UNIVERSITÀ DI PISA

Corso di Laurea in Informatica Umanistica

RELAZIONE

Progettazione e sviluppo di un modulo software per la creazione e manipolazione di animazioni facciali realistiche in Virtual Human conversazionali

Candidato: Camilla Tanca

Relatore: Marcello Carrozzino

ANNO ACCADEMICO 2014-2015

ESTRATTO:

Nel campo dell'Informatica Umanistica, come in molti altri settori legati alle tecnologie informatiche, gli ambienti virtuali immersivi (*Immersive Virtual Environments* - IVE) acquistano una posizione ogni giorno più rilevante. Gli IVE sono ad oggi già adoperati in molti campi di ricerca ed industria, come la *game industry*, i beni culturali, la riabilitazione, la medicina, l'addestramento e molti altri. Affinché un ambiente virtuale si possa dire veramente completo, tuttavia, è necessario integrarvi la presenza umana; questo è possibile tramite l'utilizzo di personaggi umani digitali, comunemente chiamati *Virtual Humans* (VH), e la gestione delle loro interazioni con l'utente.

La qualità visiva e performativa dei VH ha raggiunto negli anni dei livelli sempre più alti, sebbene manchi ancora un buon equilibrio tra realismo visivo (che negli ultimi anni ha raggiunto livelli foto realistici molto alti) e un modello comportamentale credibile. Nell'animazione dei VH è particolarmente importante il ruolo del viso, in quanto uno dei maggiori canali di comunicazione nell'essere umano. Tuttavia dato l'alto numero di muscoli facciali l'animazione delle espressioni del viso nei VH richiede particolare attenzione. È dunque estremamente importante affinare una procedura efficace per animare in modo realistico ed efficiente i tratti facciali del VH, sia al fine di realizzare espressioni facciali convincenti, sia per la più evidente funzione comunicativa: la parola.

Questa tesi si propone di sviluppare una metodologia per creare delle espressioni realistiche e una sincronizzazione credibile tra la mimica facciale del VH e la sua locuzione, che può essere espressa tramite la riproduzione di un file audio, di un flusso ricevuto in tempo reale o meccanismi di sintesi vocale, al fine di avere il massimo impatto di realismo.

Inizialmente verrà spiegato che cosa sono i VH (il loro ruolo negli ambienti virtuali, caratteristiche, procedure di creazione, etc.) e verrà condotta un'analisi sullo stato dell'arte raggiunto nel campo. In seguito sarà esposto il lavoro tecnico oggetto della tesi, dapprima si esporrà la procedura utilizzata per realizzare le animazioni facciali in un contesto immersivo, con particolare attenzione per la funzione di *lip sync*; successivamente si illustrerà l'architettura sviluppata per la loro gestione ed infine si presenteranno alcuni esperimenti e test nei quali vengono sfruttate le animazioni facciali per valutarne il loro utilizzo pratico. L'elaborato sarà concluso con summa del lavoro svolto e un indagine di eventuali sviluppi futuri.

ABSTRACT:

In Digital Humanities, as in many other Information and Communication Technologies (ICT) sectors, Immersive Virtual Environments (IVE) are assuming an increasing importance. IVEs are indeed exploited in several fields, both in research and in industry such as gaming industry, cultural heritage, rehabilitation, medicine, training and many others.

For a VE to be complete, however, it is necessary for it to integrate the human presence, achieved through digital human characters commonly called Virtual Humans(VH), and to manage their interaction with the user.

Performance and quality of VHs is constantly evolving , although a good balance between visual realism (which in the latest years has reached high photorealistic levels) and a convincing behavioural model is still lacking.

In the animation of VHs the role of the face is particularly important; in fact, it can be considered one of the most important sources of communication, and needs particular attention, as it contains the largest amount of body features.

For these reasons it is extremely important to establish a seamless methodology to effectively animate all the facial features of the VH, both in order to achieve convincing facial expressions and for an additional key communicative function: speech.

This thesis aims to develop a methodology to create a smooth lip sync between the VH and its speech, based either on audio files/streaming or on text-to-speech audio features, in order to achieve the best possible realistic impact.

In the thesis, first VH are introduced (their role in VEs, characteristics, the creation pipeline, etc.) and the state of the art of the field. Then the technical work object of the thesis will be presented, first exposing the methodology developed to animate the face of Virtual Agents in an immersive context, with a particular focus on the lip syncing, subsequently illustrating the implemented architecture, then presenting the results of user studies carried on to test the system and evaluate its usability and efficiency.

Last, conclusions are drawn, considering also possible future developments of the work done.

INDICE

INDICE DELLE FIGURE

A Poppy

1. INTRODUZIONE: VIRTUAL HUMAN E AMBIENTI VIRTUALI

La tecnologia usata per creare Ambienti Virtuali Immersivi (IVE) nell'ultima decade (Hale *et al*. 2014), ha subito una forte spinta evolutiva, trainata principalmente dalla *game industry* che domanda in modo sempre più urgente un maggiore realismo, un'interazione più naturale ed intuitiva e una maggiore usabilità. Per quanto riguarda la componente visuale l'obiettivo è di arrivare a colmare il divario con la ricchezza della visione umana in modo da creare un'immersione visiva maggiore. In questo senso lo stato dell'arte procede di pari passo con quello della computer grafica che dai poligoni piatti è arrivata velocemente a *smooth shading*, *texture maps* e *shaders* programmabili, i quali presentano una elevata flessibilità.

Ad oggi sono diversi i dispositivi che permettono di ricreare esperienze di immersione in ambienti virtuali sempre più realistiche e adoperabili per gli scopi più svariati, da attività ludiche o educative, a training in ambienti controllati, a sperimentazione e cura della mente umana. Un esempio sono i sistemi di proiezione CAVE[1]-like. I CAVE sono dei sistemi di realtà virtuale simili a vere e proprie stanze (quadrati o rettangoli tridimensionali o talvolta circolari), sviluppati nei primi anni '90 all'Università di Illinois, Chicago. In questi ambienti gli utenti sono circondati da immagini stereo proiettate sui muri e sulle pareti della stanza; inoltre vengono utilizzati dei *tracker* che rilevano la posizione e la rotazione della testa dell'utilizzatore in modo da adattare il suo punto di vista alla prospettiva mostrata. L'ampio angolo di vista creato dalle pareti che circondano l'utente realizza un'esperienza di realtà virtuale altamente immersiva. Inoltre l'utente ha la possibilità di interagire con l'ambiente tramite controllers, come ad esempio *joypad* o *wiimote*[2], che possono a loro volta essere tracciati. (Kageyama et al. 2013). Lo stato dell'arte attuale per sistemi CAVE-like è di 16

[1] Il termine CAVE é stato scelto in quanto acronimo ricorsivo di Automatic Virtual Environment. Viene inoltre specificato nel paper di battesimo la somiglianza concettuale con il mito della caverna platonica. (Cruz Neira et al., 1992)

[2] Il Nintendo's Wii Remote Control (meglio noto come Wiimote) é il controller della console Wii dotato di un acelerometro su tre assi e una camera a infrarossi ad alta precisione e velocità (Kouroupetroglou *et al*. 2012)

milioni di pixel per muro in cave a pareti multiple 10 x 10 ft; 100 milioni di pixel in schermi tessellati composti da 50 o più elementi (Cruz Neira et al. 1992, Hale et al. 2014).

Il mondo dei sistemi immersivi di realtà virtuale, nell'ultimo decennio, ha avuto uno sviluppo tecnologico estremamente rapido, aprendo le porte anche ad un'ottica di largo consumo. Infatti, ad esempio gli Head Mounted Displays (HMD), ovvero visori composti da doppio schermo (uno per ogni occhio) che adattano la propria visione stereoscopica al movimento della testa dell'utente, e che sono da sempre icona della realtà virtuale, hanno espanso enormemente la loro diffusione nel 2012 con l'Oculus Rift DK1[3]. Questo dispositivo offre diverse innovazioni rispetto ai tradizionali HMD: ad esempio un FoV[4] molto più ampio dei sistemi convenzionali e sensori integrati che registrano i cambiamenti della posizione della testa in tre dimensioni con una latenza migliore. Inoltre il dispositivo viene fornito ad un prezzo accessibile e dotato di un kit per sviluppatori che permette di implementare velocemente la sincronizzazione del display coi movimenti della testa. Per questo dispositivo attualmente si è arrivati a una risoluzione di 2,3 milioni di *pixels* per occhio (Kim 2015, Hale et al. 2014) . Dal 19 marzo 2014 è stata inoltre rilasciata la seconda versione dell'Oculus Development Kit (DK2) che presenta una risoluzione e un tempo di aggiornamento più elevati, una persistenza maggiore per evitare *motion blur* e un *tracking* posizionale per movimenti più precisi e minore latenza (Davis et al. 2015). Questo rapido sviluppo commerciale e tecnologico fa presagire un futuro nel quale l'uso degli ambienti virtuali immersivi sarà sempre più pervasivo nella vita quotidiana.

[3] http://www.oculusvr.com/

[4] Field Of View

Figura 1 Oculus rift DVK1 durante l'esperimento descritto in The Influence of an Interactive Context on Emotion Recognition from Faces in Immersive Virtual Environment (Faita, Vanni, Ruffaldi, Carrozzino, Tanca and Bergamasco , submitted)

Nella corsa allo sviluppo degli IVE, un campo che pare lievemente indietro è quello riguardante i Virtual Human (VH). Nonostante il loro uso sia essenziale, in questo genere di applicazioni, nei campi più svariati. Infatti, come vedremo più avanti, risulta ancora aperta la sfida tesa a valicare la *Uncanny Valley*, celebre definizione coniata da Mori nel 1970 che descrive il gap di credibilità percepita che esiste tra antropomorfismo e fotorealismo.

Il ruolo dei VH di aspetto realistico è molto importante nella creazione di ambienti virtuali realistici e ancor di più se immersivi. La presenza di personaggi dotati di sembianze umane è essenziale per completare un ambiente virtuale finalizzato alla simulazione di una scena di vita reale. La corrente evoluzionistica dei nuovi mezzi di interazione (tv interattiva, prodotti multimediali) rende sempre più necessario lo sviluppo di sistemi che possano integrare esseri umani simulati in giochi, prodotti multimediali e animazioni cinematografiche (Thalmann 2001). Le applicazioni esistenti e potenziali che coinvolgono attività umane negli ambienti virtuali sono infatti tra le più disparate. Per fare solo alcuni esempi possiamo citare:

- Insegnamento e training basati su simulazione
- Simulazione di ambienti di lavoro ergonomici
- Uso di pazienti per simulazioni mediche e chirurgiche

- Ortopedia, protesi e riabilitazione
- Psicoterapia virtuale
- Simulazioni architettoniche
- Narrazioni interattive

Inoltre la telepresenza[5] si va configurando come il futuro dei sistemi multimediali e permetterà ai partecipanti di condividere esperienze private e professionali, giochi, meeting ed eventi sociali. I VH hanno in questo contesto un ruolo chiave e la loro interazione realistica pone una sfida per la ricerca altamente interessante. L'uso di VH sempre più complessi e accurati accresce inoltre il livello di naturalezza nell'interazione con gli ambienti virtuali; una più naturale percezione di se stessi e degli altri utenti (ma anche dei personaggi autonomi) accresce il senso di copresenza e con esso la sensazione generale di esperienza virtuale condivisa.

Ad oggi lavoro sui VH si è concentrato principalmente verso la sfida di una interazione sempre più realistica tra agente virtuale (spesso *human-like*) e utente in carne e ossa, mirata a capire e ricreare le dinamiche del comportamento non verbale. Al fine di creare una sensazione di realismo vanno considerati molti aspetti, cercando di capire quali siano le caratteristiche che maggiormente ci fanno percepire un essere umano come tale e cercare di riprodurle in un VH, in modo da creare nell'utente una maggiore connessione ed immersione nell'ambiente virtuale.

Come fa notare nel suo libro, Hale (Hale et al. 2014), ci sono tre principali elementi sui quali è importante lavorare al fine di avere un Virtual Human che non sia più qualcosa di inanimato, ma si avvicini sempre più a un essere umano reale:

- Responsività: il VH dovrà reagire da una parte alle azioni compiute dall'utente, dall'altra all'ambiente virtuale nel quale vive.
- Credibilità: si dovrà fare attenzione alle caratteristiche comportamentali del VH in modo da renderle simili a quelle umane e trascinare in questo modo l'utente nell'interazione

[5] Per telepresenza si intende essenzialmente un qualche tipo di presenza di un individuo in un luogo fisicamente distante dalla propria posizione nello spazio (Schloerb 1995).

- Interpretabilità: bisognerà fare attenzione che comportamento e risposta del VH alle varie situazioni proposte durante la simulazione virtuale siano leggibili dall'utente, sia per quanto riguarda la risposta a determinati eventi, sia per quanto riguarda lo stato cognitivo ed emozionale che tali eventi scaturirebbero in una persona umana

Queste tre caratteristiche sono da tenere in considerazione, assieme alla ricerca di un realismo visuale, nella creazione di VH. La problematica è abbastanza pressante ed interessante da approfondire, in quanto i VH sono stati usati, lo sono e lo saranno in molte applicazioni di Realtà Virtuale sia come agenti di interazione con l'utente (Virtual Agent), sia come proiezione virtuale dell'utente stesso (Avatar).

Ritengo sia utile, prima di illustrare problematiche e caratteristiche riguardo il modello comportamentale dei VH, chiarire meglio la distinzione terminologica tra questi due termini. Chiuderà infine il capitolo una piccola introduzione all'architettura da me ideata.

1.1 Avatar e Virtual Agent

Quando si parla di VH, spesso non è facile comprendere con chiarezza la distinzione tra le due categorie di *Avatar* e *Virtual Agent*, inoltre a queste denominazioni se ne aggiungono molte altre (ad esempio *embodied conversational agent*, *virtual assistant*, *autonomous agent*) che contribuiscono a creare confusione terminologica. Molta della bibliografia in merito colloca il fulcro della questione nel modo in cui il personaggio virtuale viene controllato, ovvero la necessità o meno di un controllo umano, la sua *agentività* (Astrid *et al.* 2010).

> *An agent is defined as an acting entity, which includes the precondition of some kind of artificial intelligence that renders the control by a human dispensable (Erickson, 1997; Balakrishnan and Honavar, 2001). An avatar, by contrast, is a virtual representation of a human being, which is controlled completely by the human. Good examples of avatars are those in Second Life and World of Warcraft, where the*

user controls not only the verbal behavior, but also gestures and other body movements[6] *(Astrid et al. 2010).*

È di fondamentale importanza avere ben chiara questa distinzione, in quanto le reazioni degli utenti nel raffrontarsi ad un *avatar* o ad un *virtual agent* saranno estremamente differenti. Nel primo caso infatti si tratterà di analizzare un tipo di *Computer Mediated Communication* (CMC), mentre nel secondo caso potremo parlare più propriamente di *Human Computer Interaction* (HCI). É utile notare che in entrambi i casi si riscontra una tendenza ad agire socialmente, il che pare ovvio in una comunicazione tra esseri umani, seppure mediata dal computer, mentre lo è meno in caso di HCI. In questa dissertazione mi atterrò a questa distinzione, parlando più in generale di VH quando non è essenziale riferirsi ad una delle due sottocategorie.

1.1.1 Avatar

Avatar nella lingua inglese è ormai inteso come una personificazione, una rappresentazione corporale dell'utente (Mu *et al.* 2009). Tuttavia il sanscrito *Avatara* significa incarnazione ed anche nei testi Hindu viene frequentemente usato per indicare la personificazione del Divino: ad esempio Krishna è l'ottavo avatar (incarnazione) di Vishnu (the preserver), che molti induisti adorano come un Dio.

Non è del tutto chiaro come la parola sia arrivata ad assumere la sua accezione attuale; comunemente l'associazione viene fatta risalire al romanzo *cyberpunk* del 1992 di Neal Stephenson, *Snow Crash*, nel quale *users* di un sistema basato su computer sono in grado di entrare in mondi virtuali ed interagire tramite versioni virtuali di se stessi chiamate appunto Avatar (Ahn *et al.* 2012). É certo che il concetto di Avatar ed in generale la personificazione in ambienti virtuali è ed è stato sin dagli esordi oggetto di enorme stimolo

[6] Viene definito *agente* un soggetto che agisce; questo include un preconcetto di qualche genere sull'intelligenza artificiale che rende il controllo umano non essenziale (Erikson 1997; Balakrishman and Honavar, 2001). Un avatar, invece, é una rappresentazione virtuale di un essere umano, controllata totalmente dall'umano. Buoni esempi di avatar sono i personaggi di *Second Life* e *World of Worcraft,* nei quali gli utenti controllano non solo il comportamento verbale, ma anche i gesti e altri movimenti del corpo (Astrid *et al.* 2010)

per la fantasia umana. Lo dimostra il gran numero di opere letterarie, come *Neuromante* di William Gibson e *Cacciatore di androidi* di Philip Dick, ma anche cinematografiche come *Matrix* di Larry Wachowski ed il recentissimo *Avatar* di James Cameron, che richiama appunto il termine che ormai è entrato nel vocabolario odierno.

Si può definire Avatar qualsiasi forma di rappresentazione dell'utente, da una pedina del monopoli ad una rappresentazione umana realistica; i differenti livelli di realismo visuale e comportamentale influenzeranno la percezione della controparte umana con la quale si interagisce. Nella pratica, col passare degli anni, si è passati dalle prime forme usate in contesti di comunicazione testuale, dove una semplice immagine poteva essere un avatar, ad un aspetto sempre più dettagliato e realistico fino all'uso di veri e propri Virtual Human.

Oggi la parola Avatar non è più confinata al mondo dei *gamers* o agli appassionati di fantascienza, ma ha iniziato ad essere presente nelle più svariate applicazioni di uso quotidiano: da rappresentazioni virtuali realistiche (visualmente e nei movimenti) in console come Microsoft Kinect e Nintendo Wii, a personaggi in *online worlds* come quelli dei MMORPG[7]s, ad assistenti virtuali per acquisti online.

Sebbene gli avatar siano personalizzabili in forme sempre più fantasiose e non sempre corrispondenti al reale aspetto dell'utente, la ricerca scientifica si è concentrata soprattutto sugli Avatar di forma umana; l'uso degli stessi per usi scientifici offre infatti nuove prospettive e strumenti nei campi più disparati, ad esempio fornendo una mediazione tra validità ecologica e controllo sperimentale negli studi sociali. Nonostante il grande interesse del pubblico per gli Avatar, in ambito scientifico la disciplina si può considerare tuttavia ancora in fase nascente (Ahn et al 2012).

La ricerca sugli avatar si divide essenzialmente in due macro aree: la prima dedicata alla misura delle risposte psicofisiologiche e comportamentali degli utenti e la seconda mirata a riprodurre negli avatar emozioni, cognizioni e comportamenti umani. La ricerca da me condotta si va a collocare proprio in questa seconda area, con la prospettiva tuttavia di verificare la risposta da parte degli utenti a ciò che verrà realizzato.

[7] Massive(ly) Multiplayer Online Role-Playing Game

1.1.2 Virtual Agent

La riproduzione di comportamenti umani è di importanza ancora maggiore nella creazione di VH dotati di movimenti autonomi piuttosto che diretti da un essere umano, in quanto in questo caso si dovrà emulare in una certa misura anche il modo di pensare umano.

Il confronto tra i metodi elaborativi di un computer ed il ragionamento umano non è certo una novità, anzi si può dire che queste riflessioni abbiano accompagnato la ricerca in campo pratico fin dalla nascita dei primi computer stimolando nuovi paradigmi di ricerca e ispirando svariate opere di fantascienza che spesso tradiscono il timore che un giorno le macchine possano diventare superiori all'uomo e prendere il sopravvento sui limiti dei comuni mortali. Ne sono esempi: Pensiero profondo in *Guida Galattica per autostoppisti,* il *Tardis* nella serie *Doctor Who*, l'agente Smith, nel già citato *Matrix*, Hal 9000 in 2001*: Odissea nello spazio* di Stanley Kubrick e Skynet nella serie *Terminator*.

Era solo il 1950 quando Turing propose il suo celebre test, un approccio comportamentale per determinare l'esistenza di una *intelligenza artificiale*, ovvero della capacità di ragionamento di una macchina. Il Test di Turing si considera superato quando un agente artificiale viene scambiato per un essere umano in 5 minuti di conversazione dal 30% dei casi; non essendo infatti possibile indagare sul senso di coscienza interna di una macchina la migliore misura della suo essere senziente sarebbe la propria capacità di ingannare il giudizio umano nel reputarla tale. Sebbene la valutazione comparativa tra ragionamento umano e macchina non sia più considerata vitale per il progresso dei calcolatori, può essere invece molto interessante nel campo del 3D *gaming* e per le simulazioni di realtà virtuale. In questo campo la presenza umana realistica e autonoma aumenterebbe il dinamismo ed il coinvolgimento nei VE e creerebbe la possibilità di surrogati virtuali contribuendo alla personificazione della coscienza umana che potrà così moltiplicarsi nel tempo e nello spazio (Gilbert & Forney 2015).

Gilbert e Forney (2015) usano il test di Turing in una versione più indulgente ideata dallo stesso autore nel 1952 per mettere alla prova degli agenti virtuali reali. Secondo gli autori, agenti virtuali 3D avrebbero maggiori possibilità di superare il test, in quanto sfrutterebbero appieno la tendenza umana all'antropomorfizzazione ed a questo si aggiungerebbe il

vantaggio dei VE di rendere visibili caratteristiche e comportamenti umani. Tutto questo contribuirebbe a creare una maggiore illusione di trovarsi davanti ad una coscienza umana piuttosto che artificiale.

L'esperimento è stato condotto tramite una visita di un negozio virtuale su *Second Life* al seguito di una guida rappresentata da un VH di sesso femminile, senza specificare se si trattasse di un avatar o di un VA. Risultò che il 78% di coloro che si raffrontarono con un VA considerassero lo stesso guidato da un essere umano, mentre solo il 10% di chi si trovò di fronte ad un avatar scambiò lo stesso per un agente virtuale.

É importante notare in questo caso che i risultati, che supererebbero ampiamente il test di Turing, furono probabilmente incrementati dalla mancata consapevolezza della possibilità di trovarsi di fronte ad un'intelligenza artificiale e dal contesto della piattaforma di *Second Life* nella quale l'abitudine degli utenti è quella di raffrontarsi con avatar piuttosto che VA. Sebbene gli autori in questo caso parlino di *Artificial Intelligence* è necessario chiarire di cosa esattamente si tratti e di come, nel caso degli agenti virtuali ci siano diversi gradi autonomia del *virtual character*. É utile dunque, prima di proseguire, fare chiarezza terminologica.

Il termine AI viene fatto risalire alla conferenza di Darmouth del 1956, durante la quale McCarthy *et al.* (1955) individuarono 7 interrogativi chiave che saranno le basi di questa nuova disciplina (Rech 2004):

- *Automatic Computers*: se una macchina è in grado di fare un lavoro, allora anche un computer può essere programmato per farlo. La difficoltà non sarebbe stata tanto nelle capacità delle macchine dell'epoca di replicare la complessità del cervello umano ma nella nostra incapacità di programmarlo a tale scopo
- Come usare un computer al fine di utilizzare un linguaggio
- Reti Neurali: come un ipotetico set di neuroni può essere organizzato per elaborare concetti?
- Teoria della grandezza di un calcolo
- Auto-miglioramento
- Astrazione
- Casualità e Creatività

Pare più corretto allora, parlando di *Virtual Character* che reagiscono alle azioni dell'utente, con un certo grado di autonomia scriptata, di *Autonomous Virtual Humans*, ovvero personaggi virtuali dotati di un comportamento che ne determina i movimenti (Thalmann 2001). L'AVH dovrebbe percepire gli oggetti e gli altri VH tramite qualche tipo di sensore che può essere visuale, tattile o auditivo. In base agli impulsi percepiti il meccanismo comportamentale dell'avatar stabilirà la reazione adeguata. L'interazione può riguardare semplicemente il movimento nell'ambiente, l'interazione con lo stesso o la comunicazione con altri attori presenti, nell'ultimo caso si parlerà di avatar interattivo-percettivo.

Il canale maggiormente usato è solitamente quello visivo: il VH ottiene le informazioni relative a profondità e colore dei pixel rispetto al suo punto di vista e la propria posizione; in base a queste può localizzare gli oggetti visibili presenti nella scena 3d.

Per utilizzare le informazioni auditive, invece, è necessario creare un ambiente sonoro nel quale il VH possa accedere direttamente a posizione e semantica della fonte sonora di un evento acustico.

Sensori tattili possono ad esempio essere basati su multi sensori sferici collegati ad un oggetto, i quali verranno attivati dalla collisione con la figura (sistemi di questo genere possono essere integrati, ad esempio, in metodologie per il *grasping* automatico).

In generale gli *high level behaviour* usano impulsi sensoriali e conoscenze speciali: alcuni metodi di modellazione usano approcci automatizzati. Ogni personaggio ha uno stato interno che cambia ad ogni *step* temporale in accordo con le automazioni attive e gli input sensoriali.

Meccanismi di astrazione per simulare comportamenti intelligenti sono inoltre stati discussi nella letteratura riguardo AI e AA e sono stati introdotti molti metodi tesi alla creazione di Virtual Agent intelligenti e autonomi, sebbene la sfida al progresso sia ancora fondamentalmente aperta.

Per quanto questa prospettiva nel campo dei VH sia estremamente interessante, non è pretesa di questa tesi addentrarsi in un campo così ampio, l'obiettivo è in questo caso la realizzazione di un buon compromesso tra caratteristiche visuali, comportamentali e performance. Questo potrebbe costituire un primo passo verso meccanismi di intelligenza artificiale e *machine learning*.

1.2 UNCANNY VALLEY

Secondo Blascovich & McCall (2013) dal momento che non si hanno ancora Virtual Agent che possano essere completi dal punto di vista della AI (o non sono facilmente fruibili tecnologie abbastanza avanzate in un'ottica di largo consumo), per creare il senso di empatia e reazione sociale, che pare più naturale quando ci si trova davanti a VH animati da algoritmi, bisognerà sopperire lavorando sulla qualità impattale di modelli e animazioni. Questa non è, oltretutto, di minore importanza nella creazione di Avatar, in quanto una migliore trasposizione dei comportamenti umani aumenterà ulteriormente il realismo della simulazione.

Individuare però l'equilibrio tra caratteristiche visuali e comportamentali che ci fa percepire "credibile" un VH non è né semplice né banale; Tinwell (2015) nel suo libro *The Uncanny Valley in Games and Animation*, esplora il concetto di *Uncanny*, noto nella robotica e computer grafica (ma non solo) e fondamentale per comprendere quali siano gli elementi più importanti da tenere in considerazione nella creazione di avatar realistici.

Viene definita *uncanny* la sensazione sgradevole che si prova in presenza di oggetti dei quali non si riesce a capire la appartenenza al mondo reale o surreale, vivente o inanimato. Tale sensazione può crescere fino a provocare disgusto o orrore e permanere anche una volta accertata l'origine artificiosa dell'oggetto. L'effetto negativo è stato ampiamente sfruttato nella cinematografia dell'orrore, alla quale è difatti legato inestricabilmente.

Tinwell (2015), ripercorre le classiche teorie di Jentish e Freud per capire le origini del fenomeno in questione e cita come esempio di tale sensazione la leggenda di *Sandman*, favola tradizionale raccontata ai bambini nei quali un uomo senza faccia getterebbe sabbia negli occhi dei bambini che non si comportano bene al fine di prendere i loro occhi e adoperarli per nutrire i suoi figli, raffigurati come mostri per metà bambini e per metà volatili; in questo racconto la sensazione sgradevole è suscitata dalla natura parzialmente umana e parzialmente sovrannaturale dei personaggi.

Jentish (1906) evidenzia come questa sensazione si verifichi anche in presenza di persone con determinati disturbi mentali o fisici come, ad esempio, le crisi epilettiche nelle quali il comportamento del corpo differisce da quello ordinario.

Alle riflessioni di Jentish seguiranno nel 1919 gli studi di Freud sull'*uncanny*, il quale descriverà tale sensazione come quella che si prova ogni qualvolta ci si imbatta in qualcosa di orrorifico o aberrante e cercherà di indagarne le origini. Per fare ciò cercherà una descrizione a partire dal termine opposto, lo scozzese *canny*[8], ovvero familiare; sarebbe allora *uncanny* tutto ciò che risulta inusuale o non familiare. Questa descrizione non basta però a descrivere la sensazione di disagio che racchiude il termine; in seguito dunque il concetto verrà esteso a partire dal significato più ampio del termine scozzese e definito come la sensazione che si prova quando si viene a conoscenza di qualcosa che sarebbe dovuto rimanere nascosto o segreto.

Dalle ricerche di Jentish e Freud il concetto di *uncanny* è stato associato a un ampio spettro di argomenti, come ad esempio automazione, letteratura, linguaggio, horror, psicologia, estetica, terapia, sogni, manifestazioni di pazzia e memento mortis (Tinwell 2015). Nella seconda metà del ventesimo secolo lo stesso concetto è stato usato ampiamente per descrivere la sensazione che si prova a riguardo del design di androidi umanoidi nel campo della robotica.

L'uso di questo termine fu introdotto da Mori nel 1970 col saggio *The Uncanny Valley* e gode tutt'ora di notevole rilievo. Mori mostrò nel suo saggio un forte scetticismo nei confronti della tendenza, diffusa negli anni di redazione del saggio, verso la creazione di androidi umanoidi il più possibile realistici, atta a migliorarne il rapporto con gli utenti che avevano poco rapporto e affinità con i macchinari di tipo industriale. Mori predisse ed in seguito confutò che i fallimenti di tali androidi nel replicare il comportamento umano sarebbero stati percepiti maggiormente rispetto a robot senza pretese di somiglianza con l'essere umano e questo avrebbe potuto portare fino ad un senso di repulsione.

[8] canny |'kani| - adjective (cannier, canniest) - ORIGIN late 16th cent. (originally Scots): from can (in the obsolete sense 'know') + -y meanings: 1) having or showing shrewdness and good judgement, especially in money or business matters: canny investors will switch banks if they think they are getting a raw deal. 2) N. English & Scottish pleasant; nice: she's a canny lass.

Per raffigurare la sua teoria Mori disegnò un grafico che mostrava come nell'ascesa verso la somiglianza la percezione di familiarità si verifichi un brusco picco qualora ci sia una somiglianza realistica con la figura umana ma incompleta per qualche motivo.

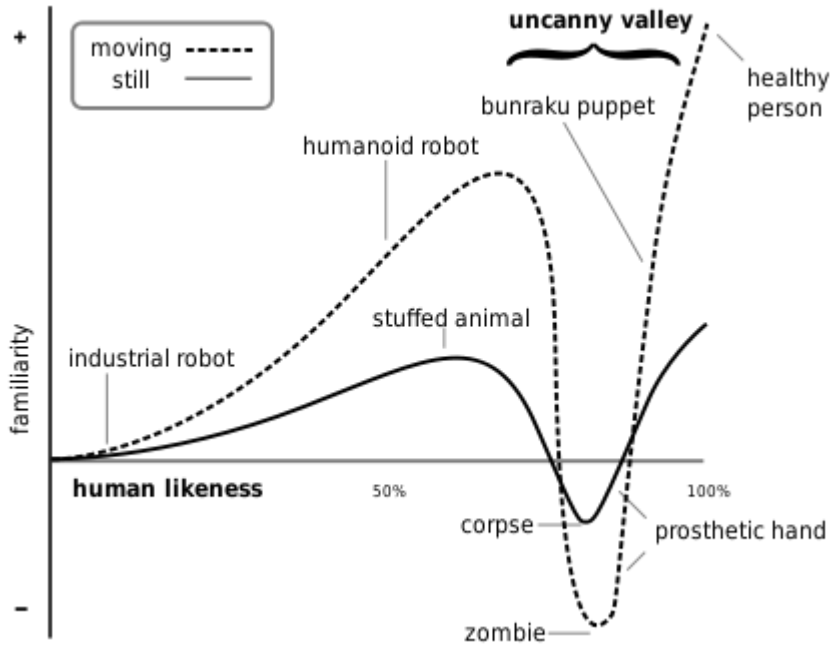


Figura 2 Grafico che illustra la conformazione della uncanny valley nel rapporto tra somiglianza alla figura umana e senso di familiarità

Mori definisce la depressione descritta come *Uncanny Valley* nella quale fa ricadere anche figure estranee alla robotica quali *zombies* e cadaveri, per meglio spiegare quale sia la sensazione elicitata da questo genere di presenze che mostrano quella incoerenza tra apparenza riconoscibilmente umana, ma con qualche disturbante elemento di incoerenza.

É essenziale sottolineare come tale sensazione sia maggiormente accentuata dal movimento dell'oggetto in questione; spesso, infatti, sarà proprio il movimento innaturale a creare disagio anche se associato a una estetica estremamente realistica. Pertanto Mori dissuade dal tentativo di raggiungere il secondo picco del suo grafico, in quanto i risultati avrebbero portato ulteriormente verso il fondo della valle (Mori 1970).

La spinta tecnologica però non segue il consiglio dello scienziato giapponese e la ricerca di un realismo che valichi la *Uncanny* Valley prosegue tanto nella robotica, quanto nella

computer grafica (Tinwell 2015 verificare). Anche in quest'ultimo campo, infatti, si possono verificare fenomeni di *uncanniness.*

Tinwell passa in rassegna diversi esempi riguardo la CG per il cinema e per la *gaming industry* e mostra come, nonostante lo sforzo verso la creazione di *features* sempre più vicine alla figura umana i risultati percepiti siano stati fondamentalmente deludenti.

Infatti, tanto nella cinematografia, quanto nella grafica real-time ogni qualvolta un lavoro viene presentato come la nuova frontiera per il fotorealismo le aspettative del pubblico vengono puntualmente deluse. Elaborazioni sempre più avanzate come quelle di *motion capture* e grafica sempre maggiormente nitida non sono ancora in grado di sopperire al divario con la figura umana ed il suo movimento, in particolare per quanto riguarda la mimica facciale.

Si sono visti esempi nei quali il personaggio viene percepito come troppo distante rispetto alle emozioni rappresentate dalla scena, ad esempio *Final Fantasy: the Spirits Within*, di Hironobu Sakaguchi (2001) e *The Adventures of Tintin: Secret of the Unicorn*, di Steven Spielberg (2011), altri come il gioco *Heavy Rain* (Quantic Dream, 2010) in cui l'audio veniva considerato incoerente con la freddezza del modello o, addirittura, in cui i modelli venivano comparati a vittime di interventi chirurgici malriusciti come nel gioco (anche se più datato) *Quake 4* (Raven Software, 2005).

É necessario allora cercare di comprendere quali siano i tratti che ci fanno percepire un essere umano come vivente e reale piuttosto che inseguire la perfezione grafica basandosi solo sulla imitazione visiva ed intuitiva.

Tinwell cerca di individuare questi elementi analizzando le linee guida per la creazione di Virtual Human realistici che diversi autori hanno cercato di definire. Molti studi hanno mostrato come vengano percepiti maggiormente vicini alla presenza umana, non tanto i personaggi con un alto grado di fotorealismo, ma personaggi anche dichiaratamente non umani con caratterizzazioni che abbiano la possibilità di esprimere emozioni umane. La natura umana, difatti, ha una grande capacità di usare l'antropomorfismo per attribuire tratti umani ad oggetti o animali. Anche Mori (1970; 2012) ha dimostrato come personaggi antropomorfi, pur non possedendo l'aspetto umano siano percepiti più positivamente di androidi iperrealisticamente umani.

Senza tralasciare l'importanza della caratterizzazione espressiva illustrata da Tinwell, va però notato molti dei casi analizzati in letteratura riguardano unicamente la componente visuale, accanto alle caratteristiche visuali andranno considerate quelle relative al modello comportamentale.

L'autrice mostra infatti come in molti casi la deviazione motoria dal comportamento umano in androidi *human-like* sia una grossa fonte di *uncanniness*, sia che si parli di fluidità dei movimenti, sia per quanto riguarda i tempi di reazione, sia per l'incongruenza acustica.

Ho *et al.* (2008) parlano in questo caso di interazione contingente, ovvero la reazione attesa dall'utente nell'interazione con un agente automatico, come chiave della percezione di realismo. Si ricerca nelle reazioni non solo la fluidità dei movimenti, ma anche la congruenza delle caratteristiche di movimento con l'aspetto visuale (Vinayagamoorthy *et al., 2005*). La congruenza comportamentale e nei confronti dell'ambiente circostante pare estremamente importante nella percezione di realismo ed influisce anche sulla percezione dell'ambiente virtuale intero.

Vinayagamoothy *et al*. (2005) sottolineano inoltre l'enorme importanza della comunicazione non verbale, come ad esempio il ruolo dello sguardo, nella interazione con personaggi artificiali. Questo tipo di interazione, sebbene non si sia provata determinante in personaggi stilizzati, è stata provata come attesa nel confronto con personaggi *human-like,* in quanto aumenterebbe la congruenza estetico-comportamentale.

1.3 IMPORTANZA DELLA COMUNICAZIONE NON VERBALE

La ricerca sulla Comunicazione Non Verbale (NVC) ha una storia ampiamente multidisciplinare che va dalla ricerca sociale, alle arti performative e abbraccia persino la cultura popolare ed i manuali di auto miglioramento (Tenebaum 2004).

Lo studio formale della NVC affonda le sue radici nell'era Vittoriana con tra i primi gli studi di Charles Darwin in *The Expression of the Emotions in Man and Animals* (Darwin, 1874). L'interesse per l'analisi del movimento crebbe poi durante la rivoluzione industriale e con gli studi sulla meccanizzazione del lavoro. In questo campo fu influenzata da un approccio scientifico e analitico allo studio dell'efficienza sul posto di lavoro, basato sullo studio fotografico dei movimenti (Moore, 2005). La macchina fotografica permise ad artisti e

scienziati di catturare e studiare in profondità ogni sfumatura dei movimenti, delle pose e dell'andatura, come si può notare dalle sequenze fotografica di Edward Muybridge del tardo diciannovesimo secolo (Muybridge, 1979).

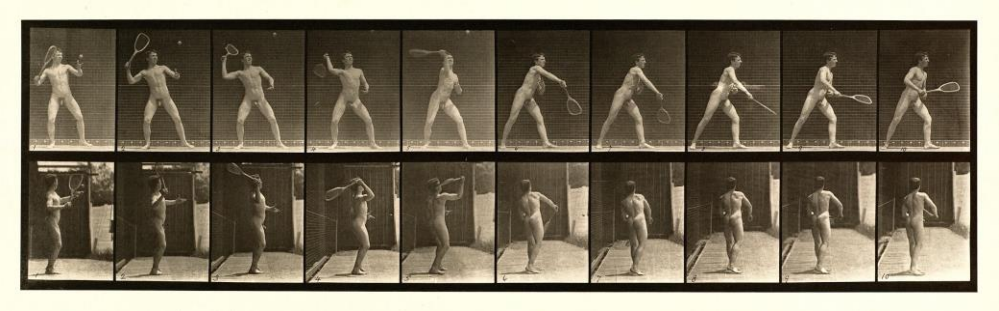


Figura 3 Sequenza fotografica di Muybrige del 1887[9]

L'inizio del ventesimo secolo vide tuttavia un declino nell'interesse scientifico verso la comunicazione corporea, ma le arti performative si avvalsero di questi studi con ricerche sulla danza come quella di Rudolf Laban e venne sviluppato un sofisticato sistema per l'annotazione e l'analisi del movimento umano (Maletic, 1987; Stebbins, 1977).

Nelle scienze sociali un nuovo vento di ricerca sulla NVC fu portato nel 1950 da Ray Birdwhistell con il suo lavoro sulle cinestetiche e, nel decennio successivo, dal lavoro di Edward Hall sulle prossemiche, che richiamò un grande interesse del pubblico verso il linguaggio del corpo alla pubblicazione del lavoro sensazionalista di Julius Fast che attirava l'audience con frasi promettenti come: "*how to penetrate the personal secrets of strangers, friends and lovers by interpreting their body movements, and how to make use of these powers*" (Fast, 1970).

Nonostante l'interesse di natura essenzialmente commerciale per il linguaggio del corpo (o forse proprio a causa di questa), negli ultimi 40 anni sono stati sviluppati svariati modelli rigorosi di comunicazione non verbale che sono stati adoperati di sovente negli ambienti virtuali, sia sfruttando gli stessi come ambiente di studio, sia essendo loro stessi sfruttati al fine di migliorare l'interazione con VH.

[9] Sequenza fotografica di Muybrige del 1887

Un campo comune di ricerca per gli studi sulla NVC[10] in ambienti virtuali è la percezione di diversi gradi di realismo negli elementi comunicativi non verbali. Esempi ne sono: la variazione del realismo estetico, consistenza tra segnali verbali e non verbali, il ruolo dello sguardo e delle prossemiche.

Nel libro *Nonverbal Communication in Virtual Worlds: Understanding and Designing Expressive Characters* (Tanenbaum 2014), che analizza le dinamiche della comunicazione non verbale nei Virtual Worlds (VW) e Virtual Environments (VE), Tanenbaum ci fornisce degli importanti elementi da tenere in considerazione sul ruolo della comunicazione non verbale in ambienti virtuali. VW e VE hanno, infatti, molte caratteristiche comuni ma si separano proprio per *Embodiment* e *Proprioception*.

L'embodiment è un fenomeno di ampia portata i cui fenomeni vengono classificati da Tim Rohrer (2007) in 12 dimensioni, di cui due sono particolarmente interessanti per la realtà virtuale:

Phenomenology: ovvero la coscienza del proprio corpo che può essere *conscious phenomenology*, quando l'immersione in un determinato scenario è consapevole e deliberata, oppure *cognitive unconscious*, quando il processo di immedesimazione é guidato dall'ambiente, ovvero quello che spesso succede nei VE.

Perspective: embodiment può riferirsi ad un particolare punto di vista, nei VE questa è intesa come la prospettiva dalla quale si osserva l'ambiente, spesso mediata da qualche tipo di Avatar, e della quale è funzione *embodiment*.

Solitamente nei VE il tipo di *embodiment* che si ricerca è quello fenomenologico. Si tratta, infatti, di esperimenti condotti singolarmente in ambienti spesso immersivi come CAVE o caschetti HMD e altamente controllati. In queste condizioni il senso di presenza dei partecipanti è focalizzato sul proprio corpo e si sovrappone alla rappresentazione virtuale (mentre, ad esempio, nei VW, spesso osservati in ambienti non immersivi e mediati da avatar personalizzabili nelle maniere più fantasiose, il senso di presenza dipende principalmente dalla capacità immaginativa del partecipante).

[10] Non verbal comunication

Anche se la mole di studi sulla NVC nei VE non è enorme possiamo trovare qualche interessante studio come ad esempio quello di Bailenson *et al.* (2001) sul realismo dello sguardo e le prossemiche, che vedremo più approfonditamente nel paragrafo 3.5.

1.4 IL PROGETTO *SPEAKTOME*

Tenendo bene a mente quanto visto in questa introduzione (tratti importanti della NVC, fattore *Uncanny*, distinzione e elementi utili ad Avatar e VH, importanza e applicazioni della presenza umana nei VE), questa tesi si propone di comporre un modulo software che costituisca una base per integrare Avatar e VA in ambienti virtuali (immersivi e non) e di gestirne al meglio le funzioni conversazionali.

L'obiettivo è di fornire una serie di strumenti e metodi per creare il più svariato numero di applicazioni nelle quali la presenza umana e la conversazione abbiano un ruolo chiave (e di conseguenza può essere utile anche per arricchire contesti nei quali la presenza di tali funzioni è marginale) e dunque siano implementate con estrema attenzione al loro impatto sull'utente.

Nel prossimo capitolo si procederà ad illustrare le metodiche attuali volte alla creazione di animazioni realistiche, sullo stato dell'arte riguardante il *rendering* dei VH, ovvero gli strumenti e le tecniche che permetteranno di fruire al meglio del prodotto realizzato. Si analizzeranno i canali principali utilizzati attualmente, in particolare quello audio-visivo, essenziale per quanto presentato in questa tesi. Verrà inoltre presentata una panoramica esemplificativa sulla realizzazione pratica del modello. É importante, infatti, al fine di "dare vita" ad esseri umani virtuali, capire come essi vengano creati nella pratica, quali siano le caratteristiche dei modelli e le relative metodiche di animazione. Sarebbe impossibile altrimenti valutare il miglior metodo e risolvere i più comuni problemi.

Successivamente si condurrà un'indagine più approfondita dello stato dell'arte dell'animazione e *rendering* nello specifico caso del viso; si esporranno inoltre le dinamiche della locuzione, dello sguardo e i riferimenti utilizzati per la creazione di espressioni realistiche in VH.

Nel quarto capitolo verrà esposta l'architettura realizzata: *SpeakToMe* (nome ispirato alla celebre serie TV statunitense *Lie to me* creata nel 2009 da Samuel Baum, nella quale uno

studioso di NVC applica, a scopo investigativo, l'interpretazione delle micro espressioni tramite il FACS[11]), introdotta da una panoramica sugli strumenti utilizzati per la sua composizione e conclusa con la presentazione di due esperimenti derivati dalle animazioni sviluppate, che hanno portato alla pubblicazione di due articoli scientifici, uno dei quali è arrivato alla pubblicazione negli atti della conferenza AVR Salento 2015

Nel quinto capitolo verranno esposti i risultati e le considerazioni riscontrate durante il test di tali esperimenti e verranno introdotti i futuri sviluppi relativi alla procedura sperimentale per le funzioni di *lip sync*.

Infine verranno esposte le conclusioni e gli obiettivi raggiunti dal lavoro svolto durante la stesura della tesi ed eventuali sviluppi implementativi futuri.

[11] Facial Acting Coding System ideato da Paul Ekman nel 1978, verrà illustrato approfonditamente nel paragrafo 3.5.1

2. ANIMAZIONE E RENDERING

Per realizzare quanto detto finora, dunque costruire e animare un Virtual Human, è necessario innanzitutto creare una struttura interna che ne determinerà la deformazione e consentirà la percezione visuale di un movimento quanto più possibile realistico.

La sintesi del movimento di animazione si configura come un misto di fenomeni naturali, percezione ed immaginazione (Thalmann 2001). L'animatore riproduce il comportamento dinamico di un oggetto secondo la propria rappresentazione intuitiva di causalità; immagina come un oggetto si muove e come reagisce quando viene spinto, premuto, tirato o torto. L'animazione dovrà quindi predisporre strumenti di controllo che permettano di tradurre le intenzioni dell'utente. Questa funzione nell'ambito dei VH è svolta dal *Motion Control Method* (MCM) che specifica come viene animato il modello e può essere caratterizzato in base alle informazioni usate per l'animazione. Ad esempio se si usa la cinematica diretta (Forward Kinematics - FK) si prediligerà l'informazione riguardo forze e torsione.

Come vedremo, esistono molti differenti tipi di MCM, ad esempio *motion capture*, *keyframe*, cinematica inversa (IK - Inverse Kinematics), FK, *walking models*, *grasping models* etc, ma nessun metodo da solo raggiunge la perfezione ed è necessaria la combinazione di più metodi per fornire risultati ottimali.

2.1 LO SCHELETRO

Sebbene i Virtual Human condividano molte caratteristiche con la figura umana reale, sono molto maggiori le differenze. Ne è un esempio la figura scheletrale, che viene usata essenzialmente per il movimento e raramente aggiunge struttura al corpo. A parte i rari casi di modelli che usano complessi sistemi muscolari e scheletrali, per l'animazione le ossa dei personaggi sono rappresentate da sistemi di forme geometriche semplici; la rotazione e traslazione di esse influisce sulla *mesh* sovrastante, sia essa composta di poligoni, NURBS o *splines*, e con qualsiasi delle tecniche che vedremo a breve.

Ci sono molti metodi per creare lo scheletro di un modello (in gergo *riggare*) e non ne esiste uno giusto o sbagliato. Solitamente il software predispone di una implementazione standard, la cui forma e proporzione saranno determinate da quelle della *mesh* o potranno esservi adattate (Ratner, 2012).

Bisogna tenere conto però che in alcuni software potrebbe verificarsi il problema del *gimbal lock* ed in questo caso l'oggetto (ovvero il singolo *bone*) non ruoterà più correttamente attorno a uno o più dei suoi assi e si dovrà effettuare una doppia rotazione che creerà delle piccole distorsioni nel movimento. Questo accade quando due assi di rotazione vengono a sovrapporsi e si verifica in tutti i sistemi che usano angoli di Eulero, in quanto ogni asse viene interpretato indipendentemente in un certo ordine gerarchico di X, Y e Z.

Un'alternativa per sopperire al problema è la manipolazione delle rotazioni tramite quaternioni; essi valutano contemporaneamente i tre assi di rotazione per determinare la direzione del movimento e un quarto valore (la componente W o *up vector*) che specifica la quantità di movimento. Lo svantaggio in questo caso consiste nella maggiore complessità di lettura e concettualizzazione.

Esistono tuttavia metodi per sopperire al problema anche utilizzando angoli di Eulero:

- Cambiare l'ordine di rotazione degli assi in base alla direzione di movimento (se permesso dal software)
- Creare ogni giunto rivolto verso la direzione nella quale dovrà ruotare

2.2 CINEMATICA INVERSA E DIRETTA

Il modello semplificato di "muscoli" nominato precedentemente, che si può meglio considerare come uno scheletro, viene chiamato *Rig* e ad esso è assegnata una determinata gerarchia, che permetterà di posizionare il modello cambiandone la configurazione e di animarlo secondo le proprie esigenze.

Vengono usati essenzialmente due tipi di configurazione, la cinematica diretta e la cinematica inversa.

2.2.1 Cinematica Diretta

La cinematica diretta è la configurazione maggiormente intuitiva e sfrutta la gerarchia dello scheletro e l'equazione cinematica per aggiornare ricorsivamente la posizione dei giunti dato un nuovo set di coordinate.

Questo sistema sarebbe computazionalmente molto oneroso se usato su un sistema scheletrico altamente realistico e questa è una delle ragioni per le quali si usa un modello semplificato nelle animazioni real-time, dove la performance è essenziale.

2.2.2 Cinematica Inversa

La cinematica inversa agisce, invece nella maniera opposta: data la posizione di arrivo desiderata per determinati giunti, viene computato e vincolato in che modo i giunti coinvolti dovranno essere traslati e ruotati in modo da raggiungere la posizione desiderata. Questo consente importanti vantaggi nel far compiere all'avatar delle azioni precise e può essere usata anche nella correzione, ad esempio, di dati ottenuti con tecniche di *motion captur*e che possono essere rumorose o non ottenere la precisione della posizione relativa all'ambiente virtuale nelle quali sono inserite.

Il processo della IK risulta più complicato rispetto alla FK, in quanto i parametri dei giunti che possono soddisfare un vincolo input non sono univoci. Ci sono dunque diversi metodi per computare l'IK e differiscono per semplicità, robustezza e performance:

- Un metodo molto sfruttato nella *game industry* è detto *Cyclic Coordinate Descent* (CCD). É un sistema euristico che itera il movimento passando attraverso ogni giunto della catena IK e ruota i segmenti in modo da far raggiungere al giunto target la posizione vincolata. Questo sistema è molto intuitivo e di facile implementazione, tuttavia può creare problemi di coerenza tra i diversi frame di animazione: infatti possono prodursi posizioni vincolanti simili a partire da parametri estremamente differenti e questo causerebbe discontinuità nel flusso di animazione.

- Il metodo analitico affronta il problema imponendo il vincolo che la catena IK non abbia più DOF[12] dei vincoli del giunto target per determinare la soluzione. Sebbene questa specifica restringa il campo di applicazione di solito sono sufficienti 7 DOF per modellare un arto (o simili). Dal momento però che il vincolo target possiede 6 DOF i parametri del giunto non saranno univoci e andrà specificato un angolo aggiuntivo per renderli tali.
- I metodi Jacobiani approssimano la soluzione risolvendo un problema di ottimizzazione numerica direttamente tramite la sua linearizzazione. Viene computata una matrice Jacobiana che include le derivate parziali della funzione tra il vettore con le posizioni target del giunto ed i parametri dello stesso. Questo riduce il problema di ottimizzazione non lineare a una serie di problemi lineari al massimo di secondo grado. Data la linearizzazione del problema la soluzione varia armonicamente in base alla configurazione dello scheletro ed i vincoli del giunto target. Tuttavia questa soluzione presenta almeno due complicazioni: il risultato può essere distorto in caso di posizioni target esterne alla catena IK e per gerarchie scheletrali complesse la computazione risulterà molto costosa, in quanto per ogni iterazione andrà calcolato un sistema lineare denso.
- Una variazione del CCD è il metodo *Particle-based* che, differentemente dal primo opera direttamente sulle posizioni dei giunti, anziché sulle rotazioni. La metafora che si può usare per descriverne il processo è quella di attaccare una molla ad ogni segmento: in un primo passaggio il giunto finale del segmento viene mosso verso la posizione desiderata ed in un secondo passaggio il vincolo di lunghezza del segmento viene riadattato con il movimento dei giunti alle sue estremità. L'operazione viene poi iterata fino alla convergenza di tutti i segmenti. Questo metodo ha proprietà essenzialmente analoghe a quelle del CCD ma il risultato appare più regolare in quanto opera nel dominio delle posizioni piuttosto che in quello degli angoli (Hale *et al.* 2014).

2.3 SKINNING

Il processo di *Skinning* è quello che riporta le trasformazioni relative all'animazione del *rig* sulla *mesh* e consiste nel *blending* delle trasformazioni dei diversi giunti applicato a ogni

[12] Degree *of Freedom*

vertice della *mesh*. In questo modo basterà studiare l'animazione a livello scheletrale per animare l'intero modello.

Molti tra i software per la modellazione 3D permettono di assegnare dei valori ai vertici della *mesh*. In questo modo è possibile controllare l'influenza di ogni bone sulle specifiche parti dell'oggetto. Ad esempio i pesi possono essere "dipinti" su un'area specifica per controllare meglio la rigidità o flessibilità del modello. In questo modo si possono creare deformazioni più sofisticate della *mesh* esterna. Molti sistemi permettono, inoltre, di creare automaticamente una mappa basilare che sarà poi possibile regolare nel dettaglio testando i movimenti per ogni singolo giunto.

Vengono impiegate diverse tecniche per realizzare lo *skinning* relativo a una *mesh* o per risolvere i problemi che possono presentarsi nel processo:

- SSD: *Skeletal Subspace Deformation* - É spesso chiamato in questo modo lo *skinning* con *blending* lineare; è il metodo di *skinning* più intuitivo: rappresenta le singole trasformazioni dei giunti come matrici 4x4 e computa la loro somma pesata per trasformare ogni vertice. Questo metodo è molto usato per via della sua semplicità, ma spesso la somma pesata di matrici non produce la qualità desiderata in quanto si possono verificare perdite di volume e forme "collassate", specialmente nella somma di matrici di rotazione.
- PSD: *Pose Space Deformation* - Questa tecnica è stata ideata per contrastare gli artefatti che possono verificarsi con lo *skinning* SSD. L'idea è quella di modellare un set di forme fisse per determinate pose chiave e computare le differenze tra queste e quelle prodotte dalla SSD. In questo modo vengono compensati molti degli errori prodotti dalla SSD, ma è richiesto uno sforzo aggiuntivo da parte del modellatore che dovrà creare tali pose e aggiungerà dei costi in termini di computazione.
- DQS: *Dual Quaternion Skinning* - É simile al SSD ma anziché usare matrici 4x4 per rappresentare le trasformazioni rigide, sfrutta doppi quaternioni che forniscono una rappresentazione compatta di rotazione e traslazione. Il vantaggio di questa tecnica è che il *blending* di due DQ produrrà a sua volta un DQ e quindi una valida trasformazione rigida; si eviteranno in questo modo i problemi della SSD di collassi e artefatti *candy wrap*.

Sebbene le tecniche di *skinning* siano una rappresentazione efficiente per l'animazione di avatar, hanno qualche limite: un esempio è la difficoltà di rappresentazione dei

rigonfiamenti muscolari a partire dalle trasformazioni rigide dei giunti scheletrici. Se in parte il problema è stato risolto dal PSD questo non basta per deformazioni con rotazione molto ampia. Recenti ricerche hanno affrontato questo problema aggiungendo parametri di scalatura e *shear* che possono essere predetti in base alla posa scheletrica, pre-computando un modello di regressione in base a una *mesh* esemplificativa. Un altro limite riguarda le *self-collision* che possono comparire con la deformazione di una *mesh*. Infatti, essendo il modello scheletrico una semplificazione, non ha una pregressa conoscenza della *mesh* corrispondente. Una possibile soluzione è quella di risolvere le collisioni utilizzando dei *bounding volumes* cubici o cilindrici associati ad ogni *bone*, come approssimazione della *mesh* corrispondente. Questo metodo risolve in maniera approssimativa i problemi di *self-collision*, sebbene non preveda trasformazioni più sottili come, ad esempio, la compressione della pelle nelle collisioni. Questo resta ancora un problema aperto e sul quale si cercano metodi maggiormente efficienti (Hale *et al.* 2014).

2.4 Motion Capture

Una tecnica di animazione enormemente usata, specialmente nel *realtime*, è l'importazione di movimenti acquisiti tramite *motion capture*. Per *motion capture* si intende sostanzialmente il processo che permette di tradurre la registrazione di un movimento in termini matematici tracciando un numero di punti chiave nel tempo e combinandoli per ottenere una singola rappresentazione tridimensionale della performance. (Menache 2011).

Le origini delle attuali tecniche di *motion capture*, sia esso basato su marker o *markerless*, si possono far risalire alla tecnica del *rotoscope*, ideata e brevettata da Max Fleischer nel 1915. La tecnica consiste essenzialmente nel copiare i movimenti di un filmato con attori reali e riportarli sui personaggi di animazione allo scopo di automatizzare parzialmente il processo di animazione.

Nonostante esempi celebri come *Biancaneve*, film Disney del 1937, la tecnica venne largamente disdegnata dal mondo dell'animazione, in quanto veniva considerata dagli artisti come una sorta di "scorciatoia". Il discorso è ben diverso però per la sua evoluzione in *motion capture*, nella grafica 3D infatti spesso prendono il sopravvento sull'originalità

e la manualità artistiche altre esigenze quali il livello più alto possibile di efficienza e riduzione dei tempi di lavorazione.

Alcune delle attuali tecniche di *Motion Capture* sono state usate, a livello di ricerca, sin dalla fine degli anni 70, sebbene abbiano fatto la loro comparsa nel mercato solo dalla metà degli anni ‘80. Un progetto pioneristico che aprì la strada al mondo dell’animazione di personaggi tramite *Motion Capture* è la sfida affrontata da Robert Abel nel 1985, quando gli venne richiesta l’animazione di una donna fatta di cromo per lo spot *Brilliance* commissionato dal National Canned Food Information Council. L'associazione era formata da Heinz, Del Monte, Campbell e altri grandi protagonisti del mercato del cibo in scatola e richiedeva uno spot estremamente innovativo con l'intento di rinnovare l'attenzione verso il cibo in scatola che si sentiva come obsoleto. Si può facilmente comprendere come una richiesta del genere, in un mondo della grafica popolato ai tempi da loghi in movimento, paesaggi e oggetti rigidi, ponesse un obiettivo altamente complesso ma che avrebbe aperto la strada dello sviluppo della figura umana digitale.

Prima di tutto bisognava però occuparsi del movimento: l’idea fu quella di fotografare il movimento di un’attrice da diverse angolazioni, utilizzando svariate camere, e usare il filmato per creare un algoritmo di movimento. Abel, insieme a molti tra i futuri protagonisti della computer grafica, fotografarono quindi da diverse angolazioni una modella-ballerina, posizionata su uno sgabello con rotazione di 360 gradi in modo da non impedire i movimenti e sul corpo della quale erano stati creati con un pennarello indelebile 18 *markers*, in corrispondenza dei suoi giunti. Le immagini vennero poi processate su una delle prime workstation della Silicon Graphics, la SGI IRIS 1000 per produrre una serie di algoritmi e animare così il modello digitale *wireframe*. Rimaneva il problema di ricoprire il tutto di cromo in maniera efficiente ed economica, risolto da Charlie Gibson che trovò il modo di applicare una *texture map* che si animasse seguendo la topologia del corpo, quella che oggi conosciamo come *Reflection Map*.

Dal sistema di *brilliance* i sistemi di *motion capture* hanno avuto una grossa evoluzione e riduzione dei prezzi; se prima infatti si trattava di tecnologie accessibili a pochi e sfruttate maggiormente nella *game industry*, proprio in questo campo il grosso potenziale delle tecniche di *motion capture* ha reso possibile una grossa diffusione e contenimento dei

prezzi. Ad oggi ci si aspetta che pressoché qualsiasi personaggio presente in un videogioco sia animato tramite dati acquisiti con tecniche di *motion capture* (Menache 2011).

Gli attuali sistemi di *motion capture* sono classificati come:

- *Outside-in*: dove sensori esterni collezionano dati da fonti posizionate sul corpo
- *Inside-out*: dove sensori collocati sul corpo collezionano dati da fonti esterne.
- *Inside-in*: dove sia sensori che fonti di ricezione sono collocati sul corpo in movimento

Le tecnologie principali sono di tipo ottico, elettromagnetico ed inerziale.

Una nota molto importante riguardo gli attuali sistemi di *motion capture* riguarda le tecnologie *markerless*, per le quali una pietra miliare è stata posta dalla Microsoft Kinect (2012), la quale rileva automaticamente le posizioni dei giunti da una singola scansione di profondità. Questo crea una soluzione robusta, veloce, economica e automatica, oltre che ovviamente non intrusiva come possono invece essere i sistemi *marker-based*. Tuttavia il *tracking* tramite sensore ottico della *kinect* non è privo di lati negativi, specialmente in caso di movimenti complessi o occlusione rilevante (Wei *et al.* 2012).

2.5 Rendering Visivo

Bisogna tenere conto, nella creazione di un umanoide virtuale, quale sarà il suo scopo ultimo ed in base a ciò privilegiare le caratteristiche appropriate. Infatti, la grafica real-time deve fare i conti con la performance molto più attentamente rispetto alla grafica pre-renderizzata delle produzioni cinematografiche. Bisognerà quindi capire qual è il modo migliore di produrre dei modelli sufficientemente credibili, ma allo stesso tempo efficienti per animazioni virtuali real-time.

2.5.1 Tools

Per ottenere modelli realistici ed efficienti che possano essere facilmente animati tramite *motion capture* è essenziale la scelta degli strumenti da adoperare:

Per quanto riguarda la modellazione di *mesh* sono disponibili diversi software *opensource* o commerciali come ad esempio: 3D Studio Max[13] (Autodesk), Maya[14] (Autodesk), Blender[15] (Blender Foundation). Questi permettono di creare il modello tenendo a bada il numero di poligoni e la topologia dell'oggetto, oltre a poter manipolare le *uv map* che potranno essere realizzate nello stesso software o più comunemente in programmi di editing grafico come Photoshop[16] (Adobe) o Gimp[17] (GNU Image Manipulation Program - The GIMP Development Team). Inoltre dispongono di ottimi strumenti per il *rigging*, *skinning* e animazione del modello e svariati formati di esportazione.

Una tecnica particolare per la realizzazione delle *textures* è la proiezione dei dettagli creati su software di *sculpting* come Zbrush[18] (Pixologic), Mudbox[19] (Autodesk) o Sculptris[20] (Pixologic), in questo modo i colori ed i dettagli più fini possono essere scolpiti direttamente sul modello *low-poly* suddiviso svariate volte come se si lavorasse su modello di cera o del materiale desiderato, in modo da avere la massima libertà artistica senza dover tenere a bada il dettaglio geometrico. I dettagli saranno poi proiettati sulle UV *map* precedentemente predisposte tramite il software di modellazione.

2.5.2 Rappresentazione classica: le mesh

Uno dei metodi maggiormente usati per la produzione di modelli umanoidi (ma anche in generale per la produzione di modelli animabili da usare in applicazioni di grafica 3D) è l'uso di *mesh* geometriche controllate da un set gerarchico di nodi che rappresentano le giunture e parti mobili del personaggio.

La *mesh* può essere costruita manualmente dall'artista 3d o acquisita manualmente tramite processi di scansione. Per realizzare manualmente la *mesh* esterna, ovvero la struttura

[13] http://www.autodesk.com/products/3ds-max/overview
[14] http://www.autodesk.com/products/maya/overview
[15]https://www.blender.org/
[16] http://www.adobe.com/it/products/photoshopfamily.html
[17] http://www.gimp.org/
[18] http://pixologic.com/zbrush/features/overview/
[19] http://www.autodesk.com/products/mudbox/overview
[20] http://pixologic.com/sculptris/

esterna di un Virtual Human, come per ogni altro modello 3D, ci sono diverse tecniche (Hale et al. 2014) :

Patch modeling

Alla base della patch *modeling* vi è il concetto di *spline* e NURBS. Le *splines* sono segmenti di linea flessibili definiti da punti di controllo chiamati vertici. Nascono dall'idea che ebbero nel 1959 Paul de Casteljau e Pierre Bézier di iterare combinazioni affini di *mesh* poligonali per ottenere superfici sinuose per la modellazione di telai di automobili. Una curva *spline* dunque è una curva morbida descritta in base a pochi punti di controllo; questa proprietà è molto importante in quanto riduce enormemente lo spazio di memoria occupato. Le *splines* hanno diverse tipologie, come ad esempio: di Bezier, Naturali, curve di Hermite, Catmull-Rom, *beta-splines* etc… Una categoria particolarmente interessante ai nostri fini è quella di B- *splines*, che hanno controllo locale sulla superficie e nessuna relazione tra grado e numero di punti di controllo.

I NURBS sono una estensione della curva razionale dove i punti di controllo che rappresentano il peso sono estesi con una coordinata.

Per le *splines* o NURBS è spesso necessaria una rappresentazione 2D, prima della più complessa trasformazione 3D. Queste sono chiamate superfici parametriche per via della loro curvatura predefinita che si adatta automaticamente ad alcuni punti dati (Panait, 2010).

Con questo metodo è possibile lavorare indipendentemente su diverse sezioni della figura e congiungerle in un secondo momento come una sorta di patchwork. In questo modo l'artista potrà avere il massimo controllo sulla modellazione. I vantaggi del patch *modeling* sono svariati ad esempio: la possibilità di dividere in sottosezioni il modello per affrontare la modellazione delle parti più complesse, la possibilità di nascondere i patch sui quali non si sta lavorando in modo da potersi concentrare meglio su quello in lavorazione e la facilità di creazione di aperture per occhi naso e bocca consentita dalla manipolazione della direzione di *splines* e NURBS (Ratner, 2012).

Subdivision modelling

La *Subdivision modeling* si può definire come una via di mezzo tra il metodo appena visto e la modellazione poligonale. É, infatti, una rappresentazione poligonale tesa ad ottenere superfici morbide tramite tecniche di suddivisione per interpolazione o schemi di approssimazione.

Per poligono intendiamo una porzione di piano limitata da tre o più linee rette, o segmenti, connessi da vertici. Chiameremo superficie poligonale una collezione di vertici connessi da triangoli o quadrilateri. Con tali superfici è possibile approssimare uno svariato numero di forme e, in quanto costruzioni compatte, il loro *rendering* è semplice ed efficiente. Tuttavia, per approssimare superfici morbide è necessario un numero di vertici elevato ed inoltre queste superfici non sono deformabili arbitrariamente. Questa permette di creare modelli dettagliati e regolari tenendo bassa la conta dei poligoni.

Il software di modellazione permette, a partire da un cubo, la suddivisione ed estrusione e rilievo dei poligoni. In questo modo si ha un controllo estremo sulla complessità della geometria del modello, problematica essenziale per le applicazioni *realtime*, in quanto la memoria occupata risulta essenziale per la performance.

2.5.3 Ricostruzione da immagini

Una diversa tecnica che potrebbe apparire molto promettente nella modellazione di personaggi fotorealistici è quella basata sulla ricostruzione da immagini. Il vantaggio di queste tecniche è quello di poter arrivare velocemente alla creazione di un modello (visivamente e fisicamente) altamente realistico personalizzato senza bisogno di conoscenze pregresse sulla fisica e anatomia degli stessi. Potrebbe quindi essere interessante per eventuali sviluppi che permettano all'utente una creazione del proprio avatar realistico in tempo reale.

Tuttavia, proprio la mancata conoscenza delle suddette caratteristiche può portare ad errori e artefatti nell'animazione. Inoltre in questo tipo di modelli il controllo sulla topologia sarà molto minore e, nonostante algoritmi di semplificazione sempre più fini, il numero di poligoni resterà più elevato di un modello manuale e spesso meno regolare.

D'altro canto questa tecnica produce modelli animati con un alto livello di fotorealismo, difficilmente riproducibili dalle attuali tecniche di modellazione e sarà molto utile per altri scopi, specialmente in modelli statici dove il livello di dettaglio è essenziale, ad esempio nelle ricostruzioni di reperti storici nei beni culturali (Hale et al. 2014) .

Dunque quando avremo bisogno di modelli "chiusi" ad esempio per la stampa 3D o con geometria regolare e numero di poligoni limitato dovremmo ricorrere alla modellazione manuale per ottenere un risultato soddisfacente.

Una soluzione alternativa è proposta da Madracevic e Sogoric in 3D *modeling* from 2D images (2010) che studiano una soluzione parzialmente automatica da completare con la modellazione diretta in diversi passaggi:

- *Model recording*: un espediente per sopperire alle distorsioni create nella ricostruzione da immagini, dovute ad inevitabili movimenti della persona, è quella di posizionare tre camere attorno al soggetto da acquisire (ai lati e di fronte) e registrare simultaneamente i tre punti di vista. In questo modo le distorsioni saranno minime grazie alla più ampia scelta di *frames*.

  Alternativamente, viene usata una unica camera che ruota intorno all'oggetto registrandolo: bisognerà, in questo caso fare attenzione agli errori prodotti non solo da movimenti dell'oggetto e cambiamento di espressione, ma anche da un movimento erroneo della camera o della scelta sbagliata dei punti di registrazione (che di solito dovranno essere ogni 45 gradi mantenendosi su un'asse orizzontale). L'ampia scelta dei frame nell'utilizzo di un video anziché scatti singoli aiuta a correggere problemi di questo genere.
- HDR: *High Dynamic Range Imaging method* e *Tone mapping* vengono usate in un secondo passaggio per rendere Illuminazione, tono, colore e *shading*. Questa tecnica consiste nel combinare tre immagini: una sovraesposta, una sottoesposta ed con esposizione ottimale, questo crea toni ottimale di valore in ogni aria, con dettaglio maggiore nelle aree chiare e scure. Questo si può realizzare anche in maniera fittizia tramite programmi di editing grafico, regolando l'esposizione artificialmente e creando così un falso HDR.

A partire da questi elementi viene creato un modello automatico basilare che ingloba la *texture* ma è ancora lontano dal fotorealismo; occorre un ulteriore passo: la modellazione diretta. Il modello è prodotto dai primi passaggi in formato .obj e può essere quindi modificato

agevolmente in software commerciali (e non) come 3Ds max, dove la rifinitura del modello può essere effettuata usando come riferimento le stesse immagini usate per la ricostruzione automatica.

Gli autori sostengono che allo stato attuale non esistano sistemi soddisfacenti a tutto tondo per la modellazione basata su design totalmente automatico o totalmente manuale. Dunque la soluzione di una combinazione di metodi automatici e modellazione diretta può essere un buon compromesso per una veloce realizzazione di modelli ad alto livello di fotorealismo (Madracevic e Sogoric, 2010).

Per motivi analoghi a quelli visti riguardo la geometria, e considerando l'impiego di risorse temporali e strumentali possibili per il progetto oggetto di questa relazione, non ho ritenuto necessario approfondire ulteriormente i meccanismi automatici di generazione di modelli. Essi non si limitano alla ricostruzione delle immagini, ma possono comprendere svariati meccanismi di acquisizione come ad esempio scanner ottici 3D, l'approfondimento di tali tecniche per la creazione di modelli *low-poly* regolari è comunque una prospettiva futura molto interessante.

2.5.4 Texture Maps

Inoltre al modello possono essere aggiunti dettagli come ad esempio colore, materiale tramite *textures* che possono contenere diverse proprietà comprese quelle relative a illuminazione ed interazione della luce con gli oggetti della scena (ad esempio specularità). Le *textures* possono essere usate per sopperire a dettagli eccessivamente complessi per essere resi al livello di dettaglio geometrico (e che appesantirebbero quindi la performance, essenziale per l'uso real-time) come capelli, rughe e piccoli dettagli simili.

L'idea di base è quella di rendere visivamente le caratteristiche interne di un essere umano, ad esempio i muscoli saranno rappresentati tramite il loro effetto superficiale, piuttosto che modellati nel dettaglio effettivo. Modellare tali strutture esplicitamente, infatti, potrebbe portare un alto livello di realismo, ma a costo di una enorme complessità in termini di creazione e di tempo computazionale.

Un esempio di tali semplificazioni è il processo di *rigging* e *skinning* visto in precedenza nel quale il numero di "muscoli" non è quello anatomicamente corretto ma piuttosto un livello

muscolare semplificato che contribuisce a simulare allungamenti e rigonfiamenti del livello superiore della pelle (Hale et al. 2014) .

Un elemento molto importante nella modellazione 3D è, dunque, la cura della superficie dell'oggetto. Infatti si può risparmiare molto del tempo e dei poligoni usati per ricreare i dettagli più fini utilizzando le *texture maps*, che possono oltretutto migliorare enormemente il risultato finale. Questo è cruciale nella creazione di figure umane, nelle quali il numero di dettagli fini è enormemente elevato. Quando ben fatta una *texture map* è indistinguibile dalla geometria e molti programmi permettono addirittura di memorizzare i dati assieme all'oggetto.

Anche se esistono molti metodi per proiettare un'immagine su un oggetto, quello maggiormente sfruttato ' l'UV *mapping*, specialmente per il posizionamento preciso di *textures* facciali. Questo tipo di mappe funzionano, infatti, particolarmente bene per oggetti che presentano forti irregolarità. Le *texture maps* sono connesse a punti specifici dell'oggetto ed in questo modo l'artista potrà avere un altissimo grado di controllo sulle stesse.

L'UV *mapping* può essere realizzato in svariate maniere: un metodo comune è quello di "srotolare e appiattire" la *mesh wireframe* ed esportare la *texture* per poterla modificare con programmi di image editing. L'immagine verrà poi salvata e assegnata al modello e la *texture* si adatterà alla forma dello stesso.

Un secondo metodo è quello di usare speciali programmi di pittura 3D che permettano di lavorare direttamente sul modello.

2.5.5 Programmable computer graphic shaders

Il termine *programmable*, in riferimento agli *shaders*, non è forse il più adatto se si pensa all'evoluzione storica degli stessi. Essi, infatti, nascono in un'epoca nella quale, in mancanza di alternative, ogni aspetto della computer grafica doveva necessariamente essere programmato manualmente. Solo in un successivo momento la scrittura di buoni programmi di grafica fu resa enormemente più facile dalle cosiddette API[21], che mettevano

[21] Application Programming Interface

a disposizione molte funzioni utili, sebbene questo a volte comportasse perdere molte funzionalità non previste dalla API in questione. Lo sviluppo della ricerca nella computer grafica fortunatamente non si arrese e sopperì al problema rendendo programmabili (di cui il nome) gli aspetti tagliati fuori dalle API. Tale funzionalità ha raggiunto degli standard veri e propri che includono il linguaggio GLSL, parte dello standard OpenGL; programmi del genere sono ormai vitali nella computer grafica.

Quando si usa la API OpenGL per sviluppare un'applicazione grafica si definiscono proprietà geometriche, visive di proiezione e di apparenza tramite funzioni specifiche e gli oggetti vengono descritti in termini di vertici, normali e primitive (Bailey et al. 2011).

In anni recenti questo ha reso possibile il loro utilizzo per la realizzazione di dettagli e tecniche di alta qualità anche per le applicazioni di tipo real-time. Infatti tramite algoritmi che sfruttano la scheda grafica anziché appesantire la CPU possono essere rese in tempo reale molte tecniche di illuminazione, deformazione e simulazioni fisiche etc.: ad esempio la simulazione di corpi morbidi o rigidi, effetti di luce sofisticati, riflessologia e comportamento dell'occhio umano (Hale et al. 2014).

2.6 Rendering Audio

Lo sviluppo di tecnologie di realtà virtuale si è concentrato in maggior misura alla cura di elementi sensoriali riguardanti la sfera visiva. Tuttavia, è stato ipotizzato che l'aggiunta di segnali sensoriali auditivi aumenti notevolmente il senso di presenza percepito dall'utente (Larsson 2002). La problematica è estremamente interessante nell'ambito di questa tesi in quanto l'obiettivo concerne almeno per metà la sfera auditiva e mira a una sincronizzazione plausibile tra l'animazione facciale di un VH e la voce ad esso associato, sia essa espressa tramite sintesi vocale (TTS Text-To-Speech), riconoscimento vocale di un file *wave* selezionato dall'utente o input real-time tramite un microfono. Vediamo dunque di capire gli elementi importanti da tenere in mente nella implementazione di un *rendering* auditivo ed in particolare nel caso di VH in IVE.

La simulazione di fenomeni acustici è un'area di ricerca molto ampia, che va dal semplice posizionamento in uno spazio 3D di effetti sonori, a complesse simulazioni di sale da concerto (Naef, 2002). Colonne sonore che accompagnino l'ambiente creato dalla

computer grafica sono in circolazione da molto tempo, ma la problematica risulta più complessa quando ci si trova a realizzare un render sonoro in un IVE nel quale il suono deve essere fornito in tempo reale. Secondo Naef (2002) per sviluppare un audio credibile in IVE ci sono almeno 4 requisiti fondamentali da tenere a mente:

- Localizzazione 3D: il posizionamento delle fonti sonore è di cruciale importanza per creare un ambiente sonoro. La posizione percepita della fonte sonora dovrà coincidere con la posizione nel VE dell'oggetto virtuale associato. In questo caso la sfida è mappare arbitrariamente le fonti e gli oggetti sfruttando il numero limitato di speaker, la cui posizione sarà oltretutto spesso vincolata dal setup fisico del *Sound Interface Device* (SID)
- Simulazione della stanza/ambiente: è importante ricreare le proprietà dell'ambiente come ad esempio ampiezza della stanza, proprietà di riflessione del suono delle pareti etc.
- Input audio Live: oltre al suono sintetizzato bisognerà tenere conto di eventuali suoni che derivano da input esterni locali o remoti, specialmente nelle situazioni di VE collaborativi dove gli input potrebbero essere molteplici.
- Velocità e efficienza: come per il *rendering* visuale (ma non solo) bisognerà tenere conto della natura real-time della simulazione; bisognerà quindi cercare il miglior compromesso tra accuratezza della riproduzione delle proprietà fisiche dello spazio sonoro e della sua realizzazione efficiente tramite l'uso di hardware standard.

2.7 Rendering Aptico

La parola "Aptico" (dal greco ἁπτικός, apticos, ovvero riguardante il tatto) viene introdotta alla fine degli anni 20 nel mondo della psicofisica per descrivere la sotto branca riguardante percezione e manipolazione basate sul tatto. In seguito, tra il 1970 ed il 1980 inizia ad esplorare il campo tattile anche la robotica, trovando non poche problematiche e campi di inquisizione. Andava quindi arricchendosi il patrimonio di studi sia dal punto di vista delle dinamiche del tatto che della loro riproduzione tramite dispositivi meccanici.

Nel 1990, proprio grazie alla confluenza di questi studi si genererà un nuovo paradigma di studi su questo settore, ovvero la *computer haptics*; analogamente alla computer grafica si rendeva possibile per la prima volta la simulazione tattile di oggetti virtuali in maniera interattiva. Questo nuovo tipo di display sensoriale permetteva di trasmettere informazioni

alla mano umana tramite una interfaccia aptica, così come nella computer grafica i display fornivano informazioni all'occhio umano tramite la luce. Tali informazioni derivano dalle risposte della mano umana, o di altre parti del corpo, agli stimoli di una macchina con uno scambio bidirezionale di energia ed informazioni.

A onore del vero, le prime interazioni aptiche con oggetti virtuali risalgono al 1960 (dimostrate da Artur Knoll - Salisbury 1999), ma solo recentemente si è arrivati a mezzi tecnologici abbastanza avanzati da consentire l'interazione con oggetti complessi. Ora, con la disponibilità commerciale di dispositivi aptici con 3 o più DOF e software *toolkit* provenienti da diverse fonti aziendali e accademiche e molte applicazioni commerciali possibili, il campo pare essere indirizzato a rapidi ed esaltanti sviluppi.

Per ciò che concerne questa relazione, sebbene non se ne faccia uso attualmente, è interessante fare una breve panoramica sulla componente del *rendering* aptico, ovvero il processo per il quale gli stimoli sensoriali desiderati vengono trasmessi all'utente in modo da comunicare informazioni sull'oggetto rappresentato, al livello più semplice in termini delle sue caratteristiche fisiche quali forma, elasticità, *texture*, massa etc.

Per proseguire il paragone con il *rendering* visivo si possono riscontrare anche in questo caso diversi livelli di dettaglio, dai soli vincoli di impedenza a tecniche che riproducono una *texture* o la frizione della superficie meccanicamente.

Le interfacce aptiche si possono descrivere come dei "piccoli robot" che scambiano energia meccanica con l'utente e si usa solitamente il termine *device-body interface* per sottolineare la connessione tra il corpo umano e la macchina. Anche se l'interazione può riguardare qualsiasi parte del corpo, lo sviluppo di questi sistemi si è attualmente concentrato particolarmente sulle mani.

Le interfacce aptiche possono essere catalogate secondo diverse delle loro caratteristiche:

- *Grounding locations*: un elemento di distinzione riguarda il posizionamento del loro punto di contatto. Vi sono i dispositivi da tenere in mano come i *force-feedback gloves* nei quali vengono lette in input le informazioni specifiche di contatto per ogni dito e trasmesse in output informazioni di resistenza, sebbene non si possa riprodurre il peso o la forza inerziale; dispositivi analoghi sono vastamente diffusi nella *game industry* e sono costituiti da trasduttori vibro tattili a basso costo che producono effetti di vibrazione sintetici. Un

secondo tipo riguarda i meccanismi esoscheletrici o interfacce *body-based* nelle quali l'utente indossa meccanismi motorizzati su un braccio o una gamba e che presentano sistemi motorizzati più complessi con molteplici DOF. Infine i sistemi locati su base a terra che includono tra gli altri joystick che riflettono la forza di resistenza o interfacce aptiche per desktop.

- Comportamento meccanico intrinseco: *Impedance haptic devices*, che simulano l'impedenza meccanica leggendo la posizione dell'utente e rispondendo con una determinata forza; *Admittance haptic devices,* che simulano l'ammettenza ricevendo in input una forza e trasmettendo una posizione. Le architetture del primo tipo, essendo più semplici da progettare e più economiche da costruire sono molto più diffuse rispetto alle seconde che vengono usate solitamente in applicazioni che richiedono forze e spazi lavorativi più ampi.
- DOF: una classificazione molto comune per le interfacce aptiche riguarda il numero di DOF presenti nel dispositivo; tali gradi possono essere attivi o passivi, percepiti o non percepiti.

Comunque si classifichino, le caratteristiche desiderabili per dispositivi aptici includono: bassa inerzia e frizione di ritorno (back-drive); massima limitazione dei vincoli imposti dalla cinematica del dispositivo, in modo da permettere movimenti più liberi; proprietà simmetriche di inerzia e frizione e di rigidità e risonanza in modo da evitare che l'utente compensi inconsciamente le forze parassitarie; equilibrio tra raggio di azione, risoluzione e banda della posizione percepita e forza riflessa; ergonomiche appropriate che permettano che l'operatore umano possa concentrarsi sul task indossando o manipolando l'interfaccia aptica, questa infatti potrebbe essere scomoda o dolorosa e distrarre l'utente, riducendo in questo modo il livello generale della performance.

Per computare la corretta interazione tra la rappresentazione aptica, ovvero l'Avatar, e gli oggetti che popolano il VE, si usano i cosiddetti *haptic-rendering algorithms.* In questo caso la scelta delle caratteristiche dell'Avatar sarà dettata sia dallo scopo della simulazione, sia dalle capacità del dispositivo aptico. Un algoritmo aptico è composto da differenti elementi nei quali possiamo distinguere tre blocchi principali:

- *Collision-detection*: rileva informazioni su momento, luogo e eventualmente misura (livello di penetrazione, area di contatto etc.) della collisione tra avatar e VE.
- *Force-response*: *computa* la forza dell'interazione al momento della collisione approssimando al meglio il contatto a quello che si avrebbe nella situazione di contatto reale.
- *Algoritmi di controllo*: minimizzano la discrepanza che viene a crearsi tra forze ideali e forze applicabili, dovuta alle limitazioni imposta dall'hardware (Salisbury 2004).

2.8 TECNICHE PRATICHE DI REALIZZAZIONE DEL MODELLO

Una volta analizzata la teoria che sta alla base della creazione di un VH, è interessante osservare le tecniche pratiche e le *pipelines* utilizzate oggi da molti modellatori per creare personaggi da usare in applicazioni real-time. É interessante, a questo proposito osservare la panoramica fatta da Danielle Todd (2012) sulle tecniche di modellazione *low poly* in 3d studio max (ma che si possono utilizzare in maniera abbastanza analoga sui software di modellazione citati precedentemente). Sebbene alcune di queste tecniche possano essere considerate da molti degli utenti un po' "datate", spesso possono essere utili per semplificare e accelerare il processo casi particolari come piccoli dettagli o forme semplici.

2.8.1 Spline modeling

Consiste fondamentalmente nella creazione di una rete di *splines* alla quale si applicherà un modificatore *Surface* per crearne la superficie. Questo tipo di modellazione può essere utile per creare forme libere, *mesh* organiche semplici e la base di modelli che abbiano spigoli rigidi; può essere utile ad esempio per modellare aperture irregolari in un avatar come quelle di occhi bocca e naso.

2.8.2 Patch modeling

È sostanzialmente analoga alla *spline modeling*, si avrà semplicemente un patch intero anziché una *spline*. I patch possono essere triangolari o rettangolari, i secondi sono suggeriti dall'autrice in quanto offrono una buona base per un modello che andrà poi suddiviso.

2.8.3 Box modeling

Una tecnica differente da quelle viste è la box *modeling*, estremamente popolare, specialmente in passato e si può dire antenata di quella che, come vedremo a breve, è la soluzione maggiormente usata nella modellazione *low poly*.

La popolarità di questa tecnica è quella di essere consentita dalla stragrande maggioranza dei programmi di grafica 3D e per molto tempo si è configurata come l'alternativa alla *poly modeling* in mancanza di un sistema robusto che la supportasse.

La differenza principale tra box e *poly modeling* è il modo nel quale si inizia a modellare: con la *box modeling* stabiliamo nelle primissime fasi quale sarà la massa finale approssimativa del modello. Il problema principale è la difficoltà nelle aggiunte di geometria, il che rende versatile la tecnica per modelli come macchine e altri maggiormente rigidi, ma più difficoltosa e meno intuitiva in caso di modellazione organica come quella dei VH.

2.8.4 Poly modeling

É il tipo di modellazione più usato e per molti artisti l'unica in grado di offrire il massimo grado di controllo, intuitività, accuratezza e velocità.

Anziché partire da un cubo che approssima la massa finale dell'oggetto come nella box *modeling*, in questo caso si è soliti partire da un poligono o gruppo di poligoni ed in seguito verranno usate varie tecniche di manipolazione che permettono enorme controllo su poligoni, vertici ed *edges* e di conseguenza su quella che sarà la struttura finale della *mesh*.

Non volendo in questo contesto entrare nel dettaglio delle tecniche e particolari di modellazione si rimanda alla bibliografia, specialmente il testo di Danielle Todd, Poly-*modeling with 3ds Max: Thinking Outside of the Box*(2012) e. *Game Art Complete* di Andrew Gaham (2008).

2.8.5 Texture maps e sculpting tools

Un punto molto interessante toccato dalla stessa Todd e sperimentato praticamente da chi scrive sono i nuovi metodi di creazione di *texture*.

Se per molto tempo la dicotomia era quella della creazione ed esportazione in formato immagine delle UV *maps* da editare in un secondo momento tramite programmi di editing grafico quale photoshop, particolarmente due programmi stanno cambiando le *pipelines* di molti modellatori *low poly*: Mudbox e Zbrush (Todd e Fitzgerald, 2012). Con queste applicazioni di *sculpting* è possibile infatti lavorare in maniera molto più diretta ed intuitiva e curare meglio i dettagli più fini. Entrambi, infatti, permettono di "scolpire" e dipingere direttamente sul modello 3d ogni minimo dettaglio e questo è reso ancora più naturale da strumenti come le tavolette grafiche che supportino livelli di pressione, ad esempio la serie Wacom[22].

L'oggetto viene facilmente importato tramite il formato obj e suddiviso in un alto numero di poligoni in modo da poter essere lavorato, una volta terminato verrà riportato in 3d studio max dove i dettagli verranno proiettati sulle UV *map* e utilizzati sul modello *low poly* mantenendo un alto livello di dettaglio ma una regolarità della topologia e un basso numero di poligoni.

Inoltre il modello ad alto dettaglio potrà essere usato all'interno di Zbrush per la creazione di render fotografici.

[22] http://www.wacom.com/

3. IL VISO

3.1 - IMPORTANZA DEL VISO NELLE APPLICAZIONI

Il viso è una *interfaccia* estremamente importante nelle comunicazioni interpersonali, in quanto il più spontaneo e ricco di segnali (Panait 2010). Si può comprendere quindi come il suo utilizzo per le comunicazioni tra Virtual Human in ambienti virtuali sia sempre stato di grossa attrattiva, se prima solo nell'immaginazione, ora anche nella ricerca di una realizzazione sempre più realistica.

Tuttavia risulta una delle parti più complesse, sia dal punto di vista della realizzazione grafica 3D, sia in termini di connessioni neurali da processare.

L'interesse che porta a cercare di superare tali difficoltà è l'uso pervasivo nelle applicazioni di computer grafica, alcuni esempi sono:

- Interfacce utente di livello avanzato
- Agenti digitali per la didattica e pedagogia
- Simulazioni in campo medico
- Simulazioni di criminalistica e analisi forensiche
- Avatar e agenti virtuali in videoconferenze
- *Game industry*
- Media e arte

Nella produzione filmica la computer grafica ha dedicato enorme attenzione alla mimica facciale, elemento essenziale sia per la caratterizzazione, sia per l'empatia dei personaggi virtuali. Non si può dire lo stesso però per la grafica real-time che, più orientata verso azioni e movimenti rapidi, resta ancora un passo indietro rispetto alla sua controparte pre-renderizzata e solo recentemente ha iniziato a collocare tra le sue priorità la cura dei movimenti del viso dei personaggi.

Personalmente ho ritenuto dunque importante creare un sistema che consentisse da un lato di sincronizzare i fonemi prodotti tramite sistemi di TTS, audio *wave* o input tramite un microfono con i movimenti delle labbra di un VH. Inoltre ho voluto aggiungere la simulazione di espressioni emozionali realistiche che creassero un maggiore senso di

empatia con l'utente (parte del progetto è poi sfociato in due esperimenti riguardanti la mimica facciale in agenti virtuali all'interno di ambienti virtuali immersivi). A questo scopo mi pare essenziale l'analisi delle metodiche e problematiche relative alla particolare animazione del viso ed i presupposti teorici alla base delle animazioni create.

3.2 L'ANIMAZIONE DEL VISO: PROBLEMATICHE E TECNICHE

Data la complessità ed il volume di calcolo necessari ad animare una scena 3D, le prime animazioni erano necessariamente pre-renderizzate (Liu 2012). Con il progresso della potenza computativa, tuttavia, è aumentata anche la possibilità di creare animazioni che valessero la pena di essere prodotte; ma la restrizione della performance, che richiede ordini di velocità estremamente superiori rispetto alla controparte pre-renderizzata, ha fatto sì che il livello di dettaglio nella grafica real-time sia altamente ridotto. Uno dei dettagli maggiormente sacrificati è stato dunque proprio il viso dei personaggi, che nei videogames degli anni '90 erano, ad esempio, *mesh* statiche animate unicamente a livello di *textures* e nei 2000 le innovazioni si limitavano all'aggiunta di palpebre e un unico giunto per il movimento della mascella per rendere il parlato.

Anche se indietro rispetto alle opere pre-renderizzate lo standard per la gran parte della *gaming industry* si è assestata in seguito a una struttura di bocca e sopracciglia più complessa nell'ottica di una mimica facciale sufficientemente soddisfacente.

In alcuni casi recenti si è iniziato però a spingersi oltre, fino a raggiungere livelli che possono iniziare a competere con le opere di animazione cinematografiche.

Un esempio è il videogioco LA Noire[23] (Appel 2011), presto seguito ed emulato da YAKUZA 4 (SEGA 2010) che hanno dedicato enorme attenzione alla mimica facciale, nel secondo caso arrivando ad usare tecnologie capaci di generare automaticamente la maggior parte delle animazioni discorsive a partire da un audio. Entrambi gli esempi non sono comunque privi di difetti: il primo, infatti, ebbe un costo spropositato ed il secondo, seppure più semplice, economico e flessibile, mancava di dettaglio rispetto al primo e faceva un grosso

[23] http://www.thegamingvault.com/2011/05/review-l-a-noire-ps3360. consultato Maggio 2011.

uso di risorse computazionali, spesso sopperendo con il passaggio in molti punti del gioco al *rig* facciale convenzionale.

3.3 Tecniche di animazione: Blendshapes o Rigging?

3.3.1 Blendshapes

Una tecnica molto interessante per animare i visi di personaggi virtuali e l'uso di *blendshapes*, ovvero l'uso di una serie di modelli modificati dall'artista in modo da mantenere la stessa geometria, ma allo stesso tempo di riprodurre espressioni nel modo più accurato possibile. Le diverse *mesh* vengono poi linkate alla *mesh* "base" con espressione neutrale e le trasformazioni dei vertici vengono interpolate linearmente tramite un processo di *blending*.

Le *blendshapes* sono state la soluzione usata tradizionalmente per le produzioni pre-renderizzate per molto tempo, in quanto consentono un altissimo livello di controllo da parte dell'artista. Tuttavia non sono prive di lati negativi, specialmente per quanto riguarda il real-time: non sono infatti supportate da molti game *engine* e, nonostante si miri alla standardizzazione, le *blendshapes* restano una soluzione caso-per-caso. Andrebbe creata una *blendshape* per ogni trasformazione e questo aggiungerebbe enormi costi, non solo in termini di lavoro di produzione artistico, ma anche per quanto riguarda la memoria che andrebbero ad occupare sulla scheda grafica, problema cruciale per le applicazioni real-time

3.3.2 Rig facciali convenzionali

Uno schema di animazione facciale che fornisca dettaglio e flessibilità con un basso costo di sviluppo non è ancora presente. Per questo motivo spesso le tecniche di *motion capture* vengono tagliate fuori e vengono usati metodi di animazione più tradizionali.

Sorge allora un altro problema: che schema di controllo adoperare? I due più comuni sono ad oggi la manipolazione diretta e quella basata su interfaccia grafica (GUI).

- GUI: crea una astrazione del viso come interfaccia 2D che può essere un viso simbolico o degli *sliders* 1D/2D. Nell'ultimo caso si tratta di semplici *sliders* che mappano le azioni facciali, nel primo viene aggiunto del tempo al processo di *rigging* per aggiungere

le zone di interesse controllate dall'interfaccia 2D. Questa tecnica, seppure molto semplice ed intuitiva, spesso non rende giustizia ai dettagli più complessi

- Manipolazione diretta: si tratta di aree di manipolazione solitamente posizionate in corrispondenza dei giunti del viso sottostanti, in modo da poter essere settata facilmente dal *rigger*

Una volta stabilito il sistema di controllo è poi possibile creare l'animazione manualmente tramite il lavoro degli animatori o tramite meccanismi di *motion capture*.

Uno strumento software molto interessante a questo proposito è il programma *faceshift studio*[24], che, oltre alla creazione di un modello di viso tramite scansioni con Kinect, permette di catturare un'animazione e utilizzarla per animare qualsiasi modello con un *rig* facciale o delle *blendshapes*. É particolarmente interessante in quanto indipendente dal tipo di *rig* o *blendshape*, e poiché comprende un plug-in per maya e 3ds max che permette di configurare il modello posizionandolo in pose standard specificate dal plug-in.

Per le animazioni utilizzate in *SpeakToMe* è stata usata la tecnica di modellazione con manipolazione diretta dei giunti per i visemi, più statici, mentre per quanto riguarda le animazioni emozionali i dati sono stati raccolti tramite *motion capture* e *faceshift*.

3.4 LE ESPRESSIONI

Le espressioni sono estremamente importanti nello stabilire un rapporto empatico e questo elemento può essere sfruttato per accrescere la sensazione di realismo di un avatar ed il conseguente senso di presenza in un ambiente virtuale immersivo, ma non solo:

Proprio con l'uso del già citato Faceshift sono state realizzate le animazioni emozionali contenute in *SpeakToMe* e sono evolute in un esperimento nel quale le espressioni di avatar hanno una estrema rilevanza; per maggiori informazioni si rimanda all'articolo relativo in via di pubblicazione *Assessment of Emotion Recognition from Faces in Immersive Virtual Environments Providing Distinct Contextual Information* di Faita *et al.* (al quale ho contribuito personalmente e che illustrerò più approfonditamente al capitolo 5.2) che ha

[24] (Faceshift AG, Zurich, Switzerland)

l'obiettivo di valutare il riconoscimento di espressioni all'interno di un contesto specifico in ambienti virtuali immersivi.

L'esperimento si basa sugli studi di Paul Ekman (1993) sulla connessione universale rilevata tra percezione dell'espressione e percezione dello stato emozionale percepito; per questo vengono adoperate quelle considerate le sei espressioni universali: felicità, rabbia, tristezza, disgusto, paura e sorpresa.

Il riconoscimento verrà poi testato in tre diversi contesti: positivo, negativo e neutrale per il controllo e mostrato ai soggetti da testare in un ambiente immersivo tramite HMD (per la precisione un Oculus Rift 1). Questo esperimento mostra come la mimica facciale di *virtual agent* sia utile non solo per creare maggiore realismo in ambienti virtuali immersivi e personaggi, ma possa essere sfruttata per testare, in ambienti controllati, fenomeni psicologici come il riconoscimento di emozioni, che alternativamente veniva testato tramite fotografie, le quali per via della loro staticità non forniscono un metro di valutazione ottimale. (Faita *et. al* 2015)

3.4.1 FACS

> *Facial expressions are one of the most important form of nonverbal communication, being one of the more immediate channels of social interaction between humans. Many previous investigations suggest that human faces, through the activation of certain muscle movements, convey specific emotions in a universal way*[25] (Faita *et al.* 2015).

. Sul filo della teoria illustrata da questa citazione, lo psicologo Paul Ekman ha sviluppato l'idea che esista una connessione universale tra le espressioni relative a espressioni

[25] "Le espressioni facciali sono una delle più importanti forme di comunicazione non verbale. Sono infatti uno dei canali più immediati tramite i quali gli esseri umani interagiscono socialmente. Molta della letteratura in merito sostiene che il viso umano, attraverso specifici movimenti dei muscoli facciali, comunichi determinate emozioni in maniera universale"

facciali di emozioni ed il relativo stato emotivo (Ekman, 1971). Ekman, contrariamente alle teorie che sostenevano che la cultura ed i processi di apprendimento fossero gli unici elementi a modellare la nostra percezione delle espressioni del viso (Mead, 1975), sostenne l'esistenza di sei emozioni basilari universali (Ekman, 1992) (felicità, tristezza, rabbia, paura, disgusto e sorpresa). Questa teoria è stata supportata da molti esperimenti basati sull'uso di fotografie statiche ed isolate di visi, che hanno dimostrato che gli umani sono in grado di distinguere le differenti caratteristiche delle espressioni che caratterizzano determinate emozioni (Adolphs, 2002; Leppänen and Hietanen, 2004). Basandosi su questo approccio, nel 1978 fu sviluppato un sistema per classificare sistematicamente i movimenti facciali: il *Facial Action Coding System* (FACS) (Ekman, 1978).

Figura 4 - Le sei espressioni delle emozioni basilari universali individuate da Ekman

Tuttavia l'uso di fotografie statiche fu una limitazione della validità ecologica di questa ricerca, in quanto non riproduceva la vivacità e la reale forma del viso così come le persone lo osservano nella realtà di tutti i giorni. Infatti, nelle interazioni in presenza l'osservatore non decodifica semplicemente il messaggio emozionale, ma ha necessità di capire anche il

modo in cui le emozioni si formano. Knight e Johnston, in *The role of movement in face recognition* (Knight & Johnston, 1997) hanno dimostrato che le emozioni di visi in movimento vengono riconosciute significativamente meglio rispetto a quelle in immagini statiche.

Dati questi presupposti, si capisce come l'interesse verso la simulazione di espressioni naturali attraverso l'animazione di personaggi tridimensionali sia cresciuto enormemente. Molti hanno inoltre dimostrato che esiste una importante correlazione tra riconoscimento di emozioni nei visi virtuali e quello in visi reali (Spencer-Smith *et al.* 2001). Inoltre l'uso dei primi aggiunge svariati vantaggi rispetto a dei filmati, in quanto possono essere adattate e manipolate, anche in tempo reale, in base allo scopo che ci si propone. Inoltre i sistemi di realtà virtuale che comprendono Virtual Human realistici in grado di comunicare emozioni si sono dimostrati utili al miglioramento della comunicazione interpersonale in individui con disturbi emozionali o comportamentali (Gutierrez-Maldonado *et al.* 2014).

A partire dalle caratteristiche dell'attività muscolare identificate in FACS, i visi 3D sono stati analizzati in molti esperimenti (Spencer-Smith *et al.* 2001; Dyck, 2001; Fabri 2004). Tuttavia, spesso è stato trascurato il livello di realismo o la rispettiva combinazione con il dinamismo dei rispettivi stimoli facciali. Tenendo conto di questo sono state sviluppate come parte di questo progetto delle animazioni dinamiche registrando le espressioni reali di un attore che imitava espressioni basilari individuate da Ekman. Questo ha permesso di creare in seguito, nel progetto che sarà esposto a breve (paragrafo 5.2), ed al quale ho contribuito personalmente, dei *Dynamic Virtual Avatar* (DVAs) che riproducessero vivacità ed intensità delle emozioni facciali come le osserviamo nella realtà. Delle emozioni individuate da Ekman per lo scopo di questa ricerca sono state considerate solo 5 su 6: Rabbia, Disgusto, Paura, Felicità e Tristezza, alle quali è stata aggiunta quella Neutrale. Ci si può riferire a questo set con EKMAN+N. L'espressione di sorpresa è stata esclusa in accordo con studi precedenti nei quali di sovente viene esclusa la sorpresa dalle emozioni base, in quanto spesso confusa con quella di paura (Faita *et. al.* submitted)

3.5 EYEGAZE

Tra i tratti essenziali della comunicazione non verbale, un altro elemento molto rilevante e affrontato abbondantemente in letteratura e il comportamento dello sguardo. I nostri occhi, infatti, forniscono segnali comunicativi riguardo i nostri sentimenti e ci aiutano a direzionare l'attenzione durante una conversazione.

Come fanno notare Colburn *et al.* (Colburn *et al.* 2000) è dunque essenziale dare importanza ai movimenti degli occhi e possibilmente cercare di tracciare lo sguardo dell'utente ed in base ad esso far reagire la sua controparte virtuale, in modo da creare una interazione più spontanea tra le due parti.

Tali riflessioni degli autori prendono spunto principalmente da due spinte: la prima è l'osservazione della mancanza di contatto visivo frequente nelle videoconferenze. Infatti dovendo in esse guardare il monitor non si potrà guardare in camera, a meno che non si ricorra ad accorgimenti speciali, ed in questo modo si perderà il contatto visivo con la persona con la quale si dialoga. Il secondo spunto proviene dalla volontà di creare migliori interfacce utente tramite rappresentazioni antropomorfe, che permetterebbero un'interazione naturale e quindi più semplice.

Tali spunti portano gli autori alla creazione di un modello stocastico comportamentale del movimento oculare nel contesto delle comunicazioni verbali tra Avatar in ambienti virtuali. Per la creazione del modello gli autori si sono basati su valori intermedi per adulti occidentali, non si tiene conto quindi di cultura, genere ed età, elementi molto importanti nel comportamento non verbale, sebbene gli autori si propongano di farlo in sviluppi futuri.

Anche Bailenson *et al*. (2001), si occupano della questione dello sguardo di agenti virtuali e Avatar in relazione alla prossemica, ovvero la gestione dello spazio personale come elemento di comunicazione non verbale. Questa è indicativa del livello di percezione del VH come presenza umana piuttosto che come oggetto inanimato. É stato inoltre analizzato come essa venga influenzata in base al contatto visivo ed è utile dunque per valutare l'influenza dello stesso in personaggi virtuali.

Le variabili manipolate nel loro esperimento sono state diverse, innanzitutto dal lato dell'agente virtuale sono state presi in considerazione due differenti gradi di risoluzione; per quanto

riguarda il partecipante, invece, sono stati considerati fattori chiave il genere e il comportamento dello sguardo.

Il modello inoltre aveva 5 differenti gradi di crescente realismo del comportamento ottico:

1. Occhi chiusi
2. Occhi aperti ma fissi
3. Battito ciliare
4. Sguardo che segue il partecipante
5. Sguardo che segue il partecipante e dilatazione delle pupille del 50% all'avvicinarsi entro 0.75 m del partecipante

Il modello non è quello di uno sguardo realistico ovviamente, ma è stato separato in diverse componenti in modo da testare separatamente l'influenza di ciascun elemento.

Per evitare di influenzare i partecipanti sul comportamento non verbale vennero dati degli elementi della scena da memorizzare e venne fatto intendere che la valutazione fosse sulla memoria.

Lo studio ha dimostrato che i partecipanti tendono a considerare gli agenti virtuali differentemente dagli altri oggetti inanimati presenti nella scena, rispettandone lo spazio personale in maniera molto simile a quella rilevata nelle conversazioni faccia a faccia. Anche la reazione ai segnali non verbali non si è dimostrata consistente con la letteratura in merito, infatti si rileva una maggiore reazione nei partecipanti di sesso femminile sia per quanto riguarda le reazioni nel comportamento non verbale, sia nel rispetto dello spazio personale che aumentava nei livelli più alti di realismo dello sguardo.

I risultati di questa ricerca sono molto interessanti, specialmente alla luce del fatto che non vi furono scambi verbali con gli agenti virtuali e che i partecipanti non erano consapevoli del vero scopo della ricerca. Questa è una ulteriore dimostrazione di come il comportamento non verbale, in particolare lo sguardo abbia un peso essenziale nella creazione di un VH credibile.

Nell'animazione degli avatar presenti in *SpeakToMe*, non è ancora presente un modello autonomo e aleatorio per il comportamento dello sguardo, tuttavia per creare il senso di empatia con l'osservatore si è pensato di implementare una funzione che segua la camera dell'ambiente relativa all'utente, e quindi lo "guardi negli occhi".

3.6 LE DINAMICHE DEL PARLATO

Nel contesto delle espressioni e la mimica del viso, un lato particolarmente interessante e centrale nella tesi qui presentata, è il *lip sync*, ovvero l'arte di coordinare il movimento labiale del personaggio virtuale con una traccia acustica o testuale.

Il problema di come realizzare una coordinazione soddisfacente è stato affrontato molto in letteratura ed è ancora uno degli *hot-topic* nell'animazione di personaggi digitali in ambienti virtuali. É utile a questo proposito citare qualche studio interessante e alcuni *tool* adoperati per svolgere il processo automaticamente.

3.6.1 Tecniche

L'animazione facciale relativa al *lip sync* è probabilmente la più semplice da comprendere, ma la più complessa da applicare, in quanto pone dei vincoli precisi di sincronizzazione rispetto ad una traccia audio e ad una appropriata deformazione della bocca. Il lavoro di *lip-syncing* è estremamente laborioso, in quanto bisogna analizzare minuziosamente la traccia sonora per determinare quale posizione delle labbra corrisponde al suono in ogni momento. Nella computer grafica però quello che è importante è creare una illusione realistica del parlato, sarebbe infatti impossibile riprodurre esattamente la pronuncia delle parole, in quanto essa è unica per ogni essere umano, a causa delle diverse conformazioni facciali, accento, lingua etc. Non è importante cosa accade realmente ma la percezione da parte dell'utente.

Ad un livello basilare, riguardo le posizioni della bocca ci sono due coppie di pose base dalle quali partire per creare una prima illusione di animazione conversazionale:

- Aperto/Chiuso
- Largo/Stretto

La prima coppia è la più importante in quanto movimento principale della bocca. I cicli del parlato sono composti da apertura e chiusura della bocca per quanto riguarda il movimento della mascella e di largo stretto rispetto al movimento delle labbra (Osipa 2007).

Word	Wide/Narrow Sequence	Open/Closed Sequence
why	narrow, wide	closed, open, closed
are	no change/ shape	closed, open, closed
we	narrow, wide	closed, slightly open
watching	narrow, slightly wide	closed, open, closed, slightly open, closed
you	narrow	closed/ no change

Figura 5 Mappatura della frase "Why are we watching you?" in termini di apertura/chiusura e ampiezza della bocca

I fonemi sono la più piccola unità linguistica di valore distintivo, ovvero che contraddistingue una parola da un'altra, come la "f" e la "t" in "detto" e "tetto" (Nespor, 1993). In altre parole i fonemi sono i suoni che la bocca produce durante la locuzione. La tecnica che sfrutta i fonemi per il *lip-synch* è stata usata per molti anni nell'animazione classica, ma risulta più ostica da applicare al mondo della computer grafica 3D.

Fonemi e visemi, che vedremo a breve, sono differenti in linguaggi differenti e le componenti del software oggetto di questa relazione utilizzano fonemi e visemi relativi alla lingua inglese, nella quale si possono individuare 38 fonemi.

Il famoso animatore Preston Blair ha individuato dieci forme base, la cosiddetta *Preston Blair phonemes series*, che viene ampiamente usata dagli animatori ancora oggi, ed è stata estesa a 12 visemi (Martin, 2006).

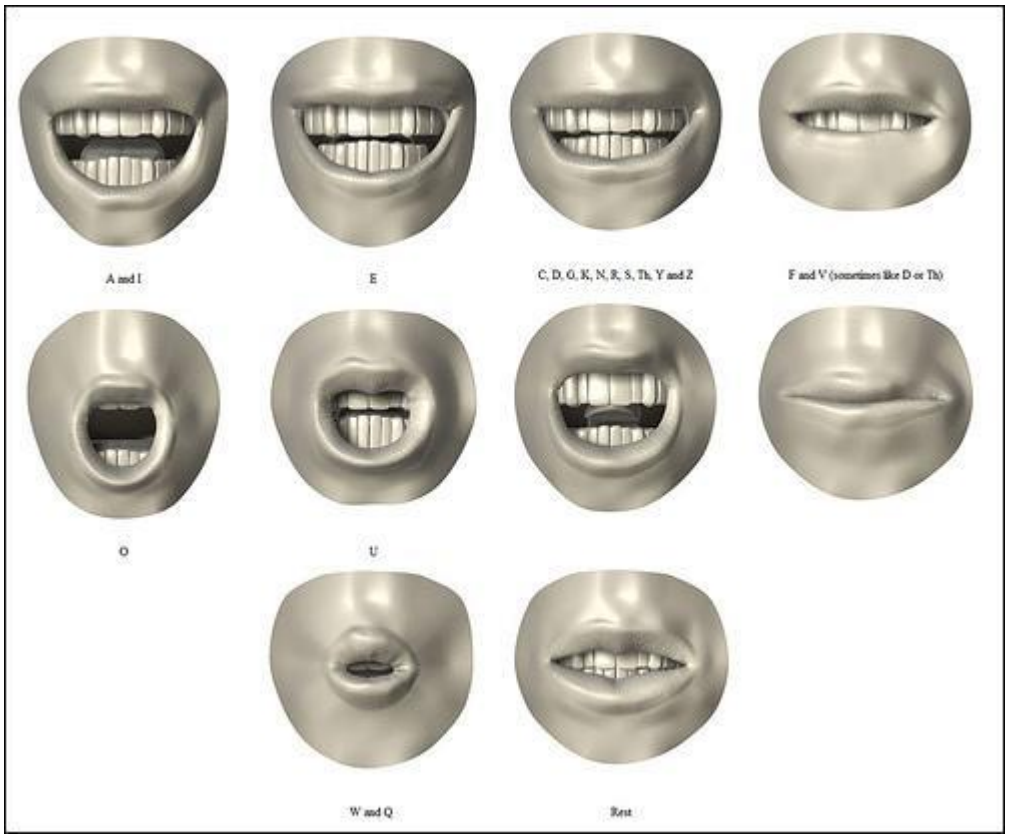


Figura 6 Preston Blair Series

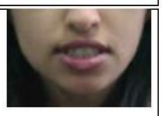
La serie consiste in forme della bocca che possono essere usate per rappresentare tutti i fonemi che utilizziamo, ma per creare una simulazione che faccia "vivere" il personaggio e riprodurre un dialogo reale è necessario l'uso di espressioni asimmetriche, in quanto la forma della bocca cambierà a seconda del viso del personaggio parlante.

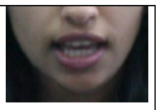
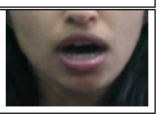
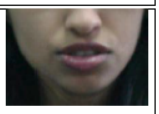
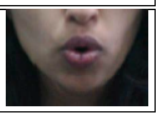
- A e I: le labbra sono leggermente tirate in larghezza, i denti aperti, la lingua visibile e appiattita sulla parte inferiore della bocca.
- E: le labbra tirate un po'più ampiamente, gli angoli più alzati e la bocca ed i denti un po' più chiusi
- U: le labbra sono protese all'esterno a e arricciate a cerchio ma leggermente aperte, i denti aperti e la lingua lievemente sollevata
- O - la bocca forma un cerchio leggermente più ampio ma le labbra non sono protese all'esterno, la lingua è appiattita sul palato inferiore
- C, D, G, K, N, R, S, Th, Y e Z: le labbra sono quasi chiuse e tirate ampiamente, i denti quasi chiusi.

- F e V: la bocca è pressoché alla sua larghezza standard, i denti sono appoggiati sul labbro inferiore, a volte la forma di D e Th può essere molto simile.

I Visemi sono dei suoni prodotti in corrispondenza di determinate posizioni o forme della bocca e che sono essenziali da rappresentare visualmente (ad esempio OO in *food*, M in *mom* etc.). In pratica sono i segnali visivi significativi composti dalle labbra. Ci sono tre tipi essenziali di suoni prodotti durante la locuzione: fatti principalmente dalle labbra (i più importanti nel processo di animazione), composti principalmente dalla lingua, composti essenzialmente da gola e corde vocali.

I visemi non sono legati ad un suono individuale, ma sono la rappresentazione di uno o più fonemi nel dominio visivo; dei 38 fonemi della lingua inglese se ne possono individuare 12 particolarmente importanti (Chikersal 2013) ed è proprio a queste 12 forme che si rifarà la successiva estensione della serie di Blair citata in precedenza e sulla quale ho basato la modellazione delle animazioni relative ai visemi.

Phoneme	Alphabetic spelling	Phonetic spelling	Viseme	
P	Pay	P EY	p	
B	Web	W EH B		
M	Lemon	L EH M AH N		
F	Four	F AO R	f	
V	Van	V AE N		
T	Raft	R AE F T	t	
D	Reed	R IY D		
S	Sit	S IH T		
Z	Arms	AA R M Z		
TH	Three	TH R IY		
DH	With	W IHDH		
DX	Forty	F AO R DX IY		
W	Swap	S W AA P	w	
R	Rest	R EH S T		
CH	Chair	CH EY R	ch	
JH	Cage	K EY JH		
SH	Blush	B L AH SH		
ZH	Asian	EY ZH AH N		

EH	Check	CH EH K	ey/iy	
EY	Take	T EY K		
AE	Bat	B AE T		
AW	Cow	K AW		
IY	Team	T IY M		
IH	Little	L IH DX AX L		
K	Coin	K OY N	k	
G	Peg	P EH G		
N	Night	N AY T		
L	Late	L EY T		
NG	Anything	EH N IY TH IH NG		
HH	Halt	HH AO L T		
Y	Yes	Y EH S		
AA	Barn	B AA R N	aa/ah	
AH	What	W AH T		
AX	About	AX B AW T		
AY	Type	T AY P		
ER	Church	CH ER CH	er	
AO	More	M AO R	ao/uh	
OW	Loan	L OW N		
OY	Hoist	H OY S T		
UH	Book	B UH K		
UW	Flew	F L UW		
IX	Action	AE K SH IX N	Silent	
SIL	-	The silence duration is variable but typically about 40 ms/ short pause	Sp	
SP	-	space btw words/characters		

3.6.2 Case study

Nel testo *Development of Real-Time Lip Sync Animation Framework Based On Viseme Human Speech* (Hoon *et al.* 2014) gli autori si occupano dello sviluppo del *lip sync* in real-time basato su visemi. Gli autori notano, osservando alcuni sistemi già adoperati, che spesso un'analisi della traccia audio basata sulla singola lettera rende il risultato finale confusionario o poco realistico. Una soluzione al problema potrebbe essere l'uso di *motion capture*, soluzione già spesso sfruttata, specialmente nelle produzioni cinematografiche ad alto livello. Tuttavia il costo di tali tecniche risulta, spesso, molto oneroso, mentre un sistema di automazione digitale come quello presentato dagli autori può essere una soluzione pratica, che allo stesso tempo produca risultati soddisfacenti.

Sfruttare i visemi anziché la singola lettera rende inoltre più semplice anche il primo passo di un sistema automatico, ovvero la coordinazione con un sistema di riconoscimento vocale.

In questo modo si potrà classificare in fonemi l'audio analizzato e riportarlo ai visemi corrispettivi predisposti sul modello in modo da sincronizzarlo.

Un altro *case-study* nel quale ha un peso fondamentale l'animazione facciale è quello condotto da Panait (2010) nella ricerca della costruzione di una libreria di animazioni per Virtual Humans. L'autrice, analizzando il miglior modo di realizzare un *lip sync* accettabile per un'animazione real-time, riporta diversi importanti schemi di semplificazione della catalogazione di visemi per la lingua inglese, a partire dalla basilare duplice dicotomia della forma della bocca aperta/chiusa e larga/stretta, che è il primo passo da comprendere prima di poter avanzare verso combinazione degli stessi in movimenti più articolati. L'autrice prosegue col citare le 10 forme base di Preston, semplificazione molto usata dagli animatori, ricordando però come sia essenziale la variazione, in quanto in un dialogo reale la forma della bocca varia da persona a persona.

Viene inoltre citata una ulteriore semplificazione, effettuata dal Toon Boom Studio[26], che riduce a 8 i visemi base e li fa corrispondere a specifici gruppi di fonemi assegnati automaticamente analizzando lo spettro acustico di un file audio importato sul software.

Per le funzioni relative al linguaggio di *SpeakToMe* si è usata una mappatura a 12 visemi. L'idea è stata quella di sfruttare le funzioni fornite dalla Speech API di Windows per comporre un sistema che aggiungesse le funzioni relative al *lip-sync* a strumenti di sviluppo per ambienti immersivi come XVR ed in particolare la libreria HALCA. Essa permette, infatti, l'uso pesato di animazioni sui giunti in modo da poter lavorare indipendentemente sull'animazione del viso e del corpo e di poter creare un *blending* tra le animazioni statiche in modo da renderle maggiormente fluide.

[26] http://www.toonboom.com/main/

4. ARCHITETTURA E IMPLEMENTAZIONE

Il lavoro di questa tesi si propone, utilizzando gli standard e le procedure illustrate, di realizzare una architettura che permetta di integrare i VH negli ambienti virtuali e di poterli gestire in tempo reale o tramite narrazioni ben definite da chi idea il programma. L'obiettivo è stato perseguito componendo una serie di moduli software che sfruttino strumenti utilizzati con successo dalla *game industry* e nella ricerca sul campo dei VH ed utilizzando dei modelli 3D altamente realistici la cui struttura standardizzata ne permette l'intercambiabilità. *SpeakToMe* si configura come una struttura facilmente personalizzabile; questo consentirà a chi crea l'applicazione, di disporre di strumenti per animare un numero arbitrario di personaggi, con dialoghi interattivi, tono emotivo, scenari e narrazioni personalizzate e finalizzate al proprio scopo. Nel presente capitolo verrà presentata, innanzitutto, una panoramica sui *tools* adoperati per la composizione del modulo software, successivamente verrà descritta nel dettaglio l'architettura nelle sue diverse funzioni: sintesi vocale da un testo o stringa, riconoscimento vocale da un file audio o in tempo reale, struttura dell'ambiente di *testing* e predisposizione a uno *storyboard* (con illustrazione di un primo *storyboard* basilare). Verranno infine presentati due esperimenti derivati dalle animazioni realistiche delle espressioni utilizzate in questa architettura.

4.1 TOOLS

4.1.1 XVR

Il progetto XVR nasce per lo sviluppo di applicazioni di realtà virtuale con accesso al web, l'ambiente di sviluppo della VR media è evoluto nel corso degli anni fino a diventare una tecnologia che comprende il più svariato numero di applicazioni interattive. Oltre a gestire il contenuto Web3D[27], infatti, XVR supporta ad oggi un enorme varietà di dispositivi per la realtà virtuale (e aumentata) come ad esempio, *trackers*, mouse 3D, dispositivi per il

[27] si intende per Web3D il contenuto 3D integrato in una pagina web.

motion capture, sistemi di proiezione stereo, ambienti CAVE-*like*, HMD etc. XVR utilizza inoltre un motore grafico in linea con lo stato dell'arte attuale che permette di visualizzare modelli tridimensionali complessi in maniera adeguata anche per installazioni di realtà virtuale off-line avanzate.

XVR usa un linguaggio *scripting* dedicato che utilizza costrutti e comandi specifici per la realtà virtuale e permette agli sviluppatori di gestire e manipolare animazioni 3D, effetti sonori posizionali, streaming audio/video ed interazione con l'utente. La piattaforma include inoltre un set di strumenti per la progettazione ed implementazione delle applicazioni di VR: l'ambiente di sviluppo integrato XVR Studio che contiene un editor, un *workspace* manager, un modulo per la pubblicazione FTP e una finestra di output per il *debugging*.

Inoltre per velocizzare il processo standard di inizializzazione di una applicazione interattiva viene fornito un *wizard* che guida il programmatore con una serie di istruzioni che lo guidano consentendo la scelta dei parametri di base della scena (numero e posizione di luci, camere etc.), la metafora di interazione ed i file iniziali da usare. Una volta completata la procedura viene generato un script s3D corrispondente ai parametri impostati e con una serie di suggerimenti (*todo sections*) che lo sviluppatore potrà completare a suo piacimento.

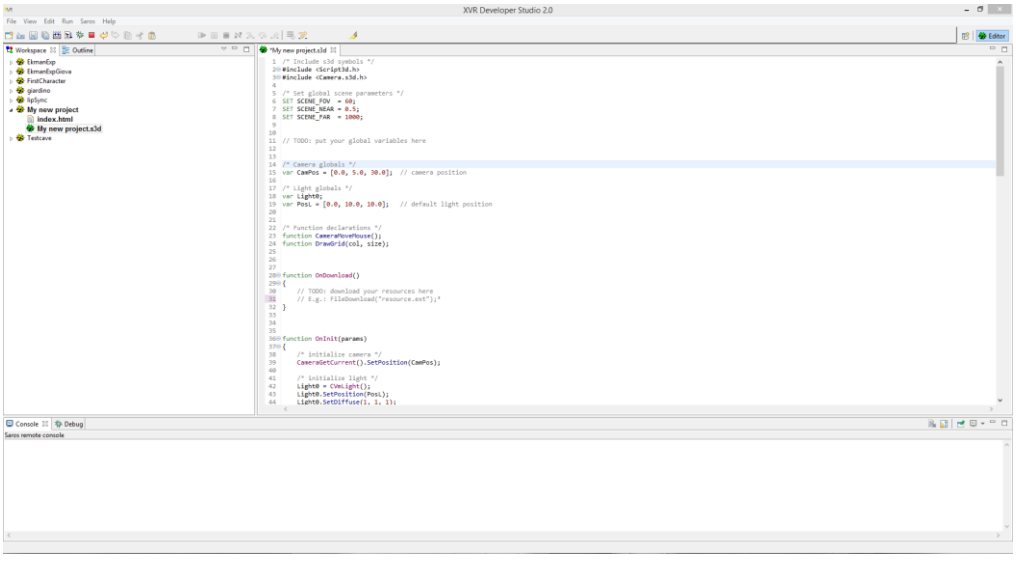


Figura 7 Xvr Studio

Le applicazioni XVR possono essere riprodotte utilizzando il player integrato (XVR player) una componente di ActiveX supportata dalle varie piattaforme di Windows e che può

essere integrata in differenti applicazioni come ad esempio il browser Internet Explorer. Il player si compone di due moduli: il modulo di controllo ActiveX, che include le componenti base come il controllo versione e le interfacce collegate e la XVR Virtual Machine (VM) che rappresenta il vero e proprio nucleo della tecnologia, include infatti il motore grafico 3D e tutti i moduli software che permettono di gestire le funzioni integrate in XVR. É inoltre possibile aggiungere dei moduli per consentire funzioni avanzate come il supporto a dispositivi per la realtà virtuale come guanti e *trackers*. Gli sviluppatori di XVR hanno ritenuto più opportuno separare queste funzioni affinché le applicazioni web, che solitamente non necessitano tali funzionalità, non siano vessate da tempi di download aggiuntivi.

La VM di XVR, come molte altre macchine virtuali contiene una serie di informazioni *bytecode*, una cosa e un'area di memorizzazione funzioni. Il linguaggio di *scripting* s3D usato da XVR consente di controllare il comportamento dell'applicazione tramite funzioni e classi che specificano le funzioni base del linguaggio e gli specifici metodi per la VR. Lo script viene poi compilato in *bytecode*, processato ed eseguito dalla VM. Il meccanismo di download, che si avvia all'accesso a una pagina web che contenga una applicazione XVR, è abbastanza standard: dopo un controllo versione ed eventuale aggiornamento, il *bytecode* viene scaricato assieme ai primi file necessari all'applicazione (modelli 3D, *textures*, suoni etc.); vengono poi eseguite tutte le inizializzazione necessarie ed infine l'applicazione inizia ad eseguire i vari cicli. Una applicazione XVR può essere rappresentata come un *loop* principale che ne contiene molti altri al suo interno ed ognuno ha una sua determinata frequenza, come grafica, fisica, network, *tracking*, e aptica. Questo semplice flusso e la sua modularità permette un facile sviluppo sia per il web che per applicazioni *standalone.*

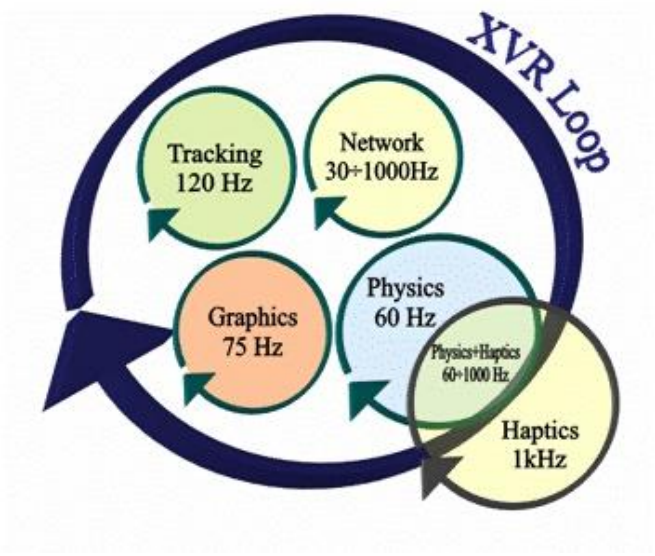


Figura 8 Struttura dei cicli di XVR

Una qualsiasi applicazione XVR si compone di 6 fondamentali funzioni principali predefinite che costituiscono la base di ogni progetto:

- OnDownload(): viene eseguita per prima e avvia i download dei file necessari all'applicazione.
- OnInit(): contiene il codice relativo alle inizializzazioni dell'applicazione, tutti i comandi vengono eseguiti sequenzialmente.
- OnFrame(): contiene tutte le funzioni ed i metodi che producono un output grafico ed è l'unica che permette di accedere al contesto grafico. Comandi grafici esterni a questa funzione non produrranno risultati.
- OnTimer(): questa funzione viene eseguita indipendentemente e dunque ad un diverso *framerate* dalla OnFrame() e contiene i comandi che si desiderano mantenere separati dalla funzione di *rendering*.
- OnEvent(); è indipendente da entrambe le funzioni precedenti e viene chiamata quando l'applicazione riceve un messaggio di tipo evento, tali messaggi possono essere esterni (provenire da Windows) oppure interni (generati dal programma stesso).
- OnExit(): viene chiamata quando si esce dall'applicazione o alla chiusura della finestra nella quale essa è contenuta.

Oltre alle funzioni basilari, XVR include molte classi, funzioni, e strutture dati predefinite e permette all'utente di definirne di nuove. Il motore 3D integrato, basato su OpenGL,

permette di gestire la grafica non solo su una finestra standard (sia essa locale o in rete), ma anche su dispositivi più avanzati come sistemi di proiezione stereo, HMDs etc.

4.1.2 HALCA

Per l'animazione e la visualizzazione degli avatar di questo progetto è stata usata la libreria HALCA (Hardware Accelerated Library for Character Animation)[28]. HALCA usa il formato file XML di Cal3D[29] per descrivere le mesh pesate sullo scheletro, le animazioni ed i materiali. Il nucleo di HALCA consiste in un *motion* mixer e un motore per la visualizzazione degli Avatar. HALCA permette di animare e visualizzare contemporaneamente su semplici display desktop, HMDs e dispositivi CAVE-*like*, uno svariato numero di Avatar realistici.

HALCA estende le funzioni di animazione e visualizzazione di Cal3D e permette allo sviluppatore di integrarle in applicazioni di realtà virtuale.

HALCA può sfruttare diverse modalità di *rendering*; nella sua versione più semplice usa funzioni base OpenGL e quindi può essere utilizzato da qualsiasi scheda grafica che supporti OpenGL. Quando viene adoperata la modalità *shader* può essere caricato e attivato uno *shader* GLSL (Rost, 2006) che viene caricato da un file di testo o presunto che il programma in cui è integrato attivi uno *shader* OpenGL.

Oltre allo *shading* standard in HALCA vengono utilizzati dei *vertex shader* per realizzare le deformazioni della pelle dell'Avatar coordinate allo stato dei giunti scheletrali o ai *morph target*. La gestione parallela di queste operazioni permette all'hardware grafico di eseguire le operazioni computazionali molto più efficientemente della CPU. Inoltre molti meno dati sono trasferiti tra CPU e GPU e questo è di vitale importanza, ad esempio, quando è necessario visualizzare ampie folle di avatar realistici. Questo è ancora più importante quando il sistema di visualizzazione consiste in una serie di proiettori multipli che utilizzano una rete telematica.

Durante l'inizializzazione, quando vengono caricati gli Avatar, le informazioni delle *mesh* ed i *morph* target sono caricanti dentro Vertex Buffer Objects (VBO) OpenGL sulla GPU.

[28] http://www.cs.upc.edu/~bspanlang/animation/avatarslib/doc/
[29] http://home.gna.org/cal3d/

HALCA inoltre permette di riutilizzare i dati di vertici ed immagini: ad esempio, due avatar che condividono la stessa *mesh* ma hanno *textures* differenti sfrutteranno lo stesso VBO con le corrispondenti *texture maps*. Questa funzionalità è molto utile quando è necessario un gran numero di VH simili tra loro. Per le animazioni in modalità *shader*, HALCA deve unicamente trasferire le trasformazioni dei giunti scheletrali dalla CPU alla GPU come matrici di trasformazione o quaternioni doppi. (Kavan et al., 2007)

Per le animazioni HALCA estende la libreria del mixer astratto di Cal3D e aggiunge le seguenti funzionalità:

- Riproduzione e *blending* delle animazioni su diversi avatar a differenti velocità
- Riproduzione e *blending* parziali (in termini di tempo) delle animazioni
- Riproduzione e animazione delle animazioni solo su parti del corpo specifiche
- Manipolare una animazione non solo in termini di tempo ma con valori esterni; queste animazioni vengono chiamate *morph animations*, da non confondere con i *morph targets*
- Accedere e manipolare direttamente le rotazioni e traslazioni dei giunti scheletrali relative ad animazioni in fase di *blending* o *morphing*
- Accedere e manipolare efficientemente l'intero stato dello scheletro (basta chiamare un'unica funzione per accedere allo scheletro e posizionarlo in una particolare posa specificando un vettore di stato di un VH da HALCA)
-

```
#ifndef AVATARS_S3D
#define AVATARS_S3D
var Avatars;
var skinShader;

//#define DEBUGAVATARS


function InitAvatars()
{
#ifdef DEBUGAVATARS
Avatars = CVmExternDLL( "HALCAWin32_d.dll" );
OutPutLn("Loaded Avatars debug dll");
#else
Avatars = CVmExternDLL( "HALCAWin32.dll" );
#endif

    Avatars.__AddFunction( C_VOID, "Idle"); //updates animations
    Avatars.__AddFunction( C_VOID, "IdleOne",C_INT);    //updates animation of character with AvatarID
    Avatars.__AddFunction( C_INT, "addCharacter",C_PCHAR,C_PCHAR);  // adds a new character first parameter directory, second, cfg name
    Avatars.__AddFunction( C_INT, "loadShaders",C_PCHAR,C_PCHAR,C_PCHAR);//loads fragment and vertex shader and returns shader linker log
    Avatars.__AddFunction( C_VOID, "Draw",C_FLOAT); //draws all avatars
    Avatars.__AddFunction( C_VOID, "DrawExtShader"); //draws all avatars with External shader
    Avatars.__AddFunction( C_VOID, "DrawExtShaderOne",C_INT); //draws avatar with the given avatarID using the external shader
    Avatars.__AddFunction( C_VOID, "drawBoundsAndSkeleton"); //draws skeleton and bounding boxes of all avatars, useful when using external shader.
    Avatars.__AddFunction( C_INT, "getProgramID"); //Gets the ID of the current shader program
    Avatars.__AddFunction( C_VOID, "loadUniAndAttrIDs",C_INT); //Gets the ID of the current shader program
    Avatars.__AddFunction( C_VOID, "setDT",C_FLOAT); //set delta t for animations

    Avatars.__AddFunction( C_FLOAT, "getAnimationDuration",C_INT,C_INT);// AvatarID, AnimationID returns the duration of the specified animation in seconds
    Avatars.__AddFunction( C_INT, "getAnimationCount",C_INT);// AvatarID, returns the number of animations that the avatar AvatarID has loaded.
    Avatars.__AddFunction( C_PCHAR, "getAnimationFilename",C_INT,C_INT);// AvatarID, AnimationID, returns the filename of the animation specified by AvatarIDn and AnimationID
    Avatars.__AddFunction( C_PCHAR, "getAnimationName",C_INT,C_INT);// AvatarID, AnimationID, returns the name of the animation specified by AvatarIDn and AnimationID
    Avatars.__AddFunction( C_FLOAT, "getAnimationTime",C_INT,C_INT);// AvatarID, AnimationID, returns the time at which the given animation is currently played
    Avatars.__AddFunction( C_VOID, "setAccumulativeRoot",C_INT,C_INT); //AvatarID, 0 or 1 for on or off
    Avatars.__AddFunction( C_INT, "getAnimationId",C_INT,C_PCHAR); //AvatarID, AnimationName

    Avatars.__AddFunction( C_VOID, "useTextureUnit",C_INT); //specifies the texture Unit that the avatar rendering use for diffues and alpha textures specified in their materials


    Avatars.__AddFunction( C_VOID, "setDTOne",C_INT, C_FLOAT); //AvatarId, DeltaT, sets deltaT for individual Avatar
    Avatars.__AddFunction( C_FLOAT, "getDTOne",C_INT); //AvatarId, returns currently set DeltaT for this Avatar

    Avatars.__AddFunction( C_VOID, "ShutDown"); //removes all characters and animations etc.
    Avatars.__AddFunction( C_VOID, "exeAct",C_INT,C_INT,C_FLOAT,C_FLOAT,C_FLOAT, C_INT);// AvatarID, AnimationID, inTime, outTime, weight, lock
    Avatars.__AddFunction( C_VOID, "exeActAt",C_INT,C_INT,C_FLOAT,C_FLOAT,C_FLOAT, C_INT,C_FLOAT);// AvatarID, AnimationID, inTime, outTime, weight, lock, startTime
    Avatars.__AddFunction( C_VOID, "exeActPart",C_INT,C_INT,C_FLOAT,C_FLOAT,C_FLOAT, C_INT,C_FLOAT,C_FLOAT);// AvatarID, AnimationID, inTime, outTime, weight, lock, startTime
    Avatars.__AddFunction( C_VOID, "removeAct",C_INT,C_INT);// AvatarID, AnimationID
    Avatars.__AddFunction( C_INT, "isExecuting",C_INT,C_INT);// AvatarID, AnimationID,

    Avatars.__AddFunction( C_VOID, "blendCycle",C_INT,C_INT,C_FLOAT,C_FLOAT); // Blends in an animation with the parameters:AvatarID, AnimationID,Weight, Delay
    Avatars.__AddFunction( C_VOID, "clearCycle",C_INT,C_INT); // Removes an Animation from the currently bleded ones with the Parameters: AvatarID, AnimationID
    Avatars.__AddFunction( C_INT, "isCycling",C_INT,C_INT); // Returns if the animation with parameres AvatarID, AnimationID is blended in at the moment
    Avatars.__AddFunction( C_INT, "setMorph",C_INT,C_INT,C_FLOAT);  // sets the animation AnimationID of character AvatarID to morph value
    Avatars.__AddFunction( C_INT, "incMorph",C_INT,C_INT,C_FLOAT);  // increases the animation AnimationID of character AvatarID by morph value
    Avatars.__AddFunction( C_INT, "addMorph",C_INT,C_INT);  // adds animationID of character AvatarID to Morphable animations.
    Avatars.__AddFunction( C_INT, "removeMorph",C_INT,C_INT);   // removes animationID of character AvatarID from morphable animations.


```

Figura 9 Estratto della libreria HALCA

Questo può essere utile ad esempio per l'integrazione con metodi di sistemi di *motion capture* real-time. Un accesso efficiente allo stato dello scheletro inoltre può essere utile per motori fisici che controllino e rispondano alle collisioni tra VH o tra un arto ed il corpo virtuale. Solo a partire dallo stato scheletrico sono stati creati per HALCA diversi algoritmi di Cinematica Inversa scritti in S3D (Jesper *et al.* 2008).

Per ogni VH le diverse animazioni possono essere specificate nell'apposito file di configurazione in formato cfg in modo da poter essere utilizzate a proprio piacimento

```
1  ################################################
2  #
3  # Cal3d cfg File
4  #
5  ################################################
6
7  scale=1.0
8
9  ################# Skeleton #################
10 skeleton=Skeleton.XSF
11
12 ################# Meshes #################
13 mesh=m008_hipoly.XMF
14
15 ################# Animations #################
16 animation=Neutral.XAF
17 animation=A.XAF
18 animation=AE.XAF
19 animation=AO.XAF
20 animation=Y.XAF
21 animation=ch.XAF
22 animation=l.XAF
23 animation=m.XAF
24 animation=n.XAF
25 animation=r.XAF
26 animation=v.XAF
27 animation=w.XAF
28 animation=bodymove2.XAF
29 animation=Neutral.XAF
30 animation=Happiness.XAF
31 animation=Anger.XAF
32 animation=Fear.XAF
33 animation=Sadness.XAF
34 animation=Disgust.XAF
35
36 ################# Materials #################
37 material=m008_hipoly_1.XRF
38 material=m008_hipoly_2.XRF
39 material=m008_hipoly_3.XRF
40
41
```

Figura 10 Esempio del file di configurazione relativo ad un personaggio usato per SpeakToMe

XVR e HALCA hanno già provato di essere una accoppiata vincente nella generazione di ambienti virtuali che adoperino avatar (Normand *et al. 2012*). Tuttavia si riscontrava la mancanza di animazioni facciali realistiche e delle funzionalità relative al *lip-sync*. Per questo si è pensato di lavorare su questi due fronti al fine di avere Avatar e Agenti Virtuali Conversazionali che possano essere più realistici e quindi maggiormente sfruttabili nel contesto di esperimenti e applicazioni che sfruttino la presenza umana.

4.1.3 SAPI

L'interfaccia di programmazione SAPI (Speech Application Programming Interface) di Windows include al suo interno le funzioni relative al riconoscimento vocale e al Text To Speech (TTS), si è pensato quindi di sfruttare tali funzioni ed integrarle in XVR con il supporto della libreria HALCA, in quanto per queste architetture parevano mancare funzioni relative al *lip sync*.

SAPI, infatti, aiuta enormemente a ridurre l'ammontare di codice necessario per implementare tali funzioni, rendendo così queste tecnologie più accessibili e robuste in svariate applicazioni. L'API costituisce una interfaccia di alto livello tra una applicazione e motori per il parlato ed implementa tutte le funzioni di basso livello necessarie alla manipolazione real-time delle funzioni relative al parlato. SAPI ha due motori principali:

- TTS: che sintetizza stringhe e file di testo in una traccia parlata tramite l'uso di una voce sintetica (con la eventuale possibilità di differenti voci). Il TTS può essere controllato in SAPI tramite l'interfaccia Component Object Model (COM) ISpVoice. Una volta creato un oggetto ISpVoice sarà sufficiente chiamare la funzione ISpVoice::Speak per generare un output parlato a partire da dati testuali. Inoltre l'interfaccia ISpVoice contiene diversi metodi che permettono di cambiare voce e proprietà attinenti come ad esempio la velocità locutoria o il volume. Possono poi essere inseriti altri controlli speciali che cambino in real-time assieme al testo, ad esempio la voce, il timbro, l'enfasi sulle parole, velocità e volume. Questo tipo di markup per la sintesi, sapi.xsd, usa un formato XML standard e rende così estremamente semplice personalizzare il TTS indipendentemente dallo specifico motore o voce in uso. Il metodo ISpVoice::Speak può operare sia in modo sincrono, restituendo un valore di ritorno solo al suo completamento, sia in modo asincrono, restituendo un valore e attivando il parlato come processo in background. In caso di utilizzo del metodo asincrono possono essere rilevate informazioni come ad esempio la posizione nel testo o lo stato del parlato (ISpVoice::GetStatus), inoltre è possibile aggiungere nuovo testo sia interrompendo quello corrente, sia appendendolo una volta concluso il testo. Inoltre sono presenti molte altre interfacce COM per applicazioni più avanzate. SAPI comunica con le applicazioni inviando eventi usando il meccanismo standard di *callback* di Windows. Per

quanto riguarda il TTS questi sono usati solitamente per la sincronizzazione dell'output. Inoltre si possono sincronizzare altre azioni come i limiti delle parole, fonemi o visemi, che andranno inizializzate con COM specifici come ISpNotifySource, ISpNotifyCallback etc.

Possono inoltre essere impostate pronunce specifiche per parola per la sintesi vocale usando metodi come ISpContainerLexicon, ISpLexicon, ISpPhoneConverter. Esiste inoltre la possibilità di effettuare ricerche e selezione dati come file vocali e lessico e di indirizzare l'output audio verso destinazioni specifiche come telefoni o hardware personalizzato.

- Riconoscimento vocale: la funzione di riconoscimento vocale elabora un testo a partire da dei dati audio. Cosí come per la sintesi vocale l'interfaccia COM principale è ISpVoice, per il riconoscimento vocale l'interfaccia sarà ISpRecoContext. Allo stesso modo si avrà un ISpEventSource che permetterà all'applicazione di ricevere notifiche per gli eventi di riconoscimento richiesti. Si può scegliere tra due differenti tipi di ISpRecognizer; per la maggior parte delle applicazioni sarà più indicato un riconoscitore condiviso che possa essere eventualmente condiviso con altre applicazioni di riconoscimento. Per fare questo è sufficiente chiamare la funzione IspRecognizer::CoCreateInstance sul componente CLSID_SpSharedRecoContext. In questo caso SAPI imposterà l'input *stream* settandolo come il suo audio di default. Per applicazioni che utilizzino server di dimensioni maggiori, che funzionino autonomamente, dove la performance è essenziale, è più appropriato un *recognizer* InProc: è necessario chiamare la CoCreateInstance sul componente CLSID_SpInprocRecoInstance, poi chiamare la funzione SetInput del COM ISpRecognizer ed ingine creare un ISpRecoContext con ISpRecognizer::CreateRecoContext.

  Il passo successivo consiste nell'impostare notifiche per gli eventi che ci interessano, ISpRecognizer è anche un ISpEventSource, il quale è un ISpNotifySource, quindi si potrà chiamare uno dei suoi metodi dal proprio ISpRecoContext in modo da indicare dove gli eventi andranno espressi. Quindi bisognerà settare gli eventi da notificare con ISpEventSource::SetInterest. L'evento più importante è SPEI_RECOGNITION, che indica il riconoscimento del parlato per un determinato contesto

Infine per la produzione di testo sarà necessario creare una determinata grammatica che indichi il tipo di proposizioni da riconoscere. Attualmente per questo non si è ritenuto necessario utilizzare le funzioni per il riconoscimento vocale in quanto per creare un *lip sync* visivamente credibile era sufficiente il riconoscimento dei fonemi tramite il software della Annosoft che permette di individuare i fonemi e la loro rispettiva durata a partire da un file audio e una simulazione di *lip sync* tramite le basilari posizioni di apertura e chiusura della bocca.

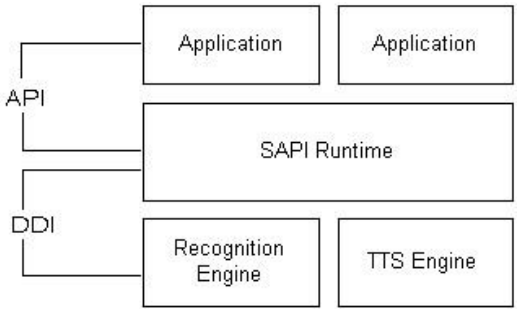


Figura 11 Struttura di SAPI

4.1.4 Annosoft voice recognizer tool

Per gli scopi di una architettura che includa VH conversazionali si assume che le tracce con audio realistico siano selezionate accuratamente ed inserite nelle risorse relative all'applicazione. Per questo motivo si è pensato di utilizzare un software robusto che analizzi la traccia e ne annoti le tempistiche in un file di testo. Per questo motivo è stata utilizzata la versione *text-based* di Annosoft Lipsync SDK 4.0[30].

Annosoft occupa una posizione molto importante nelle tecnologie di *lip sync* automatico per lo sviluppo di videogiochi e prodotti multimediali, includendo tra i suoi acquirenti i produttori di famosi giochi come Red Faction: Guerilla (THQ, 2009), Saint's Row 2 (THQ, 2008), Infamous (Sucker Punch Productions, 2009), Fable 2 (Microsoft Games, 2008), Rage, Harry Potter and the Half-Blood Prince (Electronic Arts, 2009) etc. L'SDK è stato inoltre sfruttato ampiamente inoltre per scopi di ricerca (Leff et al. 2013; Pajorová e

[30] Annosoft real-time lip-sync SDK (see http://www.annosoft.com/microphone-lipsync).

Hluchý, 2012; Jia *et al.* 2012) in quanto fornisce una marcatura accurata della fonte audio (nel nostro caso era sufficiente il tempo di inizio/fine della parola ed il fonema corrispondente) e la traduce in formato testuale facilmente integrabile in molte applicazioni.

```
1  audio prova.wav
2  phn 0 2150 75 x
3  word 2150 2630 outward
4  phn 2150 2280 75 AW
5  phn 2280 2346 75 t
6  phn 2346 2455 75 w
7  phn 2455 2564 75 AH
8  phn 2564 2630 75 d
9  word 2630 3160 bound
10 phn 2630 2785 75 b
11 phn 2785 2972 75 AW
12 phn 2972 3066 75 n
13 phn 3066 3160 75 d
14 phn 3160 5890 75 x
```

Figura 12 estratto della struttura del file prodotto a partire da una traccia audio da annosoft lipsync tool

4.1.5 OpenAL

OpenAL (Open Audio Library) è una interfaccia software per l'hardware audio che fornisce numerose funzioni che permettono di specificare gli oggetti e le operazioni per produrre un output audio di alta qualità, in particolare sistemi per il posizionamento di fonti audio 3D multicanale attorno ad un utente. L'API openAL è inoltre di facile utilizzo ed implementazione sulle diverse piattaforme. La sintassi usata dalla API è quanto più simile a quella usata da OpenGL e ne rispetta lo stile di codice e le convenzioni. L'intento principale degli ideatori è quello di generare un audio in un ambiente tridimensionale simulato, per questo motivo include estensioni compatibili con le linee guida di *rendering* IA-SIG[31] di livello 1 e 2 per gestire le fonti sonore dirette, l'attenuazione dovuta alla distanza e l'effetto Doppler, oltre ad effetti come la rifrazione del suono, l'ostruzione, la

[31] http://www.iasig.org/

trasmissione ed il riverbero; non sono invece supportati direttamente concetti sonori come il *panning* e i canali destro/sinistro. Come OpenGL, il nucleo di OpenAL non ha informazioni sul contesto di *rendering* esplicito e opera nel contesto OpenAL corrente implicito. Diversamente dalle specifiche OpenGL, tuttavia, viene incluso il nucleo dell'API ed il sistema di connessione al sistema operativo di ALC API (Audio Library Context API). Inoltre, diversamente da GLX WGK e altri BINDINGS specifici per il sistema operativo, ALC API è portabile sulle diverse piattaforme. (Kreimeier, 2001)

Il progetto che ha portato alla implementazione di OpenAL come API complementare di OpenGL nasce nel 1998, ma solo nei primi anni del 2000 viene rilasciata la specifica 1.0 assieme alle relative librerie per Linux, MacOS 8/9, Windows e BeOS. Nel tardo 1999, infatti, la Loki Entertainment software[32] aveva necessità di una API esattamente di questo tipo e di una implementazione per Linux e aveva iniziato a parlare con Creative Labs[33] di una standardizzazione della API e relativa espansione delle piattaforme supportate.

Loki Entertainment ha sviluppato in seguito diversi giochi utilizzando OpenAL, come ad esempio Heavy Gear 2 e Heretic (entrambi sotto Linux). Nel 2001 vengono rilasciate le prime librerie *hardware-accelerated* da parte dei Creative Labs, che supportano SoundBlaster Live su MacOS 8/9 e Windows. Dal 2001 ci sono state continue migliorie a OpenAL, sebbene alcune piattaforme, come BeOS e MacOS 8/9 non siano più rilevanti come nel 2000, molte altre piattaforme sono state incluse, ad esempio BSD, Solaris, IRIX, Mac OS X e le più importanti console di gioco. Per quanto riguarda l'*hardware*, OpenAL ha esteso il supporto per molti dispositivi audio Creative e NVIDIA anche sotto Windows ed è stato usato per un grande numero di prodotti su diverse piattaforme[34] (Garin, 2005).

4.2 SPEAKTOME: L'ARCHITETTURA

Per realizzare le funzioni relative al parlato emozionale ho creato una composizione di moduli software che permettesse di utilizzare e gestire le tre funzioni relative al *lip sync*, le

[32] Loki Entertainment Software, Interactive Entertainment, 1998 - http://lokigames.com/
[33] Creative Technology Limited - Consumer electronics, Singapore 1981
[34] come si può vedere nella lista all'indirizzo http://www.openal.org/titles.html

animazioni emozionali basate su FACS e le funzionalità necessarie a creare una applicazione da poter poi personalizzare per lo scopo desiderato.

Oltre alle funzioni ho realizzato una applicazione che include una versione nella quale le varie funzionalità (input testuale, input *wave*, real-time ed espressioni) possano essere provate manualmente con l'uso della tastiera e input tramite una semplice interfaccia HTML ed una seconda che presenti in automatico, tramite uno *storyboard* XML, tutte le funzioni per un determinato arco di tempo o in relazione al testo o al file adoperato. La prospettiva è quella di creare un primo ambiente di *testing* che potrà poi essere personalizzato con diversi VH, voci e testi, per la creazione di esperimenti o interviste virtuali di svariato tipo.

4.2.1 Creazione delle animazioni

I modelli adoperati per questo scopo sono selezionati dalla libreria Rocketbox[35] e sono stati animati con tre differenti tipi di animazione:

- 12 Visemi: sono stati selezionati e modellati manualmente sulla base della serie estesa di Preston-Blair 12 espressioni facciali corrispondenti ad altrettanti Visemi, ritenuti fondamentali per una corretta simulazione del parlato; tali visemi verranno poi mixati tramite HALCA in modo da avere un *lip sync* più armonioso, o eventualmente con le diverse espressioni facciali.
- FACS: le espressioni sono state create tramite il software commerciale Faceshift studio[36] un software che consente *motion capture* di alta qualità tramite l'utilizzo del sensore della Microsoft Kinect Il software prevede inoltre un plug-in che permette di trasferire, tramite Autodesk Maya le animazioni registrate su un modello che possieda un *rigging* facciale adeguato (come nel nostro caso), o legandole a determinate *blend-shapes*.
-

[35] http://www.rocketbox-libraries.com/
[36]http://www.faceshift.com

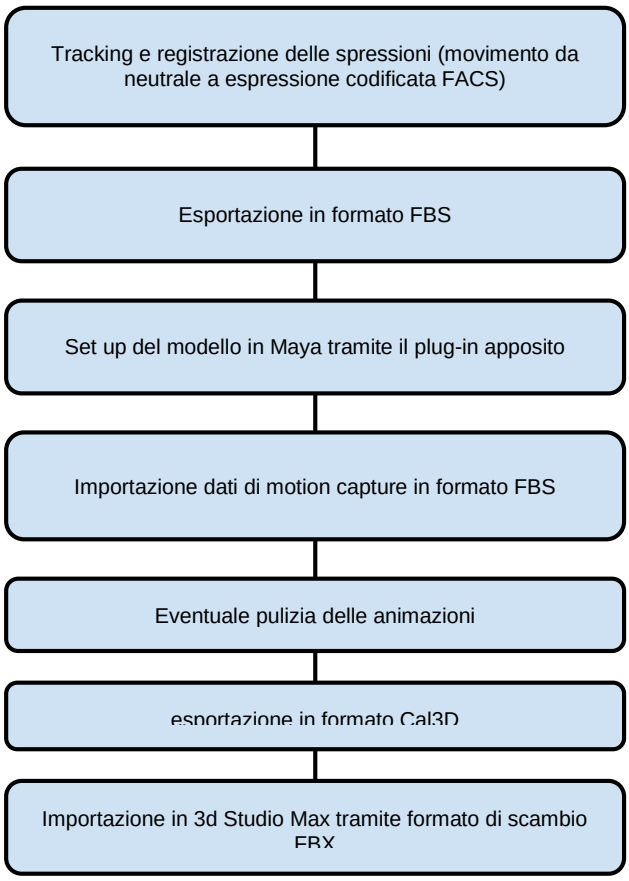


Figura 13 Pipeline di creazione delle espressioni in faceshift

- Per quanto riguarda il movimento del corpo si è usato un file bvh dalla libreria di animazioni catturate con *motion capture* dalla Carnegie Mellon University[37].

4.2.1 Sintesi vocale e DLL per SAPI

Per sfruttare le funzioni messe a disposizione da SAPI è stata creata un modulo DLL[38] esterno per XVR che permettesse di manipolare gli eventi relativi all'oggetto COM ed in seguito adoperarla all'interno dell'applicazione. Prima di andare avanti è utile analizzare nel dettaglio il modulo DLL:

Inizialmente vengono inclusi alcuni file di intestazione (*header files*):

[37] http://mocap.cs.cmu.edu/
[38] Dinamic-link Library

- windows.h: contiene tutte le funzioni che consentono di usare le API di Windows, le macro più comunemente usate e tutte le strutture dati usate dalle varie funzioni (viene limitata con la definizione WIN32_LEAN_AND_MEAN che esclude i servizi di raro utilizzo)
- math.h: contiene le macro, costanti e dichiarazioni di funzioni principali per le operazioni matematiche
- sapi.h: la più importante per lo scopo di questo progetto, include le funzioni relative alla Speech API di Windows
- sphelper.h: include alcune funzioni di supporto come ad esempio quelle per il *debugging* e le segnalazioni di errore
- initguid.h: include funzioni specifiche relative al sistema e ai driver in uso
- simpleDLL.h: semplifica il processo di esportazione della DLL

Vengono poi inizializzate le costanti e variabili che verranno usate

Dalle varie funzioni, come ad esempio gli oggetti COM ISpVoice ed iSpObjectToken.

DllMain: alla creazione viene specificato un punto di accesso in modo che processi e sotto processi possano comunicare con la DLL.

```
BOOL APIENTRY DllMain( HANDLE hModule,
                       DWORD  ul_reason_for_call,
                       LPVOID lpReserved
                              )
{
      switch (ul_reason_for_call)
      {
      case DLL_PROCESS_ATTACH:
      case DLL_THREAD_ATTACH:
      case DLL_THREAD_DETACH:
      case DLL_PROCESS_DETACH:
            break;
      }
    return TRUE;
}
```

Vengono poi dichiarate le funzioni che verranno implementate ed esportate

```
extern "C"
{
      __declspec(dllexport) int Init();
      __declspec(dllexport) int GetNumVoices();
      __declspec(dllexport) int SetRate(int r);
      __declspec(dllexport) int SetVolume(int V);
```

```
	__declspec(dllexport) int SetVoice(int V);
	__declspec(dllexport) int Speak(char *txt, DWORD flags);
	__declspec(dllexport)   int   GetStatus(SPVOICESTATUS   *viseme,   WCHAR
**ppszLastBookmark);
	__declspec(dllexport) int Exit();
	__declspec(dllexport) char* GetVoiceDescription(int id);
	__declspec(dllexport) int myGetEvents(int *duration, int *viseme);
}
```

e SplitFileName(char* filename, char *mypath, char *name) una funzione di supporto interna per gestire i percorsi dei file in formato stringa.

Le funzioni implementate sono così formate:

- Init(): è la funzione che avvia il ciclo che consentirà di rilevare il testo o l'audio desiderati ed i vari eventi collegati; rileverà inoltre il numero di voci disponibili se installate assieme a SAPI nel sistema e predisporrà le strutture relative a percorsi dei *files* e testi.
- GetNumVoices(): permette di ottenere il numero di voci rilevate nella Init().
- GetVoiceDescription(int id): permette di ottenere il nome identificativo della voce indicata da un determinato ID numerico.
- SetRate(int rate): permette di impostare la velocità della voce in un intervallo stabilito da SAPI da -10 a 10, dove 0 rappresenterà la velocità standard della voce, restituendo un valore negativo in caso il parametro indicato sia fuori dal *range* consentito o si sia verificato qualche errore precedente.
- SetVolume(int vol): permette di regolare il volume della voce prescelta in un intervallo da 0 a 100, la percentuale sul volume consentito dal sistema, restituendo un valore negativo in caso il parametro indicato sia fuori dal *range* o si sia verificato qualche errore precedente.
- SetVoice(int num): permette di impostare la voce desiderata e contraddistinta da un determinato ID differente da quello in uso,
- myGetEvents(int *duration, int *viseme): è la funzione che consente di collegare al VH, permette infatti di ottenere la durata ed il visema che vengono prodotti durante la sintesi vocale modificando i puntatori input.

- Speak(): è la funzione che permette di avviare l'output audio a partire da un testo, secondo la voce impostata ed in modo asincrono per consentire al contempo la segnalazione degli eventi (nel nostro caso visemi).
- Exit(): termina il ciclo che consente la sintesi vocale.

4.2.2 Riconoscimento vocale di un file wave

A partire dal testo contenente i fonemi e la loro tempistica, prodotto con il software annosoft, viene creata la funzione di lettura di un file *wave*.

Durante il ciclo di inizializzazione di XVR (nella funzione OnInit()) viene caricato il file di testo relativo all'annotazione del file audio e la traccia audio stessa all'interno dell'oggetto CVmVRAWav creato precedentemente.

All'avvio della funzione (che può avvenire tramite l'interfaccia grafica, la tastiera o dallo *storyboard*) viene attivato l'audio desiderato, viene mappato ogni fonema al visema corrispondente e avviata la funzione viseme(duration, viseme) che dato un determinato identificativo per visema e una durata riproduce l'animazione relativa, *blendandola* armonicamente con la precedente che verrà rimossa tramite le funzioni di HALCA *blendcycle* e *clearcycle*. La mappatura tra fonemi e visemi rispetta lo schema visto in precedenza al capitolo 3.6.

4.2.3 Simulazione riconoscimento vocale in real-time

Per la funzione di riconoscimento vocale in tempo reale attualmente non vengono utilizzati i visemi ma viene simulato variando il livello dell'apertura della bocca del VH, ovvero una delle caratterizzazioni basilari per la simulazione del *lipsync*, già adoperata spesso in letteratura e nell'industria (ad esempio Kawashima, 2010). Ho ritenuto per questa funzione che fosse sufficiente, agli scopi di questa relazione, implementare una simulazione basilare ma che consentisse una buona performance e l'utilizzo di più fonti streaming contemporanee, al fine di poter essere integrata in ambienti di copresenza virtuale.

All'interno di *SpeakToMe* tale funzione viene implementata utilizzando OpenAL. Vengono inizializzati inizialmente un numero di buffer pari al numero di fonti che si vogliono

utilizzati e si abilita l'audio 3d di OpenAL in modo che la collocazione spaziale sia credibile.

In seguito, nel ciclo OnInit() di XVR viene enumerato il numero dei dispositivi di acquisizione acustica con EnumAudioStreamingDevices(), viene impostato quello relativo al microfono (tipicamente corrispondente all'identificativo numerico 0) nella creazione di un oggetto cvmVRAWAVE e avviata la riproduzione dello streaming, che sarà poi richiamata all'avvio della funzione dentro la funzione OnTimer() di XVR. Dentro questo ciclo, richiamato tramite l'*input* da tastiera, pulsante dell'interfaccia grafica o *storyboard*, viene aggiornata la fonte di acquisizione per l'oggetto/fonte desiderato e rilevato il livello di frequenza, che sarà usato per variare la percentuale di apertura della bocca interpolando la percentuale di *blending* dell'animazione di massima apertura della bocca (in questo caso il visema relativo a A) a quella neutrale.

Il mio prospetto per il futuro è di affinare la funzione di *lip-sync* in tempo reale sfruttando le funzioni fornite da SAPI, che richiederebbero però in questo caso una grammatica ben definita per ogni linguaggio e una fase di training iniziale che dovrebbe essere il più rapida possibile in modo da non pesare troppo nella preparazione d'uso all'applicazione.

4.2.4 La classe STMtools

FirstCharacter.s3d | *STMtools.s3d

```
#ifndef STMTOOLS_S3D
#define STMTOOLS_S3D

//! Keyboardmanager is a class that eases to link keyboard events with function calls
/*
*/

class CVmSTMtools
{
// Inizializza SpeakToMe
Init();
//a partire dal nome dell'avatar lo carica tramite HALCA e restituisce il corrispettivo id.
AddAvatar(av_name);
//Carica uno storyboard in formato xml a partire dal percorso di un file e restituisce l'identificativo corrispondente
LoadStoryboard(xml_path);
///avvia lo storyboard selezionato tramite il suo identificativo.
RunStoryboard(story_id);
///carica l'animazione desiderata su un avatar, a partire dagli identificativi di avatar e animazione
RunAnimation(av_id, av_anim);
// a partire da un percorso o una stringa avvia la funzione di sintesi vocale
SpeakText(av_id, file_path);
SpeakText(av_id, text);
// a partire dal percorso di un file audio lo riproduce sincronizzandolo all'avatar prescelto
SpeakWav(av_id, wav_path);
//avvia il riconoscimento vocale sincronizzato in tempo reale all'avatar prescelto
SpeakLive( av_id);


};
```

Figura 14 Implementazione della interfaccia per la classe STMtools

Figura 5. La classe STMtools all'interno dell'editor di XVR.

Per gestire le funzioni di *SpeakToMe* ho realizzato una classe separata che contenesse tutte le funzioni per la gestione del *lip sync* e delle animazioni, che possa essere facilmente integrata in altre applicazioni da sviluppatori per gli usi desiderati. Si compone delle seguenti funzioni:

- Init(): Inizializza nelle sue diverse funzioni il programma, ad esempio: carica la DLL, gli avatar che potranno poi essere selezionati e gli scenari.
- AddAvatar(av_name): a partire dal nome dell'avatar desiderato lo carica tramite HALCA e restituisce eventualmente il corrispettivo identificativo.
- LoadStoryboard(xml_path): Carica uno *storyboard* in formato xml a partire dal percorso di un file e restituisce l'identificativo corrispondente
- RunStoryboard(story_id) avvia lo *storyboard* selezionato tramite il suo identificativo. Questa e la precedente funzione sono particolarmente utili alla creazione di scenari interattivi, dove una scelta dell'utente può determinare un differente scenario e sarà necessaria dunque una nuova narrazione
- RunAnimation(av_id, av_anim) carica l'animazione desiderata su un avatar, a partire dagli identificativi di avatar e animazione, questa funzione è usata principalmente per cambiare il tono emotivo della conversazione tramite le animazioni facciali emozionali
- SpeakText(av_id, file_path)
- SpeakText(av_id, text): a partire da un percorso o una stringa avvia la funzione di sintesi vocale e relativo *lip sync*
- SpeakWav(int av_id, string wav_path): a partire dal percorso di un file audio lo riproduce sincronizzandolo all'avatar prescelto
- SpeakLive(av_id): avvia il riconoscimento vocale sincronizzato in tempo reale all'avatar prescelto

4.2.5 Interfaccia manuale di testing e storyboard automatico

L'architettura di *SpeakToMe* si compone di due differenti versioni: una, usata da come ambiente di test per le diverse funzioni che permette di avviare manualmente le diverse

funzionalità del programma e una che carica uno *storyboard* in formato xml e fornisce una panoramica delle funzioni implementate. Questa versione potrà eventualmente essere personalizzata facilmente per gli scopi più svariati cambiando i vari elementi quali: personaggi, testi da leggere, audio da caricare, espressioni da inserire e l'ordine degli stessi nella sequenza.

Lo script principale è costituito in questa maniera rispettando i cicli previsti da XVR e descritti sopra:

Prima di iniziare ad effettuare i cicli nello script vengono settati alcuni parametri relativi a camere, luci e funzioni di OpenAL.

Vengono poi inizializzate le variabili globali e le costanti utili alle diverse funzionalità di *SpeakToMe* e definite alcune funzioni di supporto all'applicazione:

- cleanCycle(in limit): dato un limite che indica se devono essere eliminate tutte le espressioni o solo alcune il elimina eventuali visemi o espressioni emozionali *blendati* con l'animazione neutrale dell'avatar senza interferire con l'animazione del corpo.
- startFlag(string source) avvia la funzione specificata nella stringa che indica se si tratta della funzione di *stream* audio o *wave* file
- changeEmotion(string emotion) cambia espressione a partire da una stringa con il nome del visema
- viseme(duration, letter): avvia il *blending* di un visema con un codice identificativo selezionato e rimuovendo il precedente.
- reduceNVisemes(string viseme_code) mappa i fonemi ai 12 visemi.
- phonemeToCode(string phoneme): a partire da un fonema restituisce l'identificativo corrispondente in termini di visema
- AlignAvatarBoneToCamera(av_id, bone, avatar_obj): permette allo sguardo e la testa dell'avatar di seguire la camera, ovvero il punto di vista dell'utente

  Nella funzione OnDownload() viene inclusa la DLL vista in precedenza che contiene le funzioni relative a SAPI.

Viene poi avviato il ciclo di inizializzazione di XVR (la funzione OnInit())) nel quale:

- Vengono settati eventuali parametri specifici di posizione per la camera e gli oggetti relativi alla scena corrente.

- Viene caricata la DLL in una variabile contenente l'oggetto CVmExternDLL() e aggiunte le relative funzioni.
- Vengono poi settati i parametri della voce iniziale desiderata e rilevato il numero di voci disponibili.
- Viene inizializzato HALCA e caricati i personaggi desiderati (nel nostro caso solo uno) ed impostate diverse caratteristiche basilari ad essi relative
- Vengono caricati e segmentati i file di testo utili al riconoscimento vocale
- Vengono inserite le funzioni relative ai comandi attivabili tramite la tastiera
- Viene caricato il file *wave* e aperto il flusso per lo streaming

Nel ciclo OnFrame() vengono disegnati l'avatar e la scena nella quale esso è inserito assieme ad una animazione iniziale che presenterà l'ambiente di *testing*. É essenziale che il disegno di ambiente e avatar avvenga in questa funzione in quanto l'unica relativa all'output grafico. Viene inoltre impostato un testo di console da stampare a video che contiene le informazioni relative alle voci disponibili ed ai comandi attivabili tramite la tastiera. Vengono inoltre attivate le funzioni di navigazione della scena tramite mouse e l'allineamento dello sguardo dell'avatar alla camera, questo fornisce un ulteriore elemento di empatia e comunicazione verbale efficace come dimostrato dalla letteratura in merito ed illustrato nel paragrafo 2.5.5 (Colburn *et al.* 2000, Baileson *et al.* 2001).

Gli eventi relativi alle animazioni vengono avviati all'interno della funzione OnTimer():

- In caso di input attraverso l'interfaccia grafica viene acquisito l'input inviato col *form* e segmentato in modo da identificare l'identificativo della voce selezionata ed il testo da sintetizzare e viene riprodotta l'animazione per il corrispondente visema ogni qualvolta la funzione implementata dalla DLL lo segnala.
- In caso si inneschi tramite il pulsante appropriato la funzione per il riconoscimento vocale da file la corrispettiva funzione provvederà a riprodurre la traccia audio e le animazioni corrispondenti utilizzando l'array nel quale i dati su identificativo del visema e rispettiva durata sono stati memorizzati in fase di inizializzazione.
- In caso di *lip sync* in real-time tramite input da un microfono viene rilevato il livello della fonte sonora ed in base ad esso scalata l'apertura della bocca, usando le animazioni corrispondenti all'assenza di visemi o del visema A.

Infine nella funzione onExit viene terminata l'esecuzione della DLL (la funzione viene eseguita alla chiusura della finestra del player o del browser in cui è integrata).

Per la prima versione è presente una interfaccia grafica html (che utilizza il player su Windows Explorer) nella quale si offre la possibilità di selezionare un numero da 1 a 6 che identifica le differenti voci installate assieme a SAPI e un testo che verrà sintetizzato e sincronizzato con il personaggio presente sullo schermo. La schermata indica anche le funzioni dei comandi azionabili da tastiera:

- w: avvia la riproduzione e sincronizzazione del file *wave* con la mimica facciale dell'avatar
- s: avvia la funzione di streaming tramite il microfono dell'utente
- n: imposta l'espressione neutrale
- a: imposta l'espressione di rabbia (anger)
- s: imposte l'espressione di tristezza (sadness)
- h: imposta l'espressione di felicità (happiness)
- f: imposta l'espressione di paura (fear)
- d: imposta l'espressione di disgusto (disgust)

La versione scriptata, invece, si compone nella seguente maniera:

stato 1: carica una espressione realistica iniziale sull'avatar e lascia 5 secondi di tempo per osservarla

stato 2: avvia il *lipsync* di un audio selezionato caricando preventivamente il file testuale e la traccia relativi

stato 3: una volta terminata la traccia audio carica un testo tramite l'XML e lo sintetizza sincronizzandolo con il VH

stato 4: viene cambiata l'espressione dell'avatar e dato modo di osservarla per 5 secondi

stato 5: una volta trascorso il tempo utile ad osservare l'espressione viene attivata la funzione di riconoscimento vocale in real time per un tempo di 30 secondi

```
<Storyboard id="0" level="0" >
    <Scene>
        <Environments>
            <Environment id="salotto" file="salotto7.AAM"/>
        </Environments>
        <Avatars>
            <Avatar id="m008" file="m008.cfg"/>
            <Avatar id="f001" file="f001.cfg"/>
        </Avatars>
    </Scene>
    <Actions>
        <Action id="change_emotion"     av_id="m008"    anim="anger"                                    start="0"       duration="5000"         />
        <Action id="speak_wave"         av_id="m008"    source="wave"   file="prova.wave"               start="6000"    duration="65000"        />
        <Action id="speak_txt"          av_id="f001"    source="file"   file="prova.txt"    voice="0"   start="73000"   duration="20000"        />
        <Action id="change_emotion"     av_id="f001"    anim="happiness"                                start="95000"   duration="5000"         />
        <Action id="speak_live"         av_id="m008"    source="stream"                                 start="100000"  duration="30000"        />
    </Actions>
</Storyboard>
```

Figura 15 Semplice Storyboard XML implementato come demo di SpeakToMe

4.3 PUBBLICAZIONI

Dall'architettura e gli studi adoperati in questa tesi sono derivati due esperimenti relativi all'utilizzo di Virtual Human con espressioni facciali realistiche in ambienti virtuali immersivi che hanno portato alla redazione di due articoli: *The Influence of an Interactive Context on Emotion Recognition from Faces in Immersive Virtual Environment* (Faita, Vanni, Ruffaldi, Carrozzino, Tanca and Bergamasco, submitted) in attesa di pubblicazione e *Perception of Basic Emotions from Facial Expressions of Dynamic Virtual Avatar* (Faita, Vanni, Lorenzini, Carrozzino Tanca and Bergamasco, 2015) pubblicato in *Augmented and Virtual Reality: Second International Conference* (Salento AVR 2015).

Lo scopo del primo progetto era quello di valutare la percezione delle emozioni basilari in Dynamic Virtual Avatar (DVA) riproducendo la vivacità e l'intensità delle emozioni codificate nel FACS di cui sopra. Per questo sono state utilizzate le espressioni create con Faceshift e la Kinect camera e adattate manualmente tramite i software dedicati. Per questa valutazione è stato sviluppato un esperimento nel quale si richiedeva ai partecipanti di assegnare un punteggio da 1 a 5 per ognuna delle emozioni suscitate dalla visione di una particolare espressione simulata sul viso del VH.

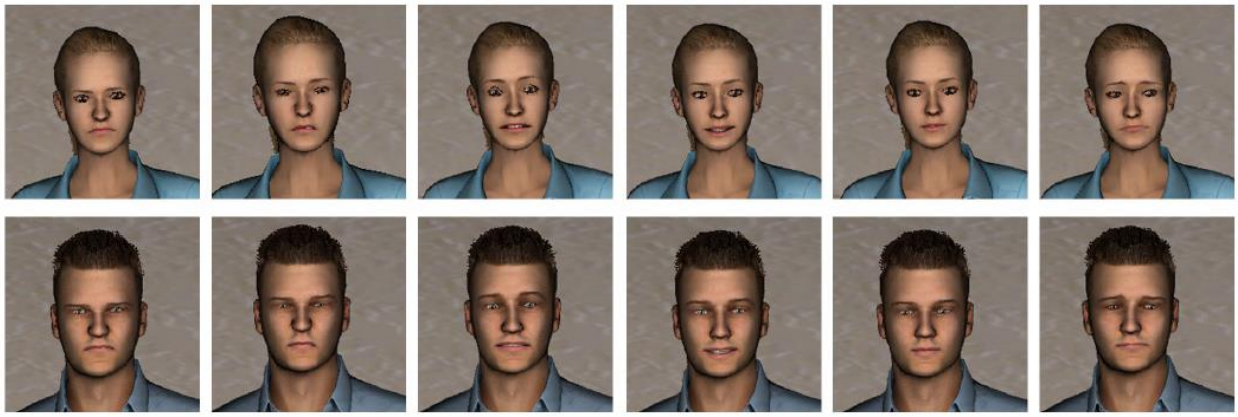


Figura 16 Le espressioni del set EKMAN+N usate negli esperimenti

Lo scopo del secondo progetto era invece quello di valutare l'impatto ambientale degli ambienti virtuali sul riconoscimento di emozioni veicolate nell'animazione delle espressioni di un VH. L'uso di espressioni standard, riconoscibili e realistiche era in questo caso estremamente importante al fine di evitare il fattore *uncanny*, che avrebbe distorto la percezione globale dell'ambiente. Per questo esperimento sono stati creati tre differenti ambienti emozionali ed indagato il loro effetto sulle emozioni create all'interno del progetto oggetto di questa tesi. Per questa valutazione è stato inoltre fatto uso della tecnologia immersiva Oculus-Rift DVK1

5. TEST

5.1 PERCEPTION OF BASIC EMOTIONS FROM FACIAL EXPRESSIONS OF DYNAMIC VIRTUAL AVATAR

Per questo esperimento sono stati selezionati 16 soggetti (7 di sesso femminile e 9 di sesso maschile) con un'età compresa tra i 29 ed i 31 anni e con un livello di istruzione uguale o superiore a una laurea triennale. Agli utenti è stato chiesto di leggere e firmare la liberatoria al consenso informato nella quale veniva brevemente spiegato l'esperimento e di rispondere ad un questionario sulla autovalutazione della propria competenza all'uso del computer (che è risultata essere in media generale di circa 4.87 su una scala di Likert di 5 punti), di video-games (media 2,5) e di VR (in media 2,5). Sulla base dei risultati ottenuti il gruppo è stato diviso in due sottogruppi: uno di utenti esperti (Exp, con media 3,75) e uno di principianti (Lay con media 1,25)

Prima dell'esperimento è stata descritta dettagliatamente all'utente la procedura sperimentale della sessione. Inoltre gli utenti sono stati sottoposti alla versione italiana del questionario PANAS (Positive and Negative Affect Schedule) (Terraciano *et al.*, 2003) per valutare il loro stato emotivo al momento dell'esperimento.

I partecipanti sono poi stati posizionati davanti a un computer portatile da 17 pollici a una distanza di 60 centimetri, sul quale hanno assistito a 24 animazioni facciali dalla serie EKMAN+N, in una sequenza casuale su due cicli in cui veniva utilizzato un avatar maschile oppure femminile. Ogni animazione è stata mostrata per 5 secondi con un tempo di passaggio da 1 a 4 tra l'espressione neutrale e quella target. Dopo ogni animazione, essi potevano assegnare un punteggio da 1 a 5, per valutare la corrispondenza tra l'emozione percepita e quelle codificate nella serie EKMAN+N, tramite di una schermata apposita di selezione. L'ordine delle etichette cambiava casualmente all'inizio di ogni ciclo.

L'interrogativo principale era la comprensione della capacità dei visi virtuali mostrati nell'esperimento di trasmettere le emozioni basilari e se esse potessero essere riconosciute da osservatori adulti. Per questo sono state isolate due variabili:

- Il tempo necessario per assegnare il punteggio, per capire quanto fosse difficile per i partecipanti percepire l'emozione sui visi virtuali
- Quanto il punteggio si avvicinava all'emozione che l'applicazione intendeva trasmettere o alla espressione neutrale (mancanza di emozioni).

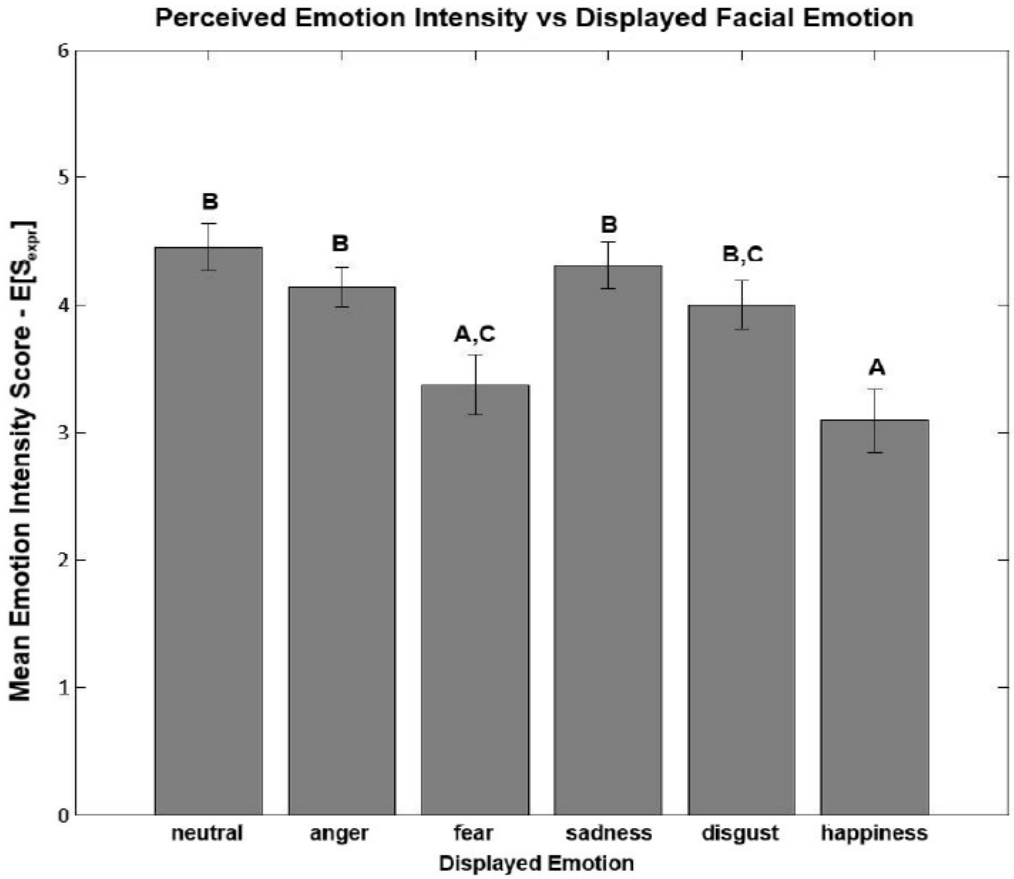


Tabella 1 Grafico che illustra il rapporto tra il punteggio assegnato e l'emozione illustrata rispetto alla percepita.

I risultati hanno dimostrato che, nonostante si assuma una maggiore facilità al riconoscimento di espressioni in visi virtuali da parte di utenti che abbiano familiarità con la realtà virtuale, non sono state ritrovate differenze significative in termini di tempo di riconoscimento nei due gruppi (Expert e Lay). Questo suggerisce che la familiarità con il dispositivo non sia un fattore di influenza per il riconoscimento delle espressioni.

Si è riscontrata invece una significativa differenza di tempo a seconda delle espressioni, in particolare tra l'espressione di rabbia, che richiedeva un tempo di circa dai 19 ai 23 secondi e quella di felicità che richiedeva solo una media di circa 13 secondi. Tuttavia, in termini di punti assegnati, l'espressione che trasmette felicità presenta un picco molto basso; questo sembrerebbe oltretutto essere in contrasto con i risultati ottenuti in precedenza che mostrano che l'espressione è solitamente riconosciuta con alti livelli di accuratezza ed intensità (Spencer-Smith *et al.*, 2001) Tuttavia si deve considerare che la presenza di 4 espressioni negative ed una sola positiva potrebbe influire sull'umore dei partecipanti e fargli dunque percepire un minore livello di intensità di tale emozione.

Il punteggio più alto è stato rilevato per espressione neutrale (circa 4.45 su un massimo di 5 assegnabile). Questo risultato potrebbe dipendere dal fatto che alle persone esaminate era stata chiaramente specificata la possibilità di trovarsi davanti ad una espressione di questo tipo. Sebbene studi precedenti abbiano infatti mostrato un alto pattern di errore nel riconoscere l'espressione neutrale (Dyck *et al*, 2008), difficilmente tale espressione veniva confusa con una totalmente differente, veniva piuttosto assegnata in caso di incertezza sull'espressione da riconoscere. Nell'esperimento presentato, tuttavia, è stato specificato di assegnare l'espressione neutrale solo in caso di completa mancanza di una espressione riconoscibile.

Un risultato inaspettato si è avuto per l'espressione di disgusto (circa 4 su un massimo 5 assegnabile). Molta della letteratura in merito ha dimostrato una particolare difficoltà nel trasmettere l'emozione di disgusto in VH in quanto è particolarmente difficile riprodurre l'arricciamento alla base del naso (Spencer-Smith *et al.* 2001; Kohler *et al.*, 2003; Moser *et al.*, 2008; Dyck *et al.*, 2008). Tali studi avevano modellato l'espressione lavorando sulle differenti unità muscolari del personaggio virtuale partendo da una espressione statica estratta dalla serie FACS. Nella ricerca presentata l'acquisizione diretta tramite una imitazione realistica eseguita da un attore ha semplificato il processo di animazione ed incrementato il valore emozionale delle espressioni, si ritiene dunque che tale procedura abbia accresciuto l'intensità emozionale per il disgusto.

Per le rimanenti espressioni i risultati si sono mostrati essenzialmente in linea con i precedenti studi in materia.

5.2 The Influence of an Interactive Context on Emotion Recognition from Faces in Immersive Virtual Environment

In questo caso prima di sottoporre gli utenti all'esperimento, essi sono stati invitati a rispondere ad un questionario al fine di raccogliere i loro dati anagrafici (età, genere, stato civile, professione) e alcune informazioni sulla salute visiva e generale e sulla familiarità con l'uso di computer, videogames e sistemi di VR. I partecipanti sono stati, inoltre, anche in questo caso sottoposti al questionario PANAS. Per quanto riguarda le informazioni relative al senso di presenza e al realismo sono state poste inoltre 5 domande tratte da IPQ[39]

La variabile primaria in questo caso era l'accuratezza nel riconoscimento delle emozioni nei visi virtuali o della mancanza di esse, calcolata sulla base del numero di identificazioni corrette nel pannello di scelta multipla che compariva dopo ogni espressione ri arrangiato in maniera casuale. Una mancata identificazione veniva contata come una identificazione errata.

É stata condotta una analisi della varianza (ANOVA) tra la variabile Group come fattore intersoggettivo a due livelli (Desktop e HMD) e Face come fattore intra soggettivo a sei livelli (uno per ogni espressione) utilizzando i dati raccolti nell'ambiente neutrale per valutare se vi fosse una differenza di identificazione e accuratezza nella percezione in un ambiente immersivo rispetto a uno non immersivo. In questo caso non si sono riscontrate differenze significative.

In seguito è stata condotta un'ANOVA tra la variabile a tre livelli Context (Neutrale, Negativo e Positivo) ed il tempo di riconoscimento del viso per capire se ci fosse una correlazione significativa tra il tipo di contesto e l'emozione percepita. É stata usata una correzione di Huynh-Feldt in quanto il test di Mauchly ha fatto emergere che i dati violavano l'assunzione di sfericità. Sia prima che dopo la correzione si è riscontrato un valore significativo del contesto, sia per quanto riguarda l'accuratezza del riconoscimento, sia sul tempo in confronto ai risultati nell'ambiente neutrale (77-78% nei due ambienti caratterizzati a fronte di 70% in quello neutrale).

[39] Igroup Presence Questionnaire (Schubert, 2003)

I punteggi più bassi ottenuti nei questionari sull'IPQ e sulla difficoltà nel riconoscere le espressioni sono stati confrontati nei due diversi contesti Negativo e Positivo ed entrambi con quello Neutrale tramite il test di Wilcoxon, ottenendo un punteggio di rilevanza del contesto del 5%.

Infine sono state condotte analisi sulla covarianza tra le variabili Context e Face come fattori intra-soggettivi ed i punteggi ottenuti nel PANAS, dell'autovalutazione sulla competenza informatica e analisi statistiche utilizzando Matlab Statistics Toolbox (MATLAB and Statistics Toolbox Release 2011b, The MathWorks, Inc., Natick, Massachusetts, United States) ed il linguaggio R 3.1.0 (R, 2014). In questi ultimi due casi non sono però stati rilevati risultati significativi

Dalle analisi descritte gli autori traggono le seguenti conclusioni:

- In un contesto neutrale l'immersione in un IVE non influenza il riconoscimento di espressioni che trasmettano emozioni, almeno in un contesto neutrale.
- In accordo alla letteratura in materia (Mobbs, 2006; Wallbott, 1988) il tipo di ambiente influenza il riconoscimento delle espressioni, sia esso negativo o positivo

Tuttavia l'effetto del contesto non è stato evidente in misura pari a quanto ci si aspettava inizialmente. Un motivo potrebbe essere una scarsa capacità degli ambienti creati nel veicolare le emozioni volute o nel creare la condizione influenzante di informazione situazionale (Carroll 1996).

5.3 TEST DELLE FUNZIONI DI LIP SYNC

Per quanto riguarda la parte relativa al *lip sync* si prospetta in futuro di fare dei test che ne validino la credibilità e l'impatto sugli utenti in ambienti virtuali. Precedenti studi sono infatti stati condotti, ma difficilmente mettono assieme espressioni, sincronizzazione col parlato, avatar realistici e un ambiente immersivo.

Ad esempio nello studio sugli *embodied conversional agent An Embodied Avatar Mediated Communication System with VirtualActor for Human Interaction Analysis* (Ishii 2007) le funzioni di comportamento non verbale sono limitate a espressioni del viso non

formalmente codificate e movimenti della testa, senza considerare lo sguardo e il sistema di codifica FACS come invece il sistema illustrato in questa tesi prevede.

Nello studio, Nudge *Nudge Wink Wink: Elements of Face-to-Face Conversation for Embodied Conversational Agents* (Cassell, 2000), invece, sebbene si valuti la reazione al cambiamento di espressione, si ritiene tuttavia che sarebbe interessante, come nella demo presentata osservare un cambiamento di espressione associato a una conversazione, in modo da dare più colore al dialogo con l'agente virtuale.

Inoltre i casi visti sono stati effettuati in ambiente desktop, ci si prospetta, invece di integrare l'ambiente creato in contesti immersivi come Oculus Rift e ambienti CAVE-*like*. Questo permetterà di valutare in che modo il sistema adoperato può essere migliorato e sfruttato al meglio per creare avatar emozionali di alto livello che creino il massimo livello di empatia con l'utente.

6. CONCLUSIONI E SVILUPPI FUTURI

L'obiettivo principale di questa tesi era in origine analizzare l'uso e le caratteristiche dei VH in ambienti virtuali immersivi e cercare di implementare una procedura che permettesse di gestire gli stessi in modo da migliorare il livello di empatia ed immersione. Questo è stato realizzato attraverso una attenta analisi della letteratura sia dal punto di vista delle caratteristiche di comportamento non verbale, sia da quello dello stato dell'arte dei prodotti presenti nella ricerca e sul mercato. L'analisi ha portato alla implementazione di un modulo software, *SpeakToMe* che integra animazioni facciali realistiche e le metodiche per un *lip sync* credibile, inoltre è stata predisposta una architettura che permetterà di creare le applicazioni più svariate consentendo di personalizzare lo *storyboard*, i personaggi adoperati, le eventuali animazioni corporee, le tracce audio, i file testuali ed il numero di fonti streaming (in caso si vogliano adoperare più personaggi che ricevano un input dal microfono). Questo permetterà di gestire un numero arbitrario di Avatar e Virtual Agent ed integrarli al meglio nello scenario desiderato per scopi specifici quali navigazione di scenari, training, sperimentazione e tutti i campi illustrati nei quali la presenza umana è vitale alla narrazione.

Si prospetta di migliorare ulteriormente le funzioni implementate creando un *lip sync* maggiormente realistico, specialmente riguardo il tempo reale dove attualmente non vengono usati i visemi ma unicamente le due posizioni basilari della locuzione (bocca aperta/chiusa, vedi paragrafo 2.5.6).

Inoltre si prospetta di integrare diversi *storyboard* esemplificativi che possano essere modificati e adoperati con semplicità per gli scopi desiderati, ad esempio in contesto sperimentale si potrà dare la possibilità di utilizzare i propri file e testi e l'ordine della sequenza.

Verranno condotti inoltre esperimenti sulle funzioni del *lip sync*, per valutare il livello di realismo e quali sono le caratteristiche essenziali per migliorarne la percezione da parte di utenti e sperimentatori in base al contesto nel quale vengono adoperate, in modo da poter eliminare elementi non necessari e migliorare la performance quando possibile.

Infine ci si propone di testare, oltre alla mimica facciale, le funzioni descritte con tecnologie di realtà virtuale immersive come Oculus Rif, ambienti Cave-like etc.

7. BIBLIOGRAFIA

Adams, Douglas. *Guida galattica per gli autostoppisti*. Edizioni Mondadori, 2012.

Adolphs Ralph, Recognizing emotion from facial expressions: psychological and neurological mechanisms, *Behavioral and cognitive neuroscience reviews 1.1*. 2002. pp 21-62.

Ahn, S. J., J. Fox, and J. N. Bailenson. Avatars. in *Leadership in Science and Technology: A Reference Handbook. SAGE Publications, Inc* 2012.

Astrid, M., et al., It doesn't matter what you are!" Explaining social effects of agents and avatars, *Computers in Human Behavior* 26.6, 2010, pp 1641-1650.

Bailenson, J., et al., Equilibrium theory revisited: Mutual gaze and personal space in virtual environments, *Presence* 10.6, 2001, pp 583-598.

Bailey, Mike; Cunningham, Steve. Graphics shaders: theory and practice. CRC Press, 2011.

Blascovich, Jim, and Cade McCall, Social influence in virtual environments, *The Oxford handbook of media psychology*, Oxford University Press, 2013, pp 305-315.

Cameron, James. Avatar [film*]. Los Angeles, CA: Twentieth Century Fox Film Corporation*, 2009.

Carrozzino, M., Tecchia, F., Bacinelli, S., Cappelletti, C., & Bergamasco, M., Lowering the development time of multimodal interactive application: the real-life experience of the XVR project, Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology, 2005, June, pp. 270-273. ACM.

Cassell, Justine, Nudge nudge wink wink: Elements of face-to-face conversation for embodied conversational agents, *Embodied conversational agents 1*, 2000.

Chikersal, Prerna, Pooja Sukheja, and Chng Eng Siong. Talking Head: Literature Review. http://prernachikersal.com/assets/pdfs/urecalitrev.pdf visitato Agosto 2015

Colburn, Alex, Michael F. Cohen, and Steven Drucker, The role of eye gaze in avatar mediated conversational interfaces, *Sketches and Applications, Siggraph'00*, 2000.

Darwin, C., *The expression of the emotions in man and animals:* D. Appleton and Company 1874.

Dick, Philip Kindred; Guasta, Maria Teresa. *Cacciatore di androidi*. Ed. Nord, 1995.

Dyck, M., et al., Recognition profile of emotions in natural and virtual faces, *PLoS One* 3.11 2008.

Ekman, P., & Friesen, W. V.,. Constants across cultures in the face and emotion. *Journal of personality and social psychology*, *17*(2), 1971, p 124.

Ekman, P., and W. Friesen. Facial action coding system: a technique for the measurement of facial movement, *Consulting Psychologists,* San Francisco, 1978

Ekman, P. *Are there basic emotions?*, 1992.

Ekman, P., Facial expression and emotion., American Psychologist, 48, 4, 1993, p 384

Ekman, Paul., Facial expressions. *Handbook of cognition and emotion* 16, 1999, pp 301-320.

Fabri, M., Moore, D., & Hobbs, D. Mediating the expression of emotion in educational collaborative virtual environments: an experimental study, *Virtual reality* 7.2, 2004, 66-81.

Faita, C., Vanni, F., Lorenzini, C., Carrozzino, M., Tanca, C., & Bergamasco, M., Perception of Basic Emotions from Facial Expressions of Dynamic Virtual Avatars, *Augmented and Virtual Reality* ., Springer International Publishing, 2015, pp. 409-419.

Faita,Claudia. Vanni, Federico. Ruffaldi, Emanuele. Carrozzino, Marcello. Tanca, Camilla e Bergamasco Massimo. The Influence of an Interactive Context on Emotion Recognition from Faces in Immersive Virtual Environment (submitted)

Fast, J., *Body Language*, New York, N.Y., Neirenberg, 1970

Freud, S.. The Uncanny, *The standard edition of the complete psychological works of Sigmund Freud,* Vol. 17, pp. 219–256,. London: Hogarth Press, 1919

Gahan, Andrew. *Game Art Complete*. Elsevier, 2008.

GIBSON, William. *Neuromante.* Edizioni Mondadori, 2014.

Gilbert, Richard L., Andrew Forney, Can avatars pass the Turing test? Intelligent agent perception in a 3D virtual environment, *International Journal of Human-Computer Studies* 73, 2015, 30-36.

Gutiérrez-Maldonado, José, Mar Rus-Calafell, and Joan González-Conde. Creation of a new set of dynamic virtual reality faces for the assessment and training of facial emotion recognition ability. *Virtual Reality* 18.1. 2014, pp 61-71.

Hale, Kelly S., and Kay M. Stanney, eds. *Handbook of virtual environments: Design, implementation, and applications*. CRC Press, 2014.

Hiebert, Garin. "Openal 1.1 specification and reference." 2005

Ho, C.-C., MacDorman, K. F. , Pramono, Z. A., Human emotion and the uncanny valley. A GLM, MDS, and ISOMAP analysis of robot video ratings," in Proceedings of the Third ACM/IEEE International Conference on Human--Robot Interaction, Amsterdam, 2008, pp. 169–176.

Hoon, Loh Ngiik, Wang Yin Chai, and Khairul Aidil Azlin Abd Rahman. Development of Real-Time Lip Sync Animation Framework Based On Viseme Human Speech. *디자인학연구* 27.4 , 2014, 19-28.

Ishii, Yutaka, and Tomio Watanabe. An embodied avatar mediated communication system with VirtualActor for human interaction analysis., *Robot and Human interactive Communication*, 2007. RO-MAN 2007. The 16th IEEE International Symposium on. IEEE, 2007.

Jentsch, E., On the psychology of the Uncanny. *Psychiat.-neurol. Wschr.*, 8(195), 219-21, 1906, pp 226-7.

Jia, X., Zhang, K., Han, Y., e Powers, D. Pronunciation quality evaluation approach based on bimodal fusion with noise adaptive weight, *Computing and Convergence Technology* (ICCCT), 2012 7th International Conference on IEEE, 2012, December pp. 1244-1247

Kageyama, Akira, and Youhei Masada., Applications and a three-dimensional desktop environment for an immersive virtual reality system., *Journal of Physics: Conference Series*. Vol. 454. No. 1. IOP Publishing, 2013.

Kavan, Ladislav. Collins, Steven . Zara, Jiri e O'Sullivan, Carol. Skinning with dual quaternions. *2007 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, pages 39–46. ACM Press, April/May 2007.

Kawashima, Hiroaki, and Takashi Matsuyama. "Interval-based modeling of human communication dynamics via hybrid dynamical systems." Workshop on Modeling Human Communication Dynamics at NIPS. 2010.

Kim, Juno, et al. "The Oculus Rift: a cost-effective tool for studying visual-vestibular interactions in self-motion perception." *Frontiers in psychology* 6 (2015).

Kouroupetroglou, Georgios, et al. "Using Wiimote for 2D and 3D pointing tasks: gesture performance evaluation." Gesture and Sign Language in Human-Computer Interaction and Embodied Communication. Springer Berlin Heidelberg, 2012. 13-23.

Kohler, C. G., Turner, T. H., Bilker, W. B., Brensinger, C. M., Siegel, S. J., Kanes, S. J., ... & Gur, R. C. Facial emotion recognition in schizophrenia: intensity effects and error pattern. *American Journal of Psychiatry.*(2003).

Knight, Barbara, and Alan Johnston. The role of movement in face recognition. *Visual cognition 4.3,* 1997, 265-273.

Kreimeier, Bernd. The Story of OpenAL. *Linux Journal 2001*.81 es, 2001, p 7.

Kubrick, S. 2001: *Odissea nello spazio* (1968).

Larsson, Pontus, Daniel Vastfjall, and Mendel Kleiner. Better presence and performance in virtual environments by improved binaural sound rendering.,*Audio Engineering Society Conference: 22nd International Conference: Virtual, Synthetic, and Entertainment Audio*. Audio Engineering Society, 2002.

Leff, Julian, et al. Computer-assisted therapy for medication-resistant auditory hallucinations: proof-of-concept study. *The British Journal of Psychiatry*, 2013, 202.6: 428-433.

Leppänen, Jukka M., and Jari K. Hietanen. "Positive facial expressions are recognized faster than negative facial expressions, but why?." Psychological research 69.1-2 (2004): 22-29.

Liu, Boqian. "GAME ASSET DEVELOPMENT PIPELINE WITH A FOCUS ON FACIAL RIGGING AND ANIMATION." (2012).

Martin, Gary C. "Preston blair phoneme series." World wide web electronic publication (2006).

Madracevic, Lana, and S. Šogorić. "3D Modeling From 2D Images." *MIPRO, 2010 . Proceedings of the 33rd International Convention*. IEEE, 2010. http://www.garycmartin.com/phoneme_examples.html. Visitato il 10/08/2015.

Magnenat-Thalmann, Nadia, and Daniel Thalmann. "Human Modeling and Animation." *Computer Animation*. Springer Japan, 1985..

Maletic, V. (1987). Body - Space - Expression: The Development of Rudolf Laban's Movement and Dance Concepts. Berlin: Walter de Gruyter & Co.

McCarthy, John, et al. "A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955." *AI Magazine* 27.4 (2006): 12.

Mead, M. (1995). Visual anthropology in a discipline of words. *Principles of visual anthropology*, *3*, 3-12.

Menache, Alberto. *Understanding motion capture for computer animation*. Elsevier, 2011.

Mobbs, D., Weiskopf, N., Lau, H. C., Featherstone, E., Dolan, R. J., & Frith, C. D. The Kuleshov Effect: the influence of contextual framing on emotional attributions. Social cognitive and affective neuroscience, 1(2), 95-106.(2006).

Moore, Carol-Lynne. *Movement and making decisions: The body-mind connection in the workplace*. Dance & Movement Press, 2005.

Mori, Masahiro, Karl F. MacDorman, and Norri Kageki. "The uncanny valley from the field." *Robotics & Automation Magazine, IEEE* 19.2 (1970 - 2012): 98-100.

Mortensen, Jesper. Yu, Insu. Khanna, Pankaj . Tecchia, Franco. Spanlang, Bernhard. Marino, Giuseppe e Slater, Mel . *Real-time global illumination for vr applications*. IEEE Computer Graphics and Applications, 28:56–64, 2008.

Moser, E., Derntl, B., Robinson, S., Fink, B., Gur, R. C., & Grammer, K.. Amygdala activation at 3T in response to human and avatar facial expressions of emotions. Journal of neuroscience methods, 161(1), 126-133. (2007).

Mu, Bo, YuHui Yang, and JianPing Zhang. "Implementation of the Interactive Gestures of Virtual Avatar Based on a Multi-user Virtual Learning Environment." *Information Technology and Computer Science, 2009. ITCS 2009. International Conference on*. Vol. 1. IEEE, 2009.

Muybridge, E. (1979). Muybridge's Complete Human and Animal Locomotion: All 781 Plates from the 1887 Animal Locomotion (Vol. 1-3): Dover Publications.

Naef, Martin, Oliver Staadt, and Markus Gross. "Spatialized audio rendering for immersive virtual environments." *Proceedings of the ACM symposium on Virtual reality software and technology*. ACM, 2002.

Nespor, Marina. "Fonologia. Le strutture del linguaggio." Bologna: Il Mulino(1993).

Normand, J.-M.. Spanlang B., Tecchia F., Carrozzino M., Swapp D. e Slater M., *Full body acting rehearsal in a networked virtual environmenta case study," Presence: Teleoperators and Virtual Environments*, vol. 21, no. 2, pp. 229-243, 2012.

Osipa J., "Stop Staring – Facial Modeling and Animation Done Right", 2nd Edition, Wiley Publishing, Inc., 2007

Pajorová, Eva, and Ladislav Hluchý. *Correct speech visemes as a root of total communication method for deaf people*."Agent and Multi-Agent Systems. Technologies and Applications." Springer Berlin Heidelberg, 2012. 389-395.

Preston Blair phoneme series: http://www.garycmartin.com/mouth_shapes.html visitato Agosto 2015

Ratner, Peter. *3-D human modeling and animation*. John Wiley & Sons, 2012.

Rohrer, Tim. "The body in space: Dimensions of embodiment." *Body, Language, and Mind: Embodiment. Walter de Gruyter, Berlin* (2007): 339-378.

Tanenbaum, Joshua, Magy Seif El-Nasr, and Michael Nixon. "Nonverbal Communication in Virtual Worlds: Understanding and Designing Expressive Characters." (2014).

Thalmann, Daniel. "The role of virtual humans in virtual environment technology and interfaces." *Frontiers of Human-Centered Computing, Online Communities and Virtual Environments*. Springer London, 2001. 27-38

Todd, D. *Poly-modeling with 3ds Max: Thinking Outside of the Box*. CRC Press, 2012.

Rech, Jörg, and Klaus-Dieter Althoff. Artificial intelligence and software engineering: Status and future trends. KI 18.3 (2004): 5-11.

Rost, Randi. OpenGL(R) Shading Language (2nd Edition) (OpenGL). Addison-Wesley Professional, 2006

Salisbury, J. K. Jr., Making Graphics Physically Tangible, *Communications of the ACM*, 42(8), 75-81, Agosto, 1999.

Salisbury, Kenneth, Francois Conti, and Federico Barbagli. Haptic rendering: introductory concepts. *Computer Graphics and Applications, IEEE* 24.2 (2004): 24-32.

Spanlang, B., & Slater, M. (2009). Full body avatar interaction., *Congreso Internacional de Interacción Persona Ordenador AIPO*.

Spencer-Smith, Jesse, et al. Making faces: Creating three-dimensional parameterized models of facial expression. *Behavior Research Methods, Instruments, & Computers* 33.2, 2001, pp 115-123.

Stebbins, G. (1977). Delsarte System of Expression. Brooklyn: Dance Horizons.

Stephenson, Neal. *Snow crash*. Spectra, 2003.

Wachoski, Andy; Wachoski, Larry. The matrix [film]. *USA: Warner Brothers Pictures*, 1999.

Terraciano A., McCrae R.R., e Costa Jr P. T., Factorial and construct validity of the italian positive and negative affect schedule (panas). *European Journal of Psychological Assessment*, vol. 19, no. 2, p. 131, 2003.

Todd, Emma. Fitzgerald, Marty . Mapping: An improved workflow for high-fidelity model textures and correlating output maps for high fidelity models., *Computer Games, Multimedia and Allied Technology* (CGAT 2012) (2012): 96.

Vinayagamoorthy, V., Steed, A. and Slater, M. (2005) Building characters: Lessons drawn from virtual environments, *Proceedings of Toward Social Mechanisms of Android Science*, COGSCI 2005, Stresa, pp. 119–126.

Wallbott, Harald G. In and out of context: Influences of facial expression and context information on emotion attributions. *British Journal of Social Psychology 27.4* 1988, pp 357-369.

Wei, Xiaolin, Peizhao Zhang, and Jinxiang Chai. "Accurate realtime full-body motion capture using a single depth camera." *ACM Transactions on Graphics (TOG)* 31.6, 2012, pp 188.

RINGRAZIAMENTI

Come sa bene chi mi conosce, per me non è semplice esprimere quello che sento intimamente in pubblica piazza, infatti forse questa è la pagina più difficoltosa della mia tesi e sicuramente se avrò modo e coraggio di rileggerla sarà una delle più grosse fonti di vergogna e contorsioni relative. Mi sembra però doveroso cogliere l'occasione per ringraziare chi, in un modo o nell'altro, mi è stato vicino nel percorso universitario e non qui a Pisa e mi ha permesso ed aiutato ad un traguardo per me estremamente importante.

Ringrazio innanzitutto il mio relatore e mentore Marcello Carrozzino per i consigli e il supporto didattico, ma anche morale, durante tutto il periodo del tirocinio e di stesura di questa tesi. Ringrazio inoltre tutto lo *staff* del PERCRO in particolare Chiara, Claudia, Cristian, Giovanni e Raffaello etc.

Desidero ringraziare inoltre, assieme a lui, tutti i docenti che hanno accresciuto il mio bagaglio culturale, e fatto appassionare anche a materie prima lontane dalle mie conoscenze e interessi e, forse inconsciamente, fornitomi gli strumenti per inseguire i miei sogni, in particolare Matteo Dellepiane, Enrica Salvatori e Marion Lamè.

Ringrazio inoltre tutti gli amici lontani e vicini che mi hanno aiutato sia nei momenti di difficoltà che in quelli di svago, facendomi divertire e ricaricare, ma anche condividere opinioni, sogni, passioni e pensieri. Penso che i risultati ottenuti siano merito non solo della formazione didattica ma frutto di ogni esperienza positiva o negativa che la accompagnata e in questo avete una grossa parte. Ringrazio in particolare:

La mia "*famigghia pisana*" che ha dovuto sorbirsi tutte le mie lamentele, paranoie e ansie in prima persona: Nicoletta e la sua empatia, perché quando qualcuno arriva a piangere al posto tuo le cose da dire rimangono poche davvero; Ciarlie, per lo spaccio di armi di piccantezza di massa e l'accompagnamento musicale (con tanto di *cantendi e pignendi*); Angelo (di cui mi tocca tacere il vero nome per educazione) per la sua spesso eccessiva ma sempre divertente ed efficace schiettezza e ad Angelica per l'associazione a delinquere nelle cospirazioni alimentari ad tasso di colesterolo; i due tornati ma mai partiti Giuseppe (che non vuol mai essere ringraziato e si becca il ringraziamento ufficiale, tié) e Silvia per il supporto morale, le consulenze e revisioni su tutto e la

sopportazione assieme a Nicoletta dei miei insulti che mi torneranno tutti, con interessi, dopo la laurea.

Ringrazio la *greffa* sassarese, lontana ma sempre presente, che mi ha (ri)accolto e trascinato in mezzo a una banda di matti e (senza forse rendersene conto) ha illuminato in questo modo uno dei periodi più bui della mia vita: Marialaura, perché quando sei al fronte serve sempre un compagno di barricata, per le serate bolognesi e per avermi insegnato a fare più domande SEMPRE; Valeria, perché i nostri aggiornamenti sono sempre travagliati ma infiniti e riescono sempre a estrarmi verità che spesso neanche io voglio ammettere; Basciu, per le mille serate pisane e il vuoto che ha lasciato a Pisa e nel mio divano; Stefania per l'ospitalità in via Organari a tutte le ore e condizioni; Anna perché ad ogni ritorno devo rinunciare alla voce; la mia probabile cugina Elena per i maglioni di babbo, le chiazze e le pasquette con radici sotto gli alberi, Andrea per il rifornimento di magliette e manicaretti da asporto; Federica per la comprensione e i consigli da donna-uomo a donna-uomo; quella matta di Ninni, Marcello e Maria Chiara.

La mia sorella di vita Elena perché da (oddio) 22 anni ormai mi è sempre stata vicina per quanti chilometri possano esserci tra noi condividendo panico, risate e ogni minima stupidaggine, finendo spesso per consolarmi facendosi consolare. Mio *marito* dalla nascita Laura, perché a volte per stare vicini e supportare non c'è bisogno di neanche una parola e l'*amante* Marina che invece spesso ha condiviso con me fiumi interminabili di filosofeggiamenti, sogni, chiacchiere e sfoghi vicendevoli. Claudia, mia compagna di sogni e utopista come me nel credere che prima o poi riusciremo a fare dei nostri sogni il nostro futuro. Ringrazio per questo anche Luigi, se anche i nostri sentimenti e pensieri hanno preso strade molto differenti, i bei ricordi, il supporto e la spinta al volere di più da se stessi sono qualcosa per il quale non posso non essere grata, nel bene e nel male.

Ringrazio Sara perché perchè non riusciamo a mantenere un litigio e qualsiasi cosa, per quanto possa essere triste, con lei finisce in lacrime, ma di risate (un saluto anche a paciocchino); Anna e Tony per gli scambi musicali, i giri di spalle, i "ti riposi quando muori" e i consigli sul vestire che non seguo mai; , Giorgio per il suo riuscire ad essere presente persino dal Giappone e per la comune passione per l'umorismo senza alcun senso; ringrazio con lui Luigi R. anche per la condivisione dall'adolescenza di hobby e passioni spesso non apprezzate abbastanza (qualcuno direbbe nerd o geek ma ormai il termine è diventato cosí modaiolo che mi fa un po' ribrezzo usarlo) e perché, come ha detto qualcuno "eravamo tutti strani, ma sono contento che fossimo cosí" assieme a lui Franzoni, Costantino, Emilio, Salvatore, Piga, Marco, Fanaro, Buscarinu, Gabriele e … Gabriele.

Ringrazio le mie coinquiline per aver condiviso gioie e dolori di Via Giordano Bruno e aver sopportato la mia acidità mattutina, i miei orari improbabili e disordine, in particolare Sonia per le scorribande

e aver cercato (inutilmente) di instillare in me un pochino di ordine e signorilità e Martina per la condivisione di avventure, conoscenze, accampamenti in biblioteca e orari indicibili.

Grazie inoltre nuovamente ai miei corelatori segreti e non ufficiali: le prof. Asara, Manconi e Demuru, il prof Renna e mia madre.

Ringrazio per ultimo chi ha il merito più grande: la mia famiglia, che mi ha permesso di inseguire i miei sogni, spesso anche a costo del proprio sacrificio. Mia Madre, che mi ha sempre insegnato che i numeri e la matematica sono un bel gioco, che si possono affrontare anche le cose più difficili con un sorriso e una battuta pungente e che non ha mai smesso di esseri più che vicina, anche a distanza, in tutti i modi possibili. Alessio, il mio fratellone, confidente e consulente che tra chiacchiere infinite su libri, sogni, vita e filosofie mi ha fatto capire che l'amore, qualsiasi tipo di amore, non è mai sprecato e per quanto possa essere incompreso è qualcosa di tuo che nessuno può toglierti e spesso lo puoi usare come come motore per le cose più grandi. Nonna e Maria per il supporto costante, i consigli e critiche, perché anche quelle non devono mai mancare. Viola, Barbara, Tore, Maria Giovanna, Ilaria, Salvatore, Giuseppina, Giulia,Antonio e Eulalia.

Riservo le ultime righe di questi ringraziamenti e di questa tesi a mio padre, che purtroppo non potrà leggere queste righe e assistere a questo momento, nonostante sia per molti versi l'artefice principale di questo percorso e della persona che (con un pochino di fierezza) penso di essere diventata. Grazie per avermi sempre dato tutto, anche più di quanto potevi, senza che mai me ne potessi accorgere fino in fondo. Grazie per avermi insegnato a capire quali sono le cose importanti nella vita. Grazie per avermi insegnato che a volte la testardaggine può non essere solo un difetto, ma anzi un dono e che se vuoi davvero qualcosa "basta con queste… *stupidaggini* e falla".

Ringrazio anche tutti quelli che sicuramente ho dimenticato, se mi conoscete sapete bene che non è mancanza di riconoscenza ma la mia testa di *parabattula*[40]. Grazie di cuore per questo e per tutto quello che verrà.

Camilla

[40] Farfalla