POLITECNICO DI TORINO

Collegio di Ingegneria Informatica, del Cinema e Meccatronica

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

Tesi di Laurea Magistrale

Rigging di un Personaggio Orientato al Motion Capture per un Prodotto di Animazione Indipendente

Relatore
prof. Antonino Riccardo Antonio Silvio

Candidato
Stefania Abate

Dicembre 2019

“Quando rovisto nel passato trovo dolore e morte, quando cerco nel futuro ansia e illusioni, ma quando frugo nel presente, trovo questo presente, questo qui, questo momento da cui ti scrivo ora, da cui ora tu leggi, questo presente infinito, luminoso, e c'è sempre la musica, ci sei tu, i tuoi occhi, c'è zio Luciano che sorride. Smetti di girare in tondo Sergio, torna a cercare le tue note migliori, alla prossima fratellino, ti amo, Alfredo”

L’Arte della Felicità

I. Abstract

Questa tesi di laurea è stata svolta presso l'azienda Robin Studio S.r.l., in particolare per la realizzazione di un teaser trailer per la serie d'animazione Reverie Dawnfall. Lo scopo di questa tesi è quello di realizzare il Rig ottimale di un modello 3D, adattabile e trasferibile a tutti i personaggi della serie, in modo da poter utilizzare la tecnica del Motion Capture come base per l'animazione e tramite il programma open source Blender.

Il progetto di tesi è diviso in tre fasi. La prima è una fase di ricerca sulle tecniche di Rigging e Motion Capture esistenti soprattutto nell'industria indipendente.

La seconda consiste nella realizzazione effettiva del Rig sul personaggio principale della serie. In questa fase si sono presentate delle problematiche, seguite dallo studio del problema ed infine dalla risoluzione delle stesse. La terza fase si è concentrata sull' applicazione delle take di Motion Capture al Rig effettuato.

L'obiettivo ultimo del mio lavoro è quello di trovare la tecnica migliore di Rig per la realizzazione del trailer e, in seguito, di tutta la serie, tenendo conto dei limiti tecnici e di risorse di una produzione indipendente.

Indice

1. Introduzione

Rigging è un termine usato per descrivere la creazione di un sistema che permette agli animatori di animare. È il meccanismo di controllo che non si vede, per lo meno dallo spettatore. A sua volta, la parola *Rigger* descrive comunemente quella persona che prende un modello 3D inanimato e crea i controlli con cui l'animatore potrà lavorare. Ci sono molti modi per indicare questa figura all'interno dell'industria, e cambiano da studio in studio. Alcuni sono chiamati *Rigger,* altri sono noti come *Setup Artist, Technical Animator, Technical Artist,* ecc. per lo scopo di questa tesi verrà utilizzato il termine Rigger.

Il Rigger ha un ruolo importante nel definire il modo in cui la pipeline procede durante una produzione. Infatti, a volte è interamente responsabile della pipeline. Ogni produzione è diversa; l'abilità di adattarsi e creare è definitivamente un tratto necessario per eccellere in questo ruolo.

Ci sono diversi modi per effettuare il rig. Uno di questi è il rig customizzato, ovvero il rig realizzato da zero, aggiungendo un osso alla volta fino a raggiungere una struttura, simile allo scheletro umano, con la conseguente creazione di relazioni tra le varie ossa e la successiva associazione, sempre manuale, tra le ossa e il modello 3D. Un altro metodo è quello del rig automatico, ovvero la realizzazione di uno scheletro tramite un software che crei automaticamente le ossa e le conseguenti relazioni tra esse, con conseguente skinning (l'associazione tra ossa e modello di cui prima) automatizzato.

Questa tesi si concentra sul paragonare un rig semi-customizzato, BlenRig, e un rig automatico fornito dall'add-on di Blender Auto-Rig Pro. In particolare, verranno riggati e animati due personaggi, Nadya e Jameela, la prima con BlenRig e la seconda con Auto-Rig. Verranno dunque esposte le caratteristiche di entrambi i rig, i loro punti di forza e i loro punti di debolezza, confrontando anche i diversi risultati ottenuti dopo l'applicazione del motion capture, che è la tecnica di animazione scelta per questo progetto.

Questo progetto di tesi, inoltre, è parte di un progetto più grande, ovvero la creazione di un teaser trailer per la serie d'animazione low budget Reverie Dawnfall, prodotta dallo studio Robin S.r.l.s.

Nadya e Jameela, sono appunto due dei personaggi principali della serie. In particolare, all'interno del teaser, Nadya è la protagonista e Jameela è il personaggio secondario.

1.1 Motivazione

Ciò che ha portato alla scelta di questo argomento di tesi, è la mia personale passione per il mondo dell'animazione e del cinema in generale, che ha influito sulla scelta del corso di laurea triennale in Ingegneria del Cinema e dei Mezzi di Comunicazione, e successivamente del corso magistrale, e che si è consolidata durante il percorso di studi, soprattutto grazie ai corsi di *Tecniche e linguaggi del Cinema, Computer Grafica* e *Computer Animation*. Inoltre, avrei voluto sviluppare una tesi che unisse una parte tecnica, quindi ingegneristica, a una parte creativa, ed è per questo che quando si è presentata l'occasione di lavorare per il progetto Reverie Dawnfall, che ci è stato presentato durante il corso di Effetti Speciali dal prof. Antonino, non ho potuto far altro che coglierla.

Pertanto, il lavoro di tesi ha avuto il seguente sviluppo. Sono entrata a far parte del team di produzione in fase di preproduzione. L'obiettivo del team era quello di creare un secondo teaser trailer da utilizzare per pubblicizzare la serie, cercare fondi, partecipando anche a bandi nazionali e internazionali. Il primo teaser era stato realizzato un paio d'anni prima, ma si era deciso un cambio di stile e un cambio modelli sia di personaggi che di ambienti, approfittando delle nuove tecnologie uscite negli ultimi anni, in particolare il nuovo motore di render real-time *eevee* di Blender 2.8. Una volta stabilito il team di lavoro, si sono svolte una serie di riunioni in cui si sono stabiliti stile visivo, topologia degli ambienti, script e storyboard del teaser. Successivamente il team ha lavorato parallelamente alla creazione dello shader, degli ambienti, dei personaggi e del rig. Scoperta la possibilità di lavorare con BlenRig, è iniziato il mio processo di rig, in cui sono sorte non poche problematiche dovute soprattutto alla pesantezza del modello in termini di poligoni. È stato dunque un processo *trial&error* che alla fine ha portato al risultato sperato, e anche un processo di crescita professionale e personale.

1.2 Outline

I capitoli di questa tesi sono suddivisi come segue:

- **Capitolo 2:** viene spiegato il processo di rigging, e i principi teorici che ne permettono il funzionamento, con un'attenzione particolare per il concetto di cinematica. Viene inoltre mostrato in cosa consiste il processo di rigging all'interno di Blender, software di modellazione e animazione 3D utilizzato per questo progetto.

- **Capitolo 3:** viene esposta brevemente la tecnica del motion capture, utilizzata come base per l'animazione del teaser, in particolare ci si concentra sul funzionamento del sistema inerziale di motion capture fornito dalla Smartsuit Pro dell'azienda Rokoko e dell'applicazione per il motion capture facciale fornita dalla stessa.
- **Capitolo 4:** si presentano le caratteristiche e il funzionamento delle due tecniche di rigging utilizzate per i personaggi del teaser, esponendo ogni passaggio effettuato per la creazione degli stessi.
- **Capitolo 5:** viene trattato il procedimento di animazione dei personaggi, sia del corpo che della faccia, applicando le take di motion capture prima tramite remap, e migliorando il movimento tramite lo spostamento delle ossa.
- **Capitolo 6:** vengono paragonati i risultati ottenuti dalle due diverse tecniche di rig ed esposti i punti di forza e debolezza di entrambi.

2. Rigging

2.1 Skeletal Animation

La *Skeletal Animation* o *Skinning* è una tecnica della computer animation in cui un personaggio, o qualsiasi oggetto articolato, è rappresentato in due modi: una rappresentazione superficiale usata per disegnare il personaggio (*mesh*) e un gruppo di ossa interconnesse in maniera gerarchica usate per animare la mesh (*rig)*. Poiché questa tecnica è spesso usata per animare personaggi umani o, più generalmente, per una modellazione organica, essa serve per rendere il processo di animazione più intuitivo, e la stessa tecnica può essere utilizzata per la deformazione di qualsiasi oggetto.

2.2 Rigging

Il rigging è una tecnica usata nella Skeletal Animation per rappresentare il modello di un personaggio 3D utilizzando una serie di ossa digitali interconnesse.

Più precisamente, rigging si riferisce al processo di creazione di una struttura ossea di un modello 3D. Questa struttura viene utilizzata per manipolare il modello 3D come un puppet per l'animazione.

Può essere "riggata" praticamente qualsiasi cosa, da una nave spaziale a un soldato, a una galassia, a una porta. Non importa quale sia l'oggetto. Aggiungere ossa permette di animare liberamente ogni cosa.

Il rig è più comune nei personaggi animati per giochi e film. Questa tecnica semplifica il processo d'animazione e migliora l'efficienza della produzione. Una volta eseguito il rig con le ossa, qualsiasi oggetto 3d può essere deformato secondo le necessità.

Nell' industria dell'intrattenimento il rig è lo step principale, nella modalità standard di animazione di un personaggio. Raggiungere animazioni complesse e regolari dipende interamente dalla qualità della fase di rig nella pipeline d'animazione.

2.2.1 Come funziona

Dopo che il modello 3D viene creato, una serie di ossa è costruita rappresentando la struttura ossea. Per esempio, in un personaggio ci sono un gruppo di ossa per la schiena, per la colonna vertebrale e per la testa.

Queste ossa possono essere trasformate usando un software di animazione digitale, il che significa che possono essere cambiate la loro posizione, rotazione e scala.

"Registrando" le posizioni di queste ossa lungo una timeline (secondo il processo definito *keyframing*) si generano le animazioni.

Un setup base potrebbe richiedere poche ore, mentre un rig complesso per un film può richiedere diversi giorni di lavoro.

Il processo di rigging si evolve in una struttura gerarchica in cui ogni osso è "imparentato" con le ossa a cui è connesso. Ciò semplifica il processo di animazione, infatti quando un artista muove l'osso di una spalla, il braccio e la mano si muoveranno di conseguenza. L'obiettivo è quello di simulare la mimica reale umana nel modo più accurato possibile.

Il modo in cui il modello interagisce con le ossa è determinato da una scala di pesi. Il peso controlla quanta influenza ha un osso su una parte della mesh. In questo modo la sensibilità della deformazione può essere regolata per un'animazione più precisa.

Il cosiddetto *Weight painting* è parte integrante del processo di rigging. Spesso è possibile eseguire un'assegnazione automatica del peso, ma a volte il risultato è insufficiente e poco funzionale.

Per ottenere una buona animazione è essenziale ottimizzare il peso di ogni osso. Poiché alcuni personaggi condividono la stessa struttura ossea, interi rig possono essere copiati e assegnati a nuove mesh. Così possono essere copiate anche le animazioni rendendo più facile produrre modelli con design simili.

Posizionare le ossa è la parte più semplice del rigging. Una volta piazzate, molte ossa avranno bisogno di un lavoro aggiuntivo per far sì che siano animate in modo corretto.

Riggare un personaggio, di solito richiede all'artista di aggiungere una cinematica inversa (IK) alle ossa, che annullerà le proprietà predefinite della cinematica diretta (FK). Entrambe le tecniche verranno illustrate nei prossimi paragrafi.

La cinematica inversa è usata principalmente per braccia e gambe, o altre estremità come la coda. Un buon setup di IK farà sì che i gomiti e le ginocchia puntino nella giusta direzione e permetterà all'animatore di ottenere più facilmente un movimento realistico.

Un altro elemento essenziale di qualsiasi rig è la presenza dei *Constraint,* ovvero i vincoli tra le ossa. Per creare animazioni fluide, alcune ossa hanno bisogno di avere delle restrizioni applicate al loro movimento, per esempio alcune ossa possono muoversi in una sola direzione o ruotare lungo un solo asse.

2.2.2 Come viene usato

Il rigging è una tecnica comune nell'animazione di personaggi in videogame, programmi televisivi e film.

Anche oggetti meccanici, come molle e porte, possono essere animati tramite la skeletal animation una volta che il rig è stato assegnato al modello 3D.

Sulla struttura ossea possono anche essere eseguite delle simulazioni fisiche. Il computer applicherà la fisica alle ossa del rig e registrerà il risultato su una serie di frame. Queste simulazioni anatomiche virtuali sono utili anche al di fuori dal contesto artistico e dall'industria dell'intrattenimento. Ad esempio, le industrie mediche e scolastiche spesso richiedono delle visual per insegnamenti e dimostrazioni.

Per un'animazione facciale di buona qualità sono spesso necessari scheletri complessi. Creare un rig facciale soddisfacente richiede a volte un progetto separato da quello del corpo.

Spesso è più opportuno usare tecniche al di fuori della tradizionale struttura ossea quando si fa il rig facciale, come ad esempio la *Morph target animation* o le *blend shapes,* che saranno trattate nei prossimi paragrafi.

Il vantaggio del rigging è quello di permettere un controllo sulla deformazione del modello, rendendo relativamente semplice l'animazione di un personaggio. Lo svantaggio di questa tecnica è che non funziona bene per l'animazione dei dettagli e può richiedere tantissimo tempo per progetti complessi.

2.3 Cinematica

Per controllare l'animazione del movimento di una figura articolata, esistono due tipi di approcci: quello dinamico e quello cinematico. In questa tesi, come accennato in precedenza, verrà esaminato l'approccio cinematico, in quanto anche il metodo scelto per animare Nadya, BlenRig, è dotato di un sistema di cinematica.

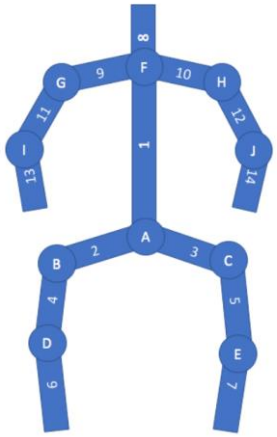
La cinematica è una branca della meccanica classica e descrive il moto di punti, oggetti, e sistemi di oggetti, senza considerare le forze che causano il loro movimento. Si occupa del calcolo di valori come posizione, velocità e accelerazione dei punti all'interno del sistema, note la geometria e le condizioni iniziali.

In animazione la cinematica si divide in due metodi: diretta e inversa.

Prima di proseguire con l'illustrazione dei due metodi, è utile specificare la definizione di alcuni concetti.

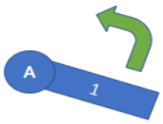
2.3.1 Corpo articolato

Un corpo articolato può rappresentare la maggior parte dei corpi, come umani e animali, provvisti di scheletri. Un corpo articolato può essere rappresentato come un albero di catene connesse. Queste sono costituite da nodi e archi. La figura seguente mostra un esempio di corpo articolato che rappresenta un umanoide.



Corpo articolato umanoide

Esistono due tipi di giunti. Il primo è il *giunto rotante*, in cui il nodo è connesso a un arco che ruota attorno a esso. È simile al centro di un orologio. Il *giunto prismatico* è fatto in modo tale che l'arco connesso al nodo trasla da esso accorciandosi o estendendosi. Può essere paragonabile alla maniglia allungabile di un trolley. Ovviamente nessun umano ha un giunto di tipo prismatico, ma solo giunti rotanti.

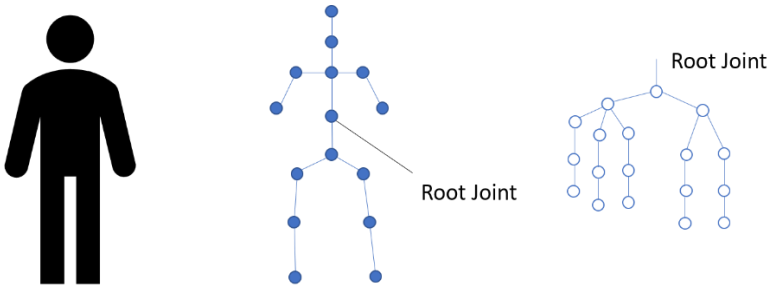


Giunto Rotante

Giunto Prismatico

2.3.2 Struttura gerarchica

Un modello gerarchico prevede che una serie di oggetti siano uniti tra loro tramite vincoli di connettività. Queste strutture sono molto utili per modellare corpi articolati, in cui la manipolazione di un giunto produce il movimento di una o più appendici. I giunti che permettono alle appendici ad esso connesse il moto in una sola direzione, si dicono giunti *a un grado di libertà* (1DOF). Giunti complessi possono avere più gradi di libertà. Un corpo articolato possiede un *root joint* (arco radice), ovvero la base della struttura e la sua posizione è nota nel sistema di coordinate globali (la posizione degli altri giunti è relativa a quella dell'arco radice). Il root joint di un umanoide è solitamente il centro dei fianchi. Il corpo articolato è dunque formato da un albero di nodi e archi che partono dal root joint.



Strutture gerarchiche: umanoide e albero

Nelle strutture gerarchiche le trasformazioni che verranno applicate all'arco radice saranno valide anche per tutti i discendenti. Dunque, per essere più specifici, la posizione di tutti i nodi è specificata in maniera relativa rispetto a quella del padre e una trasformazione applicata all'arco radice è globale e si ripercuote su tutti i nodi dell'albero.

Un arco contiene le informazioni che permettono di ruotare o traslare l'oggetto relativamente al giunto di livello superiore nella gerarchia e che permettono di stabilire la posizione relativa rispetto al giunto corrente. Un nodo contiene le informazioni relative all'oggetto e le trasformazioni applicabili all'oggetto stesso.

Ai nodi possono essere applicate diverse trasformazioni, come ad esempio traslazioni e rotazioni. L'attraversamento della gerarchia permetti di comporre tutte le varie trasformazioni associate con archi e nodi. Una struttura a stack è una soluzione semplice ed elegante per realizzare la visita dell'albero e concatenare le varie matrici di trasformazioni. Quando si visita in profondità un nuovo arco, la matrice corrispondente è inserita nello stack e pre-moltiplicata alla matrice di trasformazione corrente. Quando si attraversa l'albero dal basso verso l'alto, viene estratta la matrice dal top dello stack.

In una struttura gerarchica si possono effettuare tre tipi di visite:

- *In-Order,* prendendo il valore del nodo quando lo si visita per la seconda volta
- *Post-Order,* prendendo il valore del nodo l'ultima volta che si passa per lo stesso
- *Pre-Order,* prendendo il valore la prima volta che si passa per quel nodo

2.3.3 End Effector

Si definisce *End Effector* la posizione più esterna dell'ultimo giunto. È la terminazione della catena di alternanza di nodi e archi. L'end effector non è un articolazione ma la posizione all'estremità finale di un corpo articolato. Una figura articolata può avere diversi end effector.

2.3.4 Cinematica diretta

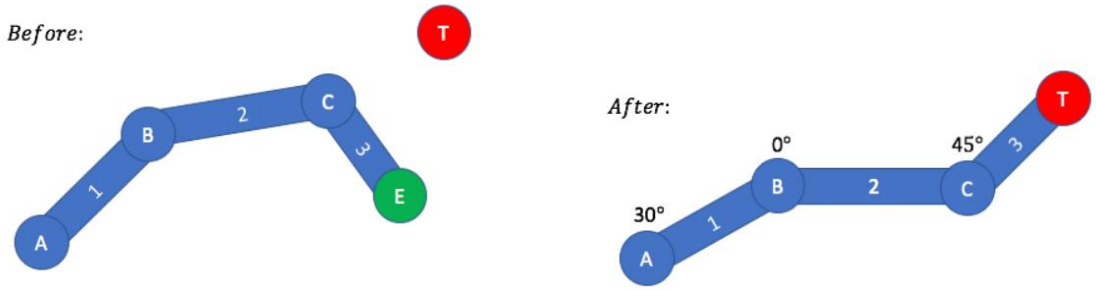
La *Cinematica Diretta* (*Forward Kinematic – FK*) può essere definita come una funzione o algoritmo che prende una posa come input, e calcola la posizione dell'end effector come output. Con la FK è necessario definire l'intera posizione di un corpo articolato in modo da fornire alla funzione l'input di posa. Dunque, è necessario specificare l'angolo che assume ogni giunto. Questo però è un processo definito *trial-and-error* in quanto bisogna provare e riprovare finché non si ottiene il risultato sperato ed è dunque una tecnica molto complessa e tediosa soprattutto quando il numero di nodi è molto elevato.

2.3.5 Cinematica inversa

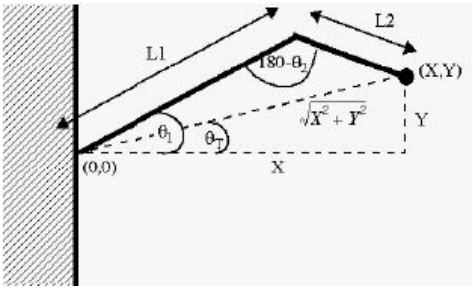
La *Cinematica Inversa* (*Invert Kinematic – IK*) è la funzione inversa della FK. Come già spiegato precedentemente, la FK prende come input una posizione e calcola la richiesta per far sì che l'end effector raggiunga quella posizione, la posizione è l'output. Per evitare questo, si specifica la posa finale dell'oggetto e gli angoli di tutti i giunti ad esso collegati vengono calcolati automaticamente. In poche parole, con l'IK l'animatore deve solo specificare la posa dell'end effector. Tra FK e IK input e output vengono invertiti. Con la cinematica inversa non si definisce la posizione dell'intera figura articolata, ma viene calcolata in automatico.

La cinematica inversa fa tutti il lavoro computazionale per calcolare la posa. Le figure seguenti lo rappresentano bene. Nella prima c'è una figura articolata con una posa nota. Viene definita una posizione target per l'end effector. Una volta applicato l'algoritmo IK, si raggiungerà la seconda

figura, che mostra che è stata calcolata una nuova posa in modo tale che l'end effector sia nella posizione target.



Potrebbe capitare a volte di avere due possibili configurazioni per la posa finale, per cui l'algoritmo non riesce a decidere. Nel caso di meccanismi molto semplici, la soluzione può essere calcolata per via analitica verificando la disuguaglianza:



$$L_1 - L_2 \leq \sqrt{x^2 + y^2} \leq L_1 + L_2$$

Per risolvere questa problematica in casi più complessi invece, è necessario introdurre dei vincoli ai giunti.

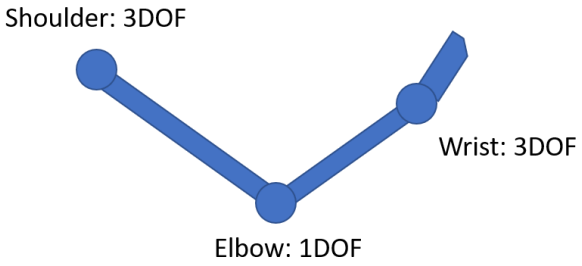
La cinematica inversa può essere usata per far raggiungere una posa target o un oggetto a un braccio di un modello umanoide, ma può anche essere usata per far fare un passo alla gamba, dicendo al piede dove posizionarsi e facendo configurare all'algoritmo IK l'articolazione della gamba. Dunque, se si sta implementando un walk cycle, ovvero un ciclo di camminata, sarà possibile posizionare alcuni dei key frame utilizzando lo strumento IK.

Un altro punto chiave su IK è che il target non è relativo alla sola posizione: l'obiettivo finale potrebbe essere definito come una rotazione. Ad esempio, se i piedi devono ruotare in base a un terreno irregolare, l'obiettivo di rotazione IK può essere definito in base alla normale del pavimento. In questo modo i piedi saranno inclinati lungo il pavimento, come quando si cammina su una pendenza. Si può usare l'IK anche per far guardare la testa (o anche gli occhi) in una certa direzione. Se si vuole che la testa del modello segua un oggetto, si può usare IK per far sì che la testa lo segui.

2.3.6 Gestione di figure articolate umanoidi

Animare le figure articolate umanoidi è un compito abbastanza complesso. Il corpo umano, oltre ad avere circa duecento ossa e seicento muscoli, ha un moto che non è ben determinabile a livello computazionale.

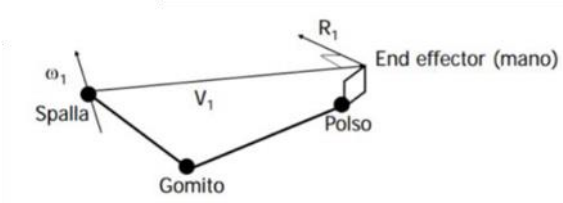
Il braccio umano può essere rappresentato, come in figura, come un manipolatore a diversi gradi di libertà. Considerando la mano come un punto, l'intero braccio consta di ben sette gradi di libertà (tre per la spalla, uno per il gomito e tre per il polso). Questo tipo di semplificazione porta a un problema riguardo il movimento dell'avambraccio in quanto la rotazione, che nella realtà è distribuita lungo l'avambraccio in quanto ulna e radio ruotano uno attorno all'altro, è qui legata al polso. I giunti del braccio, poi, devono avere dei vincoli per evitare che questo faccia movimenti innaturali.



Rappresentazione grafica del braccio

Considerando questo modello di braccio, specificando la posa dell'end effector si ha un sistema sotto vincolato, esistono soluzioni multiple e l'IK può fornire risultati non realistici. Dunque, è spesso utile fissare la posizione del polso in modo che sia subordinata alla spalla e, se tutto ciò non è sufficiente, sarà necessario specificare alcune pose intermedie.

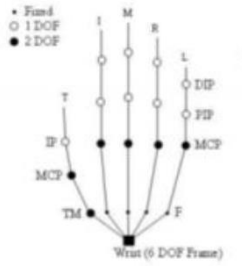
Nel calcolo della cinematica inversa, l'approccio legato allo Jacobiano inverso può essere sostituito da uno più procedurale.



La considerazione alla base del ragionamento è che gli angoli più lontani dall'end effector (ovvero dalla mano) sono quelli che influenzano la posizione in maggior misura: i giunti più vicino alla mano sono usati per attribuire "regolazioni fini". Bisogna considerare il prodotto esterno tra l'asse di rotazione del giunto e il vettore che unisce quel giunto all'end effector. Giunti lontani influenzano più di quanto facciano quelli vicini e il modulo sarà maggiore. È possibile semplificare il problema

considerando i tre giunti come se fossero su uno stesso piano. In questo modo si lavora solo su due dimensioni ma si crea un effetto ingessato, problema risolvibile permettendo la rotazione di tutti i giunti.

Il modello della mano non rientra in quello appena descritto in quanto si tratta di un sistema molto complesso e non semplice da gestire. Consta infatti di ventuno gradi di libertà. L'animazione della mano è un processo molto complicato e tedioso, ma si può semplificare tramite alcune soluzioni. Una di queste potrebbe essere l'utilizzo di un sistema di motion capture della mano, con un tracking in tempo reale che permette la gestione e la cattura di tutti i giunti e dei loro movimenti.



2.4 Rigging su Blender

Blender è un software open source e una multipiattaforma di modellazione, rigging, animazione, montaggio video, compositing e rendering di immagini 2D e 3D. Di seguito verrà illustrato come le nozioni teoriche riguardo il processo di rigging, spiegate nei paragrafi precedenti vengono applicate sul programma.

2.4.1 Introduzione

Uno scheletro o *armature* in Blender, può essere considerato simile all'armatura di un vero scheletro, e proprio come un vero scheletro, può essere costituito da molteplici ossa. Queste ossa possono essere spostate e tutto ciò a cui sono attaccate o associate si sposterà e deformerà in modo simile.

Un' armature è una tipologia d'oggetto utilizzata per il processo di rigging. Un rig consiste in un insieme di controlli che muovono un modello (puppet). L'oggetto armature prende in prestito molte idee dagli scheletri del mondo reale.

Poiché le armature sono create per essere posizionate in un certo modo, sia per una scena animata che per una scena statica, esse hanno un determinato stato, chiamato *Rest Position,* che non è altro che la forma di default dello scheletro, la posizione di default di posizione/rotazione/scala delle proprie ossa, come sono state create in Edit Mode.

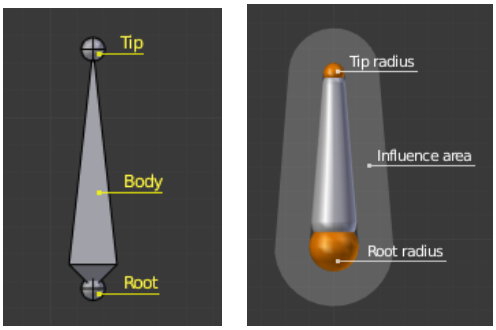
2.4.2 Bones

Gli oggetti *bones,* cioè ossa, sono gli elementi base delle armature. Possono essere definiti in due modi: ossa deformanti e ossa di controllo. Le prime sono ossa che, quando trasformate, quindi mosse, faranno in modo che i vertici a esse associate si trasformino in un modo simile. Le ossa di controllo sono invece ossa che agiscono similmente a degli interruttori, in quanto controllano il modo in cui altre ossa o oggetti reagiscono quando vengono trasformate. Un osso di controllo potrebbe ad esempio fare da switch e, quando si trova in una posizione a sinistra, indicare ad altre ossa di reagire in un certo modo mentre, quando è posizionato a destra, far sì che le stesse ossa facciano qualcosa di completamente diverso. Le ossa di controllo non vengono utilizzate direttamente per modificare le posizioni dei vertici; infatti, le ossa di controllo spesso non hanno vertici direttamente associati a se stessi.

Un osso è formato da tre parti: una base chiamata *head,* il corpo principale e una parte finale chiamata *tail.* Ogni osso ha un proprio orientamento assiale, e l'asse y, che è sempre allineato all'osso, orientato

dalla testa alla coda, viene chiamato “asse di *roll”*. Il roll dunque è l’orientamento dell’osso lungo gli assi locali.

Un osso controlla una geometria quando certi vertici “lo seguono”, proprio come i muscoli e la pelle seguono l’osso di un dito quando questo di muove. Per far questo, bisogna definire l’intensità dell’influenza che un osso ha su un certo vertice. Il modo più semplice è far sì che ciascun osso colpisca quelle parti della geometria che si trovano all'interno di un determinato intervallo. Questa è chiamata tecnica dell'*envelope*, poiché ogni osso può controllare solo la geometria all’interno della propria area di influenza.



Osso ottaedrale *Osso envelope*

Le ossa possono avere due tipi di relazioni: possono essere imparentate e in aggiunta anche connesse. Le ossa imparentate si comportano in Edit Mode come se non avessero relazioni. Possono essere spostate, ruotate, scalate senza influire sui discendenti. invece, le ossa connesse devono avere sempre le punte delle ossa madri collegati alle radici delle ossa figlie, quindi trasformando un osso verranno influenzate anche le ossa connesse.

Quando un osso viene selezionato si possono vedere le sue proprietà nella sezione *Bone* del menù laterale di Blender. In seguito, sono elencate le proprietà delle ossa:

- *Transform:* in Edit Mode si può usare questo pannello per controllare la posizione e l’orientamento di ogni osso, mentre in Pose Mode si può selezionare la posizione, rotazione e scala dell’osso principale.
- *Inverse Kinematic:* permette di controllare il modo in cui un osso si comporta all’interno di una catena di cinematica inversa.
- *Custom Properties:* le proprietà personalizzate sono un modo per archiviare i dati dell’utente nei data-block di Blender.

Le ossa di tipo Bendy (*B-Bones*) sono un modo semplice di rimpiazzare una catena di piccole ossa rigide. Un caso di uso comune per le ossa curve è quello di modellare colonne verticali o ossa facciali.

Si possono vedere i segmenti delle bendy bones solo quando viene attivata la visualizzazione B-Bone. Quando questa non è attiva, le ossa sono mostrate come stick rigidi, anche se i segmenti ossei sono presenti ed efficaci. Ciò significa che anche nella visualizzazione ottaedrica, se alcune ossa hanno diversi segmenti, deformeranno senza problemi la loro geometria.

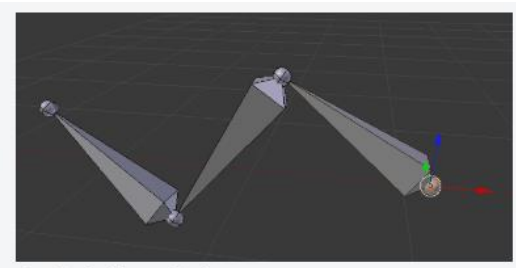
È possibile impostare il numero di segmenti in cui l'osso selezionato è diviso. I segments sono piccole ossa collegate che si interpolano tra il root e l'head. Maggiore è il numero, più "fluido" sarà l'osso ma sarà anche più pesante il calcolo della posa.

I campi *Ease In/Out* cambiano la lunghezza della maniglia della curva Bezier per controllare la maniglia della radice e la maniglia della testa dell'osso. Questi valori sono proporzionali alla lunghezza di default che varia automaticamente in base alla lunghezza delle ossa, all'angolo con la maniglia di riferimento e così via.

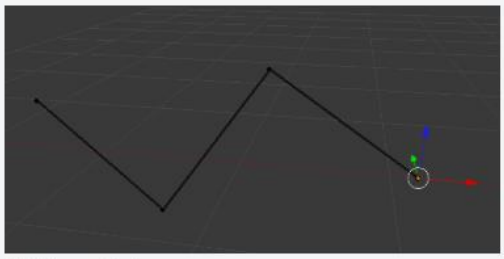
2.4.3 Proprietà

Ogni scheletro ha 32 livelli che permettono di organizzare l'armature raggruppando set di ossa in livelli. In questo modo si può scegliere di traslare un osso da un livello a un altro, nascondere o mostrare certi livelli ecc.

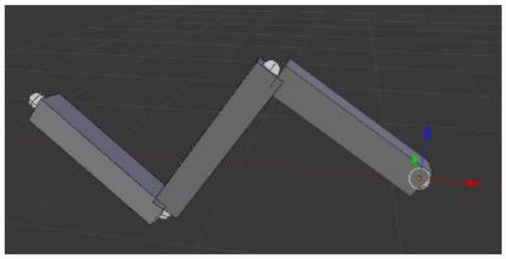
Lo scheletro consta di diverti metodi di visualizzazione: *Octahedral display, Stick Bone display, B-Bone Display, Envelope bone display.*



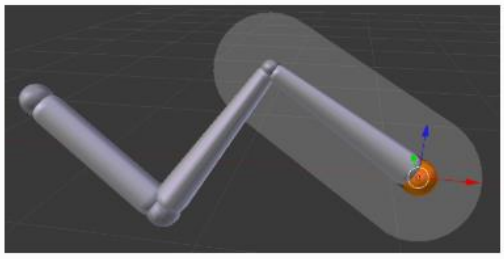
Octahedral bone display.



Stick bone display.



B-Bone bone display.



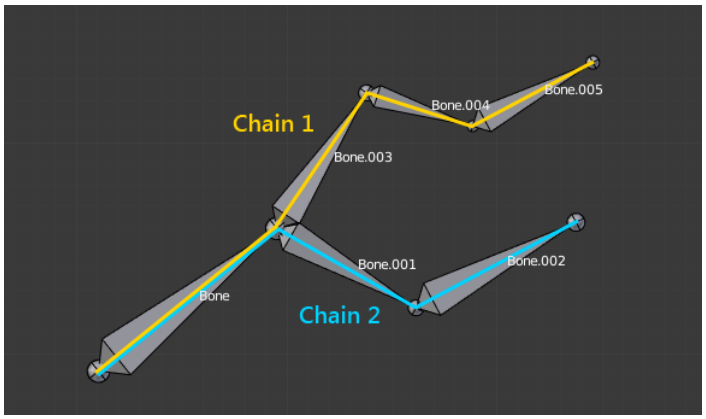
Envelope bone display.

È possibile raggruppare le ossa in gruppi (*Bone Groups*). Sono utili in quanto possono essere usati per la selezione o assegnati a un colore specifico per gruppo.

2.4.4 Struttura

L’oggetto armature imita gli scheletri reali. È fatto di ossa che, di default, sono elementi rigidi. Ma si hanno delle possibilità in più rispetto ai veri scheletri: oltre alla rotazione “naturale” delle ossa, si possono anche spostare e ridimensionare. E non devono essere collegate tra loro; possono anche essere completamente libere. Tuttavia, le configurazioni più naturali e utili implicano che le ossa siano correlate ad altre, formando le cosiddette “catene di ossa”, che creano una sorta di arto.

Le ossa all'interno di un'armatura possono essere completamente indipendenti l'una dall'altra (ovvero la modifica di un osso non influisce sull'altro). Ma questo non è quasi mai un allestimento utile: per creare una gamba, tutte le ossa “dopo” l'osso della coscia dovrebbero muoversi “con” esso in modo ben coordinato. Questo è esattamente ciò che accade nelle armature imparentando un osso a quello successivo dell'arto, creando una "catena di ossa". Queste catene possono essere ramificate. Ad esempio, cinque dita attaccate a un singolo osso "a mano".



Esempio di catena ramificata di ossa

Le ossa sono incatenate collegando la punta dell’osso madre alla radice dell’osso figlio. Radice e testa possono essere collegate, ovvero sono sempre esattamente nello stesso punto; oppure possono essere libere, come in una relazione oggetto genitore-figlio standard.

Un dato osso può essere il genitore di diversi figli e quindi far parte di più catene contemporaneamente.

L'osso all'inizio di una catena si chiama *root bone*, e l'ultimo osso di una catena è detto *tip bone*.

Le catene di ossa sono un argomento particolarmente importante nel processo di *posing* (in particolare con la cinematica diretta al contrario delle tecniche di posa della cinematica inversa "automatica").

2.4.5 Skinning

Il processo di collegamento di un'armature a un oggetto che deve essere deformato da essa, è chiamato *Skinning*.

In Blender ci sono due tipologie principali di Skinning:

1. Si possono imparentare/vincolare oggetti alle ossa, dunque quando vengono trasformate le ossa in Pose Mode anche i loro oggetti "figli" verranno trasformati esattamente come una relazione genitore/figlio standard (i figli non vengono mai deformati con questo metodo).
2. È possibile utilizzare il modificatore Armature sull'intera mesh e quindi alcune parti di questo oggetto su alcune ossa all'interno dell'armature. Questo è il metodo più complesso e potente e l'unico modo per deformare realmente la geometria dell'aggetto, ovvero per modificare le posizioni relative dei suoi vertici o punti di controllo.

Tramite il modificatore *Armature,* lo scheletro diverrà il padre degli oggetti figli a cui sarà imparentato e ogni oggetto avrà un modificatore Armature con il rig associato nella sezione *Object.*

Ci sono diverse metodologie di "imparentamento" oggetto-scheletro:

- *Con Vertex Group:* verranno creati dei vertex group vuoti su tutti gli oggetti figli (se non esistono già) per ogni ossa deformante del rig e chiamati come le rispettive ossa. Nessun peso sarà associato a questi gruppi.
- *Con Automatic Weights:* il parenting per pesi automatici funziona in modo simile a quello dei vertex group ma non lascerà i gruppi vuoti. Questo metodo calcola quanta influenza un particolare osso avrebbe sui vertici in base alla distanza da quei vertici dello stesso. Questa influenza verrà assegnata come pesi nei vertex group. Questo metodo è certamente più facile da configurare, ma spesso può portare ad armature che non deformano gli oggetti figlio nel modo desiderato. Possono verificarsi sovrapposizioni quando si tratta di determinare quali ossa dovrebbero influenzare determinati vertici quando si calcolano le influenze per armature più complesse e oggetti figlio. I segni di questa confusione si vedono quando si trasforma lo scheletro in Pose Mode, quando parti degli oggetti figlio non si deformano come dovrebbero; Se Blender non dà i risultati richiesti, si dovranno modificare manualmente i pesi dei vertici in relazione ai gruppi di vertici a cui appartengono e nei quali hanno influenza.
- *Con Envelope Weights:* funziona similmente all'Automatic Weights ma la differenza è che l'influenza è calcolata in base alle impostazioni degli envelope delle ossa. Il metodo assegnerà un peso a ogni vertex group che si trova all'interno del volume di influenza dell'osso, a seconda della loro distanza da esso.

2.4.6 Posing

Una volta eseguito lo skinning, è necessario un modo per configurare il rig in posizioni note come *pose*. Fondamentalmente, trasformando le ossa, vengono trasformati o deformati gli oggetti a esse associati. Tuttavia, non è possibile farlo in Edit Mode in quanto questa viene utilizzata per modificare la posizione predefinita, base o "di *rest*" di un'armature. Non è possibile utilizzare nemmeno la Object Mode, poiché qui è possibile trasformare solo interi oggetti.

Quindi, le armature hanno una terza modalità dedicata al processo di posa noto come *Pose Mode*. In posizione di riposo, ogni osso ha la propria posizione / rotazione / scala su valori neutri (ovvero 0,0 per posizione e rotazione e 1,0 per scala). Quindi, quando si modifica un osso in modalità Posa, si crea un offset nelle proprietà di trasformazione, dalla sua posizione di riposo.

Anche se potrebbe essere utilizzato per scopi completamente statici, la posa è fortemente connessa con le caratteristiche e le tecniche di animazione.

Il colore delle ossa si basa sul loro stato. Esistono sei diversi codici di colore, ordinati qui per precedenza:

- *Grigio*: colore di default.
- *Wireframe blu*: in Pose Mode.
- *Verde*: con Constraint.
- *Giallo*: con constraint del tipo *IK Solver*.
- *Arancione*: con constraint del tipo *Targetless Solver*.

In Pose -mode, le ossa si comportano come oggetti. Quindi le azioni di trasformazione (spostamento / rotazione / scalamento, ecc.) sono molto simili a quelle della Object. Tuttavia, ci sono alcune importanti specificità:

- Le relazioni delle ossa sono cruciali.
- Il centro di trasformazione di un dato osso (ovvero il suo punto di articolazione predefinito, quando è l'unico selezionato) è la sua radice.

Come notato in precedenza, le trasformazioni delle ossa vengono eseguite in base alla rest pose dell'armatura, che è il suo stato definito nella Edit Mode. Ciò significa che in rest pose, in Pose Mode, ogni osso ha una scala di 1,0 e rotazione e posizione nulle.

Inoltre, lo spazio locale per queste azioni è quello dell'osso (visibile quando si abilita l'opzione Assi del pannello Armature). Ciò è particolarmente importante quando si utilizza il bloccaggio dell'asse,

ad esempio, non esiste uno strumento specifico per il roll delle ossa in Pose mode, poiché è possibile ruotare attorno all'asse principale dell'osso semplicemente bloccando l'asse Y locale.

Quando si posiziona un rig, si dovrebbero avere uno o più oggetti associati a esso. E ovviamente, quando si trasforma un osso in Pose Mode, i suoi oggetti correlati o la forma dell'oggetto vengono spostati / deformati di conseguenza, in tempo reale. Sfortunatamente, se si dispone di una configurazione di rig complessa e / o di un oggetto pesante a livello di skinning, ciò potrebbe causare ritardi e rendere il montaggio interattivo molto doloroso.

2.4.7 Lattice

Un *lattice,* comunemente chiamata gabbia di deformazione, è un reticolo costituito da una griglia tridimensionale non renderizzabile di vertici. Il suo uso principale è quello di applicare una deformazione all'oggetto che controlla con un modificatore *Lattice*. Se l'oggetto è associato a Deform Lattice, viene automaticamente applicato un modificatore *Lattice*.

2.4.8 Constraints

I constraint sono un metodo per controllare le proprietà di un oggetto (ad esempio posizione, rotazione, scala), usando valori statici semplici (come quelli "limite") o un altro oggetto, chiamato "target".

Anche se i constraint sono utili nei progetti statici, il loro utilizzo principale è ovviamente nell'animazione:

- Si può controllare l'animazione di un oggetto attraverso i target utilizzati dai suoi constraint (questa è una forma di animazione indiretta). In effetti, questi target possono controllare le proprietà del proprietario del vincolo e, quindi, l'animazione dei target animerà indirettamente il proprietario.
- Si possono animare impostazioni dei constraint come l'influenza o, quando si utilizza un osso come target, animare dove lungo questo osso (tra radice e punta) si trova il vero punto target.

I constraint possono fare in modo che gli occhi di un tennista seguano una palla da tennis che rimbalza sul campo, consentono alle ruote di un autobus di ruotare tutte insieme, aiutare le gambe di un dinosauro a piegarsi automaticamente sul ginocchio e rendere facile per una mano afferrare l'elsa di una spada e la spada per oscillare con la mano.

2.4.9 Actions

Quando si animano gli oggetti e le proprietà in Blender, le Actions registrano e contengono tutti i dati. Le azioni sono blocchi di dati. Pertanto, quando si anima un oggetto modificandone la posizione con i fotogrammi chiave, l'animazione viene salvata in *Action.*

2.4.10 Driver

I driver servono per controllare i valori delle proprietà mediante una funzione o un'espressione matematica.

In sostanza, i driver consistono in:

- Una *driver configuration* che specifica zero, uno o più valori di input utilizzando altre proprietà o canali di trasformazione degli oggetti e li combina utilizzando una funzione matematica predefinita o un'espressione in Python personalizzata.
- Un' *animazione* di tipo F-Curve che mappa l'output della configurazione del driver sul valore finale da applicare alla proprietà pilotata.

Ad esempio, la rotazione di *oggetto1* può essere controllata dalla scala di *oggetto2*. Si dice quindi che la scala dell'oggetto2 guida la rotazione dell'oggetto1.

I driver non solo possono impostare direttamente il valore di una proprietà sul valore di un'altra, ma possono anche combinare più valori usando una funzione fissa o un'espressione in Python e modulare ulteriormente con una curva definita manualmente e / o uno stack modificatore.

I driver sono uno strumento estremamente potente per la di rig e sono generalmente utilizzati per guidare trasformazioni ossee e l'influenza di shape keys, constraint di azioni e modificatori, spesso utilizzando proprietà personalizzate come input.

2.4.11 Shape Keys

Le shape keys vengono utilizzate in animazione per deformare gli oggetti in nuove forme. Le shape keys, dette anche *morph target* o *blend shapes,* sono comunemente usate nell'animazione facciale del personaggio e nella modifica e perfezionamento di un rig. Sono particolarmente utili per modellare parti molli e muscoli organici in cui è necessario un maggiore controllo sulla forma risultante rispetto a quanto si può ottenere con la combinazione di rotazione e scala. Possono essere applicate su tipi di oggetti con vertici come mesh, curve, superfici e lattice.

Quando il viso viene animato, l'animatore può miscelare la forma base con le atre forme target. Tipiche forme target usate in animazione sono la bocca sorridente, un occhio chiuso, un sopracciglio alzato.

Non tutte le shape keys devono essere create muovendo la posizione dei vertici. È anche possibile spostare le ossa dello scheletro e salvare le posizioni come forme target.

Ci sono dei vantaggi nell'usare questo tipo di tecnica. L'animatore ha più controllo sui movimenti perché può determinare le specifiche posizioni dei vertici all'interno di un keyframe piuttosto che essere vincolato da uno scheletro. Questo può essere utile per animare tessuti, pelle e animazioni facciali in quanto può risultare complicato associare questi oggetti alle ossa.

Tuttavia, ci sono anche degli svantaggi. La vertex animation è di solito più laboriosa rispetto alla skeletal animation perché la posizione di ogni vertice deve essere spostata manualmente. Per questo motivo il numero di shape keys di solito è limitato. Inoltre, si potrebbero creare delle distorsioni in render che non succederebbero con la skeletal animation. La vertex animation richiede una memoria significativa, in quanto la posizione di ogni vertice deve essere salvata per ogni frame.

3. Motion Capture

3.1 Introduzione

Il Motion Capture (mocap) è una tecnica di registrazione del movimento del corpo umano (o di un altro movimento) per l'analisi e la riproduzione in tempo reale o in un momento successivo. Le informazioni acquisite possono essere tanto generali quanto la semplice posizione del corpo nello spazio o complesse come le deformazioni del viso e delle masse muscolari. L'acquisizione del movimento per l'animazione del di un personaggio 3D comporta la mappatura del movimento umano sul movimento di un personaggio del computer. La mappatura può essere diretta, come il movimento del braccio umano che controlla il movimento del braccio di un personaggio, o indiretto, come i modelli di mani e dita umani che controllano il colore della pelle o lo stato emotivo di un personaggio.

L'idea di copiare il movimento umano per personaggi animati non è, ovviamente, nuova. Per ottenere un movimento convincente per i personaggi umani in Biancaneve, gli studi Disney hanno tracciato l'animazione sulle riprese di attori in live action. Sin da allora questo metodo, chiamato *rotoscoping*, è stato usato sempre con successo per i personaggi umani. Alla fine degli anni '70, quando nacque la possibilità di animare i personaggi al computer, gli animatori riadattarono le tecniche tradizionali, incluso il rotoscoping. Ma la nascita di veri e propri sistemi di motion capture accade dall'inizio degli anni Ottanta, in ambito universitario e di ricerca. Ad esempio, nel 1982 nasce la *Graphical Marionette,* ovvero un sistema ottico che faceva uso di led posizionati al di sopra di una tuta in corrispondenza dei giunti e di una serie di camere che catturavano il movimento rendendo possibile la visualizzazione in real-time del movimento su una "marionetta" digitale; successivamente nasce *Waldo,* un sistema che gestiva in real-time i movimenti della bocca.

La tecnologia più affidabile fino ad oggi è quella che fa uso di marker, ovvero oggetti posti sulle articolazioni di una tuta che possono emettere o riflettere la luce per acquisire il movimento.

3.2 Sistemi di motion capture

I sistemi di mocap disponibili oggigiorno possono essere classificati in tre gruppi principali: sistemi ottici, sistemi magnetici e sistemi meccanici. Ogni tipologia ha i suoi punti di forza e di debolezza. Un'altra categoria è quella dei sistemi inerziali, utilizzata ampiamente nelle produzioni indipendenti e anche per l'animazione dei personaggi di Reverie Dawnfall.

Verranno di seguito analizzati brevemente i tre gruppi, con particolare attenzione per la tecnologia utilizzata per lo scopo di questa tesi.

- *Sistema ottico:* consiste di un set dalle quattro alle trentadue camere controllate da un computer. Il soggetto da catturare indossa dei marker che possono essere passivi (riflettenti – le camere in questo caso hanno un sistema di emissione di luce) o attivi (emittenti). Questo sistema offre diversi vantaggi: i dati sono molto accurati e il rate di registrazione è molto elevato (30-2000 samples/s), si possono catturare più soggetti contemporaneamente, fornisce una elevata libertà di movimento. Gli svantaggi si possono trovare nei dati rotazionali, limitati a 3 DOF, nella probabile occlusione dei marker e nei costi elevati rispetto ad altri sistemi.
- *Sistema meccanico:* si presenta come un esoscheletro con una struttura metallica o plastica e potenziometri posti sulle articolazioni del corpo. È un sistema markerless, con occlusioni minime dovute all'assenza di telecamere, che lavora in real-time ed economico ma soffre di una certa libertà nell'impostazione dei parametri e dell'impossibilità, a causa dei potenziometri, di poter definire la posizione globale dell'attore impedendo dunque la registrazioni dei salti (risolvibile con l'uso di sensori meccanici), c'è poca libertà di movimento per l'attore e ha un basso rate di registrazione e configurazione dei sensori.
- *Sistema elettromagnetico:* una serie di sensori posizionati sull'attore misurano la relazione spaziale rispetto a un emettitore di campo magnetico. È un sistema molto buono in quanto posizione e rotazione sono generate automaticamente, sono molto economici rispetto ai sistemi ottici, sono adatti per le applicazioni in tempo reale anche per più soggetti ed è privo di occlusioni. Ma c'è il rischio di subire interferenze magnetiche, è vincolato dalla durata delle batterie, il sampling rate è basso e i dati sono rumorosi.

3.2.1 Sistema inerziale e Rokoko

I sistemi inerziali nascono dalla fusione di tre diversi tipi di sensori. Sono sempre presenti un giroscopio per la misura delle rotazioni, un accelerometro che calcola la posizione e l'inclinazione e un magnetometro, che rileva l'intensità del campo magnetico circostante. Questi elementi permettono di avere 9 DOF.

Nei sistemi più diffusi, i sensori sono posizionati lungo i giunti di una tuta in tessuto (Rokoko) o in velcro, sono legati tra loro tramite cavi e inviano via wireless le informazioni calcolate visualizzabili in real-time.

Un limite di questo sistema è quello, come nei sistemi meccanici, della mancanza di una misurazione globale per la componente traslazionale che porta a errori di misurazione del movimento sull'asse dell'altezza (ad esempio nel caso di un salto). Quello che però attrae gli studi indipendenti è sicuramente il tipo di vantaggi che questo sistema offre. Non ci sono infatti occlusioni e lo spazio di cattura è potenzialmente illimitato, è possibile visualizzare in real-time senza post-processing, si possono catturare più soggetti ed è un sistema molto più economico rispetto agli altri elencati sopra. Ulteriori svantaggi si possono sicuramente trovare nell'impossibilità di calcolare la posizione globale, nel drifting dei sensori e nella limitazione dello spazio di cattura da parte del range della connessione wireless.

Per questo progetto di tesi è stata utilizzata la tuta di motion capture inerziale Smartsuit Pro di Rokoko, il cui funzionamento e utilizzo sono stati approfonditi nella tesi di laure svolta da Melissa Coarezza.

3.3 Mocap Facciale

Nella maggior parte dei casi, il motion capture facciale è realizzato con sistemi ottici, tramite marker posizionati sul viso, e separatamente dalla cattura del movimento del corpo. I dati facciali sono stabilizzati rimuovendo il movimento della testa in modo da isolare lo spostamento locale della pelle del viso causato dai muscoli facciali sotto la pelle. I dati stabilizzati vengono applicati a un rig facciale.

Tuttavia, questo capitolo non si dilungherà sulla teoria del mocap facciale bensì nel prossimo paragrafo verrà esposta la tecnica utilizzata per questo progetto di tesi, ovvero lo strumento per il mocap facciale offerto da Rokoko realizzabile semplicemente tramite uno smartphone.

3.3.1 Face Performance Capture di Rokoko

A luglio 2019 è stato rilasciato Rokoko Studio 1.13.0 che include un add-on per il motion capture facciale per iPhone X, con l'uscita nei prossimi mesi anche della versione Android e desktop. Questo prodotto è basato su ARKit, ovvero il software di Apple per la realtà aumentata introdotta con i dispositivi iPhone X. Supporta cinquantadue shape keys in real-time (60 fps).

L'acquisizione dei movimenti del viso funziona analizzando il volto dell'utente, facendo il tracking dei suoi movimenti e attivando un set di cinquantadue shape keys corrispondenti al movimento. Bisogna pensare a queste shape keys come ai diversi movimenti del viso presentati nella forma di cinquantadue differenti shape facciali. Queste sono poi sono attivate in combinazione e miscelate insieme per produrre un'espressione facciale. La rotazione e il movimento generale della testa sono gestiti separatamente, mentre le espressioni facciali sono gestite con blend shapes.

4.Applicazione del Rig

4.1 BlenRig overview

BlenRig è un sistema di auto-rigging e skinning the fornisce all'utente un rig di qualità cinematografica (incluso un sistema facciale avanzato).

La creazione del rig include:

- Uno scheletro
- Una Mesh Deform Cage e oggetti Lattices
- Una mesh base del corpo per creare una versione proxy a bassa risoluzione dei personaggi

Blenrig nasce come tentativo di creare un rig standard e riutilizzabile per Blender. La deformazione principale di BlenRig è ottenuta tramite un modificatore *Mesh Deform* che permette di ottenere una deformazione regolare e uniforme in modo veloce. La topologia della *mesh deform cage* è stata creata strategicamente per produrre un effetto pull-up sulla mesh. Questo effetto pull-up unito all' algoritmo *dual quaternion* è ciò che permette a BlenRig di avere una migliore conservazione del volume.

In generale, BlenRig non fa uso di shape keys. Tutte le deformazioni sono raggiunte meccanicamente e ciò permette al rig di essere trasferito facilmente ad altri modelli.

4.2 Rigging di Nadya con BlenRig

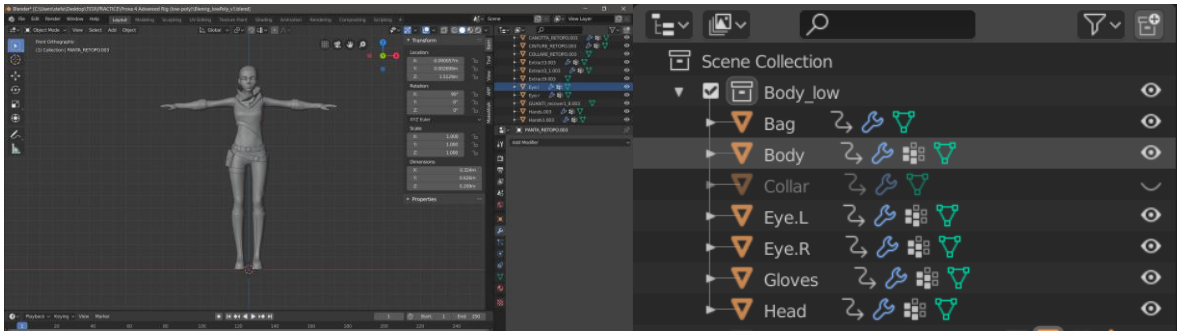
Il rig del personaggio di Nadya è stato realizzato con BlenRig, che è un rig semi customizzato, in quanto fornisce una struttura ossea già preimpostata e munita di constraint e relazioni tra ossa, e una mesh deform cage la cui struttura è stata creata appositamente per ottenere una deformazione fluida. Tuttavia, nonostante il weight paint della mesh deform avvenga in modo automatico, ha comunque la necessità di essere ritoccato, e il weight paint di mani e viso deve essere fatto manualmente. È un procedimento che richiede molto tempo, anche in base all'esperienza in campo di rigging, ma che offre un risultato di alto livello. I prossimi paragrafi illustreranno i vari step del processo di rig, dal

re-target del personaggio con il rig importato dall'add-on, al weight paint facciale e la conseguente creazione di azioni facciali.

4.2.1 Re-Target del corpo

Per prima cosa è necessario scaricare l'add-On Blenrig5 come un file zip e installarlo poi all'interno di Blender nelle *User Preferences*.

Prima di procedere con l'inserimento dello scheletro, ho aperto il file con il modello di Nadya e ho organizzato bene le mesh della scena organizzandole, tramite una serie di comandi *Join*, nel seguente modo:



In questo modo ho potuto nascondere gli oggetti che non ho incluso nel rig come il *collar* che verrà animato in seguito con una simulazione di Clothe, e inoltre separare i guanti dal corpo mi avrebbe evitato problemi futuri con il weight paint.

Per procedere con il rig, dunque, la prima cosa da fare è aggiungere Blenrig alla scena con il comando *Add>Armature>Blenrig Biped Rig*.

All'interno della collezione *BlenRig_Master_Collection* vi sono diverse collezioni. In *Blenrig Biped* appare lo scheletro (figura *1a*), nella collezione *Mesh_Deform_Cage* appare la *Mesh Deform* Cage, ovvero una mesh che verrà usata per deformare il personaggio (figura *1b*), nella collezione *Lattices* sono presenti degli oggetti *Lattices* e nella collezione *Proxy_Model* è presente un modello proxy con un piccolo numero di facce tramite cui è avere un'approssimazione a bassa risoluzione del personaggio (figura *1c*). Tutti questi oggetti sono già connessi allo scheletro di BlenRig (figura *1d*).

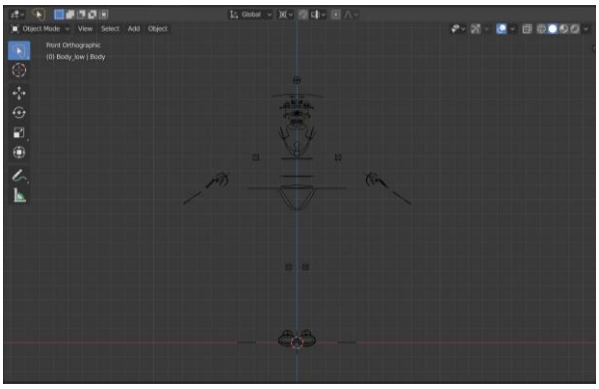


Figura 1a

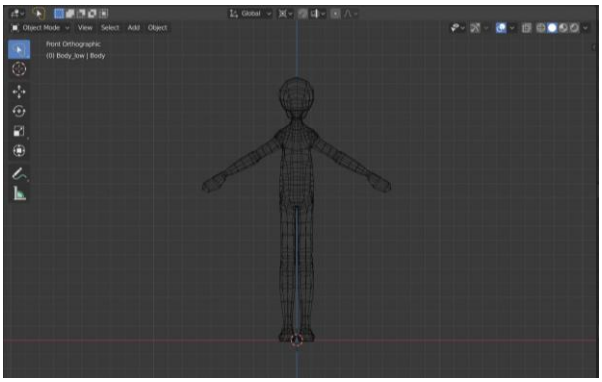


Figura 1b

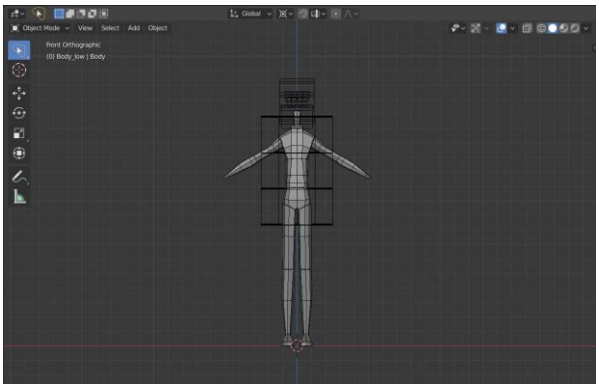


Figura 1c

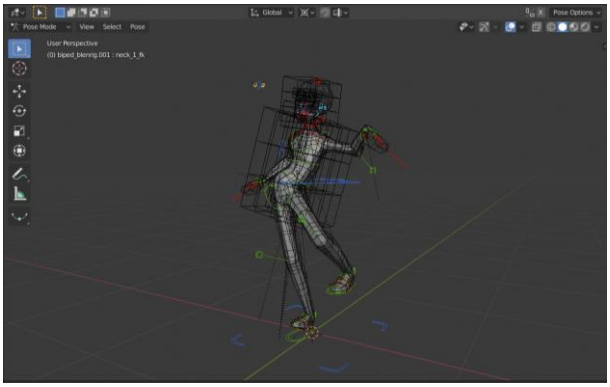


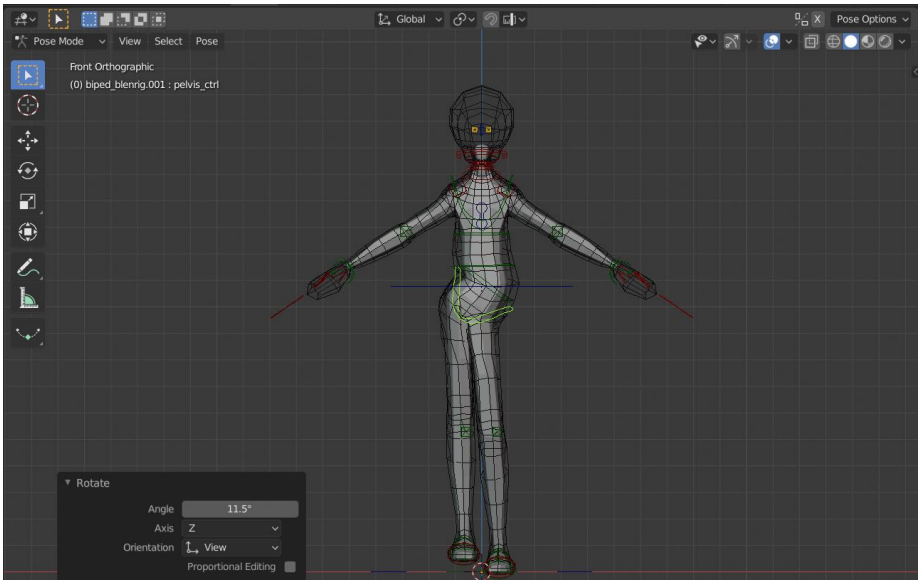
Figura 1b

Nel pannello *Tools* (premendo il tasto N) è possibile trovare un selettore di ossa per i controlli del corpo, un menù di livelli, un selettore per i controlli facciali, un menù per le proprietà extra per diverse parti del corpo, nella sezione *Custom Properties* è possibile vedere le proprietà delle ossa e infine c'è una sezione *Muscle System* che è possibile utilizzare quando si crea un sistema di muscoli o per gestire diverse proprietà.

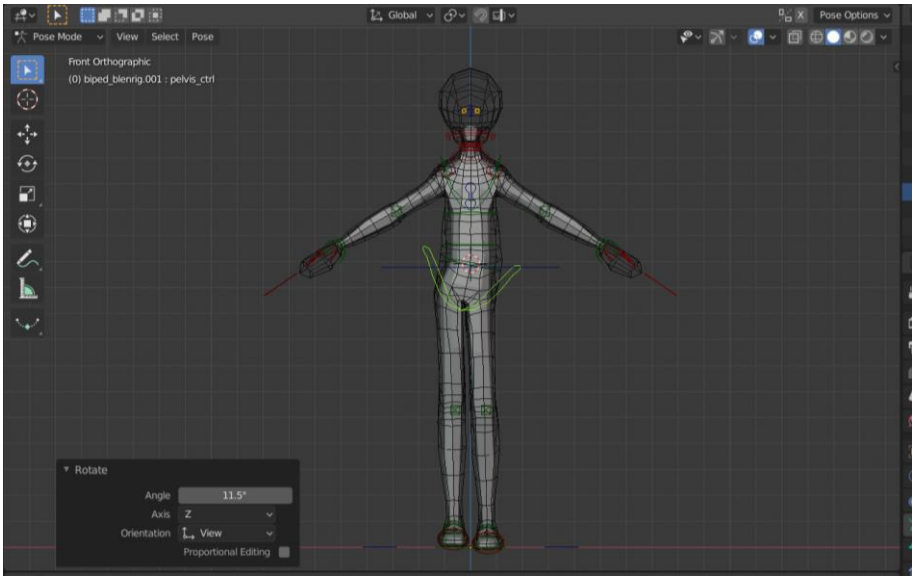
Nella sezione *Data* dello scheletro è presente un pannello Blenrig, in cui vi sono delle proprietà per il rig del corpo (*BODY SETTINGS*), per la faccia (*FACE SETTINGS*), per i livelli (*LAYERS SETTINGS*), una sezione *DYNAMIC SHAPING* che permette di modificare il personaggio in un modo simile al Flex Rig, una sezione *OPTIMIZATION* e infine la sezione *RIGGING & BAKING*. L'utilizzo di queste sezioni verrà approfondito di seguito, all'interno del capitolo.

Impostando il numero di livelli a 32, passando in modalità *Reproportion*, e isolando il livello *Reproportion*, i controllers di questo livello permetteranno di eseguire il re-target dell'intero scheletro. Si può iniziare dallo scalare il controller master, adattando l'altezza del rig a quella del modello, di seguito si posiziona il master del torso, posizionandolo alla giusta altezza, si prosegue con l'aggiustare i controller della spina dorsale. All'interno del torso sono presenti tre controller a forma di croce (*torso_ctrl_inv_str, torso_ctrl_str, pelvis_ctrl_str*) che non sono altro che punti di

pivot all'interno del torso. In base alla loro posizione cambierà il modo in cui si curverà il torso del personaggio una volta riggato. Ci sono due modi in cui impostare questi pivot. Ad esempio, se *pelvis_ctrl_str* viene posto al di sotto del controller del giro vita, si avrà un movimento automatico di questo tipo:



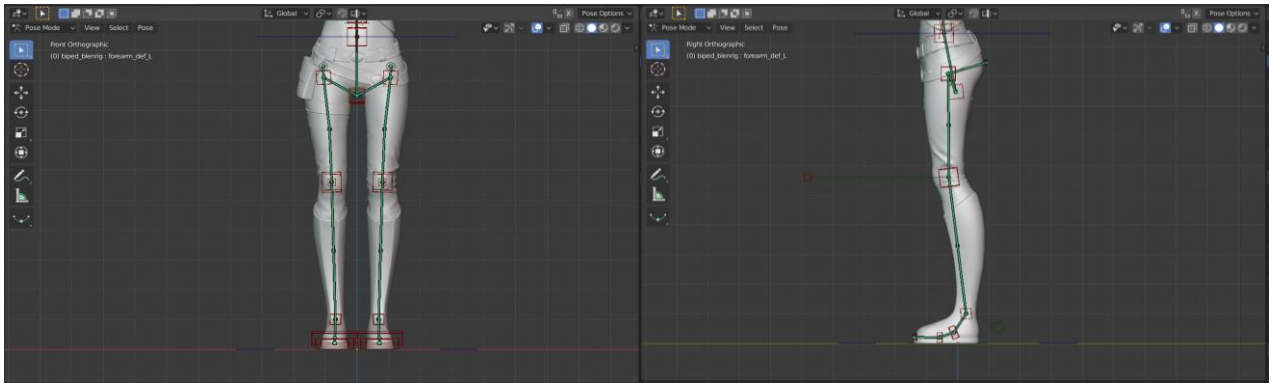
Al contrario, se lo posiziono nello stesso punto del giunto, non si avrà nessun movimento automatico in FK:



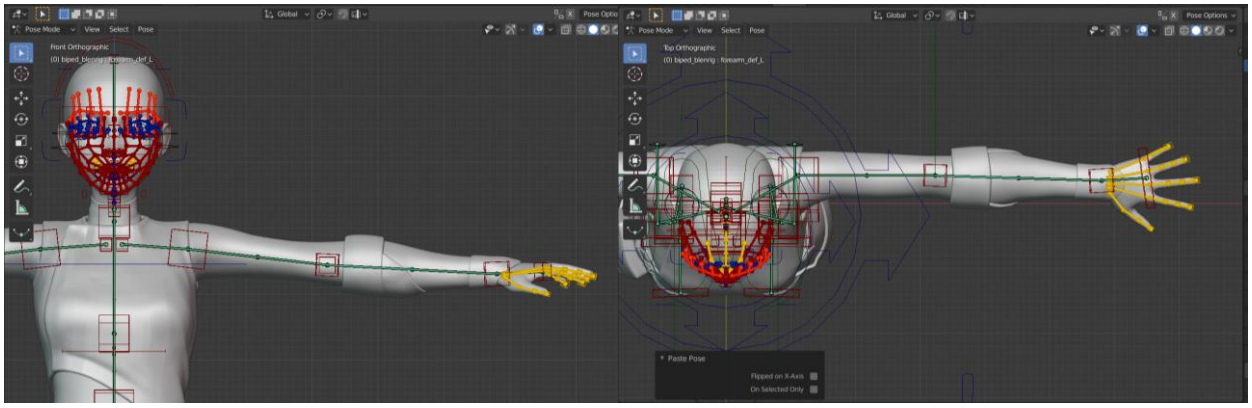
Allo stesso modo, la seconda croce *torso_ctrl_str* è il pivot per il torso ma in modalità IK. Di solito si posiziona al centro della pancia, in modo da avere una deformazione organica del modello in IK. Infine, *spine_ctrl_inv_str* è il pivot del torso in modalità *Invert Torso Control.* Di solito si usa questa

modalità quando il personaggio è appeso a qualcosa o si trova a testa in giù e, allo stesso modo del *pelvis_ctrl_str*, se questo controller viene posto al centro del giunto *spine_ctrl_3_str*, non si avrà nessun movimento automatico.

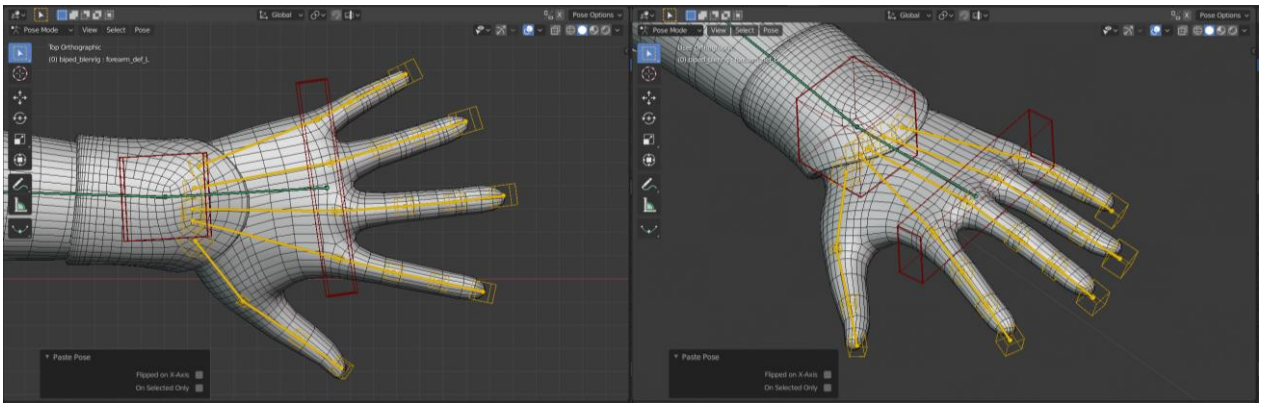
Una volta gestito il torso, è il momento delle gambe. È importante posizionare i piedi in modo appropriato, specialmente nel tacco, in modo da ottenere un buon movimento automatico per il walk cycle:



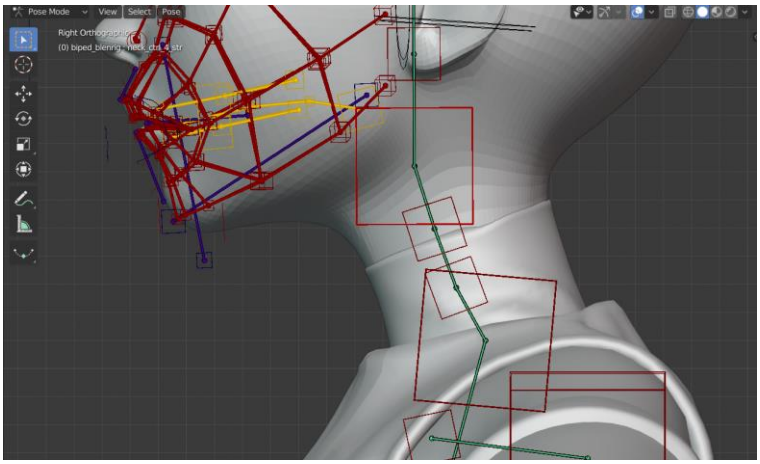
In seguito, bisogna aggiustare le braccia posizionando le ossa del collo, delle spalle, del gomito e del polso in modo adeguato. A volte è importante abilitare la visuale *wireframe* così da poter far combaciare i giunti dello scheletro ai loop effettivi del modello.



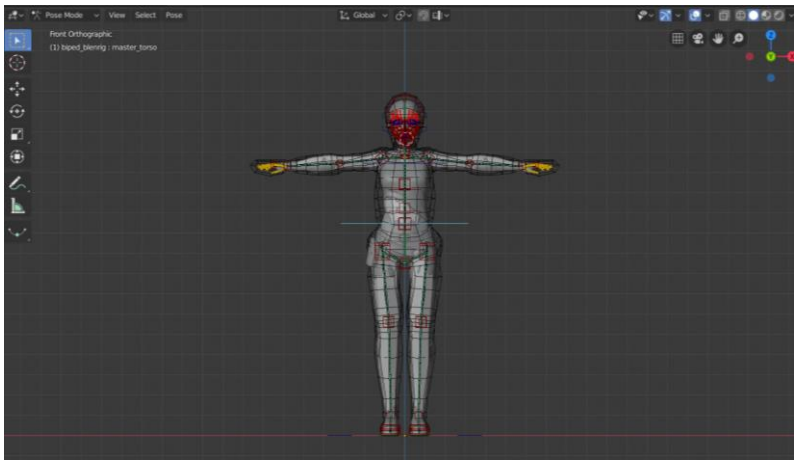
Successivamente si passa alle dita. In BlenRig le dita hanno anch'esse un sistema IK quindi è importante ricordarsi, quando si posizionano le ossa delle dita, di dare ai giunti una lieve curvatura, come una rotazione iniziale, così che l'IK sappia da che parte ruotare il dito.



Infine, è la volta del collo. È importante posizionare per prima cosa l'osso della testa nel punto di pivot della testa, e dopo si possono posizionare le altre ossa:

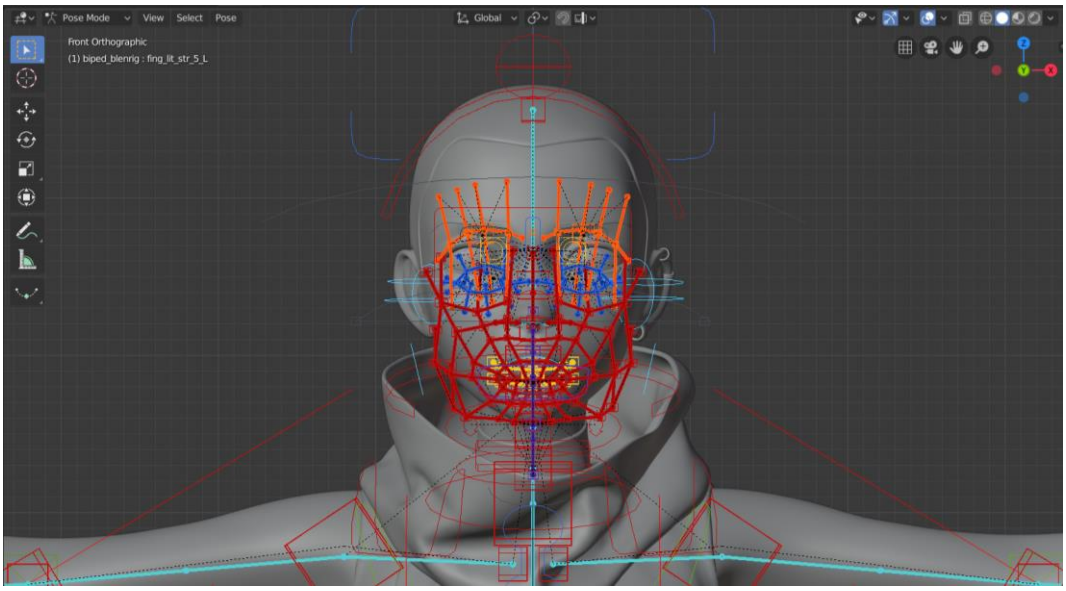


Nella seguente immagine si può notare come nel frattempo la mesh deform cage, che è già collegata al rig, si sia adattata alle proporzioni del modello. Rendendo inoltre visibile il livello *TOON* è già possibile scalare alcune parti della mesh deform cage, in modo che avvolga il personaggio.



4.2.2 Re-Target della faccia

Per prima cosa bisogna allungare l'osso della testa con il controller più in alto, dopo di che viene posizionato il primo giunto della testa a livello degli occhi e il secondo poco sotto il naso. Dopo aver aggiustato le orecchie, con il controller *face_master_str* è possibile regolare il posizionamento generale della faccia. Si può anche scalare così da adattarsi meglio al modello.

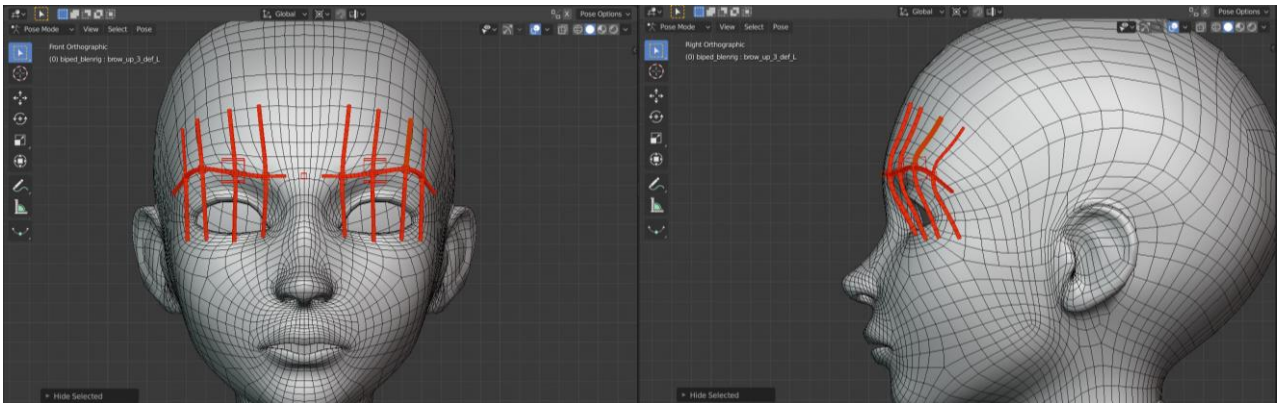


Si può modificare un controllo per un eventuale rig di un cappello e c'è anche un controller per gli occhiali.

In seguito, si può passare al riposizionamento delle ossa della faccia. È importante attivare la modalità *wireframe* per vedere i loop del volto.

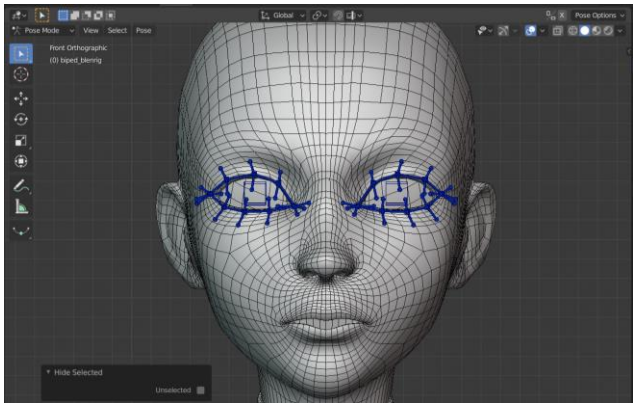
Ogni gruppo di ossa ha un proprio colore. Ogni gruppo può essere selezionato nel pannello *Armature* e isolato rispetto alle altre ossa. Inoltre, per ogni gruppo è presente un master control tramite cui è possibile spostare l'intera serie di ossa. Così facendo, è possibile modificare ogni parte del volto senza ostacoli visivi. È molto utile abilitare la modalità *B-Bone* nella sezione *Viewport Display* dello scheletro per visualizzare la curvatura effettiva delle ossa.

Ad esempio, nel caso del gruppo *STR_EYEBROWS* dopo aver aggiustato i controller principali, bisogna regolare le ossa verticali, provando a mantenerle sulla superficie del modello.

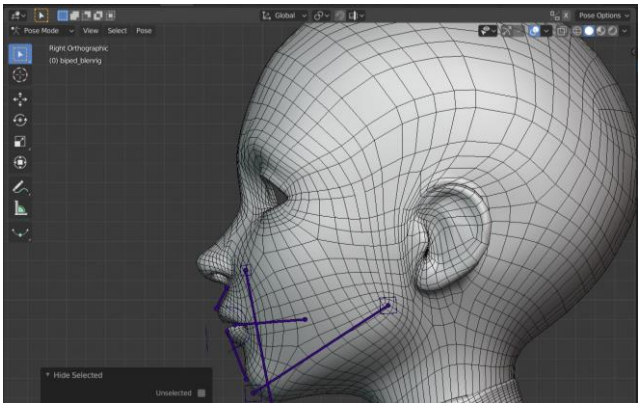


Da notare che le ossa di tipo *deformation* sono strutturate in griglie: questa è una tecnica strategica che permette di evitare che i vertici collassino quanto i giunti vengono articolati, ed è per questo che, come si vedrà nella sezione *weight paint* di questo capitolo, il weight paint verrà effettuato sia nelle ossa orizzontali che in quelle verticali.

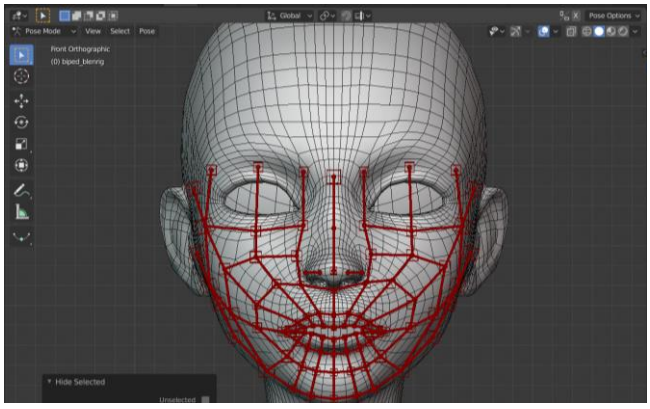
È possibile vedere il risultato del riposizionamento del resto delle *deformation bones*, gruppo per gruppo, nelle immagini seguenti:



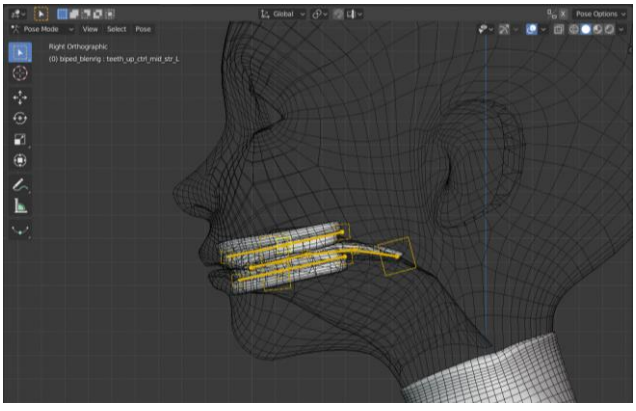
STR_EYES



STR_FACE_MECH



STR_FACE



STR_INNER_MOUTH

Nel caso del gruppo *STR_EYES* è necessario che il master controller sia posizionato al centro della mesh *eye.l*.

Il gruppo *STR_FACE_MECH* determina alcuni punti chiave della struttura facciale: l'osso *maxi_str_mstr* determina come verrà aperta la bocca, infatti il controller superiore viene posizionato nel punto di pivot della mandibola e il secondo controller viene collocato in corrispondenza del mento. L'osso *mouth_str* permetterà di curvare la parte inferiore del volto. L'osso *mouth_mstr_str* determina il posizionamento iniziale delle labbra e l'osso *mouth_mstr_ik* determina il punto di pivot per la bocca quando viene ruotata come un pezzo unico. *Mouth_ctrl_str* servirà in animazione per animare la bocca, come vedremo successivamente in questo capitolo. Infine, *lip_up_loc* e *lip_down_loc* determineranno il posizionamento dei controller secondari delle labbra superiori e inferiori.

Il gruppo *STR_INNER_MOUTH* viene posizionato in corrispondenza dei denti e della lingua.

Il gruppo *STR_FACE* è il gruppo più grande e complesso. Determina i movimenti della maggior parte del volto. Per posizionare bene le ossa di questo gruppo è necessario arrivare la modalità *B-Bone* e partire aggiustando le ossa del naso e in seguito posizionando il controller dell'angolo della bocca *mouth_corner_str_L* in corrispondenza dell'angolo sinistro. Come per tutto il resto dello scheletro, si è scelto di lavorare inizialmente sulle ossa sinistre copiando la loro posizione in seguito con il comando *copy pose>flipped on x axes.* Per questo è necessario che il modello sia perfettamente simmetrico rispetto all'asse x. Adesso si potrà procedere con il posizionamento di tutti i diversi loop delle labbra, cercando di farli combaciare il più possibile con i loop del modello. Dopo aver collocato i controller interni, si passerà a spostare quelli esterni, tenendo conto che i controller intermedi si sposteranno di conseguenza. Una volta posizionati per bene i controlli esterni, dunque si può procedere con quelli intermedi. Tutta la griglia deve corrispondere con la superficie della mesh e le curvature devono risultare levigate il più possibile e i loop delle ossa devono corrispondere il più possibile ai loop del modello.

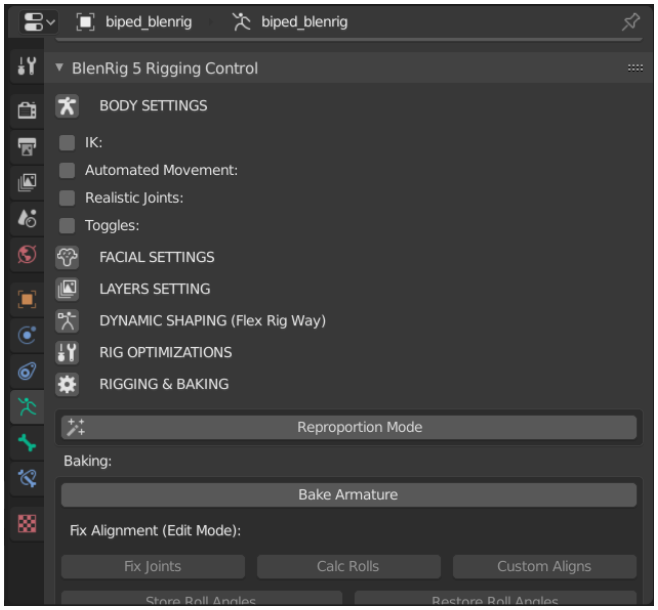
Una volta sistemate le ossa del volto, è possibile posizionare nel modo giusto i controller principali, come quelli delle labbra superiori e inferiori, quelli delle guance e dell'arricciamento del naso, che serviranno poi in animazione, in corrispondenza dei loop che andranno a muovere.

Con questo ultimo gruppo di ossa, è terminata la fase di re-target dello scheletro. La fase successiva sarà quella del *bind*, ovvero quella dell'unione tra la mesh e lo scheletro, in questo caso tramite la *mesh deform cage.*

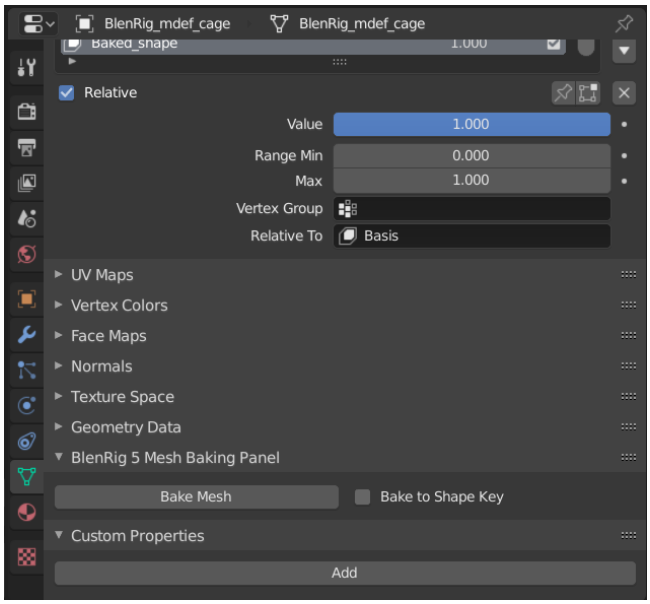
4.2.3 Baking del Rig

In questo paragrafo si procederà con il *bake* dello scheletro. In generale si fa per prima cosa il bake degli oggetti e alla fine il bake dello scheletro.

Come è già stato detto, selezionando lo scheletro e andando nel pannello *Data* nel menù laterale. Lì è possibile trovare la sezione *BlenRig* con diverse opzioni per lo scheletro BlenRig.

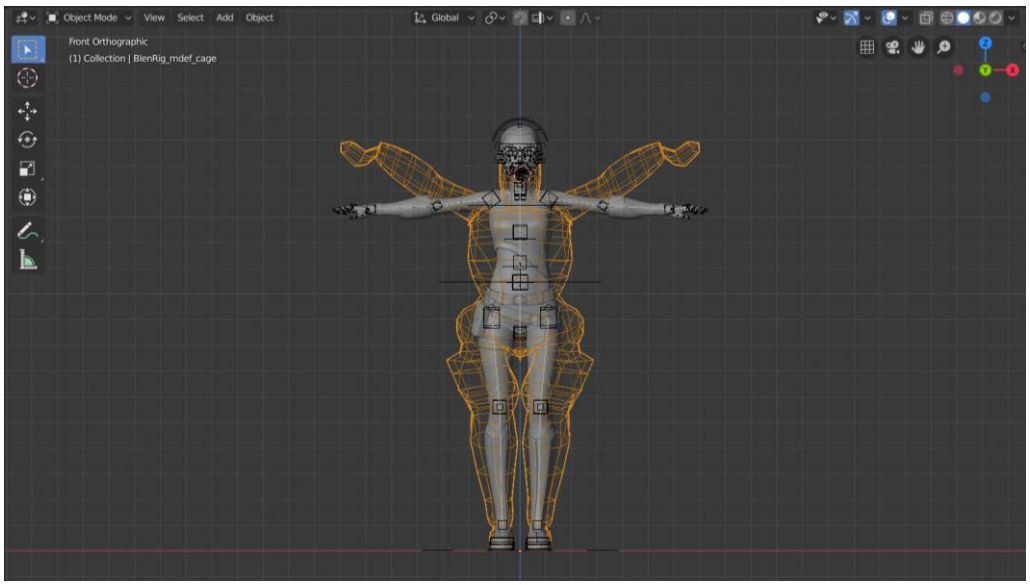


Allo stesso modo, quando si seleziona un oggetto, nel nostro caso selezionando la mesh deform cage, c'è una sezione BlenRig per quell'oggetto, ma nel caso di oggetti mesh ci sarà un bottone che permetterà di fare il bake di quegli oggetti.



In questo caso, poiché la mesh deform non ha nessuna shape key all'interno, non c'è bisogno di abilitare l'opzione *Bake to Shape Key*.

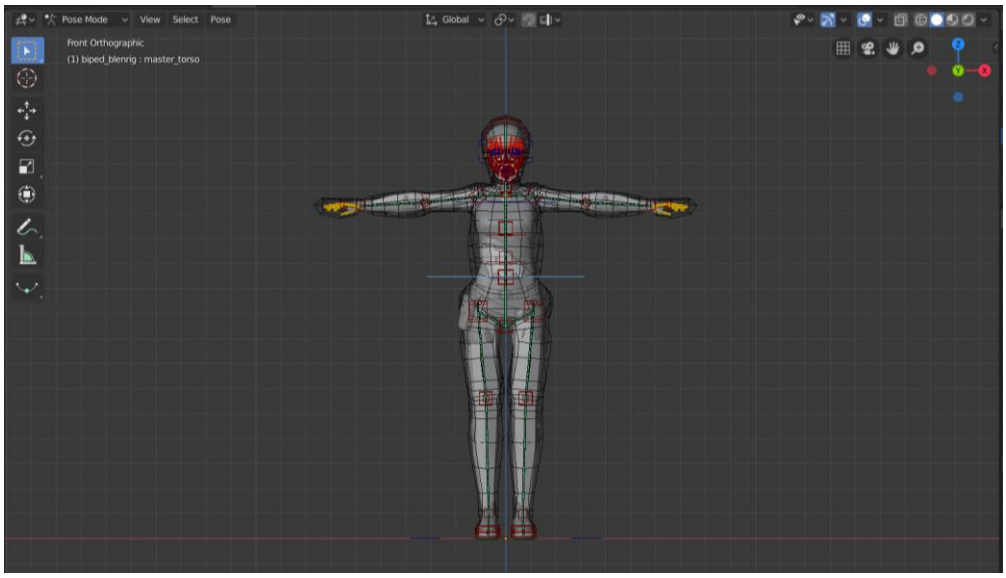
Dunque, premendo il pulsante *Bake Mesh* si potrà osservare come la mesh deform cage si deformerà in un modo ambiguo.



Questo succede perché la mesh deform cage si è adattata alle nuove proporzioni, ma lo scheletro la sta ancora deformando, quindi la mesh sta ricevendo una doppia deformazione. Tutto questo verrà risolto nel momento in cui verrà eseguito il bake dello scheletro.

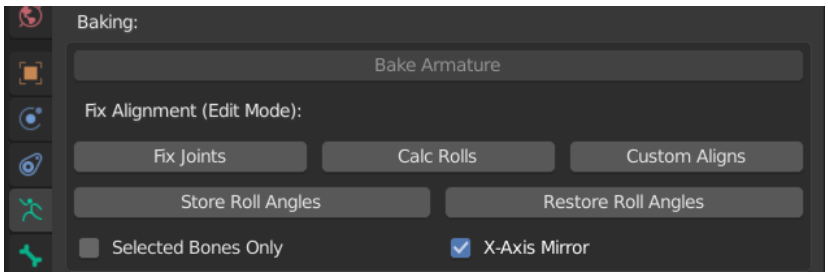
Facoltativamente si può anche effettuare il bake della mesh proxy che si utilizza per creare una versione a bassa risoluzione del personaggio.

Infine, si può eseguire il bake dello scheletro. Mantenendo attiva la *Reproportion Mode* e premendo il tasto *Bake Armature*, si può osservare come la mesh è ritornata normale, ma con le nuove proporzioni adattate al modello. Anche i Lattices si sono adattati alle nuove dimensioni.



Mode e premendo il tasto *Bake Armature*, si può osservare come la mesh è ritornata normale, ma con le nuove proporzioni adattate al modello. Anche i Lattices si sono adattati alle nuove dimensioni. Allo stesso modo, se si passa in *Edit Mode* è possibile vedere che le ossa si sono riadattate a una nuova proporzione, che è diventata la sua effettiva *Rest Pose*.

Dopo aver fatto questo e rimanendo in *Edit Mode* si procederà con l'eseguire diversi operatori:

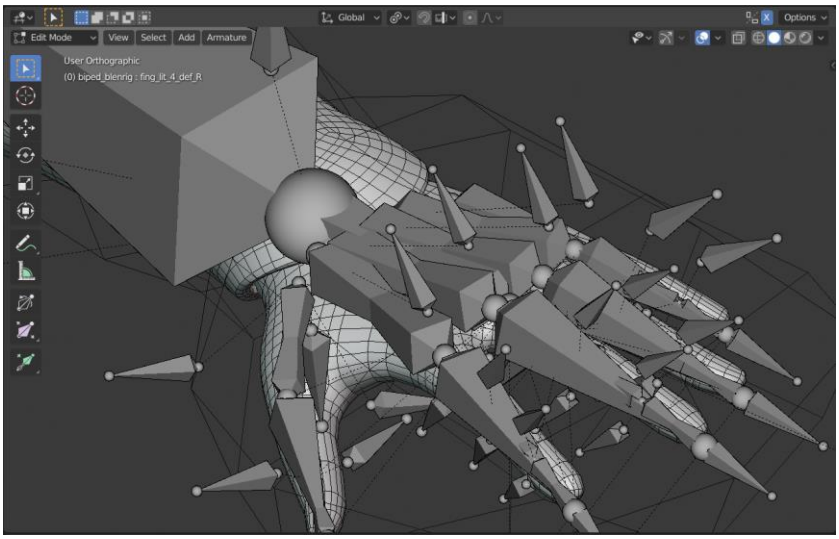


- *Fix Joint*: dopo il bake, alcune ossa dello scheletro si possono disallineare, e ciò viene risolto dopo aver premuto questo pulsante;
- *Calc Rolls*: serve per allineare automaticamente le ossa.

Nella maggior parte dei casi non è necessario selezionare l'opzione *Selected Bone Only*, ma se si stesse altamente modificando lo scheletro, potrebbe tornare utile.

Il pulsante *Store Roll Angle* serve per salvare l'angolo di roll per ogni osso. A volte l'angolo di roll può subire delle modifiche per il modo in cui Blender gestisce quei dati, dunque con quest'opzione è possibile salvare questi angoli, inoltre c'è il pulsante *Restore Roll Angle*, nel caso questi angoli si rompano, che permette di risalvarli.

Una volta allineati tutti i roll delle ossa, si procede con l'isolare il livello *Bone-Rolls*, in cui sono presenti le ossa che determinano gli angoli di roll di tutto lo scheletro. Se il personaggio si trova in t-pose, nel novanta percento dei casi questo passaggio non sarà necessario, poiché il calcolo automatico dei roll funziona con la maggior parte dei modelli, ma questo livello è utile nel caso si vogliano modificare i roll delle ossa. Dunque, se da questo livello si cambia il roll di un osso, e poi si preme il pulsante *Custom Aligns,* tutte le ossa dello scheletro si allineeranno agli angoli di roll di queste ossa. Pertanto, si procede con il modificare i roll delle dita così che combacino con l'effettiva rotazione della geometria delle dita, e dopodiché si procederà con l'allineare le falangi (visibili in modalità *Wireframe*) all'osso master tramite il comando *Recalculate Roll>To Active Bone.* Una volta orientate tutte le ossa nel modo giusto, si potrà premere il bottone *Custom Aligns*, per allineare le ossa di tutti i livelli ai roll.



Una volta soddisfatti con il risultato, bisogna premere il pulsante *Restore Roll Angles* e si potrà uscire dalla *Edit Mode*.

Prima di uscire dalla *Reproportion Mode* si dovrà premere il pulsante *Reset Constraint* perché dopo tutti questi maneggiamenti, potrebbe essere stata modificata la lunghezza di alcune ossa. Quest'operatore farà in modo che tutti i constraint funzioneranno nel modo corretto con le nuove lunghezze.

Una volta usciti dalla *Reproportion Mode*, si può vedere che lo scheletro funziona bene in pose mode e la mesh deform cage segue il rig nel modo giusto.

Questo finalizza la parte tecnica del processo di rigging. Nel prossimo paragrafo verrà sviluppata la parte più "artistica" che consiste nella deformazione.

4.2.4 La mesh deform cage

Questo paragrafo tratterà del rimodellamento della mesh deform cage in modo che circondi perfettamente il personaggio.

Prima di tutto è necessario disattivare, se presente, il modificatore *Subdivision* poiché la mesh deform verrà calcolata sulla mesh originale e nella mesh suddivisa, la posizione dei vertici cambia dall'originale e ciò potrebbe portare a posizionare male la mesh deform cage e alcuni vertici potrebbero rimanere al di fuori di essa e non essere tenuti in conto per il calcolo.

Per modellare la mesh deform cage basta andare in Edit Mode, assicurandosi che l'opzione x-mirror sia attiva, e iniziare a muovere i vertici della mesh cercando di mantenere le relazioni tra i loop della mesh deform cage e i giunti del modello. La mesh deform è stata modellata con diverse tecniche che contribuiscono alla preservazione del volume, ciò significa che quando si muove il personaggio i

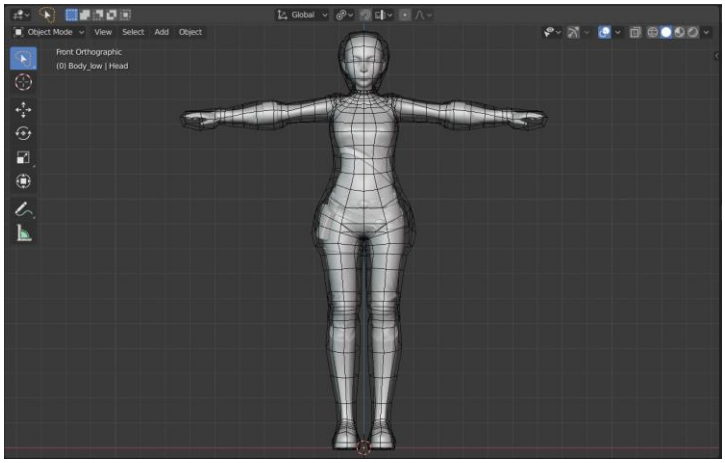
giunti non si intersecheranno né si contrarranno. È dunque necessario preservare la topologia della mesh deform cage per avere una buona deformazione del corpo. Non è indicato usare tecniche automatiche come il modificatore *Shrinkwrap* poiché si modificherebbe troppo la struttura della cage e una buona deformazione non sarebbe garantita. È utile di tanto in tanto attivare l'opzione *x-ray* per poter vedere quali vertici del modello si trovano al di fuori della mesh deform cage.

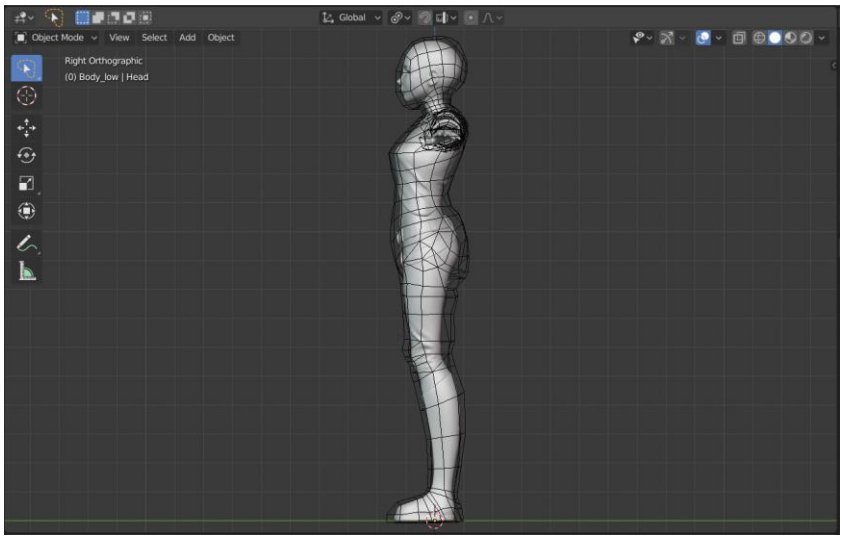
Uno degli ottimi aspetti del mesh deform che, nel gergo tecnico è definito *Harmonic Coordinates Deformation,* è che, se la mesh deform cage viene posta molto distante dal modello, si otterrà una deformazione più lieve del personaggio. D'altro lato, se la si pone più vicina al modello, si otterrà una deformazione più forte e definita. Tutto ciò risulta molto utile perché nei posti più complessi, come per esempio le spalle, si può usare questa tecnica per evitare intersezioni, e soprattutto la mesh deform consente di avere sempre una deformazione organica e lineare.

Un altro aspetto positivo della mesh deform cage è che non deve essere perfetta, ovvero, quando lo scheletro si muove, la mesh deform può avere delle intersezioni nei giunti, ma queste non si rifletteranno nel modello, che continuerà a deformarsi in maniera organica. Quindi non è necessario essere precisi, perché la mesh deform cage consiste solo nell'estrarre i vertici del personaggio così che sia sempre liscio e armonioso e permette di avere effetti di deformazione che non sono raggiungibili dalla singola deformazione delle ossa. Se si vuole comunque associare il personaggio con lo scheletro, si può comunque fare. Infatti, la tecnica del mesh deform non è compatibile con i game engine.

Nonostante sia sconsigliabile utilizzare tecniche automatiche, è sempre possibile usare lo *Smooth* per appianare certi vertici che potrebbero essere stati compromessi dopo il bake.

Una volta modificata la mesh deform cage, dovrebbe apparire come in figura:



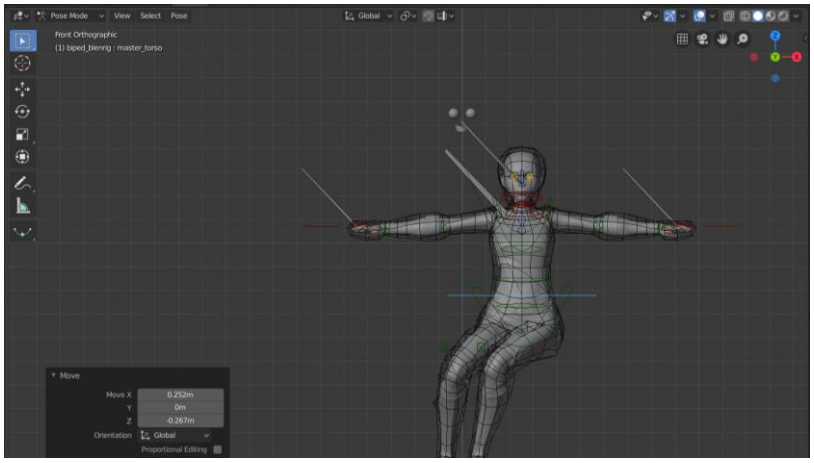


Si può passare dunque ad aggiungere un modificatore *Mesh Deform* al modello che, nella sezione *modificatori* deve essere posto al di sopra del modificatore *Subdivision*. Il target del modificatore sarà *BlenRig_mdef_cage*.

Naturalmente se il modello del personaggio è formato da più mesh, come nel nostro caso, il modificatore dovrà essere aggiunto a tutte le mesh costituenti il modello.

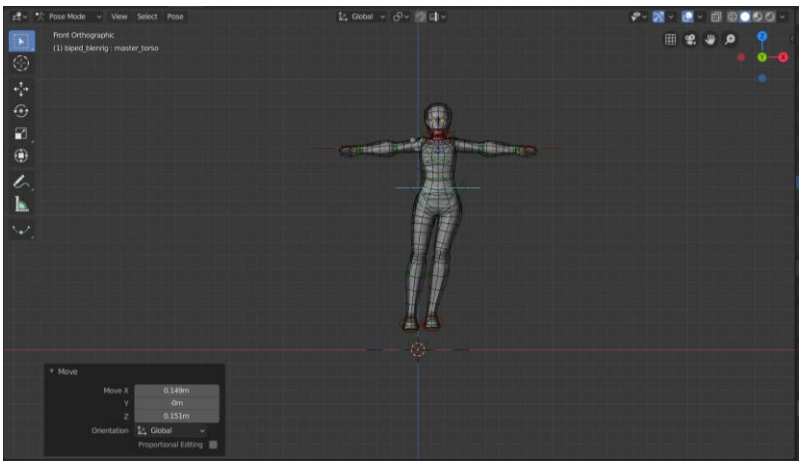
Per far sì che la deformazione funzioni non rimane altro che premere il pulsante *Bind*. Come si può notare, il modificatore *Mesh Deform* contiene il campo *Precision* il cui valore di default è 5, che è il giusto valore per capire come funziona la mesh deform, poiché rende il calcolo molto veloce.

Avendo premuto il pulsante *Bind* per ogni mesh del modello, si procede con il muovere lo scheletro in vari modi, per vedere se alcuni vertici non sono stati inclusi nel calcolo. Come è possibile notare dalla seguente immagine, funziona tutto abbastanza bene, eccetto per alcuni vertici che non vengono influenzati dalla deformazione:



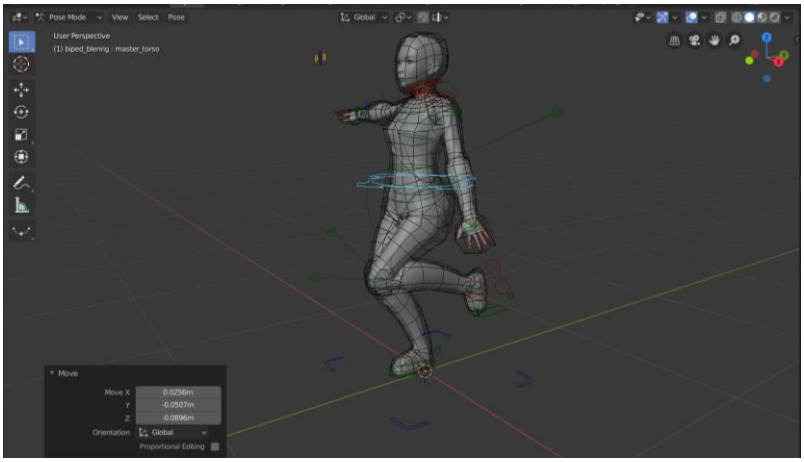
Si procederà con il premere il pulsante *Unbind* e il correggere la mesh deform cage in Edit Mode in modo che includa quei vertici che sono rimasti fuori.

Dopo aver di nuovo eseguito il Bind, si può notare come adesso tutti i vertici si muovono seguendo il rig, ciò vuol dire che sono stati inclusi nel calcolo:



Adesso si procede con il "vero" bind, ovvero quello con una precisione maggiore. Il livello 5 di precisione è ottimo per un test veloce di funzionamento, ma la deformazione che offre non è abbastanza precisa. Il livello migliore è il livello 7. Dopo aver cambiato dunque questo valore di precisione, si passa ad effettuare nuovamente il bind per ogni mesh a cui è stata associata la mesh deform cage. Il livello 7 può richiedere diversi minuti di calcolo, ad esempio in questo caso il calcolo è durato circa sei minuti.

Dopo il calcolo si può notare come il modello si deformi perfettamente:



Nel prossimo paragrafo verrà affinata la deformazione cercando di dare attenzione ai più piccoli dettagli.

4.2.5 Rifinire la deformazione della cage

In questo e nel prossimo paragrafo, verranno mostrati diversi modi di raffinare la deformazione del personaggio ma, nel caso si tratti di un personaggio secondario o se si è già soddisfatti della

deformazione ottenuta, si possono tralasciare questi passaggi e procedere con il weight paint delle mani e della faccia.

Per prima cosa bisogna controllare la qualità della deformazione in ogni parte del personaggio. Per far ciò bisogna abilitare il livello *DEFORMATION,* che mostra le ossa che si occupano della deformazione, nel caso si dovesse modificare il weight paint della mesh deform cage e abilitare il modificatore *Subdivision*.

È molto utile in questi casi, creare un driver per gestire il livello di Subdivision dell'intero personaggio, senza dovere andare, per ogni mesh che lo compone, ad attivarlo ogni volta. Nel pannello laterale di BlenRig, si può trovare una proprietà chiamata *Model_Re,* che sta per *Model Resolution*. Cliccando con il tasto destro si può selezionare l'opzione *Copy Data Path,* che copierà il percorso (data path) della proprietà. Selezionando poi uno dei modelli, si cliccherà con il tasto destro, nel pannello Modificatori, nella proprietà *View* e si selezionerà l'opzione *Add Driver*.

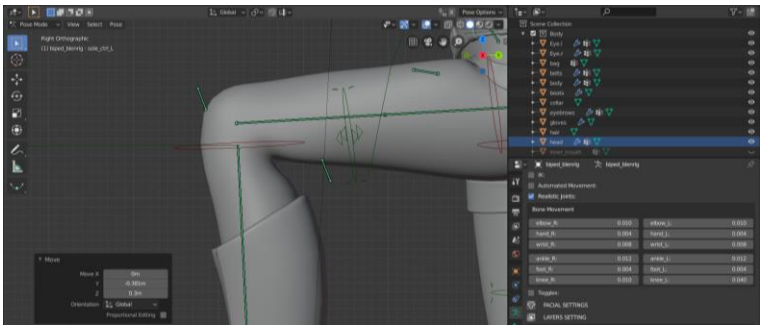
Le proprietà del driver verranno gestite, all'interno dell'editor dei driver, nel modo seguente:

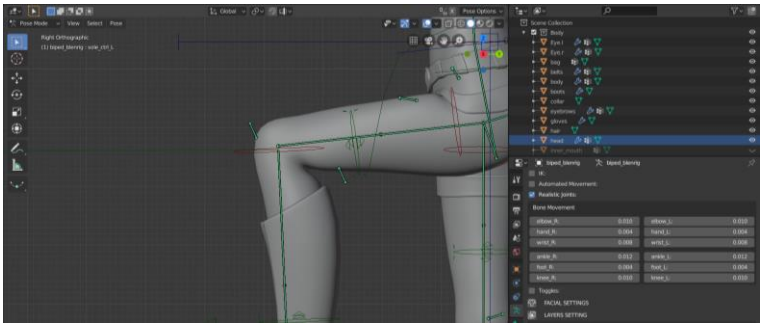
- Nella sezione *Type* verrà selezionato *Maximum Value;*
- Si definirà l'opzione *var* come *Single Property,* da qui si potrà selezionare l'oggetto *bipe_blenrig*, poiché la proprietà *Model Res* si trova all'interno del rig stesso, in particolare all'interno di un osso chiamato *properties,* e si può incollare il data path copiato in precedenza nella sezione *Path.*

Una volta copiato il driver per ogni modello, all'interno della proprietà *view* del modificatore Subdivision, il livello di suddivisioni del modello è controllato dalla proprietà all'interno del pannello di BlenRig, chiamata *Model Res.*

È conveniente lasciare al livello 1 il valore di Model Res, in quanto è più simile a ciò che verrà fuori dal render.

In BlenRig vengono usate alcune tecniche per migliorare la deformazione. Una di queste è la mesh deform cage. Un'altra è quella di utilizzare le ossa di *extra deformation* che aiuteranno nel mantenimento dei volumi. Una terza tecnica è quella dei *Realistic Joints*. È possibile trovare i parametri del *Realistic Joints* all'interno della sezione *BODY SETTINGS* nel pannello BlenRig dello scheletro. Con quest'ultima tecnica si possono emulare i volumi effettivi che hanno le ossa umane. In molti casi, quando i giunti si articolano, le ossa non solo ruotano ma si allontanano dall'osso vicino. Quindi, le ossa umane hanno spesso un dislocamento dal punto di pivot dell'articolazione, e questo dislocamento è ciò che i parametri del *Realistic Joints* cerca di emulare.

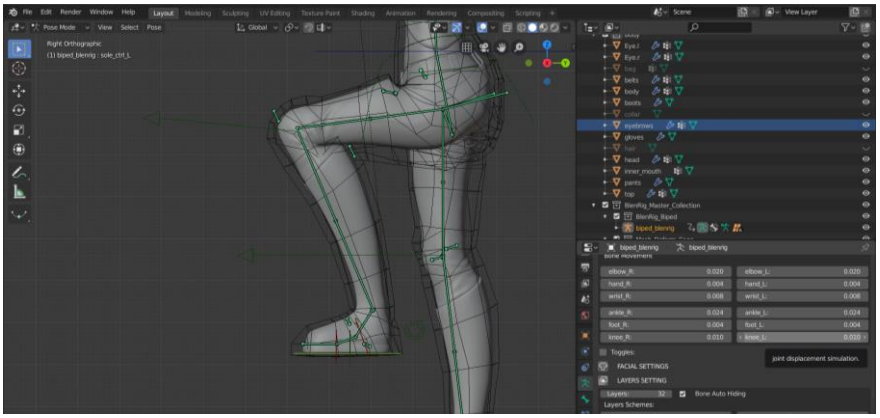




Per quanto riguarda la tecnica delle ossa di extra deformazione, queste ossa contengono un fattore di correzione e si allontanano dal giunto quando esso ruota. È possibile controllare la deformazione modificando il comportamento di queste ossa. Selezionando una di queste ossa, nella sezione *Bone Constraint* è presente un constraint di tipo *Transformation* e cambiando il valore nei campi *Destination* si può modificare il comportamento di queste ossa.

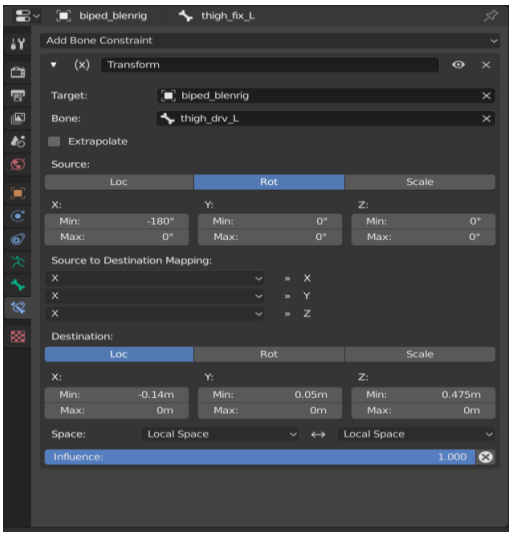
Per prima cosa comunque è consigliato modificare il weight paint delle ossa in questione sulla mesh deform cage.

Una particolare area in cui la deformazione risulta ambigua, e quella del giunto tra coscia e bacino:



Si procede inizialmente con l'abbassare il valore dell'osso *thigh_fix_L* dal pannello dei constraints. Per chi non avesse familiarità con i constraint, si può leggere il constraint nel modo seguente:

quando l'osso *thigh* ruota di -180° lungo l'asse x (come si può vedere l'asse x influenza gli assi x, y e z dell'osso di destinazione), l'osso di destinazione, ovvero l'osso di deformazione selezionato (*thig_fix_L* in questo caso), si muoverà (la sua *Location* viene influenzata) in certi valori di x, y e z. Dunque, l'unica cosa da modificare in questo constraint, sono proprio questi valori di x, y e z.



Una volta cambiati i valori nel modo più consono, bisogna ricopiarli all'interno del constraint dell'osso opposto rispetto a x (*thigh_fix_R*), in modo da avere lo stesso comportamento sia a destra che a sinistra.

È opportuno migliorare la deformazione di questo giunto anche con il weight paint sulla mesh deform cage.

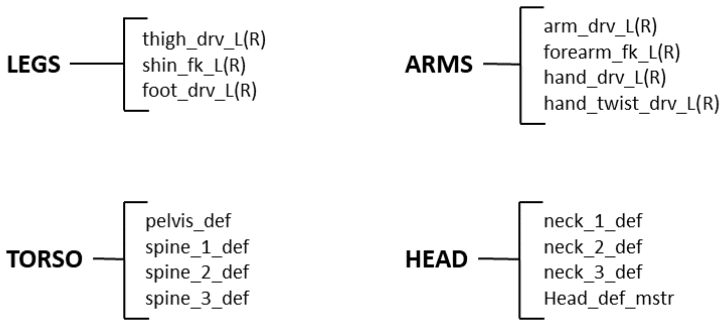
Per riassumere, l'iter che bisogna seguire è: scegliere una posa "problematica" selezionare, per quella posa, le ossa di deformazione, eseguire il weight paint e provare a dare equilibrio alla deformazione cambiando i valori in alcune ossa. Infine, come si vedrà nello specifico nel prossimo paragrafo, verranno create delle shape keys per certe pose specifiche, per definire la deformazione nel modo più preciso possibile.

4.2.6 Aggiungere le shape key alla cage

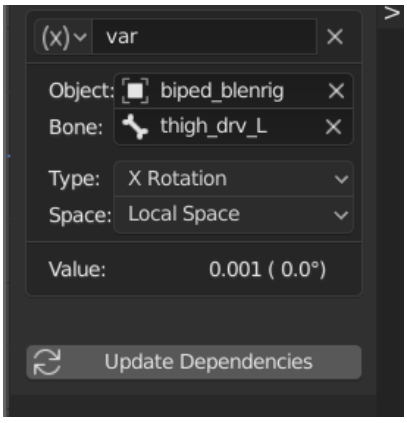
La deformazione fornita dalla mesh deform cage porta a risultati molto buoni senza ulteriori tecniche di deformazione. Anche in pose estreme la deformazione è piuttosto valida. Persino le spalle, che di solito sono un punto particolarmente difficile da gestire, funzionano bene senza ulteriori manomissioni. Tuttavia, se si vuole andare nello specifico nella deformazione, è necessario lavorare manualmente a certi dettagli, ed è qui che entrano in gioco le shape keys.

Un altro pregio della mesh deform è che, essendo una mesh con un ridotto numero di poligoni, creare le shape keys risulta un compito molto semplice, molto più semplice che aggiungere le shape keys al modello di base del personaggio.

Ecco di seguito la lista di ossa che BlenRig usa in ogni parte del corpo, utile quando si vuole creare una shape key collegata alla trasformazione di un osso tramite driver.



Quando si vuole modificare una certa posa, per prima cosa bisogna selezionare la mesh deform cage e aggiungere una shape key nella sezione *Data*, rinominare la shape key in base alla posa che si sta effettuando e al lato del corpo a cui appartiene (destro o sinistro) e aprire la finestra dell'editor dei driver. Cliccando con il tasto destro sul valore della shape key si può aggiungere un driver associato ad esso. È necessario settare il driver al *Maximum Value* e, nel campo *Object* della sezione *Trasform Channel,* selezionare *biped_blenrig.* È necessario controllare in quale asse di rotazione si deve attivare il driver e specificarlo nel driver, in questo modo:



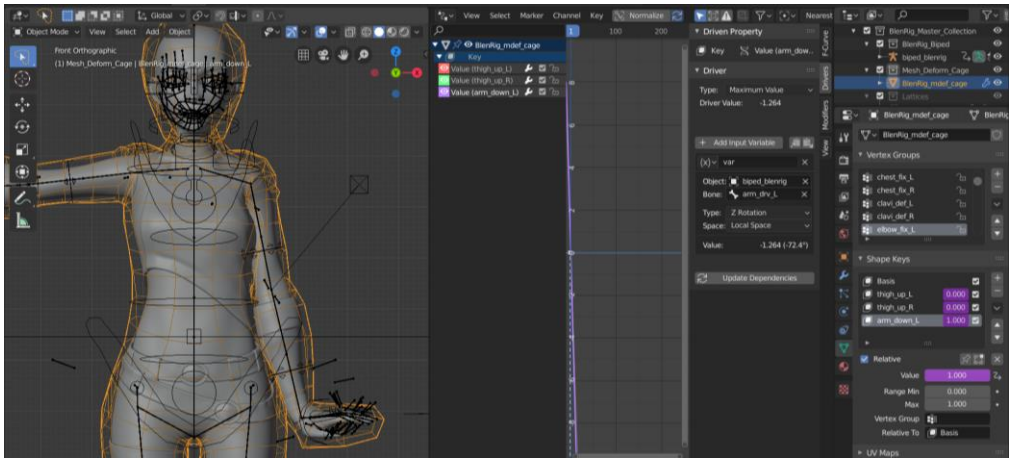
Ciò vuol dire che il driver guarderà la rotazione sull'asse x di quest'osso per attivarsi.

Il campo *Value* indica la rotazione dell'osso (*thigh_drv_L*) in radianti, e in questo valore, si vuole che il driver abbia il valore di 1, così che la shape key è pienamente attiva quando la gamba è ruotata di quel particolare angolo. Dunque, viene aggiunto un modificatore *Generator* e nel campo *x* si scriverà 1/ "value", ovvero il valore di cui prima. Adesso il valore della shape key cambierà secondo la rotazione dell'osso in questione.

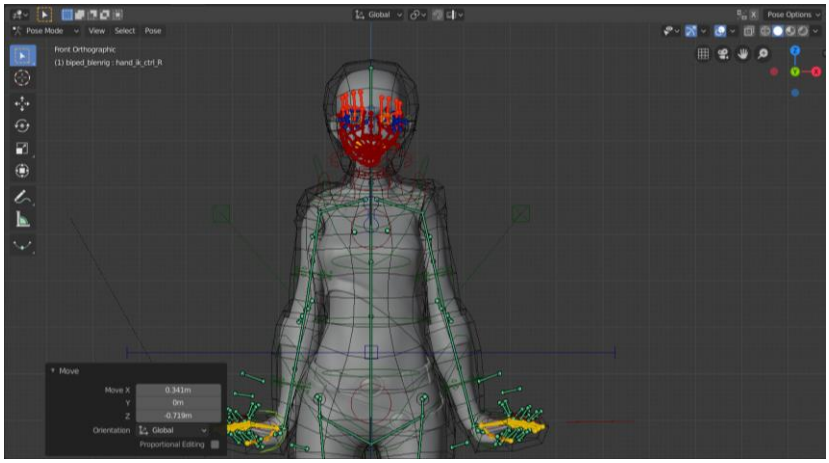
Infine, in Edit Mode, si può modificare la shape key per migliorare la deformazione d dell'area in questione (ricordando di disabilitare l'opzione *x mirror* poiché si sta lavorando soltanto con un lato del modello).

Una volta soddisfatti del risultato, si può copiare la shape key sul lato opposto del modello tramite i comandi *New Shape From Mix* e *Mirror Shape Key (Topology)* e anche il relativo driver (ricordando sempre di modificare l'osso a cui è collegato il driver e il nome della shape key).

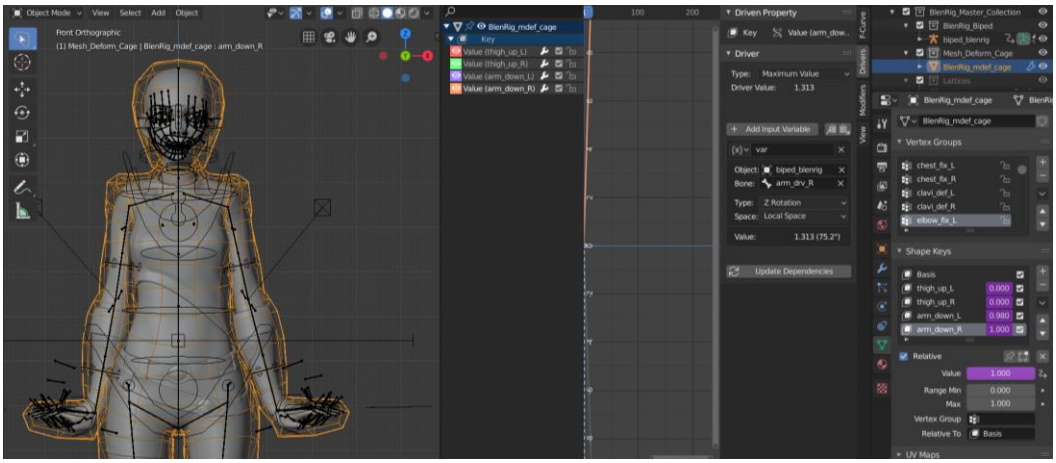
Nelle successive immagini, un esempio di come è stata migliorata la rotazione del braccio sul modello di Nadya tramite shape key e driver:



In questa prima immagine è possibile vedere la rotazione originale senza shape keys.



Nella seconda immagine i vertici della mesh deform cage sono stati modificati per ottenere una deformazione migliore.



Infine, la shape key è stata copiata sul lato opposto.

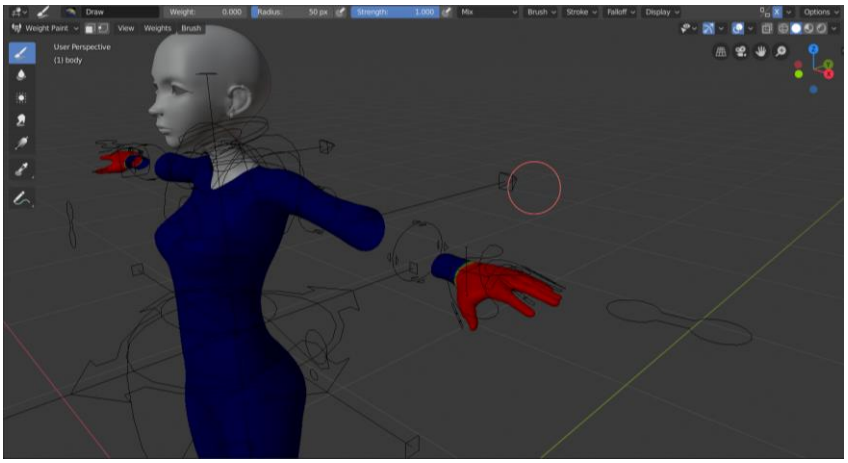
Lo stesso procedimento è stato utilizzato per le gambe e i gomiti del modello. A questo punto la fase di deformazione del corpo è terminata e non rimane che lavorare sulle mani ed il volto.

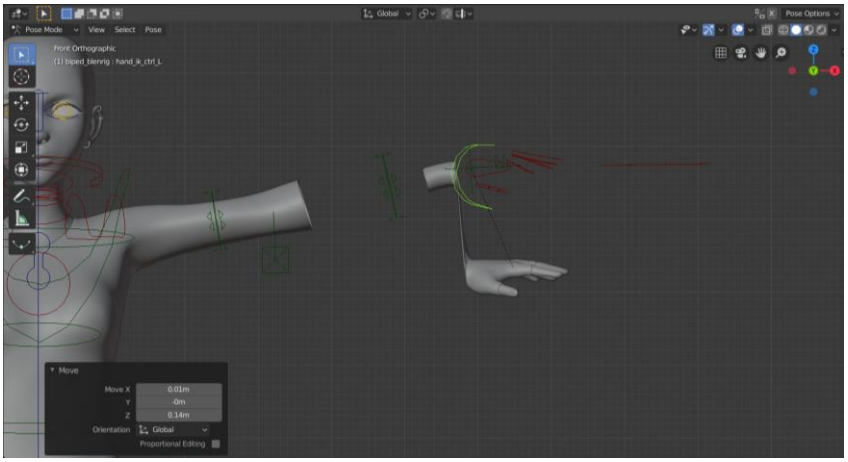
4.2.7 Weight paint delle mani

Come è già stato appurato, il metodo *Mesh Deform* è ottimo per le deformazioni generiche ma, in zone rigide o di piccole dimensioni, per esempio mani e dita, il modificatore non lavora abbastanza bene e in quei punti sarà doveroso usare il modificatore *Armature*.

Per generare la divisione tra le influenze dei due modificatori, è necessario aggiungere a ogni mesh un *vertice group* (gruppo di vertici), che nel nostro caso sarà rinominato *no_mdef* e aggiungere un modificatore *Armature* a ogni oggetto, ponendola in cima alla lista dei modificatori, e selezionare *biped_blenrig* nel campo *Object.* È in questo momento che risulta utile aver diviso il modello in diverse mesh, in particolare la testa dal corpo, poiché nelle mani si dovrà attivare l'opzione *Preserve Volume* all'interno del modificatore, ma nella testa non si dovrà applicare. Questo perché c'è la possibilità che questa opzione generi artefatti di deformazione con rare combinazioni di scalamento e rotazione di ossa e, poiché il rig facciale si basa sull'allungamento e ridimensionamento delle ossa, è consigliabile non utilizzarla. Inoltre, bisogna indicare nel campo *Vertex Group* il gruppo dei vertici creato precedentemente oltre che selezionarlo anche all'interno del modificatore *Mesh Deform* nella sezione *Vertex Group* ma con l'opzione *Invert*.

Si procede con il selezionare in modalità Weight Paint i vertici appartenenti a *no_mdef,* sial per le mani che pel la faccia, e si può notare come le parti selezionate non si muoveranno più con il resto del corpo.

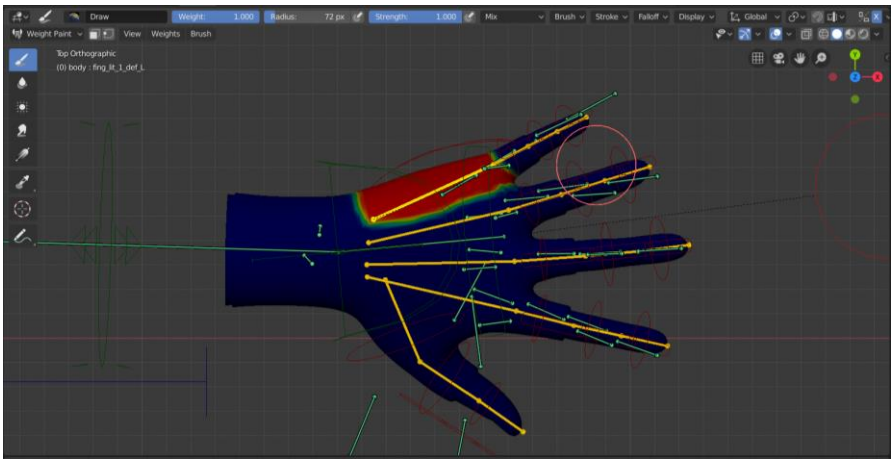


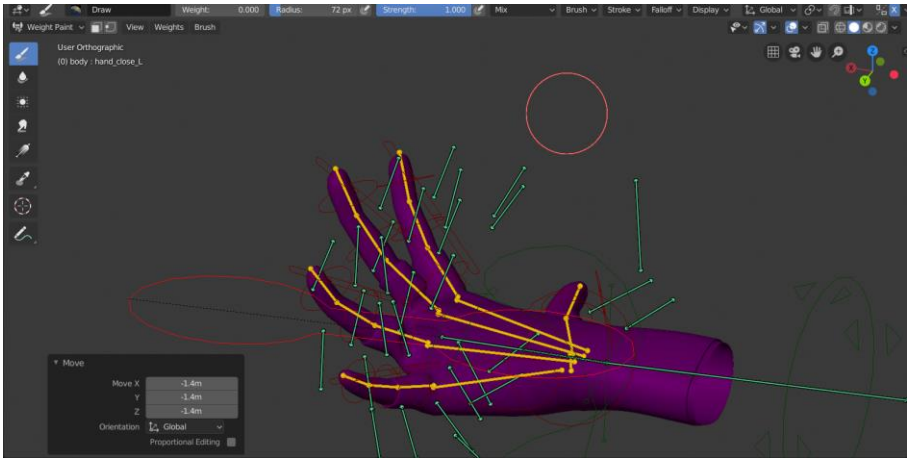


Questo succede perché tramite il modificatore *Mesh Deform* si sta dicendo a Blender che tutti i vertici che non appartengono al gruppo *no_mdef* dovranno essere deformati con il Mesh Deform (i vertici appartenenti all'area blu della prima immagine). Dall'altra parte, con il modificatore *Armature* si sta dicendo a Blender che i vertici appartenenti a *no_mdef* dovranno essere deformati con lo scheletro e poiché non si è ancora associato nessun vertice alle ossa di deformazione, il modello non si sta deformando.

Dunque, abilitando il livello *DEFORMATION BONES* dello scheletro, si può procedere con l'associare i giusti pesi alle ossa della mano dapprima in modo grossolano e poi lavorando sui dettagli per far sì che si preservino i volumi nel modo corretto.

Dalle immagini seguenti si può osservare un weight paint grossolano dell'osso *fing_lit_1_def_L,* e successivamente il risultato ottenuto dopo lo smussamento dei pesi.





Mantenendo attiva l'opzione x mirror, tutte le operazioni effettuate sulle ossa del lato sinistro saranno ripetute su quelle del lato destro del personaggio.

4.2.8 Weight paint della faccia

Una volta concluse le operazioni con le mani si procederà allo stesso modo con la testa, "tingendo" il vertex group *no_mdef* e successivamente l'osso *head_def* lavorando poi sullo smussamento della transizione tra i due gruppi di vertici (come si è fatto precedente nel caso del gruppo *hand_def_L*).

Il primo osso da "dipingere" è la mandibola (*maxy*) che determinerà l'apertura della bocca. Successivamente si procederà con il weight paint di tutte le ossa, orizzontali e verticali, di deformazione della faccia, delle sopracciglia, degli occhi e infine della bocca interna (denti e lingua), che nel caso di Nadya sono separate dalla mesh della testa e a cui è stato aggiunto il modificatore *Armature.* In seguito, sarà possibile effettuare un *Join* con l'oggetto "testa" e rifare il *bind* alla mesh deform cage.

Nonostante il weight paint sia fatto nel modo più minuzioso possibile, ci sono problematiche che potranno essere risolte solo in un secondo momento con delle shape key correttive.

4.2.9 Rig di oggetti extra

Gli oggetti esterni che finora non fanno parte del rig sono: piercing, denti e gengive, occhi destro e sinistro, guanti, collare e borsetta. Li analizzeremo pezzo per pezzo.

È stato precedentemente accennato che, per quanto riguarda denti e gengive, questi sono stati forniti di un modificatore *Armature,* di un opportuno weight paint, stavolta effettuato tramite assegnazione di vertici in Edit Mode, e infine tutta la mesh è stata unita all'oggetto *head* tramite il comando Join.

Allo stesso modo sono stati trattati i piercing. Tramite assegnazione di vertici in Edit Mode sono stati assegnati alle ossa corrispondenti e infine si è eseguito un Join alla mesh della testa.

Dopo l'operazione di join è necessario rieseguire il bind alla mesh deform cage.

Gli occhi sono rimasti oggetti separati e associati allo scheletro tramite la funzione *Set Parent To>Bone* rispettivamente alle ossa *eye_def_L* e *eye_def_R.*

Il collare e la borsetta sono stati animati con un altro metodo, che sfrutta le proprietà dei *clothes*, sperimentato da un altro componente del team, per cui non verranno analizzati all'interno di questa tesi.

I guanti, infine, sono stati riggati allo stesso modo delle mani, con un modificatore *Mesh Deform* e uno *Armature* relativo al gruppo di vertici *no_mdef.* La scelta di separazione tra mani e guanti è dovuta prima di tutto al sovrapponimento di vertici che impediva un corretto weight paint delle mesh unite, oltre al fatto che si potrebbe avere l'esigenza in un secondo momento, di creare una scena con il personaggio sprovvisto di guanti. Il contro di questa scelta è sicuramente il fatto che una volta provando le animazioni mi sono accorta che le due mesh si sovrapponevano tra loro nel momento in cui la mano veniva chiusa. Ho risolto questa problematica con un semplice ma furbo meccanismo, ovvero quello di associare alle mani un gruppo di vertici (corrispondenti ai vertici al di sotto dei guanti) di nome *mask* e un modificatore *Mask* che nasconderà i suddetti vertici e potrà essere attivato in animazione quando necessario, in modo da eliminare qualsiasi sovrapposizione di mesh.

Si è anche deciso di aggiungere al rig base, due ossa per il seno, collegate all'osso *spine_3_def* con un relativo weight paint associato.

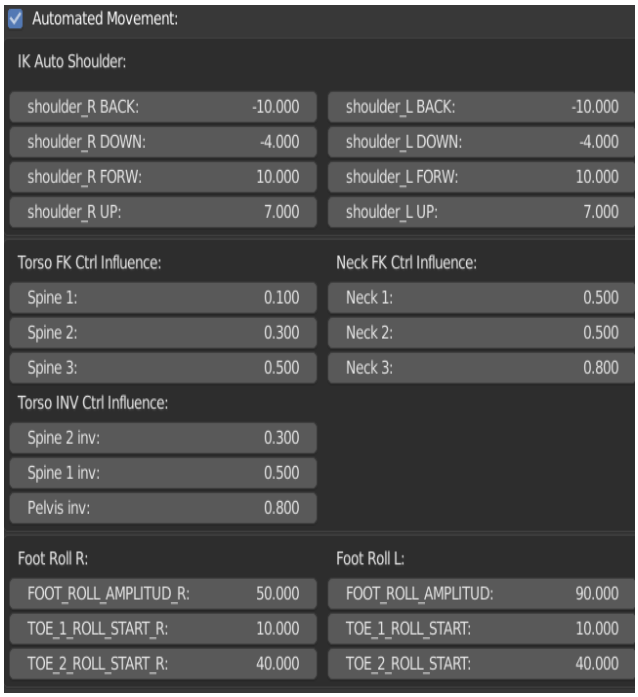
4.2.10 Parametri del corpo

Adesso che tutti gli oggetti del personaggio sono correttamente riggati non resta che cambiare qualche parametro del corpo prima di procedere con il setup del volto.

Nella sezione *BODY SETTINGS* selezionando *IK*, si potranno trovare diversi parametri che permettono di sovrascrivere il comportamento dell'IK di gambe e braccia.

I controller che hanno un comportamento IK sono *shin_ik_L(R)* e *forearm_ik_L(R)* all'interno del livello *BODY-TOON*. In generale, le ossa *IK* dovrebbero avere un angolo di inclinazione iniziale così che l'IK sappia come dovrebbero ruotare i giunti ma in alcuni casi questa inclinazione non è fornita dall'anatomia del personaggio. È qui che entrano in gioco i parametri di cui prima. Se non ci fossero questi parametri di rotazione iniziale la gamba non sarebbe in grado di ruotare in modo corretto. Quindi questi parametri sono utilissimi nel gestire l'IK anche se il modello non ha una inclinazione anatomica dei giunti. Nel caso di Nadya, l'inclinazione anatomica delle gambe e delle braccia è presente, dunque se il valore di *shin_ik_L(R)* o di *forearm_ik_L(R)* viene settato a zero, l'IK funzionerà ugualmente.

Nella sezione AUTOMATIC MOVEMENT, il primo gruppo di parametri corrisponde alle spalle. Cambiando questi valori è possibile gestire il movimento di rotazione delle spalle.



I parametri di *Torso FK Ctrl Influence, Neck FK Ctrl Influence* e *Torso INV Ctrl Influence* definiscono come questi controller principali influenzano i giunti individuali del torso e del collo. Se viene ruotato *torso_fk_ctrl* è possibile vedere come, cambiando i vari parametri, la pina dorsale ruota in modo diverso. I parametri di *Torso INV Ctrl Influence* servono quando il torso è settato in modalità *Invert.*

Infine, i parametri *Foot Roll L(R)* determinano come il piede si alza dal tacco fino alla punta. Cambiando i suddetti parametri è possibile gestire il modo in cui il piede si piega con l'IK.

4.2.11 Azioni facciali

In quest'ultimo paragrafo si procederà con l'impostazione delle espressioni facciali. Nonostante BlenRig abbia un rig facciale molto complesso, i controlli facciali non sono per niente difficili da usare, in quanto, quando si anima, si usano soltanto alcuni controlli base per muovere il resto della faccia in modo lineare, in particolare *mouth_ctrl, mouth_corner_ctrl, cheek_ctrl, eye_lid_ctrl* e *eyebrow_ctrl,* che si trovano nel livello *FACIAL 1,* e altri controlli secondari, ad esempio quelli del livello *FACIAL 2* che sono più specifici e aiutano ad aggiungere più dettagli all'espressione del personaggio, e infine nel livello *FACIAL 3* ci sono controlli per ogni giunto del rig facciale ma è improbabile che verrà utilizzato questo livello in animazione.

Per quanto riguarda tutti questi movimenti, alcuni di essi sono raggiunti usando constraints e meccanismi automatici, altri invece si ottengono tramite *actions.* Ad esempio, le sopracciglia si muovono con i constraints, quindi non c'è limite ai movimenti che possono fare. Invece, quando si sposta *mouth_corner_ctrl* verso l'interno, quel movimento è definito dalle *actions.* Queste azioni predefinite si muoveranno meglio o peggio in base alla forma del viso del personaggio, per questo è necessario modificarle.

È possibile definire il range di movimenti dei controller principali all'interno del pannello del rig, nella sezione *FACIAL SETTINGS* selezionando F*acial Movement Ranges.* Cambiando determinati valori si può fare in modo che i controller seguano perfettamente i movimenti del modello.

Quando un azione, come ad esempio l'abbassamento della palpebra, non funziona perfettamente come dovrebbe, si può modificare, facendo uso dell'*action editor* e delle ossa che si trovano nel livello *ACTIONS*. Dunque, una volta aperto l'Action editor e selezionando l'azione da modificare (in questo caso *zrig_eyelids_upper*), è possibile vedere di quanti frame è composta l'azione (in questo caso quattro). In generale il frame 0 è la rest pose. In questo caso il frame -1 è il movimento verso l'alto della palpebra, il frame 1 è la posa intermedia e il frame 2 è la posa finale. Disattivando l'opzione *keying on* sarà possibile modificare ogni frame muovendo i controller.

Questo metodo permette di modellare la posa come se queste azioni predefinite fossero shape keys dello scheletro. Quindi è possibile deformare manualmente ogni espressione del personaggio.

Una volta modificate le azioni in modo da rendere i movimenti facciali più fluidi, a questo punto tutta la meccanica del volto è definita.

4.3 Auto-Rig Pro Overview

Auto-Rig Pro è una soluzione automatica tutto-in-uno per il rigging dei personaggi e per il retarget delle animazioni.

Gli oggetti del rig di AutoRig-Pro fanno parte di una collezione madre chiamata di default *character1*.

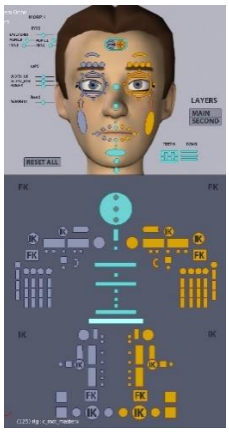
Sono presenti due oggetti armature:

- L'oggetto *rig* è quello usato dall'animatore
- L'oggetto *rig_add* è il rig aggiuntivo, nascosto di default. Non deve essere modificato, ma non bisogna cancellarlo, poiché sono ossa secondarie che potrebbero tornare utili.

Il rig, lo scheletro *rig_add* e gli oggetti della UI, fanno parte della collezione "figlia" *character1_rig.* Le custom shapes usate per i controller delle ossa fanno parte della sotto-collezione *character1_cs*. Questa collezione è nascosta di default. Gli oggetti del rig sono inoltre imparentati a un empty *char_grp*, come oggetto radice della gerarchia.

L'armature consta di 6 livelli: Main controllers, secondary controllers, other picker bones, reference bones, deforming bones e un livello per usi interni.

I controller possono essere selezionati sia nella 3D View o utilizzando un'interfaccia *picker*.



È possibile cambiare delle proprietà per quanto riguarda il movimento di braccia e gambe:

- *Stretch Length* specifica la lunghezza della catena di ossa
- *Auto Stretch* abilita o disabilita (solo in modalità IK) l'allungamento automatico degli arti
- *Fix Roll* serve per aggiustare la contorsione del piede
- *IK-FK Switch* serve per passare manualmente dalla modalità IK a quella FK e viceversa

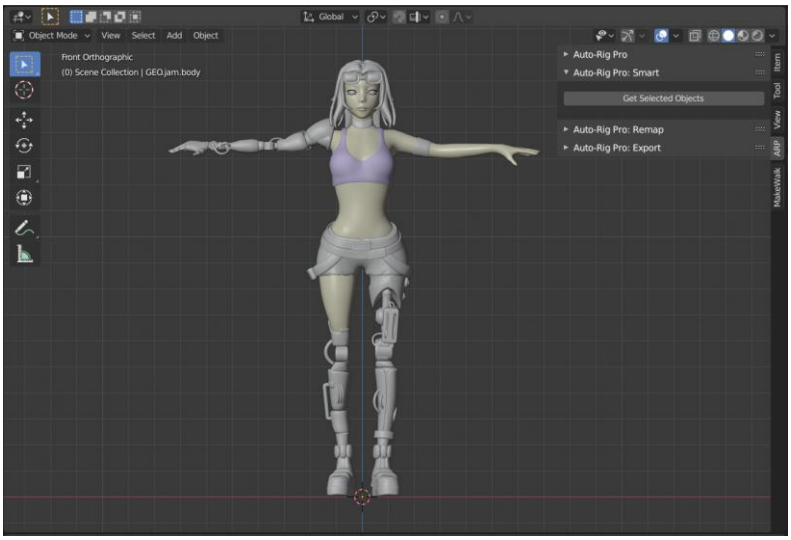
È possibile bloccare o sbloccare la rotazione della testa tramite il comando *Head Lock* e modificare l'effetto di stretch and squash delle labbra quando si muove l'osso della mandibola. Per muovere gli occhi si può usare un controller master che li muove entrambi o dei controlli figli che li muovono separatamente. Selezionando questi controller si può modificare l'influenza che ha il movimento di questi sulle palpebre.

4.4 Rigging di Jameela con Auto-Rig Pro

Essendo Jameela un personaggio secondario, viste le tempistiche di BlenRig, per lavorare parallelamente al personaggio di Jameela, si è deciso di utilizzare un rig automatico, Auto-Rig. L'esecuzione del rig non ha preso più di dieci minuti, ma l'utilizzo di questo genere di rig porta a delle imperfezioni difficili da camuffare nel caso di un personaggio principale. Nei prossimi paragrafi è illustrato l'intero processo di rig e una panoramica delle problematiche riscontrate.

4.4.1 Setup del modello

Per utilizzare la funzione *Smart,* il modello del personaggio deve trovarsi nel punto origine (0,0,0) con i piedi sul livello terra. Deve inoltre essere rivolto verso l'asse globale -y, ovvero, il volto e i piedi devono essere rivolti verso la vista frontale ortogonale, come nella seguente immagine.



Tramite i comandi *Apply>Position, Apply>Rotation&Scale* si inizializzano le trasformate del modello.

Il personaggio di Jameela è stato diviso in diverse mesh: *body, head, eyes, eyeglasses* e *hair*. La mesh *body* ha un modificatore *Mask* per i vertici della gamba sinistra in quanto nel trailer, il suo personaggio

avrà la protesi della gamba staccata dal corpo, e il modificatore Mask serve per rendere invisibili, quando necessario, i vertici su cui agisce.

4.4.2 Funzione Smart

La funzione smart è utile per posizionare velocemente le ossa di riferimento. Funziona solo per personaggi bipedi e per utilizzarla bisogna seguire le seguenti linee guida:

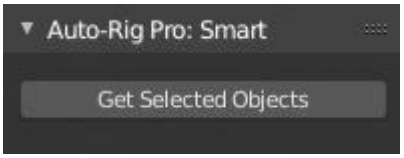
- È supportato solo un personaggio per blend file. Il tool Smart non funziona se ci sono altri rig esistenti all'interno del file.
- Il personaggio può essere in una T-Pose o in una A-Pose. Per il rilevamento delle dita ci deve essere abbastanza spazio tra le stesse. Il palmo deve essere rivolto verso il pavimento, se la mano è troppo contorta verso linterno o l'esterno, il rilevamento delle dita fallirà.

Se le dita non rispondono a questi requisiti, si può scegliere l'opzione *Skip Fingers* al di sopra del pulsante *Go.* Se il personaggio non è bipede, si possono posizionare manualmente le ossa di riferimento.

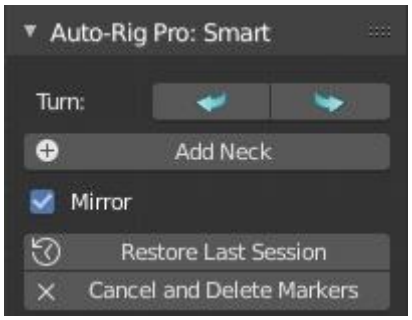
In base alle proporzioni, potrebbe essere necessario eseguire delle correzioni manuali dopo il rilevamento. Le dita potrebbero non essere rilevate correttamente se non c'è abbastanza spazio tra di loro, se sono troppo curvate, o se ci sono troppe forme complicate attorno alla mano, come per esempio dei braccialetti.

Se si vogliono impostare i marker facciali, è necessario far sì che gli occhi siano in un oggetto separato e nominato in un modo che sia facile da trovare in un secondo momento (l'oggetto può contenere una singola sfera con un modificatore Mirror oppure entrambe le sfere senza il modificatore).

Per cominciare a creare il rig automatico dunque è necessario selezionare le mesh che compongono il personaggio e premere il pulsante *Get Selected Objects* all'interno della sezione *Auto-Rig Pro: Smart* del pannello laterale *ARP.*

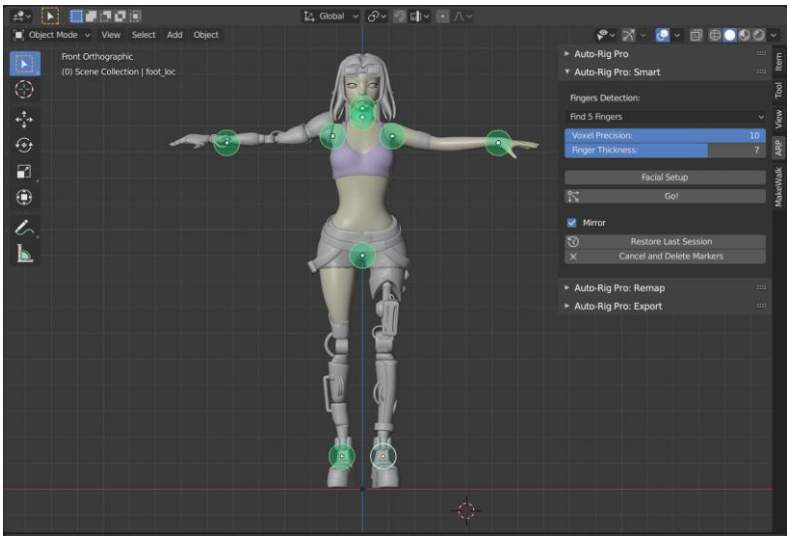


A questo punto la camera inquadrerà il personaggio nella front view. Il comando turn serve per ruotare il personaggio nel momento in cui non si trova rivolto verso la camera.

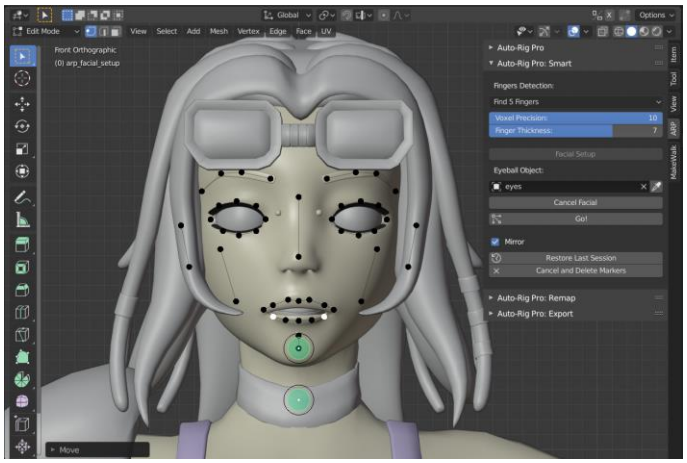


Se il personaggio non è simmetrico, bisogna deselezionare l'opzione Mirror. Nel caso di Jameela, il modello non è simmetrico, in quanto il braccio destro e le gambe sono protesi formate da oggetti diversi. Ma la funzione mirror in questo caso si è deciso di lasciarla attivata, in quanto i due lati del corpo sono simili tra loro e qualsiasi aggiustamento si sarebbe fatto in un secondo momento in Edit Mode.

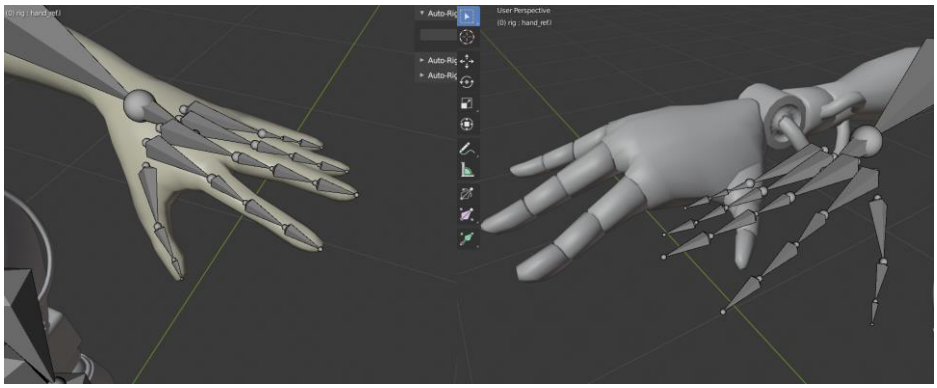
Cliccando sul pulsante *Add Neck,* questo crea un marker circolare e basta muovere il cursore per posizionarlo nel modo giusto. Non è necessario ruotare o cambiare la vista in quanto il programma riesce a trovare automaticamente la profondità del collo e anche del resto dei marker.



Una volta aggiunti tutti i marker (*Neck, Chin, Shoulders, Wrists, Spine Root, Ankles*), tramite il pulsante *Facial Setup* si può passare al setup dei marker facciali, indicando nel campo *Eyball Object* l'oggetto corrispondente agli occhi.



Dopo aver premuto il pulsante *Go!*, in pochi secondi le ossa di riferimento dovrebbero essere posizionate nel modo giusto. Ma potrebbe essere necessario dover modificare manualmente alcune ossa per un posizionamento più accurato. Come ad esempio nel caso delle mani di Jameela, Auto-Rig non è stato in grado di posizionare perfettamente le ossa della mano destra mentre non c'è stato bisogno di alcun ritocco per la mano sinistra.



Il rilevamento delle dita funziona nella maggior parte dei personaggi, ma a volte può fallire. In questo caso, come è stato fatto con Jameela nelle diverse prove di rig, si può provare a spostare leggermente il marker per il polso, poi ridurre o aumentare il valore di *Voxel Precision* (che si occupa di determinare il peso in voxel assegnato a ogni osso) e provare ad incrementare il valore di *Finger Thickness*. In ogni caso, con il modello di Jameela queste soluzioni non hanno funzionato, dunque si è proceduto con il posizionamento manuale delle sole ossa della mano sinistra.

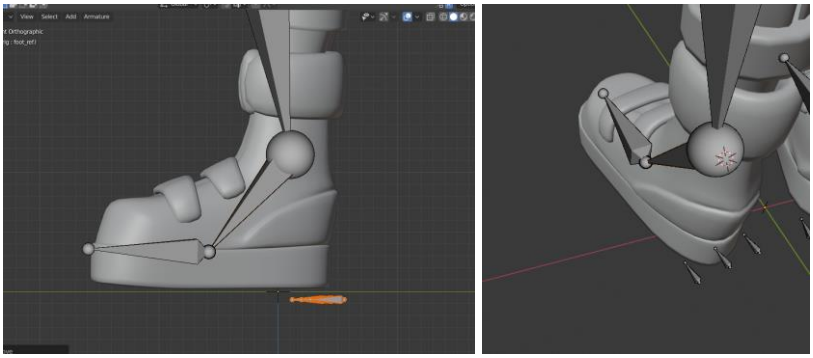
4.4.3 Configurazione del rig e posizionamento delle ossa

È il momento di definire e configurare lo scheletro. Selezionando l'oggetto *armature*, nel pannello laterale del 3D viewport nella sezione *Auto-Rig Pro>Rig* si può cliccare il pulsante *Edit Reference Bones* e modificare alcune impostazioni come ad esempio le *Limb Options,* con cui è possibile

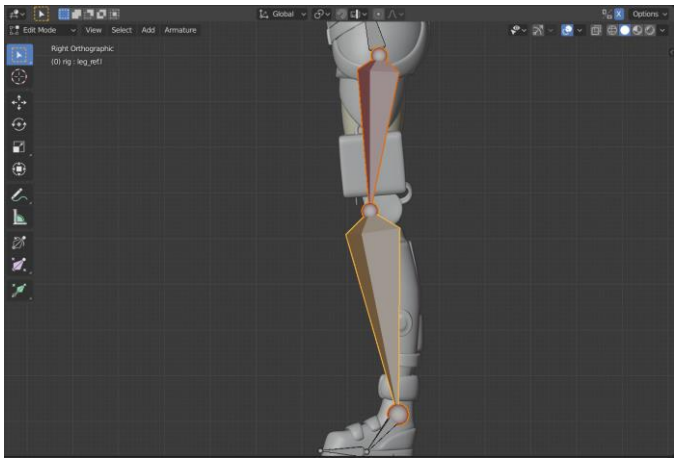
decidere il numero di parti in cui dividere un osso. Questo step, per il modello di Jameela, è stato saltato.

È necessario posizionare le ossa in modo che si adattino alle proporzioni del modello:

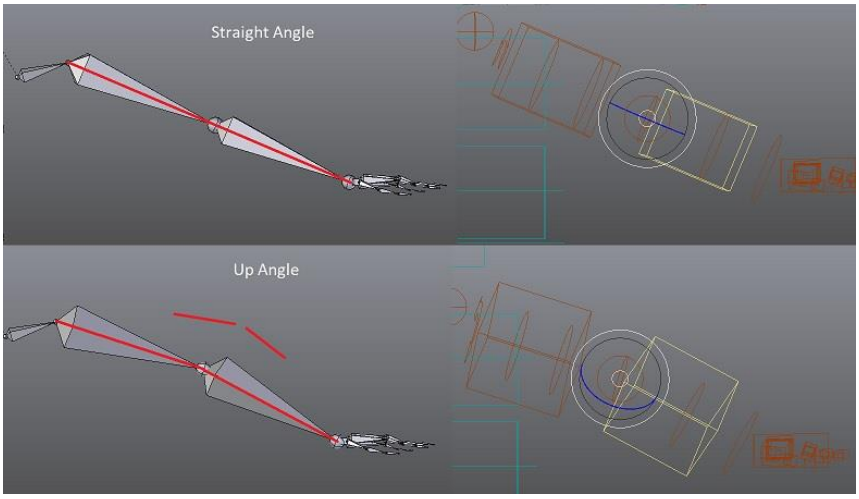
- L'osso *foot_heel* e le ossa *foot_bank* (le tre piccole ossa al di sotto del piede) dovrebbero combaciare con la posizione posteriore del tallone e alla larghezza del piede.



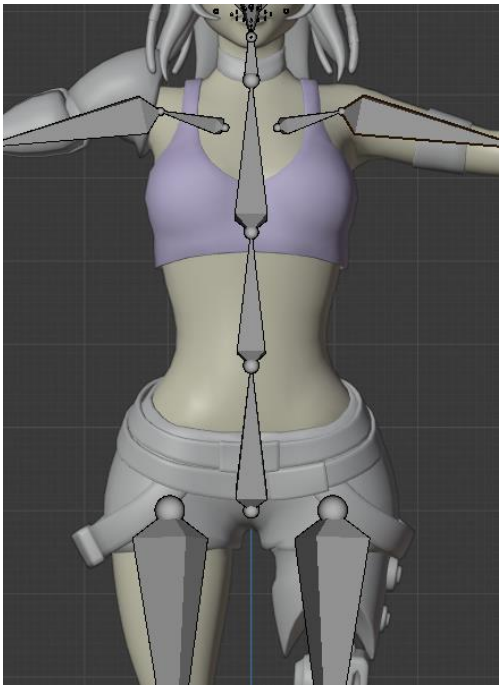
- È necessario che ci sia una leggera curvatura tra le ossa di gambe e braccia. È importante perché l'IK funzioni correttamente. Senza nessuna curvatura la direzione dell'IK potrebbe essere invertita.



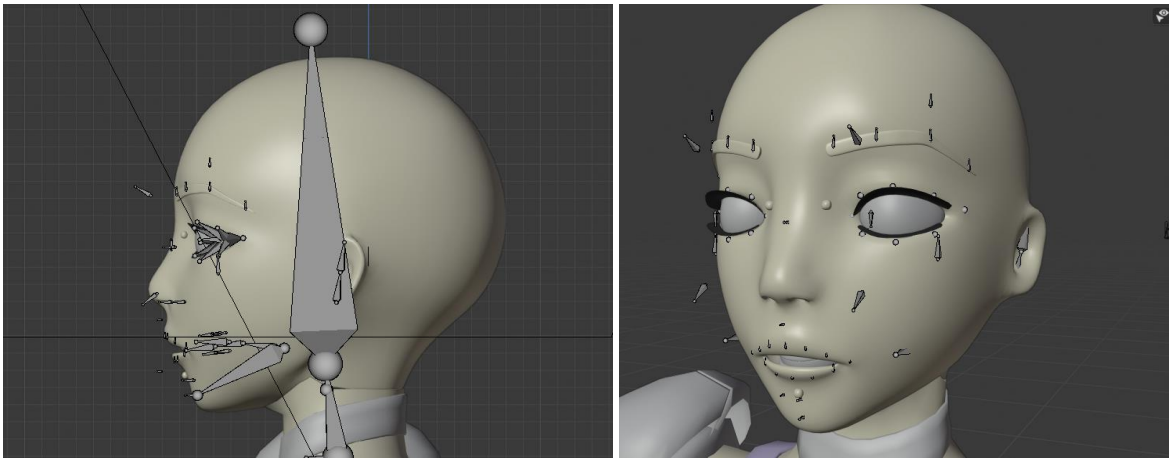
- La catena di IK non può essere aggiustata manualmente. È calcolata cliccando su *Match to Rig*. Cambiando l'allineamento delle ossa cambierebbe il roll. Nell'immagine seguente il braccio è mostrato dalla front view. Può risultare dritto da questo punto di vista ma deve comunque mantenere un'angolatura verso il dietro, visibile dalla top view.



- La punta dell'osso *roof_ref* dovrebbe essere centrata all'altezza dell'ombelico, *spine_01* alla base del petto e *spine_02* dovrebbe raggiungere la base del collo.



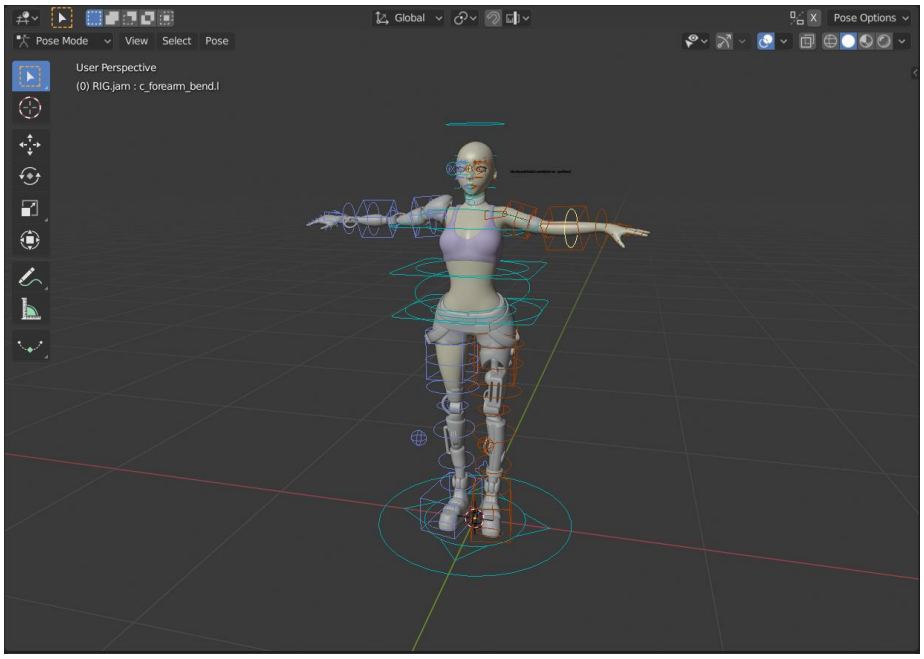
- Le ossa facciali devono essere posizionate come nell'immagine seguente, in cui è stata attivata l'opzione *x-ray*.



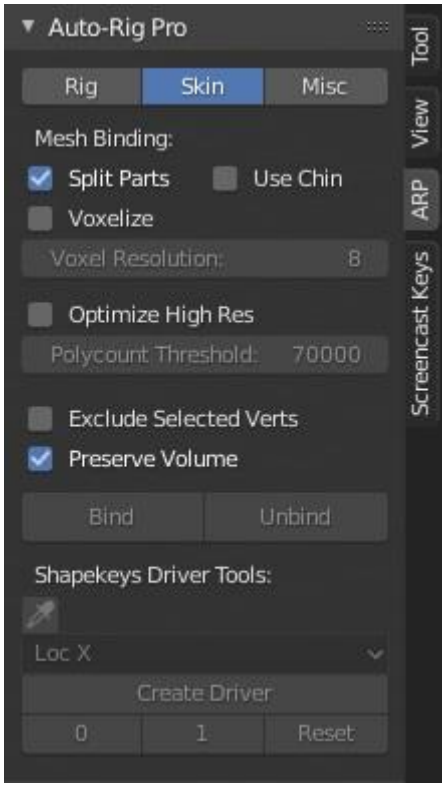
- Le ossa delle labbra devono essere molto vicine alla geometria, posizionate attorno alle labbra.
- Le teste delle ossa di ciglia e occhi devono essere posizionate al centro dell'occhio mentre le code devono raggiungere la superficie della sfera dell'occhio.

4.4.4 Generazione del rig e Skinning

Cliccando *Match to Rig* si genera il rig finale con i vari controller e tutte le impostazioni meccaniche. È possibile aggiustare le forme dei controller cliccando *Edit Shape* e *Apply Shape* una volta finite le modifiche.



Una volta generato il rig, è necessario eseguire lo *Skinning,* ovvero l'associazione automatica dei pesi. Si devono prima selezionare le mesh che compongono il personaggio, poi lo scheletro con il tasto Shift. Di seguito vengono analizzate le funzioni associate al bind dello scheletro.



- *Split Parts* viene abilitato per risultati migliori se il modello consiste di parti separate come vestiti e accessori (come nel caso di Jameela).
- *Use Chin* migliora lo skinning della testa in base alla posizione del mento (quando è disabilitato lo scheletro facciale, e solo per modelli bipedi).
- *Voxelize* fornisce migliori risultati con livelli multipli di vestiti e accessori ma è meno accurato con parti più piccole (dita, viso) rispetto ad altri metodi, quindi per il caso corrente questa opzione non è stata attivata. È comunque un metodo approssimativo non infallibile come l'add-on *Voxel Heat Diffuse Skinning* (utilizzato per la faccia di Jameela) né tanto veloce, ma produce risultati migliori rispetto ad altri metodi si skinning per la maggior parte delle volte.
- *Optimize High Res* velocizza il binding delle mesh high poly che contengono più facce rispetto alla soglia indicata dal comando.
- *Exclude Selected Verts* permette di evitare a certi vertici che si sono selezionati di essere parte del processo di binding.

Una volta scelte le opzioni più opportune si può premere il pulsante *Bind.* Per Jameela si sono create delle problematiche per cui si è dovuto procedere diversamente per il corpo e per la testa, ma questo verrà discusso successivamente all'interno di questa tesi.

Il processo di Binding può richiedere tempo, soprattutto per le mesh ad alta risoluzione.

4.4.5 Weight Paint

Una volta finito il processo di Binding, si possono fare degli aggiustamenti nella deformazione tramite weight paint manuale e il modello dovrebbe funzionare correttamente. Come già accennato, per il modello di Jameela ci sono state delle problematiche non indifferenti per cui sono state trovate soluzioni alternative. Il weight paint infatti, se fatto sul corpo, non aveva nessun effetto. Qualsiasi osso venisse "dipinto" non modificava in nessun modo la sua influenza. Questo problema è stato facilmente ignorabile in quanto il personaggio sarebbe stato seduto per tutto il tempo e dalla camera certi difetti venivano nascosti. Il problema principale è sorto sul volto. Il weight automatico talvolta prendeva parti della testa fin troppo lontane dall'area di influenza dell'osso selezionato. Muovendo le ossa della bocca si deformavano anche i denti che a volte, nonostante il weight aggiustato, si muovevano senza alcuna apparente logica. Inoltre, è spesso successo che aggiustando il weight del volto, quando veniva linkato il modello nella scena da animare, questo smetteva di funzionare, o si annullava il bind e doveva essere rifatto, e spesso durante il bind Blender smetteva di funzionare chiudendosi e facendo perdere le modifiche effettuate.

5.Remap e animazione

Come è già stato detto, per questo progetto si è deciso di utilizzare il motion capture come base per l'animazione sia del corpo che facciale, con alcuni ritocchi manuali, quando necessari, dell'output ottenuto. L'animazione di Nadya e Jameela è stata realizzata applicando le riprese di motion capture effettuate con la Smartsuit Pro, ai differenti rig creati dei modelli. Per questa operazione è stata, utilizzata la funzione *Remap* di Auto-Rig Pro che è stata creata appositamente per i rig creati con Auto-Rig, ma che il team ha riadattato per essere utilizzata anche con BlenRig. Verranno mostrati di seguito il procedimento di remap e i risultati ottenuti per le due diverse tipologie di rig, con problematiche annesse.

L'animazione e il motion capture facciale sono stati eseguiti separatamente dal corpo. Il mocap è stato realizzato tramite il tool di Rokoko che permette la cattura del movimento facciale tramite un'applicazione dell'iPhone X, e successivamente, come verrà esposto nel dettaglio in seguito, sono state eseguite delle correzioni alle espressioni gestendo i movimenti dello scheletro con keyframe, e correggendo l'output finale con le shape keys create sul personaggio. Questo procedimento è stato effettuato solo sul personaggio di Nadya, che è il personaggio principale del teaser. Per Jameela, si è scelto di utilizzare l'animazione tramite shape keys, in quanto i suoi primi piani all'interno della clip erano ridotti, ed eseguire lo stesso procedimento di Nadya avrebbe richiesto del tempo e delle fatiche inutili allo scopo di questa tesi.

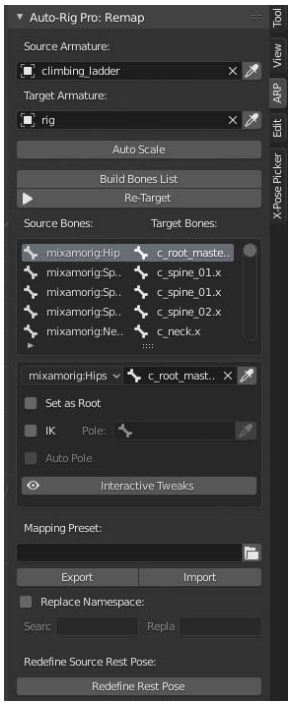
5.1 Funzionalità Remap di Auto-Rig Pro

Il *Remap* è lo strumento fornito da Auto-Rig, che permette di associare l'animazione di uno scheletro a un altro rig. Si può importare ad esempio un file di motion capture BVH e trasferire i dati dell'animazione allo scheletro di Auto-Rig Pro.

5.1.1 Workflow

Eseguire il retarget del motion capture sullo scheletro fornito da Auto-Rig Pro è un procedimento molto semplice. Basta importare il file BVH del motion capture (se si usa l'estensione FBX bisogna controllare l'orientamento delle ossa nelle impostazioni di import).

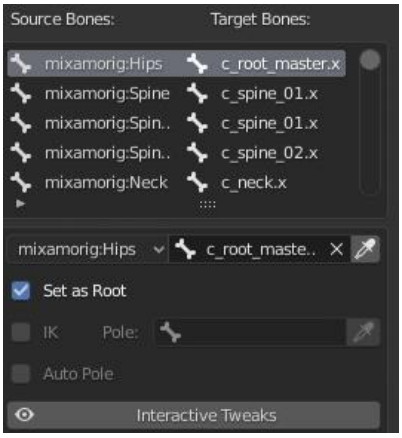
Nel menù laterale è possibile selezionare nel campo *Source Armature* lo scheletro "sorgente", ovvero il file BVH importato, e successivamente indicare lo scheletro a cui associare il motion capture, ovvero il rig del personaggio, nel campo *Target Armature.* La funzione *Auto-Scale* serve ad adattare la scala dello scheletro sorgente a quello del target, e nel caso il risultato non fosse corretto, sarebbe necessario scalarlo manualmente.



Con il comando *Build Bone List* viene eseguito in automatico il mapping delle ossa. Ovvero, a ogni osso dello scheletro sorgente viene associato un osso dello scheletro target. È possibile comunque modificare, in base alle esigenze, la lista creata automaticamente.

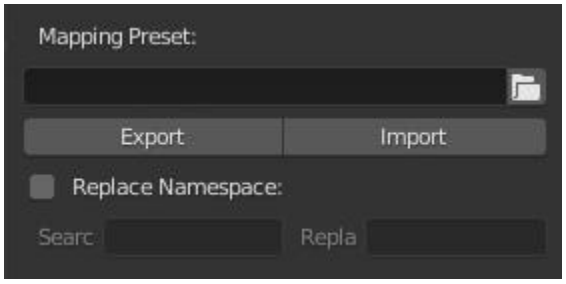
È necessario definire il *Root Bone,* ovvero l'osso radice, che tipicamente viene associato al primo o secondo osso della gerarchia (di solito i fianchi – *hips*).

Facoltativamente, si può usare la modalità IK per piedi e mani, selezionando il comando *IK* e assegnando, nella sezione *Pole,* l'osso finale della catena di inverse kinematic (lo scheletro deve essere anch'esso in modalità IK).



Se lo scheletro sorgente non ha la stessa rest pose dello scheletro target, si può ridefinire la rest pose con il comando *Redefine Rest pose*: basta selezionare le ossa che non sono posizionate come quelle del target e cliccare *Copy Selected Target Bones*, che copieranno automaticamente l'orientamento delle ossa target e successivamente cliccare su *Apply*.

In alternativa, si può esportare una mappatura delle ossa preimpostata, importandola tramite il comando *Mapping Preset*:

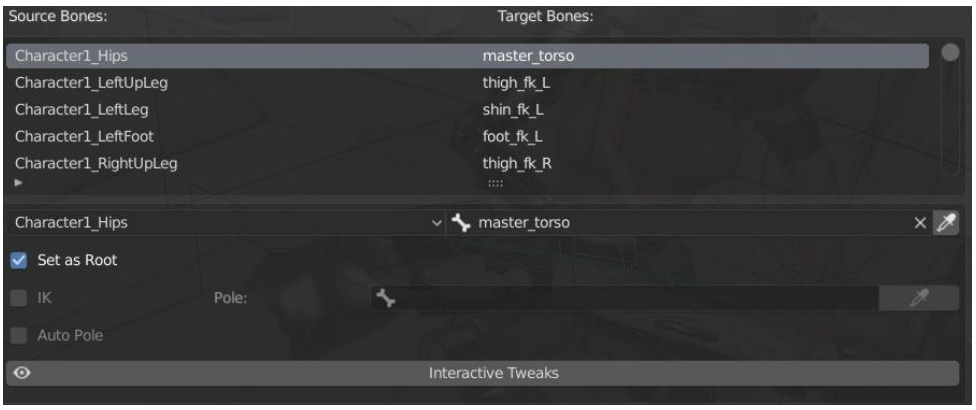


Infine, una volta eseguita la funzione *Re-Target* sarà possibile vedere che l'animazione dello scheletro proveniente dal motion capture è stata copiata correttamente sullo scheletro del personaggio.

5.1.2 Remap del corpo di Jameela

Il processo di remap di Jameela è stato molto semplice e veloce. Il motivo è semplicemente il fatto che il suo rig è stato creato automaticamente da Auto-Rig e il processo di remap è una funzione dello stesso add-on, dunque nel momento in cui il software ha dovuto fare l'Auto-Scale, il rig sorgente si è scalato nel modo corretto in base alle dimensioni del target, e quando ha dovuto creare la *bones list,* e dunque l'associazione tra le ossa del rig fornite dal mocap e le ossa del rig target, questa è stata

creata automaticamente. L'unica operazione da fare è stata impostare il *Root,* che in questo caso era l'osso nominato *master_torso.*



La funzione *Build Bones List* crea appunto la lista di ossa per il rig target (*Target Bones*), associata alle ossa del rig sorgente (*Source Bones*). Una volta premuto il tasto *Re-Target,* l'animazione del mocap viene copiata sul target.

Nel teaser, Jameela è seduta sul letto della sua stanza mentre salda la protesi metallica della sua gamba. La tuta Smartsuit utilizzata per il motion capture, non tiene conto del movimento delle mani, che sono state animate dunque tradizionalmente, tramite keyframing, muovendo osso per osso. Fortunatamente Autorig ha un sistema di IK per le mai che ha reso più semplice l'esecuzione di questo tedioso processo. Quindi nonostante la costruzione della *Bones List* è comunque possibile cambiare le ossa o il tipo di cinematica, quando necessario.

Nell'immagine è possibile vedere il retarget di Jameela. Il corpo segue perfettamente il movimento del rig sorgente (quello a destra), ma si può osservare che il braccio sinistro del personaggio è posto più in alto rispetto allo scheletro sorgente. Questo perché nella scena finale è presente un ulteriore

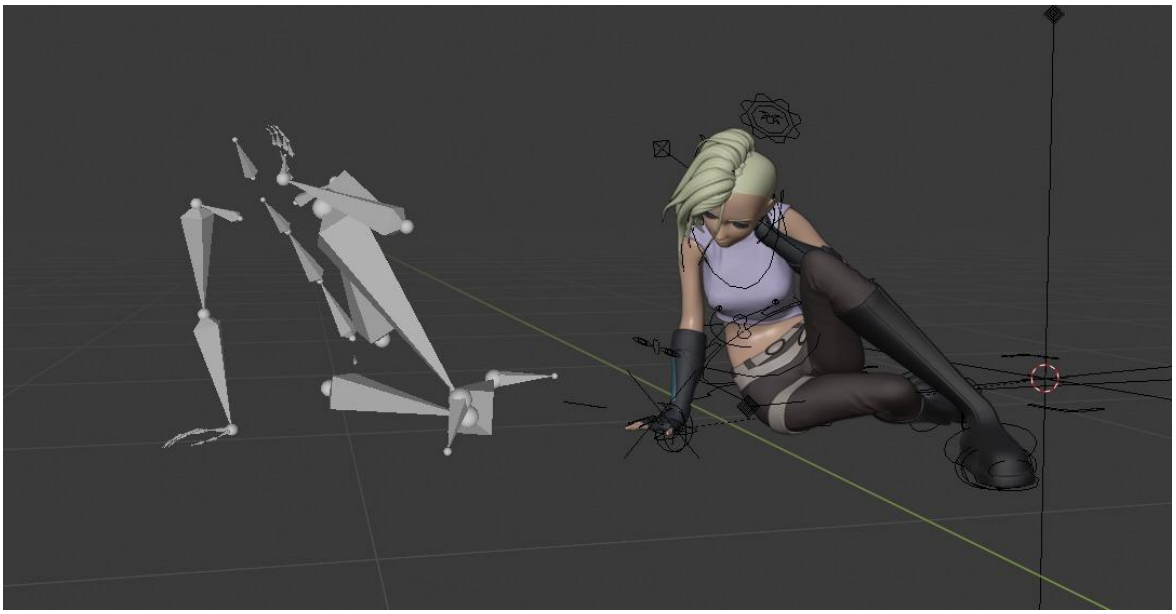
oggetto di scena (la protesi della gamba di Jameela) che in fase di ripresa non era presente. Può anche succedere che i dati provenienti dal mocap, nonostante siano stati puliti, risultino ancora rumorosi. È dunque necessario fare delle modifiche al movimento tramite lo spostamento dei controller delle ossa. Inoltre, per le mani e per i piedi non c'è un sistema di mocap annesso, è dunque necessario animarli separatamente tramite il rig.

5.1.3 Remap del corpo di Nadya

Il processo di remap di Nadya non è stato così semplice come per quello di Jameela. Dopo aver importato il file BVH, la funzione Auto-Scale non ha funzionato correttamente, e il rig sorgente è stato scalato manualmente. Dopo di che, una volta premuto il pulsante *Build Bones List,* ovviamente le ossa di BlenRig non sono state riconosciute dal software. Si è proceduto quindi con l'inserimento manuale delle ossa. Diverse prove sono servite per trovare la combinazione di ossa corrette, perché il corpo seguisse il movimento del rig sorgente nel modo più esatto possibile. Inizialmente si è scelto di lavorare in IK, inserendo dunque soltanto le ossa che avevano una catena di ossa associata, quindi l'osso del piede *sole_ctrl_L(R)* e quello della mano *hand_ik_ctrl_L(R),* selezionando il comando *IK* e indicando le relative ossa concludenti la catena (*thigh_ik_L(R)* per la gamba e *arm_ik_L(R)* per il braccio) nel campo *Pole*.

Una volta fatto il retarget, il corpo funzionava abbastanza bene, ma provando invece l'inserimento delle ossa in FK, si è potuto notare che si ottenevano una migliore deformazione e un miglior movimento del corpo.

Scelta la tipologia di mapping, è stato possibile esportare la combinazione delle ossa in un file bmap, in modo che, quando necessario, non si sarebbe dovuto rifare l'inserimento manuale, ma sarebbe bastato importare il file tramite il comando *Mapping Preset.*



Eseguire il retarget in FK ha significato che, per ogni frame c'erano, per tutte le ossa in FK, dei keyframe (provenienti dal rig sorgente). Per poter modificare in fase di animazione il movimento del corpo molto più facilmente, sarebbe stato utile utilizzare il rig in modalità IK. Lo strumento *switch IK/FK* offerto da BlenRig permette di fare, per ogni posa, lo snap, per la posa stessa, da FK a IK. Facendo però questo slittamento tra le due cinematiche, BlenRig non tiene conto dei keyframe, dunque ogni modifica fatta con questo strumento sarebbe stata persa. Per questo si è deciso di creare uno script che permettesse che il plug-in, frame per frame, per l'intervallo di frame necessario, facesse lo switch da FK a IK, salvasse per tutte le ossa in IK il keyframe e passasse al frame successivo. È possibile visualizzare questo script alla fine di questo documento, nella sezione Appendice (Script 1).

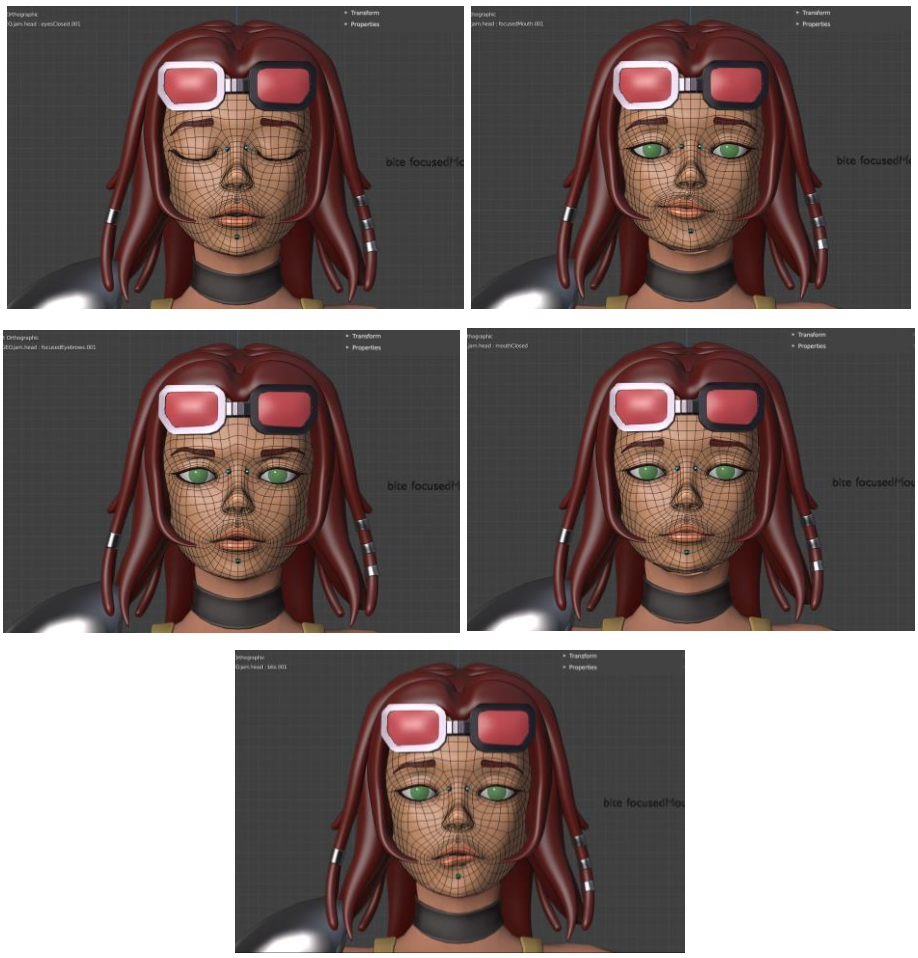
5.2 Mocap e animazione facciale

Come già anticipato, l'animazione facciale per i due personaggi è stata eseguita con metodi diversi. Vista la quantità di primi piani dedicati a Nadya, oltre al fatto di essere la protagonista principale del teaser, si è scelto per lei di utilizzare lo strumento fornito da Rokoko per il mocap facciale come base per l'animazione, con conseguente modifica del movimento tramite il rig facciale.

5.3 L'animazione di Jameela

Un altro discorso si può fare per Jameela che, all'interno del teaser, ha soltanto un primo piano e un'espressione neutra per quasi tutta la durata della sua scena. Il processo di mocap ha richiesto un lungo procedimento di creazione di shape keys sul modello di Nadya, procedimento che sarebbe stato quasi inutile nel caso di Jameela, viste anche le problematiche esplicate nel capitolo precedente

riguardo il rig facciale di Auto-Rig. Si è dunque deciso, per questioni di tempo, di creare in Jameela cinque shape key, per ogni espressione necessaria al suo personaggio.



Le shape keys sono state collegate al movimento di alcune ossa, create appositamente, tramite driver. Dunque, l'animazione del volto è stata creata semplicemente tramite keyframe sulle ossa che controllano i driver e tramite la combinazione di più shape keys.

5.4 L'animazione di Nadya

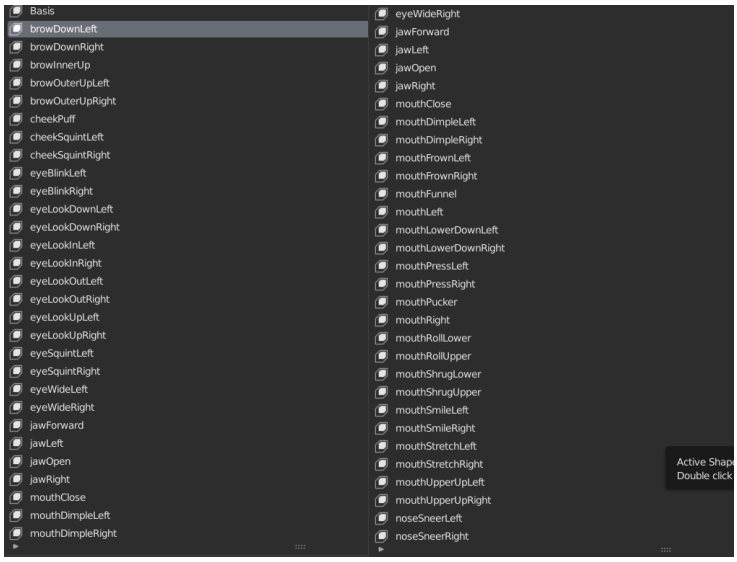
Come già spiegato nel capitolo 3, il mocap facciale di Rokoko funziona analizzando il volto dell'attore, facendo il tracking del movimento e attivando le cinquantadue shape keys corrispondenti al movimento effettuato, che sono poi attivate e miscelate per ricreare la giusta espressione.

5.4.1 Setup del personaggio

È importante impostare il volto del personaggio nel modo corretto. Una buona topologia del volto è fondamentale per ottenere la corretta deformazione ed è consigliato mantenere i poligoni della mesh

al di sotto dei cinquanta mila (il volto di Nadya ha circa quindici mila poligoni). Si può scegliere se unire gli oggetti del volto (occhi, sopracciglia, denti) o tenerli separati. Nel caso di Nadya, gli occhi sono separati dal modello, mentre i denti e le sopracciglia fanno parte della mesh del volto.

Il prossimo step è quello di creare le blend shapes per il personaggio. Si possono creare manualmente e poi testarne la funzionalità provando a combinarle nella stessa espressione, oppure si può utilizzare un tool della piattaforma Polywink, che genera automaticamente le shape keys di qualsiasi personaggio. La seconda opzione avrebbe risparmiato un po' di tempo e di processi *tryal&error* nella creazione di shape keys funzionanti, ma ha anche un costo molto elevato (circa 500 euro già scontati per i clienti Apple). Dunque, si è deciso di intraprendere la prima strada, quella della creazione manuale di ben cinquantadue shape keys. La lista di shape keys create è la seguente:

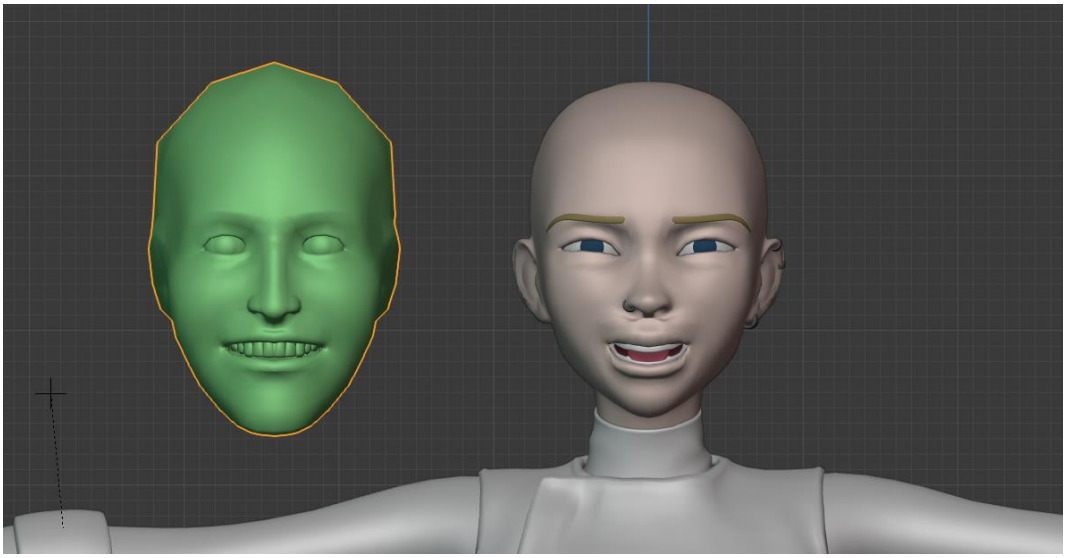


Il processo di creazione di una shape keys è già lungo di suo. Si può creare tramite lo strumento *sculpt*, tramite spostamento di vertici o tramite il rig facciale. Grazie alla buona resa del rig fornito da Blenrig e dall'esecuzione di un buon weight paint, è stata scelta l'ultima opzione. Per creare una shape key i passaggi sono molteplici: bisogna posizionare le ossa interessate in pose mode, selezionare in modalità object la mesh della testa, aggiungere un modificatore *Armature* selezionando il rig di BlenRig e selezionare la funzione *Apply as a shape key*. Dunque, per velocizzare i tempi di lavoro, si è pensato a uno script che automatizzasse il processo, in modo da non ripetere per ben cinquantadue volte i passaggi per creare una nuova shape key. Lo script è visualizzabile alla fine di questo documento nella sezione Appendice (Script 2).

Le shape keys si devono creare anche per la mesh degli occhi (anche se non tutte le cinquantadue ma solo quelle relative al movimento degli stessi), e per quella delle ciglia. In un primo momento sono state quindi create altre shape keys per le ciglia, ma successivamente si è deciso di deformare le ciglia

tramite un modificatore *Surface Deform* applicato alla testa, che ha permesso una discreta deformazione e minore lavoro per il mocap facciale.

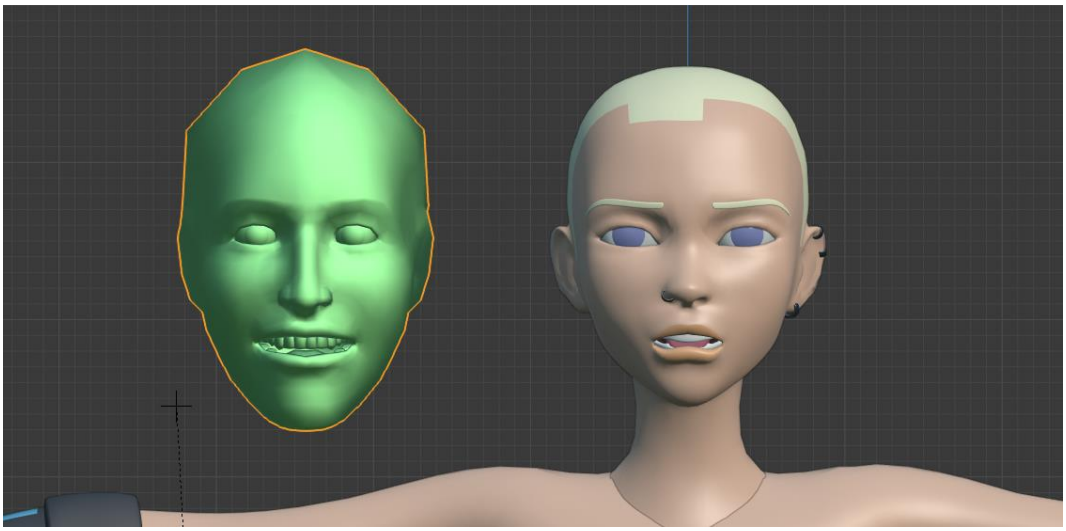
Una volta create le shape keys, si poteva notare una grossa interferenza di vertici nel momento in cui le shape keys venivano create. È possibile vedere nell'immagine successiva il risultato di questa interferenza:



Nessuna parte del volto sembra avvicinarsi all'espressione della take di mocap (la mesh verde) e soprattutto nella parte tra naso e bocca c'è un'alterazione della mesh chiaramente non gradevole alla vista.

Per risolvere il problema, ogni shape key è stata ripulita da ogni interferenza modificandola in Edit Mode e utilizzando il comando *Blend from Shape* (con l'opzione *Add* deselezionata) per rimuovere i vertici coinvolti nella shape involontariamente. In parole semplici, alcuni vertici della bocca si muovevano nonostante la shape keys fosse interessata solo al movimento di un occhio. Il comando Blend from Shape fa sì che questi vertici selezionati rimanessero fermi quando attivata la blend shape.

Dopo questo lungo lavoro di ripulitura, per cui è stato anche necessario ricreare alcune shape keys, il risultato è stato il seguente.



5.4.2 Applicazione del movimento

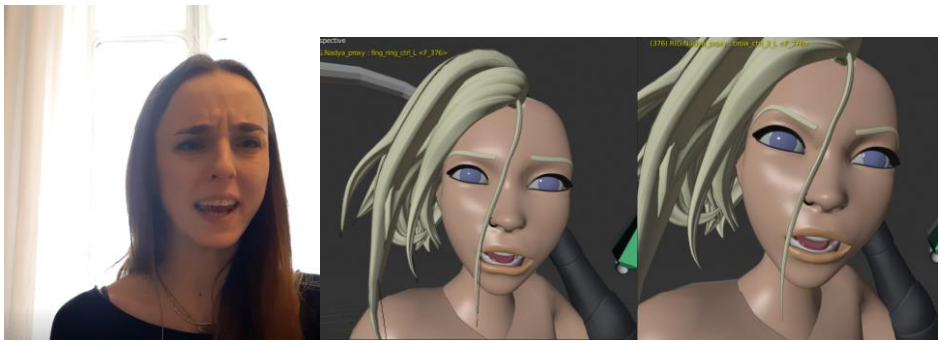
Una volta impostato il personaggio nel modo corretto, il processo di applicazione del mocap non richiede molto tempo, anzi è un'operazione abbastanza semplice. I passaggi sono i seguenti:

- Si importa sulla scena il file del motion capture, nel nostro caso un FBX. Questo si presenta come una mesh con cinquantadue shape keys (le stesse create per il modello target) e con dei key frame associate a esse.
- Si imposta il frame corrente a zero e si inizializzano le shape keys mettendo un key frame per ognuna di esse, sia per la mesh del volto che per la mesh degli occhi.
- Si selezionano tutti i keyframe della mesh del mocap sulla time table del volto e degli occhi.

Una volta fatti questi tre passaggi il volto è animato e segue perfettamente i movimenti della mesh sorgente.

5.4.3 Applicazione del movimento

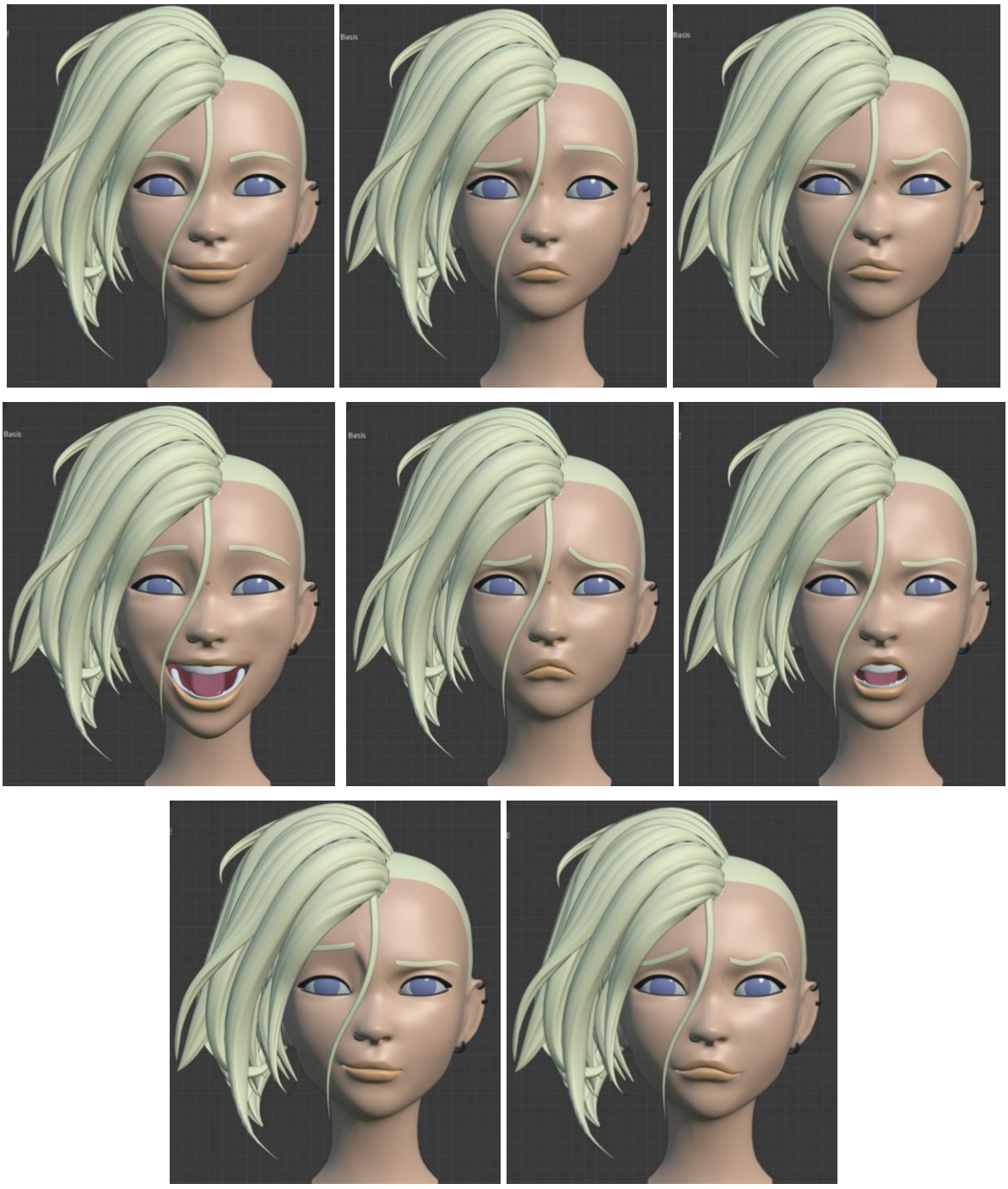
Purtroppo, con la sola animazione del motion capture, non si è ottenuta l'espressività sperata per il personaggio. Si è dunque proceduto con la modifica manuale delle ossa del volto in base anche alle espressioni fatte dall'attrice ripresa per il mocap, che era stata ripresa in contemporanea con un'altra camera. La seguente immagini è composta da tre parti. Nella prima si può vedere il video dell'attrice, in particolare nel momento in cui pronuncia la vocale "e", nella seconda il mocap facciale applicato sul modello e nella terza il risultato dell'aggiustamento di bocca e sopracciglia.



5.4.4 Shape keys aggiuntive

Oltre alle cinquantadue shape keys create per il mocap, si è deciso di creare una serie di shape keys con le espressioni principali del personaggio, in base a un disegno/reference fornito dal concept artist del progetto, Edoardo Audino.

Le prossime immagini mostrano rispettivamente lo sheet di espressioni fornite dal disegnatore e quelle effettivamente realizzate sul modello 3D. Come è possibile osservare, le espressioni sono esagerate e abbastanza irrealistiche in quanto si sta realizzando un cartone, e avere delle espressioni di questo tipo è una scelta quasi obbligata sia dal punto di vista estetico che da quello comunicativo.


Basis
Basis
Basis
Basis
Basis

6. Conclusioni

Dopo avere concluso il lavoro sul rigging, ciò che principalmente differenzia i due processi, quello semi customizzato di BlenRig e quello automatico di Auto-Rig Pro, è la tempistica di lavorazione. Non tenendo conto del tempo iniziale in cui ho dovuto imparare a utilizzare BlenRig nel modo corretto e trovare il numero adatto di poligoni della mesh per avere una buona risoluzione, questo procedimento ha richiesto molti giorni di lavoro e di processi trial&error, aggiungendo il fatto che nella mia esperienza nella computer animation non avevo mai realizzato un rig di così alto livello (si ricorda che BlenRig permette di avere un rig cinematografico) e con metodi non automatici. D'altro canto, realizzare un rig con Auto-Rig Pro non richiede più di un'ora di tempo, incluse le tempistiche di bind e di eventuale aggiustamento delle ossa. Dunque, perché utilizzare metodi di rigging come Blenrig piuttosto che metodi automatici come Auto-Rig?

Semplicemente, ogni workflow che si decide di intraprendere avrà i suoi pro e i suoi contro. Con i metodi automatici è possibile fare il rig di un personaggio in un tempo limitato, ma potrebbero verificarsi numerosi eventi impredicibili perché, per esempio, non si può fare affidamento ai pesi automatici al cento per cento, in quanto non è possibile prevedere come il personaggio si deformerà finché non si muovono tutte le parti del corpo. A volte poi i rig automatici sono funzionalità non perfette.

Nonostante i metodi automatici siano allettanti, richiederebbero comunque molte ulteriori correzioni da parte del rigger, e questi errori, che a un certo punto si vedranno, sono impredicibili. Ci si potrebbe accorgere di essi al momento dell'animazione o, ancor peggio, al momento del render. E questo è sicuramente un evento indesiderabile in una pipeline di produzione.

È dunque per questo più consigliabile avere un controllo totale sul processo di sviluppo del personaggio e agire manualmente sulle deformazioni della mesh deform cage ed eseguire manualmente il weight paint delle mani e della faccia del personaggio. Si ha così un controllo completo su come il personaggio si deforma. Così facendo si potrebbe spendere molto più tempo lavorando sulla deformazione, ma si avrà la certezza che il personaggio si deformerà in maniera corretta e che non si verrà incontro a errori impredicibili nel momento del render.

Si potrebbe fare la scelta di utilizzare metodi automatici per personaggi secondari, come effettivamente è stato fatto con Jameela, unico altro personaggio del nostro trailer, e usare BlenRig per i personaggi principali, nel nostro caso solo Nadya.

In questo specifico caso però, mentre nel caso di Nadya, una volta finito il rig, questo non è stato più toccato, se non per qualche ritocco di weight paint, il caso di Jameela ha richiesto moltissimo tempo nei ritocchi e nelle problematiche conseguenti questi aggiustamenti. È successo spesso infatti che dopo aver linkato il personaggio a una scena e applicato la take di mocap, nel momento in cui ci si accorgeva di un errore e si modificava il file originale, quando si ritornava alla scena principale ci si ritrovava con il rig non più funzionante, o il remap completamente cambiato. Il suo personaggio quindi ha richiesto moltissimo tempo e fatica per risolvere i problemi che si creavano ogni volta, mentre con il rig di Nadya non c'è mai stato nessun tipo di problema. È anche per questo che si è deciso di non eseguire il mocap facciale di Jameela. avrebbe richiesto troppi aggiustamenti e c'era il rischio che spuntassero nuovi problemi impredicibili.

Un'altra soluzione che, con il senno di poi, sarebbe stata più efficiente, era quella di riadattare lo scheletro di Nadya al corpo di Jameela ed eseguire il weight della faccia e delle mani in modo automatico.

Scegliere BlenRig, lavorare trenta ore, se non di più, su un personaggio e agire manualmente su tutti i dettagli, può sembrare una scelta folle ma l'idea è di poter fare tutto questo lavoro una volta sola, riutilizzarlo per altri personaggi ed essere sicuri di non trovare cattive sorprese nei momenti peggiori, come quello del remap, del mocap facciale, dell'animazione o peggio, del render.

Appendice

Script *IK/FK Switch*

```python
import bpy

START_FRAME = 70
END_FRAME = 140
STEP = 10
T_POSE_FRAME = 10

head_ik = bpy.data.armatures['biped_blenrig'].bones["head_ik_ctrl"]
neck_2_ik = bpy.data.armatures['biped_blenrig'].bones["neck_2_ik_ctrl"]
neck_3_ik = bpy.data.armatures['biped_blenrig'].bones["neck_3_ik_ctrl"]

hand_ik_R = bpy.data.armatures['biped_blenrig'].bones["hand_ik_ctrl_R"]
elbow_R = bpy.data.armatures['biped_blenrig'].bones["elbow_pole_R"]
hand_pivot_R = bpy.data.armatures['biped_blenrig'].bones["hand_ik_pivot_point_R"]

hand_ik_L = bpy.data.armatures['biped_blenrig'].bones["hand_ik_ctrl_L"]
elbow_L = bpy.data.armatures['biped_blenrig'].bones["elbow_pole_L"]
hand_pivot_L = bpy.data.armatures['biped_blenrig'].bones["hand_ik_pivot_point_L"]

torso_ik = bpy.data.armatures['biped_blenrig'].bones["torso_ik_ctrl"]
spine_1_ik = bpy.data.armatures['biped_blenrig'].bones["spine_1_ik_ctrl"]
spine_2_ik = bpy.data.armatures['biped_blenrig'].bones["spine_2_ik_ctrl"]
spine_3_ik = bpy.data.armatures['biped_blenrig'].bones["spine_3_ik_ctrl"]

sole_R = bpy.data.armatures['biped_blenrig'].bones["sole_ctrl_R"]
toes_mid_R = bpy.data.armatures['biped_blenrig'].bones["toes_ik_ctrl_mid_R"]
toes_R = bpy.data.armatures['biped_blenrig'].bones["toes_ik_ctrl_R"]
foot_ik_R = bpy.data.armatures['biped_blenrig'].bones["foot_ik_ctrl_R"]
knee_R = bpy.data.armatures['biped_blenrig'].bones["knee_pole_R"]

sole_L = bpy.data.armatures['biped_blenrig'].bones["sole_ctrl_L"]
toes_mid_L = bpy.data.armatures['biped_blenrig'].bones["toes_ik_ctrl_mid_L"]
toes_L = bpy.data.armatures['biped_blenrig'].bones["toes_ik_ctrl_L"]
foot_ik_L = bpy.data.armatures['biped_blenrig'].bones["foot_ik_ctrl_L"]
knee_L = bpy.data.armatures['biped_blenrig'].bones["knee_pole_L"]


for frame in range (START_FRAME, END_FRAME, STEP):
    # Set current frame
    bpy.context.scene.frame_set(frame)

    # HEAD
    # Snap current bone
    bpy.ops.head_snap.fk_ik()
    # Select IK desired bones
    bpy.data.armatures["biped_blenrig"].bones.active = neck_2_ik
    # Set loc rot keyframe
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["neck_2_ik_ctrl"].keyframe_insert("location")
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["neck_2_ik_ctrl"].keyframe_insert("rotation_euler")
    # Select IK desired bones
    bpy.data.armatures["biped_blenrig"].bones.active = neck_3_ik
    # Set loc rot keyframe
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["neck_3_ik_ctrl"].keyframe_insert("location")
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["neck_3_ik_ctrl"].keyframe_insert("rotation_euler")

    # ARM_R
    # Snap current bone
    bpy.ops.arm_r_snap.fk_ik()
    # Select IK desired bones
    bpy.data.armatures["biped_blenrig"].bones.active = hand_ik_R
    # Set loc rot keyframe
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["hand_ik_ctrl_R"].keyframe_insert("location")
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["hand_ik_ctrl_R"].keyframe_insert("rotation_euler")
    # Select IK desired bones
    bpy.data.armatures["biped_blenrig"].bones.active = elbow_R
    # Set loc rot keyframe
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["elbow_pole_R"].keyframe_insert("location")
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["elbow_pole_R"].keyframe_insert("rotation_euler")
    # Select IK desired bones
    bpy.data.armatures["biped_blenrig"].bones.active = hand_pivot_R
    # Set loc rot keyframe
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["hand_ik_pivot_point_R"].keyframe_insert("location")
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["hand_ik_pivot_point_R"].keyframe_insert("rotation_euler"

    # ARM_L
    # Snap current bone
    bpy.ops.arm_l_snap.fk_ik()
    # Select IK desired bones
    bpy.data.armatures["biped_blenrig"].bones.active = hand_ik_L
    # Set loc rot keyframe
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["hand_ik_ctrl_L"].keyframe_insert("location")
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["hand_ik_ctrl_L"].keyframe_insert("rotation_euler")
    # Select IK desired bones
    bpy.data.armatures["biped_blenrig"].bones.active = elbow_L
    # Set loc rot keyframe
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["elbow_pole_L"].keyframe_insert("location")
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["elbow_pole_L"].keyframe_insert("rotation_euler")
    # Select IK desired bones
    bpy.data.armatures["biped_blenrig"].bones.active = hand_pivot_L
    # Set loc rot keyframe
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["hand_ik_pivot_point_L"].keyframe_insert("location")
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["hand_ik_pivot_point_L"].keyframe_insert("rotation_euler"

```

```
    # TORSO
    # Snap current bone
    bpy.ops.torso_snap.fk_ik()
    # Select IK desired bones
    bpy.data.armatures["biped_blenrig"].bones.active = torso_ik
    # Set loc rot keyframe
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["torso_ik_ctrl"].keyframe_insert("location")
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["torso_ik_ctrl"].keyframe_insert("rotation_euler")
    # Select IK desired bones
    bpy.data.armatures["biped_blenrig"].bones.active = spine_1_ik
    # Set loc rot keyframe
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["spine_1_ik_ctrl"].keyframe_insert("location")
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["spine_1_ik_ctrl"].keyframe_insert("rotation_euler")
    # Select IK desired bones
    bpy.data.armatures["biped_blenrig"].bones.active = spine_2_ik
    # Set loc rot keyframe
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["spine_2_ik_ctrl"].keyframe_insert("location")
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["spine_2_ik_ctrl"].keyframe_insert("rotation_euler")
    # Select IK desired bones
    bpy.data.armatures["biped_blenrig"].bones.active = spine_3_ik
    # Set loc rot keyframe
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["spine_3_ik_ctrl"].keyframe_insert("location")
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["spine_3_ik_ctrl"].keyframe_insert("rotation_euler")

    # LEG_R
    # Snap current bone
    bpy.ops.leg_r_snap.fk_ik()
    # Select IK desired bones
    bpy.data.armatures["biped_blenrig"].bones.active = sole_R
    # Set loc rot keyframe
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["sole_ctrl_R"].keyframe_insert("location")
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["sole_ctrl_R"].keyframe_insert("rotation_euler")
    # Select IK desired bones
    bpy.data.armatures["biped_blenrig"].bones.active = toes_R
    # Set loc rot keyframe
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["toes_ik_ctrl_R"].keyframe_insert("location")
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["toes_ik_ctrl_R"].keyframe_insert("rotation_euler")
    # Select IK desired bones
    bpy.data.armatures["biped_blenrig"].bones.active = toes_mid_R
    # Set loc rot keyframe
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["toes_ik_ctrl_mid_R"].keyframe_insert("location")
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["toes_ik_ctrl_mid_R"].keyframe_insert("rotation_euler")
    # Select IK desired bones
    bpy.data.armatures["biped_blenrig"].bones.active = foot_ik_R
    # Set loc rot keyframe
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["foot_ik_ctrl_R"].keyframe_insert("location")
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["foot_ik_ctrl_R"].keyframe_insert("rotation_euler")
    # Select IK desired bones
    bpy.data.armatures["biped_blenrig"].bones.active = knee_R
    # Set loc rot keyframe
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["knee_pole_R"].keyframe_insert("location")
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["knee_pole_R"].keyframe_insert("rotation_euler")

    # LEG_L
    # Snap current bone
    bpy.ops.leg_l_snap.fk_ik()
    # Select IK desired bones
    bpy.data.armatures["biped_blenrig"].bones.active = sole_L
    # Set loc rot keyframe
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["sole_ctrl_L"].keyframe_insert("location")
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["sole_ctrl_L"].keyframe_insert("rotation_euler")
    # Select IK desired bones
    bpy.data.armatures["biped_blenrig"].bones.active = toes_L
    # Set loc rot keyframe
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["toes_ik_ctrl_L"].keyframe_insert("location")
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["toes_ik_ctrl_L"].keyframe_insert("rotation_euler")
    # Select IK desired bones
    bpy.data.armatures["biped_blenrig"].bones.active = toes_mid_L
    # Set loc rot keyframe
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["toes_ik_ctrl_mid_L"].keyframe_insert("location")
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["toes_ik_ctrl_mid_L"].keyframe_insert("rotation_euler")
    # Select IK desired bones
    bpy.data.armatures["biped_blenrig"].bones.active = foot_ik_L
    # Set loc rot keyframe
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["foot_ik_ctrl_L"].keyframe_insert("location")
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["foot_ik_ctrl_L"].keyframe_insert("rotation_euler")
    # Select IK desired bones
    bpy.data.armatures["biped_blenrig"].bones.active = knee_L
    # Set loc rot keyframe
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["knee_pole_L"].keyframe_insert("location")
    bpy.data.objects["RIG.Nadya_proxy"].pose.bones["knee_pole_L"].keyframe_insert("rotation_euler")

```

Script *Shape Key Creator*

```python
import bpy


bpy.context.view_layer.objects.active = bpy.data.objects["GEO.nadya.head"]
bpy.ops.object.modifier_add(type='ARMATURE')
bpy.context.object.modifiers["Armature.001"].object = bpy.data.objects["RIG.nadya"]
bpy.context.object.modifiers["Armature.001"].vertex_group = "no_mdef"
bpy.ops.object.modifier_apply(apply_as='SHAPE', modifier="Armature.001")
```

Bibliografia

Bibliografia classica

[1] Stewart Jones (2012). “Digital Creature Rigging: The Art and Science of CG Creature Setup in 3ds Max”.

[2] Kitagawa, M., & Windsor, B. (2012). “MoCap for artists: workflow and techniques for motion capture”.

Sitografia

[1] https://conceptartempire.com/what-is-rigging/

[2] https://medium.com/unity3danimation/overview-of-inverse-kinematics-9769a43ba956

[3] https://en.wikipedia.org/wiki/Morph_target_animation

[4] https://it.wikipedia.org/wiki/Blender_(programma)

[5] https://docs.blender.org/manual/en/latest/animation/armatures/introduction.html

[6] https://www.rokoko.com/en/explore/blog/face-capture-iphonex

[7] https://cloud.blender.org/p/blenrig/blog/blenrig-public-release

[8] http://www.lucky3d.fr/auto-rig-pro/doc/auto_rig.html#conclusion

[9] http://lucky3d.fr/auto-rig-pro/doc/rig_behaviour_doc.html

[10] http://lucky3d.fr/auto-rig-pro/doc/remap_doc.html?highlight=remap

[11] https://www.rokoko.com/en/explore/blog/face-capture-studio