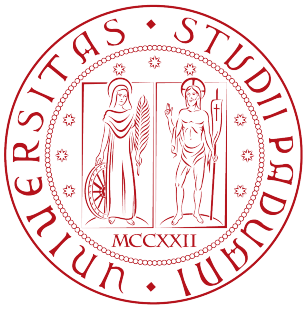


UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA
DELL'INFORMAZIONE

Corso di Laurea in
Ingegneria Biomedica

Strumenti non convenzionali per la valutazione dell'equilibrio posturale

Relatore
Prof. Piero Pavan

Laureando
Giacomo Boioli

Anno Accademico 2012/2013

ai miei genitori

SOMMARIO

Questo studio riprende quanto già svolto in precedenti lavori sullo sviluppo di metodi non convenzionali per lo studio dell'equilibrio, per proporre uno strumento completo e portabile, pronto per l'utilizzo.

Il lavoro si propone due principali obiettivi. Il primo prevede lo sviluppo di un programma MATLAB che, tramite un'interfaccia grafica, permetta di interagire con due Wii Balance Board (WBB). Il secondo obiettivo riguarda l'intenzione di rendere il programma creato portabile e indipendente con la creazione di un programma stand-alone.

Per la creazione del programma si è partiti da un software, chiamato *bbrecord*, realizzato presso l'Università del Colorado. Lo sviluppo prevede una completa ristrutturazione di *bbrecord* organizzando il software per la successiva procedura di realizzazione di uno stand-alone e implementando la possibilità di collegare più pedane contemporaneamente.

Il collegamento di due WBB rende possibile l'utilizzo di questo strumento anche per monitorare la capacità di equilibrio di soggetti obesi. Nelle specifiche di utilizzo della pedana, infatti, è indicato un peso massimo per l'utilizzatore di 150 kg, mentre il peso di un soggetto affetto da obesità grave può superare di molto questo limite.

La necessità di ottenere un prodotto a basso costo, infine, rende indispensabile la creazione di un software eseguibile libero da licenza che possa essere distribuito gratuitamente. Matlab, infatti, è distribuito solo tramite l'acquisto di licenza. L'importo relativamente basso di una WBB (circa 70 euro), l'indipendenza del programma eseguibile dal software di partenza e l'intuitività dell'interfaccia grafica conferiscono al progetto caratteristiche di semplicità ed economicità che lo rendono ideale come strumento da campo.

Indice

INTRODUZIONE

La capacità di equilibrio di una persona può essere definita come l'abilità di mantenere il corpo in una determinata posizione. L'equilibrio può essere statico, quando il corpo è fermo, o dinamico, quando il corpo è in movimento. Esso è fondamentale per la maggior parte delle attività quotidiane ed è acquisita dall'uomo durante i primi anni di vita.

La caratteristica posturale dell'uomo di poter stare in posizione eretta su due piedi, unita alle capacità di potersi spostare con un solo piede a contatto con il suolo (ad esempio durante la camminata) o senza appoggi (durante la corsa), mettono a dura prova il nostro sistema di controllo dell'equilibrio[1].

Il complesso meccanismo di stabilizzazione è controllato da tre principali sistemi: il sistema vestibolare, il sistema visivo e il sistema propriocettivo. Il sistema vestibolare è situato nell'orecchio interno e fornisce indicazioni sulla posizione della testa e del corpo, percependo accelerazioni verticali e orizzontali. Il sistema visivo si occupa di trasmettere informazioni circa la posizione del corpo nell'ambiente circostante. Il sistema propriocettivo, infine, si compone di recettori interni al nostro organismo e si occupa di definire, ad esempio, le posizioni relative delle membra che formano un'articolazione oppure di fornire le informazioni recepite dalla pelle quali vibrazioni o pressioni.

Il sistema di stabilizzazione può essere interessato da varie patologie che rendono precaria la condizione di stabilità di un soggetto, la perdita di capacità di equilibrio, inoltre, non è solo dovuta alla presenza di una specifica patologia ma può essere legata a particolari situazioni. Si osserva, infatti, che ci sono categorie di persone in cui è evidente una riduzione della capacità di equilibrio; in particolare soggetti anziani e obesi possono presentare notevoli difficoltà nel mantenimento di una condizione di stabilità.

Ogni anno, infatti, il 30-60% di anziani perde l'equilibrio e cade, il 10-20% di questi incidenti porta a gravi infortuni o morte[2]. Queste cadute, con le conseguenze che ne derivano, sono associate a paura, perdita di confidenza e alla limitazione delle attività svolte[3].

Anche nei soggetti affetti da obesità si può osservare una perdita del senso di equilibrio. Effettivamente, nella comparsa delle disabilità correlate all'aumento di peso, vengono dapprima intaccate le funzioni relative agli arti inferiori (forza e mantenimento dell'equilibrio)[4].

Riuscire a quantificare la capacità di equilibrio di un soggetto risulta quindi molto importante sia in ambito sportivo che per quanto riguarda la possibilità di prevenire cadute nei soggetti a rischio.

Sebbene il "tempo per cui si è riusciti a mantenere una data postura" sia una misura clinica molto utile, in molti studi sulla postura eretta è utilizzata la misura del "*body sway*" (misura delle oscillazioni del corpo) per caratterizzare la performance[5]. Questo tipo di misura si basa sul calcolo del CoM (Center of Mass) di un corpo. Il CoM di un sistema è inteso come baricentro, esso indica il punto geometrico dove può essere idealmente collocata la massa dell'intero sistema.

Al fine di individuare le oscillazioni del corpo si ricorre al CoP (Center of Pressure). Esso indica la media delle pressioni che il corpo esercita su una base di appoggio, questa risultante è legata principalmente all'azione dei muscoli che, a livello di caviglia e anca, sono attuate per mantenere il CoM all'interno della base di appoggio. Se un solo piede è appoggiato sulla base il CoP risiederà sul punto di appoggio del piede. Quando entrambi i piedi sono appoggiati sulla superficie, il CoP sarà situato in un punto intermedio risultante dalla quantità di peso scaricato su ogni piede[1].

Lo strumento utile alla misura del CoP è la piattaforma di forza. Si tratta, in generale, di una pedana che appoggia al suolo tramite sensori in grado di rilevare le componenti delle forze che il piede può trasmettere. Una volta che il soggetto è salito sulla pedana possono essere effettuate svariate misure sia in ambito statico che dinamico. Se si utilizza una sola piattaforma di forza si ottiene una variazione del CoP globale legata all'azione di entrambi i piedi. Due piattaforme di forza utilizzate assieme permettono, invece, di

quantificare separatamente la variazione di CoP per ogni piede[1].

Negli ultimi anni si è assistito ad un progressivo aumento dell'interattività proposta nei videogame. Si sono infatti introdotte tecnologie come la comunicazione senza fili (bluetooth o tramite IR) o l'utilizzo di accelerometri e giroscopi. In particolare per la consolle Nintendo Wii sono state create delle periferiche che si prestano bene ad utilizzi in ambiti diversi da quelli previsti. E' il caso della Wii Balance Board (Nintendo Co. Ltd., Kyoto, Japan).

Si è osservato, nella letteratura recente, la comparsa di molti studi che hanno lo scopo di analizzare questo strumento non convenzionale per un utilizzo legato alla valutazione della capacità di equilibrio di un soggetto. Molti lavori si concentrano anche sulle capacità di potenziamento del sistema di controllo dell'equilibrio legato alla possibilità di eseguire determinati esercizi di mantenimento posturale o di coordinazione eseguiti sulla Wii Balance Board con la possibilità di feedback in tempo reale.

Il lavoro di tesi svolto è parte di un progetto che si propone lo scopo di validare la Wii Balance Board come strumento per la valutazione dell'equilibrio posturale. Lo studio prevede la realizzazione di una parte software che implementi la comunicazione tra Wii Balance Board e computer. Il programma sviluppato, destinato ad un utilizzatore non esperto, deve essere flessibile e di facile utilizzo.

Capitolo 1

STRUMENTAZIONE

1.1 Wii Balance Board

La Wii Balance Board (WBB) è un accessorio a forma di bilancia introdotto da Nintendo a fine 2007 (fig. 1.1). Si tratta di una periferica di forma rettangolare le cui dimensioni sono 51,1 x 31,6 X 3,2 cm. Il peso è di circa 3.5 kg.

La periferica è alimentata da quattro batterie AA, l'autonomia dichiarata è di circa 60 ore di utilizzo[6].

Sul retro della pedana, costruita principalmente in materie plastiche, è posto come limite massimo un peso dell'utilizzatore pari a 150 kg. Il limite strutturale dell'accessorio è comunque stato ipotizzato circa sui 300 kg[6].

Figura 1.1. *Wii Balance Board*

La WBB appoggia a terra tramite quattro punti di contatto disposti agli angoli della piattaforma, ciascuno dei quali collegato a un sensore di pressione. Le informazioni ricevute dai sensori permettono di valutare la pressione sui quattro angoli esercitata dall'utilizzatore. I dati vengono poi inviati tramite bluetooth.

I sensori di pressione utilizzati sono estensimetri (*strain gauge*). Si tratta di conduttori che modificano la loro resistenza elettrica in funzione della dilatazione alla quale sono sottoposti (fig. 1.2). Infatti le modifiche di lunghezza e diametro nel conduttore determinano una variazione nella resistenza che è tradotta in termini di tensione elettrica[7].

Quantificando la variazione di tensione di un sensore è possibile determinare quanto peso è stato caricato su di esso, ricostruendo la posizione dell'utilizzatore.

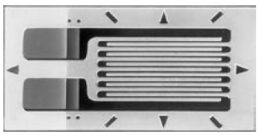


Figura 1.2. *Estensimetro*

Nel retro della pedana è presente il vano di alloggiamento delle batterie. Al suo interno si trova anche il pulsante di sincronizzazione (fig. 1.3) che serve a interfacciare correttamente la periferica con la consolle o il computer. L'operazione di sincronizzazione e le modalità di comunicazione saranno approfondite nei capitoli seguenti.

Figura 1.3. *Pulsante di sincronizzazione*

1.2 MatLab

MatLab (Matrix Laboratory) è un ambiente di sviluppo che utilizza l'omonimo linguaggio di programmazione ad alto livello. Nato come programma per il calcolo matriciale, MatLab, mette a disposizione ottimi strumenti per l'analisi di segnali e per la comunicazione con periferiche esterne tramite porta seriale e/o tramite bluetooth. Quest'ambiente si presta anche alla creazione d'interfacce utente (user interface – UI) grazie al tool GUIDE (Grafical User Interface Development Environment) che permettono una realizzazione grafica dell'interfaccia lasciando al programma il compito di generare codice. In questo modo all'utente è fornita la possibilità di creare UI per la gestione di dispositivi esterni con istruzioni semplici ed ad alto livello[8].

Un programma (script) scritto in MatLab è chiamato m-file. Un m-file può essere eseguito solamente in ambiente MatLab e questo rende il software e gli applicativi realizzati difficilmente portabili. A questo scopo mediante il toolbox MatLab Compiler (acquistabile separatamente) è possibile creare script eseguibili o librerie DLL (Dynamic Link Libraries) che non necessitano dell'acquisto di una licenza MatLab e quindi distribuibili a terzi[9].

1.2.1 Il tool GUIDE

La progettazione di un'interfaccia utente via codice è un'operazione che si può rivelare complessa. La possibilità di crearla in maniera grafica, utilizzando degli oggetti già definiti e con proprietà facilmente accessibili e ispezionabili, permette di creare programmi che interagiscono con l'utente in modo semplice e strutturato.

Per avviare il tool GUIDE si può digitare, nel prompt di MatLab, il comando "guide". Nella finestra di avvio del tool è richiesto di scegliere se iniziare un nuovo lavoro o riprendere una GUI in precedenza creata. Se si opta per iniziare un nuovo progetto si può scegliere di iniziare da un foglio bianco nel quale si andranno a definire tutti gli oggetti grafici oppure da strutture di default che permettono un rapido e predefinito interfacciamento (fig. 1.4). A titolo di esempio in figura 1.5 si può osservare l'anteprima del template definito "GUI with Uicontrols".

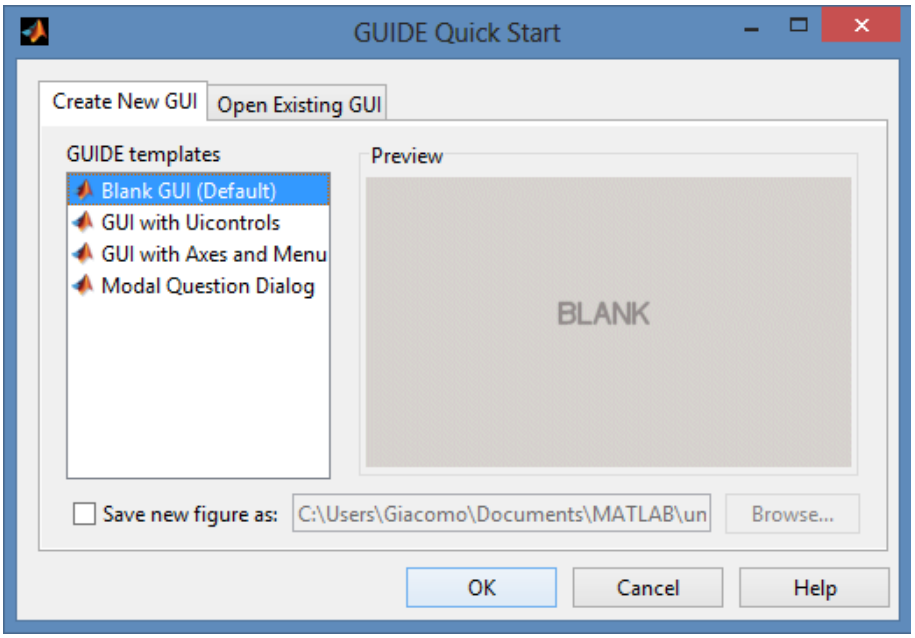


Figura 1.4. *Schermata di avvio di GUIDE*

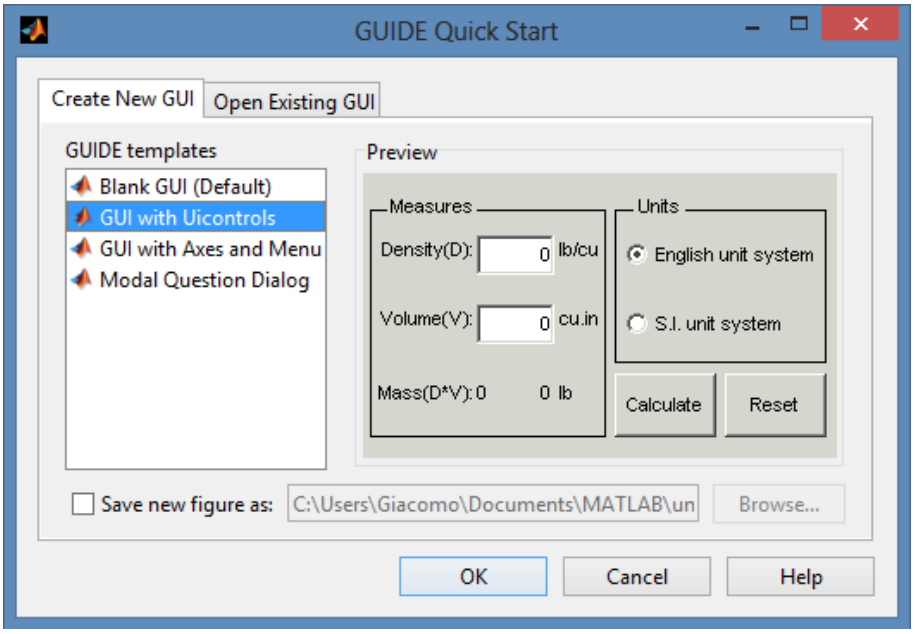


Figura 1.5. *Schermata di scelta templates*

Procediamo selezionando il template foglio bianco, indichiamo il nome e la *directory* di salvataggio e premiamo il tasto "OK". A questo punto il tool ha creato due file, un file grafico di estensione *.fig* che contiene le informazioni sul layout della nostra finestra grafica e un m-file in cui è stato già inserito del codice. Il codice generato rappresenta la struttura base della GUI. Sono presenti tre funzioni:

- $function\ varargout = gui1(varargin)$
- $function\ gui1\_OpeningFcn(hObject, eventdata, handles, varargin)$
- $function\ varargout = gui1\_OutputFcn(hObject, eventdata, handles)$

La funzione $varargout = gui1(varargin)$ è la funzione di inizializzazione, composta di una prima parte in cui si possono inserire i testi corrispondenti alla visualizzazione dell'help in linea di MatLab, e di una seconda parte non modificabile in cui avviene la vera e propria inizializzazione.

La funzione $gui1\_OpeningFcn$ è composta da tutte quelle operazioni che il programma deve eseguire prima di rendere visibile l'interfaccia utente. In questa funzione si possono inizializzare variabili o costanti e settare le proprietà dei vari oggetti.

La funzione *varargout* è composta dagli output che verranno restituiti sulla command line di MatLab.

Quando si inserisce un oggetto grafico nell'editor e si salva il lavoro, il tool modifica l'm-file inserendo nuove funzioni. Il codice inserito comprende una funzione che crea l'oggetto via codice definita $nome\_oggetto\_CreateFcn$ da non editare. L'altra tipologia di funzione è rappresentata dalla function $nome\_oggetto\_Callback$. Nel corpo delle funzioni $Callback$ sono definite le istruzioni da eseguire in corrispondenza al realizzarsi di un evento relativo all'oggetto $nome\_oggetto$.

Nella finestra di editor della figura abbiamo a disposizione numerosi oggetti grafici (osservabili sulla sinistra in fig. 1.6) che si possono aggiungere semplicemente con un operazione di trascinamento e posizionandoli nell'apposita griglia.

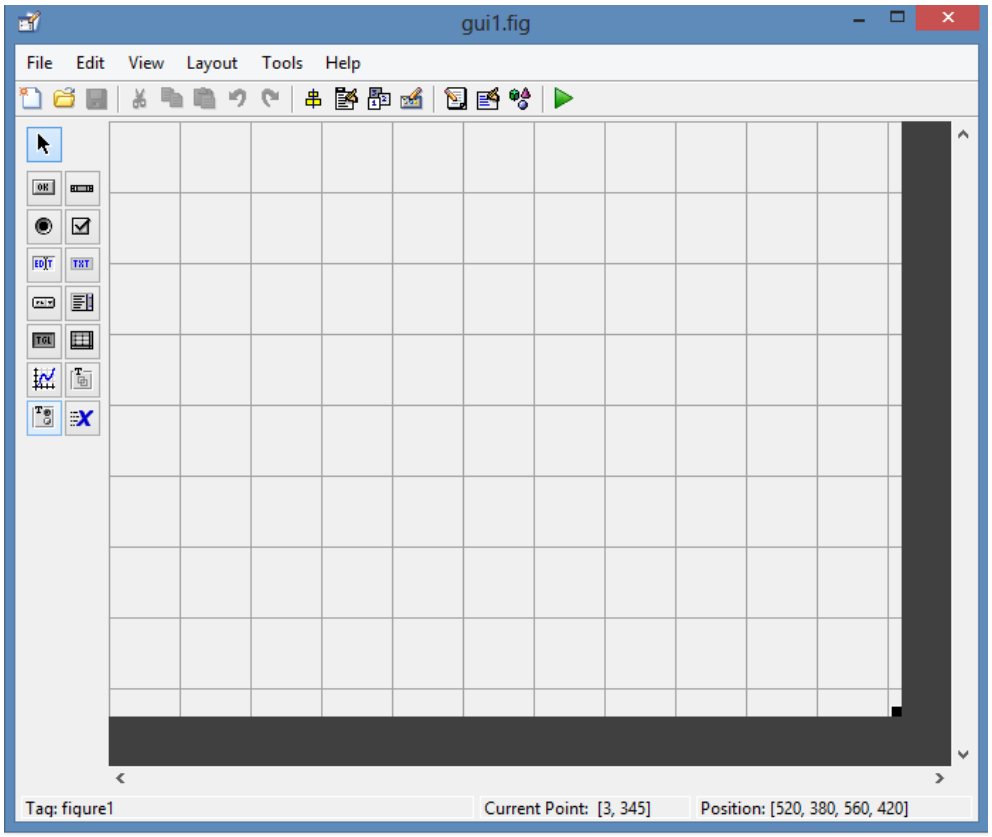


Figura 1.6. *Schermata di editor di una GUI*

I principali oggetti grafici utilizzati in questo applicativo sono:

- *Push button*
- *Axes*
- *Pop-up Menù*
- *Static text*

I *Push Button* rappresentano dei pulsanti. La relativa funzione di Callback contiene le istruzioni da eseguire quando sono premuti.

Gli oggetti *Axes* permettono di mostrare nella GUI un grafico o un immagine.

I *Pop-up Menù* sono i menù a tendina in cui è possibile selezionare una determinata opzione. In base al valore selezionato può variare, ad esempio, un'impostazione nel programma.

Infine gli oggetti di tipo *Static text* rappresentano un testo che può essere modificato solo da codice interno al programma e non rappresentano quindi un oggetto grafico di input per l'utente. Sono utilizzati principalmente come *label* per rendere l'interfaccia grafica più intuitiva.

Dalla finestra di editor è possibile, inoltre, visualizzare e modificare le proprietà di un oggetto. In questo modo si possono inizializzare i valori di un componente grafico senza dover modificare il codice, altrimenti, è possibile operare le inizializzazioni delle varie proprietà all'interno delle varie *function*. Le proprietà che sono generalmente modificate dall'editor riguardano le stringhe di testo visualizzate ad esempio da un *Push Button* oppure da uno *Static tex*t, si possono anche modificare carattere e colore del testo oppure impostare i valori di massimo e minimo delle ascisse e ordinate di un grafico. È inoltre possibile, tramite la proprietà *Enable*, attivare o rendere inattivo un componente. Tramite l'opzione *Visible*, invece, si determina se l'oggetto potrà essere visto nell'interfaccia o scomparire per lasciar posto ad altri componenti. La schermata di proprietà di un oggetto *Push Button* si può osservare in figura 1.7

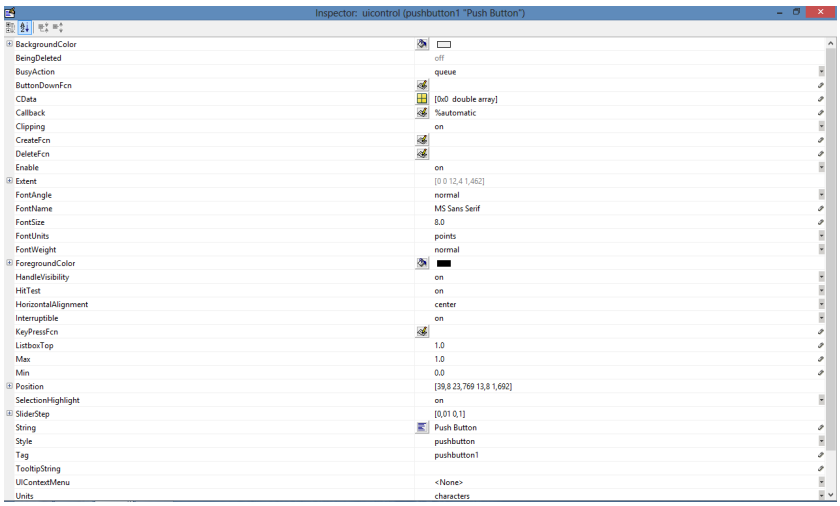


Figura 1.7. *Schermata delle proprietà di un oggetto grafico*

1.2.2 Il tool DEPLOYTOOL

Il toolbox MatLab Compiler può essere utilizzato in due modi: Il primo prevede l'inserimento di istruzioni direttamente nel *prompt* dei comandi di MatLab, il secondo prevede l'utilizzo di un tool dedicato chiamato deploytool. Questo strumento permette un approccio più semplice utilizzando un'interfaccia grafica che guida l'utente durante tutta la procedura di creazione del pacchetto indipendente.

Per avviare il tool si può agire direttamente dal *prompt* di MatLab digitando "deploytool". Dopo la pressione del tasto "INVIO" appare una finestra di dialogo che permette di scegliere se creare un nuovo progetto o selezionare un progetto già esistente. In caso di un nuovo lavoro si selezionerà anche la tipologia di progetto. La nostra scelta ricade sulla creazione di un applicativo stand alone per sistemi operativi Windows (Windows Standalone Application). Prima di procedere si dovrà selezionare la cartella di destinazione del progetto e il nome (fig. 1.8).

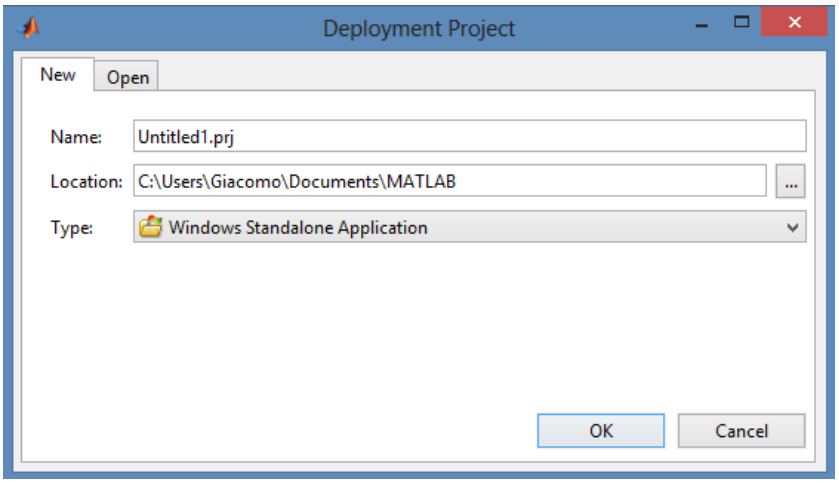


Figura 1.8. *Schermata di avvio di deploytool*

Il passo successivo consiste nell'individuare i file che compongono il nostro applicativo. In particolare bisogna selezionare un *main file*, i vari m-file *function* che contengono le funzioni richiamate nel *main*, gli eventuali file *.fig*

con le interfacce grafiche create ed infine, se presenti, i file esterni scritti in codice *C* o *C++* che interagiscono con la nostra applicazione.

Il *main file* è il file principale di avvio e ha il compito di richiamare le varie funzioni necessarie allo svolgimento del programma. La specifica richiesta da deploytool è che il *main file* contenga solamente funzioni. La prima funzione definita nel suo corpo sarà chiamata funzione principale e sarà l'unica accessibile dall'esterno.

A questo punto bisogna inserire nel progetto i vari file. Per farlo basta trascinarli all'interno degli appositi campi nella finestra del tool o ricercarli nelle cartelle del computer cliccando sui pulsanti *"Add main file"* *"Add file/folders"* (fig. 1.9).

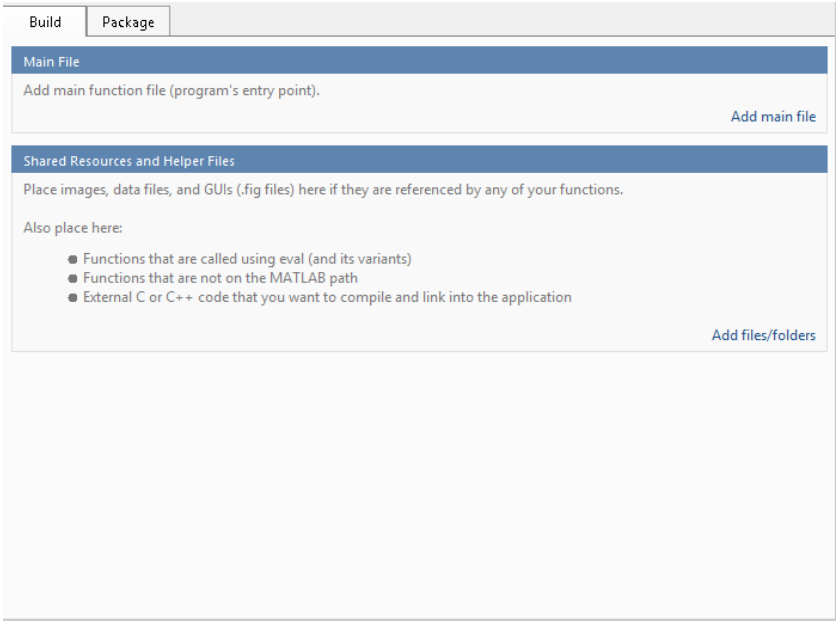


Figura 1.9. *Schermata di inserimento file*

Una volta inseriti i file necessari si procede selezionando il pulsante "Package" nel quale si definiranno i file che il pacchetto finale dovrà contenere. Si tratta di tutti i file che devono arrivare all'utente finale. Solitamente è sempre presente, oltre all'eseguibile, almeno un file *leggimi*.

Come si osserva in figura 1.10 sono già presenti i file di default: l'eseguibile con estensione *prova1.exe* e il file testo *readme.txt*. Al solito si potranno

aggiungere file semplicemente con un operazione di trascinamento oppure con una ricerca della cartella.

Come si può osservare sempre in figura 1.10 è presente il tasto "Add MCR".

Un applicativo stand alone MatLab, pur essendo indipendente, necessita sempre per poter funzionare della presenza della MCR (MatLab Compiler RunTime). Si tratta di un insieme file che forniscono le librerie necessarie al funzionamento dell'applicazione.

Questo pacchetto può essere escluso dal progetto solo nel caso in cui l'applicazione finale è destinata ad un computer nel quale è già presente l'ambiente MatLab o sono già state installate le librerie della MCR. Al fine di migliorare la portabilità è consigliabile comprenderla nel pacchetto cliccando sul pulsante "Add MCR".

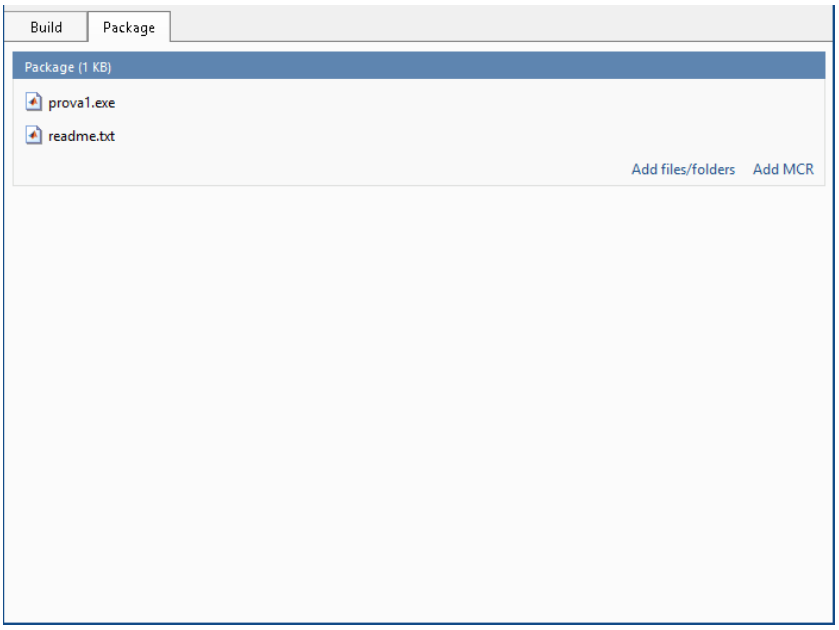
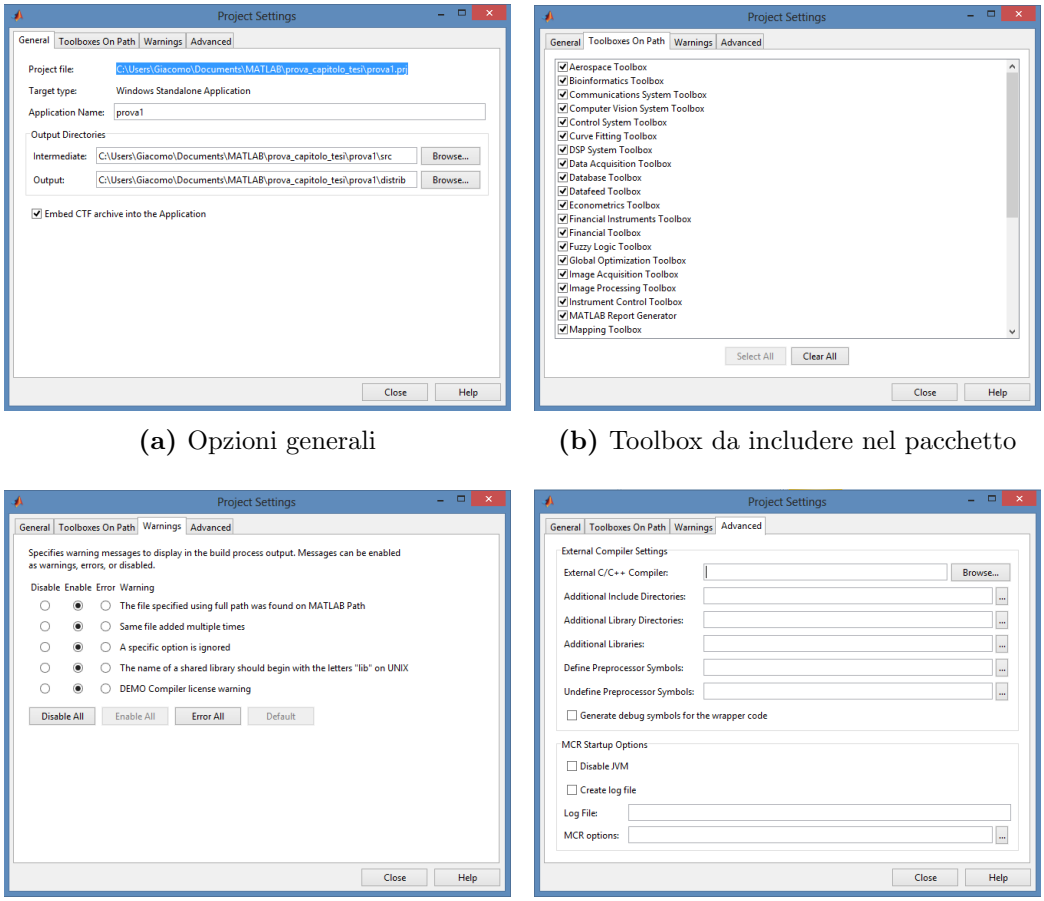


Figura 1.10. *Schermata di selezione file da distribuire*

Le ultime impostazioni per il progetto posso essere selezionate tramite il menù *settings* (fig. 1.11) individuabile in alto a destra nella finestra principale del tool.



(a) Opzioni generali **(b)** Toolbox da includere nel pacchetto

(c) Opzioni warning message **(d)** Opzioni avanzate

Figura 1.11. *Schermate di settings*

Sono presenti quattro sottomenù, un sottomenù generale nel quale si impostano nome e tipologia del progetto. In questa schermata, in basso, si osserva anche la presenza di un opzione che riguarda la possibilità di includere l'archivio CTF (Component Technology File) nell'applicazione. Si tratta di un archivio contenente tutti i file dell'applicazione in forma criptata in modo che non siano accessibili a terzi. Sono inoltre presenti tutte le informazioni di localizzazione dei vari file.

La seconda pagina di impostazioni riguarda la possibilità di scegliere quali tool disponibili in MatLab includere nell'applicazione. A questo proposito si consiglia di inserire solo i tool strettamente necessari al funzionamento

dell'eseguibile, risultano infatti essere molto pesanti e rischiano di rallentare in maniera considerevole l'applicazione, in particolare all'avvio.

La terza pagina riguarda che tipologia di messaggi di *warning* abilitare durante il processo di *build* dell'applicazione.

L'ultima pagina comprende le impostazioni avanzate. Si andranno a inserire le impostazioni di un eventuale compilatore esterno utilizzato al posto di quello fornito da MatLab. Inoltre è possibile, nella parte riservata alle opzioni della MCR, scegliere se disabilitare la JVM (Java Virtual Machine) e/o scegliere di creare un file LOG nel quale verranno riportati eventuali errori. La JVM causa un'esecuzione più lenta dell'applicazione ma risulta necessaria qualora nel programma si utilizzi una componente grafica.

Dopo aver terminato la fase delle impostazioni, si procede alla compilazione dei file e alla creazione dell'eseguibile. Si può quindi selezionare il pulsante "build" e attendere qualche secondo.

Quando l'elaborazione è terminata si può notare la creazione di una nuova cartella. In essa sono presenti due sottocartelle chiamate *src* e *distrib*. Le cartelle contengono rispettivamente i prodotti intermedi e finali dell'operazione di *build*. Il programma stand-alone è stato creato e in questa fase si può testare per verificare la presenza di errori logici o di errori nella ricerca delle dipendenze. Verificato il programma si può procedere alla creazione del pacchetto finale in cui è inclusa anche la MCR. Dalla schermata di avvio di *delpoytool* si seleziona il pulsante "Package". Al termine del processo si ottiene un file autoestraente. Al momento dell'avvio il file estrae nella cartella in cui si trova tutti i file da noi inseriti, tra cui la MCR, e invoca tramite un file batch l'MCR installer che guiderà l'utente nel procedimento di installazione. Al termine del processo l'applicativo potrà essere utilizzato correttamente.

Capitolo 2

COMUNICAZIONE

In questo capitolo si affronta il tema della comunicazione tra il computer e la periferica WBB.

Nella sezione 2.1 verrà esposto il protocollo di comunicazione e come i dati vengono inviati, nella sezione 2.2 si analizzano in particolare il tipo di dati inviati dalla WBB e, infine, nella sezione 2.3 si sviluppa l'aspetto di ricezione delle informazioni dal lato computer.

La comunicazione tra periferica e computer risulta agevole grazie a due particolari fattori. Il primo riguarda il protocollo di comunicazione bluetooth HID. Il secondo fattore è legato alla numerosa presenza di librerie create appositamente per l'analisi dei dati ricevuti dai joystick della Nintendo Wii. In particolare sono state utilizzate la libreria *WiimoteLib* e la libreria *WiiLab*.

2.1 Il protocollo HID

Al fine di determinare una modalità di comunicazione caratteristica per la vasta gamma di dispositivi che utilizzano la comunicazione bluetooth, sono stati creati diversi profili per raggruppare classi di periferiche che individuano delle specifiche in grado di ottimizzare il protocollo di comunicazione.

Il profilo HID (*Human Interface Device*), in particolare, è un protocollo creato in modo specifico per implementare la comunicazione dei dispositivi USB che operano a stretto contatto con le azioni umane.

Tipici esempi di dispositivi che appartengono a questa classe sono:

- Mouse, tastiere e touchpad
- Joystick e controller per videogiochi
- Apparecchiature multimediali

La principale caratteristica dei dispositivi appartenenti alla classe HID è di essere autodescrittivi. È compito della periferica, infatti, fornire la sua descrizione e definire il tipo di dati e le modalità di trasmissione con cui sono comunicati.

Le informazioni riguardo la tipologia di dispositivo sono salvate all'interno di segmenti (chiamati *descriptor*) (fig 2.1) situati nella *ROM* (*Read Only Memory*) del dispositivo stesso[10]. Le informazioni fornite in questi segmenti indicano anche la presenza di altri descrittori più specifici: i *report descriptor*.

Questi descrivono ogni dato trasferito indicandone la tipologia e la dimensione. Un'altra classe (facoltativa) di *descriptor* sono i *physical descriptor*, essi forniscono informazioni circa la parte del corpo a contatto con il dispositivo che ha generato il dato trasmesso[10].

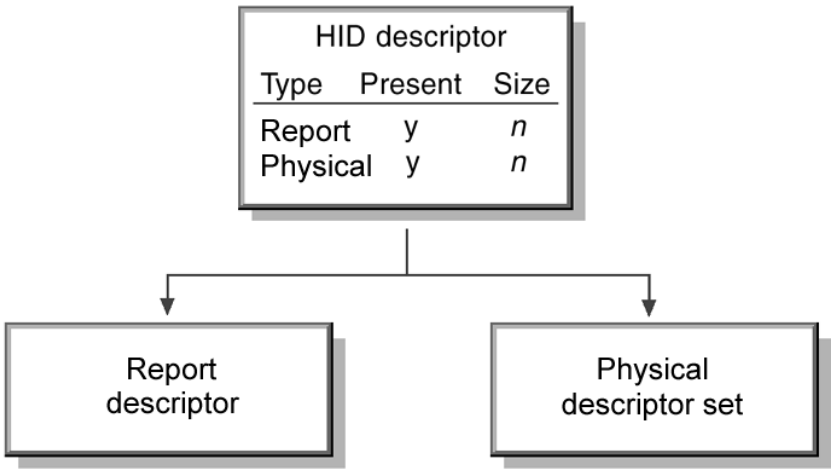


Figura 2.1. *Rappresentazione ad albero dei descriptor[10]*

La trasmissione di informazioni tra dispositivo e computer è bidirezionale. Il dispositivo potrà ricevere in input dati per le impostazioni di configu-

razione, per esempio l'illuminazione di un led su una tastiera, oppure potrà ricevere comandi di feedback per rendere un'esperienza di gioco più realistica (vibrazione di un joystick).

La comunicazione tra *host* (in questo caso il computer) e *device* può avvenire in due modi (fig. 2.2):

- Tramite *Control pipe* (*polling*)
- Tramite *Interrupt*

Se la trasmissione è impostata in modalità *polling* il computer interroga periodicamente il dispositivo per ottenere le informazioni[11]. La modalità *interrupt* prevede, invece, che sia il dispositivo a mandare una richiesta di trasmissione quando il suo stato cambia, ad esempio, in seguito alla pressione di un tasto sulla tastiera o a causa di un click del mouse[12].

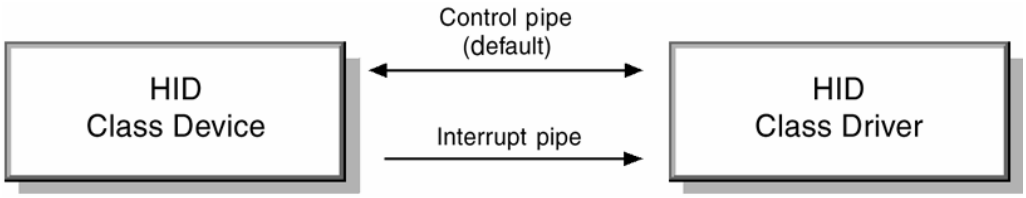


Figura 2.2. *Schema di comunicazione tra host e device*[10]

2.2 I report della Wii Balance Board

La periferica di controllo principale della Nintendo Wii è il *Wii Remote* (comunemente noto come *Wiimote*).

Si tratta di un *joystick* rettangolare dotato di un accelerometro a tre assi e di una microcamera a infrarossi. La particolarità della periferica consiste nella possibilità di aggiungere delle estensioni al *controller* principale in modo da fornire più possibilità di interazione con il giocatore. In particolare, in figura 2.3, si osserva il *Wiimote* (sulla destra) a cui è collegato come estensione un secondo controller chiamato *Nunchuck*.

Figura 2.3. *Wiimote con l'estensione Nunchuck*

Il *Wiimote* non implementa in maniera standard in protocollo HID. I *report* sulle informazioni trasmesse, infatti, riguardano solamente la lunghezza del pacchetto dati e non portano riferimenti al significato dello stesso[13].

Questa particolarità rende necessaria la creazione di driver appositi al fine di poter analizzare i dati ricevuti.

Lo studio del controller ha permesso comunque di identificare la maggior parte delle informazioni trasmesse (fig. 2.4).

I/O	ID(s)	Size	Function
O	0x10	1	Unknown
O	0x11	1	Player LEDs
O	0x12	2	Data Reporting mode
O	0x13	1	IR Camera Enable
O	0x14	1	Speaker Enable
O	0x15	1	Status Information Request
O	0x16	21	Write Memory and Registers
O	0x17	6	Read Memory and Registers
O	0x18	21	Speaker Data
O	0x19	1	Speaker Mute
O	0x1a	1	IR Camera Enable 2
I	0x20	6	Status Information
I	0x21	21	Read Memory and Registers Data
I	0x22	4	Acknowledge output report, return function result
I	0x30-0x3f	2-21	Data reports

Figura 2.4. *Tabella report del Wiimote*[13]

Ci sono due modalità di trasmissione utilizzabili, esse sono selezionate trasmettendo un comando di due byte al *report* 0x12 (voce *Data Reporting mode* in figura 2.4).

Le opzioni riguardando la possibilità di utilizzare un flusso periodico di dati tra *host* e *device* (*polling*) o di decidere per una trasmissione *on event* in cui si trasferiscono informazioni solo se i valori sono variati.

La tipologia di informazioni inviate può variare per meglio adattarsi alla tipologia di applicazione a cui è destinato il *Wiimote*. L'opzione di scelta è sempre legata alle richieste inviate dal computer corrispondenti al *report Data Reporting*.

Le modalità tra cui scegliere sono:

- Core Button – indica lo stato dei pulsanti del Wiimote
- Core Button and Accelerometer – indica lo stato dei pulsanti e il valore dell'accelerometro
- Core Button with 8 exstension byte – indica lo stato dei pulsanti e 8 byte di dati riguardo all'estensione applicata
- Core Buttons and Accelerometer with 12 IR bytes - indica lo stato dei pulsanti, il valore dell'accelerometro e 12 byte di informazioni riguardo alla microcamera IR
- Core Buttons with 19 Extension bytes – indica lo stato dei pulsanti e 12 byte di dati riguardo all'estensione applicata
- Core Buttons and Accelerometer with 16 Extension Bytes – indica lo stato dei pulsanti, il valore dell'accelerometro e 16 byte d informazioni riguardo all'estensione applicata
- Core Buttons with 10 IR bytes and 9 Extension Bytes – indica lo stato dei pulsanti, 10 byte di informazioni riguardo alla microcamera IR e 9 byte riguardo all'estensione applicata
- Core Buttons and Accelerometer with 10 IR bytes and 6 Extension Bytes – indica lo stato dei pulsanti, il valore dell'accelerometro, 10

byte di informazioni riguardo alla microcamera IR e 6 byte riguardo all'estensione applicata

- Extension Bytes – indica le informazioni riguardo alla sola estensione applicata
- Interleaved Core Buttons and Accelerometer with 36 IR bytes – indica lo stato dei pulsanti, il valore dell'accelerometro e 36 byte di informazioni riguardo alla microcamera IR

Le modalità sopra elencate ci permettono di scegliere quindi se e quali tipologie di informazioni utilizzare nell'applicazione.

Come vedremo in seguito solo una parte di queste procedure sono implementate nella libreria *WiimoteLib* utilizzata nell'applicativo realizzato.

Per quanto riguarda la WBB il nome identificativo della periferica è "Nintendo RVL-WBC-01". La piattaforma genera un *report* di dati equivalente ad un *Wiimote* con una estensione applicata, il tipo di *report* è impostato internamente e non può essere modificato[14].

	Bit							
Byte	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	**Top Right**<15:8>							
1	**Top Right**<7:0>							
2	**Bottom Right**<15:8>							
3	**Bottom Right**<7:0>							
4	**Top Left**<15:8>							
5	**Top Left**<7:0>							
6	**Bottom Left**<15:8>							
7	**Bottom Left**<7:0>							

Figura 2.5. *Tabella report dei sensori della WBB*[15]

Il pulsante di accensione, il valore dell'unico led presente sulla periferica e lo stato della batteria della periferica sono le sole informazioni conosciute che

vengono ricavate dalla parte dei dati relativi al *Wiimote*. I *report* specifici della *balance board* vengono forniti dai byte dedicati alle informazioni delle estensioni. Le informazioni ricevute dai sensori si basano su 8 byte. Ogni sensore è codificato con 2 byte. Si tratta quindi di 16 bit di precisione.

2.3 Le librerie WiimoteLib e WiiLab

Come accennato nella sezione precedente, il *Wiimote*, pur utilizzando il protocollo HID necessita di appositi *driver* per poter comunicare con il computer. Per la realizzazione dell'applicativo oggetto di questa tesi è stata utilizzata la libreria *WiimoteLib*.

Si tratta di una libreria sviluppata da Brian Peek (*Senior Technical Evangelist at Microsoft*) scritta in linguaggio *.NET*[16]. La versione utilizzata è la 1.6.0.0. Si tratta della penultima versione disponibile (l'ultima versione non apporta migliorie per quanto riguarda la gestione della WBB).

Al momento non tutte le modalità di lettura dati possibili sono state implementate. Sono disponibili le seguenti modalità:

- Buttons – fornisce solo lo stato dei pulsanti
- ButtonsAccel – fornisce lo stato dei pulsanti e dell'accelerometro
- IRAccel – fornisce lo stato dei pulsanti, dell'accelerometro e della microcamera IR
- ButtonsExtension – fornisce lo stato dei pulsanti e dell'estensione
- ExtensionAccel – fornisce lo stato dei pulsanti, dell'accelerometro e dell'estensione
- IRExtensionAccel - fornisce lo stato dei pulsanti, dell'accelerometro, dell'estensione e della microcamera IR

La libreria è costruita attorno alla classe principale *Wiimote class*. Al suo interno troviamo i metodi principali che ci permettono l'iterazione con il dispositivo:

- Connect – connette il primo Wiimote disponibile
- Disconnect – disconnette il Wiimote e blocca la lettura dei dati
- GetStatus – legge lo stato del Wiimote e dell'estensione applicata
- GetData – legge i dati dal Wiimote
- SetReportType – permette di impostare la tipologia di report desiderati (*polling* o *on event*)

Analizzando la struttura della libreria *WiimoteLib* si osserva come le varie tipologie di *report* siano state suddivise in base alle informazioni che contengono. In questo modo è possibile accedere intuitivamente alle informazioni necessarie. Ad esempio tramite la struttura *"AccelState"* possiamo ottenere i dati misurati dall'accelerometro; in egual modo tramite la struttura *"BalanceBoardState"* riceviamo le informazioni sullo stato della WBB.

La libreria *WiiLab* è stata sviluppata presso l'Università di Notre Dame, in particolare da Jordan Brindza e Jessica Szweda. La principale funzione di questo software è permettere l'interazione tra l'ambiente MatLab e i controller della Nintendo Wii. Si tratta di una libreria *wrapper*. Il concetto di un software *wrapper* è quello di fare da "involucro" a un altro software (che di solito permette interazioni di basso livello) per fornire all'utenza un insieme di funzioni o interfacce che forniscono un interfacciamento di più alto livello[17] (fig. 2.6).

La versione di *WiiLab* che è stata utilizzata è stata modificata da Ian Stevenson che ha introdotto una *patch* per implementare il supporto alla WBB[18].

Per comunicare con la periferica l'utente si avvale della classe MatLab *Wiimote.m*. La classe utilizza delle *properties* per strutturare i dati ricevuti. Si tratta di strutture utilizzate come contenitori per le istanze della classe, esse possono comprendere sia variabili che costanti. Definendo queste strutture con gli attributi di *pubblic*, *protected* o *private* si può limitare la possibilità di accesso e di eventuale modifica all'utilizzatore della classe.

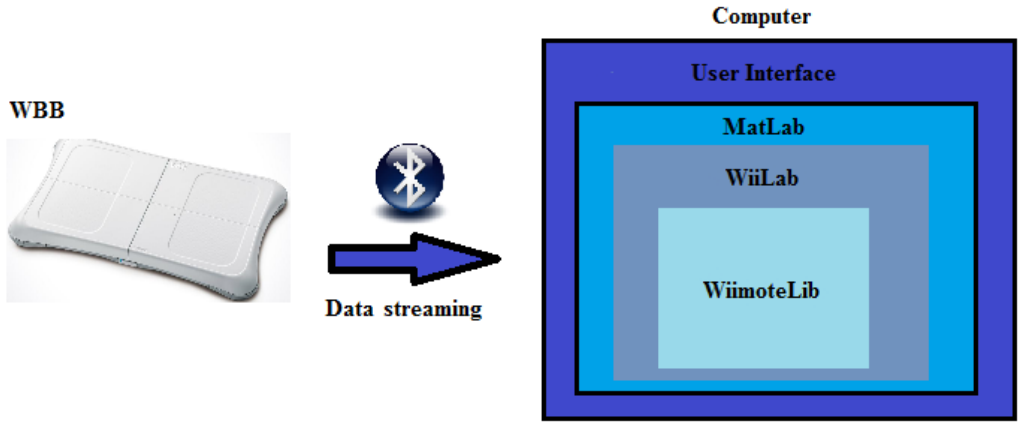


Figura 2.6. *Schema di comunicazione WBB-computer*

Nel caso specifico della *Wiimote.m* le *properties* sono quasi tutte con attributo *private* e risultano quindi accessibili ma non modificabili dall'utente[19].

Le strutture di maggior interesse nel programma realizzato sono:

- BalanceBoardSensors
- BalanceBoardCoG
- Battery

Un set di ulteriori strutture è destinato ad un utilizzo interno ed è definito privato e nascosto, risulta quindi non accessibile e non modificabile. Tale insieme è composto dai valori direttamente ottenuti dalla libreria *WiimoteLib*.

Le principali funzioni della libreria che sono state utilizzate nel software realizzato sono:

- *Connect()*
- *DisconnectAllWiimote()*
- *isConnected()*
- *GetBalanceBoardCoGState()*

- *GetBalanceBoardSensorState()*
- *GetBatteryState()*

Le funzioni *Connect()*, *DisconnectAllWiimote()* e *isConnected()* si occupano intuitivamente della connessione/disconnessione del controller.

La funzione GetBalanceBoardCoGState() è molto importante, fornisce direttamente un *array* con due valori che rappresentano le coordinate spaziali del CoG (*Center Of Gravity*).

Particolare attenzione si pone sulla convenzione utilizzata dalla funzione per l'orientamento degli assi. Come si osserva in figura 2.7, l'asse Y risulta orientato verso il retro della pedana.

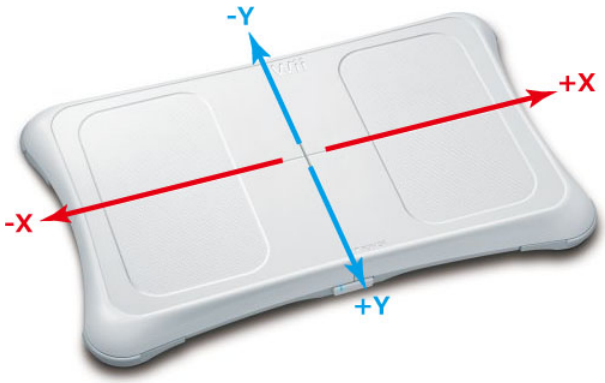


Figura 2.7. *Orientamento degli assi della funzione GetBalanceBoardCoGState()*

La funzione *GetBalanceBoardSensorState()* permette di individuare la forza peso agente sul singolo sensore. Il peso totale si ottiene sommando i valori dei quattro sensori. Infine *GetBatteryState()* fornisce indicazioni circa il livello di carica disponibile. Può risultare un dato importante, infatti si è osservato un peggioramento della qualità dei dati ricevuti quando il livello di batteria è minore del 40%.

Capitolo 3

ANALISI DEL PROGRAMMA

In questo capitolo si analizza nel dettaglio il software realizzato. Nella sezione 3.1 si definiscono l'interfaccia utente e le funzioni principali del programma mentre nella sezione 3.2 si descrivono le modalità di utilizzo disponibili.

In particolare, nella sezione 3.2.1, si tratta della procedura di installazione dell'applicativo a partire dal pacchetto *standalone* (si veda 1.2.2), nella sezione 3.2.2 si analizzano i passaggi da effettuare per una corretta connessione della periferica WBB al computer e infine, nella sezione 3.2.3, si tratta brevemente delle impostazioni da modificare per ottenere una corretta visualizzazione dei dati salvati dal programma.

3.1 Struttura dell'interfaccia grafica e funzioni principali del programma

L'interfaccia grafica (fig. 3.1) è stata realizzata per rendere semplice e intuitivo l'utilizzo del programma.

Si presenta principalmente con un oggetto grafico *Axes* centrale in cui verrà rappresentato il CoP del soggetto. Alla destra sono presenti i *Push Button* con il compito di controllare la parte di programma relativa alla registrazione e al salvataggio dei dati. In alto a sinistra si trovano gli oggetti *Static text* che indicano i valori in tempo reale (in centimetri) delle coordinate spaziali del CoP e della forza peso agente sulla pedana. In alto al centro è

presente il *Push Button CONNECT*, è l'unico pulsante attivo all'avvio del programma (ovvero con la proprietà *Enable* impostata su *ON*) e ha il compito attivare la procedura di connessione. Accanto, sulla destra, sono presenti altri due pulsanti che definiscono le impostazioni di calibrazione e di creazione di profilo utente e un *Pop-up Menù* nel quale si potrà selezionare la modalità di lavoro desiderata.

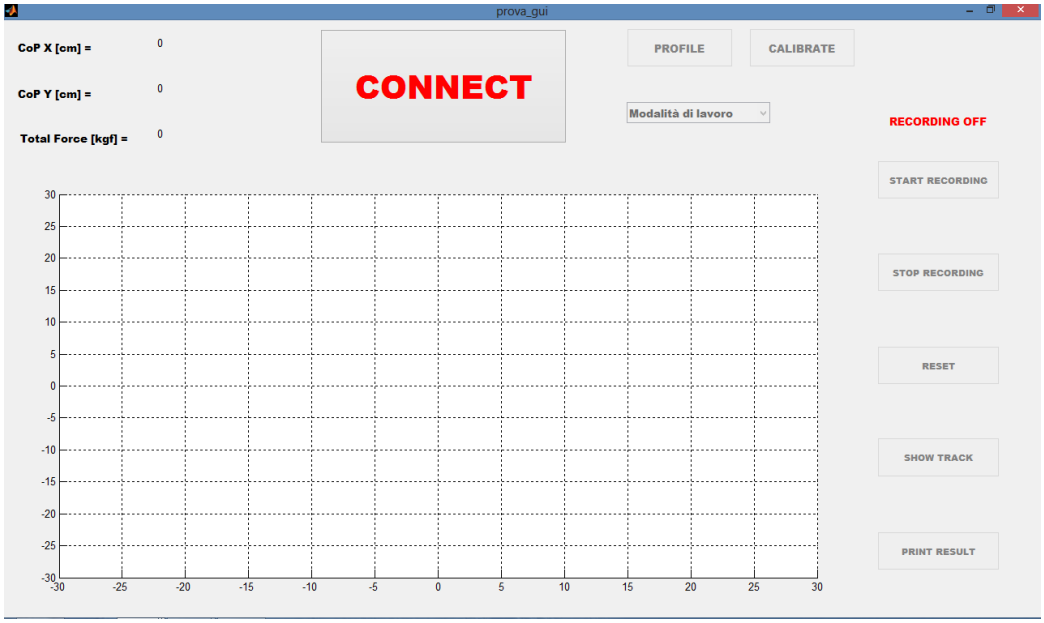


Figura 3.1. *Interfaccia utente all'avvio dell'applicazione*

La pressione del pulsante *CONNECT* consente l'esecuzione della funzione di connessione. Successivamente una finestra di dialogo permette di scegliere tra l'utilizzo di una singola WBB o l'utilizzo di due pedane contemporaneamente.

Tramite il metodo *connect()* (sezione 2.3) si tenta la connessione alla periferica. Un ciclo di controllo ne verifica l'effettivo esito.

A questo punto sono resi visibili (tramite la proprietà grafica *Visible*) i pulsanti *START* e *STOP* che avviano o terminano l'esecuzione della funzione principale.

La funzione di *Callback* associata al pulsante *START* richiama la funzione *calibrate* che si occupa di determinare i parametri di calibrazione necessari

per ottenere dei dati coerenti dalla periferica.

Il corpo principale del programma è rappresentato dalla funzione *main*. Si tratta di un processo sempre attivo sul quale sono poi richiamate le varie funzioni di *Callback* e al quale si ritorna una volta terminate le operazioni descritte in quelle funzioni. Il *main loop* ha condizione sempre vera e termina soltanto quando il pulsante *QUIT* è premuto.

La funzione *main* richiama al suo interno due funzioni in particolare: la funzione *get_data* e la funzione *plotCop*.

La funzione *get_data* (codice 3.1) fornisce in *output* le coordinate spaziali del CoP e il valore rilevato dai quattro sensori di forza. I parametri C e *sensor_avg_upright* sono i valori di regolazione definiti durante l'esecuzione della funzione di calibrazione.

Listing 3.1. *Estratto di codice: la funzione get_data*

```
function [cog,sensors] = get_data(board, C, sensor_avg_upright,
    handles)
     cog = board.wm.GetBalanceBoardCoGState();
     sensors = (board.wm.GetBalanceBoardSensorState()-
         sensor_avg_upright)./C;
```

La funzione *plotCop* (codice 3.2), invece, si occupa della parte grafica. Il CoP è rappresentato come un "pallino" all'interno di un sistema di assi cartesiano. Le dimensioni del punto variano in funzione del peso totale rilevato dalla pedana.

Per enfatizzare il passaggio alla modalità di registrazione il punto cambierà colore da nero a rosso.

Listing 3.2. *Estratto di codice: la funzione plotCop*

```
function plotCop(weight, CopX, CopY, color, handles)
    cla
    hold on
    size = weight/2;
    if size < 1
        size = 1;
```

```
end
x = CopX;
y = CopY;
c = color;
axis([-30 30 -30 30]);
xlabel('X [cm]')
ylabel('Y [cm]')
grid on
plot(x, y,'ko', 'MarkerFaceColor',c, 'MarkerSize',size);
hold off
```

La funzione *twoBoardCoP*, infine, è stata creata per la gestione specifica di due WBB. Permette, infatti, il calcolo del CoP complessivo a partire dai dati distinti delle due pedane.

Per la valutazione dell'equilibrio posturale sono stati utilizzati i seguenti parametri:

- *X range*, individua la massima escursione lungo l'asse orizzontale
- *X mean*, indica il valore medio della coordinata X del CoP
- *Y range*, individua la massima escursione lungo l'asse verticale
- *Y mean*, indica il valore medio della coordinata Y del CoP
- *Gomitolo*, rappresenta l'insieme di punti individuati dal CoP
- *Area*, calcola l'area del gomitolo utilizzando il metodo dell'inviluppo convesso[20]
- *Sway Area*, indica la velocità con cui il CoP descrive l'area
- *Lenght*, individua la lunghezza del gomitolo
- *Sway Path*, indica la velocità con cui il gomitolo è percorso dal CoP

I vari parametri potranno essere disponibili all'utente attraverso le operazioni di salvataggio descritte nella sezione seguente (3.1.1).

3.1.1 Registrazione e salvataggio

Al fine di poter registrare i dati ricevuti e processati, l'utilizzatore deve procedere alla creazione di un profilo (fig. 3.2).

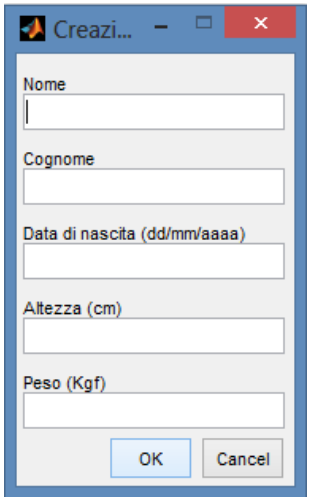


Figura 3.2. *Finestra di dialogo per le impostazioni del profilo*

L'operazione di creazione del profilo può essere effettuata subito dopo la pressione del tasto *CONNECT*. Il profilo può essere inoltre creato (o modificato) durante l'esecuzione della *main function* attraverso il pulsante *PROFILE*.

Dopo aver inserito i dati dell'utilizzatore si può scegliere la cartella di destinazione del profilo. Al termine della procedura il programma produce, nella *directory* selezionata, una cartella nominata *cognome_nome* utilizzando i dati inseriti nella finestra di dialogo precedente. All'interno di questa nuova cartella verranno salvati i dati processati dal programma.

Il programma può creare due tipologie di file:

- I file di tipo *trial*
- I file di tipo *esito*

I file di tipo *trial* sono salvati come file *.xls* con nominazione *cognome_nome_data_trial_numeroTrial*. Il "*numeroTrial*" è una variabile che individua quanti *trial* sono avvenuti per lo stesso profilo in una data sessione

di utilizzo del programma. In questo tipo di file sono rappresentati, per ogni ciclo di acquisizione dati, i valori delle coordinate spaziali del CoP e i valori dei quattro sensori (fig. 3.3).

I file di tipo *esito* (fig. 3.4) sono anch'essi salvati come file *.xls* e nominati *cognome_nome_data_esito_numeroTrial*. Questo tipo di file non viene salvato automaticamente dalla *routine* del programma. Per ottenerlo è necessario premere sul pulsante *PRINT RESULT*. Nei file di tipo *esito* sono contenuti i parametri di valutazione dell'equilibrio presentati nella precedente sezione 3.1.

	A	B	C	D	E	F	G	H
1	Cycle	Time [s]	CoP X [cm]	CoP Y [cm]	Back Left Force [kgf]	Back Right Force [kgf]	Front Left Force [kgf]	Front Right Force [kgf]
2	1.00	0.01	-3.28	-1.66	46.22	29.18	30.37	26.18
3	2.00	0.11	-3.28	-1.67	45.98	28.98	30.10	26.03
4	3.00	0.15	-3.24	-1.66	45.94	29.02	30.06	26.15
5	4.00	0.21	-3.33	-1.66	46.10	28.98	30.33	25.99
6	5.00	0.24	-3.26	-1.69	46.14	29.14	30.10	26.07
7	6.00	0.27	-3.32	-1.71	46.30	29.22	30.25	25.91
8	7.00	0.30	-3.27	-1.68	46.02	29.14	30.22	26.03
9	8.00	0.33	-3.29	-1.74	46.38	29.26	30.02	25.91
10	9.00	0.37	-3.25	-1.75	46.26	29.26	29.86	25.91
11	10.00	0.40	-3.31	-1.71	46.22	29.18	30.22	25.91
12	11.00	0.43	-3.38	-1.72	46.58	29.06	30.25	25.91
13	12.00	0.46	-3.32	-1.72	46.58	29.22	30.22	26.07
14	13.00	0.49	-3.33	-1.75	46.94	29.50	30.41	26.11
15	14.00	0.52	-3.45	-1.78	47.30	29.18	30.29	25.95
16	15.00	0.60	-3.46	-1.92	48.82	30.31	30.61	26.11
17	16.00	0.62	-3.44	-1.97	49.22	30.80	30.76	26.11
18	17.00	0.65	-3.42	-1.93	49.42	30.67	30.80	26.50
19	18.00	0.68	-3.49	-1.90	49.38	30.59	31.16	26.42

b_a_09-Jul-2013_trial_1

Pronto 140%

Figura 3.3. *Esempio di file di tipo trial salvato dal programma*

	A	B	C	D	E	F	G	H	I	J	K	L
1	Nome:	Giacomo										
2	Cognome:	Boioli										
3	Data di nascita:	31/07/1989										
4	Altezza [cm]:	192.00										
5	Peso [Kg]:	90.00										
6	Xrange [cm]:	0.59										
7	Xmean [cm]:	1.33										
8	Yrange [cm]:	1.73										
9	Ymean [cm]:	-2.48										
10	Area [cm^2]:	0.54										
11	SwayArea [cm^2/s]:	0.11										
12	Lunghezza gomitolo [cm]:	6.12										
13	SwayPath [cm/s]:	1.22										
14												
15												
16												
17												
18												
19												

boioli_giacomo_11-Jul-2013_esit

Pronto 140%

Figura 3.4. *Esempio di file di tipo esito salvato dal programma*

3.2 Utilizzo del programma

Il programma presenta molteplici modalità di utilizzo. Come già accennato si può operare con una sola pedana o con due contemporaneamente. Inoltre è possibile scegliere se utilizzare le due WBB affiancate o sovrapposte (3.5).

La modalità affiancata è utile per permettere la valutazione della capacità di equilibrio posturale di persone affette da forte obesità. Il peso dell'utilizzatore è infatti ripartito su due pedane. In questo modo la periferica potrà operare entro il suo intervallo di lavoro dichiarato (si veda 1.1).

La modalità sovrapposta può essere utilizzata per la calibrazione della WBB rispetto a una *balance board* già calibrata presa da riferimento. In questo modo non bisogna ricorrere a una pedana stabilometrica professionale per ottenere un buon livello di calibrazione

(a) modalità affiancata **(b)** modalità sovrapposta

Figura 3.5. *Modalità di utilizzo*

Avviato il programma, completate le fasi di connessione, calibrazione, creazione del profilo e scelto il numero di pedane da utilizzare si può procedere alla sceltà della modalità si lavoro.

Tramite il *Pop-up Menù* si può scegliere tra:

- Modalità *standard*
- Modalità *personalizzata*

L'utilizzo *standard* prevede un tempo di registrazione predeterminato di 30s, inoltre l'unità di misura, che non potrà essere modificata, è il centimetro.

Se si sceglie la modalità *personalizzata*, alla pressione del pulsante *START RECORDING*, due finestre di dialogo permetteranno di introdurre rispettivamente il tempo di registrazione desiderato (valore massimo 300s) e di poter scegliere tra la visualizzazione dei risultati in centimetri o millimetri.

Durante la registrazione il punto che individua il CoP cambierà colore in rosso e un oggetto grafico di tipo *Static Text* fornirà l'indicazione sul tempo trascorso da inizio registrazione (fig. 3.6). Durante la registrazione il file *trial* è continuamente aggiornato ad ogni ciclo.

Al termine della registrazione il pulsante *SHOW TRACK* permette di visualizzare il *gomitolo* (fig. 3.7) mentre il pulsante PRINT RESULT permette di creare il file di tipo *esito* nella cartella di profilo personale.

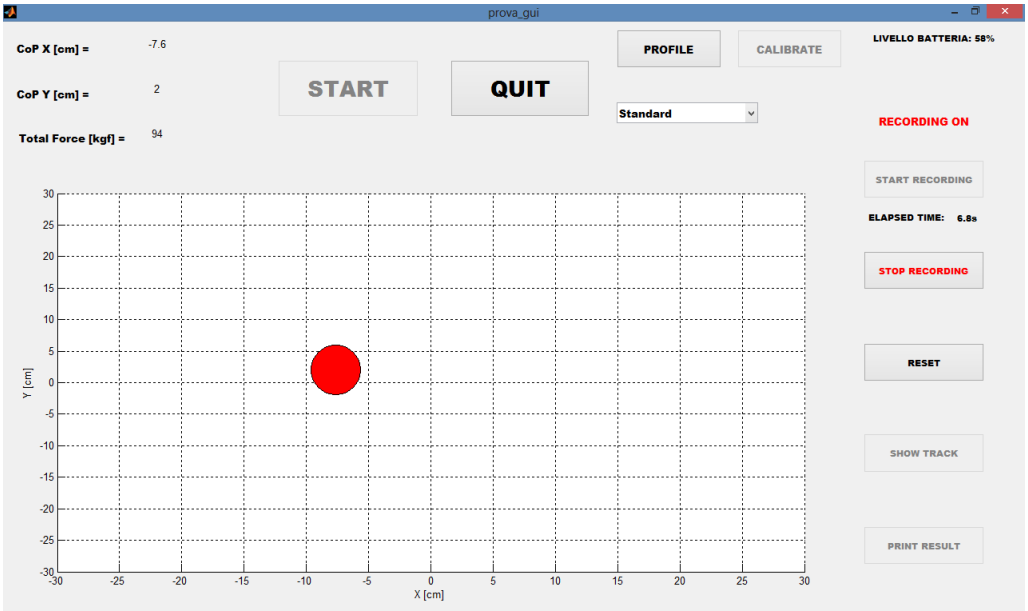


Figura 3.6. *Interfaccia grafica durante la registrazione di un trial*

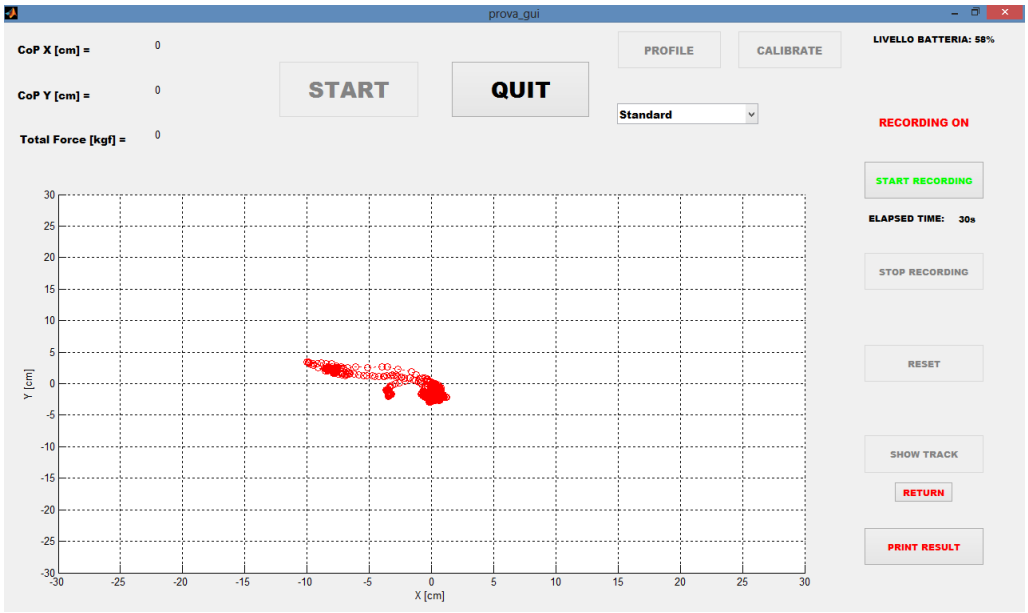


Figura 3.7. *Interfaccia grafica durante la visualizzazione del gomitolo*

3.2.1 Installazione dell'applicazione

Il pacchetto software è distribuito tramite un file autoestraente. Una volta avviato il programma si osserva l'estrazione dei file inclusi al momento della fase di *packaging* (1.2.2). Sono presenti:

- il programma eseguibile
- la MCR
- Una cartella nominata *librerie*
- Un file *batch* nominato *install_lib*
- il file *leggimi*

Nel frattempo una procedura guidata conduce l'utente all'installazione della MatLab Compiler Runtime.

L'ultimo passaggio necessario per completare il processo è l'avvio del file *install_lib* che permette la registrazione delle due librerie esterne (WiimoteLib e WiiLab descritte in 2.3) su cui si basa l'applicativo.

A questo punto, se la procedura non ha generato errori, il programma è stato installato correttamente e può essere utilizzato.

3.2.2 Connessione della Wii Balance Board

Per connettere la WBB è necessario disporre di una comunicazione bluetooth. La periferica non supporta tutti i dispositivi bluetooth (per una lista completa riferirsi a [21]). Per lo studio svolto è stata utilizzata una *Bluetooth 3.0 USB adapter (cod 17772)* prodotta dalla *Trust*. Come software per la gestione della comunicazione bluetooth è stato utilizzato il gestore di *default* disponibile in *Windows 8*.

Una volta scelto il dispositivo per attivare la comunicazione bluetooth si può iniziare l'operazione di sincronizzazione della pedana. Come accennato in 1.1, nel vano batterie, posto sul retro della piattaforma, è presente il pulsante di *sinc*. Tenendo premuto il pulsante la WBB si rende visibile (con nome identificativo: "Nintendo RVL-WBC-01") e attraverso il computer è possibile individuarla e creare l'associazione. Questa periferica non richiede chiave di accesso. Durante questa operazione l'unico *led* presente lampeggerà. La connessione di una ulteriore piattaforma non induce variazioni nella procedura di rilevamento.

Una volta avviato il programma e richiamato il metodo *Connect()* si realizzerà la vera e propria connessione e il led smetterà di lampeggiare. Nel caso di un utilizzo congiunto di due WBB il led della seconda pedana si spegnerà.

3.2.3 Visualizzazione corretta dei dati

Come osservato nella sezione 3.1.1, il programma realizzato salva i dati ottenuti ed elaborati in due file *Excel*. Per la scrittura nei file si utilizza il comando *fprintf* [22]. Si tratta, però, di una forzatura. L'istruzione, che è adatta alla scrittura dati in file di tipo testo *.txt*, è stata utilizzata, invece, per scrivere in un file creato con estensione *.xls*. A causa di questo fatto, all'apertura del file salvato, si osserverà la comparsa di un messaggio di errore. I dati saranno inoltre visualizzati in notazione esponenziale e la larghezza

delle colonne non risulterà adattata. Particolare attenzione, inoltre, si dovrà porre al tipo di convenzione utilizzata nel computer per la separazione delle cifre decimali. MatLab utilizza il punto come separatore decimale e i dati verranno scritti sul file seguendo tale metodo. L'utilizzo di una convenzione differente provocherà una visualizzazione errata dei dati.

Quindi per una opportuna visualizzazione dei dati (come nelle figure 3.3 e 3.4) bisogna procedere attraverso i seguenti passi:

1. impostare la convenzione corretta per il separatore decimale
2. eseguire le operazioni di adattamento automatico delle colonne
3. cambiare la modalità di rappresentazione dei numeri da scientifica (esponenziale) a decimale
4. salvare il file in modo da non visualizzare più il messaggio di errore

CONCLUSIONI

La lavoro svolto è mirato al raggiungimento di due principali obiettivi: la realizzazione di un software indipendente realizzato con un'interfaccia grafica semplice ed intuitiva e la possibilità di utilizzare più di una Wii Balance Board contemporaneamente.

L'applicazione realizzata presenta un buon grado di facilità di utilizzo e non richiede un addestramento specifico da parte dell'utilizzatore.

Si è ottenuto un buon livello di portabilità anche se la procedura di installazione risulta piuttosto complessa, per una eventuale distribuzione su larga scala del programma si potrebbe pensare di implementare via software una procedura guidata di installazione completa.

L'utilizzo congiunto di due pedane permette l'accesso allo strumento anche a persone affette da obesità fino ad un limite massimo di utilizzo ipotizzato di circa 250kg. Nonostante il limite strutturale di una singola pedana è calcolato sui 300kg circa, infatti, bisogna tener presente che al momento del posizionamento del soggetto sulle due pedane, può capitare che il peso sia scaricato temporaneamente su una sola *balance board*.

L'organizzazione dell'applicazione tramite il *tool* GUIDE è stata realizzata in modo da permettere a sviluppatori futuri rapidi aggiornamenti e modifiche, inoltre, l'implementazione di nuove funzioni risulta agevole grazie alla programmazione grafica permessa da *tool*.

Un ultima considerazione riguarda la frequenza di acquisizione dei dati. Il principale limite alla velocità di lettura riguarda la gestione della parte grafica. Senza modifiche il programma riesce ad acquisire i dati ad una frequenza di 30 Hz. Se la parte grafica viene esclusa dal ciclo di lettura si è osserva un incremento della frequenza fino a 60 Hz nel caso di utilizzo di una

singola *balance board*. Nel caso di utilizzo di due pedane assieme, escludendo la parte grafica, la frequenza si attesta a 50 Hz.

Bibliografia

[1] WINTER, D. A., Human balance and posture control standing and walking, Gait & Posture: 1995; Vol 3: 193-214.

[2] YOUNG, W., FERGUSON, S., BRAULT, S. CRAIG, C., Assessing and training standing balance in older adults: A novel approach using the 'Nintendo Wii' Balance Board, Gait & Posture 33 (2011): 33-305.

[3] TINETTI, M. E.& WILLIAMS, C., Fall, injuries due to falls, and the risk of admission to a nursing home, New England Journal of Medicine, 337(18), 1279-1284.

[4] CAPODAGLIO, P., CAPODAGLIO, E.M., HELMER, PRECILIOS, VISMARA, L., TACCHINI, E., FINOZZI, E., BRUNANI, A., Obesità e lavoro: un problema emergente, PI-ME, Pavia, 2011

[5] WINTER, D. A., PATLA, E. A., PRINCE, F., ISHAC, M. & GIELO-PERCZAK, K., Stiffness Control of Balance in Quiet Standing, J Neurophysiol 80:1211-1221, 1998.

[6] IGN, GDC 2008, Sawano on Wii Fit: Nintendo director Takao Sawano talks about the creation of the company's hit exercise title.

URL (luglio 2013):

`http://www.ign.com/articles/2008/02/21/gdc-2008-sawano-on-wii-fit`

[7] Wikipedia, l'enciclopedia libera, Estensimetro

URL (luglio 2013):

http://it.wikipedia.org/wiki/Estensimetro

[8] MathWorks, MATLAB GUI

URL (luglio 2013):

http://www.mathworks.com/discovery/matlab-gui.html

[9] MathWorks, MATLAB Compiler

URL (luglio 2013):

http://www.mathworks.it/products/compiler/index.html

[10] USB Implementers Forum, Device Class Definition for Human Interface Devices (HID), Version 1.11,

URL (luglio 2013):

http://www.usb.org/developers/devclass_docs/HID1_11.pdf2

[11] Wikipedia, the free encyclopedia, Polling (computer science)

URL (luglio 2013):

http://en.wikipedia.org/wiki/Polling_(computer_science)

[12] Wikipedia, l'enciclopedia libera, Interrupt

URL (luglio 2013):

http://it.wikipedia.org/wiki/Interrupt

[13] WiiBrew Wiki, Wiimote

URL (luglio 2013):

http://wiibrew.org/wiki/Wiimote/

[14] PEEK, B., BrianPeek.com, WiimoteLib - .NET Managed Library for the Nintendo Wii Remote,

URL (luglio 2013):

http://www.brianpeek.com/blog/pages/wiimotelib.aspx

[15] WiiBrew Wiki, Wii Balance Board

URL (luglio 2013):

http://wiibrew.org/wiki/Wii_Balance_Board

[16] PEEK, B., BrianPeek.com, A Compendium of Random Uselessness

URL (luglio 2013):

http://brianpeek.com/page/wiimotelib

[17] Wikipedia, l'enciclopedia libera, Wrapper

URL (luglio 2013):

http://it.wikipedia.org/wiki/Wrapper

[18] Kording Lab, Wii-project

URL (luglio 2013):

http://klab.wikidot.com/wii-proj

[19] University of Notre Dame, WiiLAB Wiki Page and Class Documentation, WiiLAB Example Code - Wiimote Class: Wiimote Properties

URL (luglio 2013):

http://netscale.cse.nd.edu/twiki/bin/view/Edu/ExWiiLabWiimoteClassProperties

[20] MathWorks, convhull

URL (luglio 2013):

http://www.mathworks.it/it/help/matlab/ref/convhull.html

[21] WiiBrew Wiki, List of Working Bluetooth Devices

URL (luglio 2013):

http://wiibrew.org/wiki/List_of_Working_Bluetooth_Devices

[22] MathWorks, fprintf

URL (luglio 2013):

`http://www.mathworks.it/it/help/matlab/ref/fprintf.html`