

# Realtà Virtuale: Ambiente 3D interattivo con Nintendo Wiimote

Federico Scozzafava

Corso di Interazione Multimodale  
Computer Science Master Degree  
Università di Roma "La Sapienza"

Docente: Maria De Marsico

Anno accademico 2015-2016

---

# Contents

---

<b>1</b>	<b>Introduzione</b>	<b>5</b>
1.1	Motivazioni . . . . .	5
1.2	Possibili applicazioni . . . . .	6
<b>2</b>	<b>Wiimote e Sensorbar</b>	<b>7</b>
<b>3</b>	<b>Realizzazione</b>	<b>9</b>
3.1	Setup . . . . .	9
3.2	Head Tracking . . . . .	10
3.2.1	Movimenti . . . . .	10
3.2.2	Tracking . . . . .	11
3.2.3	Interpretare i dati . . . . .	11
3.3	Software . . . . .	13
3.4	Grafica . . . . .	13
3.5	Implementazione . . . . .	15
<b>4</b>	<b>Specifiche dei requisiti</b>	<b>17</b>
4.1	Attori . . . . .	17
4.2	Sottosistemi . . . . .	17
4.3	Requisiti . . . . .	17
4.3.1	Req. 1 . . . . .	17
4.3.2	Req. 2 . . . . .	17
4.3.3	Req. 3 . . . . .	18
4.3.4	Req. 4 . . . . .	18
<b>5</b>	<b>Diagrammi dei casi d'uso</b>	<b>19</b>
5.1	UC Gestore Eventi . . . . .	19
5.2	UC Server Grafico . . . . .	20
<b>6</b>	<b>Specifiche dei casi d'uso</b>	<b>21</b>
<b>7</b>	<b>Diagramma delle classi</b>	<b>23</b>

<b>8</b>	<b>Diagrammi di sequenza</b>	<b>25</b>
8.1	DS AggiornaVista . . . . .	25
8.2	DS CalcolaPosizione3D . . . . .	26
8.3	DS PersonalizzaVista . . . . .	26
<b>9</b>	<b>Conclusioni</b>	<b>27</b>
<b>A</b>	<b>Questionario</b>	<b>29</b>
A.1	Utente1 . . . . .	29
A.2	Utente2 . . . . .	29
A.3	Utente3 . . . . .	30



## *Chapter 1*

---

# Introduzione

---

Lo scopo del progetto è quello di creare un ambiente 3D interattivo su monitor per usando il controllo remoto Wiimote per tracciare la posizione dell’utente nello spazio.

Il software nasce con lo scopo di visualizzare ed interagire con modelli 3D e potenzialmente potrebbe essere ampliato per permettere la modifica diretta di modelli e ambientazioni.

Sfruttando la videocamera a infrarossi del Wiimote e degli speciali occhiali provvisti di led infrarossi siamo in grado di effettuare un headtracking preciso anche in condizioni di luce non ottimali, mostrando all’utente un ambiente 3D prospettico. Il contenuto si adatta alla posizione dell’utente; l’effetto che si ottiene è simile a quello che si guardando attraverso una finestra: quello che è visibile dipende dall’angolo e dalla distanza dalla finestra.

## 1.1 Motivazioni

Con il rilascio della console per videogiochi Nintendo Wii, Nintendo ha messo in mano ai giocatori uno strumento economico e potente. Il Nintendo Wii remote (o “Wiimote”) contiene un accelerometro, bottoni e una videocamera a infrarossi.

Grazie ai sensori del Wiimote i giocatori possono interagire in maniera più diretta e naturale con i giochi. Nonostante ciò il livello di espressione concesso da questo approccio è limitato rispetto a quello consentito dai più avanzati sistemi di realtà virtuale. Per esempio non è possibile consentire al giocatore di cambiare la prospettiva muovendo la testa.

Il progetto prende ispirazione dai lavori di Johnny Chung Lee, uno studente dottorato presso lo Human-Computer Interaction Institute dell’università di Carnegie Mellon, basati sui sensori e la videocamera del Wiimote.

Il progetto è volto a dimostrare che l’attrezzatura necessaria per questo tipo di applicazioni non è necessariamente costosa e può essere realizzata con una videocamera a infrarossi (IR) e qualche diodo led. Posizionando la videocamera sotto il monitor e i led sulla testa dell’utente è possibile determinare la posizione dell’utente e calcolare l’angolo di visione di conseguenza; questo porta l’utente a percepire l’illusione che il monitor sia come una finestra affacciata verso un mondo virtuale piuttosto che una immagine statica.

## 1.2 Possibili applicazioni

Il progetto presentato rappresenta un prototipo e un esempio di cosa è possibile realizzare con un'attrezzatura relativamente economica, precisa e accessibile.

Il sistema si presta potenzialmente ad applicazioni di vario tipo:

- Visualizzazione / editing grafico 3D
- Supporto alla creazione di contenuti 3D per artisti
- Applicazione per performance artistiche (ad esempio con sensori posizionati su dei ballerini)
- Videogiochi basati su realtà virtuale

## *Chapter 2*

---

# Wiimote e Sensorbar

---

Il Wiimote (Figura 2.1) è un dispositivo molto interessante e sofisticato. Come tutti i controller per videogiochi è provvisto di un ricco set di pulsanti e ha la capacità di vibrare; inoltre grazie ai suoi accelerometri può rilevare movimenti in tutte e 3 le dimensioni e possiede una videocamera IR frontale con risoluzione 1024x768 pixel con tracciamento hardware fino a 4 sorgenti IR a 100Hz.



Figure 2.1: Il controller Nintendo Wiimote.

Posizionando la “Sensorbar” in figura 2.2 (una barra con dei led IR posti alle estremità) sotto il monitor, il Wiimote può essere usato come un dispositivo di puntamento; questo è l'utilizzo classico predisposto dai giochi per console Wii.



Figure 2.2: La Sensorbar vista frontalmente. Ogni fila di LED è identificata come un unico punto.

Questo rende possibile calcolare la posizione relativa del Wiimote rispetto alla Sensorbar. Il Wiimote è un dispositivo wireless che sfrutta la tecnologia Bluetooth per il trasferimento dati. Per minimizzare il numero di dati trasmessi dal Wiimote alla Wii, il controller calcola la posizione di ogni punto in locale e manda in un pacchetto tutte le coordinate rilevate. Questo pacchetto comprende, oltre alle coordinate dei punti, altre informazioni come ad esempio la grandezza dei punti.

Il Wiimote non acquisisce semplicemente immagini ma tiene anche traccia di ogni punto. Quando il controller individua un punto gli assegna un numero da 1 a 4. Per esempio se sono presenti due punti nell'angolo visivo della videocamera e il punto marcato come “1” esce dal campo visivo, l'altro punto sarà comunque marcato come “2”.

## *Chapter 3*

---

# Realizzazione

---

Come descritto nell'introduzione, l'obiettivo è quello di effettuare head tracking al fine di ottenere un effetto di realtà virtuale su un monitor. In monitor tradizionale, data la natura 2D dell'immagine mostrata, non importa la distanza o l'angolazione da cui si osserva l'immagine, anche se questa è una immagine 3D. La situazione cambia se invece guardiamo fuori da una finestra: quello che si osserva dipende dall'angolo e dalla distanza dalla finestra. Questo è il tipo di effetto che si vuole riprodurre su un monitor: il contenuto visibile cambia riflettendo i movimenti dell'utente.

### 3.1 Setup

Il metodo proposto utilizza un approccio diverso da quello convenzionale proposto dai creatori della Wii. Al contrario di come descritto nella sezione 2, il controller è posto sotto il monitor mentre la sensorbar è posta sulla testa dell'utente (Figura 3.1) sotto forma di occhiali provvisti di led IR (Figura 3.2). Teoricamente, vista la capacità del Wiimote di trasmettere informazioni riguardo la grandezza del punto identificato, un solo led sarebbe sufficiente, ma al fine di aumentare la precisione e la stabilità del sistema si è deciso di impiegare due fonti luminose poste sulla testa dell'utente.

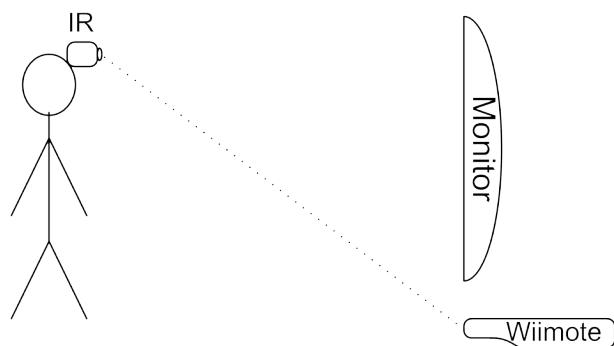


Figure 3.1: La configurazione usata per effettuare Head Tracking usando il Wiimote.



Figure 3.2: Occhiali con LED IR.

## 3.2 Head Tracking

### 3.2.1 Movimenti

I movimenti che l'utente può fare sono essenzialmente classificabili in due categorie: il movimento laterale (verticale e orizzontale), cioè cambio d'angolazione rispetto al centro del monitor, e l'avvicinarsi e l'allontanarsi dal monitor.

La prima situazione è illustrata in figura 3.3.

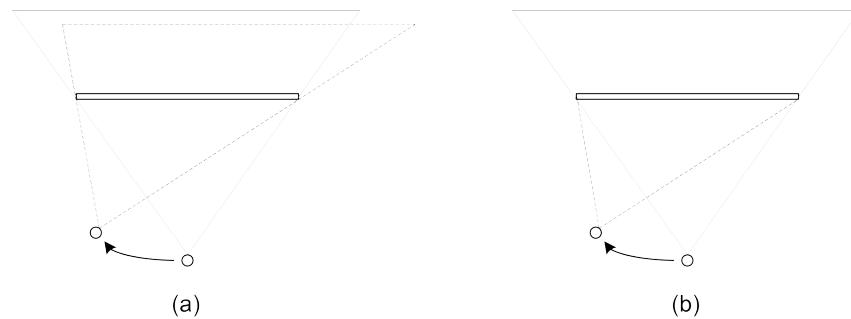


Figure 3.3: Comportamento del campo visivo quando l'utente si muove guardando attraverso una finestra (a) o un monitor (b).

La figura (a) rappresenta l'effetto desiderato, in contrapposizione la figura (b) mostra quello che accade normalmente quando l'angolo di visione non ha alcuna influenza sul contenuto che stiamo mostrando, rendendolo totalmente irrealistico. Lo stesso effetto è applicabile per movimenti verso il basso e l'alto.

La seconda situazione è illustrata in figura 3.4.

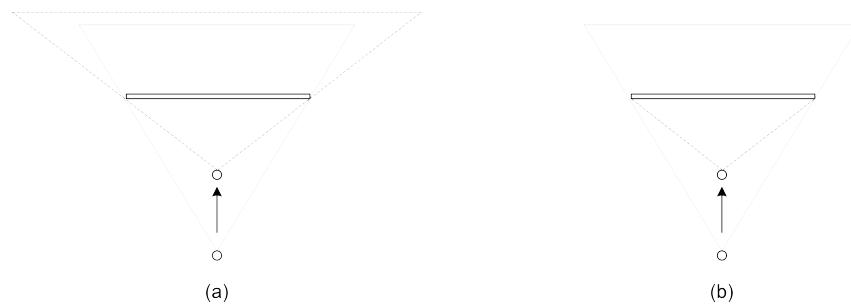


Figure 3.4: Comportamento del campo visivo quando l'utente si avvicina o allontana da finestra (a) o un monitor (b).

### 3.2.2 Tracking

Per ottenere questo tipo di interazione il sistema ha bisogno di conoscere la posizione relativa della testa dell’utente rispetto al monitor. Questo problema è conosciuto come headtracking.

Il problema in generale non è di facile soluzione; è necessario individuare un’area di interesse e analizzare un flusso costante di immagini, inoltre la qualità del tracciamento dipende molto dalle condizioni ambientali e dalla qualità delle immagini.

Grazie invece alla videocamera IR del Wiimote e del processore grafico incorporato, il compito è estremamente più semplice; non solo il tracking è effettuato dal dispositivo per noi, ma anche la qualità e l’affidabilità del sistema migliorano notevolmente dal momento che i punti di interesse sono delle fonti luminose in una determinata lunghezza d’onda.

### 3.2.3 Interpretare i dati

Nello scenario standard lo Wiimote individua i due punti, calcola le coordinate e le invia al sistema. Il sistema è in grado ora di calcolare la posizione dell’utente. La posizione nello spazio ( $x, y, z$ ) può essere determinata unicamente sulla base di questi punti.

La posizione sugli assi  $x$  e  $y$  è facilmente derivabile monitorando lo spostamento del punto medio tra i due led. La posizione sull’asse  $z$  può essere calcolata misurando la distanza tra i due led conoscendo la distanza fisica a cui sono posti i due led sulla testa dell’utente. In questo modo la proporzione tra la distanza misurata dalla videocamera del Wiimote e la distanza reale dei punti ci dirà (con una certa precisione) la distanza dell’utente dal Wiimote.

Per calcolare la distanza sull’asse  $z$  sono necessari 3 parametri: la distanza reale tra i due led, l’altezza del monitor (in pixel e millimetri) e la larghezza del monitor (in pixel). Per prima cosa viene calcolata la distanza tra i led come coordinate ricevute dal Wiimote, successivamente si usa questo valore insieme al numero di radianti per pixel della videocamera (una costante definita dividendo l’angolo di visione orizzontale di  $45^\circ$  della videocamera per la larghezza della definizione in pixel) per determinare l’angolo dal centro verso uno dei due led (Figura 3.5).

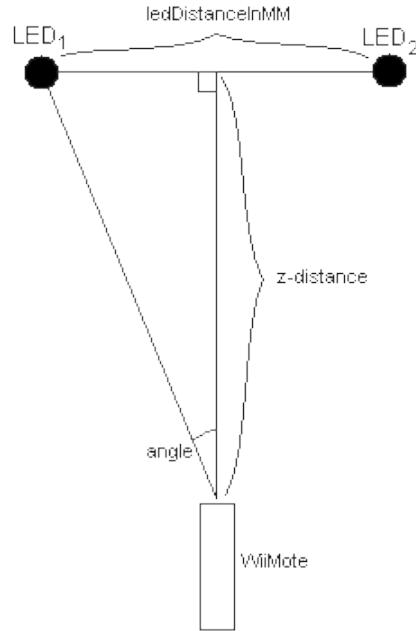


Figure 3.5: Geometria del calcolo della dimensione  $z$ .

La formula 3.1 rappresenta l’ampiezza dell’angolo sopra definito. La variabile ledDistInPixel rappresenta la distanza in pixel dei due punti visti dal Wiimote.

$$\text{angle} = \text{pixelRadians} \cdot \frac{\text{ledDist}}{2} \quad (3.1)$$

A questo punto è necessario prendere in considerazione la distanza “reale” dei led, l’angolo appena calcolato e l’altezza del monitor (Formula 3.2).

$$zDistance = \frac{\text{ledDistanceInMM}}{2 \cdot \tan(\text{angle}) \cdot \text{screenHeightInMM}} \quad (3.2)$$

Per calcolare la posizione negli assi  $x$  e  $y$  è prima necessario definire i due valori  $\text{avgX}$  e  $\text{avgY}$ , i quali rappresentano rispettivamente la media delle coordinate  $x$  dei due punti e quella delle coordinate  $y$ . Sarà sufficiente utilizzare queste misure con la distanza  $z$  calcolata precedentemente come mostrato in formula 3.3.

$$xDistance = \sin(\text{pixelRadians} \cdot (\text{avgX} - 0.5 \cdot \text{screenWidthInPixel})) \cdot zDistance \quad (3.3)$$

L’ultimo valore rimasto da calcolare è la posizione nell’asse  $y$ . Il calcolo è simile a quello effettuato per il valore  $x$ . L’unica differenza risiede nel fatto che dobbiamo considerare i casi in cui la videocamera sia posta sopra o sotto il monitor sparatamente. come possiamo notare dalla formula 3.4, è sufficiente sommare o sottrarre 0.5 per ottenere l’effetto desiderato a seconda della posizione della videocamera.

$$yDistance = -0.5 \cdot \sin(\text{pixelRadians} \cdot (\text{avgY} - 0.5 \cdot \text{screenHeightInPixel})) \cdot zDistance \quad (3.4)$$

### **3.3 Software**

Il software è stato realizzato in java e può essere eseguito su qualunque piattaforma.

Il progetto è suddiviso in due parti: la prima relativa alla comunicazione ed all'elaborazione dei dati ricevuti dal Wiimote, la seconda responsabile del rendering grafico 3D.

Per la comunicazione Bluetooth è stata impiegata la libreria java opensource Bluecove insieme alla libreria Wiiremotej per il parsing dei pacchetti ricevuti dal Wiimote, mentre per la parte grafica è stato adottato un framework per videogiochi basato su openGL, LibGDX.

Inizialmente il software inizializza il modulo bluetooth e si mette in attesa del Wiimote; una volta riconosciuto accende il primo led del controller per segnalare che la connessione è stata stabilita.

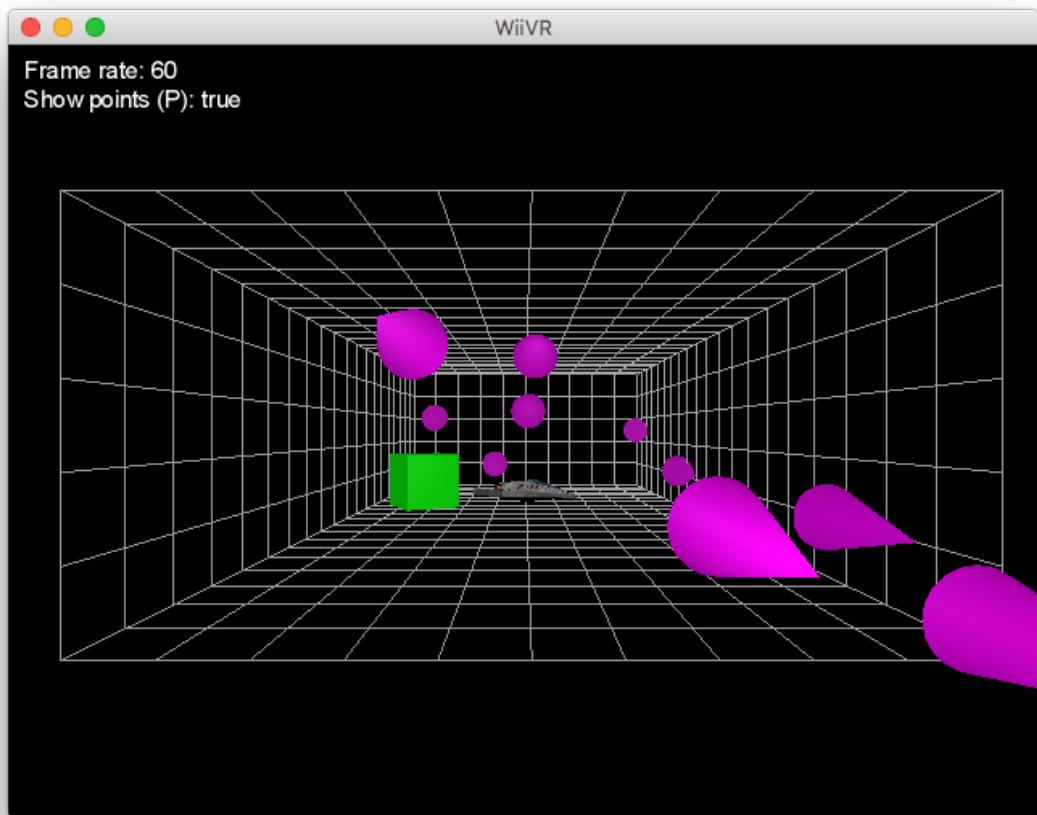
Il software acquisisce i dati dal Wiimote ad un alto rate e renderizza la scena di conseguenza. La scena è una stanza 3D formata da un reticolo in cui sono presenti dei modelli; in corrispondenza della camera virtuale è presente un punto di luce che ha lo scopo di aumentare la sensazione di prospettiva e realismo data dalla luce riflessa dagli oggetti quando l'utente si avvicina a questi.

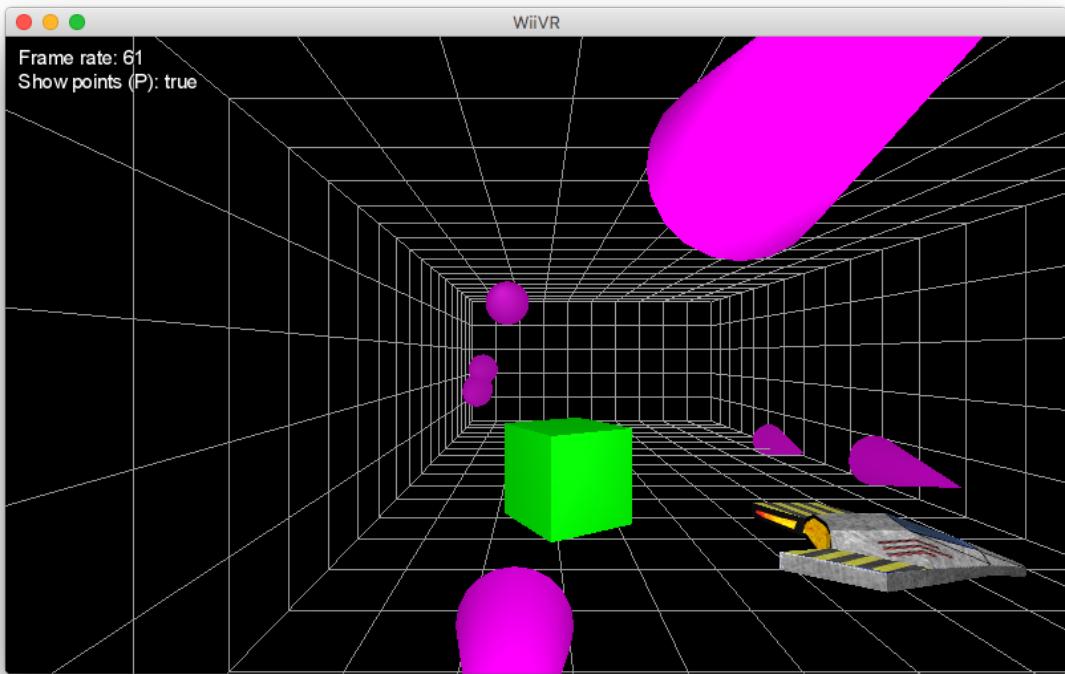
### **3.4 Grafica**

Il programma crea uno spazio tridimensionale delineato da un reticolo formato da linee verdi. Le linee creano un cubo che rappresenta una “stanza” nella quale interagiscono i vari oggetti. Le informazioni riguardo lo stato attuale del programma, le varie modalità e impostazioni sono stampate a schermo nell'angolo in alto a sinistra.

Nella “stanza” sono presenti dei modelli 3D, tra cui un cubo, una navicella spaziale con texture, coni ed altre figure geometriche (Fig. 3.4 e 3.4).

La camera è legata ad una sorgente luminosa nello spazio 3D; in questo modo l'utente può osservare la variazione di luce e riflessi sugli oggetti a seconda del materiale o texture impostata sui modelli.





### 3.5 Implementazione

Il sistema è strutturato in 3 classi:

**ROOMVR.JAVA** È la classe principale responsabile dell’interazione grafica, delle varie componenti 3D e dell’inizializzazione del controller Wiimote. La classe implementa le interfacce fornite da LibGDX (ApplicationAdapter per la struttura principale, InputProcessor per la gestione dell’input) ed espone metodi di utilità per la generazione della griglia e il disegno delle figure 3D.

**HEADTRACKER.JAVA** È la classe responsabile della comunicazione con il Wiimote e l’interpretazione dei dati ricevuti. La classe legge da file di configurazione le dimensioni del monitor ed estende la classe astratta WiiRemoteAdapter fornita dalla libreria wiiremotej.

La classe è progettata usando i design pattern Singleton e Observer-Observable e definisce quindi la classe interna Listener; in questo modo diversi listener possono registrarsi e ricevere i dati acquisiti dal Wiimote in maniera efficiente.

I calcoli della posizione descritti precedentemente sono effettuati nel metodo IRInputReceived.

**PROPERTYMANAGER.JAVA** La classe Singleton si occupa di gestire il file di configurazione esterno dal quale vengono acquisiti i parametri necessari: dimensione del monitor e id del Wiimote (opzionale).



## *Chapter 4*

---

# **Specifiche dei requisiti**

---

## **4.1 Attori**

1. Utente: Persona fisica che interagisce con il sistema muovendosi nel raggio d'azione definito dal sistema.
2. Tempo: attore concettuale che permette di modellare azioni periodiche o automatismi.

## **4.2 Sottosistemi**

1. Gestore eventi: Sistema responsabile dell'elaborazione degli eventi ricevuti dal Wiimote.
2. Server grafico: Sistema responsabile della gestione dell'ambiente grafico tridimensionale in accordo agli eventi ricevuti dal Gestore eventi.

## **4.3 Requisiti**

1. Acquisire i dati dal sensore (Req. 1)
2. Calcolare la posizione dell'utente (Req. 2)
3. Mostrare una vista grafica in accordo ai dati raccolti (Req. 3)
4. Possibilità di configurare parametri di sistema (Req. 4)

### **4.3.1 Req. 1**

1.1 Acquisire dati dal sensore un determinato numero di volte al secondo.

### **4.3.2 Req. 2**

2.1 Calcola la posizione nello spazio 3D in accordo ai dati ricevuti dal sensore.

#### **4.3.3 Req. 3**

- 3.1 Aggiornare la vista 3D in accordo alla posizione dell'utente.
- 3.2 Disegnare dei modelli 3D all'interno dell'ambiente.
- 3.3 Disegnare delle figure geometriche 3D all'interno dell'ambiente.
- 3.4 Disegnare un ambiente 3D tramite un reticolo di linee a forma di cubo.
- 3.5 Mostrare le opzioni grafiche che è possibile modificare a schermo.
- 3.6 Gestire l'input dell'utente e modificare le impostazioni grafiche.

#### **4.3.4 Req. 4**

- 4.1 Caricare il file configurazione esterno.

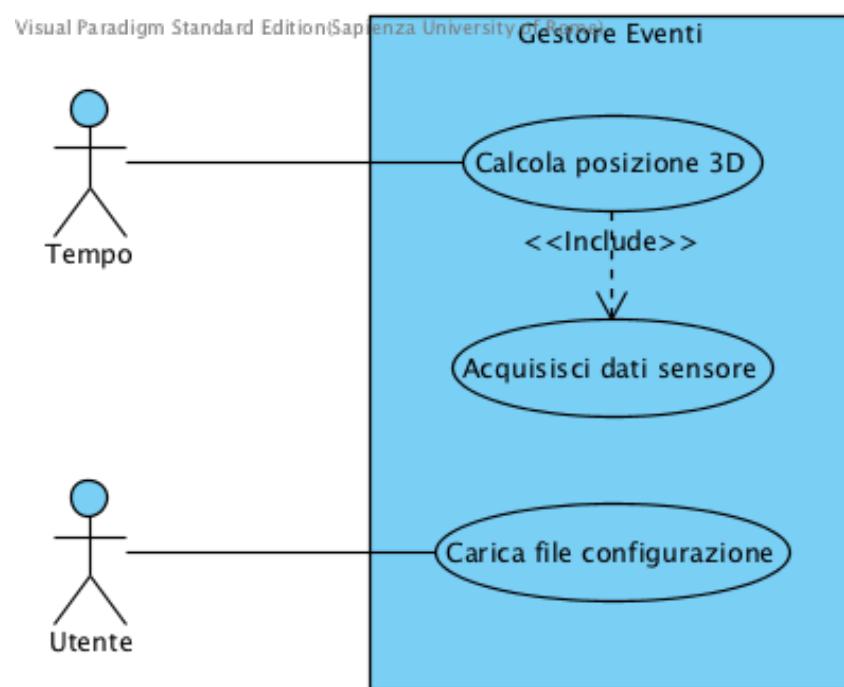
## *Chapter 5*

---

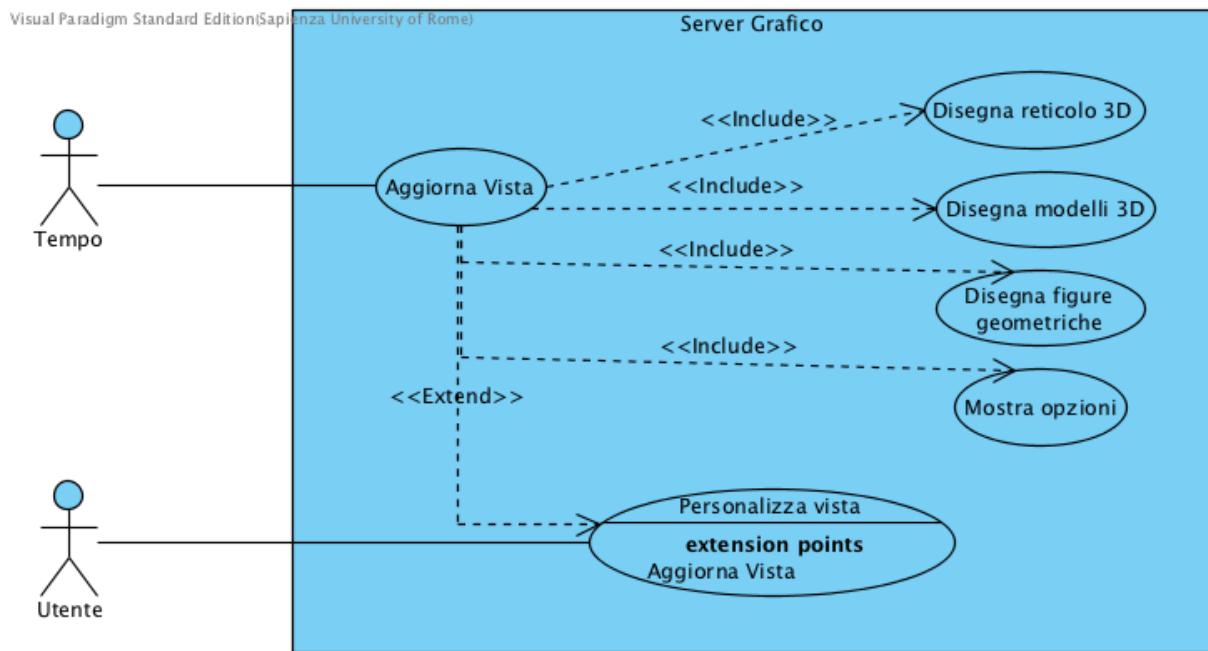
# Diagrammi dei casi d'uso

---

## 5.1 UC Gestore Eventi



## 5.2 UC Server Grafico



## *Chapter 6*

---

# Specifiche dei casi d'uso

---

Nome	Calcola posizione 3D
Attori	Tempo
Caso d'uso di inclusione	Acquisisci dati sensore
Precondizioni	Il sensore è pronto e l'utente è nel raggio d'azione del sensore
Flusso principale	<ol style="list-style-type: none"><li>1. Il sistema acquisisce i dati dal sensore</li><li>2. Il sistema calcola la posizione dell'utente</li></ol>
Post condizioni	Nessuna

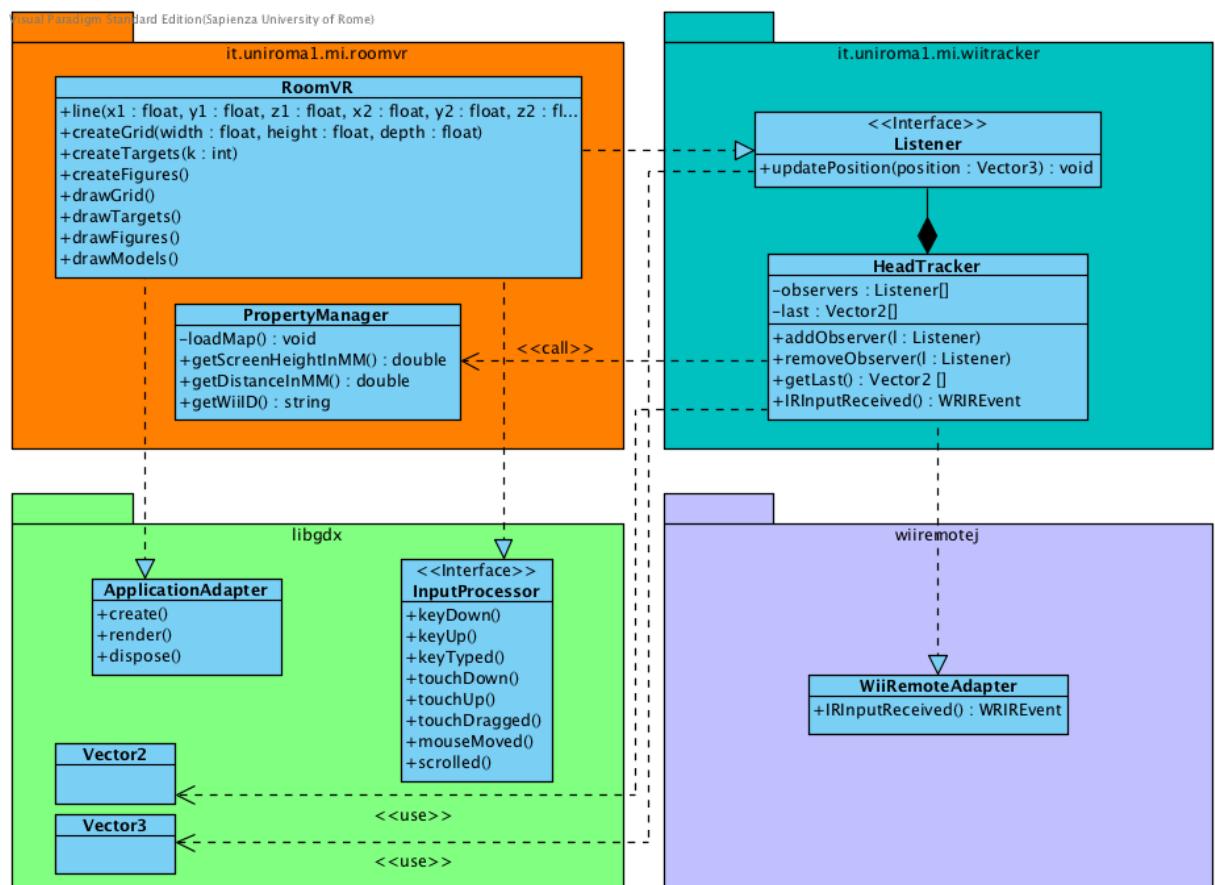
Nome	Aggiorna Vista
Attori	Tempo
Caso d'uso di inclusione	Acquisisci dati sensore Disegna reticolo 3D Disegna figure geometriche Disegna modelli 3D Mostra opzioni
Precondizioni	Nessuna
Flusso principale	<ol style="list-style-type: none"><li>1. Il sistema acquisisce i dati relativi alla posizione calcolata dal Gestore Eventi</li><li>2. Il sistema calcola calcola le matrici di trasformazione della camera 3D</li><li>3. Il sistema mostra a schermo la nuova vista</li></ol>
Post condizioni	Il sistema mostra a schermo una vista grafica coerente rispetto alla posizione dell'utente

Nome	Personalizza vista
Attori	Utente
Caso d'uso di estensione	Aggiorna Vista
Precondizioni	Nessuna
Flusso principale	<ol style="list-style-type: none"> <li>1. L'utente preme un tasto sulla tastiera esprimendo la volontà di mostrare o nascondere un elemento grafico</li> <li>2. Il sistema memorizza le nuove impostazioni</li> <li>3. Il sistema aggiorna la vista</li> <li>4. Il sistema acquisisce l'input fornito dall'utente</li> </ol>
Post condizioni	Nessuna

## Chapter 7

# Diagramma delle classi

La figura 7 mostra il diagramma delle principali classi del progetto.

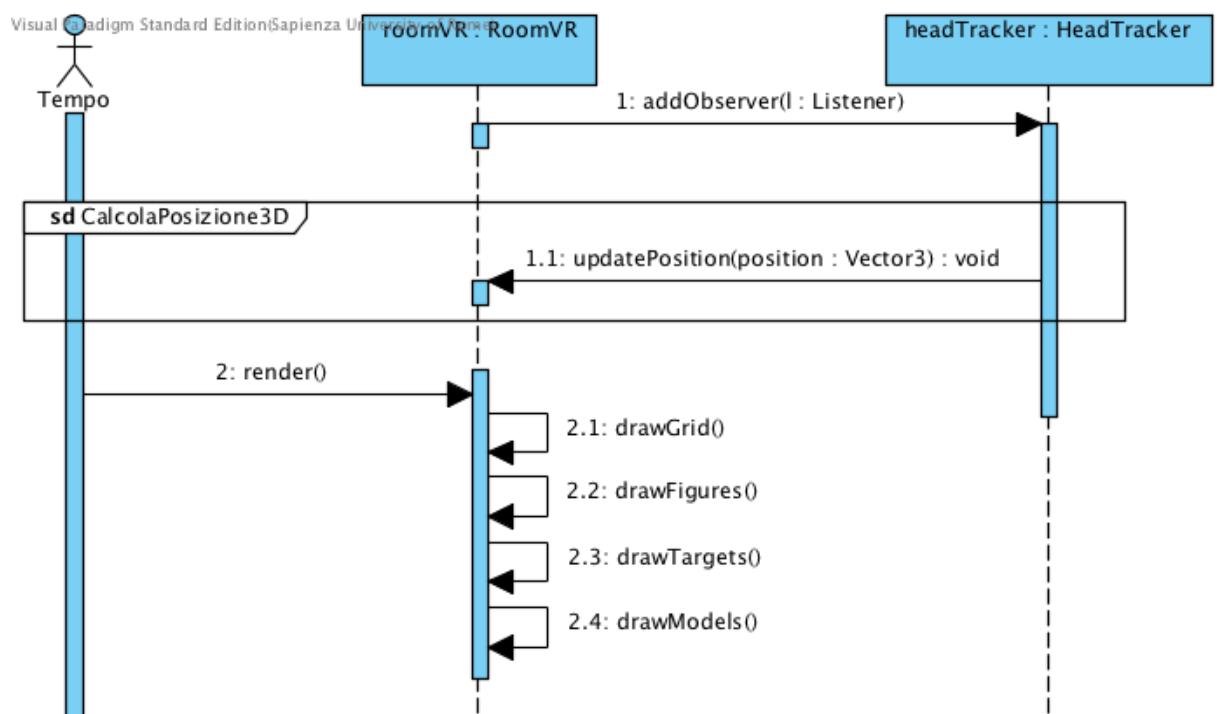




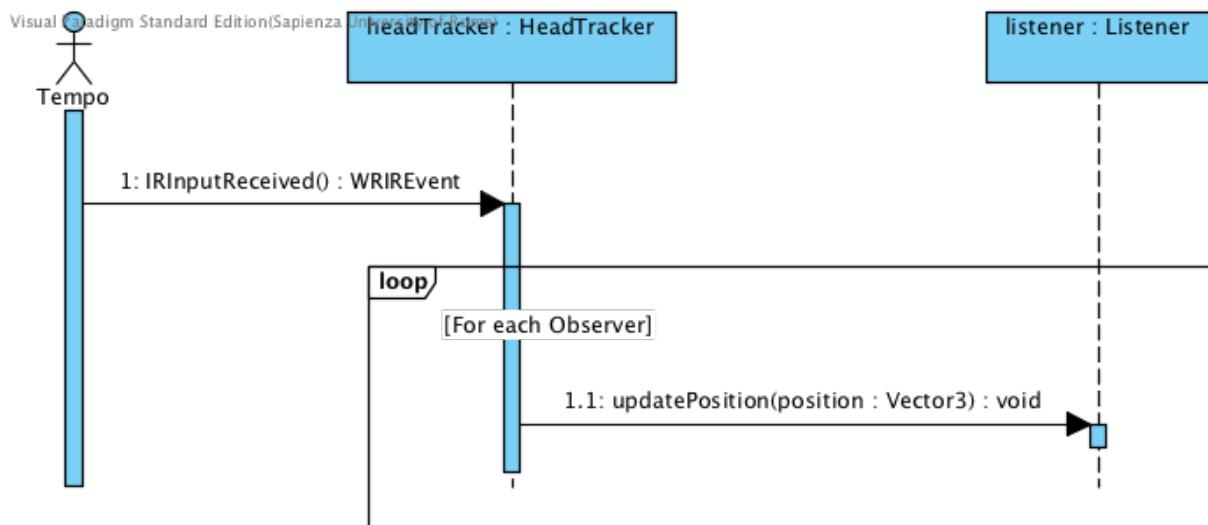
## Chapter 8

# Diagrammi di sequenza

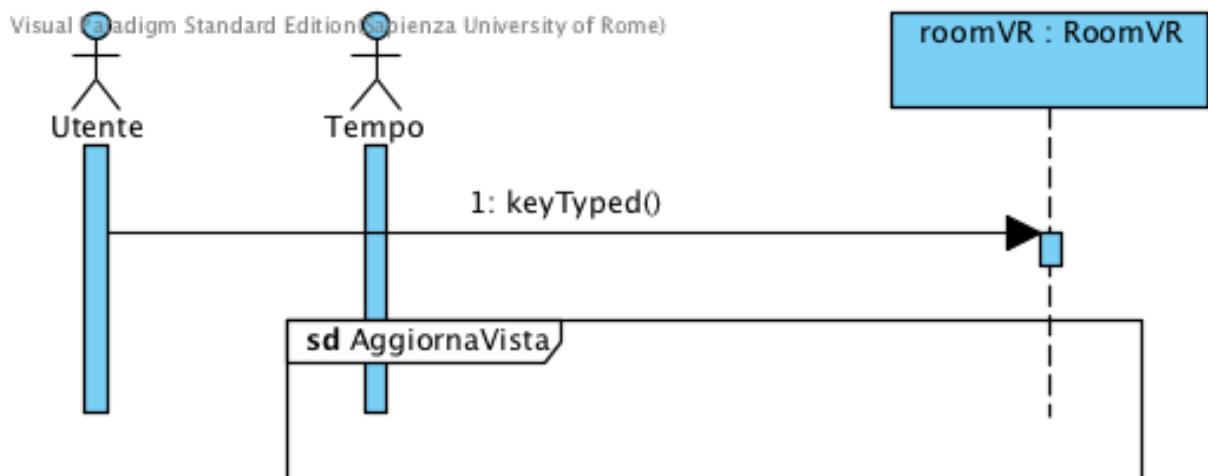
### 8.1 DS AggiornaVista



## 8.2 DS CalcolaPosizione3D



## 8.3 DS PersonalizzaVista



## *Chapter 9*

---

# **Conclusioni**

---

Il progetto sembra soddisfare i requisiti iniziali proposti. Con il costante aumentare della disponibilità di software e tecnologie sofisticate diventa sempre più facile realizzare sistemi più o meno complessi.

Il sistema è risultato fluido e robusto durante i test, seppur dimostrando qualche pecca.

Il progetto è un prototipo sperimentale e si propone come base di sviluppo per applicazioni future.



## *Appendix A*

---

# Questionario

---

### A.1 Utente1

RoomVR	- -	-	-/+	+	++
È di facile utilizzo			X		
Il sistema è immersivo e realistico			X		
Il sistema risponde in maniera reattiva					X
Ha riscontrato problemi durante l'utilizzo del sistema?	gli occhiali non sono comodi e in molti casi è necessario mantenere la testa in una determinata posizione, non è possibile avvicinarsi troppo allo schermo.				
Altri commenti	sarebbe utile aggiungere altri mezzi di interazione per ruotare l'ambiente manualmente e aiutare la navigazione.				

### A.2 Utente2

RoomVR	- -	-	-/+	+	++
È di facile utilizzo				X	
Il sistema è immersivo e realistico			X		
Il sistema risponde in maniera reattiva					X
Ha riscontrato problemi durante l'utilizzo del sistema?	L'ambiente non si muove sempre nella direzione che ci si aspetta.				
Altri commenti	asdadasd				

### A.3 Utente3

RoomVR	- -	-	-/+	+	++
È di facile utilizzo					X
Il sistema è immersivo e realistico					X
Il sistema risponde in maniera reattiva					X
Ha riscontrato problemi durante l'utilizzo del sistema?				La sensibilità del movimento è troppo elevata e i movimenti sono limitati (soprattutto quelli laterali).	
Altri commenti				bisognerebbe aumentare il raggio d'azione del dispositivo.	