POLITECNICO DI TORINO

Collegio di Ingegneria Informatica, del Cinema e Meccatronica

Tesi di Laurea Magistrale

Modellazione procedurale di ambienti 3D per la realizzazione di un prodotto indipendente di animazione: Reverie Dawnfall

Relatore

Prof. Riccardo Antonio Silvio Antonino

Candidato

Andrea Loro Piana

Aprile 2024

Sommario

Abstract

La seguente tesi prevede lo sviluppo di un progetto di animazione, Reverie Dawnfall, nella quale verrà affrontato la progettazione di un ambiente 3D, il Nightclub.
In particolare, lo studio si concentrerà su tecniche di modellazione innovative che permettano di alleggerire sia il costo computazionale che il lavoro degli environment artists. Più nello specifico si studierà la modellazione procedurale tramite l'utilizzo del software Blender che sta sviluppando la propria suite di modellazione procedurale con i Geometry Nodes.
Questo studio ha anche lo scopo di sperimentare un paradigma di modellazione stilisticamente ed esteticamente accattivante e innovativo, cercando di rendere l'ambiente caratterizzante e distintivo rispetto ad altri prodotti che hanno stesse tematiche e stesso stile.

La tesi sarà composta da una prima parte in cui saranno introdotti i concetti chiave basilari necessari per una più facile comprensione, sia dal lato più gestionale e del workflow, come la differenza tra produzione “Major” e “Indie”, e la diversificazione delle varie fasi di sviluppo di un progetto, che da quello più tecnico come software ed il motore di render utilizzati, spiegandone anche il motivo.

Dopodiché la tesi verterà sulla spiegazione del concetto di modellazione in generale e nello specifico di un ambiente 3D, per poi passare ad analizzare il modello procedurale, dapprima in maniera più generica, e poi passare più nello specifico ai Geometry Nodes, focalizzandosi sugli asset creati.
Durante la spiegazione di questi asset verrà mostrato il processo di sviluppo dall'idea alla realizzazione dell'oggetto, facendo anche un'analisi sull'utilizzo dei geometry nodes rispetto alle tecniche più classiche o ad altri metodi alternativi di modellazione, guardandone i costi e i benefici.

In conclusione, verrà presentato lo stato di avanzamento dell’intero progetto, focalizzando l’attenzione sulla resa estetica finale del progetto.

1. Introduzione

L'uso della computer grafica nei film di animazione rappresenta una rivoluzione epocale nell'industria cinematografica, dando vita a mondi fantastici, personaggi straordinari e storie avventurose che sfidano i limiti della creatività. Questa forma di tecnologia ha radicalmente trasformato la produzione di animazioni, permettendo ai registi di esplorare nuove frontiere espressive e di offrire al pubblico esperienze visive straordinarie.

Grazie alla computer grafica, ogni dettaglio, dalla texture di un singolo poro della pelle di un personaggio alla resa realistica di paesaggi fantastici, può essere plasmato con precisione millimetrica. Questo livello di dettaglio offre ai registi un controllo senza precedenti sulla narrazione visiva, consentendo loro di portare alla vita mondi fantastici o di ricreare ambientazioni storiche con una precisione straordinaria.

Oltre all'aspetto estetico, la computer grafica ha reso più efficienti i processi di produzione. Gli animatori possono manipolare digitalmente ogni singolo fotogramma, migliorando la fluidità delle animazioni e riducendo il tempo necessario per portare a termine un progetto. Questa efficienza ha permesso agli studios di animazione di esplorare una gamma più ampia di idee e di produrre opere di qualità in tempi più brevi.

In conclusione, l'uso della computer grafica nei film di animazione rappresenta una rivoluzione che ha ridefinito i confini della cinematografia. Attraverso l'impiego di tecnologie all'avanguardia, i cineasti possono dare vita a storie più coinvolgenti e visivamente straordinarie, creando un'esperienza cinematografica che va oltre il possibile.

2. Produzioni ‘Major’ vs produzioni ‘Indie’

Major è un termine utilizzato per indicare le principali case di produzione cinematografica che sono grandi, multinazionali e hanno un'influenza significativa sull'industria cinematografica. Esse hanno notevoli risorse finanziarie e sono in grado di produrre film con budget molto elevati. Hanno reti di distribuzione globali e possono investire in marketing su larga scala.
I film prodotti dalle major spesso puntano a un pubblico di massa, e spaziano dai blockbuster di successo commerciale alle produzioni di alto profilo in vari generi.
Mentre per produzioni Indie si intende produzione indipendente, ovvero si riferisce a film realizzati da case di produzione più piccole o da singoli filmmaker autonomi. Quindi hanno budget limitato, per esempio possono dipendere da fonti di finanziamento indipendenti, crowdfunding, sovvenzioni governative o investitori privati. Gli artisti indipendenti spesso godono di una maggiore libertà creativa, possono sperimentare con stili narrativi, temi e approcci innovativi senza essere vincolati dalle esigenze di un pubblico di massa o dalle richieste dei finanziatori.
La distribuzione dei film indipendenti può essere più selettiva e spesso coinvolge festival cinematografici, circuiti indipendenti o piattaforme di streaming online. I film indie possono raggiungere un pubblico più ristretto, ma spesso apprezzano il successo critico. Le produzioni indie sono conosciute per affrontare una varietà di generi e temi, spesso esplorando narrazioni non convenzionali o argomenti di nicchia che potrebbero non essere commercialmente attraenti per le major.

Bisogna anche considerare che molti studi indie nascono da poche persone, magari da realtà giovani nate dagli ambienti scolastici e universitari, ciò comporta che l'ambiente di lavoro ha dimensioni ridotte con minor esperienza e minor specializzazione su uno specifico campo.
Le major sono invece suddivise in sedi, in settori e in dipartimenti, contando un numero anche considerevole di dipendenti e di collaboratori esterni. Il risultato di queste differenze porta ad avere pipeline differenti tra loro.
Nelle major le figure lavorative richieste hanno mansioni molto specifiche, per puntare alla perfezione in ogni settore, inoltre nei vari dipartimenti si possono avere più gradi di specializzazione: entry level, stagisti, tesisti e junior, regular, senior, supervisor e capo dipartimento.
Invece negli studi indie è tutto più ‘mescolato’, questo perché essendo in pochi ci sono meno dipartimenti e normalmente esistono solo quelli principali. Quindi capita che una stessa persona debba gestire più mansioni.

Chi lavora in uno studio indie, quindi, gode di un più ampio aspetto della pipeline di lavoro e può gestire più mansioni in base alle esigenze, questo fa sì che si accumuli molta esperienza.

3. Fasi di sviluppo di un progetto

Generalmente un prodotto di destinato al pubblico, che sia di animazione 3D o un live action, passa attraverso diverse fasi prima di poter essere definito concluso e vendibile. Questi processi vengono raggruppati e divisi in tre fasi principali: la pre-produzione, la produzione e la post-produzione.

La pre-produzione è la fase iniziale di pianificazione e preparazione che precede la produzione effettiva del prodotto multimediale. Durante questa fase, vengono prese decisioni chiave, creati piani e raccolte risorse necessarie per portare avanti il progetto.

Tra le attività tipiche di questa fase si trova lo sviluppo della sceneggiatura o del concept, la pianificazione del budget e delle risorse, la selezione del cast e dell'equipaggio, la progettazione dei set e dei costumi, lo scheduling delle riprese, la ricerca di location ed infine l'acquisizione di attrezzature e la preparazione logistica.

La produzione è la fase in cui il materiale audiovisivo effettivo viene registrato o creato.

Questa fase viene chiamata fase del "punto del non ritorno", in cui tutto ciò che è stato definito nella fase precedente deve essere rispettato fedelmente, anche a livello di scadenze.

É per questo motivo che è importante fare una buona fase di pre-produzione, perché altrimenti bisogna rifare da capo il lavoro, con una perdita di fatica e tempo non indifferente.

La post-produzione è l'ultima fase della linea di produzione, in cui il materiale grezzo creato e catturato viene elaborato, modificato e assemblato per creare il prodotto finale. Durante questa fase, vengono apportate correzioni, aggiunte di effetti visivi e sonori, e si lavora sulla struttura e sulla presentazione complessiva del progetto.

Tra le attività tipiche di questa fase si trova il montaggio delle scene, l'aggiunta di effetti visivi e sonori, il color grading e correzione colore, la creazione e aggiunta della colonna sonora e degli effetti sonori, la revisione e correzione di eventuali errori o incongruenze ed infine la creazione di titoli, sottotitoli e grafiche aggiuntive.

4. Software per la modellazione 3D

Esistono svariati software dedicati alla modellazione ed animazione 3D, ognuno con
i propri punti di forza e le proprie debolezze che rispondono alle diverse esigenze del progettista.

La scelta è ricaduta su Blender [1] innanzitutto per questione di continuità con il lavoro precedente dei nostri colleghi sul trailer della serie.
Altro fattore positivo sulla scelta di Blender è che esso ha continuato a aggiornarsi e a migliorarsi, diventando non solo uno dei software di riferimento per la modellazione e animazione 3D per lavori indipendenti o di studi molto piccoli, ma facendo anche concorrenza ad altri software che sono a pagamento.
I colleghi precedenti hanno scelto di utilizzare Blender principalmente per la natura open source del software, il quale permette di modificare il layout predefinito e di creare Addon (plugin) che permettono di aggregare e integrare funzionalità in base alle proprie necessità in modo che sia tutto più comodo ed efficiente. Questo è incline anche con l'idea della tesi e della serie che trovano le loro fondamenta nella sperimentazione.

Tra i punti negativi di questa scelta è la visione che ha l'utente medio su software open source, ovvero che si tratti di qualcosa non professionale, soprattutto sapendo che i programmi più utilizzati dalle grandi case di animazione internazionali e usati nel mercato cinematografico mondiale sono a pagamento, come per esempio Autodesk Maya.
Un altro aspetto negativo rispetto ad un software a pagamento è che un software open source può avere strumenti considerati acerbi dal punto di vista tecnico, e quindi non pronti per l'uso all'interno di un ambito professionale di alto livello.
Dall'altra parte però è possibile che alcune novità vengano portate prima su software open-source in quanto la sperimentazione e condivisione stessa degli utenti porta ad avere dei plugin sperimentali con una maggiore frequenza rispetto a software a pagamento.

4.1 Blender

Blender è un software di modellazione 3D, animazione, rendering, compositing e editing video open source e gratuito. È una potente suite di strumenti che permette agli utenti di creare una vasta gamma di contenuti, tra cui modelli 3D, animazioni, effetti visivi e altro ancora.

Nasce come prodotto interno della società di animazione olandese NeoGeo, dalla mente di Ton Roosendaal, che nel 1998 fondò la società NaN (*Not a Number Technologies)* per continuare il suo sviluppo e distribuirlo come freeware, cioè come software proprietario a costo zero.

A causa della bancarotta nel 2002, Roosendal creò la fondazione no-profit Blender Foundation, che nell'ottobre dello stesso anno, riuscì a rendere Blender un progetto open source sotto la licenza GNU (General Public Licence) [2].

Negli anni Blender è sempre divenuto più importante e con un numero di utilizzatori sempre più elevato, permettendo anche a quest'ultimi di collaborare per migliorare il prodotto.

Il primo grande progetto professionale nel quale Blender è stato usato come strumento primario è stato la previsualizzazione dell'animatic di Spiderman2 [3].

Da citare anche il film di produzione italiana, Gatta Cenerentola [4] che è stato realizzato interamente con Blender.

4.2 Motori di render

Blender include due motori di rendering in modo nativo: Cycles e Eevee [5], ma supporta anche altri motori come RenderMan, motore di rendering di proprietà di Pixar, ma essi devono essere installati.

Per quanto riguarda ai due motori presenti in modo nativo, entrambi offrono approcci differenti alla generazione di immagini, rendendo Blender un'applicazione versatile adatta a una vasta gamma di esigenze di rendering.

Cycles è un motore basato su Ray Tracing, ovvero simula realisticamente il percorso della luce attraverso la scena. Ciò permette di ottenere riflessioni, rifrazioni e ombre realistiche.

Quindi Cycles è preferibile per progetti che richiedono una qualità fotorealistica, come render di architettura, prodotti o scene cinematografiche.

Ha il problema di essere molto oneroso a livello computazionale e ha tempi di rendering molto elevati.

Mentre Eevee è un motore Real-Time, cioè consente di visualizzare le modifiche nella scena istantaneamente. Questo è particolarmente utile per l'anteprima e lo sviluppo interattivo.
Eevee utilizza la tecnica della rasterizzazione per generare immagini, rendendolo veloce e adatto a progetti che richiedono tempi di risposta immediati. Inoltre supporta ombre volumetriche e effetti di profondità di campo.
Quindi Eevee è ideale per progetti in cui è più importante una resa rapida e una visualizzazione interattiva, come giochi, animazioni in tempo reale o progetti con tempi di produzione più rapidi.

La scelta tra Cycles ed Eevee dipende dalle esigenze specifiche del progetto. Alcuni utenti potrebbero utilizzare entrambi durante il processo di sviluppo, sfruttando Eevee per l'anteprima veloce e Cycles per la resa finale quando è richiesta una qualità fotorealistica. Questi motori di rendering rendono Blender un'applicazione estremamente flessibile e adatta a una vasta gamma di utilizzi nell'ambito della grafica 3D e dell'animazione.

La scelta dei miei colleghi è stata quella di utilizzare Eevee principalmente per il fatto che è più veloce, meno oneroso a livello computazionale, ma con un’ottima resa, inoltre essendo Reverie Dawnfall un progetto di animazione 3D in stile cartoon e non fotorealistico, il costo eccessivo in termini di tempo e costo computazionale di Cycles era abbastanza inutile.

5. La modellazione 3D

La modellazione 3D è il processo di creazione di rappresentazioni tridimensionali di oggetti o scene mediante l'uso di software specializzati. Essa consente di creare un modello virtuale di un oggetto tridimensionale che può essere visualizzato da diverse prospettive. I modelli 3D possono essere utilizzati in una varietà di settori, tra cui l'animazione, i videogiochi, la progettazione industriale, l'architettura, la simulazione e altro ancora. Gli artisti o progettisti 3D utilizzano strumenti software avanzati per creare, manipolare e renderizzare oggetti in modo da ottenere risultati realistici o stilizzati, a seconda delle esigenze del progetto.

Tra le prime rappresentazioni tridimensionali su calcolatore c'è quella realizzata a William Fetter[1] nel 1960, chiamata 'primo uomo' o 'Boeing Man' che rappresenta un insieme di linee che descrivono la sagoma virtuale di un pilota di aereo.

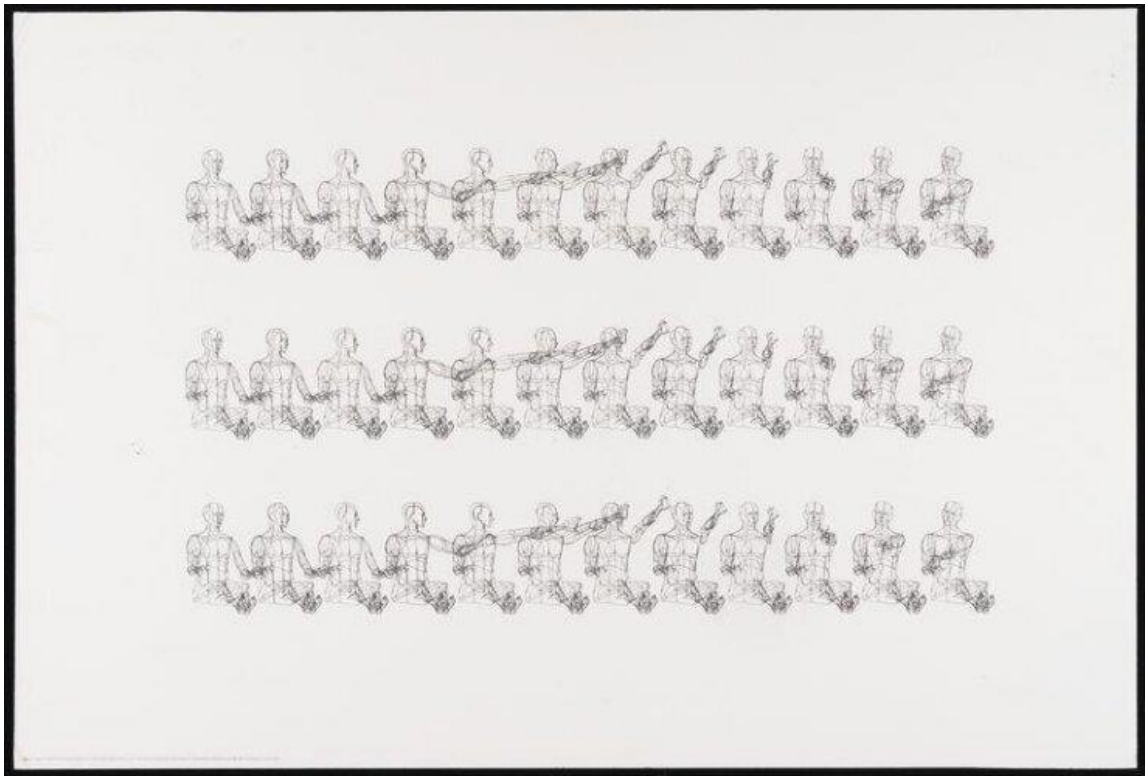


Figura 5.1: Boeing Man di William Fetter, 1960.

Quindi la nascita della modellazione 3D è avvenuta in ambito industriale come supporto alla progettazione.

[1] William Fetter (14 marzo 1928 – 23 giugno 2002) è stato un direttore artistico statunitense e un pioniere nel campo della computer grafica.

Ad oggi invece i sistemi di modellazione 3D vengono ora impiegati nei più svariati ambiti della computer grafica, da applicazioni a carattere medico, scientifico o tecnico (come nelle scienze ingegneristiche, nelle scienze applicate e nell'architettura) fino alle applicazioni puramente artistiche come nell'industria cinematografica e televisiva, nell'industria pubblicitaria e nel web design e infine nell'industria videoludica.

Dal punto di vista tipologico, la modellazione 3D si può dividere in due gruppi: la modellazione organica e quella geometrica. Quest'ultima è il primo tipo di modellazione creata, utilizzata per realizzare qualsiasi cosa che abbia una natura artificiale, mentre la modellazione organica viene utilizzata per esempio per realizzare characters o soggetti 'naturali' come piante, rocce o qualsiasi oggetto che abbia forme morbide e naturali.

5.1 Modellazione di ambienti 3D

Nelle produzioni non è da sottovalutare la scenografia, che permette di dare espressività alla storia.

Questo concetto viene ripreso anche nelle produzioni 3D dove l'ambiente e gli oggetti di scena vengono modellati per descrivere al meglio la storia del progetto che si sta realizzando, sia che si tratti di un film di animazione, di uno spot pubblicitario o di un videogame. Tali ambienti (environments) possono essere costituiti da centinaia di assets e insieme agli oggetti di scena (props) sono importanti tanto quanto i personaggi.

Questo lavoro descritto viene normalmente fatto da una persona o gruppo di persone competenti che prende il nome di Environment Artist, il cui scopo è quello di creare il senso di immersione, tramite la creazione di ambienti 3D in sintonia con la sceneggiatura della storia.

Gli Environment Artists partono dall'idea scaturita durante la pre-produzione da parte dei Concept Artists, figure artistiche che si occupano di definire lo stile degli ambienti, oggetti e personaggi ovvero il concept design della storia. Soltanto dopo aver deciso lo stile si può procedere con la fase di modellazione.

Tale fase prevede l'uso di varie tecniche e tools che gli artisti sfruttano per realizzare nel modo più coerente possibile una versione tridimensionale del prodotto creato dai Concept Artists.

La tridimensionalità e la geometria vengono resa vive grazie allo shading: fase in cui si definiscono materiali e texture degli oggetti.

Modellando gli oggetti di scena, è importante capire la loro collocazione nell'ambiente 3D rispetto alle telecamere, in quanto è una perdita di tempo la creazione di piccoli dettagli che poi non saranno visibili all'utente finale se la telecamera non inquadra o è lontana da tale oggetto.

6. Modellazione Procedurale

Il metodo procedurale è un approccio alla creazione di contenuti digitali, usata per la modellazione 3D, ma anche per attività come grafiche e texture, utilizzando procedure o regole algoritmiche invece di creare manualmente ogni singolo elemento. Questo metodo offre diversi vantaggi in termini di flessibilità, efficienza e variabilità.

Tra i concetti chiave della modellazione procedurale si hanno procedure e algoritmi, variabilità, riproducibilità, efficienza e scalabilità.
Per quanto riguarda procedure e algoritmi, essi permettono di generare dati in modo automatico, mentre la variabilità è la capacità di generare una vasta gamma di risultati variando i parametri delle procedure. Questo permette di ottenere una maggiore diversità e realisticità nei contenuti generati.
Invece per riproducibilità si intende che gli stessi risultati possono essere riprodotti fedelmente regolando i parametri delle procedure, garantendo coerenza nel processo di generazione.
La modellazione procedurale può essere più efficiente rispetto alla creazione manuale, specialmente quando si tratta di generare grandi quantità di contenuti. Gli algoritmi possono essere eseguiti in modo rapido per creare o modificare elementi in modo automatico.
Infine, la scalabilità, ovvero può essere applicata a diverse dimensioni di progetto, da dettagli microscopici a vasti paesaggi.

Esempi comuni di modellazione procedurale includono la generazione di terreni, città virtuali ed edifici. Questo approccio è ampiamente utilizzato nei settori del cinema, dei videogiochi e della grafica computerizzata per creare ambienti complessi e realistici in modo efficiente.

L'idea di utilizzare questo metodo è avvenuta per poter gestire in modo intelligente diverse questioni come il trovare l'estetica finale, che viste le molte incertezze, ha portato a procedere con un metodo che consentisse di massimizzare le possibilità creative, ma permettendo di creare un ambiente in grado di essere utilizzato dai futuri dagli artisti-colleghi, senza che quest'ultimi dovessero perdere tempo a studiarsi tutti i nodi usati per la creazione degli asset.
Inoltre, con questo metodo è possibile riutilizzare lo stesso asset con delle piccole modifiche per posizionarlo in altre ambientazioni.

6.1 Metodo Procedurale

In informatica, una procedura è una sequenza di istruzioni o passi organizzati in modo logico e coerente, scritti in un linguaggio di programmazione, che viene eseguita per compiere una specifica attività o risolvere un particolare problema. Le procedure sono comunemente utilizzate per organizzare il codice in modo modulare e per suddividerlo in unità più gestibili.

```
Type NomeProcedura(Lista input)
    //Dichiarazioni e Istruzioni
  return Output;
```

Da questa definizione si possono estrapolare alcuni punti chiave che fanno delle procedure un importante paradigma di programmazione: la programmazione procedurale.
I più importanti sono il riutilizzo del codice e l'astrazione. Il primo permette di definire una procedura una sola volta e poi richiamarla ogni volta che è necessario eseguire un determinato insieme di istruzioni. Questo contribuisce a rendere il codice più efficiente e manutenibile.
Mentre per astrazione si intende il nascondere i dettagli implementativi all'esterno della procedura stessa. Chi utilizza la procedura non ha bisogno di conoscere i dettagli interni, ma solo di comprendere come utilizzare la procedura attraverso l'interfaccia che essa fornisce: ovvero necessita di conoscere solo i parametri di ingresso (input), che sono valori che vengono passati alla procedura quando viene chiamata e consentono di rendere le procedure più flessibili, in quanto possono agire su dati diversi a seconda dei valori passati, ed il valore di ritorno (output) che consente di comunicare il risultato della procedura e fornisce i dati all'ambiente chiamante.

Adesso che è stato spiegato il concetto di procedura e del metodo procedurale, si può affrontare le modalità con cui vengono affrontati i problemi risolvibili con tale metodo.
Partendo da un problema, che nel nostro caso sarà la creazione di un modello 3D di un famoso edificio, si possono individuare due fasi principali che hanno somiglianze con il metodo scientifico:
l'analisi e la sintesi. Questi passaggi sono spesso interconnessi e si verificano in modo iterativo durante il processo di risoluzione del problema. L'immagine seguente descrive proprio questi processi, i quali verranno subito spiegati.

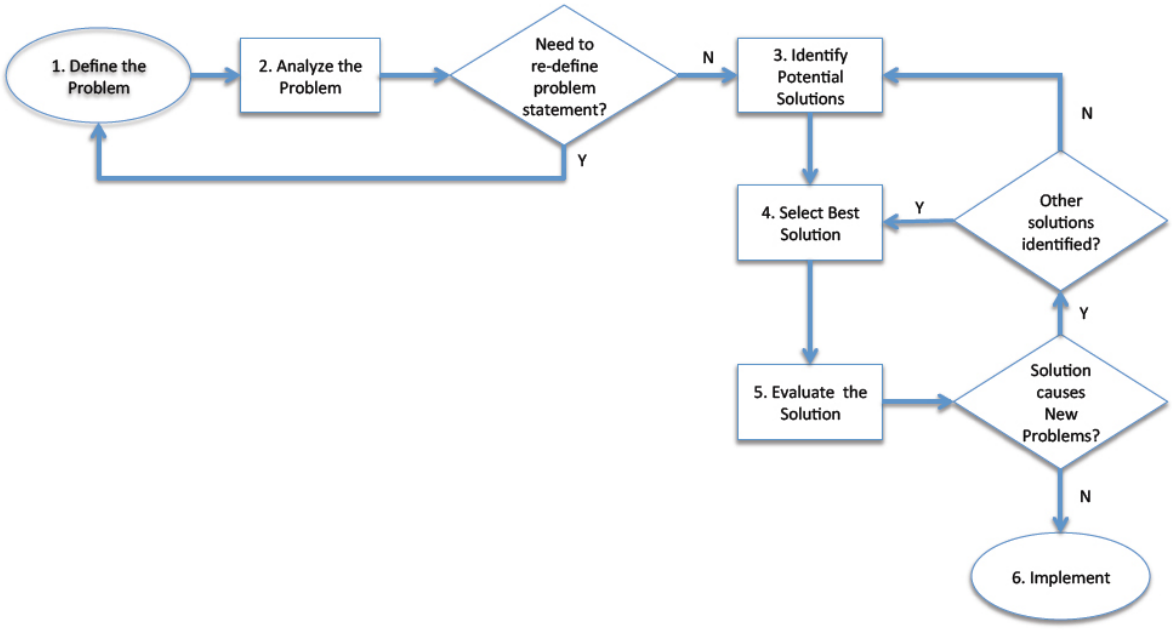


Figura 6.1: Flowchart di problem solving

La prima fase è fondamentale per far in modo che la soluzione sia efficace ed efficiente ed è costituita da diverse sottofasi: la definizione del problema, essenziale per comprendere appieno il problema, la scomposizione del problema in sotto-problemi più gestibili per comprenderne le relazioni e le dipendenze tra di essi. Questo aiuta a focalizzare l'attenzione su singoli aspetti del problema alla volta.
Dopodiché si ha la raccolta di informazioni per comprendere meglio il contesto e i fattori che influenzano il problema, e infine l'analisi dei requisiti fondamentali e le condizioni che devono essere soddisfatte per risolvere il problema.
I parametri di input possono essere eterogenei, come parametri necessari per una soluzione veloce e flessibile, ma anche parametri come modifiche da parte del cliente che commissiona il lavoro.
L'idea è quella che definendo i parametri di input si ottenga il risultato atteso, ma anche che cambiando tali parametri si possano ottenere delle soluzioni alternative, altrettanto valide, e quindi poter scegliere quella che si addice meglio al nostro obiettivo.

La seconda fase viene chiamata fase di sintesi, ed è quel momento in cui si creerà il risultato finale.
Questa fase prevede innanzi tutto la generazione di possibili soluzioni. Questo può coinvolgere la creatività e la generazione di idee innovative.
Dopodiché si passa alla valutazione delle soluzioni proposte, in base ai criteri definiti durante l'analisi. Questo può coinvolgere la valutazione della fattibilità, dell'efficacia e di altri fattori rilevanti.

Dopo l’attenta valutazione delle possibili soluzioni bisogna scegliere la soluzione che sembra essere la più adatta per risolvere il problema in modo efficace ed efficiente, cioè quella ottimale.
Infine, si ha la sintesi finale dove si integra la soluzione selezionata e si sviluppa un piano d'azione dettagliato per implementare la soluzione.

È importante notare che il processo di risoluzione dei problemi può essere iterativo. Durante l'implementazione della soluzione, potrebbero emergere nuovi problemi o sfide che richiedono un ritorno alle fasi di analisi e sintesi. Inoltre, il feedback derivante dall'implementazione può informare ulteriori miglioramenti o adattamenti alla soluzione proposta.

6.1.1 Esempio pratico: Anfiteatro Romano

Di seguito si vuole presentare un piccolo esempio per chiarire meglio il concetto di proceduralità nella modellazione 3D. Si ipotizzi di voler realizzare un modello 3D di un Anfiteatro Romano, per esempio l'Arena di Verona e si analizzi la sua struttura.

Dall'immagine seguente si notano subito molte simmetrie, infatti lo scheletro dell'Arena è costituito da tre gallerie ellittiche concentriche, formate da un susseguirsi di muri e volte. Si nota anche che esternamente l'anfiteatro di Verona era costituito da tre ordini di archi sovrapposti nell'arcata più esterna, mentre in quella intermedia da due ordini di archi. L'altezza degli archi diminuisce man mano che si sale.

Infatti, si passa da un'altezza massima di 7 metri per l'rodine inferiore, a quella centrale di 6 metri fino a 4,5 metri per quella superiore. Quindi tradotta in modellazione, risulta una scalatura sull'asse z dell'intero ordine rispetto a quello sottostante.

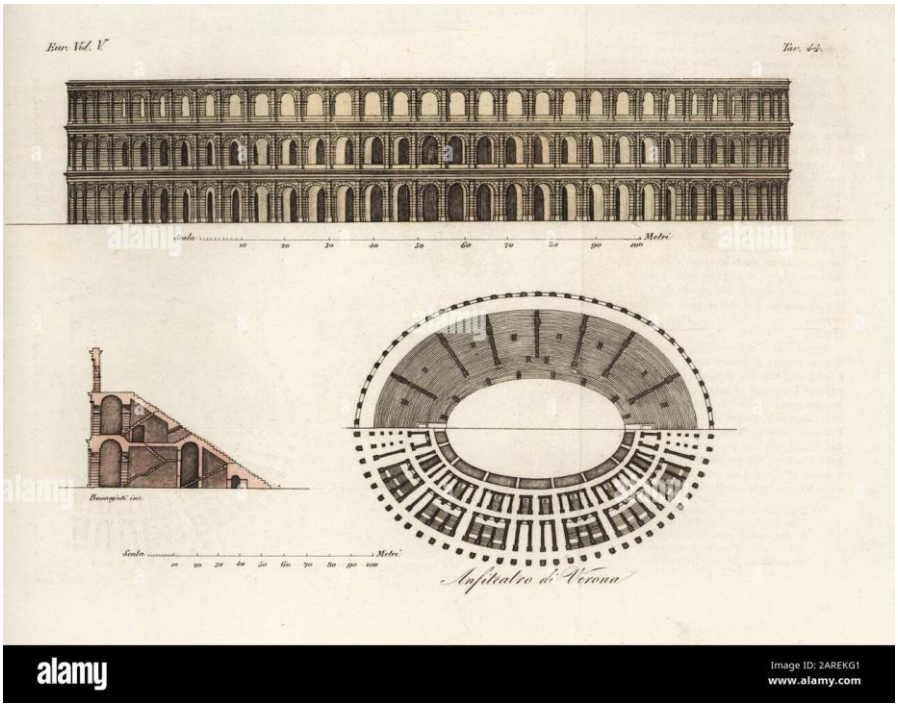


Figura 6.2: Struttura dell'Arena Di Verona

Traducendo questa analisi in metodo procedurale, partendo da una visione minuziosa e specifica, allargandosi fino ad una generale e macroscopica, si ha che, come primo lavoro, bisogna costruire la singola arcata, dopodiché con la loro duplicazione per n volte si crea l'ordine. Quest'ultimo poi viene anche lui duplicato, ma questa volta viene scalato sull'asse z, riducendo così la sua altezza e viene posto al di sopra di quello creato precedentemente; infine duplica un'ultima volta il tutto e si scala per creare le tre gallerie ellittiche concentriche.

Nella fase di analisi è importante studiare un livello di flessibilità del nostro modello che comporti sia una facilitazione nella costruzione dello stesso, ma che al tempo stesso non vada troppo a semplificarlo. L'obiettivo, quindi, è rendere statici gli elementi che si pensa non necessiteranno di modifiche.

6.1.1.1 L'arcata

Partendo dalla simmetria più specifica e muovendosi verso quella più generale, si ha come prima parte da modellare l'arcata, come quella riportata dalla figura.

Figura 6.3: Struttura di un arco

Se si volesse realizzare una procedura in pseudocodice per modellare l'arco, bisogna definire le variabili di input, che sono l'altezza della colonna, *heightColumn*, e la larghezza dell'arcata, *widthArc*.

Qui di sotto lo pseudocodice associato all'arcata:

```
Arcata newArc (heightColumn, widthArc) {
        // Generazione dell'arcata dati gli input
        return arch;
}
```

Modificando questi valori di input, il risultato sarà diverso, ovvero si potranno ottenere diverse varianti dello stesso.

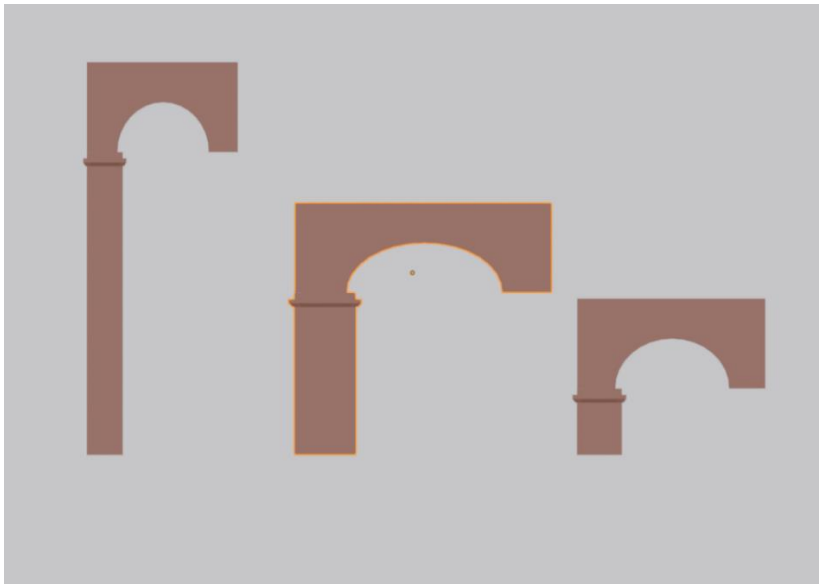


Figura 6.4: Alcune varianti delle arcate ottenibili variando gli input dell'algoritmo

6.1.1.2 L'ordine

Il singolo ordine dell'Anfiteatro viene composto da una *n* arcate, in successione, disposte lungo un'ellisse.

Data l'equazione di un'ellisse generica con centro nell'origine, vedasi sotto, in base ai parametri a e b, corrispondenti rispettivamente ai semiassi orizzontale e verticale, possono trovarsi differenti composizioni per la creazione dell'ordine.

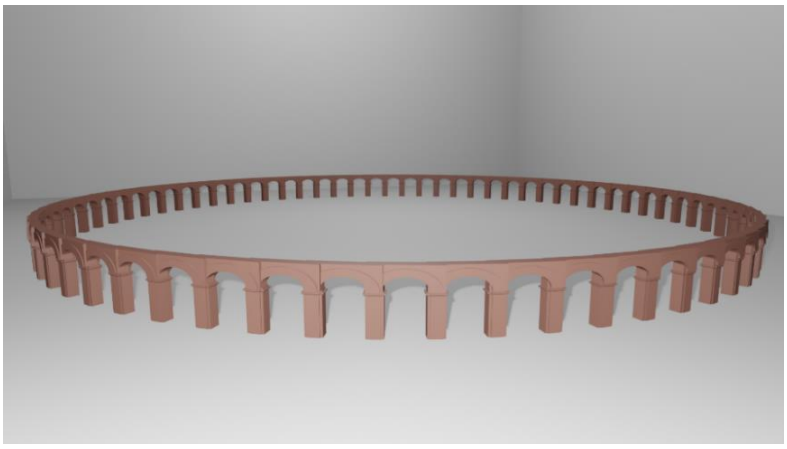


Figura 6.5: Struttura di un singolo ordine

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \; dove \; a \neq 0, \; b \neq 0$$

Quindi in pseudocodice l'algoritmo avrà come dati di input i parametri a e b, il numero di arcate, *numArc* e all'altezza delle colonne, *heightColumn*.

Ordine NewOrdine(*a, b,* numArc, heightColumn) {

// Dato il numero di arcate e i parametri a e b si calcola il valore della larghezza

// dell'arcata che sia corretta per essere disposte lungo l'ellisse scelto.

Arch = newArch (heightColumn, widthArch)

// Genera l'ordine partendo da un insieme di moduli arcata

return ordine;

}

Anche in questo caso se si Modificano i valori di input, il risultato sarà diverso.

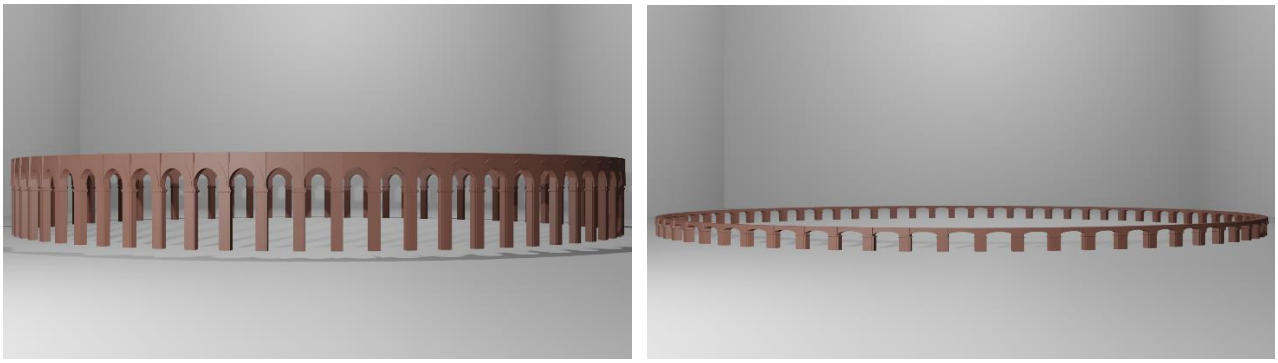


Figure 6.6: Alcune varianti degli ordini ottenibili variando gli input dell'algoritmo

6.1.1.3 L'arena

Ora che si è costruito un piano, per creare tutto l'Anfiteatro basta duplicare l'ordine, posizionarlo sopra a quello originale e scalarlo in altezza abbassandolo, così si crea la galleria centrale.

Invece per quello che rimane della galleria più esterna, si riesegue l'operazione appena citata, riscalando ancora sull'asse z rimpicciolendolo ancora un po'.

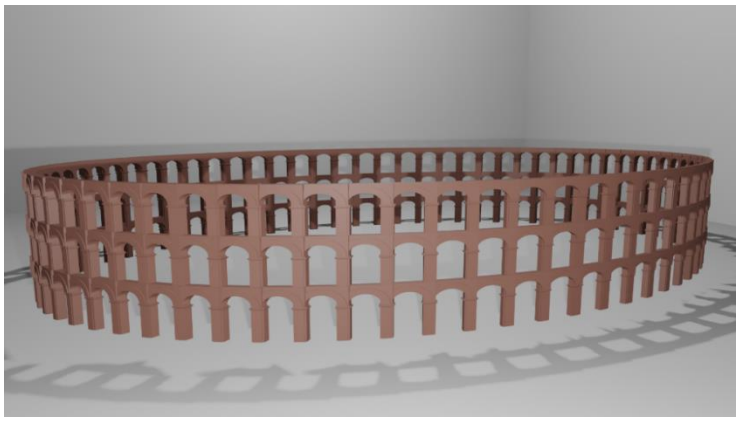


Figura 6.7: Struttura dell'Arena di Verona

In questo caso le uniche variabili caratteristiche della funzione di pseudocodice sono il numero di piani e i fattori di scalatura.

```
Floors NewArena (numFloors, scale1, scale2, a, b, numArc, heightColumn) {
        ordine = Ordine NewOrdine (a, b, numArc, heightColumn)
        // Genera l'arena partendo da una sovrapposizione dei moduli ordine sull'asse
        // verticale Z, scalando per il secondo ordine con scale1 ed eventualmente
        // per il terzo ordine con scale2.
        return arena;
}
```

Quindi definendo le variabili *numFloors, scale1, scale2, a, b, numArc* e *heightColumn* nel modo opportuno, sarà possibile generare facilmente in modo procedurale un anfiteatro romano e nello specifico di questo caso l'Arena di Verona.

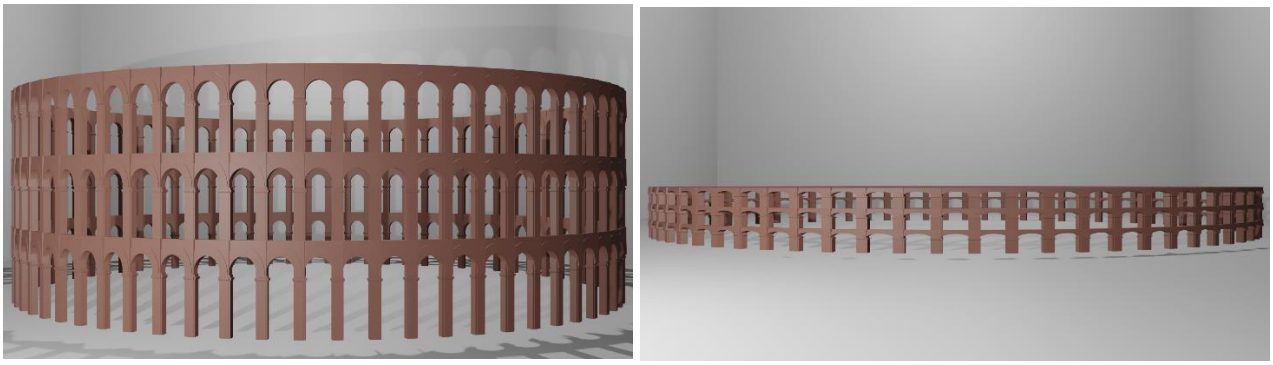


Figura 6.8: Alcune varianti dell'Arena ottenibili variando gli input dell'algoritmo

Questo perché la programmazione procedurale permette di scrivere funzioni che, in questo caso sono descritte in pseudocodice, richiamano quelle definite precedentemente. Tutto questo grazie al lavoro di analisi che ha permesso di individuare un modulo semplice ed elementare dal quale partire.
Il vantaggio di questo tipo di analisi è che se apporteremo delle modifiche ai vari valori di input, questo sarà possibile senza dover ripartire da zero con la modellazione.

7. Modellazione Procedurale in Blender

In un software di modellazione, come Blender, il metodo procedurale può essere svolto in due modi: tramite programmazione o tramite strumenti a nodi.

Nel primo caso Blender utilizza il linguaggio di programmazione Python, con una API specifica, chiamata bpy. Questa permette di creare qualsiasi script facilitando la modellazione di un oggetto, permettendo che azioni ripetute vengano fatte dagli script, o creando oggetti con una randomicità in alcune sue parti.

Nel secondo caso, invece si utilizzano degli strumenti a nodi, al di sotto della quale sono state programmate funzioni apposite, che permetto tramite collegamenti di creare forme, geometrie e oggetti. Questi strumenti in Blender vengono chiamati Geometry Nodes [6].

Ora introduciamo i concetti chiave dei Geometry Nodes per una miglior comprensione del lavoro svolto.

7.1 Nodo:

Un nodo è un elemento grafico che rappresenta una funzione o una trasformazione specifica, può essere legato ad altri nodi attraverso dei socket di input, campi o valori regolari, ricevendo informazioni specifiche dai nodi precedenti, che vengono elaborate e trasferite ai nodi successivi tramite campi di output. Alla fine, si collegano ad un nodo di output per poter visualizzare il risultato nell'ambiente 3D.

7.2 Campo:

Il campo [7] è fondamentalmente una funzione: un insieme di istruzioni che possono trasformare un numero arbitrario di input in un singolo output.

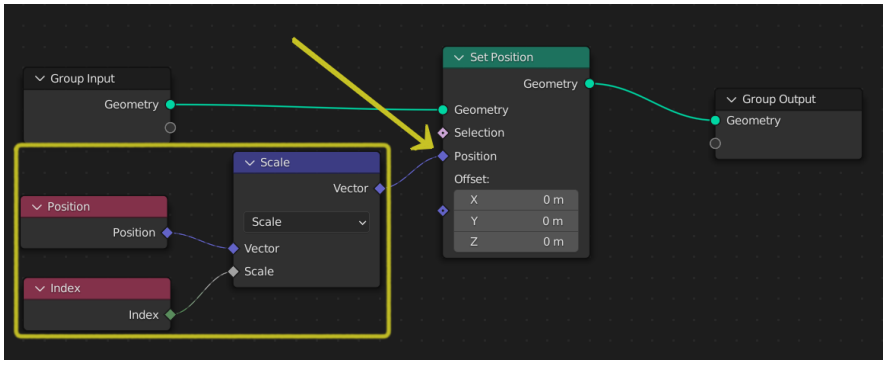


Figura 7.1: Un campo di input inserito in un nodo

Ad esempio, nella figura di sopra, il campo collegato al nodo "Set Position" dipende da due ingressi, Position e Index, e li trasforma in un vettore tramite un'unica istruzione.

Le forme dei socket vengono utilizzate per indicare quali socket sono campi e quali sono dati regolari. Si ha il cerchio per un singolo valore reale con emissione di un singolo valore, il diamante per un campo o per un singolo valore costante, ma l'uscita non varierà per elemento. Infine il diamante con punto per un campo, ma attualmente è un valore singolo, cioè tiene traccia del punto in cui vengono calcolati i singoli valori, anziché un campo con molti risultati diversi.

I nodi possono essere separati in due categorie: nodi di flusso di dati e nodi di campo. I primi passano la geometria e modificano i dati geometrici, mentre i secondi operano sui dati. Questi ultimi possono essere divisi in altrettante due categorie: nodi di input che devono essere valutati nel contesto di un nodo di flusso, per restituire effettivamente un valore introducendo dati geometrici nell'albero dei nodi, oppure nodi funzione, con input, output e socket a diamante, che assomigliano alle istruzioni che verranno valutate dai nodi di flusso. Esempi di nodi funzione sono i nodi matematici.
Il contesto su cui lavorano i nodi solitamente è costituito da un tipo di componente geometrico e da un dominio di attributi che determina quali dati vengono recuperati dai nodi di input. L'albero dei nodi del campo generato verrà valutato per ogni nodo del flusso di dati, potenzialmente recuperando i dati da una geometria diversa o modificata.

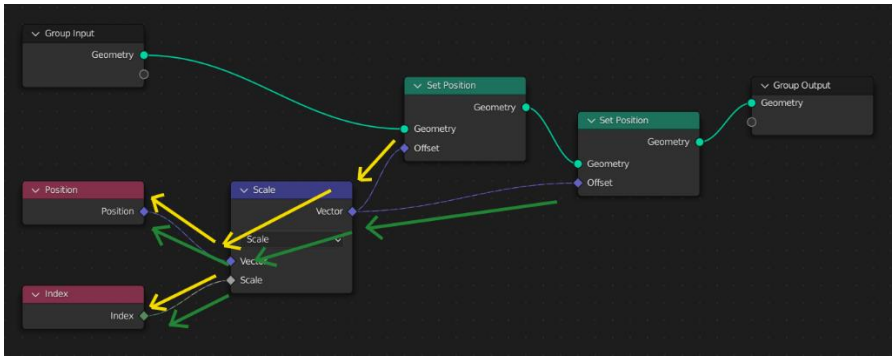


Figura 7.2: Esempio di come viene valutato un albero di nodi

Nell'immagine sopra quando viene aggiunto un secondo nodo Set Position, lo stesso albero dei nodi di campo viene valutato due volte, una per ciascun nodo del flusso di dati. Nel secondo Set Position, i risultati saranno diversi poiché il suo input geometrico avrà già la posizione modificata dal primo nodo.

7.3 Attributo:

Un attributo [8], invece, è il dato memorizzato in un blocco dati. Ad esempio, a ogni vertice può essere associato un vettore o un numero. Essi possono essere di vario tipo, tra cui: Boolean, Integer (32 bit), 8-bit Integer, float, Vector (3D), 2D Vector (UV maps), Color, Byte Color, Quaternion.
Gli attributi possono essere modificati collegando un valore al nodo Group Output, ma anche attraverso l'uso di altri nodi. Tali attributi lavorano all'interno di un proprio dominio che definisce come possano essere interpolati e utilizzati e possono essere visualizzati nello Spreadsheet Editor.

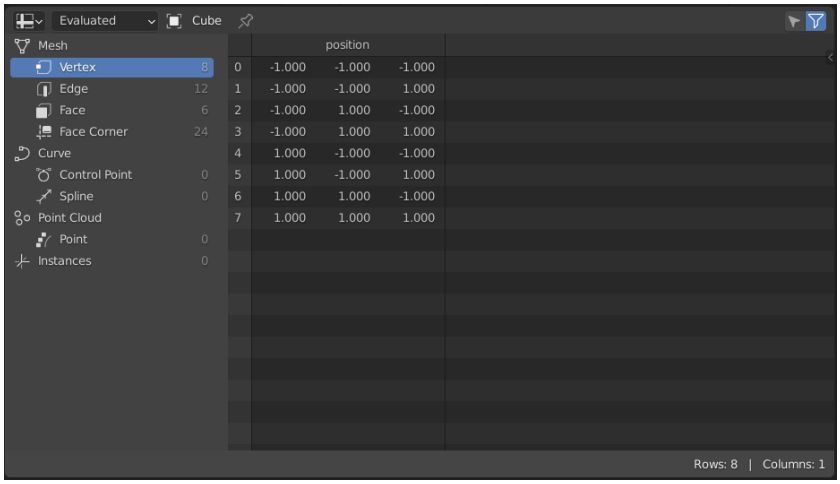


Figura 7.3: Spreadsheet Editor

Tra questi domini ci sono quelli Point, Edge, Face, Face Corner, Spline e Instance, che vengono automaticamente interpolati con altri domini. Ad esempio, quando il nodo Position viene collegato all'attributo di input del nodo Set Material, i valori vengono interpolati dal dominio Point al dominio Face.

7.4 Instances:

Oltre a memorizzare dati reali come una mesh o una curva, una geometria può memorizzare istanze [9], che a loro volta possono fare riferimento a più geometrie, a un oggetto o a una raccolta. Lo scopo dell'istanziazione è quello di consentire di includere molte più geometrie nel risultato, senza duplicare i dati effettivi, e creando anche istanze nidificate: le istanze possono memorizzare una geometria e quest'ultima può contenere istanze.
Questo perché motori di render come Cycles gestiscono meglio gli stessi dati geometrici ripetuti in molte posizioni rispetto a quando i dati vengono duplicati. Quello che rende utile questo metodo è quello di

limitare le mesh uniche che comporta migliori prestazioni nella fase di rendering. Questo però significa che il risultato di un'operazione sarà lo stesso per ogni istanza di una determinata geometria. Per avere risultati univoci per ogni istanza, è possibile utilizzare il nodo *Realize Instances*.

7.5 Node type:

Ora che si sono spiegati i concetti fondamentali dei Geometry Nodes, si mostreranno i nodi più importanti e più utilizzati per questo progetto, in quanto presentarli tutti sarebbe un processo che non arricchirebbe il lavoro svolto. Innanzitutto, i nodi si dividono in diverse categorie [10], le principali sono: Attribute Nodes, Input Nodes, Output Nodes, Geometry Nodes, Curve Nodes, Instances Nodes, Mesh Nodes, Point Nodes, Volume Nodes, Simulation Zone, Material Nodes, Texture Nodes, Utilities Nodes, Group Nodes e Hair Nodes. Poi ognuna ha le sue sottocategorie, utili trovare il nodo di interesse e per capire le sue funzionalità ed i suoi obiettivi.

7.5.1 Tipo Attribute Nodes:

Questi nodi sono utilizzati prevalentemente per gestire attributi custom, i quali possono comunicare in diverse zone dell'albero dei nodi di un geometry nodes o con altre proprietà dell'oggetto come per esempio lo shading. Un loro utilizzo, per esempio, avviene quando si deve gestire un geometry nodes di simulazione.

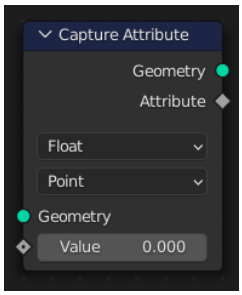


Figura 7.4

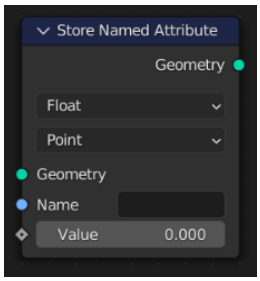


Figura 7.5

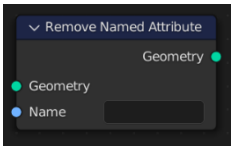


Figura 7.6

7.5.2 Tipo Output Nodes:

Il nodo *Viewer* consente di visualizzare graficamente nella Viewport 3D e matematicamente nello Spreadsheet Editor una qualsiasi geometria.

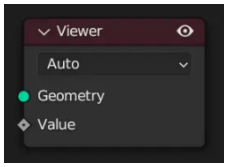


Figura 7.10

7.5.3 Tipo Input Nodes:

I nodi di input si dividono in due sottocategorie: *Constant* e *Scene.* I primi non hanno socket di input e forniscono in output il valore del nodo. Mentre i nodi *Scene* sono quei nodi che trattano informazioni legate alla scena od a oggetti all'interno di essa.

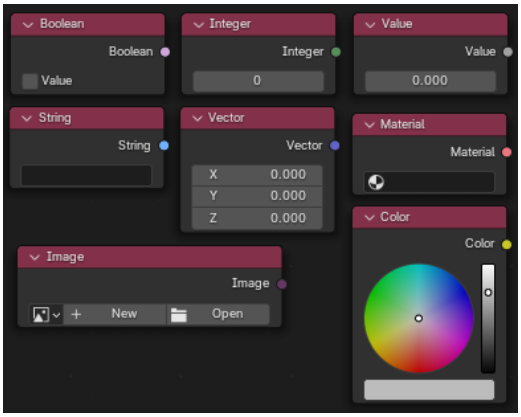


Figura 7.7

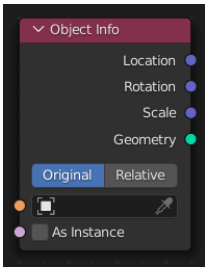


Figura 7.8

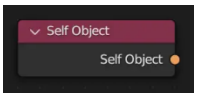


Figura 7.9

7.5.4 Tipo Geometry Nodes:

Questa categoria comprende quattro grandi sottocategorie: *Read, Sample, Write e Operations*, in più ci sono due nodi che non hanno una sottocategoria in quanto hanno caratteristiche differenti.

7.5.4.1 Sottotipo Read:

Sono nodi di tipo input, ovvero restituiscono il dato valore, ma non hanno socket di input.

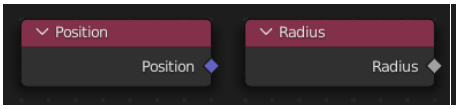


Figura 7.11

Figura 7.12

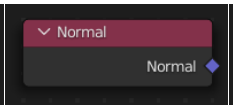


Figura 7.13

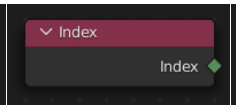


Figura 7.14

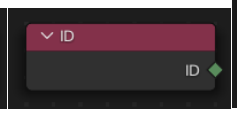


Figura 7.15

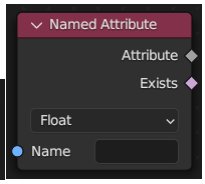


Figura 7.16

7.5.4.2 Sottotipo Sample:

Il nodo *Sample Index* recupera i valori da una geometria di origine in corrispondenza di un indice specifico.

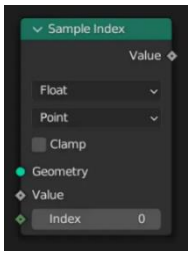


Figura 7.17

7.5.4.3 Sottotipo Write:

Per quanto riguarda invece il sottoinsieme dei Write sono stati utilizzati due nodi: Set ID e Set Position. Il primo inserisce l'attributo id sulla geometria di input. Invece il secondo controlla la posizione di ogni punto, allo stesso modo in cui controlla l'attributo posizione.

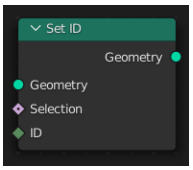


Figura 7.18

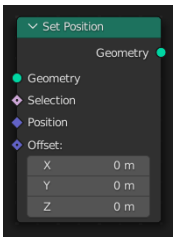


Figura 7.19

7.5.4.4 Sottotipo Operations:

La sottocategoria di Operations contiene alcuni dei nodi più utilizzati e sono quelli che chiedono in input una geometria e restituiscono in output la geometria modificata.

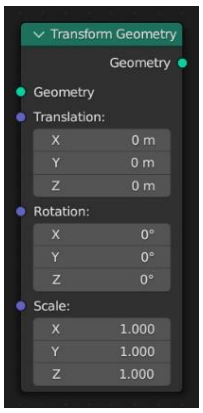


Figura 7.20

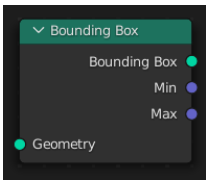


Figura 7.21

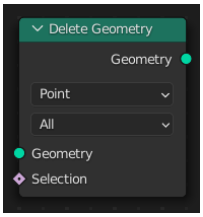


Figura 7.22

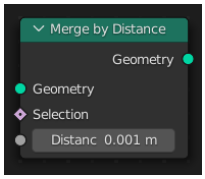


Figura 7.23

7.5.4.5 Join Geometry:

Il nodo di *Join Geometry* semplicemente unisce geometrie create separatamente, mantenendo tutte le proprietà e caratteristiche delle geometrie di input. Per gli attributi e i domini, invece vengono presi come output quelli più complessi.

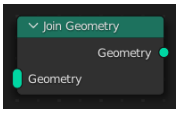


Figura 7.24

7.5.4.6 Geometry to Instance Node:

Il nodo *Geometry to Instance* trasforma ogni geometria di input connessa in un'istanza.

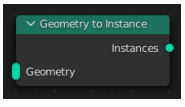


Figura 7.25

7.5.5 Tipo Curve Nodes:

Questa categoria, invece comprende le stesse quattro sottocategorie principali del tipo Geometry: *Read, Sample, Write e Operations*; in aggiunta ha anche le sottocategorie *Primitives* e *Topology*.

7.5.5.1 Sottotipo Read:

Di questo sottotipo si è utilizzato il nodo *selection endpoint,* che consente di selezionare un numero arbitrario di punti finali in ciascuna spline di una curva.

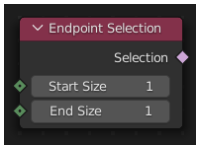


Figura 7.26

7.5.5.2 Sottotipo Write:

Questa sottocategoria permette di controllare alcuni valori della curva come il raggio e la ciclicità della curva stessa.

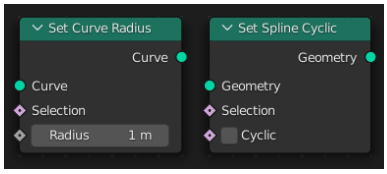


Figura 7.27 **Figura 7.28**

7.5.5.3 Sottotipo Operations:

Questa sottocategoria permette di modificare e trasformare le curve, come trasformare una curva in una mesh o in punti, aggiungere altri punti di controllo o ricalcolarli, oppure rimuovendoli. Altri nodi permettono di arrotondare gli angoli in modo simile all'effetto del modificatore *Bevel* su una mesh 2D, con la differenza che le parti arrotondate sono sempre porzioni di un cerchio, o di generare una mesh utilizzando l'algoritmo di triangolazione di Delaunay.

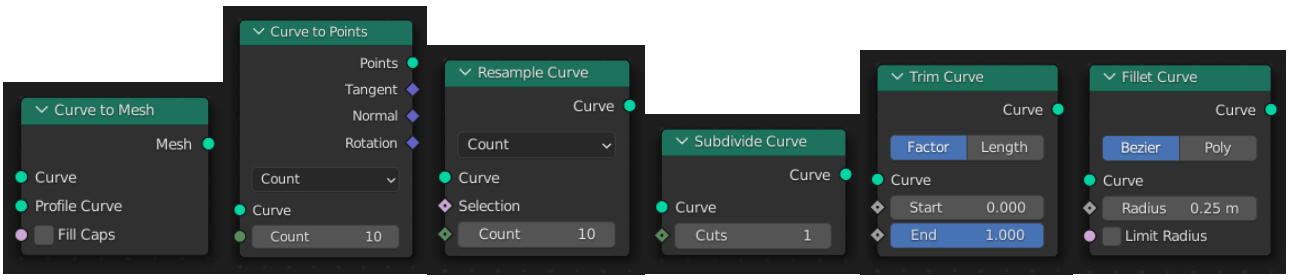


Figura 7.29 **Figura 7.30** **Figura 7.31** **Figura 7.32** **Figura 7.33** **Figura 7.34**

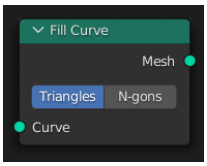


Figura 7.35

L'algoritmo di triangolazione vincolato di Delaunay[2] afferma che la triangolazione per un gruppo di punti P su un piano è una triangolazione DT(P) tale che nessun punto appartenente a P sia all'interno del circumcerchio di ogni triangolo in DT(P). A destra l'immagine che visualizza tale algoritmo.

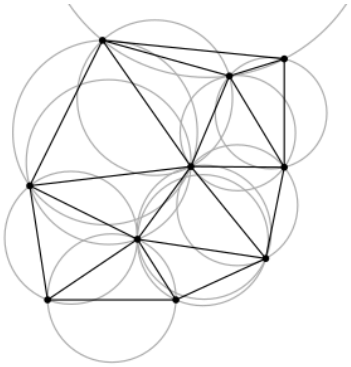


Figura 7.36

Si evitano triangoli stretti e gruppi di punti su una stessa linea in quanto non esiste triangolazione. Invece per gruppi di quattro o più punti su una stessa circonferenza, come i vertici di un rettangolo, la triangolazione non è unica: ognuno dei due possibili triangoli in cui si può dividere il quadrilatero, infatti, soddisfa i requisiti di Delaunay, ovvero che le circonferenze circoscritte ai triangoli non contengano altri punti.

7.5.5.4 Sottotipo Primitives:

Questo sottotipo presenta una serie di nodi che rappresentano i punti di partenza delle curve ovvero sono le curve basi da cui si creano gli asset.

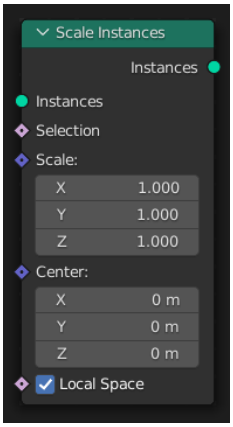
Figura 7.37 **Figura 7.38** **Figura 7.39** **Figura 7.40** **Figura 7.41** **Figura 7.42** **Figura 7.43** **Figura 7.44**

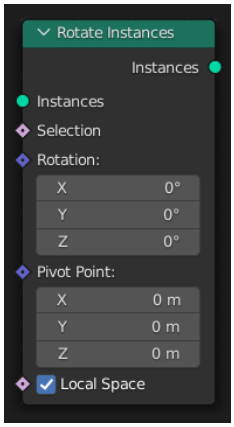
[2] Boris Nikolaevič Delone, noto anche come Delaunay (San Pietroburgo, 15 marzo 1890 – Mosca, 17 luglio 1980), è stato un matematico sovietico, noto per i suoi contributi all'algebra moderna e alla geometria dei numeri.

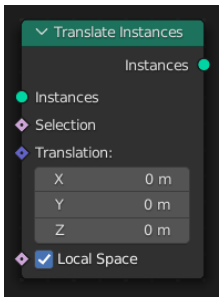
7.5.6 Tipo Instance Nodes:

Per quanto riguarda i nodi di instance, essi sono quelli che generano istanze o che operano e modificano istanze.

Figura 7.45 **Figura 7.46**







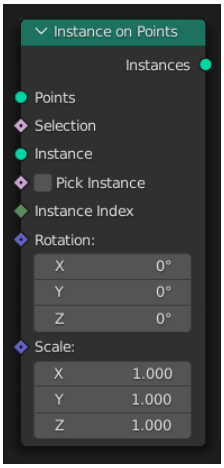


Figura 7.47 **Figura 7.48** **Figura 7.49** **Figura 7.50**

Il nodo *Realize Instance*s trasforma tutte le istanze in dati geometrici reali, in modo da lavorare su ogni singola istanza, perdendo però in termini di prestazioni quando si hanno molte istanze di geometria complessa.

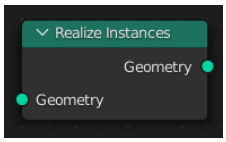


Figura 7.51

7.5.7 Tipo Point Nodes:

Questo tipo di nodi permette di manipolare i punti e nuvole di punti. Di questa famiglia di nodi si è utilizzato il nodo Set Point Radius.

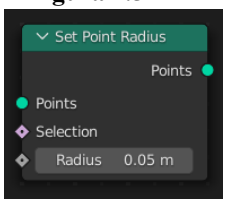


Figura 7.52

7.5.8 Tipo Texture Nodes:

Il nodo *Image Texture* a differenza degli altri nodi di questa tipologia, funziona in modo diverso rispetto al nodo equivalente dello shader, ovvero aggiunge un file di immagine come texture e i dati dell'immagine vengono campionati con il vettore di input.

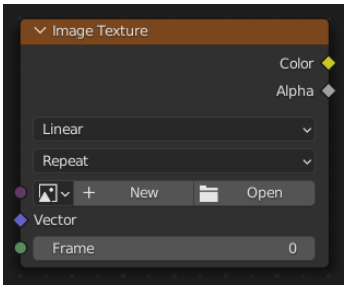


Figura 7.53

7.5.9 Tipo Mesh Nodes:

Il tipo Mesh node, anch'esso comprende le stesse sei sottocategorie principali del tipo Curve: *Read, Sample, Write e Operations*, *Primitives* e *topology,* e una nuova categoria, *UV.*

7.5.9.1 Sottotipo Read:

Di questa tipologia si è utilizzato il nodo *Edge Neighbors* che restituisce informazioni sulla topologia relative a ciascun bordo di una mesh.

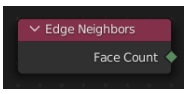


Figura 7.54

7.5.9.2 Sottotipo Write:

Il nodo *Set Shade Smooth* controlla se le facce della mesh appaiono uniformi nella viewport e nel rendering.

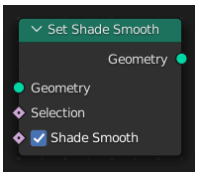


Figura 7.55

7.5.9.3 Sottotipo Operations:

Questi sono nodi che permettono di manipolare, modificare e gestire le mesh. Permettono anche di trasformare una mesh in punti o curve.

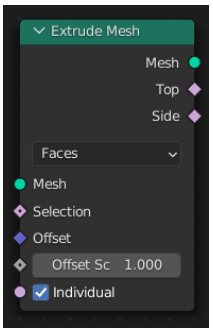


Figura 7.56

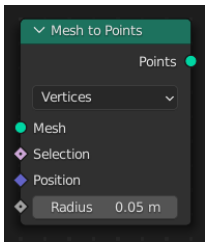


Figura 7.57

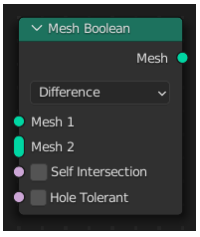


Figura 7.58

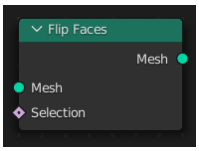


Figura 7.59

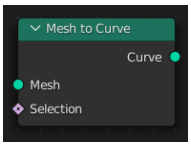


Figura 7.60

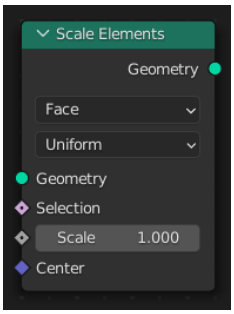


Figura 7.61

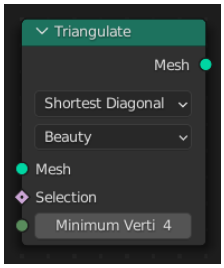


Figura 7.62

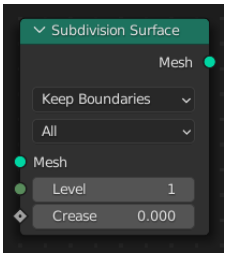


Figura 7.63

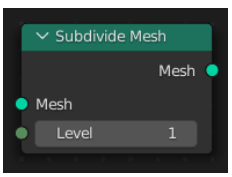


Figura 7.64

7.5.9.4 Sottotipo Primitives:

In questo sottotipo sono presenti tutte quelle forme geometriche primitive da cui si possono costruire oggetti.

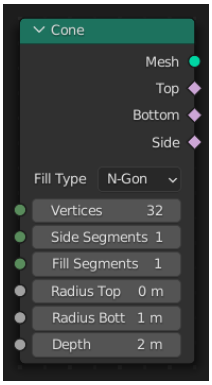


Figura 7.65

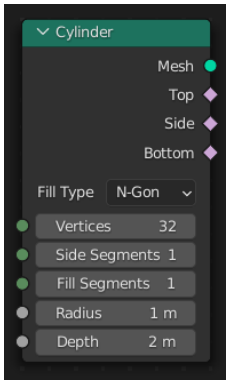


Figura 7.66

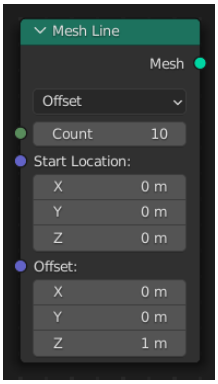


Figura 7.67

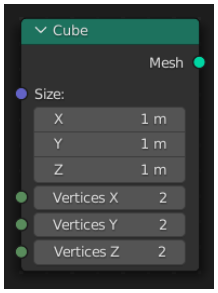


Figura 7.68

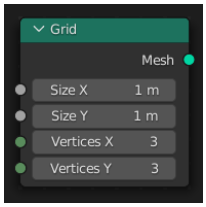


Figura 7.69

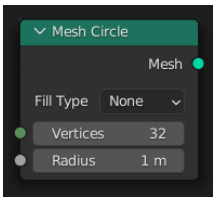


Figura 7.70

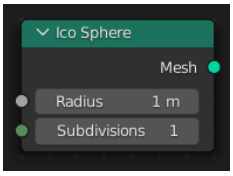


Figura 7.71

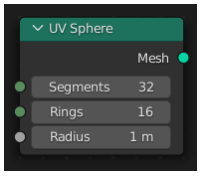


Figura 7.72

7.5.9.5 Sottotipo UV:

Il nodo *UV Unwrap* genera una *UV map* in modo simile al operazione nell'editor UV, ma con la differenza che il nodo non esegue la correzione delle proporzioni, perché è banale da implementare con un nodo matematico vettoriale.

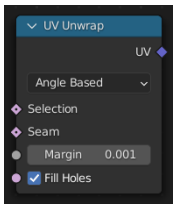


Figura 7.73

7.5.10 Tipo Material Nodes:

I Material nodes sono quei nodi che permettono di comunicare direttamente con la scheda di shading, infatti nei geometry nodes lo shading non viene gestito normalmente ma si devono usare questi nodi in tutte le parti dell'albero dei nodi in cui si vuole un determinato materiale o una determinata texture.

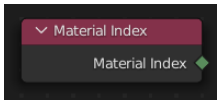


Figura 7.74

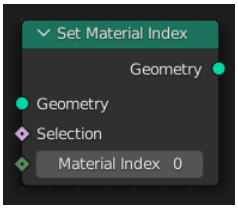


Figura 7.75

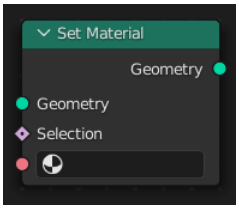


Figura 7.76

7.5.11 Tipo Utilities Nodes:

Il gruppo utilities comprende diversi sottogruppi, tra i quali ci sono i gruppi Color, Text, Vector, Field, Math, Rotation e Random Value, Repeat Zone e Switch.

7.5.11.1 Sottotipo Color:

Di questa categoria si è visto il nodo *Color Ramp* che è lo stesso utilizzato per lo shading e in questo caso viene utilizzato per mappare i valori ad un vettore, Color, con l'uso di una sfumatura.

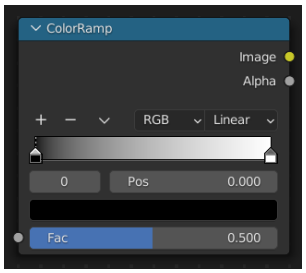


Figura 7.77

7.5.11.2 Sottotipo String:

Questo sottotipo è una categoria che si è studiata come si vedrà in seguito per la creazione di insegne luminose ed esse permettono facilmente di operare con caratteri e stringhe come se fossero delle mesh.

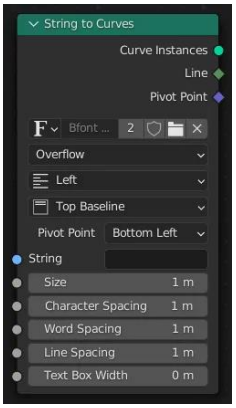


Figura 7.78

7.5.11.3 Sottotipo Vector:

Questi sono nodi che permettono di manipolare vettori o singole coordinate.

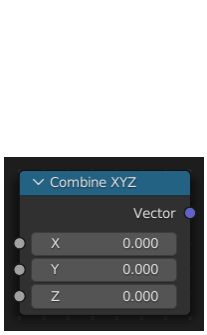


Figura 7.79

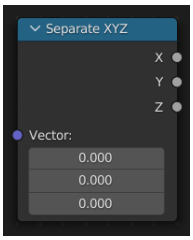


Figura 7.80

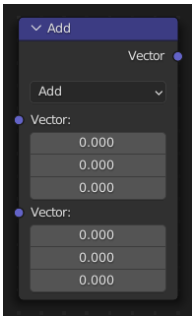


Figura 7.81

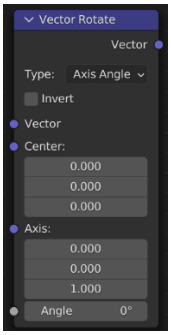


Figura 7.82

7.5.11.4 Sottotipo Math:

Essi sono nodi che permettono di eseguire operazioni matematiche, logiche, comparazioni, conversioni, rimappature, arrotondamenti e missaggio tra valori, colori e vettori.

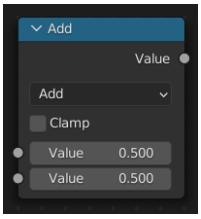


Figura 7.83

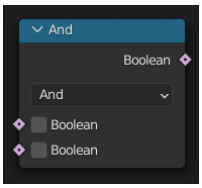


Figura 7.84

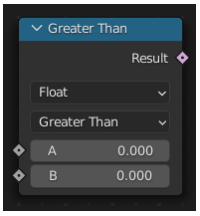


Figura 7.85

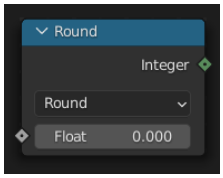


Figura 7.86

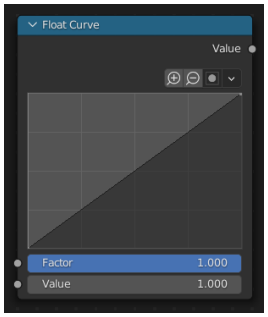


Figura 7.87

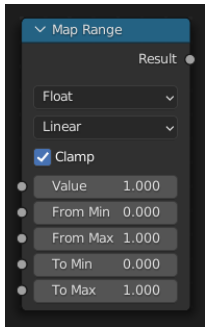


Figura 7.88

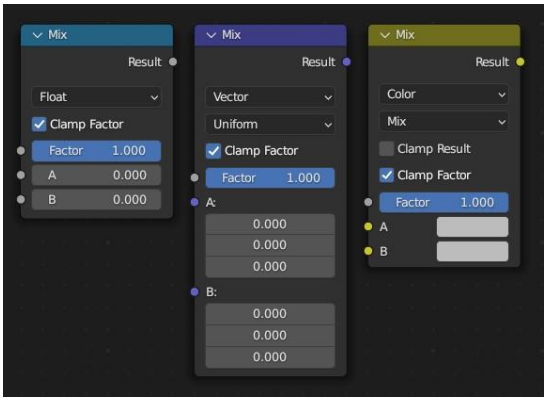


Figura 7.89

7.5.11.5 Sottotipo Rotation:

Questi sono nodi che permettono di gestire le rotazioni di mesh, curve e istanze.

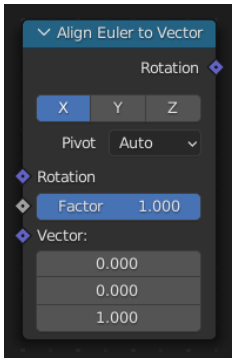


Figura 7.90

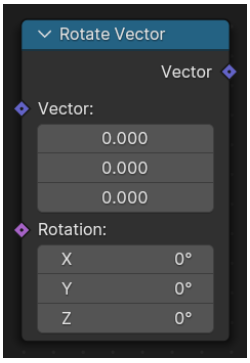


Figura 7.91

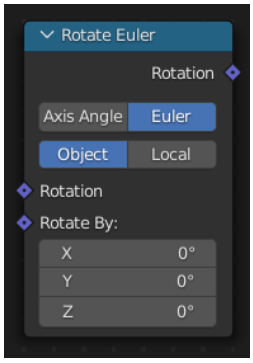


Figura 7.93

7.5.11.6 Altri Nodi:

Per quanto riguarda invece i nodi che non hanno una sottocategoria ci sono il nodo *Random value*, il nodo *Repeat*, che consente l'esecuzione di nodi più volte in un ciclo e infine il nodo *Switch*.

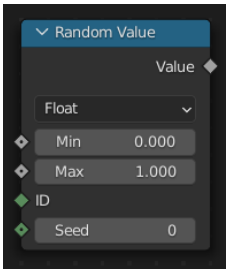


Figura 7.94

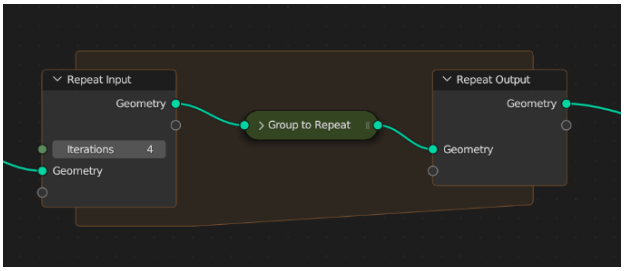


Figura 7.95

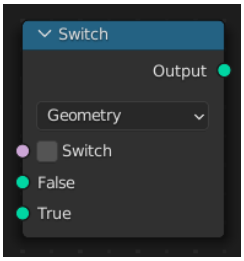


Figura 7.96

7.5.12 Tipo Group:

I nodi *Group* vengono usati per semplificare una struttura ad albero di nodi. I due nodi necessari per questo tipo di operazione sono il Group Input e Group Output. Il primo serve per poter gestire i dati al livello superiore dell'albero dei nodi oppure per poterli gestire nella finestra dei modificatori nel riquadro riguardante il geometry node stesso. Il secondo invece viene usato per poter visualizzare il risultato al livello superiore dell'albero dei nodi oppure visualizzarlo nella Viewport.

7.5.13 Tipo Simulation zone:

Il gruppo di nodi Simulation zone consente di creare simulazioni fisiche, come nel metodo tradizionale, e trattarle come se fossero mesh normali. Questo permette una maggiore possibilità di manipolazione della simulazione stessa, velocizzando il lavoro e anche velocizzando eventuali modifiche da apportare.

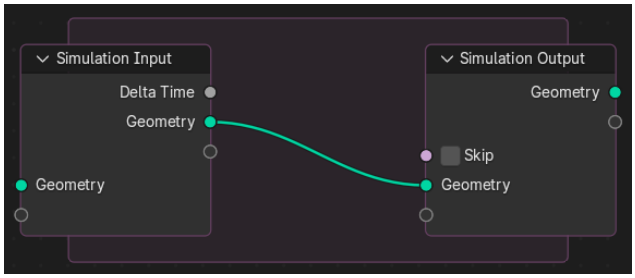


Figura 7.97

8. Reverie Dawnfall

Reverie Dawnfall ovvero “Il calare dell’alba su un sogno ad occhi aperti” è una serie d’animazione 3D cyberpunk nata nel 2017 dalle menti di Mark Gore e Riccardo Antonino, docente del corso Visual Effects al Politecnico di Torino, nonché uno dei fondatori di Robin Studio [11], studio creativo di Torino, presso il quale si svolge la produzione e dove svolgo la tesi.

Figura 8.1: Reverie Dawnfall cover

La serie pone l’attenzione su diverse tematiche di carattere sociale, climatico e scientifico. Più nello specifico tratta i disturbi neuro-psichiatrici, l’esplorazione spaziale, il desiderio di riscoprire il passato ed il collasso degli ecosistemi, a causa della super industrializzazione e dell'eccessivo consumismo, e le terribili conseguenze dell’ignorare lo sviluppo sostenibile.

Reverie Dawnfall è ambientata in un futuro distopico, su un esopianeta simile alla Terra per morfologia e caratteristiche, ma è in rotazione sincrona rispetto alla propria stella ovvero una metà del pianeta è perennemente buia, mentre l’altra metà è perennemente illuminata.
La linea di confine tra le due zone, vive nel crepuscolo, ed è qui che sorge Dome City, la città protagonista della serie. Questa particolare posizione dà alla serie elementi caratterizzanti sul look visivo come colori, contrasti e stile di illuminazione globale, che si distacca sia dal cupo notturno dei cyberpunk stile “Blade Runner”, sia dai saturi toni caldi dei post-apocalittici alla “Mad Max”.

Dome City è una metropoli di media grandezza, coperta da una cupola trasparente che serve a proteggere la città dal contesto altamente inquinato e tossico in cui si trova il modo esterno.
La conformazione della città è quindi radiale che si erge assecondando la forma della cupola, cioè gli edifici più alti sono verso il centro e via via sono sempre più bassi verso i confini della cupola, e si dispongono a settori tutti di circa la stessa superficie, su due fasce, come in un bersaglio per il gioco delle freccette.
La città è situata in una posizione strategica al bordo della notte, che si trova ad est rispetto alla stessa, a ovest vi è la zona giorno, mentre a nord si trova la baia, Poisoned Seas, abitata da enormi meduse dal veleno letale.

Attualmente l'arco narrativo è composto da una stagione di una decina di episodi e vedono come protagonista una giovane e brillante studentessa di entomologia di nome Nadya Sinkamen, e tre suoi amici: Jameela Rani, Gavril Gajdos e Breather.
Gli abitanti, tutti nati con disabilità fisiche o disturbi neuropsichiatrici a causa del gravissimo inquinamento, sono in rivolta sotto l'influenza di cinque grandi superpotenze che coincidono in multinazionali dal controllo monopolistico, ma la polizia è sempre più inefficace nel mantenere l'ordine.
In una società in cui ogni persona è in qualche modo malata, imperfetta o disabile, l'enigmatico Peregrine, antagonista della serie, inspiegabilmente nato sano e geneticamente puro, manovra i destini dalle ombre di un mondo al tramonto.
Questo in contrapposizione con la protagonista che soffre di sinestesia e iperestesia, ovvero una condizione che la porta ad avere visioni di un mondo simile ma più vivo e luminoso.
Questa situazione porta Nadya a dissociarsi completamente dalla realtà nel momento in cui queste visioni si fanno più nitide e insistenti, e incomincerà a dubitare del proprio universo.

8.1 Pre-Produzione di Reverie Dawnfall

Reverie Dawnfall è un progetto la cui idea nasce nel 2017, mentre nel 2019 è stato realizzato un trailer da dei miei colleghi, ex tesisti; dunque, vi era già un background di materiale di partenza su cui poter lavorare per raggiungere gli obbiettivi richiesti.
In particolare, i concepts dei personaggi protagonisti, creati dal grafico Edoardo Audino non sono stati più modificati in quanto ritenuti soddisfacenti dalla produzione. Per quanto riguarda invece gli environments, lo stile e lo shading invece, si è ripreso il lavoro precedente e si è cercato di migliorarli e renderli più efficienti, facendo comunque un lavoro di ricerca e di sviluppo, caratterizzante di solito la fase di pre-produzione.
Di seguito viene mostrato uno schema generico dei diversi steps che compongono la production plan per la realizzazione di un film di animazione 3D:

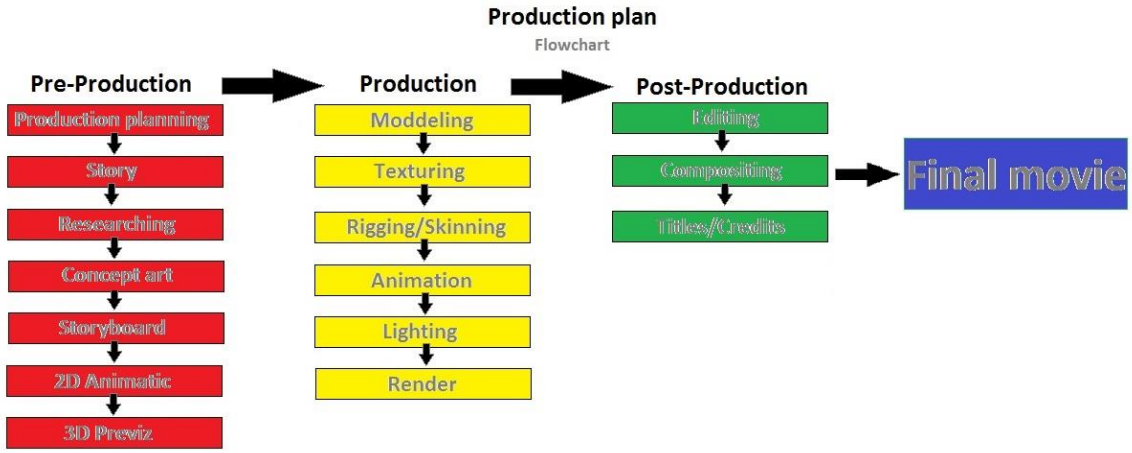


Figura 8.2: Flowchart di un generico Production plan di un film di animazione.

Questo schema permette di comprendere che all'interno di ognuna delle tre fasi produttive la pipeline di lavoro è composta di varie divisioni che normalmente equivalgono ad altrettante figure lavorative, inoltre dato questo fatto essa può essere organizzata in maniera non lineare.
Dall'altra parte però pre-produzione, produzione e post-produzione rappresentano tre fasi distinte che raramente possono coesistere in contemporanea: alla prima segue la seconda, e poi si passa alla terza.

Analizzando invece la pipeline di lavoro del nostro progetto, dato il carattere sperimentale che ne fa da padrone, non è esistita una vera e propria separazione tra le tre fasi principali.
Proprio per questo motivo, come si mostrerà in seguito è stato impossibile decidere con precisione e accuratezza lo stile desiderato del prodotto finale durante la fase di pre-produzione, ma si è proseguiti a tentativi di gusto grafico, specialmente per quanto riguarda alcuni elementi della modellazione degli environments.

Si può dire quindi che nel caso di questa tesi, pre-produzione e produzione fanno parte di un unico processo, in quanto le due fasi si intrecciano tra di loro.
Ad esempio, in corso d'opera della modellazione del night club ci si è accorti che i modelli fatti erano troppo "anonimi", ovvero rispecchiavano troppo modelli già esistenti di elementi cyberpunk; quindi, si è pensato di caratterizzarli in base al contesto della serie: un mondo dominato dalla presenza degli insetti, ciò permette di rendere la serie più originale e unica nel suo genere.

Il diagramma di seguito descrive in maniera generica questo concetto di workflow utilizzato per la realizzazione dell'ambiente 3D.

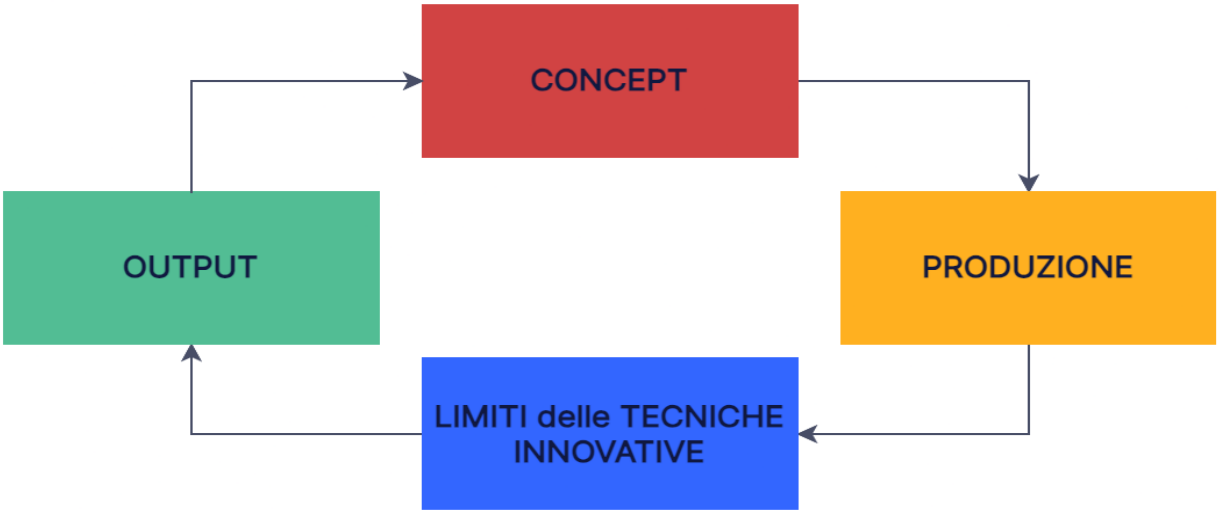


Figura 8.3: Flowchart di produzione seguito durante la tesi

Questo continuo iterare fino ad ottenere il risultato voluto, è soprattutto dovuto alla sperimentazione di tecniche che, in alcuni casi, sono ancora in fase di sviluppo e quindi non sempre è possibile creare ciò che si ha in mente.

Oltre al fattore software, è importante il fattore tempo, quindi l'idea era quella di avere più output diversi per capire quale fosse il risultato più gradevole e coerente con il resto dell'ambiente.

8.2 Risultato estetico

Lo stile della modellazione che si è cercato di realizzare ed ottenere è una fusione tra diversi stili, prendendo come riferimento sia dei prodotti già esistenti sul mercato che di invenzione.

Nel primo caso si hanno prodotti con ambienti cyberpunk, come Blade Runner 2049 o Cyberpunk 2077 di cui si sono cercate alcune forme squadrate e l'utilizzo di led anche in oggetti come le sedie e le poltrone. Come si vede in queste immagini.

Figura 8.4: Reference Cyberpunk 2077 [12]

Nel secondo caso invece si è fatto uno studio del mondo degli insetti, visto che nel mondo di Reverie le uniche forme di vita nella città, Dome City, sono proprio insetti e piccoli roditori; dallo studio degli insetti si sono prese alcune caratteristiche come ali, antenne, zampe e forme geometriche che li ricordino, come nel caso delle immagini sottostanti.

Figura 8.5: Alcuni degli insetti presi come reference [13] [14] [15] [16]

Per aiutare l'immaginazione per la creazione degli asset si è fatto uso anche di strumenti di intelligenza artificiale, Copilot di Microsoft [17] nello specifico, che permettono di generare immagini da un prompt di testo. Di seguito sono mostrate un paio di immagini usate come reference generate dall'intelligenza artificiale.

Figure 8.6: Alcune delle immagini create con l'intelligenza artificiale usate come reference [17]

8.3 Analisi ambiente 3D

L'obiettivo principale del mio lavoro in questo progetto ha riguardato la creazione di un ambiente 3D per l'episodio pilota della serie, tale ambiente 3D è il nightclub della città.

Il lavoro previsto era quello di creare tutti gli asset, arredamento interno e oggettistica, di un locale notturno effettuato tramite il software Blender.

Per quanto riguarda la versione utilizzata di Blender, dapprima ho utilizzato la 3.6, ma visto che i Geometry Nodes sono in continua evoluzione, e ad ogni nuova versione vengono aggiunti nodi e fatte migliorie su quelli esistenti, ho poi deciso di proseguire il lavoro con la nuova versione 4.0.

Di seguito verrà fatto prima una presentazione dell'ambiente in modo generale, per poi passare a spiegare nel dettaglio la creazione di ogni mobilio e ogni oggetto della stanza, facendo rifermento anche al ciclo di produzione eseguito con spiegazione di eventuali problemi riscontrati e come sono stati affrontati.

8.4 Night Club

"Un'ampia stanza rassomigliante un night club, con cameriere in minuscoli, asimmetrici abiti succinti, alcune ballerine di lapdance. Un ballatoio al piano superiore consente l'accesso ad alcune stanze chiuse; una ragazza trascina per mano un uomo, ridendo entra in una delle porte. Sul parapetto una prostituta appoggiata guarda annoiata la sala sottostante. Ai tavolini alcuni clienti bevono, altri hanno ragazze in braccio, in primo piano vediamo un tavolo a forma di piccola arena, con dentro due insetti che lottano e tre persone che li osservano incitando concitati, tenendo in mano delle carte-chip." Mark Gore

La prima fase del lavoro è stata quella di analizzare le varie parti dello script dell'episodio pilota in cui era presente l'ambientazione Night Club, questo per estrapolare più dati e dettagli possibili sulla composizione del locale.

Più nello specifico si è fatto uno spoglio della sceneggiatura, dove si sono segnati i principali asset dell'ambientazione come si vede nella seguente immagine.

Aa Name	Tags	Status	Person	Modello	Rigging	Texture	#
Carte chip	Props	In progr...	Andrea Loro	☐	☐	☐	
Apparecchietto(vicino la porta)	Props	In progr...	Andrea Loro	☐	☐	☐	
piccolo congegno dall'aspetto raffazzonato	Props	In progr...	Andrea Loro	☐	☐	☐	
Tavolo piccolo a forma di arena	Enviroment	Done	Andrea Loro	☐	☐	☐	
bancone bar	Enviroment	Done	Andrea Loro	☐	☐	☐	
bottiglie bar	Enviroment	Done	Andrea Loro	☐	☐	☐	
serratura	Enviroment	Not start...	Andrea Loro	☐	☐	☐	
porta della gambling den	Enviroment	Done	Andrea Loro	☐	☐	☐	
bicchieri	Enviroment	Done	Andrea Loro	☐	☐	☐	
Tavoli	Enviroment	Done	Andrea Loro	☐	☐	☐	
postazioni lap dance	Enviroment	Done	Andrea Loro	☐	☐	☐	
sedie	Enviroment	Done	Andrea Loro	☐	☐	☐	
divanetti		Done	Andrea Loro	☐	☐	☐	
porte stanze private	Enviroment	Done	Andrea Loro	☐	☐	☐	
impianti luci	Enviroment	Done	Andrea Loro	☐	☐	☐	
Prostituta 1	character	In progr...	Chiara Petrol	☑	☑	☐	
Prostituta 2	character	In progr...	Chiara Petrol	☑	☑	☐	
Prostituta 3	character	In progr...	Chiara Petrol	☑	☑	☐	

Figura 8.7: Spoglio della sceneggiatura ed elenco degli asset

Dopodiché si sono analizzate alcune reference che erano già presenti, dal lavoro fatto negli anni precedenti dai miei colleghi.

La prima immagine mostrata di seguito vede una piantina della stanza principale composta da due piani: il primo piano accessibile da delle scale a chiocciola, dove sono presenti stanze per spettacoli privati e

sotto il piano terra che comprende la zona bar, tavolini, poltrone, divanetti, un tavolo arena e diverse zone per spettacoli lap dance; inoltre è presente una grande struttura su uno dei lati corti.

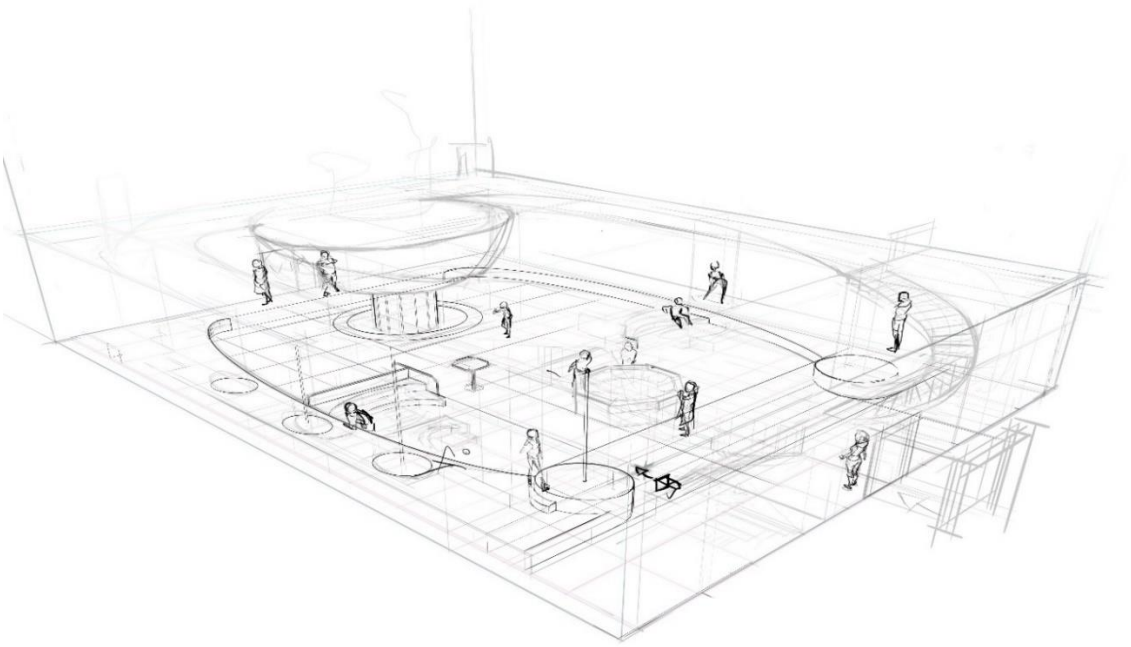


Figura 8.8: Reference Nightclub

Tale struttura viene inquadrata meglio nella seconda immagine, nella quale si può notare alla sommità di essa un ologramma di una persona e la dimensione della struttura in proporzione alla grandezza dei personaggi.

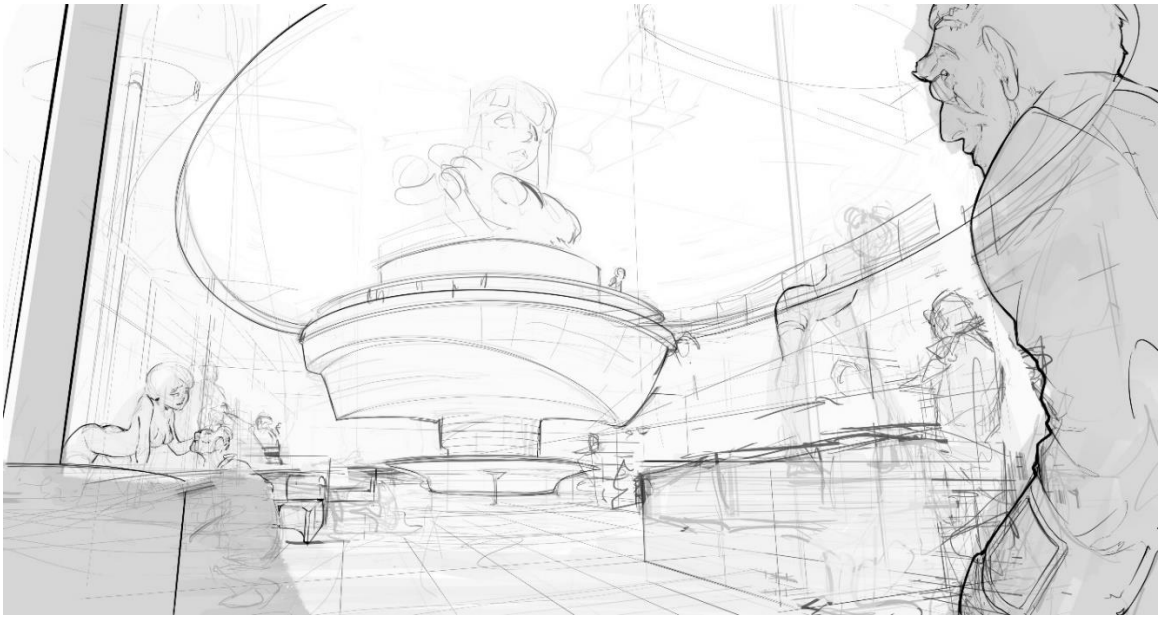


Figura 8.9: Particolare dell'ologramma e della sua struttura

Nel pensare alla struttura degli asset, ci si è trovati subito una sfida importante da tenere in considerazione: creare una scena complessa, ma che al tempo stesso abbia il minor numero possibile di vertici all'interno. Questo perché i tempi di render, se la scena è eccessivamente ricca di vertici, risultano essere molto lunghi, andando a compromettere i ritmi di produzione, soprattutto considerando che è un prodotto animato pieno di personaggi che, quindi aumentano notevolmente il costo computazionale.

Ora si passerà a trattare nello specifico ogni asset creato spiegandone nei dettagli il workflow, facendo riferimento a reference, problematiche riscontrate e soluzioni annesse.

8.4.1 Arena:

Il primo asset presentato è il tavolo-arena. Si è partiti facendo delle reference, figura a lato, con l'uso di software AI online per la creazione d'immagine tramite prompt testuale [18], partendo dall'idea di avere un arena con una caratterizzazione geometrica molto forte. Cosa anche richiesta nella sceneggiatura, infatti in essa era scritto "tavolo ottagonale".

Figura 8.10: Reference tavolo-arena [18]

Quindi come prima idea si è costruito il tavolo che può essere diviso in 4 settori; il primo è il top, ovvero la zona di gioco vera e propria, dove essa è formata da un bordo laterale abbastanza spesso dove poter appoggiare i dodecaedri, i porta insetti, ma anche drink e dove potersi appoggiare, mentre la zona di combattimento e interrata in modo da non fa cadere gli insetti durante il combattimento.
Gli altri settori fanno tutti parte del sostegno del tavolo; infatti ci sono elementi obbliqui che hanno l'obiettivo di snellire la struttura che sono alternati ad elementi luminosi tipico dello stile cyberpunk.

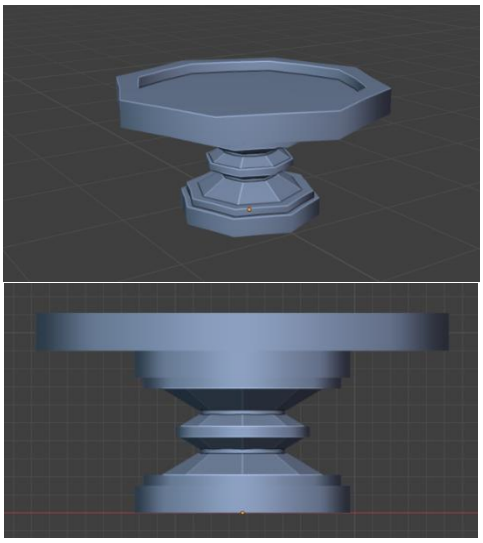


Figura 8.11

Infine l'ultimo elemento sono le due parti della gamba che sostengono le parti appena citate, ovvero i grossi sostegni appena sotto il top del tavolo e quello che tocca il pavimento, questi volevano dare l'idea di pesantezza e robustezza in contrasto con gli elementi obbliqui. Le immagini qui affianco e le due sotto

rappresentano questo step di lavorazione.

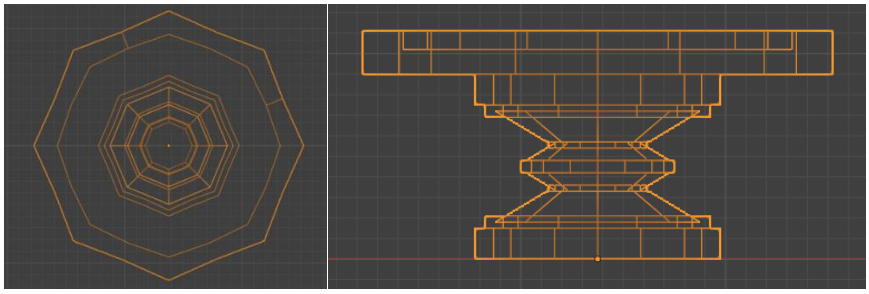


Figura 8.12

Dopo però una revisione si è notato, come anche per molti degli asset che vedremo di seguito, che questa arena risultasse anonima e non caratterizzasse abbastanza il mondo di Reverie.

Quindi dopo una ricerca si è cercato qualcosa di diverso. Questo lo si è fatto andando a sostituire gli elementi obbliqui con degli steli metallici che volevano riprendere la forma delle zampe di insetto ma che desse anche più slancio all'oggetto e rendesse ancora più snella l'arena, cosa richiesta anche sceneggiatura, in quanto in altre parti di essa l'arena veniva descritta come "strutture ottagonali fluttuanti".

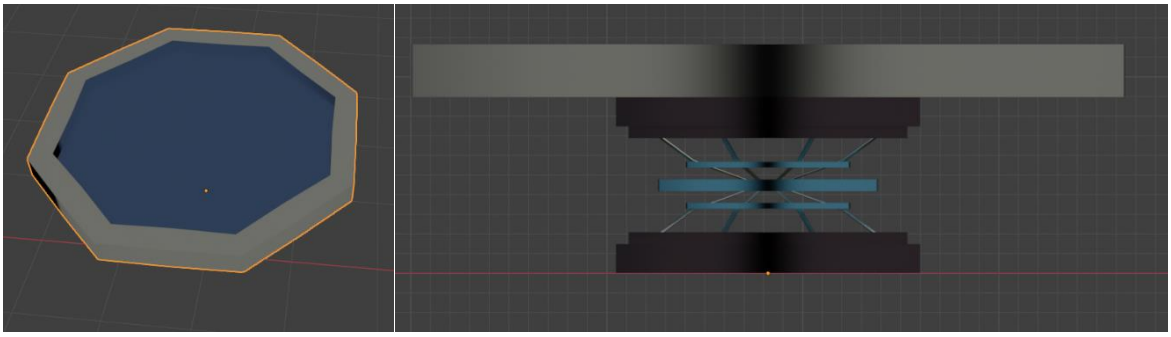


Figura 8.13: Risultato finale

Da notare anche l'immagine in Wireframe che permette di vedere come è costituita la mesh e la complessità a livello di numero di vertici. L'uso di forme geometriche semplici riduce drasticamente il numero di vertici.

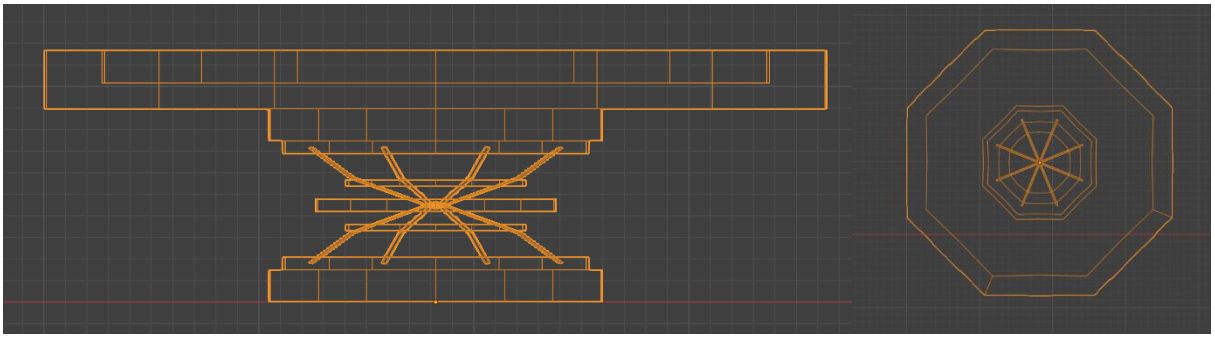


Figura 8.14

8.4.2 Bancone Bar:

Per quanto riguarda il bancone del bar, è stato uno degli asset più difficile da concepire, perché è uno di quegli arredi che all'interno di un locale notturno ne fa da padrone e rimane uno dei fulcri del locale.

All'inizio si era pensato ad un bancone molto squadrato a forma di L, diviso in due grossi blocchi, la parte inferiore, ovvero il bancone effettivo ed il soffitto, sorretto da una colonna, che copre l'intero ingombro del bancone. Tale bancone è visibile dalle due immagini a destra.

Il bancone effettivo era poi diviso in top, zona centrale che sporgeva nel lato interno creando la zona di lavoro per i baristi e una zona d'appoggio al pavimento più stretta per dare slancio all'intera struttura. Inoltre nella parte centrale vi erano due strisce a led che correvano lungo tutta la lunghezza e un poggiapiedi metallico con forme triangolari e quadrate.

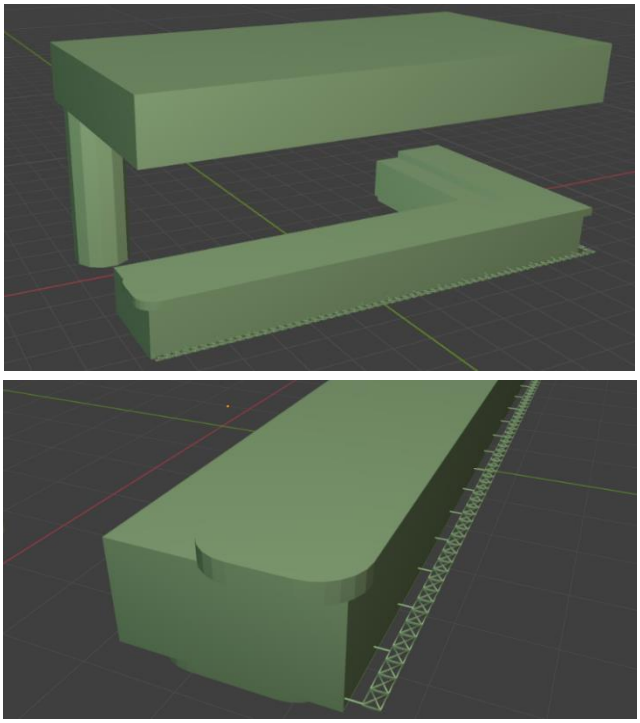


Figura 8.15

La zona del soffitto invece era un semplice rettangolo con una cavita interna in modo da alleggerire la struttura, la quale era sorretta da una colonna, nel lato corto aperto, di base dodecagonale.

Tutto questo era stato creato con l'intento di poter modificare lunghezza, larghezza, altezza e curvità del bordo del bancone nei suoi lati esterni.

Dopodiche ho provato a fare alcune modifiche, prendendo come reference l'immagine a destra, in quanto mi sembrava troppo semplice come struttura, creandone cosi una versione non più a L ma inclinando il lato corto e aggiungendo un pezzo al lato opposto, che prima era libero.

Figura 8.16: Reference Bancone bar Cyberpunk 2077 [12]

Tutto ciò è stato riportato anche al soffitto, creando così una struttura che ha come base di partenza un esagono non regolare.

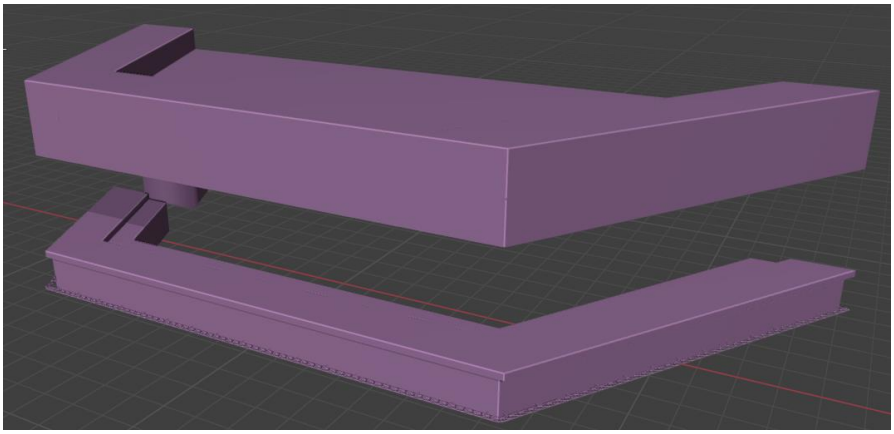


Figura 8.17

Dopo la revisione ho ragionato come poter caratterizzare ulteriormente la struttura, e ho pensato subito alla struttura dell'alveare che è ad esagoni, prendendo anche spunto da una immagine generata con la IA, mostrata qui a destra.

Quindi partendo da un esagono principale come base ho ricavato quattro lati dell'esagono, lasciando libero il lato contro il muro e quello per il passaggio.

Figura 8.18: Reference Bancone [17]

Il soffitto riprende la struttura esagonale sottostante, ma è completo con una cavita che permette di distinguere due zone, dove nella più esterna sono poi stati collocati dei faretti.

Come decorazione si è pensato di ricreare

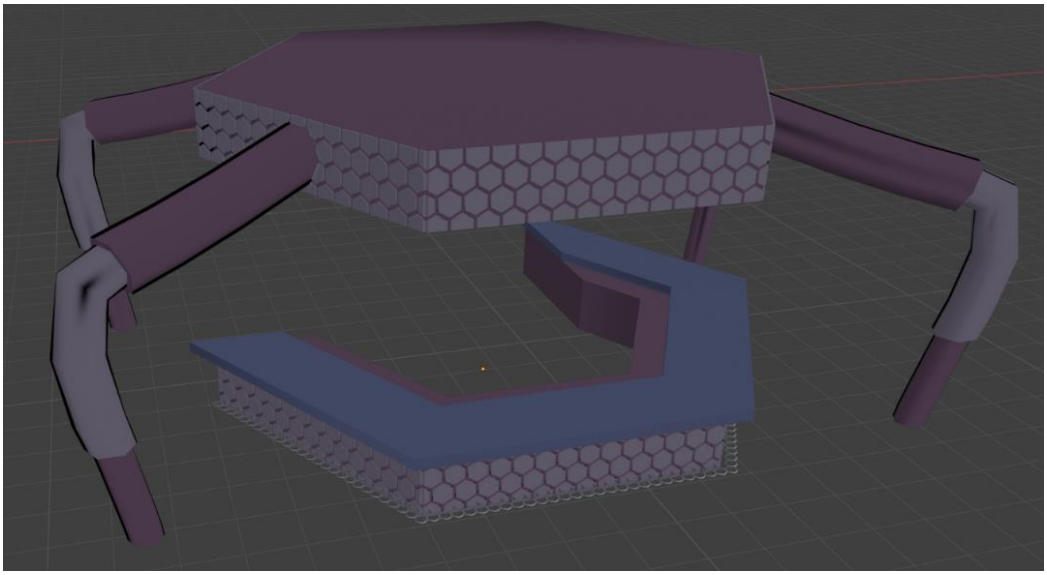


Figura 8.19: Risultato finale

altri esagoni sulle facce esterne del bancone, per caratterizzare ancora di più e dare il senso di cellette di alveare. Stessa forma esagonale anche per il poggiapiedi, in modo da renderlo coerente con tutta la struttura.

Infine il soffitto invece di essere sorretto da una colonna, è sorretto da quattro strutture a base esagonale che vogliono assomigliare a zampe di insetto, nello specifico si è preso spunto da ”Nagusta goedelii”, specie che vive nelle regioni calde, mostrata in figura a destra. Tali sostegni sono formati da tre parti che si vanno a rimpicciolire man mano che si scende verso la pavimentazione.

figura 8.20: Nagusta goedelii [19]

Si è scelto di creare solo quattro delle sei zampe in quanto sarebbero state di ingombro sia per l’uso della struttura che per l’ambientazione circostante.

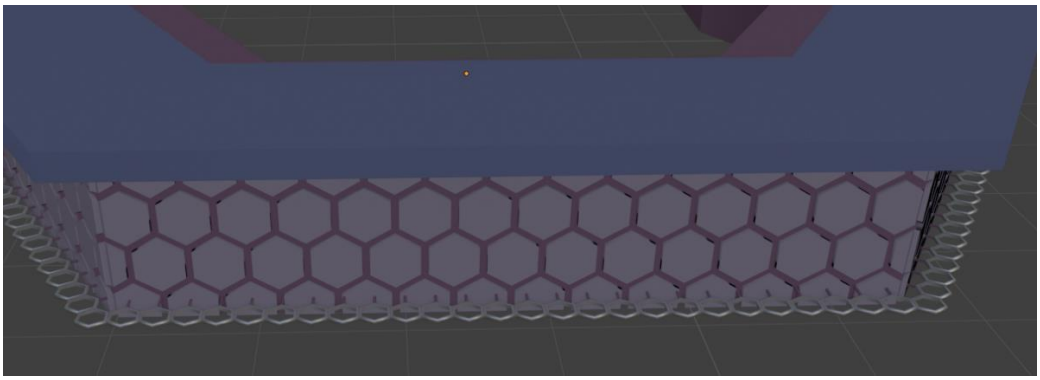


Figura 8.21: Dettaglio decorazioni e poggiapiedi

Da notare anche le immagini in Wireframe che permettono di vedere come è costituita la complessità di questa mesh. Questa uno degli asset più onerosi a livello computazionale in quanto i dettagli decorativi e la struttura ritagliata dall’esagono di partenza sono creati grazie al nodo Boolean Mesh: quindi con operazioni booleane di difference.

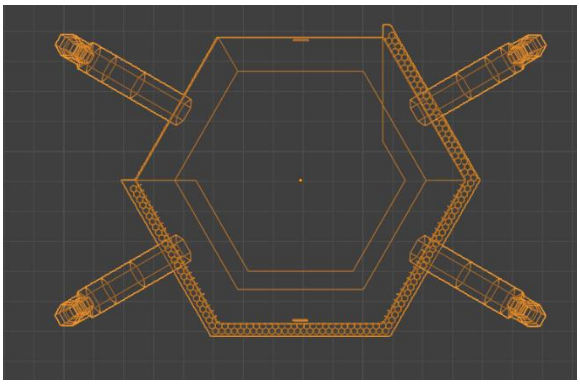


Figura 8.22

La nota positiva è che essendo la base un esagono e non una figura complessa con tanti vertici, si è limitato il costo computazionale.

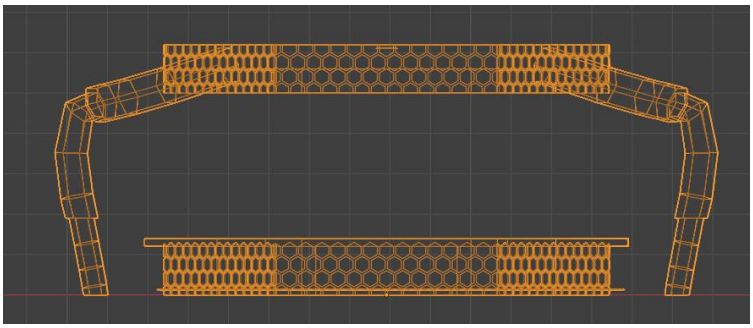


Figura 8.23

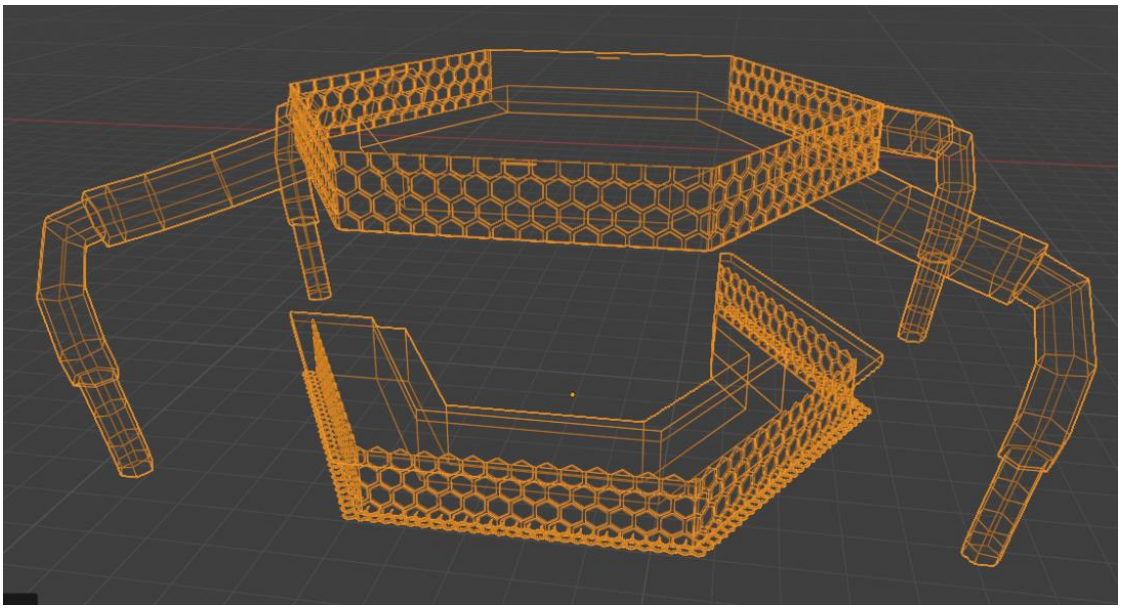


Figura 8.24

8.4.3 Bicchieri:

Per quanto riguarda i bicchieri, sono gli unici assets che non sono stati creati con i geometry nodes, ma si è voluto sperimentare con un nuovo sistema online, che permette partendo da una base di un qualsiasi oggetto, di modificarlo a piacimento: modifiche quali altezza, altezza dello stelo, larghezza e anche la forma più o meno spigolosa e tondeggiante.

Questo tool si chiama Sloyd [20], ed è un software in beta online che tramite l'uso dell'intelligenza artificiale permette di creare modelli 3D. L'obiettivo di questa web app è quella di aiutare durante la creazione di oggetti 3D in modo che diminuisca il tempo necessario alla creazione di oggetti, che magari sono di corredo all'ambiente, permettendo così di dedicarsi su elementi creativi più importanti e caratterizzanti dell'ambiente 3D. Tutto questo grazie alla generazione procedurale grazie ad algoritmi di apprendimento automatico.

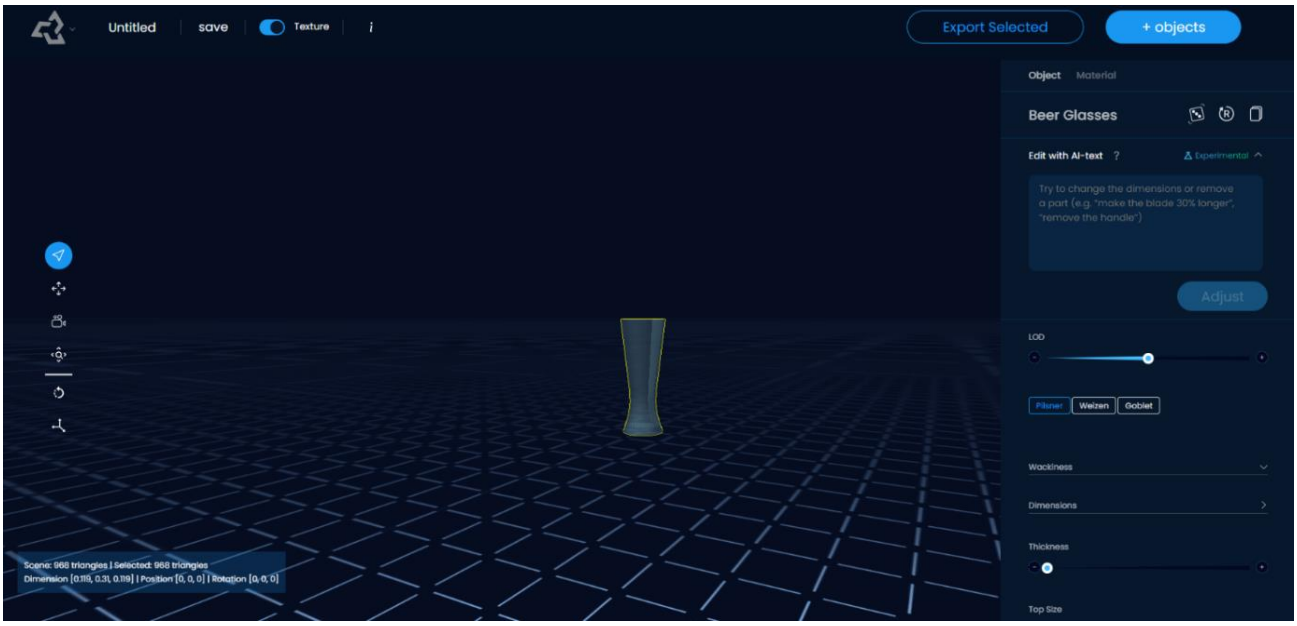


Figura 8.25: pagina workspace di Sloyd

I bicchieri creati con questo software sono presentati nelle due immagini seguenti, nelle quali si può notare che il software triangola i poligoni, ovvero che i bicchieri sono formati solo da poligoni con tre lati.

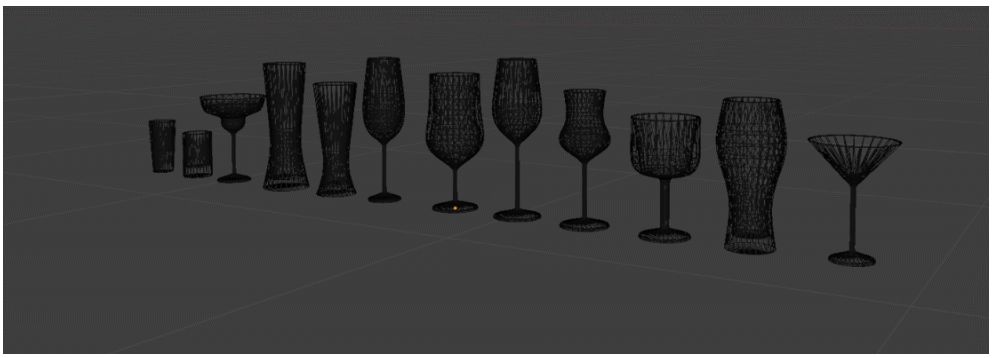


Figura 8.26

Mentre nella seconda immagine viene mostrato il render con i materiali finali.

Figura 8.27: Risultato finale

8.4.4 Bottiglie:

Le bottiglie sono state abbastanza complicate da creare in quanto sono si oggetti di piccola dimensione, ma l'idea era quella di creare un solo modello con una grande possibilità di variare i parametri per dare delle forme diverse. Quindi si è costruita come base di partenza una bottiglia cilindrica semplice, mostrata nelle immagini sotto, suddivisa in parte bassa, che è la zona che tocca con la superficie, parte centrale e parte alta, dove quest'ultima è la parte obliqua; infine il tappo, che ha le sue possibili di modifiche.

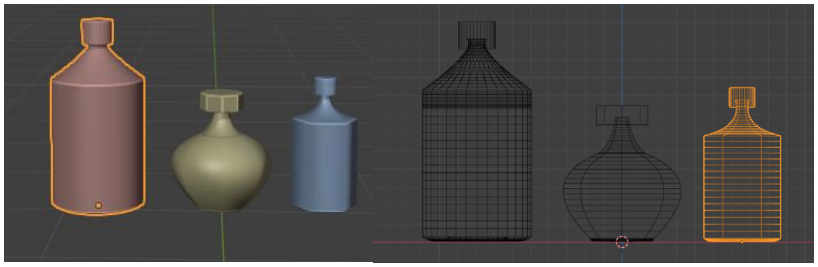


Figura 8.28

Da qui si è poi giocato con le variabili di input delle curve utilizzate per la creazione, per poter creare diverse forme, anche tra le più particolari; facendo però attenzione, grazie ai nodi *Math*, che alla modifica di un parametro tutta la bottiglia si muovesse in modo coerente per non avere errori come zone aperte o troppo spigolose. Un esempio molto semplice, modificando l'altezza della bottiglia, il tappo segue tale modifica e si troverà sempre nella posizione corretta.

Anche qua, dopo la revisione, ho voluto caratterizzare le bottiglie con elementi che riconducano al mondo degli insetti. Per questo motivo ho ripreso il concetto di zampe, in questo caso prendendo spunto da un ragno di nome Sphodros rufipes, mostrato in figura, e l'ho trasposto anche in questo asset nella parte bassa della bottiglia; questo con la specifica che la bottiglia non tocchi la superficie, in modo che la temperatura della bevanda al suo interno non venga alterata dalla quella della superficie in cui è posta. Anche il tappo è stato modificato, prendendo spunto le geometrie dei tavolini centrali che verranno mostrati in seguito. In questo caso l'idea era dare al tappo una forma particolare che però con le zampe potesse risultare migliore per l'apertura e chiusura della bottiglia.

Figura 8.29: Sphodros rufipe [16]

Figura 8.30: Risultato finale

Di seguito invece si mostrano la stessa bottiglia nell'immagine sopra, ma in versione Wireframe per vederne la complessità e il numero di vertici. Nella seconda immagine invece si può notare la grande varietà di tipologie che si possono produrre da un solo modello grazie alle possibilità date dai Geometry Nodes.

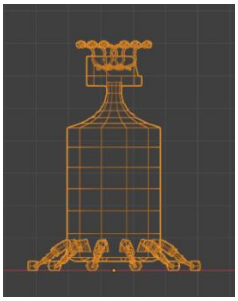


Figura 8.31

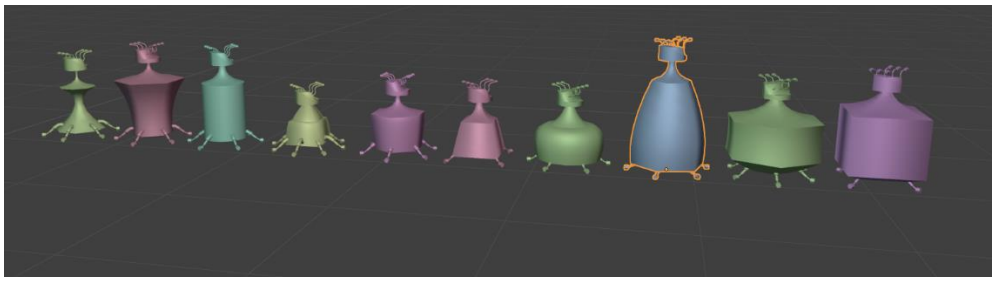


Figura 8.32

Si è anche creata una seconda tipologia di bottiglie, queste da collo allungato sono prese spunto dalla forma dell'insetto chiamato Auchenorrhyncha, mostrato nell'immagine qua affianco.

Figura 8.33: Auchenorrhyncha [21]

In questo caso l'idea era di una bottiglia che contenesse una bevanda che ha bisogno di ossigenarsi prima di essere bevuta; quindi il solo movimento del barista che versa il contenuto di questa speciale bottiglia fa sì che grazie al collo lungo la bevanda si ossigeni e sia già pronta da bere nel momento in cui il si versa tale bevanda nel bicchiere.

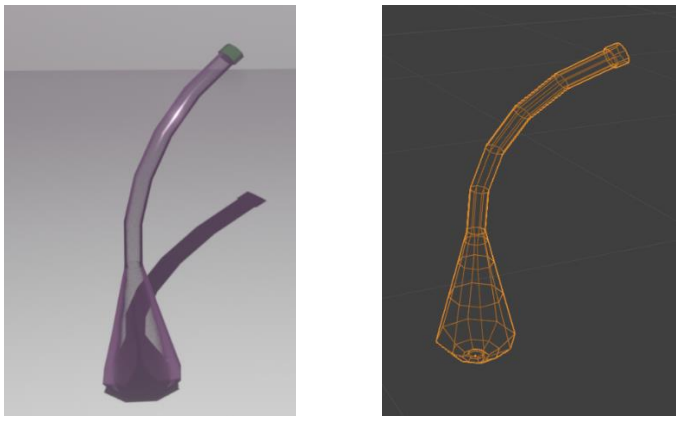


Figura 8.34: Risultato finale

Non si è riusciti ad unire le due realizzazioni in un'unica bottiglia perché sono completamente diverse l'una dall'altra, e non si è voluto snaturare troppo le due creazioni per unirle in modo forzato.

8.4.5 Divanetti:

Per quanto riguarda i divanetti invece, si sono prese come reference molte immagini tratte da videogioco Cyberpunk 2077 come nelle due immagini a destra, in cui si può notare come essenzialmente tali divanetti sono costituiti da una base che fa anche da illuminazione, poi il corpo centrale in cui si appoggiano i cuscini. Questi divanetti posso essere curvi o rettilinei in base anche al tipo di tavolino associato, infatti per comodità di utilizzo con tavolini a forma di ellisse è stato pensato un divanetto curvo, in modo che il cliente fosse comodo nel prendere e appoggiare il proprio drink sul tavolino; mentre per tavolini più piccoli o di forme più regolari si è pensato un divanetto lineare.

Figura 8.35: Reference divanetti [12]

Nella costruzione di tali divanetti la parte complicata nel divanetto curvo è stata la gestione della possibilità di modificare altezza e larghezza seduta, oltre che altezza, dimensioni e numero di cuscini.

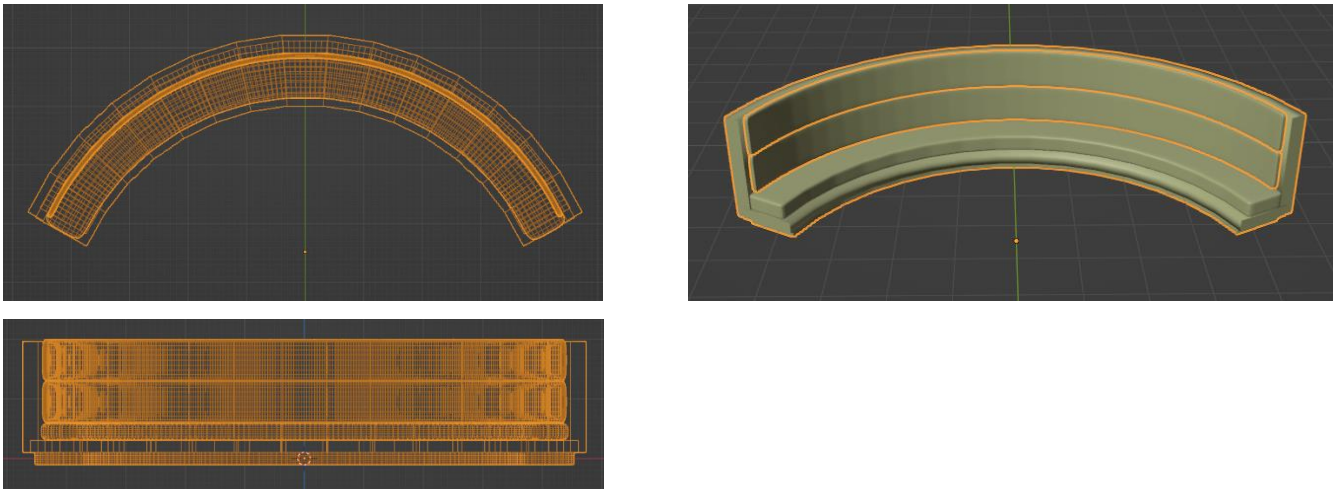


Figura 8.36

Invece in quello diritto la problematica principale, oltre alla gestione delle stesse modifiche, è stata quella di poter aumentare la dimensione stessa del divanetto. Questo si è riusciti a farlo grazie all'idea di creare un modulo divanetto da una seduta che poi venga ripetuto *n* volte in base al numero di posti di seduta

voluti In questo modo si è solo dovuti poi gestire i braccioli laterali per fare in modo che fossero solo nelle due posizioni più esterne.

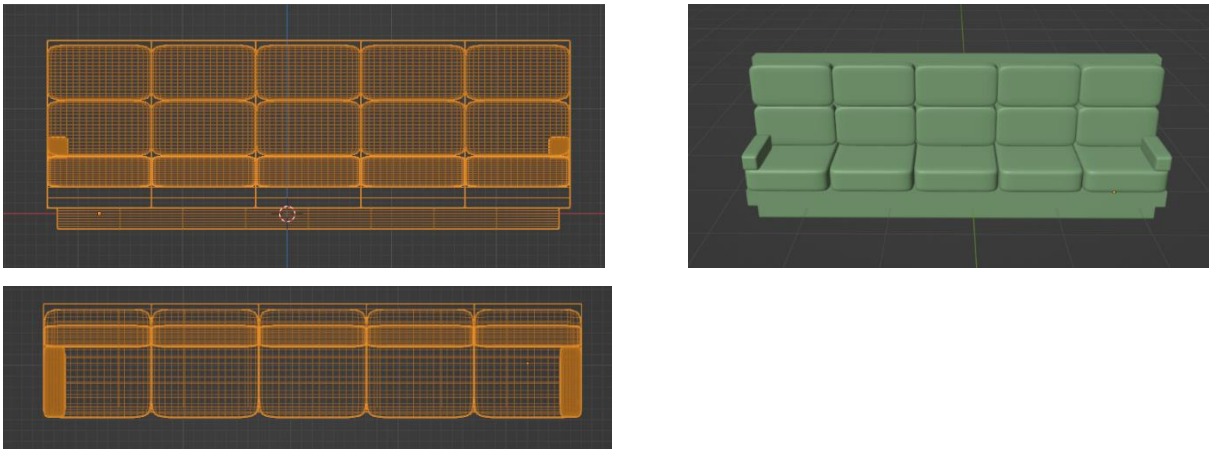


Figura 8.37

Dopo la prima revisione del lavoro, notando come i divanetti creati fossero anonimi, si è pensato un modo per caratterizzare questi due assets in modo che fossero riconducibili a questa serie. Per fare ciò dopo un'attenta analisi sul mondo degli insetti, si sono provate alcune soluzioni.

Per il divanetto diritto si sono modificati il cuscino di seduta, nella quale si è reso più geometrico, a forma di ottagono, in modo che riprendesse il tavolino corrispondente che verrà mostrato successivamente, e i braccioli che anch'essi sono stati manipolati per renderli più geometrici. Infine si è sostituita la base d'appoggio luminosa con una serie di gambe-zampe che vengono riprese da alcuni insetti, come millepiedi (Myriapoda [15]) in figura, che hanno molte basi d'appoggio. Queste gambe sono state costruite in modo da crearne una sola e poi in modo procedurale si è replicate per completare il modulo di singola seduta.

Figura 8.38: Myriapoda [15]

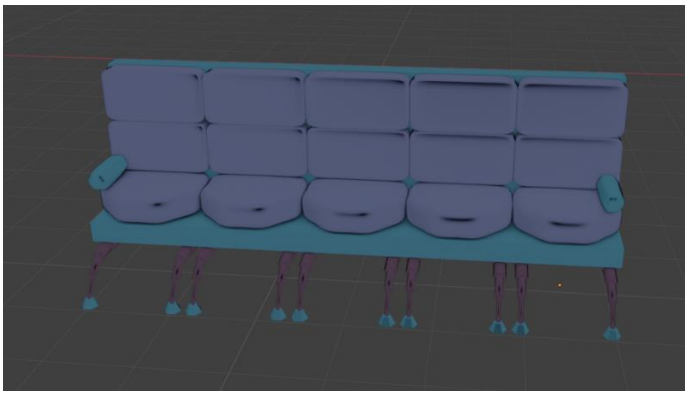


Figura 8.39: Risultato finale

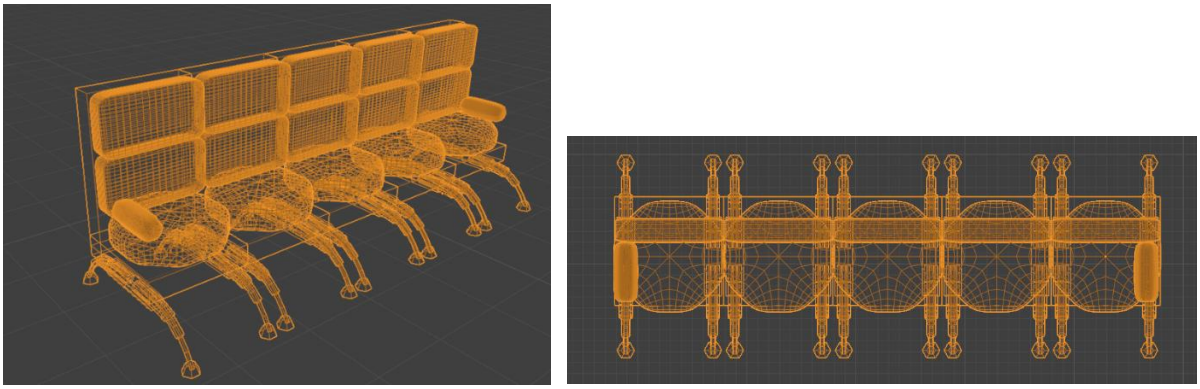


Figura 8.40

Invece per quanto riguarda il divanetto curvo, si è pensato di modificarne la curvatura per darne una simile al movimento di alcuni insetto come appunto il millepiedi dell'immagine sopra e aggiungere anche ad esso, come base d'appoggio al posto di quella luminosa, delle zampe-gambe che seguissero la curvatura principale del divanetto, mantenendo comunque la possibilità di poter modificare altezze e dimensioni del divanetto stesso e dei cuscini.

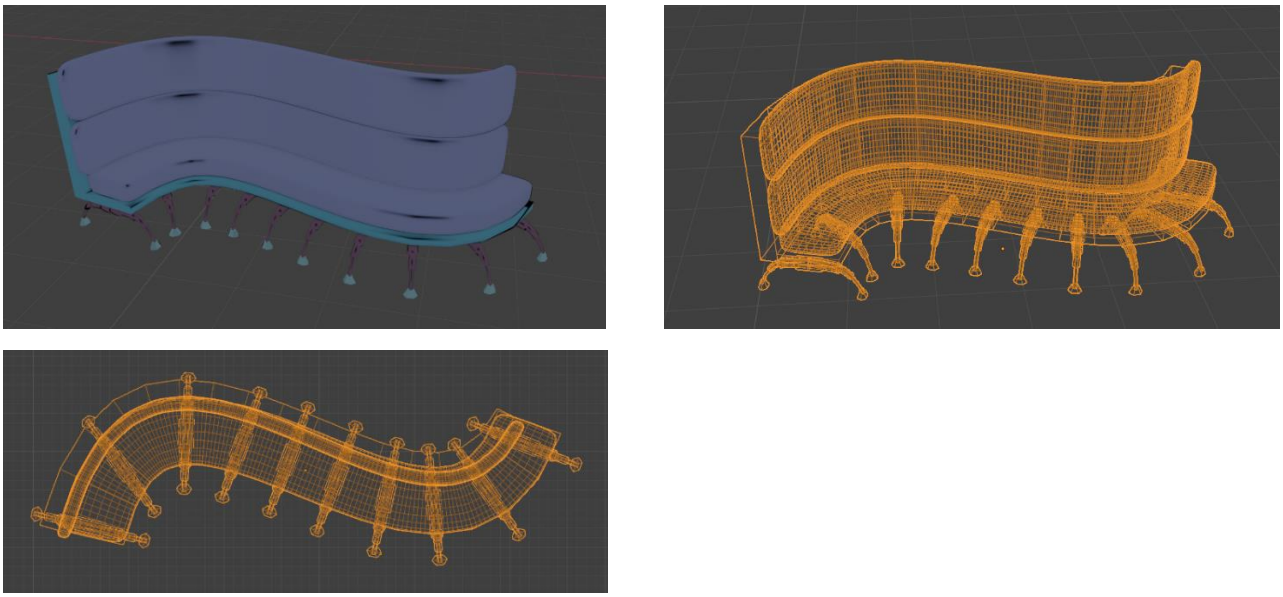


Figura 8.41: Risultato finale

Nel caso dei divanetti si può notare come ci siano molti vertici, questo è dovuto principalmente ai cuscini, volendo dare agli stessi un minimo di curvatura e non farli sembrare troppo dei blocchi duri e rigidi.

8.4.6 Sedie:

Queste sedie alte da bancone del bar sono state pensate attraverso reference tratte da Cyberpunk 2077, mostrate qua affianco. Nel nostro caso la sedia si può scomporre in quattro pezzi: la prima composta dalla seduta costituita da un cuscino e sotto una zona più rigida con delle insenature nelle quali sono ricavati delle luci a led che permettono di dare alla sedia una descrizione più futuristica. La seconda parte composta dall'unica gamba centrale formata da diverse parti cilindriche di varie dimensioni, nella quale si può notare anche qui le piccole insenature per le luci a led.

Figura 8.42: Reference sedie [12]

In generale la gamba è filiforme questo per dare una sensazione di leggerezza e di slancio.

Altra parte è il poggiapiedi anch'esso costituito da parti cilindriche e che si va ad incastrare tra la parte bassa della gamba e la base centrale, ultima parte, che riprende la forma della seduta e va in contrasto con la gamba creando una sensazione di robustezza.

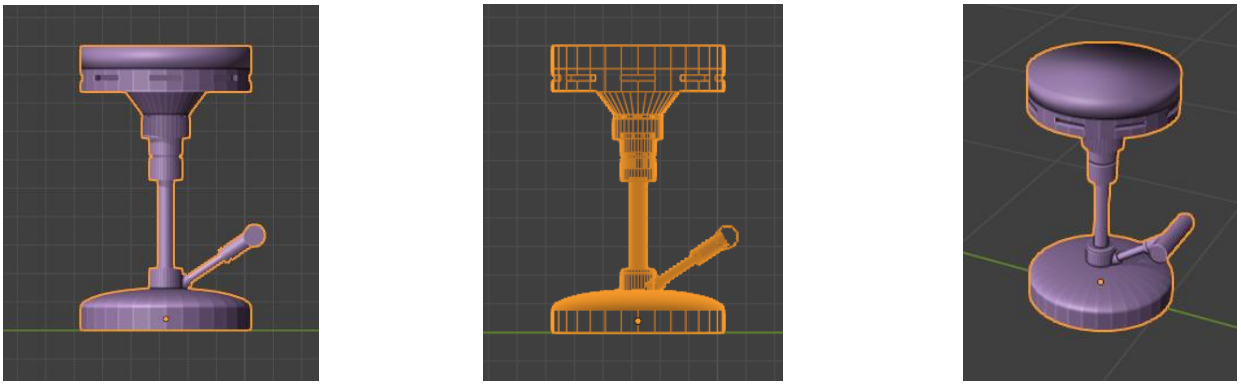


Figura 8.43

Questa sedia era stata pensata con l'obiettivo di poter modificare oltre all'altezza anche la forma per renderla più o meno geometrica.

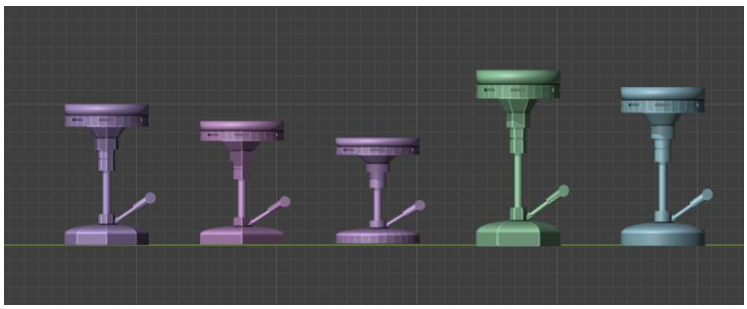


Figura 8.44

Dopo la revisione si è cercato un modo per darle un tocco di “insetto”, senza però stravolgere l’ossatura iniziale. Questo lo si è fatto in due modi: andando cioè a sostituire l’elemento obliquo che faceva da tramite tra seduta e gamba con dei bracci simili a zampe di insetto come il ”Nagusta goedelii” già visto nella sezione del bancone bar, in modo che desse anche un maggior senso di leggerezza e di slancio a tutta la sedia. L’altra modifica è stata sostituire la base, con delle zampe simili a quelle del ragno già visto nella sezione dedicata alle bottiglie. Si è comunque tenuto la possibilità di modificare l’altezza e la risoluzione della sedia, facendo in modo che anche bracci e zampe-gambe vengano aggiunte o rimosse in base alla risoluzione totale della sedia.

Figura 8.45: Reference insetti [19] [16]

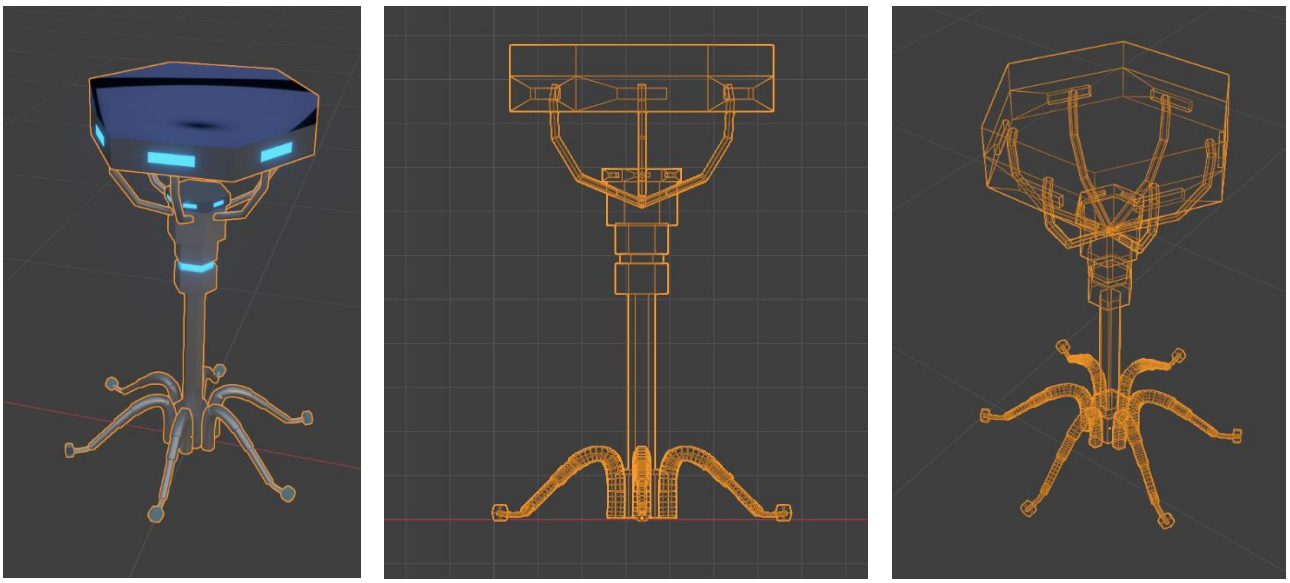


Figura 8.46: Risultato finale

Come si vede dall’immagine il poggiapiedi è stato rimosso in quanto la parte alta incurvata delle gambe forniscono una base di appoggio per i clienti.

8.4.7 Mensole:

Per quanto riguarda le mensole, all'inizio si era pensato di crearne con uno stile armonioso e curvo come nello stile dell'immagine di fianco, ma durante la costruzione mi sono reso conto che tale modello non era coerente con l'idea generale di avere forme quadrate e geometriche.

Figura 8.47: Reference mensole [22]

Inoltre l'idea di usare i Geometry Nodes era vista come una sfida per poter creare oggetti che, mettendo a posto i valori tramite nodi Math, Boolean e di altro tipo, permettessero di manipolare facilmente l'oggetto creato per essere disposto nei modi più coerenti con l'ambiente.

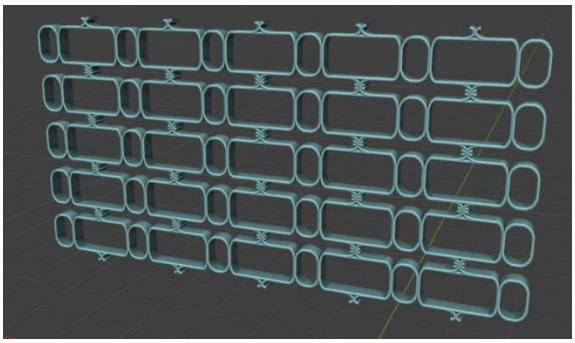


Figura 8.48

Per questo motivo ho poi preso alcune reference tra cui quella del bancone del bar nella quale dietro si intravede uno schema di scaffalature, che mi ha interessato e si è preso come punto di partenza.

Figura 8.49: Reference mensole [12]

Quindi in questo caso le mensole sono sostituite principalmente da un modulo che si ripete, questo perché appunto con la modellazione procedurale si vuole creare dei moduli che poi si possono ripetere facilmente invece di ricrearli da zero.

Questo modulo base è formato da una colonna costituita da due pareti verticali e due orizzontali, più dei

ripiani orizzontali in mezzo. Andando poi a ripetere questa colonna sull'asse orizzontale e variandone il numero dei ripiani si arriva alla costruzione di un modulo superiore formato da tre colonne, le quali vengono ulteriormente ripetute per creare la mensola definitiva.

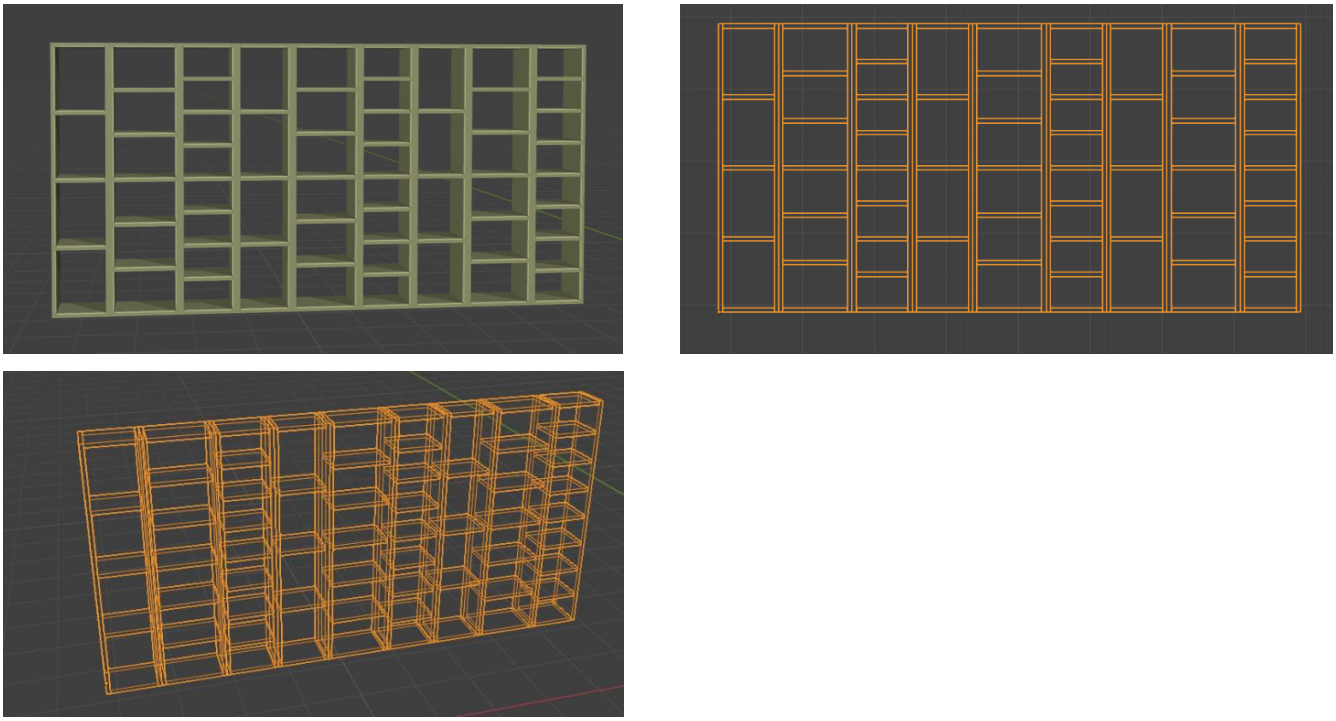


Figura 8.50

Anche qua dopo la revisione si è pensato qualcosa di diverso, che fosse coerente con le nuove parti create in stile "mondo dominato da insetti", perciò si è pensato di rimanere coerenti con l'ambiente nella quale sarebbe stata inserita questa mesh, ovvero dietro il bancone del bar. Quindi si è pensata la mensola come delle cellette di un alveare in cui inserire bicchieri e bottiglie.

Figura 8.51: Alveare usato come reference [14]

Detto questo si è quindi costruita la mensola cosi come si vede nella figura sottostante, partendo anche qua da un modulo formato da una sola cella, la quale è poi stata ripetuta in altezza e lunghezza.
Si è riusciti a tenere la possibilità di aumentare o diminuire il numero di istanze in altezza e larghezza, ma anche la possibilità di variare la profondità delle cellette.

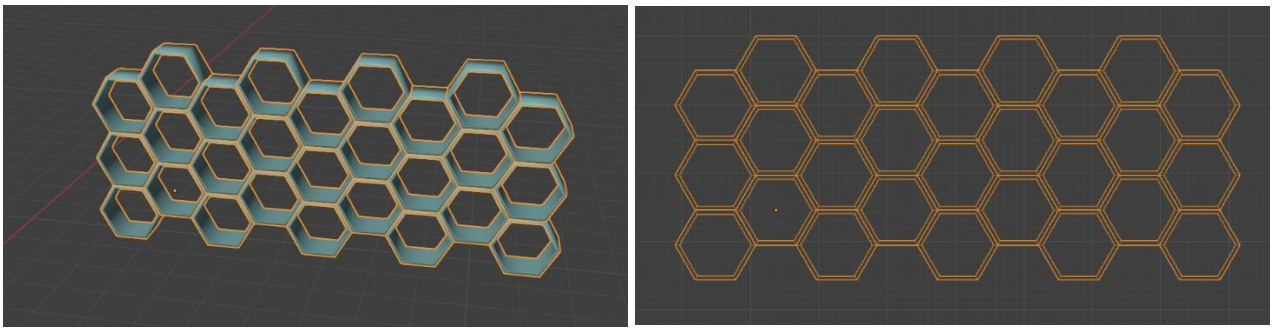


Figura 8.52: Risultato finale

8.4.8 Poltrone:

Parlando invece di poltrone, anche in questo caso si è partiti con qualche idea ispirata dal videogioco di CD Projekt RED, come si può vedere dalle immagini qui affianco.

Figura 8.53: Reference poltrone [12]

Dapprima si sono provate le due immagini 8.52 e 8.53, dove nel primo caso, la prima immagine, la struttura è formata da due braccioli laterali che tengono al centro la seduta, la quale è un tutt'uno con la parte posteriore. La base di appoggio è formata da un cilindro basso e largo nella quale si va a innalzare un tubo più sottile che collega la base alla seduta. In questo caso si poteva variare l'altezza del tubo, per modificare l'altezza della sedia.

Figura 8.54: Reference poltrone [12]

La seconda prova invece è stata la poltrona nella seconda immagine, nella quale si nota che è un pezzo unico, ed è praticamente una poltrona a "S" in cui la parte alta rimane diritta permettendo così di ottenere lo schienale.

Qua sotto vengono mostrate le prime due realizzazioni.

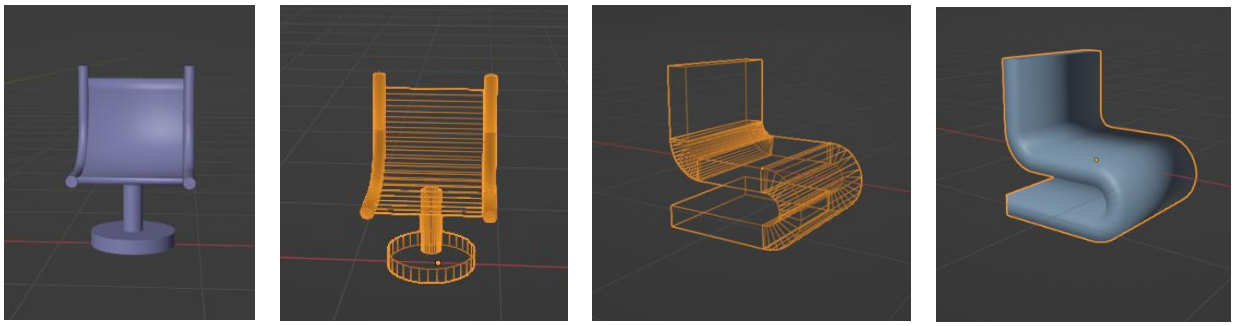


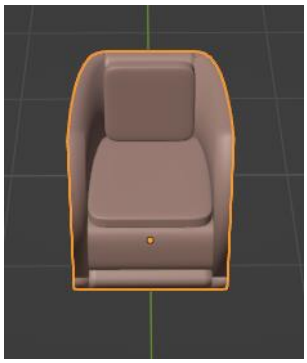
Figura 8.55

Queste due realizzazioni però non mi convincevano molto, per cui, trovando un'altra reference, mostrata a destra, ho provato a realizzarla.

Figura 8.56: Reference poltrone [12]

Essa può essere suddivisa in diverse parti: una prima parte è la base illuminata a led che permette di essere notata anche in situazioni di scarsa luminosità, quando il locale ha le luci soffuse.

Si ha poi una zona centrale che è la parte portante della poltrona e che permette di sorreggere il peso, sulla quale è posto un cuscino. Cuscino che viene posto anche nella parte posteriore. Infine la parte laterale che è un tutt'uno con la parte posteriore che permette al cliente di sentirsi avvolto all'interno di tale poltrona.



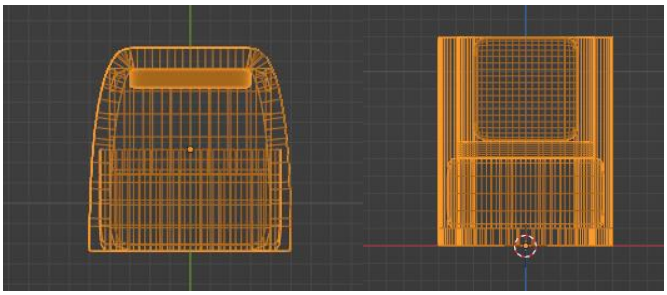


Figura 8.57

Dopo la revisione ho pensato di creare delle reference tramite IA, delle quali le più interessanti sono quelle mostrate di seguito, da qui allora ho ripensato il progetto per le poltrone e ne sono venute fuori di due tipologie. La prima versione cerca di avvicinarsi alla reference IA di sinistra, nella quale si sono aggiunte una piramide capovolta in basso che vuole assomigliare ad un pungiglione e che serve come sostegno centrale, le antenne con strisce a led, prese dalla seconda immagine, servono per poter riconoscere la posizione delle sedie anche al buio, soprattutto per i camerieri che devono servire ai tavoli. Infine i due "occhi" che in realtà sono casse acustiche per avere una musica di sottofondo anche ai tavoli.

Figura 8.58: Reference poltrone generate con IA [17]

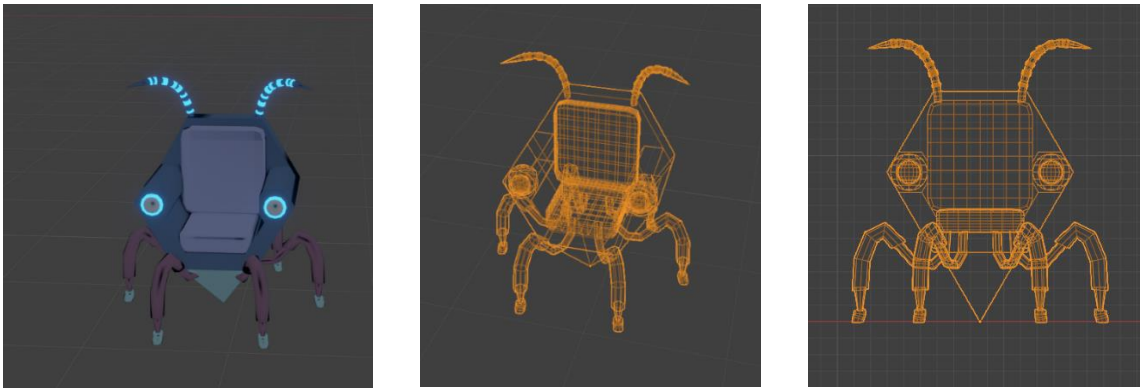


Figura 8.59: Risultato finale

Nel secondo caso si sono tenute le antenne, il pungiglione e le casse che però quest'ultime sono state spostate sullo schienale come nell'immagine di reference qua di fianco, mentre per la forma della poltrona ci si è rifatti al concetto dell'alveare in cui si può paragonare il cliente ad un ape; quindi la poltrona diventa una celletta che permette al cliente di riposarsi e consumare il proprio drink comodamente; questo anche perché tale poltrona rimane più spaziosa di quella precedente ed ha un cuscino per ogni lato di appoggio.

Per quanto riguarda le gambe della poltrona sono in entrambi i casi presi spunto sia dalla prima che da quest'ultima immagini create tramite intelligenza artificiale.

Figura 8.60: Reference poltrone generate con IA [17]

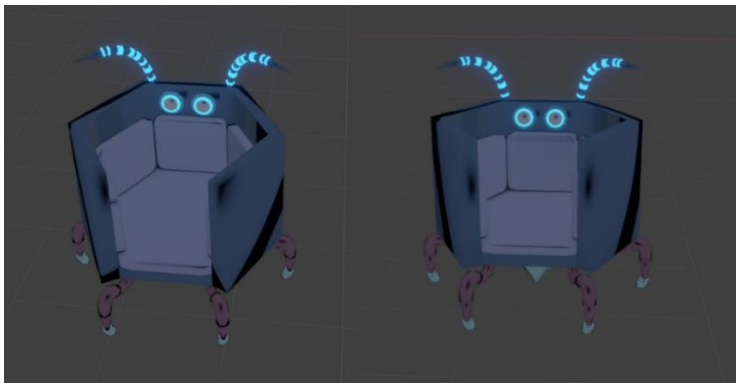


Figura 8.61: Risultato finale

Anche qua, come per i divanetti ci sono un po' più di vertici per i cuscini, anche se si è cercato di limitarne la quantità.

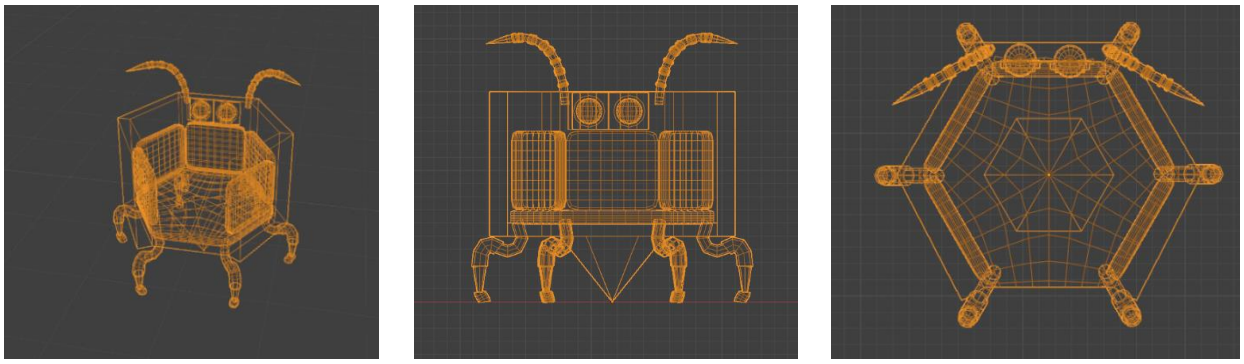


Figura 8.62

8.4.9 Porte:

Come punto di partenza per la creazione delle porte si è preso nuovamente spunto da Cyberpunk 2077. In questo caso si sono trovate diverse reference che si sono volute provare, infatti nel nightclub c'era bisogno di una porta ingresso, le porte del piano superiore che portano a stanze private e infine una porta di sicurezza nel piano terra.

Figura 8.63: Reference porta [12]

Quindi si era pensato la porta vetrata grossa come ingresso principale, la quale è formata da due blocchi laterali metallici che fanno da cardine, le due grosse vetrate e delle decorazioni anch'esse metalliche nella divisione delle due vetrate.

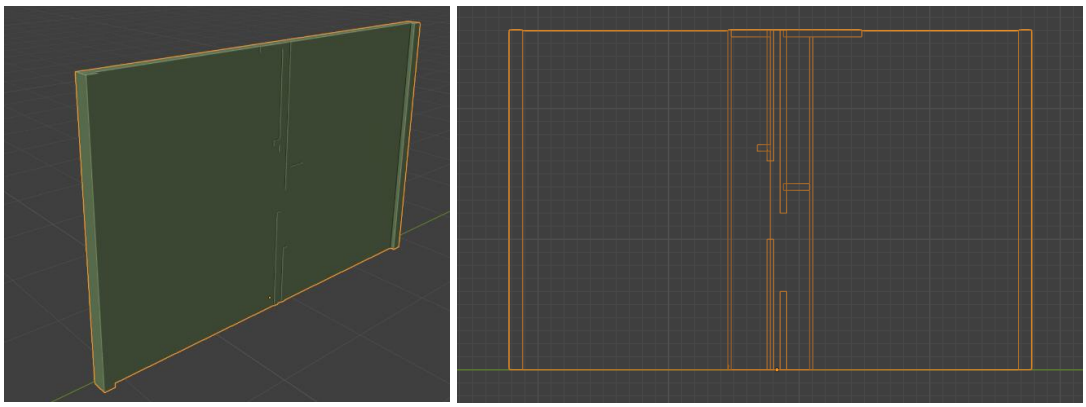


Figura 8.64

Per quanto riguarda invece la porta di sicurezza si voleva una porta possente che sembrasse impossibile da buttare giù, quindi con linee molto dritte e con una piccola finestrella o addirittura senza, mentre la maniglia ha integrato un tastierino in quanto apribile solo tramite codice d'accesso.

Figura 8.65: Reference porta [12]

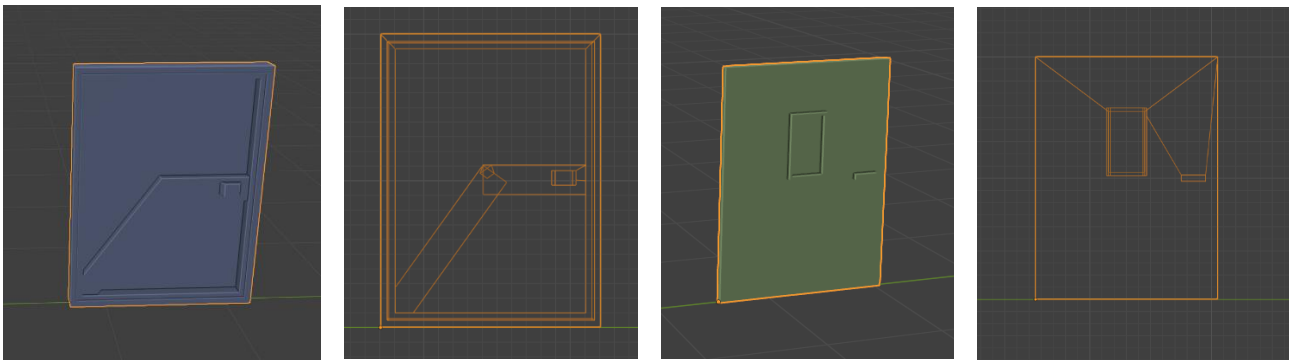


Figura 8.66

Infine per le porte delle stanzette private al piano superiore si cercava una tipologia di porta non vetrata per dare privacy ai clienti che decidevano di trascorrere del tempo con le prostitute del locale, ma che fosse più gradevole nell'aspetto. Quindi si sono pensate due tipologie di decorazioni di tipo geometrico, poi si è preferita la seconda in quanto il numero delle strisce orizzontali indicava il numero della stanza.

Figura 8.67: Reference porta [12]

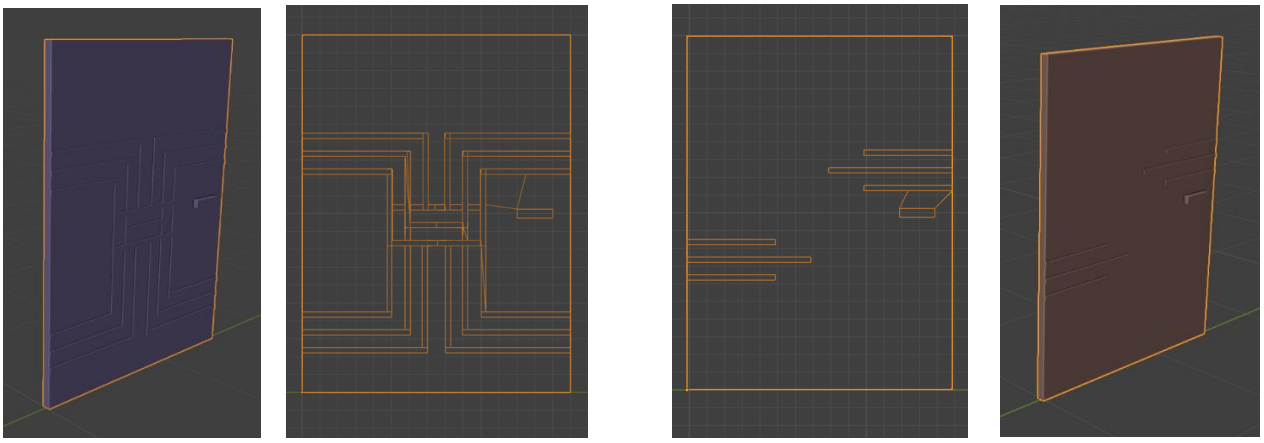


Figura 8.68

Dopo la revisione, notando effettivamente che le porte erano un po' troppo semplicistiche e che non avessero nulla a che fare con il mondo di Reverie, si è fatta un'ulteriore analisi e ricerca per capire come crearle. Grazie all'aiuto di Copilot quindi si sono riuscite a ricavare alcune reference che mi hanno dato delle idee. Quindi si sono create alcune tipologie di porta dalle immagini di reference mostrate di lato di ogni spiegazione. La prima richiama ad altri asset già creati come il bancone del bar,

Figura 8.69: Reference porte generate con IA [17]

le mensole e una tipologia di poltrona, ovvero lo schema dell'esagono e dell'alveare. Quindi si ha una porta esagonale, con dettagli e decorazioni anch'essi esagonali come fosse l'entrata di un alveare. Lo stipite della porta anch'esso esagonale ha nella parte destra un tastierino che permette di inserire il codice. Infine tutto il complesso è accessibile grazie a tre gradini, sempre di forma esagonale, che permettono agevolmente di arrivare all'altezza della porta. Questa tipologia di porta è stata pensata come porta sicurezza in quanto da una sensazione di possenza.

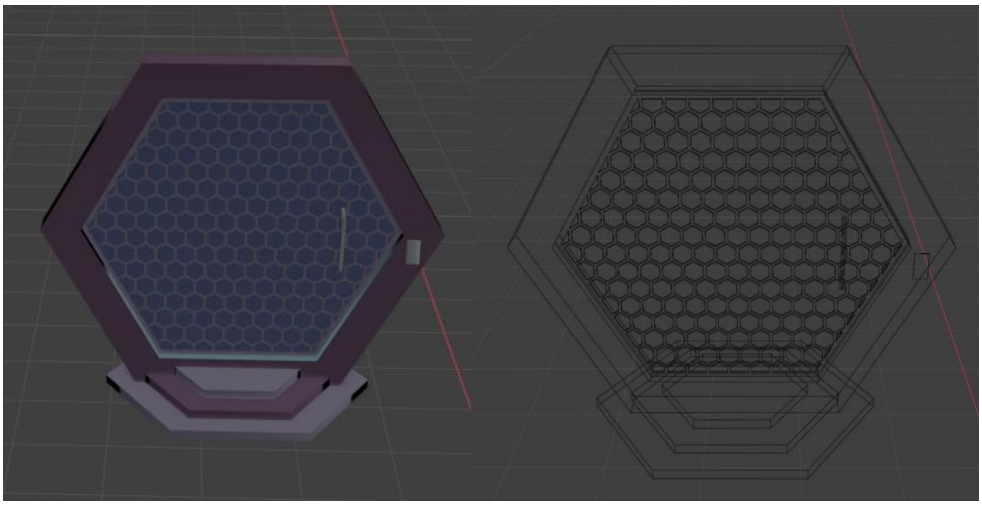


Figura 8.70: Risultato finale

Partendo poi da questa porta andando a strecciare in lunghezza la porta in modo da creare due ante, si è così creata la porta di ingresso, la quale a differenza di quella precedente, non ha né il tastierino né le maniglie in quanto è pensata per essere una porta a scorrimento. Si è voluto tenere questa porta come ingresso perché dà la sensazione di maestosità ed inoltre caratterizza molto il locale che il cliente troverà al suo interno.

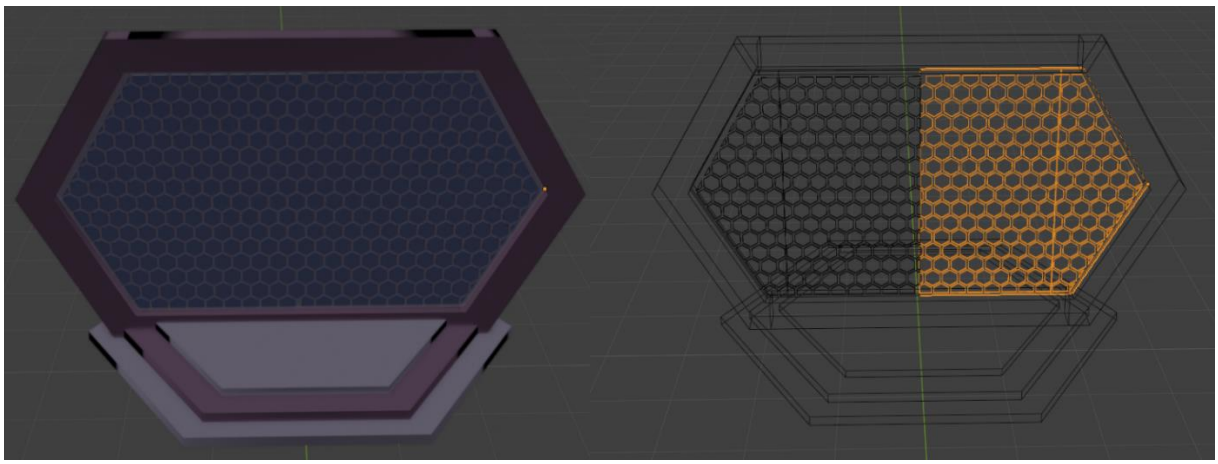


Figura 8.71: Risultato finale

Infine per quanto riguarda le porte delle stanzette al piano superiore si sono volute creare due differenti porte, in realtà sono tre ma solo due utilizzate, per distinguere i due lati lunghi del locale in cui sono presenti le stanze.

Esse sono porte di forma ovale come nella seconda immagine di reference e hanno in un caso come decorazioni delle ali di farfalla piene o vuote, dall'altra parte delle protuberanze che vogliono assomigliare a delle zampe di insetto. La loro finalità in entrambi casi è quello di indicare il numero della stanza. Entrambe le tipologie hanno tre scalini per poter accedere alla porta che non sono esagonali come la porta di ingresso e di sicurezza, ma bensì rotondi, in modo da essere coerente con l'intero oggetto. Infine tutte e due le tipologie hanno delle antenne luminose, create al posto di strisce a led poste sopra alle porte, che vogliono segnalare, con acceso-spento oppure cambiando colore, ai clienti e alle prostitute quali stanze siano occupate e quali invece siano libere.

Figura 8.72: Reference porte generate con AI [17]

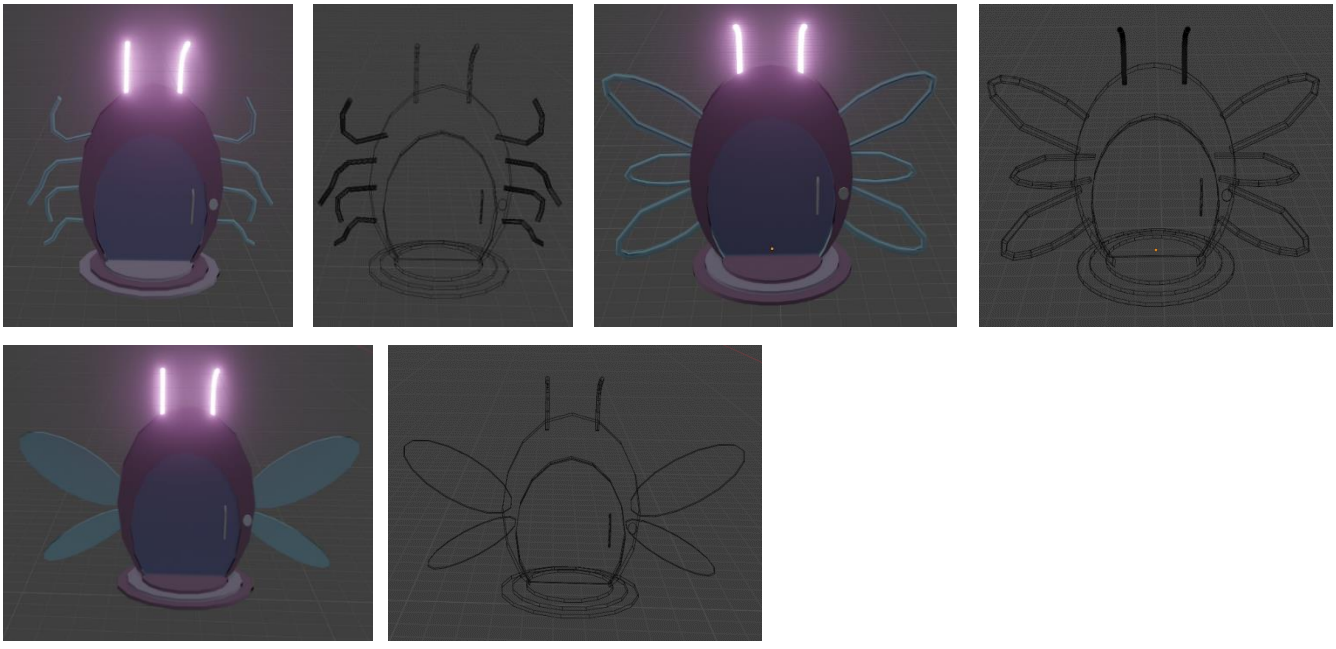


Figura 8.73: Risultato finale

8.4.10 Scala:

Per quanto riguarda le scale, esse erano già state create da un mio collega, ma dando un occhiata, ho notato che esse avevano un numero di vertici elevato; a destra la scala originale in questione.

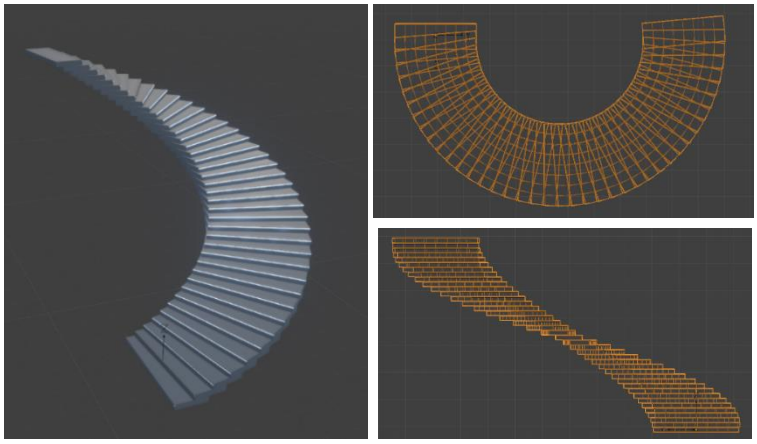


Figura 8.74

Per questo motivo ho voluto provare a crearne una copia utilizzando i Geometry nodes ed il risultato è più che soddisfacente in quanto non solo quelle create hanno praticamente la stessa curvatura di quelle originali, ma il numero di vertici è sceso di un ordine di grandezza. Inoltre grazie alla grande malleabilità dei geometry nodes, con lo stesso albero dei nodi si sono riuscite a fare non solo le due scale a chiocciola ma anche il paio di gradini che le precedono e anche i due gradini che portano alla porta di ingresso. Questo ha permesso ulteriormente di abbassare il numero di vertici oltre che essere la via più veloce per creare e modificare un ambiente.

Qua di fianco è presentata la scala con i Geometry Nodes, mentre di sotto l'insieme di tutte le scale modificate grazie all'unico oggetto.

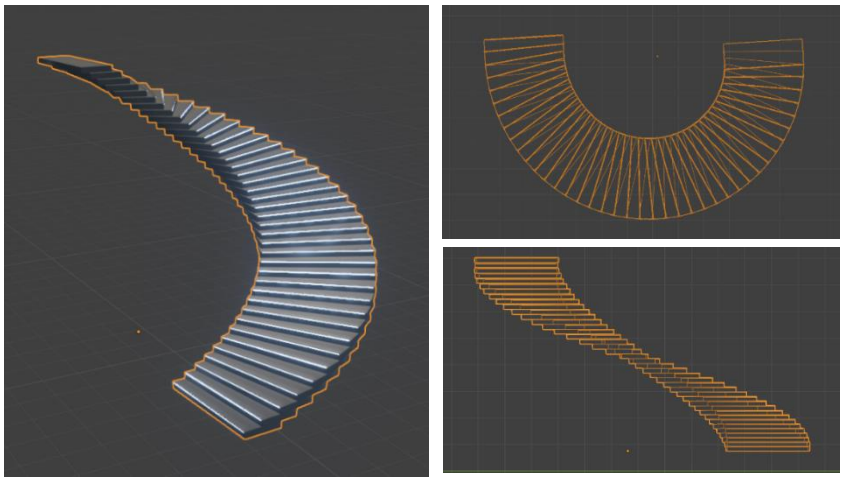


Figura 8.75

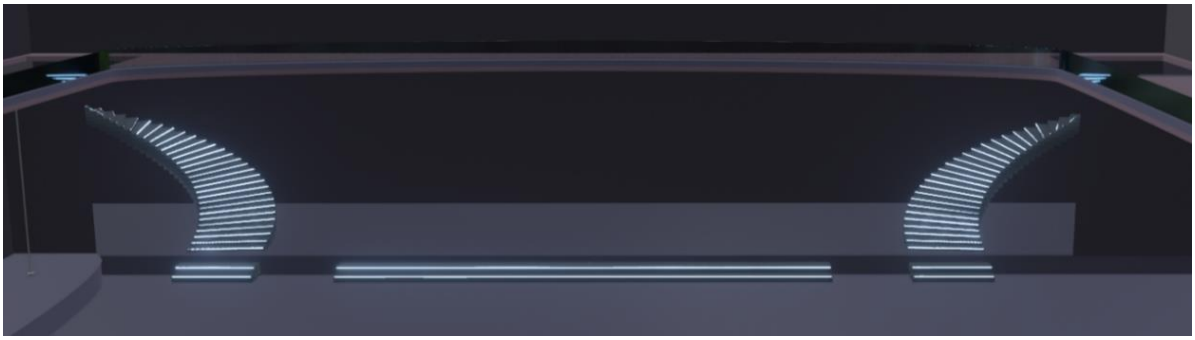


Figura 8.76: Risultato finale

8.4.11 Strutture luci:

Per quanto riguarda l'impianto luminoso, si sono cercate reference di diverso tipo perché era necessario avere diverse fonti di luci, in quanto in un ambiente come il nightclub si vuole avere una luce soffusa e debole, quindi meglio metter più fonti di luce a bassa intensità piuttosto che poche luci ma ad alta intensità.

Figura 8.77: Referenze strutture luci [12]

Come reference sono state prese alcune immagini di Cyberpunk 2077, ma anche dal canale YouTube "PRO BLENDER" [23], il quale in un tutorial ha mostrato come generale una struttura con luci in Geometry Nodes. Le immagini di queste reference sono le quattro mostrate a lato.

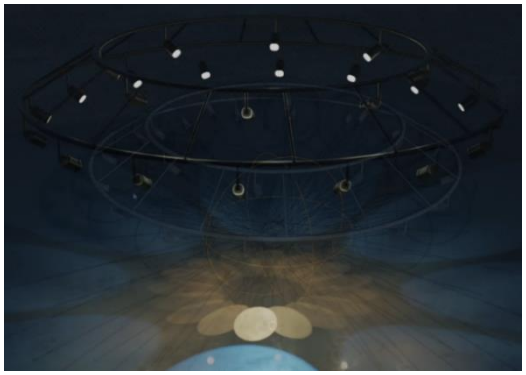


Figura 8.78 Referenza struttura luce [23]

Quindi si è pensato un piccolo faretto di forma esagonale da mettere attorno al bancone del bar, in modo da ottenere una fonte di luce per i baristi ed è stato pensato di tale geometria per essere coerente con lo stile del bancone. Questo faretto è formato da una base esagonale, una piastra a led e un vetro che va a coprire tale piastra permettendo una maggior diffusione della luce. In questo caso si è costruito il faretto in modo da poter variarne la forma geometrica, da quadrata fino a rotonda aumentano i vertici della struttura, ma anche poter aumentare o diminuire la dimensione del vetro e l'altezza di tutta la struttura.

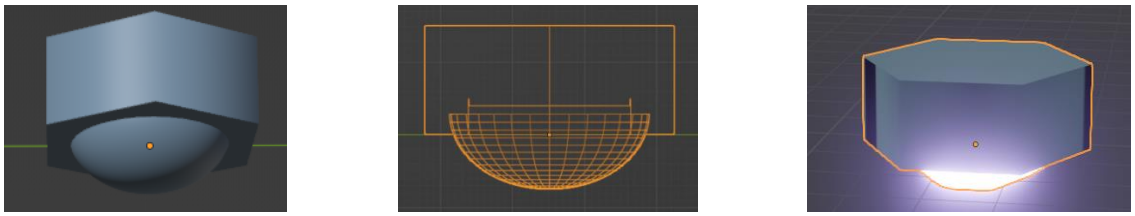


Figura 8.79

Poi si è pensato ad una struttura luminosa presa spunto dalla seconda immagine, e da posizionare sopra ai tavolini sullo sfondo della stanza, nel lato destro. Tale oggetto è composto da una struttura metallica con una piastra luminosa incastonata ed il tutto è tenuto su da alcuni fili metallici. In essa si può variare la dimensione del parallelepipedo e la lunghezza dei fili.

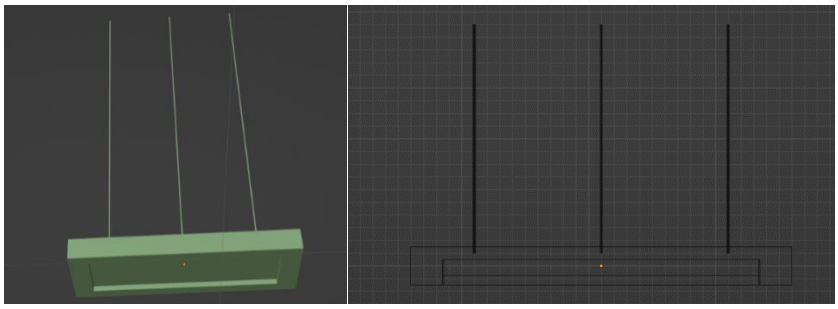
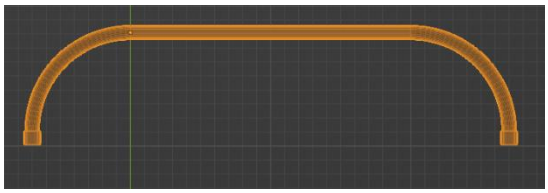


Figura 8.80

Si è creato poi un tubo a neon formato da due parti, la prima è appunto il neon di cui si può decidere la lunghezza, la seconda invece è la connessione metallica che permette di essere fisata al muro. Queste luci inizialmente sono state pensate per essere messe sopra a tutte le porte, ma dopo la revisione si sono sostituite con le antenne nelle strutture delle porte, per quanto riguarda le porte al piano superiore, mentre tale tubo è rimasto nella porta sicurezza.



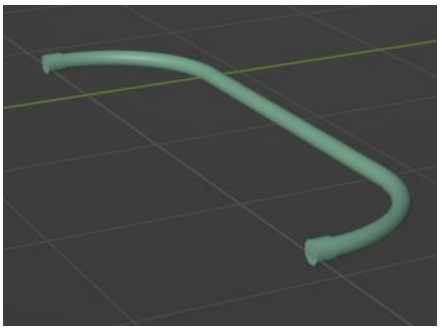


Figura 8.81

Altra fonte di luce sono i faretti a muro formati dal faretto vero e proprio come nella reference della pagina precedente e da un blocco metallico che tiene tale faretto incollato al muro. Tale connessione può essere modificata andando a decidere l'altezza delle barre metalliche e del tubo metallico, ma anche l'altezza della circonferenza che circonda il faretto. Mentre per quanto riguarda quest'ultimo si può modificare altezza, raggio e risoluzione, quindi passando da cubico a circolare, ma anche decidere la curvatura della parte alta del faretto.

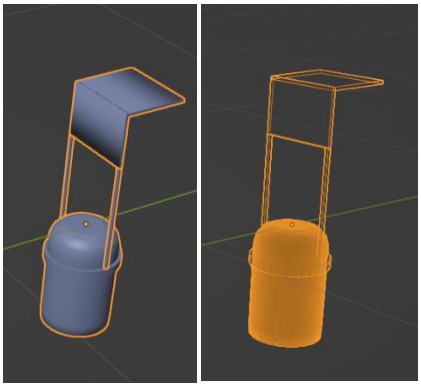
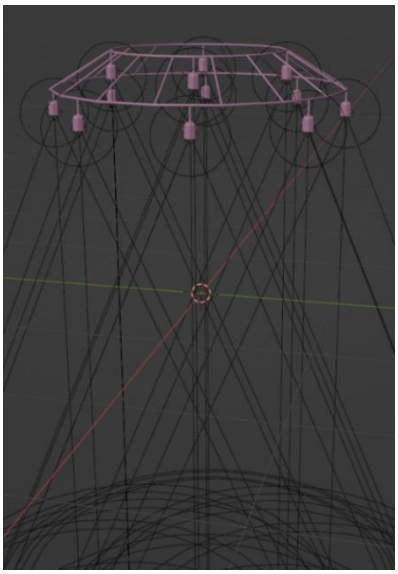


Figura 8.82

Ultima struttura è quella che permette l'illuminazione della zona centrale del locale nonché probabilmente la più presente nell'ambiente 3D. Essa è stata presa spunto dal tutorial di "PRO BLENDER" che è formata da dei faretti simili a quelli descritti in precedenza, con alcune differenze in quanto hanno un'altra struttura di contorno, la quali però può essere modificata in risoluzione, dimensione e altezza di tutti i dettagli. Inoltre ad ogni faretto è associato una luce di tipo spot, la quale può essere gestita dall'apposita scheda *object data properties*. In più possono essere modificate le altezza, grazie al tubo metallico che permette ai faretti di essere collegati alla struttura sovrastante. Tale struttura è formata da due anelli di circonferenze diverse, posizionati a diverse altezze, collegati tra di loro da tubi metallici, in cui sono collegati due file di faretti, una per ogni anello. In questa struttura si possono modificare le circonferenze dei due anelli e le altezze, facendo però attenzione a modificare correttamente i tubi metallici obliqui in modo da combaciare con i due anelli.



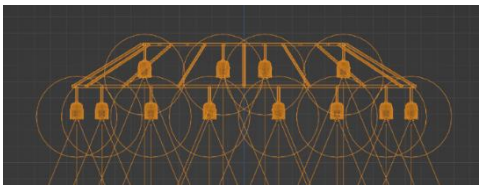


Figura 8.83

Dopo la revisione non si sono modificate troppo le strutture in quanto sono elementi piccoli e che con le fonti di luci accese non permettono troppo di vedere la costruzione dell'oggetto, ma comunque si sono apportate delle modifiche. Tra queste come abbiamo già accennato prima, ci sono l'eliminazione dei tubi a neon sulle porte del secondo piano e il cambio di geometria da rotonda a esagonale per il neon posto sopra la porta di sicurezza e per i faretti del bancone del bar.

Figura 8.84: Risultato finale

Anche per i faretti si sono modificate le risoluzioni in modo da tornare su geometrie semplici, sia per questioni di coerenza sia per ridurre il numero di vertici.

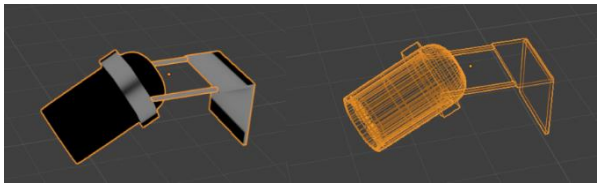
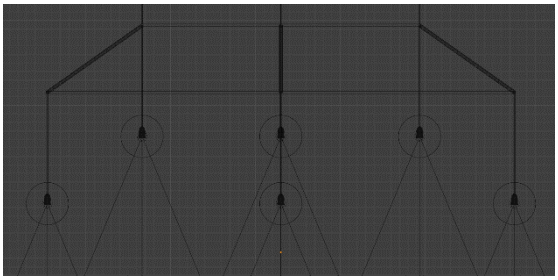


Figura 8.85: Risultato finale

Altre modifiche sono state il cambio di geometria riducendo la risoluzione fino a sei vertici per la struttura delle luci con reference il video di “PRO BLENDER”. Questo permette anche di ridurre il numero di faretti e di tenere un’illuminazione più tenue e soffusa.



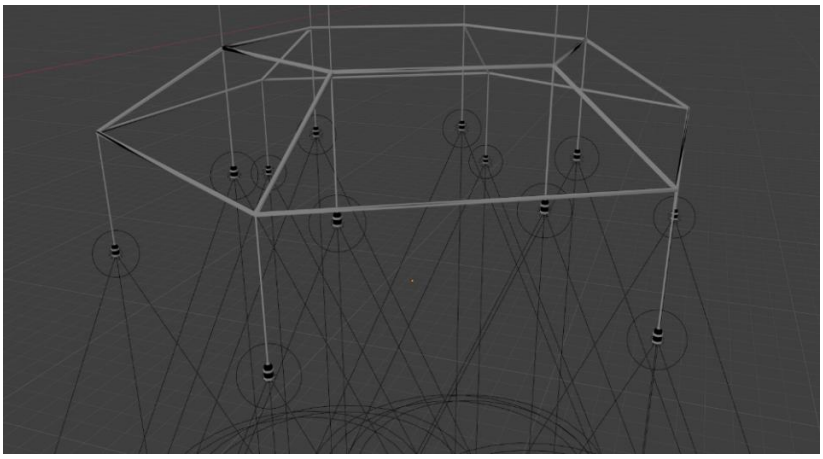
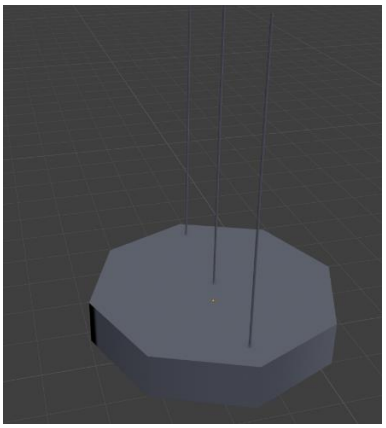


Figura 8.86: Risultato finale

Infine per la struttura a parallelepipedo, si è pensato invece ad una struttura ad ottagoni per essere coerenti con i tavolini sottostanti che verranno spiegati in seguito.



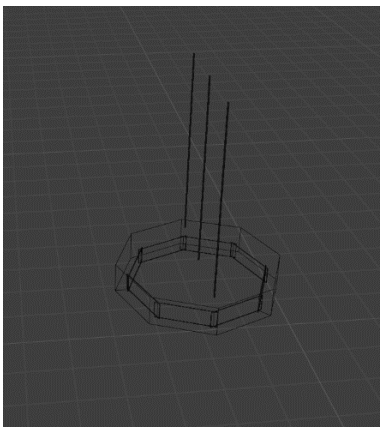


Figura 8.87: Risultato finale

In tutti gli asset modificati si sono tenute le possibili modifiche permettendo così di cambiare design in corso d’opera e poter riutilizzare gli stessi asset per altre ambientazioni cambiando soltanto alcuni valori.

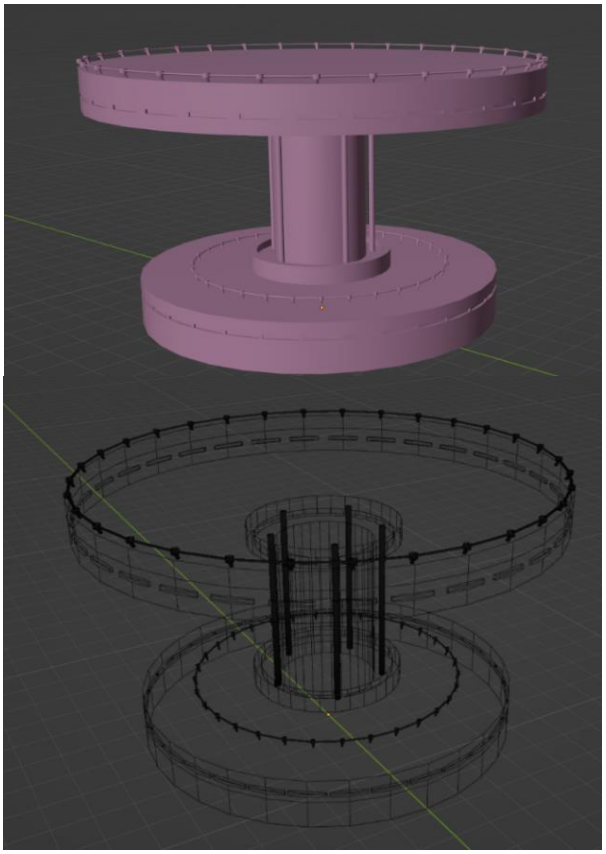
8.4.12 Struttura ologramma:

La struttura sopra la quale vi è l'ologramma della testa di una persona è stato pensato come se fosse un altare o comunque qualcosa che dovesse spiccare all'interno della sala: doveva quindi avere grosse dimensioni. Nel frattempo che si cercavano immagini di reference per altri oggetti sempre sul videogioco della casa polacca, si è trovata una struttura che in quel contesto veniva usata da alcuni personaggi come palco per ballare.

Figura 8.88: Reference Struttura ologramma [12]

Data la particolarità di questa struttura si è deciso che essa poteva fare al caso nostro.

Quindi si è provato a ricostruire tale struttura, che in generale può essere divisa in alcune zone. La base, la quale ha degli incavi in cui sono posizionate delle luci a led. Al di sopra di essa c'è una sorta di recito circolare al cui interno troviamo una grossa colonna di vetro accerchiata da alcune colonnine di metallo. Infine si ha la parte superiore, quella che sorregge l'ologramma, che ha la stessa struttura della base ma con un diametro maggiore, e ha anch'essa il recito circolare, con un raggio superiore a quello della base, ed esso fa da recito all'ologramma.



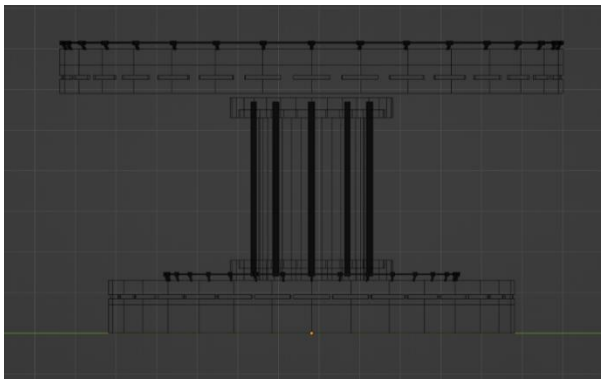


Figura 8.89

Dopo la revisione si è voluti modificare la struttura per essere più coerente con il resto dell'ambiente senza però stravolere il lavoro già svolto. Quindi si è pensato di ridurre la risoluzione di gestirla in modo autonomo grazie al nodo *Group Input.* Tale risoluzione è stata portata a otto per avere una base ottagonale che riprendesse i tavolini vicino, ma anche perché essendo una struttura centrale nella stanza si voleva qualcosa di simmetrico e che non fosse eccessivamente spigoloso come un esagono. Inoltre si è voluti cambiare le altezza della colonna centrale, della parte superiore e della base per permettere, tenendo sempre la stessa altezza, di usare anche qua come nella maggior parte degli asset, delle zampe di insetto che fanno da sostegno a tutta la struttura.

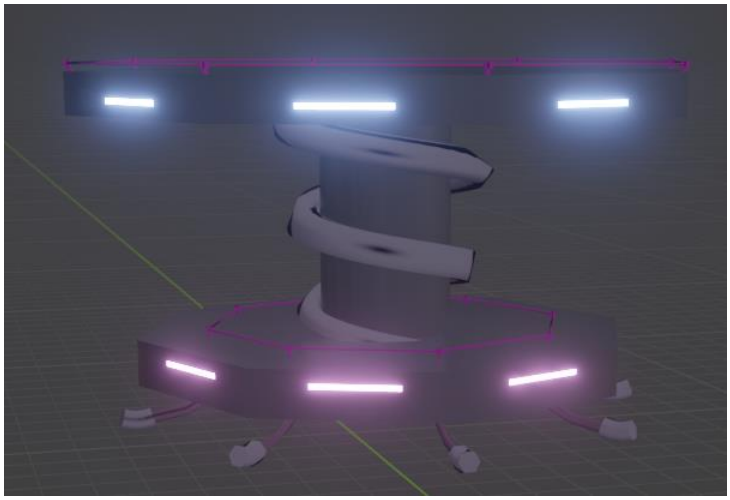


Figura 8.90: Risultato finale

Infine al posto delle colonnine di metallo si è voluto mettere una spirale di metallo, che voglio ricordare la forma di alcuni insetti invertebrati quando si arricciano su sé stessi, come i vermi nell'immagine qua affianco.

Figura 8.91: vermi usati come reference [24]

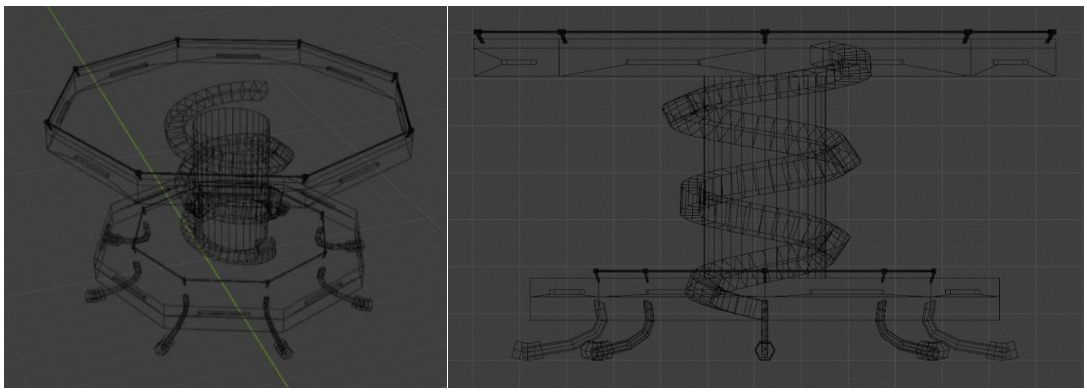


Figura 8.92

In questo caso si è anche lasciato la possibilità di modificare il raggio della struttura, e modificandolo si cambia il raggio di tutte le parti in modo omogeneo. Questo grazie all'uso di nodi *Math*, e le funzioni *Divide* e *Add*.

8.4.13 Tavoli:

Per quanto riguarda i tavolini, si sono cercate molte reference, soprattutto nel mondo di Cyberpunk, in quanto si volevano creare più tipologie da inserire, con magari "zone a tema".

Quindi si sono creati tavolini a base quadrata e a base circolare, dove in quest'ultimi poi è stato modificato l'albero dei nodi in modo da poter cambiare risoluzione dei cilindri e farli diventare da triangolari a circolari passando per le varie forma geometriche di base come esagoni e ottagoni, in modo da avere un solo asset che poteva essere modificato. In questi tavolini che partivano con una base di raggio circa 1.5 metri e una gamba sola, si potevano modificare altezza e dimensione di ogni singolo elemento.

Figura 8.93: Reference tavolo [12]

Infatti il tavolino può essere suddiviso nel top del tavolo e nella gamba, che a sua volta ha tre suddivisioni, gestibili in modo separato. Infine se il tavolo era di grosse dimensioni si poteva anche modificare il numero di gambe portandole fino a quattro.

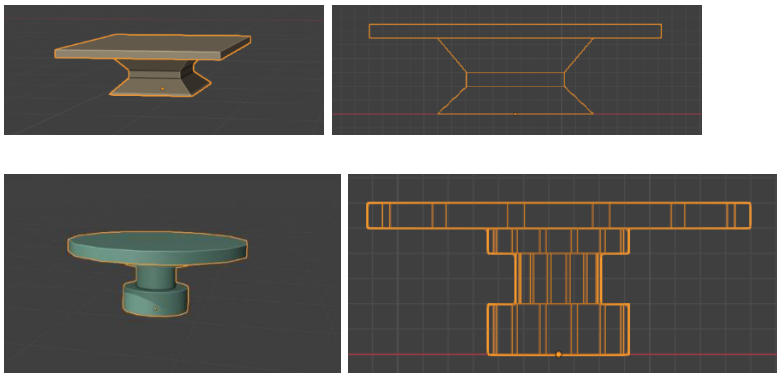


Figura 8.94

Altro tipo di tavolino era questo in figura a lato, il quale prevedeva di avere la base di appoggio larga quanto il top del tavolo e nella base aveva anche degli inserti luminosi. In realtà tale tavolo aveva la stessa base dell'albero dei nodi di quello precedente, ma è stato tolto la possibilità di modificare il numero delle gambe per permettere di inserire gli inserti luminosi.

Figura 8.95: Reference tavolo [12]

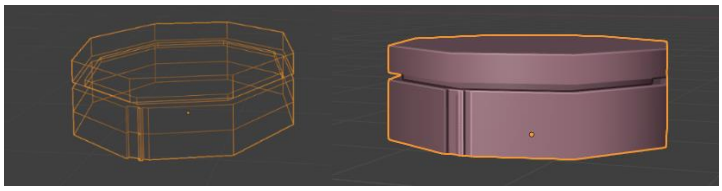
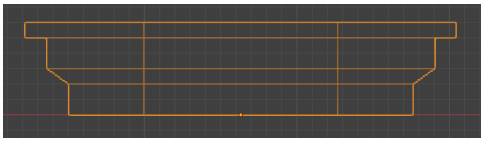
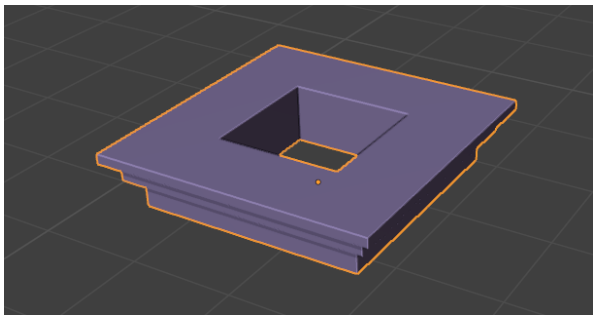


Figura 8.96

Figura 8.97: Reference tavolo [12]

Un altro tavolo sempre di forma geometrica è quello presentato a lato, il quale voleva essere un grosso tavolo, abbastanza alto da non necessitare l'uso di sedie e poltrone, ma piuttosto un tavolo dove poter rimanere in piedi mentre si guardano le ballerine danzare. Anche in questo caso esso può essere diviso in top e gamba, la quale ulteriormente è divisa in tre parti; tutte le quattro parti possono essere modificate a piacimento in altezza, larghezza e risoluzione per avere il tavolo adatto al proprio ambiente 3D.





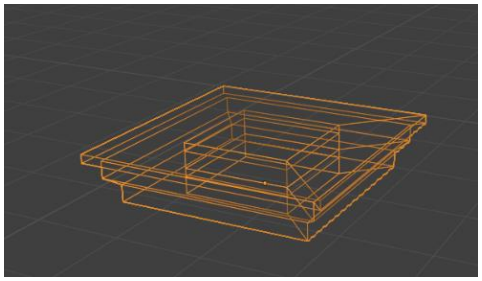


Figura 8.98

Infine l'ultimo tavolo è il tavolo ovale dedicato alla zone dei divanetti curvi, in quanto un divanetto squadrato sarebbe stato scomodo da raggiungere dai clienti da seduti in tali divanetti. Anch'esso ha la stessa divisione in quattro parti e anche in questo caso ogni parte può essere modificata in altezza, larghezza, raggio e risoluzione.

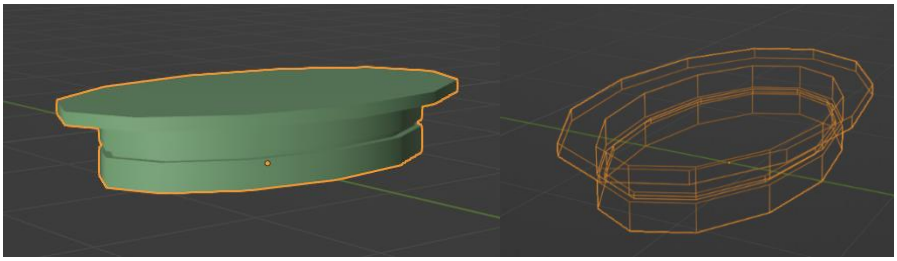


Figura 8.99

Anche in questo caso si è ripensato il concetto di tavolino dopo la revisione, per rendere tutto coerente con gli altri asset. Per il primo tavolino presentato ovvero quello centrale si è pensato di farlo diventare un vero e proprio insetto, infatti esso è stato costruito in modo dalle che abbia un top e una parte centrale che fa da tramite tra il top e le gambe. Tale parte centrale e costruita in modo che sia più spessa da una parte e più sottile dall'altra in modo da simulare la testa dell'insetto. Funzionalmente parlando questo permette di alleggerire la struttura. Infine per rendere davvero il tavolo simile ad un insetto si sono costruite sei gambe di base esagonali che simulano le zampe dell'insetto più lunghe davanti e corte dietro.

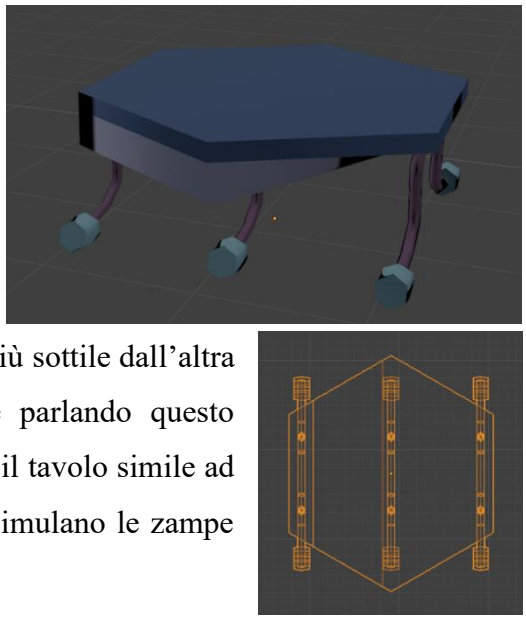


Figura 8.100: Risultato finale

Il secondo tipo di tavolo, visto che l'idea era di posizionarlo in fondo alla sala, si voleva caratterizzarlo in modo diverso. Quindi si è pensato di usare il numero otto come caratterizzazione dato anche che al di sopra dei tavoli ci sono le strutture luci ottagonali e che li affianco è presente la struttura dell'ologramma anch'essa a base ottagonale. Da questa idea poi si è aggiunta quella di associare al tavolo ad un insetto, come il ragno in figura, quindi due ottagoni uniti che descrivono il corpo e otto zampe, poste centralmente per questioni di baricentro.

Figura 8.101 [25]

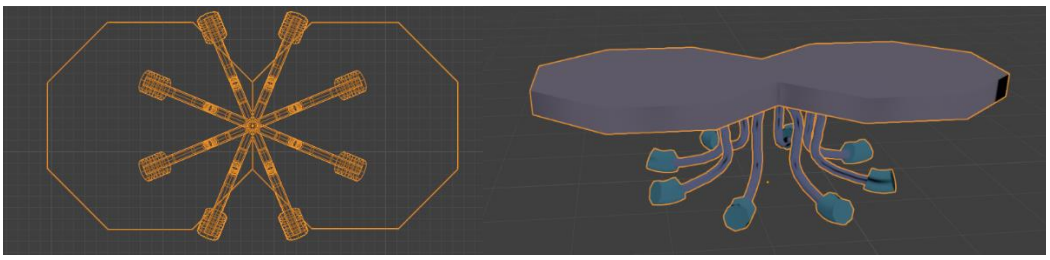


Figura 8.102: Risultato finale

Per il grosso tavolo da usare senza poltrone si è voluti avere un rimando al mondo degli *araneidi:* ovvero la ragnatela, presentata qua stilizzata. Quindi invece che del buco in mezzo al tavolo si è costruito un tavolo che assomiglia a una ragnatela. A livello funzionale i blocchi metallici grigi servono a sostenere le otto regioni a spicchio.

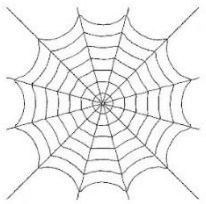


Figura 8.103 [26]

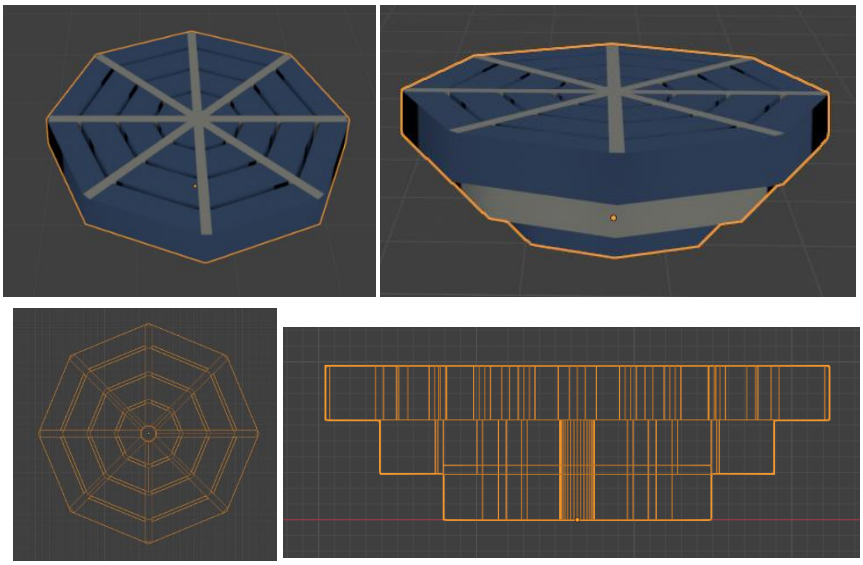


Figura 8.104: Risultato finale

Infine per i tavoli ovali si è pensato di dare la forma di guscio dell'insetto della prima foto, ma anche la forma del secondo insetto. Ovviamente al posto delle gambe, il tavolo ha sei zampe a ricordare la somiglianza con il mondo dei piccoli animali.

Figura 8.105: Insetti usati come reference [27] [13]

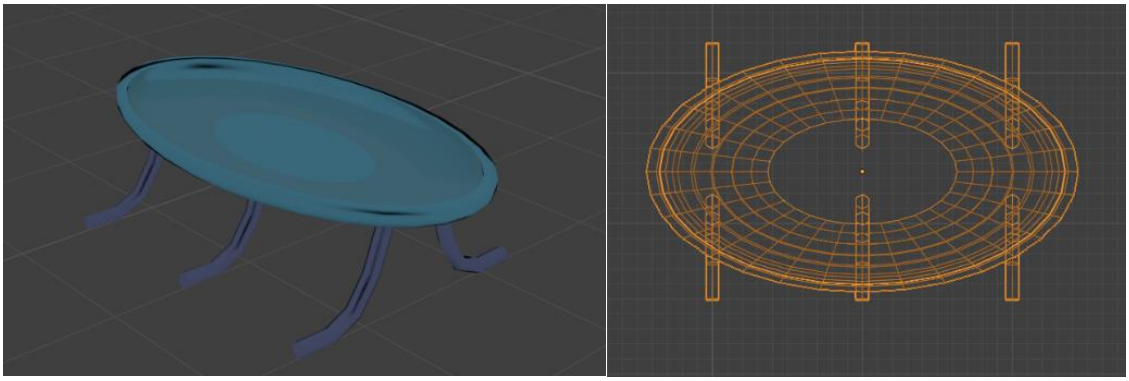


Figura 8.106: Risultato finale

In ogni caso tutti i tavoli creati possono essere modificati in risoluzione, altezza e raggio, in modo da permettere il riutilizzo degli stessi e anche per poter gestire al meglio gli spazi nel caso fosse necessario durante l'animazione dei personaggi.

8.4.14 Insegne luminose:

Le insegne luminose create possono essere suddivise in due gruppi: le scritte e i disegni. Per quanto riguarda le prime all'inizio si era pensato di creare uno script che permettesse facilmente di creare diverse scritte dati alcuni parametri di base, come il font e dimensione carattere, per poi gestire la spaziatura tra caratteri, parole e linee. Tale script viene mostrato di seguito, mentre il risultato della finestra in Panel è mostrato a lato.

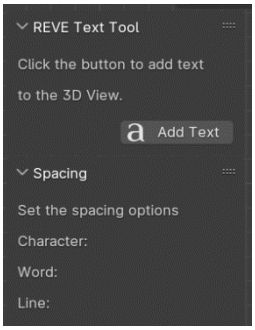


Figura 8.107

```python
import bpy

class OBJECT_TextTool(bpy.types.Panel):
    bl_label = "REVE Text Tool"
    bl_idname = "OBJECT_texttool"
    bl_space_type = 'VIEW_3D'
    bl_region_type = 'UI'
    bl_category = "REVERIE"


    def draw(self, context):
        layout = self.layout
        row = layout.row()
        row.label(text= "Click the button to add text")
        row = layout.row()
        row.label(text= "to the 3D View.")
        row = layout.row()
        row = layout.split(factor= 0.45)
        row.label(text= "")
        row.operator("wm.textopbasic", text= "Add Text", icon= 'OUTLINER_OB_FONT')


class OBJECT_Spacing(bpy.types.Panel):
    bl_label = "Spacing"
    bl_idname = "OBJECT_spacing"
    bl_space_type = 'VIEW_3D'
    bl_region_type = 'UI'
    bl_category = "REVERIE"
    bl_parentid = "OBJECT_texttool"
    bl_options = {"DEFAULT_CLOSED"}


    def draw(self, context):
        layout = self.layout
        text = context.object.data
        row = layout.row()
        row.label(text= "Set the spacing options")

        row = layout.split(factor= 0.45)
        row.label(text= "Character:")
        row.prop(text, "space_character", text= "")

        row = layout.split(factor= 0.45)
        row.label(text= "Word:")
        row.prop(text, "space_word", text= "")

        row = layout.split(factor= 0.45)
        row.label(text= "Line:")
        row.prop(text, "space_line", text= "")
```

```python
class WM_textOpBasic(bpy.types.Operator):
    """Open the Text Tool Dialog Box"""
    bl_idname = "wm.textopbasic"
    bl_label = "                          Text Tool Operator"
    text : bpy.props.StringProperty(name="Enter Text", default="")
    scale : bpy.props.FloatProperty(name= "Scale", default= 1)
    rotation : bpy.props.BoolProperty(name= "Z up", default= False)
    center : bpy.props.BoolProperty(name= "Center Origin", default= False)
    extrude : bpy.props.BoolProperty(name= "Extrude", default= False)
    extrude_amount : bpy.props.FloatProperty(name= "Extrude Amount", default= 0.06)

    def invoke(self, context, event):
        wm = context.window_manager
        return wm.invoke_props_dialog(self)


    def execute(self, context):
        e = self.extrude
        ea = self.extrude_amount
        c = self.center
        r = self.rotation
        s = self.scale
        t = self.text

        bpy.ops.object.text_add(enter_editmode=True)
        bpy.ops.font.delete(type='PREVIOUS_WORD')

        # load font
        data_font = bpy.data.fonts.load('C:\\Users\\aloro\\Polito\\Tesi\\Reverie2023\\Font\\Revery-Regular.otf')

        bpy.ops.font.text_insert(text= t)

        # get text object
        text_object = bpy.context.selected_objects[0]
        # change font
        text_object.data.font = data_font

        bpy.ops.object.editmode_toggle()
        bpy.context.object.data.size = s


        if r == True:
            bpy.context.object.rotation_euler[0] = 1.5708

        if e == True:
            bpy.context.object.data.extrude = ea

        if c == True:
            bpy.context.object.data.align_x = 'CENTER'
            bpy.context.object.data.align_y = 'CENTER'

        return {'FINISHED'}
```

```python
def register():
    bpy.utils.register_class(OBJECT_TextTool)
    bpy.utils.register_class(OBJECT_Spacing)
    bpy.utils.register_class(WM_textOpBasic)


def unregister():
    bpy.utils.unregister_class(OBJECT_TextTool)
    bpy.utils.unregister_class(OBJECT_Spacing)
    bpy.utils.unregister_class(WM_textOpBasic)


if __name__ == "__main__":
    register()
```

Qua sotto viene mostrato il risultato con la scritta BAR.

Figura 8.108

Figura 8.109: Script insegne

Si sono poi studiati i nodi String dei Geometry Nodes per avere un secondo parere e per fare un confronto. Si è notato subito come dei Geometry Nodes siano molto più facili da usare, inoltre rispetto al piccolo script scritto, erano presenti più funzioni. Quindi già il costo tempo-possibilità di operazioni è a vantaggio dei Geometry Nodes. Oltre tutto, tutti i nodi presenti nella suite possono essere usati nei giusti domini per creare l'asset voluto. Dopo questa analisi si è quindi deciso di continuare con i Geometry Nodes perché più convenienti.

Per quanto riguarda il font, esso era già stato creato negli anni precedenti con la produzione del trailer, a destra viene mostrata l'immagine con tutti i caratteri in modo da capire le parole utilizzate negli asset creati.

Gli asset creati in questo primo caso sono strutturati in due parti: la prima è la scritta, la seconda invece è l'outline, ovvero il contorno che racchiude la scritta, come nelle immagini seguenti.

Figura 8.110: Alfabeto Reverie Dawnfall

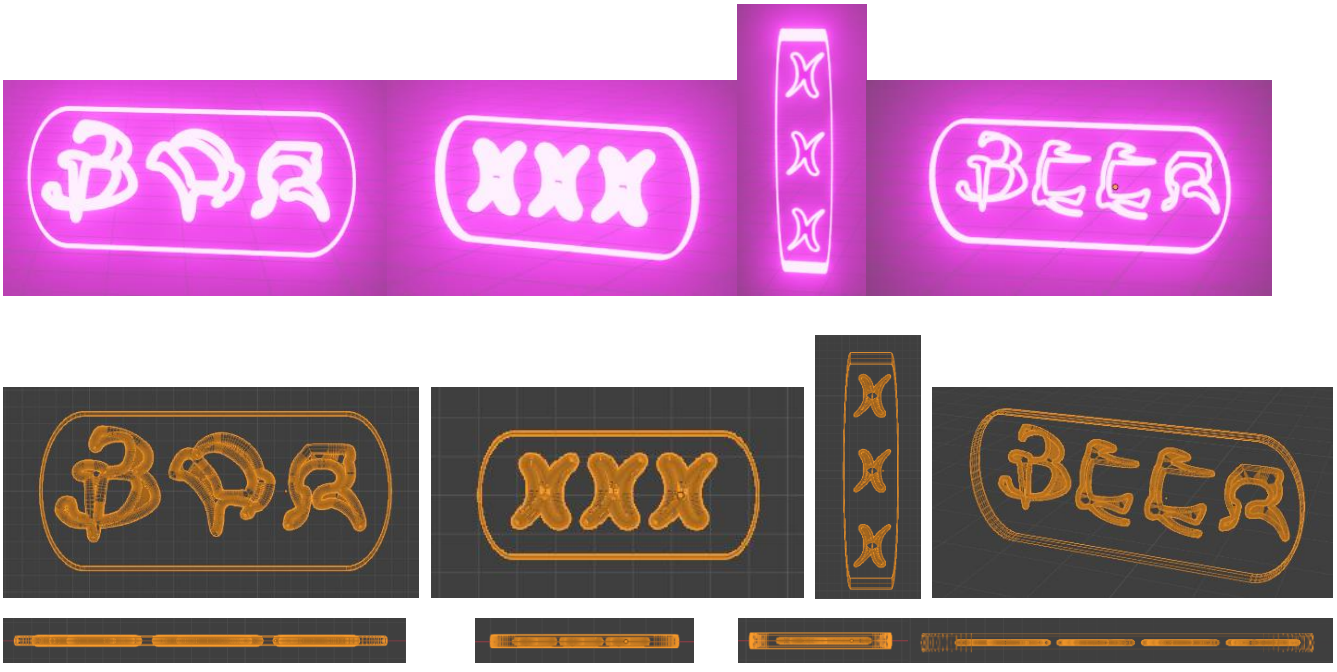


Figura 8.111: Risultato finale

Per quanto riguarda la seconda tipologia di insegne, esse prevedono l'uso di un immagine in PNG, con disegno di colore nero, come negli esempi successivi.

Queste insegne sono formate da piccole celle cubiche di led che, con una risoluzione apprezzabile, crea l'immagine scelta. Più la risoluzione sarà alta, meno si noterà la distinzione tra i vari piccoli led.

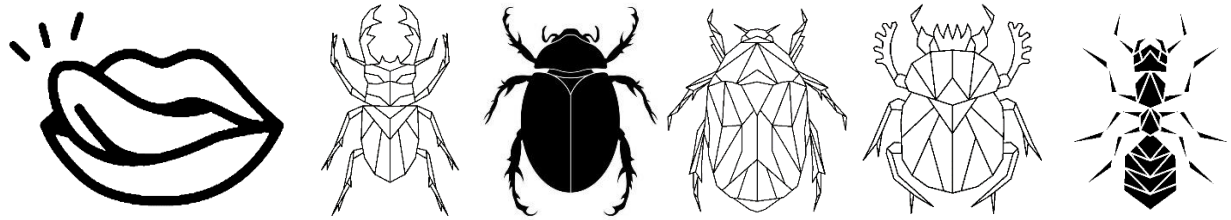


Figura 8.112: immagini usate come reference [28] [29] [30] [31] [32] [33] [34] [35]

Questo albero dei nodi è stato creato in modo tale che si possa, con un unico albero, avere più insegne diverse tra di loro, variandone la dimensione del singolo led, la risoluzione, la scala e la posizione all'interno della griglia, in modo da poter selezionare anche solo una parte dell'immagine. È data anche la possibilità di cambiare immagine.

Di seguito si mostrano alcune degli asset creati con questo metodo.

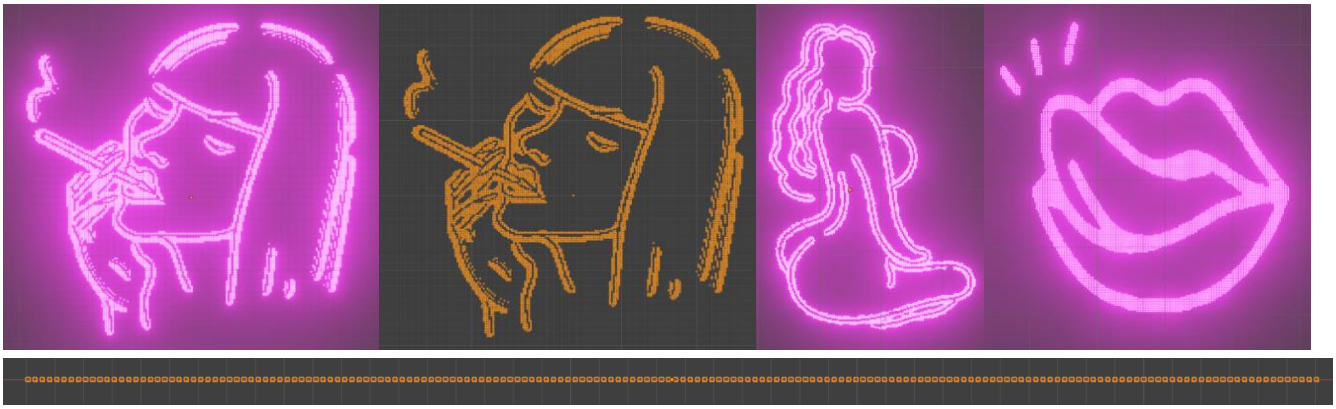


Figura 8.113: Risultato finale [28] [36]

Figura 8.114: Risultato finale [31] [34] [35]

8.4.15 Accessori bar:

Per quanto riguarda gli accessori da bar, essi sono stati tra gli ultimi asset a essere creati, quindi si aveva già la mentalità di creare qualcosa di caratteristico del mondo di Reverie.

Si è incominciato con gli spillatori di birra. Il primo creato prende spunto da un asset trovato su Sketchfab[3], immagine a destra, dalla quale però si sono leggermente modificato alcuni aspetti. Per esempio tutta la struttura di metallo prevede una base esagonale. Al posto della parte terminale del tubo sono stati posti dei piccoli tubicini, i quali servono a portare le diverse tipologie di bevande al proprio settore, che vogliono assomigliare a zampe di insetto. Altre modifiche sono le targhette che sono rese esagonali e nello spillatore vero e proprio dove si hanno diversi diramazioni che si uniscono come se fosse una morsa delle zampe di un ragno.

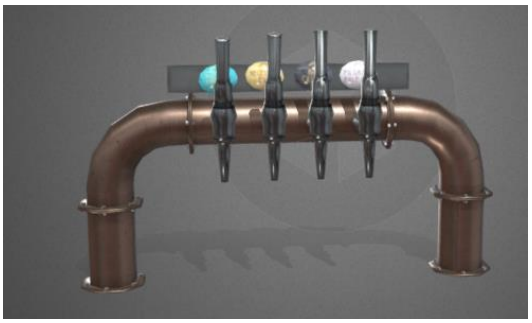
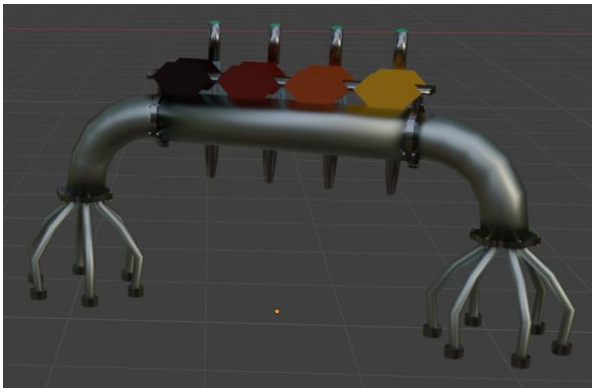
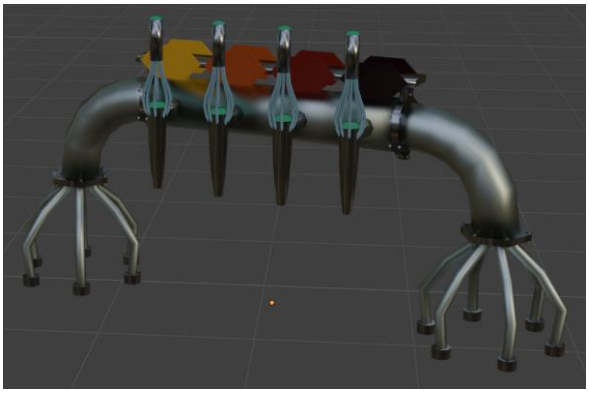


Figura 8.115: Spillatore usato come reference [37]





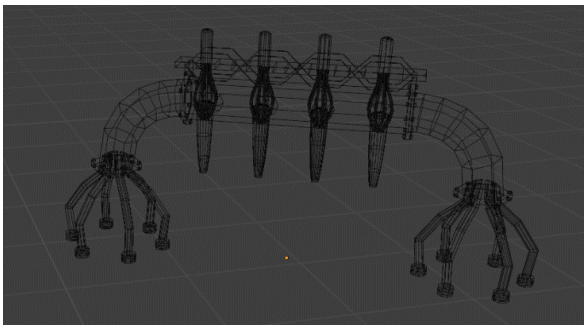
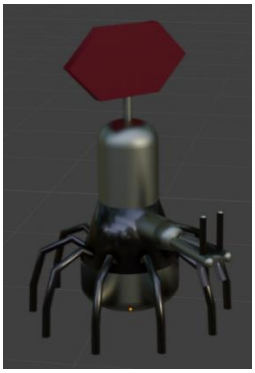


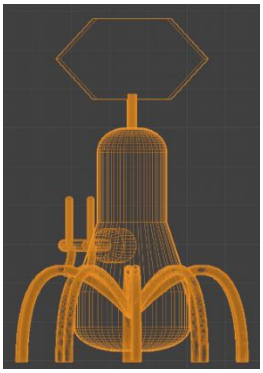
Figura 8.116: Risultato finale

[3] Sketchfab è un sito Web di piattaforma di modellazione 3D per pubblicare, condividere, scoprire, acquistare e vendere contenuti 3D, VR e AR.

Nel secondo spillatore invece si è fatto riferimento all'immagine generata con l'intelligenza artificiale qua a destra. Queste zampe servono sia al sostegno dello spillatore in quanto la struttura è rialzata, sia per prendere la bevanda e portarla dentro alla "pancia", nella quale essa viene tenuta alla giusta temperatura.

Figura 8.117 [17]





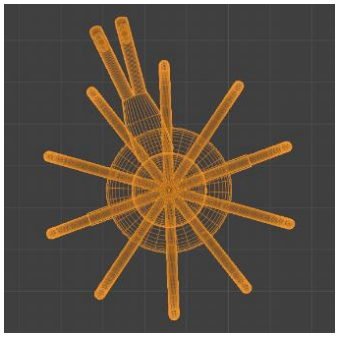


Figura 8.118: Risultato finale

Altri oggetti creati sono il misurino, che semplicemente riprende la base geometria esagonale, gli shaker che nel primo caso, cioè quello doppio anche qua semplicemente ha base esagonale, mentre quello a chiusura prevede delle ramificazioni uscenti che assomigliano a zampe di insetti e che servono per aumentare il grip durante l'apertura e chiusura dell'oggetto.

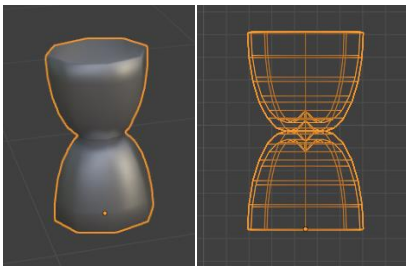


Figura 8.119: Risultato finale

Figura 8.120: Risultato finale

Ultimo accessorio è il tappetino che permette di asciugare il bicchiere prima di essere consegnato al cliente, il quale ha semplicemente una texture a base esagonale per riprendere tutto l'ambiente circostante.

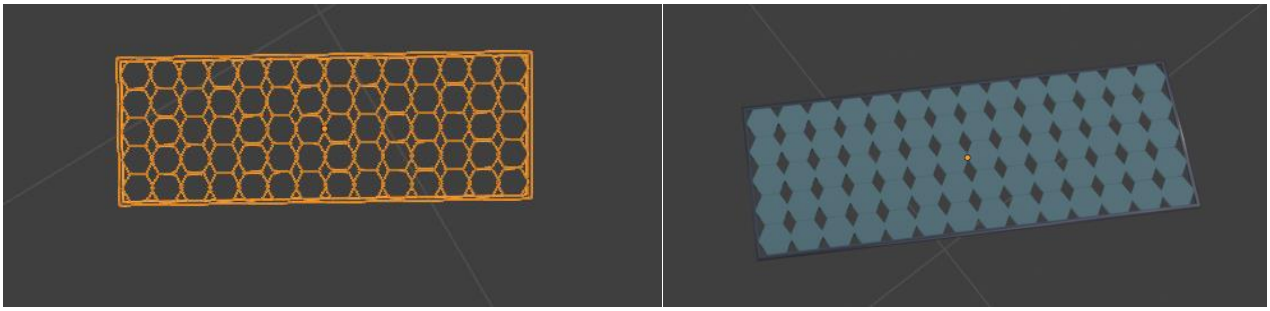


Figura 8.121: Risultato finale

8.4.16 Props:

Per quanto riguarda gli oggetti di scena invece, sono stati individuati tre oggetti.

"Ai tavolini alcuni clienti bevono, altri hanno ragazze in braccio, in primo piano vediamo un tavolo a forma di piccola arena, con dentro due insetti che lottano e 3 persone che li osservano incitando concitati, tenendo in mano delle ***carte-chip****."*

Queste carte chip sono state create basandosi sulle geometrie basilari della serie: la forma esagonale. Questo in quanto è forma comoda da tenere in mano. Esse sono costituite da un doppio contorno esagonale dove i lati interni hanno un piccolo riflesso luminoso, mentre la parte centrale dell'oggetto è rosa in quanto andrà poi sostituita con uno schermo. Infatti si è pensato che queste tessere fossero dei piccoli display in stile cornice digitale per le foto.

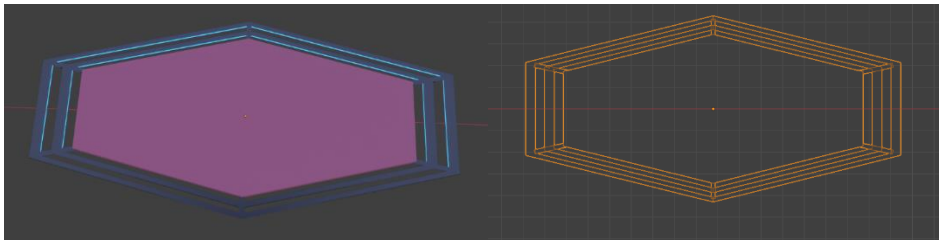


Figura 8.122: Risultato finale

"Poggia un ***apparecchietto*** *vicino alla serratura della porta."*

Per questo tipo di props si è pensato ad un piccolo oggetto con ha delle zampe di metallo che si chiudono tipo ragno, in modo che quando venga appoggiato alla serratura si aggrappi e non possa staccarsi troppo facilmente. Esso presenta internamente uno schermo che è protetto da un bordo di acciaio e avvitato con bulloni che ne conferiscono solidità a tutto l'oggetto.

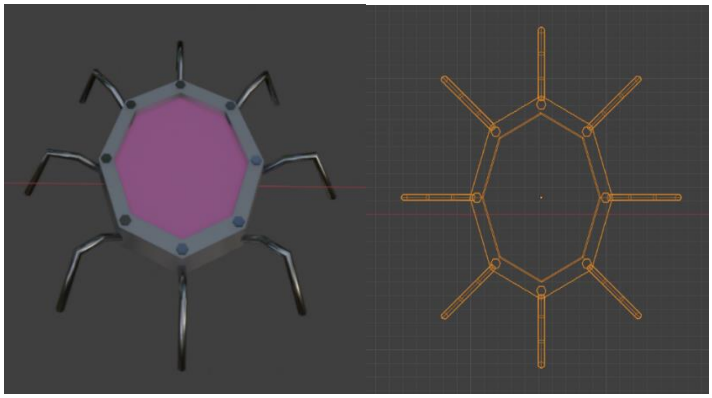


Figura 8.123: Risultato finale

*"Al bancone del bar, Jameela con un lecca lecca in bocca batte rapidamente i tasti di un **piccolo congegno dall'aspetto raffazzonato** e poi dice "done"."*

Infine a proposito del piccolo congegno dall'aspetto rafforzato si è voluto riprendere l'oggetto precedente in modo da farli sembrare oggetti complementari. Esso infatti ha la struttura di base uguale, ma è leggermente più grande; inoltre ha uno schermo centrale protetto da una cornice d'acciaio e bulloni metallici che ne conferiscono solidità. Si differenzia dall'oggetto precedente in quanto le "zampe" di metallo non sono rivolte verso il basso per aggrapparsi alla superficie, ma bensì verso l'alto come antenne di ricezione e invio del segnale.

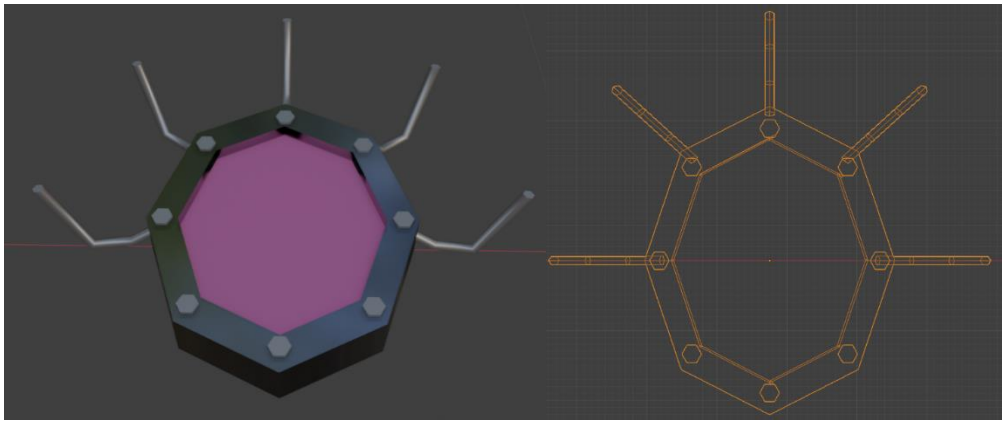


Figura 8.124: Risultato finale

8.4.17 Simulazione:

I Geometry Nodes mettono a disposizione un gruppo di nodi usato per creare simulazioni, tali nodi sono stati provati con un duplice obiettivo: creare una simulazione utile per l'animazione della serie, ma anche per analizzare le differenze con gli strumenti di simulazione tradizionale di Blender.

Per cui si è deciso che come prima simulazione potesse fare al caso nostro quella relativa ad un liquido, in modo che se durante l'animazione di un personaggio, uno di essi ha in mano un bicchiere, tale liquido possa muoversi e non rimanere innaturalmente fermo. Come reference si è usato il tutorial del canale YouTube Erindale [38] mostrato a lato.

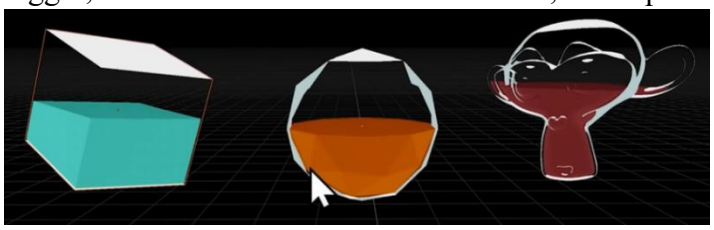


Figura 8.125: Risultato del tutorial [38]

Questa simulazione si può dividere essenzialmente in due parti: la prima riguarda l'albero dei nodi dei Geometry Nodes. In esso si usano principalmente nodi matematici, un nodo di *Sample Index* che serve per gestire la geometria all'interno della zona di simulazione e alcuni nodi di *Store Named Attribute,* i quali permettono di salvare un attributo custom. In questo albero di nodi si è data la possibilità di cambiare il colore del liquido e la sua altezza all'interno del contenitore.

La seconda parte della simulazione invece riguarda l'albero dei nodi dello shader, il quale viene suddiviso in due ulteriori parti. La prima gestisce il contenitore, bottiglia o bicchiere che sia, mentre la seconda parte riguarda il liquido. In quest'ultima parte vi è il collegamento con i Geometry Nodes, grazie al nodo Attribute che permette di selezionare l'attributo corrispondente a quello dei Geometry Nodes.

L'animazione è poi fatta inserendo il driver *#frame,* all'interno di un nodo *Value* nella sezione *Wobble* dello shader, in modo che ad ogni frame se l'oggetto viene mosso, traslato o rotato, il liquido all'interno si muove di conseguenza.

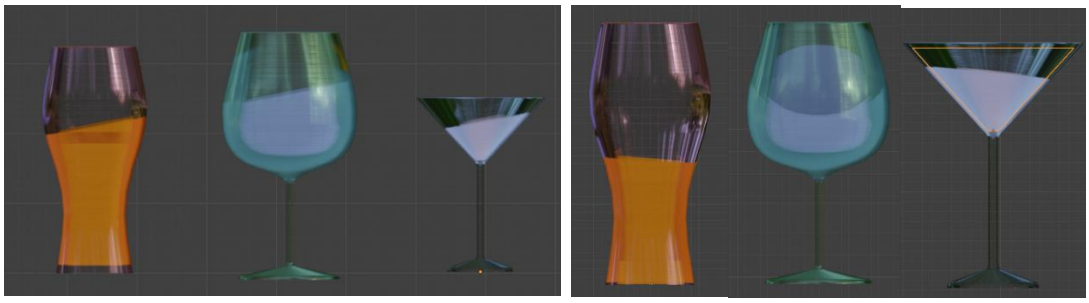


Figura 8.126: Risultato finale

8.4.18 Alberi di nodi:

Qui di seguito verranno mostrati alcune immagini riguardati gli alberi dei nodi degli oggetti appena presentati.

Figura 8.127: Bancone del Bar

Figura 8.128: Bottiglie

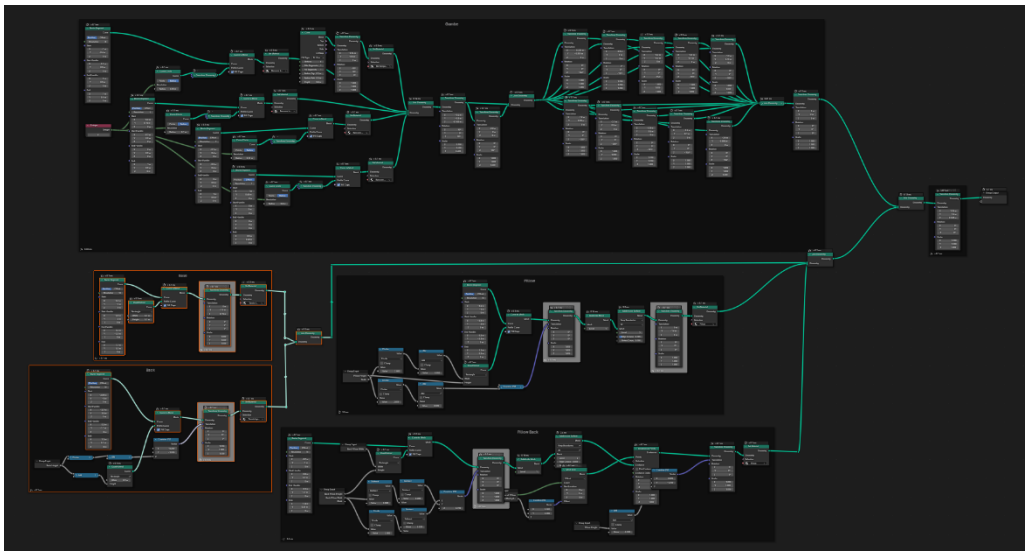


Figura 8.129: Divanetti curvi

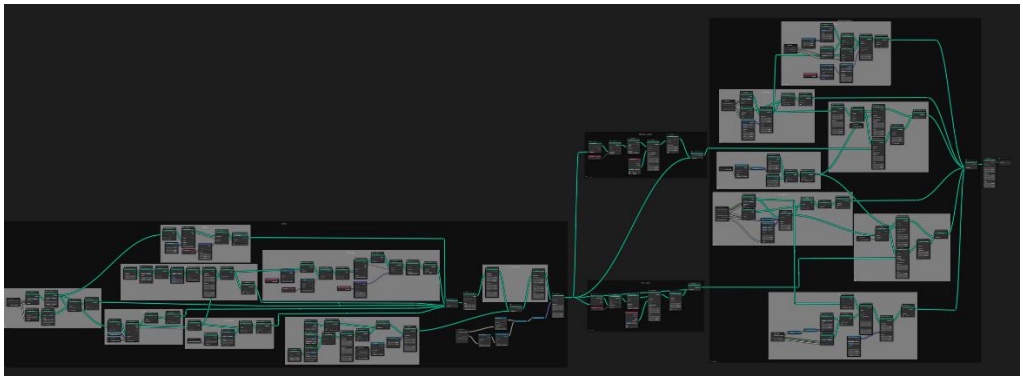


Figura 8.130: Impianto Luci esagonali

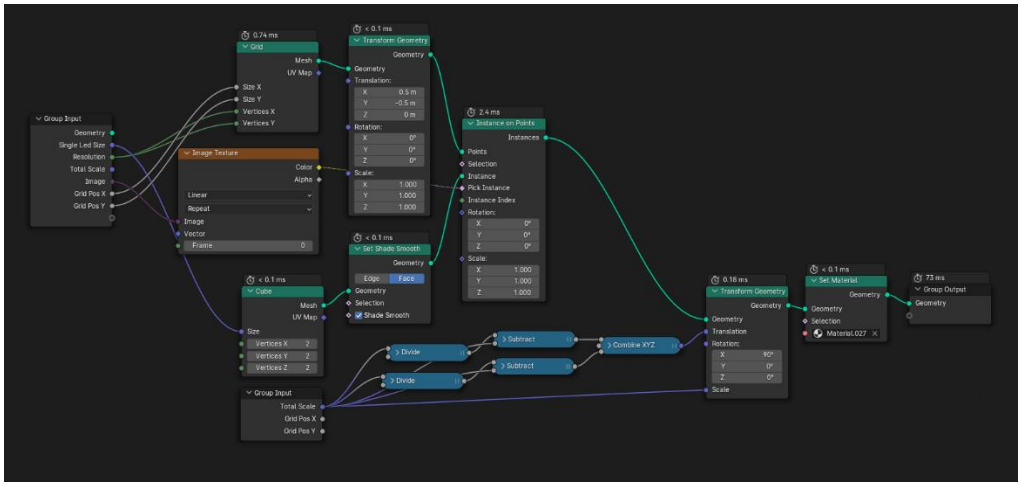


Figura 8.131: Insegne luminose immagini

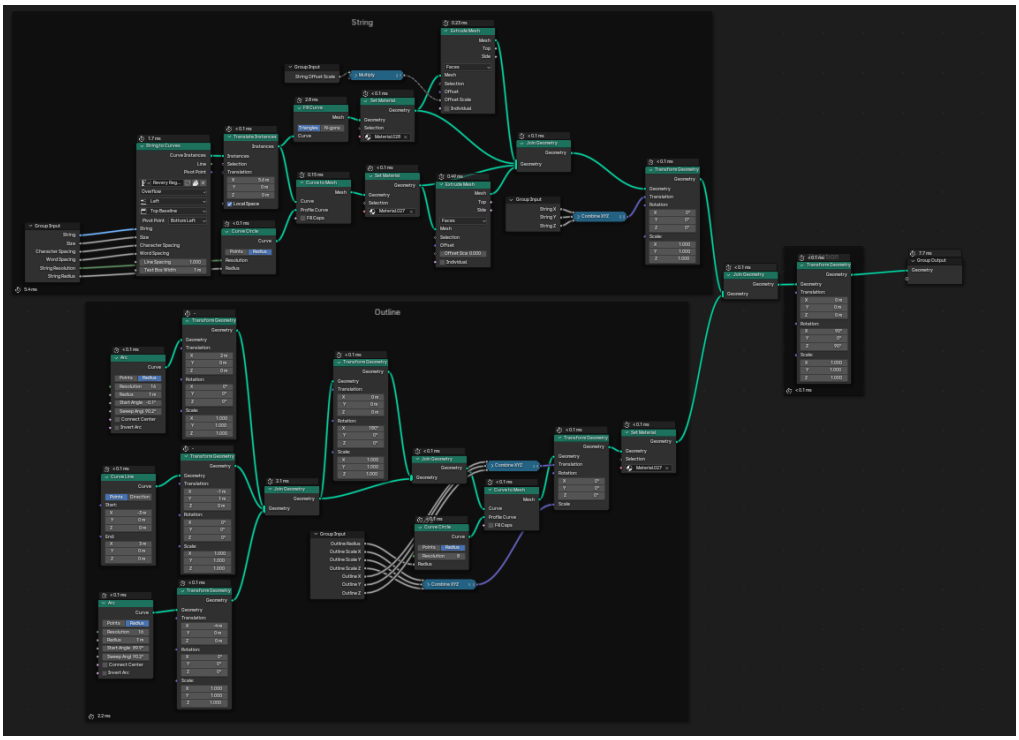


Figura 8.132: Insegne luminose scritte

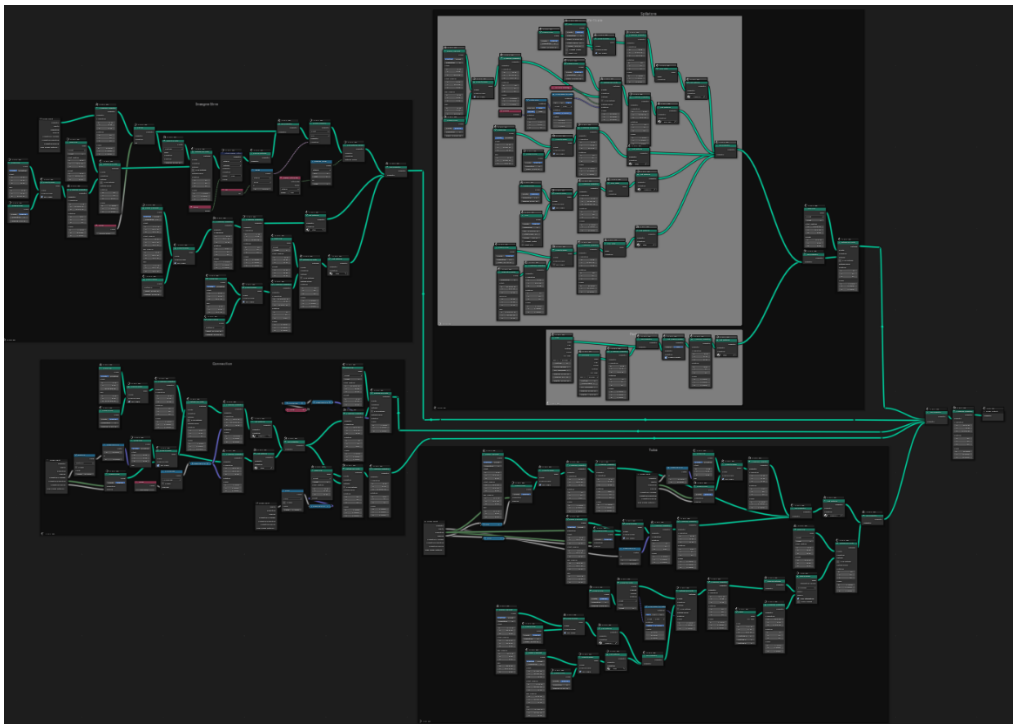


Figura 8.133: Spillatore a tubo con quattro vie

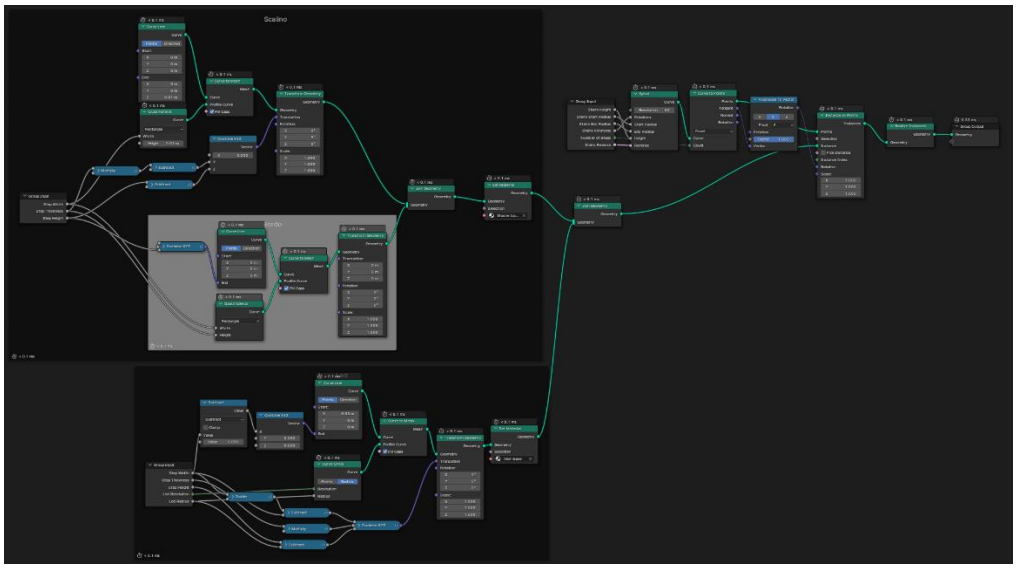


Figura 8.134: Scale

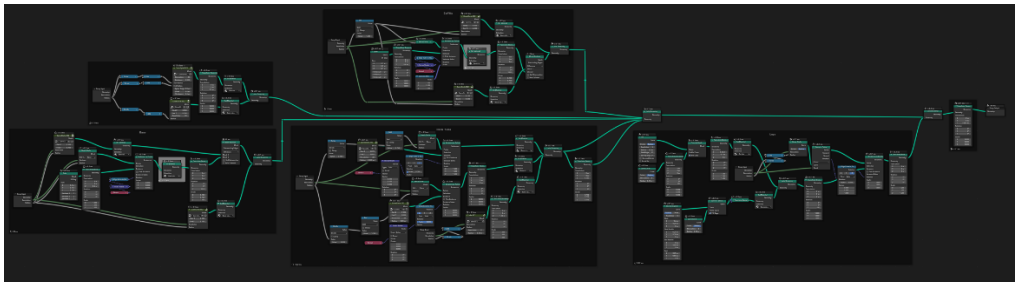


Figura 8.135: Struttura ologramma

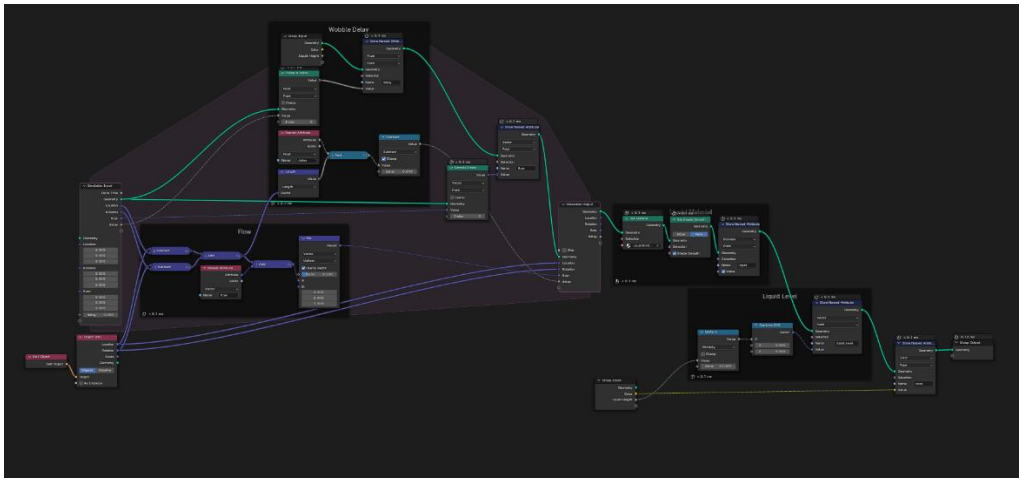


Figura 8.136: Simulazione di liquido

9. Risultati Finali

Figura 9.1

Figura 9.2

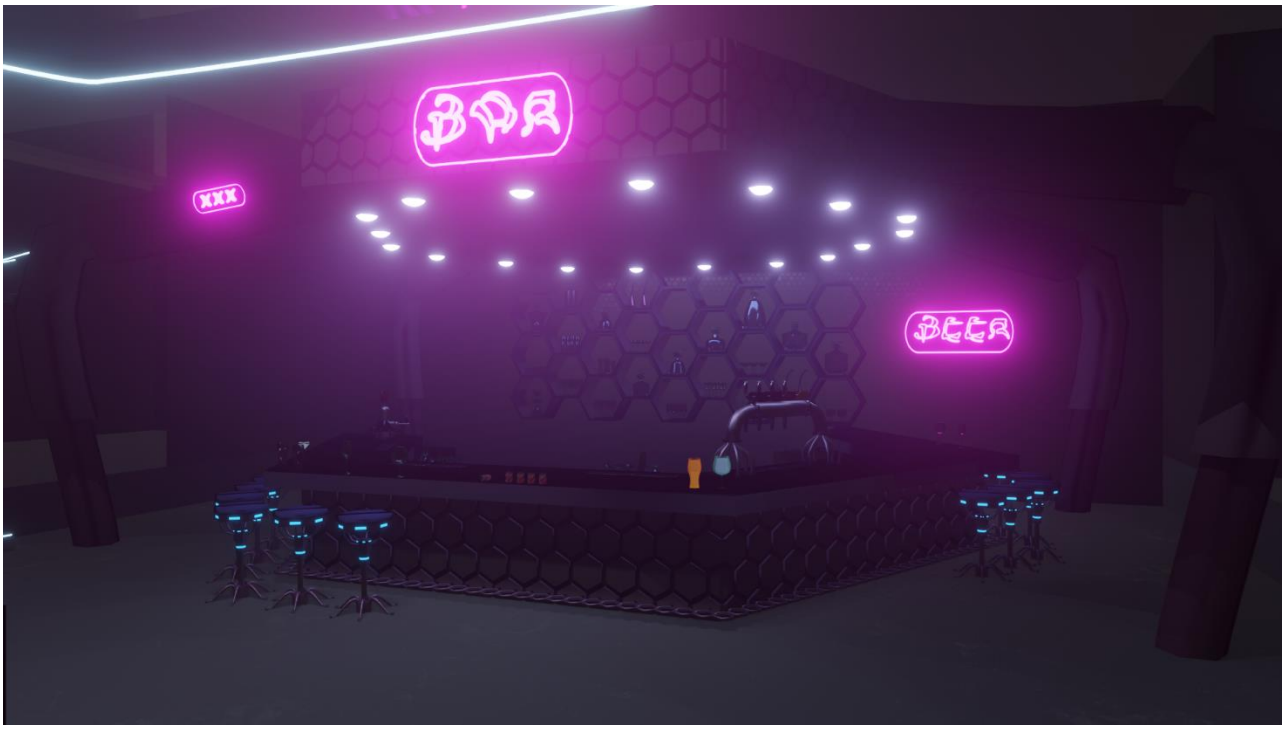


Figura 9.3

Figura 9.4

Figura 9.5

Figura 9.6

10. Conclusioni

Visto nel dettaglio ogni singolo oggetto creato e le immagini al capitolo precedente dei risultati finali ottenuti, si può tirare la somma dei risultati e degli obiettivi.

Lavorare ad un progetto di animazione indie con un alto potenziale come Reverie Dawnfall, ha permesso di ampliare la conoscenza del software Blender e di testare le proprie capacità sul proprio argomento di tesi, ma anche di imparare concetti riguardanti altri settori della produzione e creazione di un lavoro di animazione appartenenti agli argomenti di tesi dei miei colleghi.

L'uso dei Geometry Nodes ha reso possibile il compimento dell'obiettivo posto dall'azienda di sperimentare stili grafici e tecniche quanto più innovative possibili, favorendo anche alcuni processi produttivi come le tempistiche di realizzazione e il costo computazionale.

Per come sono stati strutturati i vari asset, essi offrono la possibilità di essere modificati e gestiti tramite i valori di input presenti nella sezione del modifier Geometry Nodes presente nella scheda Modifiers. In questo modo è possibile avere variazioni dello stesso oggetto per adattarlo alle proprie esigenze in modo molto facile e veloce.

Per quanto riguarda i possibili risvolti futuri, sicuramente il concetto di lavoro procedurale attraverso l'uso di nodi sarà sempre più preponderante rispetto ai suoi competitor, in quando permette un gestione del lavoro e del tempo migliore.
Inoltre potrebbe essere interessante effettuare nuovi studi sui nodi di simulazione, quelli di Repeat Zone e quelli Hair. In aggiunta con le future versioni di Blender i Geometry Nodes verranno aggiornati, aggiungendo nuovi nodi e nuove funzionalità che potranno essere sicuramente interessanti da scoprire e da provare.

11. Bibliografia e Sitografia

[1] «Blender,» [Online]. Available: https://www.blender.org/.

[2] «Storia di Blender,» [Online]. Available: https://docs.blender.org/manual/en/3.4/getting_started/about/history.html.

[3] S. Raimi, «Spider-man 2,» Columbia Pictures, Marvel Enterprises, Laura Ziskin Productions, 2004. [Online]. Available: https://www.disneyplus.com/it-it/movies/spider-man-2/1CNlLz5NUKU6.

[4] A. Rak, I. Cappiello, M. Guarnieri e D. Sansone, «Gatta Cenerentola,» Mad Entertainment, Big Sur, Sky Dancers, Tramp Ltd, O' Groove, Rai Cinema, MiBACT, 2017. [Online]. Available: https://www.madentertainment.it/produzione/gatta-cenerentola/.

[5] «Motori di render,» [Online]. Available: https://docs.blender.org/manual/en/3.4/render/introduction.html.

[6] «Introduzione ai Geometry Nodes,» [Online]. Available: https://docs.blender.org/manual/en/latest/modeling/geometry_nodes/introduction.html.

[7] «Campo,» [Online]. Available: https://docs.blender.org/manual/en/latest/modeling/geometry_nodes/fields.html.

[8] «Attributi,» [Online]. Available: https://docs.blender.org/manual/en/latest/modeling/geometry_nodes/attributes_reference.html.

[9] «Istanza,» [Online]. Available: https://docs.blender.org/manual/en/latest/modeling/geometry_nodes/instances.html.

[10] «Tipi di nodi,» [Online]. Available: https://docs.blender.org/manual/en/latest/modeling/geometry_nodes/index.html#node-types.

[11] «Robin Studio,» [Online]. Available: https://robin.studio/.

[12] CD Projekt, «Cyberpunk 2077,» CD Projekt RED, 10 12 2020. [Online]. Available: https://www.cyberpunk.net/us/it/.

[13] itchydogimages, «Tessaratomidae,» 27 04 2014. [Online]. Available: https://sinobug.aminus3.com/image/2014-04-27.html.

[14] [Online]. Available: https://www.ravennaedintorni.it/societa/2022/06/03/provincia-piu-300-aziende-apicoltura-producono-5-del-miele-italiano/.

[15] «Miriapodi,» [Online]. Available: https://disinfesta.it/insetti/miriapodi/.

[16] «Sphodros rufipes,» [Online]. Available: https://it.wikipedia.org/wiki/File:Sphodros_rufipes_non-crossing_chel.jpg.

[17] Microsoft , «Microsoft Copilot,» 7 2 2023. [Online]. Available: https://www.microsoft.com/it-it/microsoft-copilot.

[18] E. Mostaque, «Stability.ai,» 2019. [Online]. Available: https://stability.ai/.

[19] P. H. Olsen, «Nagusta goedelii,» 31 10 2022. [Online]. Available: https://insects.at/index.php/en/hemiptera/heteroptera/cimicomorpha/reduvioidea/reduviidae/harpactorinae/793-harpactorini/3346-nagusta-goedelii-140629603.

[20] Ø. Sørøy, A. Latner e A. Edesberg, «Sloyd,» 2023. [Online]. Available: https://www.sloyd.ai/?ref=futuretools.io.

[21] «Auchenorrhyncha,» [Online]. Available: https://www.stockio.com/free-photo/auchenorrhyncha-2.

[22] Assembly Point Interior Decoration LLC, «LUXURIOUS NIGHTCLUB DESIGN | Assembly Point Interior Decoration,» 2022. [Online]. Available: https://www.youtube.com/watch?v=0nwE7R_jcl4&list=PLfXUkYp3lrhVc9dYqiMugUpMlJ0h7G_e-&index=7.

[23] P. BLENDER, «Lights Group Control | Procedural Modeling | Geometry Nodes,» 2023. [Online]. Available: https://www.youtube.com/watch?v=wSL0UzWgNO4&list=PLfXUkYp3lrhXOC5BQ7ryNtlbmdmCv2M9a&index=49&t=1711s.

[24] Pixabay, «Vermi spirale,» [Online]. Available: https://pixabay.com/it/photos/vermi-spirale-calcolo-936477/.

[25] «Leptonetoidea,» 11 2010. [Online]. Available: https://the-spiders.blogspot.com/2010/11/leptonetoidea.html.

[26] [Online]. Available: https://www.etsy.com/it/listing/251669087/disegno-di-macchina-di-ragnatela-ragno.

[27] «Belostomatidae,» 2015. [Online]. Available: https://genent.cals.ncsu.edu/insect-identification/order-hemiptera-suborder-heteroptera/family-belostomatidae/.

[28] 19tom79. [Online]. Available: https://www.pngitem.com/middle/wTmR_neon-girl-smoking-neon-girl-stickers-png-transparent/.

[29] CHENXIN. [Online]. Available: https://it.pngtree.com/freepng/neon-optical-effect-sexy-girl-figure-back_6363474.html.

[30] B. Org. [Online]. Available: https://www.vhv.rs/viewpic/ixmoRTb_neon-lips-lip-sexy-neonlights-neoncolor-neoneffect-sexy/.

[31] «Marinika,» [Online]. Available: https://stock.adobe.com/images/geometric-stag-beetle-polygonal-linear-abstract-bug-vector-illustration/228244571.

[32] R. Atzeni. [Online]. Available: https://www.vecteezy.com/vector-art/24078318-black-silhouette-of-a-scarab-beetle-vector-insect-isolated-on-a-white-background.

[33] Marinika. [Online]. Available: https://stock.adobe.com/it/images/geometric-beetle-polygonal-linear-abstract-bug-vector-illustration/227634745.

[34] Marinika. [Online]. Available: https://stock.adobe.com/images/abstract-polygonal-scarab-beetle-geometric-linear-animal-vector-illustration/298850258.

[35] Marinika. [Online]. Available: https://stock.adobe.com/br/images/geometric-ant-polygonal-animal-black-silhouette-vector-illustration/280554924.

[36] CHENXIN. [Online]. Available: https://pngtree.com/freepng/neon-efficiency-sexy-women_6363482.html.

[37] S. R. Ltd, «Draft Tap Tower,» 2020. [Online]. Available: https://sketchfab.com/3d-models/draft-tap-tower-4b97aabc309745029fb3c5f57cbf1f7f.

[38] Erindale, «Fake Liquid - Simulation Nodes in Blender 3.6,» 08 07 2023. [Online]. Available: https://www.youtube.com/watch?v=4Lcwb7g3888&list=PLfXUkYp3lrhXOC5BQ7ryNtlbmdmCv2M9a&index=40.