

POLITECNICO DI TORINO

**Corso di Laurea Magistrale in
Ingegneria del Cinema e dei Mezzi di Comunicazione**

Tesi di Laurea Magistrale

**Formazione Collaborativa in Realtà
Virtuale per Sistemi Industriali**



**Politecnico
di Torino**

Relatori

Prof. Andrea SANNA
Ing. Guido COPPO

Candidato

Nicola FANTINO

Ottobre 2025

Abstract

Negli ultimi anni la realtà virtuale (VR) ha acquisito una crescente importanza nel campo della formazione industriale, grazie alla possibilità di simulare procedure complesse in ambienti immersivi, riducendo i costi e aumentando la sicurezza. La letteratura mostra inoltre come le esperienze collaborative in VR possano favorire il coinvolgimento, la motivazione e l'apprendimento degli operatori, sebbene restino ancora aperte sfide legate alla standardizzazione delle soluzioni, alla scalabilità da contesti sperimentali a scenari industriali complessi e all'ottimizzazione delle prestazioni su dispositivi autonomi. In questo contesto, la presente tesi, svolta presso l'azienda SynArea Consultants, descrive lo sviluppo di un'applicazione multiutente VR in Unity dedicata alla formazione sull'utilizzo di una stampante 3D industriale, prendendo ispirazione da un applicativo già sviluppato dalla stessa azienda per dispositivi mobili, quali smartphone e tablet. Il progetto è stato realizzato usando come punto di partenza il VR Multiplayer Template, impiegando quindi Netcode for GameObjects come soluzione per la sincronizzazione dei dati e XR Interaction Toolkit per la gestione delle interazioni immersive. L'elemento centrale del sistema è costituito da un'architettura modulare basata su procedure suddivise in passi, che possono essere avviate sia in una modalità guidata, con elementi che accompagnano l'utente passo per passo, sia in una modalità di valutazione orientata alla verifica dell'apprendimento. Inoltre, più utenti possono avviare ed eseguire in parallelo differenti procedure, favorendo la collaborazione e permettendo agli operatori di supportarsi reciprocamente durante l'esecuzione. L'ambiente virtuale è stato realizzato avendo come priorità le prestazioni su dispositivi autonomi, come Oculus Quest 2. Il risultato è un sistema flessibile ed estensibile per la formazione industriale in VR, capace di adattarsi a differenti scenari e macchinari, e che rappresenta un passo concreto verso l'integrazione di soluzioni collaborative immersive nei processi di addestramento tecnico.

Indice

Elenco delle figure	IV
Elenco delle tabelle	V
Acronimi	VI
1 Introduzione	1
1.1 Contesto generale	1
1.2 Obiettivo	2
1.3 Requisiti	2
1.4 Struttura della tesi	3
2 Stato dell'Arte	5
2.1 Applicazioni VR per la formazione industriale: una revisione della letteratura	5
2.1.1 Analisi del compito	6
2.1.2 Progettazione preliminare dello scenario	6
2.1.3 Implementazione	6
2.1.4 Formazione della forza lavoro	7
2.2 Realtà virtuale e apprendimento collaborativo	8
2.2.1 Aspetti educativi della VRCL	9
2.2.2 Sistemi tecnologici e modalità di interazione nella VRCL	9
2.2.3 Caratteristiche progettuali della VRCL	10
2.2.4 Disparità e criticità nell'uso di sistemi HMD e non-HMD nella VRCL	11
2.2.5 Conoscenza empirica consolidata riguardo alla VRCL	11
2.3 La realtà virtuale per l'addestramento procedurale	12
2.4 La creazione di un ambiente virtuale multi-utente	13
2.5 Addestramento in un laboratorio virtuale multi-utente e multiplatforma	15
2.6 Considerazioni conclusive	16
3 Tecnologie utilizzate	19
3.1 Unity	19
3.1.1 XR Interaction Toolkit	19
3.1.2 OpenXR	20
3.1.3 Netcode for Gameobjects	20
3.1.4 Unity Services	22

3.1.5	Parrel Sync	23
3.2	Visual Studio	24
3.3	Blender	24
3.4	Meta Quest 2	24
3.5	Meta Quest Pro	24
4	Progettazione e sviluppo dell'applicazione	27
4.1	Architettura dell'applicazione	27
4.2	Componenti principali del VR Multiplayer Template	28
4.2.1	Struttura e funzionalità dell'avatar	29
4.2.2	Network Manager VR Multiplayer	32
4.2.3	XRI Network Game Manager	34
4.2.4	Offline Menù UI	34
4.2.5	Tutorial UI	35
4.2.6	Spatial Panel UI	36
4.3	Gestione e realizzazione delle procedure	37
4.3.1	Gestione delle procedure	37
4.3.2	Definizione delle procedure	38
4.3.3	Definizione dei passi delle procedure	39
4.3.4	Modalità guidata	40
4.3.5	Modalità di valutazione	42
4.3.6	Componenti impiegati per realizzare le procedure	43
4.3.7	Procedure realizzate	45
4.4	Ambiente virtuale	46
4.4.1	Modelli 3D e materiali	46
4.4.2	Illuminazione	47
5	Valutazione sperimentale e risultati	49
5.1	Test effettuati	49
5.1.1	Caratteristiche del campione	50
5.1.2	Metodologia dei test	50
5.2	Risultati dei questionari	51
5.3	Analisi dei risultati	54
5.4	Valutazioni aggiuntive	54
6	Conclusioni e sviluppi futuri	57
6.1	Considerazioni finali	57
6.2	Sviluppi futuri	57
	Bibliografia	61

Elenco delle figure

1.1	Applicazione di addestramento della stampante 3D già sviluppata.	2
2.1	Sistema di evidenziazione	13
2.2	Visualizzazione del secondo scenario sviluppato	14
2.3	Operatori che si esercitano sulle diverse piattaforme	16
3.1	Topologia Client-Host	21
3.2	Topologia distributed authority	22
3.3	Architettura di rete del servizio Relay	23
3.4	Meta Quest 2.	25
3.5	Meta Quest Pro.	25
4.1	Architettura software dell'applicativo.	28
4.2	Teleport.	30
4.3	Indicazioni dei comandi disponibili.	31
4.4	Menù collegato all'utente.	32
4.5	Visualizzazione dell'avatar di un utente	32
4.6	Schermata di selezione della lobby	35
4.7	Interfaccia che mostra il tutorial dell'applicazione	36
4.8	Menù manipolabile di selezione e avvio della procedura	37
4.9	Finestra di modifica delle informazioni dei passi definita con la reflection .	41
4.10	Shader dell'ologramma	42
4.11	Lista dei passi della procedura, con indicato il loro stato	43
4.12	Operatori che si esercitano nelle due procedure.	46
4.13	Rete di light probes definita.	48
5.1	Valori SUS per utente.	53
5.2	Valori NASA-TLX per utente.	53

Elenco delle tabelle

5.1	Distribuzione e deviazione standard (σ) delle risposte al questionario System Usability Scale	51
5.2	Distribuzione e deviazione standard (σ) delle risposte al questionario NASA-TLX	52
5.3	Distribuzione delle risposte per le domande aggiuntive.	52

Acronimi

VR Virtual Reality
XR Extended Reality
AR Augmented Reality
MR Mixed Reality
HTA Hierarchical Task Analysis
CTA Cognitive Task Analysis
VRML Virtual Reality Modeling Language
SDK Software Development Kit
API Application Programming Interfaces
CAVE Cave Automatic Virtual Environment
HMD Head-Mounted Display
VRCL Virtual Reality Collaborative Learning
PBL Problem-Based Learning
VE Virtual Environment
VRTS Virtual Reality Training System
TL Traditional Learning
LBT Learning By Teaching
NGO Netcode for GameObjects
UI User Interface
IDE Integrated Development Environment
UDP User Datagram Protocol
TCP Transmission Control Protocol
RPC Remote Procedure Call

Capitolo 1

Introduzione

Il capitolo introduttivo di questa tesi ha lo scopo di fornire una panoramica generale sul contesto e il dominio applicativo in cui si inserisce il lavoro svolto. Vengono inoltre illustrati gli obiettivi principali del progetto e i risultati attesi. Infine, viene presentata la struttura complessiva della tesi, offrendo una sintesi dei contenuti trattati nei capitoli successivi.

1.1 Contesto generale

Il lavoro presentato in questa tesi è stato svolto presso SynArea Consultants, azienda torinese attiva nel settore ICT con oltre quarant'anni di esperienza nello sviluppo di soluzioni software personalizzate per diversi ambiti applicativi, tra cui il settore industriale. La crescente adozione della realtà virtuale (VR) in questo campo ha aperto nuove possibilità per la formazione e l'addestramento tecnico.

La VR è una tecnologia che consente di riprodurre ambienti tridimensionali generati digitalmente, all'interno dei quali l'utente può osservare, muoversi e interagire in tempo reale. L'obiettivo è simulare esperienze sensoriali e motorie in grado di riprodurre, in modo più o meno realistico, le condizioni del mondo fisico. Attraverso l'impiego di dispositivi come i visori indossabili head-mounted display (HMD), che proiettano immagini stereoscopiche direttamente davanti agli occhi dell'utente, i controller manuali e i sensori di tracciamento del corpo, la realtà virtuale può assumere una forma immersiva, offrendo un'interazione diretta e naturale tra l'utente e l'ambiente digitale.

A differenza di altre forme di realtà estesa (XR), come la realtà aumentata (AR), che sovrappone elementi digitali al mondo fisico, o la realtà mista (MR), che integra elementi virtuali e reali facendoli interagire tra di loro, la VR immersiva isola completamente l'utente dall'ambiente fisico, consentendogli di concentrarsi esclusivamente sull'esperienza simulata. Questa caratteristica rende la tecnologia particolarmente adatta alla formazione e alla simulazione di processi complessi, specialmente in scenari in cui sicurezza e controllo delle variabili ambientali rivestono un ruolo fondamentale.

1.2 Obiettivo

In questo contesto, il presente lavoro di tesi si colloca nel filone di ricerca dedicato all'impiego della realtà virtuale immersiva come strumento di formazione e addestramento tecnico in ambito industriale.

Il progetto mira alla progettazione e sviluppo di un sistema di training multiutente, che consenta a più operatori di collaborare in un medesimo ambiente virtuale riproducendo procedure reali di utilizzo di macchinari complessi. In particolare, il caso di studio riguarda una stampante 3D industriale basata su tecnologia di produzione additiva a polveri metalliche, impiegata in diversi settori produttivi.

Come punto di partenza è stata analizzata un'applicazione preesistente, realizzata da SynArea Consultants per dispositivi mobili, con l'obiettivo di svilupparne una versione più immersiva e interattiva ottimizzata per visori HMD. Il nuovo sistema consente all'utente



Figura 1.1: Applicazione di addestramento della stampante 3D già sviluppata.

di esercitarsi nelle principali procedure operative del macchinario, integrando al contempo una dimensione collaborativa per la formazione simultanea di più operatori.

L'obiettivo complessivo è offrire un'esperienza di apprendimento più efficace, coinvolgente e sicura, sfruttando le potenzialità della VR per la simulazione di processi industriali complessi.

Infine, il lavoro intende analizzare l'impatto dell'interazione multiutente nel contesto dell'addestramento tecnico e definire un'architettura modulare e adattabile a differenti scenari produttivi.

1.3 Requisiti

I requisiti dell'applicazione che quindi si possono definire sono i seguenti.

Il primo requisito consiste nell'implementazione di uno scenario semirealistico che riproduca in modo fedele ma ottimizzato l'ambiente operativo reale. Tale scenario deve

essere progettato sulla base di un'analisi preliminare delle procedure e delle sequenze formative da realizzare.

Un secondo requisito riguarda la possibilità di formare più operatori contemporaneamente, configurando l'ambiente in modo che più utenti possano interagire in parallelo.

Un ulteriore requisito è quello di favorire la formazione collaborativa, in cui gli utenti possano osservare l'operato dei colleghi, coordinarsi nello svolgimento delle procedure, correggersi a vicenda e apprendere attraverso la cooperazione diretta. Questa modalità consente di replicare dinamiche di apprendimento più vicine alla realtà operativa e al lavoro di squadra.

È inoltre necessario prevedere la possibilità di svolgere le sessioni formative anche a distanza, garantendo quindi un'infrastruttura di rete stabile e performante, capace di supportare la sincronizzazione in tempo reale degli utenti e degli oggetti presenti nella scena virtuale.

Infine, un requisito fondamentale è quello legato alle prestazioni dell'applicazione, che deve risultare fluida e stabile anche su visori standalone di fascia economica. Ciò implica l'adozione di tecniche di ottimizzazione grafica e computazionale tali da mantenere un framerate elevato e un'esperienza immersiva confortevole, senza compromettere la qualità visiva o l'interattività del sistema.

1.4 Struttura della tesi

La tesi è articolata in sei capitoli, ognuno dei quali affronta un aspetto specifico del lavoro svolto, dalla definizione del contesto teorico fino alla valutazione sperimentale dell'applicativo sviluppato.

- **Capitolo 1:** introduzione al contesto di lavoro e agli obiettivi della tesi, presentando le motivazioni che hanno guidato lo sviluppo e fornendo una panoramica generale sull'ambito di ricerca.
- **Capitolo 2:** presentazione dello stato dell'arte, in cui vengono analizzate le principali soluzioni e studi presenti in letteratura, utili a definire il quadro di riferimento teorico e applicativo del progetto.
- **Capitolo 3:** descrizione delle tecnologie e degli strumenti impiegati, fornendo una visione complessiva dei mezzi adottati per la realizzazione del lavoro.
- **Capitolo 4:** esposizione della progettazione e dello sviluppo della soluzione proposta. Illustrazione dei componenti principali del sistema.
- **Capitolo 5:** presentazione e analisi dei risultati ottenuti, con una valutazione delle sperimentazioni condotte e delle osservazioni emerse durante i test.
- **Capitolo 6:** conclusioni del lavoro di tesi, con una sintesi dei risultati raggiunti, delle criticità riscontrate e dei possibili sviluppi futuri.

Capitolo 2

Stato dell'Arte

In questo capitolo vengono analizzate le principali evidenze emerse dalla letteratura scientifica riguardo all'impiego della realtà virtuale (VR), con particolare attenzione alle applicazioni per l'addestramento e la formazione in contesti industriali e collaborativi. Nella prima parte si discutono le tendenze individuate negli studi accademici, mettendo in luce vantaggi, limiti e criticità dei sistemi VR applicati all'apprendimento. Successivamente vengono presentati alcuni casi studio ritenuti particolarmente significativi per l'obiettivo di questa tesi, che forniscono esempi concreti di implementazioni esistenti.

2.1 Applicazioni VR per la formazione industriale: una revisione della letteratura

Lo studio presentato in B. Xie et al. [1] analizza in maniera sistematica gli sviluppi della ricerca accademica nell'ambito della realtà virtuale applicata alla formazione. In particolare, vengono approfondite le metodologie di progettazione e implementazione delle applicazioni di addestramento in VR, i principali strumenti hardware e software utilizzati, nonché i diversi domini applicativi e le soluzioni individuate in letteratura.

Negli ultimi anni la VR ha conosciuto una rapida diffusione grazie allo sviluppo di dispositivi destinati soprattutto al grande pubblico. Il dispositivo Oculus Quest 2 [2] è divenuto nel 2020 il visore più venduto di sempre. La VR oggi è sempre più utilizzata dalle aziende e il 62% delle applicazioni riguarda l'addestramento del personale. La formazione effettuata tramite metodi tradizionali presenta diversi limiti, richiede tempo e risorse per predisporre spazi appositi, è costosa per via delle attrezzature coinvolte e in alcuni casi risulta pericolosa o addirittura impossibile. La VR al contrario permette di ridurre i costi, di svolgere l'attività da remoto, aumenta la varietà degli scenari e garantisce la sicurezza del personale.

L'analisi presentata nello studio si limita esclusivamente alla VR, escludendo altre forme di realtà estesa (XR) come Augmented Reality (AR) e Mixed Reality (MR). Le applicazioni di addestramento VR sono state sviluppate per diversi settori, in generale però vengono progettate per compiti specifici, caratterizzati da procedure fisse e da obiettivi pratici, risultando quindi molto eterogenee tra loro. Alcuni studi, come H. Xie et al. [3], hanno cercato di proporre degli approcci più generali applicabili a diversi

scenari di addestramento. Altri studi, Martin et al. [4] e Martin e Hughes [5], hanno sviluppato software per la generazione automatica di scenari. Lin et al. [6] invece hanno proposto un'architettura per sistemi di addestramento in VR insieme a un'introduzione alla pianificazione dei compiti formativi virtuali e ad alcuni modelli di scenari.

In generale però la creazione di un sistema di addestramento VR può essere suddivisa in tre fasi principali:

- Analisi del compito
- Progettazione preliminare dello scenario
- Implementazione

2.1.1 Analisi del compito

Uno dei primi passi fondamentali nello sviluppo di sistemi di addestramento è rappresentato dall'analisi del compito, che costituisce la base per la definizione degli obiettivi formativi. Tale analisi ha lo scopo di individuare le informazioni necessarie per il compito specifico e di sintetizzare i fattori che devono essere presi in considerazione nella progettazione del sistema. Le prime applicazioni di questo concetto risalgono agli anni '40. Tuttavia, queste analisi erano ancora concentrate principalmente sul comportamento fisico e su decisioni semplici, enfatizzando quindi la procedura dettagliata del compito. Con il tempo, tali approcci a basso livello sono stati sostituiti da metodi più avanzati orientati al problem-solving, capaci di includere anche compiti cognitivi complessi. In particolare, l'Hierarchical Task Analysis (HTA), che descrive obiettivi composti da unità differenti, e la Cognitive Task Analysis (CTA), più focalizzata sulla gestione di situazioni impreviste, rappresentano i metodi più diffusi e consolidati, come confermato dal confronto sistematico di Salmon et al. [7].

2.1.2 Progettazione preliminare dello scenario

La progettazione preliminare dello scenario consiste nel fornire descrizioni dettagliate delle modalità con cui gli utenti portano a termine un determinato compito. Permette di definire gli scenari e i flussi delle attività e delle interazioni all'interno di essi.

I contributi degli studi riguardanti questa fase, come ad esempio in Farmer et al. [8] e in Papelis e Watson [9], hanno permesso di delineare un quadro teorico utile alla progettazione di nuove applicazioni VR e anche una base per lo sviluppo di approcci procedurali in grado di generare automaticamente ambienti virtuali.

2.1.3 Implementazione

La fase di implementazione rappresenta il momento in cui la progettazione viene tradotta in un'applicazione VR funzionante, attraverso la realizzazione concreta del software e l'integrazione delle tecnologie necessarie.

Nel tempo sono stati sviluppati diversi strumenti per la creazione di ambienti virtuali: tra i primi, il Virtual Reality Modeling Language (VRML) [10] introdotto nel 1994, seguito da altri linguaggi e formati. Oggi, i game engines come Unity 3D e Unreal Engine sono

tra le piattaforme più diffuse grazie alla disponibilità di pacchetti e Software Development Kit (SDK) compatibili con i principali dispositivi VR, offrendo Application Programming Interfaces (API) integrate che semplificano lo sviluppo.

La modellazione 3D è un altro elemento fondamentale: chi non possiede competenze specifiche può sfruttare risorse già pronte sugli asset store, mentre per la creazione personalizzata si ricorre a software avanzati come 3ds Max [11], Maya [12] e Blender [13]. Accanto a queste soluzioni, tecniche di generazione procedurale sono ampiamente adottate per ridurre i costi e i tempi di produzione di ambienti e modelli 3D, automatizzando la creazione di scenari complessi.

Lo sviluppo delle applicazioni VR è strettamente connesso anche ai progressi dell'hardware: in passato, a causa della scarsa potenza computazionale, venivano usati sistemi basati su finestre 3D, proiezioni a specchio, simulatori veicolari o Cave Automatic Virtual Environment (CAVE), ambienti immersivi generati tramite proiezioni sulle pareti e sul pavimento. L'avvento di dispositivi a relativamente basso costo come gli (HMD) ha reso l'accesso più semplice e diffuso, permettendo di migliorare abilità cognitive, psicomotorie e affettive. Oltre ai visori, dispositivi di interazione e tracciamento come Leap Motion [14] e Kinect [15] hanno trovato applicazione in diversi progetti di training, dalla riabilitazione motoria fino all'addestramento sportivo. Più recentemente, sensori di movimento a corpo intero e tute aptiche hanno incrementato la precisione del tracciamento e il grado di immersione, mentre piattaforme basate su tapis roulant simulano camminata e corsa in ambienti virtuali. Gli studi comparativi hanno mostrato che, pur con problemi ancora presenti come la motion sickness¹, gli HMD di nuova generazione risultano più efficaci rispetto a sistemi a bassa immersività, migliorando la qualità della formazione e i risultati dell'apprendimento.

2.1.4 Formazione della forza lavoro

Un ambito di crescente interesse riguarda la formazione della forza lavoro attraverso applicativi di realtà virtuale. Numerosi progetti industriali hanno infatti adottato la VR per ridurre la perdita di produttività legata alla formazione tradizionale e colmare la carenza di istruttori qualificati.

Un ulteriore valore della VR risiede nella capacità di riprodurre ambienti di lavoro realistici e ad alto rischio con elevata fedeltà, fornendo spazi sicuri che riducono lo stress dei neoassunti. Alcuni esempi significativi includono simulatori per il lavoro in quota, Di Loreto et al. [16], la manutenzione delle linee elettriche, Borba et al. [17], o la sicurezza nei cantieri edili, Sacks et al. [18], tutti orientati a migliorare la consapevolezza dei rischi e prevenire incidenti. La ricerca ha confermato come questi sistemi siano efficaci nell'insegnare l'identificazione dei pericoli e l'ispezione dei siti, oltre a garantire un addestramento accessibile e replicabile.

¹Il termine motion sickness indica un insieme di sintomi (nausea, vertigini, sudorazione, mal di testa) causati da un conflitto sensoriale tra ciò che l'occhio percepisce e i segnali vestibolari dell'orecchio interno. Nella realtà virtuale si parla spesso di cybersickness, una forma specifica di motion sickness indotta da dispositivi immersivi come gli HMD.

In ambito manifatturiero, la VR si sta affermando come strumento per l'assemblaggio e la produzione, dimostrandosi in grado di ridurre i tempi di esecuzione e il tasso di errore rispetto alla formazione tradizionale, Abidi et al. [19], oltre a mitigare rischi legati all'uso dei macchinari o alle emergenze, Lacko [20]. Tuttavia, la dimensione collaborativa multiutente in tempo reale rimane ancora una sfida aperta: lo studio di Yildiz et al. [21] ha proposto un modello coordinato per l'assemblaggio di turbine eoliche, capace di integrare fabbriche fisiche e virtuali superando i limiti legati alla sincronizzazione dei dati multiutente.

In conclusione, la VR applicata all'addestramento della forza lavoro si concentra principalmente su due direttrici: il potenziamento delle competenze operative e l'aumento della consapevolezza sui rischi per le professioni ad alto pericolo. Nonostante ciò, la formazione di gruppo per la risposta alle emergenze resta un ambito ancora poco sviluppato, mentre l'addestramento multiutente in VR in tempo reale rappresenta una sfida ingegneristica e di ricerca aperta. Inoltre, l'adozione di nuove tecnologie come i guanti VR per la cattura precisa dei movimenti manuali promette sviluppi futuri verso esperienze formative sempre più realistiche ed efficaci.

2.2 Realtà virtuale e apprendimento collaborativo: una revisione sistematica della letteratura

Lo studio presentato in van der Meer et al. [22] presenta una ricerca sistematica riguardante la Virtual Reality Collaborative Learning (VRCL). L'obiettivo è esaminare in che modo la VR supporti e migliori l'apprendimento collaborativo secondo la letteratura esistente. Per raggiungere tale scopo, il lavoro analizza: i campi di applicazione della VRCL nell'educazione, i benefici e le potenzialità in termini di apprendimento, le caratteristiche della VRCL che consentono il raggiungimento di tali benefici, le tecnologie alla base della VRCL, confrontandole con lo stato dell'arte della VR. Infine, lo studio mira a identificare le lacune di ricerca per studi futuri, oltre a mettere in evidenza i punti di forza della VRCL a supporto delle indagini attuali. Secondo gli autori, si tratta della prima revisione sistematica sul tema della VRCL. Per fornire informazioni rilevanti, la revisione affronta quattro domande di ricerca:

- Quali competenze e abilità sono state allenate con la VRCL e cosa dovrebbe fornire un ambiente VRCL per svilupparle?
- Quali domini e discipline sono stati esplorati e perché?
- Quali sistemi sono stati sviluppati o consolidati?
- Quale conoscenza empirica è stata prodotta e con quali metodi o disegni di studio?

Dopo aver compiuto una ricerca da due database di articoli accademici e ristretto il numero tramite dei parametri definiti, si sono classificati gli studi tramite una tassonomia e quindi analizzati.

2.2.1 Aspetti educativi della VRCL

La maggior parte della letteratura selezionata (64%), si concentra sulla VRCL nell'istruzione terziaria, discutendo i possibili usi per gli studenti. Sorprendentemente, una piccola percentuale degli studi (6,5%), si è concentrata su tipologie di studenti al di fuori dell'istruzione formale, come ad esempio la formazione del personale sul posto di lavoro. Per quanto riguarda gli educatori, poco più della metà degli studi (55,4%) ha riportato che essi supportavano gli studenti fornendo diversi livelli di scaffolding (ossia un supporto didattico temporaneo e graduale che viene progressivamente ridotto man mano che lo studente acquisisce autonomia). Nel 20,9% dei casi, invece, gli educatori fornivano presentazioni e lezioni all'interno dell'ambiente virtuale, offrendo quindi un'esperienza di apprendimento più passiva. La maggior parte degli studi si è focalizzata sulla collaborazione sincrona.

Per quanto riguarda gli approcci educativi e i paradigmi di apprendimento, il 29,5% degli articoli non sembra aver discusso approcci specifici. Tra quelli che lo hanno fatto, il costruttivismo e il Problem-Based Learning (PBL)² sono risultati particolarmente presenti (33,1% e 41,0% rispettivamente).

2.2.2 Sistemi tecnologici e modalità di interazione nella VRCL

La seconda dimensione ha preso in esame più da vicino i sistemi utilizzati negli studi, includendo aspetti legati all'hardware e all'interazione degli utenti con gli ambienti virtuali. Curiosamente, la maggior parte degli studi analizzati non ha utilizzato tecnologie VR basate su HMD, ma si è concentrata su monitor e display per discutere di VRCL. Alcuni casi più rari hanno adottato sistemi CAVE.

Molti studi hanno scelto controlli di uso generale, ad esempio mouse e tastiera, piuttosto che hardware più avanzati che permettono il tracciamento posizionale. La maggioranza degli studi ha fornito ai partecipanti embodiment virtuale completo (ossia la rappresentazione dell'intero corpo tramite avatar) e la possibilità di manipolare oggetti virtuali all'interno dell'ambiente. In alcuni casi (18,0%), gli attributi fisici erano rappresentati virtualmente da strumenti, spesso per training specialistico. Gli studi che hanno impiegato HMD VR hanno usato dispositivi come Oculus Rift [23] e HTC Vive [24]. Oltre ai classici dispositivi di controllo, sono stati discussi anche altri strumenti: tavoli multitouch, dispositivi aptici, Xbox Kinect, guanti sensoriali per il rilevamento dei gesti. In pochi casi si è fatto uso del controllo della direzione dello sguardo (15,1%) e del tracciamento posizionale (11,5%).

Gli studi che hanno menzionato finalità di addestramento (17,3%) si sono concentrati sull'uso di ambienti virtuali per insegnare competenze specifiche, come ad esempio l'ispezione di aeromobili. I casi di uso terapeutico sono stati rari (2,2%), come nell'impiego della VRCL per insegnare competenze sociali a pazienti con autismo o per l'allenamento fisico tra anziani.

²Il costruttivismo è una teoria dell'apprendimento secondo cui gli studenti costruiscono attivamente la propria conoscenza attraverso l'esperienza e l'interazione e non in modo passivo, mentre il *Problem-Based Learning* (PBL) è un approccio didattico che incentiva l'apprendimento attraverso la risoluzione collaborativa di problemi reali, che vengono posti all'inizio del percorso di formazione.

Infine, per quanto riguarda l'influenza dell'utente sugli ambienti, poco più della metà degli studi (53,2%) ha utilizzato sistemi che permettevano una certa manipolazione degli oggetti virtuali, circa un quarto (26,6%) consentiva di modificare direttamente i contenuti dell'ambiente VRCL.

In termini di risultati, circa la metà degli studi ha concluso che i sistemi analizzati fossero positivi e promettenti (53,2%), mentre il 17,3% ha riportato esiti positivi basati su risultati statisticamente significativi. I risultati negativi sono stati molto rari (2,2%). Alcuni studi hanno riportato esiti misti (7,2%), mentre il 20,1% ha discusso risultati inconcludenti, nulli o non applicabili a una distinzione netta tra positivo e negativo.

2.2.3 Caratteristiche progettuali della VRCL

Affinché le tipologie di abilità e competenze riscontrate in questa revisione possano essere allenare efficacemente, un ambiente VRCL deve includere una serie di caratteristiche che supportino gli studenti nello sviluppo di tali capacità. Dalla letteratura emergono quindi nove caratteristiche e parametri di design fondamentali:

1. **Embodiment virtuale:** svolge un ruolo cruciale nel modo in cui gli studenti percepiscono sé stessi e gli altri all'interno dell'ambiente virtuale, influenzando i risultati di apprendimento e i comportamenti collaborativi grazie a una maggiore consapevolezza e senso di appartenenza.
2. **Strumenti di comunicazione efficienti:** la comunicazione verbale è essenziale per una collaborazione efficace, ma modalità aggiuntive come la tecnologia aptica possono potenziare ulteriormente la collaborazione.
3. **Usabilità e accessibilità:** i sistemi VRCL devono essere accessibili a tutti i livelli di competenza tecnica, poiché le differenze tra gli utenti possono influire negativamente sulla coesione del gruppo e sull'apprendimento.
4. **Utilità percepita:** la percezione di utilità dell'ambiente virtuale da parte degli studenti influisce sulla coesione del gruppo; fattori come consapevolezza, presenza e presenza sociale hanno un impatto significativo su questa percezione.
5. **Interazione con gli elementi:** per ottimizzare l'apprendimento, gli studenti devono poter manipolare oggetti e strumenti virtuali in modo naturale e intuitivo.
6. **Efficacia accademica:** i compiti all'interno dell'ambiente virtuale devono essere progettati in linea con obiettivi educativi e collaborativi, soprattutto se prevedono un contributo equo da parte di tutti i membri del gruppo.
7. **Supporto degli educatori:** gli insegnanti devono essere pronti a fornire supporto, motivazione e moderazione durante l'interazione degli studenti nell'ambiente virtuale.
8. **Autonomia individuale:** ogni studente deve poter mantenere un certo livello di indipendenza, non solo dagli educatori ma anche dagli altri membri del gruppo, per fornire punti di vista differenti ed esplorare rappresentazioni multiple, migliorando così l'apprendimento collaborativo.

9. **Supporto alla socializzazione:** la progettazione della VRCL deve dare priorità alla socializzazione, poiché trascurarla potrebbe trasformare la VR in un ostacolo piuttosto che in un facilitatore dell'apprendimento collaborativo.

2.2.4 Disparità e criticità nell'uso di sistemi HMD e non-HMD nella VRCL

Dalla letteratura emerge una forte disparità tra l'uso di sistemi VR basati su HMD e quelli non-HMD. Quasi l'80% degli studi analizzati ha infatti impiegato soluzioni non-HMD, come monitor o display a superficie piatta, mentre l'HMD VR compare solo nel 7,2% dei casi. Questa scarsa presenza risulta singolare se si considera che l'HMD VR è spesso percepito come la forma più rappresentativa di realtà virtuale.

Le ragioni che spiegano tale tendenza sono molteplici: negli anni '90 e 2000 i costi elevati e i limiti tecnologici rendevano gli HMD poco accessibili e meno avanzati rispetto agli standard attuali; inoltre, la loro implementazione in contesti educativi si è spesso rivelata complessa, soprattutto in ambienti collaborativi di gruppo, richiedendo competenze tecniche avanzate. A ciò si aggiungono ulteriori fattori limitanti, come le difficoltà di accesso a dispositivi avanzati in contesti multiutente, i rischi per la salute legati a fenomeni di motion sickness e cybersickness, la mancanza di linee guida pedagogiche su come integrare efficacemente l'HMD VR nei processi formativi e la scarsità di evidenze empiriche che ne dimostrino l'efficacia. Questa situazione si riflette anche nella confusione terminologica riscontrata in letteratura, dove i concetti di "VR" e "ambienti virtuali" (VE) vengono talvolta utilizzati come sinonimi e talvolta distinti, generando ulteriore frammentazione.

Nonostante tali criticità, alcuni studi mettono in luce specifici vantaggi dell'HMD VR rispetto ai sistemi non-HMD, tra cui un maggiore senso di embodiment, migliori risultati nell'apprendimento di abilità psicomotorie e un incremento di coinvolgimento e motivazione. Altri lavori, tuttavia, sottolineano come possibili svantaggi il sovraccarico cognitivo e le distrazioni che possono compromettere i risultati formativi.

2.2.5 Conoscenza empirica consolidata riguardo alla VRCL

Riassumendo la prospettiva degli studi analizzati, il 70% mostrava un atteggiamento positivo verso l'applicazione della VRCL all'educazione. Sebbene solo circa il 25% presentasse esiti statisticamente significativi, ciò evidenzia un forte ottimismo nella comunità scientifica riguardo all'uso della VRCL in diversi ambiti educativi. Dalla letteratura revisionata emergono diversi aspetti positivi:

- **Coinvolgimento e motivazione:** la VRCL si dimostra efficace nel motivare gli studenti e stimolare l'apprendimento. La possibilità di personalizzare ambienti e contenuti rende l'esperienza più adatta alle caratteristiche individuali, incrementando la motivazione sia individuale che di gruppo. Inoltre, le qualità immersive incoraggiano la partecipazione attiva, la comunicazione e la collaborazione.
- **Apprendimento a distanza e collaborazione remota:** gli ambienti virtuali permettono a studenti e insegnanti di collaborare anche a distanza, con un forte senso di immediatezza che riduce le barriere fisiche e psicologiche nella comunicazione.

L'immersione e la presenza offerte dalla VRCL consentono simulazioni realistiche di scenari formativi, promuovendo interazione sociale e apprendimento attivo. In alcuni casi, come nell'educazione di studenti con disabilità, la VRCL è risultata persino più accessibile delle controparti reali. Per essere efficace, il design deve garantire presenza, consapevolezza e senso di appartenenza al gruppo.

- **Apprendimento interdisciplinare:** la flessibilità della VRCL consente di adattare gli ambienti virtuali a molte discipline diverse, favorendo la collaborazione tra studenti con background eterogenei e sottolineando l'importanza della collaborazione interdisciplinare.
- **Sviluppo delle competenze sociali:** la VRCL facilita la socializzazione e la costruzione di identità, consentendo agli studenti di sperimentare ruoli e norme sociali attraverso gli avatar. Le competenze sociali di base, come il riconoscimento e la gestione degli stati emotivi, possono essere allenate in ambienti virtuali, migliorando comunicazione ed intelligenza emotiva. Tuttavia, l'esperienza pregressa degli studenti con i VE influisce significativamente sull'efficacia della collaborazione.
- **Compatibilità con i paradigmi educativi:** la VRCL si integra bene con approcci come il costruttivismo, il socio-costruttivismo e il PBL. Gli ambienti virtuali permettono un apprendimento esperienziale privo di rischi reali, favorendo strategie come la simulazione di ruoli e l'improvvisazione. Nel PBL, ogni studente può contribuire con strumenti virtuali per rappresentare idee e soluzioni, stimolando la co-costruzione di modelli mentali condivisi.

Lo studio quindi propone alcune direzioni per la ricerca futura. In primo luogo, evidenzia la necessità di approfondire le differenze tra VRCL HMD e non-HMD, analizzandone non solo le potenzialità ma anche le sfide. In secondo luogo, sottolinea come le abilità affettive e psicomotorie siano ancora poco esplorate, pur avendo la VR un forte potenziale in tali domini. Analogamente, anche la dimensione formativa rivolta ai lavoratori risulta poco approfondita, con una presenza marginale negli studi analizzati. Infine, raccomanda un maggior ricorso a disegni sperimentali rigorosi e a metodi di valutazione innovativi, inclusi i dati fisiologici, che restano un ambito ancora poco indagato.

2.3 La realtà virtuale per l'addestramento procedurale: Confronto tra l'apprendimento tradizionale e l'apprendimento attraverso l'insegnamento

Lo studio presentato in De Lorenzis et al. [25] ha indagato l'efficacia della realtà virtuale per l'addestramento procedurale in ambito industriale, confrontando approcci tradizionali e strategie di learning by teaching (LBT). È stato quindi sviluppato un VR-based Training System (VRTS) con lo scopo di insegnare una procedura di misurazione su quadri elettrici. Il sistema presenta due modalità, una guidata e una di valutazione: nella prima vengono illustrati tutti i passaggi necessari al completamento della procedura, tramite evidenziazione

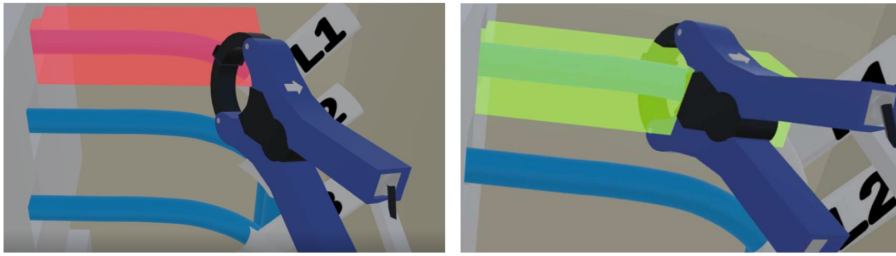


Figura 2.1: Sistema di evidenziazione adottato nella modalità guidata [25].

degli oggetti e ologrammi, mentre nella seconda qualsiasi tipo di aiuto è assente e viene generato un report finale per valutare la prestazione dell'utente.

Lo studio si è quindi articolato in due esperimenti. Il primo volto a valutare l'efficacia del VRTS rispetto a una combinazione di lezioni teoriche e sessioni pratiche in laboratorio. A tale scopo, tre diversi gruppi di partecipanti hanno seguito diverse modalità di apprendimento: usando solo il VRTS, seguendo il metodo tradizionale e infine il terzo usando una modalità mista tra i due. Si è riscontrato che, anche se la motivazione a imparare era più alta nella modalità mista, i partecipanti che hanno utilizzato unicamente il VRTS hanno dimostrato una maggiore efficienza nel svolgere la procedura. Il secondo esperimento mirava a confrontare l'apprendimento tradizionale (TL) con l'approccio LBT. Il gruppo che ha seguito l'apprendimento tradizionale ha sperimentato la modalità guidata del VRTS e successivamente la modalità di valutazione. Il secondo gruppo, dopo aver imparato la procedura con la modalità guidata, ha affiancato altri partecipanti nella modalità di valutazione. L'esperimento ha dimostrato che l'integrazione del VRTS nell'approccio LBT non è sfavorevole, anzi questo metodo è risultato superiore al TL nella curva di apprendimento e nella ritenzione delle informazioni. Lo studio cita anche il poter aggiungere eventualmente in futuro una componente multiplayer per studiare l'impatto che avrebbe sulle interazioni e sull'apprendimento.

2.4 La creazione di un ambiente virtuale multi-utente per l'addestramento degli operatori in VR

Lo studio condotto da Söderlund et al. [26] risponde alla necessità di soluzioni formative più efficaci nel settore manifatturiero, in particolare nell'industria automobilistica. Un mercato sempre più competitivo spinge i produttori a ottimizzare e migliorare l'addestramento dei propri operatori, a cui è richiesto di lavorare su una crescente varietà di prodotti in tempi brevi. Gli operatori di solito vengono sottoposti a un corso base di addestramento utilizzando strumenti fittizi. Una volta inseriti nell'ambiente produttivo è molto probabile che proseguano la loro formazione affidandosi a supervisori, colleghi, istruzioni testuali o immagini. Questo approccio tradizionale comporta diversi rischi: operatori inesperti potrebbero danneggiare la strumentazione, rallentare la produzione e compromettere la qualità del prodotto finale. Inoltre, l'utilizzo di attrezzature fisiche effettuato da utenti inesperti può esporre a maggiori rischi di sicurezza.

L'articolo presenta lo sviluppo di un VRTS multiutente alla Volvo Cars. Il sistema, realizzato in Unity utilizzando dati CAD reali, consente a più operatori e supervisori di condividere lo stesso spazio virtuale per svolgere procedure di assemblaggio. Tre scenari industriali differenti sono stati simulati per testare l'ambiente: il montaggio di una portiera, l'installazione di sensori di livellamento e il sollevamento collaborativo di un componente saldato. Per implementare la funzionalità multiutente, è stato utilizzato Photon Networking, che ha permesso di sincronizzare in tempo reale posizione e rotazione degli oggetti, così come l'avanzamento delle sequenze di training tra tutti i client connessi allo stesso server. Questo ha consentito di far collaborare fino a 16 utenti, ciascuno rappresentato da un avatar con nome e ruolo (operatore o supervisore), e di comunicare tramite chat vocale a prossimità.

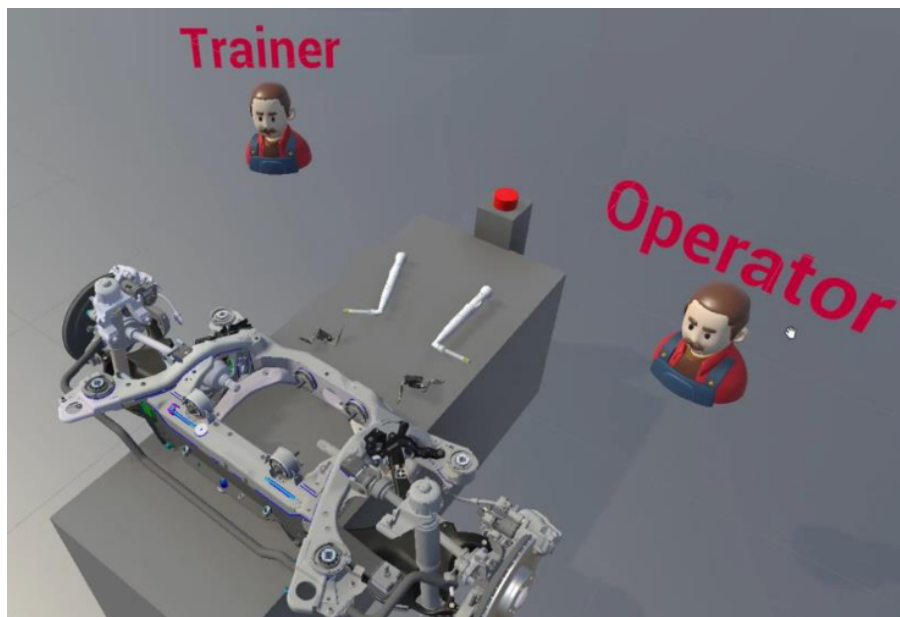


Figura 2.2: Visualizzazione del secondo scenario sviluppato [26].

Attraverso interviste con due gruppi di professionisti coinvolti nella formazione in Volvo, si sono raccolti feedback generali sul VRTS e su due casi d'uso della VR multiutente: un operatore guidato da un supervisore e, il secondo, multipli operatori che si addestrano contemporaneamente. Entrambi i gruppi hanno concordato che un VRTS multiutente ha il potenziale di esaltare il ruolo e le abilità di un supervisore. La presenza di esso nell'ambiente virtuale permette alle istruzioni impartite di essere adatte allo scenario e al singolo operatore, e stimola un dialogo tra i due che può migliorare l'apprendimento. È emerso anche come i VRTS multiutente possano potenzialmente svolgere la funzione di tour virtuali di una fabbrica, e mostrare quali azioni compiere o meno in determinate situazioni. Tuttavia, sono emerse anche alcune limitazioni, come la difficoltà nel valutare le abilità motorie dell'operatore nell'utilizzare certi strumenti e la mancanza di feedback fisico.

Nel secondo caso d'uso analizzato, la discussione nei due gruppi di focus si è concentrata

sulla possibilità che più operatori, formati insieme in un ambiente VR condiviso, possano migliorare l'apprendimento. Entrambi i gruppi hanno osservato che, all'interno dello stabilimento, le postazioni che richiedono la collaborazione diretta di più operatori sono poche, limitando così i casi in cui ha senso un addestramento multiutente sulla stessa stazione.

Tuttavia, è stata evidenziata l'opportunità di usare la VR multiutente per esercitarsi su forme di collaborazione più complesse e indirette, come la comprensione delle dipendenze tra stazioni o funzioni adiacenti, o per insegnare principi di standardizzazione, ad esempio i principi 5S. In questo scenario, più operatori condividono una postazione virtuale e sperimentano direttamente le conseguenze di una scarsa manutenzione o di strumenti non riposti correttamente.

Entrambi i gruppi hanno sottolineato che la possibilità di formarsi insieme ad altri operatori nello stesso ambiente VR potrebbe favorire l'apprendimento reciproco e aumentare il coinvolgimento e il piacere dell'attività formativa, rendendo il training più attraente ed efficace. Infine hanno concluso che il VRTS multiutente non può sostituire completamente il sistema di addestramento tradizionale già presente ma ha il potenziale di migliorare, sostituire o complementare parti di esso.

2.5 Addestramento all'uso di macchinari industriali in un laboratorio virtuale multi-utente e multiplatforma

Lo studio presentato in Chuang e Smith [27] descrive lo sviluppo di un laboratorio virtuale multiutente e multiplatforma, progettato per la formazione pratica in ambito manifatturiero, con particolare riferimento alle operazioni di fresatura. L'obiettivo principale è dimostrare come un ambiente immersivo e collaborativo possa superare i limiti delle tradizionali aule fisiche e fornire agli studenti e lavoratori un contesto interattivo per l'apprendimento delle procedure tecniche. L'architettura è stata realizzata in Unity, facendo uso del framework Photon Unity Networking per la sincronizzazione degli utenti in tempo reale e per il supporto delle interazioni tra dispositivi differenti. Un aspetto di rilievo è infatti la possibilità di accedere all'ambiente virtuale da diverse piattaforme hardware: HTC Vive Pro, Microsoft HoloLens 2 [28], PC tradizionali e smartphone Android, garantendo così la fruizione dell'esperienza anche in contesti didattici caratterizzati da disponibilità tecnologiche eterogenee.

L'ambiente virtuale sviluppato riproduce fedelmente una fabbrica con macchine utensili, e consente all'istruttore di eseguire dimostrazioni pratiche di lavorazione direttamente nello spazio condiviso. Gli studenti possono osservare da prospettive differenti, interagire con gli oggetti virtuali e replicare in prima persona i procedimenti appresi. Una caratteristica distintiva è l'adattamento dell'interazione in base al dispositivo utilizzato: gesture recognition per HoloLens 2, controller per Vive Pro, mouse e tastiera per PC e touchscreen per smartphone. Questa eterogeneità ha richiesto uno studio approfondito sull'ergonomia e sull'usabilità delle interfacce, con l'obiettivo di garantire un livello di partecipazione coerente tra le diverse piattaforme.

I risultati dei test eseguiti hanno evidenziato che le piattaforme Vive Pro e PC hanno ottenuto i punteggi più alti sia in termini di coinvolgimento che di usabilità, grazie

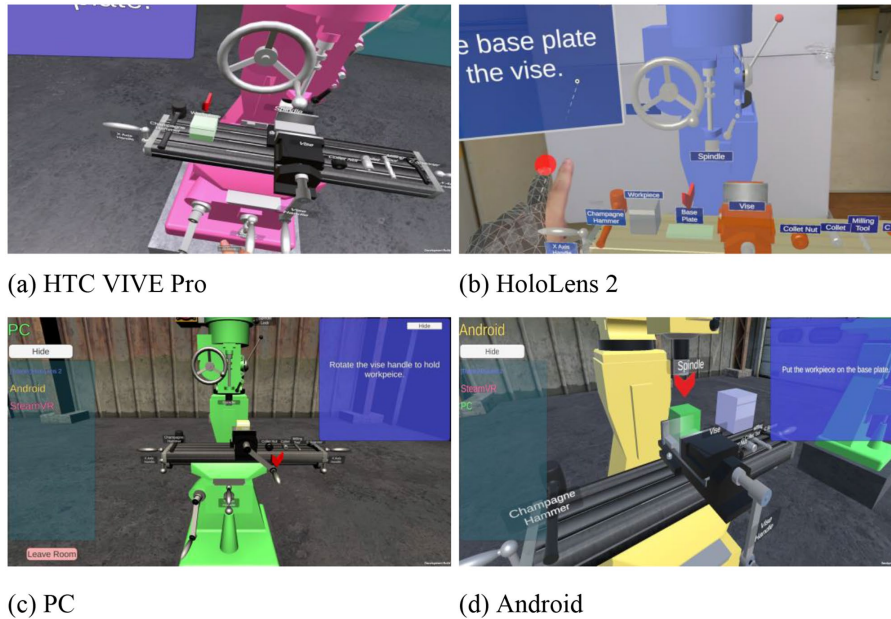


Figura 2.3: Operatori che si esercitano sulle diverse piattaforme [27].

alla maggiore stabilità delle interazioni e alla chiarezza visiva. Invece, l'esperienza con HoloLens 2 ha mostrato limiti legati al riconoscimento gestuale non sempre preciso, mentre sugli smartphone Android l'esperienza risultava penalizzata dalle dimensioni ridotte dello schermo e da un'interazione meno intuitiva. Nonostante queste criticità, l'esperimento ha confermato il valore aggiunto di un laboratorio virtuale multiplatforma, capace di consentire a più utenti di collaborare nello stesso ambiente simulato, replicando la logica delle esercitazioni di laboratorio fisiche.

Gli autori concludono sottolineando come questo tipo di soluzioni possa contribuire in maniera significativa alla formazione a distanza in ambito industriale, specialmente in contesti in cui non è sempre possibile garantire l'accesso a macchinari reali o a spazi di laboratorio. Inoltre, viene messa in evidenza la necessità di future ricerche volte a ottimizzare le interazioni cross-device e a valutare più approfonditamente l'efficacia dell'apprendimento collaborativo in ambienti VR multiutente, soprattutto per quanto riguarda l'equilibrio tra immersione, usabilità e fedeltà delle procedure industriali simulate.

2.6 Considerazioni conclusive

Dalla letteratura emerge come la VR sia ormai uno strumento maturo per la formazione procedurale e tecnica in contesti industriali ed educativi. L'impiego di essa garantisce una maggiore sicurezza, la possibilità di replicare scenari pericolosi, costosi o non disponibili nella realtà, e una migliore acquisizione delle competenze.

Approcci formativi innovativi, come il LBT, hanno evidenziato un potenziale ulteriore nel rafforzare il coinvolgimento cognitivo, aspetto che si potrebbe approfondire legandolo alla formazione collaborativa.

Viene evidenziato infatti anche il valore del VRCL, la possibilità per operatori o supervisori di condividere lo stesso spazio virtuale ha migliorato il processo formativo e facilitato il monitoraggio e la valutazione delle prestazioni. Analogamente, i laboratori virtuali multiplatforma hanno dimostrato la fattibilità di ambienti inclusivi e accessibili, capaci di integrare utenti con diversi dispositivi.

Infine, le revisioni sistematiche hanno sottolineato come la VR favorisca l'interazione sociale, la collaborazione e la costruzione condivisa della conoscenza.

Tuttavia, permangono alcune criticità. Molti studi presentano campioni ridotti o mancano di valutazioni oggettive e a lungo termine dell'efficacia formativa, rendendo difficile confrontare i risultati e generalizzarne l'impatto. L'implementazione di sistemi multiutente rimane complessa sul piano tecnico (sincronizzazione, gestione della rete, scalabilità), mentre sul piano didattico non esistono ancora modelli consolidati per sfruttare appieno le dinamiche collaborative.

In questo contesto, il presente lavoro di tesi si propone di contribuire allo scenario delineato attraverso lo sviluppo di un VRTS collaborativo. L'obiettivo è valorizzare i punti di forza emersi dalla letteratura, integrando i benefici della VR immersiva con le potenzialità della VRCL, e offrendo al tempo stesso la possibilità di arricchire il processo formativo tramite l'approccio LBT. La soluzione proposta adotta una struttura flessibile e adattabile a differenti contesti industriali, ponendosi anche come possibile riferimento metodologico per la progettazione di futuri VRTS multiutente.

Capitolo 3

Tecnologie utilizzate

In questo capitolo vengono introdotti i principali strumenti hardware e software utilizzati per la realizzazione del VRTS, con una descrizione del loro ruolo nella gestione delle interazioni immersive e della sincronizzazione tra più utenti.

3.1 Unity

Unity [29] è un game engine multiplatforma sviluppato da Unity Technologies per la creazione di applicazioni interattive in tempo reale. Come osservato negli studi analizzati precedentemente, risulta essere la soluzione più utilizzata nello sviluppo di VRTS.

Unity integra strumenti per la gestione di ambienti tridimensionali, la fisica, l'illuminazione e le animazioni, consentendo di realizzare esperienze immersive molto eterogenee tra loro. Grazie a un'architettura modulare, Unity permette l'integrazione di pacchetti dedicati che semplificano la gestione di interazioni complesse e la sincronizzazione di sessioni multiutente. Inoltre, consente di collegare un ambiente di sviluppo (Integrated Development Environment, IDE), tramite il quale è possibile progettare script in linguaggio C# che definiscono la logica e i comportamenti del progetto.

Unity ha promosso la propria presenza anche in settori non strettamente legati ai videogiochi, distribuendo ad esempio Unity Industry, che fornisce licenze, supporto e strumenti specifici per applicazioni professionali in settori come design, formazione e simulazione tecnica. La versione utilizzata nello sviluppo di questo sistema è la 6000.0.30f1.

3.1.1 XR Interaction Toolkit

XR Interaction Toolkit [30] è un pacchetto che semplifica lo sviluppo di interazioni in realtà estesa, offrendo un approccio standardizzato a funzionalità come la manipolazione degli oggetti, il tocco, la locomozione e l'interazione con interfacce utente tridimensionali.

Il toolkit fornisce un framework che collega gli eventi di input di Unity alle meccaniche di interazione 3D e con la User Interface (UI). Al centro del sistema vi sono due categorie di componenti: gli *Interactor*, che definiscono i metodi di interazione, e gli *Interactable*, che

rappresentano gli elementi con cui l'utente può interagire. Le loro relazioni vengono coordinate dall'*Interaction Manager*, mentre moduli aggiuntivi si occupano della locomozione e della rappresentazione visiva delle interazioni.

Grazie al supporto multiplatforma di Unity, XR Interaction Toolkit consente agli sviluppatori di concentrarsi sulla progettazione delle esperienze XR senza doversi confrontare con la complessità dei diversi modelli di interazione specifici di ciascun dispositivo. È inoltre incluso l'*XR Device Simulator*, uno strumento che permette di emulare un visore e i controller VR direttamente dall'editor di Unity, evitando la necessità di collegare un dispositivo fisico. Per lo sviluppo descritto in questa tesi è stata utilizzata la versione 3.1.2.

3.1.2 OpenXR

OpenXR [31] è un framework sviluppato da Khronos Group progettato per fornire un'unica interfaccia di programmazione per applicazioni di realtà estesa. La sua adozione consente di ridurre la frammentazione dovuta ai diversi SDK proprietari, garantendo compatibilità multiplatforma e supporto a visori e dispositivi eterogenei senza necessità di sviluppare soluzioni specifiche per ciascun hardware. Integrato in Unity, OpenXR permette di accedere in modo uniforme alle funzionalità dei dispositivi XR, favorendo portabilità e scalabilità del progetto. Grazie a questo approccio, gli sviluppatori possono concentrarsi sulla progettazione delle interazioni e dei contenuti, demandando allo standard la gestione delle differenze tecniche tra i vari sistemi. La versione utilizzata è stata la 1.13.2.

3.1.3 Netcode for Gameobjects

Netcode for GameObjects (NGO) [32] è uno dei framework sviluppati da Unity per la creazione di applicazioni multiutente. Si tratta di una soluzione particolarmente adatta ad applicazioni cooperative con un numero limitato di partecipanti, in cui non sono richieste le massime prestazioni di rete tipiche di alcuni giochi competitivi basati su azioni rapide. Per questi scenari Unity ha invece sviluppato Netcode for Entities. NGO supporta due principali topologie di rete: *client-server* e *distributed authority*.

La topologia client-server rappresenta il modello più diffuso, in cui le responsabilità vengono suddivise tra i dispositivi client e un server centrale, con l'obiettivo di ottimizzare le prestazioni e garantire una gestione coerente dello stato applicativo. I client corrispondono alle istanze in esecuzione sui dispositivi degli utenti: si occupano della gestione degli input locali, del rendering e di una parte della simulazione, inviando al server aggiornamenti come i movimenti del personaggio e ricevendo in cambio lo stato aggiornato del sistema. Il server, invece, mantiene la rappresentazione globale e accurata dell'applicazione, elabora gli input ricevuti e applica le regole, risolvendo conflitti e convalidando le azioni. Questo approccio assicura un'esperienza uniforme ed equa e contribuisce a ridurre i comportamenti malevoli, poiché lo stato complessivo è gestito centralmente.

All'interno del modello client-server si distinguono due tipologie di server. I server dedicati sono entità indipendenti, che non partecipano come utenti, ma si limitano a processare i dati e gestire le interazioni. Offrono le migliori prestazioni e rappresentano la scelta ottimale per applicazioni competitive e sensibili alla latenza. Tuttavia, l'architettura

introduce un inevitabile ritardo di comunicazione, in quanto tutte le azioni devono essere elaborate dal server prima di essere propagate ai client. In alternativa, è possibile adottare un client-hosted server (*host*), in cui uno dei client assume anche il ruolo di server. Questa configurazione riduce i costi, non richiedendo infrastrutture esterne. Garantisce anche al giocatore host un vantaggio in termini di latenza, poiché non deve instradare i pacchetti sulla rete. D’altro canto, comporta anche un maggior carico sulla propria macchina, che deve gestire sia l’elaborazione del server sia il rendering locale.

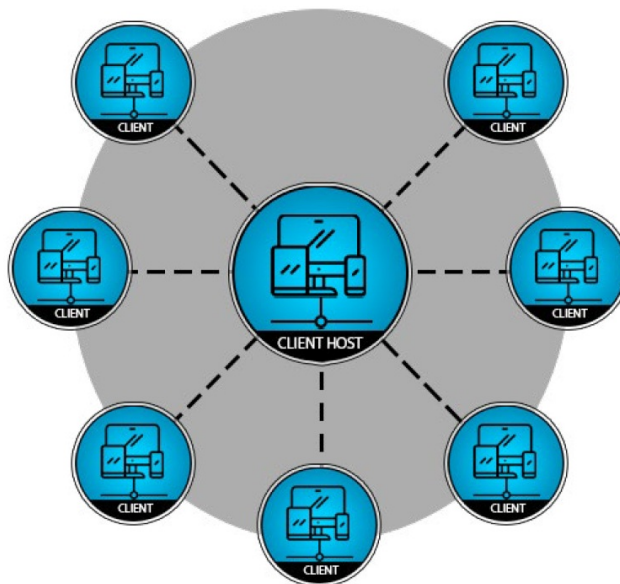


Figura 3.1: Topologia Client-Host [33].

La topologia distributed authority adotta invece un approccio decentralizzato: la gestione dello stato è suddivisa tra i client partecipanti, ciascuno dei quali è responsabile di una parte degli oggetti e può gestirne autonomamente la creazione e la sincronizzazione. Un servizio centrale, progettato per essere leggero, ha il compito di monitorare i cambiamenti negli stati degli oggetti e di gestire la distribuzione del traffico di rete, ma non esegue la simulazione vera e propria dell’ambiente di gioco. Questo modello riduce i costi e la latenza sugli input, poiché elimina la necessità da parte di un server centrale di elaborare tutte le azioni di gioco e consente ai client di gestire localmente azioni come movimenti e interazioni, offrendo così un feedback immediato e una maggiore sensazione di reattività. Tuttavia, l’assenza di un’entità autorevole rende difficile mantenere consistente lo stato dell’applicazione tra i vari utenti. Inoltre, il sistema è reso più vulnerabile a comportamenti scorretti e quindi ne limita l’applicabilità in contesti competitivi o che richiedono simulazioni di alta precisione. È quindi più indicato per applicazioni in cui la coerenza assoluta non è un requisito stringente. Il prototipo di VRTS sviluppato utilizza la topologia client-hosted server, visto che è la più veloce da implementare e non richiede nessuna architettura esterna.

La comunicazione tra client e server in NGO si basa su protocolli standard quali

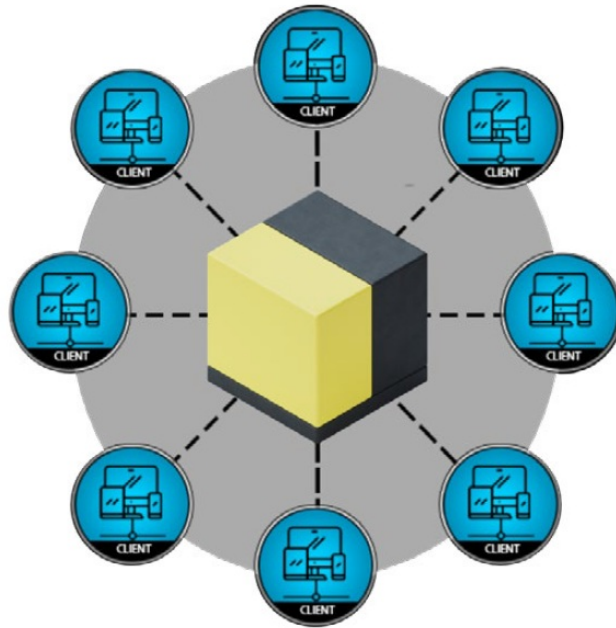


Figura 3.2: Topologia distributed authority [33].

User Datagram Protocol (UDP) e *Transmission Control Protocol* (TCP). UDP è preferito nelle applicazioni in tempo reale perché non richiede conferme di ricezione e minimizza la latenza, offrendo maggiore fluidità nelle interazioni. TCP, invece, garantisce affidabilità e ordine di consegna dei dati, a costo di una maggiore latenza e dimensione dei pacchetti.

NGO mette inoltre a disposizione una serie di componenti fondamentali per la gestione della rete. Il **Network Object** rappresenta ogni oggetto sincronizzato e gestisce operazioni quali *spawn*, *despawn* e *ownership*, ossia l'assegnazione della gestione locale a un determinato client. Ogni **Network Object** deve essere associato a un **Network Behaviour**, che definisce il comportamento dell'oggetto in rete ed estende le funzionalità di Unity con logiche specifiche lato server o client. Le *Remote Procedure Calls* (RPCs) consentono di inviare messaggi e invocare metodi su istanze remote, mentre le **Network Variable** permettono la sincronizzazione automatica di valori tra client. A coordinare l'intero sistema è il **Network Manager**, che gestisce lo stato generale della rete, incluse connessioni, disconnessioni e caricamento delle scene. La versione di NGO usata è stata la 1.10.0.

3.1.4 Unity Services

Gli Unity Multiplayer Services affiancano Netcode for GameObjects offrendo un'infrastruttura cloud che semplifica e potenzia la gestione delle connessioni multiutente.

- **Authentication Service:** gestisce l'identità degli utenti, consentendo l'accesso anonimo oppure tramite account personali, a seconda del livello di sicurezza richiesto dall'applicazione.

- **Lobby Service:** permette la creazione e gestione di stanze virtuali in cui i giocatori possono incontrarsi prima dell'avvio della sessione. Consente di invitare altri utenti e definire parametri condivisi come il numero massimo di partecipanti o le impostazioni di gioco.
- **Relay Service:** Unity Relay semplifica la connessione tra più giocatori tramite un sistema basato su join code. Quando un host avvia una sessione di gioco, viene generato un codice univoco da condividere con gli altri utenti, i quali lo utilizzano per collegarsi alla partita. Questo meccanismo rende il processo di connessione più immediato e, al tempo stesso, tutela la privacy, mantenendo nascosti dati sensibili come gli indirizzi IP. Le connessioni dirette tra client richiedono spesso configurazioni di rete complesse ed espongono gli indirizzi IP degli utenti, sollevando problematiche legate alla sicurezza e alla privacy. Il servizio Relay risolve queste criticità fungendo da server intermediario: gestisce l'instradamento delle comunicazioni e fornisce una connettività sicura e semplificata tra i client. In questo modo elimina la necessità di disporre di server dedicati.

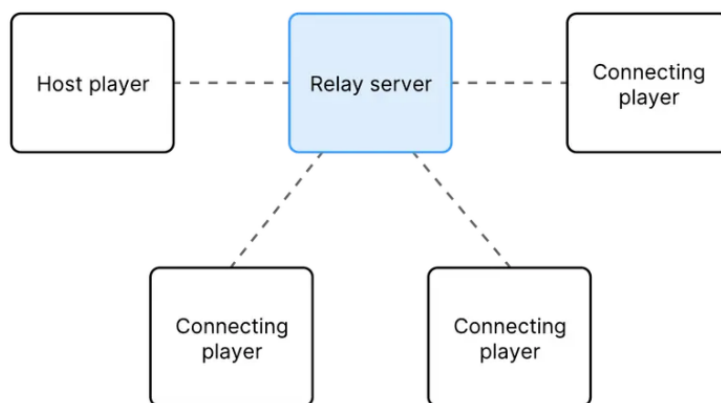


Figura 3.3: Architettura di rete del servizio *Relay* [34].

- **Vivox:** fornisce un sistema di comunicazione vocale e testuale in tempo reale, integrato nell'applicazione, che migliora la collaborazione e l'esperienza multiutente.

La versione di Unity Multiplayer Services utilizzata è la 1.0.2

3.1.5 Parrel Sync

ParrelSync [35] è uno strumento open-source che facilita lo sviluppo e il debug di applicazioni multiplayer in Unity. Il pacchetto permette di lanciare più istanze dell'editor contemporaneamente sullo stesso computer, consentendo di simulare la presenza di diversi client senza dover effettuare build separate. Questa funzionalità risulta particolarmente utile durante le fasi iterative di sviluppo, poiché riduce i tempi di test e consente di verificare in modo immediato la sincronizzazione degli oggetti e delle logiche di rete implementate con Netcode for GameObjects. La versione che è stata utilizzata è la 1.5.2.

3.2 Visual Studio

Visual Studio [36] è l'IDE adottato per la scrittura e la gestione degli script in linguaggio C# all'interno del progetto. La sua integrazione nativa con Unity consente di accedere a strumenti di auto completamento, debugging e refactoring del codice, semplificando il processo di sviluppo e riducendo la probabilità di errori. Visual Studio mette inoltre a disposizione funzioni avanzate per la gestione di progetti complessi, come il supporto al versionamento con Git e la possibilità di installare estensioni specifiche per Unity, che ne potenziano ulteriormente l'efficacia come strumento di lavoro. La versione utilizzata è Visual Studio Community 2022.

3.3 Blender

Blender [13] è un software open-source per la modellazione, l'animazione e il rendering tridimensionale, ampiamente utilizzato in ambito professionale e accademico. Grazie alla sua versatilità, consente di creare modelli 3D personalizzati, animazioni, materiali e texture, supportando formati di esportazione compatibili con Unity e altri game engines. Nel contesto di questa tesi, Blender è stato impiegato per la modifica di alcuni modelli 3D. La versione utilizzata durante lo sviluppo è la 4.2.

3.4 Meta Quest 2

Meta Quest 2 [2] è il primo dei due visori per realtà virtuale utilizzati per il testing e la validazione dell'applicativo sviluppato. È un visore di tipo stand-alone, in grado di eseguire applicazioni senza la necessità di un PC esterno. Il dispositivo integra un display LCD con una risoluzione complessiva di 1832×1920 pixel per occhio, un campo visivo di circa $90-100^\circ$ e una frequenza di aggiornamento variabile fino a 120 Hz, che garantiscono un'esperienza visiva fluida e nitida.

È dotato di due controller con tracciamento a sei gradi di libertà (6DoF) e utilizza un sistema di inside-out tracking basato su quattro telecamere grandangolari integrate nel visore, che consentono di ricostruire i movimenti dell'utente nello spazio senza ricorrere a sensori esterni. Il dispositivo impiega il sistema operativo Meta Quest OS (basato su Android) e monta un processore Qualcomm Snapdragon XR2, con 6 GB di memoria RAM, caratteristiche che gli permettono di gestire esperienze VR complesse anche in modalità autonoma.

Grazie alla possibilità di collegamento tramite Meta Quest Link o Air Link, il visore può inoltre funzionare in modalità PC-VR, sfruttando la potenza di calcolo di un computer esterno per applicazioni più esigenti e garantendo una qualità grafica superiore rispetto all'esecuzione stand-alone.

3.5 Meta Quest Pro

Meta Quest Pro [37] è un visore di realtà mista di fascia alta, sviluppato da Meta per soddisfare esigenze professionali e progettato per applicazioni che richiedono una maggiore



Figura 3.4: Meta Quest 2.

precisione nel tracciamento e una migliore qualità visiva. È dotato di due display LCD con tecnologia Quantum Dot e risoluzione di 1800×1920 pixel per occhio, che garantiscono colori più accurati e un campo visivo di circa 106° orizzontali e 96° verticali. Il refresh rate massimo è di 90 Hz, sufficiente a garantire fluidità e comfort visivo nelle sessioni prolungate.

Il visore impiega un sistema di tracciamento inside-out basato su cinque telecamere esterne per il rilevamento ambientale e altre cinque telecamere interne dedicate al tracciamento oculare e facciale, consentendo di riprodurre in modo realistico le espressioni e i movimenti degli occhi dell'utente, migliorando così la comunicazione e la presenza sociale negli ambienti collaborativi. È alimentato da un processore Qualcomm Snapdragon XR2+



Figura 3.5: Meta Quest Pro.

e dispone di 12 GB di memoria RAM, il che rappresenta un significativo incremento rispetto al Quest 2 e consente di gestire simultaneamente rendering, tracciamento e calcolo del passthrough a colori. Quest'ultimo, realizzato tramite le telecamere frontali ad alta

risoluzione, permette la sovrapposizione di contenuti virtuali all'ambiente reale, abilitando applicazioni di realtà mista.

I controller Meta Quest Touch Pro integrano sensori di tracciamento autonomo e un sistema aptico migliorato, offrendo una maggiore precisione e realismo durante l'interazione. Il dispositivo può operare sia in modalità stand-alone sia come PC-VR tramite Meta Quest Link o Air Link, offrendo la massima flessibilità d'uso a seconda delle esigenze dell'applicazione.

Capitolo 4

Progettazione e sviluppo dell'applicazione

In questo capitolo viene presentato lo sviluppo del VRTS in questione, illustrando i suoi principali componenti. Nella prima parte si analizza l'architettura dell'applicazione e l'impiego del *VR Multiplayer Template* di Unity come base di partenza per lo sviluppo, per poi approfondire la struttura e il ruolo degli script originali del sistema, con particolare attenzione alla realizzazione delle procedure e alla sincronizzazione dei dati tra più utenti. Infine viene trattata la realizzazione dell'ambiente virtuale, approfondendo l'illuminazione e gli elementi grafici utilizzati.

4.1 Architettura dell'applicazione

Il diagramma mostrato nella Figura 4.1 illustra l'architettura software dell'applicazione organizzata in livelli, evidenziando la suddivisione logica tra le varie parti del sistema.

Alla base si trovano i servizi che permettono la connessione e il supporto alle funzionalità online, su cui si appoggia il livello di rete dedicato alla gestione della comunicazione tra più utenti. Sopra di essi è collocato il livello applicativo, che costituisce il nucleo funzionale dell'esperienza interattiva e integra i componenti principali dell'applicazione, suddivisi principalmente tra gli elementi forniti dall'XR Interaction Toolkit e quelli del sistema di gestione delle procedure. Il livello successivo comprende i moduli fondamentali del motore Unity, che forniscono le infrastrutture comuni necessarie al funzionamento del sistema. A un livello superiore è presente l'interfaccia verso il sistema XR, che consente all'applicazione di comunicare con i diversi ambienti di esecuzione compatibili. Infine, al vertice, si colloca il livello di accesso, che rappresenta i dispositivi attraverso i quali l'applicazione viene effettivamente fruita.

Questa struttura a strati, disposta verticalmente, sottolinea la gerarchia delle dipendenze tra i vari moduli e garantisce una separazione chiara tra le responsabilità, migliorando la leggibilità, la portabilità e la manutenibilità complessiva del sistema.

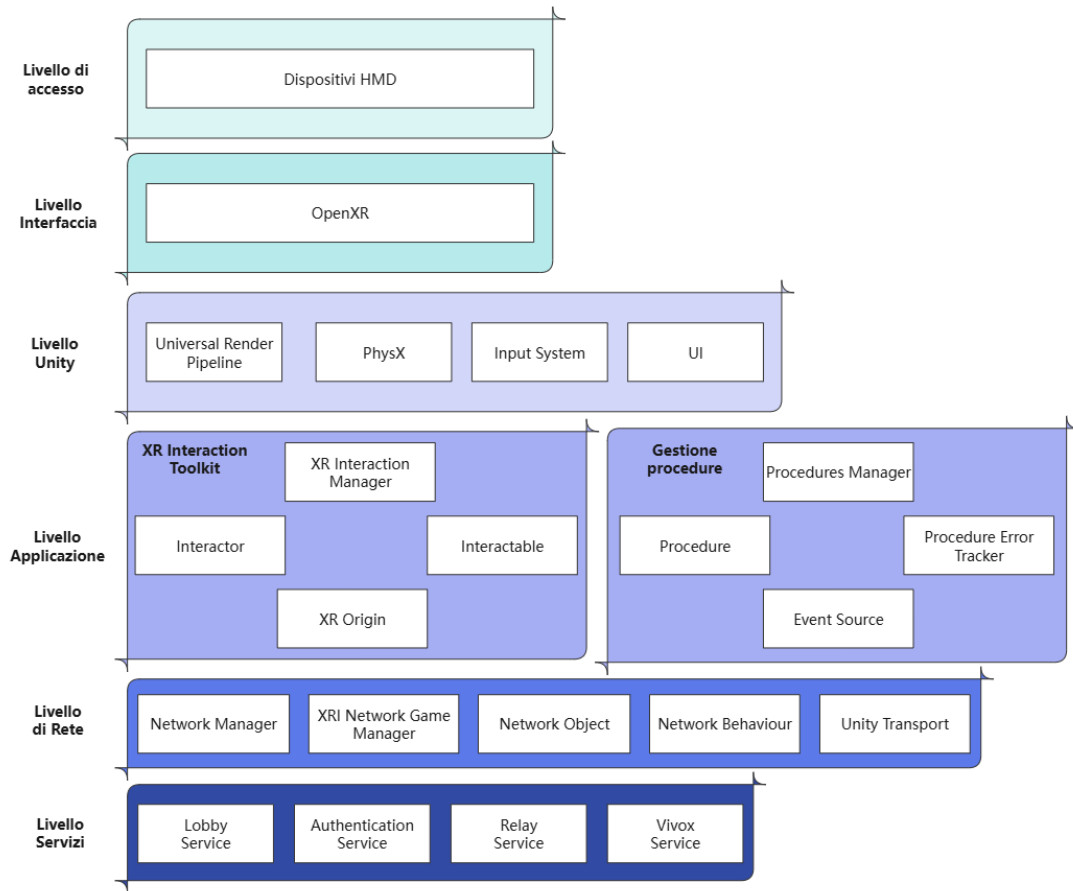


Figura 4.1: Architettura software dell'applicativo.

4.2 Componenti principali del VR Multiplayer Template

Lo sviluppo del sistema ha preso avvio dal VR Multiplayer Template, rilasciato da Unity nel luglio 2024. Si tratta di una risorsa pensata per semplificare la realizzazione di applicazioni multiutente in realtà virtuale, offrendo una configurazione iniziale robusta e adatta a supportare un'ampia gamma di scenari. Il template include un'architettura di rete pronta all'uso basata sulle tecnologie descritte nel capitolo precedente e, una volta collegato ai servizi Unity, consente di stabilire immediatamente una connessione online con altri utenti. Questo approccio permette di concentrare lo sviluppo sulla logica applicativa, riducendo il tempo investito nelle implementazioni di base delle funzionalità di rete e delle interazioni immersive. Nei paragrafi successivi verranno descritti i principali prefab¹ del

¹In Unity, un *prefab* è una risorsa serializzata che memorizza una gerarchia di oggetti con i relativi componenti e valori. Consente di creare istanze multiple e coerenti, sia durante l'esecuzione dell'applicazione sia in editor, mantenendo un'unica "fonte" modificabile: le modifiche al prefab si propagano alle istanze, mentre le variazioni effettuate sulla singola istanza possono essere mantenute come *override*.

template che sono stati utilizzati per la realizzazione dell'applicativo.

4.2.1 Struttura e funzionalità dell'avatar

Uno dei prefab principali utilizzati del template è l'**XR Origin Hands (XR Rig) MP Template Variant**. Fornisce l'avatar che rappresenta l'utente, diverse modalità di interazione e di locomozione, nonché il menù di accesso alle impostazioni e di chiusura dell'applicazione. Qui di seguito sono illustrate le principali componenti:

- **Tracked Pose Driver:** questo script è il componente che applica i dati di tracciamento forniti dal visore XR alla camera virtuale, aggiornandone posizione e rotazione in base ai movimenti della testa dell'utente. Complementare ad esso è lo script **XROrigin**, in cui vengono definite l'origine del sistema XR e la gestione dell'offset verticale della camera, in base alla modalità di tracciamento selezionata: *Floor* o *Device*. Nel primo caso, l'origine è fissata a quota zero (sul pavimento), per cui non è necessario alcun offset aggiuntivo. Nel secondo caso, invece, l'origine viene impostata all'altezza del dispositivo al momento dell'avvio e alla camera viene quindi applicato l'offset specificato, così da mantenere coerente la posizione rispetto allo spazio virtuale.
- **Character Controller:** fornisce la capsula che gestisce le collisioni e implementa una fisica semplificata del giocatore senza ricorrere a un *rigidbody*². La sua geometria è definita dai parametri centro, raggio e altezza, mentre il comportamento rispetto all'ambiente è controllato da impostazioni come la pendenza massima percorribile, l'altezza degli ostacoli superabili e la soglia minima di movimento.
- **Interactors:** Nella gerarchia, al di sotto del game object che rappresenta il controller, sono raccolti gli oggetti responsabili delle diverse modalità di interazione. Il **Poke Interactor** abilita l'interazione a tocco con superfici e pulsanti. Il **Near-Far Interactor** combina due modalità di interazione. Per gli oggetti vicini utilizza un volume sferico che consente di rilevare e afferrare elementi nello spazio circostante al controller. Per gli oggetti lontani impiega invece un raggio proiettato dal controller verso l'ambiente, permettendo di selezionare o manipolare oggetti a distanza. Nel caso di questo interactor è stato necessario disattivare l'interazione del raggio rispetto agli oggetti impiegati nelle procedure, impostandoli nel livello **Ignore Raycast** e aggiungendo quest'ultimo nei livelli interagibili con la sfera proiettata. Il **Teleport Interactor** è il componente che gestisce la locomozione basata sul teletrasporto. Attraverso un raggio proiettato dal controller consente di selezionare una destinazione valida nello spazio virtuale, mostrando un reticolo o un indicatore visivo. Per utilizzare questo interactor è stato necessario assegnare lo script **Teleportation Area** alla superficie in cui è possibile teletrasportarsi.

²In Unity, il *rigidbody* è il componente che abilita la simulazione fisica di un oggetto. Permette di applicare forze, gravità e collisioni in modo realistico, integrandosi con il motore fisico per gestire automaticamente i movimenti e le interazioni con altri corpi.



Figura 4.2: Teleport.

- **XR Interaction Manager:** rappresenta il componente centrale del XR Interaction Toolkit, in quanto si occupa di mettere in relazione gli interactor con gli interactable. Questo elemento gestisce l'intero ciclo di vita delle interazioni, dal semplice puntamento all'attivazione effettiva di un oggetto, risolvendo automaticamente conflitti e priorità qualora più interactor competano per lo stesso elemento. Inoltre, emette eventi standard a cui possono collegarsi gli script o i componenti dell'applicazione per personalizzare i comportamenti interattivi. L'Interaction Manager consente anche di applicare filtri globali, che permettono ad esempio di limitare o vincolare alcune interazioni in base allo stato dell'applicazione.
- **Suggerimenti dei comandi possibili:** Sempre in corrispondenza dell'oggetto che rappresenta il controller sono presenti alcuni prefab appartenenti all'UI, i quali forniscono indicazioni visive sulle azioni eseguibili tramite la pressione dei tasti. La loro gestione è affidata allo script `CalloutGazeController`, che ne regola la comparsa quando il controller entra nel campo visivo. Ogni elemento è inoltre associato, tramite una curva di collegamento, al tasto corrispondente all'azione rappresentata, facilitando così la comprensione dei comandi disponibili.
- **Locomozione:** Al di sotto dell'oggetto *Locomotion* sono raccolti i diversi componenti che abilitano i metodi di movimento dell'utente. La rotazione dell'utente è implementata tramite due script differenti: lo `SnapTurnProvider`, che effettua rotazioni discrete di un angolo prefissato, migliorando il comfort nelle sessioni prolungate, e il `ContinuousTurnProvider`, che abilita una rotazione continua a velocità controllata, offrendo una maggiore naturalezza nei movimenti. Allo stesso modo, gli spostamenti traslazionali sono gestiti dal `DynamicMoveProvider`, che permette di muoversi nello spazio virtuale con velocità configurabile, permettendo di cambiare certe opzioni come la possibilità di effettuare il movimento laterale o l'utilizzo della gravità. È possibile inoltre scegliere la sorgente di riferimento, testa o controller, per la direzione del movimento.

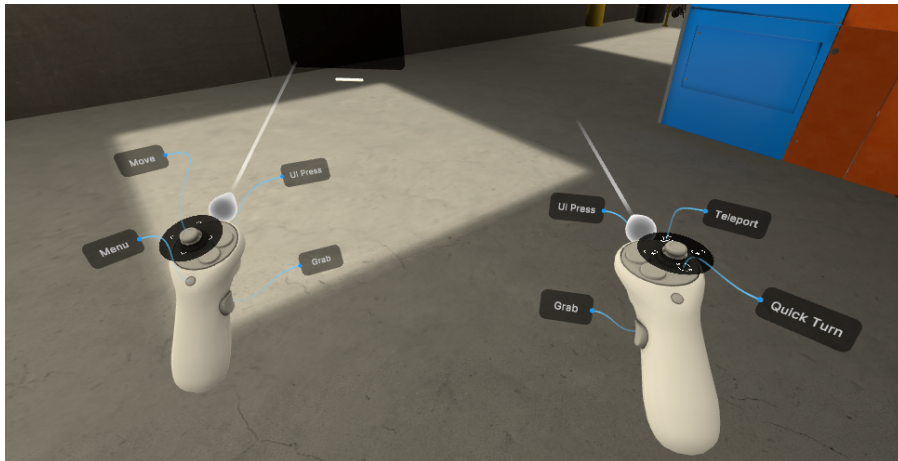


Figura 4.3: Indicazioni dei comandi disponibili.

- **Body-locked UI:** Collegato all'avatar è presente un canvas³, utilizzato per mostrare un menù con diverse funzionalità, accessibile tramite un bottone che appare ruotando il controller. Tramite questo menù è possibile modificare le opzioni legate alla lobby, come la privacy o il nome della stanza, le opzioni legate all'utente: nome, colore dell'avatar o mutare il microfono. È presente un'altra sezione dove poter cambiare stanza, ed infine le opzioni generiche dove si possono cambiare le impostazioni audio o di movimento, o chiudere l'applicazione. Inoltre lo stesso canvas viene anche utilizzato per far apparire notifiche di eventi come il lancio o la fine di una procedura.

³In Unity, un *Canvas* è un oggetto che permette di visualizzare e gestire gli elementi UI figli del canvas stesso.

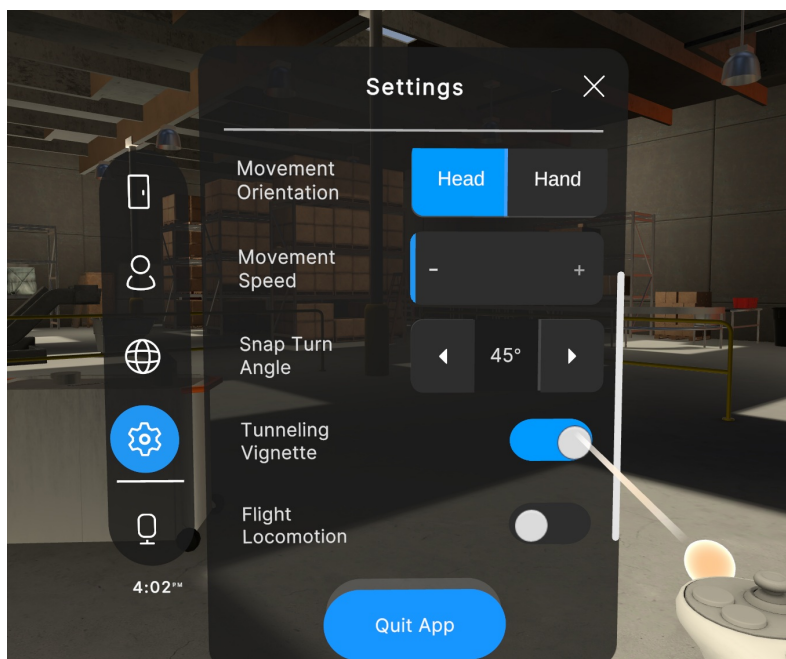


Figura 4.4: Menù collegato all'utente.



Figura 4.5: Visualizzazione dell'avatar di un utente dal punto di vista di un altro operatore.

4.2.2 Network Manager VR Multiplayer

Il prefab Network Manager VR Multiplayer rappresenta il nucleo della gestione delle funzionalità di rete all'interno del sistema. Esso integra i componenti di NGO necessari

per l'avvio, la configurazione e il mantenimento delle sessioni multiplayer, consentendo la comunicazione tra i diversi client e il server. Tramite il suo script principale, definito come *singleton*⁴, è possibile configurare diversi parametri fondamentali per il funzionamento della rete. Tra questi, uno dei più rilevanti è la definizione del prefab che rappresenta l'utente, istanziato automaticamente al momento della connessione. È inoltre possibile specificare la versione del protocollo di comunicazione, così da prevenire l'accesso ai client basati su versioni non compatibili dell'applicativo. Un ulteriore aspetto configurabile riguarda l'elenco dei prefab che devono essere creati e sincronizzati durante l'esecuzione: senza tale indicazione, infatti, i client non sarebbero in grado di sapere quali oggetti istanziare. Altri parametri includono la frequenza di aggiornamento della rete (*tick rate*), che definisce quante volte al secondo il server invia i dati ai client per mantenerne lo stato aggiornato, e il *Client Connection Buffer Timeout*, ovvero il tempo massimo concesso per completare la connessione prima della disconnessione automatica. È infine possibile regolare la dimensione massima, in byte, dei messaggi RPC e il *Spawn Timeout*, che indica per quanto tempo il client conserva le informazioni relative a un messaggio di rete in attesa che il prefab corrispondente venga istanziato.

Il secondo componente del Network Manager è rappresentato dallo script **Unity Transport**. Questo script implementa l'omonima soluzione ufficiale di Unity per il livello di trasporto della rete. Unity Transport si pone sopra UDP e lo estende con funzionalità avanzate. Aggiunge infatti una comunicazione affidabile, garantendo la consegna dei pacchetti critici senza aggiungere l'overhead di dati che caratterizza TCP, integra un sistema di cifratura e di autenticazione per proteggere i dati da accessi non autorizzati, è ottimizzato per una bassa latenza e alte capacità di throughput, infine è progettato per funzionare su diverse piattaforme e dispositivi. Attraverso questo script è possibile configurare alcuni parametri fondamentali: tra questi, il *Protocol Type*, che permette di scegliere se utilizzare una connessione diretta basata su UDP oppure passare attraverso il servizio Relay di Unity, opzione utile per garantire la raggiungibilità su Internet senza dover ricorrere a configurazioni manuali di rete. Direttamente legati a questa scelta vi sono i campi di *Address* e *Port*, che definiscono la sorgente a cui i client si connettono: nel caso di una connessione diretta corrispondono all'indirizzo del server host, mentre con Relay vengono sostituiti dai dati forniti dal servizio stesso. Altri parametri gestiscono il comportamento della connessione, come i timeout per stabilire o mantenere attive le sessioni e il numero di tentativi di connessione massimi. Particolarmente rilevante in fase di sviluppo è la sezione *Debug Simulator*, che consente di riprodurre artificialmente condizioni di rete degradate introducendo ritardo, variazioni di latenza (*jitter*) e perdita di pacchetti. Grazie a questa funzionalità è possibile testare la robustezza del sistema multiplayer senza la necessità di ricorrere a infrastrutture esterne, verificando come l'applicativo si comporta in scenari che simulano una reale connessione.

⁴Un *singleton* è un pattern di programmazione che limita l'istanza di una classe a un solo oggetto, accessibile globalmente.

4.2.3 XRI Network Game Manager

Il prefab **XRI Network Game Manager** costituisce il fulcro della gestione dei servizi di rete nel template VR Multiplayer di Unity. Lo script principale è un'istanza singleton che si occupa del ciclo di vita della connessione multiplayer, coordinando le fasi di autenticazione, creazione o ingresso nelle lobby e avvio della sessione di gioco in rete attraverso le altre componenti del prefab.

Lo script **AuthenticationManager** si occupa di fornire un **AuthenticationID** allo script principale, permettendo così alle varie istanze del progetto, che siano lanciate dall'editor di Unity o dall'eseguibile, di collegarsi ai servizi Unity. Garantisce anche la compatibilità del processo di autenticazione ai cloni creati tramite Parrel Sync.

La gestione delle lobby e l'integrazione con il servizio Relay sono affidate allo script **LobbyManager**, che fornisce metodi dedicati sia alla creazione di nuove lobby, sia alla ricerca di quelle esistenti attraverso un codice identificativo o mediante la funzione di accesso rapido (*quick join*). Questa funzionalità permette di individuare e unirsi alla prima lobby compatibile in base ai filtri definiti e, qualora nessuna sia disponibile, provvede automaticamente a crearne una nuova. Lo script si occupa inoltre di mantenere attive le sessioni tramite l'invio periodico di messaggi (*heartbeat*) al servizio Lobby e di configurare il componente Unity Transport con i parametri generati dalla sessione Relay, garantendo così la corretta instaurazione della comunicazione di rete tra host e client.

Infine, lo script **VoiceChatManager** gestisce l'intera pipeline di chat vocale posizionale basata su Vivox. Esso si sottoscrive allo stato di connessione esposto dallo script principale e, appena l'utente risulta autenticato, inizializza il servizio e gestisce i permessi legati al microfono; appena l'utente risulta connesso, esegue quindi il login a Vivox utilizzando l'**AuthenticationId** e i dati della lobby corrente. La resa spaziale dell'audio è configurabile tramite i parametri esposti dallo script: la distanza massima di udibilità (oltre la quale il parlante non è più percepito), la distanza di conversazione (superata la quale il livello sonoro viene progressivamente attenuato), l'intensità dell'attenuazione e il modello di decadimento adottato; sono inoltre definiti gli intervalli di volume minimo e massimo per i livelli di input e di output.

4.2.4 Offline Menù UI

Il prefab **Offline Menù UI** è la prima interfaccia che viene presentata all'utente all'avvio dell'applicazione. Viene gestita interamente dallo script **LobbyUI** e si articola in tre schermate principali. La prima consente di modificare il nome dell'utente e di personalizzare l'avatar cambiandone il colore del corpo.

Una volta confermate le modifiche, si accede alla sezione dedicata alle lobby, dove viene mostrata una lista delle stanze disponibili con il numero di giocatori presenti rispetto alla capienza massima. Selezionando una lobby non ancora al completo è possibile entrarvi direttamente. L'elenco può essere aggiornato manualmente tramite un apposito pulsante in alto a sinistra. Nella parte superiore della schermata è presente un campo per l'inserimento del codice di una lobby specifica, mentre un pulsante dedicato consente di sfruttare la funzione di quick join, che collega automaticamente l'utente alla prima stanza compatibile individuata dal **LobbyManager**.

Attraverso il pulsante *New Room* si accede invece alla terza schermata, dedicata alla creazione di una nuova lobby, in cui è possibile impostare se la stanza debba essere pubblica o privata, il nome e il numero massimo di utenti. Durante la fase di connessione vengono mostrate le diverse fasi del processo, con indicazione del successo o del fallimento. Una volta collegato con successo alla lobby, il menù viene disattivato insieme alla parete ad esso associata, che impediva all'utente di avanzare nello scenario.

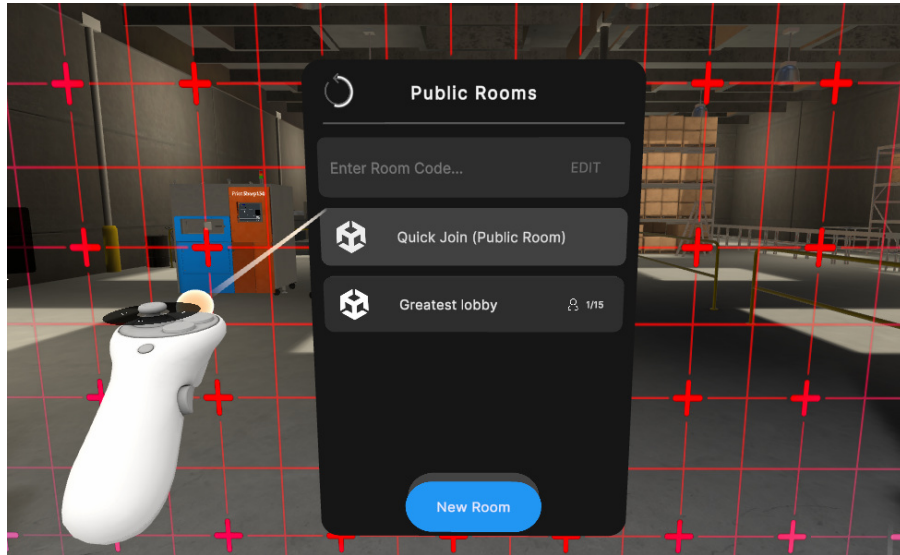


Figura 4.6: Schermata di selezione della lobby

4.2.5 Tutorial UI

Il prefab **Tutorial Player** è un elemento dell'interfaccia utente dedicato alla fruizione di un video dimostrativo che illustra i comandi e le azioni disponibili nel sistema, in modo da fornire una guida rapida e uniforme agli utenti senza dover uscire dall'esperienza VR. Integra un player video con i controlli essenziali (avvio/pausa e navigazione).

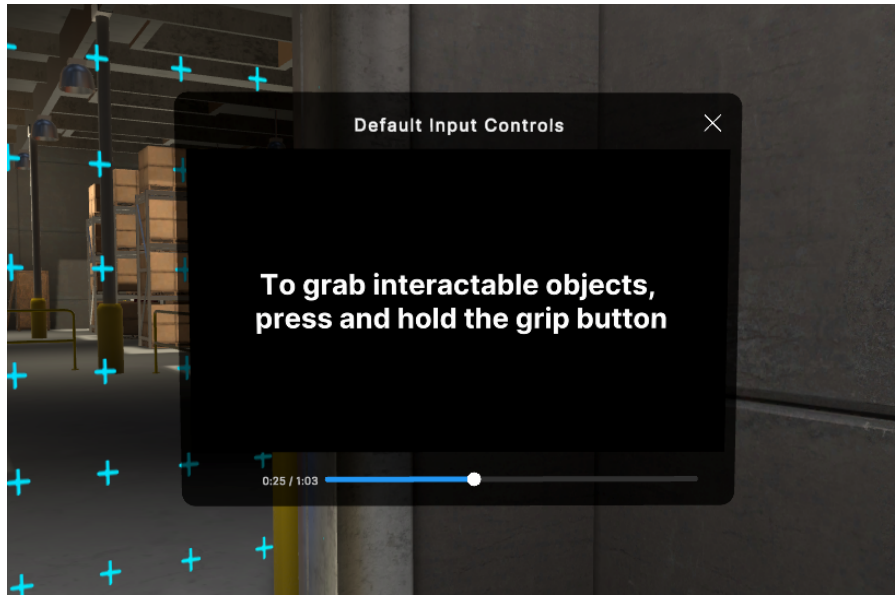


Figura 4.7: Interfaccia che mostra il tutorial dell'applicazione

4.2.6 Spatial Panel UI

Il prefab `Spatial Panel UI` è un pannello interattivo che l'utente può spostare nello spazio virtuale manipolandolo tramite l'apposita maniglia situata nella parte inferiore. L'oggetto è dotato di un `rigidbody` con gravità disabilitata e di un *collider*⁵ che, in combinazione con lo script `XRInteractableSnapVolume`, consente al raggio di interazione del `Near-Far Interactor` di agganciarsi correttamente al pannello. La pressione del tasto di presa permette quindi di traslare il menù, il quale rimane costantemente orientato verso l'utente grazie allo script `TurnToFace`, garantendo così una migliore leggibilità durante la manipolazione.

Dal pannello l'operatore può avviare le procedure relative alla macchina: un menù a tendina consente di selezionare la procedura, mentre un toggle permette di attivare o disattivare la modalità guidata. Premendo il pulsante *Start*, la procedura selezionata viene quindi eseguita secondo la modalità scelta. È inoltre presente un'icona informativa a forma di punto interrogativo che, al passaggio del raggio proiettato dal controller, mostra una schermata esplicativa contenente una descrizione della funzione del menù, del sistema delle procedure e delle due modalità operative.

Lo script `SpatialPanelManager`, eseguito all'avvio dell'applicazione, individua automaticamente il menù a tendina tra gli oggetti figli del pannello e ne popola dinamicamente le opzioni, recuperando i nomi delle procedure disponibili dal `ProceduresManager` mediante il metodo `GetAvailableProceduresNames()`, componente cardine del sistema che verrà approfondito nei paragrafi successivi.

⁵In Unity, un *Collider* è un componente che può avere diverse forme geometriche ed è usato per rilevare collisioni fisiche ed eventi di interazione.

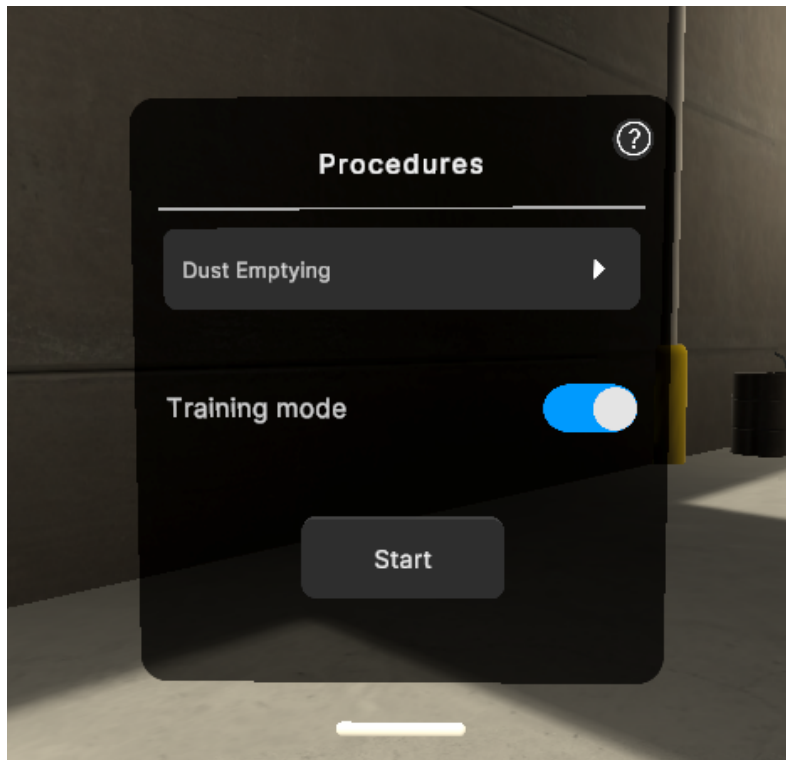


Figura 4.8: Menù manipolabile di selezione e avvio della procedura

4.3 Gestione e realizzazione delle procedure

In questa sezione si illustra come è stata realizzata la gestione e la sincronizzazione delle procedure, con particolare attenzione al ruolo del `ProceduresManager`. Successivamente vengono illustrate le due modalità operative principali, ed infine i componenti impiegati per la realizzazione delle procedure stesse.

4.3.1 Gestione delle procedure

Lo script `ProceduresManager` ha il compito di sincronizzare in rete lo stato delle procedure operative. Implementato come singleton, eredita dalla classe `NetworkBehaviour`, così da poter sfruttare le funzionalità offerte da NGO. Nell'*Inspector* espone due attributi: il riferimento al menù a tendina dello `SpatialPanelManager` e un toggle per la selezione della modalità. A livello di stato interno, definisce un enumeratore `Mode` con due valori, *Training* ed *Evaluation*, il cui valore corrente è sincronizzato tra tutti gli utenti mediante una `NetworkVariable<Mode>` modificabile esclusivamente dal server. Lo script mantiene inoltre una lista delle procedure disponibili `List<Procedure>` e una `NetworkList<int>` che rappresenta l'insieme delle procedure attive. Essendo serializzata, la lista delle procedure disponibili deve contenere i riferimenti agli oggetti figli del `ProceduresManager` che implementano la classe `Procedure`.

Quando viene stabilita una connessione, l'override del metodo `OnNetworkSpawn()` gestisce il caso dei cosiddetti *late joiners*, ossia gli utenti che si collegano a sessione già avviata. In tal caso, il metodo provvede ad avviare localmente le procedure già attive lato server, evitando disallineamenti tra host e client. Lo script si iscrive inoltre agli eventi di modifica della lista delle procedure attive tramite il metodo `OnActiveProceduresChanged` (`NetworkListEvent<int> changeEvent`). La classe `NetworkListEvent` espone un enumeratore che descrive i diversi tipi di modifiche applicabili alla lista; uno *switch* consente quindi di associare ciascun tipo di modifica all'evento corrispondente. Le variazioni vengono comunicate all'utente tramite notifiche generate dalla classe `PlayerHUDNotification`, fornita dal namespace `XRMultiplayer`, ad esempio quando viene avviata una nuova procedura.

L'avvio e la terminazione delle procedure sono gestiti mediante il meccanismo delle RPC. Il metodo `StartProcedure()`, invocato dal bottone *Start* dello `SpatialPanelManager`, verifica se la procedura selezionata sia già attiva. Qualora non lo sia e l'invocazione provenga dal server, la procedura viene avviata direttamente tramite `StartProcedureOnServer()`, che a sua volta notifica i client di avviarla attraverso la chiamata `StartProcedureClientRpc(int selectedIndex)`. Se invece è un client a richiedere l'avvio, il metodo `StartProcedureServerRpc(int selectedIndex)` inoltra la richiesta al server, che provvede a registrarla tra le procedure attive e a invocarne l'avvio sugli altri client tramite `StartProcedureOnServer()`. La terminazione di una procedura segue una logica analoga. Anche il cambio di modalità è gestito tramite RPC: se il valore del toggle varia, viene invocato il metodo `ChangeMode()`. Nel caso in cui la modifica avvenga lato server, la modalità viene aggiornata direttamente; al contrario, se la modifica è richiesta da un client, viene inoltrata al server tramite `RequestChangeModeServerRpc()`, che applica l'aggiornamento e lo propaga agli altri client.

Infine, `ProceduresManager` fornisce metodi `Get()` per ottenere la lista delle procedure attive, i relativi nomi e il valore corrente della modalità operativa.

4.3.2 Definizione delle procedure

Lo script `Procedure` rappresenta la classe astratta di riferimento per tutte le procedure implementate nel sistema. Estendendo `NetworkBehaviour`, definisce un insieme di `NetworkVariable` tra cui `currentStepIndex`, che mantiene sincronizzato il passo attuale della procedura; `trainingMode`, che indica in quale modalità è stata avviata; e `stepPanelNetworkID`, che memorizza l'identificativo del pannello responsabile della visualizzazione dei passi e del loro stato, utilizzato per aggiornare e mantenere coerente la UI. La classe contiene inoltre una lista di passi `List<ProcedureStep> steps`, un enumeratore `StepStatus` (con i valori *NotExecuted*, *Correct*, *Wrong*) e una `NetworkList<int> stepStatuses`, che garantisce la sincronizzazione dello stato dei passi e della relativa interfaccia. A supporto della logica di avanzamento, gestisce anche una lista di oggetti che implementano l'interfaccia `IEventSources`, sorgenti di eventi utilizzati per determinare la progressione della procedura. Infine, mantiene un dizionario in cui le chiavi sono i `MonoBehaviour` presenti in `eventSources`, e i valori sono tuple contenenti posizione e rotazione originali: questo consente di ripristinare, al termine della procedura, lo stato iniziale degli oggetti coinvolti.

Il metodo `OnEnable()` sottoscrive la variabile `currentStepIndex` agli eventi di cambiamento, demandando la gestione delle azioni conseguenti al metodo `OnCurrentStepChanged(int previousIndex, int newIndex)`. In `OnNetworkSpawn()` vengono invece inizializzati gli stati degli step a *NotExecuted*. L'avvio della procedura è gestito da `StartProcedure(int procedureIndex)`, invocato dal `ProceduresManager`: in questa fase la classe si registra agli eventi di tutti gli oggetti in `eventSources`, istanzia e sincronizza tra i client il pannello dei passi tramite `Spawn()` e abilita, per i late joiners, gli oggetti relativi al passo corrente. La terminazione avviene attraverso `StopProcedure()`, che esegue le operazioni inverse: rimuove le sottoscrizioni, distrugge il pannello su tutti i client, reimposta lo stato degli oggetti e richiama il metodo virtuale `OnProcedureStopped()`, sovrascrivibile dalle procedure concrete per operazioni specifiche. Il metodo `ActivateObjectsForStep(int stepIndex)`, invocato in `OnCurrentStepChanged()`, abilita gli oggetti del passo corrente tramite la componente `XRBaseInteractable`, e, se attiva, anche i componenti legati alla modalità di addestramento. I client possono richiedere il passaggio di step invocando `RequestStepChangeServerRpc(int newIndex)`, mentre `SetupStepPanelClientRpc(ulong stepPanelNetworkId)` viene eseguito sui client per inizializzare l'interfaccia del pannello dei passi. La classe dichiara inoltre il metodo astratto `UpdateCurrentStep()`, che deve essere implementato dalle singole procedure per aggiornare lo step in base a eventi specifici, e mette a disposizione vari metodi di accesso, tra cui `GetCurrentStep()` e `GetStepPanelNetworkId()`.

4.3.3 Definizione dei passi delle procedure

La classe `ProcedureStep` rappresenta l'unità elementare in cui è suddivisa una procedura. Ciascun passo racchiude le informazioni necessarie a descrivere una singola azione o sottofase dell'addestramento, includendo un nome identificativo, una descrizione testuale e le liste degli oggetti di scena che devono essere attivati o disattivati per consentirne la corretta esecuzione. Attraverso il campo delle dipendenze, uno step può inoltre richiedere il completamento di passi precedenti, garantendo così il rispetto dell'ordine logico del flusso operativo.

All'interno della classe è definita anche una struttura serializzata, `GhostTransformData`, che contiene tre attributi principali: `ghostObject`, un `GameObject` che rappresenta l'ologramma mostrato come guida per l'utente, e due campi aggiuntivi che ne specificano posizione e orientamento. Questo sistema consente di fornire un supporto visivo durante l'esecuzione dei passaggi, migliorando la chiarezza delle istruzioni.

Lo script `ProcedureStepsListEditor` estende l'editor di Unity per tutte le classi derivate da `Procedure`, con l'obiettivo di semplificare la configurazione degli step direttamente dall'*Inspector*. Unity permette infatti di personalizzare l'interfaccia di ispezione attraverso i *Custom Editor*, in cui il metodo `OnInspectorGUI()` viene sovrascritto per ridefinire la finestra dell'*Inspector*. In questo caso, lo script mantiene il comportamento di default sulle proprietà serializzate della classe, gestendo però in maniera dedicata la lista degli step.

Il funzionamento dello script si basa sull'uso della *reflection*⁶. Tramite *reflection*, l'editor ricerca nella sottoclasse concreta di **Procedure** un enumeratore denominato **StepType**. Se presente, i valori dell'enum vengono interpretati come indici e titoli dei passi della procedura. L'editor sincronizza automaticamente la dimensione della lista **steps** con il numero di valori definiti nell'enum e genera, per ciascuno di essi, un blocco di proprietà nell'Inspector che permette di inserire nome, descrizione, oggetti da attivare o disattivare e altri parametri specifici. Nel caso in cui l'enum non sia definito, l'editor mostra un messaggio di avviso (*HelpBox*), suggerendo allo sviluppatore di introdurre **StepType** nella sottoclasse per garantire la coerenza del sistema.

Questo approccio tipizzato consente di mantenere allineata la logica interna della procedura con la sua rappresentazione editoriale: se lo sviluppatore aggiunge o rimuove un valore nell'enum, la lista degli step viene adattata automaticamente, riducendo la possibilità di errori di configurazione e mantenendo un'interfaccia più chiara e coerente.

4.3.4 Modalità guidata

Per supportare la fase di addestramento e rendere più intuitiva l'interazione con gli oggetti virtuali, sono stati sviluppati due script complementari: **ShowGhostOnGrab** e **Highlighter**.

Lo script **ShowGhostOnGrab** gestisce la visualizzazione degli ologrammi introdotti in precedenza. Quando un oggetto interattivo viene afferrato, se lo step corrente della procedura lo prevede e la modalità è impostata su *Training*, l'ologramma corrispondente viene attivato e mostrato nella posizione corretta, per poi essere disattivato automaticamente al rilascio. In questo modo l'utente riceve un supporto visivo immediato sul corretto posizionamento o utilizzo dell'oggetto, riducendo la possibilità di errore. Per la realizzazione degli ologrammi è stato impiegato uno shader⁷ URP sviluppato con lo Shader Graph di Unity, basato sull'effetto Fresnel⁸ e arricchito con controlli dedicati al colore, all'opacità e all'intensità.

Lo script **Highlighter**, invece, implementa un sistema di evidenziazione dinamica degli oggetti di interesse. L'effetto si basa sulla modifica di due parametri di uno shader URP (*rim color* e *rim power*), che consentono di far pulsare colore e intensità del materiale. In questo modo, l'oggetto risulta chiaramente evidenziato finché è rilevante per la procedura; l'effetto viene automaticamente disattivato al momento dell'interazione diretta o al completamento dello step.

⁶Con il termine *reflection* si indica la capacità di un linguaggio di ispezionare e manipolare a runtime metadati e struttura del programma, come tipi, proprietà, metodi ed enumerazioni, consentendo di adattarne dinamicamente il comportamento.

⁷Uno *shader* è un programma eseguito dalla scheda grafica (GPU) che determina l'aspetto visivo di un oggetto, calcolando come la luce interagisce con la sua superficie e definendone colore, luminosità e riflessioni.

⁸In computer grafica, l'effetto *Fresnel* descrive come la quantità di luce riflessa da una superficie vari in base all'angolo di osservazione: gli oggetti appaiono più luminosi ai bordi rispetto al centro, creando un alone caratteristico.

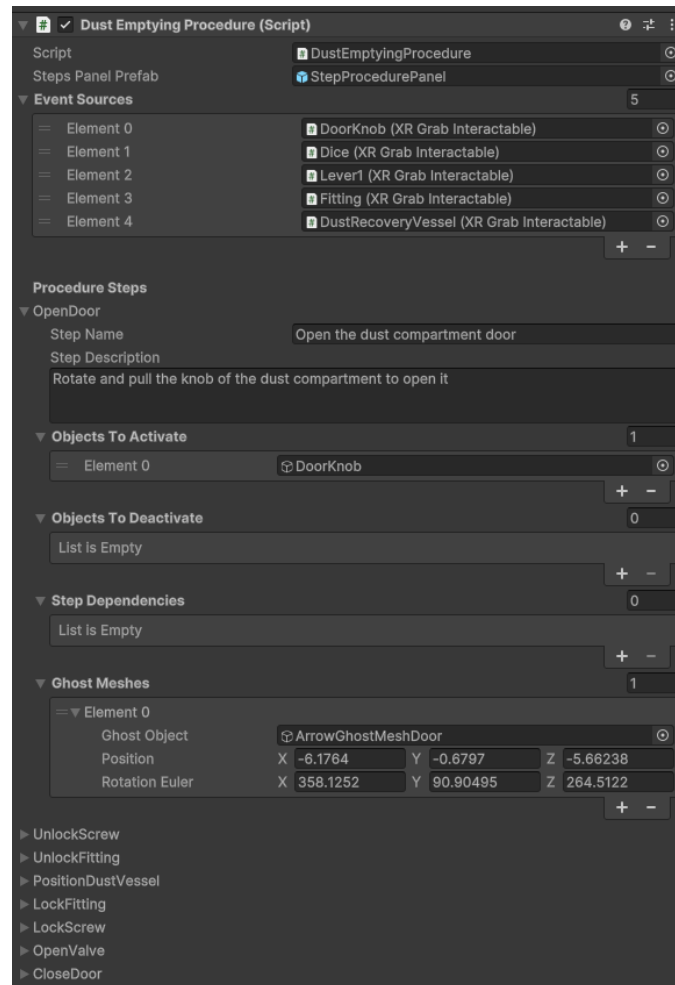


Figura 4.9: Finestra di modifica delle informazioni dei passi definita con la reflection

Combinati, questi due script costituiscono strumenti fondamentali per fornire feedback visivi chiari e immediati, migliorando l'efficacia dell'addestramento in realtà virtuale. A completare il sistema interviene lo script `StepProcedurePanelController`, che eredita da `NetworkBehaviour` e gestisce il pannello dei passi della procedura. Il metodo `Initialize (Procedure proc)` sottoscrive gli eventi di cambiamento della lista degli stati dei passi, imposta il nome della procedura come titolo del pannello e, in modalità guidata, mostra un elenco dei passi corredati da un'icona che ne rappresenta lo stato (non eseguito, corretto, errato).

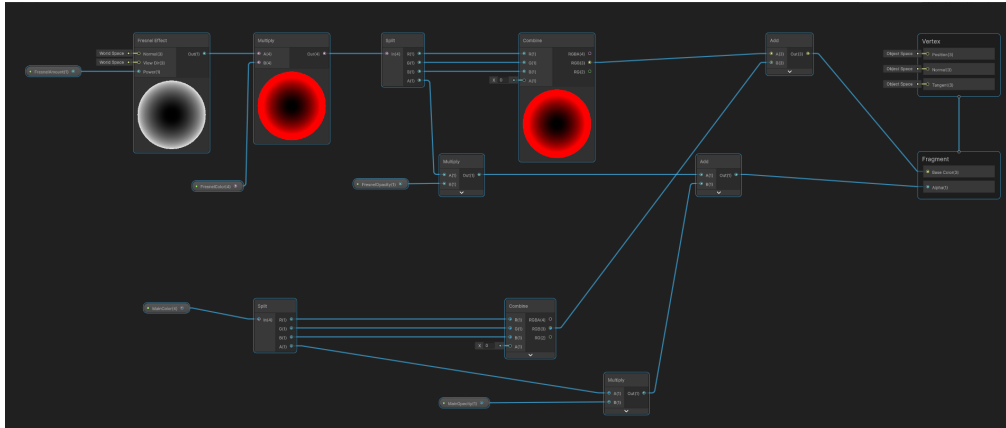


Figura 4.10: Shader dell'ologramma

4.3.5 Modalità di valutazione

Nella modalità di valutazione non viene attivato alcun tipo di supporto visivo o guida. In questo contesto, lo `StepProcedurePanelController` non mostra più la lista dei passi, ma un timer che l'utente può arrestare quando ritiene di aver completato la procedura. Una volta fermato, tramite lo stesso pulsante è possibile chiudere la schermata: viene così visualizzato l'elenco dei passi con il relativo stato, consentendo di verificare se le azioni sono state svolte correttamente.

Il sistema di gestione delle procedure è affiancato da una componente specifica per il tracciamento e la valutazione degli errori. A questo scopo è stata definita la classe astratta `ProcedureErrorTracker`, che fornisce l'infrastruttura di base per registrare la sequenza degli step eseguiti e confrontarla con il flusso previsto. Ogni volta che un passo viene completato, esso viene memorizzato in una cronologia interna e il suo stato viene aggiornato a *Correct* tramite una chiamata RPC al metodo `RequestSetStepStatusServerRpc(stepIndex, StepStatus)`. La classe definisce inoltre il metodo astratto `CheckErrors()`, che deve essere implementato dalle sottoclassi concrete per controllare la presenza di errori al termine della procedura.

Per poter effettuare il controllo degli errori, una procedura deve implementare l'interfaccia `IErrorCheckable`, definendo tra i propri attributi un `ProcedureErrorTracker`. Al termine dell'esecuzione, il tracker invoca il metodo `CheckErrors()`, che in caso di anomalie richiede al server di aggiornare lo stato di uno step a *Wrong*, sempre tramite il metodo citato precedentemente.



Figura 4.11: Lista dei passi della procedura, con indicato il loro stato

4.3.6 Componenti impiegati per realizzare le procedure

Per sincronizzare la logica e le interazioni fisiche sulla rete, ogni oggetto deve includere componenti di NGO; sono obbligatori `NetworkObject` e almeno un `NetworkBehaviour`. Quando un `NetworkObject` viene istanziato, il campo `GlobalObjectIdHash` identifica il prefab associato usato per creare i cloni sui vari client; localmente, ogni istanza è legata a un `NetworkObjectId`, che consente di indirizzare messaggi a quello specifico oggetto e alle sue controparti su tutti i client. L'ordine dei componenti è rilevante: `NetworkObject` deve precedere gli altri componenti di rete, inoltre l'ordine determina anche la sequenza delle chiamate di `OnNetworkSpawn()` sui vari `NetworkBehaviour`.

Le opzioni impostabili dal `NetworkObject` regolano diversi comportamenti rilevanti per l'oggetto: *Spawn With Observers* istanzia l'oggetto rendendolo subito visibile a tutti i client; se disattivata, l'oggetto resta visibile solo al server e la visibilità va gestita da codice. Questa opzione è attiva per tutti gli oggetti di rete del progetto. Per gli oggetti che necessitano di sincronizzare i dati della loro posizione o orientamento, l'opzione *Synchronize Transform* è attiva. *Active Scene Synchronization* e *Scene Migration Synchronization* sono disattivate poiché esiste una sola scena: la prima migrerebbe automaticamente gli oggetti tra scene quando cambia la scena attiva; similmente la seconda sincronizzerebbe gli spostamenti tra scene effettuati dallo `SceneManager`. *Always Replicate As Root* è disattiva, in caso contrario l'oggetto ignorerebbe l'eventuale padre ed effettuerebbe le trasformazioni senza vincoli; *Don't Destroy With Owner* è attiva per evitare che gli oggetti la cui gestione ricade su un client vengano distrutti alla sua disconnessione; *Auto Object Parent Sync*, che sincronizza i cambi di parentela tra gli oggetti, è disattivata.

Un aspetto centrale della sincronizzazione in NGO riguarda la gestione delle trasformazioni spaziali. A questo scopo vengono forniti due componenti distinti: `NetworkTransform`

e **ClientNetworkTransform**. Il primo adotta un modello *server-authoritative*: solo il server ha il compito di aggiornare posizione, rotazione e scala degli oggetti di rete, garantendo così la massima coerenza e sicurezza della simulazione distribuita, al prezzo però di una maggiore latenza percepita dai client. Il **ClientNetworkTransform**, invece, trasferisce l'autorità sull'oggetto al client che ne è *owner*, permettendo a quest'ultimo di inviare direttamente gli aggiornamenti al server e agli altri client. In questo modo i movimenti risultano più fluidi e reattivi lato client, caratteristica particolarmente utile in contesti immersivi, a fronte di una minore robustezza rispetto a fenomeni di incoerenza o comportamenti malevoli. Per questo motivo si è adottato questo componente per tutti gli oggetti che interagiscono con la fisica. La scelta tra i due approcci rappresenta dunque un compromesso tra consistenza globale e responsività locale. Le opzioni di sincronizzazione legate alla scala dell'oggetto sono state disabilitate visto che sono ininfluenti in questo progetto.

Altro componente presente su tutti gli oggetti interagibili che hanno interazioni fisiche è il **NetworkPhysicsInteractable**. Consente a chi non è owner dell'oggetto una manipolazione più fluida e responsiva sull'oggetto. Infatti disattiva temporaneamente il **ClientNetworkTransform** dell'oggetto quando l'utente locale che lo seleziona non è l'owner. Così facendo si evitano conflitti tra il movimento locale e gli aggiornamenti remoti dell'owner corrente. Altro comportamento importante implementato da questo script è lo scambio di proprietà in caso di collisioni tra due oggetti. Al di sopra di una certa soglia di intensità, la collisione fa cambiare owner all'oggetto con meno velocità, mantenendo quindi un comportamento più coerente della collisione.

Infine il componente **NetworkRigidbody** mantiene la simulazione della fisica solo a chi ha l'ownership dell'oggetto, mentre setta *isKinematic* a true su chi non ce l'ha, che si baserà solo sugli aggiornamenti inviati dall'owner.

Oltre ai componenti che gestiscono la sincronizzazione in rete, sono necessari quelli che implementano le interazioni. Il componente **XR Grab Interactable**, applicato a un oggetto dotato di collider, abilita la possibilità per quest'ultimo di essere selezionato e afferrato da un interactor. Il comportamento dell'interazione è configurabile attraverso diversi parametri. In particolare, è possibile definire la modalità di movimento durante la presa. In modalità *Instantaneous* l'oggetto si sposta istantaneamente seguendo il punto di attacco, senza fisica né interpolazioni, consentendo una massima reattività, ma poco realismo. In modalità *Kinematic* l'oggetto segue il movimento senza essere influenzato dalla fisica ma mantiene collisioni e interazioni di base. Infine in modalità *Velocity Tracking*, la soluzione adottata in questo caso, il movimento è ottenuto applicando velocità e rotazioni al rigidbody, risultando più realistico e fisicamente accurato, ma meno stabile.

L'**XR Socket Interactor** realizza un punto di aggancio capace di selezionare e trattenere un singolo **XR Interactable** che entra nel suo volume di rilevazione, fornendo un meccanismo di aggancio stabile e ripetibile (ad esempio per riporre utensili o inserire componenti in sede). Il socket valuta i candidati in *hover* secondo diversi criteri: *Interaction Layer Mask*, che determina i layer di interazione ammessi, escludendo quindi gli oggetti che non ne fanno parte; *Interactable Filters*, una lista di filtri utile per accettare solo tipi o ruoli specifici di oggetti; distanza e priorità, utilizzate per stabilire quale oggetto debba essere selezionato.

Il componente **Hinge Joint** di Unity implementa un vincolo fisico che collega un rigidbody a un punto fisso nello spazio o a un secondo corpo rigido, consentendo unicamente la rotazione attorno a un asse prestabilito. Dal punto di vista meccanico, tale giunto corrisponde a una cerniera ed è particolarmente indicato per la modellazione di elementi quali porte, leve o sportelli. Il componente espone diversi parametri di configurazione che ne determinano il comportamento dinamico, tra i più rilevanti: *Anchor* e *Axis* definiscono rispettivamente il punto e l'asse di rotazione del vincolo nel sistema di riferimento locale. *Use Limits* abilita la restrizione del movimento angolare entro un intervallo definito da *min* e *max*, utile per simulare escursioni controllate. Grazie a tali proprietà, l'**Hinge Joint** costituisce un elemento versatile per la simulazione di meccanismi articolati, garantendo un controllo accurato della cinematica rotazionale e una risposta fisicamente plausibile. L'impiego di questo componente risulta quindi di particolare rilievo nelle applicazioni di realtà virtuale e di training industriale, dove la fedeltà del comportamento meccanico influenza direttamente la qualità dell'interazione e il realismo percepito dall'utente.

Infine, ogni oggetto deve possedere uno script che implementi l'interfaccia **IEvent Source**, responsabile di lanciare l'evento per fare avanzare la procedura. Nel caso delle leve, ad esempio, questo ruolo è svolto dallo script **VRLever**, che genera l'evento di apertura o chiusura superata una certa soglia.

4.3.7 Procedure realizzate

Per ciascuna procedura viene definita una classe specifica che eredita da **Procedure**. All'interno di tale classe viene implementato il metodo **UpdateCurrentStep(object sender, EventArgs e)**, responsabile della gestione degli eventi generati dagli oggetti inclusi nella lista **eventSources**.

La prima procedura sviluppata riguarda lo svuotamento del vaso delle polveri e si articola nei seguenti passaggi: apertura della porta dello scompartimento polveri, sblocco della vite del morsetto, apertura del morsetto, inserimento del serbatoio, apertura della valvola delle polveri, chiusura del morsetto e, infine, blocco della vite del morsetto.

La seconda procedura implementata è invece relativa all'inserimento di una piastra metallica nella macchina. Essa prevede i seguenti passi: sblocco dello sportello, apertura dello sportello, orientamento dei blocchi, posizionamento della piastra, serraggio delle viti, pressione del tasto manuale, inserimento del valore dell'altezza lungo l'asse *z*, impostazione della velocità lungo l'asse *z*, pressione del tasto di movimento, attesa del corretto posizionamento della piastra, pressione del tasto standby, chiusura dello sportello e, infine, blocco dello sportello.



Figura 4.12: Operatori che si esercitano nelle due procedure.

4.4 Ambiente virtuale

Per la realizzazione grafica di questo sistema è stata adottata la Universal Render Pipeline⁹ (URP) di Unity, una soluzione progettata per garantire un compromesso ottimale tra qualità visiva e prestazioni. La scelta è ricaduta su URP in quanto particolarmente adatta a piattaforme mobile e dispositivi standalone per la realtà virtuale, come il Meta Quest 2, grazie alla sua capacità di gestire in modo efficiente le risorse hardware limitate e di supportare tecniche di rendering specificamente pensate per la VR.

Tra queste tecniche, un ruolo di rilievo è svolto dal Single Pass Instanced Rendering, che consente di ridurre il carico computazionale duplicando il rendering per i due occhi in un'unica elaborazione grafica, limitando così il numero di operazioni e ottimizzando le prestazioni. Questo approccio si è rivelato essenziale per mantenere una frequenza di aggiornamento stabile compresa tra 72 e 90 frame al secondo, soglia considerata fondamentale per garantire un'esperienza immersiva e confortevole, riducendo al minimo il rischio di affaticamento visivo e motion sickness.

4.4.1 Modelli 3D e materiali

L'ambiente virtuale è stato realizzato a partire da modelli 3D provenienti da diverse risorse esterne. In particolare, l'edificio principale e alcuni elementi decorativi interni provengono dalla risorsa *XR Simulation Environments* di Unity [38], una raccolta di ambienti già ottimizzati per applicazioni di realtà estesa. Il magazzino è stato successivamente modificato in Blender, intervenendo sulla struttura e sulle dimensioni per adattarlo allo scenario della

⁹Una render pipeline è il processo che, all'interno di un motore grafico, elabora gli elementi di una scena (modelli, texture, luci, effetti) attraverso una sequenza di passaggi per produrre l'immagine finale mostrata sullo schermo.

stampante 3D. Ulteriori modelli sono stati reperiti dalla libreria *FAB* [39]: trattandosi di risorse di alta qualità e quindi più complesse dal punto di vista geometrico, si è proceduto alla loro semplificazione mediante il modificatore *Decimate* di Blender, che consente di ridurre il numero complessivo di poligoni mantenendo la forma generale del modello.

Per quanto riguarda i materiali, sono state impiegate texture¹⁰ provenienti dal database *BlenderKit* [40], mentre i modelli ottenuti da *FAB* includevano già le relative mappe. Un ulteriore accorgimento adottato per ottimizzare le prestazioni ha riguardato la scelta degli shader applicati ai materiali delle superfici. In particolare, si è optato per l'utilizzo dello shader *Simple Lit* per gli oggetti e le superfici di grandi dimensioni o di minore rilevanza visiva, come pavimenti, pareti o elementi strutturali, e per lo shader *Lit* nelle aree o sugli oggetti di maggiore interesse visivo e interattivo. Lo shader *Lit* adotta infatti un modello di illuminazione fisicamente basato (PBR), capace di simulare in modo realistico il comportamento della luce su materiali metallici o dielettrici, supportando effetti avanzati come riflessioni, emissione e occlusione ambientale, a fronte però di un maggiore costo computazionale. Lo shader *Simple Lit*, al contrario, utilizza un modello di illuminazione semplificato, che considera principalmente una sola sorgente di luce direzionale, offrendo così un compromesso efficace tra qualità visiva e prestazioni.

Inoltre, Unity implementa nello shader *Lit* la tecnica del *channel packing*, che consiste nel memorizzare più texture all'interno dei diversi canali di colore di una stessa immagine, riducendo così il numero di file da caricare. In questo caso, la *Metallic Map* è salvata nel canale rosso, la *Smoothness Map* nel canale alpha e la *Ambient Occlusion Map* nel canale verde. Per ottenere questa ottimizzazione, le diverse mappe sono state combinate manualmente nei rispettivi canali tramite un software di editing di immagini.

4.4.2 Illuminazione

Anche l'illuminazione dell'ambiente è stata progettata mantenendo come requisito principale le prestazioni. Per questo motivo si è scelto di adottare un'illuminazione interamente precalcolata (*baked*), evitando così costosi calcoli di illuminazione in tempo reale e garantendo un frame rate più stabile.

La scena utilizza una luce direzionale principale, che simula la direzione del sole, affiancata da un insieme di *point light* collocate nelle aree che risultavano eccessivamente scure. Durante la fase di *bake* dell'illuminazione, Unity genera una *lightmap*, ossia una texture che memorizza le informazioni di illuminazione statica (diretta e indiretta) precalcolata per ciascuna superficie. Tuttavia, alcuni oggetti presentavano coordinate *UV*¹¹ sovrapposte nella *lightmap*, causando artefatti visivi come zone sature o mal illuminate. Per la maggior parte dei modelli, il problema è stato risolto abilitando in Unity la generazione automatica di una seconda mappa *UV* dedicata esclusivamente al *bake* dell'illuminazione.

¹⁰Una *texture* è un'immagine bidimensionale utilizzata per mappare le proprietà superficiali di un modello 3D, definendone il colore, i dettagli e le caratteristiche visive come riflessioni o rugosità.

¹¹Il termine *UV* indica le coordinate bidimensionali utilizzate per mappare una texture sulla superficie tridimensionale di un modello. Ogni vertice è associato a una coppia di coordinate (u, v) che definisce la corrispondenza tra i punti della texture e quelli del modello 3D.

Nei casi più complessi, le coordinate *UV* sono state rigenerate manualmente in Blender, assicurando una corretta proiezione. Nelle impostazioni dei modelli in Unity, sono stati regolati alcuni parametri legati al *bake* della luce: per gli oggetti di grandi dimensioni è stata ridotta la scala o la loro risoluzione nella *lightmap*, così da limitare lo spazio occupato nella texture e riservare una definizione più elevata agli elementi principali della scena.

Per consentire agli oggetti dinamici di reagire correttamente alla luce, sono stati inseriti dei *Light Probe Groups*. I *light probe* rappresentano punti nello spazio in cui vengono memorizzate informazioni sull'illuminazione indiretta calcolata durante il *bake*. Quando un oggetto mobile attraversa la scena, il suo *shading* viene interpolato in base ai valori dei probe circostanti, simulando così una reazione coerente alle variazioni luminose dell'ambiente anche in assenza di luci in tempo reale. È importante che la rete di probe sia sufficientemente densa, così da garantire una transizione luminosa fluida e un'illuminazione credibile per tutti gli oggetti in movimento.

Infine, per migliorare ulteriormente la resa visiva, sono stati aggiunti anche dei *Reflection Probes*, che si occupano di catturare l'ambiente circostante in forma di mappe cubiche (*cubemap*), consentendo così ai materiali riflettenti di mostrare riflessi coerenti con la scena. Questa tecnica risulta particolarmente efficace per superfici metalliche, vetrose o lucide, che necessitano di un minimo di realismo.

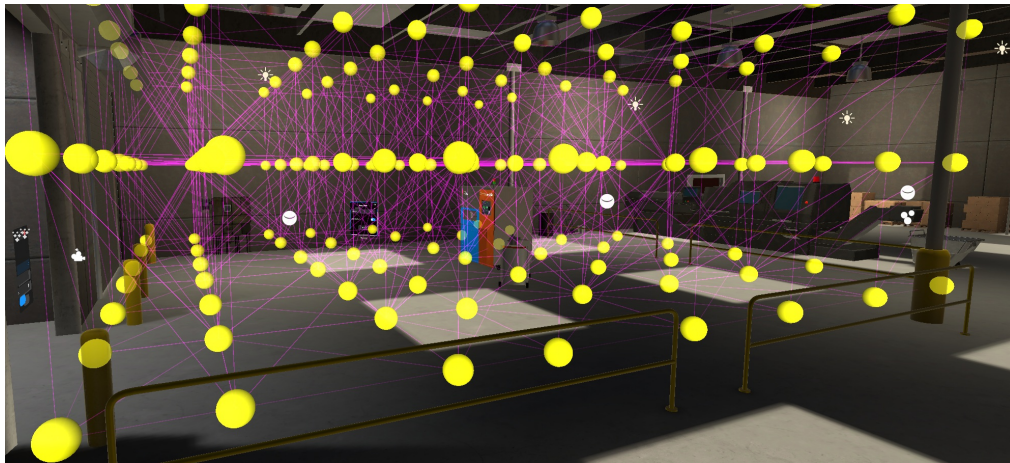


Figura 4.13: Rete di light probes definita.

Capitolo 5

Valutazione sperimentale e risultati

Il capitolo ha l'obiettivo di descrivere il processo di valutazione del sistema di training virtuale sviluppato. In questa sezione vengono illustrati gli obiettivi dei test e le metriche adottate per la loro conduzione. Viene inoltre descritto il procedimento seguito durante le sessioni di prova. Il capitolo presenta poi i risultati ottenuti dai test, con un'analisi quantitativa e qualitativa delle risposte, evidenziando i principali punti di forza del sistema e le criticità riscontrate.

5.1 Test effettuati

L'obiettivo principale dei test preliminari effettuati è valutare l'usabilità e il carico cognitivo del sistema sviluppato. A tal fine, è stato sottoposto a un gruppo di utenti un test pratico delle due procedure, seguito dalla compilazione dei questionari System Usability Scale (SUS) [41] e NASA Task Load Index (NASA-TLX) [42]. Al primo questionario sono state aggiunte alcune domande mirate sulla tipologia di sistema in questione, che approfondiscono il giudizio sull'aspetto collaborativo e industriale.

Proposto da John Brooke nel 1986, il questionario SUS è ampiamente utilizzato in ambito accademico e industriale per la sua semplicità di somministrazione e per la capacità di fornire una misura sintetica ma affidabile dell'esperienza d'uso percepita dagli utenti. Il SUS è composto da dieci affermazioni a cui il partecipante risponde utilizzando una scala Likert, uno strumento comunemente utilizzato nei questionari di valutazione per misurare il grado di accordo o disaccordo rispetto a un'affermazione. Tale scala consente di quantificare percezioni soggettive in valori numerici ordinati, generalmente compresi tra 1 (fortemente in disaccordo) e 5 (fortemente d'accordo), permettendo un'analisi statistica dei dati raccolti. Le domande sono strutturate in modo alternato tra formulazioni positive e negative, con l'intento di ridurre eventuali bias cognitivi o di risposta automatica.

Per il calcolo del punteggio complessivo, le risposte vengono convertite in valori numerici compresi tra 0 e 4: per le domande formulate positivamente si sottrae 1 al valore assegnato, mentre per quelle negative si sottrae da 5 il valore della risposta. La somma dei dieci punteggi parziali viene poi moltiplicata per 2,5 in modo da ottenere un risultato

finale su una scala da 0 a 100, dove punteggi più alti corrispondono a un livello di usabilità percepita maggiore.

Un valore medio di 68 rappresenta la soglia di accettabilità generalmente riconosciuta, mentre punteggi superiori a 80 indicano un sistema percepito come altamente usabile (Excellent), e valori inferiori a 50 suggeriscono criticità di design o difficoltà di interazione.

Il NASA-TLX è un questionario sviluppato nel 1988 dal Human Performance Group del NASA Ames Research Center con l'obiettivo di misurare il carico di lavoro soggettivo percepito da un utente durante l'esecuzione di un compito. Questo strumento è ampiamente utilizzato in ambito ergonomico, cognitivo e di interazione uomo-macchina (HCI), poiché consente di valutare in modo multidimensionale quanto un'attività risulti impegnativa dal punto di vista mentale, fisico e temporale, nonché quanto l'utente si senta soddisfatto o frustrato durante la sua esecuzione.

Il questionario si basa su sei dimensioni fondamentali. A ciascuna di queste dimensioni viene assegnato un punteggio su una scala numerica da 0 a 100, suddivisa in 20 intervalli, dove valori più alti indicano un maggiore carico percepito (ad eccezione della dimensione "successo percepito", in cui la relazione è inversa). Nel presente lavoro è stata adottata una scala semplificata da 0 a 10, i punteggi vengono poi normalizzati sulla scala 0-100, moltiplicando per 10 i valori medi delle singole dimensioni. In questo caso è stata utilizzata la versione semplificata non ponderata (Raw TLX), in cui il punteggio complessivo è calcolato come media aritmetica dei sei fattori, senza l'assegnazione di pesi relativi tra le dimensioni. Il valore ottenuto fornisce un'indicazione quantitativa del carico cognitivo e fisico complessivo percepito dagli utenti. In generale, punteggi inferiori a 40 indicano un carico basso, valori compresi tra 40 e 60 un livello medio, mentre punteggi superiori a 60 denotano un'elevata richiesta cognitiva o fisica. Il questionario rappresenta quindi un complemento ideale al SUS, in quanto consente di analizzare la fatica mentale e il livello di sforzo richiesto dall'interazione, offrendo una prospettiva più ampia sull'efficacia complessiva del sistema.

5.1.1 Caratteristiche del campione

I test hanno coinvolto un totale di 18 partecipanti, con un'età media di 25,1 anni, aventi la maggior parte un livello di istruzione universitario o equivalente. Per quanto riguarda la familiarità con la realtà virtuale immersiva, 5 dei partecipanti avevano già utilizzato un visore HMD in precedenza, principalmente per esperienze ludiche o dimostrative, mentre i restanti sperimentavano per la prima volta un sistema di questo tipo.

5.1.2 Metodologia dei test

Ogni partecipante ha eseguito una sessione in presenza con un altro utente della durata media di 15 minuti, utilizzando il visore Meta Quest 2 o il visore Meta Quest Pro, entrambi in modalità standalone. Durante la prova è stato richiesto di completare le due procedure simulate in modalità guidata e successivamente in modalità di valutazione, stando in piedi e stazionari. Gli utenti, dopo una breve introduzione sullo scopo del sistema, erano liberi di porre domande nella sezione precedente alla creazione della lobby. Successivamente, è

stato richiesto di operare con un grado maggiore di autonomia. Al termine, è stato chiesto di compilare i questionari SUS e NASA-TLX tramite Google Forms.

5.2 Risultati dei questionari

Nelle seguenti tabelle sono riportate le domande poste, la distribuzione delle risposte e la loro deviazione standard.

Affermazione (SUS)	1	2	3	4	5	σ
Ritengo che mi piacerebbe usare frequentemente questo sistema.	0.0%	0.0%	5.6%	44.4%	50.0%	0.616
Ho trovato il sistema inutilmente complesso.	50.0%	38.9%	11.1%	0.0%	0.0%	0.698
Ho trovato il sistema facile da usare.	0.0%	5.6%	22.2%	38.9%	33.3%	0.907
Penso che avrei bisogno del supporto di una persona esperta per utilizzare il sistema.	22.2%	55.6%	16.7%	5.6%	0.0%	0.802
Le varie funzioni del sistema mi sembrano ben integrate.	0.0%	0.0%	5.6%	44.4%	50.0%	0.616
Ho riscontrato troppa incoerenza nelle funzioni del sistema.	55.6%	38.9%	5.6%	0.0%	0.0%	0.618
Immagino che la maggior parte delle persone imparerebbe a usare rapidamente questo sistema.	0.0%	0.0%	22.2%	55.6%	22.2%	0.686
Ho trovato il sistema macchinoso o scomodo da usare.	66.7%	27.8%	5.6%	0.0%	0.0%	0.608
Mi sono sentito sicuro nell'utilizzare il sistema.	0.0%	0.0%	22.2%	61.1%	16.7%	0.639
Ho dovuto imparare molte cose prima di poter utilizzare il sistema.	50.0%	38.9%	11.1%	0.0%	0.0%	0.698
Punteggio medio SUS	81.7 su 100					

Tabella 5.1: Distribuzione e deviazione standard (σ) delle risposte al questionario System Usability Scale

Domanda (NASA-TLX)	0–2	3–4	5–6	7–8	9–10	σ
Quanto è stato mentalmente impegnativo il compito?	11.1%	27.8%	50.0%	11.1%	0.0%	1.42
Quanto è stato fisicamente impegnativo il compito?	66.7%	27.8%	5.6%	0.0%	0.0%	1.25
Quanto ti è sembrato di dover lavorare di fretta o sotto pressione?	66.7%	16.7%	16.7%	0.0%	0.0%	1.39
Quanto hai avuto successo nel portare a termine il compito richiesto di fare?	0.0%	0.0%	11.1%	83.3%	5.6%	1.00
Quanto impegno ha richiesto il tuo livello di prestazione raggiunto?	5.6%	33.3%	33.3%	27.8%	0.0%	1.37
Quanto ti sei sentito frustrato, irritato, stressato o scoraggiato durante il compito?	38.9%	50.0%	11.1%	0.0%	0.0%	1.08
Punteggio medio NASA-TLX	39.6 su 100					–

Tabella 5.2: Distribuzione e deviazione standard (σ) delle risposte al questionario NASA-TLX

Domanda	1	2	3	4	5
Ritieni che la presenza di un altro utente durante l'esperienza VR abbia un impatto positivo nel processo di addestramento?	0.0%	0.0%	0.0%	27.8%	72.2%
Quanto è stato efficace il coordinamento e la comunicazione con gli altri partecipanti nel completare i compiti?	0.0%	0.0%	11.1%	38.9%	50.0%
Pensi che l'esperienza VR ti abbia aiutato/a ad apprendere o memorizzare meglio la procedura rispetto a un metodo tradizionale (video, manuale, ecc.)?	0.0%	0.0%	0.0%	16.7%	83.3%
In generale, ritieni che il sistema possa essere uno strumento utile per la formazione tecnica su macchinari complessi?	0.0%	0.0%	0.0%	44.4%	55.6%

Tabella 5.3: Distribuzione delle risposte per le domande aggiuntive.

Nei seguenti grafici sono riportati i valori dei punteggi per ogni partecipante.

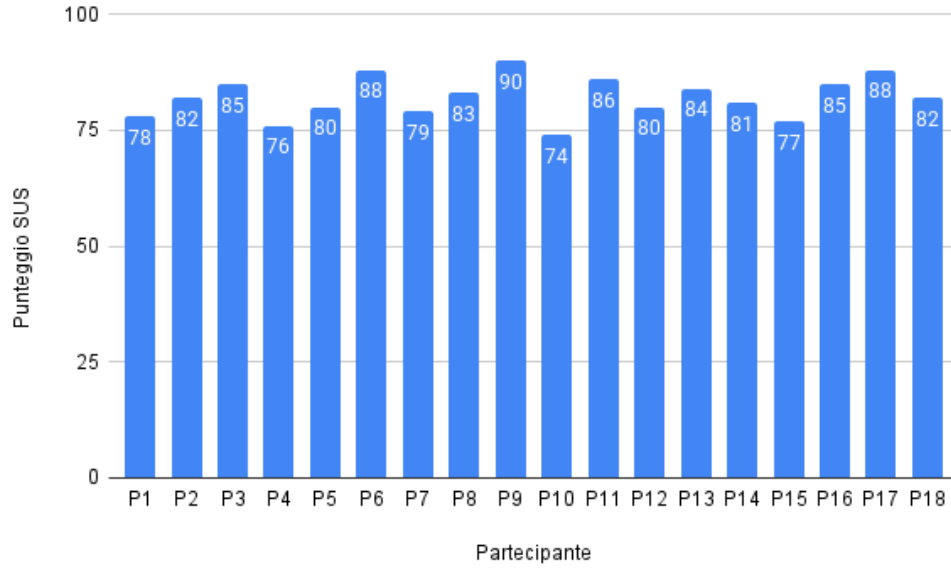


Figura 5.1: Valori SUS per utente.

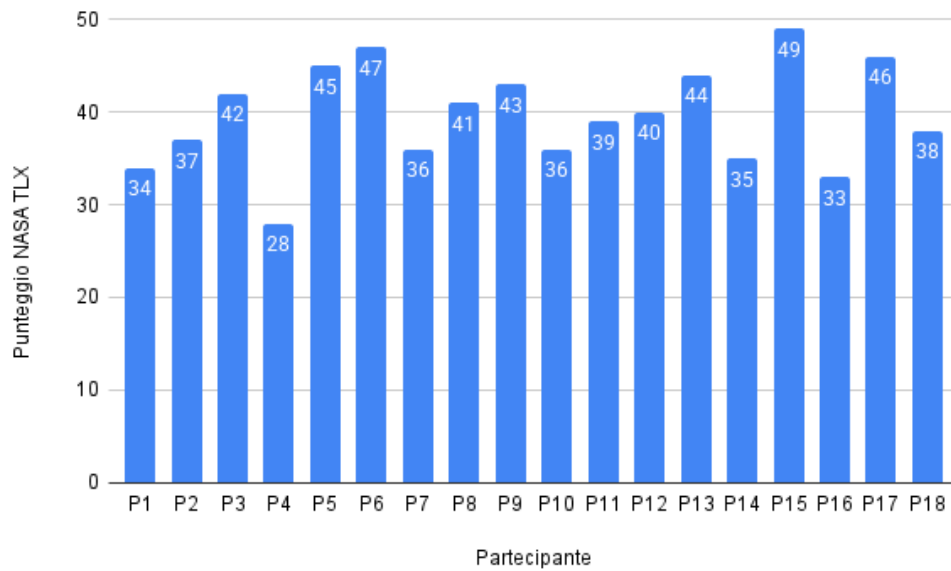


Figura 5.2: Valori NASA-TLX per utente.

5.3 Analisi dei risultati

Il questionario (SUS) ha restituito un punteggio medio pari a 81,7/100. Le risposte evidenziano come la quasi totalità dei partecipanti abbia trovato il sistema semplice da utilizzare, coerente e ben integrato. Solo una minoranza ha percepito complessità o incoerenze tra le funzioni, indicando che l'interfaccia e le modalità d'interazione risultano intuitive anche per utenti non esperti. Il risultato conferma quindi l'ottima qualità dell'esperienza d'uso.

Il questionario NASA-TLX ha restituito un punteggio medio complessivo, pari a 39,6/100, indicando un livello di stress e affaticamento moderato, segno che le attività richieste non hanno comportato uno sforzo eccessivo né dal punto di vista cognitivo né da quello fisico. In particolare, i valori più bassi sono stati registrati per le dimensioni della domanda fisica e della frustrazione, mentre punteggi leggermente più elevati si riscontrano nelle dimensioni della domanda mentale e dell'impegno richiesto. Questo risultato può essere attribuito al fatto che, oltre a dover apprendere una procedura di un macchinario industriale, alcuni partecipanti dovevano per la prima volta prendere confidenza con il visore e con le modalità di navigazione e interazione in realtà virtuale. Sono emerse infatti alcune difficoltà iniziali di interazione: nonostante il video introduttivo, alcuni utenti non avevano pienamente compreso la natura simulativa dell'ambiente e tendevano a tentare di afferrare gli oggetti a distanza, prima di adattarsi al funzionamento effettivo del sistema. La dimensione del successo percepito ha invece ottenuto valori elevati, suggerendo che gli utenti hanno avuto una sensazione di controllo e di efficacia durante l'interazione con il sistema.

Infine, le domande aggiuntive hanno indagato la percezione soggettiva dell'esperienza collaborativa e formativa in realtà virtuale. I risultati mostrano una valutazione positiva da parte dei partecipanti. La quasi totalità degli utenti (72,2% con valore massimo) ha ritenuto che la presenza di un altro utente abbia avuto un impatto positivo nel processo di addestramento, favorendo la comunicazione e la comprensione dei compiti. Anche il coordinamento e la comunicazione tra i partecipanti sono stati valutati positivamente, con il 50% delle risposte sul valore massimo e un ulteriore 38,9% sul livello immediatamente inferiore. Analogamente, l'83,3% degli intervistati ha dichiarato che il sistema può rappresentare un valido strumento di formazione tecnica per macchinari complessi, mentre oltre la metà (55,6%) ha riconosciuto che l'esperienza VR ha facilitato l'apprendimento rispetto ai metodi tradizionali.

Nel complesso, i risultati preliminari ottenuti convergono nel delineare un quadro positivo: il sistema risulta usabile, coinvolgente e a basso carico cognitivo, con un evidente potenziale come strumento formativo collaborativo in ambito industriale.

5.4 Valutazioni aggiuntive

Un ulteriore sviluppo previsto per la valutazione del sistema riguarda l'approfondimento della formazione tramite il LBT. Nel contesto del sistema di addestramento VR sviluppato, tale approccio potrebbe essere indagato attraverso nuove sessioni sperimentali in cui un utente, dopo aver completato con successo le procedure in modalità di addestramento,

assuma il ruolo di tutor o istruttore. In questa fase, egli potrebbe guidare un secondo utente all'interno della modalità di valutazione, illustrando passo dopo passo le corrette sequenze operative e intervenendo in caso di errori.

La realizzazione di test strutturati secondo questa logica permetterebbe di valutare l'efficacia dell'apprendimento attivo rispetto alla semplice esecuzione delle procedure. In particolare, si potrebbero analizzare parametri quali la memorizzazione a lungo termine, la capacità di trasferimento delle conoscenze e il grado di sicurezza operativa raggiunto dagli utenti. Inoltre, l'osservazione dell'interazione tra i due ruoli – “istruttore” e “operatore” – offrirebbe informazioni preziose sull'efficacia del canale comunicativo e collaborativo del sistema, già emerso come elemento chiave nei test precedenti.

Oltre alle prove condotte in presenza, un'estensione naturale delle attività di valutazione potrebbe riguardare test effettuati in modalità remota, con utenti fisicamente distanti tra loro. Questo tipo di sperimentazione permetterebbe di analizzare più a fondo le potenzialità collaborative dell'applicativo, verificando l'efficacia della comunicazione e del coordinamento tra operatori che interagiscono nello stesso ambiente virtuale pur trovandosi in luoghi diversi.

In particolare, sarebbe possibile valutare aspetti quali la stabilità della rete, la sincronizzazione degli oggetti e la percezione della presenza sociale in condizioni di latenza variabile, simulando scenari reali di formazione distribuita tra più sedi aziendali.

Capitolo 6

Conclusioni e sviluppi futuri

6.1 Considerazioni finali

Il lavoro svolto in questa tesi ha portato allo sviluppo di un applicativo VRTS multiutente, volto a supportare la formazione tecnica sull'utilizzo di una stampante 3D industriale. L'applicazione, realizzata in Unity e basata sul framework Netcode for GameObjects, ha permesso di ricreare un ambiente immersivo collaborativo, in cui più operatori possono interagire simultaneamente con la macchina e con l'ambiente circostante.

Attraverso l'implementazione di un sistema di gestione delle procedure e di un'interfaccia utente contestuale, è stato possibile guidare l'utente passo dopo passo nello svolgimento delle operazioni previste, integrando funzionalità di valutazione automatica degli errori e tracciamento delle azioni. I test di usabilità e di carico cognitivo condotti tramite i questionari SUS e NASA-TLX hanno evidenziato risultati incoraggianti: il sistema è stato giudicato altamente usabile e con un livello di carico cognitivo medio-basso, a conferma della validità dell'approccio immersivo adottato.

Il sistema offre un'esperienza coerente con i requisiti industriali, favorendo la memorizzazione dei passaggi procedurali e promuovendo la collaborazione tra utenti in un contesto controllato e sicuro. I feedback raccolti hanno inoltre confermato la validità dell'utilizzo della VR multiutente come strumento didattico e di addestramento, in grado di migliorare l'efficacia della formazione e ridurre i costi e i rischi associati all'utilizzo diretto dei macchinari reali.

6.2 Sviluppi futuri

Il sistema realizzato rappresenta una base solida ed estendibile per futuri sviluppi. Tra i miglioramenti proposti, si possono individuare diverse direzioni di approfondimento, sia in termini di interazione e realismo, sia di funzionalità didattiche e gestionali:

- **Sistema di ping e evidenziazione condivisa:** L'integrazione di un sistema di evidenziazione (*ping*), consentirebbe agli utenti di segnalare oggetti o aree di interesse agli altri partecipanti. Questa funzione aumenterebbe la cooperazione e

la comunicazione visiva tra gli operatori, facilitando la comprensione reciproca e la coordinazione durante le procedure.

- **Sezione offline di preparazione:** Una sezione preliminare “offline” potrebbe essere introdotta per simulare le fasi di preparazione all'utilizzo della macchina, dove si indossano i dispositivi di sicurezza, si selezionano gli strumenti da utilizzare per un dato compito o si permette una personalizzazione dell'area di lavoro.
- **Assistente vocale o AI interattivo:** Un ulteriore miglioramento consiste nell'introduzione di un assistente vocale o virtuale, rappresentato da un modello 3D o integrato tramite sintesi vocale, in grado di: fornire istruzioni dinamiche e contestuali alle azioni dell'utente; correggere o anticipare errori durante l'esecuzione della procedura; adattare il livello di supporto in base all'esperienza dell'operatore. Un tale sistema, eventualmente supportato da tecniche di intelligenza artificiale, renderebbe l'esperienza più personalizzata e immersiva.
- **Report e analisi delle prestazioni:** L'introduzione di una funzione di registrazione e salvataggio dei dati di sessione — come tempi, errori e sequenze di azioni — in formato JSON o simile, permetterebbe di generare report automatici per la valutazione oggettiva delle performance. Questa estensione sarebbe utile sia per la ricerca didattica, sia per un monitoraggio aziendale delle competenze acquisite durante il training.
- **Tracking avanzato delle mani:** L'integrazione del hand tracking nativo dei visori Meta permetterebbe di sostituire o affiancare i controller, rendendo l'interazione più naturale e diretta. Questa soluzione migliorerebbe il realismo, specialmente nelle azioni di precisione o manipolazione fine, e aumenterebbe l'accessibilità dell'applicativo.

Un ulteriore sviluppo potrebbe prevedere l'impiego di dispositivi esterni dedicati, come il *Leap Motion Controller*, capace di rilevare con elevata precisione la posizione e l'articolazione delle dita tramite sensori infrarossi. L'integrazione di tale tecnologia consentirebbe un tracciamento più accurato anche in condizioni di occlusione parziale, migliorando l'affidabilità dell'interazione nelle situazioni in cui i controller risultano limitanti o ingombranti.

In prospettiva, il sistema potrebbe essere esteso mediante l'uso di guanti aptici, che oltre al tracciamento completo delle dita forniscono un feedback tattile o di resistenza meccanica durante il contatto con gli oggetti virtuali. Ciò permetterebbe di riprodurre la sensazione fisica della presa, del peso o della consistenza dei materiali, potenziando significativamente il senso di immersione e la memoria motoria associata alle procedure apprese.

Combinando hand tracking ottico e feedback aptico, il sistema di addestramento potrebbe evolversi verso un'interazione multimodale, in cui la componente visiva, tattile e cinestesica contribuisce congiuntamente a migliorare la qualità dell'esperienza e la precisione del gesto tecnico.

- **Diversificazione dei ruoli:** Un'evoluzione significativa consisterebbe nella definizione di ruoli differenziati, ad esempio *supervisore* e *operatore*, con compiti e interfacce specifiche. Ciò renderebbe possibile simulare scenari cooperativi reali, in cui un utente fornisce istruzioni o verifica l'operato di un altro, replicando dinamiche tipiche dell'ambiente industriale.
- **Sezione teorica e quiz interattivi:** Per completare l'esperienza formativa, si potrebbe introdurre una sezione teorica integrata, con quiz o esercizi interattivi relativi alla macchina, alle procedure operative e alle norme di sicurezza. Questa funzionalità renderebbe il sistema un ambiente di apprendimento completo, in grado di unire teoria e pratica in un'unica piattaforma immersiva.
- **Sistemi inclusivi e accessibilità dell'esperienza VR:** Un aspetto di crescente importanza nello sviluppo di applicazioni immersive riguarda l'inclusività e l'accessibilità dell'esperienza per utenti con differenti esigenze fisiche o percettive. Il sistema di training potrebbe essere ulteriormente migliorato mediante l'adozione di strategie di design inclusive, volte a garantire che l'interfaccia e le modalità di interazione risultino accessibili al più ampio numero possibile di persone.

Dal punto di vista visivo, si potrebbe implementare un sistema di personalizzazione dei colori e dei contrasti, consentendo di adattare la tonalità e l'intensità degli aiuti visivi (come evidenziazioni, indicatori o testi) in base alle necessità dell'utente. Ad esempio, la possibilità di selezionare palette cromatiche alternative per utenti daltonici o l'introduzione di indicatori a forma variabile ridurrebbero il rischio di ambiguità percettiva e migliorerebbero la leggibilità delle informazioni.

- **Avatar a corpo intero e rappresentazione realistica dell'utente:** Un ulteriore sviluppo di rilievo riguarda l'implementazione di un avatar a corpo intero, capace di rappresentare fedelmente la postura e i movimenti dell'utente all'interno dell'ambiente virtuale. Attualmente, il sistema utilizza una rappresentazione parziale del corpo, limitata alle mani e alla testa, come avviene nella maggior parte delle esperienze VR incentrate sull'interazione manuale. Tuttavia, la presenza di un avatar completo, sincronizzato con i movimenti reali dell'utente, può contribuire in modo significativo a migliorare il senso di presenza, la propriocezione e la consapevolezza spaziale all'interno della scena.

L'integrazione di un corpo virtuale completo permetterebbe inoltre di rafforzare la percezione sociale nelle esperienze collaborative, poiché la visibilità dei movimenti del torso e delle gambe favorisce una comunicazione non verbale più naturale tra gli utenti. L'adozione di tecniche di inverse kinematics (IK), supportate da modelli semplificati o da reti neurali preaddestrate, consentirebbe di stimare in tempo reale la posizione degli arti inferiori e del busto anche in assenza di sensori dedicati.

Tale implementazione potrebbe essere ulteriormente arricchita attraverso la personalizzazione dell'avatar, permettendo a ciascun utente di scegliere o importare il proprio modello 3D, garantendo così un livello maggiore di riconoscibilità e immedesimazione.

- **Evoluzione dell'architettura di rete:** Un'evoluzione naturale del sistema riguarda la transizione da un'architettura client-hosted a una client-server dedicato.

Nell'attuale configurazione, uno dei client assume il ruolo di host, gestendo la sincronizzazione degli stati e il controllo della sessione. Sebbene questa soluzione risulti più semplice da gestire in fase di sviluppo e adatta per scenari di test, essa comporta alcune limitazioni, come la dipendenza dalle risorse hardware dell'host e la potenziale instabilità della connessione nel caso di disconnessione del client ospitante. Il passaggio a un server dedicato permetterebbe di ottenere una maggiore scalabilità, stabilità e controllo sull'ambiente di simulazione. In un'architettura di questo tipo, tutte le logiche di rete, l'autorità sugli oggetti e la gestione dei flussi di comunicazione verrebbero demandate a un'istanza server indipendente, riducendo i problemi di latenza e garantendo un'esperienza più coerente per tutti gli utenti connessi.

Bibliografia

- [1] B. Xie, H. Liu, R. Alghofaili, Y. Zhang, Y. Jiang, F. D. Lobo, C. Li, W. Li, H. Huang, M. Akdere, C. Mousas, and L.-F. Yu, “A review on virtual reality skill training applications,” *Frontiers in Virtual Reality*, vol. 2, p. 641553, 2021.
- [2] W. contributors, “Quest 2 — wikipedia, the free encyclopedia.” https://en.wikipedia.org/wiki/Quest_2. Consultato il: 20-09-2025.
- [3] H. Xie, E. Tudoreanu, and W. Shi, “Development of a virtual reality safety-training system for construction workers,” in *Digital Library of Construction Informatics and Information Technology in Civil Engineering and Construction*, 2006. Available online.
- [4] G. Martin, S. Schatz, C. Bowers, C. E. Hughes, J. Fowlkes, and D. Nicholson, “Automatic scenario generation through procedural modeling for scenario-based training,” in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, (Los Angeles, CA), pp. 1949–1953, SAGE Publications Sage, 2009.
- [5] G. A. Martin and C. E. Hughes, “A scenario generation framework for automating instructional support in scenario-based training,” in *Proceedings of the 2010 Spring Simulation Multiconference*, (Orlando, FL), pp. 1–6, Society for Computer Simulation International, Apr. 2010.
- [6] F. Lin, L. Ye, V. G. Duffy, and C.-J. Su, “Developing virtual environments for industrial training,” *Information Sciences*, vol. 140, pp. 153–170, 2002.
- [7] P. Salmon, D. Jenkins, N. Stanton, and G. Walker, “Hierarchical task analysis vs. cognitive work analysis: Comparison of theory, methodology and contribution to system design,” *Theoretical Issues in Ergonomics Science*, vol. 11, pp. 504–531, 2010.
- [8] E. Farmer, J. Van Rooij, J. Riemersma, and P. Jorna, *Handbook of Simulator-Based Training*. United Kingdom: Routledge, 2017.
- [9] Y. Papelis and G. S. Watson, “Using event templates to accelerate scenario development in virtual training environments,” in *Proceedings of the 50th Computer Simulation Conference*, (Bordeaux, France), Society for Computer Simulation International, July 2018.
- [10] G. Bell, A. Parisi, and M. Pesce, “Virtual Reality Modeling Language (VRML) Version 1.0 Specification.” <https://www.web3d.org/x3d/content/examples/basic/Vrml97Specification/VRML1.specification.txt>, 1995. Versione del 26 maggio 1995.
- [11] Autodesk, “Autodesk 3ds max.” <https://www.autodesk.com/products/3ds-max/>. Consultato il: 2025-09-17.

- [12] Autodesk, “Autodesk maya.” <https://www.autodesk.com/products/maya/>. Consultato il: 2025-09-17.
- [13] Blender Foundation, “Blender.” <https://www.blender.org/>. Consultato il: 2025-09-17.
- [14] Ultraleap, “Leap motion controller.” <https://www.ultraleap.com/>. Consultato il: 2025-09-17.
- [15] Microsoft, “Microsoft kinect.” <https://developer.microsoft.com/en-us/windows/kinect/>. Consultato il: 17-09-2025.
- [16] C. Di Loreto, J.-R. Chardonnet, J. Ryard, and A. Rousseau, “Woah: A virtual reality work-at-height simulator,” in *IEEE Conference on Virtual Reality and 3D User Interfaces*, (Reutlingen, Germany), pp. 281–288, IEEE, Mar. 2018.
- [17] E. Z. Borba, M. Cabral, A. Montes, O. Belloc, and M. Zuffo, “Immersive and interactive procedure training simulator for high risk power line maintenance,” in *ACM SIGGRAPH 2016 VR Village*, (Anaheim, California), Association for Computing Machinery, July 2016.
- [18] R. Sacks, A. Perlman, and R. Barak, “Construction safety training using immersive virtual reality,” *Construction Management and Economics*, vol. 31, pp. 1005–1017, 2013.
- [19] M. H. Abidi, A. Al-Ahmari, A. Ahmad, W. Ameen, and H. Alkhalefah, “Assessment of virtual reality-based manufacturing assembly training system,” *The International Journal of Advanced Manufacturing Technology*, vol. 105, pp. 3743–3759, 2019.
- [20] J. Lacko, “Health safety training for industry in virtual reality,” in *2020 Cybernetics & Informatics (K&I)*, (Velké Karlovice, Czech Republic), pp. 1–5, IEEE, Jan. 2020.
- [21] E. Yildiz, C. Møller, M. Melo, and M. Bessa, “Designing collaborative and coordinated virtual reality training integrated with virtual and physical factories,” in *International Conference on Graphics and Interaction 2019 (ICGI 2019)*, (Faro, Portugal), IEEE, Nov. 2019.
- [22] N. van der Meer, V. van der Werf, W.-P. Brinkman, and M. Specht, “Virtual reality and collaborative learning: a systematic literature review,” *Frontiers in Virtual Reality*, vol. 4, p. 1159905, 2023.
- [23] Wikipedia contributors, “Oculus rift.” https://it.wikipedia.org/wiki/Oculus_Rift. Versione in italiano; consultato il 17 settembre 2025.
- [24] HTC Corporation and Valve Corporation, “Htc vive.” <https://www.vive.com/>. Consultato il: 2025-09-17.
- [25] F. D. Lorenzis, F. G. Praticcò, M. Repetto, E. Pons, and F. Lamberti, “Immersive virtual reality for procedural training: Comparing traditional and learning by teaching approaches,” *Computers in Industry*, vol. 144, 2023.
- [26] H. Söderlund, S. Zamola, J. Boström, *et al.*, “The creation of a multi-user virtual training environment for operator training in vr,” in *Advances in Transdisciplinary Engineering*, vol. 52, pp. 173–184, 2024.
- [27] T. J. Chuang and S. Smith, “A multi-user cross-platform hands-on virtual lab within the metaverse – the case of machining training,” *Virtual Reality*, vol. 28, no. 62, 2024.
- [28] Microsoft, “Microsoft hololens.” <https://www.microsoft.com/hololens>. Consultato il: 17-09-2025.

- [29] Unity Technologies, “Unity real-time development platform.” <https://unity.com>, 2025. Sito ufficiale.
- [30] U. Technologies, “Xr interaction toolkit documentation.” <https://docs.unity3d.com/Packages/com.unity.xr.interaction.toolkit@3.1/manual/index.html>, 2025. Consultato il 28 agosto 2025.
- [31] U. Technologies, “Openxr plugin documentation.” <https://docs.unity3d.com/Packages/com.unity.xr.openxr@1.13/manual/index.html>, 2025. Consultato il 29 agosto 2025.
- [32] U. Technologies, “Netcode for gameobjects — manual.” <https://docs.unity3d.com/Packages/com.unity.netcode.gameobjects@2.5/manual/index.html>. Package com.unity.netcode.gameobjects, v2.5. Consultato il: 20-09-2025.
- [33] U. Technologies, “Introduction to multiplayer networking in unity.” E-book (PDF), 2024. Unity Technologies.
- [34] U. Technologies, “Relay servers.” <https://docs.unity.com/ugs/en-us/manual/relay/manual/relay-servers>, 2025. Documentazione ufficiale Unity, accesso: 20-09-2025.
- [35] ParrelSync Contributors, “Parrelsync: Unity editor extension for multi-instance testing.” <https://github.com/VeriorPies/ParrelSync>, 2025. GitHub repository.
- [36] Microsoft Corporation, “Visual studio — integrated development environment.” <https://visualstudio.microsoft.com/>, 2025. Sito ufficiale.
- [37] W. contributors, “Meta quest pro — wikipedia, the free encyclopedia.” https://en.wikipedia.org/wiki/Meta_Quest_Pro. Consultato il: 20-09-2025.
- [38] U. Technologies, “com.unity.xr-content.xr-sim-environments.” <https://github.com/Unity-Technologies/com.unity.xr-content.xr-sim-environments>. Accesso: 20-09-2025, GitHub repository.
- [39] Quixel, “Fab warehouse.” <https://www.fab.com/listings/a3149fab-3906-4043-b6ee-3937b752a06c>. Consultato il: 20-09-2025.
- [40] B. s.r.o., “Blenderkit.” <https://www.blenderkit.com/>, 2025. Accesso: 20-09-2025, sito ufficiale del servizio di asset per Blender.
- [41] J. Brooke, “Sus: A “quick and dirty” usability scale,” in *Usability Evaluation in Industry* (P. W. Jordan, B. Thomas, I. L. McClelland, and B. Weerdmeester, eds.), (London), pp. 189–194, Taylor and Francis, 1996.
- [42] S. G. Hart and L. E. Staveland, “Development of nasa-tlx (task load index): Results of empirical and theoretical research,” *Advances in Psychology*, vol. 52, pp. 139–183, 1988.

Ringraziamenti

*A chiunque si sia fermato un attimo e mi abbia dedicato un momento.
E soprattutto a Mamma.*