

    // Update is called once per frame
    void Update()
    {
        if (CheckRegistration())
        {
            if(_currentParent == null) //first track
            {
                _currentParent = GetCurrentParent();
                NotifyRegistration();
```

```
            }
        }
        else
        {
            NotifyLostTracking();
        }
    }

    private void NotifyRegistration()
    {
        _listeners.ForEach(l =>
            l.OnRegistrationCompleted(_currentParent));
        DeviceRegistered = true;
    }

    private void NotifyLostTracking()
    {
        _currentParent = null;
        _listeners.ForEach(l => l.OnTrackingLost());
        DeviceRegistered = false;
    }

    protected void NotifyNewMarkerLocated(GameObject offset)
    {
        _listeners.ForEach(l => l.OnNewMarkerLocated(offset));
    }

    /**
     * Return every update the state of current registration.
     **/
    protected abstract bool CheckRegistration();

    /**
     * Each update returns current worldOrigin.
     */
    protected abstract GameObject GetCurrentParent();

    [...]

}
```

Listato 4.2: La logica della classe astratta DevicePlatform che incapsula il comportamento di base dei dispositivi e notifica il Platform Manager

Nella fase di progettazione è stata anche rimodellato il supporto che era stato aggiunto in precedenza per ARCore all'interno di MiRAgE. Nel fare questo, dal momento che era stata inserita una feature di riconoscimento multi-marker per migliorare la precisione della registrazione in spazi ampi, si è scelto di conservare questa possibilità aggiungendo un metodo protetto nella classe astratta `NotifyNewMarkerLocated()` in modo che le piattaforme che supportano un riconoscimento multi-marker siano abilitate a restituire un GameObject che rappresenti la posizione del nuovo marker rilevato nel mondo fisico in termini di coordinate espresse in funzione dell'origine del mondo rilevata inizialmente.

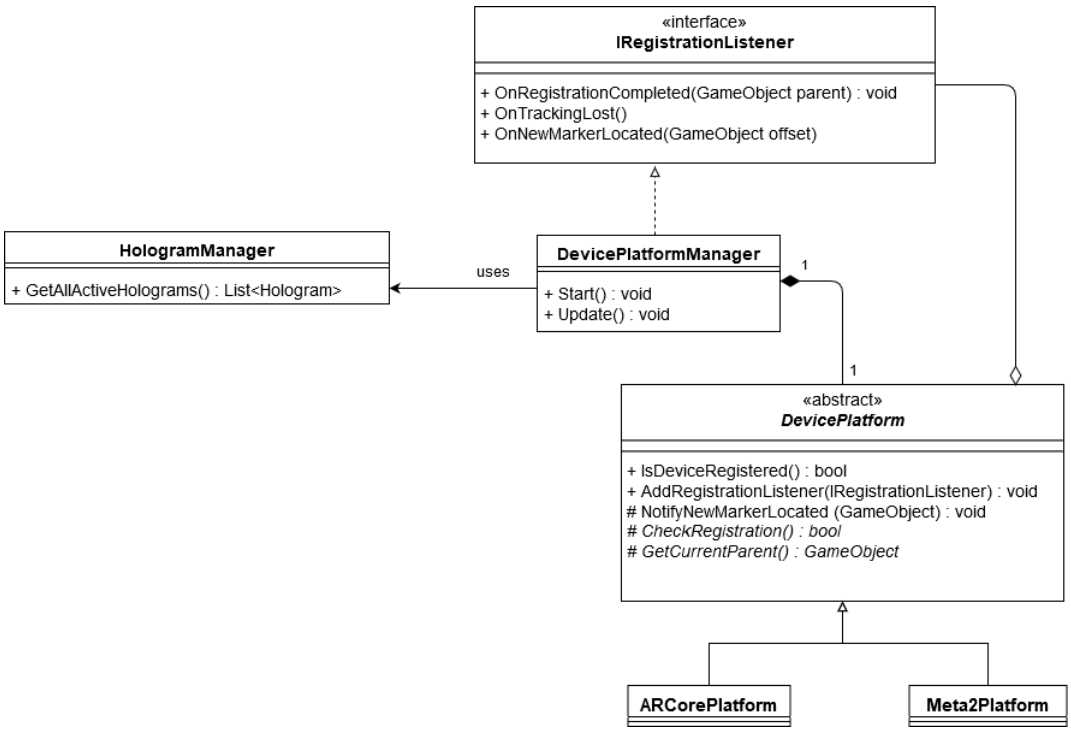

Figura 4.2: Il modello concettuale della soluzione proposta

4.4 Implementazione di un supporto per Meta2

Dopo aver progettato come estendere l'infrastruttura esistente per essere più aperta alla integrazione di altre piattaforme, il lavoro principale di questa tesi è stata l'implementazione della soluzione proposta per integrare l'HMD Meta2 e la conversione dell'estensione già realizzata per ARCore per adattarsi alla struttura corrente.

Questo esempio pratico di integrazione serve innanzitutto a dimostrare la validità dell'estensione proposta per il framework, inoltre la scelta di supportare un HMD come Meta2 è stata operata con l'intento di realizzare scenari di condivisione che integrino tecnologie molto diverse tra loro per vedere come il framework si comporti in questo contesto in termini di prestazioni e consistenza del mondo aumentato e per aprire le porte a future ricerche nell'ambito della condivisione e collaborazione a diversi livelli di immersione.

Infatti, Meta2, nonostante le limitazioni spaziali che gli sono imposte dall'essere cablato, offre la possibilità di interagire con gli ologrammi direttamente con le mani. Questa forma di interazione è molto diversa, ad esempio, da quella possibile attraverso il touchscreen di un dispositivo abilitato da ARCore.

Questi fattori sono interessanti per poter migliorare MiRAgE ed estendere le sue funzionalità a supporto degli sviluppatori.

4.4.1 Registrazione dei dispositivi ed allineamento con il mondo

Come chiarito in precedenza, uno degli obiettivi dell'integrazione riguarda la registrazione dei dispositivi per entrare correttamente all'interno del mondo e sincronizzare il mondo fisico con quello virtuale.

In generale questo problema è risolvibile fissando tre elementi che permettono di dialogare correttamente:

- Un punto nello spazio tridimensionale da considerare come origine
- Un orientamento degli assi spaziali (X, Y Z)
- Una scala delle distanze

Per quel che riguarda Meta2 questi concetti sono fissati dall'SDK contestualmente al sistema di riferimento del visore. In particolare il punto da considerare come origine del mondo viene identificato come la posizione iniziale in cui il visore viene avviato e riconosce l'ambiente. Successivamente Meta è in grado di recuperare le informazioni dello spazio e ricostruire il modello tridimensionale correttamente motivo per cui se il visore viene avviato in un punto diverso dopo aver tracciato l'ambiente in modo preliminare l'origine potrebbe essere riconosciuta nella posizione fissata precedentemente.

L'orientamento degli assi invece viene assunto con l'asse y perpendicolare al pavimento, asse z uscente dal visore e asse x di conseguenza alla destra dell'utente.

Per quel che riguarda la scala questa è gestita da Unity che richiede agli sviluppatori di inserire le dimensioni e le posizioni degli oggetti utilizzando come unità di misura il metro.

Per permettere a Meta di sincronizzarsi correttamente con il mondo aumentato esistono due strade percorribili. Entrambe prevedono un approccio basato su un marker di ingresso dal momento che è la scelta più sicura per assicurare una corretta registrazione al mondo e per avere un'azione ben precisa attraverso la quale innescare la notifica all'AW-Runtime per effettuare l'ingresso nel mondo.

L'approccio inizialmente marker-based è infatti stato inserito anche all'interno dell'estensione per ARCore.

Le due strade possibili sono le seguenti:

- Integrazione di una libreria per riconoscimento di Marker con Meta2
- Sincronizzazione basata su conferma dell'utente

Il primo approccio porta delle garanzie in termini di riconoscimento del marker e della precisione del punto stimato. Inoltre il marker riconosciuto porterebbe con sè anche l'orientamento, dettaglio importante in alcuni contesti.

Tuttavia, l'integrazione di Meta2 con queste librerie, come ad esempio Vuforia, è difficile da realizzare dal momento che, come abbiamo detto, il sistema di riferimento è fissato indipendentemente dalla posizione attuale della camera e quindi è difficile ottenere una misurazione corretta del punto individuato.

Inoltre, tecnologie completamente marker-based e completamente markerless non sono compatibili e lavorare su una integrazione potrebbe rivelarsi estremamente laborioso sebbene potrebbe effettivamente portare risultati di migliore precisione.

Il secondo approccio è più semplice a livello di prototipazione ma generalmente meno preciso perché non individua l'orientamento del marker e si affida alla corretta interpretazione dell'utente.

Per lo sviluppo di questa tesi l'obiettivo non è tanto cercare di raggiungere una registrazione perfetta, quanto più raggiungere un'operabilità del dispositivo Meta2 all'interno del sistema, per questo dopo alcuni test sperimentali effettuati con successo si è scelto di procedere con la seconda strada.

Il meccanismo di registrazione risultante è estremamente semplice, all'inizio dell'applicazione l'utente vede davanti a sè un GameObject che rappresenta il marker virtuale. A questo GameObject è assegnato il comportamento di rimanere ad una distanza fissa dal casco in modo che l'utente lo veda sempre frontale davanti a sè. (Figura 4.3) In questo modo l'utente deve sovrapporre il marker virtuale a quello fisico e premere un pulsante integrato nel visore di comodo utilizzo per confermare l'avvenuta registrazione.

Le assunzioni che vengono fatte in questa tecnica sono che il marker visualizzato virtualmente sia della stessa dimensione reale del marker fisico. Questo

è necessario perché la corretta sovrapposizione deve avvenire solo quando l'utente si trova nella giusta posizione in relazione al mondo fisico per impostare l'origine del mondo.

La seconda assunzione è sull'orientamento del marker che non può essere rilevato da questa tecnica, se non appesantendo la fase di registrazione da parte dell'utente che deve ruotare il marker virtuale in relazione a quello fisico.

Si assume quindi che il marker sia posizionato su una parete verticale e sia appeso correttamente. Questa assunzione è necessaria, ma giustificabile dal momento che per lavorare con un visore questa è la soluzione più ergonomica di posizionamento.

Inoltre il marker deve necessariamente essere installato in una precisa posizione dello spazio per assicurare l'aderenza tra il mondo virtuale e quello fisico indi per cui dal momento che è lo sviluppatore a scegliere come posizionarlo sta a lui rispettare questi vincoli.

Date queste assunzioni il sistema di assi cartesiani individuato è creato utilizzando quello standard dato dal riconoscimento dei marker ovvero prendendo come riferimento il punto nello spazio di coordinate di Meta2 a cui corrisponde il centro del marker virtuale e con gli assi orientati con l'asse Y uscente perpendicolarmente dal marker, e il piano X-Z complanare al marker e orientato come l'orientamento del marker.

Il sistema di registrazione risultante si è rivelato essere un buon compromesso tra precisione e semplicità d'uso. Può indubbiamente essere migliorato aggiungendo tecniche di riconoscimento di immagine ad esempio su una istantanea scattata alla pressione del pulsante, oppure studiando un modo più preciso per contrastare gli errori di parallasse dovuti a utenti più o meno alti, ma per il risultato necessario per questa integrazione si è rivelato più che sufficiente.

4.4.2 Implementazione del MetaDevice

Messo a punto un sistema di registrazione efficace il lavoro della tesi si è concentrato sulla realizzazione pratica di un componente che fosse sufficientemente semplice da utilizzare e che allo stesso tempo contenesse tutte le funzionalità necessarie per abilitare gli sviluppatori ad utilizzare MiRAgE attraverso il dispositivo Meta2.

La scelta è stata quella di realizzare un prefab che raccogliesse al suo interno l'insieme di script e componenti necessari all'utilizzo della piattaforma.

Il `MetaDevice` come illustrato nella figura 4.4 si compone quindi dei due prefab forniti dall'SDK di Meta `MetaCameraRig` e `MetaHands` che rispettivamente abilitano la camera del casco come camera nella scena unity e il sup-

Figura 4.3: Il marker virtuale sovrapposto a quello fisico all'avvio della fase di registrazione

porto per il riconoscimento delle mani dell'utente e delle loro interazioni con gli oggetti.

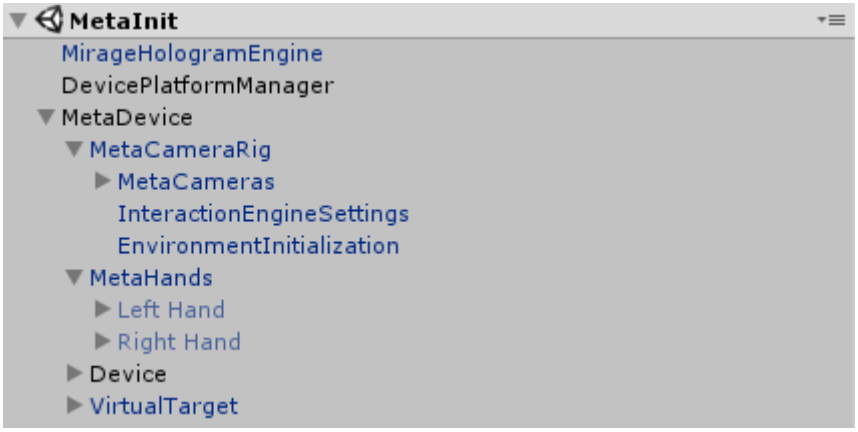

Figura 4.4: La struttura del MetaDevice visualizzato nella Hierachy Window di Unity

Questi due prefab fondamentali sono necessari per sviluppare qualsiasi applicazione per Meta2. In particolare il componente della camera permette di abilitare il tracking dell'ambiente e settare alcune impostazioni del rendering degli ologrammi come ad esempio l'occlusione dinamica da parte di altri oggetti tridimensionali rilevati, una feature caratteristica di Meta2 che rende la

visualizzazione molto più realistica. Il componente `MetaHands` si occupa invece di attivare o disattivare i gameObject relativi alla mano destra e sinistra dell'utente a seconda se questi vengono riconosciuti dal sistema oppure no. Inoltre è in grado di rilevare la gesture del *grab* quando un utente chiude la mano per interagire con un ologramma.

Oltre a questo è stato aggiunto il `MetaPlatform` uno script esteso dalla classe astratta DevicePlatform che si occupa di gestire la comunicazione tra le API del sistema con il layer di interoperabilità progettato. Nel dettaglio il `MetaPlatform` va a realizzare il sistema di registrazione descritto nella sezione precedente, registrandosi all'evento di pressione del pulsante presente sul caschetto fa in modo di restituire il GameObject che rappresenti l'origine del mondo da utilizzare come parent per tutti gli ologrammi all'interno del `DevicePlatformManager` e di orientare correttamente il sistema di riferimento come osservato nella descrizione della problematica della registrazione. Per questa implementazione prototipale la rotazione da applicare al sistema di riferimento di base di Meta2 è stata lasciata come parametro modificabile dall'esterno per una maggiore libertà.

Idealmente questo componente dovrebbe rendere il lavoro dello sviluppatore molto semplice, una volta migliorata l'interfaccia grafica in modo da semplificare le dipendenze tra i diversi componenti, per creare una applicazione con MiRAgE dovrebbe essere sufficiente fare drag and drop del prefab dell'HologramEngine con al suo interno il `DevicePlatformManager` su cui selezionare la piattaforma di riferimento tra quelle supportate. Questo dovrebbe poi automaticamente istanziare il prefab del `MetaDevice` su cui impostare alcuni parametri di base riguardanti ad esempio la dimensione del marker virtuale da visualizzare.

4.5 Validazione attraverso un caso di studio

Uno dei fattori che contraddistinguono la Realtà Mista come abbiamo detto è la sua aderenza al mondo reale e la possibilità di creare esperienze immersive per l'utente.

Per questo motivo, piuttosto che realizzare un'implementazione prototipale completamente artificiale, è stato scelto di inserire il lavoro di questa tesi in un contesto reale con il quale il PSLab[1], sede del lavoro svolto, collabora da tempo per la creazione di mondi aumentati.

[1] `http://apice.unibo.it/xwiki/bin/view/PSLab/WebHome`

4.5.1 Il mondo aumentato a Rocca delle Caminate

Rocca delle Caminate[2] è un castello situato su un colle a circa quattro kilometri da Predappio, nella provincia di Forlì-Cesena. Di origine antica, ma distrutto e ricostruito più volte nel corso degli anni attualmente è il frutto di un restauro avvenuto durante il periodo fascista che lo ha portato ad essere uno dei luoghi legati a Mussolini e alla fondazione della Repubblica Sociale Italiana verso la fine della seconda guerra mondiale.

Attualmente, dopo un nuovo restauro, è diventata sede del Tecnopolo di Forlì-Cesena cosa che la rende un contesto all'avanguardia, perfetto per lo sviluppo di Realtà Aumentata dal momento che unisce un luogo di rilievo storico ad un interesse per le nuove tecnologie.

Gli spazi della Rocca sono diventati quindi teatro della sperimentazione di mondi aumentati, in particolare si è scelto di sfruttare l'ampio cortile per popolare il castello con elementi di epoca medioevale per rendere più suggestiva l'esperienza dei visitatori e metterli in contatto con questa tecnologia. (Figura 4.5)

L'idea è quella di creare nel corso del tempo un vero e proprio mondo parallelo in cui personaggi e oggetti possano vivere all'interno del castello ed avere quindi un comportamento articolato.

È in questo contesto che si va ad inserire il lavoro svolto in questa tesi, in particolare con la resa digitale tramite Realtà Mista, di un libro del 1617[3] che racconta la storia delle famiglie che hanno abitato la Rocca nel corso degli anni.

La copia originale è conservata nella Biblioteca Nazionale Centrale di Firenze, ma, attraverso delle scansioni digitali applicate al modello tridimensionale di un libro, l'intento è quello di permettere ai visitatori della Rocca di sfogliarlo e poterlo leggere in modo semi-naturale.

L'AugmentedBook non rappresenta quindi un vero e proprio mondo aumentato, ma parte di un mondo più grande all'interno del contesto della Rocca. Per la sua natura, la possibilità di utilizzare le mani per interagire in modo diretto ci ha spinti ad indagare sull'integrazione del visore Meta2 che, non adatto per esplorare un mondo grande come quello di Rocca delle Caminate, si presta bene per contesti più riservati.

Nel corso del tirocinio svolto all'interno del laboratorio è stata realizzata un'esperienza con Meta2 nel contesto di una dimostrazione al pubblico di come la Realtà Mista possa arricchire la visita al castello.

Questa opportunità di confronto con gli utenti è stata utile per evidenziare come il supporto condiviso possa effettivamente migliorare l'esperienza gene-

[2] `http://www.roccadellecaminate.it/`

[3] Genealogia dell'antica famiglia detta delle Caminate, de' Belmonti, e de' Ricciardelli

rale. Infatti i visitatori potevano utilizzare il libro uno per volta ed osservare quello che l'utente attuale stava facendo attraverso un monitor che mostrava il libro dalla sua prospettiva.

Diversa cosa sarebbe una condivisione contestuale attraverso la quale gli utenti "di passaggio" potrebbero osservare le azioni svolte da chi utilizza il visore dalla propria prospettiva, avvicinarsi se curiosi di leggere qualche parola o addirittura utilizzando il libro collaborativamente dal momento che spesso i visitatori si muovono in piccoli gruppi.

Il caso di studio dell'AugmentedBook offre quindi un punto di partenza per realizzare un'esperienza che sia effettivamente immersiva e utile per gli utenti. In particolare, il caso specifico di un libro permette di avere un oggetto dal comportamento relativamente semplice da modellare, ma sufficientemente complesso per permettere delle interazioni di diversa natura.

Questo equilibrio tra comportamento ed interazione rende questo caso di studio perfetto per approfondire la tematica della condivisione e validare il modello proposto.

Figura 4.5: Una parte del mondo aumentato presente alla Rocca delle Caminate: elementi virtuali posizionati nel cortile.

4.5.2 AugmentedBook: Logica applicativa nel mondo aumentato

Come abbiamo detto il caso di studio di un libro si presta bene per realizzare un'entità aumentata interattiva ma sufficientemente semplice da modellare.

Le entità aumentate, seguendo la logica del paradigma a oggetti, sono dotate di proprietà che ne descrivono lo stato e azioni che possono essere invocate dagli agenti del sistema per alterare lo stato. Per quel che riguarda le azioni si è scelto di limitare gli utenti e gli agenti abilitandoli solamente ad aprire e chiudere il libro e a leggerlo sfogliandone le pagine.

Per descrivere lo stato del libro è quindi necessario avere mantenere due proprietà, una che identifica se il libro è aperto o chiuso e una che mantiene il numero di pagina.

È stato quindi definito un *template* per poter istanziare entità aumentate con le caratteristiche individuate:

```
1  @HOLOGRAM(geometry = "AugmentedBook")
2  public class AugmentedBook extends AE {
3
4     @PROPERTY private int pagenum = 0;
5     @PROPERTY private bool isopen = false;
6
7     @ACTION public void openBook(int pageNum) {
8        try {
9           customProperty("pagenum",pageNum);
10          customProperty("isopen", true);
11       } catch (PropertyNotFoundException e) {
12          e.printStackTrace();
13       }
14    }
15
16    @ACTION public void closeBook() {
17       try {
18          customProperty("isopen", false);
19       } catch (PropertyNotFoundException e) {
20          e.printStackTrace();
21       }
22    }
23
24    @ACTION public void goToPage(int pageNum) {
25       try {
26          customProperty("pagenum",pageNum);
27       } catch (PropertyNotFoundException e) {
```

```
28          e.printStackTrace();
29      }
30    }
31  }
```

Listato 4.3: Il template dell'entità aumentata descritta in java

Vediamo innanzitutto come in MiRAgE ogni entità aumentata deve necessariamente estendere dalla classe `AE` che gestisce alcune importanti proprietà dell'entità come ad esempio la posizione e rotazione rispetto al sistema di riferimento del mondo aumentato.

L'annotazione `@HOLOGRAM` invece definisce il nome del modello tridimensionale che verrà istanziato per rappresentare l'entità aumentata nelle varie applicazioni client.

Per quel che riguarda le proprietà aggiunte dall'utente queste vengono aggiornate tramite le azioni, metodi invocabili dagli agenti e devono essere aggiornate tramite il metodo `customProperty()` in modo che il sistema provveda alla notifica dell'aggiornamento della proprietà per tutti i client. Questo garantisce che il mondo aumentato venga aggiornato in modo coerente tra i vari osservatori umani.

Una volta definito un template di entità aumentata è necessario fare in modo che questa venga istanziata all'interno del mondo aumentato.

Dal momento che all'interno di MiRAgE la manipolazione delle entità aumentate è gesitita dagli agenti del sistema è necessario creare un agente che si occupi di creare un'istanza del libro.

Per fare questo è stato realizzato un agente cognitivo BDI (Belief Desire Intention) in Jason[3] che si occupasse di entrare nel mondo aumentato e successivamente creare l'istanza del libro.

L'azione `joinAW()` permette di ottenere dal server un identificatore di sessione da utilizzare per le successive modifiche al mondo aumentato. Questo meccanismo certifica che gli agenti sono autorizzati a manipolare le entità esistenti o a crearne di nuove.

Come è possibile osservare nel codice seguente alla creazione del libro viene anche stabilita la sua posizione e rotazione relativa all'origine degli assi. Le altre proprietà sono invece inizializzate come definito nel template.

```
1  +!start
2    <-    .print("=== MAIN AGENT ===");
3       makeArtifact("woatClient-BookAgent",
            "mirage4agents.WoatClient", ["127.0.0.1"], WoatClient);
4       focus(WoatClient);
5       lookupArtifact("woatTools", _);
6       joinAW("augmentedbook").
```

```
7
8  +join_done("augmentedbook")
9     <- sessionIdOnAw("augmentedbook", SessionId);
10       .concat("Agent has joined the aw! Session ID = ", SessionId,
             Msg);
11       println(Msg);
12       !createbook("book01").
13
14 +!createbook(Name)
15    <- cartesianLocation(0.5, 0.20, -0.79, L);
16       angularOrientation(90, 0, 0, 0);
17       createAE("augmentedbook", Name, "AugmentedBook", L, 0).
18
19 +ae_created("book01")
20    <- println("entity book created!").
```

Listato 4.4: L'agente deputato alla creazione dell'istanza del libro nel mondo aumentato

4.5.3 AugmentedBook: User App

Una volta configurato correttamente il mondo aumentato, la restante parte della logica applicativa dell'applicazione va gestita all'interno della User App.

Per questo esempio sono state realizzate due applicazioni client per poter effettivamente testare la condivisione su due diverse piattaforme. Come già accennato in precedenza le piattaforme utilizzate sono state un tablet Android con supporto per ARCore e un HMD Meta2.

All'interno dell'SDK per Unity di MiRAgE è fornita una classe astratta `HologramController` da estendere e accoppiare al GameObject da istanziare come rappresentazione grafica dell'entità aumentata. Questo componente riceve dall'`HologramEngineBridge` le modifiche delle proprietà e deve essere esteso dall'utente per intepretare correttamente l'update delle proprietà specifiche delle varie entità aumentate.

Estendendo l'`HologramController`, quindi si riesce a realizzare la parte di logica applicativa necessaria a visualizzare correttamente gli ologrammi e ricevere gli update da parte degli agenti che modificano le entità aumentate.

Tuttavia, in un contesto condiviso, è necessario considerare un certo grado di collaborazione tra gli utenti. Per questo motivo è necessario catturare gli input in relazione agli ologrammi e propagare le modifiche di stato al mondo aumentato.

Per cercare di individuare un pattern di sviluppo della stessa applicazione per diversi dispositivi attraverso MiRAgE si è cercato di progettare una lo-

gica applicativa che fosse sufficientemente versatile per adattarsi alle diverse piattaforme.

Per questo motivo il primo passo è stato quello di replicare un modello logico dell'entità aumentata che tenesse localmente traccia delle modifice di stato al libro.

Per mantenere la generalità riguardo la rilevazione dell'input si è scelto di realizzare un'architettura basata su un controller che permettesse di catturare gli eventi scaturiti dall'input dell'utente e applicarli al modello locale del libro. Nel compiere il suo ruolo il controller deve anche notificare il mondo aumentato quando un'azione significativa è stata effettuata.

Questo meccanismo si è rivelato efficace in quanto è stato semplice realizzare due diverse versioni dell'applicazione per le quali è stato sufficiente implementare solamente il componente deputato alla ricezione dell'input da collegare al controller.

In particolare per Meta2 si è scelto di sfruttare il supporto fornito dal riconoscimento delle mani dell'utente e della gesture principale del *grab* per identificare quando l'utente "prende" la copertina per aprire o chiudere il libro. Per sfogliare le pagine invece è sufficiente mettere la propria mano aperta sulla pagina e scorrere parallelamento al libro.

Meta2 sfrutta un sistema di eventi per notificare quando le mani dell'utente collidono con un oggetto virtuale. Sono state quindi aggiunte al modello tridimensionale del libro delle *HitBox* invisibili in punti strategici per rilevare con quale parte del libro l'utente sta interagendo e gli eventi sono stati propagati ad un componente dedicato a rilevare gli input e inviarli al controller del libro.

Per quel che riguarda ARCore le azioni di apertura e chiusura sono state implementate con delle gesture a due dita (Spread e Pinch) mentre per sfogliare le pagine è sufficiente far scorrere il dito sullo schermo.

4.5.4 Validazione

La validazione della soluzione proposta è stata effettuata valutando l'esperienza dell'utente nell'utilizzo dell'applicazione reallizzata come caso di studio. I risultati ottenuti da questa fase sono, quindi, di natura qualitativa, dal momento che, per applicazioni di questo tipo, diventa difficile effettuare una validazione di tipo quantitativo.

In generale, le applicazioni in Realtà Mista, ed in particolar modo le esperienze di condivisione come quella realizzata per questa tesi, devono necessariamente tener conto di quella che è la percezione che gli utenti finali hanno nell'utilizzo dell'applicazione stessa. Il successo di una esperienza viene, infatti, da quanto questa risulta credibile, immersiva ed intuitiva agli occhi di chi la vive, un dato difficile da misurare in modo oggettivo.

Va inoltre notato che questo aspetto, nel contesto di un'esperienza multi-piattaforma come quella realizzata, viene ulteriormente mediato dalle caratteristiche delle piattaforme in uso che, in base alle potenzialità, offrono gradi di immersione anche molto diversi agli utenti che le utilizzano.

Al contrario, la valutazione delle performance della soluzione proposta inserita all'interno del contesto di rete di MiRAgE, sono state volutamente ignorate in quanto valutare l'attività del framework non rientra tra gli obiettivi di questa tesi e la misurazione dipende fortemente, ancora una volta, dalla piattaforma utilizzata e dalla tipologia di rete impiegata.

Nel caso specifico dell'esempio realizzato una delle differenze principali è che il visore Meta2 offre la possibilità di occlusione dinamica degli oggetti virtuali in relazione agli oggetti reali, mentre Google ARCore, per la sua natura di sovrapposizione digitale e non ottica, mostra il contenuto virtuale sempre sovrapposto allo stream video ottenuto dalla fotocamera.

Questo incide negativamente sulla capacità di effettuare una valutazione tecnica, motivo per cui si è optato per una valutazione qualitativa dell'esperienza utente ed in particolare sulla percezione dell'oggetto condiviso come unica entità.

In particolare, i test effettuati coinvolgendo entrambe le piattaforme hanno cercato di valutare due aspetti principali: la coerenza degli aggiornamenti di stato e delle interazioni e la registrazione spaziale.

Per la prima viene notato che, al netto della latenza imposta dal sistema, gli aggiornamenti si propagano in modo sufficientemente fluido. In particolare, per quel che riguarda il tablet con ARCore la visualizzazione è fortemente penalizzata dalla mancanza di occlusione che impedisce di percepire correttamente le azioni compiute dalle mani di chi indossa Meta2.

Oltre a questo, per alleggerire il carico complessivo del sistema, si è scelto di propagare solamente l'evento di apertura e chiusura. Questo comporta una percezione distorta dal momento che l'utente che utilizza Meta2 vede la copertina del libro muoversi contestualmente con la propria mano che la manipola, mentre lo spettatore da ARCore vede il libro aprirsi solo una volta che l'azione è stata effettivamente completata.

Questo genere di problematiche sono comunque accettabili, dal momento che, come abbiamo detto, non tutte le piattaforme hanno le stesse capacità e di conseguenza l'esperienza non può essere esattamente identica per tutti gli utenti. IL risultato importante resta una coerenza dello stato per tutto lo svolgimento dell'applicazione. (Figura 4.6)

Inoltre, a seconda del numero di utenti coivolti e della tipologia di rete utilizzata, si potrà valutare di aumentare la frequenza degli aggiornamenti per migliorare la resa grafica complessiva anche per gli utenti che osservano gli altri interagire.

Figura 4.6: Un utente con Meta2 apre il libro mentre viene osservato tramite il tablet con ARCore. Osservando i due schermi è possibile notare come lo stato è coerente ma la percezione è diversa durante lo svolgimento delle azioni.

Per quel che riguarda la registrazione, il metodo più preciso per valutare la posizione dell'oggetto virtuale è quello di metterlo a confronto con degli oggetti reali. Per questo motivo è stato scelto di "appoggiare" virtualmente il libro su un tavolo e valutare la sua posizione in relazione ad esso da diverse prospettive.

Come è possibile notare dalla figura 4.7, la distanza tra le visualizzazioni di Meta2 e ARCore è nell'ordine di qualche centimetro. In particolare quello che si nota dai test effettuati è che la differenza significativa si ha sull'asse Y rispetto all'origine individuata tramite il marker probabilmente perché è più difficile per l'utente che si registra con Meta2 valutare correttamente la distanza a cui si trova il marker virtuale.

Questo risultato è comunque sufficiente per dare l'impressione agli utenti di guardare e interagire con lo stesso oggetto nel mondo virtuale.

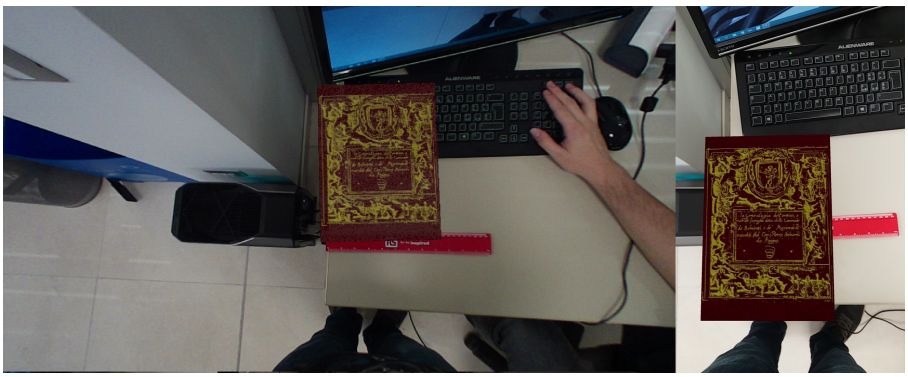

Figura 4.7: Lo stesso oggetto aumentato visto dai dispositivi da prospettive simili. A sinistra Meta2, a destra ARCore.

Conclusioni

Al termine del lavoro svolto questa tesi ha contribuito allo sviluppo di applicazioni condivise in Realtà Mista in due modi.

In primo luogo, fornendo un'analisi dettagliata del panorama delle ricerche su questo tema ed in particolare delle tecnologie abilitanti, esplorando le possibilità di realizzazione di uno scenario di condivisione utilizzando gli strumenti attualmente esistenti a disposizione degli sviluppatori.

Successivamente, l'integrazione di un supporto multi-piattaforma all'interno del framework MiRAgE ha fornito dei template semplici da riutilizzare per sviluppare nuove applicazioni basandosi sulla logica dei mondi aumentati.

Nel dettaglio, l'estensione realizzata per il framework è stata progettata per rimanere il quanto più isolata possibile sia dalla logica applicativa sia dall'infrastruttura sottostante in modo da poter essere modificata o ampliata, se necessario, senza troppe difficoltà.

L'obiettivo principale dell'integrazione era realizzare quindi un modo semplice per poter rendere operative nuove piattaforme all'interno dell'infrastruttura in previsione dell'aumento di diverse tecnologie che potranno in futuro emergere nella scena della Mixed Reality.

Per poter valutare la qualità del lavoro svolto è stato scelto di supportare l'HMD Meta2 per realizzare applicazioni più immersive per l'utente grazie alle potenzialità del visore. Questa integrazione si è rivelata efficace, seppure non perfetta dal punto di vista della registrazione che potrà essere migliorata in lavori futuri integrando tecniche di computer-vision.

Oltre a questo è stato rimodellato il supporto per la piattaforma Google ARCore già esistente nel framework abilitado due piattaforme differenti e permettendo di realizzare un caso di studio per studiare gli effetti della condivisione su due dispositivi con caratteristiche e immersività completamente differenti.

I risultati ottenuti sono stati buoni, la condivisione si è rivelata efficace e, sebbene non estremamente precisa, gli utenti del sistema possono dire di star guardando lo stesso ologramma anche se ovviamente ne possiedono una copia locale.

Lavorare con due dispositivi così diversi ha permesso di notare come le differenze a livello di funzionalità incidono nella resa della condivisione come ad esempio la mancanza di occlusione su ARCore fa visualizzare in modo incoerente le azioni che l'utente con Meta2 fa utilizzando le mani. Queste differenze, note a priori, non intaccano comunque la validità della soluzione proposta che, conscia delle limitazioni, non pone l'accento su una visione costantemente coerente che sarebbe di difficile realizzazione, bensì sul mantenimento di uno stato consistente del mondo aumentato

In generale il lavoro svolto ha messo in evidenza i punti di forza di un modello concettuale di condivisione come quello degli Augmented Worlds e ha evidenziato alcuni aspetti critici dell'infrastruttura di MiRAgE in particolare in merito alla gestione degli input degli utenti in relazione alla manipolazione diretta degli ologrammi possibile solo con un supporto come Meta2.

Il risultato ottenuto ha comunque soddisfatto il requisito più importante ovvero rendere velocemente operativi allo sviluppo di applicazioni condivise attraverso MiRAgE fornendo dei prefab intuitivi per lavorare con le diverse piattaforme e limitando il numero di azioni necessarie

Questo traguardo è fondamentale nella strada per portare il framework ad essere un elemento vivo ed utilizzabile anche all'esterno del laboratorio in modo da poter ricevere i contributi anche da parte di altri ricercatori.

Sviluppi futuri

Come ripetuto più volte questo lavoro si è concentrato sul rendere agevole lo sviluppo di applicazioni condivise tra piattaforme diverse in Realtà Mista.

Questo significa, ovviamente, che sono molti gli scenari futuri che si potranno sviluppare sulle basi di questo lavoro.

In primis, naturalmente, l'estensione del supporto ad altre nuove piattaforme per ampliare ancora di più il panorama delle tecnologie abilitate alla partecipazione nei mondi aumentati.

Oltre a questo, dal momento che il lavoro è stato realizzato con l'intento di essere velocemente operativi con Meta2, il sistema di registrazione del dispositivo potrà essere ripensato e rivisto per aumentarne la precisione dal momento che, affidandosi alla conferma da parte dell'utente, soffre dell'errore umano di chi compie la misurazione ed inoltre, come illustrato nella sezione dedicata, funziona sulle basi di alcune assunzioni che non possono essere ritenute valide per tutti i possibili contesti. In particolare andrà valutato meglio un approccio basato sul riconoscimento di marker utilizzando tecniche di computer-vision per stimare correttamente la distanza del marker dall'osservatore.

Indubbiamente un miglioramento importante potrebbe venire dal raffinamento dell'interfaccia grafica dei prefab realizzati in modo da rendere ancora più semplice per un utente inesperto la creazione di nuove applicazioni e permettere la pubblicazione di MiRAgE come framework non solo funzionante ma anche intuitivo.

Per quel che riguarda MiRAgE il lavoro svolto ha aggiunto una sezione importante e ha permesso di testare il framework in condizioni particolari, mettendo in evidenza alcune questioni riguardo la propagazione degli input da parte degli utenti umani che andranno risolte con un sistema di autenticazione che permetta di non retropropagare gli eventi generati da uno specifico utente dal momento che attualmente questo comporta alcune incongruenze grafiche che possono danneggiare la resa complessiva dell'esperienza.

Ringraziamenti

Al termine di questo percorso lungo tre anni mi sento di ringraziare per primo chi questo percorso con me lo ha iniziato, il mio "amico di vecchia data" Lorenzo che mi ha spronato ad intraprendere con lui questa avventura che si è rivelata un grande successo per entrambi.

Ci tengo a ringraziare tutti gli amici che ho conosciuto vivendo a Cesena, tra conquilini, vicine di casa, compagni di corso, lo SpaceTeam e i membri dell'associazione S.P.R.I.Te. che mi hanno fatto trovare una seconda casa sempre accogliente e mi hanno fatto vivere l'esperienza universitaria a pieno.

Un ringraziamento speciale va alla mia ragazza Giada che in questo periodo così impegnativo ha saputo starmi vicino come nessun altro sopportando anche i numerosi "discorsi da informatico".

Ringrazio i miei amici di sempre, ora diventati "di Ancona", per essere stati presenti ad ogni mio ritorno come se non fossi mai andato.

Dedico questo lavoro alla mia famiglia che mi ha dato la possibilità, mai scontata, di seguire il mio sogno e di studiare in un'altra città, sostenendomi sempre nonostante i sacrifici e gioendo con me dei miei traguardi.

Infine, ringrazio il professor Alessandro Ricci per avermi permesso di lavorare a questa tesi che mi ha fatto crescere e ha aperto i miei orizzonti. Un sentito grazie va al dottor Angelo Croatti che mi ha accompagnato personalmente in ogni passo durante la mia esperienza nel PSLab facendomi conoscere la vita all'interno di un laboratorio di ricerca e trasmettendomi la sua passione e dedizione per questo progetto, trattandomi sempre alla pari e dandomi preziosi consigli che porterò con me nella mia carriera futura.

Bibliografia

[1] I. Asimov. *I, Robot*. 1950.

[2] R. Azuma. A survey of augmented reality. 1997.

[3] R.H. Bordini, J.F. Hübner, and M. Wooldridge. *Programming Multi-Agent Systems in AgentSpeak using Jason*. 2007.

[4] D. Breen, R. Whitaker, E. Rose, and M. Tuceryan. Interactive occlusion and automatic object placement for augmented reality. 1995.

[5] A.B. Craig. *Understanding Augmented Reality - Concepts and Applications*. 2013.

[6] A. Croatti and A. Ricci. Mashing up the physical and augmented reality: The web of augmented things idea. 2017.

[7] A. Croatti and A. Ricci. Developing agent-based pervasive mixed reality systems: the mirage framework. 2018.

[8] A. Croatti and A. Ricci. A model and platform for building agent-based pervasive mixed reality systems. 2018.

[9] P. Milgram and F. Kishino. A taxonomy of mixed reality visual displays. 1994.

[10] J. Peddie. *Augmented Reality: Where We Will All Live*. 2017.

[11] J. Rousseau. The laws of mixed reality: A vision of the future, without the rose-colored glasses. `https://www.geekwire.com/2016/guest-post-laws-mixed-reality-mixed-reality-without-rose-colored-glasses/?curator=TechREDEF`, 2016. [Originally published on Artefact website].

[12] K. Ruth, T. Kohno, and F. Roesner. Secure multi-user content sharing for augmented reality applications. 2019.

[13] D. Schmalstieg, A. Fuhrmann, G. Hesina, Z. Szalavári, L. Encarnação, M. Gervautz, and W. Purgathofer. The studierstube augmented reality project. 2003.

[14] L. Tummolini, C. Castelfranchi, A. Ricci, and M. Piunti. In the mirror world: Preparing for mixed-reality living. 2015.