Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE

Corso di Laurea in Informatica

Analisi di una pipeline di animazione per la computer animation

Tesi di Laurea in Informatica

Relatore:
Chiar.mo Prof.
Giulio Casciola

Presentata da:
Francesco Orrù

Sessione III
Anno Accademico 2011/1012

alla mia famiglia, per avere investito
sul mio futuro e la mia crescita
ai miei amici, per tutte le esperienze
passate, presenti e future ...

Introduzione

Oggi la *computer graphics*, ovvero la disciplina che studia le tecniche e gli algoritmi per la rappresentazione visuale di informazioni numeriche prodotte da un elaboratore, trova applicazione nei più svariati campi di lavoro: *computer aided design* (CAD), visualizzazione scientifica, industria dell' intrattenimento, realtà virtuale, solo per citarne alcuni. In particolare negli ultimi anni l' industria dell' intrattenimento ha contribuito a investire risorse sempre più consistenti nella computer grafica, dalla realizzazione di cartoni animati, a spot pubblicitari 3D, cortometraggi o interi film interamente sviluppati con i più moderni strumenti di modellazione e animazione, per non parlare infine dei videogames. Sempre più frequentemente i risultati scientifici innovativi arrivano proprio da ricercatori operanti nei laboratori della Pixar, Disney o dell' Industrial Light & Magic di George Lucas, solo per citare alcuni colossi leader nel mercato attuale. Ma a proposito di mercato del lavoro, vediamo quali sono le aree tematiche principali nonchè le figure che si possono poi distinguere all' interno di un' azienda che sviluppa in generale applicazioni 3D:

1-Modellazione : della scena da rappresentare. Può avvenire in maniera diretta (scanner laser 3D ad esempio) o interattiva, utilizzando pacchetti di sviluppo come Maya, 3ds max, Blender, ZBrush.

2-Animazione : a partire da una serie di modelli che derivano dal processo descritto sopra, si va a "dare vita" e dinamicità alla scena sviluppata. Le componenti che interagiscono in questo processo sono diverse: dalle

sorgenti di illuminazione (statiche o dinamiche), al movimento relativo dei singoli componenti, fino al percorso della camera virtuale di ripresa.

3-Studio di curve e superfici : da un puto di vista più teorico che pratico, quindi analizzando gli strumenti matematici che permettono di definirle e manipolarle.

4-Rendering : tecnica che permette di generare immagini di sintesi a partire da rappresentazioni digitali.

5-Geometria computazionale : disciplina che si occupa di trovare soluzioni algoritmiche efficienti ai più svariati problemi geometrici, in termini di tempo e spazio.

6-Hardware per la grafica : lo sviluppo dei sottosistemi per la grafica 3D ha sperimentato negli ultimi anni un' evoluzione nettamente superiore alla famosa legge di Moore (anno 1965, "le prestazioni dei microprocessori raddoppieranno ogni 18 mesi").

Ho voluto citare i vari settori applicativi per spiegare innanzitutto i miei interessi in questa branca dell' informatica qual' è la computer grafica, per poi discutere il progetto da me realizzato e la pipeline di lavoro che ho adottato.

Il mio lavoro riprende nello specifico i due punti sopra citati: la modellazione e l' animazione.

Ho voluto cimentarmi su entrambe le discipline poichè sono strettamente correlate alla realizzazione dei modelli. Durante lo studio delle tecniche principali utilizzate in modellazione, ho riscontrato dei limiti nella realizzazione dei personaggi (characters) utilizzando i classici software di modellazione e animazione (3ds max, Blender, Maya). A questo punto mi sono immerso nel mondo dello sculpting con ZBrush, un software closed source (purtroppo come tutti gli altri , ad eccezione di Blender) ma con delle potenzialità e delle performance davvero superiori agli altri pacchetti di sviluppo elencati.

Per la parte relativa alla modellazione ho deciso di sfruttare tale tool, notando un lavoro decisamente più fluido, con un' enfasi nella realizzazione

dei *characters* che non si riesce ad avere con 3ds max ad esempio. Si tratta effettivamente di due approcci completamente diversi: con ZBrush si modella una mesh lavorando con pennelli che vanno a modificare i poligoni come se si stesse lavorando con l' argilla, in maniera quindi decisamente più fuzionale rispetto all' utilizzo di punti, lati e facce che fa 3d max.

Per quanto riguarda l' animazione, ho utilizzato uno strumento che ormai è integrato in 3ds max, ovvero CAT (Character Animation Toolkit), plugin ormai integrato in 3ds max 2013 che offre una facile interfaccia permettendo all' animatore di creare *rig* personalizzati o usare diversi preset esistenti. Col termine *rig* si intendono gli scheletri (*skeletal rigs*) che vengono applicati ad una mesh già esistente, ad esempio creata precedentemente tramite Zbrush come farò io durante il progetto.

Dopo aver spiegato ed elencato in sintesi i punti essenziali della mia tesi, o quantomeno i concetti di base, nei primi capitoli verrà fatta una descrizione e un' analisi teorica dei principi di modellazione e animazione che mi sono serviti a supporto del lavoro successivo.

Lo scopo essenziale della mia tesi è stato quello di voler studiare una precisa pipeline di animazione, che è stata volutamente suddivisa in tre flussi di lavoro:

Generazione di quattro modelli : in questa fase preliminare, dopo aver studiato diverse tecniche offerte dal programma di sculting, ho creato da zero quattro modelli per la mia scena, tre dei quali personificano tre dei browser più utilizzati al giorno d' oggi, ovvero Chrome, Firefox e Internet Explorer. Nella mia animazione il ruolo che ricoprono questi tre modelli deve essere visto come una vittima del quarto modello, ovvero Beef, il toro. Dato che a mio avviso un' animazione deve cercare di descrivere una particolare scena più o meno verosimile, ho scelto questi quattro modelli a partire da un' idea assolutamente concreta, ovvero il progetto BeEF .

Setup ed esportazione delle mesh create in Zbrush : questo step intermedio mi ha consentito di ridurre in modo considerevole il numero

dei poligoni delle mie mesh, definire le texture a partire da una colorazione utilizzando la tecnica del polypainting e per finire utilizzare il plugin Goz per esportare il mio lavoro da Zbrush a 3d max.

Definizione di una scena animata basata sui character completati : caricati i modelli in 3ds max, ho focalizzato i miei sforzi e la mia attenzione nel capire innanzitutto alcune tecniche principali offerte dal plugin CAT (*Character animation toolkit*), sperimentando vari modi di animare un singolo character attraverso questo potentissimo strumento, ormai classico e indispensabile negli studi di animazione. Prima di animare nel puro senso del termine i miei modelli, ho dovuto associare a ciascuna delle quattro mesh uno scheletro creato ad hoc per ognuna di esse. In altri termini ho dovuto generare il sistema di ossa (*bones* esattamente) che permette ai modelli di essere poi animati a partire da una struttura di questo tipo.

Riguardo al progetto ho voluto scegliere un nome che cercasse di riassumere in pochi termini l' idea che la mia animazione ha voluto rappresentare: *Hooked browsers getting pwned with BeEF*, tradotto *Browsers hookati vengono bucati cn BeEF*. Lo sviluppo dell' animazione ha generato un file *avi* che in poco più di una decina di secondi mette in evidenza i movimenti dei modelli della camera e delle gabbie che son servite per bloccare le vittime.

Per riassumere meglio il fine di questo lavoro, dunque di questo progetto, ci tengo a dire che la mia pipeline di lavoro è nata sicuramente dal mio interesse per gli ultimi software utilizzati enormemente dall' industria dell' intrattenimento (videogiochi), dal cinema e dalla pubblicità. Credo che senza la mia passione per i videogames e soprattutto per la Blizzard non sarei mai arrivato a voler intraprendere uno studio simile. Detto ciò ritengo il mio progetto un buon punto di partenza per iniziare ad entrare nel mondo della computer grafica, e soprattutto in tutto ciò in cui ci si possa inventare un' animazione.

Indice

Elenco delle figure

Capitolo 1

ZBrush e la rivoluzione della modellazione

Il 2000 è l' anno in cui viene rilasciata dalla Pixologic la prima release di ZBrush, un programma di modellazione e sculpting 3d che si pone l' obiettivo di stravolgere completamente il normale approccio alla realizzazione dei modelli o di qualunque forma o geometria abbiamo in mente di voler realizzare. Prima di ZBrush molte applicazione 3D promisero il *digital sculpting*, ma soltanto ZBrush fu la prima applicazione a rilasciare questa tecnologia. Dal 2000 ad oggi sono state rilasciate diverse versioni, e la *community* che si andò a definire per merito di tale software col passare degli anni si espanse, contribuendo enormemente al miglioramento del programma da parte degli sviluppatori.
Per capire bene perchè ZBrush abbia rivoluzionato l' approccio tradizionale alla modellazione, bisognerebbe provare a generare un *character*, che sia umanoide o meno, ma su cui si voglia cercare di mettere in risalto anche una certa qualità dei dettagli. A questo punto ci si potrebbe cimentare con un programma di modellazione *old-style* e vedere se si riesce ad ottenere una mesh simile a quella generata con il programma di sculpting, e cercare di tenere in considerazione alcuni punti fondamentali durante tale confronto: il tempo di lavoro impiegato, l' efficienza dei tool utilizzati, le difficoltà di cia-

scun approccio, e infine il livello di dettaglio e resa del modello finale. Credo che su tutti i punti appena elencati Zbrush esca certamente vincitore, proprio perchè è stato progettato per consentire di avere uno strumento in grado di concentrare tutti gli sforzi di un artista 3D nella sperimentazione e nella fantasia che si può avere come se si stesse disegnando. Gli altri programmi di modellazione a mio avviso non potranno mai rimpiazzare un tool come Zbrush, soprattutto per il dettaglio e la fluidità che quest' ultimo permette durante la creazione di personaggi e mesh dall' elevato numero di poligoni. Infatti ormai la maggior parte degli studi di modellazione e animazione che lavorano per l' uscita di qualche nuovo game oppure film, utilizzano tale programma di sculpting.

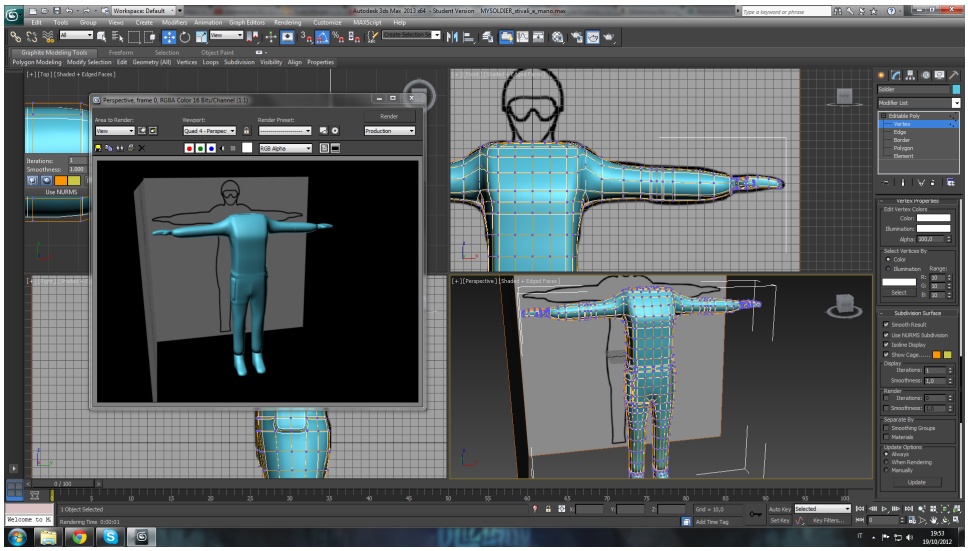


Figura 1.1: Esempio di modellazione di un soldato in 3ds max: tale approccio alla modellazione sta tutto nell' utilizzare degli strumenti di modellazione poligonale fortiti dal software per manipolare vertici, lati e facce del nostro modello

Potremmo fare un' analisi delle tecnologie che son state introdotte col passare degli anni e che hanno reso ZBrush un *tool* davvero unico e dalle estreme potenzialità per gli artisti 3D e non solo. Inizialmente ZBrush nacque come strumento che permetteva di creare illustrazioni in 2D e mezzo, ma

dalla seconda versione in poi furono introdotte delle tecnologie sempre più accessibili e utili agli artisti 3D.

1.1 ZSpheres

Con la seconda versione di Zbrush fu introdotta la tecnica chiamata ZSpheres, che permette di creare armature virtuali che possono essere successivamente convertite in poligoni e scolpite fino ad ottenere la forma desiderata. Attraverso questo tool non bisogna più faticosamente selezionare ed estrudere componenti poligonali come punti, facce e lati (come si farebbe ad esempio con un classico programma di modellazione), ma si può creare una topologia accurata se si usa correttamente tale strumento.
Zsphere è una tecnica proprietaria di tale tool che non ha assolutamente a che vedere col normale approccio alla modellazione che hanno altri programmi come 3ds max, Blender e Maya per citarne solo alcuni. Tali programmi infatti si basano sempre sull' utilizzo di primitive geometriche (cubi, piani, triangoli, curve, ecc.) che subiranno poi un processo di estrusione e modifica della loro *shape*, potendo selezionare e controllare ogni singolo vertice, lato o faccia poligonale dell' oggetto su cui si sta lavorando. Ciò può sicuramente risultare comodo durante la generazione di oggetti poligonali non troppo complessi in termini di forma e dettagli.
Ma perchè ZSphere risulta così importante in una fase preliminare del nostro lavoro? Perchè fondamentalmente va a sostituire operazioni che spesso molti *modellers* preferiscono o vogliono fare (in modo autolesionistico a mio avviso) con programmi come Maya e 3ds max. In sostanza una pipeline diversa che molti artisti utilizzano riguarda la creazione con Maya o 3ds max di una forma grezza e minimale di un modello, in modo tale da avere una base di partenza solida su cui poi aggiungere dettagli e rifiniture di ogni tipo col programma di sculpting. A questo sorge spontanea un' osservazione: perchè complicarsi il lavoro quando ZBrush con due semplici e potentissime tecniche quali ZSphere e ZSketch può risolverci ogni tipo di problema in tal senso?

Avendo provato sia la modellazione con l' uno e l' altro software (3ds max vs Zbrush), penso di aver già detto la mia riguardo alla scelta più veloce e qualitativamente superiore tra i due metodi.

1.2 SubTools

Nel caso in cui si vogliano aggiungere dettagli al nostro modello di base, ZBrush fornisce uno strumento davvero utile che prende il nome di SubTools. Attraverso questa tecnica è possibile splittare il nostro lavoro in diverse parti e concentrare la nostra attenzione e vena artistica su altri oggetti 3d che poi verranno uniti alla mesh di partenza. Chiaramente ogni oggetto può essere scolpito e modificato utilizzando tutta una serie di *modifiers* che ZBrush mette a disposizione dell' utente. Insomma quello dei SubTools è un approccio simile al principio *divide et impera* della programmazione: suddividere un problema in sottoproblemi sempre più piccoli e semplici, finendo la nostra computazione con un bel *merge* di quello che abbiamo sviluppato. Avendo la possibilità di lavorare su livelli separati che godono di proprietà indipendenti come ad esempio quella di poter avere livelli di suddivisione differenti per ciascun oggetto, è possibile avere un flusso di lavoro comodo e scalabile a qualunque esigenza di produzione. Attraverso un' intuitiva interfaccia, si può riorganizzare l' ordine dei subtools, controllare la loro visibilità, aggiungere, cancellare e fare il merge tra i vari componenti della nostra realizzazione, fino ad ottenere una mesh unica.

A proposito di merging, ci tengo a sottolineare l' indispensabile plugin a cui mi sono affidato per compiere la fusione tra gli elementi: SubTool Master. E' un plugin scaricabile come tutti gli altri dal sito ufficiale della pixologic (http://www.pixologic.com/zbrush/downloadcenter/zplugins/), e che permette di fare operazioni di *mirror*, duplicazione, *merge*, *export* su tutti I Subtools che desideriamo.

E' evidente che, per ricollegarmi al progetto, non sono riuscito a fare a meno di questo comodissimo strumento durante la realizzazione di tutti i

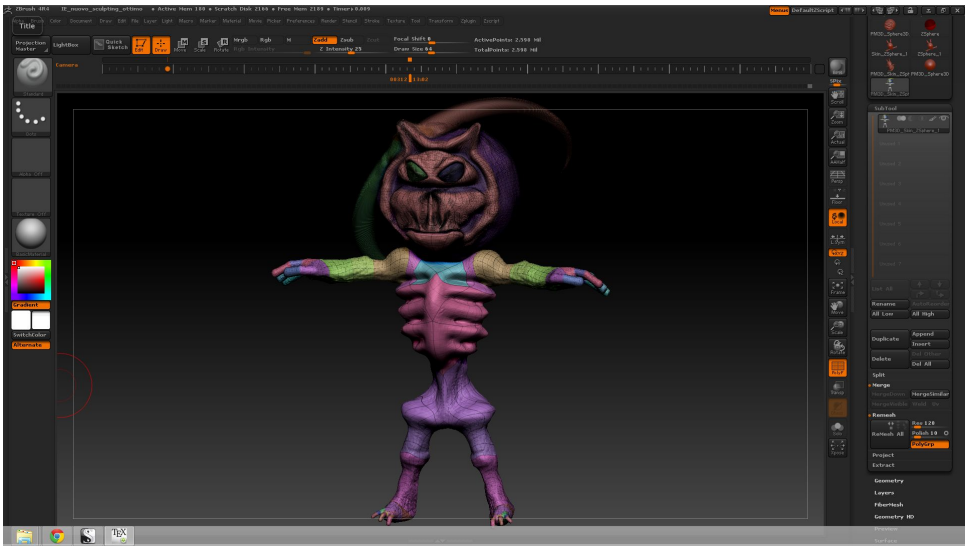


Figura 1.2: Subtools stack

modelli, ma gli sviluppi li commenterò nel dettaglio nei capitoli inerenti il progetto vero e proprio. Per il momento mi limito ad analizzare le tecniche di base che ho sfruttato nella parte pratica della tesi.

1.3 ZSketching e Hard-Surface Brushes

La versione 3.5 ha introdotto anche la tecnica di ZSketching, un processo in cui strisce di argilla virtuale sono dipinte sull' armatura generata (ad esempio utilizzando le ZSpheres sopra citate), smussate e scolpite in forme organiche. ZSketch è un tool tramite il quale si possono creare delle mesh applicando diverse pennellate (*strokes*) sull' armatura creata precedentemente. Si tratta quindi di un tool molto potente per creare complesse forme in maniera veloce e facile, cosa completamente irrealizzabile fino a questa release del software appunto. Descriverò anche il nuovo set di strumenti chiamati *hard-surface*, fornito anch' esso con questa versione e dà agli artisti la capacità di generare oggetti meccanici nello stesso modo in cui vengono elaborate le forme organiche con la tecnica di ZSchetching. La comodità dello ZSketch è resa possibile dalla gamma di brushes che il nostro software di sculpting

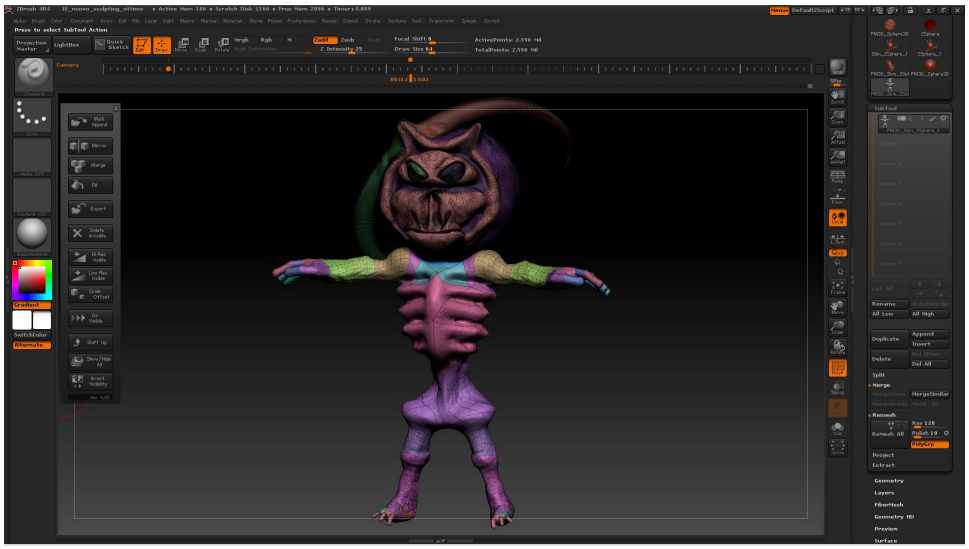


Figura 1.3: Interfaccia subtool master

fornisce per permetterci di raggiungere la forma (*shape*) voluta nel più breve tempo possibile e nel modo migliore. Il *ZSketch brush menu* non è chiaramente vasto quanto quello standard, ma fornisce un' ampia gamma di pennelli: Armature, *Bulge and Flush*, *Bulge*, *Float*, *Flush*, *Flush Dynamic*, *Flush res*, *Push Pull*, *Sketch1*, *Sketch2*, *Sketch3*, *SketchA*. Ho adottato questa tecnica per realizzare il modello di internet explorer.
Per riassumere:

ZSphere : generazione armatura di base.

ZSketch : aggiunta di dettagli alla mia mesh di sfere.

Altra tecnica che può essere estremamente utile nella realizzazione di characters realistici è l' *hard-surface sculpting*. Stiamo parlando di una varietà di *brushes* che sono stati inseriti a partire dalla versione 3.5 del programma per migliorare la realizzazione di componenti meccaniche che possono essere inserite in un contesto più ampio, quale ad esempio il miglioramento di una mesh e del suo design generale. L' attenzione ai dettagli meccanici rende ZBrush uno strumento ancora più innovativo di quanto era fino alla versione 3. Cerchiamo ora di analizzare le quattro categorie di brushes introdotte:

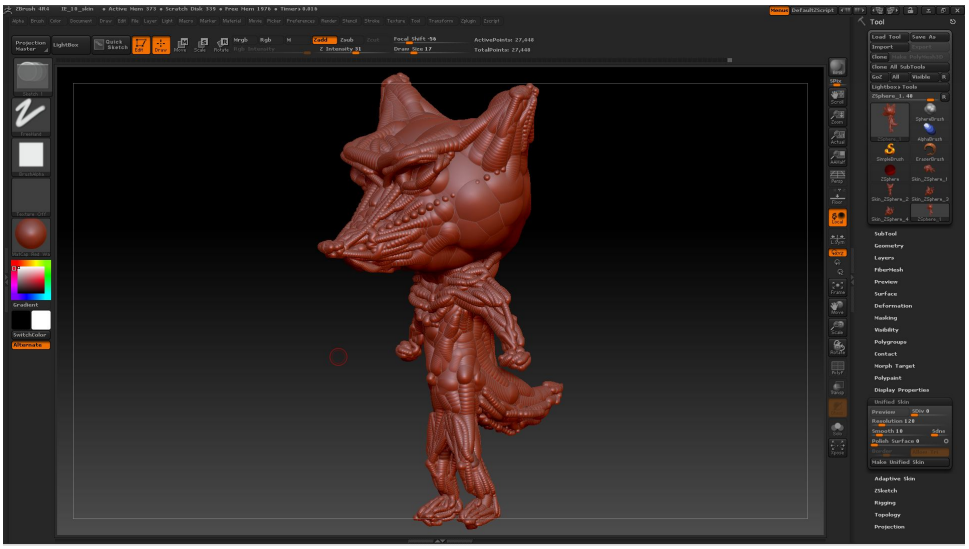


Figura 1.4: Il modello di firefox utilizzando lo Zsketch

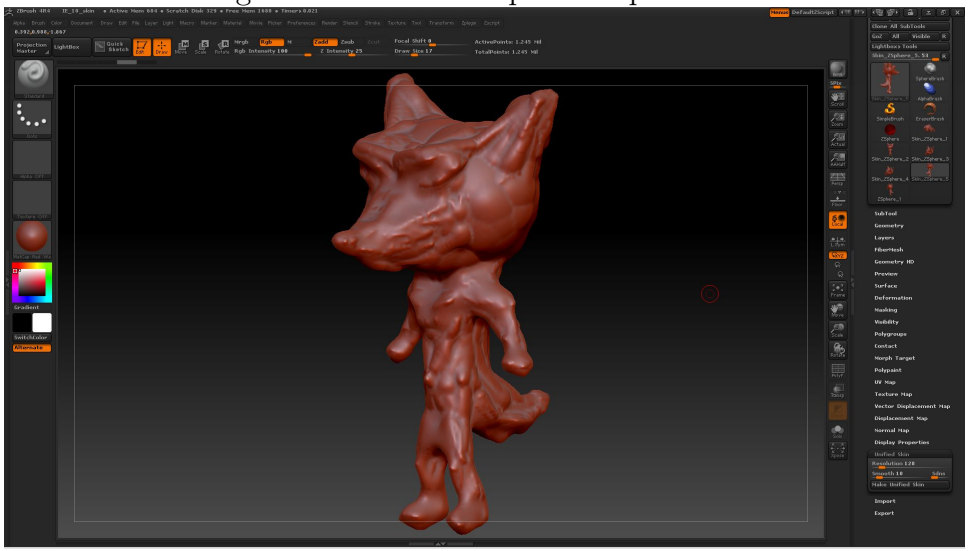
Clip : premendo ctrl+shift si può attivare tale *brush* che permette di creare una selezione che viene appiattita sul resto del modello. In sostanza l' oggetto su cui si opera con questo *brush* viene tagliato. La possibilità di agiungere anchor points premendo il tasto alt si rivela molto utile per definire forme e contorni a piacimento, fino ad ottenere la geometria desiderata. La creazione di componenti meccaniche e non solo risulta estremamente veloce, al contrario di altri programmi. Come possiamo notare i punti di forza che rendono ZBrush un software davvero unico sono innumerevoli.

Planar Brushes : usata per modificare un lato e creare degli angoli sfruttando i *Backtrack line setting.*

Trim brushes : taglia e rimuove una parte selezionata da una superficie rispettando le normali e mantendo intatti gli angoli.

Polish brushes : rispetta tutti i lati. La superficie colpita da tale pennello subisce un processo di rilassamento lungo la sua forma, fino a diventare più uniforme e schiacciata.

Figura 1.5: Firefox dopo l' adaptive skin



1.4 ShadowBox, Groups Loops e Matchmaker

E' con l' ultima release del software che si possono sfruttare le ultime tecnologie introdotte in ZBrush. Con la tecnica denominata ShadowBox, l' artista 3D è in grado di disegnare i profili di un oggetto che verrà poi generato nello spazio 3D. I profili vengono accuratamente definiti all' interno di una *box* appunto, un cubo, che permette di visualizzare tre viste: *front*, *side* e *top*.

I Group loops consentono invece di isolare i *polygroups* con un *loop* di lati tagliato da Zbrush. Ciò permette di mascherare e muovere le aree raggruppate mantenendo ad ogni modo gli angoli definiti nella mesh.

Un altro importante e utile strumento è Matchmaker. Attraverso tale tecnica si può facilmente adattare una forma ad un' altra in ZBrush. Questa funzionalità rende il tool ideale per applicare trafori, stampi o rientranze nella superficie dell' oggetto a cui si vuole aggiungere un dettaglio. Il Matchmaker è particolarmente consigliato per la generazione di pezzi di armature o uniformi durante la creazione di una mesh (ad esempio un soldato). Bisogna

ricordarsi che Matchmaker lavoro solo ed esclusivamente sotto l' asse z.
Avendo analizzato le tecniche principali e innovative introdotte col passare degli anni nel software di sculpting, adesso voglio trattare la tecnica di texture painting che ho adottato per i miei modelli.

1.5 Texture painting

ZBrush offre varie strade per creare mappe di colore:

Polypainting

Progettazione di texture a partire da fotografie

UV texture map

Quando si vuole applicare una texture su un modello, un metodo molto comune consiste nell' uso del Polypainting, che successivimente verrà aggiunto nello spazio UV. Nello specifico si attua un *baking*, ovvero si intende un processo di conversione dei dati dei *polypainting data* in *texture map 2D* che possono poi essere letti e renderizzati da programmi esterni a ZBrush. Questa pipeline di lavoro consente agli artisti 3D di concentrarsi esclusivamente nella tecnica di Polypainting, riuscendo a ottenere texture con una risoluzione massima. Si tratta di un approccio al *texturing* che permette di applicare il colore assegnando un valore RGB univoco a ciascun vertice di un poligono. Col termine UV space (spazio UV) si intende un metodo attraverso cui si associa a un oggetto 3D un corrispondente piano 2D che servirà per distendere la texture. Un punto rilevante che bisogna sottolineare è il fatto che il POlypainting in ZBrush non necessita di alcuna coordinata UV per funzionare. In questo modo l' attenzione e lo sforzo di lavoro sono unicamente immersi nella generazione dei colori applicati alle varie parti del modello su cui si sta procedendo. Bisogna però evidenziare che se si vogliono adoperare le color maps create con questa tecnica in un programma esterno come Maya o 3ds max, è necessario tuttavia convertire tali mappe in coordinate UV che potranno poi essere renderizzate dal nostro *renderer* preferito (Mental Ray per

citarne uno molto utilizzato). Ci sono diversi metodi per generare le coordinate UV in ZBrush, ma queste possono anche essere compito esclusivo di altri programmi come Maya, 3ds max o Headus UV, dipende solo dalla scelta dell' utente. Le UVs sono importanti anche perchè servono a determinare la quantità di *pixel map* che sono devoti a un' area. Il nostro programma preferito di sculpting per questo tre tecniche diverse di *mapping UV*:

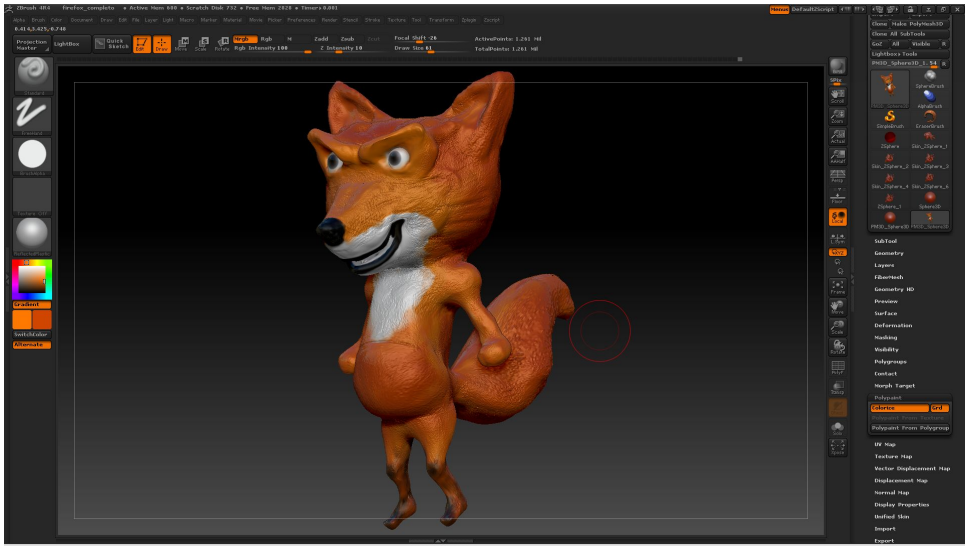


Figura 1.6: Firefox dopo il polypainting

- **AUV (*automatic UV*)** : il beneficio principale di tale sistema risiede nella velocità e nell' efficienza della generazione delle coordinate. Ma lo svantaggio sta nell' utilizzare ad esempio Photoshop per elaborare i risultati di Zbrush: non è possibile da altri programmi come Photoshop modificare una mappa UV generata con AUV poichè è in un formato codificato e non comprensibile. L' interazione successiva con la mappa sarebbe quindi irrealizzabile.

-PUV (*packed UV*) : è sicuramente il metodo più efficiente nonchè consigliato, infatti viene assegnato l' intero spazio UV al nostro personaggio.

-GUV (*group UV*) : le texture generate in questo modo sono un pò più leggibili ma meno eficienti.

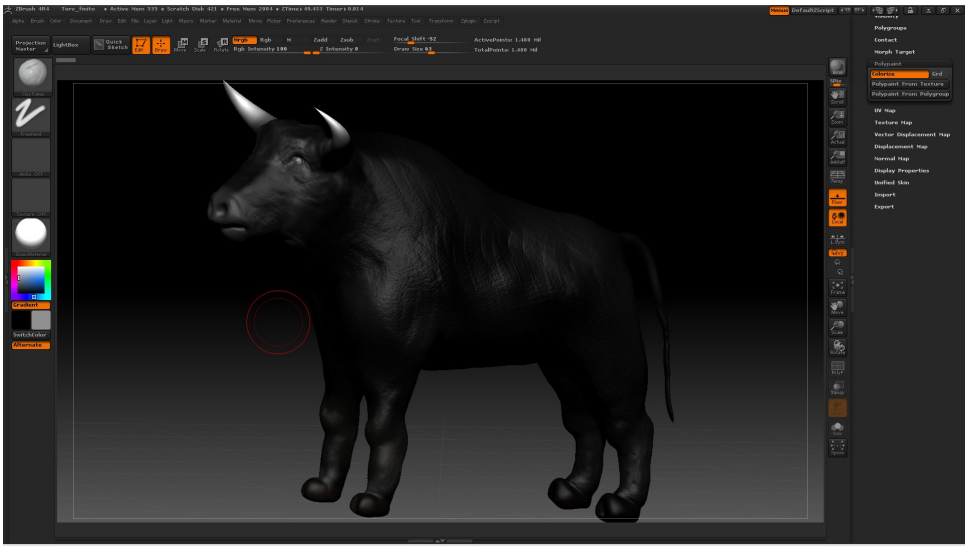


Figura 1.7: Beef completo di polypainting

Per continuare la nostra analisi sulle tecniche più utilizzate in ZBrush per generare e utilizzare esternamente le nostre texture, parliamo ora di *displacement map* e *normal maps*.

1.6 Displacement maps, Bump maps e Normal maps

Dopo aver scolpito e dipinto il nostro personaggio in ZBrush, potremmo aver bisogno di trasferire il nostro lavoro in un' altra applicazione 3D come Maya o 3ds max come abbiamo già detto precedentemente, ad esempio perchè abbiamo in mente di generare una particolare animazione o un semplice *rendering* dei nostri modelli. Nonostante molte applicazioni di computer grafica non possano renderizzare milioni di poligoni o animare mesh con un così alto numero di poligoni, si può sempre usare ZBrush in una pipeline di lavoro che spazi dai videogames ad altri effetti di visualizzazione. Utilizzando difference mapping, possiamo ricreare la nostra scultura ad alto numero di poligoni in un renderer adatto. Ci viene in aiuto un potente plugin che viene

spesso utilizzato in questo processo di esportazione del nostro lavoro preliminare completato con ZBrush: Decimation Master, un potente strumento in grado di esportare il nostro attuale modello 3D dall' elevato numero di poligoni mantenendo però un' accettabile dimensione del file suddetto. Ma che cosa si intende per *difference mapping* esattamente? Il termine si riferisce alle *texture maps* generate calcolando la differenza tra due superfici, solitamente una mesh ad alto numero di poligoni e una dal basso numero (*high-resolution* vs *low-polygon count*). Elaborando le informazioni derivate dal calcolo di tali differenze di risoluzione, ZBrush è in grado di produrre un valore appropriato per il tipo di mappa che è stata generata. ZBrush può creare due tipi di mappe: *displacement* e *normal*. Le *dispacement maps* creano dettagli ad alta risoluzione spostando fisicamente la geometria a tempo di rendering. Il modello sostanzialente viene cioè suddiviso in milioni di triangoli dal renderer che utilizziamo in un processo chiamato tassellazione, o meglio triangolarizzazione. Il perchè di tale operazione è dovuto al fatto che le primitive triangolari possono essere processate con maggiore semplicità ed efficienza rispetto alle altre primitive. Generalmente la tassellazione viene applicata per aumentare l' accuratezza della rappresentazione , dato che utilizzando un numero maggiore di componenti elementari di minore estensione si riesce in genere ad approssimare meglio la curvatura di una superficie. Bisogna sempre ricordarsi che il requisito importante di tale displacement map è che il processo di tassellazione che va a modificare la superficie verso l' interno o verso l' esterno è eseguito a tempo di rendering: ciò può impiegare anche molto tempo a seconda del livello di raffinatezza della mesh processata. Le mappe di tipo displacement sono in netto contrasto con le normal dato che queste ultime mantengono l' alto dettaglio della mesh originale senza andare ad alterare la geometria sottostante o il numero di poligoni. Inoltre non si può non evidenziare la possibilità che ZBrush offre riguardo alla generazione di *Bump maps* utilizzando il *Bump Viewer material* o utilizzando sempre mappe di tipo *displacement* ad hoc che vengono convertite in Bump. In sostanza le *displacement* e le *bump maps* sono identiche: entrambe sono

mappe in scala di grigio con valori bianchi rappresentanti l' altezza e aree scure raffiguranti la profondità. Ma la differenza principale risiede nel fatto che le *bump* non vanno ad alterare la silhouette della superficie: il motore di *rendering* le sfrutta per simulare asperità, solchi, sporgenze della superficie in esame. Questi dettagli vengono aggiunti solo in fase di *rendering*, ma come abbiamo detto non estendono la mesh su cui sono applicati. A causa della capacità del *bump mapping* di aumentare il livello di dettaglio degli oggetti, senza aumentare il numero di poligoni da renderizzare, questa tecnica viene impiegata molto nelle applicazioni dove è necessario renderizzare in tempo reale scene complesse e molto dettagliate (videogames e altre applicazioni che lavorano in *real-time*). Detto questo, l' applicazione migliore delle due tecniche di *mapping* sta nell' utilizzo di entrambe per arrivare ad un buon compromesso costo-prestazioni. Oltre alle mappe di tipo displacement e bump, ZBrush offre anche le Normal. Le *normal maps* creano solo l' illusione di una superficie dettagliata senza cambiare la geometria (al contrario delle displacement). Un *normal map* è generalmente un' immagine RGB che corrisponde alle coordinate X, Y e Z di una normale superficie da una dettagliata versione dell' oggetto. Si tratta di una tecnica molto valida quindi per migliorare l' aspetto e il dettaglio di una geometria scarna, quindi a basso numero di poligoni. Al giorno d' oggi la popolarità del *normal mapping* per *rendering* in tempo reale è dovuta all' ottimo rapporto tra qualità e requisito di processo. L' efficienza di ale tecnica è stata resa possibile soprattutto dallo scaling dei dettagli in base alla distanza degli elementi della scena renderizzata, permettendo di ridurre selettivamente i dettagli di un normal map di una data textur: ciò implica che le superfici più distanti dalle fonti di luce richiederanno simulazioni di illuminazione meno complesse.

Avendo descritto diverse tecniche offerte da ZBrush che mi son servite da base per il mio progetto di animazione, è arrivato il momento di trattare gli elementi e i punti principali riguardanti un' altro pachetto di modellazione e animazione 3D che ho scelto per il mio lavoro: 3ds max. Nel prossimo capitolo quindi andrò a sottolineare e analizzare gli argomenti che più mi sono serviti

di tale software, l' uso che ho fatto del plugin CAT (character animation toolkit), e spiegherò anche come sono riuscito a creare una pipeline di lavoro che portasse i miei modelli ad essere animati. Prima però tratterò due plugin che hanno fatto da *bridge* tra Zbrush e 3dsmax: Decimation Master e Goz, entrambi incorporati nel nostro programma di sculpting.

Capitolo 2

Decimation Master e Goz: due plugin di transizione verso 3ds max

Durante una pipeline di lavoro di animazione è necessario capire fino in fondo i tool che si vogliono utilizzare ma non solo. L' importanza che spesso ricoprono alcuni plugins spesso è sottovalutata oppure non abbastanza evidenziata all' interno di un processo lungo come quello di un' animazione. Nel mio caso specifico, ho già citato un plugin molto utile durante la fase di modellazione, ovvero Subtool Master. Altri due plugin che a mio avviso sono risultati essenziali al fine del mio lavoro sono stati Decimation Master e Goz.

2.1 Decimation Master

L' utilizzo di tale plugin non è fondamentale forse per un artista che si voglia limitare all' utilizzo di Zbrush esclusivamente per la realizzazione di un modello che non avrà un futuro animato, ma statico. Nel mio caso tuttavia risulta indispensabile, dato che si è in grado di ridurre facilmente il numero di poligoni delle nostre mesh prima che queste vengano esportate in

un altro programma come Maya o 3ds max (Goz si occuperà esattamente di tale processo, ma ne parlerò successivamente).

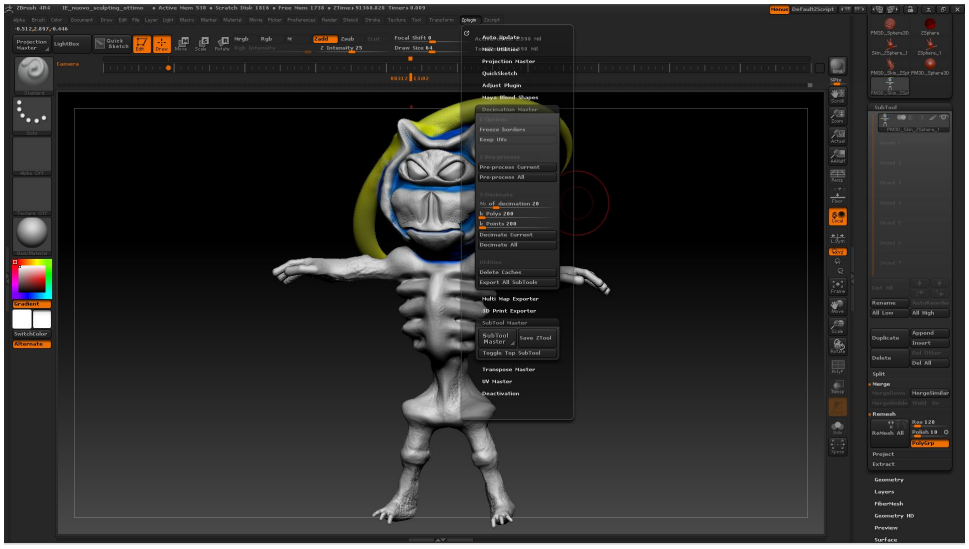


Figura 2.1: Il pannello con le opzioni che offre Decimation Master

Il processo attraverso cui opera Decimation master è abbastanza semplice, e si suddivide essenzialmente in tre step principali:

1) Settare le opzioni

Freeze borders : questa opzione evita la decimazione dei bordi e dei vertici che sono sulla superficie di un oggetto. Se il modello da decimare è parte di un insieme, sarà possibile saldare vertici e lati perfettamente dopo il processo di decimazione.

Keep UV's : con questa option possiamo mantenere le coordinate UV del modello originale che saranno pre-elaborate per una decimazione futura.

2) *Pre-Processing*

Questo è il secondo passo. Al suo interno, il plugin calcola la decimazione da una qualità del 100% a una dello 0% e crea una serie di file temporanei

(denominati Rete Progressiva). Poi, nella fase di decimazione che segue, il plugin leggerà questa maglia progressiva per applicare il risultato della decimazione. Questo pre-processo e il tempo di calcolo dipende dalla nostra ZTool attuale e dal suo numero di Active points. Utilizzando il *Pre-process current* oppure il *Pre-process All*, una barra di avanzamento verrà visualizzata nella parte superiore dell' interfaccia del programma con le informazioni sullo stato della pre-elaborazione. Altro aspetto da sottolineare è che ogni SubTool deve avere un nome univoco per l'elaborazione. Se si dispone di SubTools diversi con lo stesso nome, è necessario rinominarli prima.

3) *Decimating*

Questo è il terzo step. Si sceglie la qualità della decimazione da applicare. Se tale qualità di aggira intorno al 100%, significa che non c' è decimazione, 0.01% significa che la decimazione sarà massima. È anche possibile scegliere manualmente un valore specifico per il numero di vertici, punti o poligoni. Dopo aver scelto la qualità si possono sfruttare ozioni come *Current Decimate* per decimare il ZTool / SubTool, o *Decimate All* per tutti i SubTools visibili. Per una decimazione di alta qualità (40% -100%), il risultato visivo sarà quasi lo stesso del modello originale anche con tanti poligoni rimossi. Potrebbe a questo punto essere necessario scalare il modello per vedere la modifica. Invece per la decimazione di bassa qualità (2% - 40%), si inizieranno a vedere dei cambiamenti evidenti nei dettagli del modello. Questo può variare da modello a modello a seconda delle sue particolari caratteristiche e della sua struttura. È inoltre possibile scegliere di visualizzare il nostro target in modalità wireframe per vedere meglio le modifiche sul nostro ZTool. A tale scopo, basterà selezionare l' opzione *Frame* in ZBrush.
Per dare una dimostrazione dell' utilizzo di tale plugin ho voluto riportare alcune immagini del modello di *chrome* prima e dopo la decimazione. Ecco il risultato, con gli *active points* che cambiano drasticamente da circa 3 milioni a 64000.

Conclusione

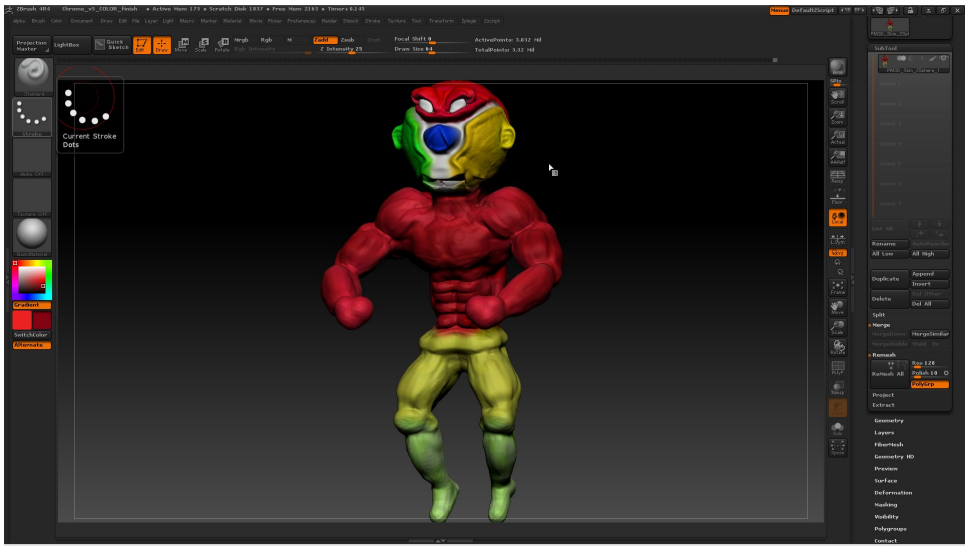


Figura 2.2: Il modello di chrome prima di utilizzare il plugin

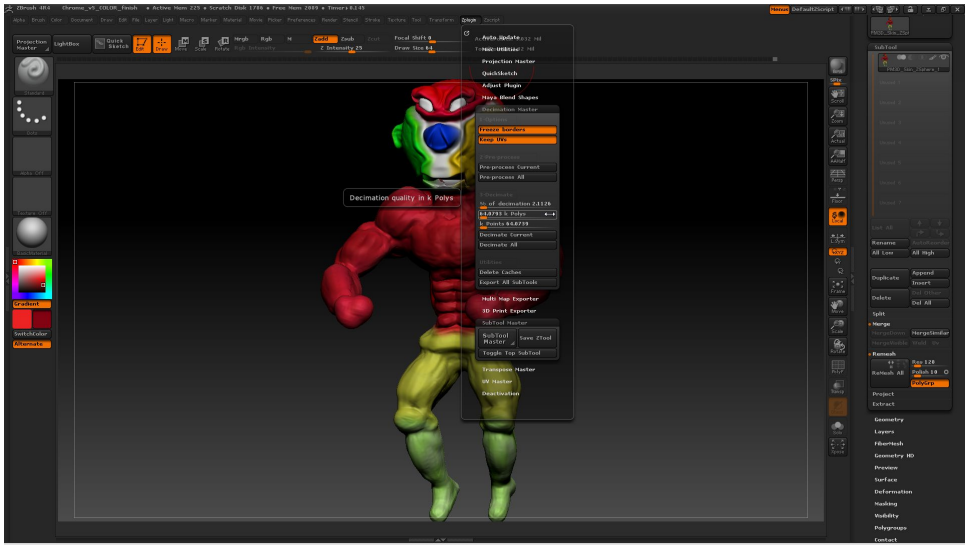


Figura 2.3: Durante il setting delle opzioni

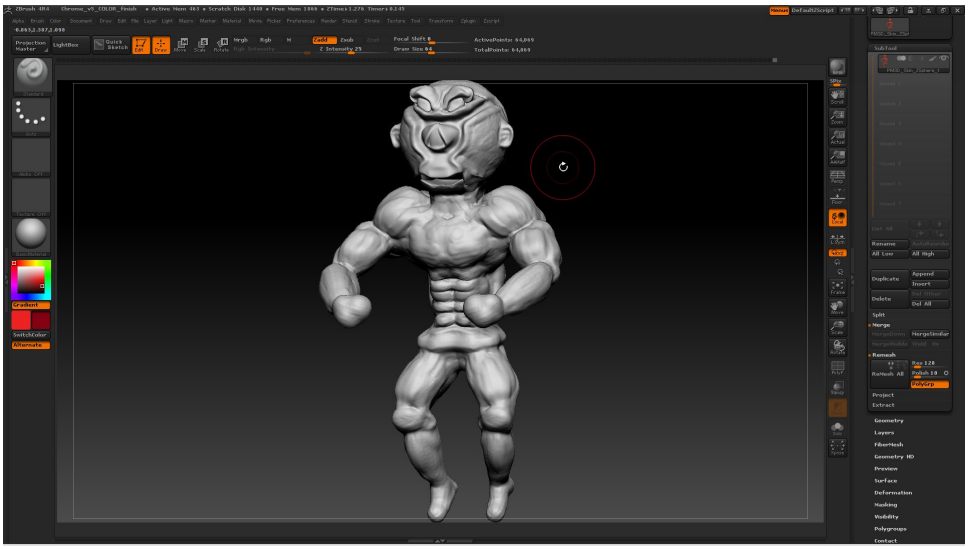


Figura 2.4: Dopo aver applicato la decimazione

Ciò che ho potuto notare dopo aver effettuato la decimazione dei poligoni è il fatto che è scomparso il *polypainting* associato alla mesh originale. Ritengo quindi che la soluzione migliore sia o generare una texture a partire dal polypainting, oppure applicarlo successivamente a tale fase, che risulta di vitale importanza per avere una mesh dal numero più limitato di poligoni. Terminato il lavoro di decimazione e polypainting sui modelli, è possibile passare ad una fase intermedia che lega i due pilastri fondamentali di tutto il lavoro svolto: l' esportazione con Goz.

2.2 Esportare direttamente da Zbrush a 3ds max usando Goz

Goz è un nuovo plugin disponibile con la quarta release di Zbrush che instaura una connessione forte tra il programma di sculpting e una determinata applicazione 3D come Maya o 3ds max ad esempio. Goz va quindi visto all' interno di una pipeline completa di animazione, dato che con un semplice bottone è in grado di esportare i modelli generati in Zbrush con i dettagli e

le texture associate.
A tal proposito è giusto evidenziare con un' immagine il modello di *firefox* dopo aver effetuato la conversione del polypainting in texture. Tale modello risulterebbe pronto quindi per l' esportazione con Goz, ma non bisogna tralasciare un fatto importante, cioè l' elevato numero di poligone delle nostre mesh.

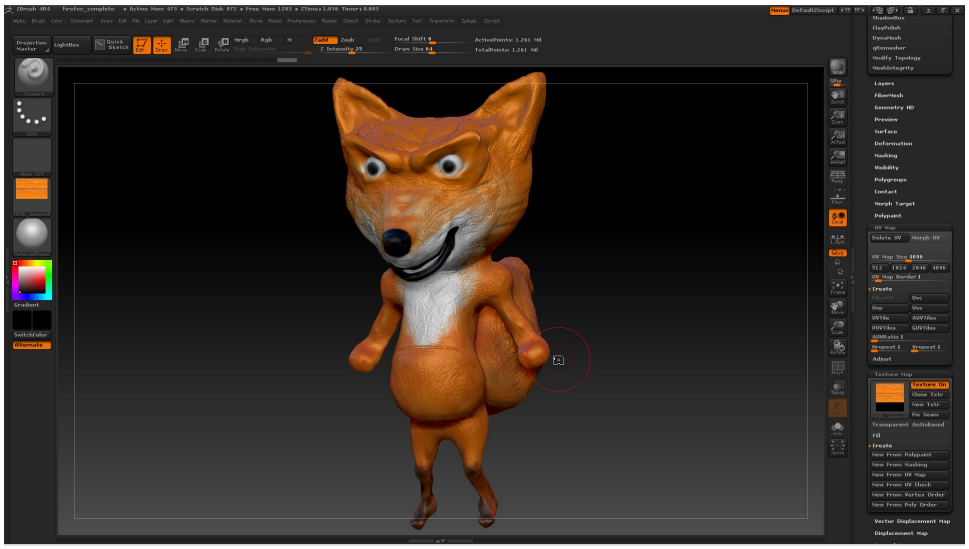


Figura 2.5: Firefox dopo la generazione della texture a partire dal polypainting, ma senza la decimazione dei poligoni

Per le animazioni i modelli con più di un milione di poligoni non sono assolutamente consigliabili, quindi prima di effettuare l' esportazione dei modelli sarebbe ragionevole utilizzare un plugin di cui ho già fatto un' analisi precedentemente: Decimation Master.

Supposto di aver completato anche la decimazione prima di utilizzare il nostro plugin che fa da ponte tra Zbrush e 3ds max, si sarà in grado di sfruttare Goz con pochi passaggi chiari e semplici. Fondamentalmente l' unica cosa che viene richiesta è di configurare l' applicazione destinataria delle nostre mesh generate in Zbrush. Dopo un semplice update dei path

delle applicazioni installate sul pc quindi, Goz è pronto per esportare quello che abbiamo generato.

Figura 2.6: Preferenze di Goz

Per spiegare meglio la potenzialità di questo fantastico plugin ho importato la mesh di *firefox* in 3ds max, riadattandola un minimo a livello di scala e posizionalmento per una resa migliore, e ho fatto un rendering velocissimo tralasciando impostazioni e miglioramenti di ogni sorta. Per quello ci sarà spazio quando analizzerò l' animazione vera e propria.

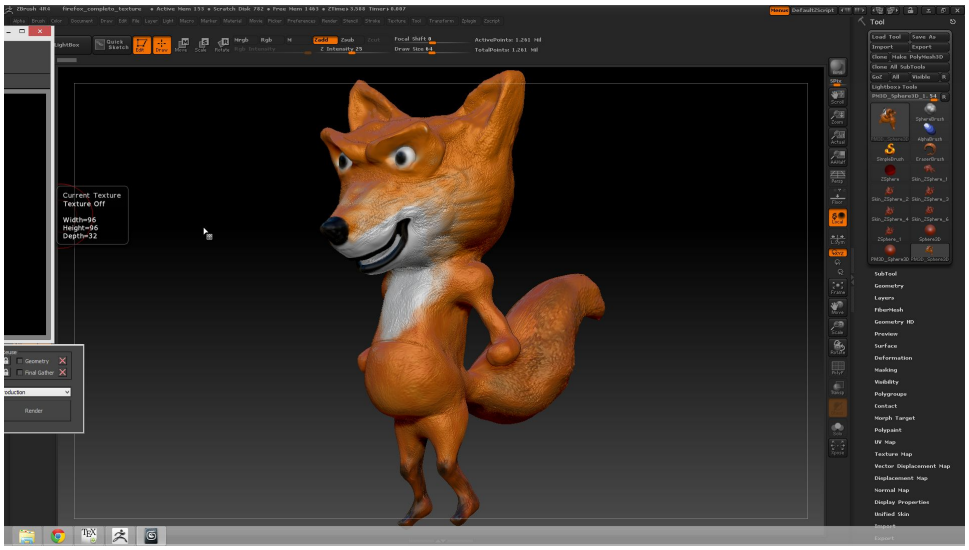


Figura 2.7: Firefox prima dell' esportazione tramite Goz

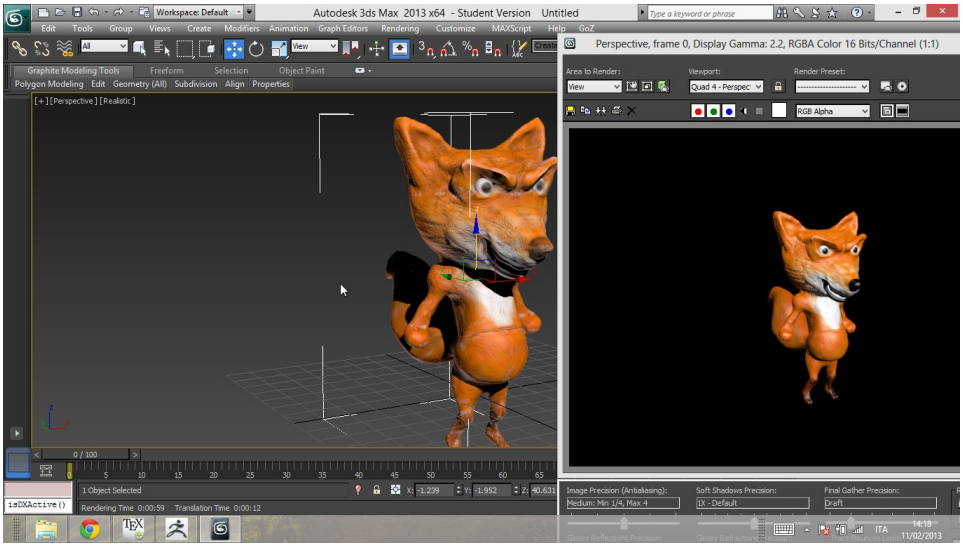


Figura 2.8: Firefox importato in 3ds max e renderizzato per una visualizzazione più realistica della texture

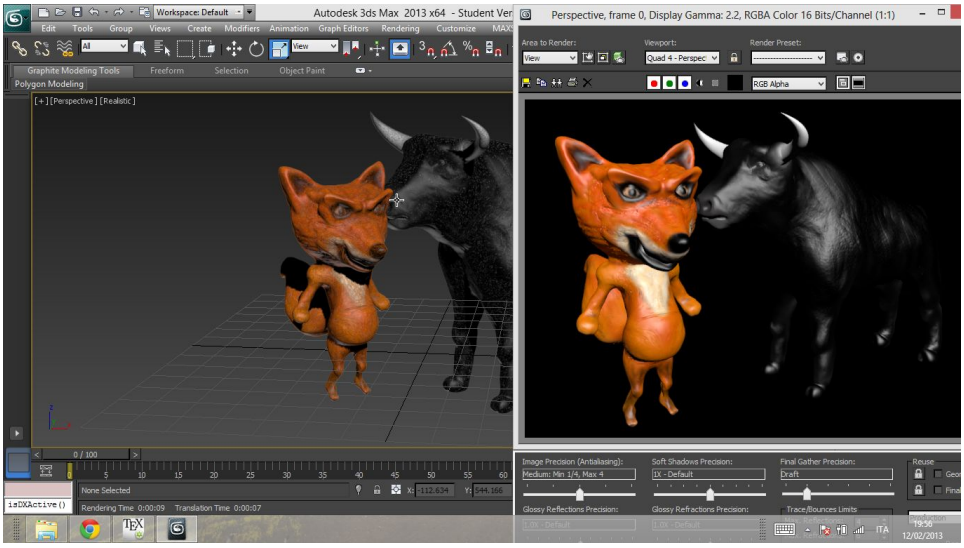


Figura 2.9: Firefox e Beef importati in 3ds max e renderizzati velocemente

Capitolo 3

Character animation toolkit

3.1 Rigging e skinning di sun modello

Prima di trattare nel dettaglio il plugin di animazione CAT che ormai tutti gli animatori 3D utilizzano con 3ds max, bisogna fare una premessa per capire il sistema che sta alla base di un processo di animazione di un modello, ovvero il rigging.
Col termine *rigging* si intendono esattamente due concetti essenziali e strettamente connessi che portano ad avere come output del nostro lavoro un *rigged character*:

Generazione di uno scheletro : si tratta della costruzione di un sistema di ossa (*bones*) gerarchiche e linkate tra loro che permette ad una mesh di essere animata a partire dalla manipolazione di questa struttura.

Skin mesh associata allo scheletro definito : lo skin permette al nostro character di seguire il sistema di ossa quando questo subisce delle modifiche e degli spostamenti. In fase di rendering, una volta che l' animazione è completa lo scheletro sarà nascosto e si riuscirà a vedere solo la nostra skin mesh.

Un' operazione da compiere dopo aver completato il primo step, quindi aver terminato la nostra gerarchia di ossa, riguarda la definizione dei limiti

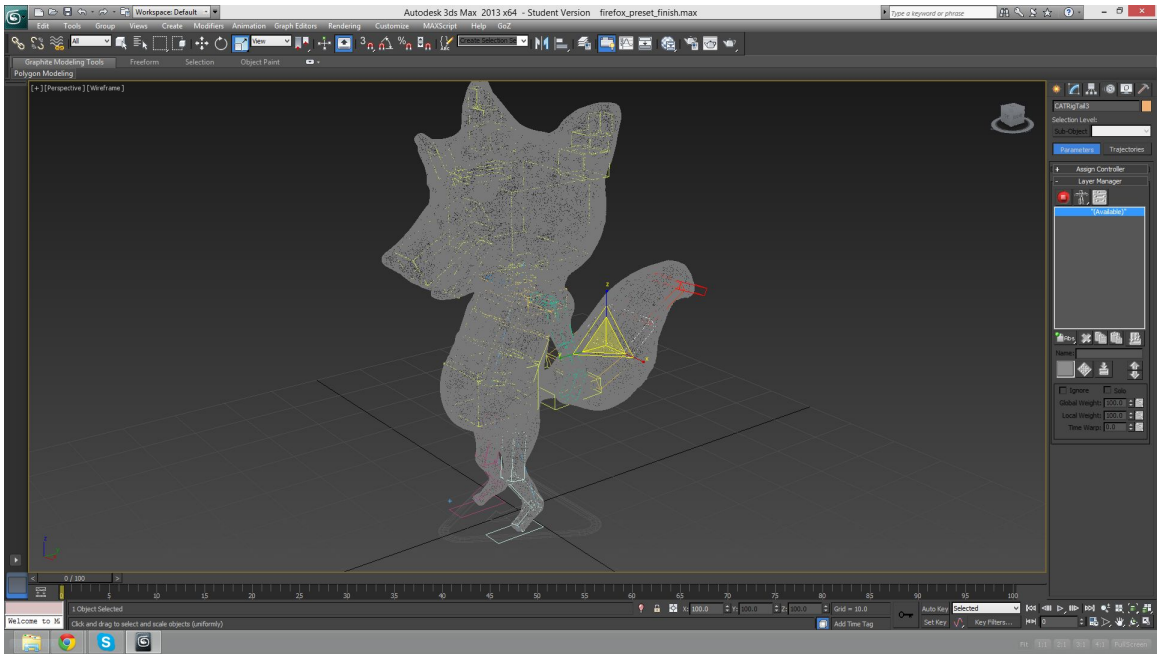


Figura 3.1: Firefox durante il rigging effettuato a partire da zero, cioè senza modificare un preset già esistente

di ogni *bone* e *joint*. Lo scopo di tale azione è quello di prevenire movimenti inaspettati della nostra mesh dopo che questa viene associata allo *skeleton*. Il processo che associa un certo modello a un sistema di ossa e che permette il *setting* dei pesi che ciascuna *bone* ricopre è definito *skinning*. A questo punto possiamo riassumere un tipico *rigging workflow* in questi termini:

1)Modellazione di un *character* .

2)Creazione dello scheletro .

3)Generazione di controlli di animazione : il *rigger* potrebbe occuparsi a questo punto di generare dei controlli di animazione che successivamente permetterebbero all' animatore di creare velocemente tutti gli *animation keys* per alcuni specifici movimenti (ad esempio per quanto riguarda le espressioni facciali oppure la manipolazione di braccia o gambe).

4)Applicazione dello scheletro alla mesh : attuato il collegamento tra scheletro e skin, può avvenire la definizione dei cosiddetti *skin weights*, associando ogni vertice della nostra skin mesh ad una specifica *bone* o ad un insieme di *bones*.

5)Animazione del modello : dopo tutta una serie di settaggi e step intermedi l' animatore può dedicarsi alla generazione di un' animazione, possibilemnte sfruttando l' ultima *feature* inclusa finalmente in 3ds max: CAT.

Col termine *animation keys* ci si riferisce a dei particolari stati di un oggetto o un modello in un determinato istante di tempo. Nel momento in cui un oggetto si muove o cambia il suo comportamento tra due *keys* differenti, si può affermare di essere davanti ad un' animazione. Senza capire questo concetto alla base di ogni sistema di animazione, non solo quello riguardante 3ds max, non si potrebbe procedere con un' analisi più accurata del plugin CAT e delle sue estreme potenzialità.

3.2 Animare un character con CAT

Negli ultimi anni il plugin CAT ha rimpiazzato il vecchio Character Studio, ormai datato e scomodo per molti animatori. Una delle *features* più innovative che hanno spinto molti animatori a scegliere CAT come plugin definitivo per lavorare sui modelli è stata sicuramente l' introduzione di una vasta gamma di scheletri già pronti. I vari *prebuilt skeletons* permettono infatti di non dover necessariamente costruire un rig da zero, ma magari semplicemente modificarne uno di default in modo tale che il sistema di ossa possa combaciare con la geometria del modello su cui si sta progettando un' animazione futura.
Vale la pena sottolineare però che spesso le modifiche a uno scheletro già pronto possono essere più costose in termini di tempo e sforzo rispetto alla generazione di uno scheletro costruito manualmente. Ad ogni modo la scelta

tra entrambi gli approcci al rigging che CAT mette a disposizione permette agli animatori di seguire flussi di lavoro indipendenti a seconda del *character* da animare.

Scelto il metodo più veloce ed efficiente tra i due a seconda dell' esigenza, bisogna cercare di visualizzare il modello in modalità *frozen and wireframe* possibilmente, in modo tale da non alterare la superficie della mesh con qualche click sbagliato, e in modo tale da riuscire a notare abbastanza facilmente come si vanno a delineare le nostre *bones* all' interno di essa.

Come già evidenziato precedentemente, a questo punto del lavoro l' animatore/rigger deciderà di applicare un modificatore di tipo *skin* alla mesh, dopo aver chiaramente aggiunto il sistema di ossa precedentemente sviluppato su di essa. Terminati questi passaggi si può iniziare ad entrare a tutti gli effetti all' interno di un processo di animazione del character in esame.

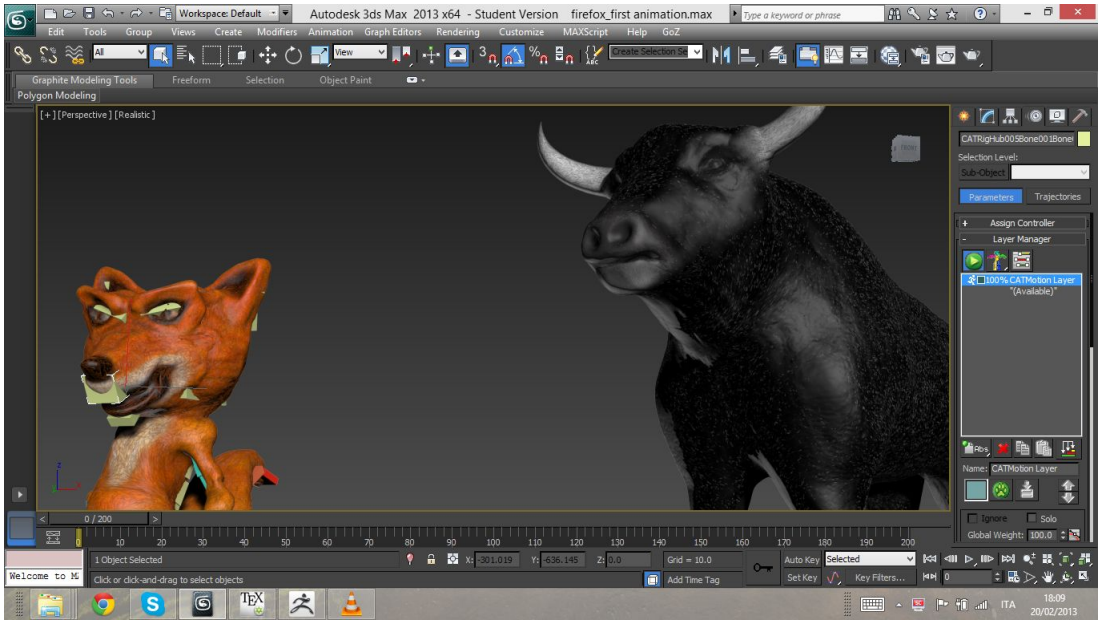


Figura 3.2: Interfaccia di CAT sulla destra con il Layer Manager

Il cuore di CAT risiede nel concetto di *animation layers*. Lavorare su livelli di animazione differenti infatti permette all' animatore di unire movimenti differenti e permettere allo stesso modello di avere espressioni diverse,

compiere gesti differenti o movimenti veri e propri allo stesso tempo. Quando si prova ad aggiungere un livello di animazione si hanno esattamente quattro possibilità, e si deve pensare di lavorare all' interno di uno stack di animazione in cui ciascun livello ha un' influenza sugli altri a seconda della posizione. Le quattro tipologie di *animation layers* sono:

1)Absolute layer : riguarda tutti gli animation keys che definiscono movimenti completi.

2)Local Adjustment layer : si riferisce ai keys relativi al sistema di coordinate locale del livello superiore.

3)World adjustment layer : applica un movimento relativo nel world space che rimane indipendente dai livelli precedenti.

4)CAT motion layer : crea loop procedurali di movimento come ad esempio un ciclo di camminata.

Selezionata una o più di tali tipologie, il passo da compiere sarà quello di aggiungere i keys ai nostri livelli di animazione appena scelti. Per compiere quest' operazione è sufficiente scegliere una tra due modalità: *Auto Key* o *Set Key*.

La differenza tra i due metodi di animazione riguarda il controllo che questi forniscono all' animatore. Con Auto Key attivo, ogni trasformazione o cambiamento di qualche parametro crea una chiave che definisce in che modo e in che posizione si trova l' oggetto su cui si sta attuando una qualche modifica, specificando quindi un determinato frame.

Si deve assolutamente capire inotre che ogni frame può contenere differenti chiavi, ma una sola per ciascun tipo di trasformazione o parametro a cui ogni chiave viene associata. Un esempio di tale operazione rguarda il fatto che se ruotiamo, scaliamo o muoviamo un oggetto nella nostra scena con Auto Key selezionato, avremo la generazione di tre keys, ognuna per ciascun tipo di trasformazione che l' oggetto ha subito.

Attivando il Set Key Mode invece, l' animatore può scegliere quali tipi di chiavi creare sulla base di una lista numerosa di opzioni: posizione, rotazione, scala, parametri, modificatori, materiali.

Altra caratteristica del plugin di animazione CAT è la possibilità di unire diversi livelli di animazioni differenti. Il *blending* avviene sulla base di una valutazione dell' ordine dei livelli nello stack di animazione che va dall' alto verso il basso. Si parte dal presupposto che ciascun livello dovrà influire un particolare movimento (quindi *bone*) o gruppi di *bones*. Nel caso in cui si voglia avere un controllo su tutto il *rig*, possiamo gestire il valore *Global weight*, mentre nei casi in cui è richiesta una particolare posizione o modifica di una singola *bone* possiamo intervenire manipolando il valore *Local weight*.

Una volta che il rig sarà associato a vari livelli, per evidenziare l' associazione tra questi e le corrispondenti parti del rig, CAT offre la *Rig coloring mode*, in modo tale che in fase di animazione, avendo numerosi livelli, si possa immediatamente avere una visione globale e precisa delle diverse parti da animare colorate diversamente.

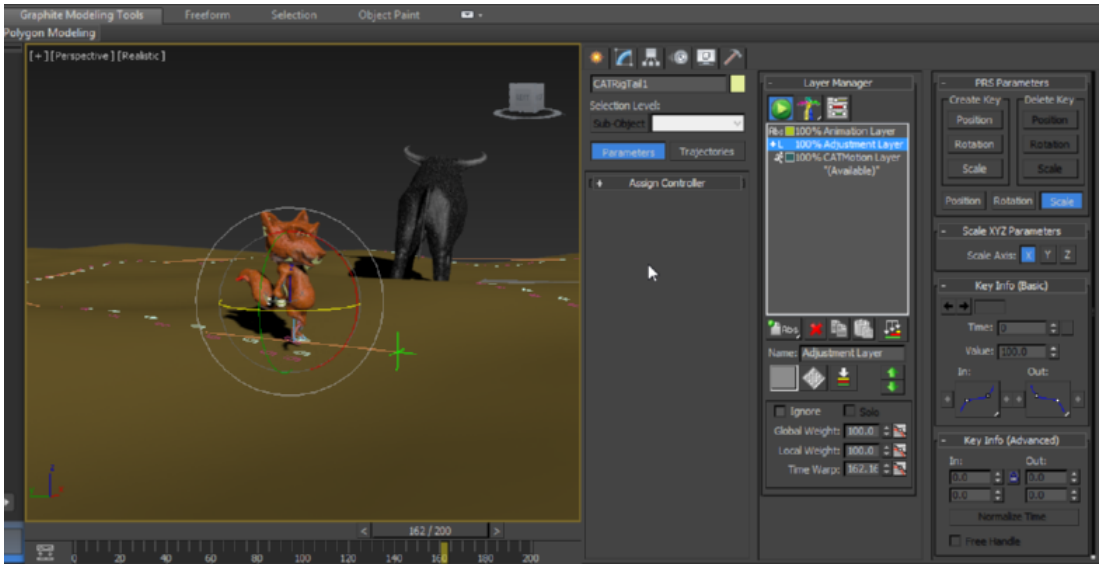


Figura 3.3: Cat layer manager e visualizzazione dei valori che vanno a influenzare i livelli

Per finire di analizzare le potenzialità del nostro Character animation

toolkit vale la pena citare un potente livello di animazione che possiamo inserire nello stack: Cat motion layer. Questo livello permette di applicare un ciclo di camminata o corsa al nostro rig. Tale operazione viene compiuta indipendentemente dalla creazione o meno di qualche keys, perchè ci sono già varie impostazioni di default per il ciclo. La gestione e la personalizzazione del ciclo chiaramente sono permesse tramite varie opzioni e menu che verranno analizzati meglio in appendice per quanto riguarda un' analisi più mirata di alcuni particolari tecnici che son serviti al fine dell' animazione.

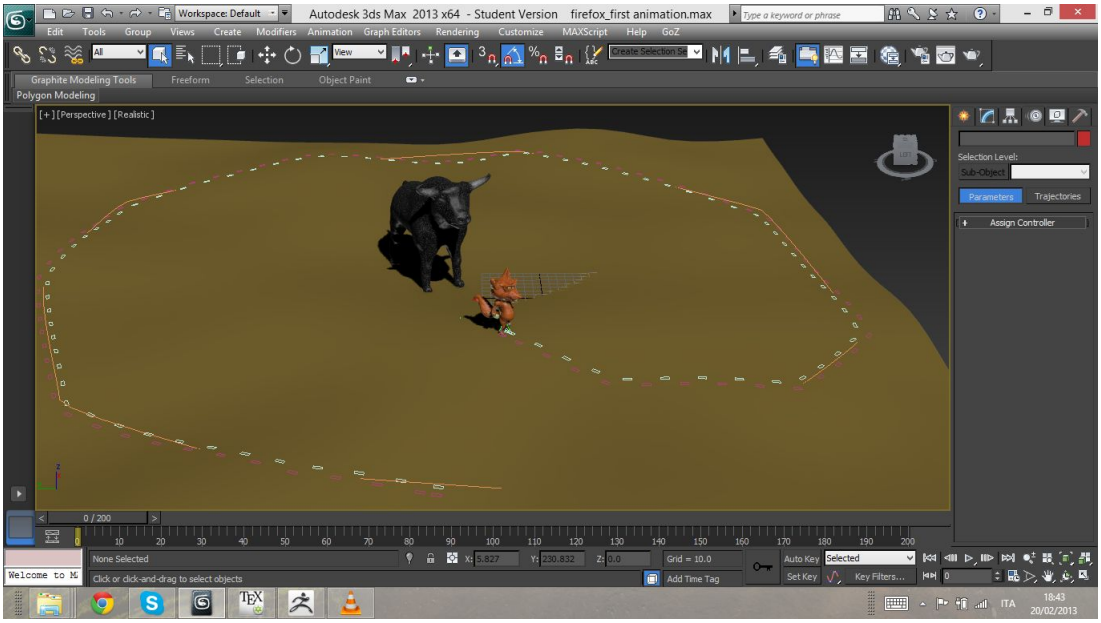


Figura 3.4: Esempio di walk cycle creato con un CAT motion layer

Capitolo 4

Il progetto: Hooked browsers getting pwned with BeEF

Come tutte le animazioni anche il progetto che ho sviluppato si è basato su un' idea, che chiaramente deve tener conto di diversi parametri per essere realizzata: tempo a disposizione, difficoltà nello studio delle diverse tecniche, qualità dei dettagli, numero degli elementi della scena. Ciò che deve fare la computer grafica, a mio avviso, tramite un' animazione, è quello di essere d' ausilio alla rappresentazione di una situazione, di un evento, di un progetto o più in generale semplicemente di un' idea che può essere frutto della fantasia dell' artista.

Nel mio caso specifico ho cercato di creare un' animazione che potesse mostrare graficamente e, da un punto di vista informativo, diverso il progetto BeEF (*The browser exploitation framework*).

4.1 Il framework BeEF

BeEF e' un tool di sicurezza open source che viene solitamente usato durante test di penetrazione per prendere il controllo del browser di una vittima a sua completa insaputa . Questo avviene convincendo la vittima ad aprire nel proprio browser un link, per esempio http://beefproject.com. Appena

il browser renderizza la pagina, che contiene il cosiddetto *BeEF hook*, ossia particolare codice JavaScript, viene inizializzato un canale di comunicazione bi-direzionale ed asincrono con il server BeEF. Da questo momento il browser della vittima viene controllato tramite BeEF.

Questo permette di enumerare esattamente che tipo di browser e' hookato, cioe' connesso, a BeEF. Tra le caratteristiche che vengono enumerate troviamo ad esempio tipo e versione del browser e dei sui plugins come Java e Adobe Reader. Queste informazioni tornano terribilmente utili quando si vuole determinare se il browser o uno dei suoi plugin sia affetto da qualche baco di sicurezza.

Se il baco, o ancor meglio i bachi, di sicurezza vengono identificati, BeEF puo' lanciare diversi exploit, ossia codice maligno che sfrutta il baco di sicurezza per, ad esempio, installare un malware nel computer della vittima.

Non solo, ma BeEF e' anche in grado di funzionare da keylogger (ogni tasto digitato viene registrato) completamente in JavaScript, e di modificare completamente il contenuto della pagina corrente per creare attacchi di Social Engineering. As esempio, registrare audio e video abilitando camera e microfono.

E' certamente complesso spiegare in modo esaustivo tutte le caratteristiche di BeEF, e non e' certamente lo scopo prefisso da questa tesi. Il lettore e' invitato a consultare: http://beefproject.com per maggiori informazioni, esempi e video.

4.2 L' animazione: dall' idea fino alla sua realizzazione

E' sempre necessario cercare di partire con qualche schizzo su un foglio di carta, anche minimale, ma che cerchi di dare una base della scena che l' animatore concepisce mano a mano che procede col lavoro. Dopo aver scelto l' obiettivo del nostro lavoro, si può entrare nella pipeline di lavoro

vera e propria, iniziando a realizzare per prima cosa i modelli che dovranno successivamente essere animati e renderizzati come ultima cosa.

Ovviamente bisogna tenere sempre in considerazione il tempo che si ha a disposizione: la qualità del lavoro finale, il numero di frames che si riuscirà a tenere in considerazione durante la nostra *timeline* saranno direttamente proporzionali al tempo che riusciremo a gestire, schedulando al meglio i vari percorsi di lavoro che si sceglieranno.

4.2.1 Modellazione

Lo sculpting con Zbrush mi ha permesso di generare quattro modelli ad alto numero di poligoni: Firefox, Internet Explorer(magari non il 6), Chrome e BeEF. Le tecniche spiegate in modo sufficientemente esaustivo nel primo capitolo come nell' appendice tecnica mi hanno portato nel giro di un mese e mezzo circa ad avere il materiale e gli elementi principali da inserire nella scena finale.

Ci tengo a sottolineare che tutti i characters sono frutto della mia fantasia, mentre per il toro ho cercato di mantenere un livello più realistico durante la realizzazione. Le difficoltà maggiori che ho incontrato in questa fase sono state legate alla scelta delle geometrie e delle dimensioni di varie parti delle mie mesh, in particolare le teste e alcuni arti, soprattutto le zampe del toro.

Non mi sembra utile rispiegare l' approccio adottato nella generazione dei modelli dato che ho evidenziato i diversi passaggi adottati nei precedenti capitoli. Di seguito mi limito a riportare i quattro modelli finiti e la gabbia che ho generato in un secondo momento per raffigurare il fatto che i browsers sono stati hookati e pwnati.

4.2.2 Setup dei modelli ed esportazione con Goz

Non si tratta di un' operazione molto dispendiosa in termini di tempo, ma bisogna stare molto attenti al grado di decimazione poligonale che varierà da modello a modello. In sostanza bisogna passare da una mesh con qualche

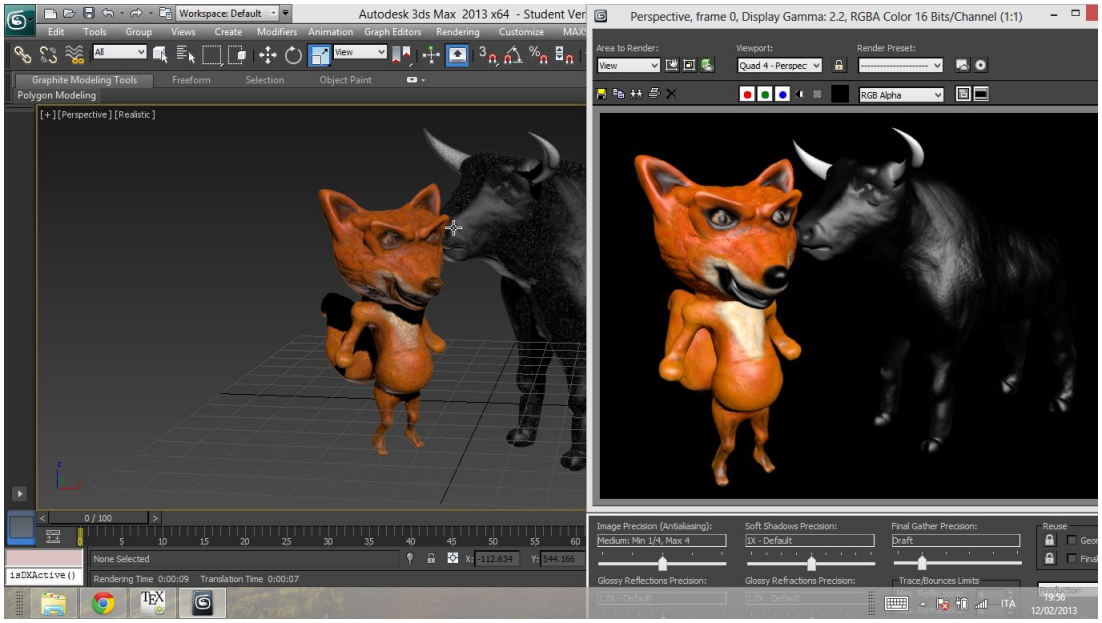


Figura 4.1: BeEF e Firefox completi

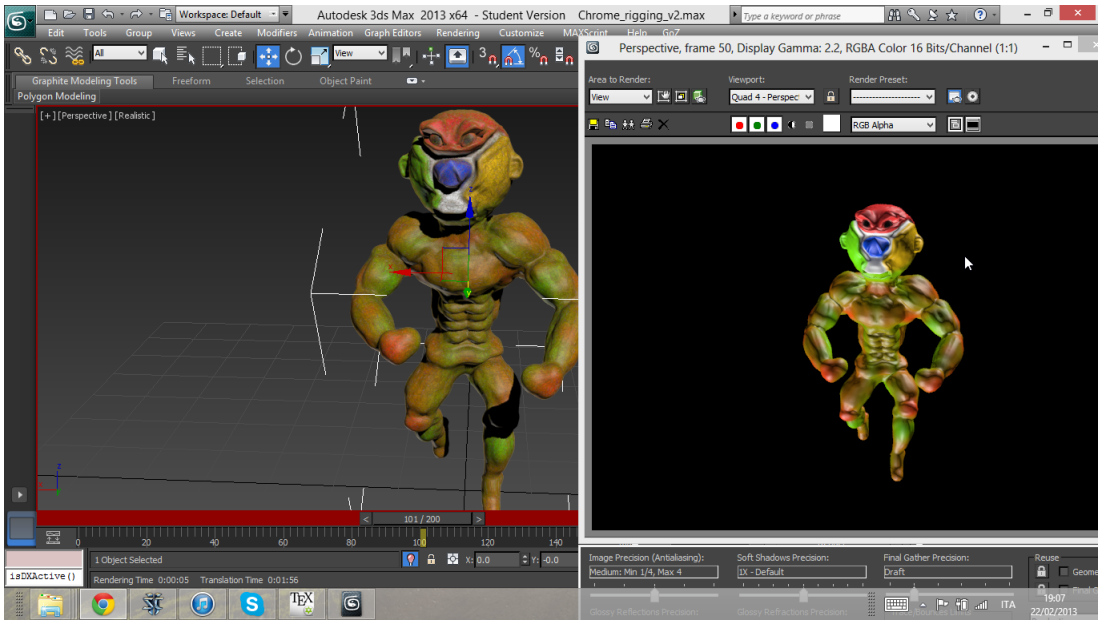


Figura 4.2: Chrome renderizzato e completo

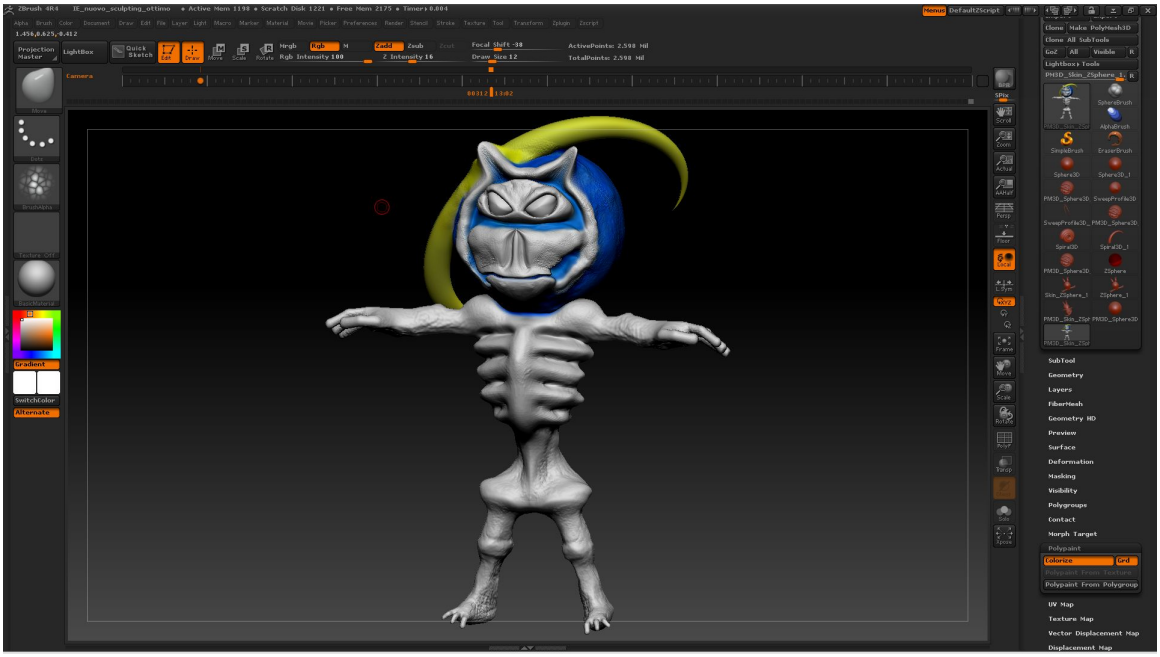


Figura 4.3: Internet explorer con polypainting che poi ho rimosso in fase di rendering finale

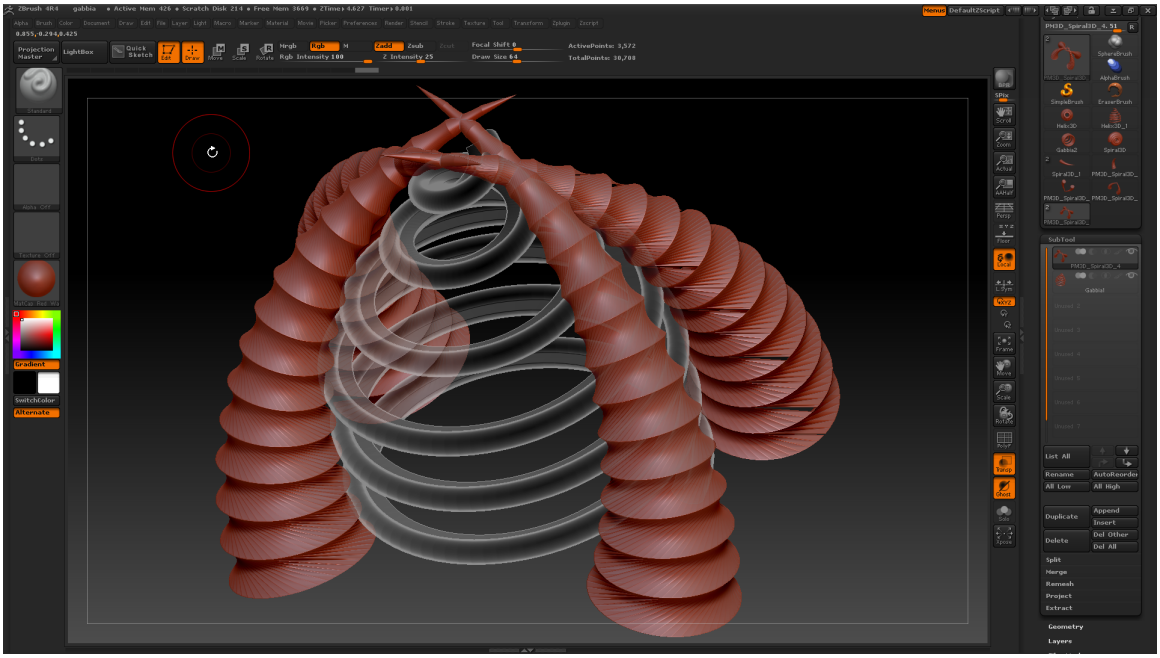


Figura 4.4: La gabbia che bloccherà i browsers

milione di poligoni ad una a bassa risoluzione, relativamente bassa dato che si tratta di mantenere dei dettagli che si vorranno mostrare in fase di rendering. I miei characters dunque dopo aver utilizzato il plugin Decimation Master sono passati da 2-3 milioni di poligoni a 30-100mila poligoni circa. Le cifre cambiano da modello a modello. Pur volendo snellire ulteriormente il carico di lavoro per 3ds max in fase di animazione, per non distruggere la geometria mi sono limitato a questa soluzione. Avendo preparato i modelli da un punto di vista geometrico, ho poi sviluppato le texture a partire dall' innovativa tecnica di Zbrush chiamata Polypainting. Per problemi legati all' efficienza del rendering, durante la generazione del video ho dovuto eliminare la texture che avevo generato in Zbrush per il modello di internet explorer. Infine, questo step intermedio tra Zbrush e 3ds max ha visto l' esportazione di tutte le mesh attraverso il plugin Goz che in modo estremamente veloce ed efficiente ha mantenuto la qualità dei characters creati nel programma di sculpting.

4.2.3 Rigging e animazione

Quest' ultima fase del mio lavoro mi ha portato via circa due mesi. E' stato infatti un processo a sua volta diviso in sottoproblemi che ho dovuto analizzare e risolvere in modo indipendente prima di avere una scena da renderizzare che potesse soddisfare la mia idea. I singoli movimenti degli attori della scena son stati studiati durante questa fase, dovendo prima predisporre un flusso di lavoro che mi consentisse di avere una visualizzazione accettabile attraverso l'uso di una camera di ripresa virtuale. Come di evince dal video che ho prodotto alla fine del mio lavoro infatti si nota come la scena, di per sè animata, non potesse avere una visualizzazione statica (quindi una camera fissa che inquadrasse l' ambientazione dall' alto per esempio), anche perchè non si sarebbe visto praticamente nulla dell' idea che ho voluto rappresentare. Potrei cercare di dare una descrizione più mirata delle diverse parti che ho dovuto sviluppare prima di arrivare a un risultato finito.

Rigging dei modelli in 3ds max : utilizzando i Cat rig, differenziati in preset già forniti dal programma e non, ho costruito 4 scheletri diversi

per i miei modelli precedentemente importati nel mio programma di animazione.

Più precisamente ho creato da zero i rig per i modelli bipedi, quindi parlo di firefox, internet explorer e chrome. Le difficoltà incontrate sono state superficiali o praticamente inesistenti se paragonate al rig del toro BeEF.

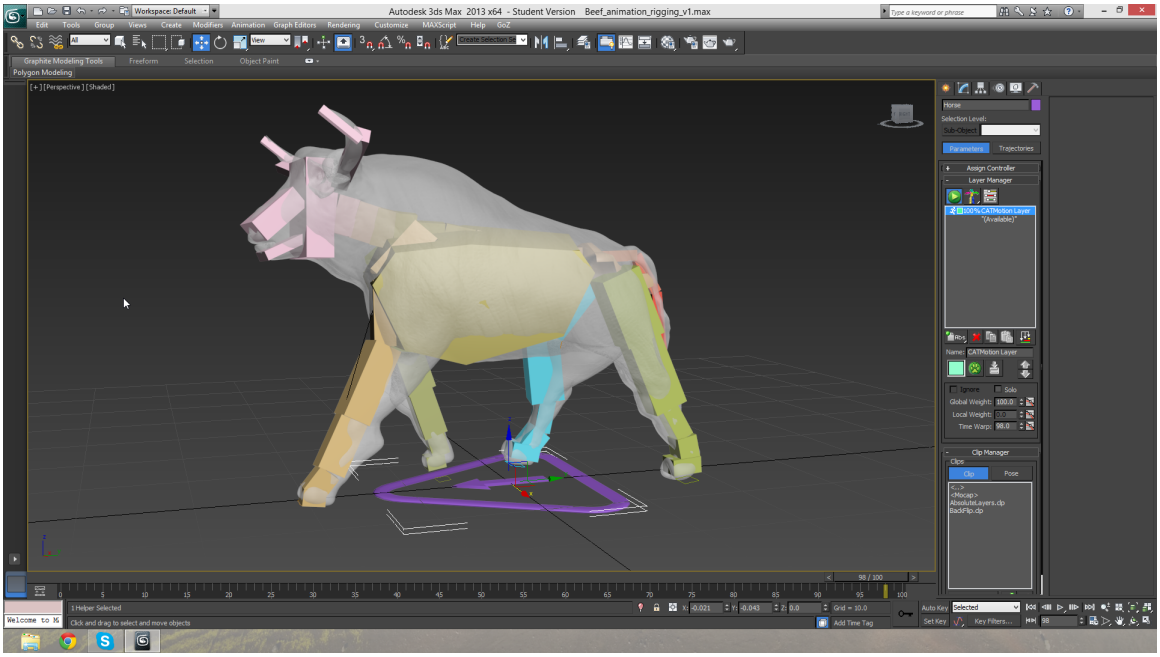


Figura 4.5: Beef surante il rigging

Per quest' ultimo infatti il lavoro è stato molto più complesso, in quanto un quadrupede con una geometria di per se non semplice come quella che avevo creato in Zbrush risulta complesso da animare, soprattutto se si tenta di creare un rig da zero. La mia esperienza a riguardo dopo tre giorni di prove e controprove mi ha poi portato ad utilizzare il rig di un cavallo, ma ho dovuto dedicare diverse ore alla personalizzazione e modifica di singole *bones* fino ad arrivare a un risultato accettabile. Mi rendo conto che ho ancora molto da approfondire sul

rigging e sul perfezionamento delle animazioni dei modelli, soprattutto sui quadrupedi.

L' operazione da compiere al termine di ogni rig riguarda l' applicazione di un modificatore al character in questione: il modifier skin. Con questo modificatore verrà legata la mesh al nostro sistema gerarchico di ossa generato in modo tale per successivamente si possa applicare al nostro personaggio un layer di aniamzione. btp

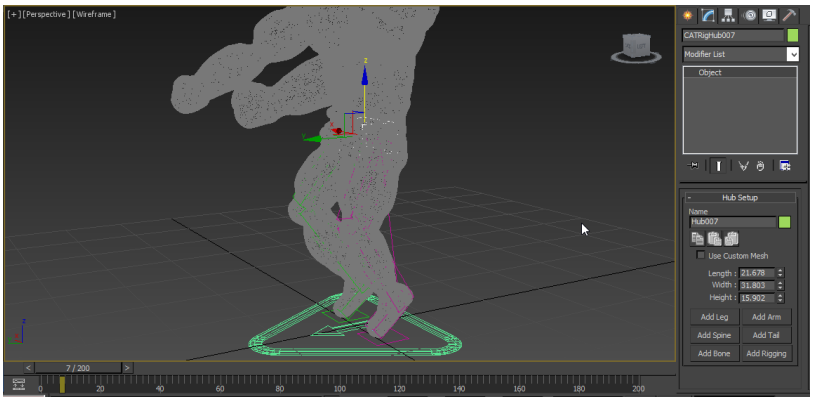


Figura 4.6: Chrome durante il rigging

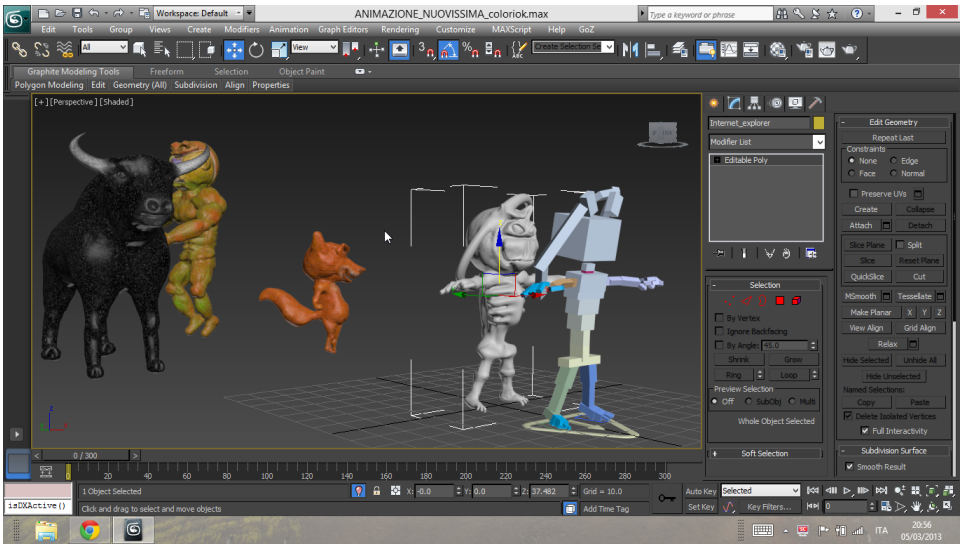


Figura 4.7: Tutti i modelli con il rig visibile di internet explorer

Il background della scena : non mi starò a dilungare sull' aspetto generale della scena riguardo al terreno e all' immagine che rappresentano un background minimale dell' animazione. Non era quello infatti lo scopo del mio lavoro. Per riassumere ho utilizzato due semplicissime primitive di tipo *plane*, applicato due texture e predisposto una luce di tipo *daylight* che simulasse un minimo una forma di illuminazione ambientale leggera.

Animazione dei modelli e della camera di ripresa virtuale : questo step finale ha visto nascere l' animazione dei characters ormai ultimati utilizzando un *animation layer* descritto nel capitolo 3 di questa tesi: *CAT motion layer*. Applicando un layer per ciascun character ho potuto creare in un secondo momento 4 diversi cicli di camminata che seguissero un *path* definito attraverso l' aggiunta di linee nella mia scena. Ovviamente per avere un ciclo di camminata efficiente è stato necessario lavorare sulla direzione e sull' orientamento dei singoli personaggi.

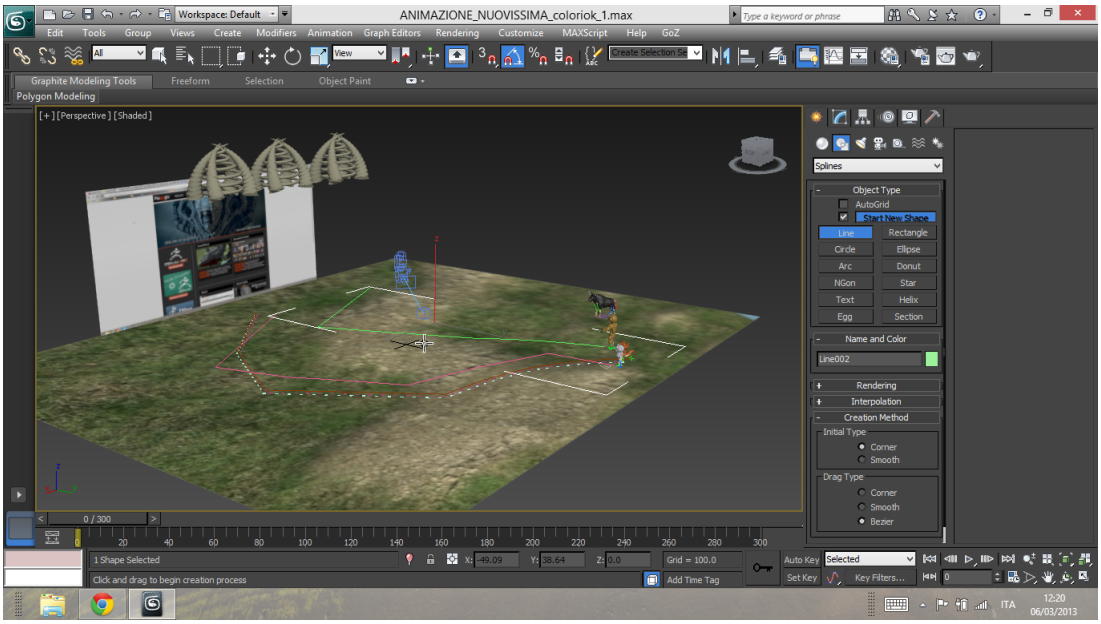


Figura 4.8: L' animazione in un determinato frame

Come si può notare dalla figura 4.8 linee differenti corrispondono a path diversi per ciascun personaggio. Ho cercato di mantenere delle traiettore non troppo complesse per non appesantire troppo il lavoro con l' animazione della camera. A questo punto mi son potuto cimentare con il movimento da attribuire alla camera per dare ulteriore dinamismo alla scena.

Per prima cosa è obbligatorio entrare in modalità *Auto key* o *Set key* per generare mano a mano che si va avanti nel nostro *time slider* le *keys* appropriate che permetteranno alla camera di assumere posizioni e rotazioni diverse, a seconda delle scelte dell' animatore. Inoltre, durante la registrazione in modalità Auto key, ho scelto di attribuire un movimento alle gabbie che avrebbero dovuto simulare i browsers hookati.

4.3 Il rendering della scena animata

La situazione generale dell' animazione quindi è risultata la seguente: i browsers scappano per così dire dal toro e si spostano verso una pagina web, come a raffigurare il fatto che stiano operando. Il toro avanza e quando i browsers sono ormai dinanzi alla pagina vengono bloccati con 3 gabbie (si presuppone che il toro beEF abbia dei superpoteri). La camera cerca di descrivere meglio la situazione che si definisce in circa 16 secondi. Avendo definito tutte le componenti della nostra animazione, l' ultimo processo da compiere è ovviamente il rendering, ovvero quell' insieme di operazioni che consente di trasformare un modello tridimensionale (o una scena più complessa chiaramente, come nel mio caso) in una rappresentazione visuale bidimensionale (bitmap).

Un altra questione da affrontare prima del *rendering* finale è la suddivisione nella nostra animazione in frames, o meglio: con quanti frames vogliamo lavorare? Quanto tempo impiegherebbe il mio renderer a generare il file con il video dell' animazione? La risposta a queste domande sta nella

sperimentazione di diverse modalità di rendering, del numero di keyframes che decidiamo di avere, e nella qualità che vorremo avere nella fase finale di renderizzazione.

Figura 4.9: Un frame della scena finale durante la registrazione in modalità Auto key

Il setup del rendering dunque è una fase essenziale se si vuole avere un prodotto finito accettabile. Non mi sono soffermato come avrei voluto su tale aspetto poichè i tempi di rendering di una scena come quella generata nel mio progetto richiedevano 6 ore con impostazioni qualitative medie potremmo dire. La risoluzione del video è stata di 800x600, mentre per quanto riguarda il renderer ho scelto di utilizzare Mental ray: tale strumento avrebbe bisogno di un' analisi a parte molto lunga e prolissa, ma non è stato l' argomento principale della mia tesi come si può dedurre.

La mia animazione dopo diverse prove e numerosissime ore di testing ampliando e diminuendo il numero di frames, è stata realizzata sulla base di 500 frames. Ho scelto questo numero perchè partendo dalla constatazione che con 300 frames avevo un output video di 10 secondi, ho pensato che per

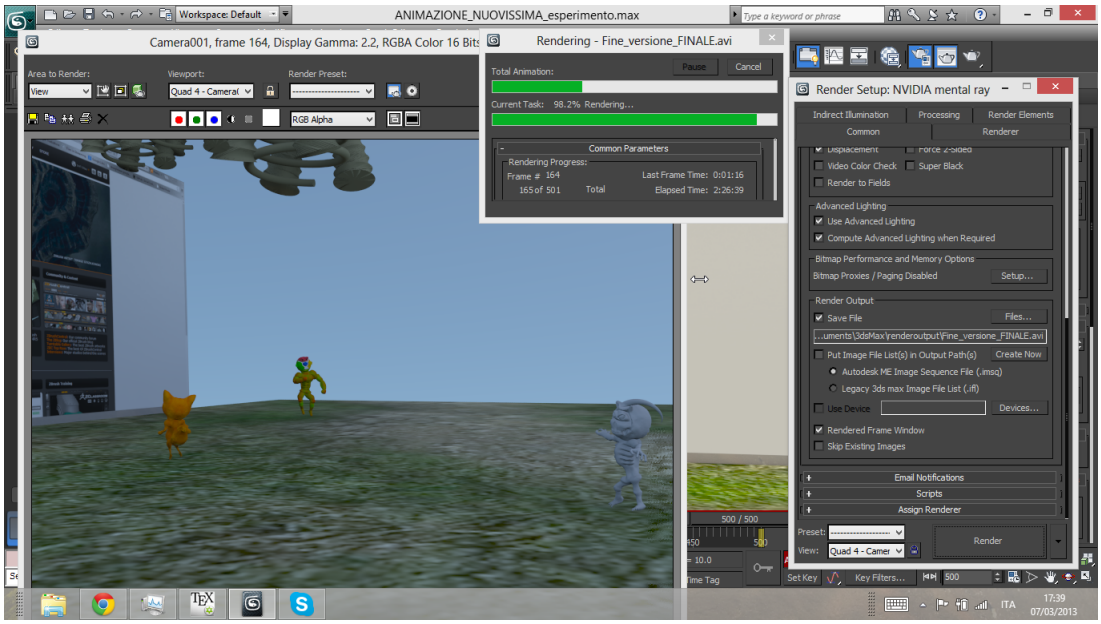


Figura 4.10: Il rendering finale con Mental Ray in fase di avanzamento

una visualizzazione migliore e meno caotica degli elementi della scena potesse aiutare tantissimo avere 6-7 secondi in più di tempo.

Conclusioni

La computer grafica è una branca dell' informatica in continua evoluzione che offre opportunità di crescita professionale sempre più specializzate e interessanti per gli appassionati del settore. E' una disciplina relativamente nuova, che col tempo si è innovata sempre più fino ad acquisire un' importanza fondamentale nel mondo del lavoro e della produttività. L' industria dell' intrattenimento oggi fattura più della televisione e del cinema messi insieme perchè alla base di videogames e prodotti 3d pubblicitari vi è un lavoro dalle immense capacità comunicative. Inoltre ormai la computer grafica in generale è stata inserita all' interno del processo di sviluppo di piccoli o grandi cortometraggi televisivi o interni film 3D. Quindi possiamo dire che all' interno dei più grandi mezzi di informazione un normale osservatore può capire quanto possa essere affascinante il semplice interessamento a questo mondo in continua evoluzione. La computer animation come è stato ripetutamente sottolineato nel corso di questa tesi è la prova di quanto la computer grafica si sia inserita nella vita di tutti i giorni, dai videogames, ai film o agli spot pubblicitari 3d. Ritengo quindi che entrare a far parte di un settore lavorativo in cui la sperimentazione, la creatività e la specializzazione delle singole figure lavorative sia un' opportunità non solo finalizzata al guadagno materiale, ma alla singola esperienza di vita che si possa intraprendere in tal senso. Il lavoro in quanto tale deve essere stimolante, ricco di colpi di scena, stabile e divertente.

Appendice A

Il progetto: fase 1 con Zbrush

A.1 Modelli preliminari con ZSphere e Zsketch

Con la tecnica ZSphere, una volta che inseriamo una prima sfera nella nostra area di lavoro, abbiamo una sfera che assume il ruolo di *root* della catena che verrà successivamente sviluppata intorno ad essa. Infatti ogni altra sfera (*child*) che si disegna sarà connessa alla prima, ovviamente mantenendo particolari accorgimenti quali ad esempio l' utilizzo dello *slider Draw Size*, quindi del nostro puntatore circolare, che per sicurezza va mantenuto a 1, fino ad arrivare quasi ad assomigliare ad un punto, evitando dunque di selezionare altre sfere con il raggio del puntatore. Le connessioni tra le sfere sono propriamente chiamate *joint*.

Realizzata la topologia desiderata, questa subirà un processo denominato adaptive skin, che consiste nella generazione della mesh applicando uno *skinning* alla catena di ZSphere generata precedentemente. ZBrush fornisce una serie di *sliders* per gestire il controllo dell' *adaptive skin*:

Preview .

Density : controlla quante suddivisioni lo skin dovrà avere.

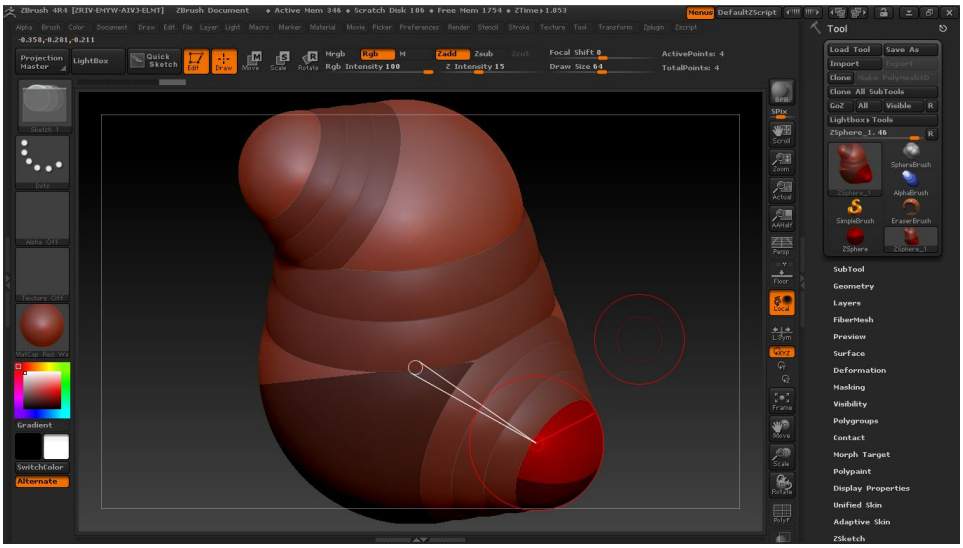


Figura A.1: Esempio di Zsphere: ogni votla che si seleziona una sfera viene visualizzato automaticamente il joint connesso ad essa

G radial : aiuta a monitorare la densità della mesh in certe aree come gli arti.

Max twist : controlla la torsione della mesh. Si sceglie la sfera su cui si vuole modificare la direzione e si possono introdurre valori negativi nello slider.

Proximity tollerance : controlla in che modo la geometria si interseca quando molte ZSpheres children si staccano da un' unica ZSphere parent.

Classical skinning : esiste anche un unico bottone che permette di adottare uno skinning di tipo classico. Funziona bene nel caso in cui si voglia avere maggior controllo sulla mesh generata rispetto alla nuova tecnica ZShere basata sullo scheletro da noi generato. A seconda delle esigenze dell' artista può tornare molto utile.

Esiste anche un altro tipo di *skinning*, chiamato *unified skin*, ma che riguarda più esattamente un' altra tecnica di ZBrush: ZSketch. Nel corso

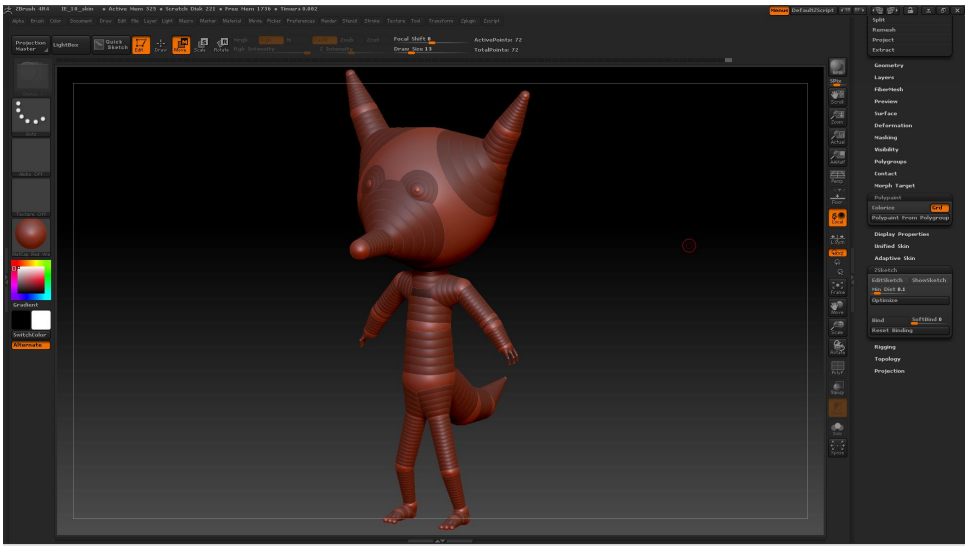


Figura A.2: Il modello di firefox nella sua fase preliminare utilizzando la tecnica Zphere

del progetto di animazione e modellazione ho utilizzato questa tecnica per generare l' armatura del modello di internet explorer e il toro (beef più esattamente). Prima di cimentarmi con l' uso di tale potentissimo strumento ho provato a realizzare i modelli partendo da primitive di base come sfere, spirali 3D, cubi, ma ho notato una perdita enorme di performance in termini di tempo e resa. Il passaggio attraverso la tecnica ZSphere credo che sia d' obbligo quindi per generare una forma preliminare della nostra mesh. L' adaptive skin in pochi secondi (avendo una buona scheda grafica e più ram possibile) elabora le informazioni generando un telo che si va ad applicare alle sfere.
Terminato il lavoro con la tecnica ZSphere, si possono scegliere due strade:

1) Applicare anche la tecnica ZSketch : partendo dall' armatura del nostro modello grezzo finalmente completata on le Zsphere si può attuare un *painting* sopra la superficie delle sfere, come se si stessero aggiungendo strisce di argilla o creta esattamente. Applicando la funzionalità *EditSketch* si può notare subito come l' area dedicare ai vari brushes dell' interfaccia viene convertita in funzione della nuova tecnica

che si sta adottando. Le categorie principali di brushes riguardanti lo Zsketching sono tre: *Sketch brushes*, *Smoothing brushes* e *Manipulation brushes*.

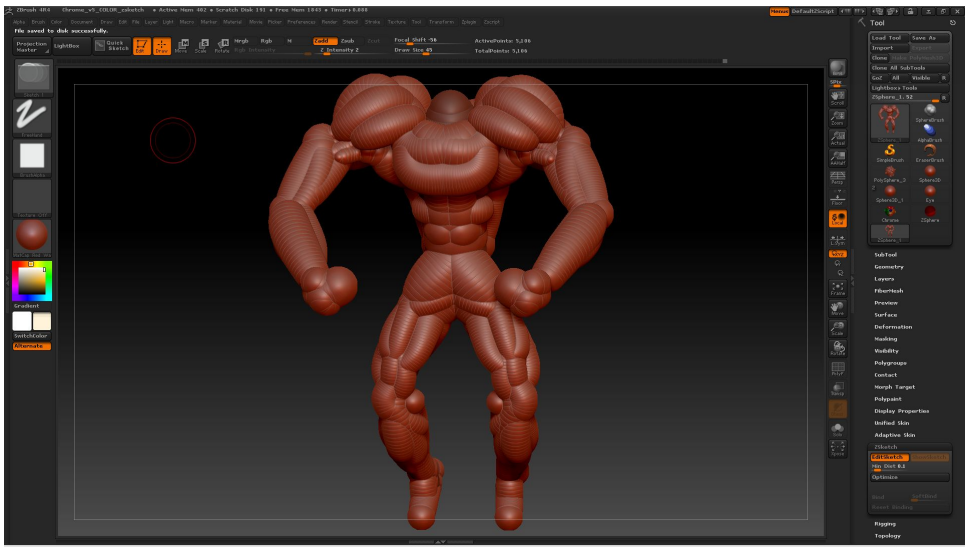


Figura A.3: Chrome dopo lo Zsketch

2) Iniziare lo sculpting : si passa allo sculpting vero disponendo di una vastissimo numero di brushes diversi e sfruttando una potenzialità di ZBrush che da certi punti di vista lo accomuna a Photoshop, la possibilità di lavorare su livelli differenti.

Con questo voglio dire che Zbrush permette di poter suddividere un lavoro di modellazione di grosse dimensioni in modo intelligente e dinamico, consentendo di avere sessioni indipendenti di lavoro focalizzate su determinate parti di una mesh complessa ad esempio. Ciò fu reso possibile a partire dalla versione 3 di ZBrush, che introdusse i Subtools.

Supponendo di aver scelto la prima opzione, quindi aver completato uno Zsketch, si può passare alla creazione della mesh vera e propria, in modo tale che questa possa passare attraverso una fase di sculpting.

Il punto più interessante di utilizzare lo Zsketch a mio avviso riguarda la creatività che questo strumento porta a venir fuori in modo davvero unico. In

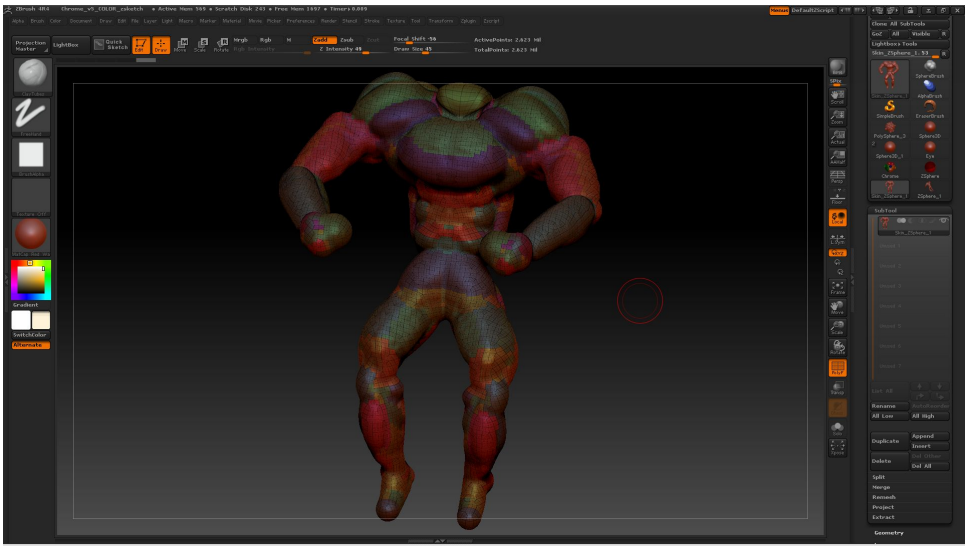


Figura A.4: Chrome skin ad alto numero di poligoni

poco tempo infatti lo stesso modellatore, partendo da un' armatura Zsphere, potrebbe generare più di uno Zsketch proprio per avere tante idee diverse da poter rianalizzare in futuro. Ovviamente bisogna sempre ricordarsi di attuare un *Preview mode* che serve per risolvere ogni problema prima del processo vero e proprio di conversione che il software applicherà.

Una volta che la mesh è pronta si entra in fase di sculpting, facendo uso essenzialmente di due strumenti: i brushes e i subtools.

A.2 Varie tipologie di brushes e utilizzo dei Subtools

Zbrush ha rivoluzionato il mondo della modellazione 3D introducendo i *modellers* al *digital sculpting*. Come è stato più volte evidenziato nel corso di questa tesi, i vecchi metodi adottati dagli altri software 3D basati sulla modifica di singoli vertici, facce e lati poligonali vengono ormai rimpiazzati in Zbrush dall' utilizzo di mesh che vengono modellate sfruttando numerosi tipologie di brushes differenti.

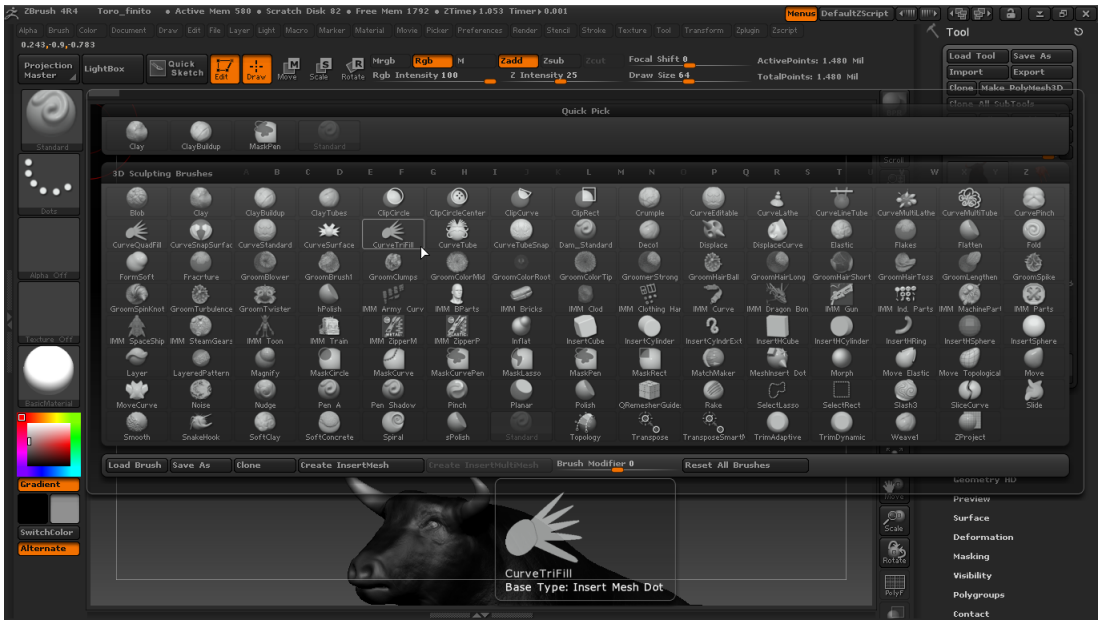


Figura A.5: Brushes disponibili di default

I pennelli che si possono adottare in fase di modellazione sono tanti, e sta a noi scegliere quali prediligere durante il nostro lavoro.

A *run-time*, durante l' utilizzo dei brushes, Zbrush attraverso algoritmi proprietari che purtroppo non sono resi disponibili (dato che il software è closed-source) aggiorna continuamente informazioni sulla direzione di ogni singola faccia poligonale coinvolta nella modifica e sul numero di poligoni all' interno dell' area che si sta manipolando.

Dopo aver pianificato un flusso di lavoro nelle sue fasi essenziali, quindi ad esempio sfruttando altre tecniche come Zsphere e Zsketch, il modellatore si ritroverà ad utilizzare, come nel mio caso, sicuramente tre tra i brushes più importanti:

1) Standard brush : selezionando e trascinando il pennello sulla superficie della nostra mesh poligonale, Zbrush tiene conto della direzione delle normali alle facce selezionate col puntatore circolare, ne fa una media, e sposta tali superfici basandosi sulla media campionata inizialmente.

Con normale si intende una linea perpendicolare uscente da ciascuna faccia poligonale e centrata in essa.

2) Smooth brush : serve a smussare la superficie racchiusa all' interno del raggio d' azione del nostro puntatore. Vale la pena notare come si debba cercare di tenere lo *slider Z intensity* ad un valore tra 20 e 40, in modo tale da non smussare o alterare eccessivamente la mesh durante il lavoro. L' esperienza e le prove con tale brushes servono a capire quanto ci si debba applicare per padroneggiare a pieno tutti i vari tipi di brushes, e questo in particolar modo.

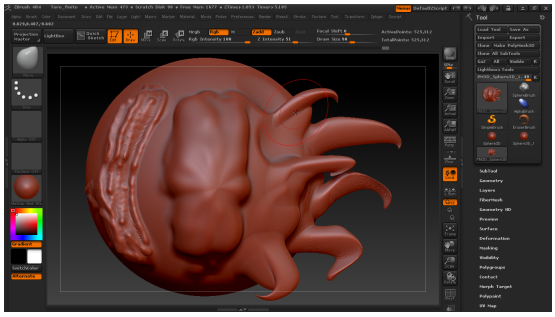


Figura A.6: Da sinistra a destra rispettivamente utilizzo dei tre brushes: Standard, Smooth e Move

3) Move brush : l' azione di questo pennello si potrebbe riassumere con due termini inglesi, ovvero *push and pull*. Muovendo i poligoni verso l' esterno o l' interno della nostra mesh si può velocemente cambiare in maniera più o meno drastica la *shape* della nostra *digital clay*.

Altri brushes che sono risultati estremamente utili al fine dei miei modelli sono stati: Clay, Clay Build Up e Clay Tubes. Con questi pochi brushes credo che ci si possa sbizzarrire in innumerevoli creazioni. Il mio primo esperimento che mi ha permesso di capire a fondo le potenzialità di diversi brushes è stata la generazione di una creatura alata dalle sembianze molto particolari e se vogliamo dire fantasiose.

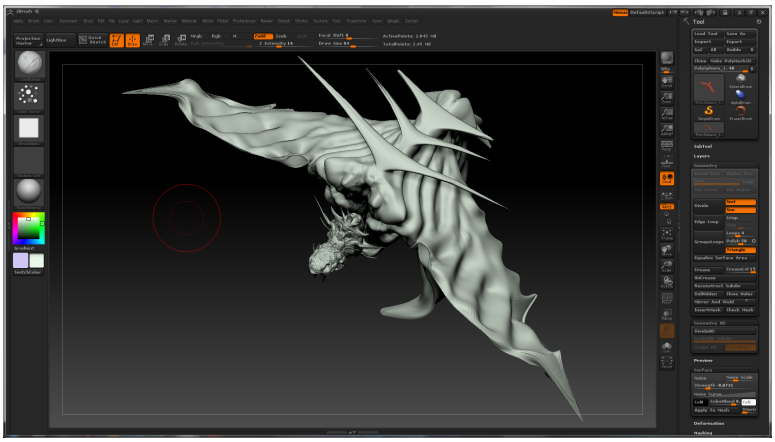


Figura A.7: Primo esperimento con Zbrush e i vari brushes

Durante lo sviluppo dei modelli spesso potrebbe risultare utilissimo sfruttare i Subtools per lavorare su parti diverse della stessa mesh. Nel caso del toro sviluppato durante il progetto, corna e occhi sono stati generati a parte per poi subire un processo di *append* allo stack di subtools dove compariva già la mesh iniziale. Ricorrendo infine ad un *merge* delle varie parti, è semplice ottenere una mesh unica su cui poi poter continuare il nostro perfezionamento del modello.

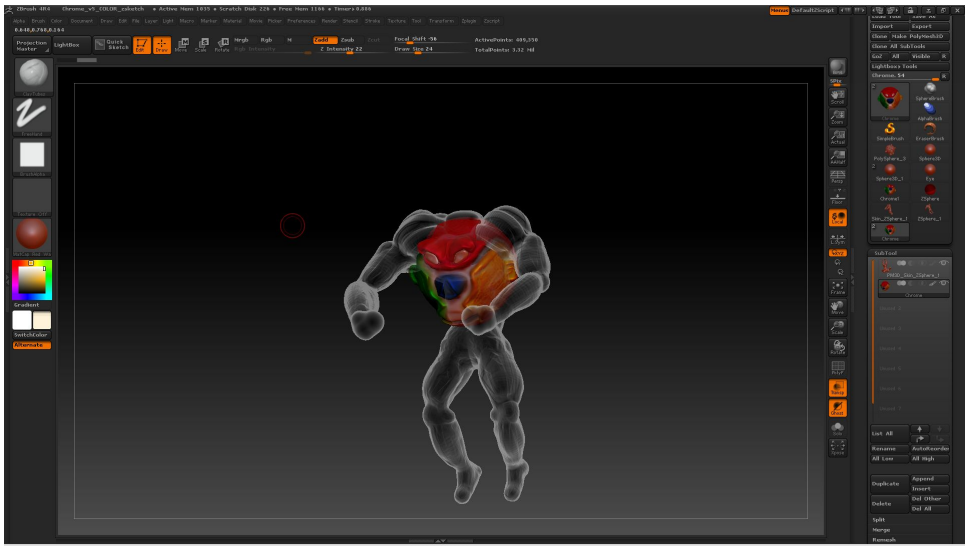


Figura A.8: Chrome durante il merging tra body e testa

Una volta che i modelli sono pronti a livello di forma e dettagli, il passo successivo ma che ho già ampiamente trattato durante la tesi riguarda la decimazione dei poligoni in vista dell' esportazione dei nostri *characters* in 3ds max attraverso Goz. Bisogna chiaramente ricordarsi di sfruttare anche la tecnica del polypainting associata alla generazione automatica della texture a partire dall colorazione dei modelli.

Ovviamente sta al modellatore scegliere se completare anche quest' ultimo step di lavoro e progettazione dei modelli in Zbrush oppure no. Infatti si potrebbe anche optare per uno studio dei materiali in 3d max, ma che sicuramente richiederebbe molto più tempo e a mio avviso non sarebbe comodo al fine di focalizzarsi poi in un' animazione.

Appendice B

Il progetto: fase 2 con 3ds max CAT

Importati i quattro modelli che saranno gli attori della scena da animare, il primo passo da compiere durante il setting dei *characters* riguarda il *CAT rigging*, per poi passare alla generazione dell' animazione sui nostri sistemi di *bones*.

B.1 Il rigging dei quattro modelli

Il *CAT rig* è una gerarchia che definisce il sistema di animazione scheletrico che è CAT. Si tratta come abbiamo già analizzato precedentemente nel corso di questa tesi di un sofisticato *character rig* progettato in modo tale da creare i personaggi che l' animatore vorrà in secondo luogo animare a tutti gli effetti. Il fatto che tale sistema sia prevalentemente scritto nel linguaggio C++ piuttosto che in qualche altro linguaggio di scripting permette di avere una velocità e delle caratteristiche uniche impossibili da ottenere in *scripting code*.

Differenti elementi possono essere aggiunti o rimossi dal rig che si sta definendo fino ad ottenere quello più adatto alle esigenze dell' animatore.

Ciascun *CAT rig* innanzitutto avrà un elemento che costituirà il *character node* del rig: CAT Parent. Tutti gli elementi del rig saranno quindi dipendenti da questo nodo root su cui si basa tutta la struttura gerarchica che verrà costruita.

Come è stato più volte sottolineato, dopo aver scelto di lavorare sulla creazione di un rig da zero oppure sulla modifica di un rig già pronto, il rigger può scegliere la modalità *None* o uno dei vari preset dal *CAT rig load save* menu.

Per quanto riguarda il mio progetto, ho voluto sperimentare entrambi gli approcci al rigging, e per esattezza ho lavorato su tre *custom rig* da zero per i modelli bipedi, mentre per il toro ho scelto di modificare un *preset rig*, quello dello gnu. Dato che non posso mostrare passo passo ogni singola modifica al mio rig, mostrerò tramite alcune immagini i rig che sono andato a costruire sui singoli modelli.

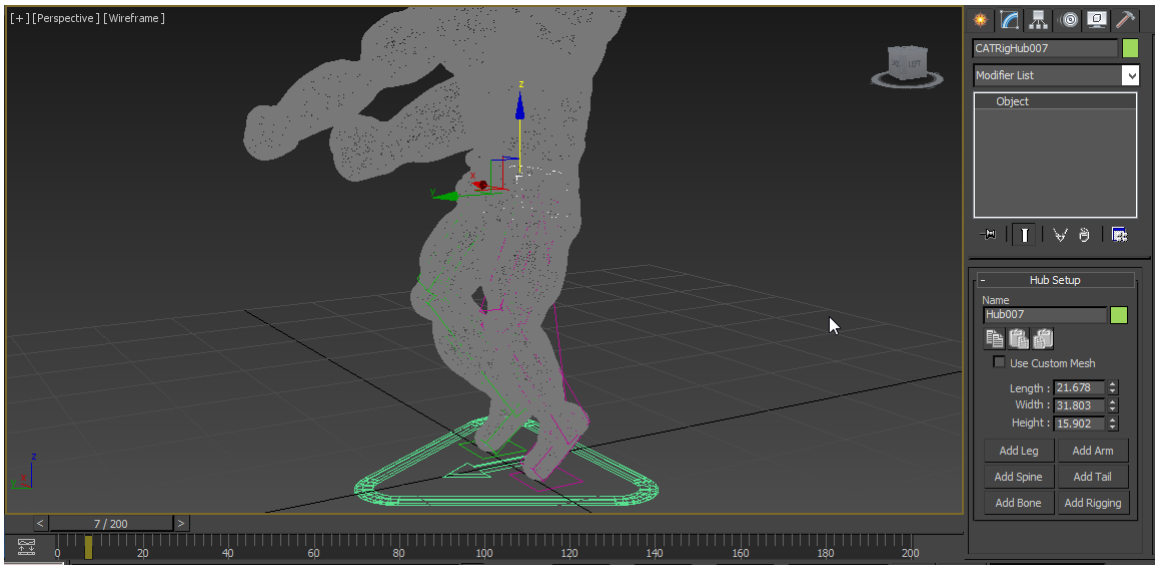


Figura B.1: Chrome durante il rigging da zero

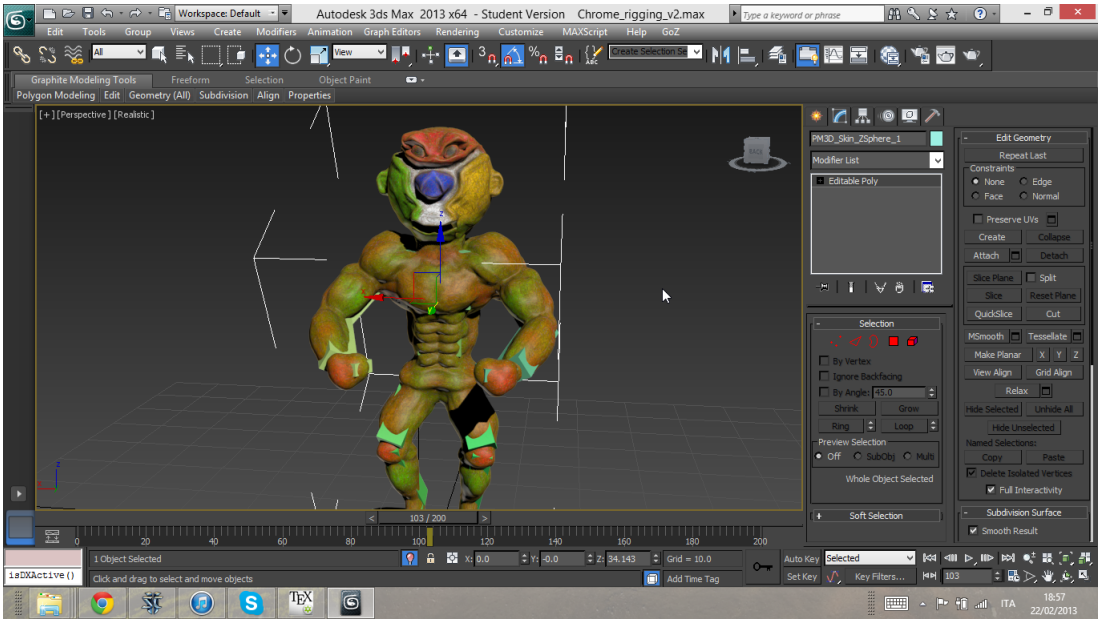


Figura B.2: Chrome skinned rig

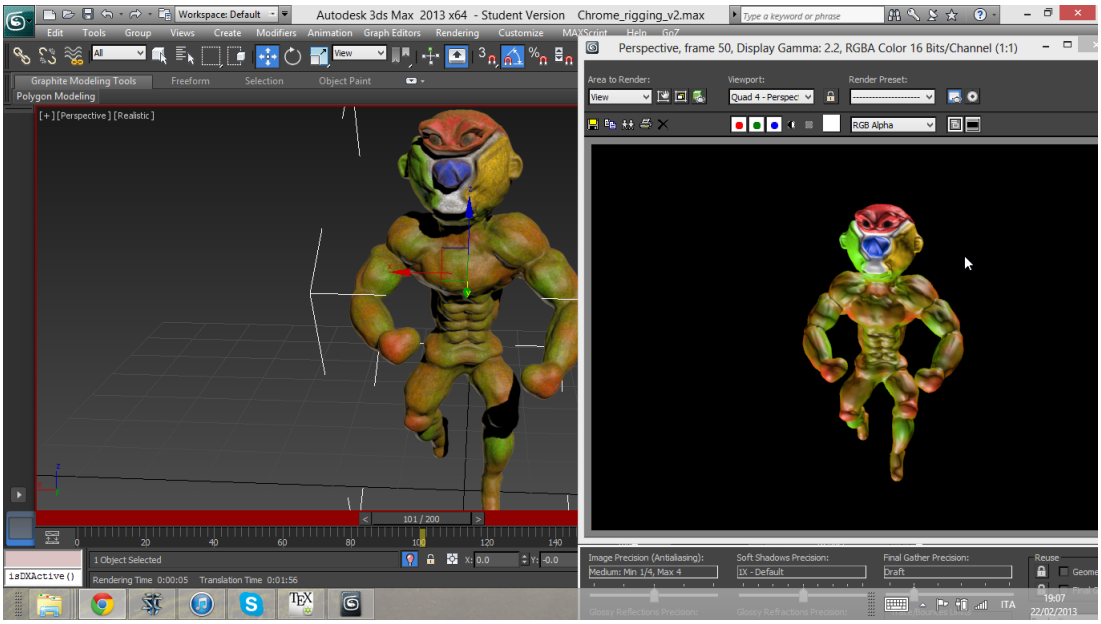


Figura B.3: Chrome rendering dopo il rigging

Bibliografia

[1] Riccardo Scateni Paolo Cignoni Claudio Montani Roberto Scopigno, Fondamenti di computer grafica, McGraw-Hill 2005 tridimensionale interattiva.

[2] Eric Keller, Introducing ZBrush 4, Sybex 2011.

[3] Kelly L.Murdock, Autodesk 3ds Max 2013 Bible, The comprenhensive, tutorial resource, Wiley, 2012.

[4] Richard Lapidus, tradigital 3ds Max, A CG Animator's Guide to Applying the Classic Principles of Animation, Focal Press, 2012.

[5] Scott Spencer, ZBrush Character Creation Advanced Digital Sculpting, 2nd Edition, Sybex, 2011 .

[6] The BeEF Project, http://beefproject.com.

[7] Zbrush offcial site, http://pixologic.com/.