

REAL2VIRTUAL

Trasporre un oggetto dal mondo reale al mondo virtuale tramite
fotogrammetria



Giulio Zanchetta
Department of Computer Science
VR413626

Davide Benini
Department of Computer Science
VR409482

Giovanni Danieli
Department of Computer Science
VR408623

Università degli Studi di Verona
Relatore: Marco Cristani
Anno Accademico : 2018/2019

Indice

	Pagina
1 Prefazione	2
1.1 Introduzione	2
1.2 Metodo di lavoro / processo di sviluppo	2
1.3 Obiettivi	3
2 Documentazione e primi passi	4
2.1 Installazione e utilizzo di Zephyr	4
2.1.1 Zephyr e ricostruzione tridimensionale	4
3 Ricostruzione e Fotogrammetria	9
3.1 Model Processing	9
3.1.1 Acquisizione delle immagini dell'oggetto da ricostruire	9
3.1.2 Elaborazione del dataset	9
3.1.3 Ricostruzione dell'oggetto acquisito	10
3.1.4 Baking delle normali	11
3.1.5 Baking delle texture	12
3.1.6 Pulizia della texture	12
3.1.7 Finalizzazione dell'oggetto ed esportazione dell'asset	12
4 Creazione del gioco nel game engine Unity	13
4.1 Descrizione del gioco	14
4.1.1 Primo livello del gioco	14
4.1.2 Secondo livello del gioco	16
5 Conclusioni	18
5.1 Bibliografia	18

Capitolo 1

Prefazione

1.1 Introduzione

La decisione di intraprendere questo percorso di trasporre oggetti dal mondo reale in un ambiente virtuale deriva dalla nostra comune passione nell'ambito videoludico.

1.2 Metodo di lavoro / processo di sviluppo

Prima di iniziare il progetto ci siamo documentati su Unity e su tutte le fasi della fotogrammetria. Avendo già delle basi riguardo Unity ci siamo limitati ad approfondire lo scripting e a eseguire migliorie grafiche da applicare al nostro caso. Per le basi di Unity abbiamo fatto riferimento alla guida offerta da [HTML.it](#), mentre per le fasi più avanzate di scripting e fisica del gioco ci siamo basati sulla documentazione e progetti di template realizzati da Unity. Per quanto riguarda la parte di fotogrammetria ci siamo basati sulla guida ufficiale messa a disposizione da Unity intitolata "Unity Photogrammetry Workflow" che ci ha fornito gli strumenti necessari per realizzare al meglio i nostri oggetti. Prima di iniziare il nostro progetto abbiamo identificato le fasi di sviluppo da seguire:

- La prima fase consiste nel testare le funzionalità e potenzialità del software 3DF Zephyr e per questo prima di arrivare a creare i nostri oggetti finali abbiamo effettuato svariate prove con oggetti di varie dimensioni e di varie proprietà.
- Abbiamo creato una repository su GitHub destinata a contenere tutti i file del nostro progetto (inclusi gli script Matlab utilizzati per le ottimizzazioni delle immagini dei datastore acquisiti). Abbiamo suddiviso la repository nelle directory: "Documentazione", "Modelli", "Presentazione", "Script" e "Progetto di Unity". Infine abbiamo scritto un file Markdown di documentazione del progetto. L'intera repository è stata resa pubblica al seguente link:
<https://github.com/zk-g/real-to-virtual>.

- A questo punto eravamo pronti a realizzare l'ambientazione ove inserire il nostro oggetto ricostruito. Per questo abbiamo realizzato una scaletta con tutti gli elementi in ordine da inserire nel progetto.

1.3 Obiettivi

Gli obiettivi del nostro studio erano:

- Imparare ad utilizzare il framework di ricostruzione 3D professionale Zephyr, imparandone i passi fondamentali (incluse operazioni di filtraggio di immagini).
- Imparare ad utilizzare il framework di sviluppo Unity, imparandone i passi fondamentali.
- Ricostruire un oggetto reale, e suo inserimento in un ambiente virtuale, definendo uno scenario che possa essere esplorato.

Capitolo 2

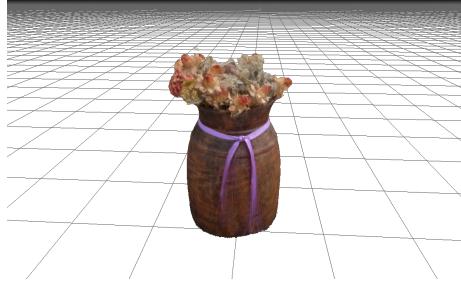
Documentazione e primi passi

2.1 Installazione e utilizzo di Zephyr

Per iniziare abbiamo consultato il materiale on-line del software Zephyr e grazie ai tutorial messi a disposizione nel canale ufficiale di 3Dflow ([3DflowChannel](#)) abbiamo appreso appieno l'utilizzo del software di fotogrammetria avanzata.

2.1.1 Zephyr e ricostruzione tridimensionale

Grazie a Zephyr è possibile ricostruire complessi elementi tridimensionali semplicemente scattando delle foto da varie angolazioni. È possibile poi esportare gli oggetti creati in vari formati il che da molta libertà per future modifiche con il software di modellazione più opportuno. Come primo passo Zephyr analizza le immagini caricate per poi creare una nuvola di punti che coincide con i vertici dell'oggetto da creare, l'utente potrà poi decidere di addensare la nuvola di punti per poi applicargli una mesh e la texture. Per iniziare abbiamo sperimentato l'utilizzo di diversi oggetti con diverse caratteristiche per capire come il software li analizzasse, utilizzando oggetti di diversa altezza, spessore e diametro. Osservando le differenze in termini qualitativi con differenti quantità di immagini abbiamo capito quale fosse il numero minimo necessario per riscontrare un risultato di buona qualità per oggetti di piccole dimensioni (quali tazze o libri), individuando che per un buon risultato sono necessarie almeno 30 fotografie.



(a) Oggetto reale da ricostruire. (b) Oggetto ricostruito da Zephyr.

Figura 2.1: Un vaso di fiori (40 foto); nonostante il basso livello di contrasto fra i fiori e lo sfondo zephyr ha ricreato l'oggetto egregiamente

Successivamente abbiamo utilizzato i tool per il mascheramento e la modifica degli oggetti messi a disposizione dal software Zephyr per capire come effettuare delle semplici modifiche senza dover ricorrere ad altri software. Questi strumenti servono principalmente per rimuovere eventuali sfondi o punti d'appoggio dall'oggetto creato.



(a) Oggetto reale da ricostruire. (b) Oggetto ricostruito da Zephyr.

Figura 2.2: Un coltello apribuste (50 foto), malgrado il lieve spessore la ricostruzione ha mantenuto fede ai dettagli

Ci siamo poi spinti oltre mettendo alla prova le capacità di Zephyr di ricostruire elementi tridimensionali partendo da immagini con rumore gaussiano e successivamente elaborate con Matlab per ridurre il suddetto rumore.

Per iniziare abbiamo scattato le foto in una stanza buia con ISO pari a 25.600 per ricreare un rumore gaussiano senza però ottenere il risultato sperato (probabilmente la macchina fotografica, Fujifilm X-T1, ha applicato in modo automatico un filtro antirumore) così abbiamo creato un filtro passa basso ideale con matlab:

$$H(u, v) = \begin{cases} 1, & se D(u, v) \leq D_0 \\ 0, & se D(u, v) > D_0 \end{cases} \quad (2.1)$$

$$D(u, v) = \sqrt{(u - M/2)^2 + (v - N/2)^2} \quad (2.2)$$

ottenendo il seguente risultato



(a) Immagine originale.

(b) Dopo aver applicato il rumore.

Togliendo il rumore generato grazie a matlab e deep learning si ottiene un risultato molto vicino all'originale. Per fare ciò è stato necessario convertire l'immagine da RGB a scala di grigi.

```
net = denoisingNetwork('DnCNN');
denoisedI = denoiseImage(noisyI, net);
figure
imshow(denoisedI)
title('Denoised Image')
imwrite();
```

È stato scelto questo procedimento perché è quello che ha dato i risultati migliori.



(a) Immagine originale.

(b) Dopo aver applicato il rumore.



(c) Dopo aver tolto il rumore con MatLab

Abbiamo poi provato a sistemare un’immagine fuori fuoco grazie all’algoritmo di convoluzione per rimuovere il blur dall’immagine:

$$f_1 * f_2(t) = \int_{-\infty}^{\infty} f_1(\tau) f_2(t - \tau) d\tau \quad (2.3)$$



(a) Immagine sfocata.

(b) Dopo l’algoritmo.

Infine abbiamo utilizzato le immagini restaurate per creare i modelli con zephyr ricavando dei risultati soddisfacenti. Nel caso della rimozione del rumore, trascurando la perdita del colore, l’oggetto ottenuto ha un qualità quasi pari all’originale. Nel

caso del blur la differenza sul pdf è meno percettibile ma si può comunque notare una riduzione della sfocatura.

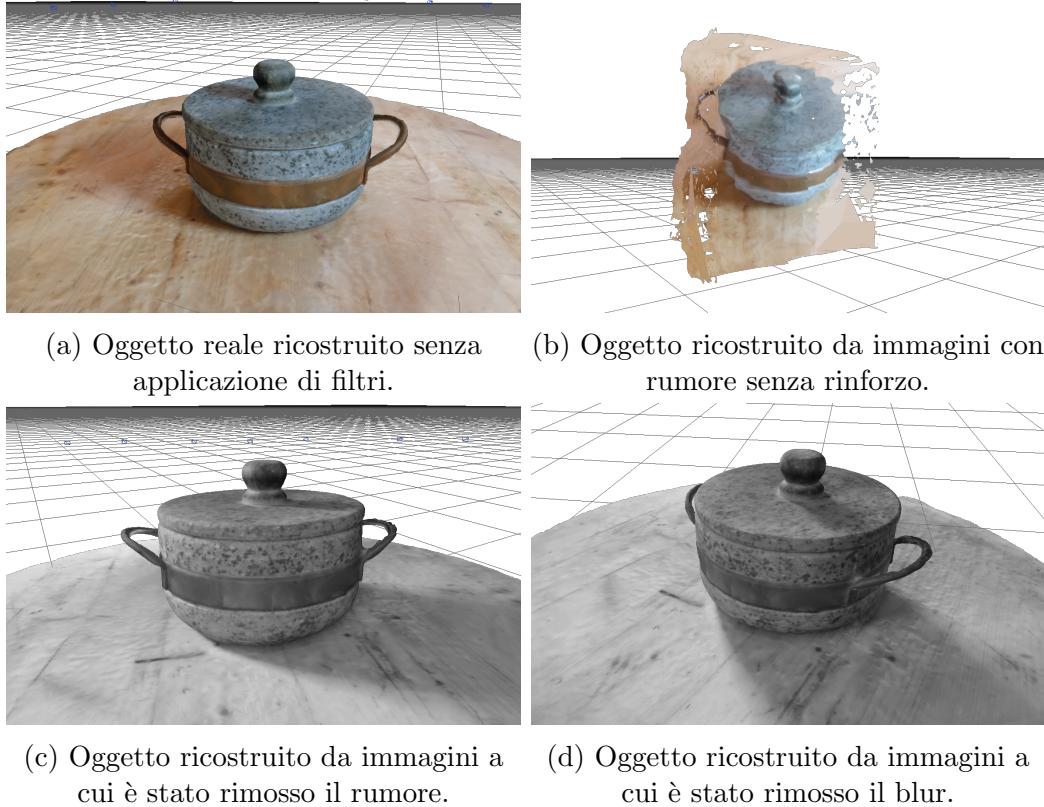


Figura 2.6: Pentola (64 immagini), nel caso della ricostruzione con le immagini senza rinforzo zephyr non è riuscito ad elaborare tutte le immagine o parzialmente ed è stato ottenuto un risultato scadente, recuperato poi con il rinforzo delle immagini su matlab

Capitolo 3

Ricostruzione e Fotogrammetria

3.1 Model Processing

Come oggetto da ricostruire è stato scelto un oggetto reale di interesse storico ovvero una statua del Giardino Salvi di Vicenza e con il software Zephyr ne abbiamo riscosstruito il modello 3D inserendolo successivamente all'interno del mondo di gioco. Il processo per ricreare un oggetto adatto al funzionamento all'interno di un motore grafico è composto da vari processi.

3.1.1 Acquisizione delle immagini dell'oggetto da ricostruire

Per acquisire il dataset ci siamo recati di pomeriggio in una giornata nuvolosa per evitare una eccessiva presenza di luce. Sono state utilizzate due fotocamere: una Fujifilm X-T1 e una Canon EOS 70D. Sono state scattate circa 700 foto da varie angolazioni per essere sicuri di acquisire la statua nella sua interezza.

3.1.2 Elaborazione del dataset

Bisogna assicurarsi che ogni foto all'interno del set abbia i requisiti per ricostruire un modello di buona qualità; bisogna escludere quindi le immagini con esposizione troppo bassa o troppo alta, immagini mosse o sgranate perchè anche una piccola percentuale di foto di bassa qualità possono compromettere il risultato finale.

Se le immagini sono state catturate in formato RAW è necessario convertirle in un formato che sia adatto al programma di ricostruzione, per mantenere la miglior qualità è consigliato convertirle nel formato TIFF, per questo si può utilizzare "Camera RAW" di Adobe oppure "DC RAW", un programma open-source creato da Dave Coffin.

Successivamente bisogna bilanciare i bianchi delle immagini acquisite mediante Adobe Photoshop.

Bisogna quindi convertire le immagini da 32 bit a 8 bit, questo per evitare problemi con la correzione di gamma all'interno del software di ricostruzione; in ogni caso la

maggior parte dei software di ricostruzione converte in automatico le immagini a 8 bit, per cui non vi è perdita di informazioni.

Come ultimo passaggio abbiamo mascherato il nostro set di foto utilizzando 3DF Masquerade.

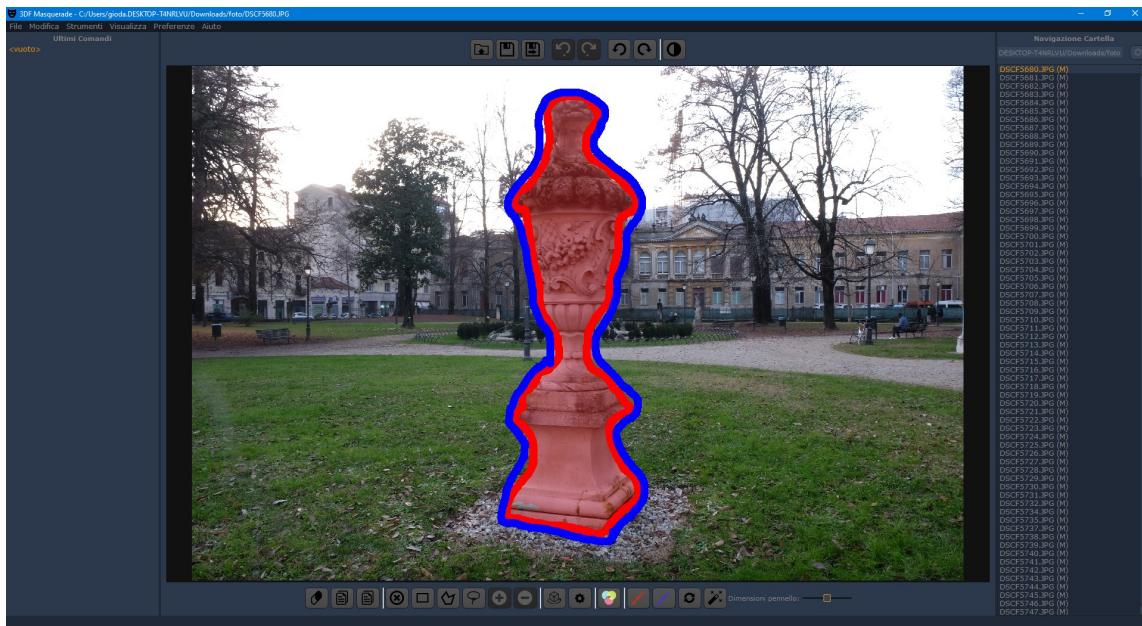


Figura 3.1: Una foto mascherata del nostro dataset all'interno del software 3DF Masquerade

3.1.3 Ricostruzione dell'oggetto acquisito

Per costruire l'oggetto principale della scena abbiamo utilizzato il software "Zephyr" di 3DFlow. Aperto il programma abbiamo importato il nostro set di immagini (composto da 350 elementi), a questo punto abbiamo calcolato la nuvola di punti sparsa, la nuvola di punti densa, le mesh ad alta e a bassa risoluzione e infine la texture.

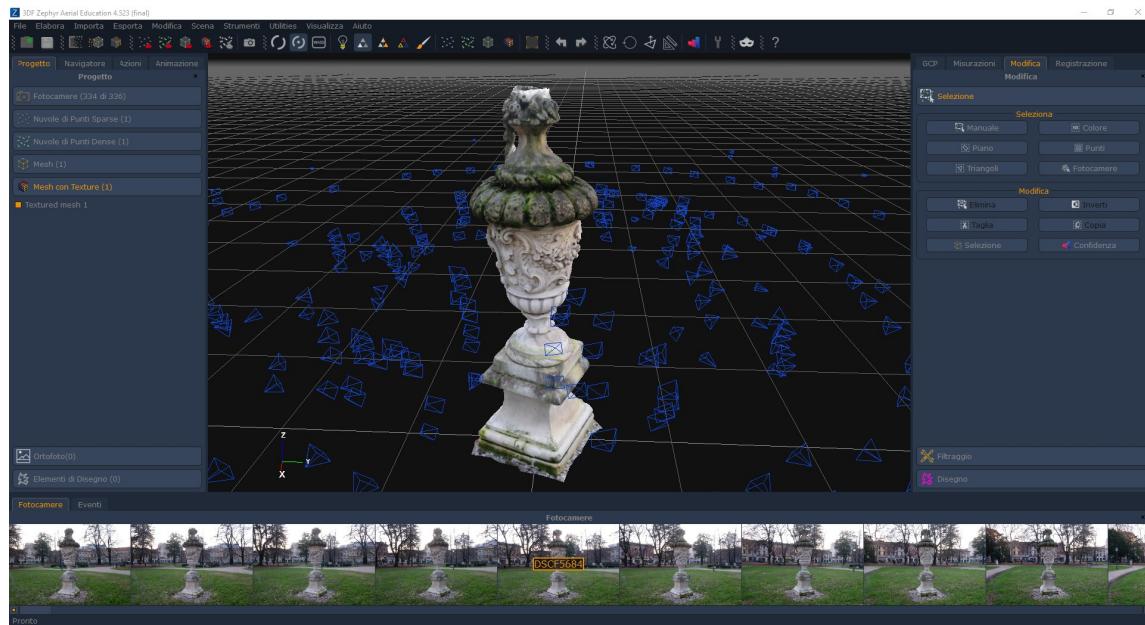


Figura 3.2: Mesh della statua ricostruita in 3DF Zephyr

3.1.4 Baking delle normali

Esportata la mesh ad alta risoluzione l'abbiamo importata nel programma "3DSMax" per calcolarne le normal map.

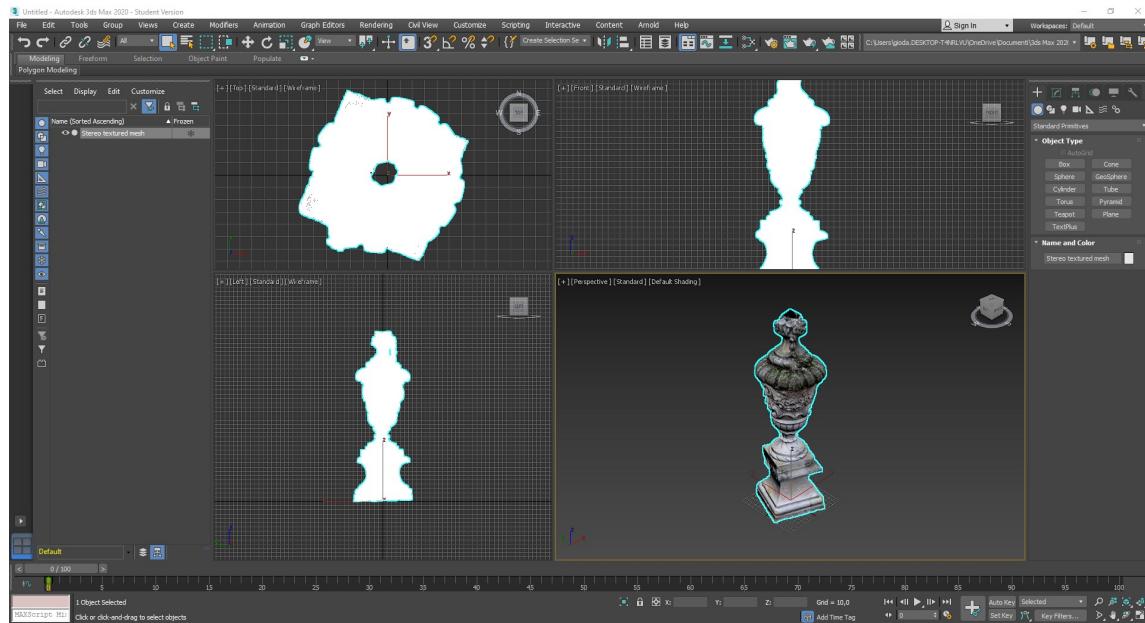


Figura 3.3: Mesh ad alta risoluzione all'interno del software 3DSMAX

L'obiettivo è quello di effettuare il baking solo nelle alte frequenze della normal map per avere un materiale di dettagli elevati.

3.1.5 Baking delle texture

Per il processo di baking delle texture, ovvero il trasferimento dei dettagli dalla mesh ad alta risoluzione alla mesh a bassa risoluzione, abbiamo deciso di utilizzare il software "Knald" in quanto è risultato essere il più veloce e accurato nel nostro caso. Inoltre grazie al supporto del formato .ply ci ha concesso più flessibilità per lavorare su Unity. Abbiamo anche testato "Substance Designer" ma non abbiamo avuto gli stessi risultati soddisfacenti ottenuti con Knald.

3.1.6 Pulizia delle texture

A questo punto abbiamo rimosso le luci e le ombre eccessive dalle texture. Per questo processo abbiamo utilizzato il software automatico di rimozione delle luci di Unity (Delightning tool). A volte il risultato di questo tool non è totalmente perfetto ed è consigliato utilizzare software aggiuntivo. Tuttavia nel nostro caso il risultato era più che buono poichè le texture non presentavano particolari problemi.

3.1.7 Finalizzazione dell'oggetto ed esportazione dell'asset

Dato che non avevamo bisogno delle texture ripetibili siamo passati alla finalizzazione e creazione dell'asset pronto per essere importato nella scena di Unity. Per fare ciò siamo nuovamente ricorsi al software 3DSMax.

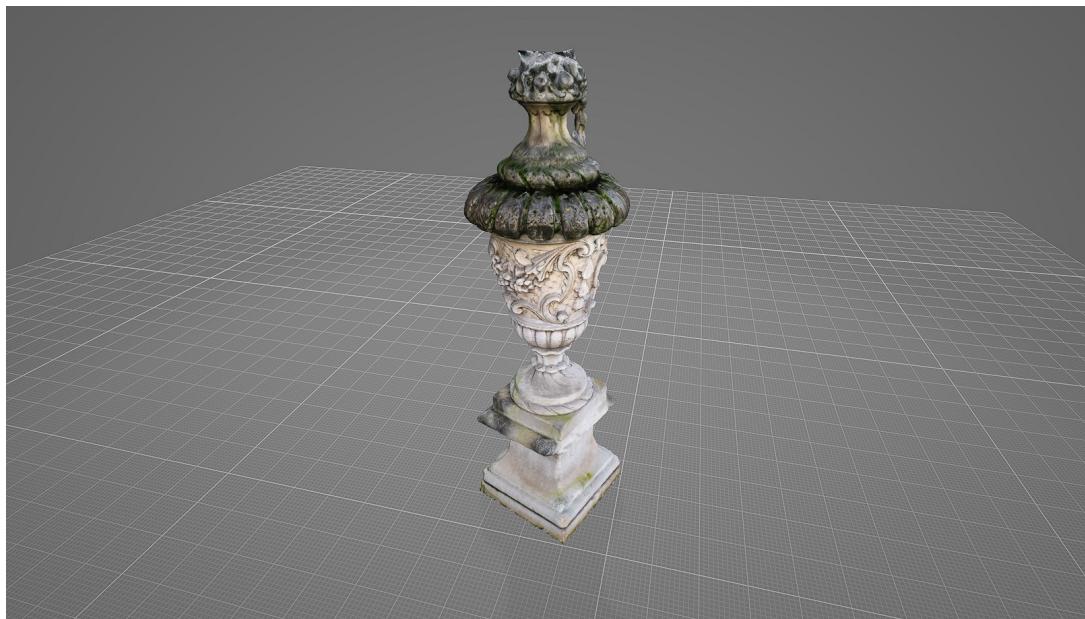


Figura 3.4: Risultato finale

Capitolo 4

Creazione del gioco nel game engine Unity

Per comporre la scena e gli oggetti creati con Zephyr ci siamo affidati al motore grafico più utilizzato al mondo: **Unity**

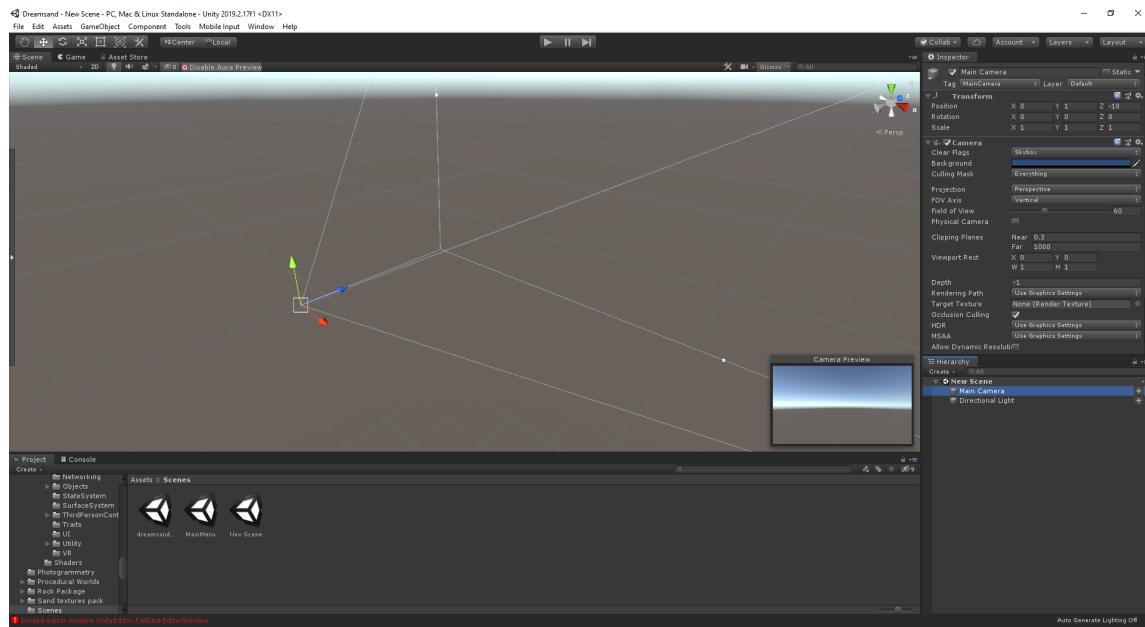


Figura 4.1: Una schermata dell'Editor di Unity

Per apprendere questo software ci siamo affidati alla documentazione ufficiale ([Unity User Manual](#)) e alla sperimentazione sulle template gratuite create da Unity. ([Standard Assets](#)). Unity permette di creare giochi compatibili su diverse piattaforme dando allo sviluppatore tutti gli strumenti necessari per concentrarsi sulla creazione e non sull'ottimizzazione.

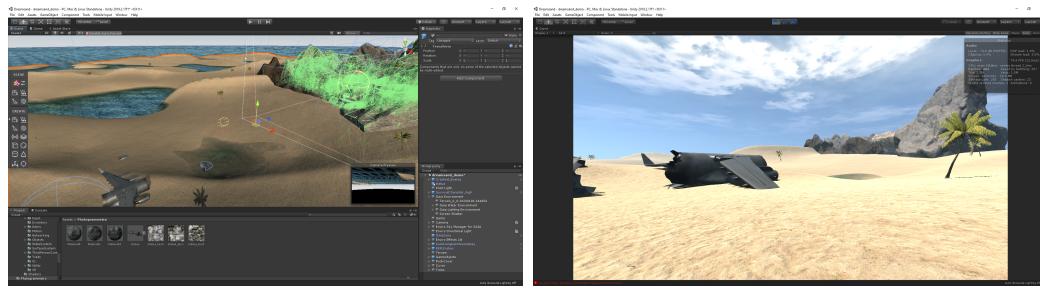
4.1 Descrizione del gioco

Siamo partiti con l'intenzione di creare due livelli di gioco, uno più esplorativo in cui ci siamo concentrati maggiormente sui dettagli grafici dei singoli elementi che compongono la scena e l'altro più interattivo per dimostrare l'ottimizzazione dei modelli generati.

4.1.1 Primo livello del gioco

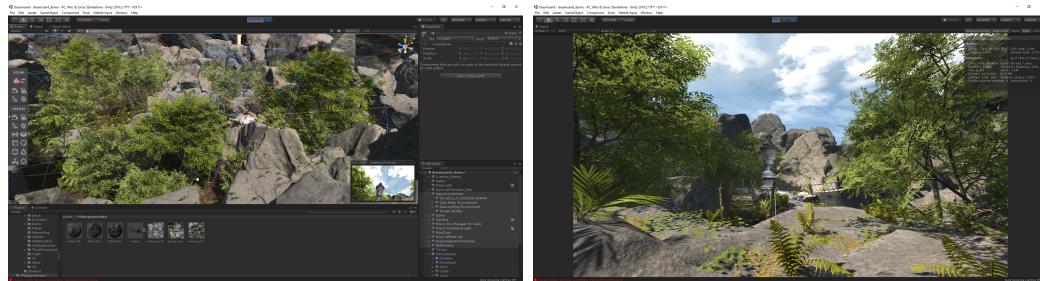
Ambientazione

Il primo livello consiste in una zona desertica con al centro una piccola oasi protetta da delle piccole montagne, dalle quali sgorga una fonte d'acqua formando una piccola cascata. L'ambiente all'interno dell'oasi è caratterizzato da una folta vegetazione simile a quella pluviale, troviamo inoltre ponti sospesi che collegano varie zone sopraelevate.



(a) Area desertica vista dall'editor di Unity.

(b) Area desertica vista all'interno del gioco



(c) L'oasi vista dall'editor di Unity.

(d) L'oasi vista all'interno del gioco

Gameplay

L'obiettivo del primo livello è quello di esplorare l'ambiente di gioco cercando di trovare uno dei nostri oggetti ricostruiti tramite la fotogrammetria, in questo caso la statua ricostruita nella sezione precedente.

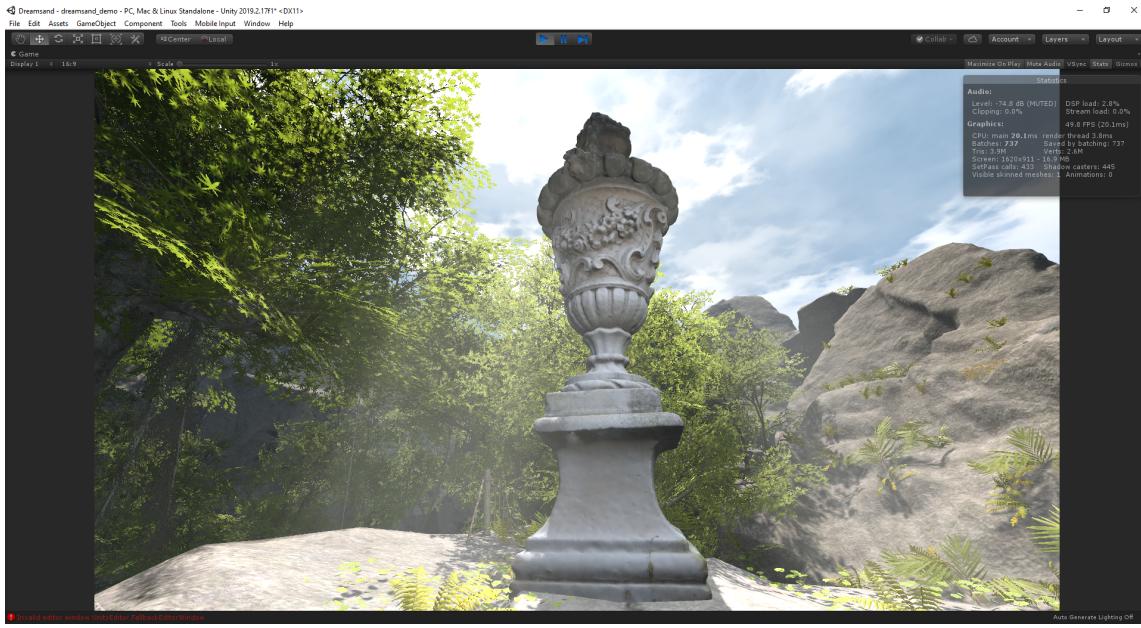


Figura 4.3: L'asset della statua all'interno del gioco

Nel corso dell'esplorazione il giocatore dovrà essere cauto, in quanto una caduta da una determinata altezza comporterà la morte del protagonista.

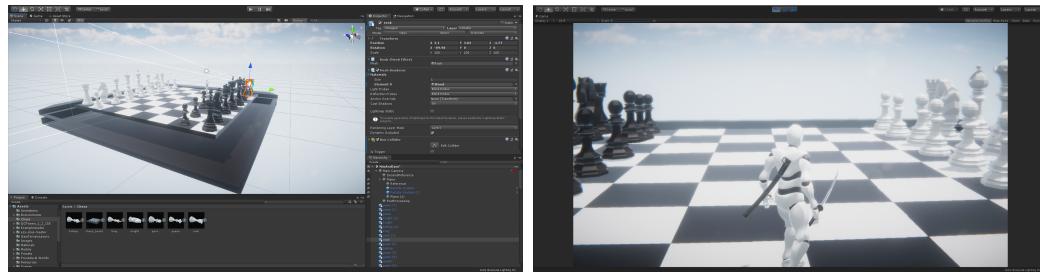
Tecniche utilizzate

- Per ridurre il carico di lavoro della gpu abbiamo utilizzato un algoritmo LOD (level of detail). Il LOD riduce la complessità della rappresentazione del modello 3D in base alla distanza dell'osservatore oppure basate su altre metriche come l'importanza di un oggetto oppure la velocità relativa al punto di vista.
- Per i collider abbiamo preferito utilizzare i *mesh collider* ai *box collider* ove possibile. I mesh collider prendono la mesh dell'asset e costruiscono i collider basandosi sulla suddetta mesh. La mesh è una collezione di vertici (vertex), bordi(edge) e facce(face) che descrivono la forma di un oggetto 3D. Un vertice è un singolo punto, un bordo(edge) è un segmento che connette due vertici e infine una faccia è una superficie piana racchiusa tra bordi(edges).
- Per definire la forma dei nostri oggetti all'interno dello spazio 3D abbiamo utilizzato la tecnica del Mesh renderer. Questa tecnica prende la geometria del filtro delle mesh e ne reindirizza la posizione definita dalla componente di trasformazione del game object.

4.1.2 Secondo livello del gioco

Ambientazione

Il protagonista si ritroverà su di una scacchiera con relativi pezzi a grandezza d'uomo. L'ambiente circostante è assente se non per una skybox per dargli una parvenza surreale.



(a) La scacchiera vista dall'editor di Unity.

(b) La schacchiera vista all'interno del gioco.

Gameplay

Nel secondo livello è possibile interagire con gli oggetti ricreati (i pezzi degli scacchi) tagliandoli in vari pezzi la cui forma dipende dal movimento di taglio della spada del protagonista.



(a) Il protagonista del gioco mentre taglia i pezzi

(b) Il risultato dei precedenti tagli sui pezzi degli scacchi

Tecniche utilizzate

- Il gameplay risulta fluido poichè gli oggetti ricostruiti hanno una geometria semplice e composta da un numero limitato di poligoni abbassando così il numero di calcoli da fare per effettuare il taglio.
- Per implementare il taglio degli oggetti abbiamo realizzato un piano parallelo al pavimento e l'abbiamo impostato come child della camera, in modo che si spostasse al movimento del giocatore. Per farlo ruotare sull'asse z lo abbiamo collegato all'asse x del movimento del mouse. Per effettivamente tagliare

l'oggetto presente nel piano abbiamo utilizzato un framework open-source chiamato Ezy-Slice creato da DavidArayan (<https://github.com/DavidArayan/ezy-slice>). L'algoritmo utilizza delle coordinate baricentriche per generare dei nuovi set UV(ovvero le proiezioni di texture bidimensionale su un oggetto tridimensionale) per i triangoli. In sostanza l'algoritmo utilizza dei "piani" per triangolare quali intersezioni tagliare individuate dai triangoli della mesh.

Capitolo 5

Conclusioni

Ringraziamo il Professor Marco Cristani per averci offerto questa opportunità in quanto è stato di nostro particolare gradimento lavorare su queste tematiche. Grazie a questa nostra esperienza abbiamo arricchito il nostro bagaglio di conoscenze, questo ci consentirà di affrontare al meglio gli studi a venire.

5.1 Bibliografia

- Tutorial 3DF Zephyr su YouTube: https://www.youtube.com/playlist?list=PLj8MbHn-bXVR5_e_h4LeG4Jzu1CM5atRH
- Materiale del corso di Elaborazione di Segnali e Immagini del Professor Marco Cristani
- Photogrammetry Workflow di Unity: https://unity3d.com/files/solutions/photogrammetry/Unity-Photogrammetry-Workflow_2017-07_v2.pdf
- Documentazione ufficiale di Unity: <https://docs.unity3d.com/Manual/index.html>
- Wikipedia: <https://www.wikipedia.org/>