Progetto di

Human Computer Interaction

A.A. 2009/2010

Implementazione di un gioco robotico

Cristian Mandelli - 739242
Deborah Zamponi - 739284

Tabella dei contenuti

1. Introduzione

Robowii 2.1 è un gioco basato sull'interazione tra un robot autonomo e un giocatore umano tramite l'utilizzo del controller Wii Remote [1] , il controller della console Nintendo Wii .

Questo progetto si pone l'obiettivo di implementare un sistema di gioco che permetta al robot di prendere decisioni autonome e di affrontare un avversario umano in un classico gioco "*spara e scappa*".

Sono infatti numerosi gli studi teorici fatti su tale sistema di gioco, ma non è ancora stata realizzata una vera implementazione che metta in pratica quanto appreso fino ad ora.

2. Risorse

Il progetto è stato realizzato da un team composto di due persone, le quali vi hanno lavorato per un periodo complessivo di circa 2 mesi (tra luglio, agosto e settembre). Tale periodo è stato impiegato sopratutto per lo sviluppo dei vari moduli in cui è suddiviso il sistema, per lo studio di fattibilità (atto ad individuare le funzionalità implementabili nel gioco) e per trovare le soluzioni più appropriate alle varie problematiche affrontate (oltre, ovviamente, alla fase di test ed al tempo dedicato agli affinamenti finali).

Il lavoro è stato svolto prevalentemente all'interno del laboratorio di Robotica ed intelligenza artificiale AIRLab, sotto la supervisione del prof. Andrea Bonarini, così da avere a disposizione tutti i componenti necessari quali:

- Robot SpyKee
- Sensori Sonar
- Ambiente di collaudo
- Controller Wii Remote

La suddivisione del sistema in moduli (sistema MRT [2]), ha permesso una maggiore parallelizzazione del lavoro di sviluppo sopratutto nelle prime fasi di implementazione.

3. Stato dell'arte

In questo capitolo verrà svolta una breve analisi dei vari studi condotti, durante gli anni, nei vari corsi offerti dal Politecnico. Tale analisi si è resa necessaria all'interno del progetto qui presentato per raccogliere alcune idee e soluzioni atti alla realizzazione di un gioco altamente interattivo e sulle quali basare l'implementazione dello stesso.

3.1 RoboWii

Il progetto RoboWii 2.1 rappresenta l'estensione di differenti alternative di design relative al più generico progetto RoboWii [3] ,sviluppato dal Politecnico di Milano nel corso degli anni.
Tale progetto si pone l'obiettivo di implementare giochi in cui sia presente una forte componente d'interazione tra uomo e macchina.

Tale interazione viene supportata, dal lato umano, tramite l'utilizzo dei più moderni controller in commercio e tipicamente utilizzati nelle principali console di gioco (con particolare riferimento al Wii Remote [1] della Nintendo).

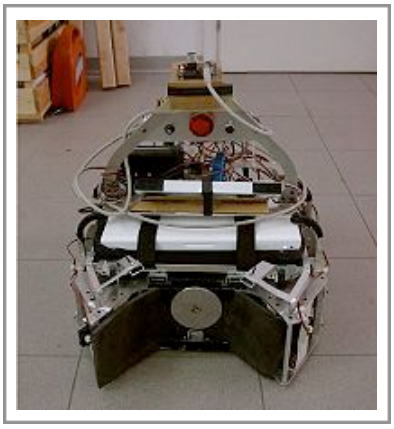


Figura 3.1: Robot utilizzato per il progetto RoboWii 1.0

3.1.1 RoboWii 1.0

Il progetto RoboWii 1.0 [4] è stato sviluppato da Antonio Bianchi e Ben Chen. In questo caso lo scopo del gioco è quello di puntare un LED posizionato su un robot tramite la telecamera ad infrarossi del Wii Remote, mentre l'automa cerca di raggiungere una posizione prefissata cercando di non venir intercettato dall'utente.

Per la realizzazione del gioco è stato utilizzato un robot - Figura 3.1 - progettato per competere nella RoboCup [5], mentre per l'implementazione del modulo di controllo del Wii Remote sono state utilizzate le librerie WiiUse [14]. Il sistema complessivo è stato sviluppato tramite l'utilizzo di un framework originariamente implementato per il progetto LURCH [6].

3.1.2 RoboWii 2.0

RoboWii 2.0 [7] è stato sviluppato da Antonio Micali con l'obiettivo di poter sfruttare i concetti e le dinamiche di gioco del progetto RoboWii 1.0 in un'implementazione realizzata con un robot a basso costo, venduto tramite i consueti canali commerciali e quindi con una maggior potenzialità di diffusione. Il robot in questione è SpaKee [8] - Figura 3.2.
Il robot SpyKee è stato equipaggiato con una serie di LED in modo da essere riconosciuto dalla telecamera IR del controller Wii Remote e di una "cintura" di sensori di prossimità per implementare un valido sistema di individuazione e superamento degli ostacoli.
La piccola telecamera di cui SpyKee è originariamente dotato è stata utilizzata, invece, per l'implementazione del sistema di visione artificiale.

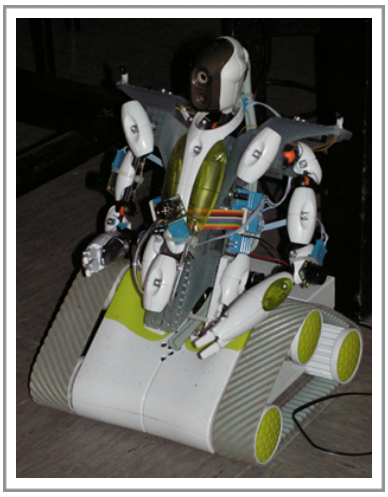


Figura 3.2: Robot SpyKee (modificato) utilizzato nel progetto RoboWii 2.0

3.1.3 RoboWii 2.0L

Il progetto RoboWii 2.0L [9] è stato sviluppato da Gabriele Pallotta e da Luigi Parpinel. Il progetto si pone l'obiettivo di eseguire una valutazione circa l'impiego di prodotti LEGO Mindstorms [10] ed in particolare del kit NXT - Figura 3.3 - per la realizzazione di giochi robotici.

Lo scopo del gioco implementato è sostanzialmente una gara di caccia fra due robot; un robot costituisce la preda, mentre il secondo robot, controllato dall'utente, rappresenta il cacciatore. Possiamo quindi vedere il gioco come una caccia al robot da parte dell'utente.

Il giocatore, in tale contesto, controlla con opportuni comandi i movimenti e le azioni proprie della caccia, nel contempo, Il robot "preda", dotato di propria intelligenza, cerca di sottrarsi alla cattura reagendo con movimenti adeguati.

Figura 3.3: Tipologie di robot utilizzati nel progetto RoboWii 2.0 L

3.1.4 RoboWii 3.0 (Jedi Training)

Il progetto RoboWii 3.0 [12], è sviluppato attualmente da Adrien Servier e da Andrea Nico, presso la sede di Como del Politecnico di Milano. Il progetto consiste nell'implementazione di un sistema che simuli l'addestramento Jedi visto nel famoso film di Guerre Stellari.

L'obiettivo del gioco consiste nel respingere, con la propria spada laser, il colpo diretto verso l'apprendista Jedi lanciato dal drone mobile.
Lo schema di sviluppo del gioco è schematizzato nella Figura 3.4

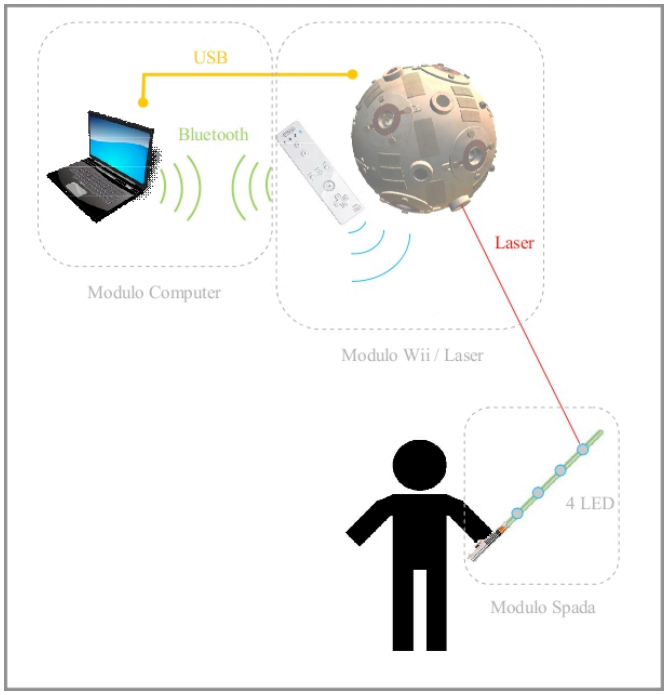


Figura 3.4: Schema di gioco RoboWii 3.0

3.2 Considerazioni finali

In questo capitolo sono stati rapidamente citati i vari sistemi aventi, come obiettivo primario, lo sviluppo di giochi robotici caratterizzati da una forte interazione tra uomo e macchina autonoma. Sopratutto oggi, date le tecnologie disponibili, è infatti possibile realizzare sistemi avanzati che consentano di portare a livelli mai raggiunti il "rapporto" uomo-robot.

La varietà di prototipi realizzati mette, inoltre, in evidenza come tale campo offra svariate opportunità di sviluppo, incentrate sia sull'utilizzo di diversi robot, sia sui differenti aspetti di interazione che possono essere sviluppati.

Nel corso dei capitoli successivi verrà analizzato nel dettaglio il sistema di gioco da noi proposto e interamente implementato.

4. Descrizione del gioco

In questo capitolo verrà descritto nel dettaglio il gioco RoboWii 2.1 e le dinamiche ad esso collegate.
Verranno inizialmente descritti gli obiettivi degli attori facenti parte del gioco, successivamente verranno date alcune informazioni di carattere generale sul sistema descrivendo, infine, le principali regole di gioco.

4.1 Obiettivi

Il gioco implementato consiste in una caccia al robot, durante la quale il giocatore si pone l'obiettivo di impedire al robot di raggiungere la propria casa base.

Il giocatore è dotato di 3 differenti tipologie di armi virtualizzate dal controller Wii Remote:

- Pistola
- Fucile
- Katana (spada)

Per aumentare la giocabilità del gioco stesso sono state definite alcune caratteristiche per ciascuna delle precedenti armi. In particolare, ad ognuna, è associata una potenza di fuoco espressa in unità di energia; in questo modo un colpo di pistola permetterà di togliere una unità di energia al robot, un colpo di fucile 2 unità di energia e 3 unità di energia per un taglio eseguito con la katana.
Ad ogni arma è associato anche un movimento di selezione, che il giocatore dovrà compiere con precisione per selezionare l'arma desiderata.
Come detto, l'obiettivo del giocatore sarà quello di impedire al robot di raggiungere il proprio rifugio.

Il robot deve, d'altro canto, evitare i colpi del giocatore e raggiungere la casa base prima di essere messo fuori gioco, possibilmente mantenendo un alto livello di energia.

Il robot, non dotato in questa implementazione di armi, potrà contare quindi solamente su alcuni punti di recupero energia disseminati nel campo di gioco (di cui non conosce la posizione a priori) che, se raggiunti, permettono di ricaricare completamente le energie perse nel combattimento.

4.2 Requisiti

Il gioco è adatto ad un target di utenza di età compresa tra i 9 e i 50 anni (se non oltre!). L'ambiente di gioco può essere una qualunque area, sia all'aperto che al chiuso, a disposizione del giocatore, purché risulti adatto al libero movimento del robot, senza danneggiarlo.
Ambienti chiusi con alcuni ostacoli potrebbero rendere più coinvolgente il gioco stesso, data la natura "spara e scappa" del gioco sviluppato.

4.3 Regole del gioco

Verranno dapprima presentate tutte le regole di base alle quali giocatore e robot devono sottostare per una corretta sessione di gioco. Successivamente verranno trattate alcune regole "avanzate" con l'obiettivo di approfondire particolari aspetti citati nelle regole di base.

4.3.1 Regole di base

Sono state definite alcune regole di base alle quali sia il giocatore "umano" che il robot devono sottostare per un corretto svolgimento del gioco. Sono di seguito elencate le regole principali:

- Il giocatore può liberamente sparare al robot tramite l'utilizzo del Wii Remote a patto che le seguenti condizioni siano soddisfatte:
  - Deve essere stata selezionata opportunamente un'arma (tramite l'esecuzione del movimento opportuno).
  - L'arma in uso deve essere carica (ad eccezione della katana).
  - Il giocatore deve puntare il robot per circa 2 secondi prima di poter colpire il robot stesso.

Se una delle precedenti condizioni non risulta essere soddisfatta, il Wii-Remote non risulterà abilitato allo sparo.

- Il giocatore dispone di tre tipologie di armi per affrontare il robot tutte con potenze e modelli di utilizzo differenti.

- Gli effetti di un colpo andato a segno sono direttamente dipendenti dalla tipologia di arma utilizzata dal giocatore.

- Ogni tipo di contatto fisico tra robot e giocatore non è ovviamente permesso.

- Il robot per raggiungere la casa base, può muoversi liberamente nell'ambiente di gioco, evitando ovviamente eventuali ostacoli che incontra sul suo cammino.

- Il robot non conosce a priori né la posizione della casa base, né la locazione dei vari punti di recupero energia disseminati nell'ambiente di gioco. Sarà quindi costretto a ricercarli durante la sessione di gioco.

- Il robot è in grado di sapere quando viene puntato dal Wii Remote e la direzione di tale puntamento.

- Il robot non può colpire il giocatore, ma semplicemente deve evitare di essere colpito.

- Al robot è assegnato un certo quantitativo di energia, il quale diminuirà con i colpi inflitti da parte del giocatore, mentre sarà ripristinato quando verrà raggiunto un punto di recupero energia. Se l'energia dovesse arrivare a zero il robot si ferma e la vittoria va al giocatore.

- Il robot deve raggiungere la casa base, con un valore minimo di energia, evitando i colpi dell'avversario.

4.3.2 Regole e comportamenti avanzati

Movimento del robot: Il robot può muoversi a differenti velocità, nel tentativo di fuggire dal giocatore e/o raggiungere nel più breve tempo possibile la casa base o i punti di recupero energia disposti nell'ambiente di gioco.

Punti di recupero energia: Nell'ambiente di gioco sono presenti alcuni punti di recupero che il robot dovrà prima ricercare e successivamente usare per recuperare l'energia persa durante il combattimento. Tali punti di recupero sono quindi in grado di ripristinare l'energia del robot non appena raggiunti.
Quando il robot ha raggiunto un punto di ricarica di energia, è protetto sia dagli sparti del giocatore sia da eventuali puntamenti.

Effetto dei colpi del giocatore: Sul robot sono presenti alcuni LED che mostrano, al giocatore, il livello di energia residuo dell'avversario. Una volta spenti tutti i LED, la vittoria viene assegnata al giocatore.
La quantità di energia tolta in seguito ad un colpo andato a segno dipende esclusivamente dalla tipologia di arma utilizzata.
Eseguito un colpo, (a segno o meno) il giocatore potrà continuare ad utilizzare la stessa arma oppure eseguire il movimento di estrazione di una nuova. Nel caso della katana, data la mancanza di un'operazione di ricarica, il giocatore dovrà estrarla dopo ogni colpo eseguito.
Prima di eseguire un colpo, l'arma dovrà essere ricaricata, con l'apposito movimento, e successivamente dovrà essere eseguito correttamente il *targeting* del robot, al termine del quale si accenderanno i LED del Wii Remote per indicare che l'arma è pronta per essere utilizzata.

Armi disponibili: Il giocatore ha la possibilità di scegliere, in qualsiasi momento del gioco una tra le seguenti armi:

- *Pistola*: La pistola è l'arma standard del giocatore. Le sue caratteristiche sono le seguenti:
  - *Danno*: 1 punti di energia.
  - *Politica di ricarica*: Deve essere ricaricata dopo 4 colpi eseguiti.

- *Fucile* : Questa è la prima arma opzionale a disposizione del giocatore. Ha una potenza di fuoco maggiore rispetto alla precedente, ma dispone di un caricatore più piccolo.
  - *Danno:* 2 punti di energia:
  - *Politica di ricarica:* Deve essere ricaricata dopo ogni colpo.

- *Katana*: Quest'arma differisce dalle precedenti in quanto non dispone di un'azione di ricarica. Essa ha una potenza di fuoco elevata, ma necessita di maggiore precisione per poter colpire il robot. Un "taglio" per essere considerato a segno deve infatti passare per il centro esatto del robot e deve essere eseguito ad una distanza inferiore ai 2 metri.
  - *Danno:* 3 punti di energia
  - *Politica di ricarica:* Come detto non esiste una vera e propria politica di ricarica, ma è necessario estrarre la katana dopo ogni esecuzione di un colpo ed eseguire il puntamento prima di provare a colpire.

5. Interfaccia utente

Nel corso di questo capitolo verrà introdotta l'interfaccia utente da noi implementata, la quale si basa quasi interamente sull'utilizzo del controller Wii Remote.

5.1 Funzionalità

Il controller Wii Remote, noto in gergo con il termine WiiMote, rappresentato nella Figura 5.1, mette a disposizione 12 pulsanti più una serie di sensori interni che garantiscono un grado di interazione con l'ambiente circostante e con gli elementi del gioco mai visto prima.

Figura 5.1: Tre prospettive del rivoluzionario controller Nintendo Wii Remote.

Il controller è dotato nella parte frontale di una telecamera IR che gli consente di individuare nell'ambiente circostante fino a 4 sorgenti infrarossi, come ad esempio i LED montati sul robot SpyKee.
Nella parte anteriore, è stato inserito un pad direzionale, con un grosso tasto A sotto di esso ed un grilletto nella parte inferiore del controller noto come tasto B. Al di sotto del tasto A vie è una fila di 3 tasti: "-", Home, "+". Vicino all'estremità inferiore del controller sono invece presenti altri due pulsanti identificati come 1 e 2, al di sotto dei quali, sono presenti 4 LED azzurri.

All'interno del controller è anche presente un accelerometro a tre assi, in grado di registrare le accelerazioni imposte al telecomando durante un movimento dell'utente.

L'interfaccia di gioco è stata sviluppata ed incentrata sulle potenzialità offerte dal controller scelto, il quale viene usato come unico mezzo di interazione a disposizione del giocatore umano.

L'idea alla base dell'implementazione dell'interfaccia utente, dato l'elevato numero di combinazioni di tasti e movimenti eseguibili, consiste nell'utilizzo di un ristretto sottoinsieme degli stessi, in modo da rendere più rapido l'apprendimento da parte dell'utente. Le combinazioni "*tasto +movimento*" adottate permettono inoltre all'utente di imporre le proprie volontà di controllo in modo molto naturale e divertente.

Nella sezione successiva verranno presentate nello specifico le associazioni tra le funzionalità proprie del gioco e come queste possano essere facilmente richiamate dal giocatore tramite il WiiMote.
Per una spiegazione circa le scelte effettuate si rimanda al capitolo 6, dove verrà presentato lo studio di fattibilità condotto.

5.1.2 Associazione funzionalità - interfaccia

Le azioni base che il giocatore può eseguire consistono nelle seguenti operazioni:

- Mirare il Robot
- Sparare verso il robot
- Ricaricare la propria arma
- Estrazione di una nuova arma
- Terminare il gioco e blocco di sicurezza

Ad ognuna di queste azioni è stato associato un comando in modo da poter essere richiamata tramite l'uso del WiiMote. Tale comando è da intendersi sia come semplice pressione di un tasto, sia come esecuzione di un particolare movimento. La disposizione dei tasti sul controller è indicata nella Figura 5.1.

Mirare il robot: A questa azione non è associato un particolare comando che deve essere eseguito tramite il WiiMote, ma semplicemente è sufficiente puntare il controller verso il Robot, in modo che esso possa essere individuato. (Si ricorda che il robot è in grado di sapere se il puntamento è attivo).

- *Movimento:* Puntamento del WiiMote in direzione del robot.
- *Tasto:* N/A

Sparare verso il robor: Questa azione è combinata con la precedente di puntamento. Se al momento dello sparo il robot è puntato, tale sparo sarà considerato valido, altrimenti il bersaglio verrà mancato.

- *Movimento:* N/A (Vedi: Mirare il robot)
- *Tasto:* B - grilletto

Ricaricare la propria arma: L'azione di ricarica è unica a prescindere dall'arma selezionata dall'utente (ad eccezione della Katana).

- *Movimento:* Movimento veloce prima verso l'alto e poi verso il basso del WiiMote (in posizione orizzontale), ripetuto due volte. (Vedi tutorial nella sezione dedicata del DVD)
- *Tasto:* A (premuto per tutto il movimento)

Estrazione di una nuova arma: I movimenti di estrazione sono visibili all'interno del tutorial presente nel DVD. Essi sono schematizzati di seguito:

- Pistola:
  - *Movimento:* Estrazione dalla fondina (Destra o Sinistra) sulla cintura del giocatore.
  - *Tasto:* A (premuto per tutto il movimento)

- Fucile:
  - *Movimento:* Estrazione dalla fondina posta sulla schiena del giocatore (Estrazione eseguita dal basso).
  - *Tasto:* A (premuto per tutto il movimento)

- Katana:
  - *Movimento:* Estrazione dalla fodera (Destra o Sinistra) posta sulla schiena del giocatore (Estrazione eseguita dall'alto).
  - *Tasto:* A (premuto per tutto il movimento)

Terminazione del gioco: Permette di terminare il gioco in un qualsiasi momento. Tale funzionalità è stata pensata anche come "blocco di sicurezza", infatti , se eseguita, blocca immediatamente il robot e la sessione di gioco, evitando così possibili danni (per quanto lievi possano essere) a cose o persone

- *Movimento:* N/A
- *Tasto:* "-" [meno] - Il Tasto risulta infatti in posizione scomoda per le impugnature utilizzate durante la normale sessione di gioco, e quindi difficilmente selezionabile per errore.

5.1.3 Funzionalità secondarie

Il controller WiiMote, oltre che essere impiegato per le interazioni descritte nella sezione precedente, dispone di un motorino vibrante, di 4 LED luminosi e di un piccolo altoparlante. Questi dispositivi sono stati utilizzati per fornire delle informazioni aggiuntive al giocatore e per rendere ancora più "reale" e coinvolgente l'esperienza di gioco.

In particolare, dei dispositivi offerti, è stato utilizzato il *modulo vibrante* con l'obiettivo principale di svolgere attività di conferma; esso infatti esegue delle piccole vibrazioni quando viene eseguito un colpo andato a segno, in modo da dare al giocatore un ulteriore feedback della propria azione.

5.1.3 Indicatori del robot

Sono disponibili anche una serie di indicatori installati direttamente sul robot; in particolare, esso è dotato di:

- una serie di 4 LED rossi e di 4 LED gialli posti entrambi sulle spalle (destra e sinistra) del robot stesso
- un LED verde posto sulla testa del robot
- due LED infrarossi posti anch'essi sulle spalle del robot, ma diretti verso il retro dello stesso.

Figura 5.2: immagine frontale del robot con i vari Led di cui è dotato

La serie di LED rossi viene utilizzata come indicatore dell'energia residua del robot; così facendo, il giocatore può sempre sapere, durante la sessione di gioco, l'energia a disposizione dell'avversario e quindi impostare al meglio la propria strategia di gioco.
Ciascun LED acceso indica un'unità di energia disponibile al robot e, ad ogni colpo andato a segno, verranno spenti tanti LED quant'è il danno causato dall'arma in uso.

Il LED verde posto sulla testa di SpyKee è stato qui utilizzato per indicare lo stato di *targeting* del robot stesso: quando il giocatore riuscirà a puntare il robot per un tempo sufficiente ad abilitare lo sparo, infatti, tale LED viene acceso e rimane tale fino a quando il giocatore non avrà sparato.
Il LED si spegnerà sia se il colpo del giocatore andrà a segno sia nel caso in cui il colpo vada perso (infatti la fase di targeting è richiesta dopo ogni sparo).

6. Documentazione Tecnica e studio di fattibilità

In questo capitolo verranno descritti gli aspetti più tecnici relativi alla realizzazione del gioco, mentre nella sezione seguente verrà descritto e spiegato nel dettaglio lo svolgimento del gioco in tutte le sue fasi, sfruttando alcuni diagrammi.

Verrà successivamente descritto l'hardware utilizzato per la realizzazione del gioco presentando l'implementazione dei vari moduli che dovranno controllare l'intero svolgimento del gioco, l'intelligenza del robot, il controllore del robot ed il modulo dedicato al riconoscimento dei movimenti e delle volontà del giocatore.

Infine verranno descritte le principali problematiche affrontate nello studio di fattibilità condotto nelle fasi preliminari dell'implementazione del gioco stesso.

6.1 Architettura del sistema e descrizione dei moduli

In questa sezione verrà presentata l'architettura del sistema complessivo, con tutti i moduli software che la compongono.

Per l'implementazione dei giochi robotici, il politecnico di Milano ed in particolare l'AIRLab, ha sviluppano negli anni un framework chiamato MRT [2], in grado di permettere una semplice gestione dei vari moduli che compongono il gioco.

Sistemi complessi di questo tipo sono infatti costituiti da decine di moduli separati ed indipendenti gli uni dagli altri, che cooperano in parallelo, eventualmente scambiandosi messaggi, per il raggiungimento dell'obiettivo finale. Il framework MRT si occupa quindi della parallelizzazione di tali moduli e della gestione delle comunicazioni tra i moduli stessi (Figura 6.1).

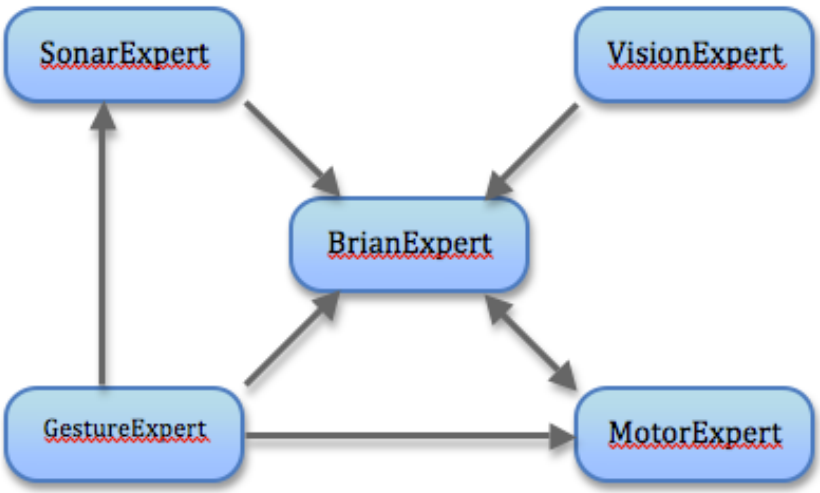


Figura 6.1: Principali moduli RoboWii 2.1 implementati all'interno del framework MRT. Le frecce rappresentano i messaggi che vengono scambiati tra i vari moduli.

L'architettura del sistema sviluppato è quindi composta da 5 moduli principali che si occupano della gestione del gioco. I suddetti moduli hanno la necessità di comunicare tra loro ed inviare i vari dati elaborati provenienti dai sensori al controllore del robot che ne gestirà i comportamenti.

Il sistema risulta composto da:

- ***SonarExpert***: Modulo che si occupa della gestione dei sonar (o sensori di prossimità) posizionati sul robot. Questo modulo inoltre si occupa anche della gestione dei LED del robot, in quanto essi vengono gestiti dal medesimo modulo XBee che si occupa dei sonar.

- ***GestureExpert***: Modulo che si occupa della gestione dell'interfaccia utente rappresentata dal Wii Remote. Questo modulo si occupa del riconoscimento dei movimenti e, più in generale, delle azioni compiute dal giocatore tramite l'utilizzo del WiiMote. Per motivi di sicurezza, inoltre, si è fatto in modo che tale modulo possa comunicare direttamente con il modulo "*MotorExpert*", al fine di poter effettuare un controllo manuale sui movimenti del robot, nel caso in cui ciò si ritenga necessario.

- ***VisionExpert***: Tale modulo si occupa unicamente della parte relativa alla visione artificiale del robot.

- ***BrianExpert***: È il modulo principale del gioco e gestisce l'intelligenza del robot. Esso riceve tutti i dati provenienti dall'ambiente esterno e mette in atto determinati comportamenti che il robot deve eseguire. Per questo è l'unico modulo (fatta eccezione di eventuali blocchi di sicurezza) che dialoga direttamente con il gestore dei motori *MotorExpert*.

- ***MotorExpert****:* È il modulo che si occupa della gestione dei motori del robot. Sulla base delle direttive impartite da *BrianExpert* esso si occuperà di mandare gli impulsi corretti ai motori.

Ogni modulo del sistema MRT deve implementare tre funzioni:

- ***void RunInit()****:* Questa funzione verrà eseguita una sola volta al momento dell'inizializzazione del modulo. Viene quindi utilizzata come fase di inizializzazione dei vari costrutti/variabili presenti nel modulo o per l'inizializzazione delle connessioni con Robot e WiiMote.

- ***void RunDuty():*** Questa funzione rappresenta il cuore di ogni modulo. Essa verrà eseguita periodicamente ad ogni "iterazione del sistema". Identifica quindi le operazioni che il modulo deve eseguire

- ***void RunClose():*** Questa funzione viene chiamata al termine del gioco in fase di chiusura per liberare le varie strutture dati e liberare eventuale memoria precedentemente allocata.

Il sistema MRT permette, come detto, di avviare periodicamente tutti i precedenti moduli. Lo sviluppatore si occuperà di specificare le periodicità con cui eseguire i vari moduli all'interno del file *Kernel.cpp*. Per una trattazione più completa e per meglio capire come lavori nel dettaglio il framework in esame si rimanda alla documentazione di MRT [2].

Per i maggiori dettagli implementativi, circa i vari moduli si rimanda, invece, alle sezioni successive del capitolo e all'Appendice B dove è presente l'implementazione di tali moduli, oppure si veda l'implementazione dell'intero progetto presente nel DVD, all'interno del quale il codice risulta opportunamente commentato.

6.1.1 Modulo: SonarExpert

Il modulo in esame è stato, in realtà, implementato all'interno dei precedenti progetti RoboWii e qui è stato semplicemente ripreso ed adattato ai nostri scopi.

Il modulo permette di comunicare con l'interfaccia presente sul robot avente il compito di leggere i valori emessi dai vari sonar. Esso esegue quindi interrogazioni periodiche di tali sensori in modo da creare una rappresentazione "virtuale" dell'ambiente intorno al robot.
Tali informazioni, espresse in termini di distanza tra robot ed ostacolo, vengono inviate (tramite messaggio) al modulo "*BrianExpert*" che le elabora, avviando così il corretto comportamento del robot.

In questo modo è stato possibile implementare un sistema in grado di reagire alla presenza di eventuali ostacoli all'interno del campo di gioco.

Il modulo in esame, inoltre permette il controllo dei LED presenti sulle spalle del robot, i quali sono controllati dalla stessa interfaccia che gestisce i sonar. Per questo motivo nella sua implementazione sono state aggiunte apposite funzioni che consentano di accendere/spegnere ogni singolo LED presente sul robot.

6.1.2 Modulo: GestureExpert

Il modulo in esame si occupa della gestione di tutte le funzionalità che il WiiMote mette a disposizione del giocatore umano.

- *Riconoscimento dei movimenti*
- *Gestione degli spari (o colpi di spada)*
- *Gestione del puntamento del robot*

Per tale scopo si è ricorsi alle librerie WiiCpp [13] sviluppate dall'università La Sapienza di Roma. Queste librerie, basate sulle più conosciute librerie WiiUse [14](progetto non più supportato), permettono il quasi totale controllo del Wii Remote e di tutte le possibili espansioni fornite dalla Nintendo, come ad esempio il WiiMotion Plus [17].

Le tre funzioni principali del modulo si occupano quindi di:

- ***RunInit():*** si occupa di creare un canale di comunicazione con il Wii Remote e si pone in attesa degli eventi generati da esso e riconosciuti dalle librerie WiiCpp.

- ***RunDuty():*** Tale funzione, richiamata ad intervalli regolari di tempo (specificati tramite MRT), consente di interrogare il WiiMote circa le azioni eseguite dal giocatore. Sarà possibile quindi abilitare, a seconda dello stato del sistema, le varie funzionalità che potranno essere rese disponibili all'utente. Al termine di ogni ciclo di esecuzione vengono inviati tutti i messaggi destinati agli altri moduli.

- ***RunClose():*** Deallocazione delle risorse e chiusura della connessione con il WiiMote.

Riconoscimento dei movimenti:

Il modulo in esame si occupa della gestione delle armi del giocatore, definendo come quest'ultimo possa selezionare e/o ricaricare una delle armi a sua scelta.

Tale funzione viene attivata solamente nel caso in cui l'utente stia premendo il tasto A sul controller Wii.

Il riconoscimento dei movimenti del gioco, quali cambio di arma e ricarica della stessa, viene effettuato tramite l'elaborazione dei valori degli accelerometri presenti all'interno del Wii Remote. Questi segnali, uno per ogni asse, rappresentano le accelerazioni, lungo le 3 direzioni, esercitate sul controller da parte dell'utente.

In questo modo è possibile identificare con una buona precisione quale sia la tipologia di movimento eseguita dall'utente.

La soluzione da noi implementata è basata sui seguenti passi:

- Definizione di un set di azioni che il giocatore può eseguire. Nel nostro caso sono stati quindi definiti i seguenti movimenti:
  - Ricarica
  - Estrazione di pistola, fucile e katana.

- Per ciascuna delle 4 precedenti azioni sono stati campionati *n* modelli di riferimento, sulla base dei quali eseguire la classificazione.

Le figure seguenti mostrano le curve di riferimento per ognuna delle azioni sopra citate; la sostanziale differenza tra le varie curve viene sfruttata per l'ottenimento di un riconoscimento robusto, a discapito del numero di possibili movimenti eseguibili dall'utente.

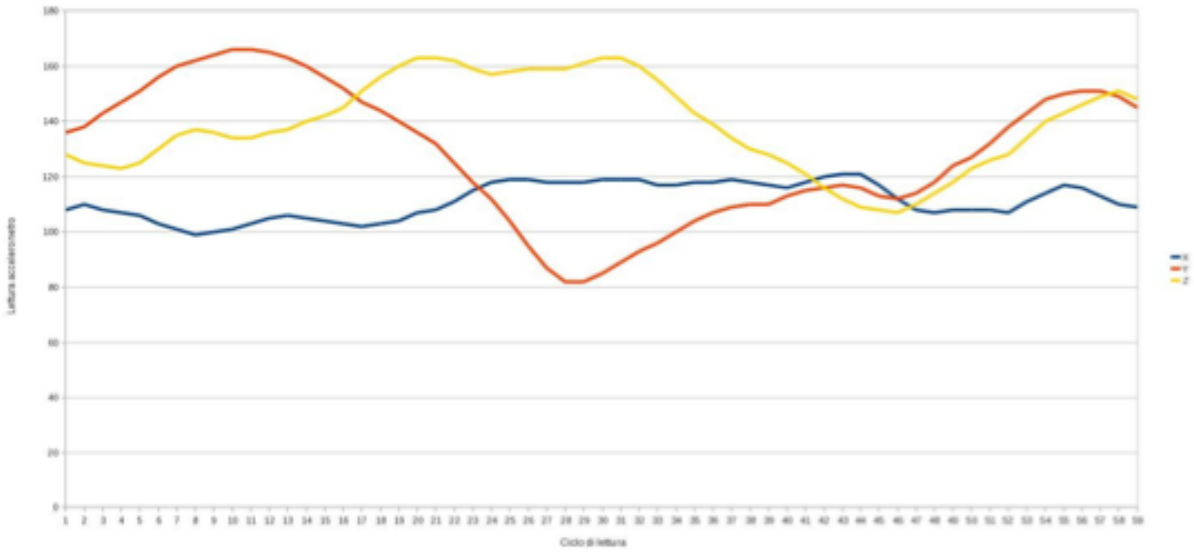


Figura 6.2: Estrazione della pistola

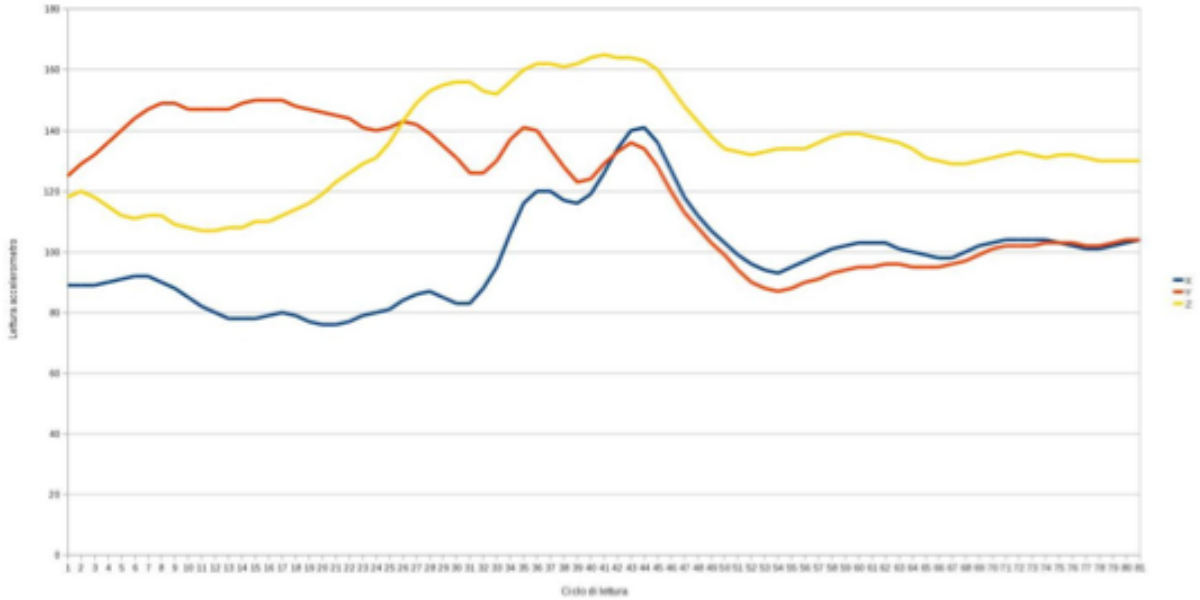


Figura 6.3: Estrazione del Fucile

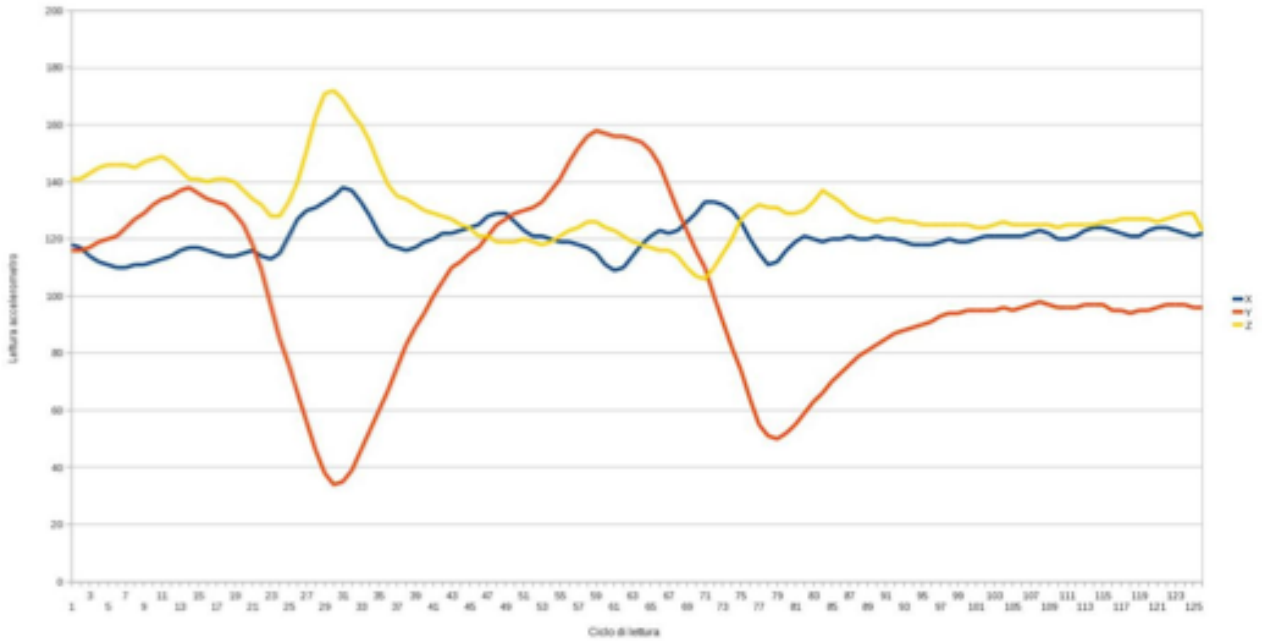


Figura 6.4 Estrazione della katana

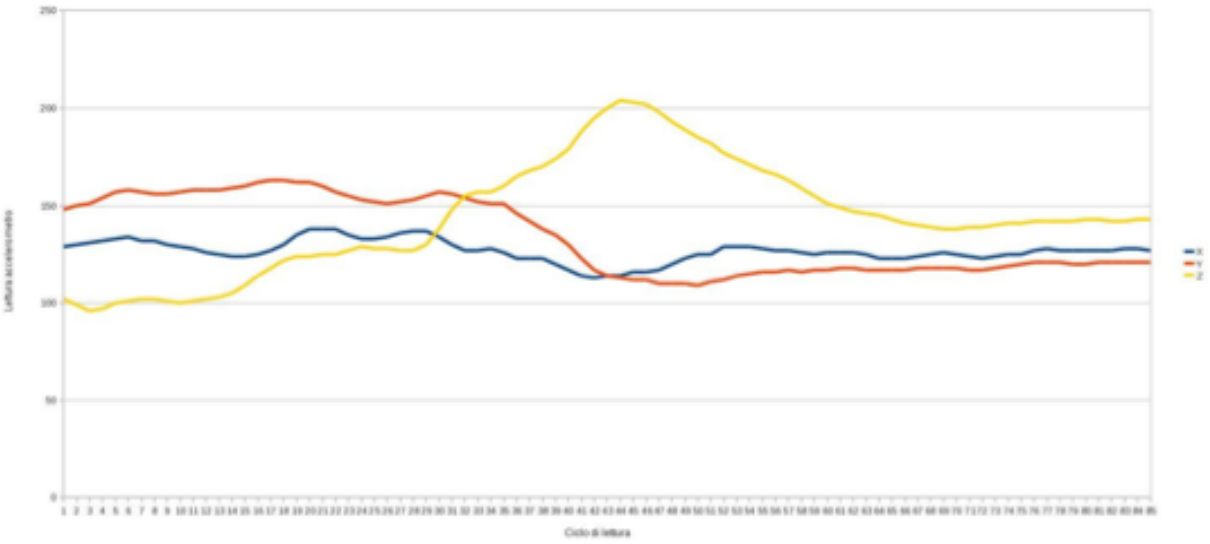


Figura 6.5 Ricarica di Pistola e Fucile

La classificazione del movimento compiuto dall'utente viene eseguita grazie ad un classico algoritmo KNN da noi implementato con l'ausilio delle librerie ANN [15]. Per ottenere una classificazione robusta, sono stati appresi "*n*" modelli di riferimento per ciascun movimento che dovrà essere eseguito.
Tali modelli, costituiti da un numero variabile di campioni (a seconda della durata del movimento stesso), sono stati normalizzati su un range fisso di 150 campioni che equivalgono pressappoco ad un movimento di 1,5 secondi (il WiiMote ha una frequenza di invio pacchetti di circa 100Hz).
Per eseguire la normalizzazione è stata utilizzata la Spline Cubica di Hermite [16], funzione di 3° grado i cui polinomi sono nella forma di Hermite [16].

Ad ogni punto, (costituito da 3 coordinate x,y,z) di ogni curva di riferimento, è stata assegnata una classe di appartenenza a seconda del movimento rappresentato: le classi utilizzate sono state:

- ***S:*** Indica un punto appartenente ad un modello di curva riferito all'estrazione della spada
- ***R:*** Indica un punto appartenente ad un modello di curva riferito al movimento di ricarica dell'arma
- ***G:*** Indica un punto appartenente ad un modello di curva riferito all'estrazione della pistola
- ***D:*** Indica un punto appartenente ad un modello di curva riferito all'estrazione del fucile

Tutti i modelli appresi, con le relative classi, sono stati salvati su file, per poter essere caricati in fase di avvio dal modulo in esame.

Il modulo di classificazione viene attivato dal sistema ogni volta che l'utente mantiene premuto il pulsante A del WiiRemote in modo da registrare tutti i dati provenienti dall'accelerometro del controller durante il movimento eseguito.
A fine di tale campionamento, i dati vengono normalizzati in base al criterio definito in precedenza (150 campioni) e successivamente si avvia la classificazione.

L'algoritmo KNN esegue una ricerca all'interno dei campioni dei modelli, restituendo la classe più probabile associata ai "*k"* punti più vicini ad ogni punto P(x,y,z) registrato.

Dopo ripetuti test sono stati fissati i seguenti valori:

- *K* = 10 (Numero di K punti da considerare)
- d = 7 (Distanza massima tra il campione considerato ed il punto più vicino)
- n = 10 (Numero di modelli di riferimento per ogni movimento)

Inoltre è stato anche implementato un modulo aggiuntivo che permette di apprendere nuovamente i modelli di riferimento su cui eseguire la classificazione. Questo potrebbe essere utile sia per migliorare le prestazioni del classificatore sia per aggiungere o modificare i movimenti eseguibili dall'utente e ri-addestrare il classificatore.

Puntamento del robot:
Il modulo "*GestureExpert*" permette inoltre di gestire la fase di puntamento del robot da parte del giocatore. Per eseguire con successo il *targeting* del robot sarà necessario mantenere il WiiMote puntato verso il robot per alcuni secondi.

Il LED verde posto sulla testa del robot indicherà, tramite l'accensione, l'avvenuto puntamento, abilitando così automaticamente l'uso dell'arma precedentemente selezionata

Il meccanismo di puntamento funziona come segue: esso ha inizio subito dopo la selezione di un'arma e dopo ogni sparo, fintantoché l'arma risulterà essere carica.

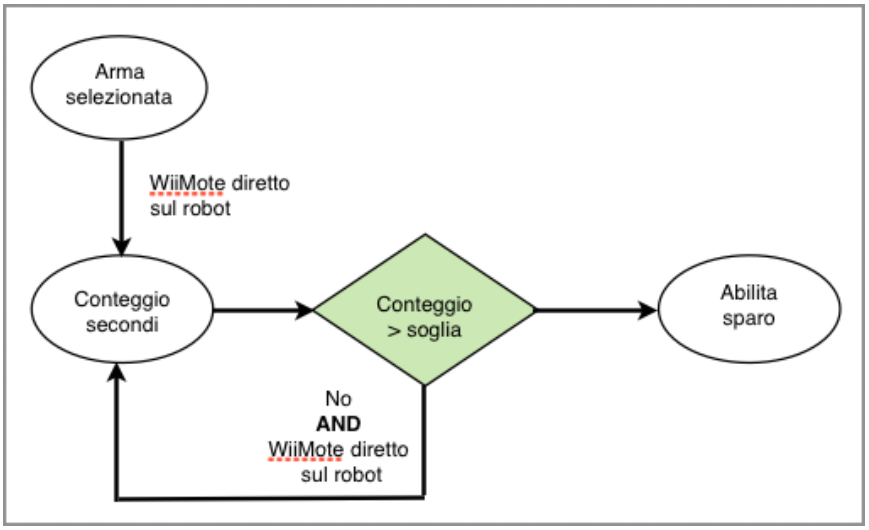


Figura 6.6: Sistema di "targeting" del robot da parte del giocatore.

Il puntamento viene gestito tramite l'utilizzo della telecamera IR montata sul Wii Remote, la quale consente di individuare fino a 4 sorgenti di infrarossi contemporaneamente.

Il Robot dispone, a tale scopo, di 2 LED infrarossi che vengono utilizzati proprio come marker. Quando vengono visualizzati e le precondizioni di tale azione sono soddisfatte (Arma carica), si avvia il "periodo" di targeting. Se durante questo periodo il giocatore perde "di vista" il robot, il conteggio verrà azzerato e si dovrà iniziare nuovamente la procedura.

Una volta eseguito correttamente il puntamento del robot, verrà abilitata la funzione che consente al giocatore di mirare e sparare al robot.

Gestione degli spari e del colpo di spada:

Il modulo *GestureExpert* si occupa inoltre della gestione degli spari eseguiti dall'utente. Se il giocatore colpisce con successo il robot, tale modulo si occuperà anche di decrementare l'energia del robot in accordo con la tipologia di arma utilizzata, tramite la generazione di un messaggio da inviare al modulo *SonarExpert* di competenza.

La gestione degli spari risulta abbastanza semplice: il modulo, infatti, controlla quando avviene uno sparo semplicemente interrogando le librerie Wiicpp circa la pressione del tasto B (grilletto) del Wii Remote.

A seconda dell'arma utilizzata si eseguono due diversi controlli per verificare se il colpo è andato a segno. Per quanto riguarda le armi da fuoco, quindi pistola e fucile, si verifica se, nell'istante in cui l'utente ha premuto il "grilletto", la telecamera IR registri la presenza almeno due punti. In tal caso

lo sparo è considerato “a segno” e si procede con la decurtazione di alcune unità di energia del robot. In caso contrario lo sparo sarà considerato “nullo”.

Per quanto riguarda, la spada, invece, ci si aspetta che l’utente esegua un taglio netto passante per il centro del robot. In questo caso la pressione del tasto B del controller sarà prolungata. La verifica circa la precisione del colpo viene eseguita, anche in questo caso, controllando se per almeno un punto della traiettoria eseguita dal giocatore, vengano rilevate le sorgenti infrarosse.

Messaggi in ingresso ed in uscita dal modulo:
GestureExpert si deve occupare di comunicare al modulo “*BrianExpert*” le azioni compiute dal giocatore umano. Per tale motivo invia dei messaggi contenenti lo stato di targeting del robot ed eventualmente il numero di unità di energia decurtate al robot.

Il modulo inoltre si occupa di mandare dei messaggi al modulo *SonarExpert* per controllare lo spegnimento dei LED di energia in seguito ad un colpo eseguito dal giocatore.

6.1.3 Modulo: VisionExpert

Il modulo di visione è un componente fondamentale che consente di analizzare tutte le immagini che il robot invia al computer principale. Questo sistema permette di definire in maniera precisa quello che il robot “vede” dal suo occhio elettronico:

- Casa Base
- Punti di recupero energia

Le tre funzioni principali del modulo si occupano quindi di:

- ***RunInit():*** Viene utilizzata per la dichiarazione del sistema di classificazione e blobAnalysis e per la registrazione ai messaggi destinati a tale modulo.
- ***RunDuty():*** Tale funzione consente di gestire la ricezione dei messaggi in arrivo dagli altri moduli ed esegue, ad ogni ciclo, l’analisi dell’immagine proveniente dal robot.
- ***RunClose():*** Deallocazione delle risorse.

L’analisi condotta sulle immagini provenienti dalla telecamera del robot si basa su un algoritmo KNN basato sui colori dell’immagine analizzata. Quest’ultimo necessita di una prima fase di addestramento, la quale ha come risultato l’ottenimento di un classificatore colore (*classifier.kcc*)

che permette di classificare l'immagine sulla base dei valori delle 3 componenti RGB di ogni suo pixel.
La fase di addestramento consiste nell'evidenziare, in alcune immagini campione, le zone di cui siamo interessati. Ad ogni pixel di tali sezioni verrà associata una classe di identificazione a seconda dell'oggetto rappresentato.

In questo modo si ottiene un elenco di pixel (punti a 3 dimensioni P(r,g,b)) ad ognuno dei quali viene associata una classe:

- Y: Casa Base (Yellow)
- R: Punto di recupero energia (Red)

Questa prima fase è schematizzata nella Figura 6.7:

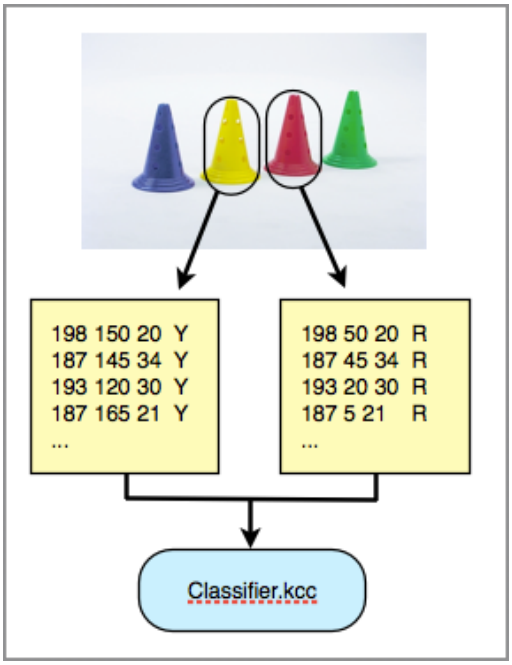


Figura 6.7: Creazione del classificatore a partire da un'immagine campione e da delle sezioni di interesse

Il classificatore viene quindi salvato sotto forma di file di testo, in modo da poter essere caricato durante l'inizializzazione del modulo (*RunInit)*.
Il modulo di visione riceve circa 15 immagini al secondo direttamente dal robot. Esso esegue una rapida classificazione delle immagini utilizzando le informazioni contenute nel classificatore precedentemente creato ed opportunamente caricato. Tale operazione consente di associare a tutti i pixel dell'immagine una classe. La selezione della classe di ogni pixel avviene tramite l'utilizzo dell'algoritmo KNN precedentemente citato.

Così facendo si identificano tutti i possibili pixel aventi come valori caratteristici (ovvero come valori dei canali R, G e B) valori simili a quelli dei pixel evidenziali in fase di apprendimento(con relativa classe). Quelli che invece avranno valori RGB troppo diversi da quelli dei possibili campioni saranno classificati come "U" (Undefined). (Figura 6.8)

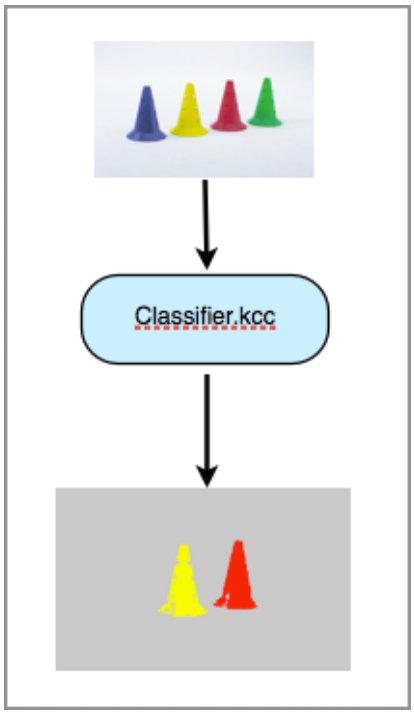


Figura 6.11: Risultato della classificazione. Sono stati colorati in giallo tutti i pixel classificati come Y, in rosso quelli con classe R e in grigio quelli con classe U (Undefined).

Per identificare le regioni di interesse nell'immagine classificata, viene utilizzato un algoritmo di Blob Analysis implementato dal Prof. Matteucci, il quale consente di identificare insiemi di pixel classificati come appartenenti alla stessa classe sull'immagine in analisi, rilevando così gli eventuali oggetti di interesse.
Questo sistema permette facilmente di eliminare eventuali falsi positivi tramite l'implementazioni di alcuni filtri, sulla base delle dimensioni delle regioni e sulla sua posizione all'interno dell'immagine. (Figura 6.9)

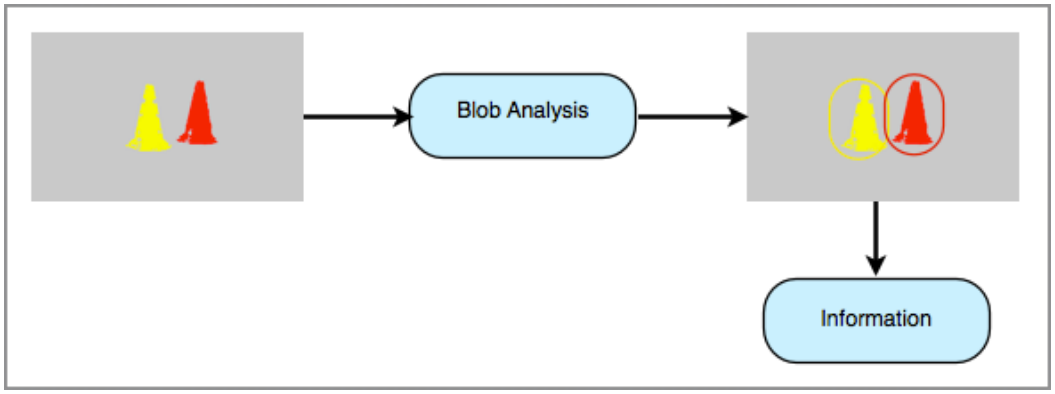


Figura 6.9: Risultati ottenuti con l'applicazione dell'algoritmo di Blob Analysis.

Le informazioni estratte dalle varie elaborazioni, eseguite su tutte le immagini proveniente dal robot, permettono di eseguire un robusto controllo dei motori e del comportamento del robot stesso.

Dopo ripetuti test, per il classificatore KNN qui utilizzato sono stati fissati i seguenti valori:

- $K = 10$ (Numero di K punti da considerare)
- d = 10 (Distanza massima tra il campione considerato ed il punto più vicino).

Messaggi in ingresso ed in uscita dal modulo:

Inizialmente vengono inoltre eseguiti dei controlli sui messaggi in arrivo dagli altri moduli del sistema ed alla fine di ogni ciclo di esecuzione del modulo in esame, vengono raccolte tutte le informazioni utili da inviare al modulo di gestione dei comportamenti *BrianExpert*.

6.1.4 Modulo: BrianExpert

BrianExpert è il modulo che controlla la logica di gioco e definisce quali comportamenti attivare in base ai messaggi che gli vengono inviati dagli altri expert.
In questa sede, sono stati implementati 6 comportamenti distinti che verranno di seguito analizzati, insieme agli ingressi ed alle uscite del modulo in analisi.

Comportamenti

I comportamenti implementati per il presente progetto sono i seguenti:

- *Go_To_Home:* permette al robot di raggiungere la casa base e quindi vincere la partita
- *Go_To_Energy:* permette al robot di raggiungere, quando necessario, i punti di ricarica dell'energia
- *Search_Home:* permette al robot di ricercare nell'ambiente circostante la casa base
- *Search_Energy:* permette al robot di ricercare nell'ambiente circostante i punti di ricarica dell'energia
- *RobotWin:* rappresenta lo stato in cui il robot ha raggiunto la casa base e quindi vinto la sessione di gioco.
- *RobotRecharge:* rappresenta lo stato in cui il robot ha raggiunto un punto di ricarica dell'energia.

Ingressi

Tutti gli ingressi al modulo in esame subisco una fase di *fuzzyficazione* in base alle membership functions specificate nei file di configurazione.

In particolare, sono state definite le seguenti tipologie di ingresso:

- *RANDOM_VAR:* si occupa di modellare una variabile casuale che viene utilizzata per attuare i comportamenti di ricerca di casa-base e punti di ricarica e per evitare gli ostacoli. A seconda del valore di tale variabile, dunque, il robot eseguirà tali comportamenti in modo diverso.

```
(RANDOM_VAR
(SNG (RAN_LOW 0))
(SNG (RAN_HIGH 1))
)
```

- *OBJECTDISTANCE:* rappresenta la distanza misurata da ogni singolo sonar montato sul robot. Tali ingressi vengono utilizzati per lo più per l'attuazione del comportamento di *ObstacleAvoidance*

```
(OBJECTDISTANCE
(TOL (VERYNEAR 350 450))
(TRA (NEAR 350 450 700 800))
(TOR (FAR 700 800))
)
```

- *OBJECTPOSITION:* rappresenta la posizione del blob individuato nel frame corrente. Viene utilizzato dai comportamenti "*Go_To*" per raggiungere e centrare l'oggetto in analisi.

```
(OBJECTPOSITION
(TOL (LEFT 40 50))
(TRA (CENTER 40 50 270 280))
(TOR (RIGHT 270 280))
)
```

- *ENERGYLEVEL:* indica il valore attuale dell'energia del robot.

```
(ENERGYLEVEL
(TOL (LOW 20 30))
(TOR (HIGH 20 30))
)
```

- *TRUE_FALSE:* variabile utilizzata per fuzzyficare gli ingresso booleani.

```
(TRUE_FALSE
(SNG (T 1))
(SNG (F 0))
)
```

Uscite

Sulla base degli ingressi precedentemente presentati, Mr.Brian si occupa di verificare l'attivabilità dei comportamenti specificati.

Un comportamento viene quindi attivato sulla base delle condizioni specificate nei file di configurazione sotto forma di predicati fuzzy; di particolare rilevanza sono i file CANDO e WANT.

Per maggiori circa l'utilizzo di Mr. Brian si rimanda alla documentazione presente sull'AirWiki [2].

In particolare, sono state definite le seguenti uscite:

- *TANSPEED:* esprime il set point di velocità tangenziale da imprimere ai motori

```
(TANSPEED
(SNG (VERY_FAST_FORWARD 80))
(SNG (FAST_FORWARD 50))
(SNG (FORWARD 35))
(SNG (SLOW_FORWARD 20))
(SNG (VERY_SLOW_FORWARD 10))
(SNG (STEADY 0))
(SNG (VERY_SLOW_BACKWARD -10))
(SNG (SLOW_BACKWARD -20))
(SNG (BACKWARD -35))
(SNG (FAST_BACKWARD -50))
(SNG (VERY_FAST_BACKWARD -80))
)
```

- *ROTSPEED:* esprime il set point di velocità rotazionale da imprimere ai motori

```
(ROTSPEED
(SNG (VERY_FAST_RIGHT 80))
(SNG (FAST_RIGHT 40))
(SNG (RIGHT 30))
(SNG (SLOW_RIGHT 15))
(SNG (VERY_SLOW_RIGHT 1))
(SNG (AHEAD 0))
(SNG (VERY_SLOW_LEFT -1))
(SNG (SLOW_LEFT -15))
(SNG (LEFT -30))
(SNG (FAST_LEFT -40))
(SNG (VERY_FAST_LEFT -80))
)
```

- *WIN:* indica lo stato di vittoria da parte del robot

```
(WIN
(SNG (F 0))
(SNG (T 1))
)
```

- *RECHARGE:* indica il raggiungimento, da parte del robot, di un punto di ricarica dell'energia.

```
(RECHARGHE
(SNG (F 0))
(SNG (T 1))
)
```

Tutti i valori elaborati in tale modulo, verranno opportunamente inviati a "*Motor Expert*" per essere attuati dai motori del robot.

6.1.5 Modulo: MotorExpert

Il modulo "*MotorExpert*" ha come unico compito quello di ricevere in ingresso i messaggi provenienti da "*BrianExpert"* e tramutare i valori così ottenuti in comandi per i motori.

In modo particolare, il modulo *BrianExpert* invia i valori calcolati relativi alle velocità tangenziale e rotazionale ed il modulo "*MotorExpert*" si occupa di trasformare tali valori nelle velocità da imprimere ai cingoli di cui il robot SpyKee è dotato.

Tale trasformazione avviene rispettando le seguenti due regole:

cingolo destro = velocità tangenziale + velocità rotazionale
cingolo sinistro = velocità tangenziale - velocità rotazionale

6.2 Hardware utilizzato

In questa sezione verrà definito l'hardware utilizzato per la realizzazione del gioco RoboWii 2.1; ovvero:

- Robot autonomo SpyKee
- Controller Wii Remote
- Cilindri di cartone di colore giallo per indicare la casa base
- Cilindro di cartone di colore rosso per indicare i punti di ripristino dell'energia
- Un PC/MAC come unità di elaborazione esterna
- Interfacce di comunicazione di rete

Nelle sezioni successive verranno analizzati, con maggiore dettaglio, tutti i precedenti componenti, spiegando il ruolo che essi occupano all'interno del gioco.

6.2.1 Robot SpyKee

Per l'implementazione del gioco è stato scelto il robot SpyKee - Figura 6.10 - disponibile nel laboratorio di robotica ed intelligenza artificiale (AIRLab) del Politecnico di Milano.

Figura 6.10: Il robot SpyKee utilizzato nell'implementazione del gioco.

SpyKee, insieme al giocatore umano, è l'attore principale del gioco, il suo ruolo consiste nel competere con il giocatore per la vittoria finale. Esso è dotato di movimento mediante cingoli, in modo da poter raggiungere i principali traguardi ed evitare i colpi "lanciati" dall'avversario umano.

La scelta del robot è stata condizionata dalle numerose funzionalità che spyKee mette a disposizione.

Il robot è dotato di una telecamera posizionata sulla fronte, la quale è in grado di scattare fotografie e registrare video a 15fps, anche se con bassa risoluzione. La telecamera sarà sfruttata dal modulo di visione del robot per individuare, nell'ambiente circostante, la casa base e i punti di ricarica.

Oltre alla telecamera, il robot è stato dotato di una cintura di sensori sonar (8 sensori in corrispondenza dei 4 punti cardinali Nord, Sud, Est ed Ovest e 4 per NE, NW, SE e SW). Tali sensori sono stati utilizzati per il rilevamento e l'aggiramento degli ostacoli incontrati dal robot.
La cintura di sensori, non presente sul robot originale, è stata aggiunta successivamente tramite l'utilizzo di un modulo Xbee che si occupa di controllare anche i vari Led posizionati sul robot.

Come già detto in precedenza, il robot è dotato di due strisce di LEDs gialli e rossi e due LEDs infrarossi posizionati sulle spalle, oltre ad un LED verde posto sulla sua testa.
Tutti questi dispositivi (LEDs gialli esclusi) giocano un ruolo fondamentale nell'interazione uomo-robot, infatti:

- i LED infrarossi sono usati per permettere il puntamento del robot mediante WiiMote;
- i LED rossi permettono al giocatore di sapere in tempo reale quale sia lo stato del robot circa le unità di energia a disposizione;
- il LED verde permette al giocatore di capire quando la fase di targeting è andata a buon fine.

6.2.2 Il controller Wii Remote

Il controller WiiMote è lo strumento principale utilizzato dal giocatore umano per interagire con il robot. Tramite i sensori presenti al suo interno, il giocatore potrà eseguire tutti i movimenti ed i comandi descritti nel capitolo 5, mentre con la telecamera ad infrarossi potrà mantenere il robot sotto puntamento.

6.2.3 Identificatori dei punti di recupero energia e della casa base

Per definire la posizione dei punti di recupero di energia e della casa base e rendere tale informazione chiara sia al robot che al giocatore, è stato scelto di utilizzare dei semplici cilindri di cartone, di altezza di circa 20 cm, di colori differenti.

Il numero di punti di recupero di energia non è prestabilito a priori, quindi è modificabile a piacimento; per tale scopo sono stati impiegati dei cilindri di colore rosso.

Per quanto riguarda la casa base, invece, è stato pensato di utilizzare un cilindro identico ai precedenti ma di colore giallo.

È importante sottolineare che il robort inizialmente non è a conoscenza della posizione dei vari cilindri e del loro numero. Inoltre non è in grado di memorizzare la loro posizione durante la sezione di gioco.

6.2.4 Unità di elaborazione esterna

Tutta l'elaborazione (troppo complessa ed onerosa dal punto di vista computazionale, dei dati e delle immagini provenienti sia dal robot che dal Wii Remote) è gestita da un calcolatore esterno, il quale provvederà a ricevere i dati, ad elaborarli e a trasmettere i risultati alle interfacce che si occuperanno di tradurli in azioni. L'unità di elaborazione esterna svolgerà dunque il ruolo di *sistema.*

6.2.5 Interfacce di comunicazione di rete

Utilizzando un'unità di elaborazione esterna è necessario utilizzare opportuni protocolli di rete per convogliare i dati all'interno del sistema principale e per inviare i dati elaborati alle interfacce che si occuperanno di trasformarli in azioni reali.

Per quanto riguarda la comunicazione con il robot autonomo (SpyKee) sono stati utilizzati due sistemi di comunicazione senza fili. L'utilizzo di due sistemi è dovuto sostanzialmente alle modifiche eseguite sul robot (Sonar e LED) che non potevano essere controllate con il medesimo sistema di controllo del robot.

Il robot è controllato tramite connessione Wifi che garantisce stabilità di connessione e banda sufficiente per la trasmissione delle immagini provenienti dalla telecamera del robot (15 per secondo).
Inoltre la comunicazione tra il sistema di controllo dei sensori di prossimità e dei LED infrarossi e l'unità esterna di elaborazione è gestito tramite il modulo Xbee. Come detto in precedenza, l'utilizzo di tale sistema è stato dettato dal fatto che essi sono tutti componenti aggiuntivi non presenti originariamente nel robot e quindi non integrabili nel suo sistema di controllo. Inoltre, un altro aspetto non trascurabile che ha fatto prediligere questo standard rispetto ad altri, è stato il suo basso consumo energetico.

Il protocollo di comunicazione tra il WiiMote e l'unità di elaborazione esterna, infine, avviene tramite una connessione Bluetooth, esattamente come avviene sulla console Nintendo Wii.

6.3 Studio di fattibilità

Questa sezione si pone l'obiettivo di introdurre le principali problematiche (e relative soluzioni) che sono state incontrate, sia nella prima fase di progetto, sia durante l'implementazione dei vari moduli.

Lo studio di fattibilità, quanto mai necessario quando ci si pone l'obiettivo di implementare un sistema di controllo intelligente capace di interagire sia con il mondo esterno che con un giocatore umano, ha permesso di scegliere le soluzioni più adatte (secondo noi) sulla base dell'hardware a disposizione.

I punti toccati da tale studio sono molteplici, tra i principali troviamo:

- *Definizione dei movimenti eseguibili dal giocatore umano e gestione dell'interfaccia utente.*
- *Gestione del sistema di obstacle avoidance*
- *Definizione dei materiali da utilizzare per l'identificazione della casa base e dei punti di recupero di energia*
- *Gestione dei sistema di puntamento del robot da parte del giocatore*

6.3.1 Interfaccia utente e definizione dei movimenti

Il primo modulo ad essere implementato è stato quello relativo alla gestione delle funzionalità attuabili dall'utente.
Le problematiche principali di tale modulo sono da associare alla definizione di un insieme di movimenti (ricarica e selezione delle armi) che siano al tempo stesso intuitivi da parte dell'utente, ma che permettano la creazione di un sistema di classificazione robusto e con un margine d'errore molto basso.

Il controller Wii Remote è sembrata subito la scelta più corretta proprio per le sue caratteristiche, che consentono una elevata interazione tra il giocatore ed il mondo che lo circonda.
Esso, infatti, è dotato di un accelerometro, i cui dati appresi hanno permesso la classificazione dei movimenti

Inizialmente si era pensato di utilizzare, oltre all'accelerometro, un'espansione del Wii Remote nota con il nome di Wii Motion Plus [17], prodotta sempre dalla Nintendo, che consente di aumentare sia la precisione del Wii Remote, sia il numero di movimenti che esso può rilevare. Tale espansione consiste sostanzialmente nell'aggiunta di un sensore giroscopico.
La collaborazione giroscopio-accelerometro, dunque, permette di tracciare nello spazio una qualsiasi traiettoria compiuta dal controller prendendo in considerazione non più le sole forze esercitate dal braccio dell'utente, ma anche le rotazioni compiute dal controller sui propri assi.
L'utilizzo delle librerie WiiCpp tuttavia, non ha permesso l'utilizzo di tale sistema in quanto in fase di test si sono riscontrati numerosi problemi di connessione tra il WiiRemote e la sua espansione.

Si è scelto quindi di definire dei movimenti che risultassero, dal punto di vista delle forze impresse sul controller, molto diversi tra loro, in modo da poter eseguire una classificazione robusta.

Altra limitazione è stata imposta dalle librerie Wiicpp le quali non hanno permesso, ad esempio, la gestione dell'altoparlante del Wii Remote.

Limitazioni sono state imposte anche dall'utilizzo delle librerie per il controllo di spyKee, le quali sono state implementate tramite un attento lavoro di reverse engineering sui messaggi scambiati tra calcolatore e robot, durante l'utilizzo dello stesso, con i programmi forniti dal costruttore. Tali librerie non hanno permesso, ad esempio, la gestione dei suoni riproducibili dal robot, che sicuramente avrebbero potuto incrementare il coinvolgimento dell'utente nel gioco.

6.3.2 Gestione del sistema di obstacle avoidance

Il Robot da noi utilizzato per lo sviluppo del gioco è stato dotato di una cintura di sensori sonar (8 per la precisione) in modo da poter implementare rapidamente un sistema di riconoscimento degli ostacoli.

La gestione, l'interpretazione e l'utilizzo dei dati provenienti da tale strumentazione sono già stati oggetto di studio dei progetti RoboWii 2.0 con esito positivo e quindi sono state qui riportate, apportando solo qualche modifica per meglio adattarli al progetto in esame.

6.3.3 Sistema di visione

Il sistema di visione qui implementato è basato sull'utilizzo di un insieme di algoritmi già impiegati nel progetto di Tesi di uno dei componenti di questo team [18], con l'obiettivo di identificare gli occhi e la pupilla all'interno di un'immagine di un volto umano. Il sistema inoltre è lo stesso utilizzato nella RoboCUP.

Il sistema di visione è fortemente condizionato dalla scelta dei materiali e dei colori per la rappresentazione degli oggetti di interesse per la rilevazione.

Altra difficoltà è dovuta alla qualità delle immagini provenienti dalla telecamera di SpyKee, immagini a bassa risoluzione e decisamente mosse a causa delle vibrazioni del robot durante il movimento e dal frameRate di soli 15fps.

Dopo svariati test, per individuare la scelta del colore migliore, siamo giunti alla conclusione che i colori opachi, poco riflettenti alla luce, sono i più adatti per le analisi che devono essere eseguite. Tali colori infatti variano di poco a seconda delle condizioni di luce dell'ambiente e anche con immagini a bassa risoluzione, come quelle a disposizione, hanno permesso l'ottenimento di risultati soddisfacenti. Tuttavia si deve garantire anche che essi risultino ben distinguibili sul campo di gioco.

È stata inoltre valutata la possibilità di far indossare un giubbotto, opportunamente colorato, al giocatore umano in modo da rendere possibile il suo riconoscimento da parte del robot. Questa soluzione infatti è stata proposta da numerosi progetti nel campo RoboWii. Tuttavia la telecamera di SpyKee, ancora una volta, ha imposto forti limitazioni, impedendo l'implementazione di tale sistema di riconoscimento.
Il problema principale è infatti da ricercarsi nella necessità primaria che tale telecamera inquadri costantemente il campo di gioco, in modo da rendere possibile l'identificazione della casa base e dei punti di recupero di energia. Il ristretto angolo di veduta della telecamera non permette tuttavia di inquadrare, nel contempo, il campo di battaglia ed il giocatore il quale uscirebbe dall'inquadratura.
Sviluppi futuri potrebbero sicuramente optare per la sostituzione della telecamera in dotazione, fonte - come dimostrato - di grandi limitazioni.

6.3.4 Puntamento del WiiMote

Il puntamento del WiiMote verso il robot ha, come idea base, quella di sfruttare la telecamera IR presente all'estremità di ogni controller Wii Remote. Tramite questa telecamera vengono infatti rilevati i LED infrarossi montati sul robot, in particolare è possibile ottenere le coordinate e la distanza di 4 fonti luminose contemporaneamente.

Il posizionamento dei vari LED sul robot è già stata fonte di studio nei progetti precedenti. I due LED infrarossi sono posizionati sulle spalle del robot in modo da essere facilmente visibili dalla telecamera del controller con un puntamento diretto alla schiena del robot stesso - Figura 6.11.
Tale sistema è stato utilizzato anche per determinare la distanza tra il giocatore umano ed il robot, in modo di controllare la distanza minima che il giocatore deve mantenere prima di poter eseguire il puntamento del robot. Le librerie utilizzate per la comunicazione con il WiiMote permettono, infatti, completo accesso ai dati relativi alla telecamera in modo di controllare le distanze tra la telecamera e le sorgenti infrarosse.

Figura 6.11: Puntamento del giocatore alla schiena del robot

7. Scenari di gioco

In questo capitolo verranno illustrati due principali scenari d'uso del gioco implementato. In particolare, verranno presentate due partite tipiche, una prima, nella quale il robot uscirà come vincitore, ed una seconda dove il robot uscirà sconfitto a favore del giocatore umano.

7.1 Scenario di vittoria del robot

In questo scenario viene descritta una tipica partita a RoboWii 2.1, terminante con la vittoria del robot SpyKee. Di seguito sono elencati i passi salienti:

- Il robot parte alla ricerca della casa base
- Il giocatore impugna la pistola e punta il robot
- Il robot cerca di evitare il puntamento
- Il giocatore fa fuoco mentre i 4 LED sul WiiMote sono accesi e colpisce il robot
- Il robot ricerca ed individua un punto di ricarica di energia e si dirige verso di esso
- il giocatore ripone la pistola e seleziona il fucile
- Il robot raggiunge il punto di ricarica
- Il giocatore punta il robor facendo accendere il LED verde del robot
- Il robor cerca di evitare il puntamento

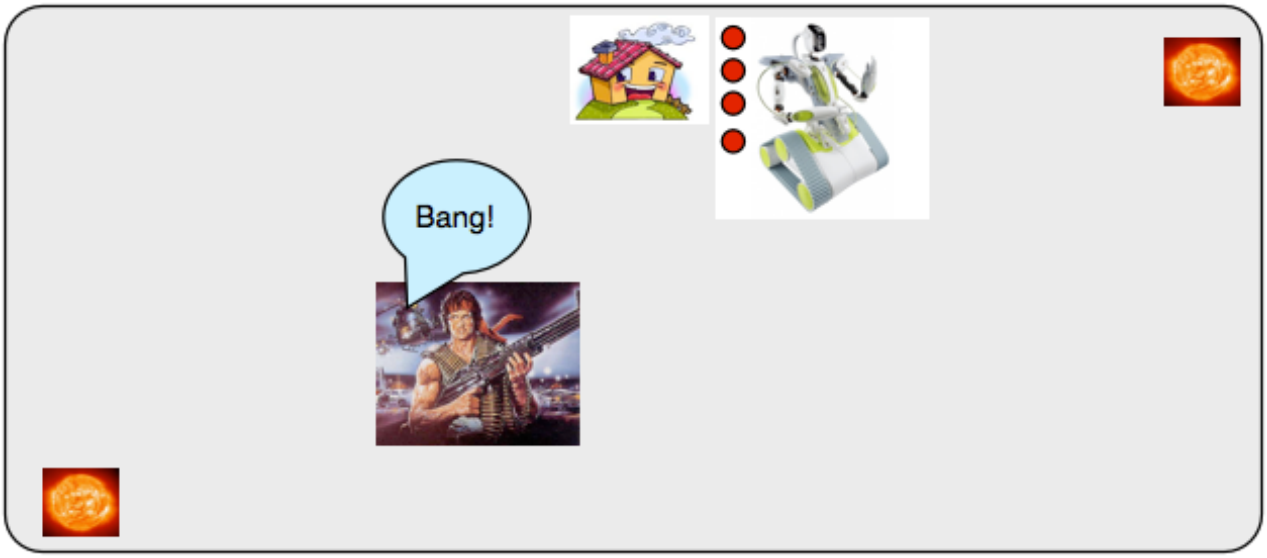
- Il giocatore spara ma manca il bersaglio

- Il robot individua la casa base e la raggiunge con il massimo dell'energia, mentre il giocatore è intendo a ricaricare l'arma e prendere la mira.

(a): Il gioco ha inizio. SI ricorda che il robot non conosce la locazione dei punti di energia e della casa base. Esso ricerca la casa base e cerca di raggiungerla, intanto il giocatore impugna la pistola,

(b): Il giocatore mira il robot che cerca di scappare, spara e colpisce il robot facendogli perdere 1 unità di energia. L'obiettivo del robot diventa ora raggiungere un punto di ricarica energia ed inizia la ricerca

(c): Il robot trova e raggiunge il punto di ricarica e la sua energia viene ripristinata, mentre il giocatore decide di cambiare la pistola con il più potente fucile.

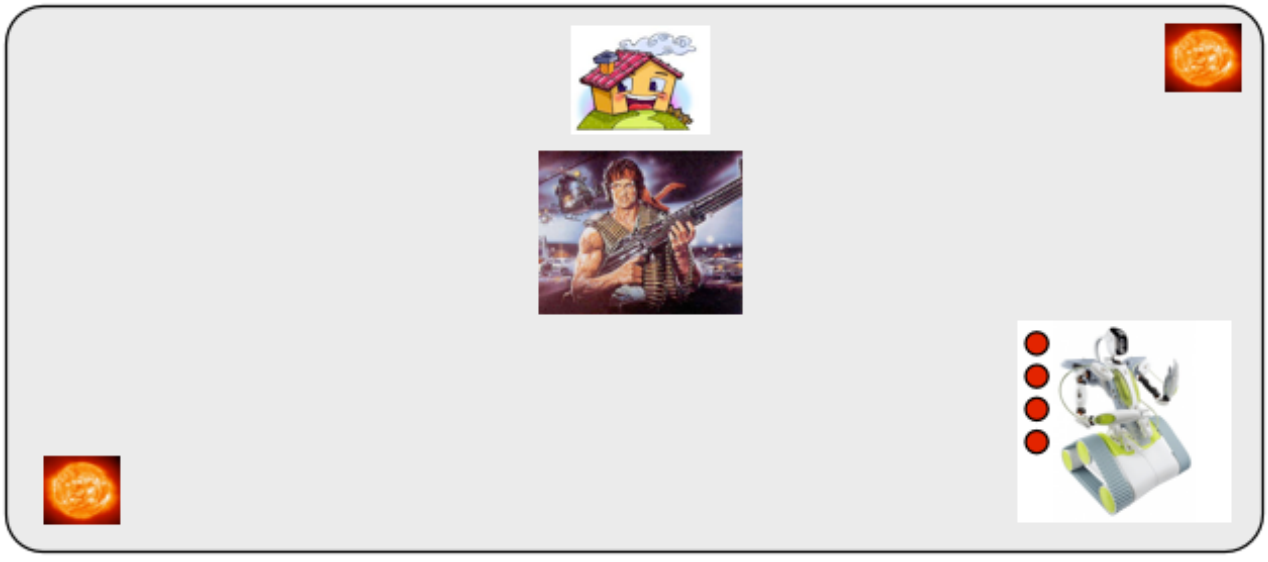


(d): Il robot individua la casa base e cerca di raggiungerla a velocità massima, mentre il giocatore mira al robot spara ma manca il bersaglio, consegnando la vittoria al robot che raggiunge la casa base.

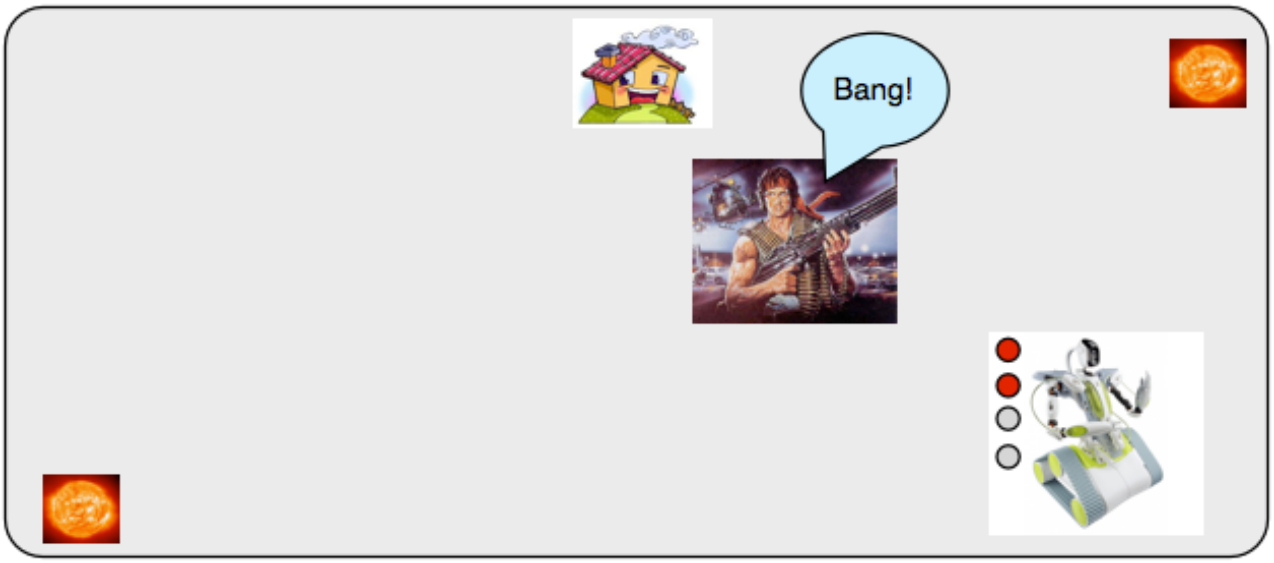
7.1 Scenario di vittoria del giocatore

In questo scenario viene descritta una tipica partita in cui la vittoria verrà assegnata al giocatore umano.

- Il robot parte alla ricerca della casa base
- Il giocatore estrae il fucile e punta il robot
- Il giocatore attende il puntamento e fa fuoco
- Il robot perde 2 unità di energia a causa del colpo appena subito
- Il robot cerca di raggiungere un punto di ricarica di energia
- Il giocatore estrae la spada e colpisce il robot che esaurisce l'energia e consegna la vittoria al giocatore

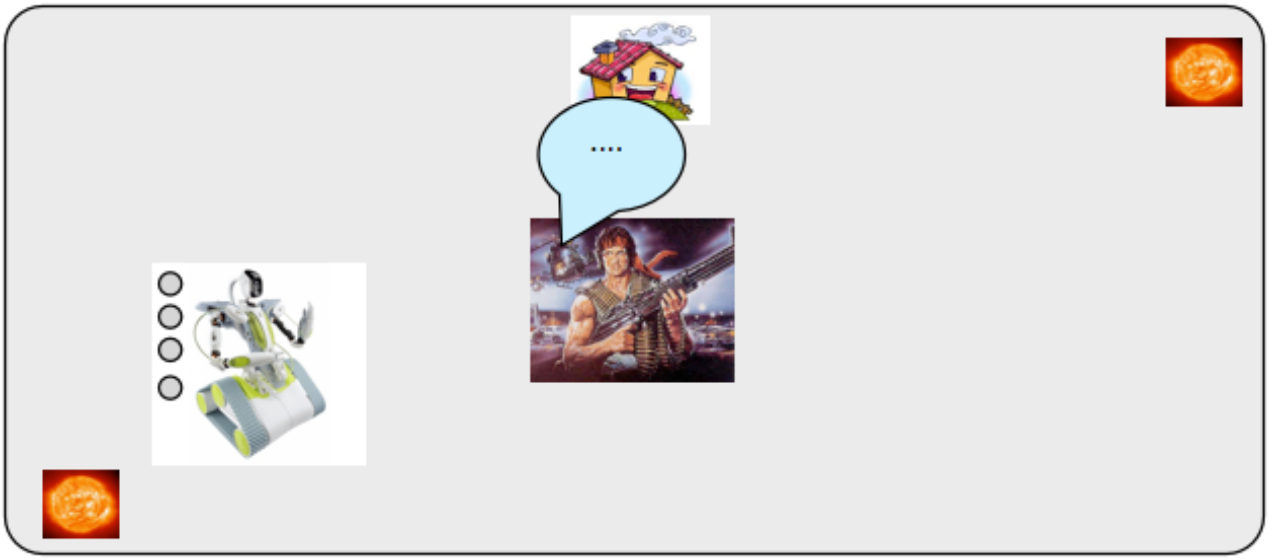


(a): Il gioco ha inizio, il giocatore impugna il fucile e punta il robot, che intanto ricerca la casa base



(b): Il giocatore prende la mira e spara al robot; quest'ultimo perde due punti di energia, ed inizia la ricerca di un punto di ricarica

(c): Il robot trova il punto di ricarica e data la distanza da esso cerca di raggiungerlo a massima velocità, il giocatore intanto impugna la Katana.



(d): Il giocatore precede il robot e sfodera un taglio netto al centro dell'automa, il quale perde tutta la sua energia. Il giocatore ha vinto!

8. Conclusioni finali

La realizzazione di questo progetto ha preso in esame diverse problematiche relative alla realizzazione di un gioco di interazione tra un robot ed un umano. La progettazione dei vari sistemi che permettono l'ottenimento dei risultati raggiunti ha permesso di capire ed evidenziare quali fossero gli aspetti essenziali per la riuscita del progetto, a partire dalla semplice idea di gioco, fino all'implementazione delle più piccole funzionalità che devono essere correttamente gestite dal sistema.

Durante la progettazione sono state analizzate numerose alternative, le quali prevedevano situazioni aggiuntive a quelle implementate, come la capacità del robot di sparare, l'utilizzo di più armi contemporaneamente etc. Tuttavia sia le tempistiche imposte dai corsi accademici sia le risorse hardware utilizzate non hanno permesso lo sviluppo di tali miglioramenti che rimangono quindi un ottimo punto di partenza per eventuali sviluppi futuri.

Appendice A: Manuale utente

1. Installazione del software:

Per utilizzare il sistema implementato è sufficiente copiare dal DVD (fornito con il presente documento) la cartella /Progetto/MRT, sul proprio computer.
Il sistema è stato correttamente testato su un MacBookPro intel con sistema operativo Linux (Ubuntu 10.04 LTS).

1.1 Compilazione

Se vengono eseguite delle modifiche ai vari moduli (o librerie) del sistema è necessario compilare tutto il progetto, a partire dai singoli moduli modificati.

Per la compilazione di un modulo *moduleName* è sufficiente accedere alla cartella *moduleName/src* ed eseguire il comando

```
MacBookPro:>  make lib
```

Se necessario, è possibile compilare anche tutti gli esperti presenti all'interno della cartella *robot/ src* con il medesimo comando visto sopra.

Per generare l'eseguibile (nella cartella *spykee),* è invece necessario eseguire il seguente comando dalla cartella *MRT*.

```
MacBookPro:>  make ROBOT=spyKee
```

2. Avvio del sistema.

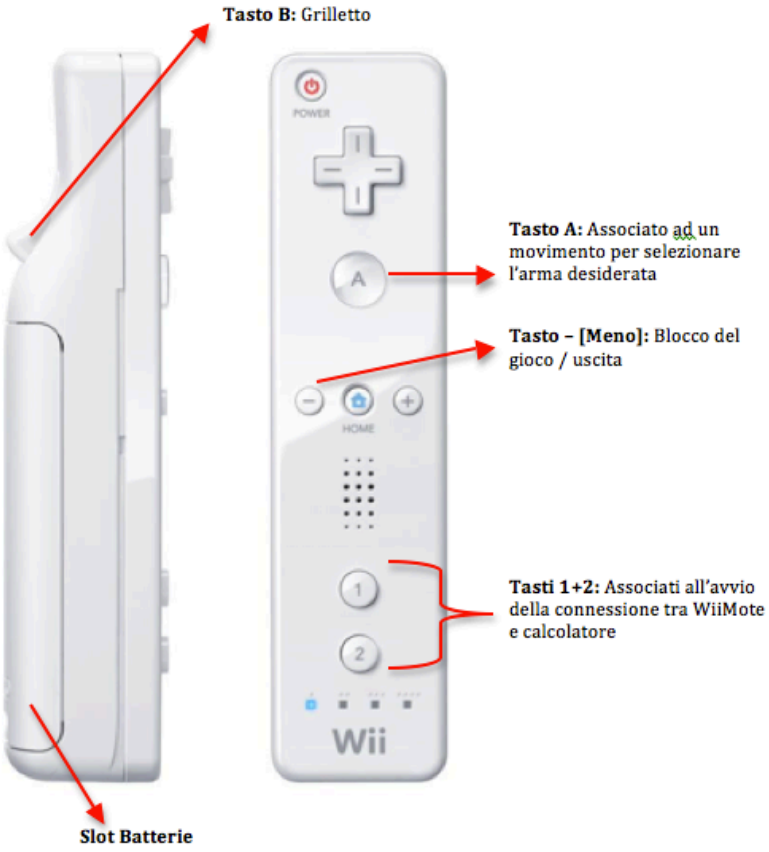
È sufficiente eseguire l'eseguibile con il comando

```
MacBookPro:> ./MRT/spyKee/SpyKee
```

All'avvio, il sistema controllerà che il computer ed il robot siano collegati via Wifi, e ed eseguirà una ricerca per collegarsi al Wii Remote. In questa fase è sufficiente seguire le istruzioni sullo schermo del computer, concluse le quali il gioco verrà avviato in modo automatico dopo qualche istante di attesa.

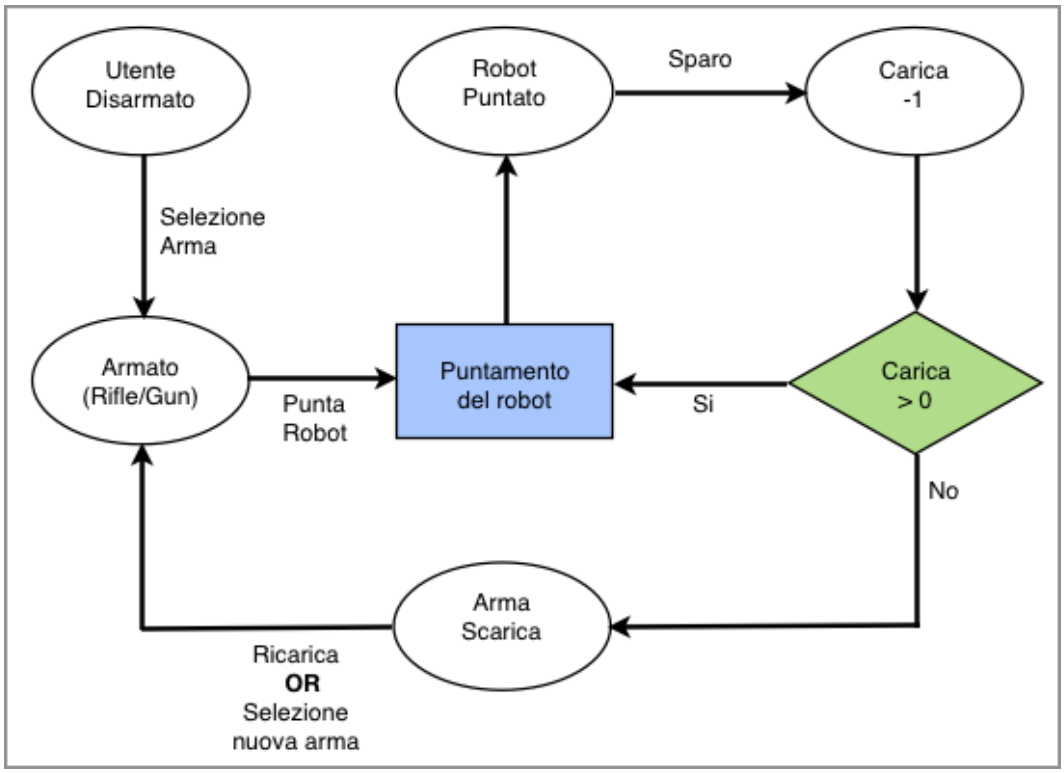
3. Come giocare

Per prima cosa viene fornita, nell'immagine seguente, una rapida descrizione dei vari comandi disponibili con relative funzioni:



3.1 Sparare al robot

Lo schema seguente indica le azioni che devono essere compiute, per abilitare l'arma selezionata.



Per eseguire un colpo è necessario nell'ordine:

- Selezionare l'arma desiderata mantenendo premuto il pulsante A ed eseguendo il movimento corretto(per i movimenti associati alle armi vedere i video tutorial presenti nel DVD nella sezione VideoGuida).
- Puntare l'arma sul robot per qualche secondo sino a quando i LED del controller non rimangono accesi
- Mirare al robot
- Sparare premendo il tasto B
  - A seconda dell'arma selezionata è possibile eseguire in successione diversi colpi (4 per la pistola, 1 per il fucile).
  - Se l'arma è scarica è possibile ricaricarla tramite l'opportuno movimento (Pulsante A premuto).

3.1 Colpire il robot con la katana

Lo schema seguente indica le azioni che devono essere compiute, per abilitare la katana.

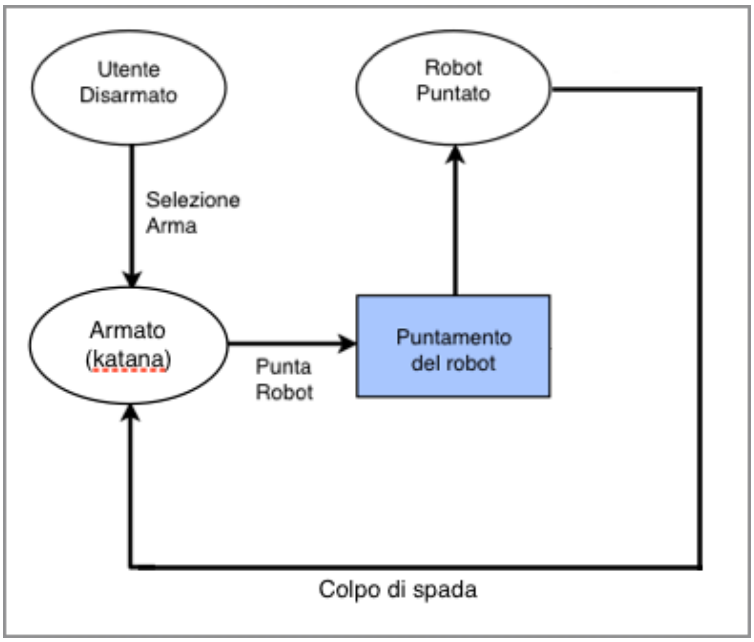


Figura 6.2: Modulo di gestione della Katana del giocatore.

Per eseguire un colpo è necessario nell'ordine:

- Selezionare l'arma desiderata mantenendo premuto il pulsante A ed eseguendo il movimento corretto(per i movimenti associati alle armi vedere i video tutorial presenti nel DVD nella sezione VideoGuida).
- Puntare l'arma sul robot per qualche secondo sino a quando il LED verde del robot non si accende
- Mantenendo premuto il tasto B del controller simulare l'esecuzione di un colpo di spada cercando di passare per il centro esatto del robot.

Appendice B: Codice

Il codice, per motivi di spazio é stato inserito solamente all'interno del DVD allegato.

Bibliografia

[1] - *Controller Wii Remote:* http://en.wikipedia.org/wiki/Wii_Remote

[2] - *Sistema MRT:* http://airwiki.ws.dei.polimi.it/index.php/MRT

[3] - *RoboWii: http://airwiki.ws.dei.polimi.it/index.php/ROBOWII*

[4] - *RoboWii 1.0:* http://airwiki.ws.dei.polimi.it/index.php/ROBOWII#RoboWII_1.0

[5] - *RoboWii 1.0:* http://airwiki.ws.dei.polimi.it/index.php/RoboWII1.0

[6] - *LURCH:* http://airwiki.ws.dei.polimi.it/index.php/Lurch

[7] - *RoboWii 2.0:* http://airwiki.ws.dei.polimi.it/index.php/RoboWII2.0

[8] - *SpyKee: http://airwiki.ws.dei.polimi.it/index.php/Spykee*

[9] - *RoboWii 2.0L: http://airwiki.ws.dei.polimi.it/index.php/RoboWII2.0.L*

[10] - *Lego Midstorms: http://airwiki.ws.dei.polimi.it/index.php/Lego_Mindstorms_NXT*

[12] - *RoboWii 3.0: http://airwiki.ws.dei.polimi.it/index.php/RoboWII3.0*

[13] - *WIICpp: http://wiic.sourceforge.net/*

[14] - *WiiUse: http://wiiuse.sourceforge.net/*

[15] - *ANN: http://www.cs.umd.edu/~mount/ANN/*

[16] - *Interpolazione Hermite: http://en.wikipedia.org/wiki/Hermite_interpolation*

[17] - *Wii MotionPlus: http://en.wikipedia.org/wiki/Motion_Plus*

[18] - *Tesi Cristian Mandelli: http://airwiki.ws.dei.polimi.it/index.php/Interpretation_of_facial_expressions_and_movements_of_the_head*