Project: Non-Euclidian

1 Introduzione 3

1.1 Informazioni sul progetto 3

1.2 Abstract 3

1.3 Scopo 3

2 Analisi 4

2.1 Analisi del dominio 4

2.2 Analisi e specifica dei requisiti 4

2.2.1 Spiegazione elementi tabella dei requisiti: 5

2.3 Use case 5

2.4 Pianificazione 5

2.5 Analisi dei mezzi 5

2.5.1 Software 6

2.5.2 Hardware 6

3 Progettazione 6

3.1 Design dell’architettura del sistema 6

3.2 Design dei dati e database 6

3.3 Design delle interfacce 6

3.4 Design procedurale 6

4 Implementazione 7

5 Test 7

5.1 Protocollo di test 7

5.2 Risultati test 8

5.3 Mancanze/limitazioni conosciute 8

6 Consuntivo 8

7 Conclusioni 8

7.1 Sviluppi futuri 8

7.2 Considerazioni personali 8

8 Glossario 8

9 Bibliografia 9

9.1 Bibliografia per articoli di riviste: 9

9.2 Bibliografia per libri 9

9.3 Sitografia 9

10 Allegati 9

# Introduzione

## Informazioni sul progetto

Allievi:

* Nicolò Fadda
* Matteo Ambrosone
* Lorenzo Di Stefano

Docente:

* Geo Petrini

Scuola, sezione e materia:

* Centro Professionale Tecnico Trevano, Sezione SAM, INFORMATICA AZIENDALE, M306

Date Importanti:

* Data di Inizio: 12.01.2024
* Data di fine: 03.05.2024

## Abstract

*In the immersive gaming landscape, the project "Project: Non-Euclidian" presents a unique foray into non-Euclidean spatial exploration. The simulation creates a three-dimensional environment where illusionary portals serve as a central element, enabling instantaneous player movement between distinct sections of the virtual world. This abstract spatial concept is realized through viewports linked to dynamic cameras, enhancing the perception of a spatially unconventional realm.*

*Motivated by the desire to redefine traditional spatial logic in gaming, the project offers a novel experience where players control movement through W, A, S, D keys and navigate full-screen camera rotation using the mouse. The manual design of levels, coupled with strategic portal placements, ensures a captivating and immersive journey for players.*

*Developed exclusively for Windows, the project also provides customization options, allowing users to adjust screen resolution, music and effects volume, and remap controls according to personal preferences.*

## Scopo

Lo scopo del progetto "Project: Non-Euclidian" è fornire un'esperienza di gioco innovativa e coinvolgente attraverso l'esplorazione di uno spazio tridimensionale che sfida le leggi euclidee. L'obiettivo principale è creare un ambiente virtuale unico, dove i giocatori possono immergersi in un concetto di spazio distorto e fuori dagli schemi convenzionali.

Attraverso l'introduzione di portali illusori, il progetto mira a trasportare i giocatori in un mondo virtuale dove la percezione spaziale tradizionale viene deliberatamente alterata. L'illusione di uno spazio tridimensionale senza senso viene accentuata grazie all'utilizzo di viewports collegate a telecamere che seguono il movimento del giocatore, consentendo cambiamenti istantanei tra diverse sezioni del terreno attraverso i portali.

Il cuore del progetto risiede nell'offrire una nuova prospettiva di gioco, sfidante e coinvolgente. "Project: Non-Euclidian" incoraggia i giocatori a esplorare un mondo virtuale non euclideo, rompendo le convenzioni spaziali tradizionali e offrendo un'esperienza di gioco unica nel suo genere.

# Analisi

## Analisi del dominio

* Questo gioco è inteso per essere giocato da una fascia di età compresa tra i 8 e i 99+ anni.
* Il gioco è progettato per girare sulla piattaforma Windows 10-11 e Linux.

Attualmente esistono pochi prodotti simili, questa è una grande opportunità per far scoprire un gioco del genere a un ampio pubblico videoludico.

## Analisi e specifica dei requisiti

### Portali

|  |  |
| --- | --- |
| **ID: REQ-001** | |
| **Nome** | Viewports con telecamera |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Si necessitano viewport che renderizzino una telecamera |

|  |  |
| --- | --- |
| **ID: REQ-002** | |
| **Nome** | Telecamera che segue i movimenti del giocatore |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Si necessitana che la telecamera segua i movimenti del giocatore per dare l’impressione che non ci sia alcun portale |
| **Sotto requisiti** | |
| **001** | Viewports con telecamera |

### Audio

|  |  |
| --- | --- |
| **ID: REQ-003** | |
| **Nome** | Colonna sonora |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** | Il gioco deve avere una colonna sonora che giri un loop infinito per tutta la durata della partita |

|  |  |
| --- | --- |
| **ID: REQ-004** | |
| **Nome** | Effetti sonori |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** | Il gioco deve avere degli effetti sonori per passi e raccolta oggetti, di default avranno lo stesso livello di volume. |

### Gameplay

|  |  |
| --- | --- |
| **ID: REQ-005** | |
| **Nome** | Giocatore funzionante |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Il giocatore deve muoversi avanti, indietro, desta, sinistra (WASD), girare la telecamera e saltare.  Inoltre il movimento deve seguire la direzione della telecamera. |

|  |  |
| --- | --- |
| **ID: REQ-006** | |
| **Nome** | Livello di prova iniziale |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** | Il gioco deve contenere un livello iniziale che mostri all’utente in modo semplice il funzionamento del gioco |

|  |  |
| --- | --- |
| **ID: REQ-007** | |
| **Nome** | Menu di pausa |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Il giocatore deve avere la possibilità di mettere in pausa il gioco in ogni momento andando nel menu di pausa |

|  |  |
| --- | --- |
| **ID: REQ-008** | |
| **Nome** | Menu iniziale |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Il giocatore quando avvia il gioco deve vedere il menu iniziale con la possibilità di iniziare una nuova partita, continuarne una già iniziata o entrare nelle impostazioni |

|  |  |
| --- | --- |
| **ID: REQ-009** | |
| **Nome** | Raccogliere oggetti |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Il giocatore deve poter raccogliere degli oggetti contrassegnati da materiali definiti |

|  |  |
| --- | --- |
| **ID: REQ-010** | |
| **Nome** | Far cadere oggetti |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Il giocatore deve avere la possibilità di far cadere gli oggetti raccolti |

|  |  |
| --- | --- |
| **ID: REQ-011** | |
| **Nome** | Algoritmo di collegamenti randomici |
| **Priorità** | 4 |
| **Versione** | 1.0 |
| **Note** | Il gioco deve contenere un algoritmo che mischi i collegamenti tra i portali per disorientare l’utente |

|  |  |
| --- | --- |
| **ID: REQ-012** | |
| **Nome** | Salvare la partita |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** | Il giocatore deve avere la possibilità di salvare la partita in qualsiasi momento |

|  |  |
| --- | --- |
| **ID: REQ-021** | |
| **Nome** | Gestione gravità |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Il programma deve essere in grado di cambiare la direzione della spinta gravitazionale in modo da far accedere il giocatore a più percorsi |

#### Impostazioni di gioco

|  |  |
| --- | --- |
| **ID: REQ-013** | |
| **Nome** | Risoluzione |
| **Priorità** | 3 |
| **Versione** | 1.0 |
| **Note** | Il giocatore deve avere la possibilità di modificare la risoluzione della schermata di gioco |

|  |  |
| --- | --- |
| **ID: REQ-014** | |
| **Nome** | Volume audio ed effetti |
| **Priorità** | 3 |
| **Versione** | 1.0 |
| **Note** | Il giocatore deve avere la possibilità di modificare il volume della colonna sonora e degli effetti sonori separatamente |

|  |  |
| --- | --- |
| **ID: REQ-015** | |
| **Nome** | Layout tastiera |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** | Il giocatore deve avere la possibilità di reimpostare i tasti della tastiera |

#### Tutorial

|  |  |
| --- | --- |
| **ID: REQ-016** | |
| **Nome** | Spiegare obbiettivo |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Il gioco deve spiegare l’obbiettivo tramite una finestra di dialogo e a voce (in inglese) |

|  |  |
| --- | --- |
| **ID: REQ-017** | |
| **Nome** | Spiegare come si gioca |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** | Il gioco deve spiegare al giocatore come muoversi e girare la telecamera tramite un’immagine sul muro |

### Modelli 3D

|  |  |
| --- | --- |
| **ID: REQ-018** | |
| **Nome** | Set di statue |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Il gioco deve contenere dei modelli artistici che saranno gli obbiettivi da raccogliere |

|  |  |
| --- | --- |
| **ID: REQ-019** | |
| **Nome** | Set di stanze |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Il gioco deve contenere un set di stanze che saranno poi usate per creare la struttura |

|  |  |
| --- | --- |
| **ID: REQ-020** | |
| **Nome** | Collegamenti |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Il gioco deve contenere un set di collegamenti che saranno usati per collegare le stanze |

### Spiegazione elementi tabella dei requisiti:

**ID**: identificativo univoco del requisito

**Nome**: breve descrizione del requisito

**Priorità**: indica l’importanza di un requisito nell’insieme del progetto, definita assieme al committente. Ad esempio, poter disporre di report con colonne di colori diversi ha priorità minore rispetto al fatto di avere un database con gli elementi al suo interno. Solitamente si definiscono al massimo di 2-3 livelli di priorità.

**Versione**: indica la versione del requisito. Ogni modifica del requisito avrà una versione aggiornata.

Sulla documentazione apparirà solamente l’ultima versione, mentre le vecchie dovranno essere inserite nei diari.

**Note**: eventuali osservazioni importanti o riferimenti ad altri requisiti.

## Use case

## Pianificazione

|  |
| --- |
| Figura 1: Gannt preventivo |

### Milestones

|  |  |  |
| --- | --- | --- |
| Nome | Scadenza | Descrizione |
| Portale Funzionante | 23.02.2024 | Si necessita per la scadenza un prototipo di portale funzionante |
| Strutture terminate | 15.03.2024 | Per la scadenza si necessita un prototipo di mappa generato a mano privo di portali |
| Portale pronto per il gioco | 29.03.2024 | Si necessita per la scadenza di un portale funzionante adatto alla mappa |

## Analisi dei mezzi

Per la realizzazione del progetto “Project: Non-Euclidian”, sono stati impiegati diversi mezzi tra software e hardware per garantire un ambiente di sviluppo adeguato ed efficiente.

Per la pianificazione è stato scelto lo schema Agile scrumban.

### Software

Godot v4.1.3: Game engine free e open source per lo sviluppo di giochi in 2D e 3D

GDscript: Linguaggio basato su python per lo sviluppo di giochi in godot

GitHub (Desktop & Web)

### Hardware

Hardware per sviluppo: 3 PC desktop HP elite desk Windows 10

Specifiche:

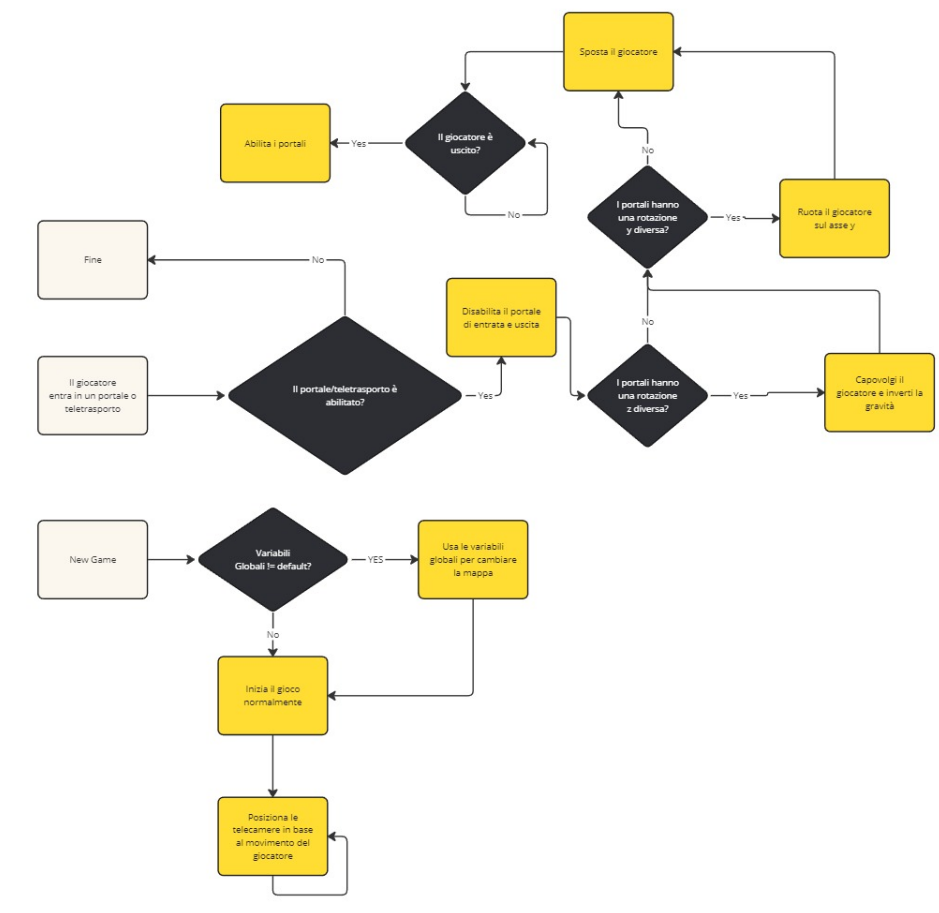
* Processore Intel(R) Core(TM) i7-9700 CPU @ 3.00GHz 3.00 GHz
* 32GB RAM
* Sistema operativo a 64 bit

# Progettazione

## Design delle interfacce

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

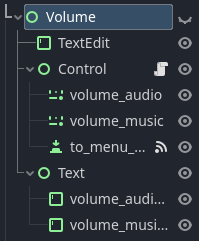
## Design procedurale



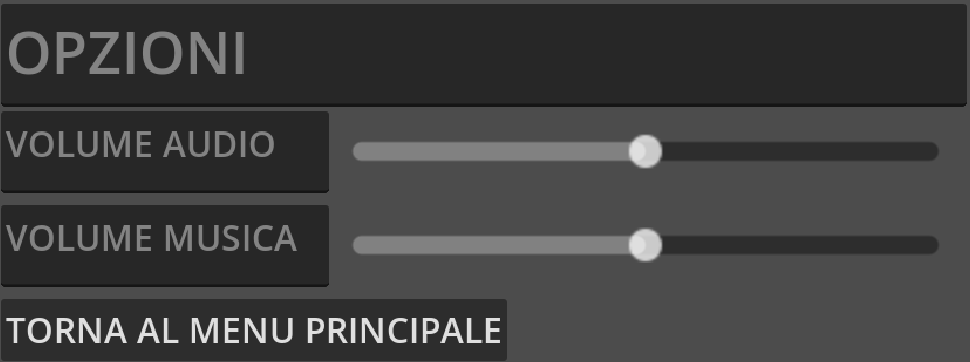
# Implementazione

## Impostazioni Utente

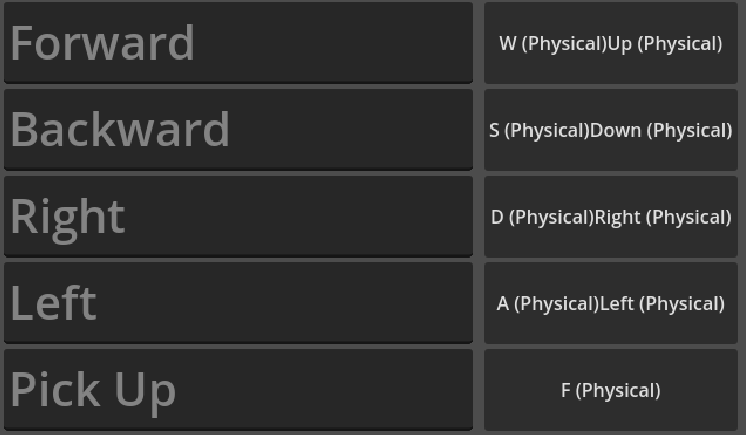
### Volume

Il Volume dell’audio è gestito nel MainMenu e nel menù opzioni durante il gioco, tramite dei nodi Hslider vanno a modificare la variabile globale del volume della musica e degli effetti sonori come la voce del tutorial, dei passi e della raccolta degli oggetti.

La musica di sottofondo è un nodo figlio del nodo padre start (è la radice della scena), mentre il suono dei passi (sound\_player), quello della raccolta dei sassi (pickup\_sound) e della voce del tutorial (tutorial\_voice) vengono aggiunti come nodi figlio del giocatore e avviati quando necessario (es il pickup sound viene avviato quando il giocatore raccoglie un oggetto)



### Layout tastiera



Il layout della tastiera viene modificato dall’utente quando clicca su uno dei buttoni, i buttoni richiamano il metodo \_on\_key\_pressed() nello script Keys.gd che assegna l’input selected, in seguito il programma ascolta gli input del utente,

se l’utente dovesse cliccare un tasto della tastiera la funzione \_input lo catturerebbe aggiornando l’InputMap con il tasto premuto dal utente

#When a button is clicked it updates the input\_select variable

#with the InputMap action that must be updated

func \_on\_key\_pressed(input: String):

    input\_selected = input

#when the player has clicked a button and pressed a key it updates the InputMap

#and resets the input\_selected variable

func \_input(ev):

    if input\_selected!="null" and not ev is InputEventMouse:

        InputMap.action\_erase\_events(input\_selected)

        InputMap.action\_add\_event(input\_selected, ev)

        var search = input\_selected.substr(0,1).to\_upper()+input\_selected.substr(1,len(input\_selected))+"\_button"

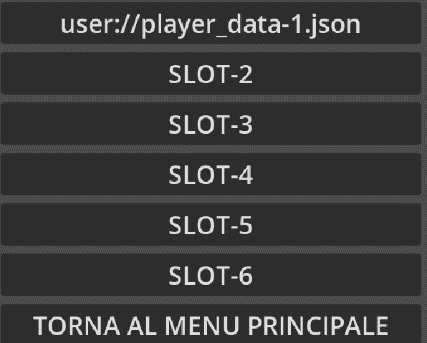
        get\_node("Keys/VBoxContainer/"+search).text = ev.as\_text()

        input\_selected = "null"

### Risoluzione

Il giocatore può cambiare la risoluzione della finestra dalle impostazioni del menu, il codice viene eseguito quando l’optionButton cambia item e setta la grandezza x e y secondo l’item selezionato.

### Caricamento partita

Quando l’utente inizia il gioco, quest’ultimo cerca nella cartella %AppData% se esistono dei salvataggi e, se così, stampa all’interno della finestra di caricamento partita (come nella foto di esempio).

Quando l’utente clicca uno slot contenente una partita aggiorna le variabili globali secondo il json caricato, starà poi allo script start.gd modificare i nodi della mappa per rispecchiare la partita salvata in precedenza (il gioco non salva i collegamenti tra i portali, vengono rigenerati ogni volta che si ricarica la partita)

Global.gd

# Player scores

var numero\_sassi := 20

var sassi\_posionati := []

var statue\_posizionate := []

Esempio di salvataggio

{

    "graviy\_direction":"upwards",

    "numero\_sassi":20,

    "player\_position":"(-49, 29.951, -91.97)"

    ,"player\_rotation":"(0, 0, 0)",

    "rock\_picked":{},

    "sassi":[],

    "saved\_date":{"day":12,"month":4,"weekday":5,"year":2024},"statues\_captured":{}

}

## Gameplay

### Movimento

Il movimento del giocatore viene gestito in Player.gd, lo script prende gli input da Input.get\_vector() che tramite la mappatura dei tasti in InputMap capisce in che direzione sta andando. In seguito usa il basis del twistpivot moltiplicato per il proprio per capire in che direzione sta andando e tramite quello ottiene il vettore x e z che fa muovere il giocatore nella mappa

    # Get the input direction and handle the movement/deceleration.

    # As good practice, you should replace UI actions with custom gameplay actions.

    var input\_dir = Input.get\_vector("left", "right", "forward", "backward")

    #the variable gravitation changes the way the player moves horizontally when he's upsidedown

    var direction = (twist\_pivot.basis \* transform.basis \* Vector3(input\_dir.x\*gravitation, 0, input\_dir.y)).normalized()

    if direction:

        velocity.x = direction.x \* SPEED

        velocity.z = direction.z \* SPEED

    else:

        if is\_on\_floor():

            sound\_player.play()

        velocity.x = move\_toward(velocity.x, 0, SPEED)

        velocity.z = move\_toward(velocity.z, 0, SPEED)

    move\_and\_slide()

#### Salto

Il salto viene eseguito se il giocatore preme spazio, quello che fa è settare la velocità y alla JUMP\_VELOCITY che verrà poi diminuita dalla gravita creando un moto parabolico (se il giocatore è capovolto la gravità rimane comunque assoluta e quindi JUMP\_VELOCITY dev’essere negativa)

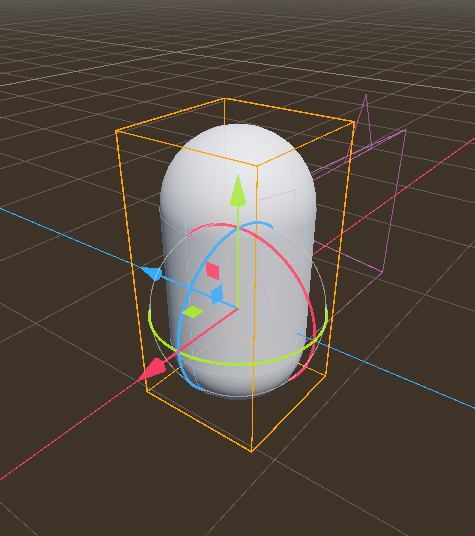
    if Input.is\_key\_pressed(KEY\_SPACE) and ((is\_on\_floor() and gravitation==1) or (is\_on\_ceiling() and gravitation==-1)):

        velocity.y = JUMP\_VELOCITY\*gravitation

        print("is on floor and jumps")

Player.gd

### Rotazione telecamera

La rotazione della telecamera avviene tramite 2 nodi figli del nodo player: PitchPivot e TwistPivot.

Il TwistPivot gestisce la rotazione della telecamera sull’asse y (verticale) quindi sposta la telecamera del giocatore a destra e a sinistra, mentre il PitchPivot gestisce la rotazione della telecamera dall’alto verso il basso. Il motivo per cui non ruotiamo direttamente la telecamera è perché se dovessimo ruotarla sull’asse y la rotazione sull’asse x rimarrebbe assoluta, ruotando la visuale dell’utente invece di spostarla.

    twist\_pivot.rotate\_y(twist\_input)

    pitch\_pivot.rotate\_x(pitch\_input)

    pitch\_pivot.rotation.x = clamp(pitch\_pivot.rotation.x, deg\_to\_rad(-80),deg\_to\_rad(80))

    twist\_input = 0.0 #it stops the camera from slipping

    pitch\_input = 0.0

func \_unhandled\_input(event: InputEvent) -> void:

    if event is InputEventMouseMotion:

        if Input.get\_mouse\_mode() == Input.MOUSE\_MODE\_CAPTURED:

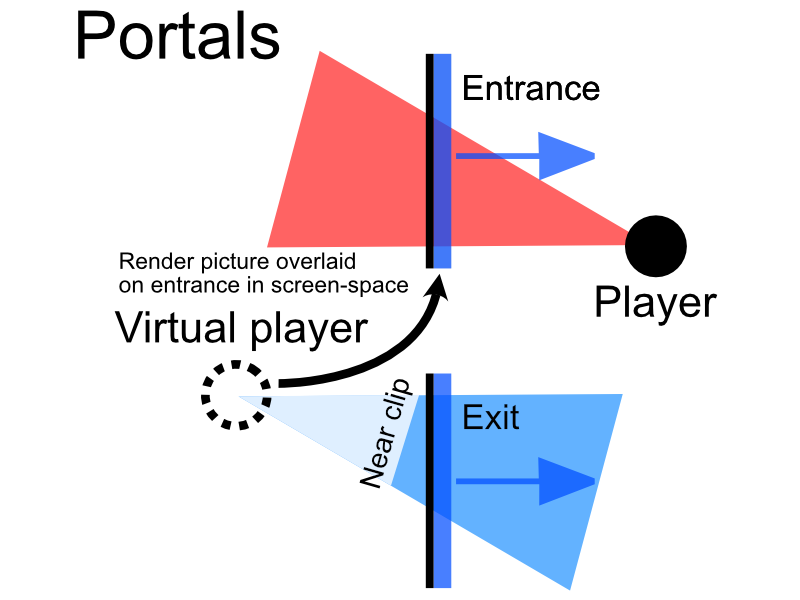
            twist\_input = - event.relative.x \* sensivity \* gravitation

            pitch\_input = - event.relative.y \* sensivity \* gravitation

Player.gd, rotazione telecamera

### Portali

I portali sono delle Mesh in 2D che proiettano su di loro un viewport, i viewport che usano sono collegate a delle telecamere che funzionano come dei giocatori virtuali, questi giocatori virtuali replicano il movimento del giocatore rispetto al portale a qui sono collegati, creando così un’illusione.

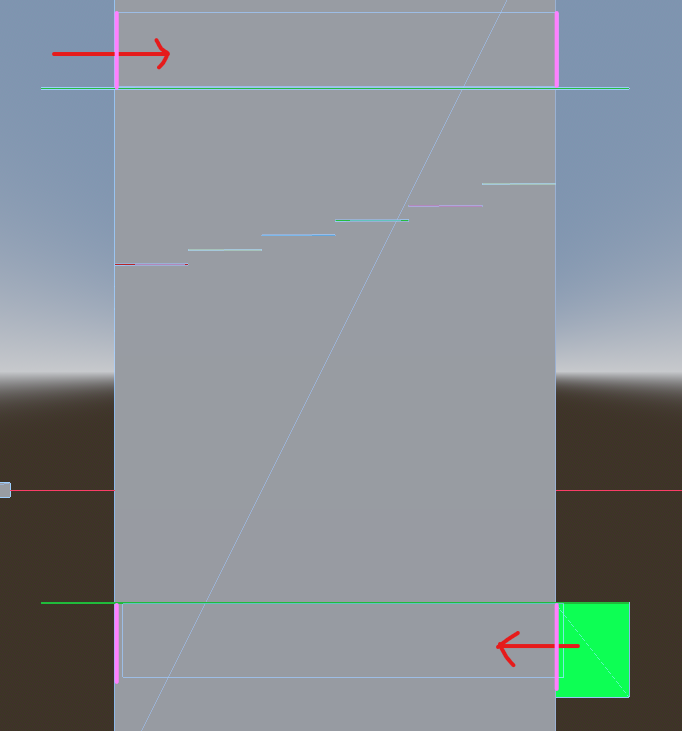
Quando il giocatore entra viene portato al portale collegato al portale in qui è entrato, questa informazione è

Contenuta all’interno dello script dei portali.

### Teletrasporti

I teletrasporti hanno la stessa funzione dei portali ossia portare il giocatore da A a B, ma i teletrasporti vengono implementati solamente quando il giocatore in A si trova in un ambiente simile a quello in qui si ritroverebbe al punto B (es se A è un corridoio e B è un corridoio uguale ad A finché si mantengono le proporzioni e le distanze il teletrasporto può essere applicato senza farsi notare dall’utente)

Esempio di situazione

In questa situazione sia sopra che sotto il viewport della telecamera del giocatore non cambierà affinché si mantiene la stessa distanza dal limite del portale (segnato in rosa), ovviamente devono essere effettuati dei controlli che accertano che il giocatore guardi in una certa direzione per evitare di rompere l’illusione

### Sassi

I sassi all’interno del gioco sono necessari a creare dei punti di riferimento, per facilitare il movimento del giocatore aiutandolo ad orientarsi. Questi possono essere posizionati e raccolti nel caso si desiderasse spostarli.

È possibile posizionare i sassi nella mappa, premendo il tasto **[Q]** sulla tastiera.

Il giocatore ha a disposizione un numero limitato di sassi. In partenza verranno forniti 20 sassi da posizionare a piacimento.

Avvicinandosi ad un sasso e guardando verso di esso, se il giocatore sarà abbastanza vicino, il sasso si colorerà di blu e apparirà a schermo la scritta “Press F to pickup”, adesso premendo il tasto **[F]**, come suggerito, il sasso verrà rimosso dalla scena e tornerà disponibile nell’inventario.

Immagine che contiene testo, logo, Elementi grafici, schermata

Descrizione generata automaticamente

I sassi sono dei mesh cubici di colore rosso, il colore di questo tipo di oggetti è necessario a contraddistinguere i mesh dal resto della mappa, essendo questo un colore che all’interno del gioco è associato agli oggetti interagibili.

Il metodo necessario alla generazione dei mesh rappresentanti i sassi è presente nello script **Global.gd**, esso è denominato **create\_stone()**.

Questo metodo ha come parametro la posizione in **Vector3** di dove posizionare il mesh.

Il body generato è un oggetto fisico (con collisione) di dimensioni ridotte.

|  |
| --- |
| func create\_stone(position):      var body = RigidBody3D.new()      var mesh = BoxMesh.new()      var particle = MeshInstance3D.new()      var collision = CollisionShape3D.new()      collision.scale\_object\_local(Vector3(0.1, 0.1, 0.1))      var new\_material = StandardMaterial3D.new()      new\_material.albedo\_color = Color(1, 0, 0)        # Imposta il colore rosso      mesh.material = new\_material      mesh.size = Vector3(0.1, 0.1, 0.1) # Modifica le dimensioni lungo gli assi      particle.mesh =mesh      collision.shape = BoxShape3D.new()      body.add\_child(particle)      body.add\_child(collision)      body.position=position      sassi\_posionati.append(position)   # Salva la posizione del sasso in un array      start.add\_child(body) |

La posizione del body viene salvata nell’array contenete le posizioni di tutti i sassi posizionati.

Infine il body viene inserito nella scena aggiungendolo come figli al Node3D start.

Il sasso viene spawnato nella posizione nella quale il player guarda.

La posizione di spawn del sasso viene ottenuta grazie al Raycast3D che permette di risalire all’oggetto presente nella direzione dello sguardo del player e alla posizione di impatto con il Raycast3D con questo.

Immagine che contiene schermata, Modellazione 3D

Descrizione generata automaticamente

Il Raycast3D è presente nel player come nodo figlio del PitchPivot, questo permette al raggio del Raycast3D di muoversi e ruotare assieme alla visuale del player.

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Il sasso verrà posizionato solo nel caso in cui la distanza tra il player e la presunta posizione di spawn fosse inferiore a 4.

Questa distanza viene verificata grazie al seguente metodo.

|  |
| --- |
| # Ritorna true se la distanza tra i 2 vettori è minore di max\_distance  func is\_near(pos,max\_distance):      var distance = calculate\_distance(pos,player.position)      if distance < max\_distance:          return true      return false |

Il metodo è presente nello script del player.

Questo verifica la distanza tra il vettore della posizione del player e il vettore passato come parametro, per fare ciò viene utilizzato il metodo **calculate\_distance()** che ritorna la distanza tra i 2 vettori passati come parametro.

|  |
| --- |
| # Ritorna la distanza tra i due vettori  func calculate\_distance(vector1: Vector3, vector2: Vector3) -> float:      return vector1.distance\_to(vector2) |

All’interno dello script del player è presente il metodo **place\_stone()**, questo serve a posizionare i sassi allo scatenarsi dell’evento **“drop”**.

L’evento **“drop”** si scatena alla pressione del tasto [**Q]**.



Nello script del player è presente anche il metodo \_input(event) che viene chiamato allo scatenarsi di un evento.

All’interno di questo è stato aggiunto una condizione che chiama il metodo place\_stone nel caso fosse scatenato l’evento “drop”.

|  |
| --- |
| func \_input(event):      # Controlla che sia stato premuto il tasto per droppare      if Input.is\_action\_just\_pressed("drop"):          place\_stone() |

Il metodo **place\_stone()**, prima di posizionare il sasso verifica che il numero di sassi presenti nell’inventario sia maggiore di 0. Nel caso il numero di questi sia effettivamente maggiore di 0, ottiene la posizione nella quale spawnare il sasso grazie al **Raycast3D** e nel caso questa posizione distasse meno di 4 dal vettore posizione del player, modificherebbe il numero di questi, sottraendolo di 1 e chiamerebbe il metodo **create\_stone()** dallo script Global usando la posizione ottenuta come parametro.

|  |
| --- |
| func place\_stone():      if(Global.stones\_number>0):          var pos = raycast.get\_collision\_point()          if is\_near(pos,4):  Global.stones\_number = Global.stones\_number - 1              Global.create\_stone(pos) |

### Statue

Il codice per gestire la raccolta delle statue è contenuto nello script **playerRayCast3D.gd**. In parallelo all’if che controlla la selezione dei sassi da parte del RayCast3D, è presente anche un secondo if che verifica che la selezione delle statue.

|  |
| --- |
| # Raccolta Statua  if get\_collider().name.contains("Body"):      if get\_collider().get\_parent().get\_parent().name.contains("statua"):          set\_message\_visibility(true)          pickup\_statua(get\_collider().get\_parent().get\_parent().get\_parent())      else:          set\_message\_visibility(false) |

Nel caso il controllo effettuato dall’if che verifica la collisione con il RayCast3D con una statua restituisca true, viene richiamato il metodo **pickup\_statua()** che rimuove la statua selezionata dalla scena start e aggiunge alla variabile di tipo array **Global.collected\_statues** il valore corrispondente al numero presente nel nome della statua (es: la statua si chiama “statua-1”).

|  |
| --- |
| func pickup\_statua(body):      if Input.is\_action\_just\_pressed("pickup"):          Global.start.remove\_child(body)          var statue = body.get\_child(0).name.split("statua-")          for value in statue:              if len(value)>0:                  Global.collected\_statues.append(value) |

## Funzionalità

### Esci

Quando il giocatore esce dal menu cliccando il pulsante esci, il gioco richiama la funzione exit() dello script Menu.gd che ferma l’intero processo dalla radice.

func exit():

    get\_tree().quit()

### Menu di pausa

Quando il giocatore attiva l’azione esc del InputMap il gioco mette il processo principale in pausa il gioco. Tutti i nodi 3D essendo che ereditano il processo principale non reagiranno più mentre le interfacce e la musica di sottofondo non ereditano nessun processo per questo continueranno a funzionare.

### Salvataggio Partita

Quando il giocatore clicca uno slot di partita dal menu di pausa richiama il metodo **on\_slot\_button\_pressed** dello script player.gd; quello che fa è salvare un file json nella directory %AppData% con le variabili globali di partita come posizione del giocatore, se è capovolto, le statue che ha raccolto e i sassi che ha lasciato in giro.

Le statue essendo fisse e non potendole spostare possono essere salvate come nome che verrà poi usato per capire quali nodi eliminare dal contenitore Statua.

Invece i sassi essendo che non hanno una posizione fissa bisogna salvare la posizione in modo che al rientro del giocatore siano allo stesso posto in cui li ha posizionati.

# Test

## Protocollo di test

Definire in modo accurato tutti i test che devono essere realizzati per garantire l’adempimento delle richieste formulate nei requisiti. I test fungono da garanzia di qualità del prodotto. Ogni test deve essere ripetibile alle stesse condizioni.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-001  REQ-006 | **Nome:** | Livello di prova iniziale |
| **Descrizione:** | Al primo avvio di una partita, il gioco partirà dal livello di prova iniziale. | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Avviare il gioco 2. Cliccare su “Nuova Partita” 3. Premere il tasto Invio | | |
| **Risultati attesi:** | Il gioco ci proietta nel primo livello di prova | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-002  REQ-016 | **Nome:** | Spiegazione dell’obiettivo del gioco |
| **Descrizione:** | Al primo avvio di una partita, il gioco partirà dal livello di prova iniziale dove comparirà una finestra di dialogo che spiegherà l’obiettivo sia a voce che scritto. | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Avviare il gioco 2. Cliccare su “Nuova Partita” | | |
| **Risultati attesi:** | Il gioco ci proietta nel primo livello di prova ed esce la finestra di dialogo con l’obiettivo assieme alla spiegazione a voce in inglese | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-003  REQ-017 | **Nome:** | Spiegazione di come si gioca |
| **Descrizione:** | Al primo avvio di una partita, il gioco partirà dal livello di prova iniziale dove comparirà un’immagine a muro che spiegherà i comandi di movimento principali. | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Avviare il gioco 2. Cliccare su “Nuova Partita” 3. Premere Enter | | |
| **Risultati attesi:** | Il gioco ci proietta nel primo livello di prova e una volta premuto enter vedremo davanti a noi un’immagine con i principali comandi di movimento. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-004  REQ-008 | **Nome:** | Menu’ Iniziale |
| **Descrizione:** | All’avvio del gioco l’utente si trova un menu’ dove può selezionare varie opzioni. | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Avviare il gioco | | |
| **Risultati attesi:** | Il gioco si presenta con un menu’ iniziale con alcune opzioni principali. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-005  REQ-007 | **Nome:** | Menu’ di pausa |
| **Descrizione:** | All’avvio della partita l’utente deve poter mettere in pausa il gioco in qualsiasi momento del gioco. | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Avviare il gioco 2. Cliccare su “Nuova Partita” 3. Premere Enter 4. Premere Esc | | |
| **Risultati attesi:** | Il gioco va in pausa, bloccando i movimenti delle cose attorno al giocatore e permette di scegliere alcune opzioni. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-006  REQ-005 | **Nome:** | Comandi giocatore funzionanti |
| **Descrizione:** | All’avvio della partita l’utente deve poter muoversi avanti, indietro, desta, sinistra (WASD), girare la telecamera e saltare.  Inoltre il movimento deve seguire la direzione della telecamera. | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Avviare il gioco 2. Cliccare su “Nuova Partita” 3. Premere Enter 4. Provare tutti I tasti (W-A-S-D-Spazio) e muovere la telecamera (Mouse) | | |
| **Risultati attesi:** | Il giocatore riesce a fare tutti I movimenti di base. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-007  REQ-001 | **Nome:** | Viewports telecamera |
| **Descrizione:** | I portali utilizzati per teletrasportarsi da una parte all’altra devono renderizzare una telecamera per fare in modo che si veda dall’altra parte del portale. | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Avviare il gioco 2. Cliccare su “Nuova Partita” 3. Premere Enter 4. Controllare che il viewport sia funzionante | | |
| **Risultati attesi:** | Il giocatore riesce a vedere dall’altra parte del portale cosa lo aspetta correttamente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-008  REQ-002 | **Nome:** | Telecamera del giocatore |
| **Descrizione:** | Si necessita che la telecamera segua i movimenti del giocatore per dare l’impressione che non ci sia alcun portale (il giocatore vede come se fosse solo una “porta”) | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Avviare il gioco 2. Cliccare su “Nuova Partita” 3. Premere Enter 4. Controllare che la telecamera proietti correttamente l’altro lato del portale. | | |
| **Risultati attesi:** | Il giocatore riesce a vedere dall’altra parte del portale cosa lo aspetta correttamente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-009  REQ-009 | **Nome:** | Raccolta di oggetti |
| **Descrizione:** | Il giocatore deve poter raccogliere le statue / oggetti correttamente tramite l’utilizzo del tasto “F” | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Avviare il gioco 2. Cliccare su “Nuova Partita” 3. Premere Enter 4. Avvicinarsi ad un oggetto / statua raccoglibile 5. Premere il tasto “F” | | |
| **Risultati attesi:** | Il giocatore raccoglie senza problemi l’oggetto facendolo scomparire. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-010  REQ-010 | **Nome:** | Drop di oggetti |
| **Descrizione:** | Il giocatore deve poter far cadere gli oggetti correttamente tramite l’utilizzo del tasto “Q” | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Avviare il gioco 2. Cliccare su “Nuova Partita” 3. Premere Enter 4. Premere il tasto “Q” | | |
| **Risultati attesi:** | Il giocatore fa cadere un oggetto. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-011  REQ-012 | **Nome:** | Salvataggio partita |
| **Descrizione:** | Il giocatore deve poter salvare i suoi progressi per fare in modo che una volta uscito dal gioco e rientrato possa continuare la sua partita | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Avviare il gioco 2. Cliccare su “Nuova Partita” 3. Premere Enter 4. Giocare (muoversi, raccogliere / lasciare oggetti, ...) 5. Premere Esc 6. Cliccare su “Salva” 7. Scegliere uno slot a scelta e cliccarci sopra 8. Cliccare su “Torna al menu principale” 9. Cliccare su “Esci” 10. Cliccare su “Carica” | | |
| **Risultati attesi:** | Il giocatore carica correttamente la partita precedentemente salvata. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-012  REQ-003 | **Nome:** | Colonna sonora |
| **Descrizione:** | Il giocatore all’avvio della partita dovrà sentire una colonna sonora per tutta la durata del gioco. | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Avviare il gioco 2. Cliccare su “Nuova Partita” 3. Premere Enter 4. Giocare (muoversi, raccogliere / lasciare oggetti, ...) 5. Ascoltare la colonna sonora abilitando l’audio | | |
| **Risultati attesi:** | Il giocatore sente la colonna sonora. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-013  REQ-004 | **Nome:** | Effetti sonori |
| **Descrizione:** | Il giocatore dovrà sentire degli effetti sonori come i passi, la raccolta di oggetti, … | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Avviare il gioco 2. Cliccare su “Nuova Partita” 3. Premere Enter 4. Giocare (muoversi, raccogliere / lasciare oggetti, ...) 5. Ascoltare gli effetti abilitando l’audio | | |
| **Risultati attesi:** | Il giocatore sente la colonna sonora. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-014  REQ-011 | **Nome:** | Algoritmo di collegamento |
| **Descrizione:** | Il giocatore quando attraverserà svariati portali, essi dovranno teletrasportarlo in altri portali randomicamente (una volta teletrasportato, quei portali saranno collegati tra loro e non più randomici) | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Avviare il gioco 2. Cliccare su “Nuova Partita” 3. Premere Enter 4. Attraversare un portale | | |
| **Risultati attesi:** | Il giocatore attraversa il portale e viene teletrasportato correttamente. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-015  REQ-021 | **Nome:** | Gestione gravità |
| **Descrizione:** | Il giocatore quando attraverserà un portale che lo porterà sottosopra, la gravità dovrà tenere “incollato” il giocatore al pavimento (soffitto) | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Avviare il gioco 2. Cliccare su “Nuova Partita” 3. Premere Enter 4. Attraversare un portale che fa andare sottosopra | | |
| **Risultati attesi:** | Il giocatore attraversa il portale e viene teletrasportato correttamente, rimanendo attaccato al pavimento / soffitto. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-016  REQ-013 | **Nome:** | Risoluzione schermo |
| **Descrizione:** | Il giocatore deve poter modificare la risoluzione dello schermo a proprio piacimento dal menu delle impostazioni | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Avviare il gioco 2. Andare nelle impostazioni 3. Cliccare su “risoluzione” 4. Modificare la risoluzione | | |
| **Risultati attesi:** | Il giocatore modifica correttamente a proprio piacimento la risoluzione del gioco. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-017  REQ-014 | **Nome:** | Volume audio |
| **Descrizione:** | Il giocatore deve poter modificare il volume del gioco in qualsiasi momento tramite uno slider. | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Avviare il gioco 2. Andare nelle impostazioni 3. Cliccare su “Opzioni” 4. Modificare il volume | | |
| **Risultati attesi:** | Il giocatore modifica correttamente a proprio piacimento il volume del gioco. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-017  REQ-015 | **Nome:** | Layout tastiera |
| **Descrizione:** | Il giocatore deve poter modificare i tasti con cui gioca a proprio piacimento. | | |
| **Prerequisiti:** | - | | |
| **Procedura:** | 1. Avviare il gioco 2. Andare nelle impostazioni 3. Cliccare su “Opzioni” 4. Modificare i propri tasti | | |
| **Risultati attesi:** | Il giocatore modifica correttamente a proprio piacimento i tasti con cui giocare. | | |

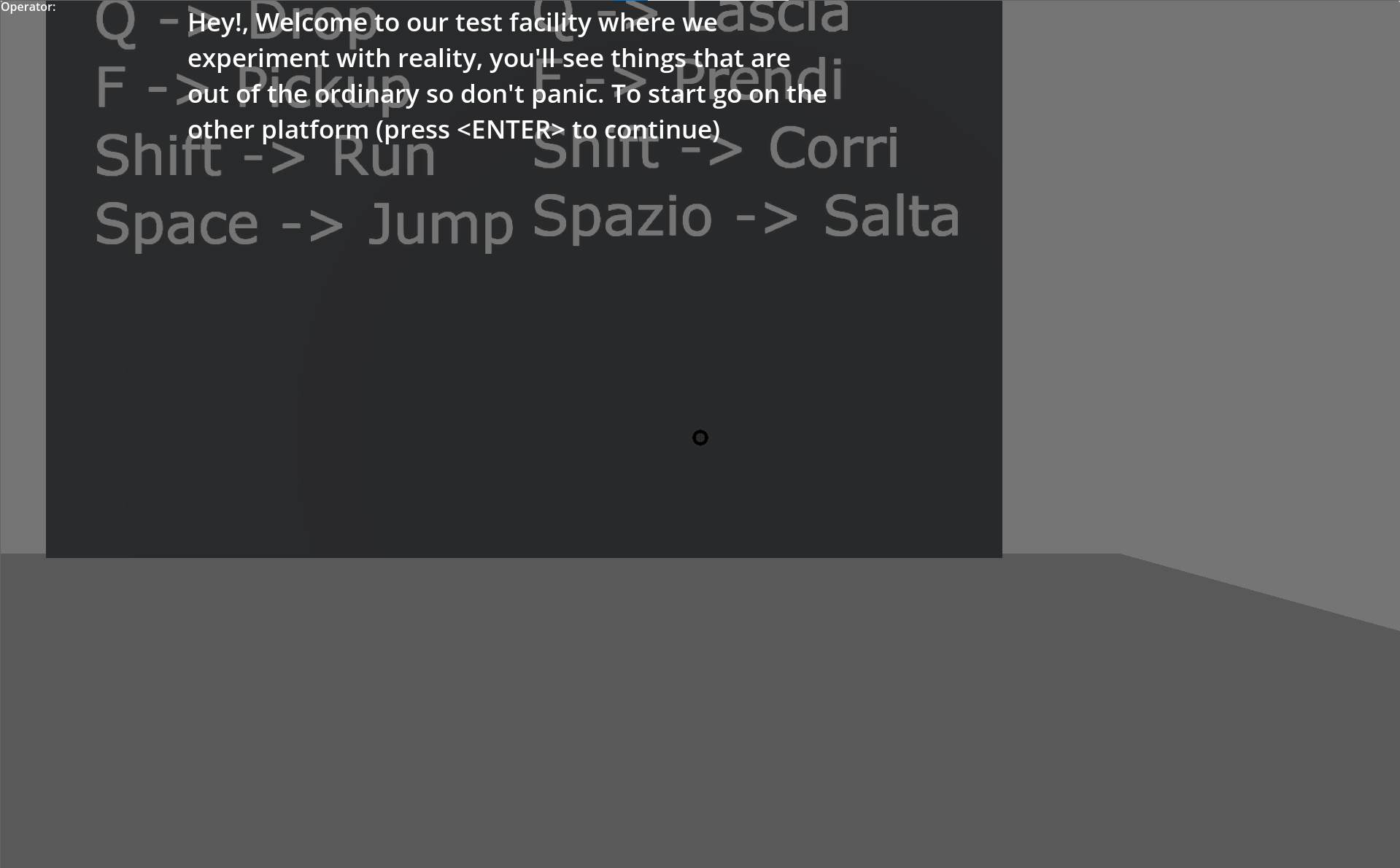
## Risultati test

|  |  |
| --- | --- |
| Test Case | Esito |
| TC-001 | PASSATO |
| TC-002 | PASSATO |
| TC-003 | PASSATO |
| TC-004 | PASSATO |
| TC-005 | PASSATO |
| TC-006 | PASSATO |
| TC-007 | FALLITO |
| TC-008 | FALLITO |
| TC-009 | PASSATO |
| TC-010 | PASSATO |
| TC-011 | PASSATO |
| TC-012 | PASSATO |
| TC-013 | PASSATO |
| TC-014 | PASSATO |
| TC-015 | PASSATO |
| TC-016 | PASSATO |
| TC-017 | PASSATO |
| TC-018 | PASSATO |

### TC-001

Esito: Passato

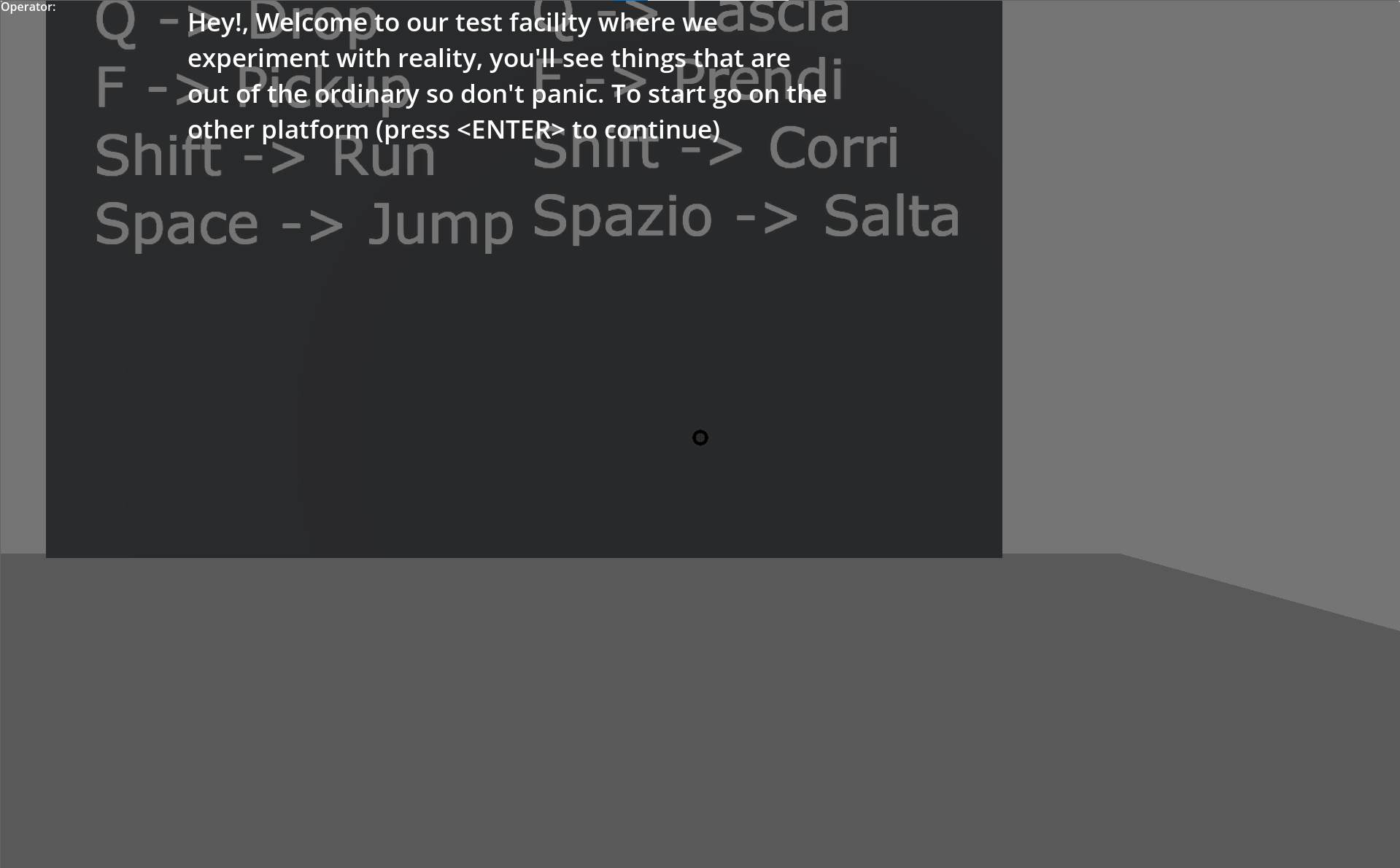
Per questo test bisognava soltanto fare in modo che il player parta dal livello di prova al primo avvio di una nuova partita.



### TC-002

Esito: Passato

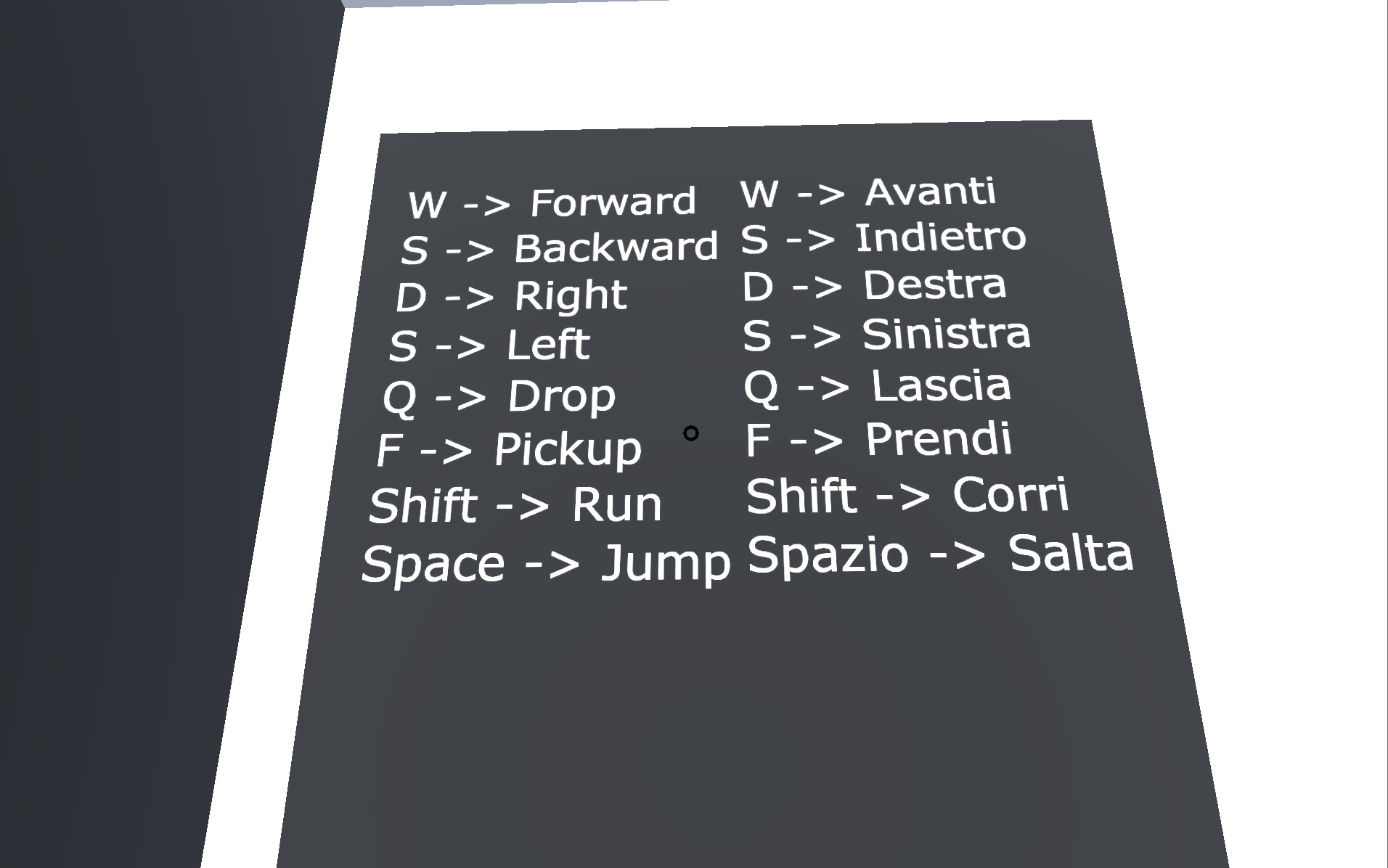
Per questo test bisognava fare in modo che ci sia una spiegazione sia a voce che scritta del gioco.



### TC-003

Esito: Passato

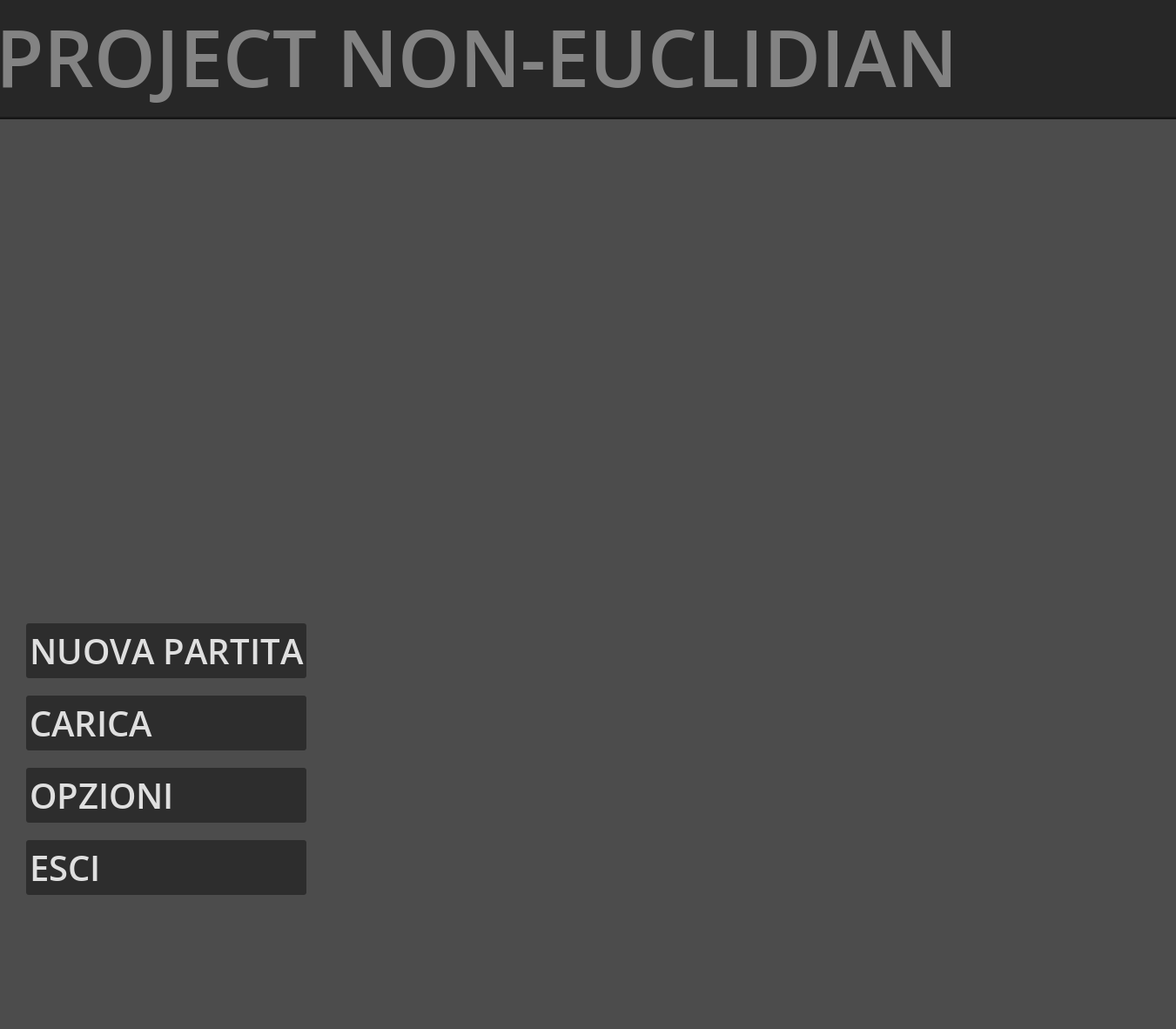
Per questo test bisognava creare un immagine a muro che spieghi i comandi del gioco.



### TC-004

Esito: Passato

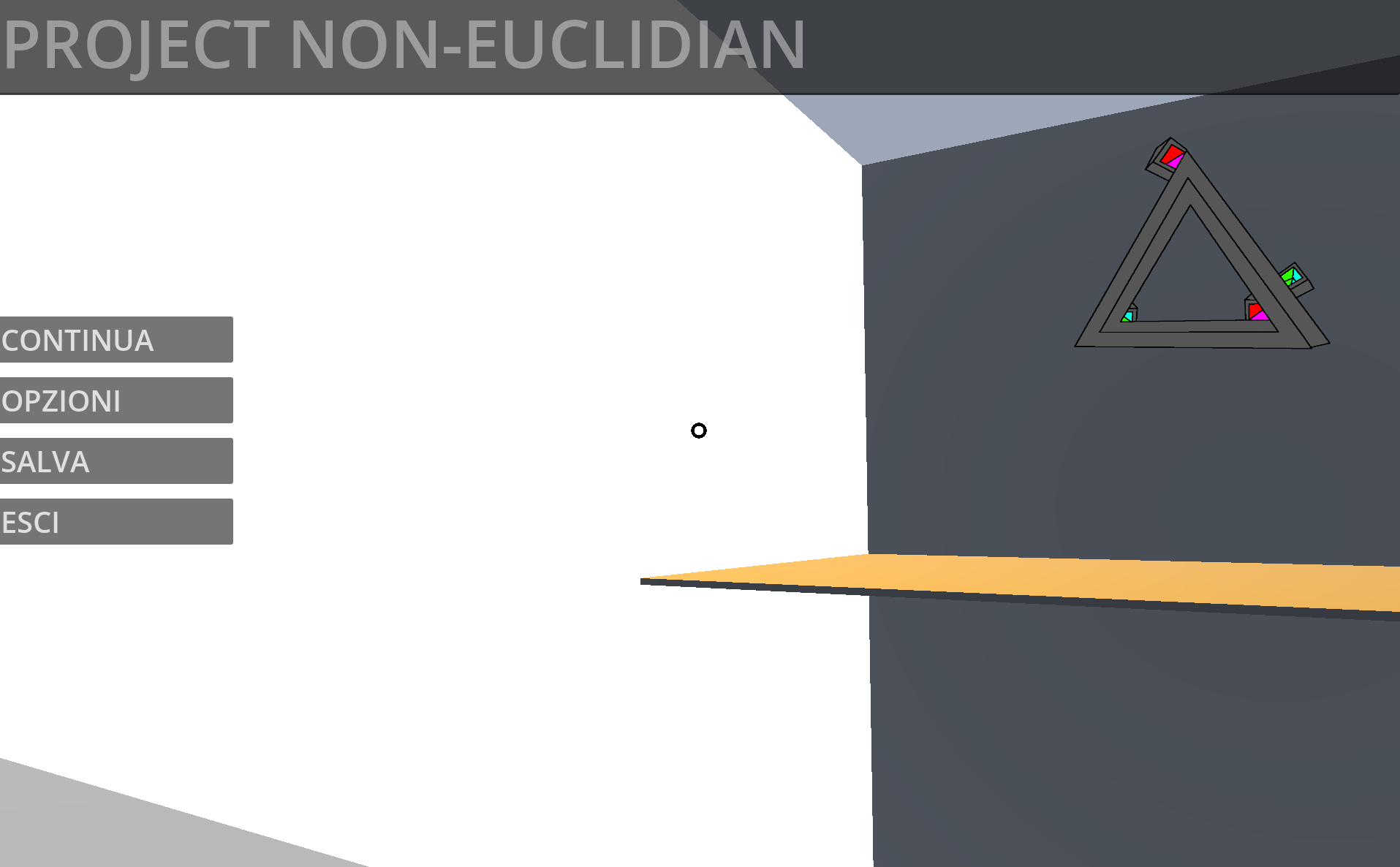
All’avvio del gioco l’utente dovrà avere un menu’ dove si possono selezionare varie opzioni.



### TC-005

Esito: Passato

L’utente può mettere in pausa il gioco in qualsiasi momento.

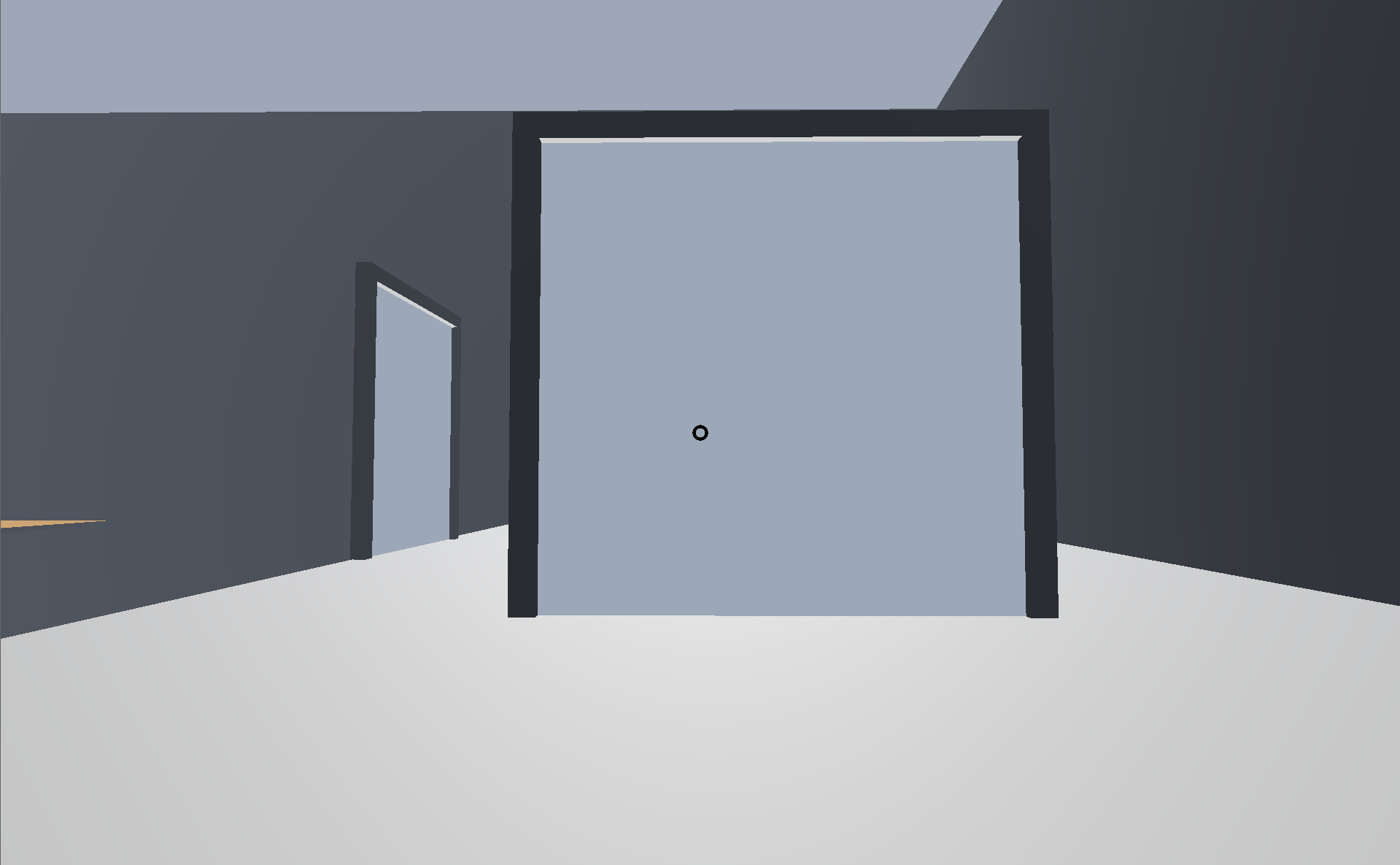
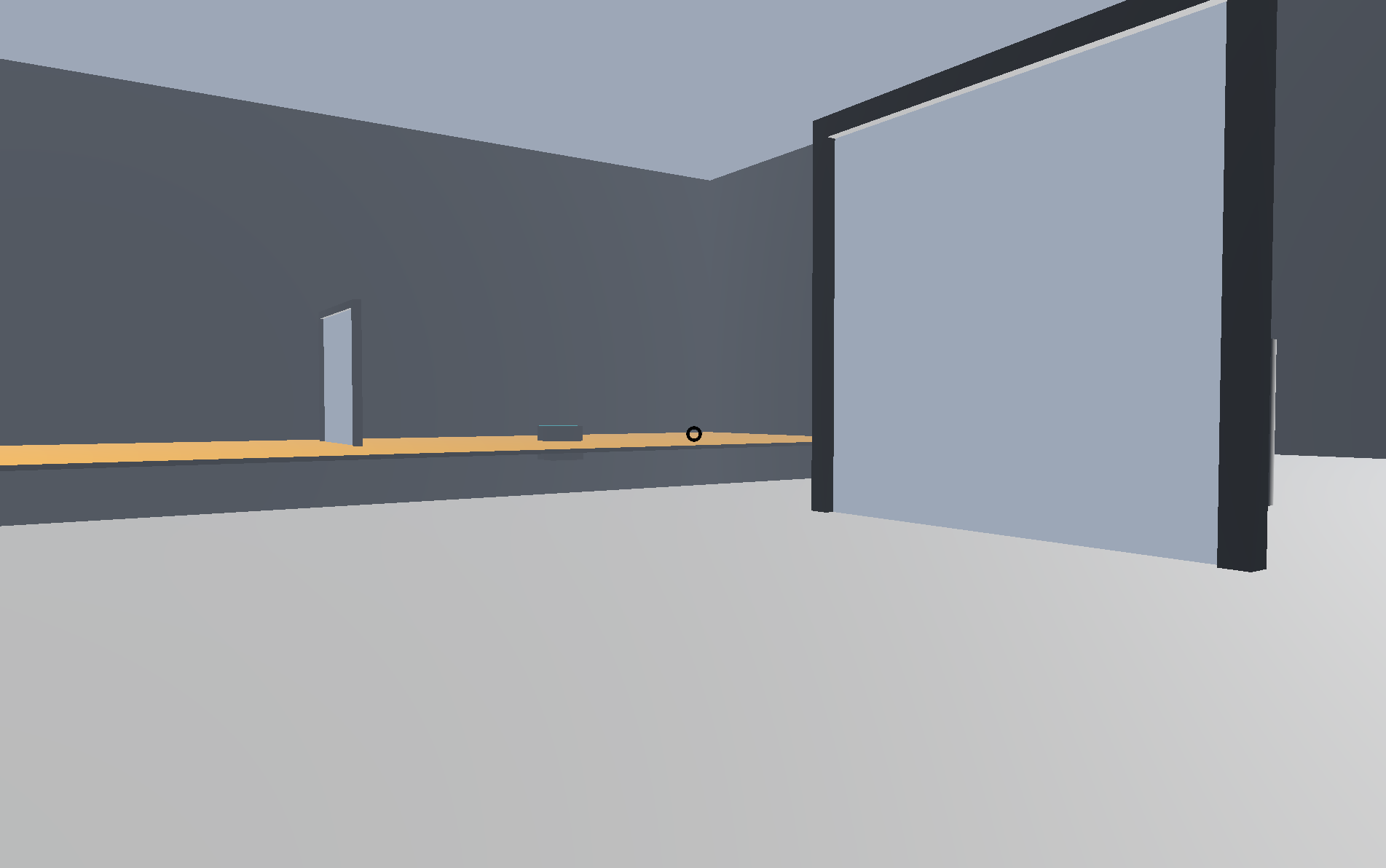


### TC-006

Esito: Passato

Il giocatore può muoversi nelle varie direzioni.

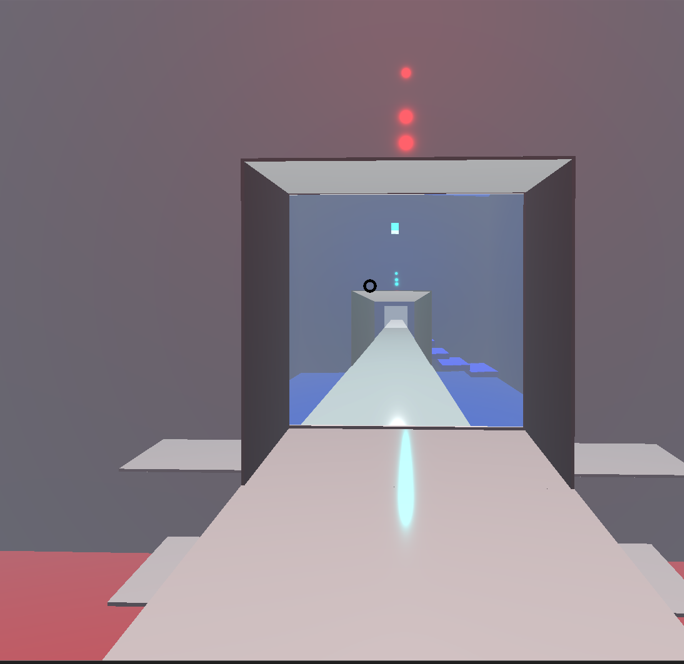
Prima del movimento Dopo il movimento

### TC-007

Esito: Passato

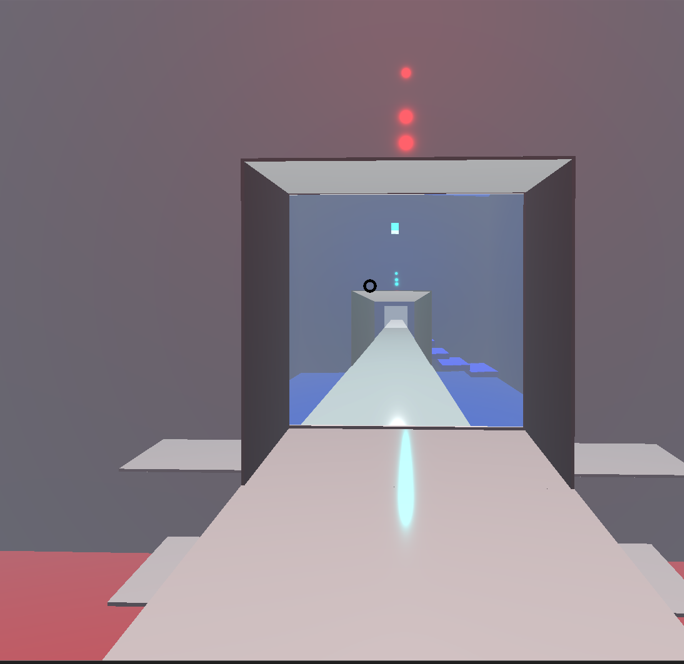
Viene renderizzata la telecamera dall’altra parte del portale.



### TC-008

Esito: Parzialmente Passato

La telecamera segue i movimenti del giocatore ma non ha un effetto eclatante in modo tale da non far capire che sia un portale.



### TC-009

Esito: Passato

Il giocatore può raccogliere gli oggetti tramite il tasto “F”

Immagine che contiene testo, logo, Elementi grafici, schermata

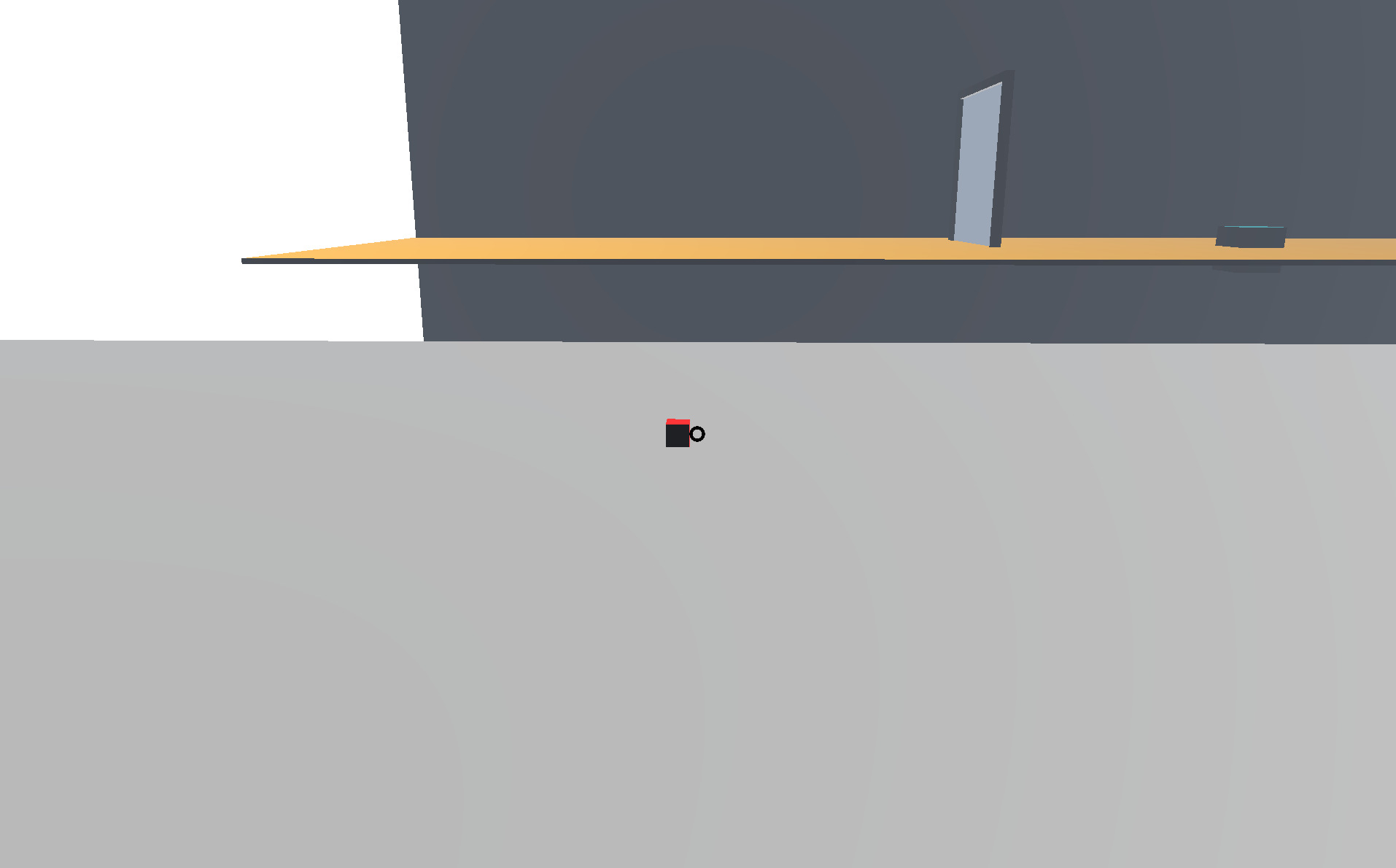
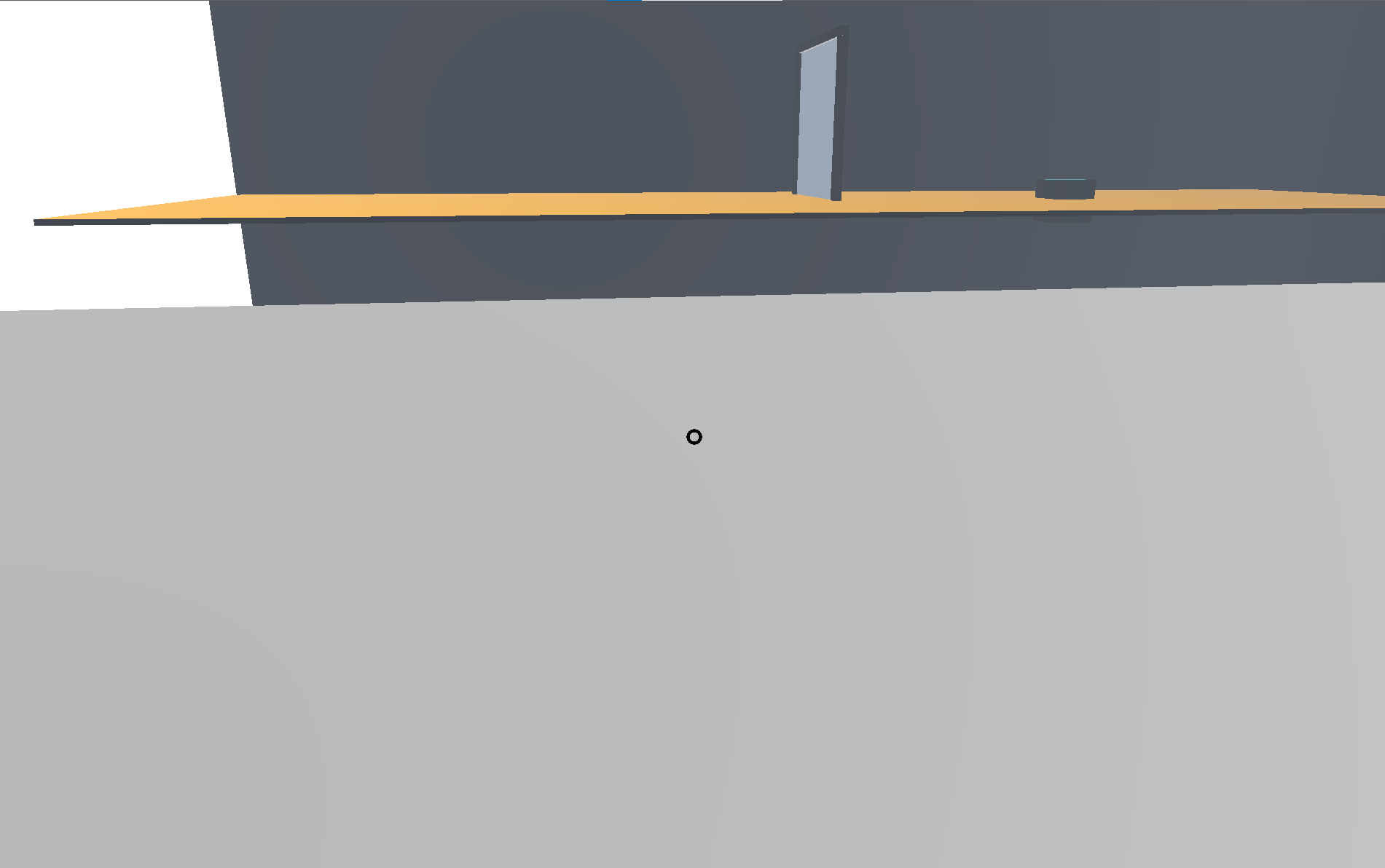
Descrizione generata automaticamente

### TC-010

Esito: Passato

Il giocatore piazza correttamente il sasso con il tasto “Q”.

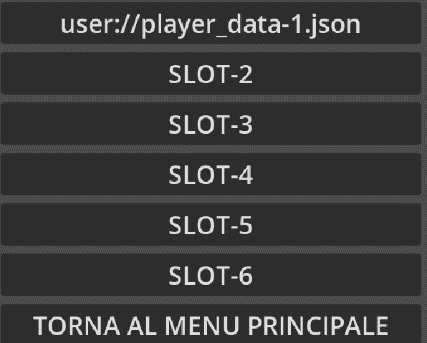
Prima di aver premuto il tasto Dopo aver premuto il tasto



### TC-011

Esito: Passato

Il giocatore carica correttamente la partita dopo averla salvata.



### TC-012

Esito: Passato

Il giocatore sente correttamente la colonna sonora se ha il volume del gioco attivo.

### TC-013

Esito: Passato

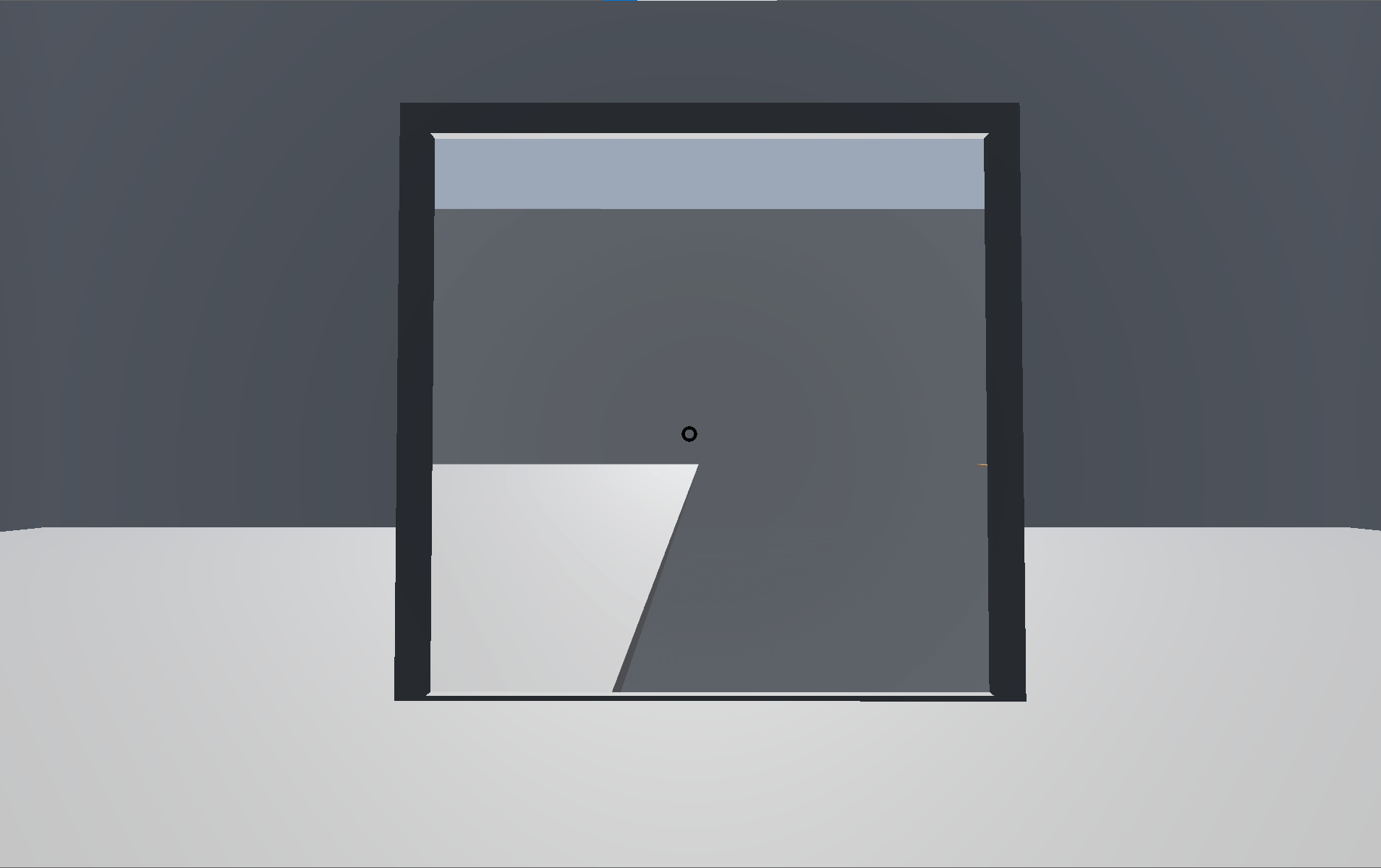
Il giocatore sente correttamente gli effetti sonori se ha il volume del gioco attivo.

### TC-014

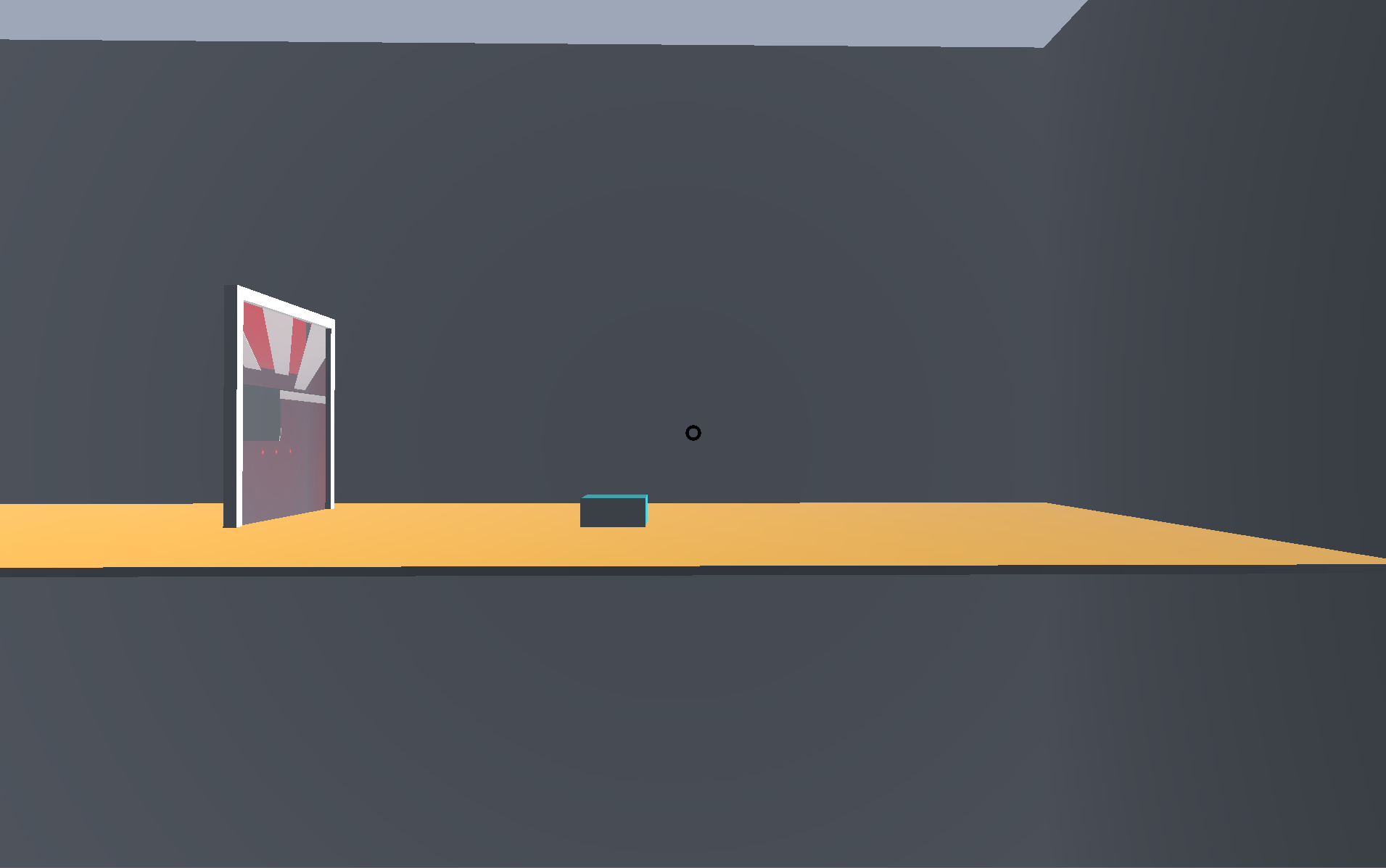
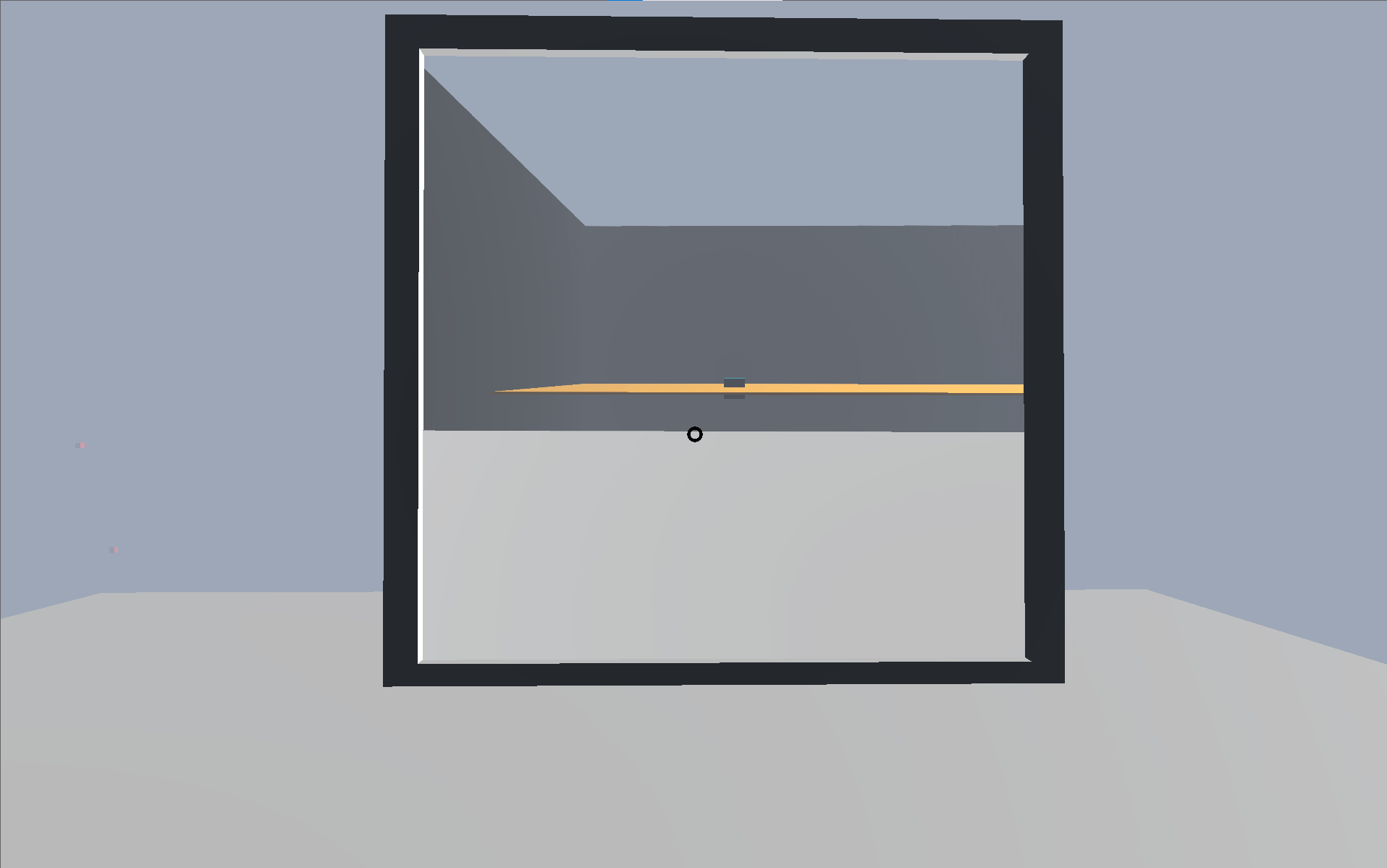
Esito: Passato

I portali sono randomici.

Prima di averlo attraversato (1° volta) Dopo averlo attraversato (1° volta)



Prima di averlo attraversato (2° Partita) Dopo averlo attraversato (2° Partita)

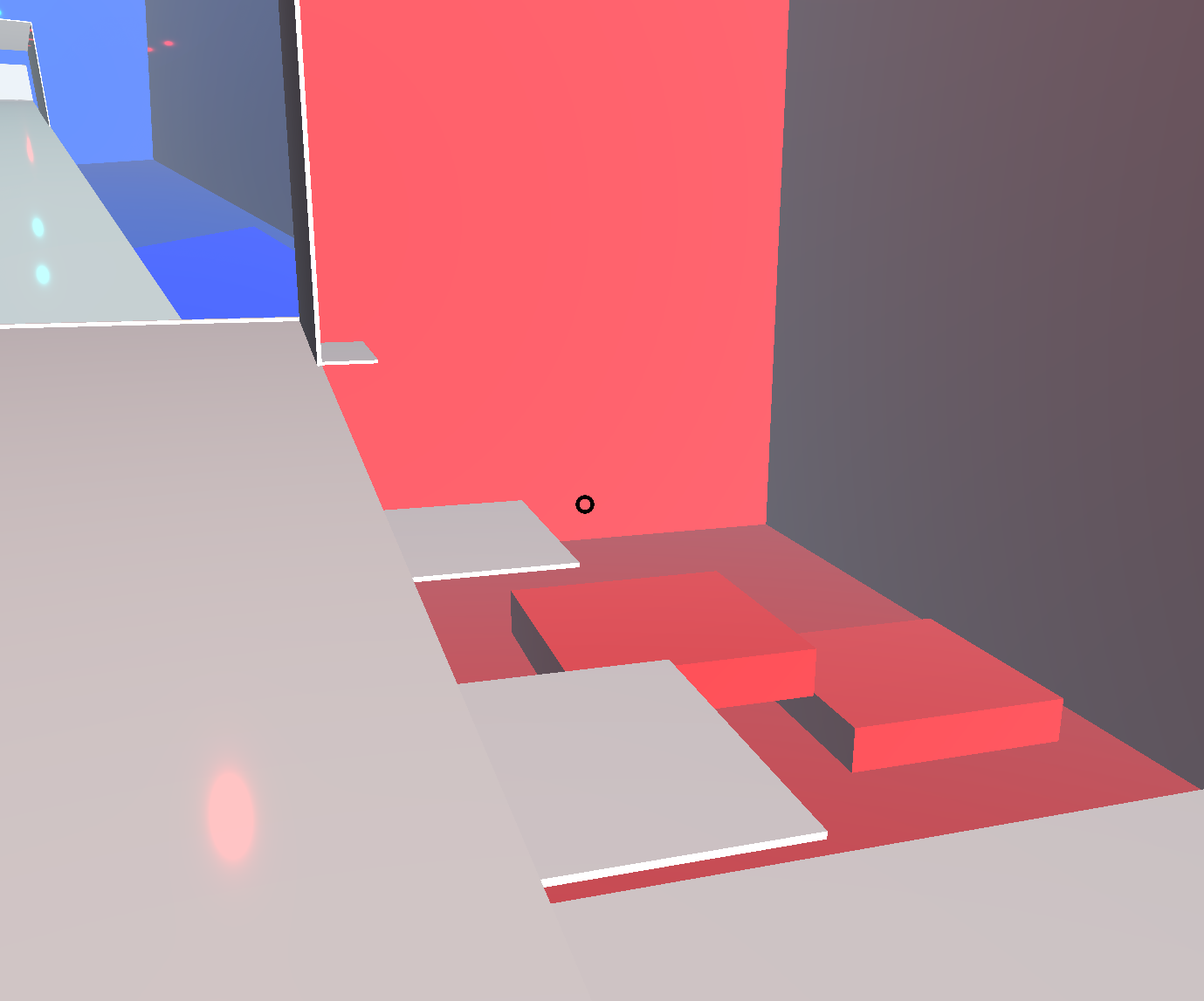
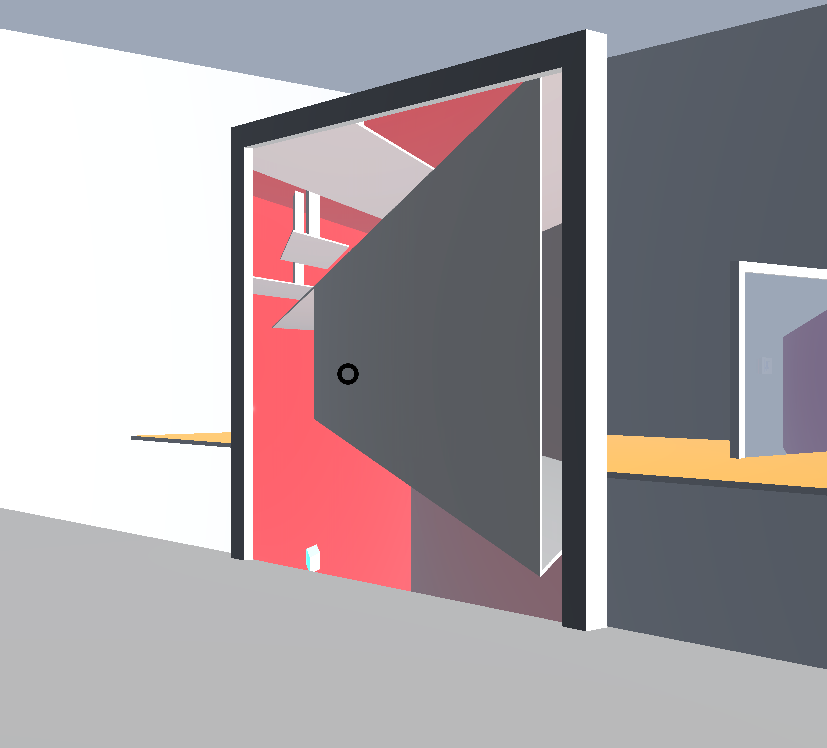


### TC-015

Esito: Passato

Il portale tiene “sottosopra” il giocatore.

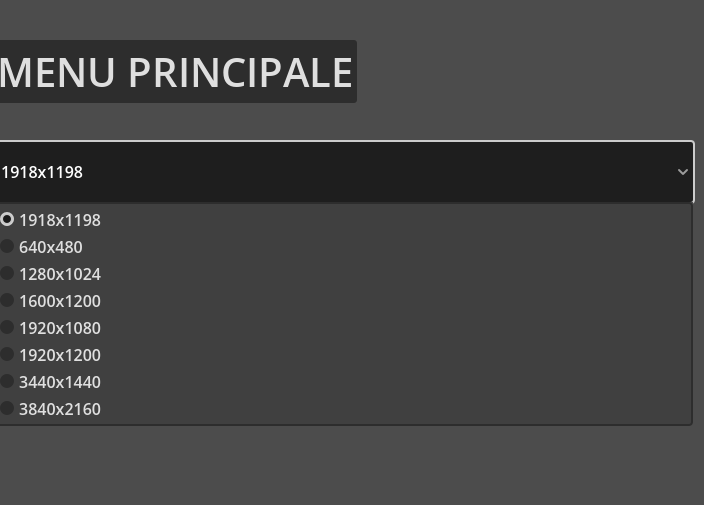
Portale che mostra l’altra parte sottosopra Una volta attraversato il giocatore è attaccato a terra



### TC-016

Esito: Passato

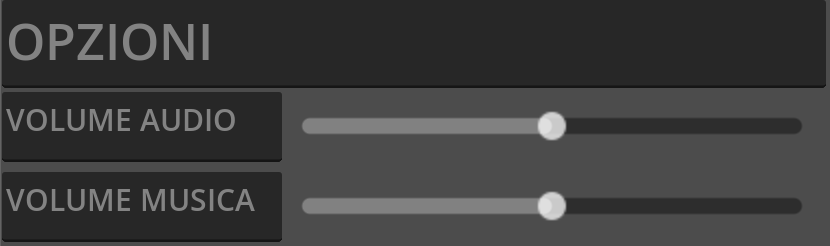
Il giocatore nel menu’ principale può modificare a proprio piacimento la risoluzione grafica.



### TC-017

Esito: Passato

Il giocatore può modificare il volume del gioco in qualsiasi momento tramite uno slider.



### TC-018

Esito: Parzialmente Passato

Il giocatore può modificare i tasti con cui gioca a proprio piacimento.

Tuttavia se si modificasse il tasto per raccogliere gli oggetti, non funzionerà se non mappato sul tasto “F”.



# Mancanze/limitazioni conosciute

* Texture Mapping Portali
  + Purtroppo per tempo non siamo riusciti a mettere una texture decente per i portali per fare l’effetto eclatante per far sembrare che non ci sia un portale.
* Tasto della raccolta differente
  + Di questa mancanza ci siamo accorti l’ultimo giorno di lavoro durante la documentazione dei test-case, purtroppo anche questo per tempo non siamo riusciti ad aggiustarlo.

# Consuntivo

****

****

# Conclusioni

Il nostro progetto non cambierà il mondo ne avrà un impatto così eclatante, però siamo sicuri che aver sviluppato un videogioco del genere avrà sicuramente un impatto interessante sulle persone che non sapevano dell’esistenza di questo tipo di gioco.

L’esperienza Non-Euclidea offre un “viaggio mentale” abbastanza particolare da far capire che potrebbe essere interessante e valerne la pena di provare un gioco del genere.

È stato molto interessante lavorare con un Engine mai utilizzato prima e aver scoperto molte cose nuove in questa esperienza.

Abbiamo inoltre capito l’importanza di una pianificazione accurata e avere comunicazione tra di noi per fare un lavoro di squadra fatto coi fiocchi.

## Sviluppi futuri

* Aggiungere le mancanze conosciute
* Creare un effetto più eclatante del portale
* Aggiungere una grafica non poly
* Avere dei movimenti più fluidi
* Migliorare la grafica della UI
* Aggiungere traduzioni per altre lingue

## Considerazioni personali

**Nicolò Fadda:**

Questo progetto è un’ottima opportunità per cimentarsi con un nuovo engine di gioco (Godot) e approfondire concetti di programmazione avanzati, come la gestione di ambienti non euclidei e la simulazione di gravità variabile.

Con uno sviluppo impegnativo, con imprevisti e problemi da risolvere, che ha permesso di acquisire nuove competenze e accrescere la propria abilità nel problem solving questo progetto è stato stimolante da realizzare.

Sono contento di quello che abbiamo fatto ma avremmo potuto fare molto di più.

**Lorenzo Di Stefano:**

Lo sviluppo non è stato privo di imprevisti e problemi da risolvere. Affrontare queste sfide mi ha permesso di accrescere la mia abilità nel problem solving, imparando ad adattarmi e a trovare soluzioni creative in situazioni complesse.

Nonostante le difficoltà incontrate, il progetto è stato indubbiamente stimolante. La soddisfazione di superare gli ostacoli e di vedere il gioco prendere forma è stata immensa.

Sono felice di come è uscito il prodotto finale.

**Matteo Ambrosone:**

"Project: Non-Euclidian" si è rivelato un'opportunità fantastica per immergermi nel mondo di Godot, un engine di gioco nuovo per me.

Nonostante il progetto sia giunto a termine, la mia avventura con Godot e con lo sviluppo di giochi non euclidei non è finita. Sono entusiasta di continuare ad esplorare questo campo affascinante e di creare nuove esperienze di gioco uniche e coinvolgenti.

Il progetto è stato un'esperienza indimenticabile, ricca di sfide, soddisfazioni e insegnamenti preziosi. Sono grato a tutti coloro che hanno collaborato con me a questo progetto e non vedo l'ora di affrontare nuove sfide del genere.

# Glossario

Inserite una semplice tabella con due colonne che spieghi i termini specifici del progetto (lista dei termini in ordine alfabetico A-Z)

Esempio:

|  |  |
| --- | --- |
| **Termine** | **Descrizione** |
| JSON | JavaScript Object Notation è un formato per lo scambio di dati basato sul linguaggio di programmazione JavaScript |
| Mesh | Un mesh 3D è una costruzione strutturale di un modello 3D costituito da poligoni. Viene utilizzato per la visualizzazione e la prototipazione. Scopri i tipi di modelli mesh 3D. |
| RigidBody | Un RigidBody è un corpo controllato direttamente dal motore fisico per simulare il comportamento degli oggetti fisici |
| Viewport | dispositivo di visualizzazione. Viene utilizzato sia nel codice che nella progettazione analogica come modo per fare riferimento allo schermo di visualizzazione e al modo in cui il layout si adatta a quello schermo |
| Non-Euclideo | Una geometria non euclidea è una geometria costruita negando o non accettando alcuni postulati euclidei (geomeria normale) |
| Renderizzare | identifica il processo di resa, ovvero di generazione di un'immagine a partire da una descrizione matematica di una scena tridimensionale, interpretata da algoritmi che definiscono il colore di ogni punto dell'immagine digitale. |
| Loop | successione di operazioni che vengono eseguite ciclicamente nello stesso ordine tipicamente con modifiche dei parametri e/o degli operandi a ogni iterazione, finché non sia soddisfatta una qualche condizione di uscita prefissata. |
| Engine | struttura software progettata principalmente per lo sviluppo di videogiochi e generalmente include librerie pertinenti e programmi di supporto come un editor di livelli.  Detta anche “Motore di gioco”. |
| MainMenu | In questo caso si tratta di una scena nel gioco dove viene mostrato il menu’ principale. |
| HSlider | Componente graico con il quale un utente può impostare un valore muovendo un indicatore |
| Nodo | Oggetto per cui può avere dei sottonodi |
| Layout | La disposizione sulla pagina degli elementi che costituiscono un bozzetto |
| InputMap | Nodo per cui vengono mappati i tasti del gioco |
| Mappare | Assegnare |
| Input | qualsiasi elemento necessario a provocare l'inizio di un dato procedimento |
| Player | Si riferisce al giocatore |
| Script | Un file con estensione .gd, contenente il codice sorgente in linguaggio GDScript |
| Funzione | unità di organizzazione del codice che permette di raggruppare una sequenza di istruzioni in un unico blocco |
| Slot | Spazio dedicato per salvare i progressi del giocatore |
| Texture | immagine bidimensionale che viene riprodotta su una o più facce di un modello poligonale tridimensionale. |

# Bibliografia

## Sitografia

**PLAYER**

Il player utilizzato è stato creato a partire dal video dello youtuber **Bramwell** video:

<https://www.youtube.com/watch?v=v4IEPi1c0eE&ab_channel=Bramwell> - First Person Movement In Godot 4 – 19.01.2024

**PORTALI**

A inizio progetto, per la creazione dei portali ci si è ispirati al video dello youtuber **3rd Monkey**, il video è il

seguente:

<https://www.youtube.com/watch?v=ReMKWYmifN8&ab_channel=3rdMonkey> - Smooth portals in GODOT – 23.02.2024

**RACCOLTA OGGETTI**

Nello script del Raycast3D posizionato sul player, il codice per la raccolta dei sassi e delle statue è ispirato dal seguente video:

<https://www.youtube.com/watch?v=QY-h1IBS_9c&ab_channel=EightFaun> - Raycasting, Recoil, and More using Godot 4.1! – 26.04.2024

dello youtuber **EightFaun**.

**CREATE STONE**

Il metodo create\_stone, nello script Global.gd, per la generazione dei mesh rappresentati i sassi, è stato creato quasi per intero dal chatbot Copilot. Le parti generate dal chatbot sono quelle nella quale vengono generati i mesh e i rigidbody.

# Allegati

Elenco degli allegati, esempio:

* Diari di lavoro
* Codici sorgente
* QdC
* Prodotto
* Gannt preventivo
* Gannt consuntivo
* Use Case
* Modelli 3D
* Abstract