Alma Mater Studiorum · Università di Bologna

SCUOLA DI INGEGNERIA E ARCHITETTURA
Corso di Laurea Magistrale in Ingegneria Informatica

B-R1ING MoCap: registrazione e riproduzione dei movimenti umani su avatar 3D in realtà aumentata

Tesi di Laurea
in
Computer Graphics

Relatore:
Prof.ssa
Serena Morigi

Candidato:
Simone Galvani

Sessione III
Anno Accademico 2021/2022

dedica

Abstract

Già da qualche anno si è stati introdotti alla possibilità di vivere in un mondo virtuale: basta indossare un paio di visori di realtà aumentata, virtuale e mista che riproducono nell'ambiente circostante oggetti che fisicamente non esistono. Negli ultimi mesi, inoltre, questa possibilità sta diventando sempre più concreta con l'introduzione, da parte dei colossi dell'informatica, del concetto di Metaverso: un universo parallelo completamente digitale dove sarà possibile svolgere ogni attività sociale.

L'obiettivo di questa tesi è quello di contribuire in piccola parte a questo enorme progetto, creando, tra utenti, una modalità di interazione virtuale ma che si basa su comportamenti del tutto reali. A questo proposito il titolo dell'elaborato è *"B-R1NG MoCap: registrazione e riproduzione dei movimenti umani su avatar 3D in realtà aumentata"*. Lo scopo del progetto è quello di permettere a una persona di registrare un video in cui c'è un soggetto in movimento, salvare i movimenti del soggetto in un pacchetto dati e infine riprodurlo su un *avatar* 3D che viene fatto agire in realtà aumentata. Il tutto farà parte di un'applicazione "social network" che permette l'interazione tra utenti nel modo precedentemente spiegato. Un utente può quindi registrare i movimenti umani e inviarli ad un altro utente che può riprodurre il messaggio in realtà aumentata tramite il suo smartphone.

Viene introdotto, così, un nuovo tipo di comunicazione digitale indiretta

passando dalla comunicazione scritta, ormai salda da decenni nei messaggi, alla comunicazione orale, introdotta da qualche anno tramite i messaggi vocali, alla comunicazione gestuale resa possibile dal lavoro in oggetto.

Le fasi principali del progetto sono state due: una in cui, dopo aver individuato la tecnica migliore, è stato effettuato il "motion capture", l'altra in cui il movimento registrato è stato trasformato in animazione per un soggetto 3D che viene visualizzata in realtà aumentata.

Indice

1 Specifiche di progetto **7**

1.1 B-R1NG® | Rising Connections 7

1.1.1 B-R1NG® MoCap 8

2 Introduzione **13**

2.1 Metaverso 14

2.2 Realtà aumentata, realtà virtuale, realtà mista 15

2.2.1 Aspetti psicologici 18

2.2.2 Perché usare la realtà virtuale 19

3 Motion Capture **22**

3.1 Tecniche di motion capture 23

3.1.1 Sistemi ottici 23

3.1.2 Sistemi non ottici 26

3.2 Applicazioni esistenti 27

4 MediaPipe **28**

4.1 MediaPipe Holistic 28

4.1.1 Modello dei *Landmark* 29

4.2 MediaPipe Pose 31

4.2.1 BlazePose 31

4.2.2 Modello di rilevamento posa 32
4.2.3 Modello dei *landmarks* della posa 32
4.2.4 Applicazioni 32

5 Fase 1: registrazione movimento 35
5.1 Unity 3D 35
5.2 MediaPipeUnity plugin 36
5.3 Soluzione Pose Tracking 37
5.4 Test cases fase 1 41

6 Fase 2: ricostruzione movimento 43
6.1 Tentativo 1 44
6.2 Tentativo 2 45
6.3 Soluzione 47
6.3.1 Rotazioni in Unity: Quaternioni 48
6.3.2 Ricostruzione del modello coi *landmarks* di riferimento 49
6.3.3 Script "AnimationRenderer" 51
6.4 Test cases fase 2 52

Funzionamento B-R1NG® MoCap 54

Conclusioni e Sviluppi Futuri 56

Bibliografia e Sitografia 59

Capitolo 1

Specifiche di progetto

Il progetto nasce come parte integrante di un progetto aziendale già avviato da anni e lanciato a febbraio 2022. Si tratta di B-R1NG®, social network innovativo che si propone di introdurre una nuova modalità di scambio di messaggi.

1.1 B-R1NG® | Rising Connections

B-R1NG® viene definito dai creatori come una nuova esperienza virtuale che permette di chattare con amici tramite ologrammi, creare contenuti 3D ed entrare nel mondo della realtà aumentata di amici e influencer. È un progetto che nasce dall'incontro tra il virtual hub Touchlabs e la web agency Spaghetti Digitali. Il suo obiettivo è quello di creare connessioni tra persone, utilizzando la cosiddetta Wearable Technology, la tecnologia indossabile.

Si tratta di un bracciale dotato di una tecnologia interna che gli permette di riconoscere l'account associato all'applicazione tramite un chip NFC.

Quindi, proprio come nei social network, è possibile registrarsi alla piattaforma creando il proprio account, personalizzare il proprio profilo, cercare

Figura 1.1: B-R1NG® | Rising Connections

altri utenti registrati e mettersi in contatto con loro facendo richiesta di "amicizia" e creando la propria rete di utenti. L'attività principale sarà produrre contenuti e inviarli ad altri utenti che potranno visualizzarli in tempo reale sul bracciale, tramite la realtà aumentata.

1.1.1 B-R1NG® MoCap

Questo progetto è stato intitolato B-R1NG® MoCap e il suo goal finale è quello di integrare il sistema di B-R1NG® nella versione 2.0.

L'obiettivo è quello di creare una funzionalità, all'interno dell'applicazione, per registrare i movimenti del corpo tramite la fotocamera dello smartphone e salvarli in un pacchetto dati. Una volta che questo pacchetto sia stato salvato si avrà la possibilità di inviarlo ad un altro utente, che potrà

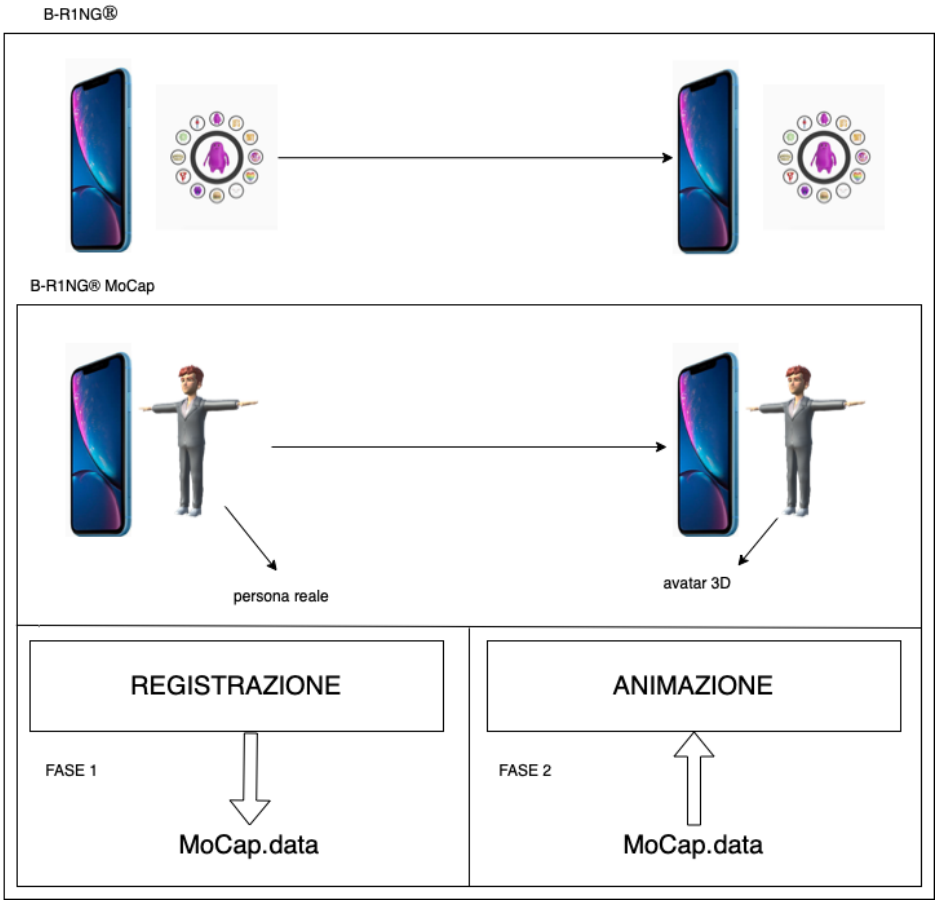


Figura 1.2: B-R1NG® MoCap

riprodurre i movimenti su un *avatar* 3D da lui stesso creato e vederlo in realtà aumentata, sempre grazie alla fotocamera dello smartphone.

È un progetto che esclude tutta la classe medio-bassa degli smartphone perché la realtà aumentata richiede molta potenza per essere supportata. Si ritiene però che, nel giro di pochi anni, tutte le case produttrici di smarphone e tablet avranno le risorse per fornire i chip adeguati anche ai dispositivi più economici. Ecco alcuni chip del sistema operativo Android che supportano questa tecnologia: `Snapdragon 865 Qualcomm SDM865`, `Snapdragon 765G Qualcomm SDM765`, `Snapdragon 720G Qualcomm SM7125`, `Snapdragon 460`

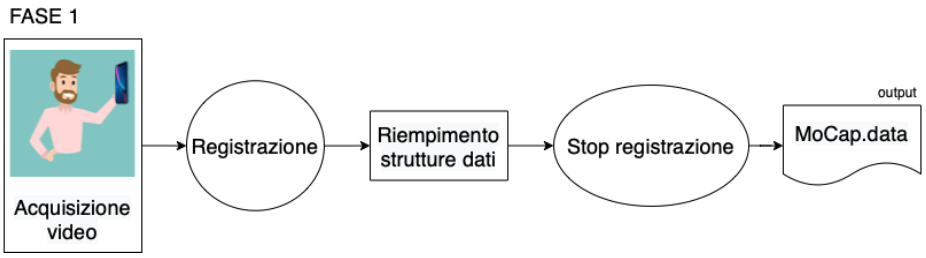


Figura 1.3: B-R1NG® MoCap Fase 1

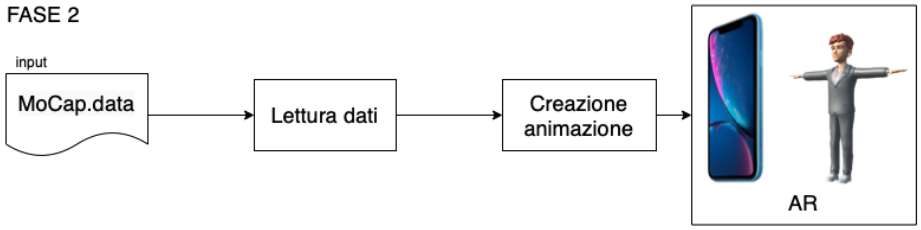


Figura 1.4: B-R1NG® MoCap Fase 2

`Qualcomm SM4250`, `Helio G85 MediaTek`, `Dimensity 1000 Plus MediaTek`, `Dimensity 800U MediaTek`, `SAMSUNG Exynos 7 Octa 7884B`, `SAMSUNG Exynos 990`.

Per quanto riguarda casa Apple il chip meno potente che supporta la tecnologia è `Apple A12 Bionic` presente sugli iPhone X.

Milestones Durante le prime settimane è stato studiato il Motion Capture e sono state analizzate le varie tecnologie esistenti per capire quale potesse essere la più adeguata al caso in oggetto.

Dopo il primo mese si stava già lavorando alla fase di registrazione dei movimenti grazie ad un plugin per Unity 3D.

L'attività più lunga è stata quella di ricostruzione dei movimenti sul modello 3D e di compilazione dell'applicazione mobile.

L'obiettivo prefissato all'inizio del progetto era quello di avere, entro la fine del tirocinio, un prototipo che simulasse l'animazione ottenuta. Il goal finale, ossia l'integrazione col progetto esistente, verrà poi raggiunto nei mesi successivi chiamando al lavoro anche il resto del team di sviluppo di B-R1NG®.

Esperienza lavorativa Il tirocinio presso Touchlabs è nato dal mio interesse per il mondo della realtà aumentata scaturito in conclusione del percorso magistrale in Ingegneria Informatica.

Durante l'estate del 2021, dopo aver superato l'esame di Computer Graphics che mi aveva introdotto a questo tema dell'informatica, ho contattato Andrea Bortolotti di Touchlabs per chiedere se ci fosse una possibilità di stage presso la sua azienda. Grazie alla sua disponibilità è nata questa collaborazione che mi ha permesso di entrare nel mondo del lavoro durante gli ultimi 6 mesi.

Touchlabs è un'azienda di Bologna specializzata nella creazione e nello sviluppo di siti web, app mobile (iOS, Android, Windows Phone), app per smartwatch (Apple Watch), strategie di web marketing, configuratori di prodotto 3D, VR, AR.

Costruito da Andrea, grazie alla sua passione per la tecnologia e ad un'instancabile voglia di innovazione, lo studio si trova in centro città ed è un vero e proprio museo tecnologico che mette in mostra dai più nuovi dispositivi di visione aumentata alle console storiche degli anni '90. È un ambiente che stimola la creatività e che in questi mesi mi ha aiutato ad aumentare anche la mia concezione della tecnologia.

Oltre ad un'opportunità di crescita personale questo tirocinio è stato anche l'occasione per scontrarmi con i miei limiti tecnici ed iniziare a percepire

il primo distacco tra mondo universitario e lavorativo. Per di più in questa esperienza ho potuto avvalermi dell'aiuto di ragazzi disponibili ed esperti nel settore che hanno reso possibile la mia crescita professionale non soltanto dal punto di vista tecnico, ma anche per l'attitudine al lavoro.

Capitolo 2

Introduzione

A ottobre 2021, Mark Zuckerberg, fondatore del social network Facebook, ha annunciato il cambio di identità del suo brand da "Facebook Ireland Limited" a "Meta Platforms Ireland Limited". Con il termine "Meta" ha voluto fare un chiaro riferimento al Metaverso, la nuova tecnologia in cui la presenza virtuale sarà equivalente e parallela a quella fisica, grazie all'ausilio di un dispositivo di realtà virtuale. Si potrà parlare di un "internet immersivo", che può essere vissuto e non solo guardato.

Questo annuncio è stato, per i più introdotti nel settore, un piccolo ulteriore passo di consapevolezza per quanto riguarda il mondo della realtà virtuale, mentre, per i più distanti dal mondo tecnologico, è stato la presentazione ufficiale di un nuovo concetto di realtà, supportata dalle tecnologie esistenti con la promessa di crescere con quelle future. Si tratta di una prospettiva che sembra ancora lontana nel tempo, eppure molti brand stanno già lavorando per creare contenuti e servizi per il Metaverso. Abbiamo già assistito ad alcuni eventi che dimostrano che questo processo può diventare qualcosa di concreto. Il 20 aprile del 2020, 12 milioni di persone hanno seguito in live streaming sulla piattaforma virtuale Fortnite la tappa

dell'"Astronomical Tour" del rapper statunitense Travis Scott. Il 17 maggio 2021 la nota casa di moda italiana Gucci ha aperto su Roblox, altra piattaforma virtuale, il suo "Gucci Garden", uno spazio virtuale in cui gli utenti hanno potuto esplorare sale a tema e assistere alla presentazione dell'edizione limitata di borse virtuali, che gli *avatar* hanno potuto indossare per 350.000 Robux (4115 dollari).

2.1 Metaverso

Il concetto di Metaverso, però, nasce molti anni fa quando, nel 1992, lo scrittore statunitense Neal Stephenson pubblicò il romanzo di fantascienza "Snow Crash". Le vicende del romanzo sono svolte all'interno di un parallelismo tra la realtà fisica e la realtà del Metaverso, cioè la realtà virtuale. Nello svolgimento dei fatti è possibile notare analogie tra i due mondi: una particolare droga fisica corrisponde ad un virus nella realtà virtuale, il bibliotecario corrisponde ad un software 3D per la ricerca di informazioni nella rete, il linguaggio primordiale di una civiltà antica può essere visto come un linguaggio di basso livello tipo quello binario, mentre la lingua parlata viene accostata ad un linguaggio di programmazione di alto livello. Il romanzo "Snow Crash" è stato fonte di ispirazione per la società americana Linden Lab nella creazione della piattaforma Second Life nell'anno 2003. Si tratta di un ambiente virtuale multiutente che offre una possibilità di azione in molteplici campi della creatività: intrattenimento, arte, formazione, musica, cinema, giochi di ruolo, architettura, impresa e molti altri. Gli utenti di questa realtà, chiamati *residenti*, possono accedere al mondo virtuale attraverso un *avatar* tridimensionale e, nel rispetto dei termini di servizio, possono fare qualunque cosa: esplorare il mondo virtuale, socializzare con altri utenti

tramite chat o voce, partecipare ad attività quali concerti, mostre, lezioni e moltissime altre attività proprio come nel mondo fisico. Queste caratteristiche rendono Second Life differente da un videogioco, in cui è possibile solo perseguire un obiettivo per concluderlo o avanzare nello stesso. Da ciò si capisce che, ad un certo punto, nasce la necessità di dare importanza ad una identità precisa, rappresentabile e personalizzabile, anche nel mondo virtuale dei social network. È chiaro, dunque, che ormai da 30 anni si sta cercando di costruire una realtà parallela a quella fisica dove poter esprimere se stessi. L'espressione scritta e digitale attraverso contenuti multimediali è stata possibile grazie all'avvento molto rapido dei social network. Negli ultimi anni si sta cercando di concretizzare sempre di più anche una forma di comunicazione gestuale attraverso la riproduzione di sé stessi sotto forma di *avatar*. È da qui che nasce il progetto descritto, cioè rendere possibile questo nuovo tipo di comunicazione digitale attraverso lo strumento che inevitabilmente ci troviamo tutti i giorni tra le mani, lo smartphone.

2.2 Realtà aumentata, realtà virtuale, realtà mista

Si procede ora alla descrizione dei concetti principali, le terminologie e i processi psicologici e cognitivi che entrano in gioco quando si parla di questa branca dell'informatica.

Realtà aumentata: l'ambiente reale è "aumentato" da oggetti virtuali (creati al computer). L'utente ha la possibilità di osservare e sentire suoni provenienti dall'ambiente reale circostante.

Realtà virtuale: è un ambiente in cui le persone sono completamente immerse e in cui possono interagire con un mondo creato interamente al

computer.

Realtà mista: l'utente può interagire con l'ambiente fisico e con quello virtuale. Gli oggetti digitali, a differenza della realtà aumentata, non si sovrappongono ma si integrano nell'ambiente e sono manipolabili da parte dell'utente.

I supporti a queste tecnologie sono visori "aperti", visori "chiusi" oppure smarphone e tablet.

Figura 2.1: Esempio realtà aumentata

"La realtà virtuale è una delle frontiere scientifiche, filosofiche e tecnologiche della nostra era. È un mezzo per creare una completa illusione di trovarsi in un posto diverso, che può essere un ambiente fantastico, alieno, con un corpo lontano da quello dell'essere umano. È al tempo stesso lo strumento più avanzato per comprendere ciò che un essere umano è in termini di cognizione e percezione. Mai un medium è stato così potente per la sua bellezza e vulnerabile all'inquietudine. La realtà virtuale ci metterà alla prova. Amplificherà il nostro carattere più di quanto abbiano fatto altri media. La realtà virtuale è tutto questo e molto di più."

Figura 2.2: Esempio realtà virtuale

[Jaron Lanier, 2017]

Questa definizione di Lanier, informatico statunitense che ha reso popolare la locuzione "virtual reality", fa capire che la realtà virtuale è molto più di un semplice mezzo di divertimento o svago. Essa permette di esaltare la parte creativa ed emozionale dell'essere umano, è uno straordinario strumento di formazione e informazione e, soprattutto, non isola il fruitore dal contesto, ma ne crea uno nuovo, parallelo, in cui è possibile entrare e uscire facilmente.

La realtà virtuale trasforma l'utente da osservatore di un'esperienza a **protagonista** della stessa, essendo in grado di modificare in tempo reale i contenuti della propria esperienza con le proprie scelte o azioni.

Può essere considerata un'interfaccia **esperienziale**, in cui la componente percettiva (visiva, tattile e uditiva) si fonde con l'interattività. L'interazione con questa realtà genera quello che viene definito "**senso di presenza**": la sensazione di essere "dentro" l'ambiente virtuale anche se fisicamente ci

Figura 2.3: Esempio realtà mista

troviamo in uno spazio differente. È proprio questo senso di presenza che il nostro progetto vuole andare a soddisfare rendendo concreta e condivisibile la comunicazione gestuale di cui ci serviamo inconsciamente tutti i giorni.

2.2.1 Aspetti psicologici

Entrando più approfonditamente nel concetto di "senso di presenza", esso è definibile come la risposta psicologica dell'utente all'interno di sistemi di realtà virtuale. È il grado soggettivo con cui una persona percepisce di essere fisicamente e mentalmente "presente" all'interno della simulazione. Bisogna tenere conto delle varie nature che può avere questo aspetto: si parla di presenza spaziale, presenza sociale e presenza personale.

Presenza spaziale: illusione di trovarsi all'interno del mondo virtuale ("being there").

Presenza sociale: stato psicologico in cui l'individuo percepisce sé stesso come esistente all'interno di un ambiente interpersonale. È la sensazione di stare con qualcun altro all'interno di un ambiente virtuale ("being with").

Presenza personale: modello mentale che gli utenti si creano di sé stessi all'interno del mondo virtuale, in riferimento alla percezione del proprio corpo, degli stati fisiologici, degli stati emotivi e dell'identità.

Il livello di presenza dipende da caratteristiche tecniche e, in particolar modo, dal livello di fedeltà sensoriale: maggiore sarà quest'ultimo, maggiore sarà anche il senso di presenza. Lo stesso accade in correlazione con i compiti proposti: più il compito è complesso, maggiori sono le possibilità che l'utente focalizzi l'attenzione, sperimentando un elevato senso di presenza; inoltre, maggiore sarà il coinvolgimento, maggiore sarà il senso di appartenenza. Alla base del concetto di presenza sta l'aspetto fisico che, nella realtà virtuale, è rappresentato da un *avatar*. L'*avatar* di una persona è la rappresentazione digitale del suo corpo umano in 3D. L'*embodiment* in un corpo virtuale è un fenomeno complesso, che può dare luogo a diverse illusioni soggettive, legate ad aspetti sia spaziali sia motori sia affettivi. All'interno di mondi virtuali condivisi con altri utenti, il comportamento delle persone è influenzato dalle caratteristiche del loro *avatar*: le persone si comportano, in una determinata situazione, come gli altri si aspetterebbero che una persona con quel corpo si comporti. La realtà virtuale, grazie alle sue caratteristiche uniche di immersione e interattività, è in grado di:

- Far vivere l'esperienza come se fosse reale $\rightarrow$ concetto di presenza.
- Far percepire il proprio *avatar* come il proprio corpo $\rightarrow$ concetto di *embodiment*.
- Suscitare risposte emotive più intense di altri dispositivi $\rightarrow$ medium affettivo.

2.2.2 Perché usare la realtà virtuale

È interessante utilizzare la realtà virtuale per diversi motivi:

- Offre un ambiente arricchito in grado di rispondere alle esigenze del mondo reale.

- Permette di creare delle vere e proprie esperienze digitali coinvolgenti e autorealizzanti che possono avere un impatto trasformativo sulla vita delle persone.

- Utilizza strumenti di intelligenza artificiale per proporre un livello di sfida coerente con le capacità cognitive degli individui (abilità), mantenendo allo stesso tempo, un elevato livello di presenza nell'ambiente col fine di supportare un'esperienza ottimale.

- Collega l'esperienza ottimale provata nell'ambiente virtuale ad un'altra esperita nella dimensione reale, in cui l'individuo è sfidato a mettere in atto le competenze apprese.

- Può essere considerata come un mezzo di apprendimento esperienziale.

"La realtà virtuale può essere considerata una tecnologia positiva e trasformativa grazie alla capacità di far vivere e sperimentare "mondi possibili", generando una nuova consapevolezza in grado di spingere il soggetto verso il cambiamento."

[Muratore et altri, 2019]

Figura 2.4: Cono di Dale

"Io sento e dimentico. Io vedo e ricordo. Io faccio e capisco."

[Confucio]

Capitolo 3

Motion Capture

La motion capture è una tecnica conosciuta prevalentemente dagli specialisti del settore, ma tutti in realtà ne abbiamo avuto esperienza da spettatori. Si tratta, infatti, del processo di analisi del movimento del corpo umano o di oggetti. Le informazioni raccolte attraverso tale processo possono essere utilizzate in svariati ambiti di ricerca. In quello sportivo si usa per studiare i movimenti degli atleti al fine di migliorare la loro tecnica e aumentare le performances, educando il corpo a disperdere meno energie possibili e prevenendo infortuni grazie allo studio approfondito del movimento di ossa e articolazioni. Le stesse applicazioni vengono sfruttate in ambito medico per studiare l'anatomia del corpo in movimento in condizioni non riproducibili all'interno di un laboratorio. Nell'ambito dello spettacolo questa tecnica viene usata per gli effetti speciali di molti film al fine di riprodurre movimenti realistici su soggetti disegnati al computer. Il primo esempio di quest'applicazione è stato il personaggio di Gollum ne "Il Signore degli Anelli". L'attore, in questo caso, indossava, per registrare la sua immagine stilizzata, un vestito ricoperto da sensori il quale ha permesso di riprodurre digitalmente i suoi movimenti sull'*avatar* del personaggio. Allo stesso modo il motion cap-

ture viene usato per creare personaggi dei videogiochi che riproducano con realismo e precisione i movimenti umani.

Questi sono solamente alcuni esempi di come viene utilizzata questa tecnica, ma gli ambiti sono molti di più e vanno dall'arte alla scienza, dallo sport all'intrattenimento digitale.

3.1 Tecniche di motion capture

Esistono varie tecniche per realizzare il tracciamento dei movimenti del corpo. Le prime, utilizzate alla fine degli anni 90, si basano su marcatori posizionati sui vestiti della persona in corrispondenza delle articolazioni, per identificare il movimento in base alle posizioni o agli angoli tra i marcatori.

Dall'inizio del ventunesimo secolo, grazie alla crescita tecnologica, sono stati sviluppati nuovi metodi: la maggior parte dei sistemi moderni può estrarre la forma del corpo in base a come è posizionato rispetto allo sfondo dell'immagine catturata. In seguito, tutti gli angoli delle articolazioni sono calcolati inserendo un modello matematico nella figura che rappresenta la forma del corpo.

3.1.1 Sistemi ottici

Marker

I sistemi ottici si servono di una telecamera puntata sul soggetto per analizzare i suoi movimenti. Questi sistemi possono essere di due tipi: quelli passivi sono caratterizzati da marcatori simili a palline argentate, che riflettono la luce, mentre quelli attivi utilizzano marcatori, ad esempio LEDs, che inviano segnali ad un sistema di acquisizione.

Figura 3.1: Esempio tracciamento con markers passivi

Nei sistemi ottici passivi i marcatori sono rivestiti con un materiale retroriflettente per riflettere la luce verso la telecamera. La soglia di cattura della telecamera può essere regolata in modo che solo i marcatori vengano campionati, ignorando la pelle e il tessuto. Tecniche di computer vision sono utilizzate per elaborare le immagini e riconoscere i marcatori tramite analisi sui pixels. Un oggetto con dei marcatori in posizioni note viene usato per calibrare le telecamere e misurare la distorsione della lente di ognuna, ottenendo le posizioni dei marcatori. Se due telecamere calibrate vedono un marcatore, si può ottenere un fix tridimensionale. Tipicamente un sistema è composto da 2 a 48 telecamere, ne esistono, però, di oltre 300 per cercare di ridurre lo scambio di marcatori. Le telecamere extra sono necessarie per la copertura completa intorno al soggetto di cattura e ai soggetti multipli. A differenza dei sistemi di marcatori attivi e dei sistemi magnetici, i sistemi passivi non richiedono all'utente di indossare fili o apparecchiature

elettroniche, ma centinaia di palline di gomma sono attaccate direttamente a un performer che indossa, su tutto il corpo, una tuta in spandex/lycra progettata appositamente per il motion capture.

Figura 3.2: Esempio tracciamento con markers attivi

I sistemi ottici attivi triangolano le posizioni illuminando molto rapidamente un LED alla volta o più LEDs con un software, per identificarli in base alle loro posizioni relative. Piuttosto che riflettere la luce generata esternamente, i marcatori stessi sono alimentati per emettere la propria luce. Grazie a leggi fisiche legate alla luce, questi sensori permettono di aumentare le distanze e il volume per la cattura e forniscono prestazioni migliori nel riconoscimento dei movimenti. È proprio utilizzando questo tipo di sensori che, durante le scene dei film, i registi possono vedere la performance degli attori in tempo reale e osservare i risultati sul personaggio in Computer Graphics guidato dal motion capture.

Markerless

Le tecniche emergenti e la ricerca nella computer vision stanno portando al rapido sviluppo dell'approccio markerless, ossia senza marcatori. Algoritmi speciali sono progettati per permettere al sistema di analizzare flussi multipli

di input ottici e identificare le forme umane. Reti neurali vengono addestrare per riconoscere la forma del corpo umano e assegnare un punto di riferimento ad ogni articolazione senza l'ausilio di sensori.

Grazie ai sistemi markerless il soggetto non deve indossare nessuna tuta speciale per il tracciamento; questo rende il tutto molto più economico e accessibile a chiunque. Per il progetto da noi realizzato ci siamo serviti di una libreria di soluzioni che sfrutta questi sistemi.

3.1.2 Sistemi non ottici

La tecnologia di cattura del movimento non ottica si basa su sensori inerziali miniaturizzati, modelli biomeccanici e algoritmi di fusione dei sensori. I dati di movimento dei sensori inerziali sono spesso trasmessi in modalità wireless a un computer, dove il movimento viene registrato o visualizzato. La maggior parte dei sistemi inerziali utilizza unità di misura inerziali (IMU), che contengono una combinazione di giroscopio, magnetometro e accelerometro, per misurare i tassi di rotazione. Queste rotazioni sono tradotte in uno scheletro nel software. Molto simile ai marcatori ottici, più sensori IMU sono presenti e più naturali sono i dati. Non sono necessarie telecamere esterne, emettitori o marcatori per i movimenti relativi, anche se sono necessari per dare la posizione assoluta dell'utente, se lo si desidera.

I sistemi di cattura inerziale del movimento registrano, in tempo reale, tutti i sei gradi di libertà del corpo umano. I vantaggi dell'uso dei sistemi inerziali includono l'utilizzo in un'ampia varietà di ambienti, nessuna risoluzione, portabilità e grandi aree di cattura. Gli svantaggi sono una minore accuratezza posizionale e la deriva posizionale che può aggravarsi nel tempo. Questi sistemi sono simili ai controller Wii, ma sono più sensibili e hanno una maggiore risoluzione e velocità di aggiornamento. Non essendo materia

del progetto ci si limita a questa descrizione sommaria dei sistemi non ottici al solo scopo di rendere nota la loro esistenza e accennare alla differenza con i precedenti.

3.2 Applicazioni esistenti

DeepMotion e RADiCAL sono servizi a pagamento che permettono di creare un'animazione 3D partendo da un video registrato. Per quanto riguarda i modelli 3D è possibile crearli direttamente dalla piattaforma online oppure caricare i propri modelli FBX o GLB personalizzati. Abbiamo individuato questi due software online, che offrono il servizio di motion capture e ricostruzione dell'animazione, poichè presentano somiglianze col progetto che si intende illustrare in questa tesi. La differenza consiste nell'elaborazione del movimento.

Per soddisfare i requisiti del progetto si cercherà di salvare un file dati contenente informazioni relative al movimento, senza necessità di caricare un video sul server. Infatti, i software precedentemente impiegano molto tempo ad analizzare il video caricato. Mentre, nel nostro progetto, ciò che verrà salvato sarà appunto un pacchetto dati molto leggero in modo da essere inviato ed elaborato in pochi secondi. Inoltre, avendo salvato i dati dei movimenti, sarà possibile animare diversi soggetti 3D e non soltanto un solo *avatar*, come invece fanno questi software che forniscono un'animazione legata ad un singolo modello.

Capitolo 4

MediaPipe

MediaPipe è una libreria software che realizza un sistema ottico senza marcatori e di cui noi ci siamo avvalsi.

Il tracciamento dal vivo della posa umana e dei punti di riferimento del viso e della mano sui dispositivi mobili può trovare varie applicazioni nella vita moderna: analisi del fitness e dello sport, controllo dei gesti e riconoscimento del linguaggio dei segni. In particolare, la stima della posa umana gioca un ruolo significativo in varie applicazioni in ambito sportivo e medico, come la quantificazione degli esercizi fisici (es: conteggio delle ripetizioni di piegamenti durante una sessione di allenamento) o il comportamento posturale di una persona. Combinare tutti i tracciamenti citati sopra in tempo reale in una soluzione semanticamente coerente è un problema particolarmente difficile che richiede l'inferenza simultanea di più reti neurali dipendenti.

4.1 MediaPipe Holistic

MediaPipe offre, per i vari tipi di tracciamento, soluzioni veloci e accurate, ma separate. All'interno di questa grande libreria di programmazione, in-

fatti, troviamo varie soluzioni che permettono di analizzare singolarmente il movimento di mani, testa e corpo.

È, tuttavia, presente una soluzione più elaborata, chiamata "Holistic", che integra i tre modelli, ognuno dei quali è ottimizzato per il suo particolare dominio. Si tratta di un modello molto complicato che modifica l'input in base alla parte del corpo da analizzare. Infatti, a causa delle diverse specializzazioni delle soluzioni, l'input di un componente non è adatto agli altri. Il modello di stima della posa, per esempio, prende come input un fotogramma video a risoluzione fissa bassa (256x256). Ma se si dovessero ritagliare le regioni della mano e del viso da quell'immagine per passarle ai rispettivi modelli, la stessa risoluzione sarebbe troppo bassa per un'articolazione accurata. Pertanto, MediaPipe Holistic è stato progettato come una pipeline a più stadi, che tratta le diverse regioni utilizzando una risoluzione dell'immagine appropriata a ciascuna di esse. In primo luogo, si stima la posa umana (in alto in figura 4.1) con il rilevatore di posa di BlazePose e il successivo modello di riferimento. Poi, usando i punti di riferimento di posa dedotti, si desumono tre regioni di interesse (ROI) per ogni mano (2x) e il viso e si impiega un modello ritagliato per migliorare la ROI. In seguito viene ritagliato il fotogramma di input a piena risoluzione in queste ROI e si applicano modelli specifici del viso e della mano per stimare i loro corrispondenti punti di riferimento. Infine, tutti i punti di riferimento vengono uniti con quelli del modello di posa per ottenerne più di 500.

4.1.1 Modello dei *Landmark*

I punti di riferimento vengono chiamati *landmark* e MediaPipe Holistic utilizza i modelli di *landmark* di posa, faccia e mano in MediaPipe Pose, MediaPipe Face Mesh e MediaPipe Hands rispettivamente per generare un totale di 543

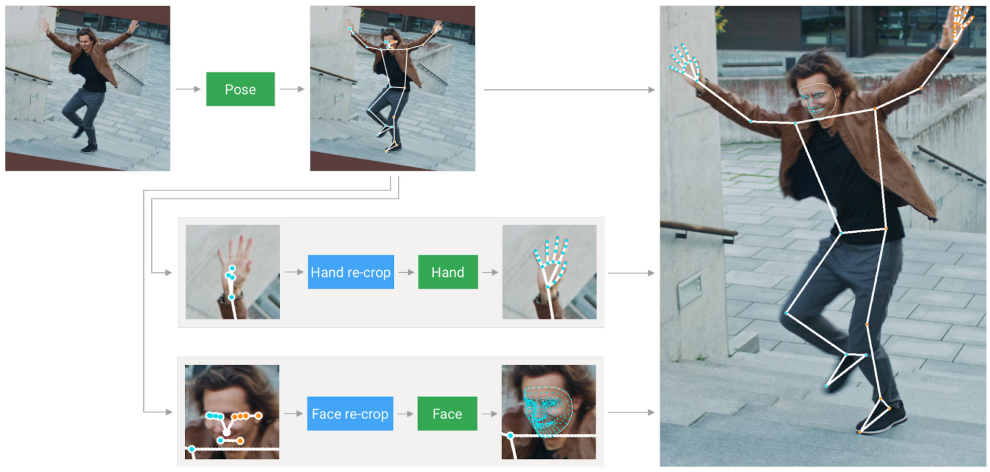


Figura 4.1: Holistic Pipeline

landmarks (33 *landmarks* di posa, 468 di faccia e 21 di mano per mano). Un singolo *landmark* è composto da:

- coordinate x e y
- coordinata z che rappresenta la profondità del punto di riferimento relativamente alla profondità del punto medio dei fianchi che è l'origine, e più piccolo è il valore più il punto di riferimento è vicino alla telecamera.
- visibilità: un valore in [0.0, 1.0] che indica la probabilità che il punto di riferimento sia visibile (presente e non occluso) nell'immagine.

Il modello MediaPipe Holistic sarebbe il modello ottimale per soddisfare tutti i requisiti del progetto, tuttavia si è scelto di utilizzare solamente il modello MediaPipe Pose per ricostruire inizialmente il movimento principale del corpo, trascurando le articolazioni delle mani e la mimica facciale. Negli sviluppi futuri il primo passo sarà quello di prendere in considerazione anche gli altri due modelli per realizzare una riproduzione sempre più fedele a quella reale.

4.2 MediaPipe Pose

MediaPipe Pose è una soluzione per il tracciamento della posa del corpo ad alta fedeltà, che deduce 33 punti di riferimento 3D utilizzando la ricerca BlazePose. Gli attuali approcci allo stato dell'arte si basano principalmente su potenti ambienti desktop per l'inferenza, mentre il metodo di MediaPipe raggiunge prestazioni in tempo reale sulla maggior parte dei moderni telefoni cellulari, desktop/laptop, in python e persino sul web.

4.2.1 BlazePose

Lo standard attuale per la posa del corpo umano è la topologia COCO, che consiste in 17 punti di riferimento per il torso, le braccia, le gambe e il viso. Tuttavia, i punti chiave COCO localizzano solo i punti della caviglia e del polso, mancando di informazioni di scala e orientamento per mani e piedi, che sono vitali per applicazioni pratiche come il fitness e la danza. L'inclusione di più punti chiave è cruciale per la successiva applicazione di modelli di stima della posa specifici del dominio, come quelli per mani, viso o piedi. Con BlazePose, si presenta una nuova topologia di 33 punti chiave del corpo umano. Questo ci permette di determinare la semantica del corpo dalla sola previsione della posa che è coerente con i modelli del viso e della mano.

BlazePose consiste in due modelli di apprendimento automatico: un *Detector* e un *Estimator*. Il *Detector* taglia la regione umana dall'immagine di input, mentre l'*Estimator* prende un'immagine con risoluzione 256x256 della persona rilevata come input e produce i punti chiave.

Il *Detector* ha un'architettura basata sul Single-Shot Detector (SSD). Data un'immagine di input produce un bounding box che racchiude il sog-

getto e fornisce un punteggio di fiducia per ogni punto di riferimento trovato. L'*Estimator* calcola i valori della coordinata z basati sui fianchi della persona. I *landmarks* sono situati tra i fianchi e la telecamera quando il valore è negativo, dietro i fianchi quando il valore è positivo.
Inoltre vengono forniti altri due importanti parametri: la presenza che restituisce la probabilità che i *landmarks* esistano nel quadro, la visibilità che restituisce la probabilità che i *landmarks* non siano occlusi da altri oggetti.

4.2.2 Modello di rilevamento posa

Il rilevatore prevede come punti chiave virtuali il riquadro contenente la faccia, ottenuto da MediaPipe Face Detection, e due ulteriori punti che descrivono saldamente il centro del corpo umano, la rotazione e la scala. L'ispirazione di questo modello viene dall'uomo vitruviano di Leonardo, pertanto si prevede il punto medio dei fianchi di una persona, il raggio di un cerchio che circoscrive l'intera persona, e l'angolo di inclinazione della linea che collega i punti medi delle spalle e dei fianchi.

4.2.3 Modello dei *landmarks* della posa

I *landmarks* vengono predetti secondo lo schema riportato nella figura numero 4.3.

4.2.4 Applicazioni

Sulla base della posa umana, possiamo costruire una varietà di applicazioni, come tracker di fitness o yoga. Come esempio, presentiamo contatori di squat e flessioni, che possono contare automaticamente le statistiche degli utenti, o verificare la qualità degli esercizi eseguiti.

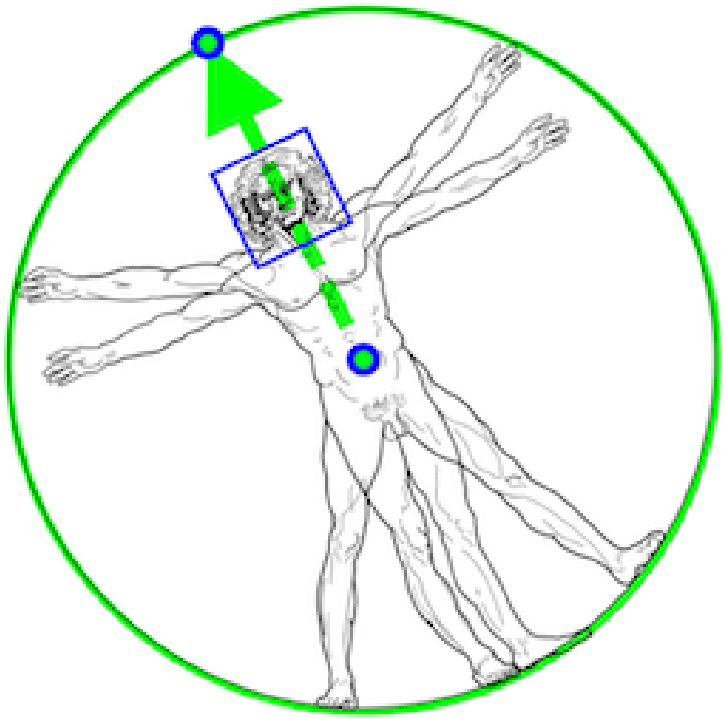
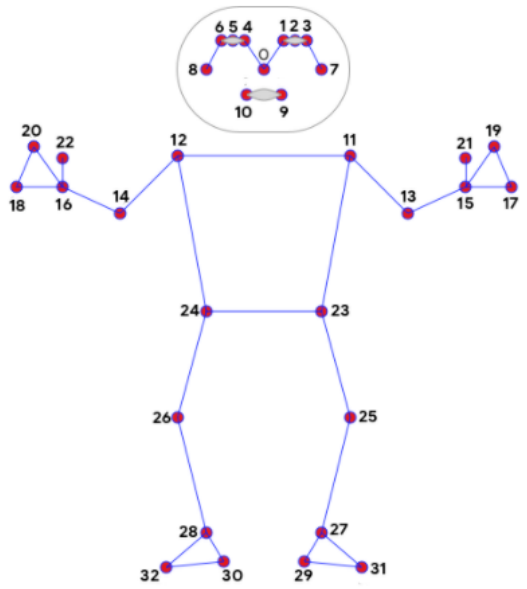


Figura 4.2: Modello Uomo Vitruviano



0. nose
1. left_eye_inner
2. left_eye
3. left_eye_outer
4. right_eye_inner
5. right_eye
6. right_eye_outer
7. left_ear
8. right_ear
9. mouth_left
10. mouth_right
11. left_shoulder
12. right_shoulder
13. left_elbow
14. right_elbow
15. left_wrist
16. right_wrist
17. left_pinky
18. right_pinky
19. left_index
20. right_index
21. left_thumb
22. right_thumb
23. left_hip
24. right_hip
25. left_knee
26. right_knee
27. left_ankle
28. right_ankle
29. left_heel
30. right_heel
31. left_foot_index
32. right_foot_index

Figura 4.3: *Landmarks* della Pose Tracking - Topologia Mediapipe

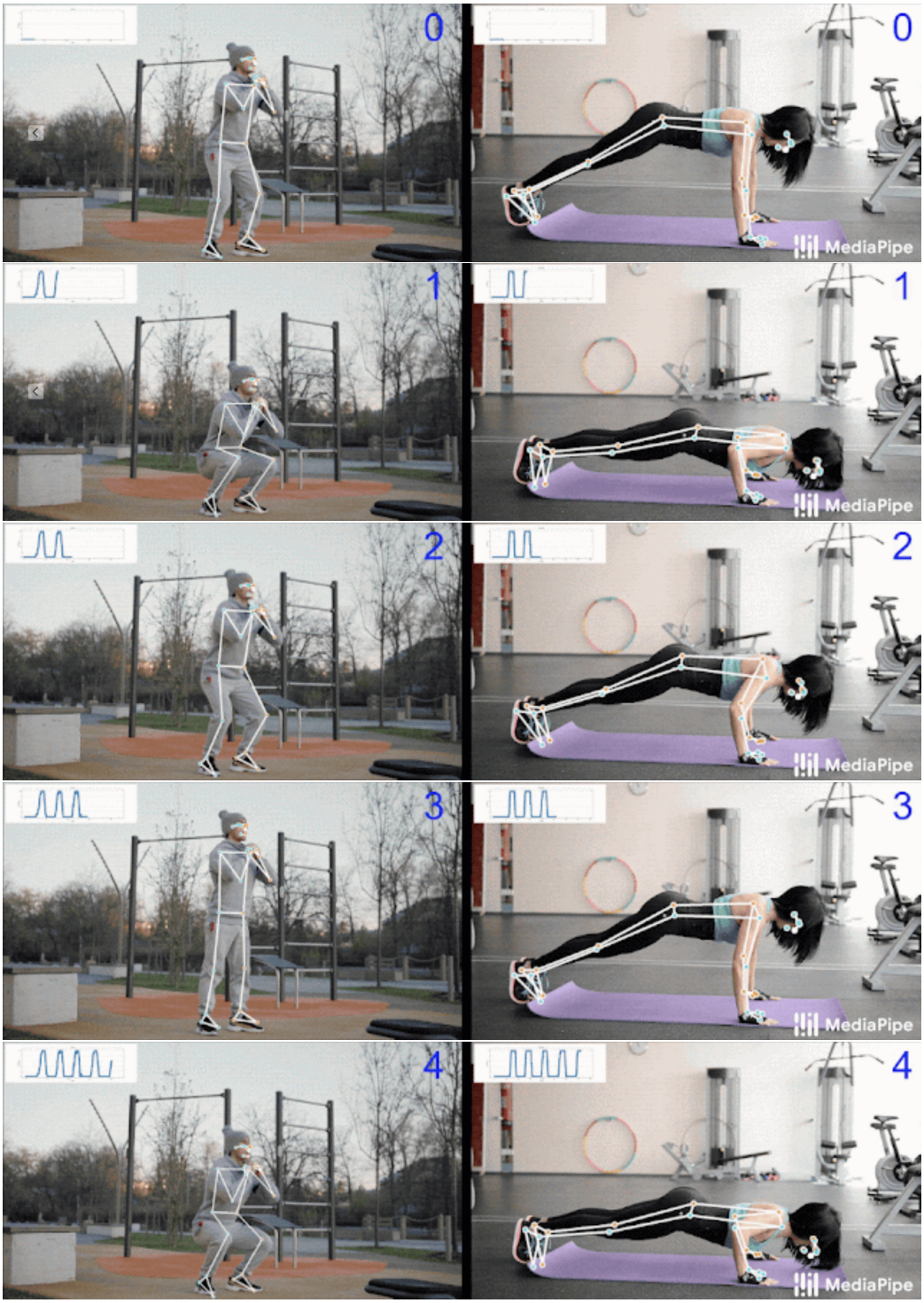


Figura 4.4: Esempio di applicazioni di conteggio squat o piegamenti

Capitolo 5

Fase 1: registrazione movimento

Il progetto ha avuto due fasi principali di svolgimento.

Nella prima fase ci si è focalizzati su quali fossero le modalità di registrazione dei movimenti e quale fosse la piattaforma più adatta per tale scopo. Si è deciso di utilizzare il software Unity, ideale per la gestione e modellazione di ambienti 3D, e di servirsi di un pacchetto di funzioni messo a disposizione da MediaPipe proprio per questo software.

5.1 Unity 3D

Unity 3D è un motore grafico multipiattaforma che consente lo sviluppo di ambienti tridimensionali interattivi in tempo reale.

Figura 5.1: Unity 3D

Questo ambiente di sviluppo può essere eseguito sia su Microsoft Windows sia su macOS sia su Linux, e produce applicazioni che possono essere

eseguite su smartphone, tablet, computer e la maggior parte delle console esistenti.

5.2 MediaPipeUnity plugin

MediaPipe fornisce un plugin per Unity tramite il quale sono state tradotte tutte le API della libreria da C++ a C# in modo da essere utilizzate anche sul motore grafico di cui si è appena parlato. All'interno del plugin è presente una scena per ogni soluzione offerta da MediaPipe. Le soluzioni offerte sono:

- *Box Tracking*: riconosce la presenza di una persona e disegna un riquadro che la contiene.

- *Face Detection*: disegna un riquadro contenente il volto e i keypoints rispettivamente di occhi, naso, bocca e orecchie.

- *Face Mesh*: riconosce più di 400 keypoints del volto e disegna i contorni di viso, naso, bocca, occhi e sopracciglia. Inoltre disegna un riquadro contenente il volto.

- *Hair Segmentation*: individua la zona dei capelli del soggetto e la evidenzia colorandola di blu.

- *Hand Tracking*: individua 21 keypoints per ognuna delle due mani colorandoli diversamente a seconda che siano della mano destra o della sinistra.

- *Holistic*: individua contemporaneamente i keypoints della faccia, del corpo e delle mani.

- *Instant Motion Tracking*: individua il sistema di riferimento della faccia.

- *Iris Tracking*: molto simile a *Face Mesh*.

- *Object Detection*: riconosce alcuni oggetti e ne disegna un riquadro che li contiene etichettandoli secondo la loro classificazione.

- *Pose Tracking*: individua i 33 keypoints del corpo.

5.3 Soluzione Pose Tracking

Come già accennato nel capitolo precedente la soluzione utilizzata per soddisfare i primi requisiti del progetto è stata MediaPipe Pose.

Per utilizzare il plugin di Unity di cui si è parlato sopra è necessario selezionare la scena principale "Start Scene". Viene presentata un'interfaccia in cui poter selezionare quale telecamera si vuole utilizzare, nel caso fossero più di una quelle installate sul computer, e in altro a destra c'è un bottone che apre l'elenco delle soluzioni offerte. Si sceglie dunque quella denominata "Pose Tracking".

Lo schermo mostra ciò che viene ripreso dalla telecamera e i vari *landmarks* sono posizionati nei corrispettivi punti del corpo e uniti secondo lo schema di figura 4.3. È necessario che il volto del soggetto sia individuato dalla telecamera affinché il sistema funzioni correttamente.

La figura 5.2 rappresenta l'ambiente Unity: il riquadro centrale mostra la scena in tempo reale e nella colonna di sinistra sono mostrati tutti gli elementi attivi che fanno parte della scena. Tra questi c'è un oggetto denominato "Point List Annotation" che contiene al suo interno l'insieme di tutti i *landmarks* di tracciamento mostrati nella figura. Selezionando uno di questi, che viene chiamato "Point Annotation(Clone)", possiamo vedere sulla destra della schermata tutte le sue caratteristiche tra cui la sua posizione che nell'esempio della figura è di (-0.590, 32.526, -5.317). In basso a destra

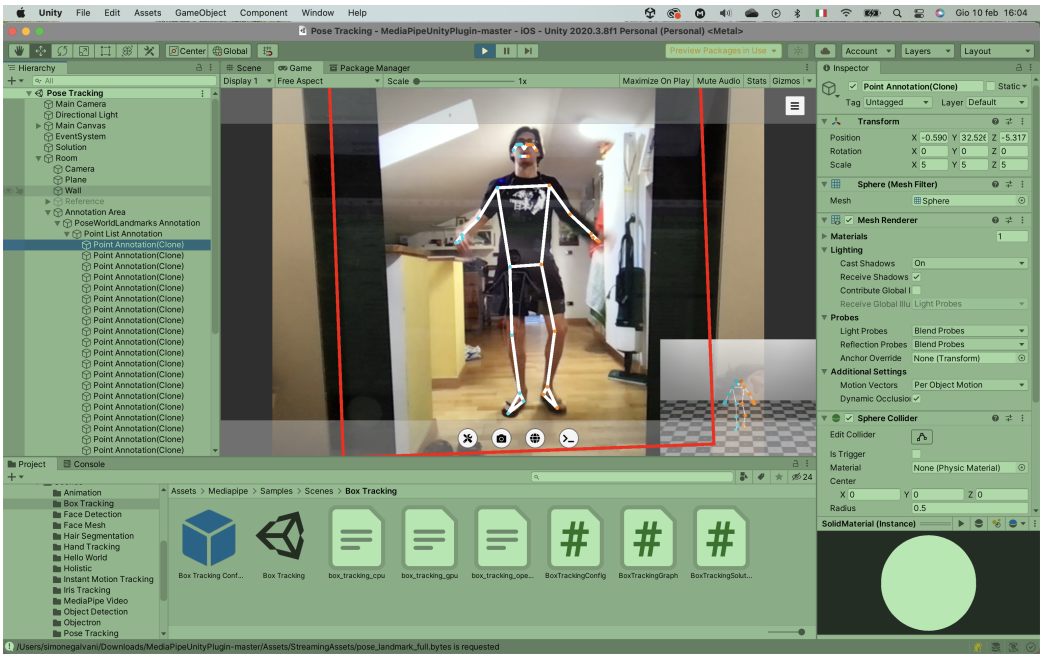


Figura 5.2: MediaPipe Pose

nella scena, in una zona denominata "Room", viene riprodotto il modello tracciato compreso di punti e segmenti che li uniscono utile per evidenziare ed analizzare lo schema in movimento. È da qui che si è partiti per andare ad analizzare le caratteristiche dei *landmarks*. Si è dunque ipotizzato di creare uno script che andasse a leggere la posizione di tutti i *landmarks* frame per frame.

Script "Recorder"

Per ordinare al meglio la raccolta dei dati è stata creata una struttura dati che contiene le **coordinate** di un *landmark*, il suo **identificatore** da 0 a 32, in riferimento allo schema di figura 4.3, una lettera per indicare la **parte del corpo** a cui fa parte e un intero che indica il **frame** di appartenenza. La lettera per indicare la parte del corpo è stata pensata perché in sviluppi

futuri sarà necessario registrare anche la posizione di *landmarks* appartenenti a viso e mani, dunque sarà necessario distinguerli.

Ad ogni frame tutti i *landmarks* vengono salvati in una lista di strutture tramite la funzione `Detection()` e scritti in un file di testo tramite la funzione `Serialization()` secondo lo schema:

numero frame, parte del corpo, identificatore, coordinate.

Per quanto riguarda il file dati è necessario che questo sia il più leggero possibile per rendere la comunicazione quasi istantanea. Si è riusciti ad ottenere, per una registrazione di 10 secondi, un file di 630 righe per un peso di 700KB. Il risultato è accettabile, basti pensare che tramite i servizi di messaggistica esistenti è già possibile inviare file pesanti più di qualche megabyte e il tempo stimato di ricezione è di qualche secondo.

Riguardo alla grafica di registrazione l'obiettivo è quello di creare un'interfaccia simile ai più famosi social networks in cui si preme il bottone di registrazione per avviare un video e lo si preme nuovamente per terminare la sua registrazione. Nelle prime versioni della demo di Unity questa operazione viene eseguita tenendo premuto il tasto "R" come è possibile vedere dal codice.

```
// Test.cs
using System.Collections.Generic;
using UnityEngine;
using System.IO;
using Mediapipe.Unity;

public class Test : MonoBehaviour
{
  [SerializeField] private PointListAnnotation _poseListAnnotation;
  private int _i = 0;
  private int _frameCounter = 1;
```

```
private List<DataStruct> _frameList;
private float _time;

[System.Serializable]
public class DataStruct
{
  public int f;
  public string p;
  public int iD;
  public Vector3 kp;
}

public void Update()
{
  _time += Time.deltaTime;
  if (Input.GetKey(KeyCode.R))
  {
    Debug.Log("Recording..." + _time);
    Detection();
    _frameCounter++;
  }
  else if (Input.GetKeyUp(KeyCode.R))
  {
    Debug.Log("...stop recording. File saved!");
    Serialization();
  }
}

private void Detection()
{
  while (_i < _poseListAnnotation.count)
  {
    AddFrameToList("b", _poseListAnnotation);
    _i++;
  }
}

private void Serialization()
{
  using (TextWriter tw = new StreamWriter("test2.txt"))
  {
```

```
      foreach (var s in _frameList)
      {
        if (s.iD == 32)
        {
          tw.Write(s.p + " " + s.iD + " " + s.kp.x + " " + s.kp.y + " " + s.kp.z + "\n");
        }
        else
        {
          tw.Write(s.p + " " + s.iD + " " + s.kp.x + " " + s.kp.y + " " + s.kp.z + " ,");
        }
      }
    }
  }

  private void AddFrameToList(string body_part, PointListAnnotation whichPointList)
  {
    var position = whichPointList[_i].GetComponent<Transform>().position;
    _frameList.Add(new DataStruct
    {
      f = _frameCounter,
      p = body_part,
      iD = _i,
      kp = position
    });
  }
}
```

5.4 Test cases fase 1

Nella prima fase di testing del software di registrazione è stato creato un modello a punti del corpo umano per verificare che la registrazione dei *landmarks* venisse effettuata correttamente.

Come previsto i punti assumono le posizione registrate dal sistema di MediaPiepe e il movimento è fedelmente riprodotto.

È stato confermato che la libreria MediaPipe registra i punti solamente se

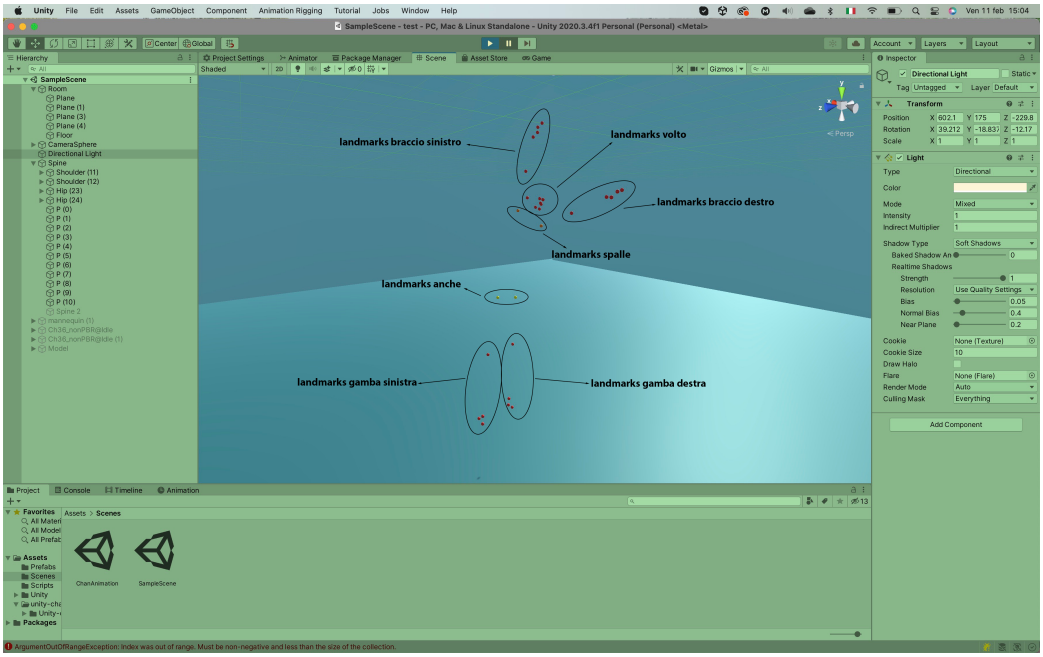


Figura 5.3: Modello a punti

viene individuato il volto e per una corretta riproduzione delle coordinate anche la zona delle anche deve essere individuata. Durante le varie registrazioni effettuate, infatti, nei casi in cui il volto non veniva ripreso il modello non dava alcun risultato e nessun *landmark* era visibile. Nella documentazione fornita dagli sviluppatori era accennato che, nel caso in cui il sistema avesse rilevato solamente il volto, i punti non visibili sarebbero stati predetti. In effetti dall'animazione del modello a punti è possibile analizzare movimenti consoni ad un corpo umano anche se viene ripreso il corpo parzialmente. I *landmarks* delle parti non individuate dalla telecamera assumono posizioni coerenti a tutto il resto del corpo.

Capitolo 6

Fase 2: ricostruzione movimento

La seconda fase del progetto è stata caratterizzata dalla sperimentazione su Unity dell'animazione. Si è cercato di lavorare seguendo il comportamento del modello a punti descritto alla fine del capitolo precedente.

Per assegnare le posizioni ai keypoints viene letto il file di testo precedentemente scritto dal sistema di registrazione. Ad ogni riga letta vengono aggiornate le liste che contengono i keypoints in base al numero identificativo. In questo modo alla fine della lettura si avranno 33 liste contenenti in ogni elemento la posizione di un determinato punto. Per individuare una posa del modello 3D basterà interrogare le liste allo stesso indice e si avranno le posizioni di tutti i punti per quella posa.

Mixamo Per quanto riguarda i modelli 3D utilizzati per applicare le animazioni si è ricorsi alle risorse di Mixamo. Mixamo è una libreria online di modelli 3D accessibile gratuitamente grazie ad un account Adobe. Ci sono modelli più o meno stilizzati a cui sono già state applicate animazioni di par-

tenza come balzi, camminate o anche più articolate come balletti di break dance o mosse di arti marziali. È possibile scaricare il modello in formato FBX, compatibile con varie piattaforme, ed è possibile scegliere se scaricare anche l'animazione associata o solamente il modello in T-pose.

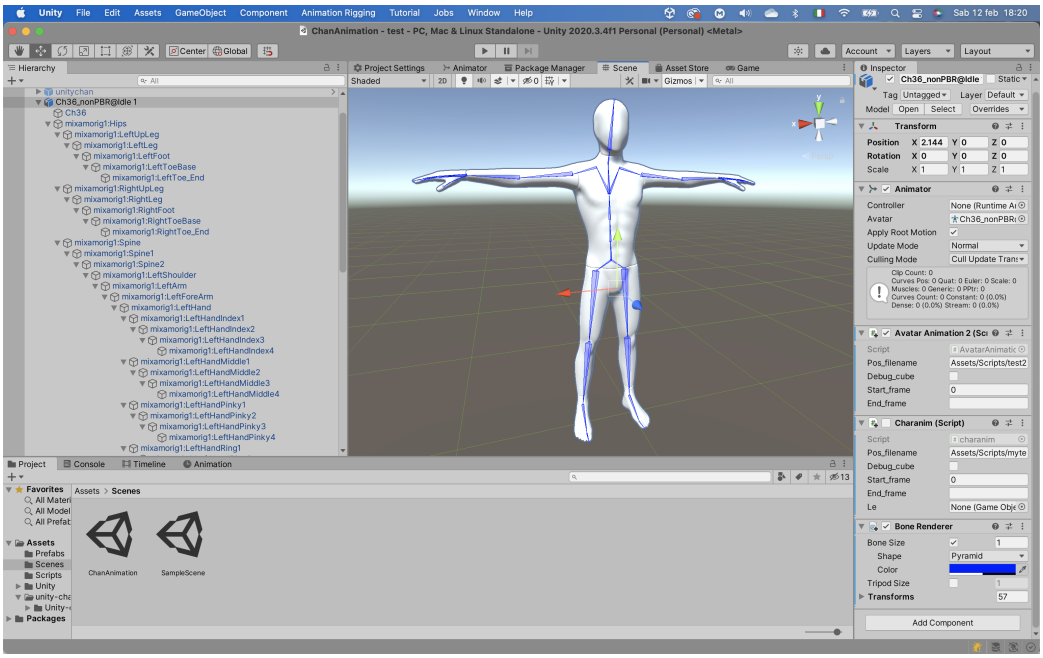


Figura 6.1: Modello in T-pose di Mixamo con gerarchia scheletro

Come si può vedere dalla schermata in figura 6.1 nella colonna di sinistra è mostrata la gerarchia delle ossa del modello.

6.1 Tentativo 1

La prima idea per realizzare l'animazione è stata quella di applicare le posizioni dei keypoints a ogni osso del modello secondo i punti corrispondenti. Il risultato ha prodotto un modello completamente spalmato sull'ambiente 3D evidenziando il primo problema di scala. Le coordinate registrate non erano

in scala col modello e inoltre quest'ultimo andava riposizionato su una sorta di baricentro tra tutti i keyponts.

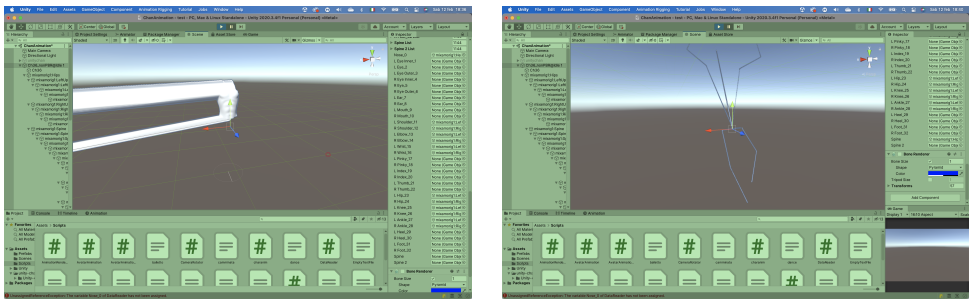


Figura 6.2: Modello deformato

Anche ricentrando il modello e provando a cambiare scala i risultati non sono migliorati tanto. Modificare la posizione delle ossa in questo modo ha portato alla deformazione del corpo. Non è possibile creare un'animazione soddisfacente andando a modificare direttamente le posizioni dello scheletro. Altra considerazione da fare è che il modello dei punti registrati non tiene in considerazione la rotazione delle ossa perché registra solamente traslazioni di punti nello spazio. Anche questo è sicuramente motivo di deformazione.

6.2 Tentativo 2

Un secondo approccio è stato quello di capire come funzionava la cinematica inversa del corpo umano.

La cinematica inversa è il processo di determinazione del posizionamento di una catena in base al posizionamento delle sole estremità.

È il processo che usa il cervello umano per muovere gli arti. Ci basti pensare che per prendere un oggetto con la mano il nostro cervello calcola la transizione che deve fare il polso per raggiungere l'oggetto. Le articolazioni di gomito e spalla si muoveranno di conseguenza. Se invece dovessimo calcolare

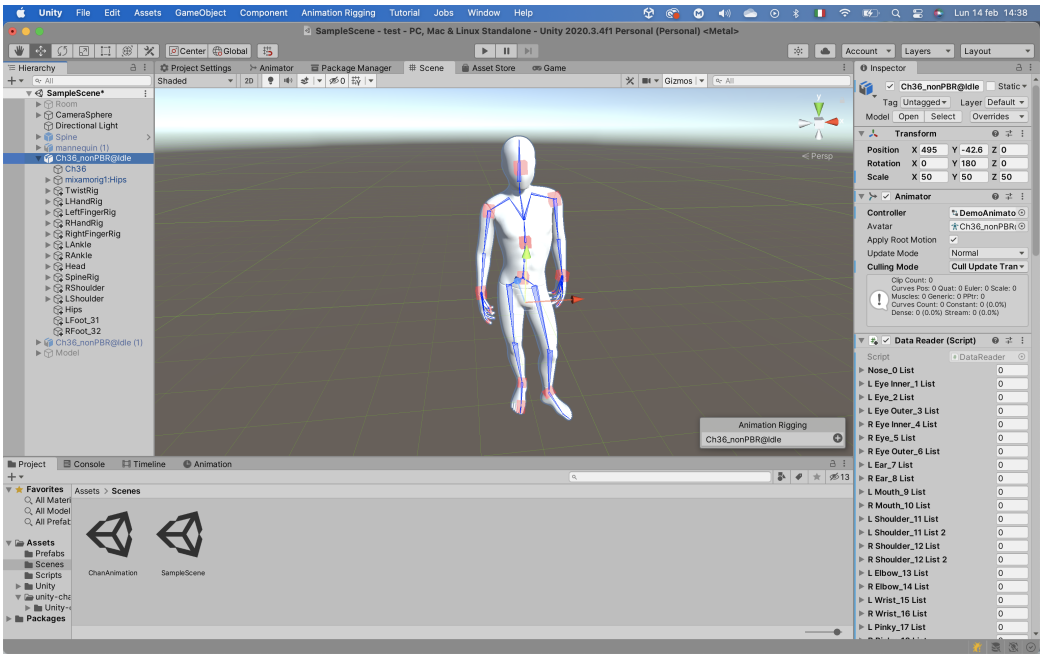


Figura 6.3: Modello con target di riferimento

la posizione di ogni articolazione dello scheletro parleremmo di cinematica diretta. Quest'ultima viene usata molto raramente nelle animazioni perché chiaramente richiede molto più tempo e inoltre si rischierebbe di creare movimenti innaturali.

Per usare la cinematica inversa in Unity si usa il package "Animation Rigging" che permette di gestire i vincoli tra le varie ossa. Ci sono vari vincoli che si possono dare allo scheletro per gestire i suoi movimenti modificando la posizione solamente di alcune ossa e calcolando anche le conseguenti rotazioni.

Inoltre grazie a questo package è possibile creare oggetti target che le ossa possono seguire evitando così di andare a operare direttamente sugli oggetti ossa. In questo modo, qualora una posizione fosse troppo distante dal reale raggiungimento dell'articolazione, quest'ultima eviterebbe di deformarsi fermandosi alla sua naturale estensione massima. In figura 6.3 si è cercato

di assegnare le posizioni solamente ad un numero limitato di ossa andando a modificare la posizione dei loro oggetti target. È possibile vedere che sono stati assegnati target al volto, ai polsi, alle spalle, alle caviglie ed ad alcune parti del tronco.

Sono stati fatti numerosi tentativi. Si è cercato di testare la maggior parte dei vincoli per fornire regole di movimento esatte, cercando anche di andare ad operare sulle rotazioni delle ossa. Anche la scelta degli oggetti target è stata messa molte volte in discussione in quanto i numerosi tentativi sono sempre stati fallimentari.

Questa fase è stata quella più problematica del progetto e che ha portato via più tempo. Si è stati in grado di vedere qualche possibilità di animazione, ma non si è mai riusciti a trovare la strada giusta per ottenere un risultato accettabile. La conclusione è stata che questo strumento che permette di utilizzare la cinematica inversa viene utilizzato per creare animazioni personalizzate di cui si conosce il risultato a priori, ossia andando a modificare le posizioni delle ossa a proprio piacimento.

6.3 Soluzione

La svolta nell'ottenimento di un'animazione accettabile è stata la ricerca di progetti simili e analisi della ricostruzione dell'animazione. A poco a poco si è scoperto che per risolvere il problema della cinematica inversa il modo migliore è quello di animare il modello 3D tramite le rotazioni. Bisogna quindi ad ogni frame assegnare una rotazione a tutte le ossa e non più una posizione.

Grazie a questa soluzione si va a risolvere anche il problema della scala, poichè il calcolo delle rotazioni non è influenzato dalla posizione delle coordi-

nate: si può dare una direzione ad un osso anche se i suoi punti di riferimento sono distanti.

6.3.1 Rotazioni in Unity: Quaternioni

Per operare con le rotazioni in Unity ci sono diversi modi: ci sono le proprietà eulerAngles e localEulerAngles, che sono Vector3 e rappresentano l'angolo di rotazione nei tre diversi assi X, Y e Z in gradi (quindi da 0 a 360).

Ruotare un oggetto impostando i suoi angoli di Eulero però può presentare il problema del blocco cardanico (gymbal lock) che si verifica quando due assi, che ruotando verso la stessa direzione, si allineano. Il blocco causa la perdita di un grado di libertà corrispondente all'asse bloccato. In breve, gli angoli di rotazione che vediamo nell'Inspector di Unity hanno un ordine di applicazione, quindi sommare rotazioni su diversi assi non produce i risultati sperati perché la rotazione su un asse dipende anche dalla rotazione degli altri due.

Per risolvere questo problema, in Unity vengono usati oggetti matematici, detti quaternioni, che incorporano numeri complessi.

Il quaternione è una sorta di estensione del numero complesso. È un oggetto matematico definito come

$$Q = q_0 + q_1 i + q_2 j + q_3 k.$$

Parte scalare $\rightarrow q_0$

Parte vettoriale $\rightarrow q_1 i + q_2 j + q_3 k$

In confronto agli angoli di Eulero presentano funzioni più semplici da comporre e confrontati con le matrici di rotazione i quaternioni sono più stabili numericamente e addirittura più efficienti.

Quasi mai si accede o si modificano i singoli componenti del quaternione (q_0, q_1, q_2, q_3); il più delle volte si prendono le rotazioni esistenti (ad esempio da Transform) e le si usa per costruire nuove rotazioni. Le funzioni Quaternion che si usano il 99% delle volte sono: Quaternion.LookRotation, Quaternion.Angle, Quaternion.Euler, Quaternion.FromToRotation.

- LookRotation(`Vector3 forward`, `Vector3 upwards = Vector3.up`): crea una rotazione in cui l'asse Z è allineato con il vettore `forward`, l'asse X è allineato con il prodotto vettoriale tra `forward` e `upwards`, e l'asse Y è allineato con il prodotto vettoriale tra Z e X.

- Angle(`Quaternion a, Quaternion b`): ritorna l'angolo in gradi tra le due rotazioni `a` e `b`.

- Euler(`float x, float y, float z`): ritorna una rotazione che ruota z gradi intorno all'asse Z, x gradi intorno all'asse X e y gradi intrno all'asse Y; applicate in ordine.

- FromToRotation(`Vector3 fromDirection, Vector3 toDirection`): crea una rotazione che ruota da `fromDirection` a `toDirection`.

6.3.2 Ricostruzione del modello coi *landmarks* di riferimento

Durante i vari tentativi di animazione era sorto anche il problema che il modello di registrazione di MediaPipe in figura 4.3 è diverso dalla fisionomia dello scheletro umano e di conseguenza dai modelli 3D forniti da qualsiasi software. Per operare sulle rotazioni delle ossa in modo preciso si è dovuti tornare alla topologia COCO con 17 *landmarks*, almeno per la parte principale dell'animazione. Pertanto ci si è dovuti attenere allo schema in figura

6.4. Senza fare riferimento alle zone di mani e piedi, l'unica differenza sta nella parte centrale del modello. Infatti la topologia COCO prevede un tronco centrale, molto più relistico in riferimento allo scheletro umano, mentre la topologia di MediaPipe è senza tronco e la zona del busto è descritta da un rettangolo.

Dunque per ricostruire il modello in figura è bastato calcolare i punti 0, 7, e 8 mancanti dal modello ottenuto in precedenza. Il calcolo non è stato eccessivamente complicato in quanto 0 è il punto medio tra 4 e 1 (24 e 23 nel modello MediaPipe) 8 è il punto medio tra 11 e 14 (12 e 11 nel modello MediaPipe) e 7 è il punto medio tra i due punti appena calcolati. L'unico dato che viene perduto in questo calcolo è la profondità (coordinata z) del punto 7 che dipenderà sempre da quella di 8 e 0; ma ai fini del risultato si è visto che il dato è irrilevante.

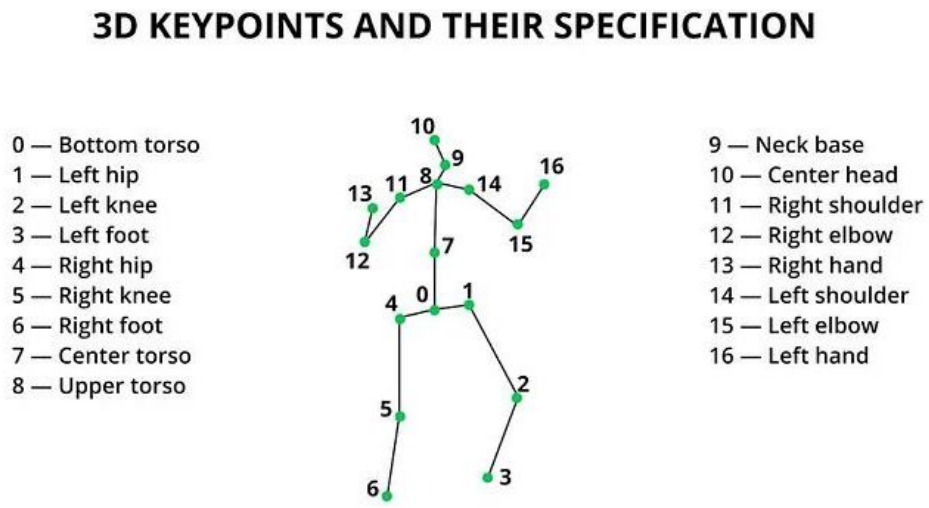


Figura 6.4: Landmarks della Pose Tracking - Topologia COCO

6.3.3 Script "AnimationRenderer"

Si procede ora alla descrizione del codice sviluppato per calcolare ad ogni frame la rotazione di ogni singolo osso utilizzando le funzioni fornite da Unity che sfruttano le proprietà dei quaternioni.

Per prima cosa è stata creata una struttura dati che descrivesse lo scheletro definita da un array di oggetti `Transform`. La struttura è stata poi riempita seguendo l'ordine in figura 6.4 grazie alla classe `HumanBodyBones` che permette di accedere alle proprietà dello scheletro in Unity.

```
bone_t = new Transform[bone_num];

bone_t[0] = anim.GetBoneTransform(HumanBodyBones.Hips);

bone_t[1] = anim.GetBoneTransform(HumanBodyBones.LeftUpperLeg);
bone_t[2] = anim.GetBoneTransform(HumanBodyBones.LeftLowerLeg);
bone_t[3] = anim.GetBoneTransform(HumanBodyBones.LeftFoot);
bone_t[4] = anim.GetBoneTransform(HumanBodyBones.RightUpperLeg);
bone_t[5] = anim.GetBoneTransform(HumanBodyBones.RightLowerLeg);
bone_t[6] = anim.GetBoneTransform(HumanBodyBones.RightFoot);
bone_t[7] = anim.GetBoneTransform(HumanBodyBones.Spine);
bone_t[8] = anim.GetBoneTransform(HumanBodyBones.Chest);
bone_t[9] = anim.GetBoneTransform(HumanBodyBones.Neck);
bone_t[10] = anim.GetBoneTransform(HumanBodyBones.Head);

bone_t[11] = anim.GetBoneTransform(HumanBodyBones.RightUpperArm);
bone_t[12] = anim.GetBoneTransform(HumanBodyBones.RightLowerArm);
bone_t[13] = anim.GetBoneTransform(HumanBodyBones.RightHand);
bone_t[14] = anim.GetBoneTransform(HumanBodyBones.LeftUpperArm);
bone_t[15] = anim.GetBoneTransform(HumanBodyBones.LeftLowerArm);
bone_t[16] = anim.GetBoneTransform(HumanBodyBones.LeftHand);
```

Lo 0 è il punto cardine dello scheletro, può essere considerato come la radice da cui si diramano tutte le altre ossa ed è quello che viene usato per determinare l'orientamento della parte bassa del corpo. Per fare ciò ad ogni frame viene calcolato il vettore ortogonale al triangolo di vertici 7, 4, 1. Lo

stesso viene fatto anche per il punto 8 usando come triangolo di riferimento quello di vertici 7, 14, 11. È importante distinguere le due rotazioni in quanto l'orientamento delle anche può non essere lo stesso delle spalle. Per quanto riguarda invece l'orientamento di tutte le altre ossa è stato preso come vettore direzione la risultante della differenza tra un punto e il suo successivo con riferimento al modello COCO.

A titolo di esempio, per calcolare il vettore direzionale del femore si calcola la differenza tra il punto 2 e il punto 1 ottenendo così il vettore congiungente i due punti con direzione 1-2. Grazie alla funzione LookRotation() di Unity si orienta il femore secondo la direzione ottenuta.

Pur avendo usato il modello COCO, i punti delle dita di mani e piedi sono stati utili per calcolare l'orientamento, almeno approssimativo, di questi ultimi che altrimenti non avrebbero avuto riferimenti.

6.4 Test cases fase 2

Nella seconda fase di test sono state effettuate varie registrazioni per verificare che l'animazione ottenuta venisse riprodotta il più verosimilmente possibile. Sono stati filmati soggetti che effettuavano alcuni tra i movimenti più comuni tra cui camminate, salti, sedute e alcuni movimenti un po' più complicati come passi di danza.

In generale le animazioni analizzate garantivano un risultato accettabile ma grazie a questa fase di test ci sì è accorti di due aspetti che avrebbero migliorato notevolmente il risultato ottenuto.

Primo Aspetto Per prima cosa si è visto che nell'animazione l'*avatar* muove correttamente tutte le ossa ma non si muove nello spazio, nel senso che

il suo punto chiave (punto 0 del modello COCO) rimane fisso. Dopo varie analisi si è notato che nella zona denominata "Room" (vedere figura 5.2), i *landmarks* dell'animazione non riproducono gli spostamenti del corpo ma sono fissi nel centro della "Room". Per questo si è deciso di cambiare il codice per prelevare i dati in questo modo: dalla zona centrale della schermata sarebbero stati prese le coordinate x e y per salvare gli spostamenti del corpo nello spazio, mentre dalla zona della "Room" sarebbe stata presa la coordinata z che nella zona precedente non era presente ma viene calcolata dal sistema.

Secondo Aspetto Inoltre si è notato che dal sistema di registrazione è impossibile ottenere il dato della profondità del punto 0 e, in effetti, il soggetto animato si muove solo lungo l'asse orizzontale senza riprodurre i movimenti in avanti e indietro. L'unico dato di profondità è quello calcolato da MediaPipe su ogni *landmark* rispetto al punto 0, ma quest'ultimo ha sempre la stessa coordinata z.

Si è così pensato di calcolare questa coordinata in relazione alle dimensioni del modello in registrazione. Più il soggetto è grande più è vicino alla telecamera e viceversa. Come riferimento alla grandezza del modello si è calcolata la distanza tra i punti 4 e 1. Quando questa distanza aumenta da un frame al suo successivo la coordinata z del punto 0 viene incrementata di un fattore di scala, in caso opposto viene decrementata. Il fattore di scala viene calcolato in base alle dimensioni dell'*avatar* nella scena.

Funzionamento B-R1NG® MoCap

Per dimostrare il funzionamento del progetto realizzato è stata creata un'applicazione demo che mostra le due fasi del progetto.

L'applicazione si apre con la scena di registrazione denominata "Pose tracking" ricavata dal plugin di Unity descritto nel capitolo 5.

Figura 6.5: Selezione della sorgente

Tramite il bottone in basso è possibile scegliere quale telecamera utilizzare, se quella frontale o quella principale. Rispetto alla scena offerta dal plugin di Unity è stato aggiunto un bottone di registrazione che, se cliccato, fa partire un countdown di 3 secondi prima di iniziare a registrare.

Nella zona in basso a destra è possibile vedere la riproduzione in tempo reale dei movimenti del corpo.

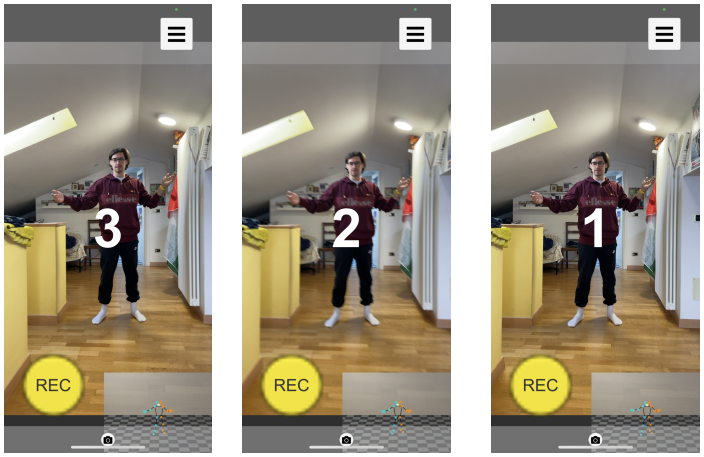


Figura 6.6: Scena "Pose Tracking"

Col bottone in alto a destra è possibile cambiare scena, avviando quella denominata "Play", per vedere l'animazione realizzata in realtà aumentata.

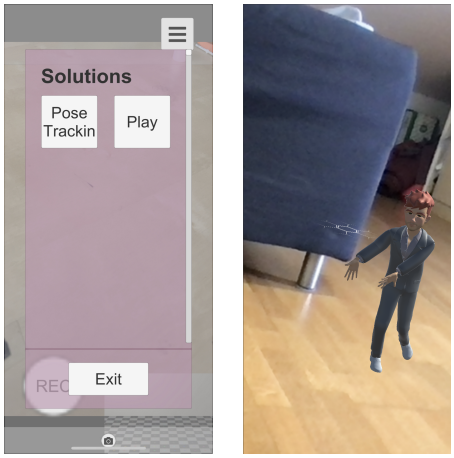


Figura 6.7: Schermata cambio scena e Scena "Play"

Conclusioni e sviluppi futuri

Questo lavoro si inserisce nel contesto dell'applicazione di realtà aumentata B-R1NG® per aumentarne il prestigio integrandola con il servizio che è stato denominato B-R1NG® MoCap. Si tratta di un sistema di registrazione dei movimenti del corpo e riproduzione degli stessi su *avatar* 3D in realtà aumentata.

In particolare è stato studiato il "Motion Capture" che è stato descritto nel capitolo 3 dove ne sono stati illustrati alcuni usi, descritte le varie tecniche per metterlo in pratica e accennati alcuni esempi che si servono di questa tecnica.

In seguito è stata selezionata la tecnica più adeguata, tra quelle esistenti, per registrare i movimenti umani. Sono stati catturati i dati sui movimenti grazie all'integrazione tra la libreria open source MediaPipe e il motore grafico Unity 3D. Le informazioni salvate sono state impacchettate all'interno di una struttura dati sotto forma di file di testo, facilmente leggibile e comodamente condivisibile grazie alla sua dimensione ridotta.

Poi, sempre grazie a Unity 3D, è stato realizzato un modello 3D in grado di leggere le coordinate precedentemente registrate e riprodurle sotto forma di animazione.

Il tutto è stato trasformato in applicazione mobile in grado di mostrare l'animazione ottenuta in realtà aumentata.

Le esigenze da cui nasce il progetto sono quelle di soddisfare il cosiddetto "senso di presenza", descritto nel capitolo 2, che viene reso possibile dall'*embodiment* nella realtà virtuale e aumentata. Si è cercato, dunque, di creare un nuovo modo di comunicare che andasse oltre al semplice scambio di messaggi scritti e vocali o tramite immagini. Si è resa possibile una nuova modalità di comunicazione, quella gestuale.

Durante lo sviluppo del progetto sono sorte idee di come implementare nuove funzionalità per rendere l'applicazione sempre più attraente.

Tra i primi obiettivi di sviluppi futuri c'è sicuramente quello di ampliare l'animazione dei movimenti anche alle mani e al viso per aumentare la gamma dei messaggi gestuali. Questo migliorerebbe di gran lunga il livello comunicativo, basti pensare a quanto sono forti gesti come l'occhiolino o un pollice alzato per dare consenso. Per entrambi gli obiettivi si partirà, come in questo progetto, dall'uso della libreria MediaPipe che, come già accennato nel capitolo 5, fornisce le soluzioni *Face Mesh* e *Hand Tracking* utili per registrare i *landmarks* di mani e volto. Inoltre per quanto riguarda la riproduzione della mimica facciale si potranno prendere come riferimento applicazioni già esistenti come quella fornita dal servizio di messaggistica di Apple denominato "Memoji".

Figura 6.8: "Memoji" di Apple

Altra funzionalità che potrebbe aumentare notevolmente il livello di pre-

stigio dell'applicazione è quella di poter registrare l'audio. In animazioni come balli o messaggi vocali sarebbe un modo per aumentare sempre di più l'immedesimazione del destinatario del messaggio. Come requisito fondamentale di questa funzionalità dovrà esserci la sincronizzazione tra audio e animazione, soprattutto tra voce e mimica facciale per realizzare una riproduzione perfetta.

Come ultima funzionalità per ampliare il progetto si è pensato che sarebbe utile creare un archivio di animazioni personalizzate, in modo che un utente possa recuperare un'animazione particolarmente gradita, salvarla con un nome all'interno del proprio database e poterla riprodurre o inviare all'occorrenza.

Bibliografia e Sitografia

[1] Federica Lovisato, Andrea Bortolotti, *Le nuove tecnologie*.

[2] https://it.wikipedia.org/wiki/Metaverso, *Metaverso*.

[3] https://it.wikipedia.org/wiki/Snow_Crash, *Snow Crash*.

[4] https://it.wikipedia.org/wiki/Second_Life, *Second Life*.

[5] https://en.wikipedia.org/wiki/Motion_capture#Applications, *Motion Capture*.

[6] https://mediapipe.dev, *MediaPipe*.

[7] https://github.com/homuler/MediaPipeUnityPlugin, *MediaPipeUnity-Plugin*.