ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA
CAMPUS DI CESENA

DIPARTIMENTO DI INFORMATICA — SCIENZA E INGEGNERIA
CORSO DI LAUREA IN INGEGNERIA E SCIENZE INFORMATICHE

Tecniche di animazione 3D nella realizzazione di un cortometraggio

---

Elaborato in
Computer Graphics

Relatore:
Prof. Damiana LAZZARO

Presentata da:
Leonardo MARINI

SESSIONE UNICA
Anno Accademico 2018 — 2019

«People gain a tremendous amount of wisdom when they travel…

get lost, and maybe you'll find yourself somewhere.»

Bryan Cranston

ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA
CAMPUS DI CESENA

Abstract

Dipartimento di Informatica — Scienza e Ingegneria
Corso di Laurea in Ingegneria e Scienze Informatiche

Laurea

Tecniche di animazione 3D nella realizzazione di un cortometraggio

di Leonardo MARINI

Questa tesi vuole mostrare il processo di produzione di un cortometraggio, da un punto di vista progettuale. Vengono quindi analizzate le fasi della produzione, come la modellazione e la realizzazione delle animazioni. Oltre ad esse sono trattate anche alcune fasi precedenti alla produzione, come la progettazione delle animazioni, e del cortometraggio in generale. Il focus principale sono quindi proprio le animazioni e la teoria dietro di esse. Infatti per capire come progettare e realizzare animazioni in maniera efficiente, si rende necessario capire come funzionino da un punto di vista scientifico e ingegneristico. A tale scopo, vengono illustrati alcuni concetti teorici delle animazioni, con alcuni richiami alla matematica, quali matrici e formule trigonometriche, ed altri di programmazione lineare, come il metodo del simplesso per risolvere un problema di ottimizzazione.

La scelta di realizzare un cortometraggio deriva dal fatto che questo è il prodotto che meglio rappresenta un lavoro sulle animazioni. Un altro approccio sarebbe potuto essere quello di realizzare un videogioco, ma in esso ci sono innumerevoli altri aspetti da progettare, e le animazioni non ricoprono altro che una minima frazione del prodotto finale. La storia dietro a questo progetto, così come tutti i personaggi, sono opera della creatività di Alberto Uras, caro collega con cui ho avuto il piacere di lavorare a questo progetto.

In questo progetto entrambi abbiamo preso parte in tutte le fasi, ognuno di noi ha però avuto più influenza su alcune di esse, nel mio caso le animazioni. Per progettarle e realizzarle è stato fatto uso di tecniche che mirano a semplificare il lavoro dell'animatore. Il risultato non è da considerare sorprendente: sono state utilizzate tecniche già da tempo consolidate. Il punto non è infatti quello di sperimentare nuove tecniche in assoluto. Ciò nonostante, le tecniche che sono state utilizzare erano comunque relativamente nuove per noi, in quanto nessuno di noi le aveva mai messe in pratica.

Ringraziamenti

La stesura di questa tesi è stata una bella sfida. Il doverla portare a termine lontano da casa mi ha messo a dura prova, soprattutto psicologicamente. Per questo motivo vorrei ringraziare tutti coloro che, nonostante i chilometri che ci dividessero, mi sono stati vicino. In particolare la mia famiglia, che mi ha supportato in ogni scelta, anche in quelle più rischiose, come appunto quella di partire per la Svezia prima di aver completato gli studi. Tutti i miei amici, ai quali non ho mai smesso di pensare. Ed infine, un grazie immenso alla mia relatrice, che mi ha aiutato a redarre questo documento in maniera meticolosa.

Indice

Elenco delle figure

Lista delle abbreviazioni

GCI	**C**omputer **G**enerated **I**magery
GC	**C**omputer **G**raphics
DOF	**D**egree(s) **O**f **F**reedom
FK	**F**orward **K**inematics
IK	**I**nverse **K**inematics

Capitolo 1

Introduzione

La computer grafica (CG) gioca un ruolo molto importante nell'ambito delle scienze informatiche. In particolare, negli ultimi anni si è visto un notevole miglioramento nelle tecniche di rendering fino ad arrivare a risultati di fotorealismo nei quali è praticamente impossibile riconoscere se un'immagine sia stata generata artificialmente (CGI), o se si tratti di una foto. Ovviamente questi progressi non sono limitati alle immagini statiche, anzi, trovano molte più applicazioni in sequenze video, dove queste immagini sono animate. Di fatto, i settori che maggiormente fanno uso di CGI sono quello cinematografico e quello videoludico.

Quello delle animazioni digitali, è un sotto settore della CG, tuttavia molto ampio. Questa tesi vuole quindi analizzare questo aspetto, in particolare le esistenti tecniche di animazione 3D, e il contesto in cui queste vengono utilizzate. Per quanto riguarda il contesto, come già detto sopra, quelli principali sono due: film e videogiochi.

Da qui deriva la scelta di realizzare un cortometraggio: la differenza principale tra i due settori, per quanto riguarda le animazioni, è che il primo utilizza animazioni *offline*, mentre per il secondo si parla di animazione in *tempo reale*. Ciò significa che nel primo caso è possibile sapere in anticipo esattamente quali azioni saranno da animare mentre, nel caso dei videogiochi, è l'utente finale a scegliere quali azioni fare ed in che ordine. Ciò rende più difficile progettare le azioni necessarie, che devono quindi essere spezzate in animazioni cicliche per rendere possibile transitare da un'animazione all'altra. Quest'ultimo aspetto complica ulteriormente le animazioni dei videogiochi: è particolarmente difficile transitare da un'animazione all'altra e rendere tale transizione fluida e realistica lo è ancora di più. Un altro aspetto, che ha favorito la scelta di realizzare un cortometraggio rispetto ad un videogioco, è che, da un punto di vista progettuale, il primo è costituito da poche semplici fasi, contrariamente ai videogiochi che racchiudono anche il *game-design*, ovvero rendere il gioco in sé giocabile. Sintetizzando, le animazioni rappresentano un aspetto molto piccolo nella realizzazione di un videogioco, mentre ricoprono la parte centrale nella realizzazione di film d'animazione.

L'obiettivo di un cortometraggio è quello di raccontare una storia semplice e a sè stante che possa, quindi, essere rappresentata come filmato dalla breve

durata. La creazione di un cortometraggio animato è quindi un processo che coinvolge l'intera pipeline grafica con l'aggiunta dell'ideazione di una storia da raccontare. L'intero processo di produzione può quindi essere suddiviso in 3 fasi principali: *pre-produzione*, *produzione* e *post-produzione* [15].

In questo caso le fasi differiscono leggermente da quelle proposte da K.Roy in [15]: modellazione e texturing qui vengono trattate come parte della produzione del trailer. Soltanto il rigging viene anticipato nella progettazione siccome va attentamente studiato in base alle animazioni che si vogliono realizzare. Come descritta da K. Roy, la produzione (Capitolo 5) comprende anche le fasi di animazione, illuminazione e rendering. Una descrizione più dettagliata di ogni sotto-fase verrà fornita nel rispettivo paragrafo.

Tuttavia, prima di illustrare come sia stato realizzato il corto, verranno spiegate le fasi di analisi e progettazione. Come per ogni progetto infatti, è indispensabile partire da un'attenta analisi (Capitolo 2) del problema per capire cosa si vuole realizzare e studiarne la sua fattibilità. Di conseguenza sarà necessario definire una metodologia, a questo proposito viene illustrata la progettazione (Capitolo 4), che ha l'obiettivo di studiare *come* il progetto verrà realizzato. Quest'ultima fase, come già detto, vuole analizzare l'aspetto metodologico del progetto. Per rendere possibile ciò, vengono quindi riportate le varie tecniche di animazione al capitolo precedente (Capitolo 3).

In ogni capitolo verrà data un particolare attenzione alla parte relativa alle animazioni. Sono infatti esse, l'oggetto principale di questa tesi, a differenza del corto, che rappresenta il prodotto finale: un'applicazione della teoria che sta dietro ai vari metodi di animazione, in un contesto lavorativo reale. Vorrei quindi sottolineare come questa tesi inverta il ruolo delle animazioni e del corto. Normalmente, le animazioni, ed in particolare i *rig* che permettono di animare un oggetto 3D, sono il mezzo che permette di raggiungere lo scopo (oggetto animato). In questo caso invece, è il corto a diventare un semplice mezzo attraverso il quale è possibile mostrare l'efficacia della di diverse tecniche di animazione.

Capitolo 2

Analisi

2.1 Requisiti

L'obiettivo di questo progetto è quello di realizzare un cortometraggio animato in 3D. Questo dovrà quindi rappresentare una storia. Non ci sono requisiti su ciò che possa essere rappresentato, quindi si è deciso per una storia di nostra immaginazione, senza aggiunta di particolare realismo.

Per la storia ne è stata scelta una scritta precedentemente da Alberto, il che ci ha permesso di concentrarci sulla realizzazione del corto, senza spendere tempo per l'ideazione della storia. Questa scelta ha quindi introdotto alcuni nuovi requisiti, come la realizzazione dei modelli dei personaggi e degli ambienti rappresentati nella storia, e la fedeltà allo storyboard realizzato dal mio collega.

Cosa più importante: il corto dovrà essere realizzato utilizzando diverse tecniche di animazione, ciascuna adatta al proprio dominio, al fine di mostrarne le diverse caratteristiche, i vantaggi e gli ambiti in cui ha senso utilizzarle.

Come requisito aggiuntivo, non funzionale, si è puntato ad avere animazioni quanto più realistiche possibili. Questo al solo scopo di avere un prodotto finale piacevole da guardare, e poter mostrare con orgoglio.

2.2 Vincoli

Visti i limiti di tempo, sia per la realizzazione del video, che per poterlo esporre, si è deciso di proseguire con la realizzazione di un trailer. Ciò non differisce molto da quella che era l'idea iniziale, e soddisfa i requisiti menzionati nella sezione precedente (2.1 Requisiti). Questa scelta offre anche il vantaggio di poter affiancare scene temporalmente distanti tra loro, e anche in ordine cronologico non lineare, per mostrare scene topiche in un crescendo di drammaticità.

Inoltre, in questo modo è stato anche eliminato il problema di doppiare i personaggi. Infatti in un trailer, è possibile utilizzare una colonna sonora di sottofondo, al posto dei dialoghi, generalmente per nascondere *spoiler* presenti in ciò che i personaggi dicono.

Infine, un vincolo fondamentale, e alla base di ogni processo di realizzazione di qualsiasi film, è quello di seguire fedelmente lo storyborad, fornito dal direttore artistico, in questo caso sempre Alberto Uras.

2.3 Organizzazione del lavoro

Come per ogni progetto, è importante definire una *tabella di marcia* in maniera tale da non dover rifare le cose più volte, e procedere da un passo all'altro con la consapevolezza di sapere a che punto della produzione ci si trova e, attraverso una stima, capire quanto manca al completamento del prodotto. Il lavoro è stato quindi suddiviso nelle seguenti task:

- Realizzazione e posizione delle scene principali, come key-frame fissi.
- Creazione di cicli di animazione da utilizzare in diverse scene.
- Implementazione delle animazioni cicliche nelle scene che ne necessitano.
- Aggiunta di frame *in-between* e correzione dei movimenti.
- Animazione dei dialoghi:
  - Creazione delle shape-keys per le diverse espressioni facciali.
  - Alternare le schape-keys per adattarle all'audio del dialogo.
  - Aggiunta delle animazioni facciali alle scene animate.

Capitolo 3

Concetti di animazione

Animare significa "muovere" un modello, altrimenti statico, nel tempo. Questi movimenti sono definiti attraverso delle trasformazioni geometriche (traslazione, rotazione e scalatura). Di seguito vengono riportate alcune tecniche di animazione 3D, spiegandone le metodologie, i pregi, i difetti e il contesto in cui possono venire utilizzate.

3.1 Rappresentazioni di rotazione

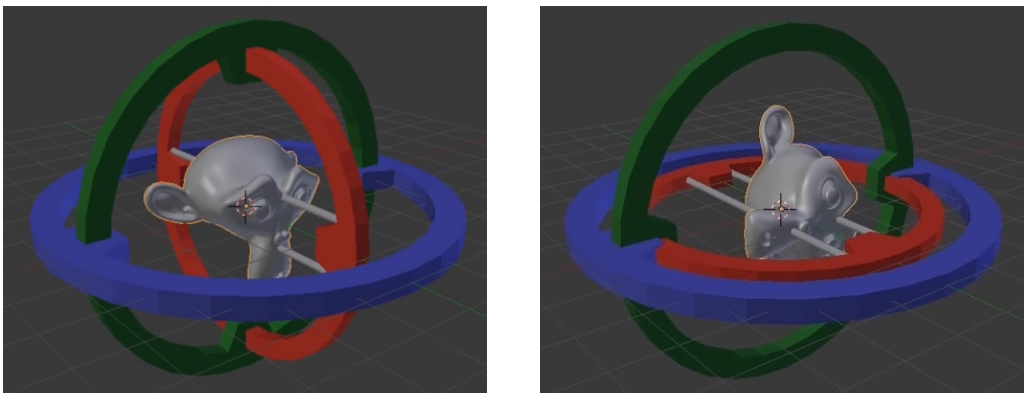
Lo scopo di questo paragrafo è descrivere le possibili rappresentazioni matematiche di una rotazione. Alcuni di questi concetti sono validi anche per altre trasformazioni. Tuttavia verranno considerate soltanto le rotazioni, in quanto queste ultime sono uno degli aspetti principali nella realizzazione di animazioni — in particolare di animazioni 3D — e alla base di alcune tecniche di animazione — *Inverse Kinematics* (IK) — come anche uno dei più complessi.

Come vedremo in seguito sarà indispensabile capire quale delle seguenti rappresentazioni conviene utilizzare in base all'animazione che si vuole realizzare.

3.1.1 Angolo-asse

Questo tipo di rappresentazione è senza dubbio la più semplice. Utilizza 4 valori: 3 per specificare l'asse, ed 1 per l'angolo. In questo modo, con una singola rotazione, è possibile raggiungere qualsiasi orientamento dell'oggetto che si sta ruotando. Così come esiste sempre una linea retta che collega due punti nello spazio, si può pensare ad una rotazione angolo-asse come una singola rotazione che collega due orientamenti.

Questa rappresentazione è ottima per rotazioni di articolazioni con un solo *Degree Of Freedom* (DOF). È anche possibile definire articolazioni con più di un DOF semplicemente concatenando n articolazioni su singolo asse connesse da $n - 1$ ossa (i.e. oggetti connessi in un'articolazione) di lunghezza 0. Ciò però rende la giuntura risultante inutilmente complessa da animare. Per questo motivo le rappresentazione con angoli di Eulero, o sotto forma



(A) Posizione neutra. (B) Gimbal lock, l'asse X e Z sono allineati.

(Immagini di Nathan Vegdahl)

FIGURA 3.1: Rappresentazione di rotazione Euleriana attraverso un giroscopio a tre assi.

di quaternione, sono quelle più spesso utilizzate. Inoltre la rappresentazione dell'asse attraverso 3 componenti numeriche non risulta di facile comprensione per l'utente che di solito preferisce usare quella Euleriana anche per rotazioni su singolo asse.

3.1.2 Euleriana

Concettualmente è la più intuitiva di tutte: utilizza 3 assi di rotazione (X, Y, Z) ed il funzionamento è analogo a quello di un giroscopio. Ogni asse offre un DOF, quindi sono possibili rotazioni con 3 DOF. Tuttavia sono necessarie due accortezze:

1. ordine degli assi;
2. gimbal lock problem.

L'ordine degli assi è decisivo, in quanto quello più interno dipende dalla rotazione di quelli esterni. Di conseguenza ruotando gli assi in un ordine diverso da quello specificato porta a risultati diversi da quello atteso. In più, interpolazioni tra diversi orientamenti, possono a loro volta risultare sgradevoli, poiché la rotazione viene spezzata in 3 movimenti.

Il problema del gimbal lock [19], in italiano blocco cardanico, sorge dall'allineamento di due assi: quello più interno e quello più esterno. Ne deriva che ruotando uno di questi 2 assi si ottiene la stessa rotazione, perdendo quindi un DOF. È quindi importante scegliere l'ordine degli assi in maniera tale che il primo e il terzo non risultino mai allineati.

Per ovviare a questo problema può essere conveniente bloccare uno degli assi in una posizione fissa. Per questo motivo, la forma Euleriana è meglio utilizzata nelle giunture con 1 o 2 DOF.

3.1.3 Quaternione

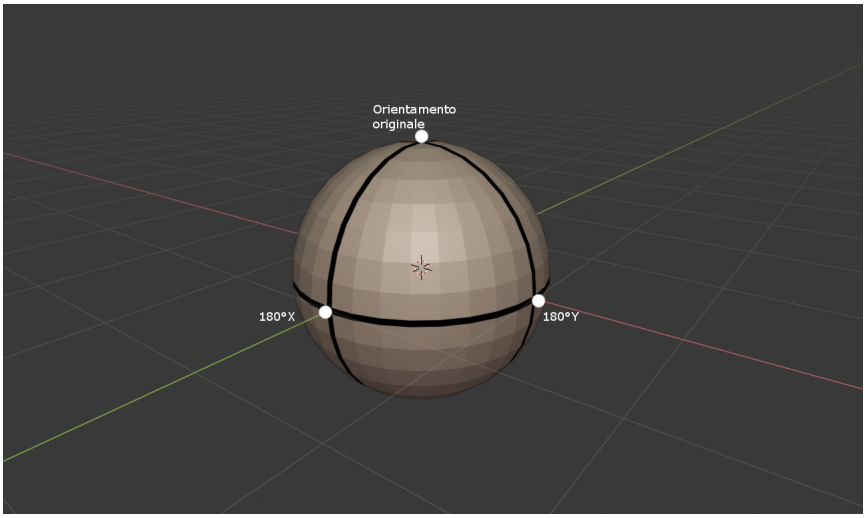


FIGURA 3.2: Rappresentazione di un quaternione 2D su una sfera 3D.

Diversamente da quanto appena visto, la forma di quaternione è concettualmente complessa, ma in pratica diventa molto utile e senza molti dei difetti della controparte Euleriana.

I principali benefici della rappresentazione a quaternione includono l'eliminazione del gimbal lock. Infatti è presente una componente in più, ma ciascuna componente non rappresenta un asse quanto piuttosto l'orientamento dell'oggetto ruotato intorno a quell'asse. La quarta componente serve, quindi, a definire la posizione neutra.

L'interpolazione risulta diretta e dolce, infatti il movimento non è spezzato sui diversi assi e l'ordine di questi non è importante. Infine rende comodo calcolare una rotazione opposta semplicemente invertendo il segno della componente W.

Per rendere tutto ciò possibile servono 4 componenti (X, Y, Z, W), come già detto in precedenza, una per la posizione originale più una per la rotazione su ogni asse. Se si prende il caso di rotazioni su due assi si può immaginare che queste 3 componenti siano 3 punti su una sfera, come mostrato in Figura 3.2. Ogni altra rotazione è data dall'interpolazione di questi tre punti.

La ragione per cui i due punti sulla sfera rappresentano una rotazione di 180° anziché 90° è resa necessaria in quanto, altrimenti, questi due punti coinciderebbero nel punto più basso della sfera. Come effetto aggiuntivo è possibile rappresentare rotazioni fino a 720° ed ogni orientamento equivale a due possibili rotazioni dando all'animatore l'abilità di decidere in che verso ruotare l'oggetto durante un'animazione.

3.1.4 Matriciale

Quest'ultima è probabilmente la rappresentazione più ottimale in termini di flessibilità, in quanto permette di rappresentare anche traslazioni, scalature e altre trasformazioni come *shear*. È, infatti, la rappresentazione che Blender utilizza internamente [2] [18] proprio perché offre la maggior flessibilità.

Siccome permette di rappresentare qualsiasi tipo di trasformazione, questa struttura viene solitamente chiamata *Matrice di Trasformazione*. Nel caso di uno spazio a 3 dimensioni, essa ha dimensione 4×4.

Questa sua caratteristica la rende indispensabile per rappresentare articolazioni formate da più ossa (i.e Kinematics Linkages), che verranno illustrate qui di seguito (sezioni 3.2 e 3.3), poiché rende possibile convertire le coordinate locali di un osso (rappresentato in Figura 3.3, evidenziato in azzurro) figlio a quelle locali del suo padre, fino ad arrivare alle coordinate globali.

L'unico difetto è che, come la rappresentazione sotto forma di quaternione, non mantiene l'informazione sul percorso della rotazione. Infatti è ancora più simile a una delta-rotazione (i.e. differenza di orientamento), rispetto ad un quaternione poiché copre una rotazione di soli 360°, rispetto ai 720° del quaternione.

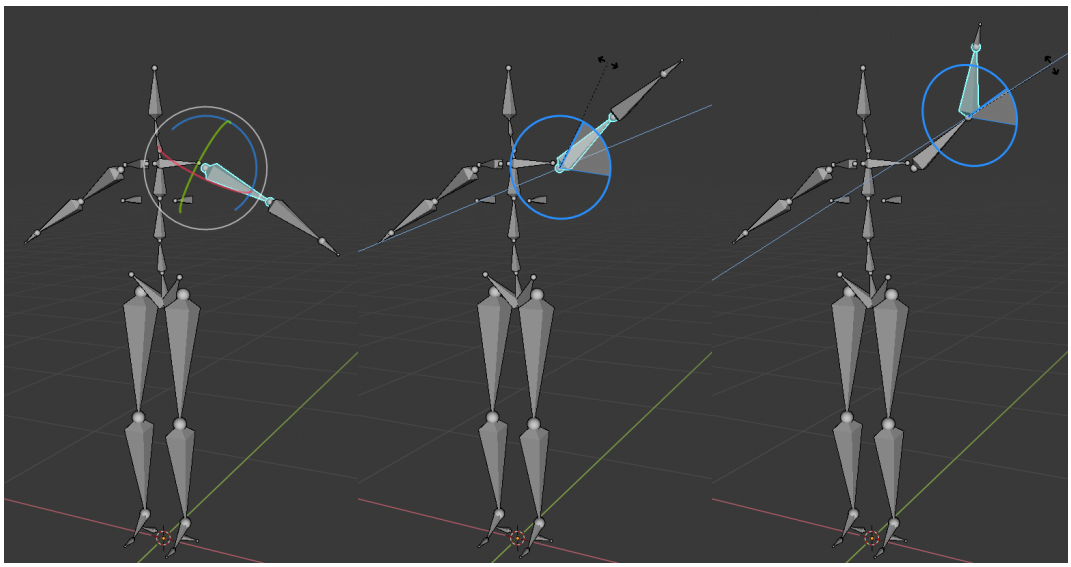


FIGURA 3.3: Esempio di posizionamento tramite Forward Kinematic. La figura rappresenta la stessa armatura, formata da diverse ossa, in tre posizioni differenti.

3.2 Cinematica Diretta

Per figure complesse, come quella umana (da qui in avanti tutti gli esempi saranno riferiti alla figura umana, siccome, nella realizzazione del corto, è stata il centro della maggior parte delle animazioni), è utile avere un sistema che permetta di posizionare le sue componenti in maniera relativa ad altre. Per esempio, una volta posizionato il busto, posizionare il braccio, poi la mano ed infine le dita, tutto in maniera relativa a quanto posizionato in precedenza.

Questa tecnica permette infatti di specificare, attraverso una rotazione, la posizione di un osso relativa al suo osso padre. Per questo motivo è necessario che le ossa di un'armatura (vedi Figura 3.3) siano organizzate in maniera gerarchica (i.e. ad albero). In questo modo, quando l'armatura viene posizionata basterà moltiplicare tutte le matrici di trasformazione dalla radice ai nodi foglia in maniera *deep-first*. Per definire una posa è quindi necessario specificare la rotazione di ogni osso. Questo permette un controllo preciso su ogni DOF dell'armatura, il che è ottimo, perché permette all'animatore di realizzare pose perfette, senza lasciare che nulla venga calcolato automaticamente.

Lo svantaggio è che questa metodologia rende difficile animare azioni comuni in cui gli arti si muovono in uno spazio non relativo al resto del corpo, ma rispetto allo spazio circostante. Ad esempio, per aprire una porta tenendo la mano sulla maniglia, il corpo si può spostare di lato, ma la mano deve rimanere aggrappata alla maniglia. Con una metodologia FK, l'animatore dovrebbe infatti spostare il corpo e, ogni volta, riposizionare la mano sulla

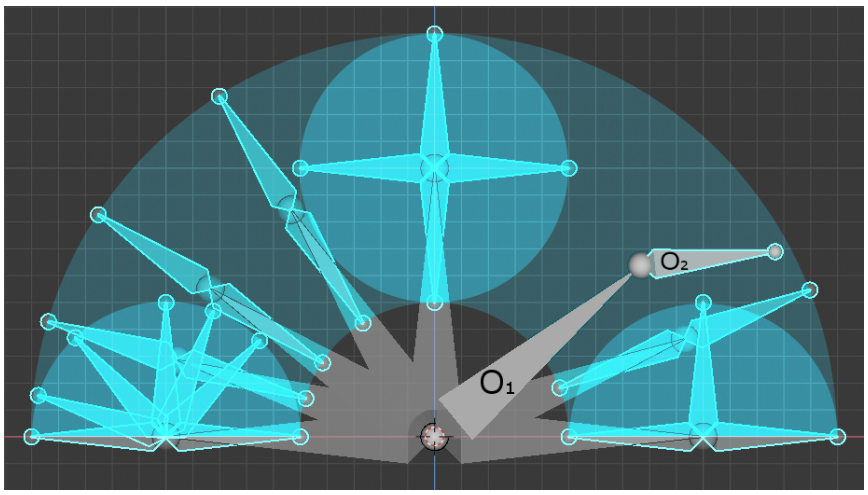


FIGURA 3.4: Esempio di segmento di scheletro formato da due ossa in uno spazio 2D. Quello evidenziato in azzurro è lo spazio raggiungibile dall'*end-effector*

maniglia, introducendo un numero notevole di *counter-animation*, ovvero l'animazione inversa per riposizionare la mano nella posizione in cui era prima del movimento del resto del corpo.

3.3 Cinematica Inversa

Il concetto di Inverse Kinematic (IK) è, fondamentalmente l'inverso di Forward Kinematic. In quest'ultimo ogni osso influenza la rotazione di quelli che lo seguono (i.e. figli). Al contrario, in IK, muovere un osso alla fine di una sequenza (i.e. *end-effector*), influenza la posizione di quelli che lo precedono. Ciò è molto utile nel caso in cui si vogliano posizionare gli arti di una figura umana, in quanto rende il tutto più intuitivo. IK serve anche a fare in modo che una serie di ossa mantenga un estremo (i.e. *end-effector*) connesso a qualcosa. Definite la posizione della mano e della radice (spalla) la posizione delle ossa intermedie viene calcolata automaticamente. Esistono diverse tecniche per il calcolo degli angoli delle articolazioni intermedie.

3.3.1 Soluzione analitica

Nel caso di articolazioni relativamente semplici è possibile calcolare l'intero spazio delle soluzioni analiticamente, ed individuare tra di esse quelle che soddisfano l'obiettivo. Ad esempio, nel caso di un sistema di due ossa (O_1 e O_2) e due articolazioni in uno spazio 2D come in Figura 3.4 lo spazio raggiungibile dall'*end-effector* può essere descritto come

$$\{x : |O_1 - O_2| \leq x \leq O_1 + O_2\}$$

Data la posizione (X, Y) obiettivo dell'*end-effector*, se questa cade all'interno dello spazio raggiungibile, è possibile attraverso le formule trigonometriche, ricavare gli angoli delle 2 articolazioni:

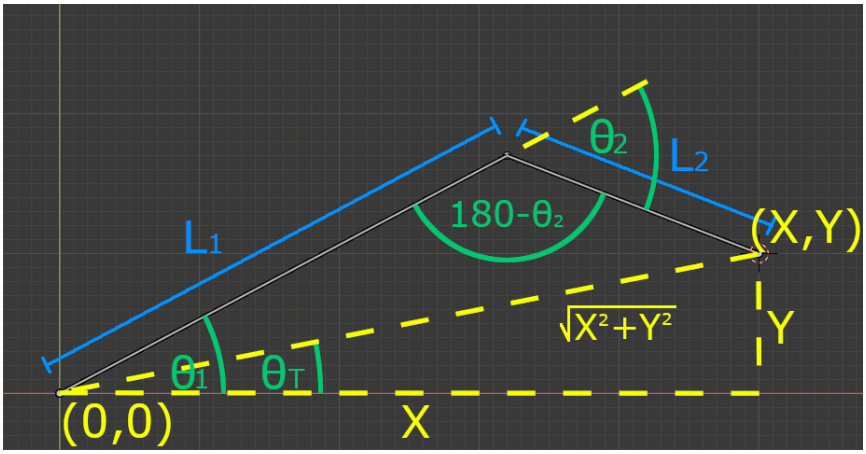


FIGURA 3.5: Soluzione analitica ad un semplice problema di cinematica inversa

$$\arccos(\theta_T) = \frac{X}{\sqrt{X^2+Y^2}} \tag{3.1}$$

$$\theta_T = \arccos\left(\frac{X}{\sqrt{X^2+Y^2}}\right) \tag{3.2}$$

$$\cos(\theta_1 - \theta_T) = \frac{O_1^2 + X^2 + Y^2 - O_2^2}{2O_1\sqrt{X^2+Y^2}} \tag{3.3}$$

$$\theta_1 = \arccos\left(\frac{O_1^2 + X^2 + Y^2 - O_2^2}{2O_1\sqrt{X^2+Y^2}}\right) + \theta_T \tag{3.4}$$

$$\cos(180 - \theta_2) = -\cos(\theta_2) = \frac{O_1^2 + O_2^2 - (X^2+Y^2)}{2O_1O_2} \tag{3.5}$$

$$\theta_2 = \arccos\left(\frac{O_1^2 + O_2^2 - X^2 + Y^2}{2O_1O_2}\right) \tag{3.6}$$

Nell'equazioni sopra riportate θ_1 e θ_2 si riferiscono agli angoli rispetto alla posizione naturale dei rispettivi ossi. θ_T è l'angolo formato dall'ipotenusa immaginaria che si forma tracciando un segmento dalla radice alla posizione obiettivo dell'end-effector, rispetto alla posizione naturale di O_1.

Per casi più complessi, come spesso capita nelle animazioni digitali, viene usato un metodo iterativo analogo alla risoluzione di un problema di ottimizzazione in programmazione lineare [8], in cui la funzione obiettivo è

data dalle variabili che compongono la posizione e l'orientamento dell'end-effector.

3.3.2 Lo Jacobiano

Molti meccanismi utilizzati nell'ambito delle animazioni digitali sono troppo complessi per essere risolti analiticamente. Per questi, il movimento può essere costruito in maniera incrementale. Ad ogni intervallo temporale dell'iterazione, viene calcolato in che modo cambiare gli angoli di ogni articolazione per far sì che la posizione e l'orientamento corrente dell'end-effector si avvicini a quella desiderata. Per fare ciò esistono diversi metodi [9, 20] [13] [12] [3], molti dei quali utilizzano una matrice di derivate parziali chiamata lo Jacobiano, o matrice Jacobiana.

Per spiegare lo Jacobiano da un punto di vista matematico, si considerino le sei funzioni rappresentate in Equazione 3.7, ciascuna delle quali è una funzione in sei variabili indipendenti tra loro. Dati i valori di input per le variabili x_i, ciascun valore di output y_i può essere calcolato dalla rispettiva funzione f_i.

$$\begin{aligned} y_1 &= f_1(x_1, x_2, x_3, x_4, x_5, x_6) \\ y_2 &= f_2(x_1, x_2, x_3, x_4, x_5, x_6) \\ y_3 &= f_3(x_1, x_2, x_3, x_4, x_5, x_6) \\ y_4 &= f_4(x_1, x_2, x_3, x_4, x_5, x_6) \\ y_5 &= f_5(x_1, x_2, x_3, x_4, x_5, x_6) \\ y_6 &= f_6(x_1, x_2, x_3, x_4, x_5, x_6) \end{aligned} \tag{3.7}$$

Queste equazioni possono anche essere usate per descrivere il cambiamento nelle variabili di output relativo alle variabili di input. I differenziali di y_i possono essere scritti in termini dei differenziali delle x_i utilizzando la regola della catena, generando l'Equazione 3.8.

$$dy_i = \frac{\partial f_i}{\partial x_1}dx_1 + \frac{\partial f_i}{\partial x_2}dx_2 + \frac{\partial f_i}{\partial x_3}dx_3 + \frac{\partial f_i}{\partial x_4}dx_4 + \frac{\partial f_i}{\partial x_5}dx_5 + \frac{\partial f_i}{\partial x_6}dx_6 \tag{3.8}$$

Le Equazioni 3.7 e 3.8 possono essere possono essere rappresentate in forma vettoriale, producendo le equazioni 3.9 e 3.10, rispettivamente.

$$Y = F(X) \tag{3.9}$$

$$dY = \frac{\partial F}{\partial X}dX \tag{3.10}$$

Una matrice di derivate parziali, $\frac{\partial F}{\partial X}dX$, viene chiamata *Jacobiana* ed è una funzione dei valori correnti delle x_i. Lo Jacobiano può essere visto come l'associamento delle velocità (ad intervallo di tempo regolare) X alle velocità

Y (Eq. 3.11).

$$\dot{Y} = J(X)\dot{X} \tag{3.11}$$

In qualunque momento, lo Jacobiano è una funzione delle x_i. All'iterazione successiva, X sarà cambiata e, allo stesso modo lo sarà la trasformazione rappresentata dallo Jacobiano.

Quando si applica lo Jacobiano ad una catena di ossa, le variabili di input, x_i, rappresentano i valori delle articolazioni, e le variabili di output, y_i, quelli dell'end-effector (rappresentati come angoli x, y, z).

$$Y = \begin{bmatrix} p_x & p_y & p_z & \alpha_x & \alpha_y & \alpha_z \end{bmatrix}^T \tag{3.12}$$

In questo caso, lo Jacobiano associa le velocità degli angoli delle articolazioni, $\dot{\theta}$, alle velocità della posizione e orientamento dell'end-effector, $\dot{Y}$ (Eq. 3.13).

$$V = \dot{Y} = J(\theta)\dot{\theta} \tag{3.13}$$

V è il vettore delle velocità lineari e angolari, e rappresenta la trasformazione che si vuole applicare all'end-effector. Tale trasformazione si basa sulla differenza tra la posizione/rotazione corrente e quella specificata come obiettivo. Queste velocità sono vettori in uno spazio tridimensionale, pertanto ciascuna ha tre componenti, x, y e z (Eq. 3.14).

$$V = \begin{bmatrix} v_x & v_y & v_z & \omega_x & \omega_y & \omega_z \end{bmatrix}^T \tag{3.14}$$

$\dot{\theta}$ è un vettore delle velocità delle articolazione, in altre parole, contiene i cambiamenti dei parametri di ogni articolazione, ovvero le incognite dell'equazione (Eq. 3.15).

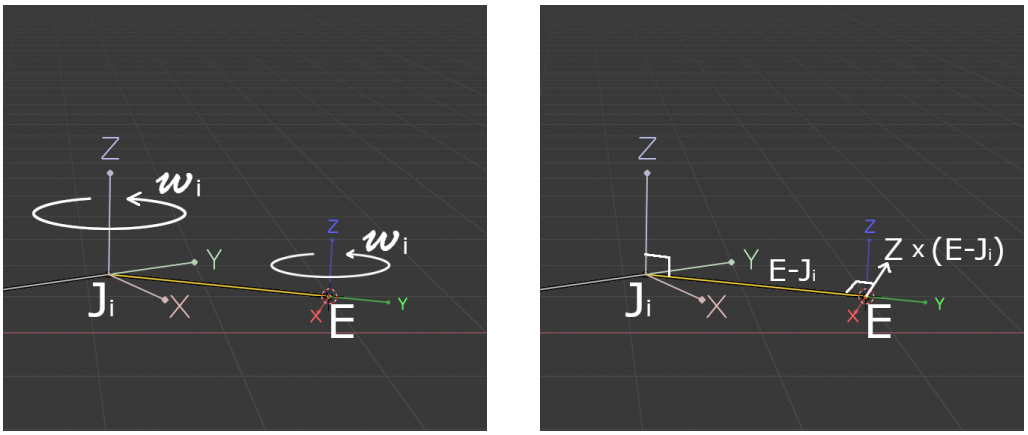
$$\dot{\theta} = \begin{bmatrix} \dot{\theta}_1 & \dot{\theta}_2 & \dots & \dot{\theta}_n \end{bmatrix}^T \tag{3.15}$$

La matrice Jacobiana, J, è una matrice che associa i due vettori appena visti, ed è una funzione della posa corrente (Eq. 3.16).

$$J = \begin{bmatrix} \frac{\partial p_x}{\partial \theta_1} & \frac{\partial p_x}{\partial \theta_2} & \cdots & \frac{\partial p_x}{\partial \theta_n} \\ \frac{\partial p_y}{\partial \theta_1} & \frac{\partial p_y}{\partial \theta_2} & \cdots & \frac{\partial p_y}{\partial \theta_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \alpha_z}{\partial \theta_1} & \frac{\partial \alpha_z}{\partial \theta_2} & \cdots & \frac{\partial \alpha_z}{\partial \theta_n} \end{bmatrix} \tag{3.16}$$

Ogni termine dello Jacobiano rapporta il cambiamento di una specifica articolazione ad uno specifico cambiamento dell'end-effector. Per una giuntura

rivolutiva (i.e. che ruota su un singolo asse), il cambiamento di rotazione nell'end effector, ω, è semplicemente la velocità angolare dell'articolazione intorno all'asse di rotazione per la giunzione che si sta considerando. Per un'articolazione prismatica, in cui un osso trasla rispetto al padre, la rotazione dell'end-effector rimane inalterata dal cambiamento nell'articolazione. Per un'articolazione rotazionale, la trasformazione lineare nell'end-effector è il risultato del prodotto vettoriale tra l'asse di rivoluzione e il vettore che parte dall'articolazione, fino all'end-effector. La rotazione di un articolazione rotazionale induce uno spostamento istantaneo dell'end-effector. Per un'articolazione prismatica, lo spostamento equivale a quello dell'articolazione (vedi Fig. 3.6).



(A) Velocità angolare, ω_i (B) Velocità lineare, $Z \times (E - J_i)$

E—end effector
J_i—i-esima articolazione
Z—asse di rotazione
ω_i—velocità angolare dell'i-esima articolazione

FIGURA 3.6: Velocità angolare e lineare indotte dalla rotazione di un'articolazione intorno al proprio asse.

Le velocità lineari e angolari desiderate sono calcolate trovando la differenza tra la configurazione dell'end-effector corrente e quella obiettivo. Le velocità indotte dalla rotazione di una specifica articolazione, sono determinate dai calcoli mostrati in Figura 3.6. Il problema sta nel determinare la migliore combinazione lineare delle velocità, indotte dalle varie articolazioni, che risulti nelle velocità desiderate nell'end-effector. Lo Jacobiano è il risultato della rappresentazione del problema descritto, in forma di matrice.

È importante che, nel rappresentare il problema in forma Jacobiana, tutte le coordinate dei valori delle articolazioni, siano rappresentate come coordinate dello stesso sistema (e.g. coordinate globali). Spesso infatti, le informazioni relative ad una particolare articolazione sono rappresentate nel sistema di coordinate locale a quell'articolazione. Durante la creazione della matrice Jacobiana, queste informazione devono essere convertite in un sistema di

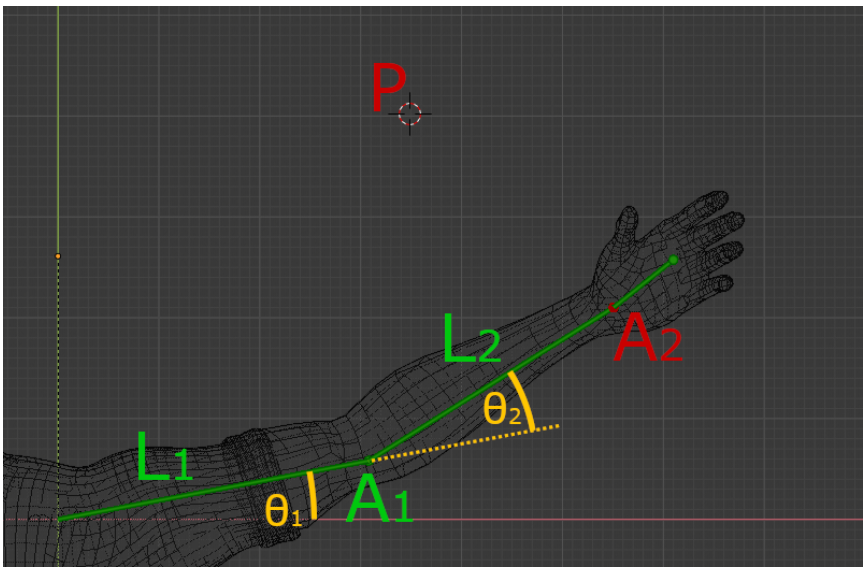


FIGURA 3.7: Esempio di catena formata da due articolazioni che possono ruotare su un piano.

coordinate comune, come quello delle coordinate globali, o dell'end-effector. Esistono diversi metodi per calcolare lo Jacobiano che si basano sul massimizzare l'efficienza computazionale date le informazioni in coordinate locali, tutti però producono il risultato in un sistema di coordinate comune.

Per fare un esempio inerente a questo progetto, si consideri lo scheletro di uno qualunque dei personaggi, in particolare il suo braccio. In questo caso ci si può limitare ad un movimento in un piano fissato, che quindi esclude dal problema una delle tre dimensioni e vincola le rotazione ad un unico asse (z). In Figura 3.7 si vuole spostare l'end-effector, rappresentato dal polso (A_2), nel punto P. In questo caso particolare, l'orientamento della mano non ha importanza. L'asse di rotazione di ogni articolazione (A_i) è perpendicolare alla figura ed esce dal foglio. L'effetto di una rotazione incrementale, r_i, di ciascuna articolazione, è dato dal prodotto vettoriale tra l'asse dell'articolazione e il vettore che collega quest'ultima all'end-effector, $\vec{V_i}$ (vedi Fig 3.8), e costituisce le colonne della matrice Jacobiana. È importante notare come ogni rotazione di un'articolazione influenzi tutte quelle successive. Di conseguenza, le prime hanno un'intensità maggiore rispetto alle ultime ($r_1 > r_2$). In termini tecnici la lunghezza di ogni r_i è funzione della distanza tra la posizione della rispettiva articolazione dall'end-effector.

La trasformazione da applicare all'end-effector è il risultato della differenza tra la sua posizione corrente e quella obiettivo (Eq. 3.17).

$$V = \begin{bmatrix} (P - A_2)_x \\ (P - A_2)_y \\ (P - A_2)_z \end{bmatrix} \tag{3.17}$$

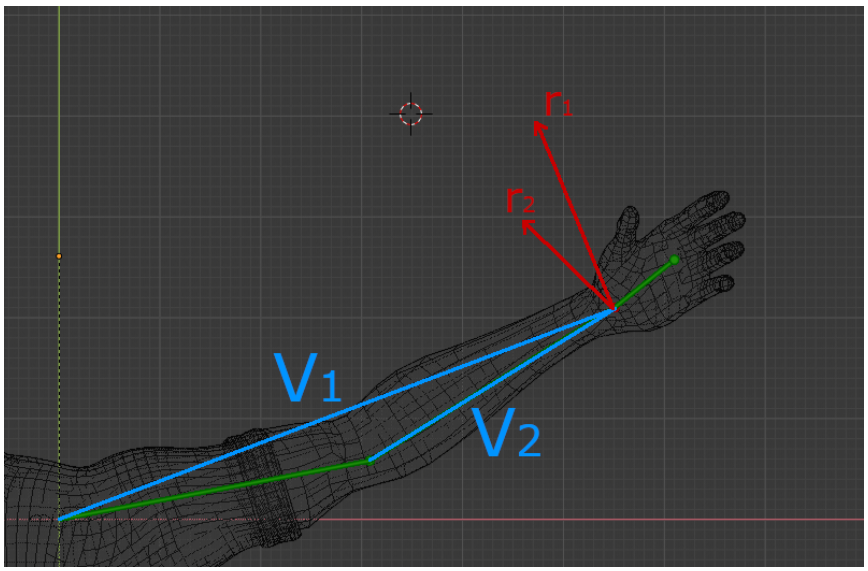


FIGURA 3.8: Modifiche istantanee della posizione indotte dall'angolo di rotazione delle articolazioni.

Il vettore delle trasformazioni che si vogliono applicare è uguale alla matrice Jacobiana (Eq. 3.18) moltiplicata per il vettore delle incognite, date dai cambiamenti negli angoli delle articolazioni.

$$J = \begin{bmatrix} ((0,0,1) \times A_2)_x & ((0,0,1) \times (A_2 - A_1))_x \\ ((0,0,1) \times A_2)_y & ((0,0,1) \times (A_2 - A_1))_y \\ ((0,0,1) \times A_2)_z & ((0,0,1) \times (A_2 - A_1))_z \end{bmatrix} \tag{3.18}$$

Una volta calcolato lo Jacobiano, bisogna risolvere un'equazione nella forma mostrata in Equazione 3.19. Nel caso J sia una matrice quadrata, l'inversa dello Jacobiano, J^{-1}, può essere utilizzata per calcolare le velocità angolari delle articolazioni, date quelle dell'end-effector (Eq. 3.20).

$$V = J\dot{\theta} \tag{3.19}$$

$$J^{-1}V = \dot{\theta} \tag{3.20}$$

Se l'inverso dello Jacobiano non esiste, allora si dice che il sistema è singolare per gli angoli assegnati tra le articolazioni. Una singolarità si verifica quando non è possibile formare una combinazione lineare delle velocità dell'angolo articolare per produrre le velocità desiderate dell'effettore finale. Come semplice esempio di una situazione del genere, consideriamo un braccio planare completamente esteso con una posizione dell'obiettivo posizionata sull'avambraccio (vedi Figura 3.9). In tal caso, un cambiamento in ciascun angolo dell'articolazione produrrebbe un vettore perpendicolare alla direzione desiderata. Ovviamente, nessuna combinazione lineare di questi vettori potrebbe produrre il vettore di movimento desiderato. Tutte le singolarità

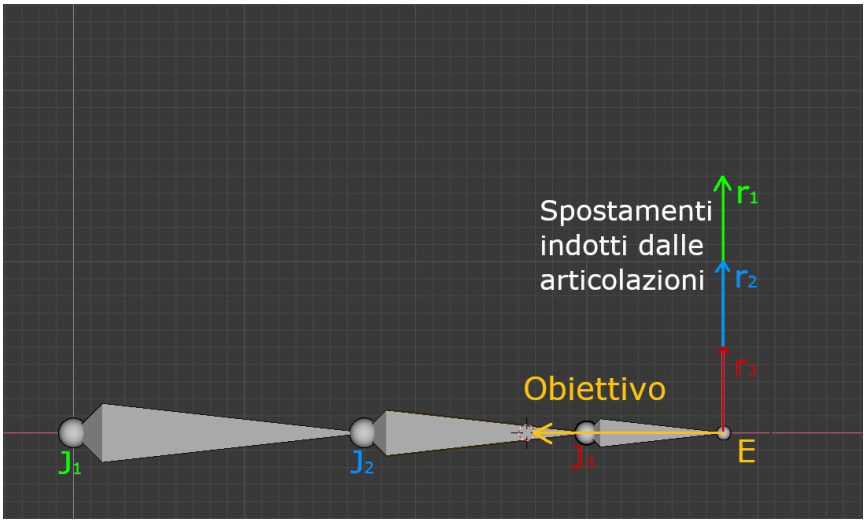


FIGURA 3.9: Semplice esempio di una configurazione singolare.

di un sistema non possono essere determinate semplicemente ispezionando visivamente le possibili configurazioni geometriche del collegamento.

Inoltre, anche una configurazione che si avvicina solo all'essere una singolarità può ancora presentare gravi problemi. Se i giunti del collegamento nella Figura 3.9 sono leggermente perturbati, la configurazione non è singolare. Tuttavia, per formare una combinazione lineare dei vettori di cambiamento istantaneo risultanti, è necessario utilizzare valori molto grandi. Ciò si traduce in grandi impulsi vicino ad aree di singolarità. Questi devono essere bloccati a valori più ragionevoli. Anche in questo caso, l'errore numerico può provocare movimenti imprevedibili. I problemi con le singolarità possono essere ridotti se il manipolatore è ridondante, quando vi sono più DOF di quanti siano i vincoli da soddisfare. In questo caso, lo Jacobiano non è una matrice quadrata e, potenzialmente, esiste un numero infinito di soluzioni al problema IK. Poiché il Jacobiano non è quadrato, non esiste l' inverso convenzionale. Tuttavia, se le colonne di J sono linearmente indipendenti, esiste la Pseudoinversa. Numerose sono le tecniche numeriche basate sulla regolarizzazione dell'operatore Jacobiano per risolvere il problema limitando l'effetto dei problemi numerici [5] [6].

Blender propone due metodi per risolvere la cinematica inversa, uno di questi tiene conto anche di altri vincoli definiti dall'utente [1]. Questo metodo si chiama iTaSC e, come il metodo standard, utilizza una matrice Jacobiana, ma il modo in cui la calcola è differente [3].

Un motivo per cui il metodo iterativo è migliore di quello analitico, oltre ad essere l'unico metodo per risolvere articolazioni complesse, è che, nel caso in cui la posizione obiettivo si trovi al di fuori dello spazio raggiungibile, (e.g. la mano deve raggiungere un punto troppo distante dalla spalla) non esiste

nessuna soluzione e in questo caso il risultato può e dev'essere approssimato.

Siccome quello della cinematica inversa è un problema sotto-vincolato, è molto probabile che esista più di una soluzione. Nel caso visto in Figura 3.4 ad esempio esistono 2 soluzioni, per un qualsiasi punto (X, Y) interno allo spazio raggiungibile, simmetriche rispetto alla linea immaginaria che collega la radice di O_1 alla punta di O_2. In tal caso il risultato è scelto dall'algoritmo tra una delle possibili soluzioni. Non sempre però, sono tutte realistiche o visivamente gradevoli. Per questo motivo è possibile aggiungere dei vincoli per limitare il numero di soluzioni possibili, molto utili sono quelli per l'imitare il movimento di un arto, per renderlo più realistico.

Capitolo 4

Progettazione

Così come nella progettazione di un software si passa dall'analisi alla sua progettazione, prima di svilupparlo, anche in questo caso, è opportuno progettare l'intero cortometraggio, prima di realizzarlo. Questo è un passaggio importante poiché non solo permette di capire come il prodotto dovrà essere realizzato, ma permette anche di stabilire delle convenzioni standard (e.g. nomi dei file) da mantenere durante il progetto. Quest'ultimo aspetto è indispensabile soprattutto nel caso in cui ci siano più persone a lavorare allo stesso progetto.

4.1 Descrizione del Cortometraggio

Il cortometraggio da realizzare narra le storie di un ragazzo e un ex pirata spaziale, e dell'intreccio delle loro storie. Questo corto rappresenta solo il probabile primo episodio di una serie, in esso si vedrà l'incontro dei due protagonisti che, immediatamente, porterà una svolta alla routine quotidiana del ragazzo. I due, insieme ad una donna extraterrestre, dovranno affrontare diversi nemici, per poi scappare e far perdere le proprie tracce. In particolare il trailer mostrerà alcune delle scene più salienti del cortometraggio (vedi Figura 4.1).

Esse mostrano alcune azioni dei personaggi che andranno animate come camminata, corsa, sedersi, impugnare oggetti... è quindi necessaria una progettazione accurata del corto ed in particolare delle animazioni.

4.2 Progettazione generale

Normalmente la fase di progettazione di un cortometraggio serve a definire l'aspetto dei personaggi e dell'ambientazione (i.e. Character Design, Environment/Prop Design). Tuttavia questi aspetti sono stati prevalentemente coperti da A. Uras, in quanto il corto rappresenta una storia di sua creazione, e non verranno trattati. La progettazione delle animazioni verrà invece trattata dettagliatamente, in quanto oggetto principale del mio lavoro.

(Disegni di Alberto Uras)

FIGURA 4.1: Alcune delle scene, tratte dallo storyboard, che verranno mostrate nel trailer

A questo punto del progetto era chiaro che l'obiettivo finale era quello di realizzare un breve trailer di una possibile serie piuttosto che un cortometraggio. È stato quindi necessario selezionare le scene da realizzare per stare entro i due minuti di tempo di riproduzione.

Fatto ciò si è creato un flusso di lavoro, in maniera tale da procedere in modo organizzato. Siccome non ero il solo a lavorare a questo progetto ho scelto un approccio orientato alla produzione, ovvero in cui diversi team si occupano di diversi aspetti del progetto ma che, ciò non di meno, sono collegati e dipendono gli uni dagli altri.

Ad esempio, una volta realizzato il rig base per un modello, è necessario poterlo animare direttamente, anche senza avere il rig avanzato finale. Ancora più importante è che sia possibile aggiungere dettagli ad un modello, come materiali e textures, anche dopo che questo è stato animato. Per fare ciò, ho seguito una semplice filosofia: creare l'oggetto una volta e linkarlo in diverse scene invece che copiarlo. Questa procedura è analoga a quella di utilizzare delle interfacce nella programmazione ad oggetti invece che le classi direttamente. In questo modo, è possibile modificare una classe in un secondo tempo senza alterarne il suo comportamento e, di conseguenza, chi la utilizza non nota alcuna differenza.

Ciò è stato possibile anche grazie alla caratteristica delle armature (o scheletro), di poter essere animate anche quando linkate da un altro file. In pratica, è possibile definire un proxy che permette di accedere a tutte le proprietà di un'armatura come se questa fosse l'oggetto originale e non una copia (o meglio un link).

4.3 Progettazione delle animazioni

Quando si tratta di sviluppare qualcosa di complesso come una figura umana, è opportuno progettarla, per trovare un modello di astrazione che la semplifichi, pur mantenendo le proprietà che ci interessano e quindi ci permetta di animarla in maniera efficiente. Il corpo umano è infatti composto da circa duecento DOF [14]. Nonostante ciò, la struttura esteriore è fondamentalmente composta da un'unica mesh. Quindi, rispetto a quanto visto fin'ora, dove la struttura da animare era un'armatura composta da più ossa, sarà necessario deformare la mesh per adattarla all'armatura sottostante. Ciò è possibile associando determinati vertici della mesh ad ogni osso.

La progettazione del rig (Figura 5.9) serve ad ottenere una serie di controlli interattivi mirati a facilitare l'animazione del personaggio. L'usabilità di un rig è un aspetto fondamentale, in quanto, se il rig non è semplice da utilizzare per un animatore, esso è completamente inutile. Il rig, infatti, non è fine a se stesso, ma è lo strumento che permetterà (solitamente ad altri) di animare i nostri modelli. La progettazione delle animazione consiste dunque nell'ideare un rig che permetterà poi di animare i modelli efficientemente. Parlare

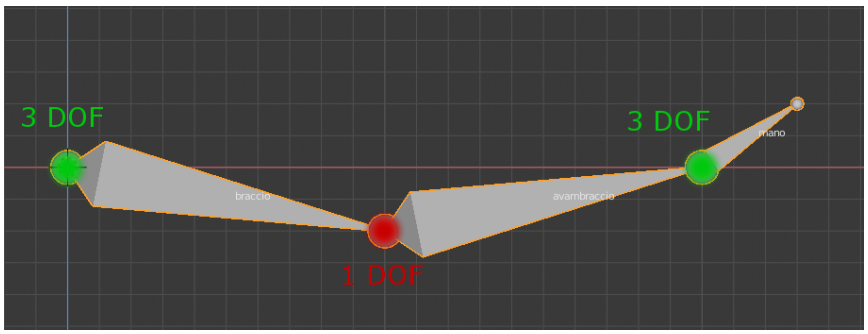


FIGURA 4.2: Esempio di un semplice rig per il braccio.

di progettazione di animazioni o di rig è quindi indifferente. Per progettare un buon rig serve, prima di tutto, individuarne l'obiettivo, per rendere il rig il più semplice possibile. Di seguito sono riportati alcuni degli obiettivi individuati e per le diverse parti di un rig.

4.3.1 Arti superiori

Le braccia devono essere in grado di estendersi per raggiungere e afferrare oggetti. Come spiegato in precedenza (Capitolo 3), ha senso fare utilizzo dell'IK per posizionare la mano dove si vuole, e lasciare che il resto del braccio venga posizionato di conseguenza. In tal caso è opportuno identificare quante articolazioni sono necessarie nel braccio e quanti DOF servono in ciascuna di esse. Un buon modo di rappresentare gli arti superiori è quello di utilizzare 7 DOF: 3 per la spalla e il polso, 1 per il gomito (vedi Figura 4.2).

Un altro modo è quello di aggiungere un articolazione a metà dell'avambraccio per permettere la rotazione longitudinale di esso. In altri casi l'avambraccio è staccato dal braccio, questo permette di evitare di dover deformare la mesh, ma il taglio netto tra avambraccio e braccio non sempre è conveniente in termini di visualizzazione. Per tutte le figure umane nel nostro progetto è stata utilizzata la prima rappresentazione (Figura 4.2). Nel corto, appaiono anche dei robot, per i quali si sarebbe potuto utilizzare la terza rappresentazione, più semplice. È stato tuttavia scelto di utilizzare comunque la prima per permettere di avere lo stesso scheletro per tutti i personaggi, in modo da avere un rig uniforme e non dover fare lavoro doppio.

Un'ultima considerazione da fare è che ci sono alcuni casi in cui è preferibile poter animare il braccio attraverso FK. Per questo motivo è stato scelto di utilizzare un rig che permetta di alternare tra FK e IK. Nel caso della FK è inoltre opportuno rendere la rotazione del braccio indipendente da quella del corpo. Questa caratteristica è quasi sempre indispensabile, se si pensa ad esempio al ritardo del movimento delle braccia, quando le si lascia andare (a peso morto), rispetto a quello del busto, facendolo ruotare a destra e sinistra. Esistono però dei casi in cui è preferibile che le braccia mantengano la rotazione del busto, ad esempio nel caso di un robot. Siccome lo stesso rig verrà

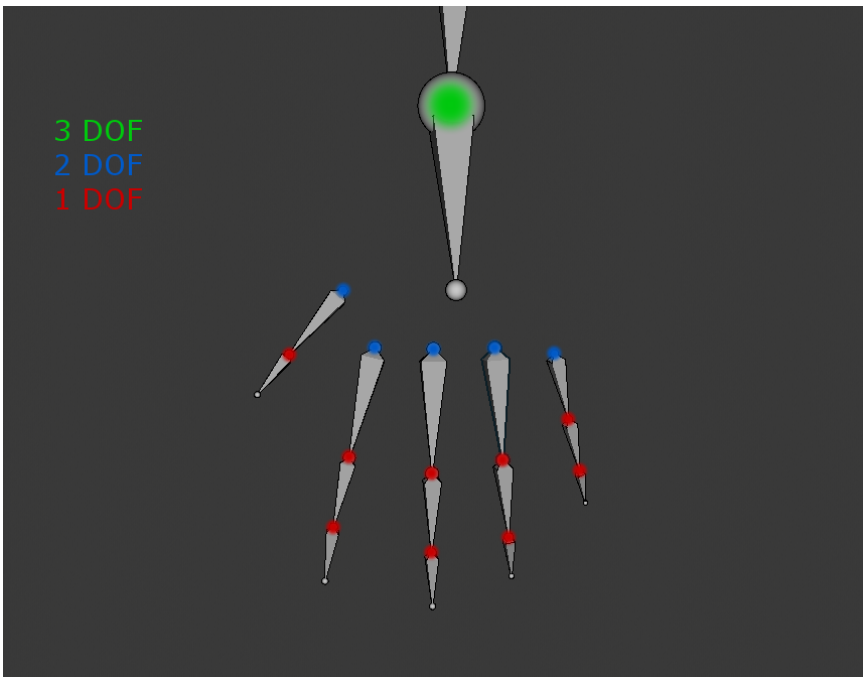


FIGURA 4.3: Esempio di un semplice rig per la mano

utilizzato sia per umani che robot, ha senso far sì che questa caratteristica sia attivabile e disattivabile a seconda di chi stia implementando il rig.

Per quanto riguarda il rig della mano un modello molto utilizzato è quello rappresentato in Figura 4.3. Esso consente di avere un controllo piuttosto semplice che permetta di eseguire le azioni più comuni relative alla mano, come indicare e afferrare oggetti. Per semplificare quest'ultima è possibile aggiungere un driver che controlla allo stesso tempo la chiusura di falangina e falangetta. Quando afferriamo un oggetto infatti queste due parti di ogni dito si muovono in maniera coordinata e, a differenza della falange, non è possibile muoverne una in maniera indipendentemente dalle altre. In alcuni casi un osso aggiuntivo è presente per ogni dito, tra la falange e il polso, per permettere un maggiore controllo e livello di dettaglio. Nel nostro caso, questa scelta non è stata ritenuta necessaria, preferendo un rig più semplice e più veloce da animare. Un modo più semplice invece, soprattutto usato in videogiochi che utilizzano il minor numero possibile di poligoni, è quello di raggruppare alcune dita. È molto importante scegliere il giusto bilancio tra livello di controllo e semplicità d'uso, questo trade-off infatti non permette di massimizzare entrambe le caratteristiche. Siccome per le dita non è necessario avere un IK, non sono necessarie ulteriori progettazioni. Verrà semplicemente usato l'FK per il resto della mano.

Ora che le articolazioni sono state individuate e, per ciascuna di esse, si sa quanti DOF serviranno, è possibile procedere con la scelta della giusta rappresentazione di rotazione (3.1 Rappresentazioni di rotazione). Basandoci su quanto visto nel Capitolo 3, si può dedurre che per le articolazioni con 3 DOF

andranno utilizzati dei quaternioni. Mentre per le altre è possibile usare gli angoli di Eulero.

Nel caso di rotazione euleriana è anche necessario scegliere l'ordine degli assi, in base a quelli sui quali verrà effettuata la rotazione:

- più usato -> anello (Figura 3.1) esterno, allineato globalmente/osso genitore;
- secondo più usato -> anello interno, rotazione locale;
- meno usato/non usato -> secondo, causa gimbal lock;

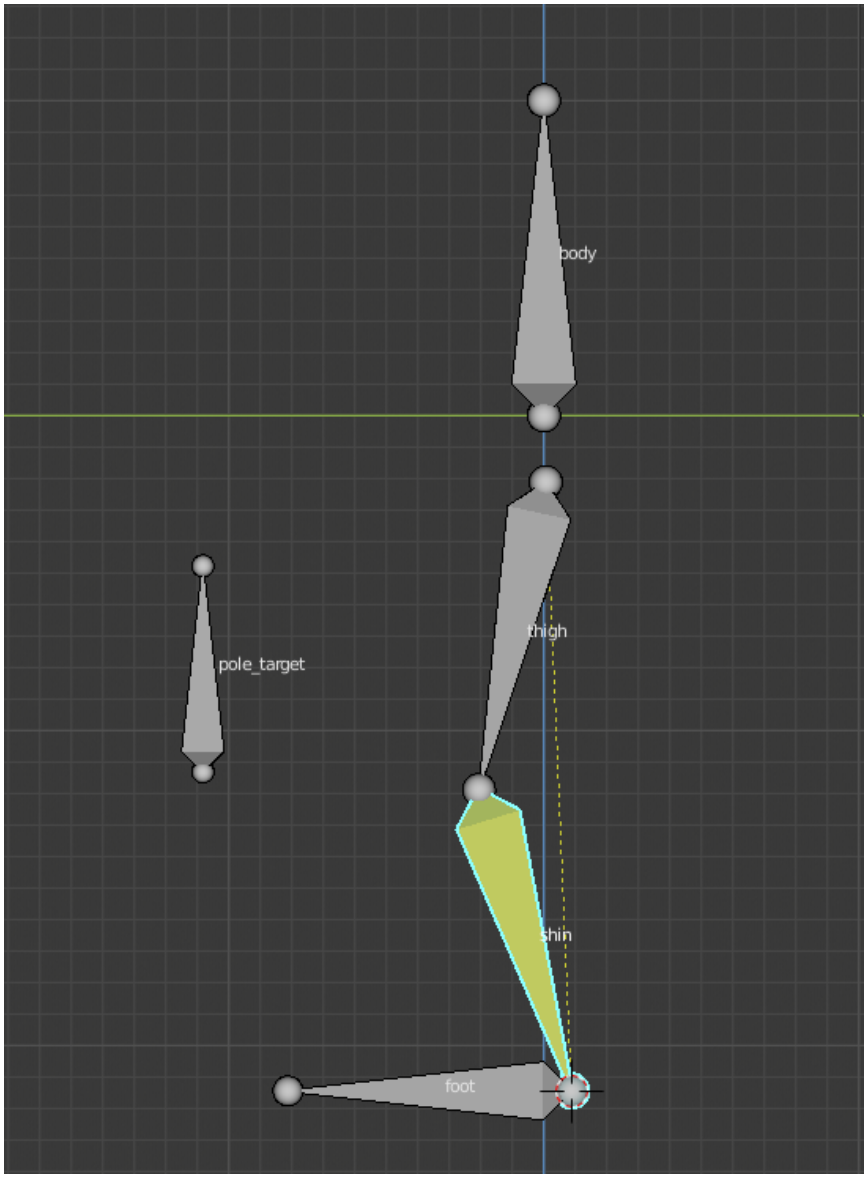


FIGURA 4.4: Esempio di un semplice rig per la gamba

4.3.2 Arti inferiori

Per progettare il rig delle gambe in maniera efficacie, bisogna pensare a qual è il modo più conveniente di animare una camminata.

Ad un certo punto il piede poggia a terra e deve rimanere fisso in quella posizione. Il corpo, invece, continua a spostarsi in avanti. Se usassimo la cinematica diretta, ogni volta che il corpo viene spostato in avanti, dovremmo riposizionare il piede nella posizione in cui era (i.e. counter-animation), in quanto quest'ultimo è discendente del corpo. In alternativa potremmo rimodellare l'intera gerarchia per far sì che il piede si trovi più in alto nella gerarchia rispetto al corpo. Ma questo sarebbe una pessima scelta per ovvi motivi. Come nell'esempio della mano sulla maniglia, anche qui serve utilizzare una IK, per muovere il resto del corpo indipendentemente dal piede. Le gambe possono quindi essere controllate interamente attraverso la cinematica inversa (Sezione 3.3). Siccome questo metodo è sotto-vincolato (i.e. esistono più soluzioni), resta da specificare un vincolo aggiuntivo, relativo all'articolazione intermedia (il ginocchio).

Per fare ciò, è possibile aggiungere un osso, figlio del piede, che servirà per direzionare il ginocchio. Il motivo per cui, quest'ultimo osso, viene solitamente definito come figlio del piede, è per il semplice motivo che piede e ginocchio puntano solitamente nella stessa direzione. In questo modo il ginocchio eredita la rotazione del piede, semplificando ulteriormente il lavoro dell'animatore.

In Blender questo osso viene chiamato *Pole Target* [1], ed il suo funzionamento è molto semplice: tracciando una linea immaginaria dalla base della gamba all'end-effector, viene individuato l'asse che collega i due poli della sequenza delle ossa che costituisce la cinematica inversa. In Figura 4.4 questo asse è rappresentato da una linea tratteggiata gialla. Ruotando l'intera sequenza di ossa intorno a questo asse, si ottengono infinite soluzioni ammissibili, in cui l'articolazione del ginocchio è l'unica a cambiare posizione, descrivendo una sorta di "equatore" intorno all'asse. Ora, tracciando un'altra linea, che collega la base del *pole target* all'asse precedentemente menzionato, e perpendicolare a quest'ultimo, avremo un modo di rappresentare la posizione esatta del ginocchio, attraverso l'angolo formato da quest'ultima linea rispetto alla posizione originale. Si noti che la seconda linea rimane sempre perpendicolare all'asse. Quindi una traslazione del *pole target* parallela all'asse, non ha effetto sulla rotazione della gamba.

In fine, come è stato fatto per il gomito, si può forzare il ginocchio a piegarsi su un singolo asse, garantendo un movimento ancora più realistico in fase di animazione.

Capitolo 5

Produzione

5.1 Modellazione

Con modellazione digitale, o modellazione 3D, o semplicemente modellazione, per brevità, si intende la manipolazione di veritci in uno spazio tridimensionale per realizzare solidi più o meno complessi denominati modelli. Questa è senz'altro la fare più artistica dell'intero processo di produzione, e una di quelle che più mi ha appassionato.

Una volta definiti i concept art, in gran parte realizzati da A. Uras, il primo passo della produzione è appunto quello di realizzare i modelli dei personaggi, dei prop e delle ambientazioni. Per fare ciò, sono state usate diverse tecniche di modellazione 3D, che possono essere divise nelle seguenti due categorie:

- Modellazione hard-surface [7].
- Scultura digitale[10].

La differenza fondamentale sta nel cosa si vuole realizzare. La prima è ottima per quasi tutto ciò che è realizzato dall'uomo (e.g. macchinari, edifici) ed è stata quindi stata usata nella realizzazione di ambientazioni e prop. Consiste nel partire da una forma geometrica semplice, ad esempio un cubo o una sfera, ed aggiungere dettagli estrudendo sezioni o utilizzato operatori booleani in combinazione con altre forme geometriche. (Figura 5.1) Il risultato è un modello dagli spigoli ben definiti, che comunque può avere superfici curve e smussate. La caratteristica che lo distingue è la simmetria della struttura e la precisione della posizione dei dettagli.

Al contrario, il secondo metodo viene utilizzato per ottenere modelli organici, in pratica tutto ciò che è presente in natura. Questi modelli sono particolari, poiché andranno deformati per essere animati. Un esempio lampante, in questo progetto, sono i personaggi. Nonostante anche in questo caso la simmetria sia fondamentale (ogni personaggio ha due braccia e due gambe perfettamente simmetriche), vengono solitamente aggiunti dettagli per rompere la simmetria, siccome in natura nulla è perfettamente simmetrico. Anche in questo caso si parte solitamente da forme geometriche semplici,

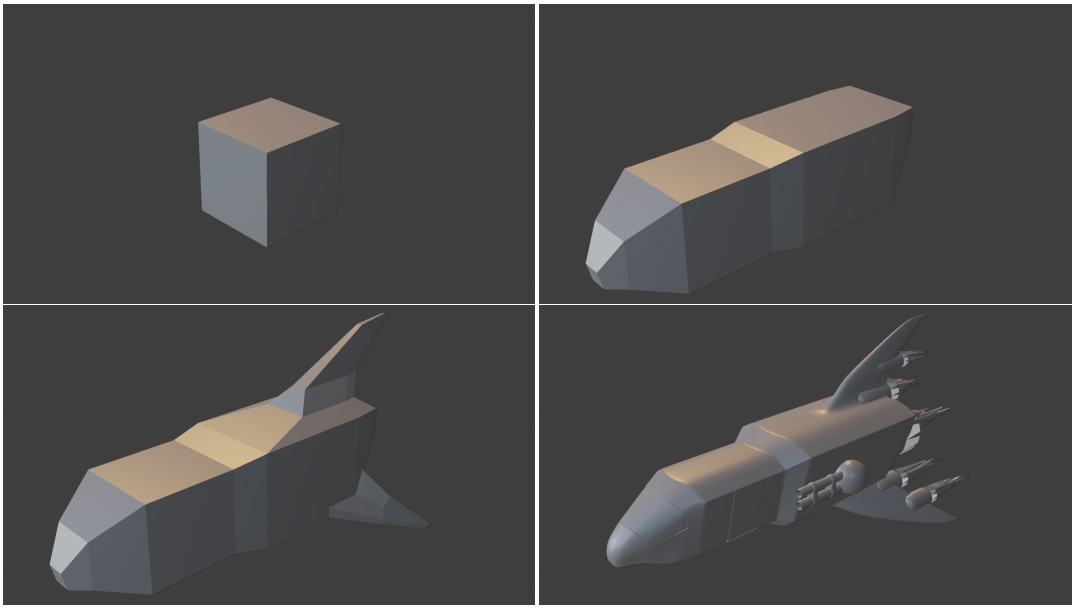


FIGURA 5.1: Esempio di modellazione hard-surface per modellare un'astronave.

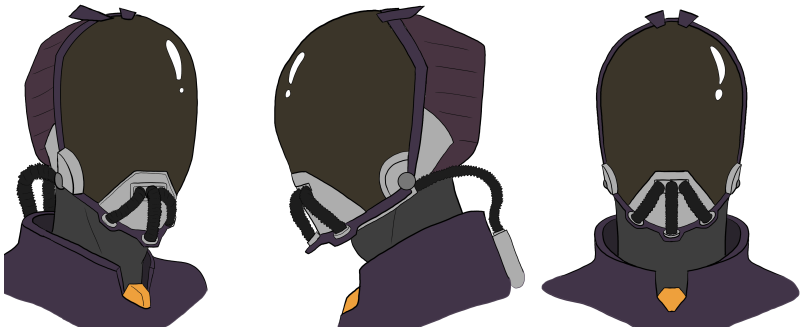


FIGURA 5.2: Concept art di uno dei porsonaggi

tuttavia il processo di modellazione è completamente differente, tant'è che spesso si fa distinzione tra modellazione e scultura digitale (come se i due concetti si escludessero a vicenda). Nella scultura digitale infatti non si usano mai operazioni come l'estrusione di una faccia, di fatto il concetto di faccia non viene neanche utilizzato. La mesh viene vista come un oggetto compatto, i cui vertici possono essere manipolati attraverso strumenti che emulano le azioni di uno scultore. Per questo motivo la scultura digitale è anche il metodo di modellazione preferito dagli artisti.

Affinché un modello si possa definire ben fatto e ultimato, è necessario soddisfare i seguenti criteri. In primo luogo, deve ben rappresentare in 3D quello che il concept si limitava a mostrare in 2D. In Figura 5.2 e 5.3 è possibile notare quanto il modello 3D si avvicini al concept originale. Nonostante, in

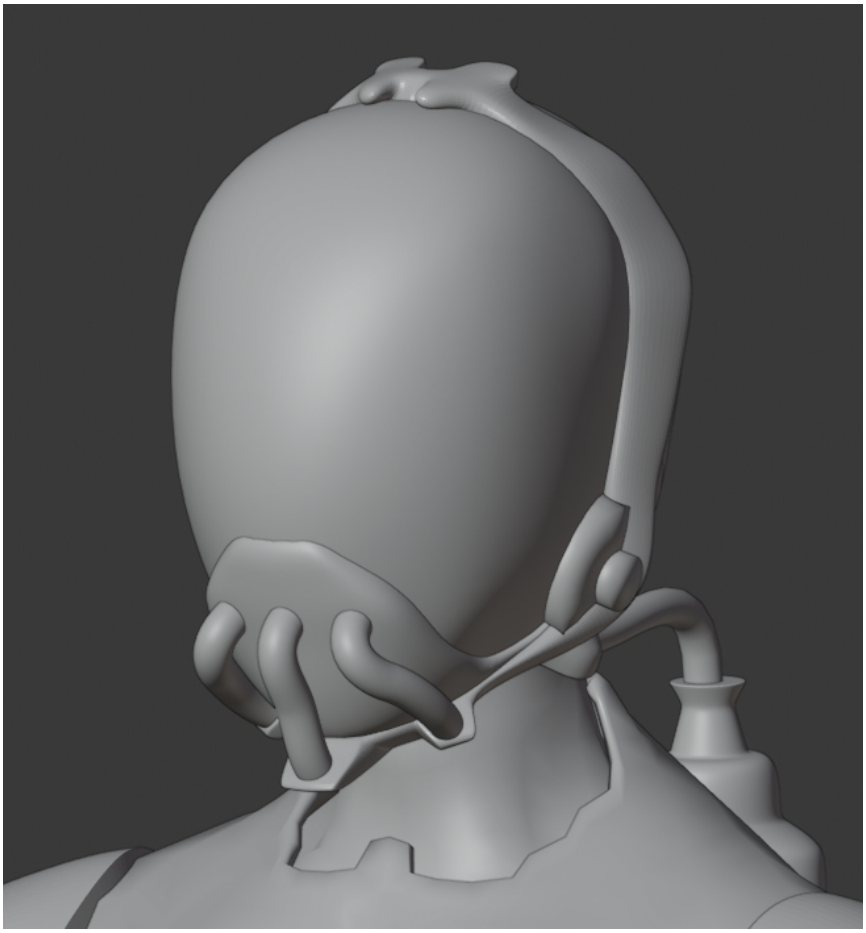


FIGURA 5.3: Rappresentazione 3D del concept art mostrato in Figura 5.2

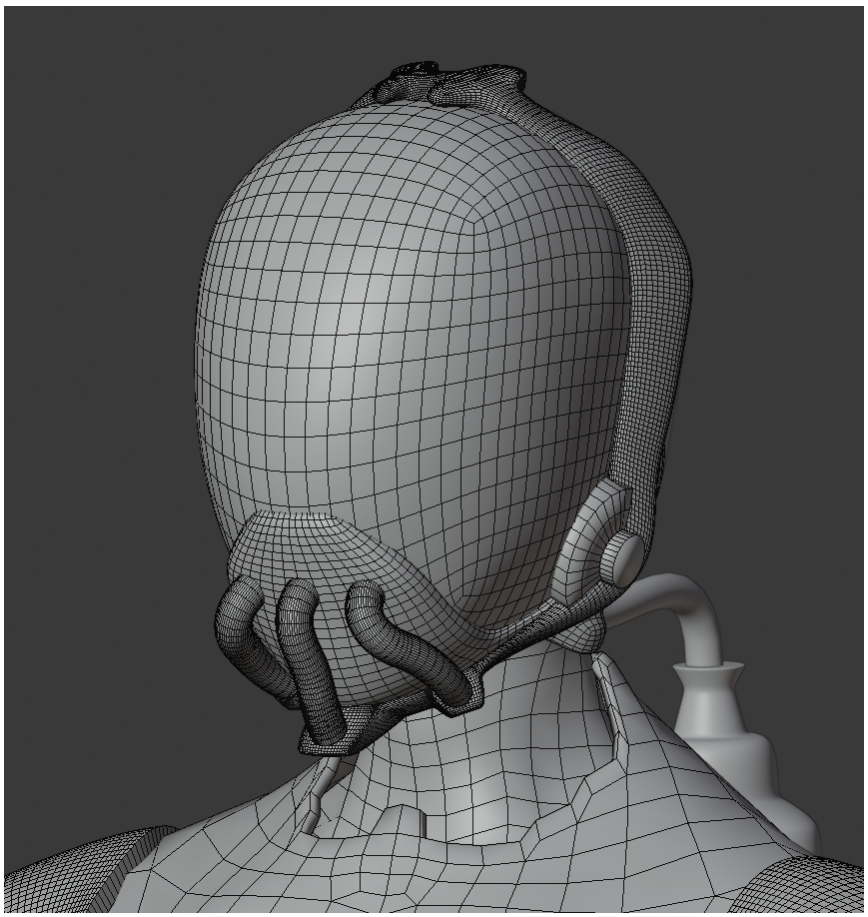


FIGURA 5.4: Modello 3D con topologia della mesh visibile

questo caso, il modello rappresenti molto bene il concept, è possibile notare come alcuni dettagli differiscano. Questo è dovuto principalmente a due motivi: il primo è che, semplicemente, il concept art serve a dare un'idea di come deve apparire il personaggio, e non vincola quindi a mantenere gli stessi dettagli durante la fase di modellazione. Il secondo è che in un disegno 2D la forzatura della prospettiva può nascondere dettagli a cui un modello 3D non può evadere. Per tanto, sono spesso necessarie modifiche poiché non sarebbe possibile realizzare in 3D ciò che era rappresentato in 2D.

Il secondo criterio, non meno importante del primo, è la *topologia* della mesh. A dir la verità, questo aspetto non ha alcuna importanza nel caso di modelli statici, mirati allo scopo di essere rappresentati in un render fisso, o stampati in 3D. Tuttavia, nel caso delle animazioni, questo è un aspetto fondamentale. Infatti dal posizionamento dei vertici dipende la deformazione della mesh che, com'è già stato detto, è necessaria nelle animazioni dei modelli organici (e.g. personaggi). Per avere una buona topologia, è fondamentale che le facce siano composte di quattro lati. Questo serve a definire dei percorsi, che attraversano la faccia da un lato a quello opposto e continuano nelle facce adiacenti. Questi percorsi servono a far si che quando si aggiunge un maggiore

livello di dettaglio, la mesh non venga deformata in maniera inaspettata. Un altra caratteristica importante, per una buona topologia, è quella di avere al massimo quattro spigoli (o archi) che partano da ogni vertice. Questo non è sempre possibile, tuttavia è necessario che vertici con più di cinque spigoli siano presenti nel minor numero possibile, e siano posizionati in punti strategici dove la mesh verrà difficilmente deformata. Questo perché se vertice è adiacente a molti vertici, genera anche molti incroci di percorsi che sono da evitare per quanto detto prima.

Infine, un aspetto non poco importante nella definizione qualitativa di un modello, è il numero totale di vertici. L'obiettivo è quello di avere il minor numero possibile di vertici per poter realizzare un certo livello di dettaglio. Il motivo anche qui è duplice, seppure i due effetti sono strettamente correlati. Innanzi tutto, un ridotto numero di vertici permette un *frame-rate* più alto in fase di animazione. In secondo luogo, in fase di rendering, ogni frame impiegerà meno tempo ad essere renderizzato.

Per mantenere un livello di dettaglio, discretamente alto, è stato fatto uso di una tecnica denominata *baking*, che verrà approfondita nel paragrafo successivo. Per ora mi limiterò a dire che per ogni modello è stato fatta prima una versione "*low-poly*", ovvero a basso contenuto di poligoni (o vertici, analogamente). Dopodiché i dettagli sono stati modellati su una copia di questo modello, dopo averne aumentato il numero di vertici suddividendo ogni faccia. Avere due modelli separati, ci permette di utilizzare il primo nelle animazioni e, in un secondo momento, aggiungere i dettagli "cucinati".

Qui i vantaggi sono molteplici: è possibile iniziare ad animare un modello subito, parallelizzando la fase di animazione a quella di modellazione dei dettagli. In più come già detto, si avrà un frame-rate più alto in fase di animazione, ed un rendering più veloce una volta ultimate le animazioni.

In Figura 5.5 5.6 5.7 e 5.8 sono riportati altri esempi di comparazione tra concept e modello 3D dei modelli da me realizzati. Tutti i concept, a parte quello del tenente, sono stati forniti da Alberto Uras.

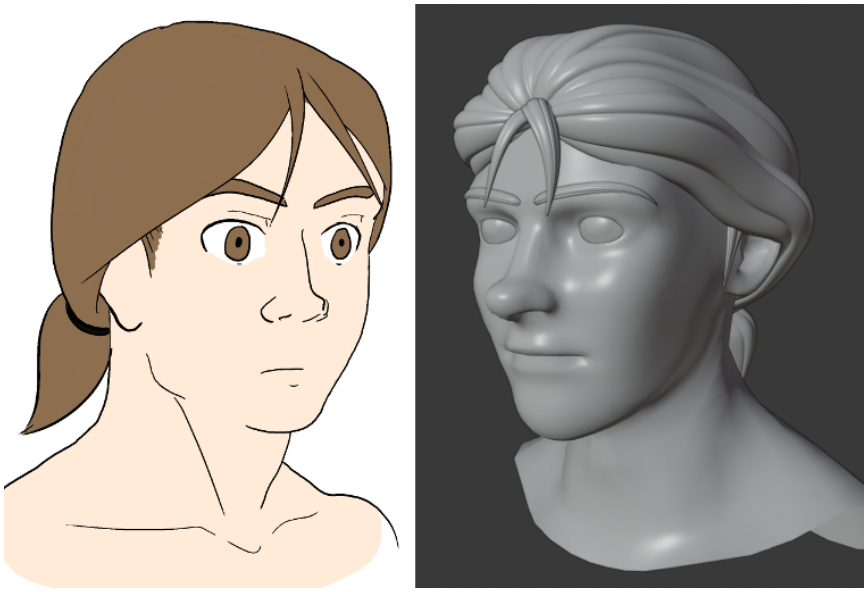


FIGURA 5.5: Modello 3D del volto del ragazzo a confronto con il relativo concept.

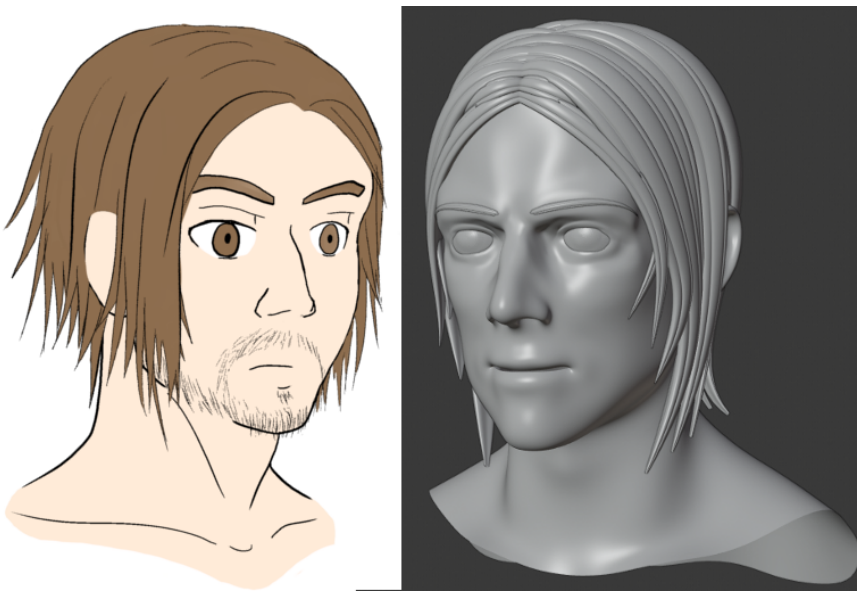


FIGURA 5.6: Modello 3D del volto del capt. Lawrence a confronto con il relativo concept.

FIGURA 5.7: Modello 3D del volto del tenente a confronto con il relativo concept.

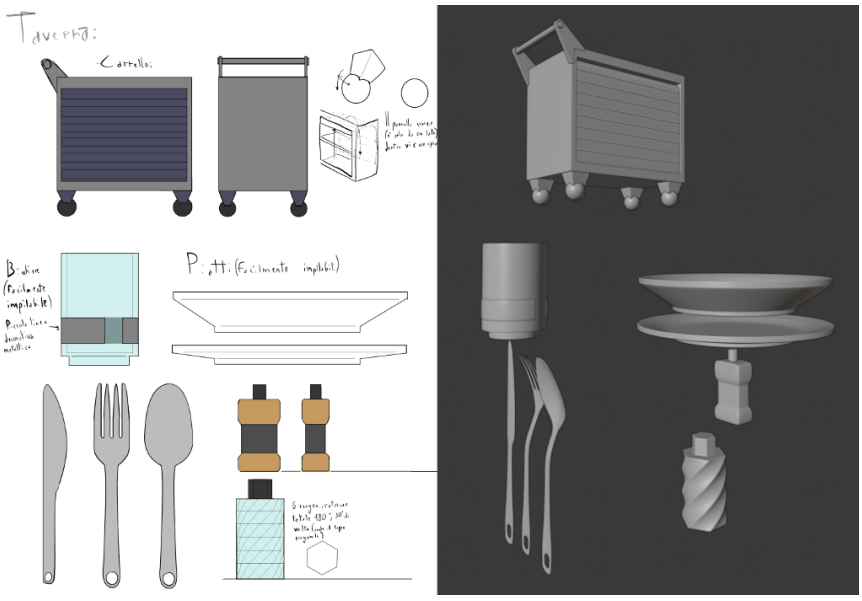


FIGURA 5.8: Alcuni modelli 3D dei prop a confronto con il relativo concept.

5.2 Texturing

Il processo di texturing serve ad applicare un'immagine alla superficie di un oggetto, allo scopo di aggiungergli colore e dettagli di rilievo, detti appunto texture. Chiaramente i modelli rappresentati nelle figure non possono considerarsi completi senza l'aggiunta di colore. Uno degli scopi delle texture è proprio questo, esse infatti permettono di mappare un'immagine sulla superficie della mesh. Per fare ciò però, la mesh deve prima essere stesa su un piano. Questo è possibile farlo in maniera automatica, il problema è che spesso la topologia deve venire spezzata per poterla stendere su un piano. In alternativa, per figure complesse, è consigliato farla manualmente, in modo da limitare i "tagli", che creerebbero discontinuità una volta applicata la texture.

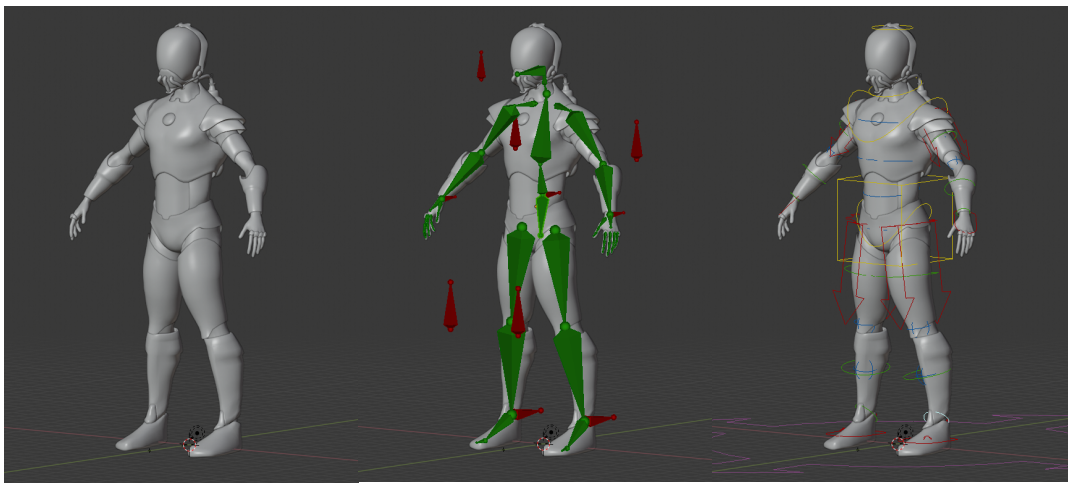
Dopo aver eseguito questa procedura, è possibile creare una texture da associare a ciò che è stato ottenuto. Esistono diverse tecniche per fare ciò:

- pittura manuale;
- generate proceduralmente.
- generata dalla geometria della mesh;

La prima, come suggerisce il nome, è la più artistica. Si tratta fondamentalmente di disegnare sul modello. Questa tecnica, seppure non sia affatto automatizzabile, è la più utilizzata, in quanto spesso non c'è altro modo di aggiungere dettagli irregolari. Essa è ad esempio stata usata per aggiungere il dettaglio ai capelli del capitano, per non dare l'idea che la sua chioma fosse un blocco unico, ma formato da tanti capelli.

Per i capelli di tutti gli altri personaggi, la texture è stata generata proceduralmente (vedi Figura 5.16). Questo è stato possibile poiché, avendo questi altri, dei capelli lunghi, questi sono stati modellati con l'uso di curve in varie ciocche. Dopodiché è stato possibile definire la texture in modo che fosse formata da tante linee che partissero da un estremo della curva all'altro.

Ad ogni modo le texture non sono usate solo per dare colore ad un oggetto. Esse sono utilizzate anche per aggiungere dettagli in rilievo. Attraverso l'uso di particolari immagini, come mappe normali, che utilizzano i colori per rappresentare informazioni come la direzione della superficie, è possibile dare maggiore profondità ad un oggetto. Questa tecnica è detta "baking", come era già stato menzionato in precedenza, permette di aggiungere dettagli ad un oggetto senza aumentarne il numero di vertici. Il concetto che ne sta dietro è quello di pre-calcolare, una sola volta i dettagli, e poi applicarli al modello in bassa risoluzione. In questo caso la texture viene generata direttamente dalla geometria della mesh.



(A) Modello senza controlli. (B) Armatura o meta-rig. (C) Rig avanzato con forme personalizzate.

FIGURA 5.9: In figura è mostrato il modello di uno dei personaggi, con diversi tipi di controlli per l'animazione.

5.3 Animazione e Rigging

Rigging è un termine generale che si riferisce all'aggiunta di controlli ad un oggetto, tipicamente allo scopo di animarlo [1]. Consiste nell'assegnare relazioni tra oggetti [4].

Nel Capitolo 4 è stato visto come progettare un rig orientato all'animazione che bisognerà eseguire. Nella realizzazione di questi si è partiti definendo il meta-rig, aggiungendo tutte le ossa, dalla radice alle foglie, e associando ciascuna alla rispettiva porzione del modello, per permettere una deformazione soddisfacente di quest'ultimo. Per generare il rig avanzato, è stato in parte usato rigify [1]: una tecnologia che permette di automatizzare alcuni processi della realizzazione di un rig. Purtroppo questa tecnologia è stata scoperta verso la fine del progetto e non è quindi stata utilizzata al massimo. Infatti i rig avanzati non sono quasi per nulla stati utilizzati, anche perché, essendo io e Uras già familiari nell'animare direttamente il meta-rig, si è preferito procedere direttamente alla realizzazione delle animazioni. In Figura 5.9 è possibile vedere il rig finito applicato ad un modello.

5.3.1 Keyframe e metodi di animazione

Ora che è possibile animare i modelli deformandoli, si può procedere con la realizzazione delle animazioni, precedentemente descritte al Capitolo 4. Esistono principalmente due metodi di animazione [21]:

- dall'inizio alla fine
- da posizione a posizione

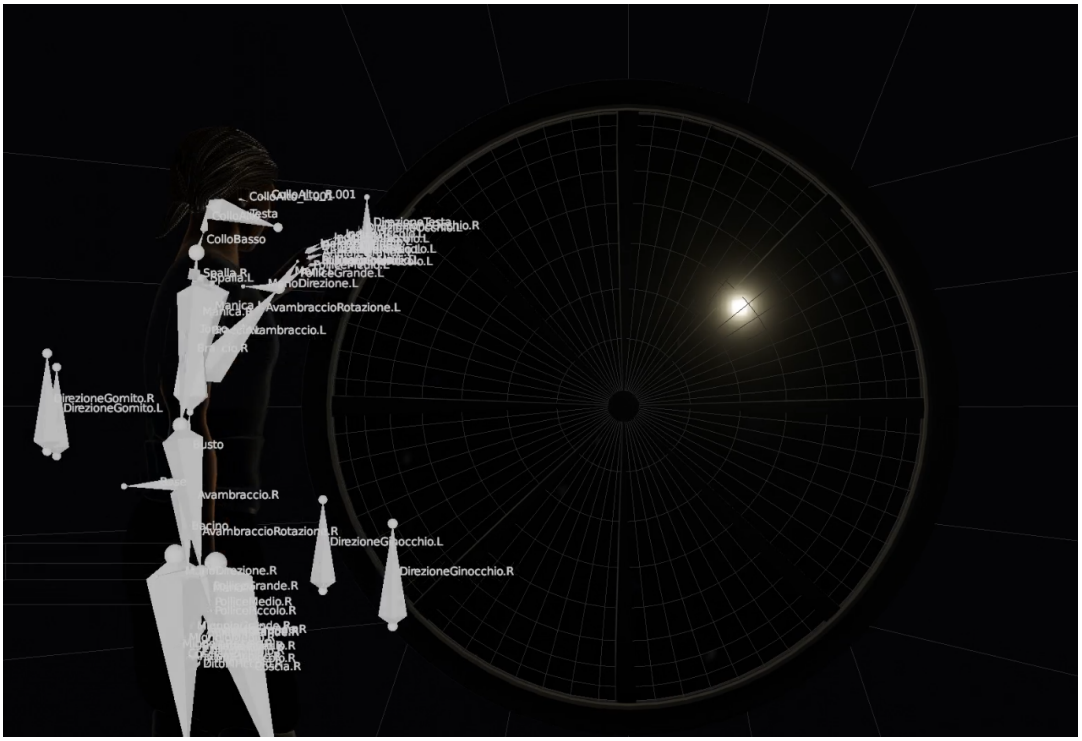


FIGURA 5.10: In figura è rappresentata una delle prime scene, in cui il ragazzo si sveglia.

FIGURA 5.11: In figura sono rappresentati capitano e capitana che si preparano ad uno scontro a fuoco.

FIGURA 5.12: In figura sono rappresentati due robot e la capitana, a mezz'aria, che sferra un calcio ad uno di essi.

Il primo è molto semplice: si parte dal primo frame, dove si posizionano all'interno della scena tutti i personaggi e la telecamera. Poi si passa al secondo animando ogni cosa, poi al terzo e così via. Questo metodo è molto utilizzato nell'animazione tradizionale, soprattutto nella realizzazione dei flip-book. Ha il vantaggio di essere naturale: per arrivare in una determinata posizione devo spostarmi facendo un passo alla volta partendo dal primo punto in avanti. Di conseguenza anche le animazione che ne risultano hanno un aspetto più naturale. Questo metodo però rende difficile pianificare le azioni future per questo motivo non è ottimale.

Al contrario, animare da posizione a posizione permette di pianificare un'intera scena definendo delle posizioni chiave. Queste ultime, anche dette *key-frame*, provengono direttamente dallo storyboard e definiscono cosa deve accadere in una scena. Definite queste posizioni principali, distribuite nella sequenza dei frame della scena, si possono aggiungere i frame di intermezzo tra due posizioni. Nell'animazione digitale questo metodo è il più efficiente in quanto i frame di intermezzo possono venire automaticamente calcolati come interpolazione delle due posizioni. Il lato negativo è che le animazioni risultanti non sono molto naturali: spesso è necessario rendere un personaggio più veloce o più lento, per farlo arrivare in un ponto al tempo giusto e rispettare i piani preposti.

Per realizzare tutte le animazioni di questo cortometraggio è stato fatto ovviamente uso di quest'ultimo metodo. Va aggiunto però che, per rendere le animazioni più naturali, dopo aver definito tutti i frame di intermezzo. Questi ultimi sono stati modificati con un approccio dal primo verso l'ultimo. In questo modo l'azione è più naturale, e spezza la linearità dell'interpolazione

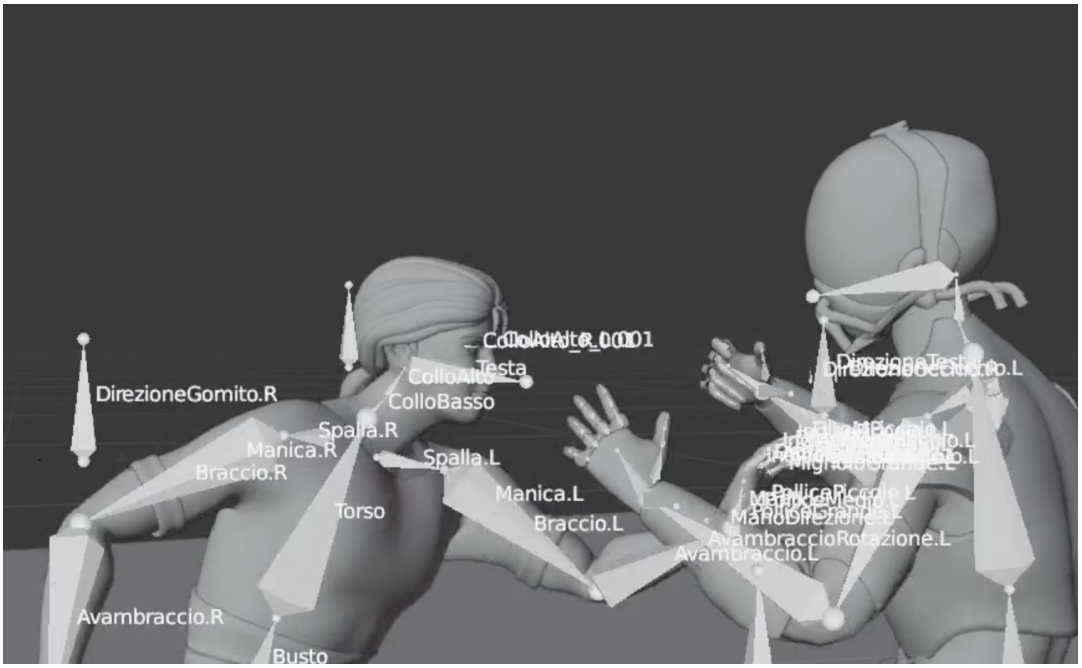


FIGURA 5.13: In figura è rappresentato il ragazzo che spinge uno dei banditi spaziali.

che, essendo calcolata dal computer, non mostra mai un movimento realistico fin da subito. È quindi più giusto dire che nella realizzazione del trailer, è stato fatto uso di una combinazione delle due tecniche.

5.3.2 Animazioni cicliche

Inoltre, per le animazioni più comuni, esse sono state rese cicliche, per poterle iterare per la durata necessaria, e riutilizzarle in ogni scena necessaria. In questo modo le animazioni sono state modularizzate, ovvero spezzate in più azioni ripetibili. Alcune animazioni come la camminata, infatti, seguono dei pattern ben definiti, costituiti da varie fasi e, grazie alla loro uniformità è possibile renderle cicliche in maniera tale da non dover rifare la stessa animazione più volte.

Nel caso del ciclo della camminata queste fasi sono dette a supporto singolo e doppio supporto [14]. Quest'ultima inizia con la posa di *contatto*, in cui il secondo piede poggia a terra col tallone, e termina con la posizione di passaggio. Viceversa la fase a singolo supporto inizia dalla posizione di passaggio e termina con la posizione di contatto. Aggiungendo un solo frame di intermezzo per ogni fase, rispettivamente posa bassa (doppio contatto) e posa alta (contatto singolo), si ottiene un passo completo. Ripetendo il tutto per l'altra gamba l'animazione può considerarsi ultimata. I restanti frame verranno calcolati tramite interpolazione.

La corsa è molto simile alla camminata, ma differisce nella durata delle fasi

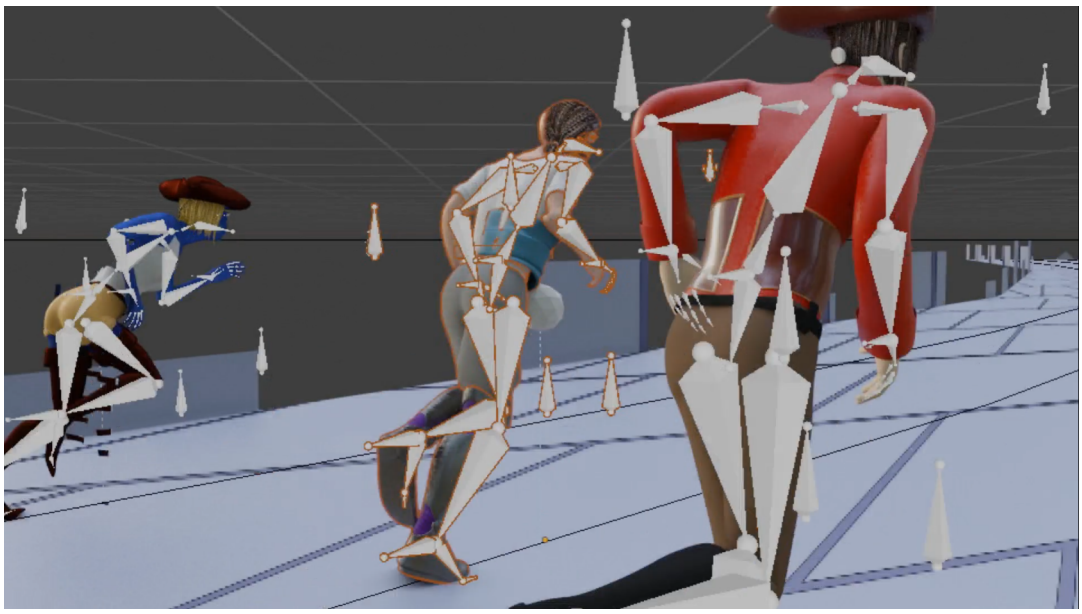


FIGURA 5.14: In figura sono rappresentati Capitana, ragazzo e Capitano mentre corrono, per seminare i loro inseguitori.

e nel fatto che, a differenza della camminata, dove almeno un piede poggia sempre a terra, in questo entrambi i piedi non sono mai a contatto col suolo nello stesso momento. Le due fasi sono quindi chiamate supporto singolo e volo, in cui nessun piede si trova a contatto col terreno.

Un volta ottenuta l'animazione ciclica, è possibile inserirla in ogni scena necessaria. Nel caso di una camminata, è inoltre necessario spostare il modello mentre cammina. Il modo migliore per farlo è quello di definire un percorso, che il modello dovrà seguire durante l'animazione. Il percorso non è altro che una curva, solitamente NURBS o B-spline. Il vantaggio di usare le curve in questo caso è che il percorso risultante avrà curve molto soffici. Inoltre è possibile, attraverso i punti di controllo, modificare la curva a posteriori e, con essa, il modello si sposterà di conseguenza.

5.3.3 Espressioni e Dialoghi

Un altro aspetto importante delle animazioni, soprattutto in questo progetto, sono i dialoghi. Nonostante si sia scelto di non doppiare i dialoghi dei personaggi, ma piuttosto tenere una traccia audio di sottofondo, per aggiungere drammaticità, è comunque stato necessario animare le facce dei personaggi per permettergli di parlare e esprimere sentimenti attraverso espressioni facciali. Per fare ciò sono state utilizzate due tecnologie: *shape-keys* [1] e *Rhubarb* [22].

Siccome la faccia è un concentrato di muscoli servirebbe un rig con centinaia di DOF. Avere molte ossa per controllare ogni muscolo non è conveniente,

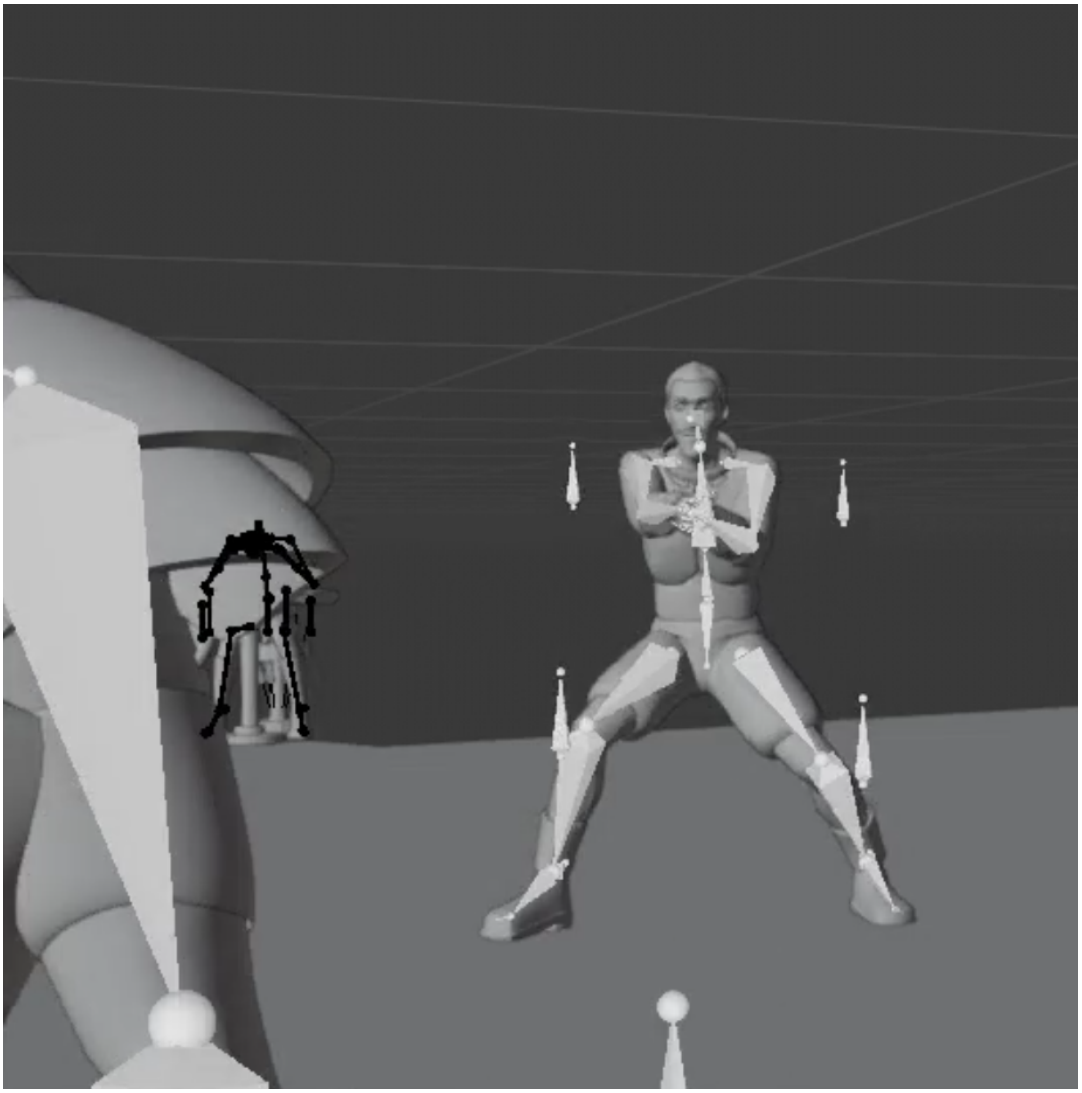


FIGURA 5.15: In figura è rappresentato il tenente mentre spara ad uno dei banditi spaziali.

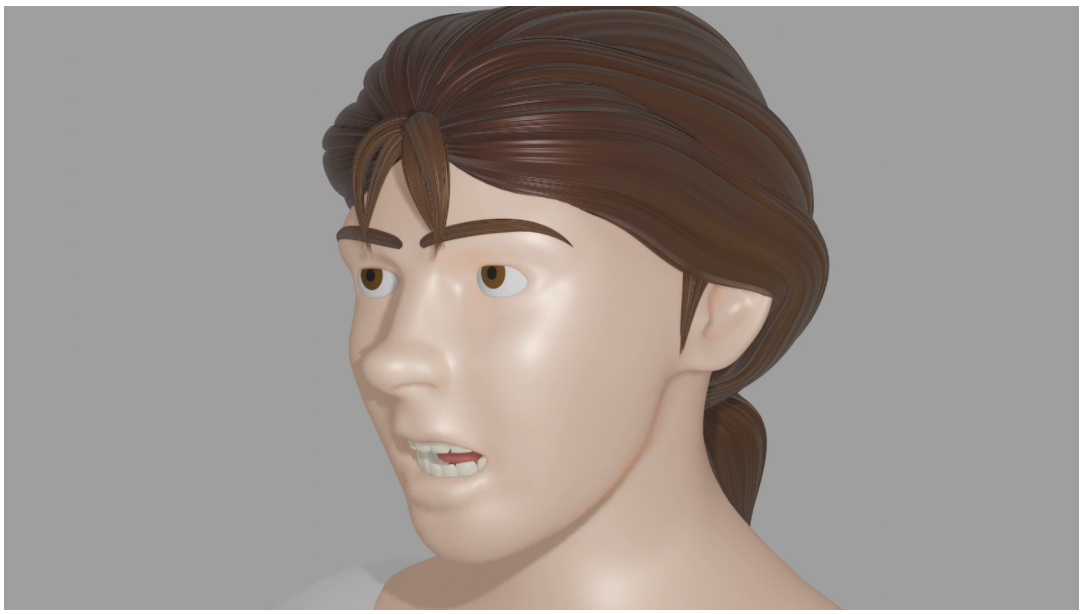


FIGURA 5.16: In figura è rappresentato il ragazzo mentre viene animato per un dialogo

poiché sarebbe difficile da animare. Per semplificare il processo di animazione è conveniente parametrizzare ogni muscolo con una semplice variabile di range $0 - 1$ dove 0 indica il muscolo rilassato e 1 contratto. Tuttavia a causa dell'elevato numero di muscoli presenti nella faccia sarebbe comunque difficile animare una singola espressione. Un modo per ovviare a questo problema e quello di usare appunto delle shape-keys, conosciute anche come blend-shapes, dove ognuna rappresenta una posa della faccia (e.g. triste, felice, arrabbiato, sorpreso...), o meglio dividere ogni posa in aree della faccia e poi parametrizzarle e combinarle in modo che la loro somma sia uguale a 1. Questo metodo è molto restrittivo rispetto al grado di controllo che ha l'animatore, poiché le espressioni rappresentabili dipendono dalle shape-keys definite precedentemente (e dalle loro combinazioni interpolanti). Nonostante ciò rappresenta un buon compromesso tra semplicità d'uso e numero di espressioni rappresentabili.

Per quanto riguarda i dialoghi, Rhubarb Lyp Sync, permette di animare la bocca, e le aree limitrofe del viso, attraverso una traccia audio e scritta del dialogo da riprodurre. Ciò è possibile perché a ogni fonema corrisponde un visema (i.e. espressione della faccia). Per tanto, è stato sufficiente modellare un set limitato di espressioni e associarle ad il rispettivo fonema, registrare l'audio, e Rhubarb ha fatto il resto. Come riportato da Parent [14], esistono 42 diversi fonemi. Tuttavia molti di questi possono essere rappresentati dallo stesso visema, o da una interpolazione di questi. Inoltre Williams afferma che non è necessario animare una frase in ogni sua singola sillaba, è sufficiente selezionare quelle più evidenti [21]. Questo ha permesso di ridurre il numero di shape-keys necessarie per i dialoghi a 9 che sono anche quelle utilizzate

da Rhubarb.

5.4 Illuminazione

Quello dell'illuminazione è un passaggio fondamentale per poter renderizzare una scena. Infatti senza luci il risultato sarebbe ovviamente quello di una schermata nera. Non solo, esistono diversi tipi di luce [16], e da essi, oltre che dal loro posizionamento, l'aspetto di una scena può cambiare completamente.

Esistono fondamentalmente due categorie di settaggio di luci: per ambienti e per soggetti in primo piano. Quest'ultimo è quello solitamente utilizzato dai fotografi nei loro studi fotografici: l'obiettivo è quello di creare una luce ad hoc, che risalti le forme del soggetto che si vuole rappresentare. Questo è molto importante perché nella fotografia, così come nel rendering, si passa da una scena in tre dimensioni a una sua rappresentazione bidimensionale. È quindi facile perdere le informazioni di profondità, e le ombre servono proprio a questo. Solitamente per raggiungere questo scopo si utilizza un sistema a tre luci [17].

L'illuminazione di un ambiente può spesso risultare più complessa. È importante indirizzare la luce per evidenziare cosa si vuole mostrare. Il tutto è reso complesso dal fatto che ogni luce deve avere un contesto: aggiungere una luce che fluttua a mezz'aria non sarebbe realistico, bisogna che provenga da una fonte luminosa, come ad esempio una lampada. Tutto comunque dipende dal contesto: non avrebbe senso aggiungere una lampada da tavolo in una scena all'esterno.

Indipendentemente dal tipo di scena che si vuole realizzare, il metodo in cui la luce viene calcolata per mostrare oggetti o oscurare l'ambiente circostante con ombre è sempre lo stesso, e dipende dal motore di renderizzazione.

5.5 Renderizzazione

Il processo di renderizzazione è ciò che permette di proiettare la scena tridimensionale in un'immagine in una finestra contenuta nello schermo bidimensionale [11]. Per fare ciò, è necessario definire una camera virtuale, attraverso un punto nello spazio tridimensionale che ne definisce la posizione, un vettore che indica la direzione in cui essa è orientata, un angolo che definisce il campo visivo ed infine la profondità oltre la quale non è necessario renderizzare.

Attraverso una serie di trasformazioni, ogni oggetto presente nella scena viene proiettato nel sistema di riferimento della telecamera. Esiste comunque più di un modo per calcolare queste trasformazioni, due di questi sono rasterizzazione e ray-tracing.

Nel primo caso il calcolo è incentrato sugli oggetti presenti nella scena. Per ognuno di essi viene tracciato un raggio che parte da ogni vertice verso la telecamera. Collegando i punti sulla griglia di visualizzazione, è possibile capire quale area dello schermo l'oggetto ricopre. Dopodiché attraverso un buffer di profondità viene calcolato quali oggetti sono più vicino alla camera e quindi coprono gli oggetti più lontani. Questo è un processo relativamente veloce perché il numero di raggi è limitato al numero di vertici presenti nella scena. Di fatti, questo è il metodo che fino a qualche anno fa veniva usato in tutti i videogiochi per permettere una renderizzazione in tempo reale. Questo spiega anche perché nei videogiochi ci sia la tendenza a tenere un numero di vertici molto più basso rispetto ad un film realizzato con la computer grafica. Tuttavia, questo metodo presenta lo svantaggio di non riuscire a calcolare le luci in maniera realistica.

Il secondo metodo invece è incentrato sulla vista: dalla telecamera viene proiettato un raggio che interseca la griglia di visualizzazione per ogni pixel dell'immagine risultante. Questi raggi, detti primari, continuano in linea retta finché non trovano un oggetto. Dal punto di incidenza con l'oggetto, dipendentemente dalle proprietà fisiche di quest'ultimo, partono altri raggi, alcuni sono quelli secondari, che si dirigono direttamente verso le sorgenti luminose, per capire se l'oggetto è illuminato direttamente, oppure esiste un altro oggetto che proietta un ombra. I restanti sono raggi di riflessione/rifrazione e permettono di calcolare la luce ambientale o indiretta che non era possibile calcolare nella rasterizzazione. Per via dell'alto numero di computazioni necessarie, questo è un metodo molto più lento rispetto al primo, ma permette di raggiungere risultati molto più fotorealistici.

Recentemente comunque quest'ultimo metodo, è stato velocizzato a tal punto da poter essere implementato nei videogiochi, con una renderizzazione in tempo reale. Il primo invece è stato sviluppato a tal punto da ottenere immagini fotorealistiche, quasi quanto il secondo. Blender propone due motori di renderizzazione, ognuno implementante uno di questi due metodi, rispettivamente Eevee e Cycles.

Nella realizzazione di questo cortometraggio è stato scelto di utilizzare Eevee come motore di renderizzazione. Il motivo di questa scelta deriva dal fatto che ciò ci ha permesso di accorciare notevolmente i tempi di rendering, al costo di un risultato meno realistico, che comunque non era uno dei requisiti.

Capitolo 6

Conclusioni

6.1 Risultati

Questo progetto si può considerare come il più ambizioso, che io abbia mai realizzato in ambito di Computer Graphics, e la prima volta in cui lo scopo non era semplicemente di testare qualcosa, ma di arrivare ad un prodotto che soddisfacesse dei requisiti ben definiti. Molte delle tecniche e dei processi usati per realizzare il progetto mi erano, fin'ora, completamente sconosciuti. Ciò ha richiesto un notevole quantitativo di tempo solo per apprendere tutto ciò che c'era di nuovo, in modo da farne uso in maniera ottimale. C'è da dire che tutto ciò che è stato appreso ha portato risultati notevoli nel progetto e, senz'altro, tornerà utile in futuro, qualora dovessi realizzare qualsiasi cosa riguardante la grafica 3D.

Il trailer ottenuto è stato utile a comprendere molti aspetti teorici che sono stati solo leggermente visti durante il corso di Computer Graphics. Inoltre, esso rappresenta un pezzo importante da aggiungere al mio portfolio, in quanto rappresenta molto bene tutte le mie abilità riguardanti la grafica 3D. Realizzare un prodotto attraversando tutte le fasi di un progetto è stato anche utile a capire come ottimizzare il processo e vedere applicazioni di quanto appreso nel mondo reale.

6.2 Lavori futuri

Come già detto, il trailer realizzato è da considerarsi già un ottimo traguardo. Ovviamente dopo la realizzazione di un trailer il passo successivo è quello di realizzare un intero film o una serie ad episodi. In questo caso si è già pensato a proseguire con una serie, il cui nome sarebbe *"Space Wanderer"*.

Oltre a continuare la serie, che sarebbe soprattutto un buon allenamento e un'interesse personale, ho intenzione di continuare sulla via della computer grafica con un Master. Ho infatti già intrapreso la via per continuare i miei studi iscrivendomi alla Kungliga Tekniska Högskolan (KTH) con un Master of Science in Computer Science, orientato su Visualization and Interactive Graphics.

Il mio obiettivo a lungo termine è quello di diventare un Technical Animator. Questa posizione è una buona combinazione tra le abilità informatiche come problem solving, applicate alla grafica. L'obiettivo di un animatore tecnico è infatti quello di aiutare gli artisti a fare il loro lavoro, scollegandoli dagli aspetti tecnici delle grafica e delle animazioni. In pratica si tratta di applicare concetti di programmazione a oggetti come quello dell'information-hiding nel campo delle animazioni. Questa è una posizione che mi interesserebbe molto ricoprire, in quanto mi permetterebbe di lavorare con ciò che mi appassiona, come la grafica e le animazioni, pur facendo uso di tutto ciò che ho appreso dall'ingegneria informatica.

Bibliografia

[1] *Blender 2.80 Manual*. Blender Foundation. URL: https://docs.blender.org/manual/en/2.80/animation/introduction.html#rigging.

[2] *Blender 2.80 Python API*. Blender Foundation. URL: https://docs.blender.org/api/current/mathutils.html.

[3] *Blender Code Documentation*. Blender Foundation. URL: https://wiki.blender.org/wiki/Source.

[4] *Blender Fundamentals 2.8*. [Consultato il 24 Ottobre 2019]. Blender Foundation, lug. 2019. URL: https://www.youtube.com/watch?v=-gIL6VZ-bkE&list=PLa1F2ddGya_-UvuAqHAksYnB0qL9yWDO6&index=28.

[5] S. Buss. «Introduction to Inverse Kinematics with Jacobian Transpose, Pseudoinverse and Damped Least Squares Method». In: *Unpublished survey* (lug. 2006). URL: https://www.math.ucsd.edu/~sbuss/ResearchWeb/ikmethods/index.html.

[6] S. Buss. «Selectively Damped Least Squares for Inverse Kinematics». In: *Journal of Graphics Tools* 10.3 (2005), pp. 37–49. URL: https://www.math.ucsd.edu/~sbuss/ResearchWeb/ikmethods/index.html.

[7] Thomas Denham. *What is 3D Hard Surface & Organic Modeling?* [Consultato il 17 Ottobre 2019]. URL: https://conceptartempire.com/hard-surface-organic-modeling/.

[8] The Editors of Encyclopaedia Britannica. *Linear programming*. [Consultato il 22 Novembre 2019]. Lug. 2017. URL: https://www.britannica.com/science/linear-programming-mathematics.

[9] E.G. Gol'shtein. «Simplex method». In: (feb. 2011). [Consultato il 22 Novembre 2019]. URL: http://www.encyclopediaofmath.org/index.php?title=Simplex_method&oldid=18646.

[10] Claire Heginbotham. *What is 3D Digital Sculpting?* [Consultato il 17 Ottobre 2019]. URL: https://conceptartempire.com/what-is-3d-sculpting/.

[11] Damiana Lazzaro. *Rendering Pipeline - dispense delle lezioni*. URL: https://iol.unibo.it/pluginfile.php/264133/mod_unibores/content/0/Lezione16Ottobre.pdf.

[12] Aristide Mingozzi. *Simplesso Duale - dispense delle lezioni*.

[13] Aristide Mingozzi. *Simplesso Revisionato - dispense delle lezioni*.

[14] Rick Parent. *Computer Animation: Algorithms and Techniques*. 3ª ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2012. ISBN: 9780124158429, 9780124159730.

[15] Kenny Roy. *Finish your film! Tips and tricks for making an animated short in maya*. Routledge, 2014.

[16] Thomas Tanner. *What is a 3D Lighting Artist? (A Beginner's Guide On What They Do)*. [Consultato il 17 Ottobre 2019]. URL: https://conceptartempire.com/lighting-artist/.

[17] *Three-Point Lighting*. URL: https://courses.cs.washington.edu/courses/cse458/05au/reading/3point_lighting.pdf.

[18] Nathan Vegdahl. *Humane Rigging*. [Consultato il 1 Novembre 2019]. Blender Foundation, 2012. URL: https://www.youtube.com/playlist?list=PLE211C8C41F1AFBAB.

[19] *What is Gimbal Lock and why does it occur?* [Consultato l' 8 Novembre 2019 (archiviato dall'url originale il 24 giugno 2016)]. URL: https://web.archive.org/web/20160624170903/http://www.anticz.com/eularqua.htm.

[20] Elmer G. Wiens. *Simplex Algorithm*. [Consultato il 22 Novembre 2019 (archiviato dall'url originale il 18 maggio 2006)]. URL: https://web.archive.org/web/20060518080331/http://www.egwald.com/operationsresearch/lpsimplex.php.

[21] Richard Williams. *The Animator's Survival Kit–Revised Edition: A Manual of Methods, Principles and Formulas for Classical, Computer, Games, Stop Motion and Internet Animators*. Faber & Faber, Inc., 2009. ISBN: 0571238343, 9780571238347.

[22] Daniel Wolf e Scaredyfish. *Rhubarb Lip Sync*. URL: https://github.com/scaredyfish/blender-rhubarb-lipsync.