POLITECNICO DI TORINO

Corso di Laurea in Ingegneria Informatica

Tesi di Laurea Magistrale

A.a 2023/2024

Studio e Sviluppo di Esperienze Museali Virtuali Immersive

Analisi della posizione e dell'attenzione degli utenti tramite heatmap

Relatori:

Andrea Bottino

Francesco Strada

Candidato:

Luca Baratta

Sessione di Laurea Novembre/Dicembre 2024

Dichiarazione

Questa è una tesi di Ingegneria Informatica, specializzazione in Grafica, in collaborazione con il Politecnico di Torino, il Dipartimento di Informatica e Automatica DAUIN e il Museo Nazionale Etrusco di Villa Giulia a Roma.

Ringraziamenti

Ringrazio la mia famiglia e i miei amici per il supporto che mi hanno dato in questi anni. Ringrazio in particolare per i preziosi consigli mio fratello Marco, Costantin Ionescu e Francesco Zumbo.

Indice

1 Introduzione

1.1 Esperienze immersive

Nel corso degli ultimi anni l'evoluzione della realtà virtuale e aumentata è cresciuta considerevolmente. A pari passo è cresciuto anche l'interesse in vari campi applicativi. Tra questi campi ha avuto fortuna quello museale, di cui lo studio oggetto di questa tesi si occupa principalmente.
Grazie alla collaborazione del Museo Nazionale Etrusco di Villa Giulia, che ha fornito il materiale, è stato possibile ricreare un ambiente virtuale con riferimento a stanze realmente presenti all'interno del museo.
L'uso della realtà virtuale per ricreare esperienze museali ha lo scopo sia di permettere a chi non ne ha la possibilità di beneficiare del patrimonio culturale che il museo ha da offrire, sia, come nel caso di questa tesi, di poter raccogliere dati riferiti ad un campione di utenti, per analisi interne e per migliorare il servizio offerto.

1.2 Sommario della tesi

In particolare, lo studio questa tesi si prefigge di visualizzare a schermo i punti di interesse sui quali gli utenti hanno prestato più attenzione, per conoscere il percorso che hanno eseguito e gli oggetti o le parti degli oggetti su cui si sono più soffermati.
Si è proceduto quindi ad una parte di registrazione dei dati, vale a dire posizione e orientamento della visuale dell'utente durante la visita, ed alla lettura grafica dei risultati per un numero prefissato di utenti. Viene restituito un risultato normalizzato per i valori massimi, in modo che siano indicati i punti con più o meno attenzione all'interno della scena. Questo risultato viene visualizzato utilizzando una heatmap, chiamata in questo modo perché generalmente è applicata nei rilevamenti di calore, dove i colori caldi (rosso, arancione, giallo), rappresentano i punti di massimo, mentre i colori freddi (blu, viola), rappresentano i punti di minimo. L'utente finale potrà quindi visionare la scena con l'heatmap applicata a tutti gli oggetti presenti, oltre al percorso visualizzato sul pavimento.
In sintesi la rassegna dei capitoli:

- Il capitolo 2 (Letteratura e stato dell'arte) presenta il campo della realtà virtuale e le sue applicazioni, in particolare nell'ambito della rappresentazione di musei e del patrimonio culturale. Vengono poi menzionati i casi precedenti riguardanti il lavoro in questa tesi, con particolare attenzione all'applicazione delle heatmap su oggetti 3D e all'interno dell'ambiente Unity.

- Il capitolo 3 (Ambiente museale) presenta una breve descrizione dell'ambiente virtuale dove verranno applicate le heatmap. L'ambiente fa riferimento al Museo Nazionale Etrusco di Villa Giulia a Roma.

- Il capitolo 4 (Metodologia) illustra il metodo con cui si sono sviluppate le mappe, attention map e heatmap, nell'ambiente di Unity. Partendo dall'attention map che rileva l'attenzione (sia per quando riguarda la posizione sia i punti di fissazione dello sguardo dell'utente), si arriva ad avere una rappresentazione tramite heatmap, con una scala di colori tipica per quel tipo di mappe. A seguire vengono presentate varie sottosezioni per la preparazione dell'ambiente antecedente alla generazione della mappe e la post-produzione di esse. È presente anche una guida all'utilizzo per l'utente finale.

- Il capitolo 5 (Sperimentazione) introduce alla fase pratica della tesi. Nell'esperimento eseguito con un campione di utenti che hanno compiuto una visita virtuale all'interno delle stanze del museo, sono stati monitorati la posizione e l'orientamento del visore in intervalli di tempo regolari. Oltre a ciò sono state registrate le reazioni emotive degli utenti. Vengono di seguito illustrate le elaborazioni dei dati che sono stati estrapolati.

- Il capitolo 6 (Risultati) chiude la parte applicativa della tesi illustrando i risultati ottenuti con l'applicazione delle mappe e l'elaborazione dei dati registrati all'interno dell'ambiente virtuale.

1.3 Tecnologie utilizzate

L'ambiente museale è stato ricreato con il motore grafico Unity, proprietà di Unity Tecnologies. Anche la creazione delle heatmap è stata sviluppata usando Unity, attraverso la URP (Universal Render Pipeline), mentre l'ambiente del museo è stato sviluppato con l'HDRP (High Definition Render Pipeline). I linguaggi utilizzati sono C#, sviluppato da Microsoft e diffuso per la piattaforma .NET e High Level Shader Language (HLSL) per la scrittura degli shader, sempre appartentente a Microsoft. Sono stati usati come VR headset il Meta Quest 2 e il Meta Quest 3, proprietà di Meta. Per l'elaborazione dei dati è stato usato il linguaggio Python, della Python Software Foundation, e sono state sfruttate le librerie pandas (`https://pandas.pydata.org/`), per la creazione di dataframes e la visualizzazione dei dati; scipy (`https://scipy.org/`) per l'individuazione di picchi; matplotlib e mpl toolkit (`https://matplotlib.org`) per la visualizzazione dei grafici, sklearn e seaborn(`https://scikit-learn.org`; `https://seaborn.pydata.org/tutorial/color_palettes.html`) per il clustering.

1.3.1 Unity

La tesi è stata sviluppata interamente con il motore grafico Unity. Unity è un motore sviluppato da Unity Technologies che fornisce strumenti per creare e gestire videogiochi e altre esperienze interattive in tempo reale, per poi essere pubblicati su varie piattaforme. Tra le varie applicazioni che hanno avuto successo negli anni sviluppate in Unity si possono citare Pokemon GO, Among Us, Genshin Impact e altri (`https://unity.com/games`).
Attraverso un editor visivo e la programmazione tramite degli script è possibile gestire in maniera agevole lo sviluppo completo di un videogioco o un'applicazione, sia bidimensionale che tridimensionale, fino ad arrivare ad una build finale come eseguibile.
Nel caso di questa tesi, si è deciso di programmare ed eseguire l'applicazione direttamente sull'Editor di Unity. L'Editor è l'interfaccia grafica che Unity offre per lo sviluppo di applicazioni, la maggior parte delle interazioni con Unity avvengono all'interno dell'Editor. Viene creato un oggetto empty invisibile nella scena che gestisce le azioni da compiere per generare le heatmap, utilizzando l'Inspector presente a destra nel layout di figura 1. Interagendo con l'Inspector la scena centrale cambia a seconda dei comandi impartiti. Per gestire al meglio il programma all'interno dell'Inspector bisogna comprendere che, a differenza dello sviluppo classico di Unity, lo stato delle modifiche si mantiene nel tempo, anche quando si riavvia l'applicazione. Per questo è meglio mantenere degli stati e avere delle funzioni che permettano di riportare la scena allo stato originale.

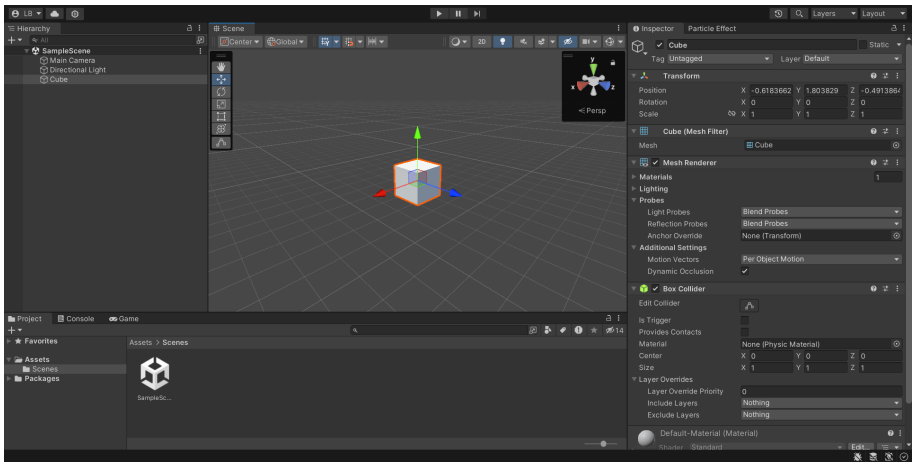


Figura 1: Esempio base del layout dell'Editor di Unity. Come si può vedere, nella parte centrale è presente l'ambiente virtuale

1.3.2 Universal Render Pipeline

La Universal Render Pipeline è una soluzione di rendering multipiattaforma costruita sul framework Scriptable Render Pipeline (SRP). La SRP è una funzionalità di Unity che consente di scrivere script C# per controllare il modo in cui Unity esegue il rendering di ciascun fotogramma. Unity fornisce due pipeline di rendering predefinite: la Universal Render Pipeline (URP) e l'High Definition Render Pipeline (HDRP). Un'altra caratteristica che contraddistingue le pipeline scriptabili è la loro natura open source.
In generale una pipeline grafica, nella computer grafica 3D, è uno speciale sottosistema software hardware che disegna in modo efficiente le primitive 3D in prospettiva. Solitamente questi sistemi sono ottimizzati per l'elaborazione di triangoli 3D con vertici condivisi. Le operazioni di base nella pipeline mappano le posizioni dei vertici 3D alle posizioni dello schermo 2D e ombreggiano i triangoli in modo che appaiano realistici e nel corretto ordine di visione [7].

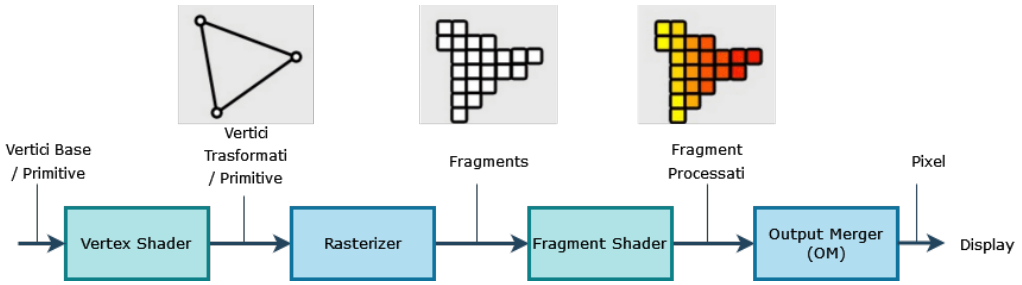


Figura 2: Esempio di pipeline grafica

In particolare, la URP è una pipeline leggera, progettata per fornire grafica di alta qualità mantenendo prestazioni ottimali. Grazie a questo la pipeline riesce ad essere compatibile con una vasta gamma di piattaforme.

1.3.3 High Definition Render Pipeline

L'HDRP si rivolge ad applicazioni realistiche, concentrandosi sulla fornitura di immagini ad alta fedeltà. Questa pipeline utilizza funzionalità hardware grafiche moderne, che consentono di creare una grafica fotorealistica ma che è meno adatta per piattaforme di fascia bassa. Le caratteristiche principali sono:

- Grafica high-fidelity: illuminazione, ombre e materiali realistici.
- Modelli di illuminazione avanzati: inclusa illuminazione volumetrica, reflection probes e layer di luce.
- Integrazione di Shader Graph e del VFX Graph: per creare shader complessi ed effetti visivi.

- Screen Space Reflection e Ray Tracing (se supportato dall'hardware): per riflessi realistici.

1.3.4 CG/HLSL

Cg (abbreviazione di C for Graphics) e High-Level Shader Language (HLSL) sono due nomi dati al linguaggio di shading di alto livello sviluppato da Nvidia e Microsoft per la programmazione degli shader. Simile al linguaggio C e usato con gli shader programmabili in DirectX. Ad esempio, è possibile usare CG/HLSL per scrivere un vertex shader o un fragment shader e usare tali shader nell'implementazione del renderer nell'applicazione Direct3D . L'unica differenza tra il linguaggio CG e HLSL è che il compilatore CG di Nvidia è compatibile sia per OpenGl che DirectX, mentre Microsoft HLSL restituisce solamente shader DirectX in formato bytecode. Gli shader HLSL possono abilitare molti effetti speciali nella computer grafica sia 2D che 3D. Il linguaggio Cg/HLSL originariamente includeva il supporto solo per vertex shader e pixel shader, ma gradualmente furono introdotti anche altri tipi di shader (`https://en.wikipedia.org/wiki/Cg_%28programming_language%29`):

- DirectX 10 (Shader Model 4) e Cg 2.0 hanno introdotto gli shader geometrici.
- DirectX 11 (Shader Model 5) ha introdotto shader di calcolo (GPGPU) e i texel shader. Questi ultimi sono presenti in Cg 3.1.
- DirectX 12 (Shader Model 6.3) ha introdotto gli shader di ray tracing (generazione di raggi, intersezione, hit / miss).

2 Letteratura e stato dell'arte

2.1 Realtà virtuale

ll fine della realtà virtuale è simulare un ambiente reale per mezzo di tecnologie elettroniche, sino a dare a chi la sperimenta l'impressione di trovarsi realmente immerso in quell'ambiente. Al giorno d'oggi il termine VR (Virtual Reality) è molto più ampio di un tempo, partendo dalle applicazioni desktop a quelle immersive.
Il tipo più immersivo di realtà virtuale utilizza un display stereoscopico posto davanti agli occhi dell'utente con una qualche forma di tracciamento del movimento per determinare dove l'utente stia guardando. La visione del mondo esterno è completamente esclusa, creando una forte sensazione di immersione e fornendo allo stesso tempo una visione senza ostacoli del mondo virtuale. Sebbene gli head mounted display esistano da decenni, sono diventati disponibili a livello consumer nel 2014 con l'introduzione dell'Oculus Rift. Ad oggi i modelli di head mounted display hanno avuto un'ampia crescita, con caschetti come HTC Vive, Google Cardboard e Samsung Gear VR, tra gli altri.
La VR si è espansa in diversi ambiti nel corso degli anni, tra i quali il cinema e i videogiochi [3], ma anche nell'ambito lavorativo, militare [1], clinico, dell'insegnamento e della cultura [15].
Oltre alla VR bisogna citare l'AR (Augmented Reality) dove, a differenza della realtà virtuale, gli utenti sperimentano un ambiente del mondo reale a cui si sovrappongono informazioni generate dal software. L'AR ha avuto successo grazie alla facilità con cui si può usufruirne, basta infatti uno smartphone o un PC dotato di webcam per avere un'esperienza in realtà aumentata.

2.2 Applicazioni in campo museale

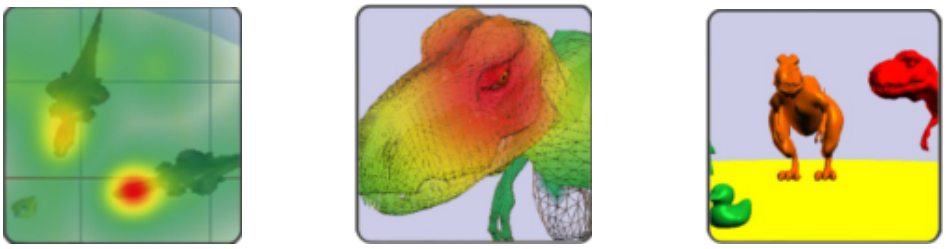
Come già è stato introdotto, uno dei campi in cui la realtà virtuale ha avuto più applicazione è quello delle rappresentazioni museali.
Il primo utilizzo di una presentazione VR in un'applicazione riguardante il patrimonio culturale risale al 1994, con l'osservazione di un visitatore al museo che poteva compiere un "walk-through" interattivo di una ricostruzione 3D del castello di Dudley in Inghilterra com'era nel 1550. Questo sistema consisteva in un disco laser controllato da un computer, progettato dall'ingegnere britannico Colin Johnson. Il lavoro finale fu presentato in una conferenza tenuta dal British Museum nel 1994 e successivamente pubblicata nella rivista Imaging the Past - Electronic Imaging and Computer Graphics in Museums and Archaeology [5].
Queste applicazioni possono avere sviluppi sia in realtà aumentata, direttamente in loco, cercando di aumentare l'immedesimazione durante la visita, sia con l'ausilio

di un visore VR, cercando di ricreare l'ambiente del museo. Oltre a questo, musei e siti storici sono in grado non solo di fornire una maggiore accessibilità al pubblico in generale, ma anche ai ricercatori che altrimenti non sarebbero in grado di usufruire di questi manufatti e di questi siti. In particolare per i siti archeologici, il sistema non presenta alcun rischio di danni al sito, né visivi né di disturbo dei visitatori o interruzione della normale attività funzionamento. Inoltre, la VR presenta un'interfaccia molto semplice per creare nuovi contenuti e arricchire i database con informazioni scientifiche.[14]
Gli strumenti digitali consentono alla comunità accademica globale di visionare questi manufatti e siti in un qualsiasi luogo senza rischi e possono fornire al ricercatore strumenti anche più accurati dei metodi analogici.
Per fare sì che la ricreazione dell'ambiente avvenga, è necessario un lavoro di digitalizzazione dei manufatti. Prima di digitalizzare i manufatti, i musei devono innanzitutto proporre una mostra virtuale, stilando un elenco di manufatti da esporre e identificando gli altri oggetti multimediali che comporranno l'ambiente. Una volta che queste informazioni sono acquisite, il museo può iniziare a digitalizzare i manufatti e a inserire tutti i dati necessari in un database [16].
Da notare però che il museo virtuale non deve essere considerato come una trasposizione di un museo reale in forma elettronica o sul web, né può essere inteso come uno strumento supplementare per completare il museo reale, come una sorta di spazio espositivo o di ulteriore catalogo digitale. La sua natura è strettamente legata al suo scopo finale, ovvero quello di comunicare la conoscenza al più vasto pubblico possibile, senza fermarsi alle generalizzazioni tra "studiosi" e "pubblico". Questo obiettivo viene raggiunto con l'utilizzo di strategie di comunicazione basate su narrazioni visive, interattive e multimediali che raccontano la storia di ciascun manufatto, contestualizzandolo geograficamente, storicamente e culturalmente e inserendolo in una rete di informazioni che va oltre il manufatto stesso e ciò che il museo reale contiene. In questa prospettiva, il museo virtuale trova la sua massima espressione nella versione del "museo del territorio" che virtualmente fa uscire dalle mura chiuse del museo il bagaglio culturale originario di un intero territorio [13].

2.3 Casistiche precedenti

La ricerca di una raffigurazione visiva dei dati dell'attenzione posta da un utente è stata proposta molte volte, come si evince nell'articolo "Visualization of Eye Tracking Data: A Taxonomy and Survey: Visualization of Eye Tracking Data" [2]. In questo articolo vengono presentate le tecniche e i punti in comune tra circa 90 studi eseguiti nel corso degli anni. L'evoluzione di questi studi parte da esperimenti su l'analisi di immagini bidimensionali, con l'utilizzo di grafici e heatmap, per giungere a rappresentazioni grafiche tridimensionali, tenendo conto anche del tempo come fattore, fino all'analisi di scene virtuali solamente

nell'ultimo periodo. I dati in input, solitamente di dimensioni molto grandi, possono essere più o meno precisi in termini sia di campionamento, sia con l'aggiunta di dati più specifici (come ad esempio la dilatazione delle pupille). Le tecniche utilizzate si dividono principalmente in due categorie: tecniche di visualizzazione basate su punti, tecniche basate su AOI (Area Of Interest) e una combinazione delle due. Nel svolgimento di questa tesi è stata adottata una tecnica basata su punti, anche se sarebbe facilmente espandibile a quella basata su Aree di Interesse, dividendo i valori dell'heatmap in modo che ogni oggetto abbia un suo valore (uniforme) rispetto all'ambiente globale, come mostrato in figura 3c.



(a) Tecnica basata su punti in una proiezione 2D

(b) Tecnica basata su punti in una scena 3D

(c) Tecnica basata su AOI in 3D

Figura 3: Le tre rappresentazioni delle tecniche utilizzate per generare un'heatmap

Ulteriori informazioni possono essere estrapolate dalla visualizzazione dei dati di attenzione, come le fissazioni; gruppi di punti di attenzione; le saccadi, movimenti veloci dell'occhio che passano da una fissazione all'altra. Le aree di interesse sono un raggruppamento di fissazioni e saccadi, e l'utente può passare da un'area all'altra attraverso le transizioni. Nel caso delle heatmap una fissazione con una concentrazione di punti più alta viene rappresentata con colori caldi, una più bassa con colori freddi.

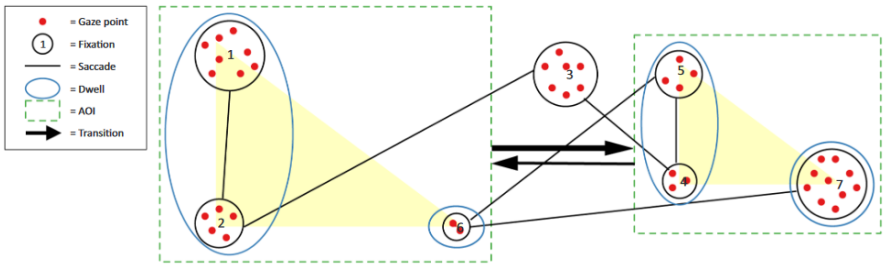


Figura 4: I punti di attenzione sono raccolti nelle fissazioni, le saccadi collegano le fissazioni tra loro, le AOI sono le aree di interesse e il passaggio tra loro si chiama transizione. Una sequenza di fissazioni e saccadi è chiamata *scanpath*.

Inizialmente le visualizzazioni venivano rappresentate con diagrammi che mostravano le statistiche dei dati quantitativi. Questa rappresentazione però è limitante se si vuole fare un'analisi di carattere più qualitativo e immediato. Con le heatmap si riesce a sopperire a questa mancanza. Nell'articolo viene anche affrontato il problema della presenza di più utenti, che possono generare troppi dati e causare "ingombro visivo". Tecniche come la media o il raggruppamento di dati possono essere utilizzati in questo caso.

Per lo sviluppo della tesi si è fatto riferimento all'articolo "GPU-accelerated attention map generation for dynamic 3D scenes" [9]. Nello specifico lo sviluppo base delle attention map e delle heatmap. Il progetto dell'articolo va oltre, sviluppando le heatmap anche per mesh animate e in movimento, che però non è di interesse in questa tesi. Diversamente, i dati di input binoculari, catturati con particolari occhiali che tracciano il movimento delle pupille, potrebbero aggiungere un fattore di precisione in più, ma si è deciso, nella ricerca oggetto di questa tesi, di limitarsi alla visualizzazione mono-oculare.

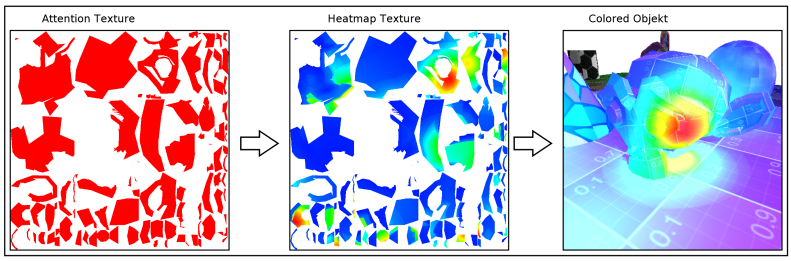
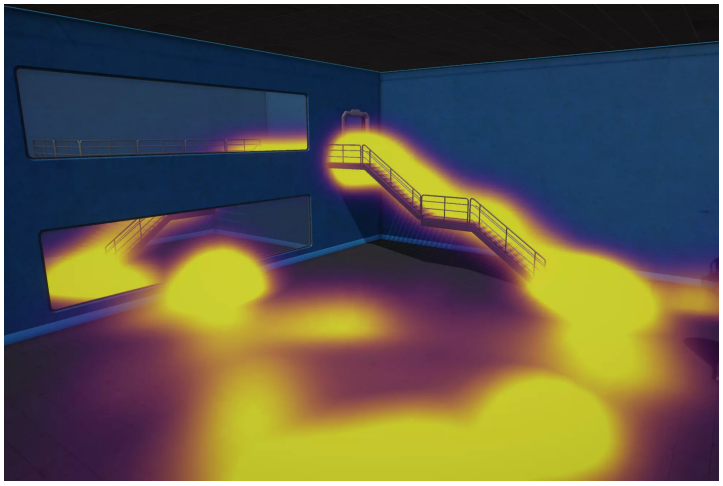


Figura 5: Workflow: i valori di attenzione vengono aggiornati nell'attention texture. Dopo aver determinato la scala globale, i valori di attenzione possono essere mappati su una rampa di colore definita e visualizzati sulla superficie del modello.

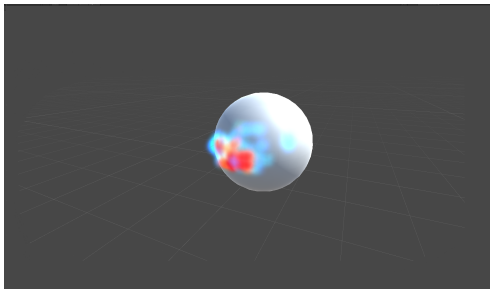
Ogni modello possiede una texture che memorizza i valori di attenzione. In questo modo, ogni regione della mesh che possiede coordinate di texture sarà automaticamente in grado di memorizzare i valori di attenzione nelle regioni di contorno della texture di attenzione.
L'idea di base del progetto è quella di lasciare che la GPU si occupi dei calcoli per la generazione delle heatmap, in modo che si possa applicare anche in real-time. Nel caso di questa tesi, invece, si è proceduto sul caricamento di dati da file di testo. Seppur non in real-time, la velocità della GPU rende i tempi di generazione delle heatmap assai più veloci, data la gran mole di dati a cui far fronte.
Una parte di sviluppo in real-time è stata comunque fatta per questioni di debugging, e avere questa caratteristica come supporto potrebbe aiutare nelle fasi di testing.

2.3.1 All'interno dell'ambiente Unity

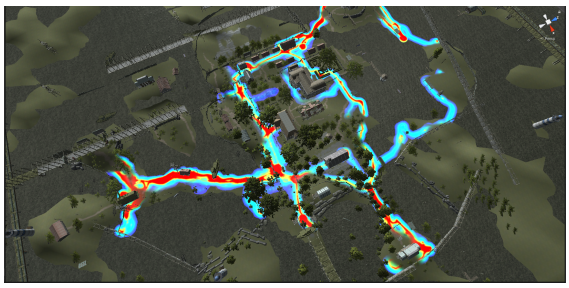
Per casistiche precedenti sviluppate con la piattaforma Unity, si fà riferimento a due progetti: Volumetric Heatmap (`https://assetstore.unity.com/packages/tools/utilities/volumetric-heatmap-247453?srsltid=AfmBOopvZhECmw73y559fceNUHQGS6mwOmxHheZOwasRmNMpygTgHRKD`) (figura 6a) presente nell'Asset Store ufficiale di Unity, e "heatmap-unity" (`https://github.com/kDanik/heatmap-unity?tab=readme-ov-file`) (figura 6b e 6c) pubblicato su GitHub.



(a) Heatmap volumetrica



(b) Heatmap particellare sull'attenzione



(c) Heatmap particellare sulla posizione

Figura 6: Versioni di heatmap in Unity, generate con nuvole particellari

In entrambi i casi si utilizza un'heatmap volumetrica, facendo uso dei sistemi particellari presenti in Unity. Un sistema particellare simula e renderizza molte piccole immagini o mesh, chiamate particelle, per produrre un effetto visivo. Ogni particella in un sistema rappresenta un singolo elemento grafico nell'effetto. Il

sistema simula contemporaneamente ogni particella per creare l'impressione dell'effetto completo.
Seppure il risultato sia funzionale per determinare la posizione dell'utente in uno spazio tridimensionale, per quanto riguarda la generazione dell'attenzione sugli oggetti, questa è stata sperimentata solo nel secondo progetto e, a parere di chi scrive, in uno stato embrionale.
In questa tesi invece si farà ricorso agli shader e alla scrittura dei valori direttamente sulle texture.

3 Ambiente museale

L'ambiente applicativo della tesi è stato ricreato a partire da tre stanze del Museo Nazionale Etrusco di Villa Giulia, di cui due completamente visitabili. L'utente comincia l'esperienza da un'anticamera con un'architettura neutra. Attraversando un corridoio si entra nella prima stanza, dove sono presenti prevalentemente ceramiche e utensili dell'epoca etrusca, racchiusi in teche di vetro. Dall'entrata si può intravedere la seconda stanza, la sala 12 Cerveteri, con presente al centro il Sarcofago degli Sposi. Questa scultura, fulcro dell'ambiente, è datata nel periodo compreso tra il 520 e il 530 a.C. in età Etrusca e rappresenta una coppia adagiata su un lettino reclinato. Ricomposto da circa quattrocento frammenti, il Sarcofago degli sposi è in realtà un'urna destinata ad accogliere i resti materiali dei due defunti. Plasmata a tutto tondo, l'opera rappresenta una coppia di coniugi distesi su un letto con il busto sollevato frontalmente nella tipica posizione del banchetto (`https://www.museoetru.it/opere/sarcofago-degli-sposi`). Infine, preceduta da una rampa di scale, in un piano sopraelevato, è presente la terza stanza, incompleta perché inaccessibile all'utente. Sono presenti comunque delle ceramiche visibili dall'utente nella terza stanza per mantenere l'immersione nell'ambiente.
Sono state successivamente create altre due varianti che riguardano principalmente modifiche alla terza stanza.
Nella prima variante la teca singola più vicina alla stanza Cerveteri è stata rimossa. Al suo posto è stato aggiunto un pannello per nascondere il sarcofago alla vista quando comincia la visita. Delle frecce segnalatrici indicano il proseguimento alla terza stanza. L'accesso alla terza stanza è stato bloccato e la stanza eliminata.
La seconda variante riprende le modifiche della prima, ma la stanza con il sarcofago è stata modificata con una volta celeste. La stanza viene oscurata per avere un effetto notte ed è avvolta da una calotta stellata per rendere l'esperienza più suggestiva.

(a) Anticamera

(b) Terza stanza(incompleta)

(c) Prima stanza con ceramiche

(d) Seconda stanza sala Cerveteri

Figura 7: Le quattro stanze del museo

4 Metodologia

4.1 Premessa

In questo capitolo verrà presentata la metodologia riguardante l'attention map e l'heatmap.
L'attention map è una texture che permette all'utente di visualizzare i punti di interesse, prendendo come input dati che sono stati registrati nella fase di esperienza all'interno dell'ambiente virtuale del museo. L'heatmap è una texture derivata dall'attention map, che permette di visualizzare i dati in maniera più esaustiva con l'aiuto di una colorazione diversa a seconda del grado di interesse. Queste due texture possono indicare sia il percorso compiuto dall'utente, in questo caso la texture viene proiettata su una superficie piatta, il pavimento, sia che cosa l'utente ha visto, in questo caso la texture è proiettata direttamente sulle superfici degli oggetti.
Verranno poi esposte le successive elaborazioni di queste mappe, come la possibilità di cambiare la densità dei valori massimi e di quelli minimi, con un conseguente cambiamento generale nell'aspetto della scena. Il filtraggio dei dati può avvenire per utente o per intervalli di tempo. Inoltre, si procede alla preparazione della scena all'interno di Unity, antecedente alla generazione della heatmap, affinché queste funzionino a dovere. La creazione di "cache" per una miglior gestione della memoria e il salvataggio delle texture su disco. Il procedimento effettivo, sempre all'interno dell'ambiente di Unity (nell'Editor in particolare), per generare le mappe.

4.2 Attention Map

4.2.1 Shaders

Gli shader sono programmi di basso livello che vengono compilati ed eseguiti in fasi specifiche della pipeline grafica. (`https://learn.microsoft.com/en-us/windows/win32/direct3dgetstarted/work-with-shaders-and-shader-resources`). Solitamente sono suddivisi in due categorie principali: i vertex shader e i fragment shader.

- I vertex shader vengono eseguiti per ogni vertice nella scena, dal vertex buffer verranno poi rasterizzati nella posizione del pixel nello schermo. Possono venir usati per la manipolazione dei vertici in tempo reale senza avere un carico pesante sulla CPU, essendo eseguiti interamente dalla GPU (ad esempio: i fili d'erba in un prato che vengono mossi dal vento).
- I fragment shader vengono eseguiti per ogni pixel e ne determinano il colore finale visualizzato a schermo. Il fragment shader riceve come input i dati del vertex shader e usa dei metodi di interpolazione per tradurre questi dati da vertice a pixel. Nel nostro caso il fragment shader permette di generare l'attention map con valori corrispondenti al rosso. Un altro shader traduce questi valori di rosso nei colori dell'heatmap.

4.2.2 Fase di Registrazione

La fase di registrazione avviene all'interno dell'ambiente virtuale. I dati vengono salvati all'interno di un file .csv(Comma Separated Values). Grazie a questo formato, si possono salvare liste di dati divise in categorie, ognuna identificata con un titolo sulla prima riga
(`https://en.wikipedia.org/wiki/Comma-separated_values`). Le categorie a cui si farà riferimento sono:

- l'istante (time) di campionamento dei dati, a partire dall'inizio della sessione
- la posizione (position) della telecamera durante quel preciso istante
- l'orientamento (rotation) della telecamera in quell'istante

L'input necessario per generare l'attention map sarà la posizione e l'orientamento della telecamera. L'orientamento, in particolare, servirà per trovare la direzione perpendicolare alla vista della telecamera, dove, facendo passare un raggio fittizio, colpirà il punto della mesh dell'oggetto che si troverà davanti all'utente. Il tempo ha lo scopo di comprendere il percorso dell'utente fino a quell'istante, o il percorso compiuto fra due istanti di tempo.

4.2.3 Double Buffering

Per fare in modo che l'attention map funzioni, c'è bisogno che il calcolatore ricordi i momenti passati in modo che sia fatta una somma finale di tutte le attenzioni registrate.
Nel progetto iniziale si è provato a salvare le posizioni generate all'interno di un vettore, o per meglio dire un buffer. Questo metodo è però limitante, in quanto viene sprecata una gran quantità di memoria, oltre che infattibile in questo caso, in quanto i vettori all'interno degli shader in Unity possono essere solamente di dimensioni limitate.
Avendo a che fare con una grande quantità di dati, si è deciso di adottare la tecnica del double buffering. Questo metodo alloca in memoria due buffer, che vengono comunemente chiamati front buffer e back buffer (`https://wiki.osdev.org/Double_Buffering`). In questo caso, per semplicità e chiarezza, verranno chiamati buffer A e buffer B. Mentre il buffer A ha immagazzinato l'output dello shader, nel buffer B vengono scritti i risultati dell'ultima attention, avendo in input i risultati del buffer A. Il passaggio successivo consiste nello scambiare il buffer A con il buffer B e viceversa, in questo modo quello che precedentemente era il buffer A, ora buffer B, immagazzina i risultati, mentre il buffer A scrive i risultati dell'ultima attention avendo già in memoria tutti i dati delle attention precedenti. A ogni aggiornamento dell'attention vengono scambiati i buffer, il back buffer avrà dunque al suo interno tutti i dati salvati fino a quel momento e il front buffer alla fine mostrerà i risultati a schermo.

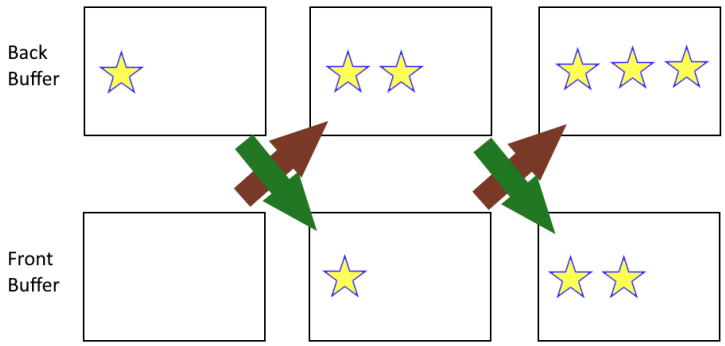


Figura 8: Funzionamento base del double buffer

4.2.4 Camera Secondaria

All'interno della scena è stata prevista una seconda telecamera, che chiameremo SubCamera, che servirà a renderizzare su una delle RenderTexture usate per il double buffering. In Unity, la RenderTexture può venire impostata come proprietà "Render Target" di una telecamera (`Camera.targetTexture`), in modo che la telecamera stessa esegua il rendering in una texture invece di eseguirlo sullo schermo(`https://docs.unity3d.com/ScriptReference/RenderTexture.html`). Un oggetto invisibile che chiameremo ShadowPrefab verrà instanziato all'interno della scena per ogni oggetto, da cui prenderà le informazioni della mesh dell'oggetto originale. In questo modo potranno essere usati in particolare le informazioni dei vertici del modello originale.
Infine, la mappa UV dell'oggetto verrà renderizzata tramite lo shader presente negli ShadowPrefab davanti alla SubCamera, che renderizzerà l'attention map sulla RenderTexture, che infine verrà copiata sull'oggetto visibile in scena. Con la mappa UV la mesh viene "appiattita" su un piano, sul quale giace un'immagine. A questo punto ogni vertice dell'oggetto tridimensionale disporrà di un set di coordinate bidimensionali condiviso con l'immagine, che potrà essere quindi associata alle sue facce, risultando visibile nello spazio 3D. Le coordinate UV fanno quindi da ponte tra lo spazio bidimensionale delle immagini (texture) e quello tridimensionale della mesh (`https://it.wikipedia.org/wiki/Mappatura_UV`).

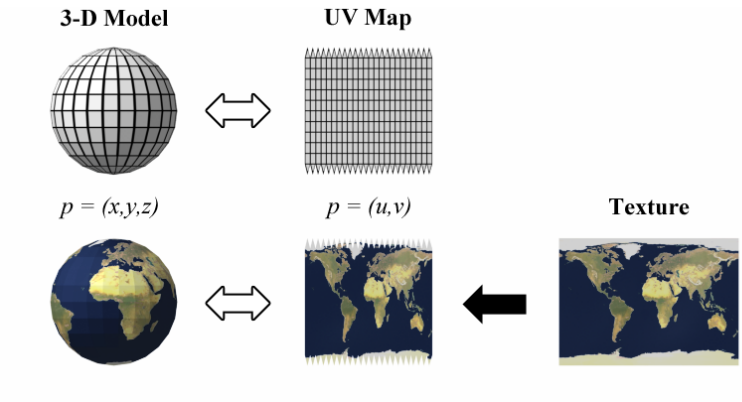


Figura 9: Rappresentazione della mappatura UV

Per evitare che la SubCamera renderizzi un oggetto alla volta, viene impostata una culling mask. Quest'ultima è impostata con un layer apposito così che la SubCamera renderizzi solamente gli oggetti con quel layer. In questo modo, quando è tempo di renderizzare la mappa UV, verrà posto automaticamente sull'oggetto il layer apposito, mentre tutti gli altri oggetti che avranno un layer diverso non verranno considerati. Per quanto riguarda la culling matrix, vale a dire

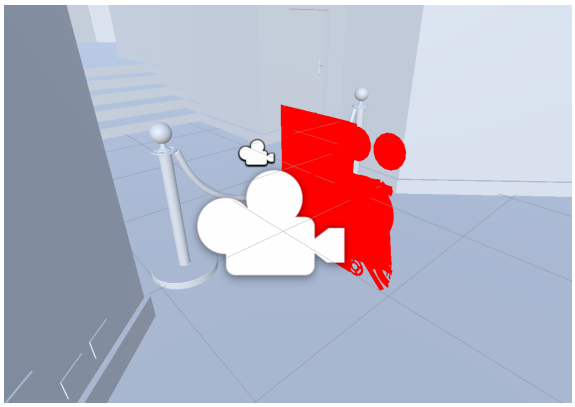


Figura 10: SubCamera rappresentata come gizmo all'interno della scena di Unity. Davanti ad essa sono presenti le mappe UV degli oggetti in scena, generate dagli shader presenti negli ShadowPrefab di ogni oggetto.

la matrice che determina la zona visibile davanti alla camera, dove tutti gli oggetti al di fuori di essa vengono "nascosti", c'è il bisogno di replicare la camera come se fosse davanti all'oggetto considerato in quell'istante. Dato che la culling matrix è il prodotto fra la view matrix e la projection matrix, si replica la view matrix con la posizione e la direzione della camera che si ha come dati in input. Per quanto riguarda la projection matrix, è stata usata la matrice di proiezione standard, prospettica, fornita da Unity. Di default la SubCamera è centrata nelle coordinate (0.5, 0.5, -0.5), in modo che punti nella direzione dell'asse z del mondo, vale a dire alle coordinate (0.5, 0.5, 0) dello spazio globale, dove sono presenti le proiezioni UV di tutti gli oggetti in scena, generati dagli shader collegati ai ShadowPrefab. Aggiungendo una matrice di traslazione classica, in rapporto alla posizione della SubCamera, è possibile spostarla in qualsiasi punto, in modo che questa non sia visibile all'interno della scena. Una matrice di traslazione sposta un oggetto, come insieme di vertici, lungo uno o più dei tre assi del mondo (`https://www.brainvoyager.com/bv/doc/UsersGuide/CoordsAndTransforms/SpatialTransformationMatrices.html`). Una matrice di trasformazione che rappresenta solo la traslazione ha questa forma:

$$T = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

dove il vettore (t_x, t_y, t_z) rappresenta la traslazione, nel nostro caso lo spostamento della SubCamera dal centro.

4.2.5 Procedimento

Per ogni oggetto su cui occorre fare l'analisi bisogna aggiungere un mesh collider. Un collider è un componente di Unity che definisce la forma di un oggetto al fine di rilevarne le collisioni fisiche. I collider sono invisibili e non devono necessariamente avere la stessa forma della mesh dell'oggetto di gioco(`https://docs.unity3d.com/Manual/CollidersOverview.html`). In particolare tra i vari tipi di collider, il Mesh Collider ha la stessa forma dell'oggetto. Ha però lo svantaggio di essere molto pesante in termini computazionali ed è in generale sconsigliato. Nel nostro caso però si rivela necessario per avere dei risultati precisi quando si andrà a disegnare l'heatmap sugli oggetti.
All'oggetto originale viene poi legata la sua "ombra", un oggetto chiamato ShadowPrefab invisibile, che recupera le informazioni della mesh a cui è attaccato lo shader che calcolerà l'attention map. Questo oggetto "di appoggio" è necessario per il rendering dell'attention map, in quanto la texture su di esso viene posta, tramite lo shader, davanti alla SubCamera.
Per ogni intervallo di tempo omogeneo e arbitrario, viene lanciato un Raycast perpendicolare al piano della camera.
Un Raycast è una funzione appartente all'interfaccia Physics di Unity, che proietta un raggio nella scena, restituendo un valore booleano se un bersaglio è stato colpito con successo. Quando ciò accade, le informazioni sul colpo, come la distanza, la posizione o il riferimento alla transform dell'oggetto, possono essere memorizzate in una variabile Raycast Hit per uso successivo. Questo rende il Raycast estremamente utile per ottenere informazioni su altri oggetti, per eseguire controlli sul terreno o, in generale, per fare qualsiasi cosa che implichi una connessione tra due oggetti(`https://gamedevbeginner.com/raycasts-in-unity-made-easy/`).
Il raggio indica quindi la direzione della visuale dell'utente, che servirà a calcolare l'attenzione complessiva. Le coordinate del punto di intersezione tra la mesh dell'oggetto e il raggio vengono usate come input dello shader. Oltre al punto di intersezione, lo shader riceve come input la posizione della camera, il raggio dell'area di effetto dell'attention map e l'attention map precedente, necessaria per il double buffering.
Lo shader, che, come esposto nella sezione Shaders a pagina 15, fa due cicli (uno per ogni vertice della mesh e uno per ogni fragment), si occupa della visualizzazione dell'heatmap per ogni oggetto interessato. Al suo interno troviamo la struttura base di Unity, lo ShaderLab (`https://docs.unity3d.com/Manual/SL-Reference.html`), con i dati di input per il vertex shader, nella `struct appdata`, che Unity genera in automatico partendo dalle informazioni della scena. Questi dati sono:

- `vertex`: la posizione del vertice in rapporto alle coordinate del modello
- `uv`: le coordinate della mappa UV della texture

- `normal`: il versore normale al vertice

Questi dati vengono poi manipolati nella funzione `v2f vert(appdata v)`, che avranno come output una `struct v2f`(vertex to fragment), che sarà anche l'input per il fragment shader. Particolare attenzione è stata prestata alla prima coordinata, la `SV_POSITION`, quella che renderizza nella scena il risultato del fragment shader. Contrariamente a come si è soliti fare, renderizzando la mesh esattamente dove sono i vertici nello spazio globale (in realtà nel Clip Space, che è lo spazio visualizzato sullo schermo) , tramite la funzione `o.position = UnityObjectToClipPos(v.vertex)`, che equivale a `o.position = mul(UNITY_MATRIX_MVP, v.vertex)`, quello che viene visualizzato è la mappa uv posta davanti alla SubCamera, a una distanza tale che occupi tutto il Clip Space di quest'ultima. Questo viene ottenuto moltiplicando il valore dell'uv (trasformato in un vettore di quattro dimensioni) con la matrice `UNITY_MATRIX_VP`. Questa costante rappresenta il prodotto tra la View Matrix e la Projection Matrix. La View Matrix trasforma i vertici da World Space a View Space ed è l'inverso della matrice di trasformazione della telecamera (`https://www.3dgep.com/understanding-the-view-matrix/`). La Projection Matrix è progettata per trasformare un punto 3D nello spazio della telecamera nella sua controparte proiettata sulla piano d'immagine. In sostanza, quando si moltiplica un punto 3D per una matrice di proiezione, si determinano le sue coordinate 2D sul piano d'immagine(`https://www.scratchapixel.com/lessons/3d-basic-rendering/perspective-and-orthographic-projection-matrix/projection-matrix-introduction.html`). La Model Matrix, che in questo caso manca, trasforma lo spazio da quello del modello a quello globale. L'omissione di quest'ultima matrice permette al fragment di visualizzare la mappa uv al centro del mondo, davanti alla SubCamera, posta di base alle coordinate (0.5, 0.5, -0.5), rivolta verso l'asse z. La variabile `offset` usa la matrice di traslazione della telecamera, in coordinate del mondo, per spostare la SubCamera arbitrariamente, senza il vincolo di dover essere centrata nell'origine. L'offset farà seguire il fragment sempre davanti alla SubCamera.
All'interno del fragment shader viene calcolata e visualizzata l'effettiva attention map. Il valore di attention viene calcolato per ogni fragment rispetto alla distanza dal punto di contatto. Il punto di effettivo contatto avrà un valore massimo, mentre i valori più distanti avranno valori sempre più bassi, secondo una distribuzione normale. La distribuzione normale(o gaussiana), genera una curva con una forma a campana, simmetrica rispetto al suo punto di massimo: inizia a salire lentamente, poi in modo sempre più ripido fino a raggiungere il valore massimo; da quel punto in poi scende in maniera simmetrica(`https://www.prismamagazine.it/2020/05/06/la-curva-gaussiana/`. La formula della distribuzione normale nella sua forma generale si presenta come segue:

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} exp^{-\frac{((x-\mu)^2}{(2\sigma^2)}}$$

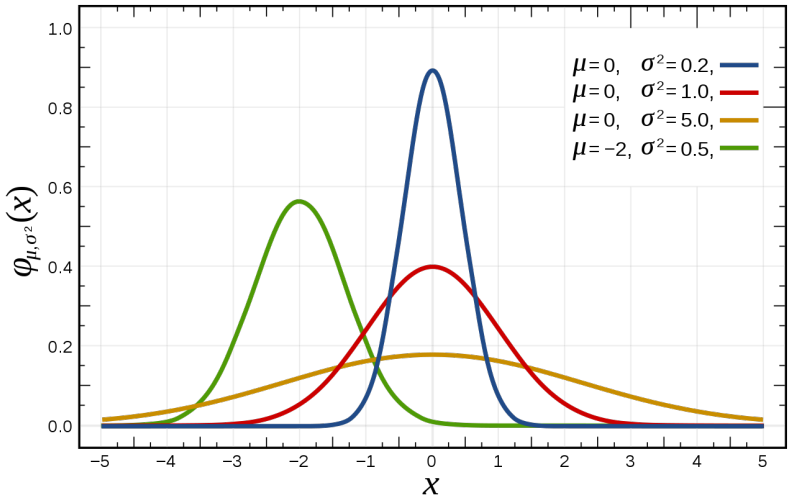


Figura 11: Curve gaussiane con diversi valori di σ e μ

Come si può vedere, la curva rappresenta una distribuzione massima al centro, che scende in maniera esponenziale mano a mano che ci si allontana. Nella forma semplificata che viene usata, x indica la distanza tra il punto di collisione e il fragment, μ è nullo perché il valore massimo è centrato, σ è il raggio di applicazione dell'attention map, diviso per due per avere un'ampiezza della curva unitaria a raggio unitario. La funzione utilizzata è:

```
funzione di distanza gaussiana(punto a, punto b, raggio):
    calcola exp(-pow(distanza(a, b) / raggio, 2)) /(sqrt(2*PI))
    ritorna il valore
```

Che si traduce in:

$$f(a,b,r) = \frac{1}{2\pi} \exp -(\frac{g(a,b)}{r})^2 \tag{1}$$

con a: posizione del punto a, b posizione del punto b, g funzione di distanza fra due punti, r raggio di interesse

Il valore di attention ottenuto viene poi aggiunto a tutti i valori precedenti, che vengono salvati in una RenderTexture di formato RFloat. Il formato RFloat immagazzina nel red channel di RGB un valore di 32 bit floating point (`https://docs.unity3d.com/ScriptReference/RenderTextureFormat.RFloat.html`), che può raggiungere un valore massimo di $3.4028235 * 10^{38}$. Per questo motivo l'attention texture è visualizzata in un colore rosso di diversa intensità.

4.2.6 Sphere Collider

Per fare in modo che la mappa sia continua fra i diversi oggetti, è stato utilizzato la funzione SphereCollider fornita da Unity (`https://docs.unity3d.com/ScriptReference/SphereCollider.html`). La funzione genera una sfera che include tutte le mesh che hanno un Mesh Collider, partendo da un centro, che nel nostro caso è il punto di contatto tra il Raycast e l'oggetto, e un raggio, che sempre nel nostro caso è il raggio di interesse come indicato nella funzione 1. In questo modo tutti gli oggetti all'interno del raggio attiveranno lo shader per la generazione dell'attention map, sempre in relazione alla distanza del punto di contatto, che però riguarderà un altro oggetto nelle vicinanze.

4.2.7 Shadow mapping

Lo shadow mapping è un processo che consente di aggiungere ombre alla grafica computerizzata 3D. Le ombre vengono create verificando se un pixel è visibile dalla sorgente luminosa, confrontando il pixel con uno z-buffer o depth image della vista della sorgente luminosa, memorizzata sotto forma di texture. Se si guarda attraverso una fonte di luce, tutti gli oggetti che si vedono appaiono illuminati. Tutto ciò che si trova dietro questi oggetti, invece, sarebbe in ombra.
Questo è il principio di base utilizzato per creare una shadow map. La vista della luce viene renderizzata, memorizzando la depth di ogni superficie che vede (la depth map). Successivamente, la scena regolare viene renderizzata confrontando la profondità di ogni punto disegnato con questa mappa di profondità(`https://en.wikipedia.org/wiki/Shadow_mapping`).

4.2.8 Shadow masking

Nel caso dell'attention map questa tecnica viene applicata per non considerare le facce nascoste rispetto alla visuale. Anziché da una fonte di luce, si usa direttamente la vista della camera principale e si applica un mascheramento direttamente nello shader.
Per prima cosa si ottengono nel vertex shader le Normalised Device Coordinates(NDC), ovvero la posizione sullo schermo in cui (0,0) rappresenta l'angolo in basso a sinistra e (1,1) quello in alto a destra, indipendentemente dal display del dispositivo. La componente z delle NDC indica invece la depth in quel determinato punto (x,y). La componente w serve a compensare la distorsione prospettica, causata dalla Projection Matrix. La funzione `Linear01Depth`, presente nella libreria "UnityCG.cginc" di Unity, trasforma la componente z in forma lineare in un range tra 0 e 1(`https://docs.unity3d.com/2021.3/Documentation/Manual/SL-BuiltinMacros.html`). Si ottiene così la distanza effettiva

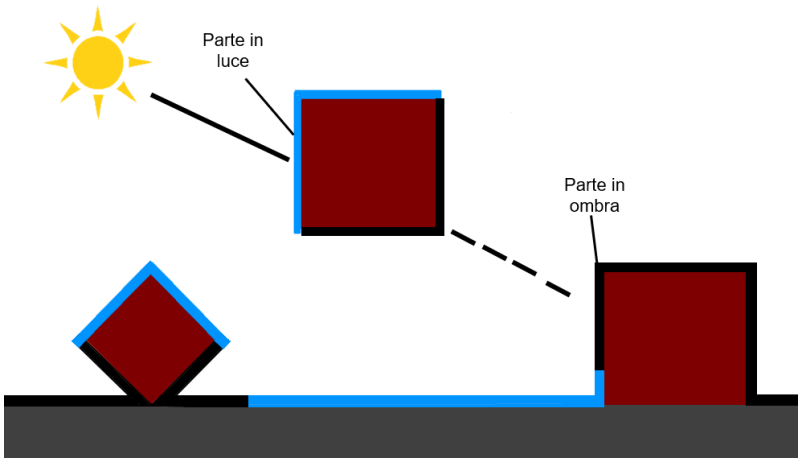


Figura 12: Schema dello shadow mapping

della telecamera e la posizione del punto. La distanza del fragment viene calcolata prendendo la coordinata z nel View Space. Questa coordinata indica la distanza della perpendicolare dalla posizione del fragment al piano ortogonale della camera.

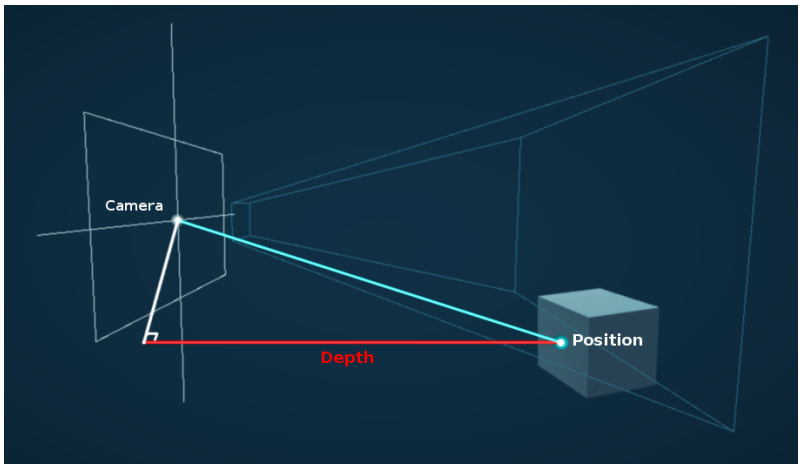


Figura 13: Depth effettivo del fragment, cioè la distanza tra il piano ortogonale alla camera e la posizione del fragment

Il risultato viene reso positivo con il suo opposto, ponendo un meno davanti, visto che in View Space la camera è diretta verso l'asse z negativo. Viene successivamente diviso per la distanza del Far Plane dalla telecamera.

I Near e Far Plane, chiamati anche Piani di Clipping, fanno parte del Volume di Vista(View Frustrum). Essi indicano il piano più vicino e più lontano visualizzati dalla camera, tutto quello al di fuori del volume formato dai due piani non verrà renderizzato. Dividendo per la distanza del Far Plane, si ottiene il valore 0 in coincidenza della telecamera e 1 in coincidenza del Far Plane.
Infine si confrontano la distanza effettiva del punto e la depth del fragment visto dalla telecamera. Se la prima variabile è maggiore della seconda, significa che il punto è in una posizione più lontana da quella più vicina alla telecamera, e quindi in "ombra". Altrimenti, il punto è in vista alla telecamera e il contributo gaussiano viene aggiunto al risultato.

(a) Oggetto con shadow masking applicato (b) Oggetto senza shadow masking applicato

Figura 14: Confronto con e senza shadow mapping, l'utente osservava il sarcofago di fronte

Nella figura 14, che mostra il retro del sarcofago, in uno scenario dove l'utente osserva le due figure in posizione frontale, si può osservare che nell'immagine a sinistra, con il shadow mapping applicato, non viene preso in considerazione il retro, al contrario rispetto all'immagine di destra, dove lo shadow mapping è applicato.

4.3 Normalizzazione

Il Ridimensionamento, anche noto come Scalatura min-max o Normalizzazione min-max, consiste nel ridimensionare l'intervallo di caratteristiche per l'intervallo in [0, 1]. La formula generale per un intervallo min-max di [0, 1] è la seguente:

$$x^{'} = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{2}$$

dove x è un valore originale, $x^{'}$ è il valore normalizzato(`https://en.wikipedia.org/wiki/Feature_scaling`). Visto che il valore minimo dell'attention map è sempre 0, la formula diventa:

$$x^{'} = \frac{x}{\max(x)} \tag{3}$$

Ogni valore viene quindi diviso per il massimo globale.

```
funzione di riduzione(bigTexture, shader di riduzione):

        dimensione texture = dimensione bigTexture >> 1;
        if(dimensione == 0) ritorna Texture iniziale;
        genera smallTexture con la nuova dimensione di texture;
        applica lo shader e stampa il risultato sulla smallTexture;
        ritorna funzione di riduzione(smallTexture, shader di riduzione);
```

In questa funzione ricorsiva, che riceve come input l'attention texture, la texture viene ridotta alla dimensione di un singolo pixel. Ad ogni iterazione la dimensione della texture viene ridotta di una potenza di 2 (`>>1` indica l'operatore shift a destra, che equivale a dividere per 2), se la dimensione della `bigTexture` è uguale a uno, la ricorsione finisce, altrimenti la funzione prosegue e si passa all'iterazione successiva. Nell'ultimo passaggio, applicando lo shader, all'interno del pixel si avrà il valore massimo globale dell'attention texture. All'interno dello shader avviene la riduzione della texture ad un quarto della dimensione originale e viene ritornato il massimo. La funzione principale, contiene l'informazione della dimensione dei texel della texture in questione, in particolare viene usato il reciproco della larghezza della texture, con la variabile `{TextureName}_TexelSize`, fornita da Unity (`https://docs.unity3d.com/Manual/SL-PropertiesInPrograms.html`).

In generale, il texel può essere definito come l'elemento unitario di una texture. Grazie a questo valore, moltiplicato per 0.5, vengono campionati i valori di quattro texel vicini tra loro e viene restituito il massimo fra questi. Infine, il valore massimo può essere usato in un altro shader, dove i valori della mappa, che sia attention o heatmap, vengono normalizzati, secondo la funzione 3.

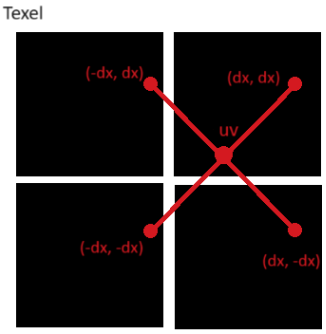
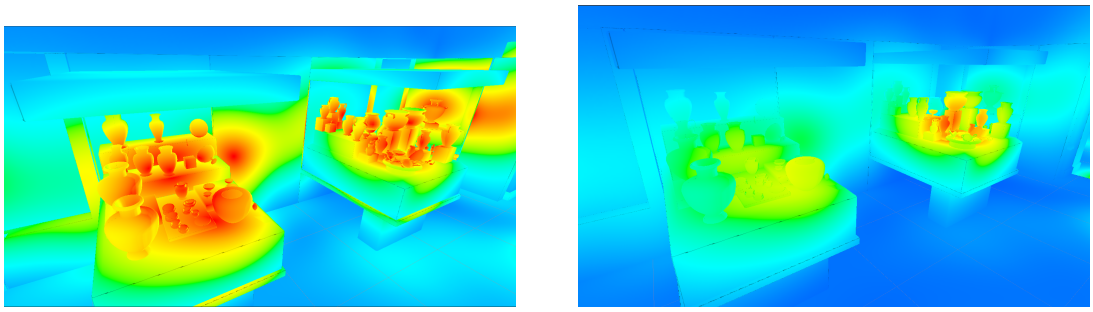


Figura 15: Campionamento di quattro texel data una coordinata uv

4.4 Normalizzazione globale

Dato che la normalizzazione avviene per ogni singolo oggetto, è necessario che venga normalizzata tutta la scena, cioè che la normalizzazione avvenga tra tutti gli oggetti presenti in essa. In questo modo l'heatmap risulta più omogenea e le informazioni sono più coerenti fra loro. Per trovare il massimo globale si estrae il massimo di ogni oggetto, si confronta con i valori massimi degli altri oggetti e si trova il massimo globale. A questo punto si divide ogni valore per il massimo globale.
Nella figura 16, a sinistra la normalizzazione è stata applicata localmente per ogni oggetto, a destra è invece stato applicato a tutti gli oggetti il massimo globale della scena. Con la normalizzazione globale si può notare che la teca a destra è stata osservata per più tempo rispetto che a quella a sinistra, mentre con la normalizzazione locale si ha solamente un'idea di dove gli utenti abbiano guardato.



(a) Normalizzazione standard (b) Normalizzazione globale

Figura 16: A sinistra si può notare la normalizzazione locale, a destra la nomalizzazione globale

Per evitare che dei valori troppo elevati di massimo locale abbiano un peso troppo eccessivo nel calcolo del massimo globale, questi ultimi vengono filtrati. Si calcola la media di tutti i valori locali degli oggetti e la deviazione standard (o scarto quadratico medio) rispetto alla media calcolata. La deviazione standard è una misura di dispersione statistica che quantifica la variabilità o la propagazione di un insieme di dati: più i dati si discostano dalla media più questo valore è alto. Fra i valori di massimo locale quelli che si discostano troppo dalla media (nel limite superiore) non possono venire assegnati come massimo globale.
Il limite che si è scelto è quello di tre deviazioni standard, seguendo la regola empirica, che afferma la banda compresa fra sei deviazioni standard di una distribuzione normale contiene il 99,7% dei valori. Da notare però che i valori massimi localmente non rappresentano una distribuzione normale, si è comunque scelto questo approccio dato che il risultato è accettabile, si è notato infatti che per valori che non si discostavano troppo dalla media questi rientravano comunque

nell'intervallo. Da notare anche che i valori massimi locali che superano il valore massimo vengono comunque normalizzati per il massimo filtrato, risultando in un valore maggiore di 1. Questo fatto non ha importanza dato che nello shader che si occupa della normalizzazione tutti i valori vengono limitati a 1, per essere poi convertiti nel colore rosso (valore massimo) con la generazione dell'heatmap, che verrà esposta nella prossima sezione. Le zone che avranno i massimi che superano il massimo globale avranno delle aree di rosso più pronunciate.
L'utilizzo di questo metodo è consigliabile solamente se si notano zone dove la concentrazione dell'attenzione o della posizione è eccessiva, vale a dire che è presente un punto con il valore massimo rosso, mentre tutti gli altri punti hanno solamente valori freddi dell'heatmap. Se non è possibile rimuovere questi valori allora il metodo aiuta ad avere una visione globale più esaustiva della scena.

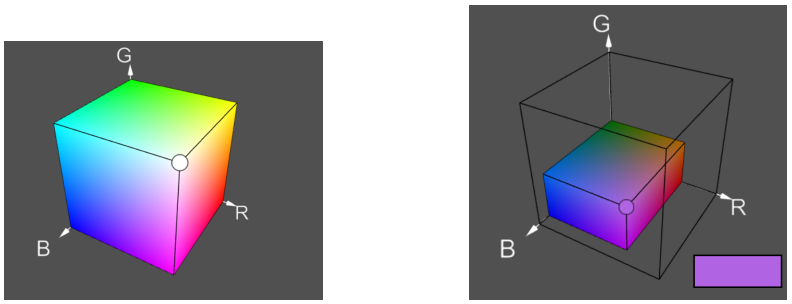
4.5 Heatmap

4.5.1 Input dell'attention

A questo punto i valori dell'attention, o per meglio dire, le texture con immagazzinati i valori dell'attention, possono essere utilizzate come input per generare le heatmap. Bisogna precisare che le heatmap non aggiungono informazioni, ma convertono i dati già ottenuti per avere una visualizzazione più chiara visivamente per andare poi ad analizzare i dati. Le heatmap possono essere generate sia in real-time, sempre all'interno del double buffering per l'attention map, allocando due buffer specifici per le heatmap, sia una volta che l'attention map finale è stata generata, in questo caso l'attention map viene manipolata attraverso uno shader.

4.5.2 Conversione dei colori

La conversione dei colori avviene valutando il valore dell'attention come se fosse il parametro 'hue' (tonalità) del metodo additivo di composizione dei colori HSV (Tonalità, Saturazione, Valore). Ricordiamo che l'attention map è gestita come un parametro float, salvato nel canale rosso dell'RGB (Red-Green-Blue), e quindi di base visualizzata come zone nere e zone rosse di diversa intensità. Il modello di colore HSV condivide la maggior parte delle sue proprietà con HSL(Hue, Saturation, Lightness). Questi tre modelli possono venire rappresentati visivamente con figure geometriche tridimensionali, dove ogni punto all'interno del volume rappresenta una combinazione dei tre valori.



(a) Composizione RGB, rappresentata come un cubo

(b) Le coordinate RGB in questo caso danno il colore viola

Figura 17: Si può vedere che, modificando le coordinate tridimensionali, si può determinare il colore dato dal vertice

Invece di trasformare il cubo RGB in una forma biconica, come fa HSL, HSV ha la forma di un singolo cono invertito (`https://web.cs.uni-paderborn.de/cgvb/colormaster/web/color-systems/hsv.html`).

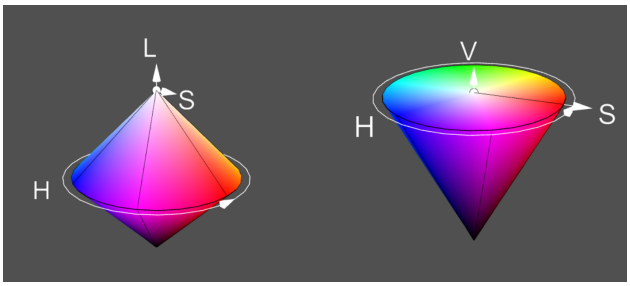


Figura 18: Rappresentazione di HLS(destra) e HSV(sinistra)

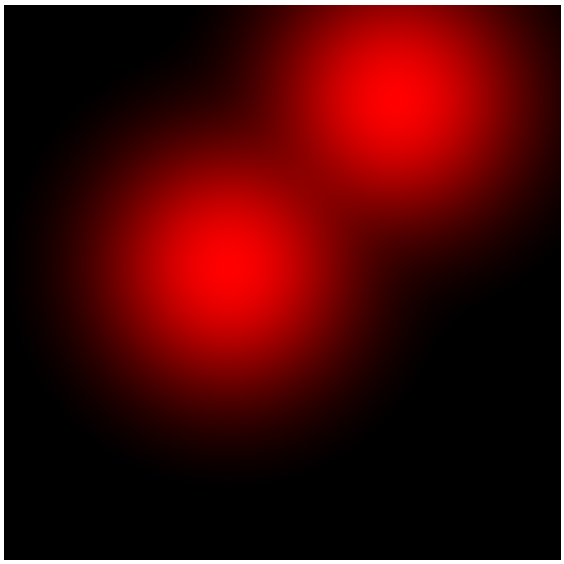
Si è scelto di lavorare con HSV perché si è ritenuto che la conversione della tonalità sia più intuitiva a livello visivo, rispetto a HLS. Il valore dell'hue è rappresentato come i gradi della base del cono, rappresentando la tonalità, che parte dal rosso al grado 0, a 120 gradi si raggiunge il verde, a 240 il blu e così via. Dato che il valore dell'attention viene normalizzato per il massimo, si può convertire da valori compresi tra 0 e 1 ai gradi di un angolo giro, da 0 a 360 gradi. Per prima cosa, il valore viene ribaltato, in modo che i valori minimi corrispondano al blu, poi viene limitato a due terzi del valore originale, in modo che il valore massimo sia il rosso e non il viola. In questo modo si ha la rappresentazione di uso comune dell'heatmap. Avviene quindi la conversione da HSV a RGB, mantenendo le coordinate di saturazione e valore massime, usando la formula(con H tonalità, S saturazione, V valore):

$$f(n) = V - VSmax(0, min(k, 4-k, 1) \tag{4}$$

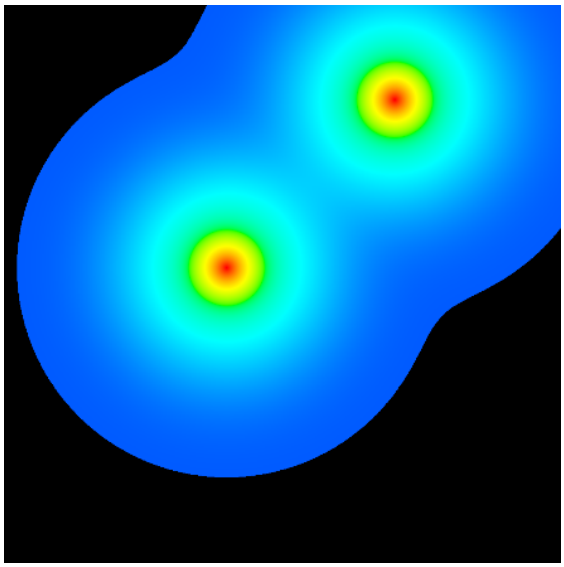
$$\text{dove k, n}\in \mathbb{R}_{\geq 0},\ k = (n + \tfrac{H}{6})\text{mod}6$$

$$(R, G, B) = (f(5), f(3), f(1)) \tag{5}$$

Grazie a questo metodo, la transizione di colori per diversi valori di interesse avviene in maniera continua e uniforme.



(a) Texture dell'attention map



(b) Texture dell'heatmap

Figura 19: Conversione da attention map a heatmap

4.6 Heatmap 2D

La visualizzazione del percorso compiuto da uno o più utenti è realizzata con un'heatmap bidimensionale, partendo dall'attention map presente nel capitolo 4.2. Dato che il pavimento può considerarsi un piano, è stato possibile attuare delle modifiche e delle semplificazioni. Per prima cosa l'input dello shader in questo caso è la posizione della telecamera, o più precisamente, il punto di contatto tra la verticale della telecamera e il piano d'appoggio. Partendo dalla posizione della telecamera, viene generato un versore che ha direzione verso il basso in coordinate del mondo. Un filtro viene applicato alle mesh che corrispondono al pavimento in modo che siano le uniche che vengono prese in considerazione. Il punto di contatto viene dato allo shader che calcola la distanza gaussiana di tutti i punti(o fragment) del pavimento rispetto al punto. Lo shader è stato semplificato in modo che si evitino i calcoli per lo shadow mapping, dato che si tratta di un piano del quale la faccia posteriore non è d'interesse e che teoricamente non nasconde ulteriori poligoni. Un'altra semplificazione possibile, che si sarebbe potuta applicare, è quella di usare le coordinare della texture nel punto di contatto invece che il punto nel mondo, in modo da non dover calcolare le coordinate globali all'interno dello shader. Le coordinate della texture vengono date direttamente da Unity con la struttura RaycastHit, per venire poi confrontate con le coordinate uv della texture del pavimento dello shader. Tuttavia, si è deciso di non adottare questa tecnica per coerenza con l'attention map.
Per quanto riguarda la culling matrix della SubCamera, come viene spiegato nella sezione 4.2.4, questa deve replicare la camera come se fosse davanti a quell'oggetto in quell'istante. Nel caso dell'heatmap 2D la view matrix prende in input il punto di contatto con il pavimento, questo punto viene alzato in verticale, lungo l'asse Y, di un'unità, e viene direzionato verso il basso.

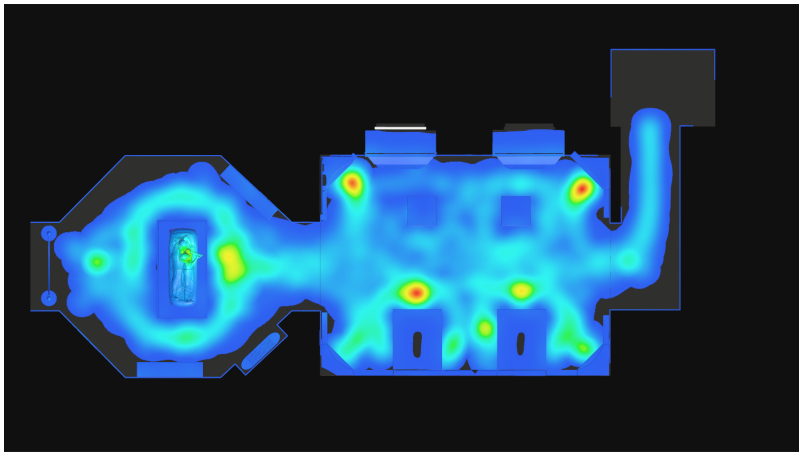


Figura 20: Visuale della pianta del museo con l'heatmap applicata

4.7 Distribuzione

Nella curva gaussiana standard, la funzione base che è stata utilizzata è

$$\exp -9x^2/\sqrt{2\pi}$$

in modo che il la maggior parte dei valori sia compreso fra i valori $(-1, 1)$, aggiungendo una variabile raggio k, impostabile dall'utente, la funzione diventa

$$\exp -9(x/k)^2/\sqrt{2\pi} \tag{6}$$

in questo modo la maggior parte dei valori è contenuta nell'intervallo $(-k, k)$. Oltre al raggio, l'utente può determinare la distribuzione della curva, vale a dire cambiare l'intensità dei valori massimi e di quelli minimi dell'heatmap. Per fare ciò viene usato uno shader apposito, che prende in input la curva gaussiana e la eleva a potenza. Come si può vedere nella figura 28, elevando ad una potenza maggiore di 1 la curva si restringe e diventa più ripida, così che i punti si concentrino su valori più alti (curva nera). Al contrario, elevando ad una potenza compresa fra 0 e 1, la curva si espande ma diventa anche più piatta. In questo caso i punti sono più concentrati sui valori bassi (curva verde).

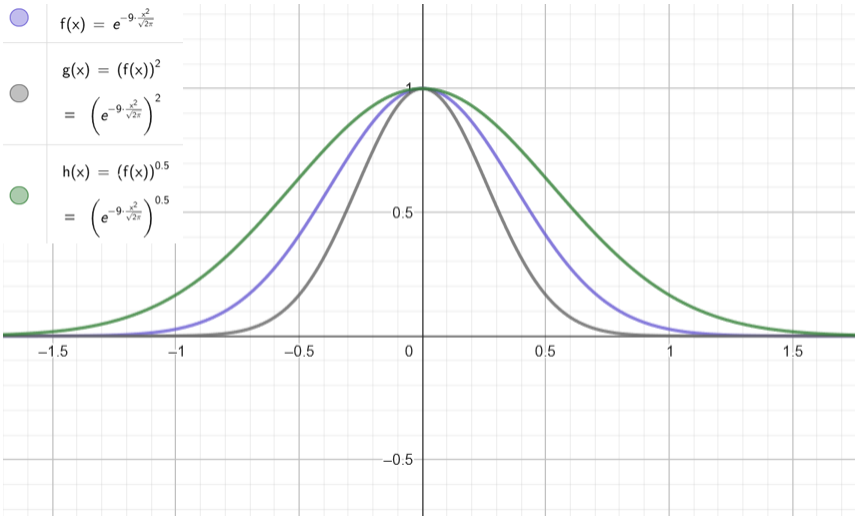
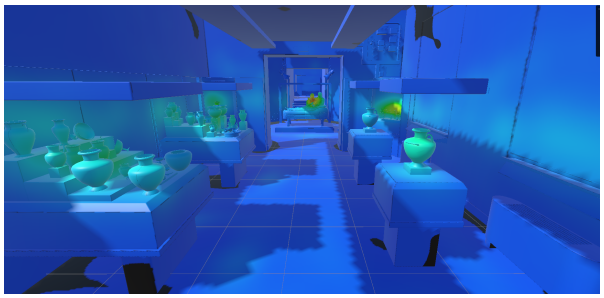
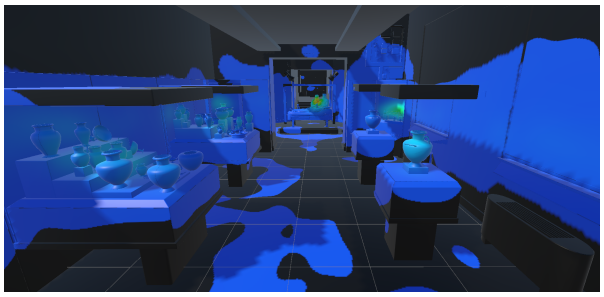


Figura 21: Andamento della gaussiana con il cambio di distribuzione

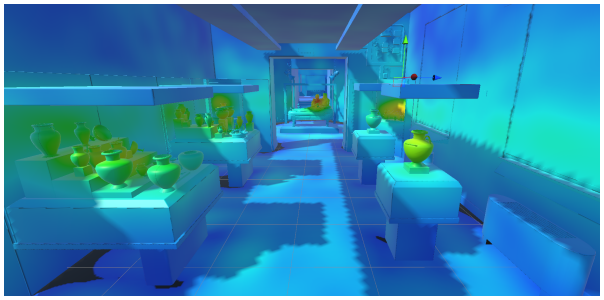
Come risultato, l'heatmap con valori maggiori di 1 l'effetto generale è una diminuizione dei colori freddi, con un divario fra i colori più alto. Al contrario, con una distribuzione compresa fra 0 e 1, si ha un aumento dei colori e il divario fra colori caldi e freddi diminuisce. Questo risultato si può osservare nella figura 22.



(a) La distribuzione è a 1



(b) La distribuzione è a 2, la mappa è pià concentrata sui valori alti



(c) La distribuzione è a 0.5, la mappa è concentrata sui valori bassi

Figura 22: Risultato nell'ambiente con tre diverse distribuzioni, la mappa è più concentrata sui valori bassi

4.8 Fase realtime e fase postprocessing

La fase di realtime è stata necessaria per testare la correttezza dell'heatmap. In questa fase, prevalentemente di testing, viene controllato il livello di crescita delle zone calde dell'heatmap, usando una first person camera in movimento. In questo modo viene monitorata in particolare la velocità in cui i valori si avvicinano al rosso, in modo da non avere né una crescita troppo ripida né una troppo lenta. Questa fase è resa possibile in real time grazie alla GPU che si occupa di disegnare le mappe sui modelli. Il trasferimento di dati da GPU a CPU causerebbe dei ritardi di attesa fra le due componenti che non permetterebbe di avere risultati soddisfacenti in tempo reale.
Nel caso della fase di post processing, l'heatmap viene prima calcolata e infine visualizzata sulle texture. In questo caso la quantità di dati da caricare è maggiore, dato che possono essere calcolate in simultanea tutte le texture degli oggetti presenti in scena, per un dato periodo di tempo e non solo istantaneamente. Oltre a questo viene calcolato il massimo globale dei valori dell'attention map, che come algoritmo ha complessità $\mathcal{O}(n)$ (con n il numero di oggetti presenti in scena) e che subisce un passaggio di dati da GPU a CPU e viceversa. Per calcolare la complessità di calcolo dell'algoritmo in totale, bisogna moltiplicare il numero di oggetti presenti nella scena, per il numero di istanti di tempo presi in campione. In più viene aggiunta una funzione che è una variabile che dipende dal raggio di interesse nell'heatmap e la vicinanza degli oggetti fra di loro. Tutti questi fattori precedentemente elencati hanno anch'essi complessità $\mathcal{O}(n)$. Infine bisogna aggiungere la ricerca del massimo globlale.

$$\begin{aligned}\text{complessità} = \text{numero di oggetti} * \text{istanti campionati} + g(\text{raggio, oggetti vicini}) \\ + \text{ricerca massimo globale} = \mathcal{O}(n)\end{aligned} \tag{7}$$

dove g è in funzione del raggio e degli oggetti vicini presenti in scena

Nella fase in tempo reale, una scena con una concentrazione di oggetti troppo alta potrebbe avere un impatto significativo sulle performance.
Per il post processing, la mole di dati da elaborare può portare a tempi di attesa molto lunghi. Per questo motivo si è infatti deciso di non considerare gli oggetti all'interno delle bacheche, ma "oscurare" i vetri e proiettare l'heatmap direttamente su di essi. In questo modo si è notato che il tempo di generazione delle heatmap si ridotto considerevolmente. Il passaggio ad HDRP aumenta invece la complessità ulteriormente, data la natura stessa della pipeline, che rispetto all'URP richiede risorse più ingenti.

4.9 Classificazione Dati

Come già esposto nella sezione 4.2.2, i dati in input sono salvati in un file .csv. Ogni file rappresenta la sessione di un utente. Dato che all'interno di ogni file è salvato anche lo user id, è possibile filtrare i dati a seconda dell'utente. I dati finali verranno poi salvati in una lista divisa per ogni utente, che poi verrà processata per generare le heatmap. Oltre agli utenti, è possibile filtrare per il tipo di scena. Ci sono infatti tre varianti in totale del museo: quella base, nominata MAIN, che riproduce fedelmente il museo (come si può vedere dalla figura 18), la VARIANTE 1, dove la sala del sarcofago è separata da un muro, la VARIANTE 2 dove la sala del sarcofago è stata decorata con un ambiente stellato (si possono osservare queste ultime due nelle figure 35). Per quanto riguarda la suddivisione in intervalli temporali, come verrà esposto nella sezione 4.11, è risultato conveniente ordinare i dati in modo crescente rispetto al tempo per poi essere processati. Nel caso si abbia intenzione di dividere i vari intervalli per utente, è necessario fare una distinzione preliminare per l'identificativo di ogni utente, salvare i dati in una lista divisa per quest'ultimo, ordinare i dati per il tempo e infine processarli.

UserId	Time	position-X	position-Y	position-Z	rotation-X	rotation-Y	rotation-Z	rotation-W
1	212,4	6,05	1,51	0,18	-0,06	-0,69	-0,06	-0.99
1	212,5	6,04	1,51	0,182	-0,06	-0,70	-0,070	-0.99
1	212,6	6,046	1,51	0,178	-0,07	-0,70	-0,074	-0.99
1	212,6	6,046	1,51	0,17	-0,07	-0,70	-0,07	-0.99

Tabella 1: Esempio del formato di dati in ingresso

Come si può vedere nella tabella 1, il formato dei dati è suddiviso in: id utente, istante di campionamento, coordinate cartesiane della posizione dell'utente in quell'istante e l'orientamento. Il valore W nella rotazione rappresenta la quarta dimensione del quaternione. Un quaternione è una quadrupla di numeri reali {x, y, z, w}. È un'alternativa matematicamente conveniente alla rappresentazione dell'angolo di Eulero e viene usato internamente da Unity per rappresentare tutte le rotazioni.
Da notare che le coordinate fanno riferimento alla Camera virtuale presente in Unity, che si traduce nella posizione del visore all'interno della scena.

4.10 Caching

La cache è una memoria che memorizza dati temporanei per un veloce riuso. In generale la funzione della memoria cache è di velocizzare gli accessi alla memoria principale aumentando le prestazioni del sistema.
Nel nostro caso è stata sviluppata una cache apposita, collocata nel disco, per salvare le texture di ogni oggetto, per poi essere caricate in modo veloce nella memoria principale. In questo modo si diminuisce il carico sulla memoria principale con l'immagazzinamento di un numero di texture variabile. Per ogni oggetto vengono salvate due texture per il double buffering (sezione 4.2.3), una texture per salvare la mappa normalizzata (sezione 4.3), una texture per salvare l'heatmap (sezione 4.5) e una texture per salvare la distribuzione della mappa (sezione 4.7).
La creazione di una cache permette di dover inizializzare le texture solamente la prima volta e poter avere a disposizione immediatamente queste una volta generate. Da considerare però che, di conseguenza, verrà allocata della memoria su disco, anche ingente, data la natura delle texture e dei loro dati.
Per questo è stata aggiunta la possibilità di scegliere la risoluzione delle texture, in base alle proprie necessità. Per un ulteriore risparmio di memoria, la risoluzione delle texture dipende dalla dimensione di ogni oggetto: un oggetto più piccolo avrà una risoluzione minore rispetto ad un oggetto più grande, nell'ordine della potenza di 2. Per esempio, se la texture di una parete ha una risoluzione 1024x1024 pixel, la risoluzione del Sarcofago degli Sposi avrà una risoluzione 512x512p, mentre la risoluzione di un vaso sarà di 256x256p. Dato che in questo modo non cambiano i PPI (Pixels Per Inch), il risultato visivo finale rimane più o meno invariato.

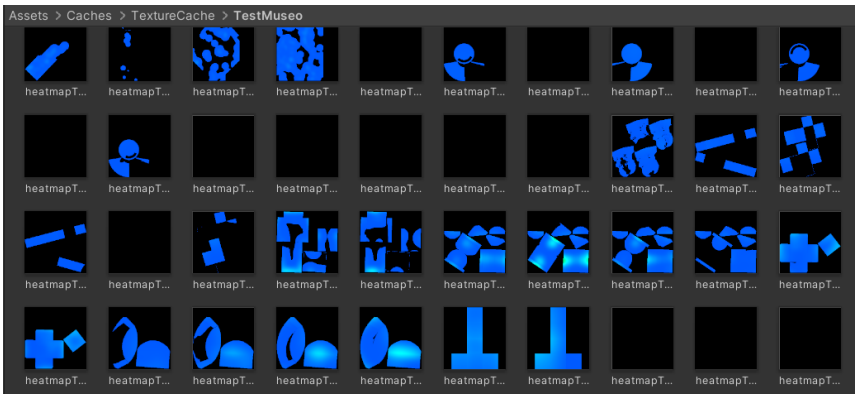


Figura 23: Mappe UV delle heatmap salvate come texture su disco

4.11 Intervalli temporali

Grazie all'informazione temporale, è possibile filtrare i dati in input (sezione 4.2.2), secondo un intervallo fra due valori, compresi fra 0 secondi e il tempo massimo registrato. Per ottenere questo c'è bisogno di ordinare in maniera crescente rispetto al tempo tutti i valori, dato che i valori di input inizialmente sono ordinati oltre che per il tempo, anche per l'id dell'utente. Successivamente si escludono tutti i valori non compresi nell'intervallo di tempo considerato e si processano solo quelli compresi.
Viene poi implementato all'interno di Unity uno slider che permette di scegliere i valori di massimo e di minimo. Questi due valori non possono superare il massimo valore registrato fra tutti gli utenti. Se viene selezionato un intervallo di tempo che supera l'intera durata della registrazione di un utente, questo non verrà automaticamente considerato.

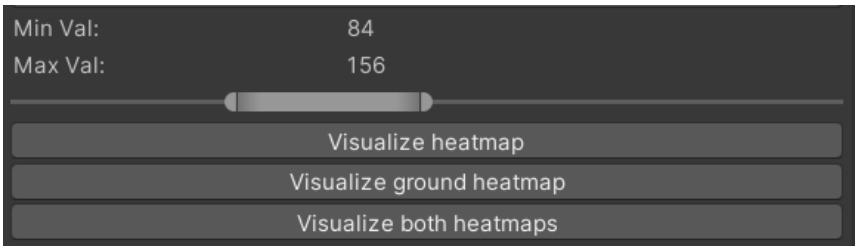


Figura 24: Esempio di intervallo selezionato all'interno dell'ambiente Unity

L'implementazione di questo metodo permette di poter leggere un file con una lista di intervalli temporali divisi per utente, funzione che verrà sfruttata nella sezione 5.4.

1	5	7	8	11	12
(47, 51)	(91, 95)	(5, 9)	(24, 28)	(35, 39)	(6, 10)
(143, 147)	(119, 123)	(55, 59)	(94, 98)	(79, 83)	(8, 12)
NaN	(217, 221)	(96, 100)	(171, 175)	(141, 145)	(61, 65)
NaN	NaN	(119, 123)	(190, 194)	(209, 213)	(123, 127)

Tabella 2: Esempio del formato del file di intervalli temporali, divisi per un campione di utenti. Per rendere coerente la tabella i valori non considerati sono indicati con NaN (Not a Number) per riempire le colonne di lunghezza minore. Questi valori non vengono considerati nell'analisi finale.

Per ogni intervallo (in secondi) vengono letti i valori e salvati in un buffer, che viene incrementato per ogni intervallo preso in considerazione. Quando gli

intervalli di un utente sono terminati, si passa all'utente successivo e vengono caricati i suoi intervalli, fino alla fine del file. Una volta accumulati tutti i dati, questi vengono normalizzati e generata l'heatmap corrispondente. Il formato dell'intervallo immesso è (*inizio intervallo*, *fine intervallo*). Gli intervalli non considerati vengono segnati come *NaN* (Not a Number). Gli istanti temporali devono comunque fare riferimento ai dati in input (sezione 4.9).
Successivamente gli intervalli temporali verranno utilizzati per l'identificazione dei picchi emozionali nella scena, come spiegato nella sezione 5.4.1.

4.12 Inizializzazione della scena

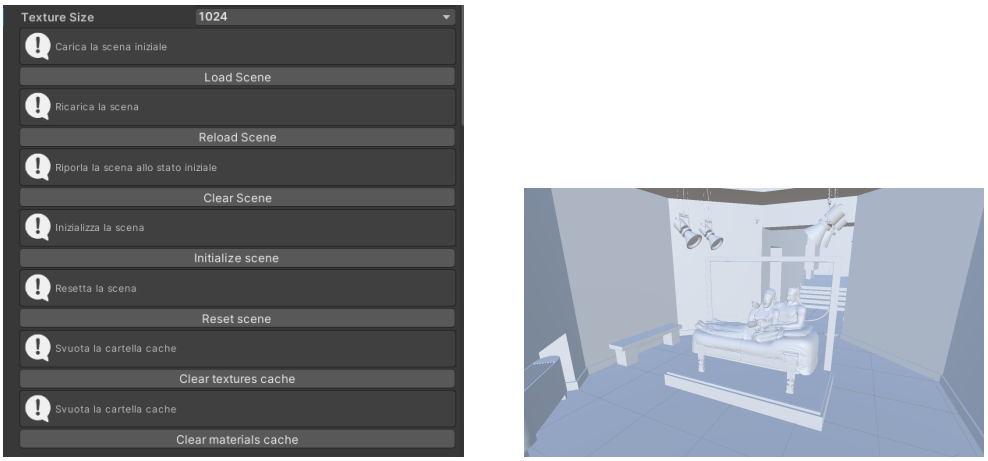
Per visualizzare le heatmap, bisogna rendere l'ambiente in Unity favorevole alla loro generazione. In questo caso per ogni oggetto viene creato un materiale diverso, e dato che alcuni oggetti possono avere più materiali assegnati ad esso, ne viene forzato ad uno solo. È necessario poi distinguere gli oggetti che non devono essere considerati, il pavimento, dove verrà applicata l'heatmap 2D, e gli oggetti tridimensionali. Nel nostro caso si assegnano manualmente agli oggetti dei layer. In Unity i layer sono uno strumento che consente di separare gli oggetti nelle scene. I layer vengono modificati tramite l'interfaccia utente e script per modificare il modo in cui gli oggetti interagiscono fra di loro. In particolare il Raycast (sezione 4.2) di Unity può filtrare gli oggetti a seconda del loro layer. I valori del pavimento devono essere manualmente assegnati, dato che un algoritmo di identificazione delle superfici calpestabili sarebbe oneroso e inaccurato, rispetto che assegnare le superfici a mano. Gli oggetti trasparenti invece non vengono automaticamente considerati. Successivamente per ogni oggetto con il layer corretto vengono inizializzate le texture utili per l'heatmap e salvate in una particolare cache come spiegato nella sezione 4.10. In particolare ci sono 3 layer a cui l'algoritmo fa riferimento:

- Heatmap: assegnato automaticamente, calcola l'attenzione per tutti gli oggetti con quel layer
- GroundHeatmap: assegnato manualmente, calcola la posizione per tutte le superfici assegnate
- NoLoad: layer opzionale, utile se si vuole evitare che un particolare oggetto nella scena venga processato

Viene inizializzata la SubCamera e per ogni oggetto viene istanziato un suo duplicato invisibile che contiene le informazioni della mesh, chiamato ShadowPrefab (sezione 4.2.4).
In questo modo, anche importando la scena da ambienti esterni è possibile averla pronta all'uso.
Nel caso di modifiche irreversibili involontarie o indesiderate, è possibile far ritornare la scena allo stato originale. Vengono ridisegnate tutte le texture al valore di default e vengono liberate dalla memoria tutte quelle allocate. Per ogni oggetto presente in scena viene riassegnato il materiale originale e vengono distrutti tutti gli oggetti ShadowPrefab.

4.13 Workflow per la generazione delle heatmap

Inizialmente si importano le mesh interessate nella scena, nel nostro caso il modello del museo. Si seleziona la dimensione delle texture, che può essere originale, se ci sono delle texture già presenti sui modelli, oppure con una risoluzione variabile fino a 2048 pixel. Vengono così salvati i materiali nella cache (sezione 4.10) ed assegnati all'oggetto corrispondente. Per quanto riguarda la Universal Render Pipeline, i materiali inizialmente non presentano nessuna texture applicata, la scena si presenta quindi con il colore di default di Unity (bianco - figura 25b). Vengono anche salvate le texture nella cache che serviranno da buffer per salvare i dati, oltre a quelle finali con sopra l'heatmap.
Ad ogni oggetto nella scena viene associato un oggetto figlio con le informazioni della mesh (ShadowPrefab) e i componenti necessari per procedere. Cambiando il layer (sezione 4.12) per il pavimento verranno instanziati l'oggetto figlio e i componenti specifici per le heatmap in 2D (sezione 4.6).
In caso di necessità, è possibile riazzerare la scena, questo comporterà però in ogni caso la perdita delle texture originali. È possibile svuotare le cache manualmente, in questo caso sarà necessario rigenerarle quando si ricarica la scena. Inizializzando la scena verranno caricate le texture della cache, se presenti; altrimenti verranno nuovamente generate e salvate. Se, una volta visualizzate le heatmap, si vuole partire con delle nuove mappe, è necessario eseguire un reset della scena, in modo da liberare le texture e svuotare i buffer.



(a) Controllo in Unity (b) Stanza del sarcofago inizializzata

Figura 25: Inizializzazione della scena

Con la scena pronta, si passa al caricamento dei dati di input. Questo passaggio preliminare permette di selezionare i dati e filtrarli secondo due metodi: per l'ID dell'utente o per il tipo di scena. Per quanto riguarda gli ID utente, si possono

selezionare utenti singoli o intervalli di utenti; per la scena, nel nostro caso, si possono selezionare le tre tipologie: Main, Variante 1 e Variante 2. Vengono così filtrati tutti gli utenti che hanno compiuto la visita in quella variante del museo.

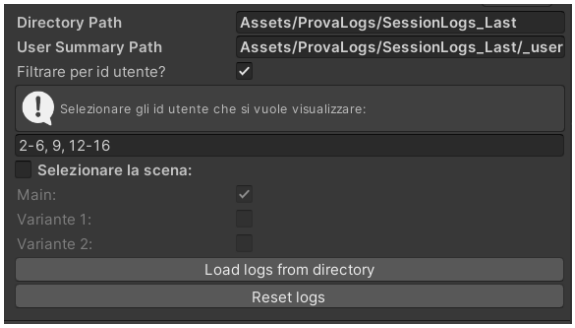
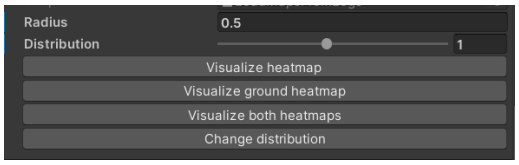
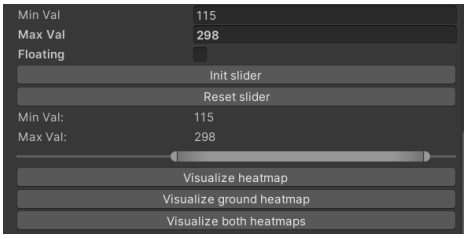


Figura 26: Controllo in Unity per caricare i dati

Con la scena e i dati predisposti, è possibile generare le heatmap. Si possono impostare il raggio d'azione dell'heatmap e la distribuzione dei valori (sezione 4.7), ma mentre è possibile modificare la distribuzione dei valori anche successivamente con un effetto immediato, per modificare il raggio è necessario ricalcolare tutte le mappe, essendo questo legato allo shader delle attention map (sezione 4.2).
L'utente può decidere se visualizzare l'heatmap per l'attenzione, l'heatmap per la posizione o entrambe le heatmap contemporaneamente. Oltre a ciò può considerare solamente un intervallo di tempo limitato (sezione 4.11) oppure tutta la durata delle visite.



(a) Controllo per la generazione delle heatmap



(b) Slider con gli intervalli di tempo

Figura 27: Generazione heatmap in Unity

A questo punto è possibile salvare le texture su disco (sezione 4.15) e caricarle in un altro momento se necessario, oppure fare il reset delle mappe.
Con il reset delle mappe è possibile ritornare allo stato iniziale e ricominciare con la generazione di nuove heatmap.

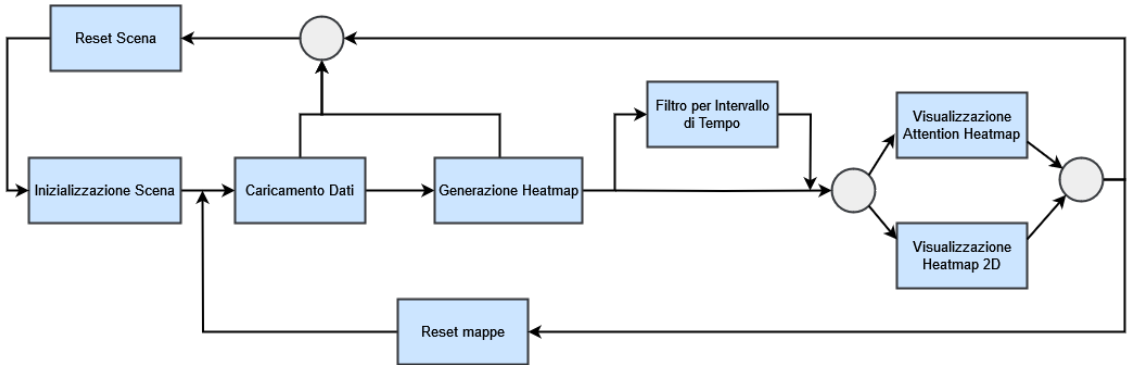


Figura 28: Controllo in Unity per caricare i dati, il passaggio iniziale è l'inizializzazione della scena, quello finale è la visualizzazione delle heatmap

4.14 Passaggio all'HDRP

La pipeline HDRP (High Definition Render Pipeline) è una nuovissima pipeline di rendering avanzata disponibile in Unity per creare contenuti ad alta definizione e con un'elevata qualità visiva. La HDRP offre una serie di tecniche avanzate di illuminazione basate sulla fisica, illuminazione lineare , illuminazione HDR; per il rendering, la gestione delle luci volumetriche, il rendering di ombre ad alta qualità e l'effetto di rimbalzo della luce. Offre gli strumenti necessari per creare applicazioni come giochi, demo tecniche e animazioni con un elevato standard grafico. La HDRP offre una maggiore efficienza di rendering ma non è supportata da tutte le piattaforme. Per questo motivo, attualmente, viene usata principalmente per applicazioni realtime dall'elevata qualità visiva supportata da un potente sistema di calcolo.
Ai cambiamenti effettuati nel caricamento della scena (sezione 4.12), si è aggiunta la texture presente di default sull'oggetto, in modo da avere facile accesso all'originale una volta che questa viene modificata. La texture viene salvata nella cache (sezione 4.10) assieme alle altre texture, e infine salvata in formato PNG con il metodo esposto nella prossima sezione, dato che Unity non permette di avere due riferimenti allo stesso asset presenti nel progetto.
Per poter vedere le texture originali insieme alle heatmap, è stato disposto uno shader con la funzione lerp. La funzione lerp prende due valori in input, in questo caso i fragment (o pixel) e compie un'interpolazione lineare secondo un terzo parametro, che sposta l'interpolazione a favore di un valore o dell'altro. Ne risulta una dissolvenza delle due texture che dipende dal terzo parametro, se il parametro è 0, la texture originale si vedrà completamente, mentre l'altra texture sarà completamente dissolta. Al crescere del valore d'interpolazione, le due texture cambiano di dissolvenza fino al valore 1 dove solo la seconda texture sarà visibile. La formula del lerp si presenta come:

$$x * (1 - s) + y * s \tag{8}$$

con x la prima texture, nel nostro caso, y la seconda, e s il valore d'interpolazione

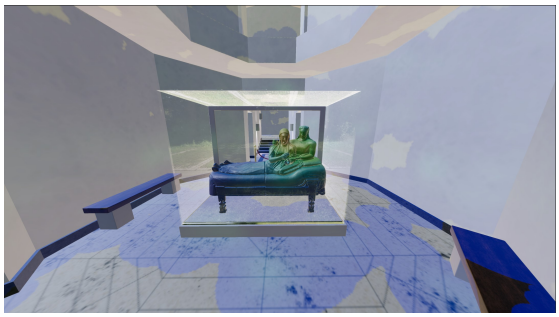
La funzione può anche essere scritta:

$$x + s * (y - x)$$

con gli stessi valori della funzione precedente

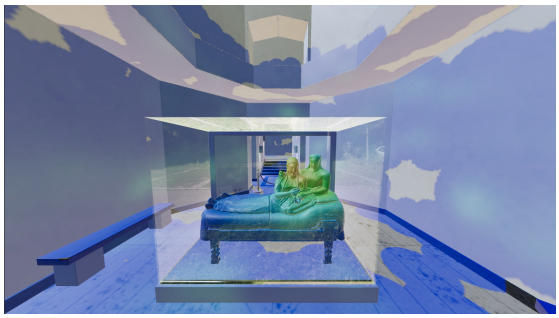
Con il valore posto a 0.5 entrambre le texture, quella originale e l'heatmap, saranno visibili allo stesso modo sulla superfice. Il passaggio di pipeline da URP all'HDRP permette di preservare le texture iniziali dell'ambiente, a scapito di una maggiore complessità e conseguente aumento di tempo nella generazione delle heatmap.

(a) Dettaglio bacheche, heatmap al 25%



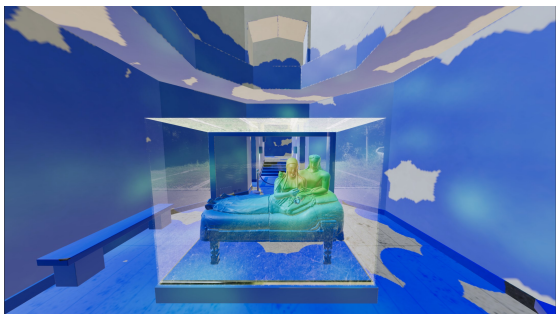
(b) Sala sarcofago, heatmap al 25%

(c) Dettaglio bacheche, heatmap al 50%



(d) Sala sarcofago, heatmap al 50%

(e) Dettaglio bacheche, heatmap al 75%

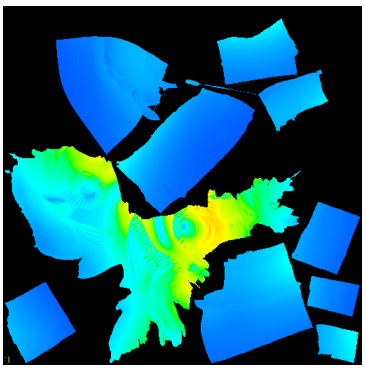


(f) Sala sarcofago, heatmap al 75%

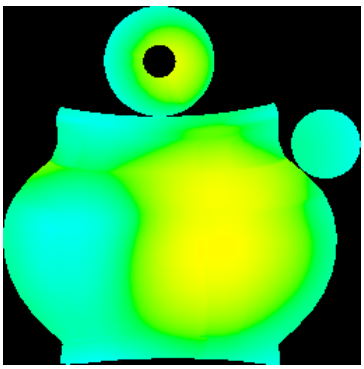
Figura 29: In questo caso sono state caricate le heatmap di due utenti, cambiando il valore di intensità della funzione lerp si può gradualmente vedere la mappa applicata sugli oggetti. In questo modo vengono preservate le texture originali.

4.15 Salvataggio delle texture

Per avere facile accesso a heatmap già generate, è stato implementato il salvataggio e caricamento delle texture sotto forma di immagini bidimensionali. In particolare viene salvata l'heatmap applicata alla mappa UV dell'oggetto. È possibile salvare le mappe in quattro formati differenti: PNG (Portable Network Graphics), JPEG (Joint Photographic Experts Group), TGA (Targa), EXR. Il formato PNG è caratterizzato da compressione lossless, senza perdita di informazioni. Il formato JPEG compie una compressione dei dati, quindi occupa meno memoria ma ha perdita di accuratezza dell'immagine. TGA è un formato di file raster utilizzato per memorizzare fotografie e immagini digitali, salva immagini non compresse, salvaguardando il risultato da quelle alterazioni che spesso si verificano con i formati compressi (`https://it.wikipedia.org/wiki/JPEG`). Lo scopo del formato EXR è quello di rappresentare in modo accurato ed efficiente immagini ad alta gamma dinamica. È ampiamente utilizzato nei software applicativi in cui la precisione è fondamentale, come rendering fotorealistico, accesso alle texture e composizione di immagini(`https://openexr.com/en/latest/`).
Per riassumere, il formato JPEG si può utilizzare solo se è necessario risparmiare memoria, i formati TGA e EXR sono molto accurati, a scapito della memoria, mentre il formato PNG è un compromesso fra quest'ultimi.



(a) Mappa UV del sarcofago



(b) Mappa UV di un vaso

Figura 30: Esempi di mappe UV salvate con l'heatmap applicata

Ogni oggetto presente in scena che ha generato un'heatmap avrà una copia salvata in memoria; le dimensioni delle texture rispecchiano quelle presenti nella scena. Caricando le texture già salvate rispetto che a doverle rigenerare ogni volta permette un risparmio in termini di tempo che può anche essere considerevole. Per il caricamento delle texture vengono selezionate le texture interessate, nel formato in cui sono salvate (PNG, JPEG, TGA, EXR), e vengono caricate come un array di byte, per poi essere convertite in texture all'interno dell'ambiente

Unity e assegnate alla mesh corrispondente. La funzione utilizzata, LoadImage dall'API di Unity (`https://docs.unity3d.com/ScriptReference/ImageConversion.LoadImage.html`), sostituisce il contenuto della texture con i dati dell'immagine corrispondente. Questa funzione può anche modificare la dimensione e il formato della texture. I file JPEG vengono caricati nel formato RGB24, i file PNG vengono caricati nel formato ARGB32 e i file EXR vengono caricati in RGBAFloat.
Liberando le texture è possibile fare un reset della scena per caricare heatmap già salvate o generarne delle nuove.

5 Sperimentazione

Per quanto riguarda l'attuazione pratica degli obbiettivi descritti in questa tesi, sono state effettuate delle sessioni con dei soggetti per testare l'applicazione dell'ambiente virtuale, con la cattura dei dati che andranno poi visualizzati tramite heatmap.
Le sessioni, della durata di quattro minuti ciascuna, hanno previsto una visita virtuale con l'ausilio del visore VR di due stanze ricreate partendo da quelle presenti al Museo Nazionale Etrusco di Villa Giulia. Ogni utente ha potuto eseguire la visita potendo camminare liberamente in una stanza, seguendo le dimensioni reali delle due stanze presenti nel museo, solamente un leggero ridimensionamento del modello 3D verso dimensioni minori è stato effettuato per garantire la sicurezza degli utenti, in modo che questi non si avvicinassero troppo ad oggetti e alle pareti della stanza una volta indossato il caschetto VR. La durata massima dell'esperimento è stata di quattro minuti, al termine dell'esperienza un indicatore sonoro ha indicato all'utente di ritornare alla posizione di partenza e terminare la visita. Ogni utente è stato comunque libero di terminare la visita prima del tempo che ha avuto a disposizione ritornando nella zona di partenza.

5.1 Partecipanti

Il totale dei partecipanti che ha preso parte all'esperimento è stato di 69 utenti. Di questi, 42 di sesso maschile, 26 di sesso femminile e 1 utente come "altro/preferisco non rispondere"; di età compresa fra i 19 e 34 anni. A tutti gli utenti è stato presentato un questionario pre e post visita, non è stato detto loro cosa sarebbe stato presentato loro, a parte la precisazione che sarebbe stata una mostra virtuale. Nel questionario antecedente all'esperienza sono state poste domande come le esperienze pregresse con la realtà virtuale o con ambienti virtuali, così come esperienze in musei e frequenza di visite in siti culturali durante l'anno. Nel questionario post-esperienza sono state domandate impressioni sulla visita, come valutare il grado di immersione e di realismo, così come eventuali sensazioni di nausea e malessere.

5.1.1 Igroup Presence Questionnaire

Queste domande post-esperienza fanno parte del questionario IPQ (Igroup Presence Questionnaire) (igroup.org – project consortium) che è una scala per misurare il senso di presenza vissuto in un ambiente virtuale (VE - Virtual Environment). Questa scala comprende i valori da 0 a 6, con 0 che indica l'esperienza in realtà virtuale come la peggiore possibile e il 6 che indica la migliore possibile [8]. Il questionario è stato costruito utilizzando un ampio pool di elementi e due sondaggi con circa 500 partecipanti. [12]

L'attuale versione dell'IPQ prevede tre sottoinsiemi e un item generale aggiuntivo che non appartiene a nessun sottoinsieme. I tre sottoinsiemi sono emersi dall'analisi delle componenti principali e possono essere considerati fattori abbastanza indipendenti. Essi sono:

- Presenza spaziale: la sensazione di essere fisicamente presenti nel VE
- Coinvolgimento: misurare l'attenzione dedicata al VE e il coinvolgimento sperimentato
- Realismo sperimentato: misurare l'esperienza soggettiva del realismo nel VE

L'item generale aggiuntivo (Presenza generale) valuta il generale "senso di essere in quel posto in quel momento" e presenta un elevato carico su tutti e tre i fattori, con un carico particolarmente forte sulla Presenza spaziale [10][11]. Un quinto fattore, denominato PRES (che misura direttamente la Presenza generale con una singola domanda), è stato considerato e confrontato con il punteggio della Presenza generale per avere un'analisi più precisa delle diverse versioni del museo.

Factor	Item
Presence	In the computer generated world I had a sense of "being there"
Spatial Presence	Somehow I felt that the virtual world surrounded me.
	I felt like I was just perceiving pictures.
	I did not feel present in the virtual space.
	I had a sense of acting in the virtual space, rather than operating something from outside.
	I felt present in the virtual space.
Involvement	How aware were you of the real world surrounding while navigating in the virtual world? (i.e. sounds, room temperature, other people, etc.)?
	I was not aware of my real environment.
	I still paid attention to the real environment.
	I was completely captivated by the virtual world.
Experienced Realism	How real did the virtual world seem to you?
	How much did your experience in the virtual environment seem consistent with your real world experience?
	How real did the virtual world seem to you?
	The virtual world seemed more realistic than the real world.

Figura 31: Immagine ripresa da [8], rappresenta i set di domande divisi nei tre sottoinsiemi più l'item aggiuntivo

Dall'analisi dei risultati dei questionari, si è trovato che il punteggio della Presenza generale è di 4.16, corrispondente ad un rating "Molto buono", indicando un ambiente con un forte senso di immersione. Il fattore PRES ha indicato un punteggio medio di 5.01, corrispondente ad un rating "Eccellente". La differenza fra il fattore PRES e il rating di Presenza generale suggerisce che alcuni aspetti dell'ambiente, indicati dai sottoinsiemi Presenza spaziale, Coinvolgimento e Realismo sperimentato, ha intaccato l'esperienza.

Per quanto riguarda i sottoinsiemi, la Presenza spaziale è risultata in un punteggio di 4.79 (Molto buono), ne risulta che gli utenti si sono sentiti immersi nell'ambiente con successo. Il Realismo sperimentato ha raggiunto il rating di 3.44 (Soddisfacente), indicando che i partecipanti hanno trovato l'ambiente sufficentemente realistico. Il punteggio più basso lo ha ottenuto il sottoinsieme Coinvolgimento, notando che, seppur immersi nell'ambiente, gli utenti avessero un livello di interazione limitato, potendo solamente osservare l'ambiente circostante percorrendo le stanze del museo.
Come si può notare dall'immagine sottostante, tra le tre configurazioni della scena, non vi sono differenze significative in termini di punteggio fra una variante e l'altra.

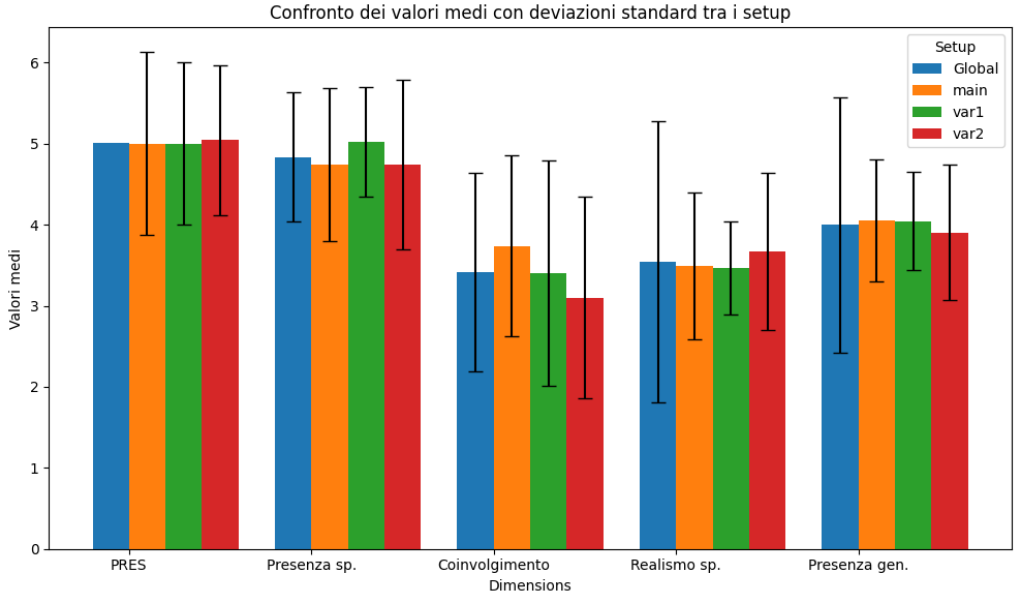


Figura 32: Diagramma che rappresenta i risultati dell'IPQ

5.1.2 VNRQ e VRISE

C'è un forte interesse sullo studio dell'idoneità dei sistemi di realtà virtuale immersiva (VR), per essere implementati in contesti clinici e di ricerca, a causa della presenza di nausea, vertigini, disorientamento, affaticamento e instabilità (VR Induced Symptoms and Effects: VRISE). Il Virtual Reality Neuroscience Questionnaire (VRNQ) è stato sviluppato per valutare la qualità del software VR in termini di esperienza utente, meccanica di gioco, assistenza in-game e VRISE [6]. I punteggi sono indicati con una scala Likert con un sistema a 7 punti, con i valori più alti che indicano un minor senso di disagio (1: estrema sensazione di disagio, 7: assenza di disagio). Per tutte le varianti il senso di disagio nell'ambiente virtuale è

stato rilevato come basso o molto basso. Nausea e Fatica sono stati quasi assenti per tutti gli utenti (valori medi di 6.67 e 6.71 rispettivamente). Valori più bassi sono stati rilevati invece per il senso di Disorientamento (5.86) e Instabilità(5.87). Fattori quindi presenti, ma solo in parte e che quindi probabilmente non significative per l'esperienza utente globale. Anche in questo caso, non sono state rilevate differenze significative tra le varianti del museo.

Setup	Nausea	Disorientamento	Vertigini	Fatica	Instabilità
Globale	6.67(1.08)	5.86(1.26)	6.25(1.33)	6.71(0.88)	5.87(1.36)
MAIN	6.83(0.39)	6.17(1.03)	6.52(0.85)	6.70(0.76)	6.04(1.02)
VARIANTE 1	6.57(1.31)	5.91(1.24)	6.43(1.41)	6.83(0.39)	5.87(1.18)
VARIANTE 2	6.61(1.31)	5.48(1.44)	5.78(1.57)	6.61(1.27)	5.70(1.79)

Tabella 3: Risultati del questionari. I risultati sono rappresentati come la media delle risposte con la deviazione standard tra parentesi.

5.2 Dispositivi

Grazie all'XR Interaction Toolkit fornito da Unity, è stato possibile usufruire della scena in modo interattivo con il visore Oculus Quest 3.
Oltre al visore, gli utenti hanno indossato un dispositivo per il riconoscimento delle emozioni che traccia gli impulsi elettrici della corteccia prefrontale tramite elettroencefalogramma (EEG - ElectroEncephaloGraphy), sviluppato da BrainSigns (Università la Sapienza, Roma, Italia, `https://www.brainsigns.com/en/`) – Mindtooth®(`https://www.mindtooth.com/`). Oltre al dispositivo di BrainSigns, agli utenti è stato disposto sul polso e sulle dita un dispositivo per la valutazione delle risposte biologiche, quali attività elettrodermica (EDA - ElectroDermal Activity) e frequenza cardiaca (HR - Heart Rate). Questi parametri sono stati registrati usando l'unita Shimmer3+ GSR, prodotta da Shimmer Sensing (`https://shimmersensing.com/`). I sensori sono stati posizionati sul secondo e terzo dito della mano non dominate. [4]

5.3 Preparazione

Prima dell'inizio della visita vera e propria, ai visitatori è stato chiesto prima di rimanere con gli occhi chiusi per un minuto per la calibrazione dei dispositivi, successivamente di guardare ad una parete bianca in assenza di stimoli per avere

(a) Stanza dove ha avuto luogo la visita

(b) Utente all'inizio della visita

(c) Utente nel corso della visita

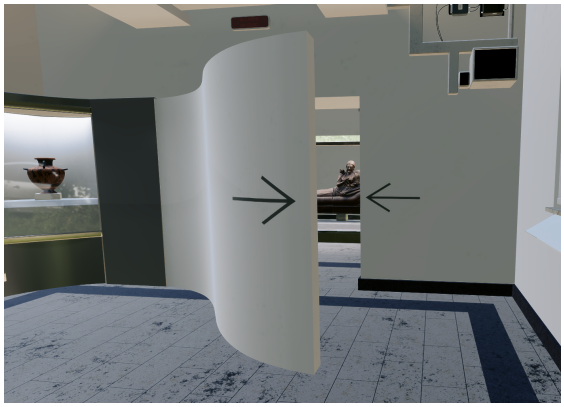
Figura 33: Luogo dell'esperimento, la stanza è stata adibita per avere le stesse dimensioni percorribili delle tre stanze del museo

un riferimento base per l'analisi (baseline)[4]. Uno stimolo uditivo, simile a quello utilizzato per terminare la visita, indica all'utente che può iniziare la visita, come precedentemente spiegato allo stesso.
La visita è stata monitorata con un normale PC desktop, dove è possibile selezionare tre scenari: la prima scena, chiamata MAIN, è una ricostruzione fedele del museo in tutte le sue parti. Nella seconda scena, chiamata VARIANTE 1, vi è stato aggiunto un separé artificiale prima della stanza del sarcofago per confrontare la reazione degli utenti rispetto alla casistica precedente e il proseguimento è stato bloccato con un muro. Nella terza scena, chiamata VARIANTE 2, oltre al separé, la stanza in cui è presente il sarcofago è stata modificata in modo da sembrare che l'utente sia immerso in un cielo stellato, così da fornire un effetto sorpresa appena

Figura 34: Utenti con indosso il dispositivo, come si può vedere, sono presenti dei conduttori sul capo e sulla fronte

entrato. 22 utenti hanno partecipato alla variante MAIN, 23 utenti hanno partecipato alla VARIANTE 1 e altri 23 alla VARIANTE 2.
All'interno dell'ambiente di Unity è stato possibile registrare i dati di posizione e orientamento del visore durante la visita, con una frequenza di campionamento di circa 15 Hz. I dati sono stati poi classificati come descritto nella sezione 4.9 e usati come input per la generazione delle heatmap. I dati iniziali, che sono concentrati nell'anticamera dove gli utenti sono dovuti rimanere fermi ad osservare una parete bianca per un minuto, sono stati eliminati, per non avere una concentrazione eccessiva di dati in un punto e non compromettere la visione globale (normalizzata) delle heatmap.

(a) Museo nella sua variante originale



(b) Variante del museo con un muro separatore per la stanza del sarcofago

(c) Variante del museo con il cielo stellato

Figura 35

5.4 Interpretazione ed elaborazione dei dati

5.4.1 Identificazione ed analisi dei picchi

Con i dati prodotti dai due dispositivi (sezione 5.2) ed elaborati dall'Università Sapienza di Roma, è stato possibile estrapolare i risultati di:

- indice di interesse/piacevolezza/approach e allontanamento/withrawial (asimmetria alfa frontale)
- indice di sforzo cognitivo (workload)
- indice di attivazione emozionale (arousal - conduttanza cutanea SC), divisa in Skin Conductance Reaction—SCR e Skin Conductance Level—SCL

I primi due set di dati oscillano fra l'origine, che viene determinata dalla baseline (la registrazione in condizioni neutre prima dell'inizio dell'esperienza), con valori negativi che indicano un approccio verso l'allontanamento e un valore di sforzo cognitivo minore rispetto alla baseline rispettivamente per l'indice approach/withrawial e l'indice di workload. I valori della skin conductance sono assoluti e normalizzati. Questi dati sono divisi nell'asse delle ascisse per il tempo, in valori discreti di secondi.
Considerato che i dati hanno zone mancanti, causate dalla mancata o corrotta registrazione da parte dei dispositivi, sono stati filtrati gli utenti con almeno l'80% dei valori validi, mentre i dati mancanti sono stati interpolati con una funzione di interpolazione lineare.
A questo punto, applicando la media mobile su un intervallo di tempo di 8 secondi è stato possibile "smussare" i valori, pronti per l'identificazione dei picchi. Sono stati utilizzati due metodi, uno base con due parametri, con l'altezza minima che i picchi possono avere (height) e la distanza minima sull'asse delle ascisse(distance). L'altro metodo invece trova i picchi che non superano la soglia (threshold) di n*deviazioni standard la media dei valori.
Una volta trovati i picchi, si identificano gli istanti di tempo in cui questi hanno luogo, e si generano degli intervalli di tempo attorno a ciascun picco. Si è deciso di scegliere una durata di 8 secondi per ogni intervallo, abbastanza ampia perché sia comprensibile, una volta generate le heatmap, il percorso visivo e di posizione dell'utente in quel lasso di tempo. Gli intervalli vengono poi salvati in un file con lo stesso formato della tabella 2 nella sezione 4.11, pagina 38.

5.4.2 Creazione di etichette

Per la creazione delle etichette si è fatto ricorso allo stesso metodo usato per l'identificazione dei punti di contatto tramite RayCast per la generazione delle heatmap. Per l'attenzione è stato istanziato un raggio con origine e direzione della

camera in scena. Per quanto riguarda la posizione invece il raggio parte sempre dalla Camera, ma punta verso il basso nelle coordinate globali. La funzione Physics.Raycast ha un attributo che restituisce in automatico il nome della mesh colpita. Un terzo dizionario è preimpostato con la corrispondenza nome oggetto - label (ad esempio: *Bacheca_Sarcofago*, *Sarcofago degli sposi*, *Pavimento_inizio*, *Stanza inizio*). Dato che il tempo dei dati di input è discreto e nell'ordine dei secondi, ma la frequenza di cattura di posizione e orientamento della Camera è su 14 Hz, è stato creato un dizionario che, per ogni secondo, calcola le volte in cui occorre ogni label in quel secondo. La label con le occorrenze maggiori è stata poi selezionata come quella da aggiungere al dizionario con i valori secondo/label. Si è proceduto quindi a caricare il risultato su un file .xlsx e a dividere lo stesso per ogni variante del museo. Ne è risultata una tabella con gli identificativi utente per ogni colonna, ogni secondo come indice di riga, e le label come valori utente/secondo.
Le label del pavimento sono state suddivise in: Stanza inizio, Stanza bacheche, Stanza sarcofago. Per le bacheche le label sono: Sarcofago sposi, Bacheca E, Bacheca D, Bacheca B1 (angolo), Bacheca C (angolo), Bacheca B, Bacheca A1, Bacheca A (angolo), Bacheca F (angolo). Per le bacheche si è fatto riferimento alla planimetria originale del museo, come mostrato nell'immagine sottostante.

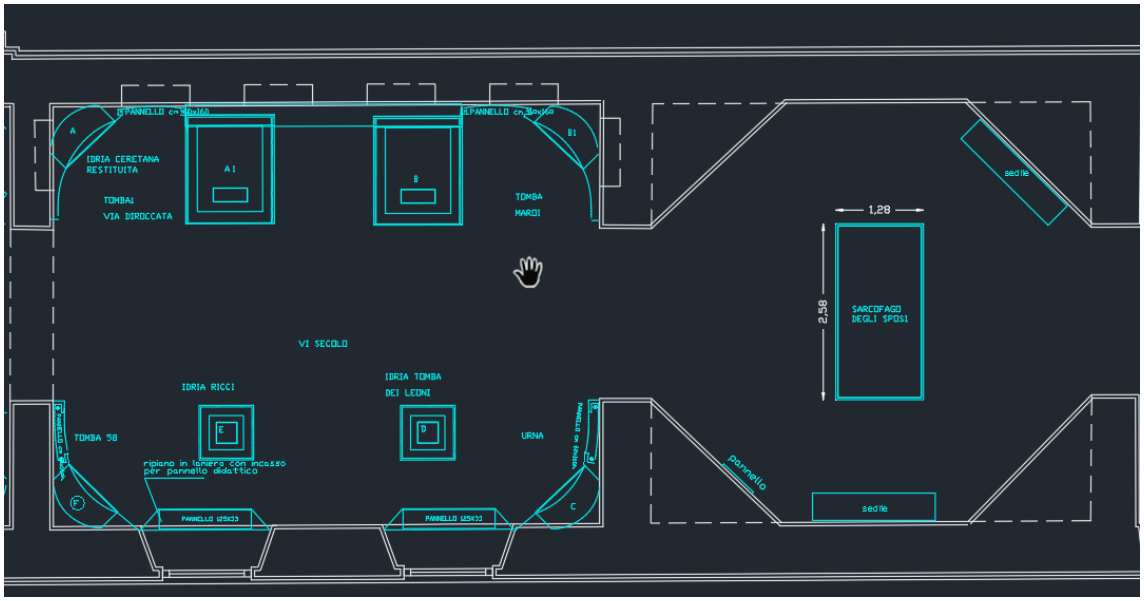
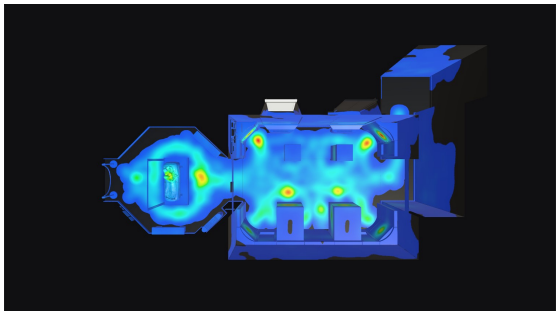


Figura 36: Planimetria delle due stanze principali del museo: la sala delle bacheche e la stanza del sarcofago, a cui si è fatto riferimento per le etichette.

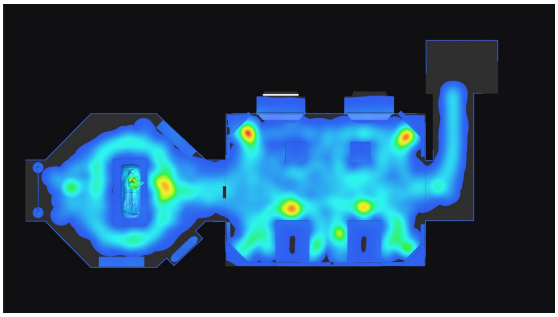
6 Risultati

6.1 Heatmap globale

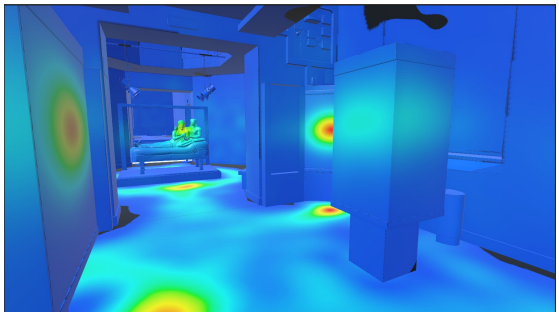
Gli utenti monitorati sono stati suddivisi per ogni scena, filtrati delle parti superflue e generate le heatmap corrispondenti. Le scene sono state precedentemente divise nella parte per l'heatmap 2D posizionale (sezione 4.6) e la parte di gaze, o visualizzazione.
Si è deciso preliminarmente di modificare la scena, facendo in modo che i vetri delle teche non fossero trasparenti, ma opacizzati con un materiale apposito, in modo che fosse visibile solamente l'esterno della bacheca. Per quanto riguarda gli utensili, i vasi e le ceramiche all'interno delle teche si è preferito assegnare il layer NoLoad (introdotto nella sezione 4.12, layer che non considera gli oggetti nella generazione delle heatmap), in quanto si è notato che in questo modo i tempi di caricamento delle mappe si riducono notevolmente, fino a tre volte in meno rispetto al tempo necessario considerando tutte le mesh.
Per tutte le varianti è stato posto un raggio di 0.5 unità e un valore di distribuzione di 1.5, concentrato più sui valori massimi. Tutte le texture sono state poi salvate su disco per ulteriore visualizzazione o elaborazione.
Nella variante MAIN non si è tenuta in considerazione la terza stanza, dato che non è di interesse nel confronto con le altre due varianti.
Il risultato è una mappa generale dell'heatmap con tutti gli utenti, dove le zone che sono state oggetto di maggior attenzione sono evidenziate in rosso, degradando mano a mano verso l'arancione, il giallo, il verde ed infine l'azzurro ed il blu. Le zone non considerate da nessun utente sono segnate in nero.
Nelle pagine seguenti è presentata una rassegna di immagini che mostrano tutte le varianti con l'heatmap applicata, con particolare attenzione alla pianta del museo, alla stanza delle bacheche e alla stanza del sarcofago (figure 37 - 38 - 39). Da queste immagini si può notare come gli utenti si siano soffermati principalmente sulle teche ad angolo, così come sulle teche grandi a muro, in particolare la seconda in ordine dall'entrata. Per quando riguarda il Sarcofago degli sposi, in generale gli utenti hanno girato intorno alla scultura con particolare attenzione alla parte frontale e al busto delle due figure.
Fra una variante e l'altra non vi sono differenze significative. Si può solamente notare qualche leggera differenza, ad esempio nella VARIANTE 2 gli utenti si sono soffermati di più all'entrata della sala del sarcofago rispetto alla VARIANTE 1. Nelle seconda variante c'è anche stata una maggiore concentrazione davanti alla teca nell'angolo sud-est della seconda stanza. Nella variante MAIN gli utenti hanno girato di più attorno al sarcofago, tenendo conto però che la stanza successiva non era nascosta, ma vi era solamente impedito l'accesso tramite un separatore.



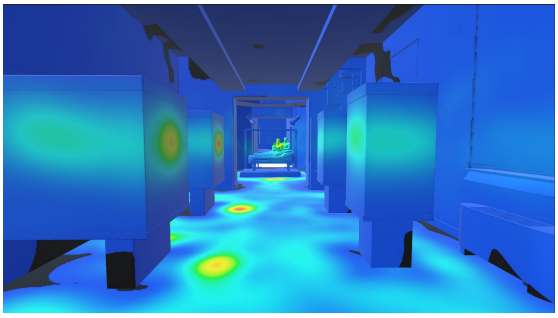
(a) Pianta delle stanze, dove si possono vedere i movimenti degli utenti e dove si sono fermati più tempo



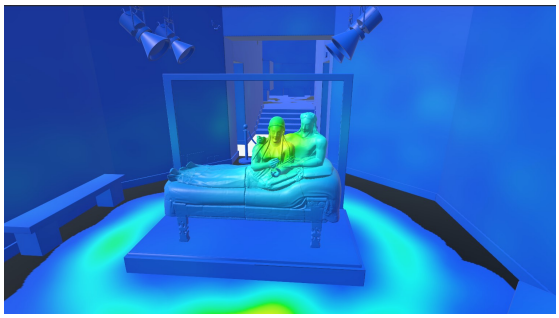
(b) Pianta con prospettiva ortogonale dall'alto



(c) Stanza con presenti le bacheche, qui si può notare che la concentrazione maggiore è posta nella bachecha nell'angolo



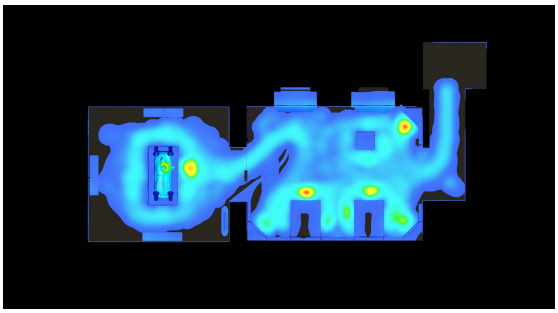
(d) Visuale del museo dall'entrata, qui si può vedere il percorso fatto dagli utenti e dove si sono soffermati



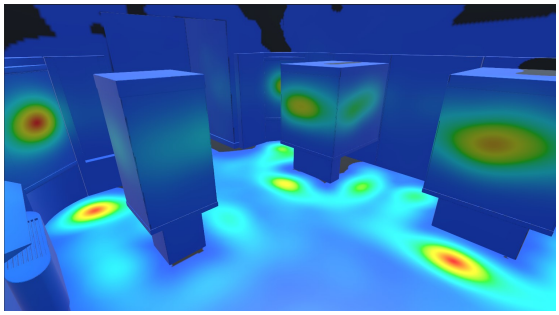
(e) Stanza del sarcofago, in questo caso si è deciso di non rendere la bacheca opaca per la centralità dell'opera

Figura 37: Scena MAIN

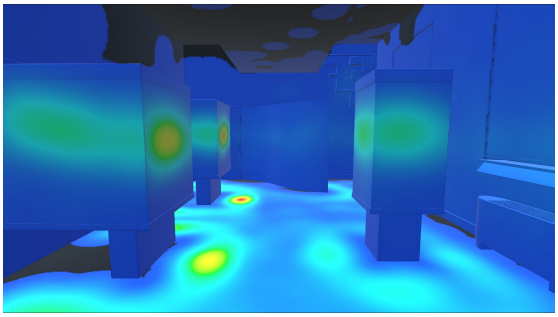
(a) Pianta del museo, si può notare il cambio di percorso dato dal muro e la rimozione della bacheca all'angolo rispetto alla scena MAIN



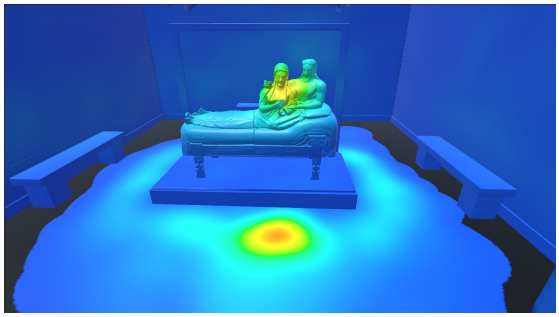
(b) Pianta con prospettiva ortogonale dall'alto



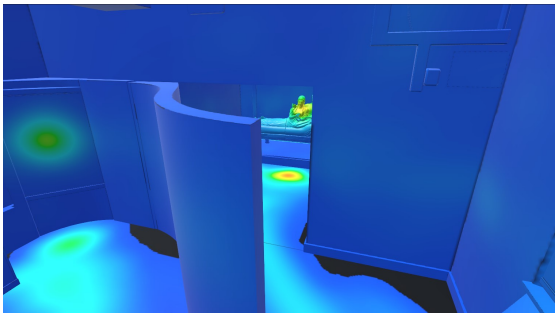
(c) Visuale dall'angolo della stanza con le bacheche, si può notare la concentrazione dei valori massimi sulla bacheca all'angonlo e la bacheca grande a destra



(d) Visuale dall'entrata del museo, in questo caso la visuale iniziale del sarcofago è stata impedita da un muro separatore

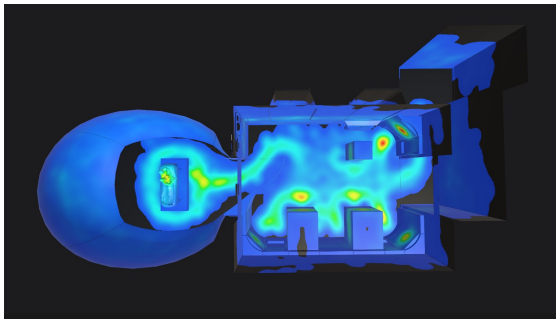


(e) Stanza del sarcofago, in questo caso chiusa, si può notare infatti una maggior concentrazione di attenzione sulla parte frontale del sarcofago

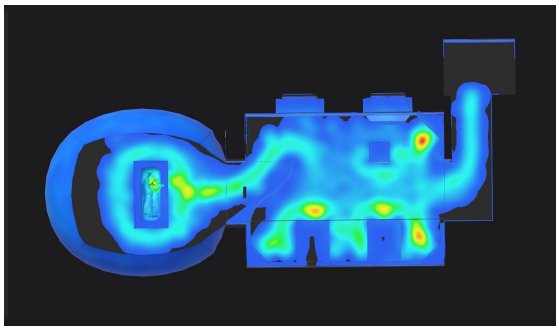


(f) Dettagli del muro separatore, che porta alla stanza del sarcofago

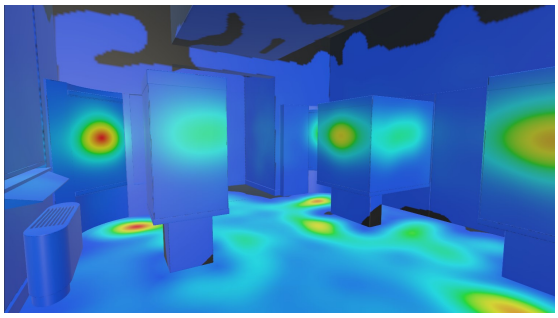
Figura 38: Scena VARIANTE 1



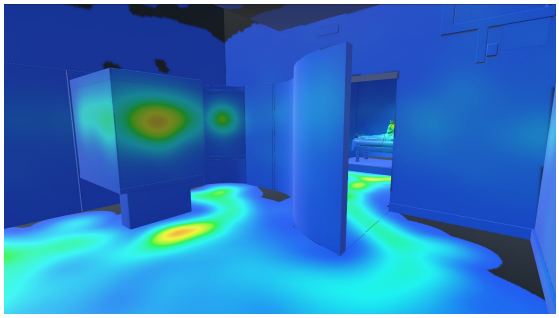
(a) Pianta del museo, in cui spicca la calotta stellata rispetto alle altre due varianti



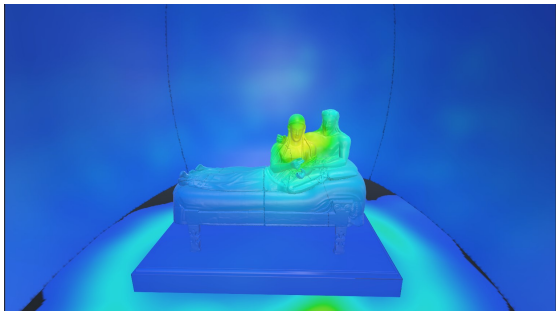
(b) Pianta con prospettiva ortogonale dall'alto



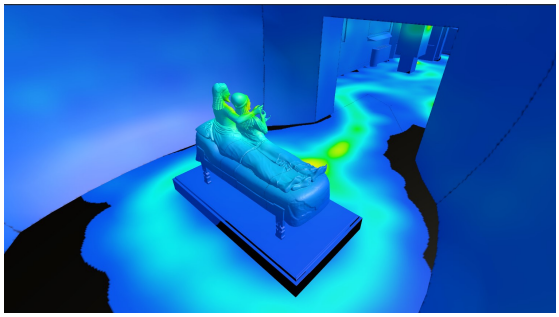
(c) Visuale dall'angolo della stanza delle bacheche, in questo caso la VARIANTE 1 e la VARIANTE 2 hanno una disposizione molto simile



(d) Anche in questo caso un muro separatore impedisce la vista della stanza del sarcofago all'inizio della visita



(e) Visuale anteriore della stanza del sarcofago, con la calotta stellata che avvolge il Sarcofago degli Sposi



(f) Retro del sarcofago e della calotta stellata, con dettaglio della stanza precedente

Figura 39: Scena VARIANTE 2

6.1.1 Analisi dei dati fisiologici a livello globale

Con i dati prodotti dai dispositivi ed elaborati dall'Università di Sapienza di Roma, si è notato come i valori di Skin Conductance, cioè di attivazione emozionale, sono molto vicini fra loro nelle tre varianti, con i valori delle due varianti 1 e 2 che superano appena la variante MAIN. Per quanto riguarda l'indice di piacevolezza, la crescita è più marcata fra la variante MAIN e la VARIANTE 1, così come fra la VARIANTE 1 e la VARIANTE 2. Purtroppo l'analisi non raggiunge la significatività statistica, ma il confronto fra le varianti mostra una forte tendenza all'approccio per la VARIANTE 2 e all'evitamento per la variante MAIN. Anche il workload presenta una marcata differenza fra il MAIN e le due varianti, differenza che però si assottiglia per le varianti 1 e 2, rispetto all'indice di piacevolezza.
Per l'analisi delle tre varianti si è usato il metodo di analisi della varianza ANOVA. Il test viene eseguito se si sospetta che una determinata variabile di processo indipendente possa influire in maniera determinante sui risultati di tale processo. Nel nostro caso la variabile è il tipo di variante del museo, utilizzando tre livelli (main, var1 e var2) di questa variabile indipendente (o fattore) e si ricavano dei campioni per l'osservazione. Se dal confronto delle medie di ogni gruppo di osservazione tramite ANOVA emergono delle differenze, allora si dimostra che il fattore analizzato influisce sui risultati ottenuti. In generale, una tabella ANOVA include:

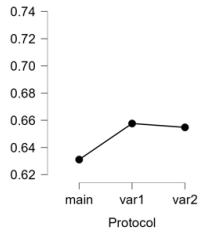
- Origine: le origini della varianza.
- DF: gradi di libertà per ogni origine della varianza.
- Somma dei quadrati per ogni origine della varianza, insieme al totale di tutte le origini.
- Media quadratica: la somma dei quadrati divisa per i relativi gradi di libertà.
- Statistica F: la media quadratica del fattore divisa per la media quadratica dell'errore.
- Prob $>$ F: il p-value.

In figura 40 sono riporate le tabelle con i risultati.
È stato eseguito anche un confronto fra le singole varianti con il t-test. Il test t (noto anche come test t di Student) è uno strumento per valutare le medie di una o due popolazioni tramite verifica d'ipotesi. Il test t può essere usato per determinare se un singolo gruppo differisce da un valore conosciuto (test t a un campione), oppure se due gruppi differiscono l'uno dall'altro (test t a due campioni indipendenti).
Nelle figure 41-42, pagine 63-64, vengono mostrate le tabelle e i diagrammi con i risultati del t-test.

ANOVA - GSR

Cases	Sum of Squares	df	Mean Square	F	p
Protocol	0.009	2	0.005	0.442	0.645
Residuals	0.635	62	0.010		

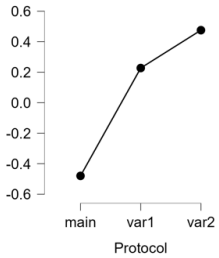
Note. Type III Sum of Squares



ANOVA – Approach/Withdrawal

Cases	Sum of Squares	df	Mean Square	F	p
Protocol	10.670	2	5.335	1.126	0.331
Residuals	303.188	64	4.737		

Note. Type III Sum of Squares



ANOVA - Workload

Cases	Sum of Squares	df	Mean Square	F	p
Protocol	26.281	2	13.141	1.249	0.294
Residuals	673.181	64	10.518		

Note. Type III Sum of Squares

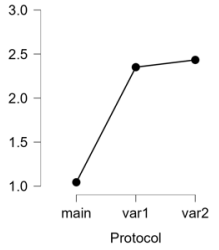


Figura 40: Le tre tabelle con il test ANOVA

6.1.2 Analisi delle etichette

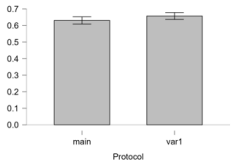
Dalla produzione delle etichette nella sezione 5.4.2, l'Università di Roma La Sapienza ha prodotto i risultati ANOVA sulla media dei valori. Sono state solamente selezionate le etichette delle bacheche in comune tra le tre varianti, facendo sempre riferimento alla figura 36. Purtroppo i risultati non hanno portato ad un esito statisticamente significativo. Nelle figure 43 - 44 - 45 sono rappresentate le tabelle con i risultati.

6.1.3 Correlazioni tra IPQ e dati

Per identificare le relazioni tra i risultati dei test IPQ e gli indici in approach/withrawial e di workload, si è utilizzato il coefficiente di Spearman. Il coefficiente di Spearman è un indice statistico utilizzato per misurare il grado di relazione tra due variabili, in modo da valutare la presenza di relazioni tra i dati delle coppie di variabili. Nel campione che ha visitato la variante MAIN si è notato che all'aumentare dello sforzo cognitivo è aumentato il rating alla seguente domanda dell'IPQ: "In qualche modo mi sentivo circondato dal mondo virtuale". Per gli utenti che hanno fatto visita nella VARIANTE 1 all'aumentare dell'indicde

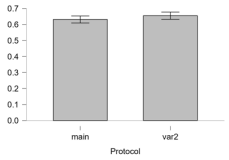
Independent Samples T-Test

	Test	Statistic	df	p	Effect Size	SE Effect Size
GSR	Student	-0.880	41	0.384	-0.269	0.308
	Mann-Whitney	205.000		0.539	-0.113	0.176



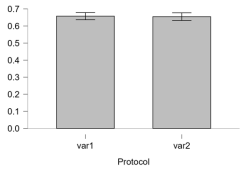
Independent Samples T-Test

	Test	Statistic	df	p	Effect Size	SE Effect Size
GSR	Student	-0.754	41	0.455	-0.230	0.307
	Mann-Whitney	202.000		0.493	-0.126	0.176



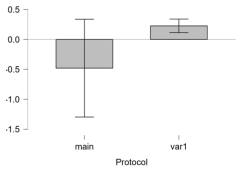
Independent Samples T-Test

	Test	Statistic	df	p	Effect Size	SE Effect Size
GSR	Student	0.092	42	0.927	0.028	0.302
	Mann-Whitney	245.000		0.954	0.012	0.174



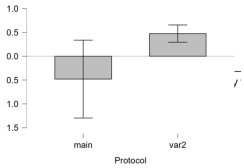
Independent Samples T-Test

	Test	Statistic	df	p	Effect Size	SE Effect Size
Piacevolezza	Student	-0.897	42	0.375	-0.271	0.305
	Mann-Whitney	195.000		0.283	-0.193	0.174



Independent Samples T-Test

	Test	Statistic	df	p	Effect Size	SE Effect Size
Approach/ Withdrawal	Student	-1.191	42	0.240	-0.359	0.307
	Mann-Whitney	170.000		0.096	-0.296	0.174



Independent Samples T-Test

	Test	Statistic	df	p	Effect Size	SE Effect Size
Piacevolezza	Student	-1.164	44	0.251	-0.343	0.299
	Mann-Whitney	231.000		0.472	-0.127	0.170

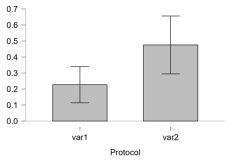
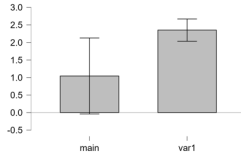


Figura 41: Tabelle e diagrammi con t-test per quanto riguarda il GSR e l'approach/withrawal

di piacevolezza aumentava il rating per le domande: "Mi sentivo presente nello spazio virtuale" e "Quanto eri consapevole del mondo reale circostante mentre navigavi nel mondo virtuale?". Quest'ultima domanda è correlata all'aumento dell'indice di piacevolezza anche per la VARIANTE 2.

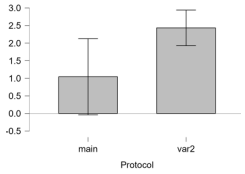
Independent Samples T-Test

	Test	Statistic	df	p	Effect Size	SE Effect Size
Workload	Student	-1.204	42	0.235	-0.363	0.307
	Mann-Whitney	215.000		0.545	-0.110	0.174



Independent Samples T-Test

	Test	Statistic	df	p	Effect Size	SE Effect Size
Workload	Student	-1.196	42	0.238	-0.361	0.307
	Mann-Whitney	257.000		0.727	0.064	0.174



Independent Samples T-Test

	Test	Statistic	df	p	Effect Size	SE Effect Size
Workload	Student	-0.138	44	0.891	-0.041	0.295
	Mann-Whitney	290.000		0.586	0.096	0.170

Note. For the Student t-test, effect size is given by Cohen's d. For the Mann-Whitney test, effect size is given by the rank biserial correlation.

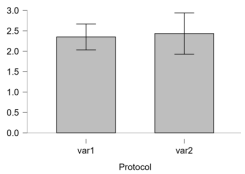


Figura 42: Tabelle e diagrammi per quanto riguarda il workload

ANOVA - Gaze_AW_mean_Sarcofago sposi

Cases	Sum of Squares	df	Mean Square	F	p
Experiment	$1.249\times10^{+8}$	2	$6.243\times10^{+7}$	0.883	0.419
Residuals	$4.100\times10^{+9}$	58	$7.068\times10^{+7}$		

ANOVA - Gaze_AW_mean_Bacheca A (angolo)

Cases	Sum of Squares	df	Mean Square	F	p	η^2_p
Experiment	$1.472\times10^{+8}$	2	$7.361\times10^{+7}$	0.799	0.456	0.035
Residuals	$4.056\times10^{+9}$	44	$9.219\times10^{+7}$			

ANOVA - Gaze_AW_mean_Bacheca A1

Cases	Sum of Squares	df	Mean Square	F	p	η^2_p
Experiment	$1.224\times10^{+8}$	2	$6.122\times10^{+7}$	0.836	0.439	0.029
Residuals	$4.100\times10^{+9}$	56	$7.321\times10^{+7}$			

ANOVA - Gaze_AW_mean_Bacheca B

Cases	Sum of Squares	df	Mean Square	F	p	η^2_p
Experiment	$1.237\times10^{+8}$	2	$6.185\times10^{+7}$	0.860	0.429	0.029
Residuals	$4.100\times10^{+9}$	57	$7.192\times10^{+7}$			

ANOVA - Gaze_AW_mean_Bacheca E

Cases	Sum of Squares	df	Mean Square	F	p	η^2_p
Experiment	$1.394\times10^{+8}$	2	$6.972\times10^{+7}$	0.923	0.404	0.033
Residuals	$4.080\times10^{+9}$	54	$7.556\times10^{+7}$			

ANOVA - Gaze_AW_mean_Bacheca F (angolo)

Cases	Sum of Squares	df	Mean Square	F	p	η^2_p
Experiment	$1.353\times10^{+8}$	2	$6.764\times10^{+7}$	0.845	0.435	0.032
Residuals	$4.080\times10^{+9}$	51	$8.000\times10^{+7}$			

Figura 43: Le tabelle con il test ANOVA per quanto riguarda l'approach/withrawal

ANOVA - Gaze_Workload_mean_Sarcofago sposi

Cases	Sum of Squares	df	Mean Square	F	p	η^2_p
Experiment	$1.248\times10^{+8}$	2	$6.242\times10^{+7}$	0.883	0.419	0.030
Residuals	$4.099\times10^{+9}$	58	$7.068\times10^{+7}$			

ANOVA - Gaze_Workload_mean_Bacheca A (angolo)

Cases	Sum of Squares	df	Mean Square	F	p	η^2_p
Experiment	$1.473\times10^{+8}$	2	$7.364\times10^{+7}$	0.799	0.456	0.035
Residuals	$4.056\times10^{+9}$	44	$9.218\times10^{+7}$			

ANOVA - Gaze_Workload_mean_Bacheca A1

Cases	Sum of Squares	df	Mean Square	F	p	η^2_p
Experiment	$1.225\times10^{+8}$	2	$6.123\times10^{+7}$	0.836	0.439	0.029
Residuals	$4.099\times10^{+9}$	56	$7.320\times10^{+7}$			

ANOVA - Gaze_Workload_mean_Bacheca B

Cases	Sum of Squares	df	Mean Square	F	p	η^2_p
Experiment	$1.237\times10^{+8}$	2	$6.183\times10^{+7}$	0.860	0.429	0.029
Residuals	$4.099\times10^{+9}$	57	$7.192\times10^{+7}$			

ANOVA - Gaze_Workload_mean_Bacheca E

Cases	Sum of Squares	df	Mean Square	F	p	η^2_p
Experiment	$1.394\times10^{+8}$	2	$6.971\times10^{+7}$	0.923	0.404	0.033
Residuals	$4.080\times10^{+9}$	54	$7.555\times10^{+7}$			

ANOVA - Gaze_Workload_mean_Bacheca F (angolo)

Cases	Sum of Squares	df	Mean Square	F	p	η^2_p
Experiment	$1.351\times10^{+8}$	2	$6.757\times10^{+7}$	0.845	0.436	0.032
Residuals	$4.080\times10^{+9}$	51	$8.000\times10^{+7}$			

Figura 44: Le tabelle con il test ANOVA per quanto riguarda il workload

ANOVA - Gaze_SCL_mean_Sarcofago sposi

Cases	Sum of Squares	df	Mean Square	F	p	η^2_p
Experiment	7.804×10^{-4}	2	3.902×10^{-4}	0.010	0.990	3.473×10^{-4}
Residuals	2.246	59	0.038			

ANOVA - Gaze_SCL_mean_Bacheca A (angolo)

Cases	Sum of Squares	df	Mean Square	F	p	η^2_p
Experiment	0.036	2	0.018	0.332	0.719	0.015
Residuals	2.412	45	0.054			

ANOVA - Gaze_SCL_mean_Bacheca A1

Cases	Sum of Squares	df	Mean Square	F	p	η^2_p
Experiment	0.033	2	0.017	0.494	0.613	0.017
Residuals	1.907	57	0.033			

Note. Type III Sum of Squares

ANOVA - Gaze_SCL_mean_Bacheca B

Cases	Sum of Squares	df	Mean Square	F	p	η^2_p
Experiment	0.043	2	0.021	0.552	0.579	0.019
Residuals	2.234	58	0.039			

ANOVA - Gaze_SCL_mean_Bacheca E

Cases	Sum of Squares	df	Mean Square	F	p	η^2_p
Experiment	0.015	2	0.007	0.183	0.833	0.007
Residuals	2.212	55	0.040			

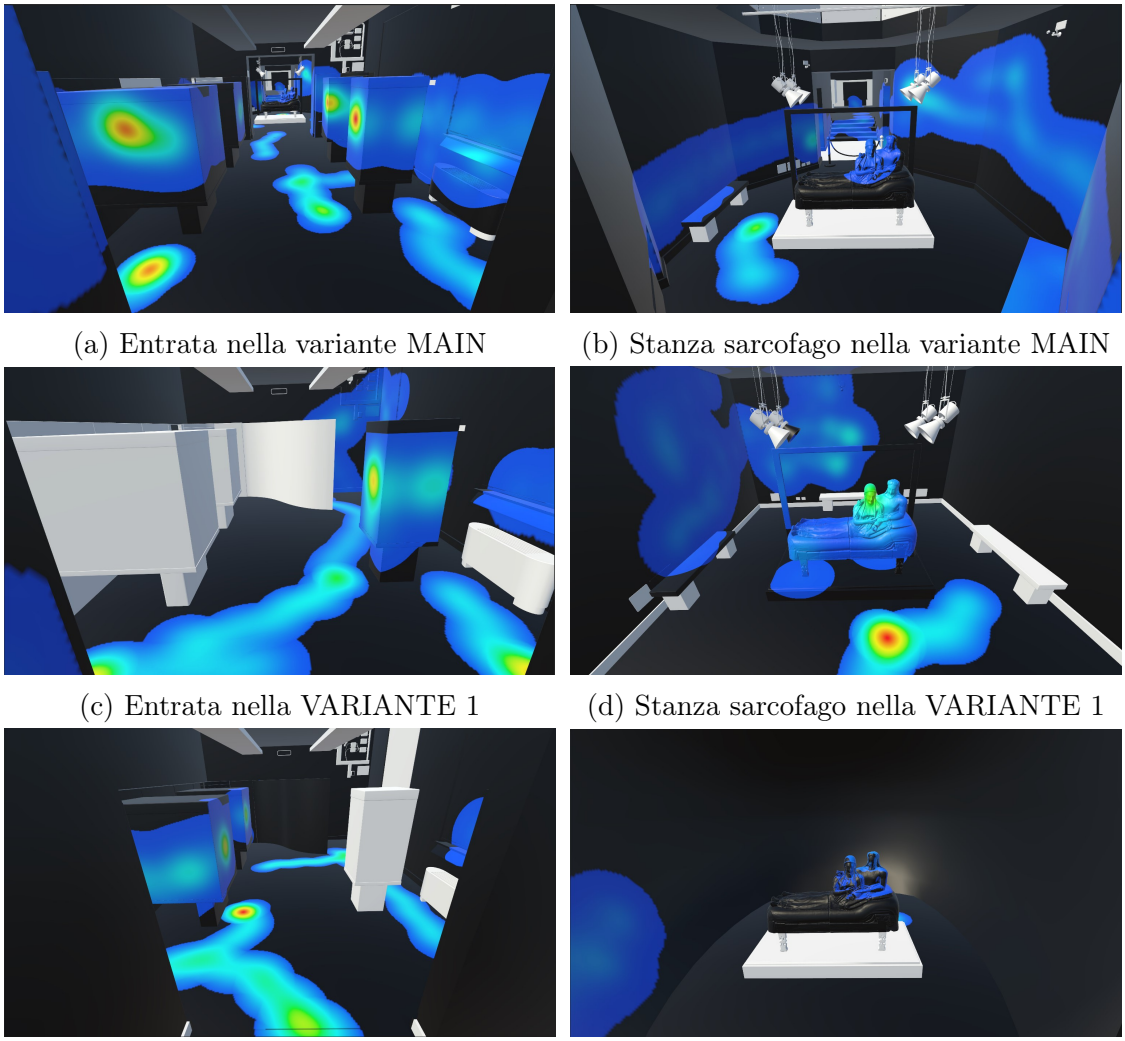
ANOVA - Gaze_SCL_mean_Bacheca F (angolo)

Cases	Sum of Squares	df	Mean Square	F	p	η^2_p
Experiment	0.056	2	0.028	0.552	0.579	0.021
Residuals	2.634	52	0.051			

Figura 45: Le tabelle con il test ANOVA per quanto riguarda lo Skin Conductance Level

6.2 Picchi emotivi

Con l'individuazione dei picchi emotivi, in particolare dell'indice di attivazione emozionale, Skin Conductance Level, sono state prodotte le heatmap per le tre varianti del museo, basandosi sui file prodotti come spiegato nella sezione 5.4.
In questo caso alcune differenze si possono notare: nella scena MAIN le teche principali, quelle al centro della stanza, hanno avuto un'attenzione più marcata rispetto alle altre due varianti (figura 46a). Nella VARIANTE 1 le bacheche sulla sinistra non hanno avuto nessun riscontro emotivo (gli oggetti di colore bianco non sono stati colpiti dallo sguardo dell'utente, figura 46c), al contrario della VARIANTE 2, dove la teca sulla destra non ha suscitato picchi emotivi (figura 46e). Per quanto riguarda la stanza del sarcofago, solo gli utenti della VARIANTE 1 hanno avuto dei picchi di fronte alla statua (figura 46c), mentre nel MAIN ne hanno avuti sulla parte laterale (figura 46b) e quelli della VARIANTE 2 solamente sul retro (figura 46f). Queste differenze si possono notare nella figura 46. Nelle figure 47-48-49 sono raffigurati dei dettagli delle tre varianti.



(a) Entrata nella variante MAIN

(b) Stanza sarcofago nella variante MAIN

(c) Entrata nella VARIANTE 1

(d) Stanza sarcofago nella VARIANTE 1

(e) Entrata nella VARIANTE 2

(f) Stanza sarcofago nella VARIANTE 2

Figura 46: Heatmap che evidenziano le differenze tra i picchi emotivi nelle tre varianti

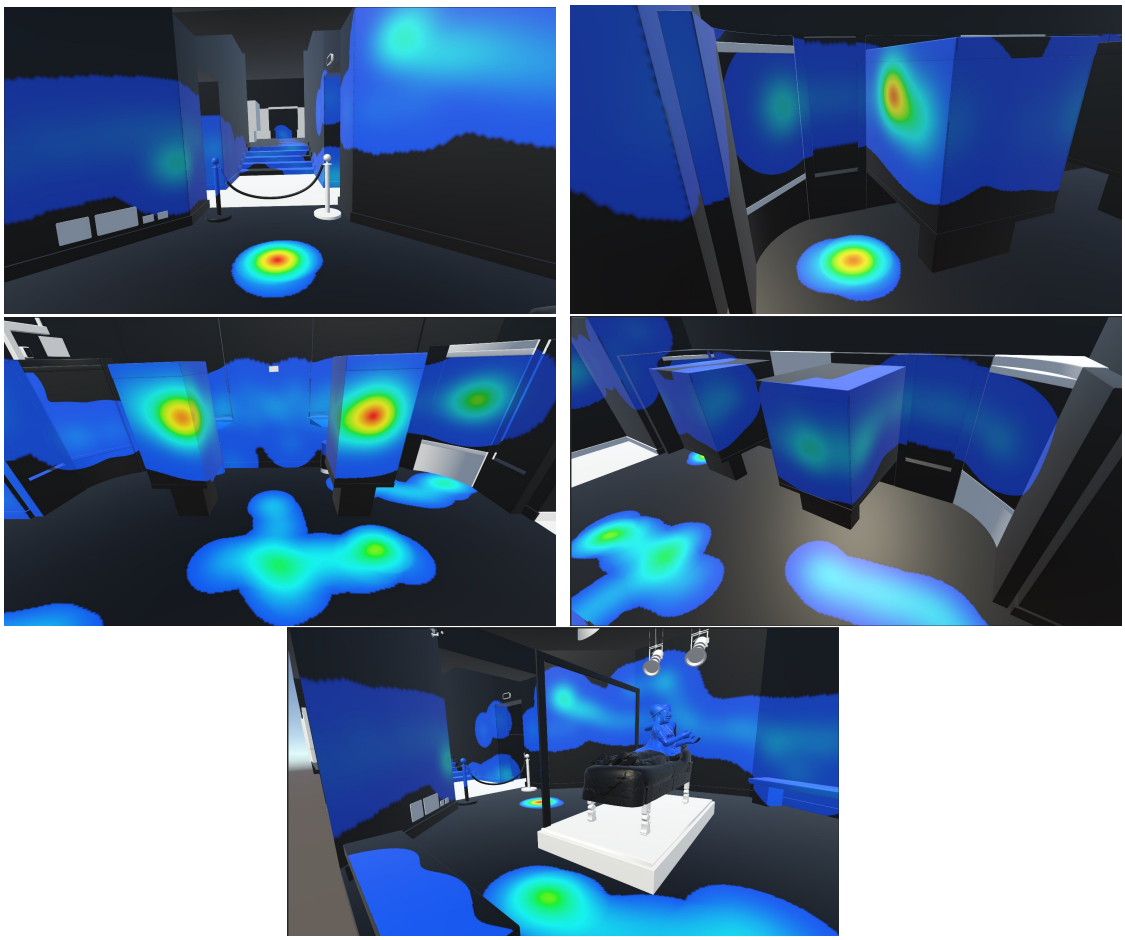


Figura 47: Scena MAIN

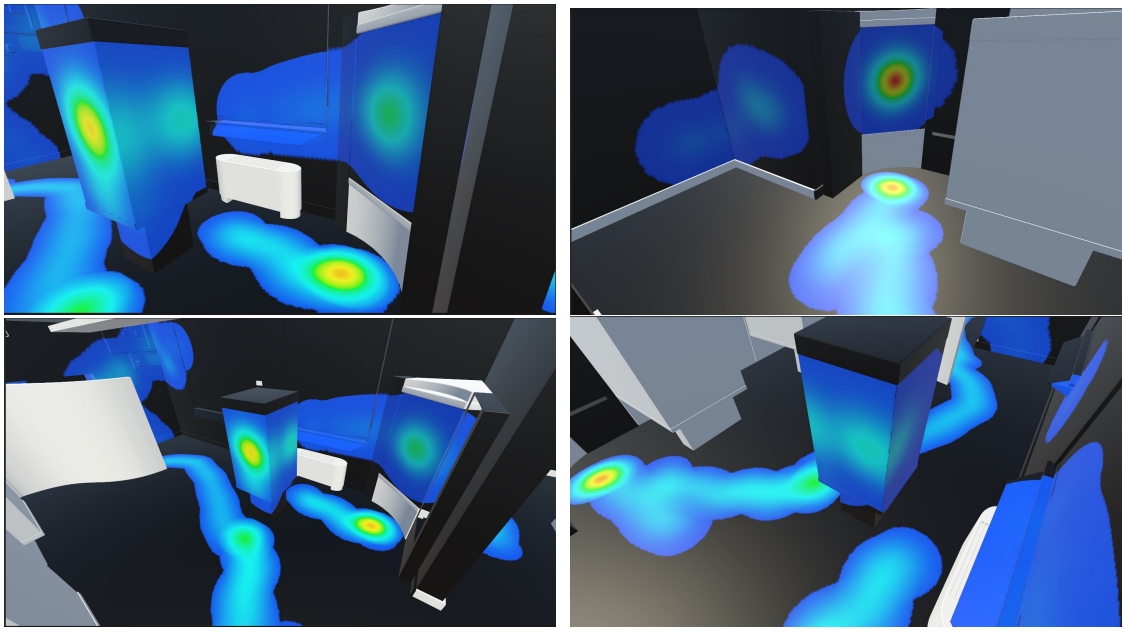


Figura 48: Scena VARIANTE 1

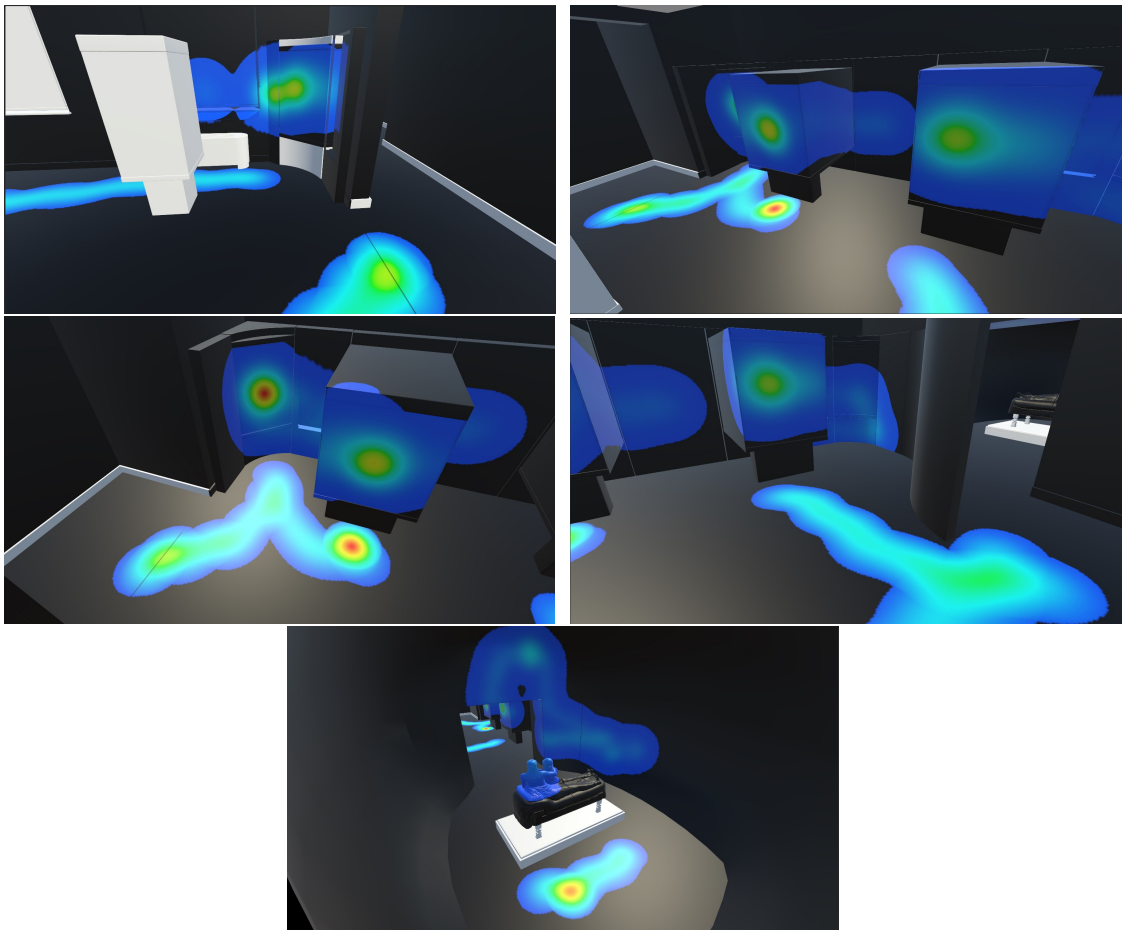


Figura 49: Scena VARIANTE 2

7 Conclusione

In questa tesi viene presentata l'applicazione di heatmap in un ambiente virtuale tridimensionale, partendo dalla ricostruzione di un museo reale. Sono state sviluppate sia heatmap tridimensionali, per il monitoraggio dell'attenzione degli utenti, sia heatmap bidimensionale, per il monitoraggio della posizione e il percorso eseguiti. Il sistema preliminarmente genera un'attention map, texture mono-canale che immagazzina i valori al suo interno. I valori sono estrapolati da dati che contengono informazioni sulla posizione e orientamento del visore. Un raggio perpendicolare al piano del visore indica il punto di contatto con l'oggetto dell'attenzione o della posizione. Sull'attention map vengono poi applicate lo shadow mapping e la normalizzazione. Con una conversione di colori basandosi sulla scala di colori HSV, si è trasformata l'attention map in heatmap. In un esperimento che ha avuto luogo nel mese di luglio 2024, ad un campione di utenti è stato chiesto di effettuare una visita virtuale all'interno del Museo Etrusco di Villa Giulia a Roma, con due varianti che modificano la struttura delle stanze. Queste tre varianti sono state rappresentate con l'heatmap applicata, potendo compiere un'analisi qualitativa del percorso e dell'attenzione degli utenti. Con i dati elaborati dall'Università di Roma la Sapienza, è stato possibile confrontare, sempre qualitativamente, le differenze tra le varianti.
In conclusione, è stato possibile fornire uno strumento per l'analisi qualitativa di una scena museale virtuale. Questo strumento può essere applicato anche in altri campi, usando scenari e set di dati diversi.
Lo studio oggetto di questa tesi può essere ampliato grazie alle tecnologie che si stanno sviluppando in questi anni, sia a livello software che hardware. Con l'utilizzo delle nuove tecnologie, come i compute shaders, si potrebbero sviluppare delle mappe interamente all'interno della GPU, accelerando il processo di generazione delle stesse, che può essere problematico all'aumentare dei dataset in input. L'aggiunta di un dispositivo di eye-tracking potrebbe inoltre aumentare la precisione dell'analisi, monitorando direttamente la direzione delle pupille anziché quella dell'intero visore. In questo caso sarebbe necessario distinguere la pupilla destra dalla sinistra, e calcolare simultaneamente i punti di contatto del raggio proiettato da esse con l'oggetto dell'attenzione.

Riferimenti bibliografici

[1] A. Bc. Enhancing military training through vr applications. *International Scientific Journal of Engineering and Management*, 03:1–9, 05 2024. doi: 10.55041/ISJEM01739.

[2] T. Blascheck, K. Kurzhals, M. Raschke, M. Burch, D. Weiskopf, and T. Ertl. Visualization of eye tracking data: A taxonomy and survey: Visualization of eye tracking data. *Computer Graphics Forum*, 36, 02 2017. doi: 10.1111/cgf.13079.

[3] S. Fang. The application of vr in films and games. *Highlights in Science, Engineering and Technology*, 85:1300–1305, 03 2024. doi: 10.54097/ydf30368.

[4] A. Giorgi, S. Menicocci, M. Forte, V. Ferrara, M. Mingione, P. Alaimo Di Loro, B. M. S. Inguscio, S. Ferrara, F. Babiloni, A. Vozzi, V. Ronca, and G. Cartocci. Virtual and reality: A neurophysiological pilot study of the sarcophagus of the spouses. *Brain Sciences*, 13(4), 2023. ISSN 2076-3425. doi: 10.3390/brainsci13040635. URL `https://www.mdpi.com/2076-3425/13/4/635`.

[5] T. Higgins, P. Main, J. Lang, and B. Museum. *Imaging the Past: Electronic Imaging and Computer Graphics in Museums and Archaeology*. British Museum London: Occasional paper. British Museum, 1996. ISBN 9780861591145. URL `https://books.google.no/books?id=PopiQgAACAAJ`.

[6] P. Kourtesis, S. Collina, L. A. A. Doumas, and S. E. MacPherson. Validation of the virtual reality neuroscience questionnaire: Maximum duration of immersive virtual reality sessions without the presence of pertinent adverse symptomatology. *Hum. Neurosci.*, 13, 2019. doi: https://doi.org/10.3389/fnhum.2019.00417.

[7] S. Marschner and P. Shirley. *Fundamentals of Computer Graphics, Fourth Edition*. A. K. Peters, Ltd., USA, 4th edition, 2016. ISBN 1482229390.

[8] M. Melo, G. Gonçalves, j. Vasconcelos-Raposo, and M. Bessa. How much presence is enough? qualitative scales for interpreting the igroup presence questionnaire score. *IEEE Access*, 11:24675–24685, 2023. doi: 10.1109/ACCESS.2023.3254892.

[9] T. Pfeiffer and C. Memili. Gpu-accelerated attention map generation for dynamic 3d scenes. pages 257–258, 03 2015. doi: 10.1109/VR.2015.7223393.

[10] H. Regenbrecht and T. Schubert. Real and illusory interactions enhance presence in virtual environments. *Presence: Teleoperators & Virtual Environments*, 11(4):425–434, 2002.

[11] T. Schubert, F. Friedmann, and H. Regenbrecht. Decomposing the sense of presence: Factor analytic insights. 01 1999.

[12] T. W. Schubert. The sense of presence in virtual environments: A three-component scale measuring spatial presence, involvement, and realness. *Z. für Medienpsychologie*, 15(2):69–71, 2003.

[13] F. Stanco and D. Tanasi. *Experiencing the past: Computer Graphics in Archaeology.*, pages 51–88. 01 2011.

[14] V. Vlahakis, J. Karigiannis, M. Tsotros, M. Gounaris, L. Almeida, D. Stricker, T. Gleue, I. Christou, and N. Ioannidis. Archeoguide: first results of an augmented reality, mobile computing system in cultural heritage sites. pages 131–140, 01 2001. doi: 10.1145/584993.585015.

[15] K. Wang. Research and application of vr in training and learning. *Highlights in Science, Engineering and Technology*, 85:1306–1313, 03 2024. doi: 10.54097/anhqyy16.

[16] M. White, K. Walczak, and N. Mourkoussis. Arco: Augmented representation of cultural objects. 18:14–15+46, 06 2003.