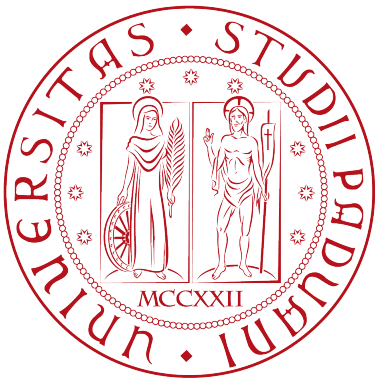


Università degli Studi di Padova

Dipartimento di Ingegneria dell'Informazione

Corso di Laurea in Ingegneria Elettronica

Sviluppo di applicazioni basate su Wiimote per scopi didattici

Laureando:
Daniele Rucatti

Relatore:
prof.ssa Giada Giorgi

Anno accademico 2009/2010

A mia madre...

Sommario

L'obiettivo di questo lavoro di tesi è stato lo sviluppo di semplici applicazioni in *Labview* per scopi didattici basate sull'utilizzo del controller *Wiimote* prodotto da *Nintendo* per la console per videogiochi *Wii*.
Nella prima parte dello scritto verranno analizzati il controller ed i sensori in esso contenuti, con particolare attenzione all'accelerometro *Anolog Devices ADXL330* di cui dispone. Verranno analizzate le principali tipologie di accelerometri ed i parametri più importanti che ne descrivono le caratteristiche. Si focalizzerà quindi l'attenzione sugli accelerometri MEMS (*Micro Electro Mechanical Systems*) di dipo capacitivo, categoria alla quale appartiene il sensore del *Wiimote*, analizzandone modello fisico, circuito di trasduzione e condizionamento del segnale, conversione analogica/digitale. Nella seconda parte dello scritto verranno affrontati i temi della connessione tra il *Wiimote* ed un *PC Windows* e dell'acquisizione dei dati, attraverso l'uso delle librerie driver *WiimoteLib* e *WiiLAB* e l'utilizzo dei software *Matlab* e *Labview*. L'ultima parte della tesi è dedicata alla descrizione delle applicazioni realizzate: "*Wiimote in Catuda Libera"* permette lo studio della caduta di un corpo; *"Wiimote Inclinometro"* permette di eseguire misure di inclinazione; *"Wiimote Frequenzimetro"* permette di determinare la frequenza di un movimento periodico applicato al controller e di visualizzare lo spettro del segnale acquisito da esso. Queste applicazioni possono essere un valido strumento per uno studente che si trova ad affrontare per la prima volta i concetti fisici/matematici elencati.

Indice

INTRODUZIONE

Nintendo è una famosa azienda giapponese sviluppatrice di videogiochi e console. Fondata nel 1889, è passata dai giochi di carte e meccanici ai videogiochi agli inizi degli anni 80. Nel 2006 mette in commercio "Wii" [2] (figura 1), che si differenzia dalle altre console[1] presenti sul mercato, grazie al controller[2] innovativo di cui dispone: il Wiimote (o *Wii Remote*).

Figura 1: La Console Wii assieme al controller Wiimote

Il Wiimote ricorda, nella forma, un telecomando, ed è l'interfaccia mediante la quale un utente può interagire con il videogioco (figura 2). Si tratta di un dispositivo wireless che contiene al suo interno diversi sensori (di posizione, di accelerazione) che permettono di rendere molto realistico il videogioco. Il giocatore, infatti, è in grado di partecipare al gioco con il proprio corpo, utilizzando il Wiimote, di volta in volta come una mazza da baseball, una racchetta, ecc; sullo schermo l'alter ego virtuale utilizzerà tale attrezzo per compiere delle azioni come ad esempio colpire la pallina. Il successo della console Wii è dovuto proprio al modo con il quale il giocatore si relaziona con il gioco, ovvero all'elevato livello di interattività tra uomo e macchina.
Il Wiimote è, dal punto di vista ingegneristico, un sistema di misura composto da: sensori, blocchi di elaborazione dei dati, moduli di comunicazione radio. In questo capitolo verrà descritto il sistema di misura "Wiimote". Si analizzerà la dotazione hardware del Wiimote, in modo da mettere a nudo tutte le potenzialità di questo dispositivo, e quindi si focalizzerà l'attenzione sulla principale funzione del

[1] *Console*: nel mondo videoludico, dispositivo elettronico concepito esclusivamente o primariamente per giocare con videogiochi.

[2] *Controller*: nel mondo videoludico, dispositivo che permette il controllo dell'azione di gioco (escluse tastiere e mouse).

Figura 2: Uso del Wiimote

controller, ovvero la misura dell'accelerazione. A tal fine verà richiamata la teoria che sta alla base dei sensori di accelerazione fino all'analisi approfondita dell'accelerometro di cui è dotato il Wiimote e del sistema di misura necessario alla sua lettura. Obiettivo finale sarà la realizzazione di alcune applicazioni per PC Windows, che sfruttano il controller per eseguire semplici esperimenti di fisica.

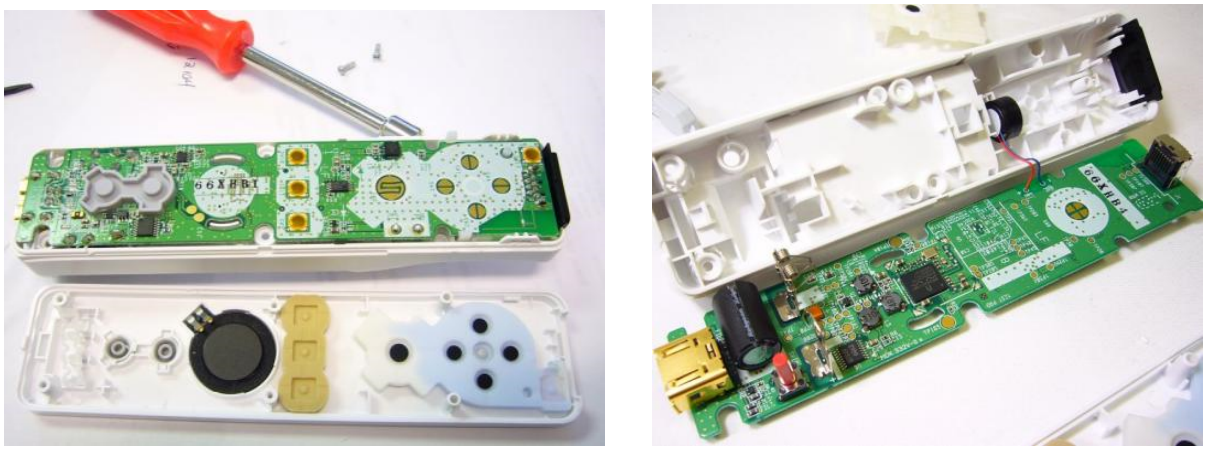
Capitolo 1

WIIMOTE: HARDWARE

L'interno del controller si presenta come un unico PCB (*Printed Circuit Board*) su cui sono saldati diversi componenti (figura 1.1). I principali sono:

- Una microcamera infrarossa.
- L'accelerometro ADXL330, prodotto dalla *Analog Devices*, oggetto di approfondimento delle sezioni successive.
- Uno speaker audio per produrre suoni e relativi driver di pilotaggio.
- Un motorino in corrente continua per la generazione di vibrazioni (usate come mezzo di interazione con l'utente).
- Pulsanti e LED di segnalazione posizionati in appositi alloggiamenti nell'involucro plastico.
- Una memoria E^2PROM usata per il salvataggio di dati e configurazioni durante il funzionamento del controller. In essa viene memorizzata anche una parte del firmware del dispositivo (è possibile effettuare aggiornamenti del firmware via Bluetooth).
- L'integrato BCM2042, prodotto dalla *Broadcom*, che costituisce uno dei componenti principali del controller.
  Esso si occupa sia della comunicazione wireless Bluetooth che dei diversi dispositivi di interfaccia e servizi. Il chip integra al suo interno un microprocessore 8051, memorie ROM, FLASH e RAM, stack Bluetooth e profilo *Human Device Interface* (HID), modulo per la trasmissione radio completo (necessaria solo l'antenna esterna), regolatore switching per l'alimentazione dello stesso e di eventuali periferiche esterne [11].

La comunicazione tra i vari dispositivi ed il BCM2042 avviene tramite un bus a due linee basato sul protocollo seriale sincrono I^2C (*Inter Integrated Circuit*).
Il controller è inoltre dotato di un connettore di espansione (posto sul lato inferiore), al quale arriva direttamente il bus I2C, che viene usato principalmente per interfacciarsi con le "Estensioni" della Nintendo, accessori aggiuntivi da collegare al Wiimote per ampliare le modalità di gioco previste.
In figura 1.2 viene presentato uno schema a blocchi che raffigura a livello concettuale l'hardware del controller.



(a) Lato Superiore (b) Lato Inferiore

Figura 1.1: L'interno del Wiimote [10]

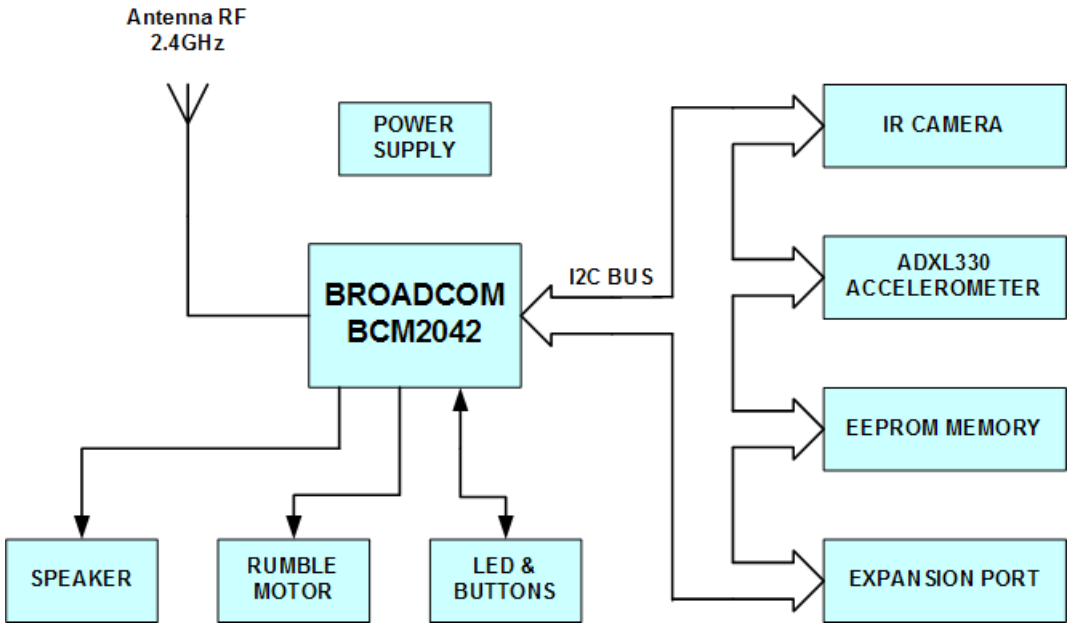


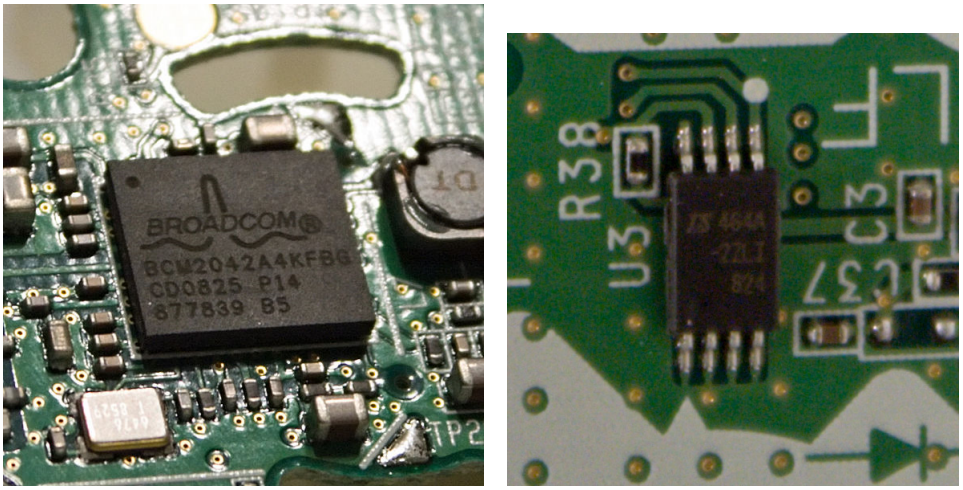
Figura 1.2: Schema concettuale Hardware del Wiimote

Verranno ora analizzati alcuni elementi dell'hardware del controller, per approfondimenti si vedano [8] e [10].

1.1 Broadcom BCM2042 ed M24128 EEPROM

Il Broadcom BCM2042 (figura 1.3a) è un integrato a basso costo molto usato per le Tastiere ed i Mouse wireless. Esso si occupa di gestire la comunicazione Bluetooth con la console (o altro dispositivo) utilizzando il profilo Bluetooth HID (*Human Interface Device*) [3, 5]. Tale profilo è basato sullo standard USB-HID [4], usato per periferiche di puntamento quali appunto mouse, tastiere e controllers. Il Wiimote viene riconosciuto dall'esterno come una qualsiasi periferica di puntamento,

tuttavia il protocollo HID che implementa non fa uso dei tipi di dati definiti dallo standard HID, bensì utilizza dei formati proprietari quindi ne rendono impossibile l'utilizzo per mezzo dei driver HID Standard disponibili attraverso ogni sistema operativo. Sono comunque reperibili in internet delle librerie driver per poter comunicare con il controller su piattaforme Linux/Windows [9] ed utilizzare quindi l'hardware del Wiimote per diverse applicazioni, come quelle che verranno presentate in questa tesi.
Una breve descrizione della comunicazione tra il controller e l'host, con la descrizione della composizione dei pacchetti dati sarà data in seguito.
Il BCM2042 si occupa della gestione dei dati da trasmettere e della comunicazione con l'host (la console). Il programma per il processore, ovvero il firmware del Wiimote, è contenuto in parte nell memoria EEPROM M24128 (figura 1.3b) ed in parte nella ROM interna all'integrato, questo ne rende possibili eventuali aggiornamenti. Il firmware si occupa del controllo completo del sistema, dalla lettura dei sensori (Accelerometro e Camera IR) all'elaborazione dei dati, alla loro trasmissione attraverso il ponte radio.
Lo spazio in memoria non occupato dal firmware contiene informazioni legate al dispositivo, codici identificativi, registri per la comunicazione con i sensori e per il salvataggio dei dati delle letture, infine parte dello spazio viene utilizzato liberamente dall'host per la scrittura/lettura di informazioni.



(a) Il Broadcom BCM2042 (b) La E^2PROM M24128

Figura 1.3: Il chip Broadcom e L'EEPROM esterna [8]

1.2 La IR Camera

Il controller include una microcamera monocromatica con filtro infrarosso e con microcontrollore integrato per l'elaborazione dell'immagine (figura 1.4). La microcamera comunica attraverso il bus seriale I2C restituendo direttamente le coordinate degli oggetti luminosi individuati. Per essere correttamente identificati, tali oggetti luminosi devono essere sorgenti puntiformi di luce infrarossa, in numero inferiore a 5. Tracciando il movimento di queste fonti luminose è possibile stabilire la posizione del Wiimote nello spazio.
Il microcontrollore interno alla microcamera elabora le foto scattate e restituisce le posizioni dei punti luminosi, che costituiscono un sistema di riferimento fisso, con una risoluzione di 1024x768 pixel. Successivamente la posizione del controller può essere calcolata misurando le distanze tra i punti ed effettuando ragionamenti geometrici.

La microcamera è dotata di un filtro plastico che permette il passaggio della sola luce infrarossa, senza questo filtro verrebbe tracciato qualsiasi oggetto luminoso. Assieme alla console Wii viene fornita una barra lunga una ventina di centimetri con dei led infrarossi ad elevata luminosità alle estremità. La barra viene posta sotto il televisore e, grazie ad una procedura di taratura guidata effettuata per mezzo della console, il Wiimote può fornire ai videogiochi informazioni sulla sua posizione nella stanza ed essere usato come sistema di puntamento assoluto (ad esempio in un gioco "sparatutto" è usato per muovere il mirino dell'arma). Le operazioni da eseguire per effettuare la taratura consistono nel posizionarsi a circa 1.5m dal televisore, puntare il controller verso di esso ed eseguire i passi proposti dalla procedura guidata, che permette di definire i punti di riferimento (angoli dello schermo) da utilizzare per il calcolo della posizione del dispositivo.
Il campo di visione della fotocamera risulta essere di circa 33° come apertura orizzontale e di 23° come apertura verticale.

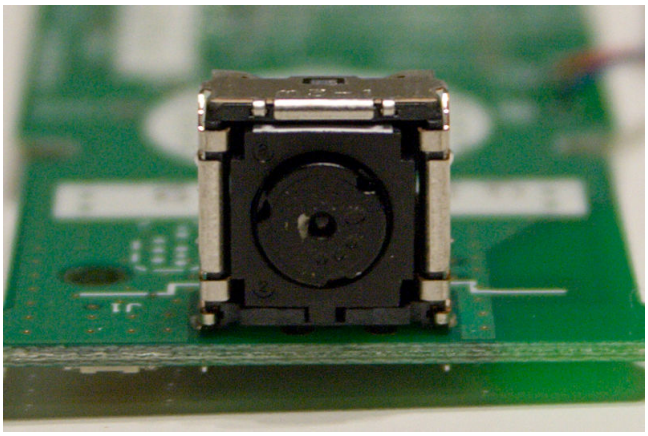


Figura 1.4: La IR Camera del Wiimote (Lato superiore PCB) [8]

Utilizzando un dispositivo composto da 4 led infrarossi disposti nello spazio, dei quali si conosce la disposizione e la distanza, è possibile grazie alla microcamera risalire alla posizione ed all'orientazione del Wiimote nello spazio 3D [12]. Infatti conoscendo la posizione del punto focale della fotocamera rispetto al piano dell'immagine è possibile calcolare le coordinate relative alla posizione della IR Camera rispetto alla posizione dei led infrarossi (usati come punti di riferimento). In figura 1.5 viene rappresentato schematicamente quanto detto.

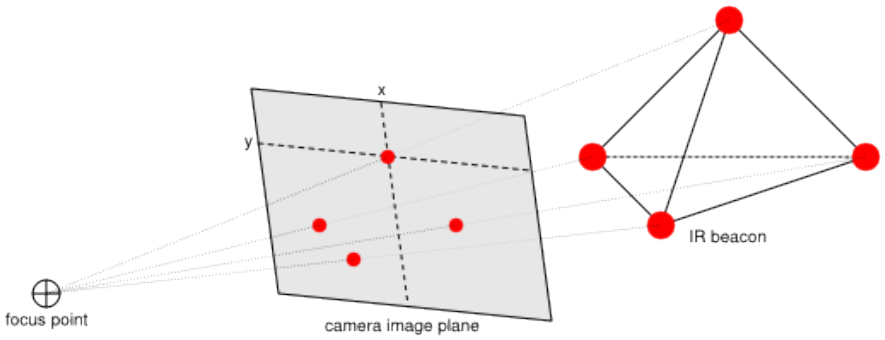


Figura 1.5: La proiezione sul piano dell'immagine di 4 Led disposti nello spazio permette risalire alla posizione della IR Camera [12]

1.3 Accelerometro ADXL330

Il controller integra l'accelerometro a 3 assi *ADXL330* [15] prodotto dalla *Analog Devices* per la misura dell'accelerazione, mostrato in figura 1.6.

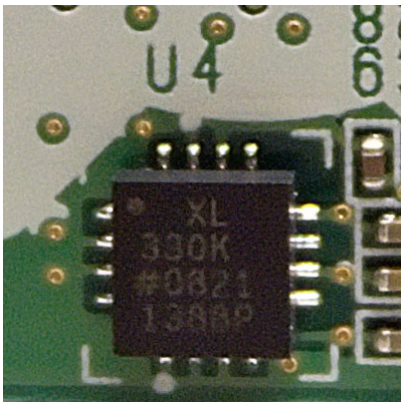


Figura 1.6: ADXL330 Linear 3-axis accelerometer [8]

Il sensore fornisce i valori di acclerazione sui tre assi per mezzo di 3 uscite analogiche proporzionali all'accelerazione a cui il sensore è sottoposto, una per ogni asse. Il range di misura (*measure range*) del sensore va da $-3g$ a $+3g$ (dove l'unità g è pari a $9.81 m/s^2$), l'accelerazione massima misurabile risulta quinti essere pari, in modulo, a 3 volte l'accelerazione gravitazionale terrestre. Nel caso comune di tensione di alimentazione pari a $V_s = 3V$ la sensibilità (*sensitivity*) del sensore risulta essere pari a $0.3V/g$ (valore tipico), inoltre la tensione in uscita corrispondente alla misura di accelerazione nulla (0g) risulta essere pari a 1.5V (valore tipico), ovvero pari alla metà della tensione di alimentazione. Una tensione compresa tra 0V e questo valore rappresenta un'accelerazione negativa, positiva altrimenti. Il segno dipende dalla direzione con cui è applicata l'accelerazione in riferimento alla definizione degli assi usata dal sensore. Lo scostamento dalla linearità (*nolinearity*) del sensore (nel range $\pm 3g$) è bassa, il datasheet del sensore [15] riporta come valore tipico $\pm 0.3\%$ (espresso come percentuale sul valore di fondoscala). Un estratto del datasheet che riporta i parametri menzionati è riportato in figura 1.7.

Parameter	Conditions	Min	Typ	Max	Unit
SENSOR INPUT	Each axis				
Measurement Range		±3	±3.6		*g*
Nonlinearity	% of full scale		±0.3		%
Package Alignment Error			±1		Degrees
Interaxis Alignment Error			±0.1		Degrees
Cross Axis Sensitivity[1]			±1		%
SENSITIVITY (RATIOMETRIC)[2]	Each axis				
Sensitivity at X_{OUT}, Y_{OUT}, Z_{OUT}	V_S = 3 V	270	300	330	mV/*g*
Sensitivity Change Due to Temperature[3]	V_S = 3 V		±0.015		%/°C
ZERO *g* BIAS LEVEL (RATIOMETRIC)	Each axis				
0 *g* Voltage at X_{OUT}, Y_{OUT}, Z_{OUT}	V_S = 3 V	1.2	1.5	1.8	V
0 *g* Offset vs. Temperature			±1		mg/°C

Figura 1.7: Parametri Range di misura, scostamento dalla linearità, sensibilità (estratto dal datasheet dell'accelerometro ADXL330 [15])

I parametri fondamentali che caratterizzano un accelerometro verranno discussi nel capitolo 2 sezione 2.4, nel capitolo 3 verrà invece analizzato in modo più approfondito il sensore ADXL330.

I segnali analogici provenienti dal sensore vengono elaborati dal Broadcom BCM2042 e successivamente le informazioni relative all'accelerazione misurata vengono inviate via Bluetooth .
L'accelerometro misura la forza applicata ad un insieme di "masse di prova" contenute nel sensore stesso. Il modo in cui tali masse sono costruite, ed il blocco di trasduzione e condizionamento adottato all'interno di questa tipologia di sensori sarà discusso in un'apposita sezione.
Quando il sensore non è soggetto ad alcuna forza (come nel caso ideale di un ambiente privo di gravità) la misura dell'accelerazione è nulla su tutti e tre gli assi. Nella realtà invece non è così, infatti è sempre presente la forza di gravità che agisce sulla "massa di prova" interna. Quindi le accelerazioni misurate sui tre assi rappresentano le componenti dell'accelerazione gravitazionale. In figura 1.8 è mostrato il Wiimote con gli assi di riferimento associati all'accelerometro interno.

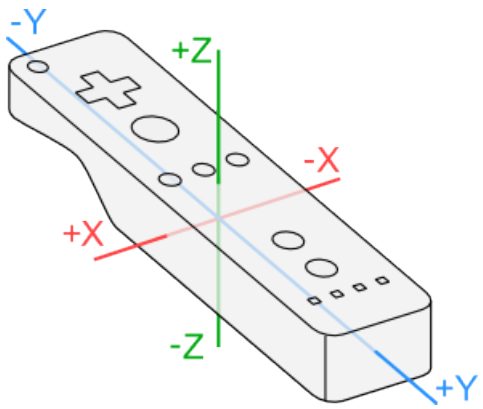


Figura 1.8: Assi di riferimento per la misura dell'accelerazione del controller Wiimote

Se ipotizziamo di tenere il controller su un piano in modo che l'asse z positivo sia rivolto verso l'alto allora il sensore misurerà una accelerazione costante negativa sull'asse z pari a $-g$, dovuta alla forza di gravità, mentre sugli altri due assi l'accelerazione risulterà nulla.
Il sensore può in questo modo anche essere usato come un "tilt sensor" [35], ovvero come un dispositivo in grado di distinguere l'inclinazione di un oggetto: semplicemente osservando le componenti della forza di gravità misurate dai 3 assi dell'accelerometro si può stabilire l'inclinazione dell'oggetto (figura 1.9), questo però se il sensore non viene mosso lungo gli assi (condizioni statiche). In questo caso risulta impossibile separare il contributo della gravità dal contributo della forza applicata dall'utente sull'accelerometro (movimento del sensore), per cui difficilmente si riuscità a stimarne l'inclinazione. Per misure precise di inclinazione è necessario un giroscopio, a differenza dell'accelerometro esso permette di effettuare misure di inclinazione anche con il sensore in movimento (condizioni dinamiche). La Nintendo fornisce questo sensore come *Estensione del Wiimote*, commercialmente chiamata *Wiimote Plus*.

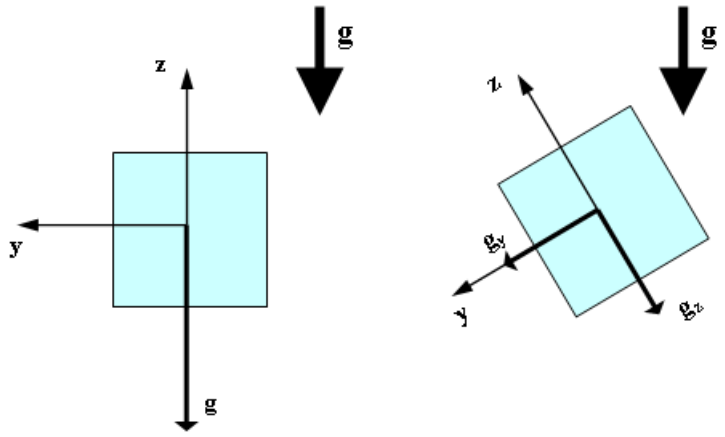


Figura 1.9: Suddivisione delle componenti della gravità a seconda dell'inclinazione (esempio)

1.4 Comunicazione, lettura dati ed applicazioni

1.4.1 Profilo Bluetooth HID

Il profilo Bluetooth HID (*Human Interface Device*) usato dal controller per la comunicazione tramite protocollo Bluetooth è identico a quello USB HID usato da alcune periferiche USB come ad esempio mouse e tastiere (si vedano le specifiche in [4]). Il protocollo HID (Human Device Interface) è stato appositamente pensato per tutti quei dispositivi che realizzano una "Interfaccia umana", cioè dispositivi di puntamento, gamepad per il gioco, strumenti multimediali ed anche strumenti medici. Il vantaggio sta nel fatto che sono le stesse periferiche HID a comunicare al dispositivo host le loro "credenziali", ad identificarsi, a definire quanti dati trasmettono ed il loro significato.
Un'unità HID fornisce all'host diverse *interfacce* (in inglese *interface*), ogni interfaccia rappresenta un particolare dispositivo o servizio presente nell'unità (per esempio se l'unità è dotata di una tastiera e di un mouse verranno create due interfacce distinte). L'interfaccia descrive se stessa e come intende comunicare con l'host attraverso un *descriptor block*, contenuto nell'interfaccia stessa, che definisce tutti i *reports* che vengono trasmessi. I reports non sono altro che i dati effettivi che vengono poi inviati all'host: avendo ricevuto il *descriptor block* l'host è in grado di prepararsi a ricevere i dati, perchè conosce cosa dovrà ricevere. Un report è unidirezionale, può essere solo di ingresso o di uscita, i report di ingresso contengono dati per l'host (per esempio i valori dei tasti di una tastiera), mentre i report di uscita contengono i dati di configurazione/controllo dell'interfaccia (per esempio l'accensione dei leds indicativi della tastiera). Ad ogni report è associato un ID univoco, che viene usato dall'host per la lettura o scrittura del dato. I campi informativi associati ad un report sono il numero di bytes che compongono il report ed un'eventuale descrizione dei dati contenuti.
La comunicazione con l'host può avvenire in due modi, o tramite polling o tramite interrupt. Nella prima modalità è l'host ad inviare periodicamente al dispositivo una richiesta di invio dati, il dispositivo prepara i report e li invia all'host come risposta. Nel secondo modo invece è l'*interface* che, quando ha dei dati da inviare all'host, genera un interrupt ed invia i reports. La cosa può avvenire in modo sincrono o asincrono. Nel modo sincrono la trasmissione dei dati avviene periodicamente ad intervalli regolari, vengono sempre trasmessi gli ultimi dati disponibili in memoria (*streaming*), nel modo asincrono (*On Event*) invece la trasmissione avviene successivamente ad un evento (ad esempio la pressione di un pulsante del dispositivo).
Per molti dispositivi di uso comune sono già stati definiti i reports che devono essere inviati/ricevuti, come nel caso di mouse e tastiere HID, che sostituiscono le vecchie PS2, e quindi esistono dei drivers standard che supportano in maniera nativa queste periferiche. Tuttavia il profilo HID è espandibile ed

un produttore può decidere autonomamente caratteristiche e funzioni dei report del proprio oggetto, in questo caso però deve provvedere a realizzare un apposito driver affinché il dispositivo host sia in grado di interpretare i dati e comunicare con esso.
Questo è il caso del controller Wiimote, che implementa il profilo HID in maniera non standard, non rientrando quindi in nessuna categoria di uso comune già implementata nei sistemi operativi o di cui comunque esiste un driver "universale". Il Wiimote nel *descriptor block* comunica solo le dimensioni dei reports, tuttavia alcune informazioni sui report trasmessi si possono trovare in internet, ad esempio in [8] viene specificata la tabella dei report HID usati dal Wiimote e che verranno ampiamente descritti nel seguito di questa tesi.

1.4.2 Lettura dei dati tramite PC

Nonostante il Wiimote sia progettato per funzionare solo con la console Wii, sono nati diversi progetti che mirano a costruire un driver completo per tutte le funzionalità del dispositivo e pienamente funzionante sui comuni sistemi operativi.
È stata realizzata una libreria che permette la comunicazione via Bluetooth tramite lo standard HID e l'interpretazione dei dati (*reports*) inviati dal Wiimote. Richiamando la libreria, attraverso un qualsiasi linguaggio di programmazione, *C*, *Java*, *.Net*, è possibile accedere ai dati inviati dal controller, ovvero i valori di accelerazione misurati dall'accelerometro, le coordinate dei punti luminosi individuati dalla IR Camera, lo stato dei pulsanti ed il livello della batteria.
Alcune librerie disponibili, ancora in via di sviluppo (per la lista completa si veda [9]):

- *WiiUse* [25], scritta in C e compatibile con i sistemi operativi *Linux* e *Windows* (presto anche *Mac*). Supporta gran parte delle funzioni del controller ed anche alcune estensioni, è compatibile con gli stack Bluetooth più usati (*Microsoft* e *Windcomm*).

- *WiimoteLib* [23], scritta in *.Net*, per sistemi Windows, ha più o meno tutte le caratteristiche della *WiiUse*.

- *WiiLAB* [31, 32], scritta in *.Net*, è una libreria *wrapper* che permette la comunicazione tra l'ambiente *MATLAB* e la libreria *WiimoteLib* (vedi sezione 4.2). La libreria *WiimoteLib* deve comunque essere sempre presente e collocata nella stessa directory della *WiiLAB*. La libreria rende disponibili al programmatore dei *metodi* di più alto livello rispetto a quelli disponibili nella *WiimoteLib*, basta infatti la chiamata di un solo *metodo* per ottenere i dati dal dispositivo. La libreria può essere utilizzata anche da applicazioni diverse da *MATLAB*.

Grazie alle librerie sono stati realizzati dei software che permettono di controllare il mouse del computer, di usare il Wiimote per scorrere i menù di un *PC-MediaCenter* (per la visione di contenuti multimediali), per giocare nei videogiochi su PC e moltissime altre applicazioni.
Uno dei software più usati per l'interfacciamento con il Wiimote in ambiente Windows è *GlovePie*, realizzato da Carl Kenner [21], si veda figura 1.10.

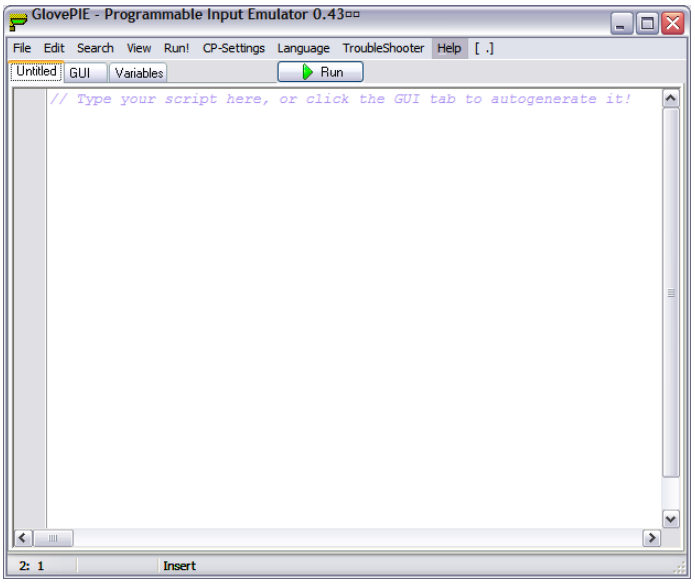


Figura 1.10: La schermata iniziale del Software GlovePie

Interessante risulta essere la possibilità, tramite le librerie sopra citate, di "interfacciare" il Wiimote con *LabView* [22], noto software prodotto dalla National Instrument per l'acquisizione e l'elaborazione dei dati da strumenti di misura. Labview è in grado di interagire con le librerie scritte in *.NET* ed utilizzarne i *metodi*, in questo modo è possibile comunicare con il controller attraverso la libreria *WiimoteLib* (oppure la *WiiLAB*). L'argomento verrà approfondito nella sezione 4.2.5, infatti Labview sarà oggetto principale del capitolo 5, dove verrà utilizzato per la realizzazione di alcune applicazioni che fanno uso del controller per scopi didattici. In figura 1.11 è mostrato il pannello frontale di un VI per l'acquisizione e la visualizzazione dei dati trasmessi dal Wiimote.

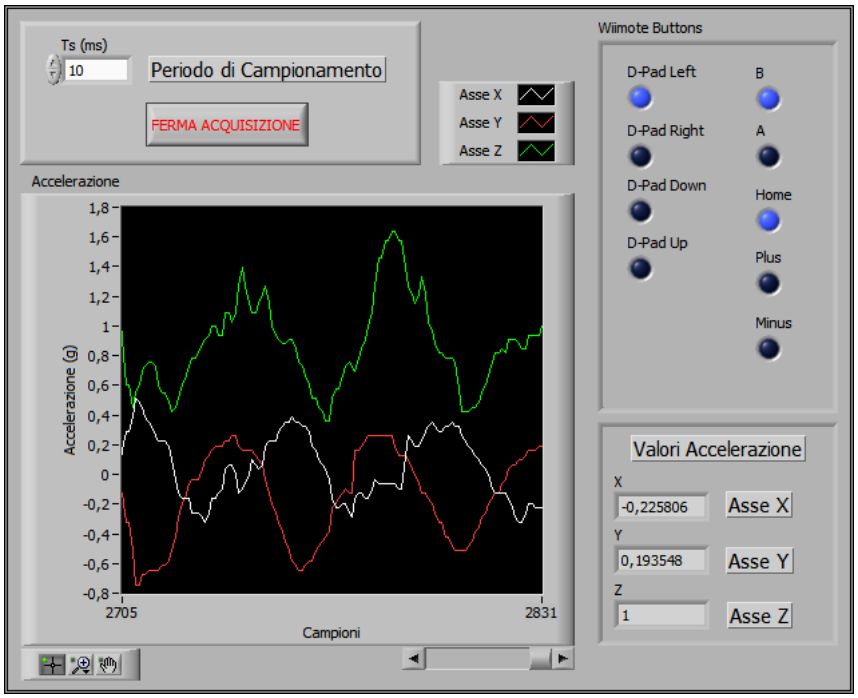


Figura 1.11: VI dimostrativo per la lettura dei sensori del Wiimote. Si notino il grafico che rappresenta i valori di accelerazione inviati dal dispositivo ed i led che segnalano la pressione dei pulsanti.

Il Wiimote è un dispositivo economico che al suo interno contiene già i sensori necessari per condurre test fisici di vario tipo (studi ed esperimenti sull'accelerazione, simulazioni, sviluppo software...). L'acquisto del singolo accelerometro richiederebbe invece anche la costruzione di un adeguato circuito dotato di microcontrollore per la sua lettura e l'invio dei dati al calcolatore.

1.4.3 Applicazioni del Wiimote

Gli accelerometri sono entrati nella vita di tutti i giorni, trovano spazio nei moderni cellulari per l'orientazione automatica del display quando il dispositivo viene inclinato, nelle moderne automobili per la rilevazione dell'accelerazione laterale del veicolo, allo scopo di controllare le sbandate (azionando opportunamente il sistema di frenatura) e nel mondo delle console per videogiochi, come l'oggetto di cui si parla in questo scritto.
Molto particolare e degno di nota è il lavoro che i professori Maurizio Vannoni dell'Istituto Nazionale di Ottica Applicata, e Samuele Straulino (Università di Firenze) stanno facendo, utilizzando il Wiimote nella sua interezza per condurre esperimenti fisici, principalmente per uso didattico. Il progetto a cui stanno lavorando, chiamato *OpenLab Project*, mira infatti ad utilizzare sensori integrati in oggetti comuni (dai mouse ottici al controller oggetto di questa tesi) come strumenti di misura per apprendere la fisica e familiarizzare con argomenti quali l'acquisizione, l'elaborazione dei dati di misura, la simulazione eccetera (vedi [18]):

> «Un esempio di applicazione nel mondo reale è il *Crash Test*. Per misurare ciò che avviene nel momento dell'impatto vengono usati degli accelerometri analoghi a quelli contenuti nel Wiimote. Per esempio è possibile simulare ed analizzare ciò che avviene in un Crash Test utilizzando il controller: viene inserito in un'opportuna "scatola" riempita di schiuma e fatto scontrare con una parete oppure fatto cadere a terra dall'altezza di alcuni metri. La schiuma serve per evitare che il dispositivo si danneggi nello schianto.»

I due docenti hanno inoltre realizzato un pendolo (Figura 1.12) utilizzando il Wiimote: il controller viene usato come massa del pendolo, il passaggio per l'asse verticale è rilevato dalla IR Camera (puntata verso il basso) grazie ad un led fissato sul pavimento, l'effetto della forza di gravità è misurata dall'accelerometro a 3 assi. Tutti i dati sono inviati al calcolatore dove vengono raccolti ed esaminati. Tale struttura viene usata nelle lezioni di laboratorio con gli studenti per studiare il moto del pendolo (si veda [17]).

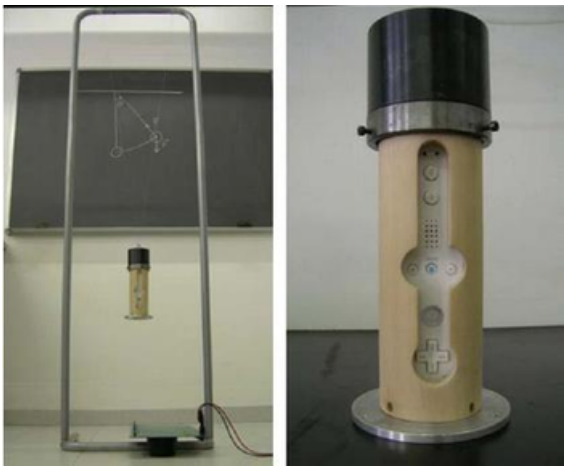


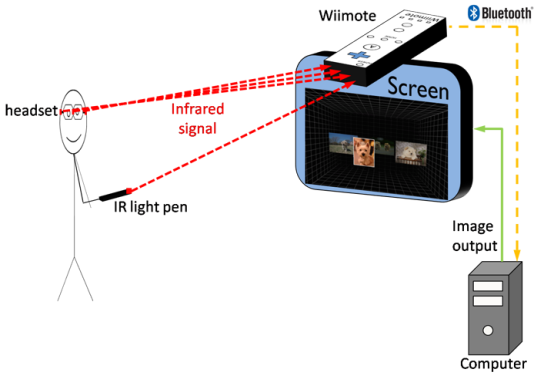
Figura 1.12: Il pendolo realizzato dai docenti Vannoni e Straulino [17]

Altre applicazioni che sfruttano il Wiimote sono sono state considerate da Johnny Chung Lee (figura 1.13a). Ricercatore nel Reparto Scienze Applicate presso la Microsoft, Lee si occupa dello studio delle interfacce Uomo-Macchina per la comunicazione tra Persona e Computer. Il controller nipponico è stato per lui fonte di idee innovative:

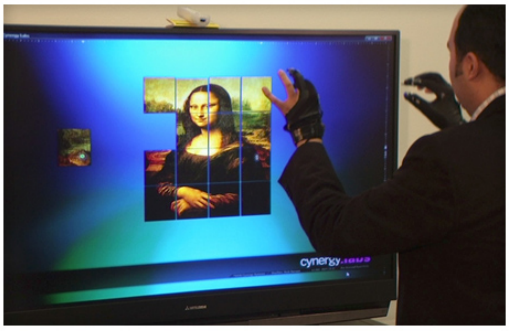
- Tracciare le dita della mano (Finger-Tracking) grazie alla IR Camera, per poter controllare un computer con le dita (prendendo ispirazione dal noto film di Spilberg, Minority Report, Figura 1.13d)
- Tracciare la posizione degli occhi (Eye-Tracking) grazie ad un paio di occhiali equipaggiati di led infrarossi, questo viene usato per modificare le immagini sullo schermo a seconda della distanza da esso e dalla posizione della testa (vedere schema Figura 1.13b).
- Realizzare una Lavagna Interattiva (WhiteBoard) grazie all'uso della telecamera IR e di una penna modificata a cui è stato messo un led al posto della punta (vedere schema Figura 1.13b). In tal modo è possibile scrivere sullo schermo (monitor oppure videoproiettore)

Per ulteriori approfondimenti sul lavoro svolto da Lee si veda [19]. I progetti elencati usano il Wiimote in posizione fissa mentre a muoversi sono i led infrarossi. In figura 1.13c è mostrato un altro progetto che sfrutta il Wiimote per rendere più interattivo l'uso di un computer (si noti la somiglianza con la 1.13d).

(a) Johnny Lee mentre parla delle sue creazioni [19]



(b) Schema funzionamento Eye-Tracking + WhiteBoard



(c) Project Maestro [20], prodotto dal reparto R&D di Cynergy Labs, fa uso del Wiimote

(d) Confronto con un'immagine tratta dal film fantascientifico "Minority Report"

Figura 1.13: Il Wiimote ispira nuove idee: Johnny Lee e Cynergy Labs rendono la fantascienza realtà

Capitolo 2

INTRODUZIONE AGLI ACCELEROMETRI

Nella maggior parte degli accelerometri [6] il principio di funzionamento è lo stesso: una piccola massa (detta massa di prova, *Proof Mass*, o massa campione) viene tenuta sospesa grazie ad un elemento elastico ed in qualche modo ne viene rilevato lo spostamento quando una forza agisce su di essa. In presenza di un'accelerazione la massa si sposta dalla sua posizione di riposo di una quantità proporzionale all'accelerazione rilevata, la variazione della sua posizione viene convertita in un adeguato segnale elettrico contenente l'informazione relativa alla grandezza fisica misurata. Il sistema è dotato di un elemento smorzante per ridurre eventuali oscillazioni della massa.
Quello che si ottiene è un "classico" della meccanica noto come sistema Massa-Molla-Smorzatore (MMS). Per approfondire si veda la [13].

2.1 Teoria: Sistema Massa-Molla-Smorzatore

Si consideri un sistema MMS formato da una *Proof Mass* di massa m, una molla di costante elastica k, che costituisce l'elemento elastico a cui è applicata la massa, ed uno smorzatore con costante di attrito (viscoso) b, tale attrito normalmente è dovuto all'aria o altro fluido. Alla massa viene applicata una forza $F_{ext} = ma$, dove a è l'accelerazione subita dalla massa. In Figura 2.1 è mostrato lo schema meccanico del sistema appena descritto, che in ogni caso rimane una approssimazione rispetto al funzionamento reale di un accelerometro.

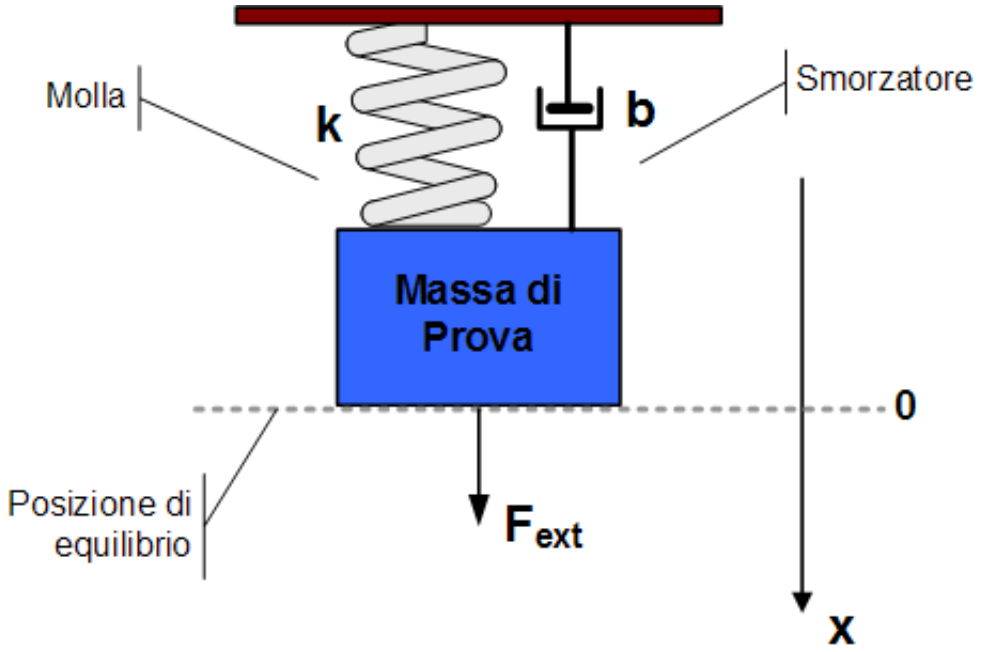


Figura 2.1: Sistema Massa-Molla-Smorzatore di un Accelerometro

Il comportamento del sistema risulta essere del secondo ordine. L'equazione che mette in relazione la forza F_{ext}, e quindi l'accelerazione a applicata alla massa m, con lo scostamento x dalla posizione di equilibrio della stessa è la seguente:

$$m\frac{\partial^2 x}{\partial t} + b\frac{\partial x}{\partial t} + kx = F_{ext} = ma \tag{2.1.1}$$

Dove k è la costante elastica della molla e b è la costante di attrito dello smorzatore. Dividendo tutto per m si ottiene la seguente espressione:

$$\frac{\partial^2 x}{\partial t} + \frac{\omega_n}{Q}\frac{\partial x}{\partial t} + \omega_n^2 = a \tag{2.1.2}$$

Dove sono stati definiti i parametri:

$$\omega_n = \sqrt{\frac{k}{m}} \qquad \text{Pulsazione di risonanza}$$

$$Q = \frac{\omega_n m}{b} \qquad \text{Fattore di qualità}$$

Valori tipici per un accelerometro MEMS (*Micro Electro Mechanical System*) sono sono ad esempio $50\mu g < m < 150\mu g$, $0.5\frac{N}{m} < k < 2.5\frac{N}{m}$ e quindi frequenze di risonanza ($f_n = \frac{\omega_n}{2\pi}$) che variano da qualche centinaia di Hertz fino a poco più di $1kHz$.
La funzione di trasferimento del sistema, mediante la quale è possibile dedurne il comportamento si ottiene passando al dominio di Laplace:

$$\frac{x(s)}{a(s)} = \frac{1}{s^2 + \frac{b}{m}s + \frac{k}{m}} = \frac{1}{s^2 + \frac{\omega_n}{Q}s + \omega_n^2} \tag{2.1.3}$$

Dove $x(s)$ ed $a(s)$ sono rispettivamente le trasformate di Laplace dello scostamento $x(t)$ e dell'accelerazione $a(t)$, variabili nel tempo. In condizioni statiche ($s = 0$) si ottiene la seguente relazione tra accelerazione e scostamento:

$$\frac{x_{static}}{a} = \frac{m}{k} = \frac{1}{\omega_n^2} \quad \Rightarrow \quad X_{static} = \frac{m}{k} \cdot a \tag{2.1.4}$$

Dove la costante m/k è definita sensibilità del trasduttore (da accelerazione a posizione):

$$S = \frac{m}{k} = cost \tag{2.1.5}$$

Valori tipici per la sensibilità sono $20\frac{\mu g \cdot m}{N} < S < 300\frac{\mu g \cdot m}{N}$. La transcaratteristica del trasduttore è rappresentata in figura 2.2, nel caso particolare di $S = 150\frac{\mu g \cdot m}{N}$.

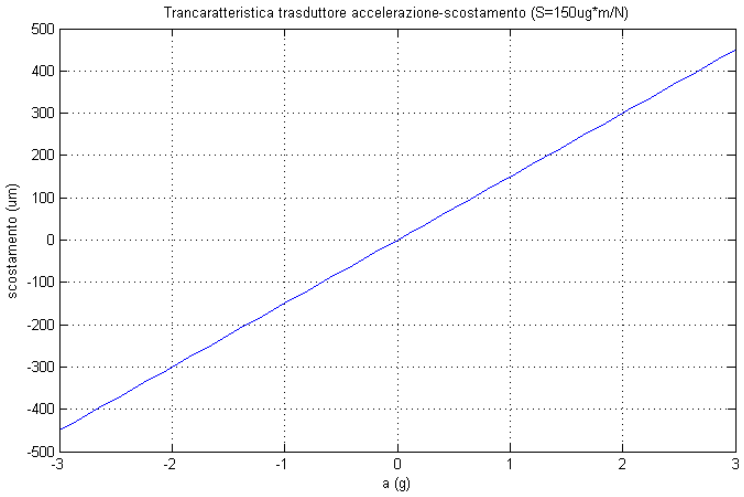


Figura 2.2: Transcaratteristica trasduttore accelerazione-scostamento (grafico puramente a scopo illustrativo)

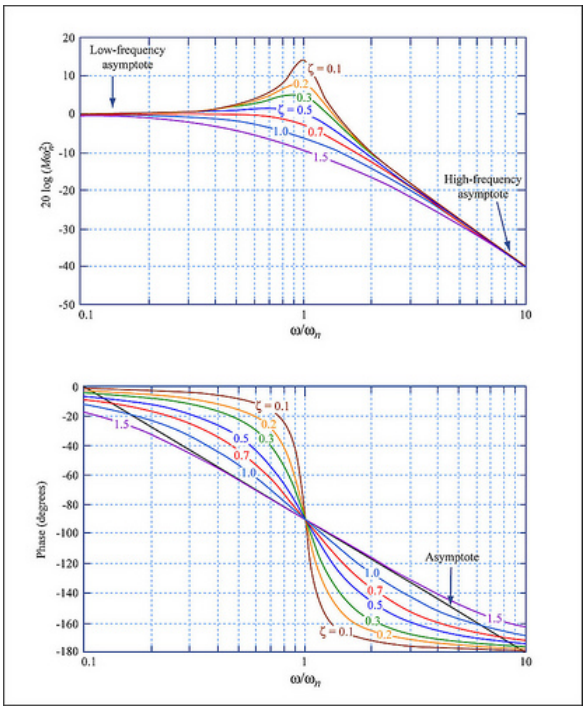
Il comportamento dinamico del sistema è descrivibile mediante il diagramma di Bode di Figura 2.3a, che riporta nel dominio della frequenza ($s = j2\pi f$) la funzione di trasferimento 2.1.3 al variare del parametro ξ (coefficiente di smorzamento).
Ricordando che per un generico sistema del secondo ordine vale:

$$\frac{Y(s)}{X(s)} = \frac{1}{s^2 + 2\xi\omega_n s + \omega_n^2} \tag{2.1.6}$$

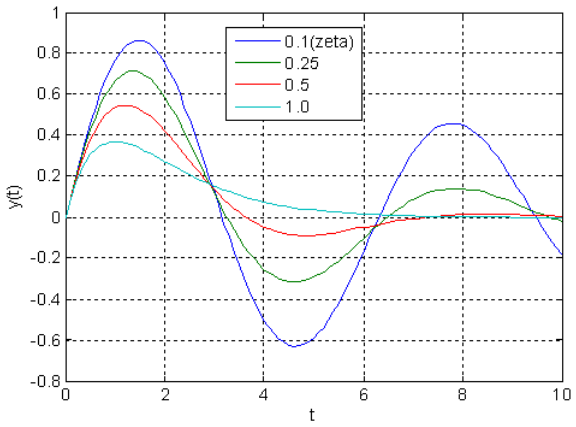
Dove $X(s)$ è l'ingresso del sistema mentre $Y(s)$ è l'uscita del sistema. Confrontando con l'espressione 2.1.3 si può ricavare la relazione:

$$\xi = \frac{b}{2\sqrt{m \cdot k}} = \frac{1}{2Q} \tag{2.1.7}$$

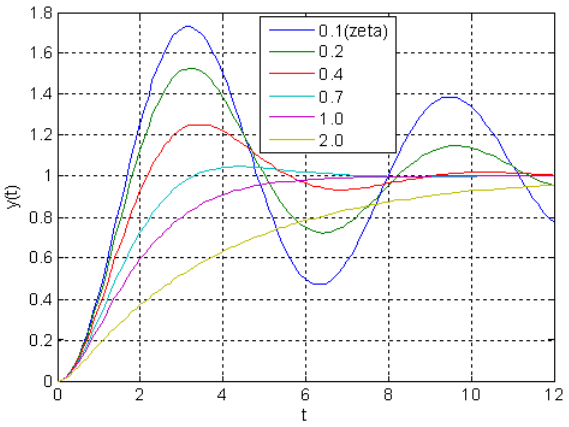
In figura 2.3b invece è riportata la risposta all'impulso, sempre al variare di ξ, tale risposta (dominio del tempo) mostra come il sistema reagisce, oscillando, quando viene sottoposto ad una variazione brusca, impulsiva dell'ingresso, come ad esempio la l'applicazione per un brevissimo tempo di una forza. In figura 2.3c invece è rappresentata la risposta al gradino: al sistema viene applicata una forza costante e tale forza viene mantenuta lasciando il tempo al sistema di portarsi a regime. I grafici proposti sono a scopo illustrativo e non si riferiscono al caso particolare di un accelerometro.



(a) Diagramma Di Bode



(b) Risposta all'impulso



(c) Risposta al gradino

Figura 2.3: Caratteristiche dinamiche e risposte ai segnali canonici

Caratteristica voluta per l'accelerometro è avere ξ prossimo ad $1/\sqrt{2}$ (sistema criticamente smorzato), in questo modo si massimizza la banda del sistema e si garantisce la più piccola distorsione di ampiezza possibile, evitando allo stesso modo fenomeni oscillatori indesiderati nelle risposte agli stimoli esterni (impulsi, gradini). Questo si traduce nelle seguenti condizioni:

$$\xi = \frac{1}{\sqrt{2}} \Leftrightarrow Q = 2 \cdot \sqrt{2} \Leftrightarrow b = \sqrt{2km} \tag{2.1.8}$$

Nella 2.1.5 si è definita la sensibilità del trasduttore, si vorrebbe che tale sensibilità sia elevata, questo richiede una massa grande, allo stesso tempo si vorrebbe una massa piccola in modo da garantire che sia soddisfatta la condizione di smorzamento critico 2.1.8, compatibilmente con b.
Anche la pulsazione di risonanza, che determina la banda del sistema è influenzata dalla massa, per garantire una banda larga è necessaria una massa piccola.

Tutti i parametri in gioco sono inoltre fortemente influenzati dalla tecnologia costruttiva e dal processo di fabbricazione adottato per il sensore.
Un altro importante parametro per un accelerometro è la minima accelerazione rilevabile, il sistema meccanico presenta un rumore dovuto alla presenza dello smorzatore, la minima accelerazione rilevabile è quindi pari proprio al rumore meccanico del sistema MMS descritto. L'influenza del rumore sul sistema può essere calcolato e simulato con appositi software in modo rapido, al variare dei diversi parametri.
Per ora è stata presentata un'analisi meccanica del funzionamento di un sensore di accelerazione, l'accelerazione viene convertita in una differenza di posizione. Verranno analizzate ora le principali tipologie di accelerometri, soffermandosi su quelli di tipo capacitivo, dato che il sensore contenuto nel Wiimote è un sensore MEMS (*Micro Electro Mechanical System*) di tipo capacitivo.

2.2 Classificazione e tipologie di Accelerometri

Quasi tutti i tipi di accelerometri utilizzano il principio fisico appena discusso, si diversificano tuttavia per il modo in cui dalla misura dello scostamento della massa di prova, viene generato un segnale elettrico proporzionale all'accelerazione.
Si possono dividere gli accelerometri in due grandi categorie:

- **Accelerometri per misure di accelerazione statica**: sono in grado di rilevare accelerazioni continue e statiche (frequenza 0Hz), come ad esempio l'accelerazione di gravità. Di solito la banda di questo tipo di sensori non supera i 500Hz, possono inoltre essere usati come *Tilt Sensors* (sensori di inclinazione). A questa categoria appartiene il sensore capacitivo ADXL330.

- **Accelerometri per misure di accelerazione dinamica**: non sono in grado di rilevare accelerazioni statiche, ma solo accelerazioni che variano nel tempo, come ad esempio quelle generate da urti o vibrazioni. La banda di questi sensori è passante e compresa tra qualche decina di Hz fino alle decine di kHz. Sensori di questo tipo sono ad esempio quelli realizzati con tecnologia piezoelettrica.

Gli accelerometri possono inoltre essere classificati in base alla loro sensibilità, al range di misura $\pm N \cdot g$ ed all'applicazione per cui sono destinati. Qui di seguito è proposta una classificazione in base al meccanismo di trasduzione (tipi di accelerometri di uso più comune [6]), viene tralasciato l'accelerometro capacitivo che verrà discusso a parte:

- **Accelerometro Estensimetrico** (Resistivo): come principio sfrutta lo stesso di un'estensimetro, la massa viene connessa fisicamente a degli estensimetri, lo scostamento della massa dalla posizione di equilibrio causa un allungamento del sensore estensimetrico, quindi una variazione della sua resistenza. I sensori sono caratterizzati da una bassa sensibilità, ridotta accuratezza e risultano fortemente influenzati dalla temperatura. Il loro campo d'impiego si limita alla misura di accelerazioni statiche.

- **Accelerometro LVDT** (Linear Variable Differential Transformer): la massa costituisce il nucleo ferromagnetico del sensore LVDT, intorno al nucleo vi sono delle bobine destinate alla rilevazione della posizione della massa, un circuito elettrico apposito si occupa di generare un adeguato segnale elettrico. Questo sensore funziona anche per misurare accelerazioni statiche.

- **Accelerometro piezoelettrico**: come principio per la rilevazione dello spostamento della massa sfrutta le proprietà di un cristallo piezoelettrico, il quale genera un segnale elettrico quando

è sottoposto a compressione. La massa viene sospesa sul cristallo, che viene usato anche come elemento elastico. In presenza di una accelerazione il cristallo si comprime e genera un segnale elettrico proporzionale alla compressione. I sensori non possono essere usati per misurare accelerazioni statiche, infatti il quarzo genera un segnale solo al momento della compressione (variazioni di accelerazione), mentre il segnale è nullo se l'accelerazione applicata alla massa interna risulta costante. Le costati elastiche dei cristalli impiegati non sono elevate, quindi sensori di questo tipo non sono adatti a rilevare lievi accelerazioni. Presentano una sensibilità piuttosto bassa, hanno però un range molto elevato (svariate centinaia di g) rendendoli utili alla misura di grossi shock. L'accelerometro piezoelettrico appartiene alla categoria dei sensori attivi.

- **Accelerometro Termico**: la massa è costituita da un gas che viene riscaldato da un elemento riscaldante. Il gas è racchiuso in una cavità dotata di sensori termici (termo-pile). L'accelerazione causa una spostamento della bolla di gas caldo (si vedano le leggi sulla convezione del calore) che viene rilevata dai sensori termici. Questi accelerometri, detti MEMSIC, fanno parte della categoria dei "*Micro-Machined Accelerometers*" e presentano molti vantaggi, tra cui la resistenza agli shock e la minor usura nel tempo, dovute al fatto che non sono presenti elementi meccanici mobili all'interno del dispositivo.

- **Accelerometro Laser**: questo tipo di accelerometro è usato per effettuare misure altamente precise, infatti i sensori basati su principi ottici sono altamente immuni al rumore ed hanno un comportamento estremamente lineare. A differenza degli accelerometri discussi in precedenza, l'accelerometro laser non utilizza il principio della massa di prova: tramite un interferometro laser viene misurata istante per istante la distanza dell'oggetto in movimento e tramite un calcolatore ne viene fatta la derivata seconda rispetto al tempo (per misure ancora più accurate viene usato un orologio atomico per il riferimento temporale).

2.3 Accelerometri Capacitivi MEMS

Gli accelerometri capacitivi sfruttano come principio per il rilevamento dello spostamento della massa di prova, la variazione della capacità elettrica di un condensatore, associata alla variazione della distanza tra le sue armature. La massa stessa, realizzata con materiale conduttivo, costituisce un'armatura del condensatore, l'altra, o le altre due (se costruito con tecnologia differenziale), sono invece fisse alla struttura del dispositivo. La massa viene tenuta sospesa grazie ad un elemento elastico, in modo che le armature non si tocchino. Uno schema funzionale dell'accelerometro capacitivo è riportato in figura 2.4.

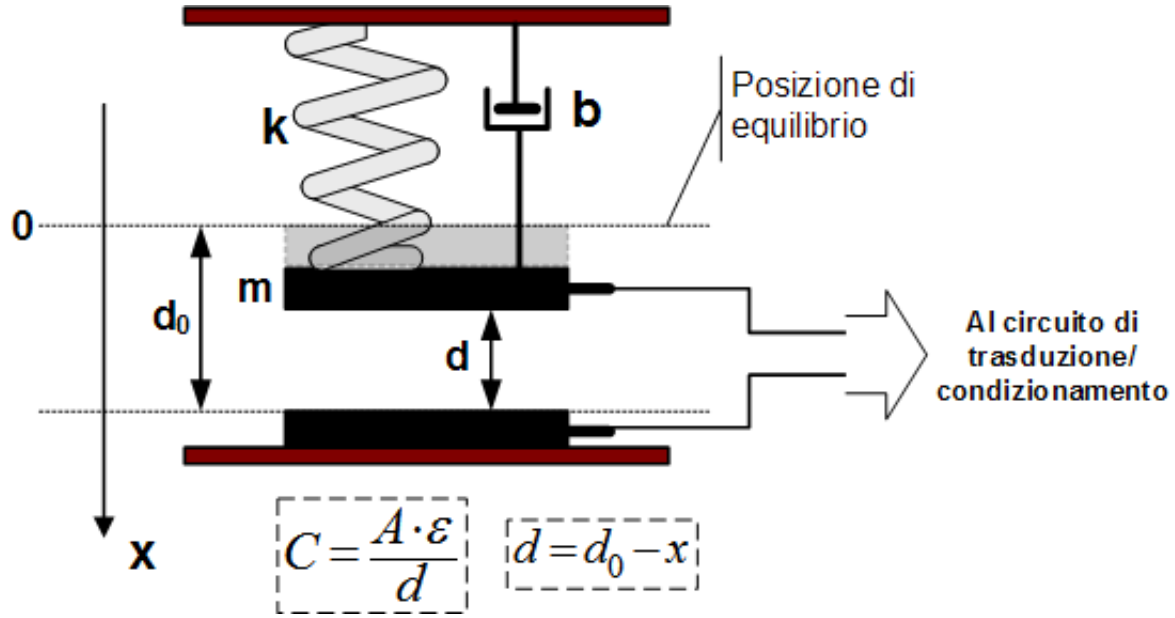
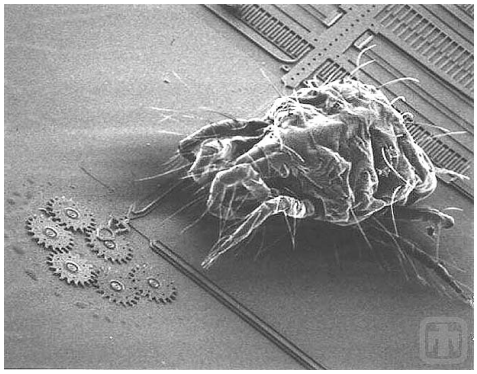
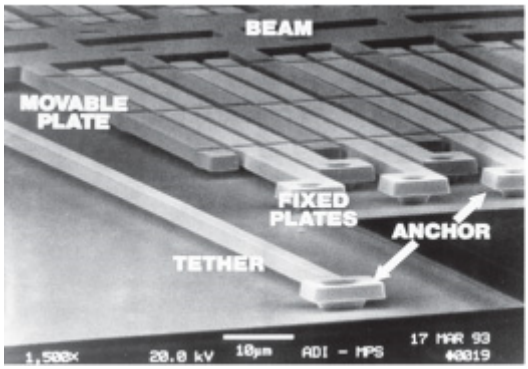


Figura 2.4: Schema funzionale accelerometro capacitivo

La massima deflessione x che può subire la massa è pari alla distanza tra le armature in condizione di equilibrio d_0 (con forze applicate nulle), infatti uno scostamento pari a $x = d_0$ causa il contatto tra l'armatura mobile e quella fissa. Un apposito circuito infine genera un segnale elettrico proporzionale alla capacità del condensatore e quindi all'accelerazione.
Questi sensori sono adatti alla misura di accelerazioni statiche, sono poco sensibili alle variazioni di temperatura, hanno un'elevata sensibilità, alte prestazioni, bassa dissipazione di potenza ed un costo molto basso. La tecnologia capacitiva tuttavia, a differenza di quello che avviene per sensori basati su altre tecnologie, rende questi sensori suscettibili alle interferenze elettromagnetiche; una possibile soluzione consiste nell'utilizzare un'adeguata schermatura.
Gli accelerometri di tipo capacitivo si prestano bene ad essere integrati, assieme al relativo sistema di trasduzione e condizionamento del segnale, in un unico chip di silicio. Si parla in questo caso di dispositivi MEMS (*Micro-Electro-Mechanical-Systems*), chiamati anche "Micro Macchine" [7]. Come dice il nome stesso, sono dispositivi elettro-meccanici le cui dimensioni sono dell'ordine dei micron, realizzati direttamente su un substrato di silicio, che è lo stesso substrato su cui sono realizzati i transistor che compongono il circuito che si occupa della generazione e del condizionamento dei segnali necessari al funzionamento della micro macchina. Nel nostro caso, il micro sistema meccanico è costituito dal sistema Massa-Molla-Smorzatore, che viene realizzato completamente in silicio. La realizzazione è abbastanza semplice, il modo è lo stesso con il quale si realizza un qualsiasi circuito integrato (processo fotolitografico).
In figura 2.5a è mostrato un sistema MEMS le cui dimensioni sono confrontabili con quelle di un acaro (0.5mm). Nella figura 2.5b invece è mostrata la struttura interna di un accelerometro capacitivo realizzato in tecnologia MEMS.



(a) Un micro sistema meccanico confrontato con un acaro



(b) Struttura reale di un accelerometro capacitivo MEMS

Figura 2.5: MEMS: Micro-Electro-Mechanical-Systems

2.3.1 Realizzazione fisica

Nell'accelerometro MEMS [13], come già accennato, la massa costituisce un'armatura di un condensatore, di solito si tratta di un'armatura centrale, compresa tra due armature fisse. Quello che elettricamente si va a realizzare è schematizzabile come due condensatori in serie con una connessione centrale comune corrispondente alla massa mobile (che è conduttiva). In figura 2.6 è rappresentato un modello della struttura interna del sensore, viene evidenziato l'effetto provocato da un'accelerazione applicata al package del sensore: la massa interna si sposta e come si può vedere le capacità dei due condensatori variano. In questa configurazione, in condizioni di assenza di forza applicata, le capacità sono uguali, mentre quando è applicata una forza esse variano, precisamente una aumenta e l'altra diminuisce, in modo complementare (la loro somma è sempre costante).

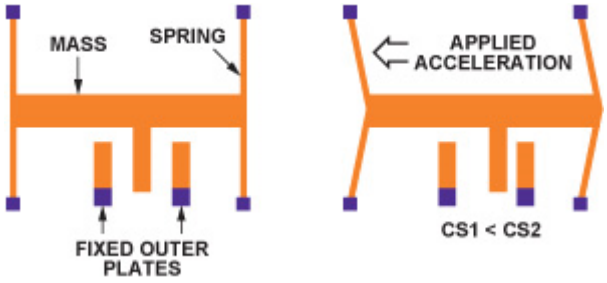


Figura 2.6: Modello struttura Accelerometro capacitivo MEMS di tipo differenziale, funzionamento in assenza ed in presenza di accelerazione

La realizzazione del condensatore gioca un ruolo fondamentale: la distanza tra l'armatura mobile e quella fissa fornisce lo scostamento massimo possibile per la *Proof Mass* e quindi impone un limite sulla massima forza applicabile alla massa, ovvero un limite alla massima accelerazione misurabile. Tale distanza, indicata con d_{max}, è fortemente dipendente dal processo costruttivo del sensore e da limitazioni tecnologiche, non può essere scelta grande a piacere (anzi deve essere più piccola possibile). Ne consegue che gli accelerometri MEMS capacitivi di solito hanno range di misura limitato a pochi g. Si ricorda che l'unità di misura $[g]$ è definita come l'accelerazione di gravità terrestre al livello del mare e corrisponde a circa $9.81 m/s^2$ (unità del Sistema Internazionale). Ad esempio l'accelerometro ADXL330 del Wiimote ha un range (in unità g) che va da $-3g$ a $3g$, ovvero che risulta variare tra $-29.4 m/s^2$ e $29.4 m/s^2$.

La massima accelerazione misurabile (in modulo) è data dalla seguente espressione:

$$a_{max} = \frac{k \cdot d_{max}}{m} \tag{2.3.1}$$

Nel caso di un accelerometro con $S = m/k = 150\frac{\mu g \cdot m}{N}$ e $d_{max} = 7.35\mu m$ si ha un'accelerazione massima misurabile $a_{max} = 49m/s^2 = 5g$.
In un accelerometro capacitivo inoltre si definisce la sensibilità S_C, dovuta alla trasduzione accelerazione-scostamento-variazione di capacità:

$$S_C = \frac{\Delta C}{\Delta a} = \frac{C - C_0}{a} \tag{2.3.2}$$

dove C è la capacità del condensatore quando è applicata l'accelerazione a, mentre C_0 è la capacità del condensatore in condizioni di riposo (assenza di accelerazione), si veda la figura 2.4. Eseguendo la sostituzione:

$$C - C_0 = \frac{A \cdot \varepsilon \cdot x}{d_0 \cdot (d_0 - x)} = \frac{C_0}{d_0 - x} \cdot \frac{m}{k} \cdot a \tag{2.3.3}$$

dove A è l'area della sovrapposizione tra le due armature (fissa e mobile), ε è la permittività elettrica dell'aria, d_0 è la distanza tra le armature in condizioni di riposo (assenza di accelerazione), m è la massa di prova e k è la costante elastica della membrana che sorregge la massa. Si noti che nell'ultimo passaggio è stata applicata la 2.1.4. Si ottiene infine:

$$S_C = \frac{C_0}{d_0 - x} \cdot \frac{m}{k} = \frac{C_0 \cdot m}{k \cdot d} \tag{2.3.4}$$

dove $d = d_0 - x$ è la distanza tra le armature del condensatore nel caso in cui venga applicata un'accelerazione a e quindi avvenga uno scostamento x della massa.
Interessante è capire come vengono calcolati i parametri k (costante della molla) e b (attrito smorzatore) in base alle caratteristiche geometriche ed alle proprietà del materiale (silicio) del sensore:

- Costante k della molla. Tale costante è legata alle caratteristiche dell'elemento elastico ed influisce sulla frequenza di risonanza (e quindi sulla la banda) e sulla sensibilità del sensore. Le caratteristiche da considerare dell'elemento elastico, che nel sensore MEMS è composto da silicio, sono la lunghezza L, lo spessore t, la larghezza W ed il modulo di Young della membrana che sorregge la *Proof Mass*. In figura 2.7 sono riportate le dimensioni geometriche considerate. Se si suppone che k non subisca variazioni durante il sollecitamento del materiale può essere applicata la seguente equazione [13]:

  $$k = \frac{16Wt^3}{L^3}E \tag{2.3.5}$$

  Dove il modulo di Young per il silicio vale $E = 190\,GPa$.

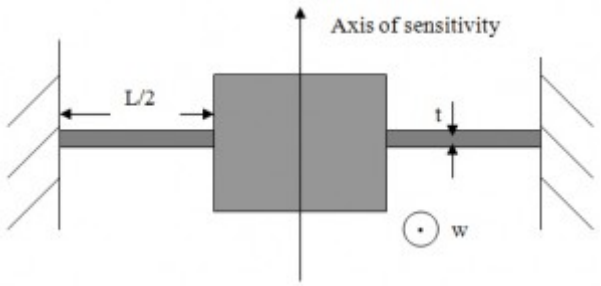


Figura 2.7: Dimensioni Geometriche membrana elastica [13]

- Costante di attrito viscoso b dello smorzatore. Tale costante è legata all'effetto di compressione dello strato d'aria compreso tra la massa mobile e l'elettrodo fisso del sensore, che si verifica quando la massa è sottoposta ad una forza che fa variare la sua posizione. Se è verificata la seguente condizione ("*Squeeze Number*") entro i limiti della banda dell'accelerometro [13]:

$$\sigma = \frac{12\mu A\omega}{(pd)^2} << 1 \tag{2.3.6}$$

  Allora b può essere calcolato grazie alla seguente relazione [13]:

$$b = \frac{0.42\mu A^2}{d^3} \tag{2.3.7}$$

  Dove nelle formule sopra d è la distanza tra la massa mobile e l'elettrodo fisso, $\mu = 1.85 \cdot 10^{-5} N/m^2$ è la viscosità dell'aria, $p = 1.013 \cdot 10^5 Pa$ è la pressione atmosferica, A è l'area dello strato di aria ed ω è la pulsazione di una sollecitazione sinusoidale applicata al sensore.

Come precedentemente detto il sensore viene realizzato direttamente sul wafer di silicio del chip, i materiali usati sono silicio, polisilicio, eventuali elementi metallici, ecc. Gli stessi elementi che si usano per realizzare un comune circuito integrato. Il procedimento che si segue nella realizzazione dei MEMS è quindi analogo: si tratta di un procedimento fotolitografico, il substrato di silicio viene ricoperto con un polimero sensibile alla luce UV (ultravioletta), grazie ad una maschera ottica si espone solo la zona del materiale che si vuole trattare, infine si procede ad un attacco chimico. Ripetendo questa procedura più volte, usando diverse maschere ottiche è possibile realizzare fisicamente la massa e la membrana elastica dell'accelerometro. In Figura 2.8 sono riportati gli step del processo di fabbricazione di cui si è appena parlato.

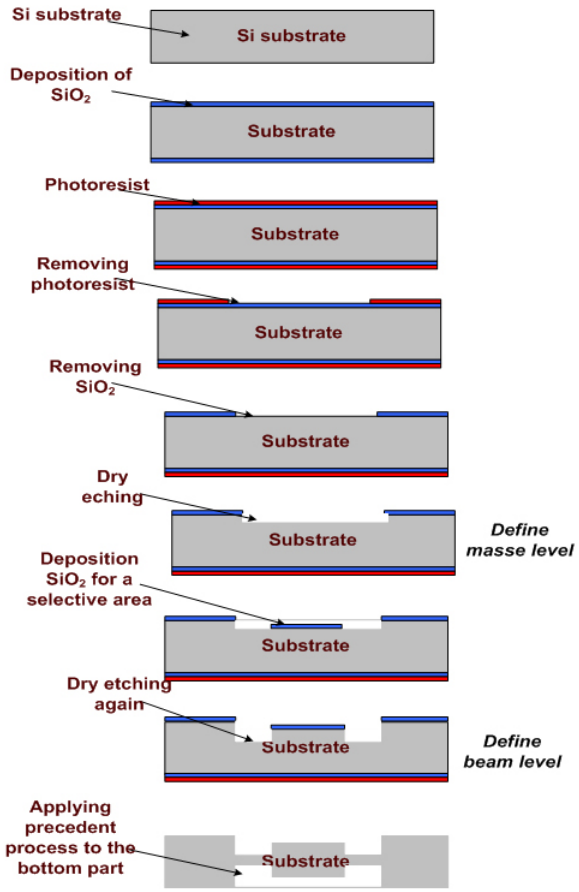


Figura 2.8: Steps processo fabbricazione massa e membrana elastica di un accelerometro MEMS [13]

In Figura 2.9 è rappresentato il risultato finale, complessivo degli elettrodi che costituiscono le armature fisse del condensatore. La figura costituisce solo un modello funzionale, la reale struttura di un accelerometro MEMS capacitivo è già stata vista in figura 2.5b.

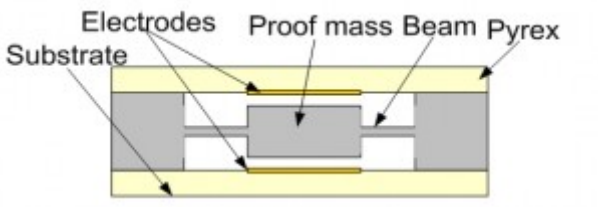


Figura 2.9: Risultato del processo di fabbricazione di un accelerometro a singolo asse [13]

2.3.2 Generazione segnali elettrici: Trasduzione, Condizionamento

Verrà ora discusso il circuito di trasduzione/condizionamento usato per passare dalla variazione di capacità del condensatore ad armatura centrale (condensatore differenziale) ad un segnale in tensione la cui componente continua (DC) è proporzionale all'accelerazione misurata (risulta costante se la forza applicata al sensore è costante). Viene in sostanza utilizzata un'architettura di misura ad anello aperto, schematizzata in Figura 2.10. Si veda [14] per ulteriori approfondimenti.

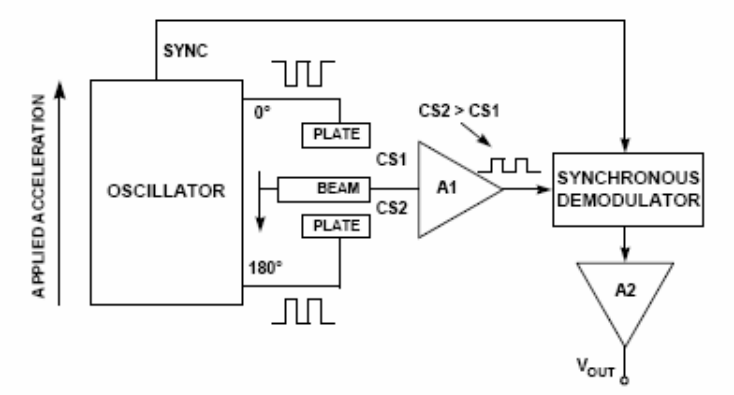


Figura 2.10: Circuito di trasduzione/condizionamento accelerometro capacitivo

Le armature fisse del condensatore differenziale vengono pilotate da un segnale sinusoidale (oppure da un'onda quadra) ad alta frequenza (es. 100kHz) generato da un circuito oscillante: alla prima armatura viene fornito il segnale non sfasato, mentre alla seconda viene fornito lo stesso segnale ma sfasato di 180° (negato). Il segnale in uscita dall'elettrodo centrale viene applicato ad un opportuno amplificatore, in uscita a quest'ultimo si ha un segnale periodico (del tutto analogo a quello generato dall'oscillatore) la cui ampiezza è proporzionale alla capacità differenziale. Successivamente un demodulatore ed un filtro "passa basso" fanno si che il segnale in uscita (disponibile all'utente) contenga l'informazione sul segno dell'accelerazione applicata e sia pulito dalle componenti ad alta frequenza dovute al segnale di pilotaggio.
Il circuito che può essere usato come amplificatore prevede l'uso di un *op-amp* ed è mostrato in figura 2.11 (è stata fatta l'ipotesi di segnale di pilotaggio sinusoidale):

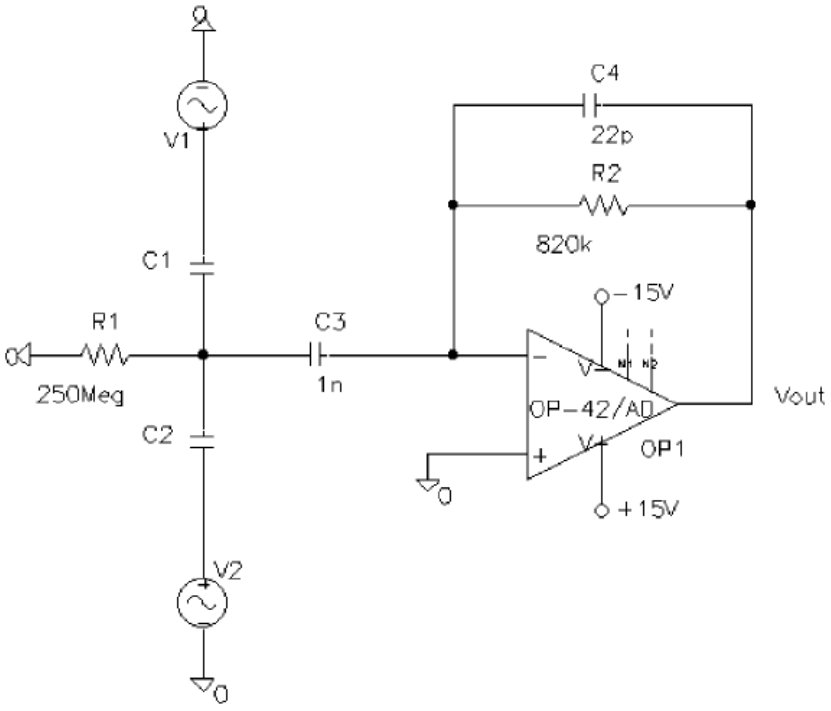


Figura 2.11: Circuito Amplificatore [14]

I segnali V_1 è V_2 sono sfasati di 180°, C_1 e C_2 rappresentano le capacità del condensatore differenziale, C_3 è un condensatore di disacoppiamento, R_1 è una resistenza di *pull-down*, deve essere molto grande, infatti si richiede che la costante di tempo $\tau = C_{1/2} \cdot R_1$ sia almeno 10 volte più grande rispetto al periodo del segnale di pilotaggio. La resistenza R_2 è necessaria per evitare che l'operazionale vada in saturazione (percorso di *feedback*). Assumendo l'operazionale ideale la tensione in uscita (Vout in figura) risulta essere (nel dominio di Laplace):

$$V_{co}(s) = -\frac{sC_3R_2}{1+sC_4R_2} \cdot \frac{sV_1C_1 + sV_2C_2}{sC_1 + sC_2 + sC_3 + 1/R_1} \tag{2.3.8}$$

Ponendo $C_1 = C_0 + \Delta C$, $C_2 = C_0 - \Delta C$ e $V_1 = -V_2$ si può riscrivere tutto come:

$$V_{co}(s) = -\frac{sC_3R_2}{1+sC_4R_2} \cdot \frac{2V_1s\Delta C}{s(C_1 + C_2 + C_3) + 1/R_1} \tag{2.3.9}$$

Se si fanno le seguenti assunzioni (approssimazioni):

$$C_3 >> C_1 + C_2 \tag{2.3.10}$$

$$1/R_1 \to 0 \tag{2.3.11}$$

$$|sC_4R_2| >> 1 \tag{2.3.12}$$

Allora risulta che la tensione in uscita è proporzionale alla capacità differenziale:

$$v_{co}(t) = -\frac{2\Delta C}{C_4} \cdot v_1(t) = -\frac{C_1 - C_2}{C_4} \cdot v_1(t) \tag{2.3.13}$$

Le capacità di C_1 e di C_2 possono essere calcolate grazie alla formula del condensatore a facce piane:

$$C_1 = \varepsilon \frac{A}{d-x} \tag{2.3.14}$$

$$C_2 = \varepsilon \frac{A}{d+x} \tag{2.3.15}$$

dove ε è la costante dielettrica dell'aria, A è l'area delle armature, d rappresenta la distanza tra l'armatura centrale e l'armatura esterna in condizioni di assenza di sollecitazioni (si trova al centro tra le due armature), mentre x rappresenta lo scostamento dalla posizione centrale quando viene applicata una sollecitazione. Si noti che con le convenzioni adottate uno scostamento positivo causa un aumento di C_1 ed una diminuzione di C_2 (la massa si sta muovendo verso l'armatura 1). Risulta quindi:

$$v_{co}(t) = -\frac{C_1 - C_2}{C_4} \cdot v_1(t) = -\frac{2\varepsilon A x}{C_4(d^2 - x^2)} \cdot v_1(t) \simeq -\frac{2\varepsilon A}{C_4 d^2} \cdot x \cdot v_1(t) \tag{2.3.16}$$

L'ultima approssimazione è dovuta all'ipotesi di piccoli scostamenti di posizione, ponendo $d^2 >> x^2$, e come si può notare risulta che la tensione in uscita dall'amplificatore ha un'ampiezza lineare con lo scostamento x della massa di prova.
Il segnale $v_{co}(t)$ entra in ingresso al demodulatore, il quale genera un segnale contenente l'informazione sul segno associato alla deflessione della massa. In uscita dal demodulatore viene messo un filtro "passa basso" del second'ordine per prelevare solo la componente continua del segnale demodulato. Il filtro ha una frequenza di taglio pari circa a 500Hz. Il funzionamento del demodulatore è molto semplice ed è illustrato nella in figura 2.12.

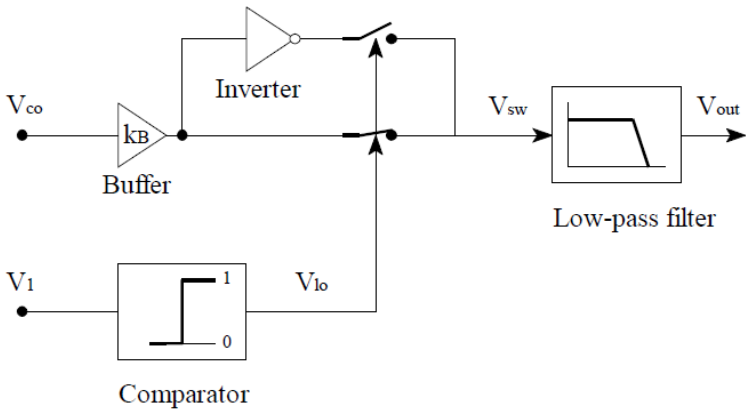


Figura 2.12: Circuito Demodulatore [14]

Il segnale v_1 generato dall'oscillatore viene fornito ad un comparatore che pilota gli switch elettronici (realizzati con dei transistor). Il segnale $v_{sw}(t)$ all'ingresso del filtro risulta essere $v_{sw}(t) = v_{co}(t)$ nel caso in cui $v_1 < 0$, risulta invece essere $v_{sw}(t) = -v_{co}(t)$ nel caso in cui $v_1 > 0$. L'andamento di $v_{sw}(t)$ è riportato in figura 2.13 evidenziando le due condizioni $\Delta C = \frac{C_1 - C_2}{2} > 0$ (scostamento x positivo) e $\Delta C = \frac{C_1 - C_2}{2} < 0$ (scostamento x negativo).

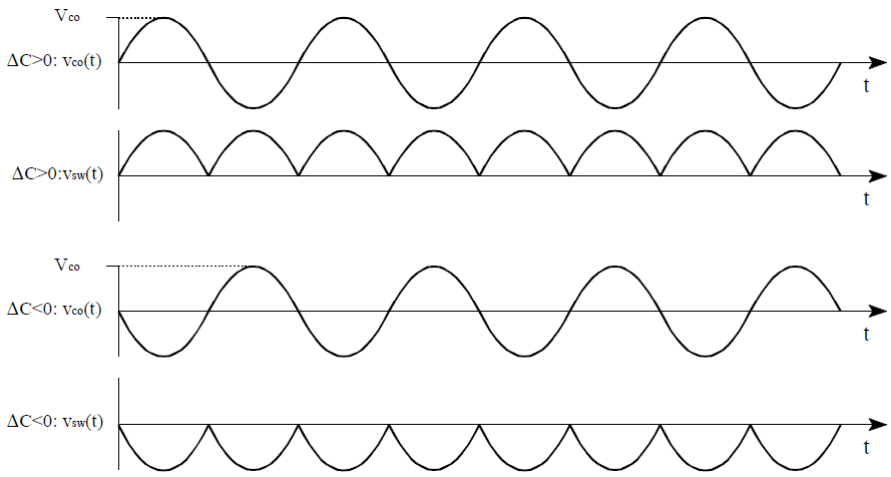


Figura 2.13: Segnale in uscita al demodulatore nei casi $\Delta C > 0$ sopra e $\Delta C < 0$ sotto [14]

2.4 Parametri Caratteristici di un Accelerometro

Qui di seguito verranno discussi i parametri tipici che caratterizzano un accelerometro integrato analogico. I parametri che si esamineranno vengono riportati nei datasheet rilasciati dai produttori per caratterizzare il sensore e relative performance (si vedano [16] e [15]).

- **Range di misura** (*measurement range*): definisce il range di accelerazioni misurabili con il sensore per cui si garantisce un segnale in uscita lineare. Di norma è espresso usando l'unità di misura g ed il range è simmetrico, del tipo $\pm 3g$, la massima accelerazione misurabile, in modulo, è strettamente collegata alla massima deflessione che può avere la massa interna al dispositivo. Di solito comunque ci si riferisce alla massima accelerazione misurabile senza distorsioni dovute alla non linearità, tale accelerazione è pari all'estremo superiore del range di misura, ad esempio $3g$ (la deflessione della massa non è massima). Oltre tale valore si ha un errore di linearità (figura 2.14).
  Bisogna osservare anche che la massima accelerazione misurabile non ha niente a che fare con la massima accelerazione sopportabile dal sensore: quest'ultima infatti si riferisce allo stress oltre il quale si ha la rottura fisica del dispositivo, e di norma viene indicata in una tabella chiamata "*Absolute Maximum Ratings*".

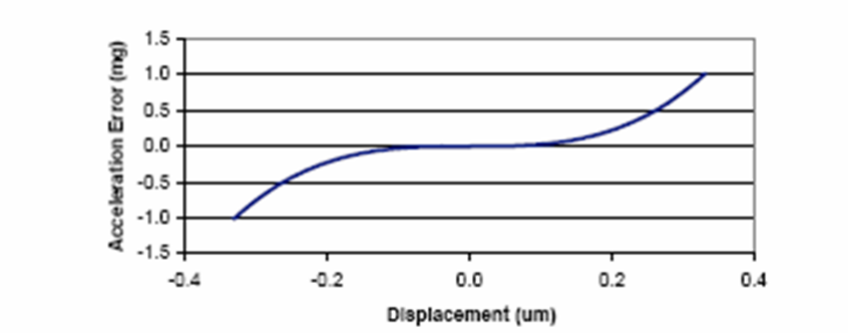


Figura 2.14: Scostamento dalla linearità oltre il range di misura. L'asse delle x rappresenta lo scostamento della massa dalla posizione centrale. Il grafico ha scopo puramente illustrativo, i valori riportati sono solo indicativi.

- **Fattore di scala** (sensibilità o *sensitivity*): è definita come il rapporto tra la variazione del segnale in uscita e la relativa variazione di accelerazione in ingresso. Viene espressa di norma in mV/g, definisce il fattore di scala della retta ideale che costituisce la transcaratteristica ingresso-uscita del dispositivo:

$$Sensitivity = \frac{\Delta V_{out}}{\Delta g} \tag{2.4.1}$$

La sensibilità può essere calcolata, nel caso particolare in cui si conoscano i valori di tensione in uscita corrispondenti alle accelerazioni $-g$ e g, nel seguente modo:

$$Sensitivity = \frac{V_{out,+1g} - V_{out,-1g}}{2g} \tag{2.4.2}$$

Le tensioni $V_{out,+1g}$ e $V_{out,-1g}$ possono essere misurate posizionando l'accelerometro con l'asse di riferimento orientato rispettivamente verso l'alto e verso il basso lungo la direzione sulla quale agisce l'accelerazione gravitazionale.
La sensibilità di solito è data compresa tra un valore minimo ed uno massimo. La tensione di alimentazione di solito causa una variazione della sensibilità (nei datasheet è riportata la scritta "*Ratiometric*", per indicare la dipendenza dalla tensione di alimentazione), ad esempio un raddoppio della tensione causa un raddoppio della sensibilità, viceversa un dimezzamento. Come ultima cosa va osservato che anche la temperatura influisce sulla sensibilità, lo scostamento dal valore nominale viene espresso in $\%/°C$.

- **Non-linearità** (*Nonlinearity*): definisce lo scostamento dalla linearità della caratteristica ideale (retta) individuata dal *Fattore di scala* e viene fornita come percentuale rispetto al fondo scala, oppure rispetto al range di fondo scala. Negli accelerometri MEMS assume valori molto bassi (ad esempio per l'ADXL330 è il 0.3% del fondo scala) ed in molte applicazioni la sua influenza può essere trascurata. In Figura 2.15 è riportato un grafico che mostra un esempio di scostamento dalla linearità per un accelerometro (la figura ha scopo puramente illustrativo): la linea grigia indica la caratteristica ideale, quella nera un possibile andamento reale (non lineare).

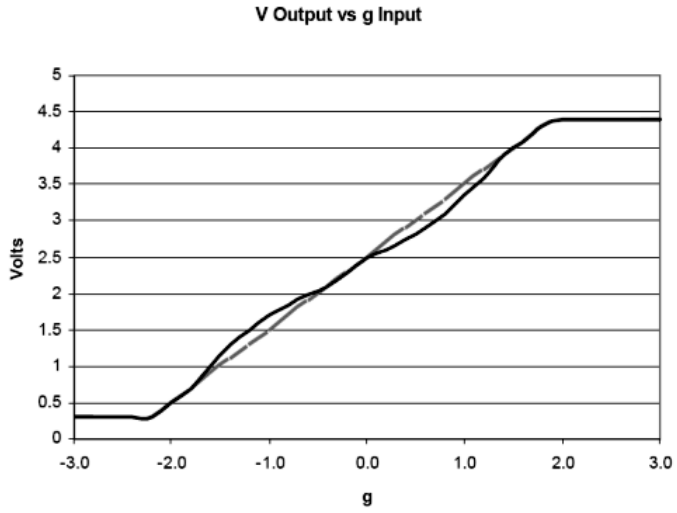


Figura 2.15: Transcaratteristica e scostamento dalla linearità [16]

- **Banda** (o larghezza di banda, *Bandwidth*): definisce la banda del segnale in uscita, associata alla banda delle sollecitazioni applicate al sensore. Rappresenta la massima frequenza che

può avere il segnale in uscita, quindi la massima frequenza con cui può variare l'accelerazione in ingresso affinché la variazione sia rilevabile. Di solito questo valore è compreso tra le centinaia e le migliaia di Hz e può essere limitato in uscita con l'aggiunta di condensatori di filtraggio. A seconda dell'applicazione per cui il sensore è destinato, si possono voler tagliare le frequenze che non interessano, ad esempio nella rilevazione di vibrazioni può essere richiesta una banda elevata mentre per misurare l'accelerazione di un veicolo normalmente si preferisce bassa. La dinamica indotta in uscita influisce anche sulla progettazione del blocco di conversione analogico-digitale (A/D): in base al teorema di Nyquist la frequenza di campionamento dell'ADC (*Analog to Digital Converter*) dovrà essere almeno 2 volte tale valore.
La banda del sensore è determinata dal circuito di condizionamento interno del sensore e dal sistema meccanico Massa-Molla-Smorzatore. La frequenza di risonanza, è invece ampiamente maggiore della banda di utilizzo (di solito diversi kHz).

- **Offset in uscita a 0g** (*Zero-g bias level*): specifica la tensione in uscita quando non è applicata accelerazione (0g), viene espressa in V o mV ed è riferita ad una particolare tensione di alimentazione (*ratiometric*). Viene normalmente anche indicata la deviazione massima da tale valore (che è ideale), lo scostamento dovuto alla temperatura (in $mg/°C$), la sensibilità alla variazione con la tensione di alimentazione, o comunque altri parametri o diagrammi che riassumono le variazioni dovute alla non idealità del dispositivo.

- **Tensione di alimentazione** (*Supply voltage*): definisce il range delle possibili tensioni di alimentazione del sensore, molti parametri del sensore dipendono dalla tensione di alimentazione usata (*ratiometric*).

- **Errore di allineamento del package** (*Package alignment error*): definisce l'angolo tra il piano del *package* (riferimento assi esterno) ed il *die* (riferimento assi interni), figura 2.16. L'errore è dovuto al processo tecnologico del *packaging*, che non riesce ad allineare il *die* perfettamente con l'involucro plastico, di norma lo scostamento è di circa 1°. Nel caso di accelerometri con più assi è definito anche l'errore di allineamento tra un asse e l'altro. L'impatto dell'errore di allineamento im molte applicazioni può comunque trascurato.

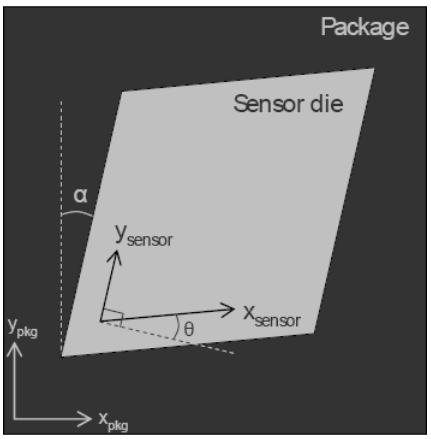


Figura 2.16: Allineamento tra Package e Die [16]

- **Sensibilità asse-asse** (*Cross-axis sensitivity*): specifica l'entità del segnale che compare sulle uscite corrispondenti agli assi ortogonali all'asse a cui è applicata l'accelerazione, ovvero la percentuale di accoppiamento tra 2 assi (in un accelerometro dotato di più assi). Applicando

un'accelerazione lungo l'asse z, le uscite degli assi x ed y subiscono anch'essi una lieve variazione. Questo è dovuto agli errori di allineamento tra assi, processo tecnologico del sensore, circuito di condizionamento interno.

- **Rumore** (*Noise density*): la potenza totale di rumore RMS (*root-medium-square*) come la deviazione del segnale in uscita rispetto al caso ideale, come riportato nei datasheet (ad esempio [15]), si può calcolare dalla densità di rumore, conoscendo la banda in uscita:

$$Total\,Rms\,Noise = Noise\,Density \cdot \sqrt{Bandwidth \cdot 1.6} \tag{2.4.3}$$

La banda può essere limitata grazie ai condensatori in uscita, imponendo la minima frequenza richiesta per l'applicazione che si vuole realizzare si riduce al minimo il rumore e si massimizza quindi la risoluzione dell'accelerometro.

Capitolo 3

ACCELEROMETRO SOTTO ESAME: ADXL330

In questa sezione verrà analizzato l'accelerometro presente nel controller Wiimote, come già detto, si tratta di un accelerometro a tre assi realizzato con tecnologia MEMS, di tipo capacitivo, con uscite in tensione analogiche e proporzionali all'accelerazione misurata. Quanto detto nel precedente capitolo ha permesso di definire tutte le caratteristiche di un accelerometro, quindi l'analisi dell'integrato ADXL330 sarà più semplice. Come prima cosa si analizza il datasheet del produttore [15] e si riportano le caratteristiche del sensore, successivamente verrà invece analizzata la conversione analogico-digitale [1] dei segnali del dispositivo: i dati vengono poi elaborati in un mondo digitale, per esempio da un microcontrollore.

3.1 Analisi del Datasheet

L'ADXL330 è un accelerometro a tre assi a basso consumo di potenza, il circuito di condizionamento è interno al dispositivo, per cui le uscite risultano già proporzionali all'accelerazione misurata. Le accelerazioni misurate appartengono ad un range di $\pm 3g$ (*full-scale*), è possibile misurare l'accelerazione statica dovuta alla gravità e quindi può essere usato come sensore di inclinazione, ovviamente misura anche l'accelerazione dinamica associata a movimenti, urti, vibrazioni. Le principali applicazioni per questo sensore vanno dai dispositivi mobili (cellulari, lettori mp3...), alle console per giochi (come nel nostro caso), alla protezione per gli Hard Disk (in caso di urti), alla stabilizzazione delle immagini, ai dispositivi medici.
In figura 3.1 viene riportato il blocco funzionale dell'accelerometro, che ne descrive la struttura interna.

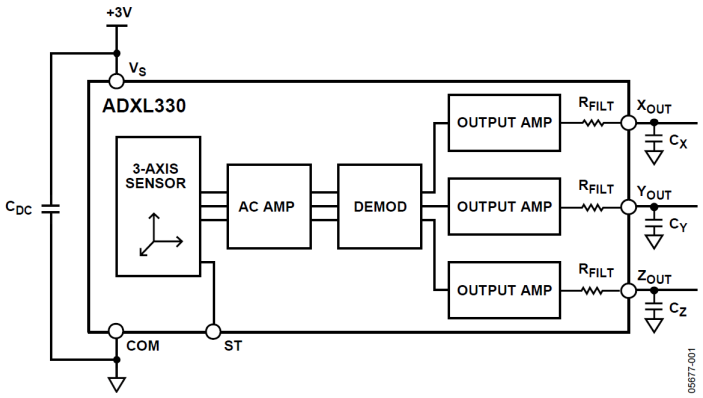


Figura 3.1: ADXL330 Blocco funzionale [15]

Essendo un accelerometro a 3 assi, si ha un elemento sensibile per ogni asse di riferimento. I tre assi x, y e z sono ortogonali tra loro, quindi si può misurare un'accelerazione proveniente da qualsiasi direzione. Ogni elemento sensibile è seguito da una catena di circuiti necessari alla trasduzione ed al condizionamento del segnale, la cui struttura è stata ampiamente discussa nel precedente capitolo. In figura 3.2 è raffigurato uno schema che riassume le operazioni eseguite sul singolo canale (Nell'ADXL330 il segnale generato dall'oscillatore interno è un'onda quadra a 50kHz).

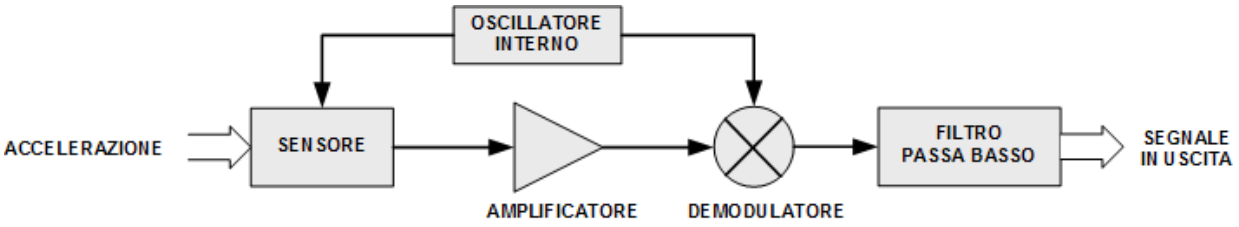


Figura 3.2: Trasduzione e condizionamento per singolo asse

Le resistenze in uscita dagli amplificatori di uscita ("*output amp*") schematizzati Figura 3.1 sono fissate e sono pari a $R_{filt} = 32k\Omega$. L'utente, in base alle esigenze sulla banda in uscita, deve inserire tra l'uscita e massa un adeguato condensatore in modo da realizzare un filtro passa basso. Il filtraggio migliora la risoluzione della misura e previene fenomeni di *aliasing*. La banda in uscita influenza direttamente la frequenza di campionamento del convertitore ADC.
La scelta dei tre condensatori C_x, C_y, C_z, uno per ogni asse, permette di definire la banda voluta nel range $[0.5Hz, 1600Hz]$ per gli assi x ed y (uscite X_{out} e Y_{out}) e nel range $[0.5Hz, 550Hz]$ per l'asse z (uscita Z_{out}). La scelta della banda, come detto precedentemente, dipende dall'applicazione a cui è destinato il sensore. Il condensatore C_{DC} viene usato per disaccoppiare l'accelerometro dal rumore presente nell'alimentazione, normalmente viene messo un condensatore di $0.1\mu F$, tuttavia se l'alimentazione contiene disturbi ad alta frequenza il valore deve essere aumentato, consigliata anche l'introduzione di una resistenza da 100Ω sulla linea di alimentazione, come riportato nel datasheet: il rumore può seriamente interferire con l'oscillatore interno al dispositivo e falsare completamente le misure.
Per il calcolo della frequenza di taglio (a -3dB) del filtro in uscita realizzato grazie all'aggiunta dei condensatori C_x, C_y, C_z si utilizza sulla seguente formula [15]:

$$F_{-3dB} = \frac{1}{2 \cdot \pi \cdot R_{filt} \cdot C_{(x,y,z)}} \simeq \frac{5\mu F}{C_{(x,y,z)}} \tag{3.1.1}$$

Dove il valore della capacità $C_{(x,y,z)}$ è espressa in μF. La resistenza R_{filt} ha una tolleranza del 15% rispetto al suo valore nominale di $32k\Omega$, quindi anche la banda varia di conseguenza. Un valore minimo di capacità di $4.7\mu F$ è comunque raccomandato in uscita. In tabella 3.1 sono mostrate alcune possibili bande e le relative capacità da mettere in uscita.

Bandwidth (Hz)	Capacitor (µF)
1	4.7
10	0.47
50	0.10
100	0.05
200	0.027
500	0.01

Tabella 3.1: Scelta del condensatore di filtraggio[15]

La tabella 3.2, estratta dal datasheet, riporta le specifiche dell'accelerometro. Le specifiche sono riferite alla temperatura ambiente di $T_A = 25°C$, ad una tensione di alimentazione $V_s = 3V$. In caso non fosse diversamente specificato si assumono $C_x = C_y = C_z = 0.1\mu F$ (external filter) ed un'accelerazione pari a $0g$. I valori massimi e minimi sono garantiti, quelli tipici non sono garantiti.

Parameter	Conditions	Min	Typ	Max	Unit
SENSOR INPUT	Each axis				
Measurement Range		±3	±3.6		*g*
Nonlinearity	% of full scale		±0.3		%
Package Alignment Error			±1		Degrees
Interaxis Alignment Error			±0.1		Degrees
Cross Axis Sensitivity[1]			±1		%
SENSITIVITY (RATIOMETRIC)[2]	Each axis				
Sensitivity at X_{OUT}, Y_{OUT}, Z_{OUT}	V_S = 3 V	270	300	330	mV/*g*
Sensitivity Change Due to Temperature[3]	V_S = 3 V		±0.015		%/°C
ZERO *g* BIAS LEVEL (RATIOMETRIC)	Each axis				
0 *g* Voltage at X_{OUT}, Y_{OUT}, Z_{OUT}	V_S = 3 V	1.2	1.5	1.8	V
0 *g* Offset vs. Temperature			±1		mg/°C
NOISE PERFORMANCE					
Noise Density X_{OUT}, Y_{OUT}			280		µg/√Hz rms
Noise Density Z_{OUT}			350		µg/√Hz rms
FREQUENCY RESPONSE[4]					
Bandwidth X_{OUT}, Y_{OUT}[5]	No external filter		1600		Hz
Bandwidth Z_{OUT}[5]	No external filter		550		Hz
R_{FILT} Tolerance			32 ± 15%		kΩ
Sensor Resonant Frequency			5.5		kHz
SELF TEST[6]					
Logic Input Low			+0.6		V
Logic Input High			+2.4		V
ST Actuation Current			+60		µA
Output Change at X_{OUT}	Self test 0 to 1		−150		mV
Output Change at Y_{OUT}	Self test 0 to 1		+150		mV
Output Change at Z_{OUT}	Self test 0 to 1		−60		mV
OUTPUT AMPLIFIER					
Output Swing Low	No load		0.1		V
Output Swing High	No load		2.8		V
POWER SUPPLY					
Operating Voltage Range		1.8		3.6	V
Supply Current	V_S = 3 V		320		µA
Turn-On Time[7]	No external filter		1		ms
TEMPERATURE					
Operating Temperature Range		−25		+70	°C

[1] Defined as coupling between any two axes.
[2] Sensitivity is essentially ratiometric to V_S.
[3] Defined as the output change from ambient-to-maximum temperature or ambient-to-minimum temperature.
[4] Actual frequency response controlled by user-supplied external filter capacitors (C_X, C_Y, C_Z).
[5] Bandwidth with external capacitors = $1/(2 \times \pi \times 32\ k\Omega \times C)$. For C_X, C_Y = 0.003 µF, bandwidth = 1.6 kHz. For C_Z = 0.01 µF, bandwidth = 500 Hz. For C_X, C_Y, C_Z = 10 µF, bandwidth = 0.5 Hz.
[6] Self-test response changes cubically with V_S.
[7] Turn-on time is dependent on C_X, C_Y, C_Z and is approximately 160 × C_X or C_Y or C_Z + 1 ms, where C_X, C_Y, C_Z are in µF.

Tabella 3.2: Tabella Specifiche[15]

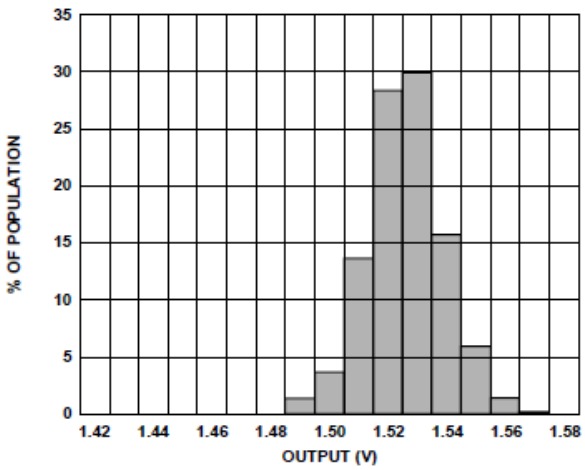
Come si può osservare sono indicate tutte le specifiche discusse nella sezione 2.4. Le voci più importanti sono la "*Measurement Range*" (range delle accelerazioni misurabili), la "*Sensitivity*" (fattore di conversione $mV \leftrightarrow g$), la "*Zero-g Bias Level*" (tensione in uscita quando è applicata accelerazione nulla), la "*Frequency Response*" (banda, frequenza di risonanza...), la "*Noise Performance*" (riporta solo la densità di potenza, per il calcolo della potenza totale RMS del rumore si usa la formula già vista) e la "Power Supply" (range delle tensioni di alimentazione, corrente assorbita e tempo di accensione). Si ricorda che i parametri del tipo "*ratiometric*" variano al variare di V_s, in caso di tensione di alimentazione diversa dal valore preso come riferimento è necessario tener conto del loro cambiamento.
Attenzione merita la voce indicata come "*Self test*": il dispositivo integra un sistema per testare il funzionamento del sensore di cui sono riportati i parametri. Attraverso un pin denominato *ST* (figura

3.1) è possibile "eccitare" il sensore: portando *ST* alla tensione V_s viene esercitata una forza elettrostatica sulla massa interna, viene misurata un'accelerazione di $-500mg$ sull'asse x, $500mg$ sull'asse y e $-200mg$ sull'asse z. Questo può essere usato per verificare eventuali guasti. Nell'uso normale del sensore *ST* viene posto a GND (*ground*).
Nella tabella *Absolute Maximum Ratings* (tabella 3.3) è riportata per esempio la massima accelerazione sopportabile (non misurabile!) dal dispositivo, oltre tale soglia è altamente probabile che il dispositivo venga danneggiato irreparabilmente. Viene riportato anche il range di tensioni di alimentazione che il sensore è in grado di tollerare, valori non inclusi nell'intervallo indicato possono causare un danneggiamento del dispositivo. Sono permesse sollecitazioni del sensore, a patto di rimanere sufficientemente all'interno dei margini di sicurezza imposti.

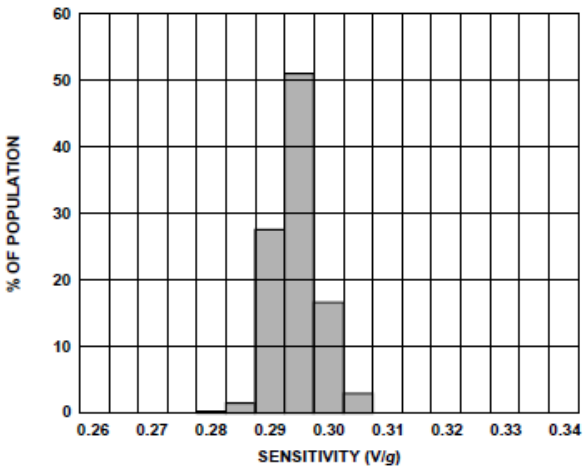
Parameter	Rating
Acceleration (Any Axis, Unpowered)	10,000 *g*
Acceleration (Any Axis, Powered)	10,000 *g*
V_S	−0.3 V to +7.0 V
All Other Pins	(COM − 0.3 V) to (V_S + 0.3 V)
Output Short-Circuit Duration (Any Pin to Common)	Indefinite
Temperature Range (Powered)	−55°C to +125°C
Temperature Range (Storage)	−65°C to +150°C

Tabella 3.3: Absolute Maximum Ratings[15]

Il datasheet riporta anche dei diagrammi ("*Typical Performance Characteristics*") che rappresentano lo scostamento dei parametri quali il *Zero-g Bias*, la *Sensitivity*, dai valori tipici. Le informazioni sono rappresentate tramite istogrammi, sulla scala x è rappresentata la grandezza di interesse, mentre sulla scala y il numero in percentuale di prove per cui la grandezza ha assunto un determinato valore (le prove effettuate sono superiori a 1000). I diagrammi, estratti dal datasheet, del *Zero-g Bias Level* per l'asse x (per $T_A = 25°C$ e per $V_s = 3V$) e della *Sensitivity* dell'asse x (per $T_A = 25°C$ e per $V_s = 3V$) sono riportati rispettivamente nelle figure 3.3a e 3.3b. Nel datasheet sono riportati anche grafici che mostrano l'andamento del *Bias* e della *Sensitivity* con la temperatura ed altre caratteristiche di interesse (anche per gli altri assi).



(a) *X-axix Zero-g Bias* a 25°*C* e con $V_s = 3V$



(b) *X-axix Sensitivity* a 25°*C* e con $V_s = 3V$

Figura 3.3: *Typical Performance Characteristics[15]*

Nella Figura 3.4 è rappresentato il *package* del sensore con indicati gli assi di riferimento per la misura dell'accelerazione. Un'accelerazione nella direzione dell'asse causa un aumento della tensione in uscita. Quando non è applicata accelerazione l'uscita si porta al valore "*Zero-g Bias Level*" V_0 che nominalmente uguale a metà della tensione di alimentazione. Quindi sarà possibile misurare il modulo e segno dell'accelerazione grazie alle seguenti relazioni:

$$a = \frac{(V_{mis} - V_0)}{Sensitivity} \tag{3.1.2}$$

$$a > 0 \Leftrightarrow V_{mis} > V_0 \tag{3.1.3}$$

$$a > 0 \Leftrightarrow V_{mis} < V_0 \tag{3.1.4}$$

dove V_{mis} è la tensione misurata e V_0 la tensione *Zero-g*, entrambe espresse in mV. L'accelerazione è espressa in g, per passare alle unità del sistema internazionale (SI) basta moltiplicare tale valore per la costante $g = 9.81 m/s^2$ (accelerazione di gravità terrestre).

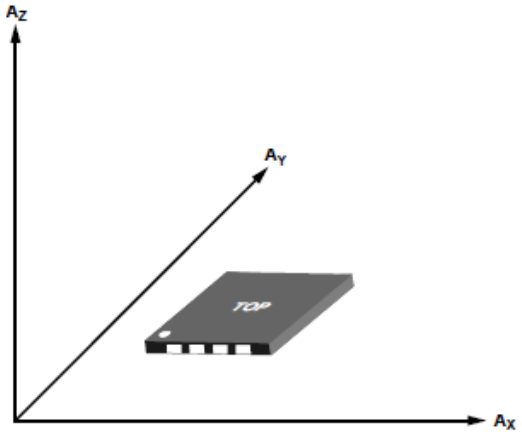


Figura 3.4: ADXL330 Assi di riferimento (rispetto al Package) [15]

Il sensore come discusso precedentemente misura l'accelerazione statica di gravità, a seconda della posizione del *package* nello spazio si registrerà, a dispositivo fermo, un valore costante di tensione sull'uscita corrispondente ad uno degli assi, o su più assi se la direzione della forza gravitazionale non coincide esattamente con uno degli assi di riferimento. In figura 3.5 è rappresentato il comportamento del sensore (le accelerazioni misurate sui singoli assi) a seconda dell'orientamento del *package*.

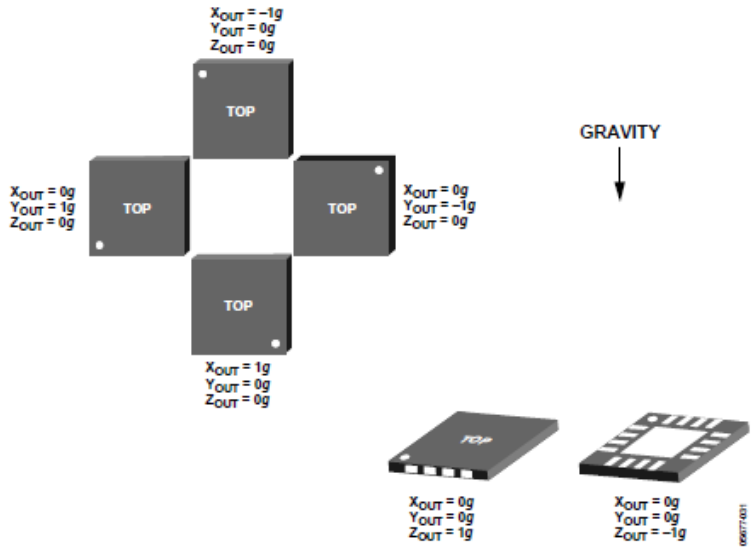
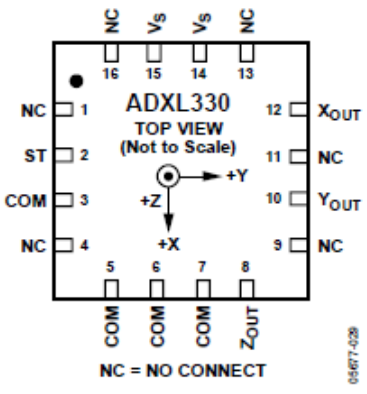
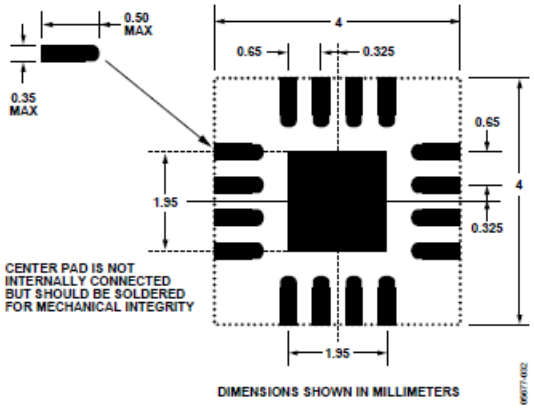


Figura 3.5: Comportamento nei confronti della gravità [15]

Per completezza infine si riporta la descrizione dei *pin* del sensore, con riferimento ai contatti elettrici del *package*. In figura 3.6a è riportato il *package* (con gli assi di riferimento). Si noti la posizione in alto a sinistra di un pallino nero, per identificare l'orientamento del chip. In figura 3.6b sono riportate le dimensioni in millimetri del *package* e dei relativi contatti elettrici (necessarie per il disegno del PCB). In tabella 3.4 sono riassunte le funzioni svolte dai singoli pin.



(a) *Top View* e *Pin Configuration*



(b) *Package Dimensions*

Figura 3.6: Configurazione Pin e *Package Layout [15]*

Pin No.	Mnemonic	Description
1	NC	No Connect
2	ST	Self Test
3	COM	Common
4	NC	No Connect
5	COM	Common
6	COM	Common
7	COM	Common
8	Z_{OUT}	Z Channel Output
9	NC	No Connect
10	Y_{OUT}	Y Channel Output
11	NC	No Connect
12	X_{OUT}	X Channel Output
13	NC	No Connect
14	V_S	Supply Voltage (1.8 V to 3.6 V)
15	V_S	Supply Voltage (1.8 V to 3.6 V)
16	NC	No Connect

Tabella 3.4: *Pin Function Descriptions* [15]

3.2 Conversione Analogica-Digitale

Scopo della conversione A/D (figura 3.7) è rendere disponibile il valore misurato ad un sistema di elaborazione numerico, ad esempio un microcontrollore. La tensione analogica in uscita dal sensore viene convertita in una parola composta da M bit rappresentanti numericamente il risultato della misura (l'operazione va fatta per ogni uscita, quindi sono necessari 3 convertitori).

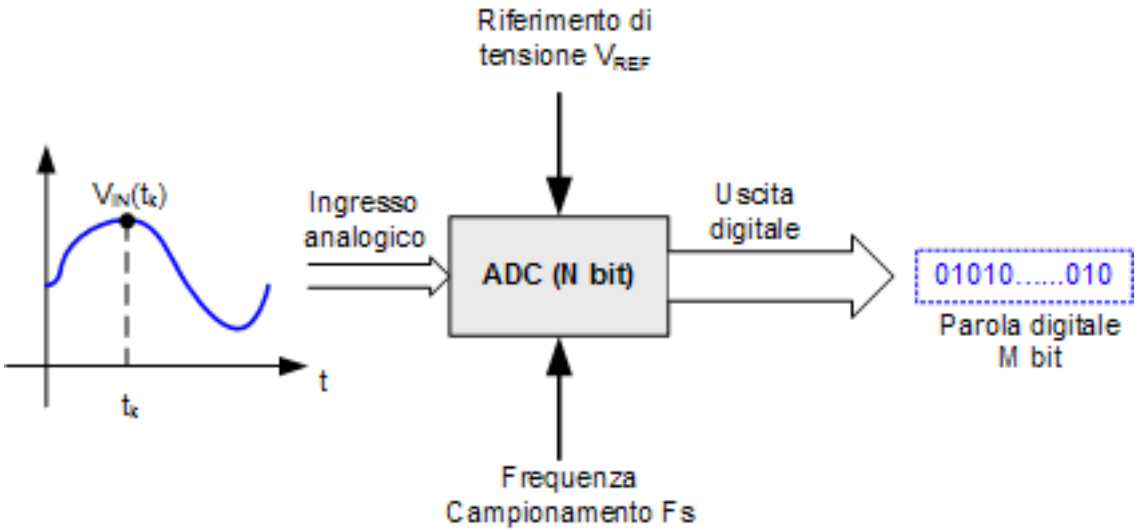


Figura 3.7: Convertitore ADC

Per la scelta del giusto ADC (*Analog to Digital Converter*) si devono valutare alcuni importanti parametri:

- **Frequenza di campionamento** F_s. La frequenza di campionamento va scelta in base al teorema del campionamento di Nyquist secondo la seguente relazione:

$$F_s \geq 2 \cdot B \tag{3.2.1}$$

dove B è la banda del segnale in uscita dall'accelerometro. Considerando il caso peggiore in cui si ha $B = 1600Hz$ (assi x ed y) risulta $F_s \geq 3.6kHz$. Di norma è sempre bene, per evitare

fenomeni di *aliasing*, tenersi al di sopra di questo valore.
Si dovrà provvedere quindi alla scelta di un ADC che sia in grado di campionare a tale velocità, ovvero il tempo di conversione T_{conv} deve essere inferiore a $T_s = 1/F_s = 277\mu s$.

- **Range del Quantizzatore** V_{FS}. Il range del quantizzatore dell'ADC, che nel nostro caso è unipolare, va da 0V a V_{FS}. Se V_s è la tensione di alimentazione del sensore, corrispondente anche alla massima tensione possibile del segnale in uscita, si ha che il range deve essere $[0\,,\,V_s]$. Considerando il caso in cui $V_s = 3V$ si ha che $V_{FS} = 3V$. Si deve prevedere un ADC compatibile con tale range, alternativamente il segnale del sensore va condizionato in modo da essere compatibile con il range del quantizzatore dell'ADC di cui si è in possesso.

- **Numero di bit M del Quantizzatore**. Più è elevato il numero di bit usati dal quantizzatore, più è piccolo l'errore dovuto al processo di quantizzazione (si raggiungono risoluzioni più alte). Nella scelta del numero di bit si deve valutare l'influenza del rumore dell'accelerometro: è inutile prendere un numero di bit troppo elevato in quanto l'accelerazione minima rilevabile è data dal valore del rumore, che dipende dalla banda B scelta.
Ricordando che il rumore si può calcolare la formula 2.4.3, nel caso peggiore in cui $B = 1600Hz$, in riferimento all'asse x (Sensitivity pari a 300mV/g), si ha che

$$Total\,Noise = 280 \cdot \sqrt{1.6 \cdot 1600} \simeq 14mg \tag{3.2.2}$$

$$V_{noise} = Total\,Noise \cdot Sensitivity = 14mg \cdot 300mV/g = 4.2mV \tag{3.2.3}$$

sarebbe quindi inutile scegliere un numero M di bit per il quale il passo di quantizzazione risulti inferiore a V_{noise}.
Si ha che il passo di quantizzazione è dato da:

$$\Delta q = V_{FS}/2^M \tag{3.2.4}$$

e quindi risulta:

$$M = log_2(V_{FS}/\Delta q) \tag{3.2.5}$$

il massimo numero M di bit "utili" è dato infine da:

$$M_{max} = log_2(V_{FS}/V_{noise}) = log_2(3V/4.2mV) = 10 \tag{3.2.6}$$

Se la banda richiesta è inferiore a 1600Hz il rumore risulterà minore, la risoluzione maggiore ed M_{max} assumerà valori più alti. In figura 3.8 è raffigurata schematicamente l'operazione svolta dal quantizzatore.

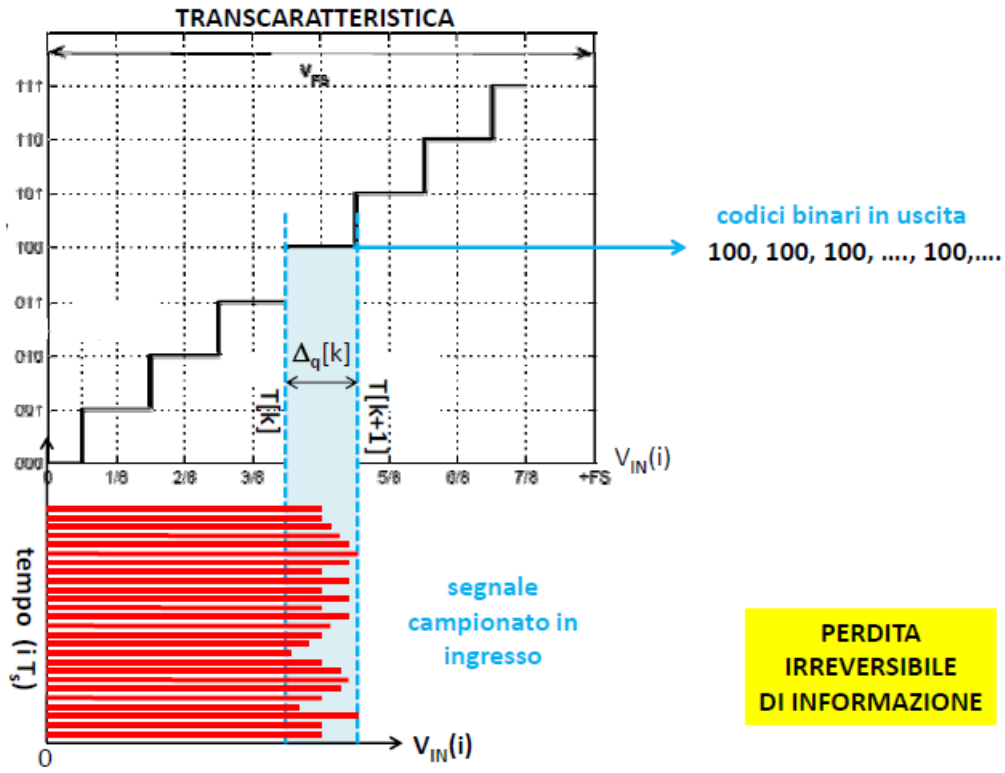


Figura 3.8: Esempio operazione di quantizzazione (M=3)

Definiti i parametri dell'ADC si può calcolare il passo di quantizzazione effettivo (si suppone di seguito $M = M_{max} = 10$ e $V_{FS} = 3V$):

$$\Delta q = \frac{V_{FS}}{2^M} = \frac{3V}{2^{10}} \simeq 2.93mV \tag{3.2.7}$$

La tensione misurata V_{mis} viene quantizzata e risulta:

$$V_{mis,q} = N \cdot \Delta q \tag{3.2.8}$$

con N numero naturale compreso tra 0 e $2^M - 1$.
N è il numero decimale associato alla parola binaria di M bit restituita dal quantizzatore (il quantizzatore si comporta da codificatore binario). Il massimo valore rappresentabile risulta essere:

$$V_{mis,max} = (2^M - 1) \cdot \Delta q = (2^{10} - 1) \cdot 2.93mV \simeq 2997mV \tag{3.2.9}$$

Data la sequenza binaria di M bit $b_{M-1}, b_{M-2}, \cdots\cdots\cdots b_0$ la tensione misurata quantizzata può essere espressa come:

$$V_{mis,q} = N \cdot \Delta q = \Delta q \cdot \sum_{i=0}^{M-1} b_i \cdot 2^i = V_{FS} \cdot \sum_{i=0}^{M-1} \frac{b_i}{2^{M-i}} \tag{3.2.10}$$

L'incertezza associata alla conversione analogica-digitale è pari a $\pm\Delta q/2$, infatti risulta che:

$$V_{mis,q} - \frac{\Delta q}{2} \leq V_{mis} \leq V_{mis,q} + \frac{\Delta q}{2} \tag{3.2.11}$$

La conversione A/D dei segnali in uscita dall'accelerometro normalmente avviene all'interno di un microcontrollore, che è dotato di un ADC interno. Di solito questo tipo di convertitori ha un numero di bit che va da 8 ai 12, la frequenza di campionamento ed altri parametri del modulo ADC possono essere impostati programmando opportunamente alcuni registri interni. Il risultato della misura viene caricato un un apposito registro che può essere letto via software, esso contiene il valore binario

associato a $V_{mis,q}$. Il tipo di convertitore ADC che usualmente trova impiego nei microcontrollori è quello "ad approssimazioni successive": l'architettura utilizzata sfrutta un convertitore DAC (*Digital to Analog Converter*) per generare delle tensioni di riferimento interne, e sfrutta una sorta di algoritmo di "ricerca binaria" per risalire al valore analogico in ingresso con la miglior approssimazione. Rispetto all'architettura *Flash*, questo convertitore impiega M passi (cicli di clock) per effettuare la conversione, quindi il tempo richiesto per la conversione risulta essere pari a:

$$T_{conv} = M \cdot T_{clk} \tag{3.2.12}$$

dove T_{clk} deve essere scelto superiore ai tempi di commutazione della logica e del DAC.
In Figura 3.9 è mostrata l'architettura di un ADC SAR (*Successive Approximation Register*).

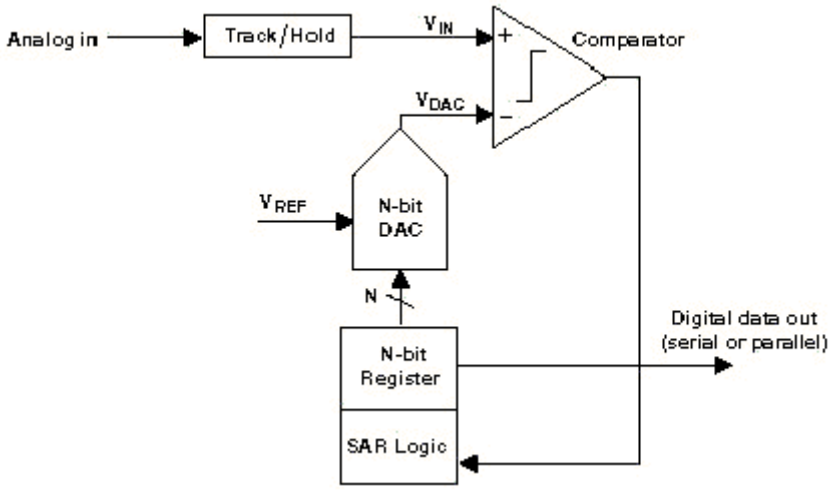


Figura 3.9: Architettura ADC ad approssimazioni successive

Il registro SAR viene inizializzato al valore "100....0" corrispondente a metà dell'escursione del DAC, $V_{DAC} = V_{REF}/2 = V_s/2$. Successivamente la logica di controllo memorizza in ogni bit del registro, partendo dal più significativo, il risultato della comparazione: viene mantenuto il bit a '1' se $V_{IN} > V_{DAC}$, altrimenti viene posto a '0'. Nel ciclo di clock successivo il registro SAR viene inizializzato al valore "X10...0", dove la X rappresenta il risultato della precedente comparazione, e la procedura si ripete. Ad ogni iterazione il registro SAR viene quindi aggiornato e l'uscita V_{DAC} cambia avvicinandosi sempre di più alla tensione da misurare V_{IN}. L'algoritmo si ferma quando viene raggiunto il bit meno significativo: il contenuto del registro corrisponde al valore numerico associato alla tensione analogica in ingresso e può essere letto dal programma in esecuzione sul microcontrollore. L'ADC è pronto per una nuova conversione. In figura 3.10 è mostrato un esempio della procedura di conversione (con convertitore a 4 bit).

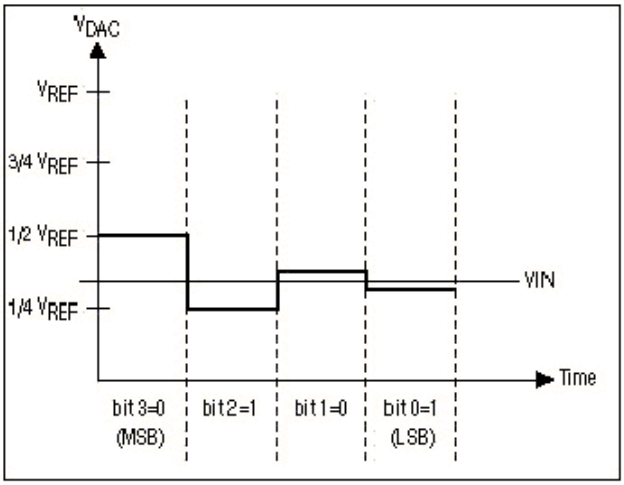


Figura 3.10: Conversione ad approssimazioni successive (4 bit)

Il microcontrollore, elaborati i dati di misura, può essere usato per controllare opportunamente attuatori esterni (ad esempio per applicazioni robotiche), per comunicare i dati ad altri dispositivi (in modo *wired* o *wireless*), visualizzarli su un *display,* ecc (figura 3.11). Nel Wiimote avvengono operazioni analoghe: il microprocessore BCM2042 invia i valori di accelerazione misurati attraverso una connessione Bluetooth ad un dispositivo host (la console Wii, il computer...) dove vengono ulteriormente elaborati.

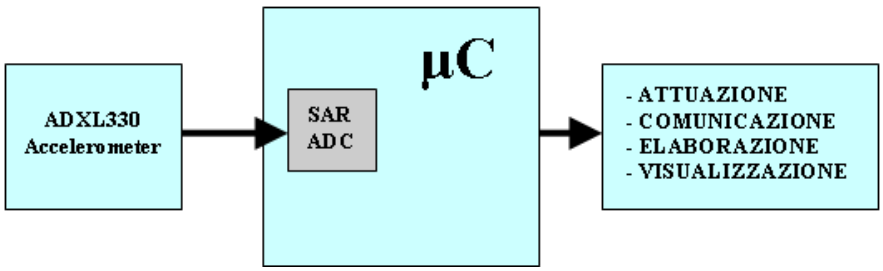


Figura 3.11: Interfaccia Sensore-Microcontrollore

Capitolo 4

ACQUISIZIONE DEI DATI

Dopo aver studiato la struttura ed il funzionamento del Wiimote, ed aver analizzato in dettaglio diverse tipologie di sensori per la misura dell'accelerazione, si procede a presentare la modalità di comunicazione tra controller ed *host*. L'*host*, nel caso di questa tesi, è costituito da un PC con sistema operativo Windows XP. I dati trasmessi dal Wiimote verranno acquisiti per mezzo della libreria driver *WiiLAB* [31, 32]. La libreria, come si vedrà, permette la comunicazione tra il controller e gli ambienti di programmazione *Matlab*[1] e *Labview*[2]. Quest'ultimo sarà utilizzato nel capitolo 5 per realizzare semplici applicazioni per scopi didattici che fanno uso del dispositivo, visto come strumento per la misura di accelerazione.
Nella prossima sezione si analizzeranno i *reports HID* scambiati tra controller e computer, per poi esaminare in dettaglio, nella sezione 4.2, il funzionamento delle librerie *WiimoteLib* e *WiiLAB*. La libreria *WiiLAB* si appoggia alla *WiimoteLib* per la comunicazione con il Wiimote, quindi è necessario studiare il funzionamento di entrambe per comprendere come i dati del dispositivo vengono resi disponibili ai software che ne fanno uso. Infine verranno esaminate, sia per *Matlab* (sottosezione 4.2.2) che per *Labview* (sottosezione 4.2.5), le operazioni necessarie ad effettuare l'acquisizione dei dati trasmessi dal Wiimote sfruttando la libreria, la loro elaborazione e la loro visualizzazione. In figura 4.1 è mostrato uno schema del sistema di acquisizione che si andrà a realizzare.

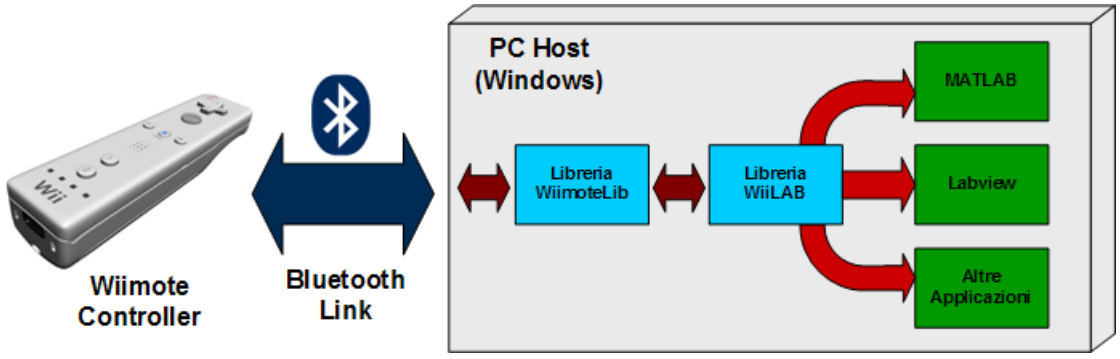


Figura 4.1: Il sistema di acquisizione

[1]**MATLAB** (**Mat**rix **Lab**oratory) è un ambiente di programmazione per il calcolo numerico e l'analisi statistica creato dalla The MathWorks. MATLAB consente di manipolare matrici, visualizzare funzioni e dati, implementare algoritmi, creare interfacce utente, e comunicare con altri programmi

[2]**LabVIEW** (**Lab**oratory **Vi**rtual **I**nstrumentation **E**ngineering **W**orkbench) è un ambiente di sviluppo integrato (IDE) creato da National Instrument (NI). Sfrutta un linguaggio di programmazione grafico chiamato "Linguaggio G". Labview viene utilizzato principalmente per acquisizione, elaborazione e visualizzazione dati dati, ad esempio per il controllo di processi in ambito industriale.

4.1 La comunicazione ed i Reports HID

Quando il dispositivo *Host* effettua la ricerca delle periferiche Bluetooth raggiungibili (utilizzando il *Service Discovery Protocol*), il Wiimote risponde, se si trova in "*Discovery Mode*", con un proprio *report* identificativo, apparendo come una generica periferica di puntamento con il nome "*Nintendo RVL-CNT-01*". Per abilitare la modalità *discovery* del Wiimote è necessario premere e tenere premuti entrambi i pulsanti "1" e "2" del controller durante tutta la fase di ricerca e la connessione (*pairing*) con il dispositivo *Host*. Durante questa fase i 4 led presenti sul controller lampeggiano. Il numero di led che lampeggiano sono proporzionali al livello della batteria: quattro led indicano che la batteria è completamente carica, mentre uno solo indica che la batteria deve essere sostituita/ricaricata. Se la connessione con il computer ha successo essi continuano a lampeggiare anche dopo il rilascio dei pulsanti. In figura 4.2 sono evidenziati i pulsanti "1" e "2" sul controller.

Figura 4.2: Pulsanti "1" e "2" per abilitare la modalità *discovery*

Dopo questa fase di inizializzazione il Wiimote viene rilevato dal *driver* che si occupa della comunicazione, che ne prende il controllo in scrittura (invio richieste dati e configurazioni) ed in lettura (recupero valori dei sensori, stato pulsanti, livello batteria...).
In tabella 4.1 vengono riportati i *report* in ingresso ed in uscita usati dal Wiimote [8].

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

Tabella 4.1: Reports HID del Wiimote: Un report di tipo I (Input) è inviato dal Wiimote all'Host, mentre un report di tipo O (Output) viene inviato dall'Host al Wiimote [8]

Ad ogni *report* è associato un identificativo, rappresentato nella colonna *ID* della tabella; il campo *Size* specifica la dimensione in bytes del relativo *report*. Le modalità di lettura dei sensori del Wiimote sono impostate attraverso i bytes di configurazione del *report* "*Data Reporting Mode*" ($ID = 0x12$), che permette di stabilire quali dati devono essere restituiti attraverso i "*Data Reports*" (da $ID = 0x30$ a $ID = 0x3f$). La libreria che si occupa della comunicazione fornisce un set di funzioni di basso livello che permettono di impostare questi parametri e di interpretare i dati ricevuti.

4.1.1 Data reporting e Data Reports

Le modalità utilizzate dai *"Data Reports"* possono essere suddivise in due categorie principali:

1. **Dati provenienti dai sensori del dispositivo** (pulsanti, accelerometro, *camera IR*). I dati sono organizzati in bytes consecutivi; la dimensione del *report* varia a seconda della modalità scelta attraverso il *"Data Reporting Mode"*: vengono restituiti solo i dati specificati attraverso i bytes di configurazione del *report*. Le modalità impostabili verranno elencate e discusse più avanti.

2. **Dati provenienti dalle espansioni**, che possono essere eventualmente connesse al dispositivo mediante un connettore dedicato. In questo caso vengono trasmessi i dati riferiti all'espansione attualmente connessa; il modo in cui tali dati sono organizzati dipende dalla tipologia di periferica interfacciata al Wiimote. Un *driver* che implementa funzionalità aggiuntive legate ad una specifica espansione deve conoscere l'organizzazione dati vengono all'interno del *report*.

Vi sono due modalità di aggiornamento dei *reports*, impostabili attraverso il primo byte di configurazione del *"Data Reporting Mode"*:

1. **Report Continuo**: il Wiimote invia all'*host* i *reports* dei sensori senza considerare i valori precedentemente inviati.

2. **Report a Evento**: il Wiimote invia all'*host* i *reports* dei sensori solo se i valori letti dai sensori sono cambiati rispetto all'ultimo invio.

Il secondo byte di configurazione del *"Data Reporting Mode"* stabilisce la dimensione dei *"Data Reports"* ed il loro significato Di seguito vengono elencate le possibili modalità operative del controller ed i dati trasmessi all'*host*:

- **Pulsanti** (valore *0x30*): in questa modalità il *"Data Report"* avente $ID = 0x30$ restituisce 2 bytes contenenti lo "stato dei pulsanti". Per "stato dei pulsanti" si intende l'insieme di bit rappresentanti lo stato (pulsante premuto o non premuto) di tutti i pulsanti del controller.

- **Pulsanti ed Accelerometro** (valore *0x31*): in questa modalità il *"Data Report"* avente $ID = 0x31$ restituisce 5 bytes conteneti lo stato dei pulsanti ed i valori di accelerazione misurati dall'accelerometro sui tre assi x, y e z.

- **Pulsanti ed Estensione a 8 bytes** (valore *0x32*): in questa modalità il *"Data Report"* avente $ID = 0x32$ restituisce 10 bytes. I primi due contengono lo stato dei pulsanti mentre i rimanenti 8 provengono dall'estensione eventualmente connessa al Wiimote.

- **Pulsanti, Accelerometro e Camera IR a 12 bytes** (valore *0x33*): in questa modalità il *"Data Report"* avente $ID = 0x33$ restituisce 17 bytes. I primi 5 sono analoghi alla modalità "Pulsanti ed Accelerometro", i rimanenti 12 contengono le informazioni sulla posizione dei 4 punti luminosi individuati dalla microcamera infrarossa. Questa è la modalità più usata per accedere ai dati del Wiimote quando nessuna estensione è connessa.

- **Pulsanti ed Estensione a 19 bytes** (valore *0x34*): in questa modalità il *"Data Report"* avente $ID = 0x34$ restituisce 21 bytes. I primi due contengono lo stato dei pulsanti mentre i rimanenti 19 provengono dall'estensione eventualmente connessa al Wiimote.

- **Pulsanti, Accelerometro, Estensione a 16 bytes** (valore *0x35*): in questa modalità il *"Data Report"* avente $ID = 0x35$ restituisce 21 bytes. I primi 5 contengono lo stato dei pulsanti ed i dati dell'accelerometro, mentre i rimanenti 16 provengono dall'estensione eventualmente connessa al Wiimote.

- **Pulsanti, Camera IR a 10 bytes, Estensione a 9 bytes** (valore *0x36*): in questa modalità il *"Data Report"* avente $ID = 0x36$ restituisce 21 bytes contenenti le informazioni elencate.

- **Pulsanti, Accelerometro, Camera IR a 10 bytes ed Estensione a 6 bytes** (valore *0x37*): in questa modalità il *"Data Report"* avente $ID = 0x37$ restituisce 21 bytes contenenti le informazioni elencate.

- **Estensione a 21 bytes** (valore *0x3d*): in questa modalità il *"Data Report"* avente $ID = 0x3d$ restituisce 21 bytes contenenti solo dati provenienti dall'estensione.

- **Pulsanti, Accelerometro e Camera IR a 36 bytes** (valore *0x3e* oppure *0x3f*): in questa modalità sono necessari 2 *"Data Reports*", quelli aventi $ID = 0x36$ ed $ID = 0x3f$. Ognuno di essi è composto da 21 bytes e vengono restituiti lo stato dei pulsanti, i dati dell'accelerometro e 36 bytes associati alla *IR Camera*. In questa modalità, a differenza da tutte le altre, vengono restituiti solo gli 8 bit più significativi dei valori rappresentanti le accelerazioni misurate dai 3 assi dell'accelerometro.

Quando il Wiimote viene acceso viene predefinito il *Data Report* "Pulsanti" ($ID = 0x30$), l'applicazione che fa uso del controller dovrà eventualmente provvedere ad impostare una modalità alternativa attraverso il *report "Data Reporting Mode*". Nel caso in cui si fosse interessati ai valori misurati

dall'accelerometro questa operazione è necessaria. Quando un'estensione viene connessa o disconnessa dal Wiimote il controller interrompe l'invio dei *"Data Reports"* ed attende che l'host imposti una delle modalità descritte: l'applicazione con la quale il Wiimote è interfacciato può scegliere una modalità che permetta la ricezione dei dati dell'estensione, nel caso in cui avvenga la connessione dell'estensione, oppure una modalità che non preveda la ricezione dei dati dall'estensione, nel caso in cui l'estensione venga disconnessa. Attraverso il *report "Status Information*", che verrà ora analizzato, è possibile individuare a quale dei due casi corrisponde l'interruzione dell'invio dei *"Data Reports*".

4.1.2 Status Reporting

Il *report* denominato *Status Information* ($ID = 0x20$) contiene informazioni sullo stato del Wiimote. Il *report* è composto da 6 bytes ed è usato per comunicare all'Host parametri quali il livello di batteria, la configurazione dei led di indicazione, lo stato dell'estensione, lo stato dei pulsanti.
Esso è composto da:

- 2 bytes contenenti lo stato dei pulsanti;
- 1 byte contenente diversi flag di stato: segnalazione batteria quasi scarica, segnalazione connessione di un'estensione, stato microcamera (accesa/spenta), stato speaker interno (acceso/spento), stato dei quattro led di segnalazione (acceso/spento per ogni led);
- 2 bytes fissi a 0;
- 1 byte contenente il livello della batteria.

Il *report "Status Information"* è automaticamente inviato all'*host* quando un'estensione viene connessa/disconnessa al dispositivo, alternativamente è necessario inviare la richiesta di stato attraverso il *report "Status Information Request"* ($ID = 0x15$).

4.1.3 Significato dei Reports rimanenti

Tra gli altri *Output Reports* vi sono:

- *"Player LEDs"* ($ID = 0x11$): attraverso questo *report* è possibile controllare i 4 led di segnalazione del Wiimote.
- *"IR Camera Enable"* ($ID = 0x13$) ed *"IR Camera Enable 2"* ($ID = 0x1a$): permettono di attivare la microcamera infrarossa. I parametri di configurazione della *IR Camera*, come ad esempio la sensibilità devono essere salvati in opportuni registri di controllo all'interno della memoria EEPROM di cui è dotato il Wiimote.
- *"Speaker Enable"* ($ID = 0x14$): attraverso questo *report* è possibile accendere o spegnere lo *speaker* di cui è dotato il Wiimote. Il controller dispone di un piccolo altoparlante in grado di riprodurre dei semplici suoni, che vengono salvati sulla memoria EEPROM. I bytes contenenti il suono da riprodurre vengono inviati grazie al *report "Speaker Data"*, lo *speaker* può essere messo in modalità "*mu*te" grazie al *report "Speaker Mute"*.
- *"Read Memory and Registers"* ($ID = 0x17$): tramite questo *report* è possibile definire quali indirizzi della memoria si vuole leggere, compresi i "registri di controllo", le aree di memoria riservate alla configurazione delle periferiche interne al Wiimote e quelle relative alla configurazione ed allo scambio di dati tra il controller e le estensioni. I dati letti sono restituiti dal *report "Read Memory and Registers Data"*, a pacchetti di 16 bytes.

- *"Write Memory and Registers Data"* ($ID = 0x16$): il *report* permette la scrittura di dati nella memoria EEPROM del Wiimote. Nel *report* viene indicato l'indirizzo in cui scrivere e la dimensione, in bytes, dei dati da scrivere. Possono essere scritti al massimo 16 bytes alla volta.

- *"Acknowledge Output Report"* ($ID = 0x22$): il *report* è usato per comunicare l'esito di operazioni ed eventuali errori di comunicazione, scrittura/lettura in memoria.

- ON/OFF Vibrazione: il Wiimote integra un piccolo motorino in corrente continua. Non esiste però un *report* dedicato all'attivazione/disattivazione di tale periferica. Il motorino tuttavia può essere controllato attraverso diversi *report* semplicemente ponendo a '1' oppure a '0' l'LSB del primo byte di controllo. Questa operazione ha effetto solo sui *reports "Data Reporting Mode"*, *"IR Camera Enable 1 e 2"*, *"Speaker Enable"* e *"Speaker Mute"*.

4.2 Libreria WiimoteLib e Wrapper WiiLAB

In questa sezione verranno analizzate le librerie *WiimoteLib* e *WiiLAB*, si forniranno alcuni esempi su come utilizzare quest'ultima libreria per acquisire i dati trasmessi dal Wiimote attraverso i linguaggi di programmazione *Matlab* e *Labview*, focalizzandosi sui valori relativi alle accelerazioni misurate dall'accelerometro interno al controller. Le prime acquisizioni permetteranno di determinare sperimentalmente alcuni parametri legati al dispositivo ed all'acquisizione dei dati:

- Minimo periodo di campionamento, dovuto alle latenze introdotte dalle librerie ed all'uso di un sistema di comunicazione con il Wiimote basato sul polling.

- Range di misura, ovvero accelerazioni massime e minime misurabili con il Wiimote (non corrisponde al range di misura dell'accelerometro).

- Passo di quantizzazione (misure di accelerazione), dovuto all'operazione di quantizzazione eseguita dal convertitore analogico-digitale. Rappresenta la risoluzione del sistema di misura "Wiimote".

4.2.1 WiimoteLib

La libreria *WiimoteLib*, creata dal programmatore Brian Peek (si veda [23]), fornisce un driver completo per il Wiimote. Per funzionare ha bisogno di uno *stack* Bluetooth Microsoft, Windcomm o compatibili, che sono la quasi la totalità di quelli in commercio.
La libreria implementa la comunicazione HID e comunica con il controller attraverso i *reports* descritti precedentemente. All'utilizzatore della libreria vengono messe a disposizione delle funzioni di livello abbastanza basso per poter interagire con il Wiimote (in linguaggio *.Net*). La comunicazione con il Wiimote avviene instanziando un oggetto di tipo *Wiimote*: l'oggetto contiene vari metodi per la connessione, disconnessione del controller, la lettura dei dati ricevuti, la configurazione del Report Mode, ecc.
Stabilita la connessione vi sono due possibili modi per leggere i dati provenienti dal Wiimote:

- **Modalità Polling**. Viene usata nel caso in cui il Wiimote sia stato configurato in modalità "Report Continuo". Per ottenere gli ultimi dati che il controller ha inviato al computer è necessario invocare il metodo *GetStatus()* dell'oggetto *Wiimote*. Il metodo restituisce un oggetto di tipo *WiimoteState*, quest'ultimo costituisce la struttura dati all'interno della quale è memorizzato lo

stato[3] di tutte le periferiche contenute nel controller. Le proprietà dell'oggetto rappresentano i valori di interesse, quali ad esempio le accelerazioni misurate sui tre assi dell'accelerometro, lo stato dei pulsanti, il livello della batteria, i dati della *IR Camera*, ecc.

- **Modalità ad Eventi**. Viene usata nel caso in cui il Wiimote sia stato configurato in modalità "Report a Evento". Ogni volta che viene ricevuto un *report* dal controller si verifica un evento in corrispondenza del quale viene invocato il metodo *WiimoteChanged()*. Il programmatore che utilizza la libreria deve solo scrivere, all'interno del metodo *WiimoteChanged(),* il codice che gestirà i dati ricevuti. Anche in questo caso lo stato delle periferiche viene memorizzato in un oggetto *WiimoteState* nel momento in cui si verifica l'evento.
  È possibile gestire tramite eventi anche i casi di connessione e disconnessione di un'estensione al Wiimote.

La libreria nella versione attuale non è completa, al momento in cui si scrive essa non è in grado di sfruttare tutte le modalità di *report* disponibili, inoltre non gestisce le funzionalità legate allo *speaker* ed ha un supporto delle estensioni limitato agli accessori *WiiMotePlus*, *Nunchuck*, *Guitar Hero*.
Le modalità di *report* supportate dalla libreria sono:

- *Buttons*: dati riferiti allo stato dei pulsanti, corrispondente al *Data Report* avente $ID = 0x30$;
- *ButtonsAccel*: stato dei pulsanti e valori dell'accelerometro, corrispondente al *Data Report* avente $ID = 0x31$;
- *IRAccel*: stato dei pulsanti, valori dell'accelerometro e della *IR Camera*,corrispondente al *Data Report* avente $ID = 0x33$;
- *ButtonsExtension*: stato dei pulsanti e dati provenienti dall'estensione, corrispondente al *Data Report* avente $ID = 0x32$;
- *EstensionAccel*: stato dei pulsanti, valori dell'accelerometro e dati provenienti dall'estensione, corrispondente al *Data Report* avente $ID = 0x35$;
- *IRExtensionAccel*: stato dei pulsanti, valori dell'accelerometro e della IR Camera, dati provenienti dall'estensione, corrispondente al *Data Report* avente $ID = 0x37$.

I nomi usati per indicare le modalità sono quelli usati nella libreria, vi è una corrispondenza univoca con le modalità già descritte. *WiimoteLib* riorganizza inoltre i dati contenuti nei *report* ricevuti, dividendoli in base alle periferiche di appartenenza (accelerometro, pulsanti...), per permetterne un accesso più trasparente. Per comunicare al controller la modalità usata esiste il metodo *SetReportType()* dell'oggetto *Wiimote*.
Di seguito viene riportato uno spezzone di codice (scritto in C#) per chiarire i concetti finora espressi [24].

[3]Con la parola *stato* ci si riferisce all'insieme di tutti i dati inviati che il Wiimote invia all'*Host*, non solo al report di stato descritto nella sottosezione 4.1.2.

```
using WiimoteLib;

private void Form1_Load(object sender, EventArgs e)
{
    // create a new instance of the Wiimote
    Wiimote wm = new Wiimote();

    // setup the event to handle state changes
    wm.WiimoteChanged += wm_WiimoteChanged;

    // setup the event to handle insertion/removal of extensions
    wm.WiimoteExtensionChanged += wm_WiimoteExtensionChanged;

    // connect to the Wiimote
    wm.Connect();

    // set the report type to return the IR sensor and accelerometer data (buttons always come back)
    wm.SetReportType(Wiimote.InputReport.IRAccel, true);
}


void wm_WiimoteExtensionChanged(object sender, WiimoteExtensionChangedEventArgs args)
{
    if(args.Inserted)
        wm.SetReportType(Wiimote.InputReport.IRExtensionAccel, true);    // return extension data
    else
        wm.SetReportType(Wiimote.InputReport.IRAccel, true);            // back to original mode
}

void wm_WiimoteChanged(object sender, WiimoteChangedEventArgs args)
{
    // current state information
    WiimoteState ws = args.WiimoteState;

    // write out the state of the A button
    Debug.WriteLine(ws.ButtonState.A);
}
```

L'esempio riporta il codice mediante il quale gestire la comunicazione dei dati provenienti dai sensori tra Wiimote e PC Host, tramite modalità ad eventi. Tuttavia, la libreria WiiLAB, che verrà descritta in seguito, farà uso di una modalità di comunicazione basata sul sistema di polling.

4.2.2 WiiLAB e MATLAB

La libreria *WiiLAB* permette di gestire in modo semplice ed immediato le principali funzionalità della libreria *WiimoteLib* dall'ambiente di programmazione *Matlab* [28, 31].
La *WiiLAB* dispone di metodi di più alto livello rispetto alla *WiimoteLib*, il programmatore può dimenticarsi dei *report* e dell'organizzazione dei dati ricevuti dal controller. I principali metodi messi a disposizione, che verranno utilizzati anche nella sottosezione 4.2.5 per la comunicazione tra la libreria e l'ambiente di programmazione *Labview*, sono i seguenti:

- *Connect()*. Il metodo esegue la connessione con il Wiimote impostando, attraverso la libreria *WiimoteLib*, la modalità "Report Continuo".
- *Disconnect()*. Il metodo termina la connessione con il Wiimote.

- *GetAccelState()*. Il metodo restituisce un *array*[4] di tre elementi rappresentanti le accelerazioni misurate dal Wiimote.
- *GetButtonsState()*. Il metodo restituisce un *array* rappresentante lo stato dei pulsanti del controller.

Per la descrizione di tutti i metodi disponibili e gli esempi sul loro utilizzo si faccia riferimento alla *wiki* online della libreria [32]. È necessario mettere in evidenza alcune ulteriori considerazioni a proposito delle librerie: per la comunicazione con la *WiiLAB* viene utilizzata una versione modificata della *WiimoteLib*. Le modifiche, introdotte dai programmatori del progetto *WiiLAB*, permettono una migliore comunicazione con il controller grazie alla riduzione delle latenze nell'invio dei *reports* al dispositivo ed alla gestione degli errori di connessione, attraverso l'introduzione di un nuovo tipo di dati per la comunicazione di tali errori alla libreria *WiiLAB*. Quest'ultima modifica è stata necessaria in quanto la disconnessione dell'adattatore Bluetooth, mentre la libreria era in esecuzione, causava delle *eccezioni I/O* non gestite e quindi il crash dell'applicazione.
La connessione alla libreria *WiiLAB* avviene grazie ad un *COM Server,* attraverso il quale è possibile utilizzare tutte le funzioni da essa messe a disposizione. La lettura dei dati restituiti dal Wiimote avviene attraverso un polling: il controller, come visto precedentemente, invia i dati al computer in modo continuo; per recuperare gli ultimi valori ricevuti è quindi necessario effettuare continue richieste alla *WiimoteLib* attraverso la *WiiLAB*.
Per poter interagire con il Wiimote in modo rapido i creatori del progetto *WiiLAB* forniscono la classe "*Wiimote Class*" in linguaggio *Matlab*, che contiene tutte le strutture dati per la memorizzazione delle informazioni ricevute dalla libreria. Tale classe contiene i metodi associati a tutte le funzioni della libreria, in tal modo l'utente che ne fa uso non ha bisogno di conoscere la sintassi per accedere ad un *COM Server* ma può focalizzarsi solo sullo sviluppo di algoritmi che fanno uso dei dati acquisiti.
La classe integra anche delle funzioni extra, create per facilitare la lettura dei dati del Wiimote: esse comprendono il plot automatico dei dati restituiti dall'accelerometro, la visualizzazione di tali valori per mezzo di animazioni grafiche, ecc. Queste caratteristiche aggiuntive non verranno discusse.
Assieme alla *"Wiimote Class"* vengono fornite numerose funzioni "globali" che gestiscono le procedure di interrogazione del controller e semplificano ulteriormente lo sviluppo di applicazioni: la loro chiamata restituisce un vettore contenente i dati richiesti. Esempio più importante è costituito dalla funzione *getWiimoteAccel()* (usata ampiamente in seguito) che restituisce un vettore di 3 elementi contenente i valori di accelerazione misurati rispettivamente dagli assi x, y e z dell'accelerometro, di tale funzione. In figura 4.3 è riportato uno schema che illustra l'interazione tra *WiimoteLib*, *WiiLAB* e *Matlab*.

[4]Le informazioni vengono restituite all'interno di arrays in quanto strutture più facilmente interpretabili dall'ambiente Matlab.

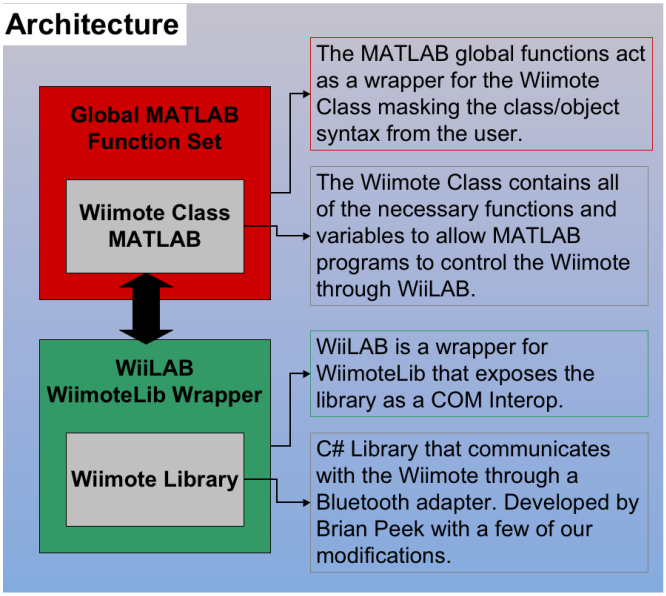


Figura 4.3: Schema interazione tra WiimoteLib, WiiLAB e Matlab [32]

Gli elementi base che costituiscono la *"Wiimote Class"* sono le *properties* pubbliche e private (l'equivalente delle *struct* pubbliche e private in C), il *constructor* (o *costruttore* della classe) ed i metodi della classe:

1. Le *properties* definiscono la struttura dati delle informazioni lette attraverso la libreria *WiiLAB*. La classe fa uso di properties private, l'utilizzatore può solo leggere le strutture dati ma non scriverle. Alcune delle strutture dati di interesse sono:

   (a) *LEDs*, che a sua volta contiene 4 campi corrispondenti allo stato dei 4 led del Wiimote:

      i. *LED1*
      ii. *LED2*
      iii. *LED3*
      iv. *LED4*

   (b) *Rumble*, che contiene lo stato attuale della vibrazione (on/off);

   (c) *Accel*, che contiene gli ultimi valori letti dall'accelerometro; è composto da 3 campi corrispondenti ai valori corrispondenti ai tre assi:

      i. *X*
      ii. *Y*
      iii. *Z*

   (d) *Buttons*, che contiene lo stato dei pulsanti; è composto da 11 campi corrispondenti ai singoli pulsanti:

      i. *A*
      ii. *B*
      iii. *Up*
      iv. *Right*
      v. *Down*
      vi. *Left*

vii. *Home*
viii. *Plus*
ix. *Minus*
x. *One*
xi. *Two*

(e) *isConnected,* che è costituito da due campi:

i. un intero corrispondente allo stato della connessione del Wiimote;
ii. un intero contenente il riferimento associato al Wiimote connesso (possono essere gestiti fino a 4 controller contemporaneamente).

2. Il costruttore della classe inizializza la connessione con il *COM Server* e quindi con la libreria *WiiLAB*.

3. I metodi della classe svolgono funzioni quali la connessione/disconnessione al Wiimote, l'aggiornamento dei dati contenuti nelle *properties* ed il controllo degli elementi di feedback, vibrazione e LEDs. Tali metodi sono:

(a) *Connect()* esegue la connessione con il Wiimote. Nel caso di più controller connessi al PC verrà utilizzato il primo dispositivo disponibile.

(b) *Disconnect()* e *DisconnectAllWiimotes()* permettono di disconnettere il Wiimote oppure tutti i Wiimote connessi (vi è la possibilità infatti di gestire eventualmente più di un controller contemporaneamente). Il Wiimote quando disconnesso non è tuttavia spento, in quanto ancora connesso al PC, per enfatizzare questo fatto i 4 led del controller vengono accesi. Se la vibrazione era attiva viene automaticamente disattivata.

(c) *CheckConnection()* permette di interrompere l'esecuzione del codice *Matlab* nel caso in cui il Wiimote venga disconnesso in modo imprevisto, evitando il crash del programma (si ricordino le modifiche fatte alla *WiimoteLib* a questo scopo).

(d) *UpdateWiimoteState()* permette di aggiornare le strutture dati delle *properties*; in sostanza è questo metodo che esegue il polling alla libreria ed acquisisce i dati dei sensori del wiimote. Il metodo chiama i singoli sottometodi *GetAccelState()*, *GetButtonsState()*, ecc, i quali si occupano di fornire lo stato delle singole periferiche (viene eseguito anche il check della connessione). Il metodo *GetBatteryState(),* che restituisce il livello di carica della batteria, è invece a parte e non è chiamato dal metodo di update generale. I vari metodi possono comunque essere chiamati singolarmente.

(e) *SetRumble()* e *SetLEDs()* permettono di attivare o disattivare il motorino del Wiimote e di accendere o spegnere i singoli led blu di segnalazione.

Una descrizione dettagliata della *"Wiimote Class"* è fornita nella *wiki* online progetto *WiiLAB* [32], contenente per ogni metodo il relativo codice sorgente. Il sorgente completo è contenuto nel file *wiimote.m*, mentre in altri files (*AccelState.m, ButtonState.m, ...*) sono definite le strutture dati. Un esempio di inizializzazione del Wiimote e lettura dell'accelerometro è il seguente:

```
%%% ESEMPIO USO METODI DELLA CLASSE %%%

% Creazione di una istanza della classe Wiimote
wiimote = Wiimote();
% Connessione al primo Wiimote disponibile
wiimote.Connect();
% Aggiorna la struttura dati associata all'accelerometro (POLLING)
wiimote.GetAccelState();
% Costruisce una variabile Z ed assegna ad essa il valore contenuto
% nel campo Z della struttura Accel (ovvero la il valore di accelerazione
% dell'asse z del Wiimote
Z = wiimote.Accel.Z;
% Accende i leds nella configurazione 0101 (off - on - off - on)
wiimote.SetLEDs(0, 1, 0, 1);
% Visualizza il valore memorizzato in Z
fprintf('Asse z = %d\n', Z);
```

Le operazioni più comuni che interessano la gestione di un controller Wiimote vengono ridefinite in funzioni "globali" (il cui codice è contenuto in omonimi files *".m"*). In questo modo risulta immediata la creazione di programmi che sfruttano il dispositivo, riducendo sia il tempo necessario per lo sviluppo di un progetto, che la possibilità di introdurre accidentalmente degli errori di programmazione. Di seguito, a titolo di esempio, si riporta il codice della funzione globale *getWiimoteAccel()*:

```
function [X Y Z] = getWiimoteAccel ()
% usage:    getWiimoteAccel ()
% purpose:  Gets the Acceleration values from the wiimote
% return:   Returns a three element array with the following entries
%           index:        1 2 3
%           element:    [ X Y Z ]

global wiimote;

% update the wiimote state values
wiimote.GetAccelState();

% assign outputs
X = wiimote.Accel.X;
Y = wiimote.Accel.Y;
Z = wiimote.Accel.Z;

end
```

La funzione chiama il metodo *GetAccelState()* della *"Wiimote Class"*, che si occupa di aggiornare i valori relativi alle accelerazioni misurate dal Wiimote, contenuti nella *property "Accel"*. Restituisce infine tali valori attraverso un vettore di tre elementi.
Di seguito viene riportato l'elenco delle funzioni globali usate in circostanza di tesi. Tali funzioni permettono di gestire la comunicazione tra Wiimote e PC Host, l'acquisizione dei dati provenienti dal Wiimote e la loro elaborazione; esse sono:

- *initializeWiimote()* - Stabilisce la connessione con il Wiimote. Questa funzione deve essere chiamata una sola volta prima di tutte le altre.
- *disconnectWiimote()* - Disconnette il Wiimote. Questa funzione deve essere sempre chiamata prima di terminare il programma.

- *isButtonPressed('button')* - Controlla se il pulsante *button* passato come argomento (stringa) è attualmente premuto. Il valore del parametro *button* può corrispondere ad una delle seguenti stringhe:
  A, B, UP, RIGHT, DOWN, LEFT, PLUS, MINUS, HOME, ONE, TWO.

- *getWiimoteButtons()* - Ritorna un *array* di 11 elementi che contiene lo stato degli undici pulsanti del Wiimote (1 o *true* se è premuto, 0 o *false* se non è premuto). L'ordine dei pulsanti all'interno dell'*array* è il seguente:
  [A, B, UP, RIGHT, DOWN, LEFT, PLUS, MINUS, HOME, ONE, TWO]
  Quindi se la funzione restituisce il seguente array $[1,0,1,0,0,0,0,0,0,0,0]$ significa che sono stati premuti contemporaneamente i pulsanti 'A' ed 'UP' (la funzione è utile nel caso si voglia sapere se sono premuti più di un pulsante in contemporanea).

- *waitForButtonPress('button')* - Ferma il programma e rimane in attesa che venga premuto il pulsante *button* passato come argomento. Il parametro *button* è una stringa scelta tra le 11 parole già viste.

- *getWiimoteBattery()* - Restituisce un valore decimale indicante il livello di carica della batteria (in percentuale).

- *getWiimoteAccel()* - Restituisce un vettore di 3 elementi contenente i valori dell'accelerazione misurata sui tre assi dell'accelerometro. L'ordine dei valori all'interno dell'*array* è il seguente: *[X, Y, Z]*.
  Se la funzione restituisce l'array $[-2,0,1]$ significa che sull'asse x agisce un'accelerazione pari a $-2g$, sull'asse y un'accelerazione nulla, mentre sull'asse z un'accelerazione pari a g. Questa funzione è alla base dell'intero sistema di acquisizione considerato in questa tesi e consente di campionare l'accelerazione misurata dal sensore interno al Wiimote.

- *setWiimoteLEDs('led1','led2','led3','led4')* - Accende i led del Wiimote secondo la configurazione impostata tramite i parametri booleani *led1*, *led2*, *led3*, *led4* (possono essere un intero 1/0 oppure un valore booleano *true/false*). Il valore 1 accende il led, il valore 0 lo spegne. Il parametro *led1* si riferisce al led all'estrema sinistra del Wiimote mentre *led4* a quello all'estrema destra.

- *setWiimoteRumble('on')* - Accende/Spegne la vibrazione del Wiimote a seconda del parametro *on* passato. Il valore di *on* può essere 1 o 0 (oppure *true/false*), rispettivamente 1 abilita la vibrazione, 0 la disabilita.

Per l'elenco completo delle funzioni globali disponibili si rimanda alla documentazione del progetto *WiiLAB* [32]. Un esempio di codice che fa uso di alcune delle funzioni globali descritte è riportato di seguito: il programma inizializza il Wiimote ed attende la pressione del pulsante *A*, a questo punto a seconda che il Wiimote sia tenuto con i tasti verso il basso o verso l'alto viene attivata o rispettivamente disattivata la vibrazione. Il programma attende quindi la pressione del tasto *HOME* per disconnettere il controller e terminare l'esecuzione.

```
% crea l'oggetto corrispondente al Wiimote connesso ed inizializza la
% connessione
initializeWiimote();
%Attende che l'utente premi il tasto A del Wiimote
waitForButtonPress('A');
%Misura l'accelerazione rilevata dal sensore e la salva in un vettore
[X Y Z] = getWiimoteAccel();
%Se l'accelerazione sull'asse z è negativa (Wiimote capovolto)
if (Z < 0)
    %accende la vibrazione del Wiimote
    setWiimoteRumble(1);
else        %altrimenti
    %spegne la vibrazione del Wiimote
    setWiimoteRumble(0);
end
%Attende che l'utente premi il tasto HOME del Wiimote
waitForButtonPress('HOME');
%Disconnette il Wiimote (ed interrompe la vibrazione)
disconnectWiimote();
```

4.2.3 Un esempio di acquisizione con Matlab

In questo semplice esempio, corrispondente alla funzione Matlab *acquire(Ts,T)*, viene descritta la procedura per mezzo della quale acquisire i dati provenienti dall'accelerometro per un tempo *T* (in millisecondi), impostabile dall'utente, campionando con un periodo *Ts* (in millisecondi). I campioni acquisiti, suddivisi in in tre vettori, uno per ogni asse del sensore, vengono riportati in rispettivi grafici che mostrano l'andamento dell'accelerazione nel tempo. A schermo vengono infine stampate alcune informazioni utili ai fini diagnostici, come ad esempio la massima e la minima accelerazione registrate durante l'acquisizione (figura 4.4).

```
>> acquire(10,3000);
Frequenza campionamento dati = 100
Periodo campionamento dati = 10
Durata Acquisizione = 3000
Massima accelerazione asse x = 1.6
Massima accelerazione asse y = 15.2
Massima accelerazione asse z = 28.5
Minima accelerazione asse x = -21.8
Minima accelerazione asse y = -1.6
Minima accelerazione asse z = -0.3
```

Figura 4.4: Esempio output informazioni diagnostiche durante un'acquisizione

Il campionamento viene gestito per mezzo di un ciclo *FOR* nel quale vengono chiamate le funzioni *getWiimoteAccel()*, per la lettura dei dati dell'accelerometro, e *delay(Ts)* per ottenere la giusta temporizzazione. La funzione *delay(Ts)*, appositamente creata, interrompe l'esecuzione del programma per il tempo *Ts*.
Di seguito è riportato un frammento di codice della funzione *acquire()* che chiarisce quanto descritto.

```
%% Creazione vettori dati e tempi
%costruisco la scala dei tempi
timescale = 0:Ts:T;
%crea per ogni asse il vettore dati
data_x = zeros(1,length(timescale));
data_y = zeros(1,length(timescale));
data_z = zeros(1,length(timescale));

%% Acquisizione dei dati
for i = 1:length(timescale)
    [data_x(i), data_y(i), data_z(i)] = getWiimoteAccel();    %acquisisce
    %la funzione pause() non ha una precisione molto elevata per cui
    %si è scritta la funzione delay() che è più precisa e riesce ad
    %avere risoluzioni più elevate di 10ms
    delay(Ts);              %attende per Ts prima di acquisire nuovamente
end

%% Conversione unità di misura da g a m/s^2
data_x = data_x * g_CONST;
data_y = data_y * g_CONST;
data_z = data_z * g_CONST;
```

Una prima semplice prova è stata avviare l'acquisizione e muovere il Wiimote per vedere l'andamento dell'accelerazione nel periodo di tempo considerato. L'esperimento è stato ripetuto più volte variando i parametri T e Ts in gioco: scendendo con Ts sotto i 10ms si sono osservate nei grafici delle ripetizioni dello stesso valore in campioni consecutivi anche nel caso di bruschi movimenti (ad esempio rilascio del controller ad una certa altezza da terra), questo lascia intuire che il minimo periodo di campionamento sia proprio pari a circa 10ms, e quindi la frequenza massima di campionamento sia pari a circa 100Hz. In figura 4.5 è mostrata l'acquisizione di una caduta del Wiimote con conseguente urto su un cuscino (campionamento con periodo 10ms): si notino i momenti del rilascio, la caduta libera (la forza di gravità viene compensata), l'urto ed il conseguente assestamento (l'andamento oscillatorio smorzato è tipico dei sistemi del secondo ordine). Nelle figure 4.6a e 4.6b viene evidenziato l'intervallo di tempo corrispondente alla caduta libera nei due casi di $Ts = 10ms$ (figura 4.6a) e $Ts = 5ms$ (figura 4.6b).

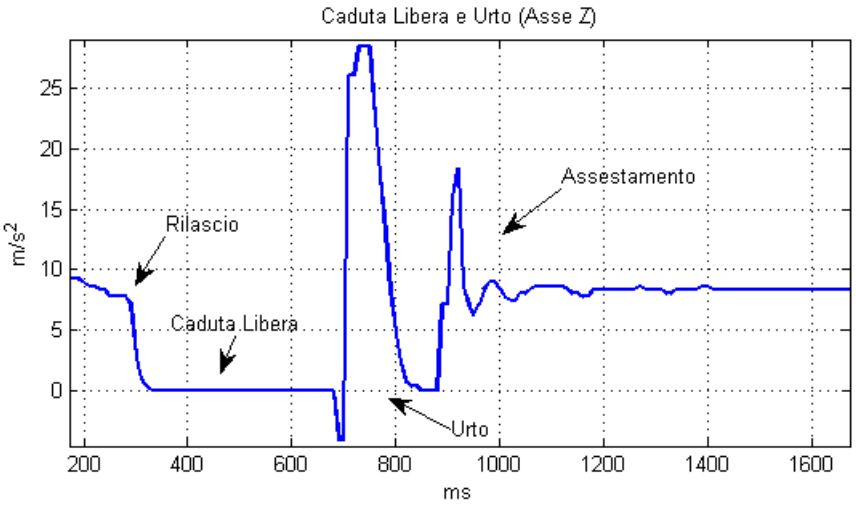
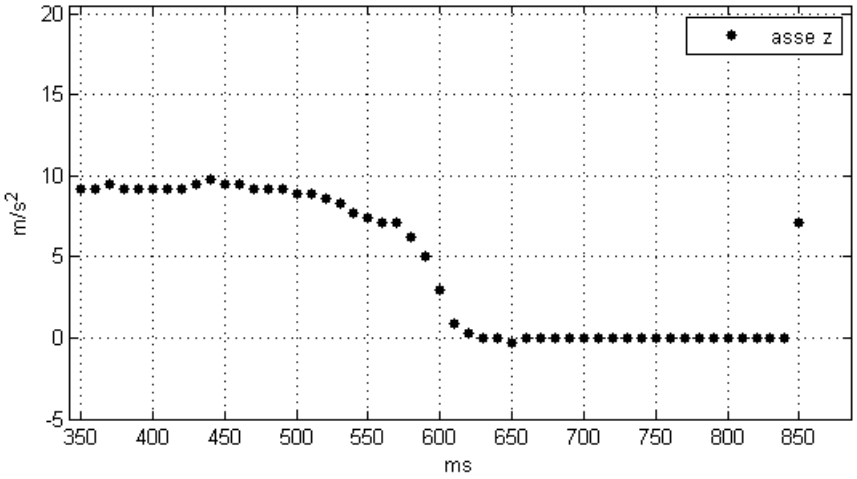
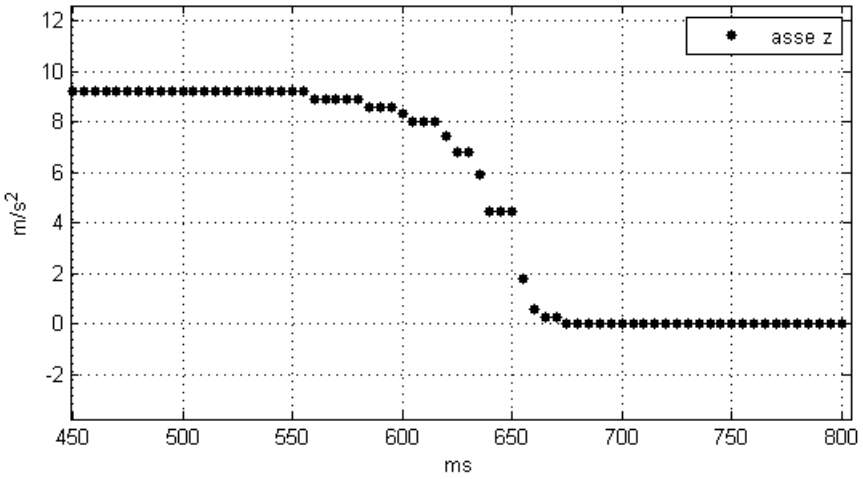


Figura 4.5: Andamento dell'accelerazione con il Wiimote in caduta libera e successivo urto



(a) Periodo di campionamento 10ms



(b) Periodo di campionamento 5ms

Figura 4.6: Intervallo corrispondente alla caduta libera nei casi $T_S = 10ms$ e $T_S = 5ms$

Un ulteriore esperimento fatto, più per divertimento che per un fine utile, è stato misurare, grazie al Wiimote, le pulsazioni cardiache di un collega: il controller è stato posizionato sul torace del volontario, che per qualche secondo ha cercato di trattenere il respiro per rendere il segnale acquisito più pulito. Il risultato della misura è mostrato in figura 4.7.

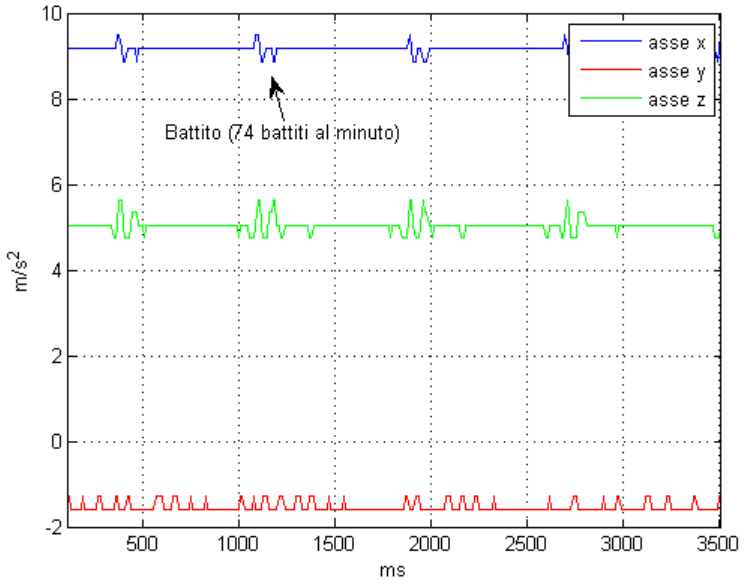


Figura 4.7: Cardiogramma con il Wiimote

Come si può osservare sono raffigurati tutti e tre gli assi, ma quello che consente di individuare il battito del cuore è l'asse X. Semplici calcoli basati sulla distanza temporale tra una pulsazione e l'altra hanno permesso di stimare una frequenza cardiaca di 74 battiti/minuto.

4.2.4 Ulteriori Osservazioni

Per mezzo di altre acquisizioni effettuate grazie alla funzione *acquire()* realizzata, è stato possibile determinare sperimentalmente alcuni parametri caratterizzanti del Wiimote, visto come strumento per la misura di accelerazioni.

Range di misura
Il range delle accelerazioni misurabili dal Wiimote, per un singolo asse, è stato determinato applicando al dispositivo una forte accelerazione lungo la direzione dell'asse considerato (esempio asse X) ed effettuando un'acquisizione, attraverso la funzione *acquire()* realizzata. Nel caso dell'asse X il grafico generato è riportato in figura 4.8. Nel grafico sono mostrati gli andamenti dell'accelerazione sia nel caso in cui la forza applicata al Wiimote risulta concorde all'asse X, sia nel caso in cui la forza applicata risulta discorde (accelerazione negativa).

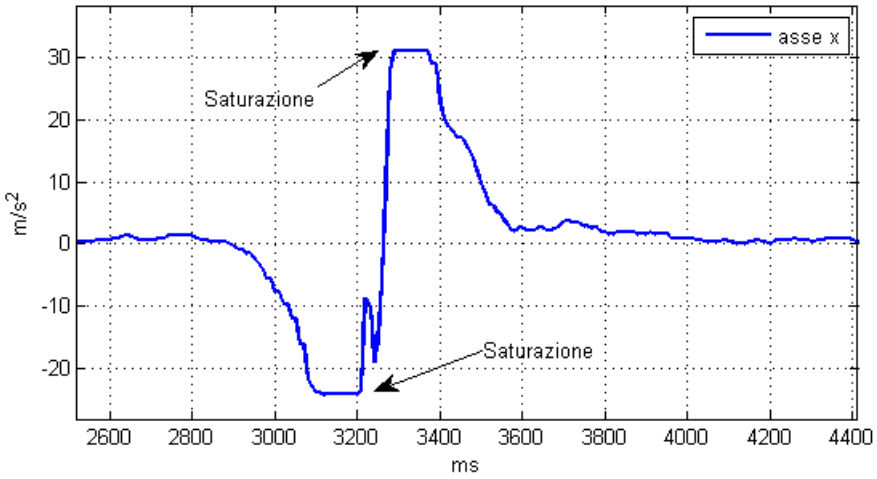


Figura 4.8: Range di misura asse X

La massima accelerazione applicata al Wiimote ha un valore superiore (in modulo) alla massima accelerazione misurabile dal dispositivo, questo è evidenziato nel grafico dai fenomeni di saturazione. I valori massimo e minimo di accelerazione nell'acquisizione effettuata corrispondono quindi agli estremi del range di misura. Nel caso dell'asse X la massima accelerazione misurata risulta essere $a_{x,max} = 31m/s^2$, la minima accelerazione risulta essere $a_{x,min} = -24.4m/s^2$ e quindi il range di misura risulta essere $[a_{x,min}\,,\, a_{x,max}] = [-24.4m/s^2\,,\, 31m/s^2]$.
Le stesse considerazioni vengono fatte per gli assi Y e Z. Nel caso dell'asse Y, massima e minima accelerazione misurate sono rispettivamente $a_{y,max} = 23.4m/s^2$ e $a_{y,min} = -32.9m/s^2$, di conseguenza il range di misura per quest'asse risulta essere $[a_{y,min}\,,\, a_{y,max}] = [-32.9m/s^2\,,\, 23.4m/s^2]$. Nel caso dell'asse Z, massima e minima accelerazione misurate sono rispettivamente $a_{z,max} = 28.5m/s^2$ e $a_{z,min} = -22.9m/s^2$, di conseguenza il range di misura per quest'asse risulta essere $[a_{z,min}\,,\, a_{z,max}] = [-22.9m/s^2\,,\, 28.5m/s^2]$. I grafici ottenuti per gli assi Y e Z vengono riportati rispettivamente nelle figure 4.9 e 4.10.

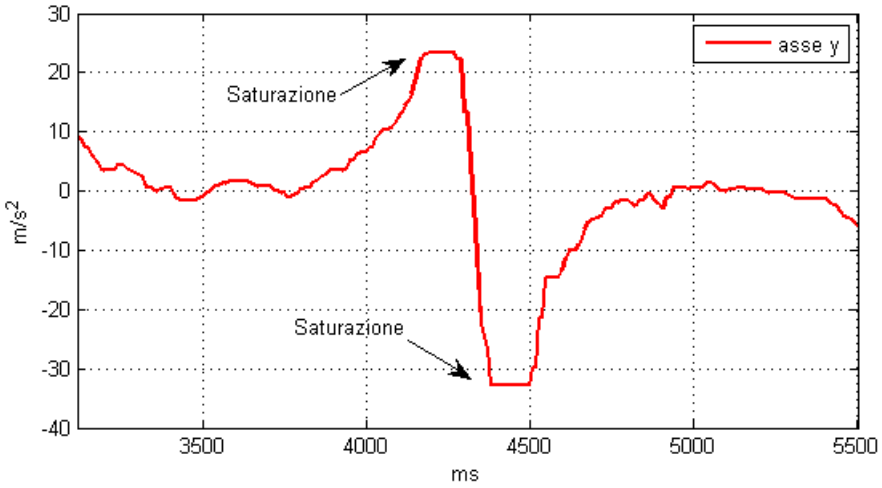


Figura 4.9: Range di misura asse Y

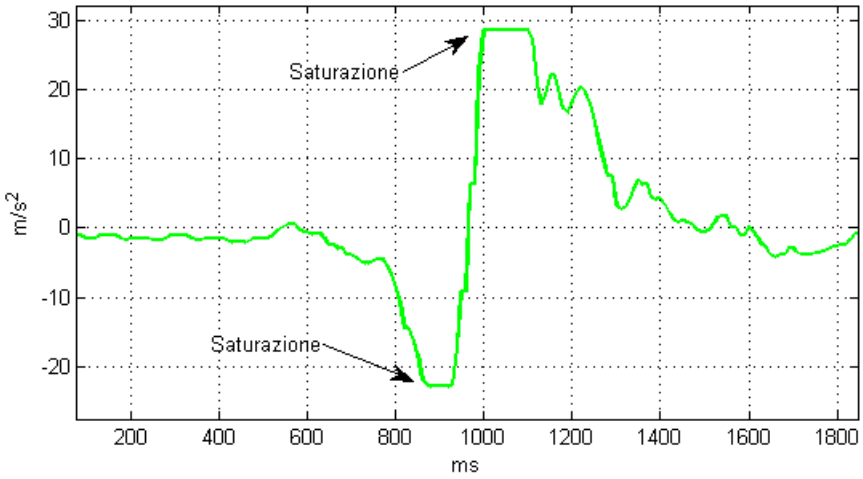


Figura 4.10: Range di misura asse Z

Passo di Quantizzazione

Il passo di quantizzazione corrisponde alla più piccola variazione di accelerazione rappresentabile in una misura effettuata con il Wiimote. Si può determinare il valore del passo di quantizzazione Δq per i singoli assi X, Y e Z del controller tenendo il dispositivo fermo ed effettuando un'acquisizione: gli effetti del rumore e le piccole vibrazioni a cui è soggetto il dispositivo fanno si che le accelerazioni misurate oscillino tra due valori, corrispondenti a due livelli di quantizzazione adiacenti, come mostrato in figura 4.11, nel caso particolare dell'asse Y. Il comportamento per gli assi X e Z è analogo. Dall'analisi dei grafici è possibile osservare che il passo di quantizzazione, dato dalla differenza tra i due valori osservati, è pari a circa $\Delta q \simeq 0.3 m/s^2$ (pari a circa $0.03g$).

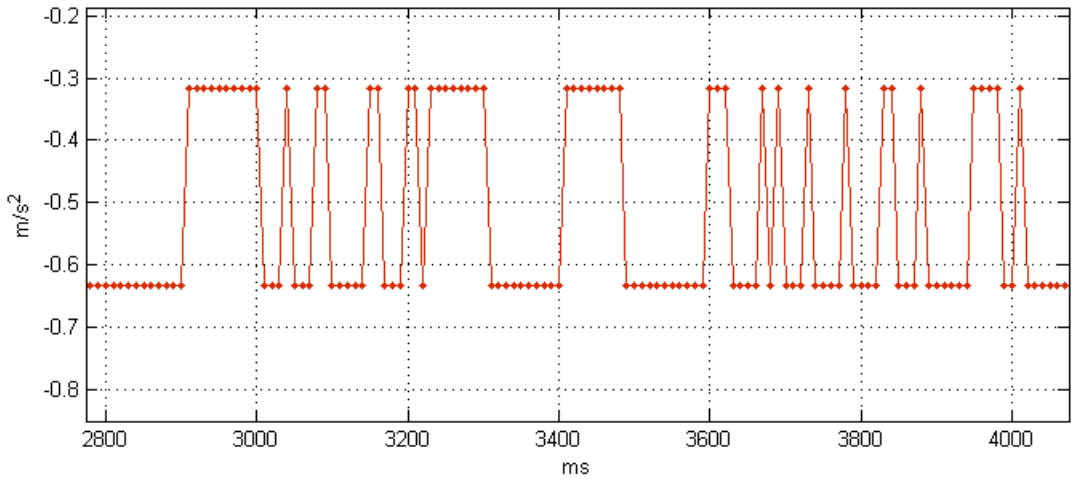


Figura 4.11: L'accelerazione misurata oscilla tra due livelli di quantizzazione a causa del rumore e di piccole vibrazioni

Per stimare il passo di quantizzazione è stato scritto il seguente *script* per Matlab (riportato solo lo spezzone di codice relativo all'asse X):

```
%% Calcolo il passo di quantizzazione sugli assi del Wiimote %%
[acc_x, acc_y, acc_z] = getWiimoteAccel();          %leggo l'accelerazione
g_cost = 9.80665;    %costante accelerazione di gravità convenzionale
while(1)             %% asse x
    [acc2_x, acc2_y, acc2_z] = getWiimoteAccel();   %leggo l'accelerazione
    %% visualizzo l'accelerazione solo se vi è un cambiamento, poi esco
    if (acc_x ~= acc2_x)
        fprintf('passo quantizzazione asse x (g) = %.2f\n', abs(acc_x - acc2_x));
        %conversione in m/s^2-units
        fprintf('passo quantizzazione asse x (m/s^2) = %.1f\n', abs(acc_x - acc2_x)*g_cost);
        break;
    end
```

Viene eseguita una prima misura di accelerazione ed i valori ottenuti dal Wiimote vengono salvati nelle variabili *acc_x*, *acc_y* ed *acc_z*. Successivamente un ciclo *WHILE* esegue ripetute misure di accelerazione e confronta il valore misurato con quello salvato precedentemente. Se i due valori confrontati non coincidono ne viene stampata a schermo la differenza (in valore assoluto) ed il ciclo termina. Se sul Wiimote agiscono solo piccole vibrazioni il valore visualizzato corrisponde al passo di quantizzazione cercato. In figura 4.12 viene riportato l'*output* del programma.

```
passo quantizzazione asse x (g) = 0.03
passo quantizzazione asse x (m/s^2) = 0.3
passo quantizzazione asse y (g) = 0.03
passo quantizzazione asse y (m/s^2) = 0.3
passo quantizzazione asse y (g) = 0.03
passo quantizzazione asse z (m/s^2) = 0.3
```

Figura 4.12: Informazioni stampate a schermo

Incertezza dovuta alla quantizzazione
Si può determinare la componente di incertezza dovuta alla quantizzazione mediante la relazione [29]:

$$u_{acc} = \frac{\Delta q}{\sqrt{12}} \simeq 87 \cdot 10^{-3} m/s^2 \tag{4.2.1}$$

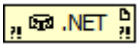
dove u_{acc} è l'incertezza standard, dovuta al processo di quantizzazione, associata alla misura di accelerazione e Δq è il passo di quantizzazione.
L'incertezza estesa U_{acc} associata alla misura risulta inoltre essere [29]:

$$U_{acc} = u_{acc} \cdot k = u_{acc} \cdot \sqrt{3} = \frac{\Delta q}{2} \simeq 0.15 m/s^2 \tag{4.2.2}$$

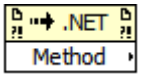
dove $k = \sqrt{3}$ è il fattore di copertura (distribuzione di probabilità uniforme). I valori misurati possono essere espressi nella forma $a_{mis} \pm U_{acc}\ [m/s^2]$.

4.2.5 WiiLAB e LABVIEW

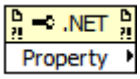
L'ambiente di programmazione *Labview* [26] permette di comunicare con le librerie scritte in linguaggio *.Net*, nella *functions palette* del software sono presenti infatti dei *nodi* che permettono di interagire in modo semplice con *oggetti .Net*, i più importanti sono [27]:

- ***Constructor Node***: 
  Il nodo crea una nuova istanza di un oggetto *.Net*. Nell momeno in cui il nodo viene inserito

all'interno del *block diagram* appare una finestra dove viene chiesto il percorso del file *.dll* della libreria da utilizzare. Il *constructor node* fornisce in uscita il *riferimento* (*reference*, *refnum* o *handle*) dell'oggetto istanziato, che viene usato successivamente per invocarne i metodi.

- ***Invoke Node***: 

Il nodo invoca un metodo dell'oggetto *.Net* passato come riferimento. Il metoto da invocare è selezionabile cliccando sulla scritta *"Method"* dell'*invoke node*. Possono essere passati dei parametri in ingresso al metodo e vengono restituiti i parametri in uscita. Sia i dati in ingresso che in uscita possono essere tipi di dati standard (*int*, *float*, *double*...), gestiti anche da *Labview*, oppure strutture dati definite nella libreria.

- ***Property Node***: 

Il nodo permette di leggere o scrivere le *properties* dell'oggetto *.Net* passato come riferimento (ad esempio una struttura dati contenente le informazioni volute). La *property* sulla quale effettuare l'operazione è selezionabile cliccando sulla scritta *"Property"* del *property node*. La modalità lettura oppure scrittura è impostabile attraverso il menù contestuale del nodo, visualizzabile facendo click con il tasto destro del mouse su di esso. I dati da leggere/scrivere possono essere di tipo standard oppure strutture dati definite nella libreria.

- ***Close Reference***: 
Il nodo permette di chiudere il riferimento associato all'oggetto *.Net* (passato come riferimento). Viene utilizzato per eliminare i riferimenti ad un oggetto non più utilizzato, che può quindi essere scaricato dalla memoria.

Per poter comunicare con il Wiimote si può quindi interagire con la libreria *WiiLAB* direttamente all'interno di Labview. La scelta della libreria *WiiLAB* è dovuta al suo più immediato utilizzo rispetto alla *WiimoteLib*, inoltre è possibile mantenere un filo logico con quanto già detto nella sottosezione 4.2.2.
Le operazioni principali da eseguire per sfruttare la libreria *WiiLAB* con Labview verranno illustrate attraverso un esempio: è stato realizzato un semplice programma che permette la visualizzazione in *real-time* dei dati provenienti dal Wiimote, vengono visualizzati i valori di accelerazione misurati dall'accelerometro e lo stato dei pulsanti del controller. La comunicazione con la libreria avviene grazie ai *nodi* descritti sopra. In figura 4.13 è riportato il codice dell'applicazione.

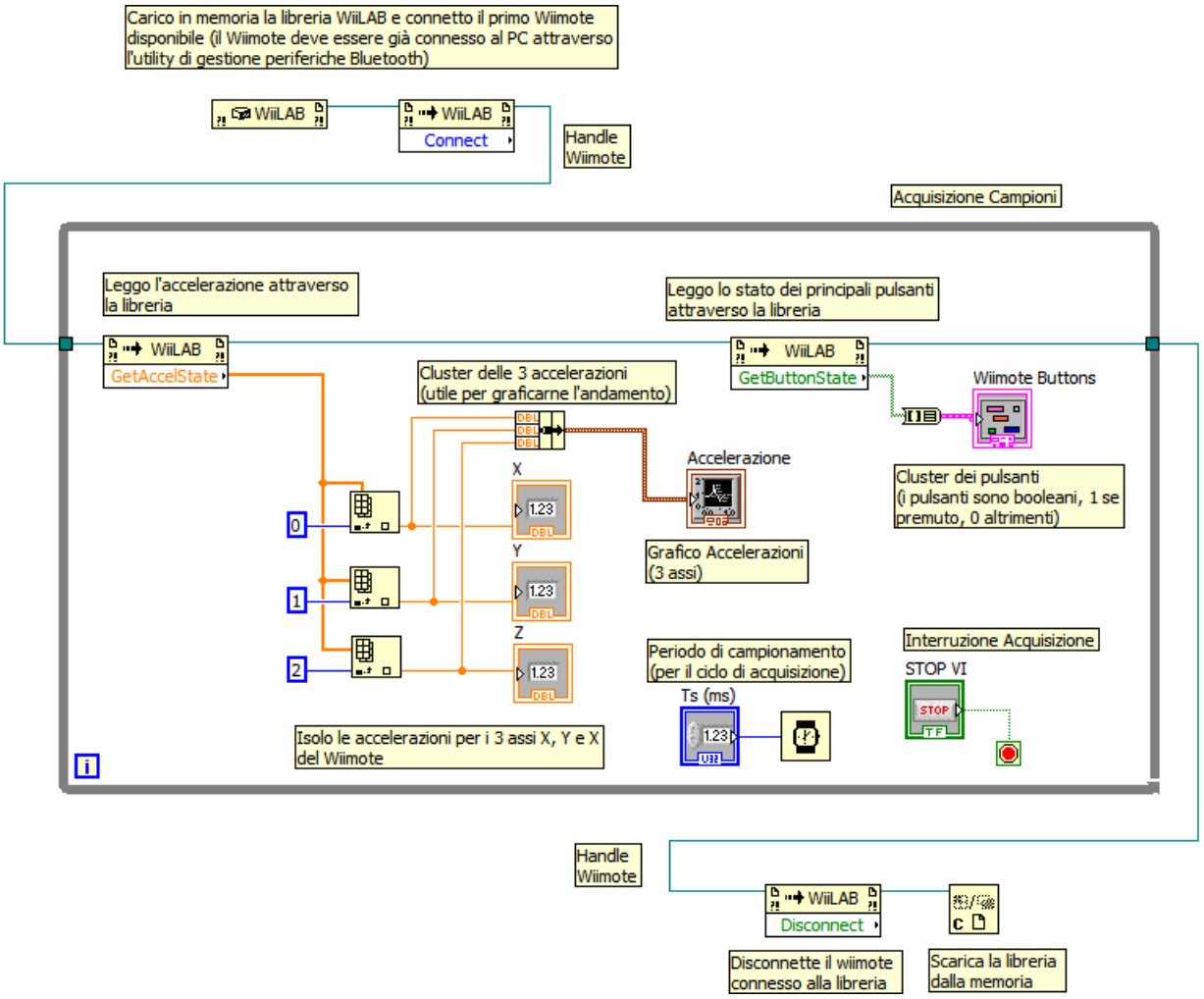


Figura 4.13: Codice per l'acquisizione dei dati dal Wiimote

Prima di acquisire i dati del dispositivo è necessario creare un'istanza dell'oggetto *WiiLAB* associato alla libreria attraverso il *Constructor Node* ed invocare il metodo *Connect()* dell'oggetto per effettuare la connessione con il Wiimote associato al PC (il controller deve già essere connesso con il computer). Successivamente è possibile acquisire i dati del controller attraverso i metodi *GetAccelState()* e *GetButtonState()* già visti, sfruttando gli *Invoke Node*. Nel codice proposto l'acquisizione dei campioni avviene all'interno di un ciclo *WHILE*: attraverso un primo *Invoke Node* viene invocato il metodo *GetAccelState()* che restituisce un'array di 3 elementi di tipo *double* rappresentanti, in ordine, l'accelerazione misurata sugli assi X, Y e Z del controller. I valori vengono visualizzati in forma numerica attraverso tre *numeric indicator* ed in forma grafica attraverso un *chart*. Per mezzo di un secondo *Invoke Node* viene invocato il metodo *GetButtonState()* che restituisce un'array di valori *booleani* rappresentanti lo stato (premuto/non premuto) dei singoli pulsanti del controller. Lo stato dei pulsanti viene visualizzato attraverso un insieme di led indicatori (*cluster*). Il ciclo è temporizzato e viene eseguita una iterazione ogni *Ts* millisecondi: l'acquisizione dei dati del Wiimote avviene con periodo di campionamento *Ts (ms)*. Al termine dell'acquisizione (pressione pulsante *"STOP VI"*) il controller viene disconnesso invocando il metodo *Disconnect()* e l'oggetto *WiiLAB* viene scaricato dalla memoria attraverso il nodo *Close Reference*. In figura 4.14 viene riportata l'interfaccia grafica associata al codice descritto, gli indicatori presenti mostrano i dati provenienti dal Wiimote durante un'acquisizione di prova.

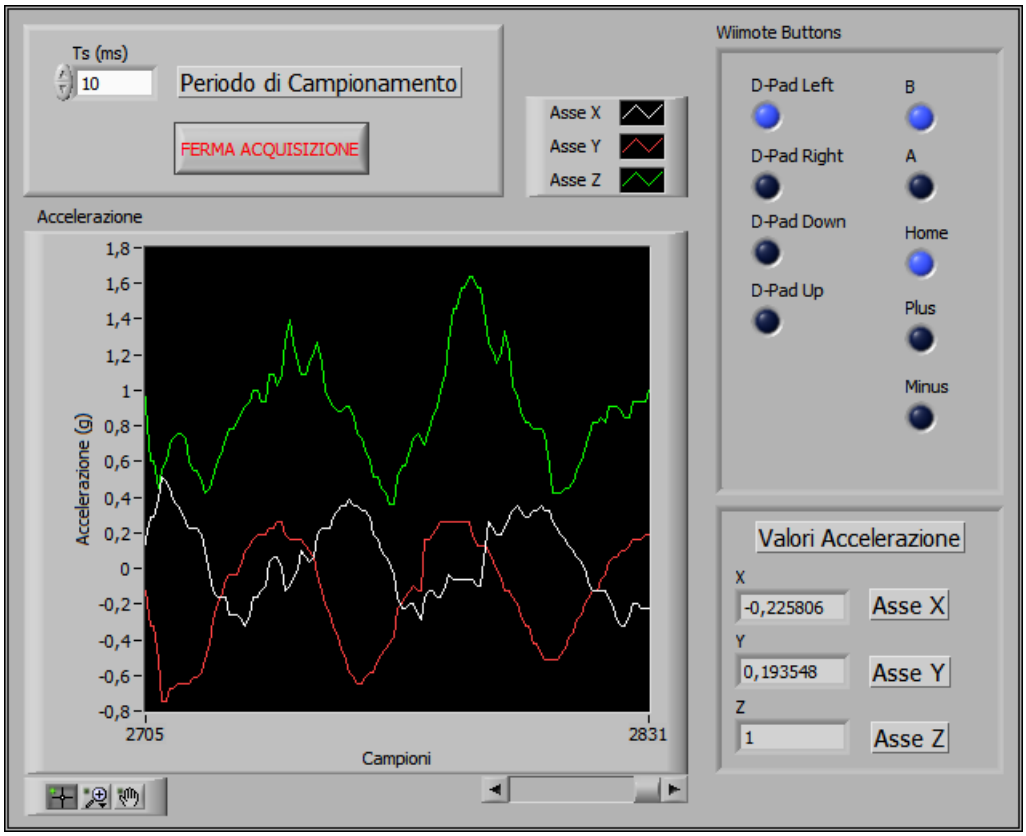


Figura 4.14: Interfaccia grafica associata al codice di figura4.13

Le operazioni da compiere per completare l'acquisizione dati dal Wiimote possono essere divise in tre *"blocchi funzionali"*:

1. **Inizializzazione del controller:** corrisponde alle operazioni che precedono l'acquisizione dei dati, ovvero l'istanziazione dell'oggetto *WiiLAB* e la connessione con il dispositivo. Nel *block diagram* di figura 4.13 è considerata la parte codice precedente al ciclo *WHILE* (in alto nell'immagine).

2. **Acquisizione campione:** corrisponde all'acquisizione dei valori correnti di accelerazione misurati dal controller e dello stato attuale dei pulsanti. Nel *block diagram* di figura 4.13 è considerata la parte codice interna al ciclo *WHILE*, esclusi gli indicatori.

3. **Disconnessione del controller:** corrisponde alle operazioni successive all'acquisizione dei dati (intesa come acquisizione di una sequenza di campioni), ovvero la disconnessione del dispositivo e la chiusura del riferimento associato all'oggetto WiiLAB. Nel *block diagram* di figura 4.13 è considerata la parte codice successiva al ciclo *WHILE* (in basso nell'immagine).

Per dividere i tre "Blocchi Funzionali" e rendere il codice più leggibile e meglio interpretabile sono stati creati 3 Sub-VI:

1. Sub-VI *InitWii*: si occupa dell'inizializzazione del controller.
   In figura 4.15 mostrato il codice del *VI*, mentre in figura 4.16 è raffigurata l'icona associata al *VI* con relativo *connector pane*.

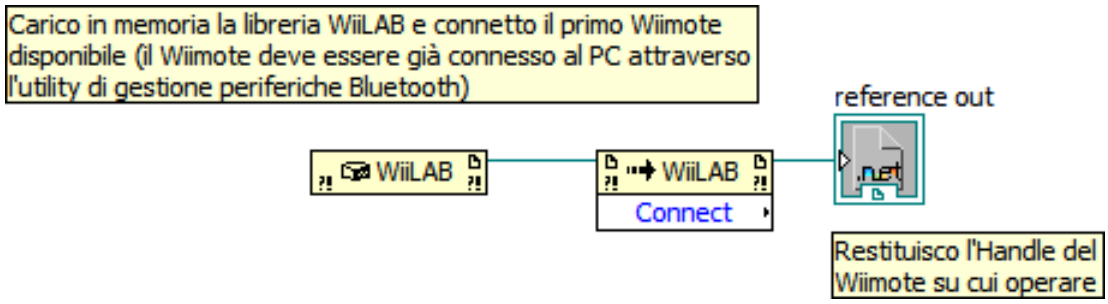


Figura 4.15: Codice del Sub-VI *InitWii*

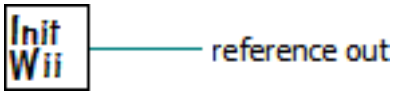


Figura 4.16: Icona e pannello connettori *InitWii*

2. Sub-VI *ACQWii*: si occupa dell'acquisizione di un singolo campione.
   In figura 4.17 mostrato il codice del *VI*, mentre in figura 4.18 è raffigurata l'icona associata al *VI* con relativo *connector pane*.

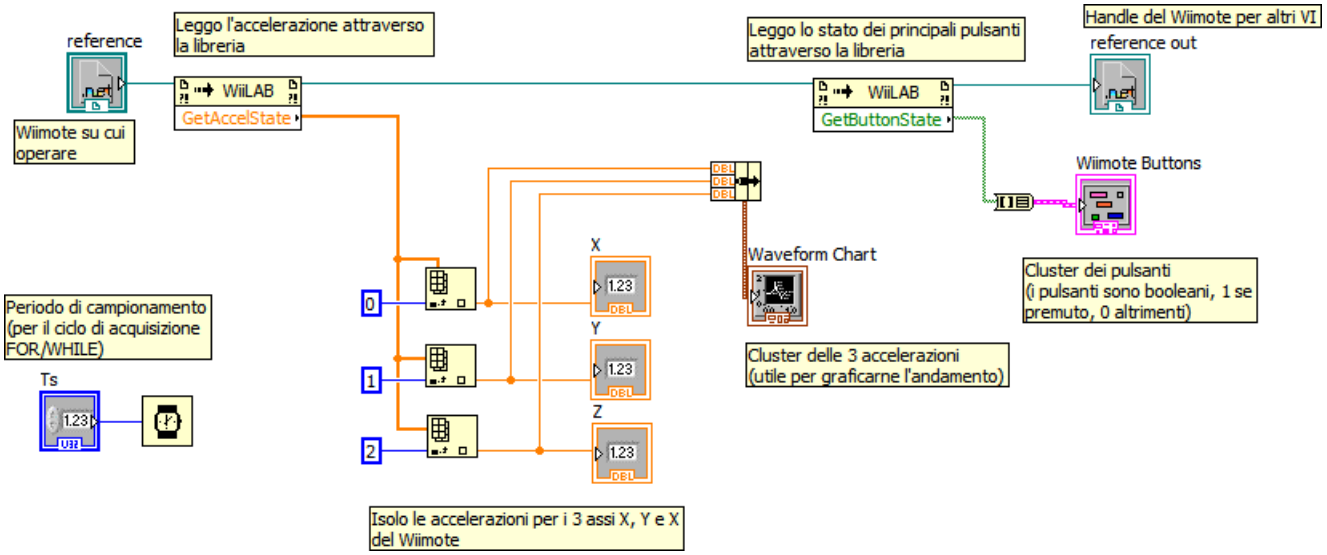


Figura 4.17: Codice del Sub-VI *ACQWii*

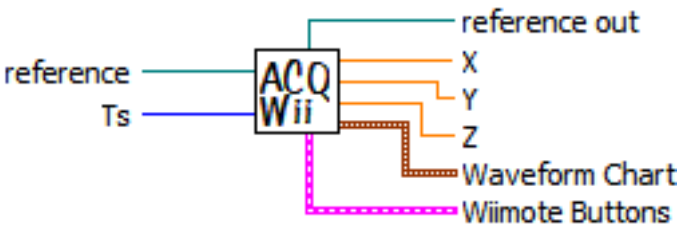


Figura 4.18: Icona e pannello connettori *ACQWii*

3. Sub-VI *DisconnectWii*: si occupa della disconnessione del controller.
   In figura 4.19 mostrato il codice del *VI*, mentre in figura 4.20 è raffigurata l'icona associata al *VI* con relativo *connector pane*.

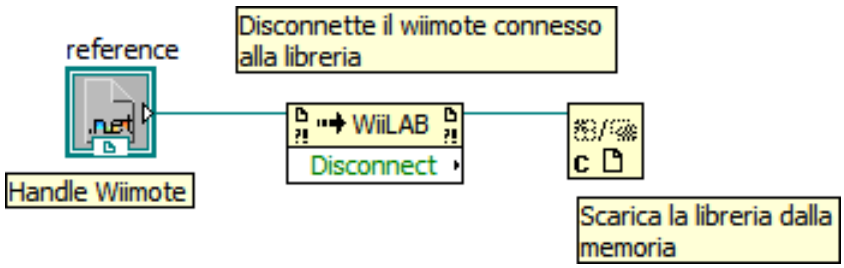


Figura 4.19: Codice del Sub-VI *DisconnectWii*

Figura 4.20: Icona e pannello connettori *DisconnectWii*

Il codice proposto in figura 4.13 è stato riscritto utilizzando i Sub-VI realizzati, il risultato è riportato in figura 4.21.

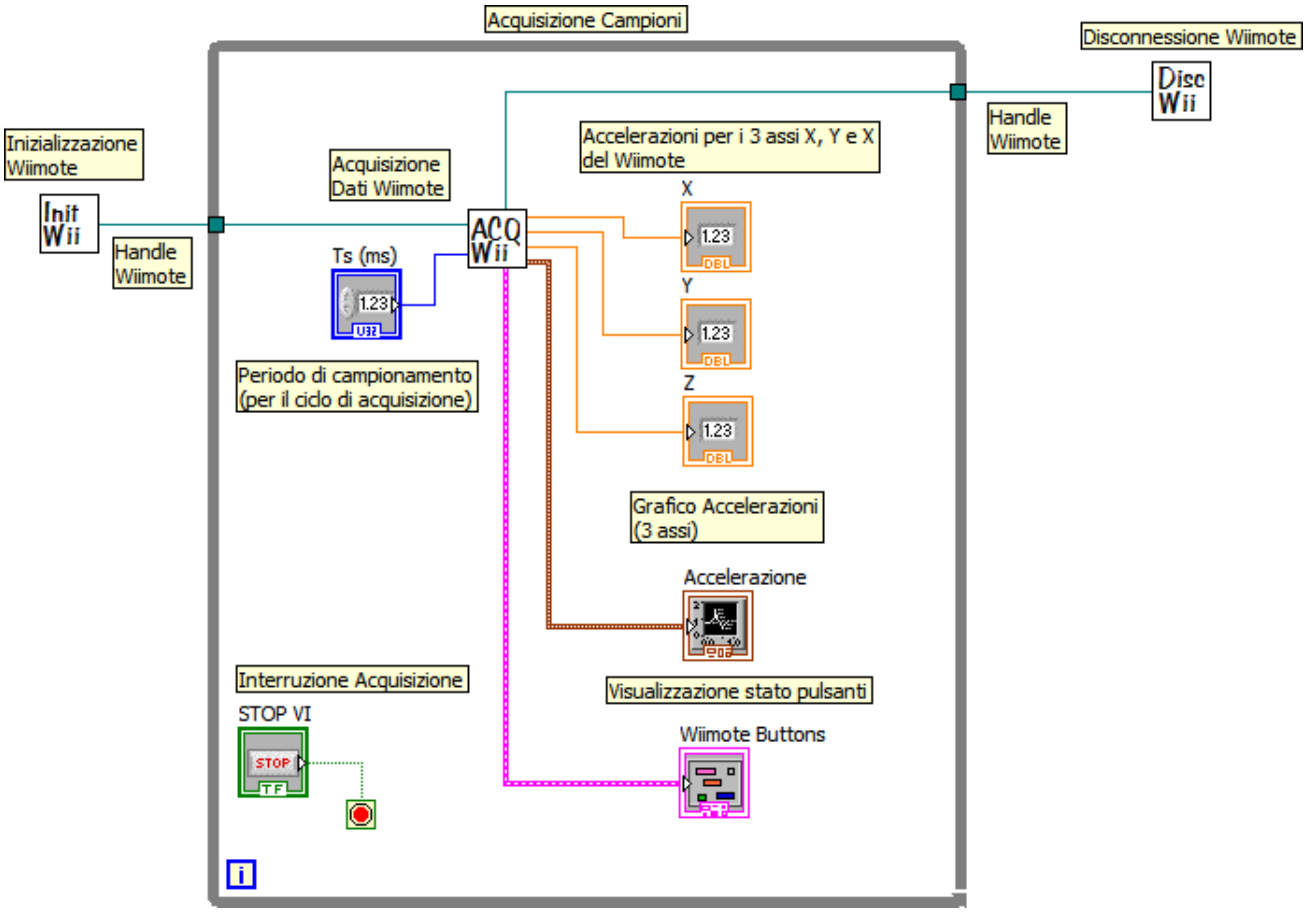


Figura 4.21: Codice per l'acquisizione dei dati dal Wiimote attraverso l'uso dei Sub-VI *InitWii*, *ACQWii* e *DisconnectWii*

4.2.6 Un esempio di acquisizione con Labview

Viene ora analizzato un programma di esempio che permette di acquisire i valori di accelerazione misurati dal Wiimote per un tempo T_W (finestra acquisizione) campionando con frequenza $F_S = 1/T_S$, dove T_S è il periodo di campionamento. L'applicazione, successivamente all'acquisizione, visualizza in un unico grafico i tre segnali di accelerazione corrispondenti all'andamento dell'accelerazione per i tre assi X, Y e Z del controller durante il tempo T_W. In figura 4.22 è riportata l'interfaccia grafica dell'applicazione realizzata.

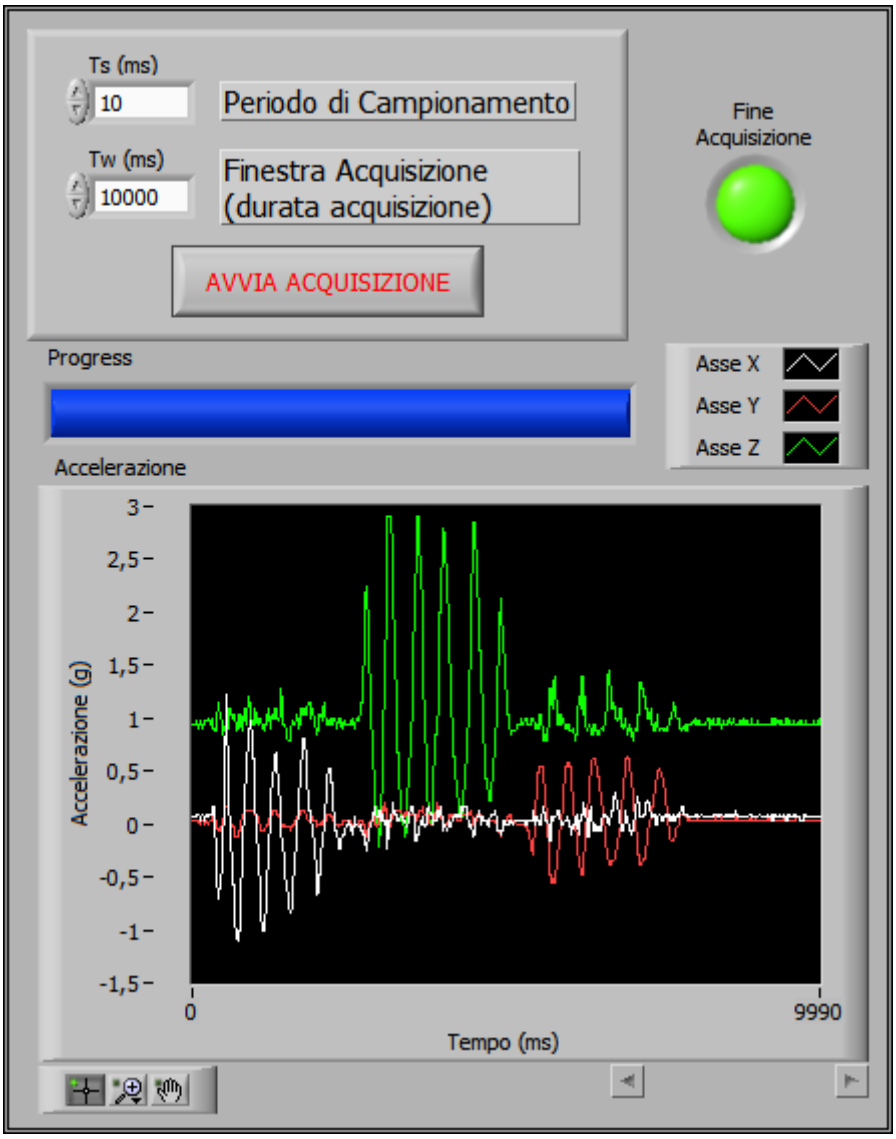


Figura 4.22: Interfaccia grafica programma acquisizione accelerazione

La procedura per eseguire un'acquisizione è molto semplice:

1. Connettere il Wiimote al PC ed eseguire l'applicazione.
2. Inserire il periodo di campionamento T_S voluto (di default $10ms$).
3. Inserire la durata dell'acquisizione T_W voluta (di default $5000ms = 5s$).
4. Premere il pulsante *"Avvia Acquisizione"*.

5. Attendere che la barra *"Progress"* arrivi fino in fondo. Durante questo intervallo di tempo il programma esegue l'acquisizione dei dati, l'avanzamento della barra indica il tempo rimanente alla fine dell'operazione. Quando l'acquisizione è terminata si accende anche il led *"Fine Acquisizione"*.

6. I campioni ricevuti vengono visualizzati nel grafico *"Accelerazione"*: nel grafico viene rappresentato l'andamento dell'accelerazione, in unità *g*, al variare del tempo, in unità *ms*.

7. Il grafico risultante può eventualmente essere esportato come file immagine (opzione *"export simplified Image"* dal menù contestuale).

In figura 4.23 viene riportato un grafico, generato con l'applicazione realizzata, che mostra la caduta del Wiimote da qualche decina di centimetri di altezza ed il conseguente urto su un cuscino, in analogia con immagine 4.5 ottenuta con Matlab.

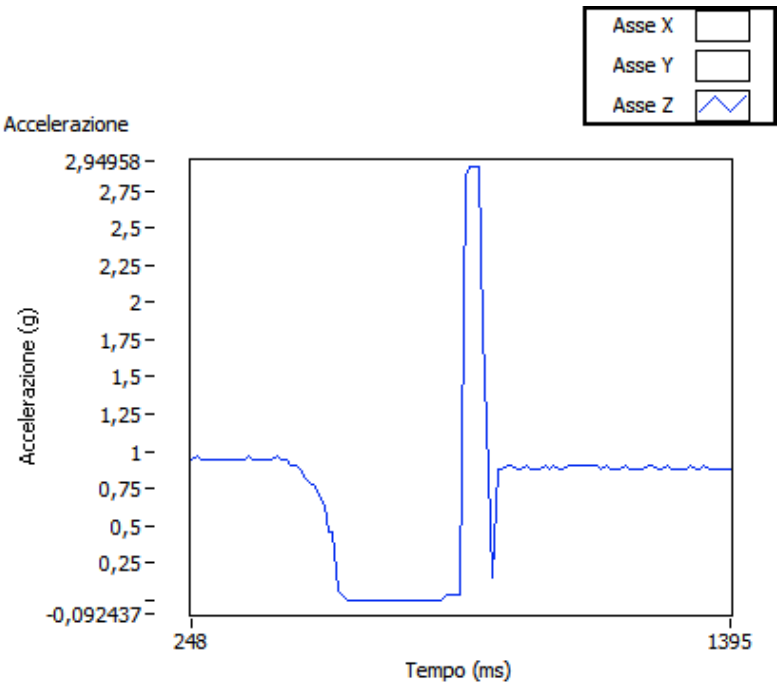


Figura 4.23: Caduta libera e urto con Labview

A differenza del programma proposto in figura 4.21, l'acquisizione avviene attraverso un ciclo FOR, infatti si conosce già il numero di campioni da acquisire: dati T_W e T_S risulta $N = T_W/T_S$. Il pulsante *"Avvia Acquisizione"* (*START*) è associato ad una struttura *CASE*, quando viene premuto il codice interno alla struttura viene eseguito: il Wiimote viene inizializzato, viene poi effettuata l'acquisizione ed infine il dispositivo viene disconnesso ed i dati acquisiti vengono rappresentati graficamente. Il ciclo *FOR* si occupa di generare i vettori contenenti i campioni acquisiti (un vettore per ogni asse del controller) ed il vettore contenente la scala dei tempi, per la successiva visualizzazione su grafico *XY Graph*. Il codice dell'applicazione è riportato in figura 4.24, si noti l'uso dei Sub-VI esaminati nella sottosezione 4.2.5. Alla base delle applicazioni che verranno realizzate nel capitolo 5 vi sarà una struttura per l'acquisizione dei dati simile a quella raffigurata.

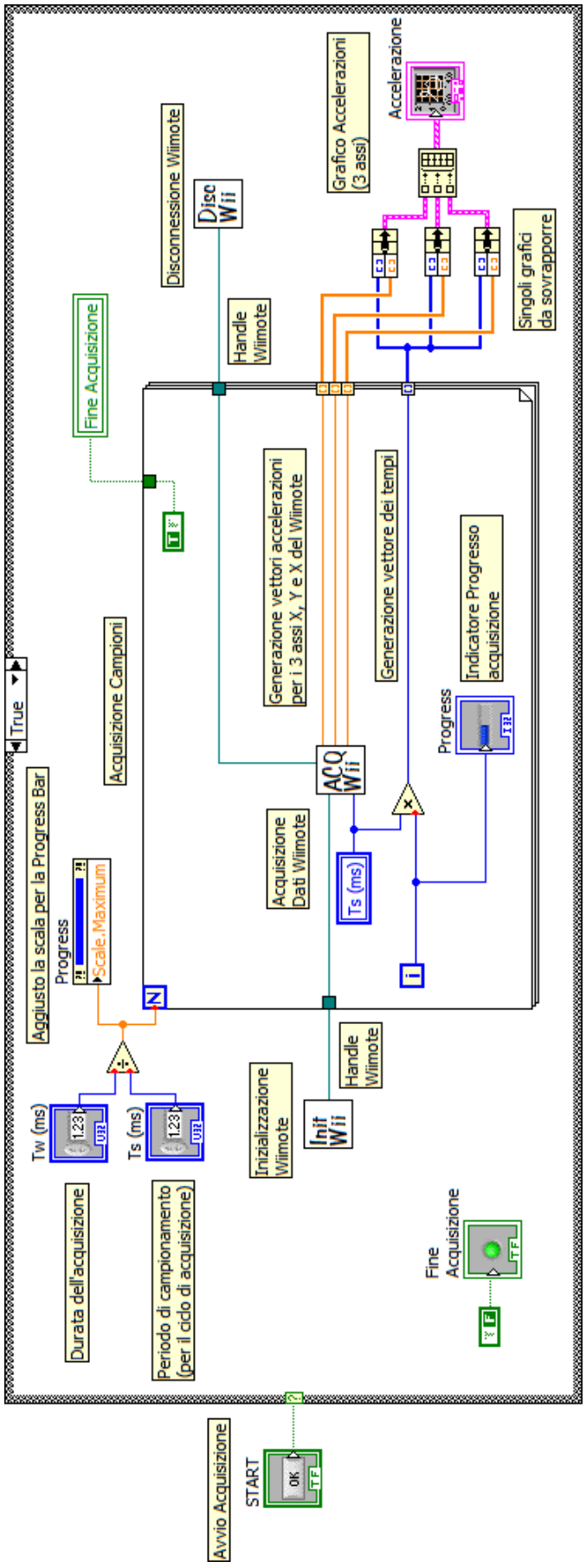


Figura 4.24: Codice programma acquisizione accelerazione

4.3 Il sistema di acquisizione: riassunto

Riassumendo, per la gestione di un controller Wiimote mediante PC sono necessari:

1. Un Wiimote.
2. Un PC con sistema operativo Windows, consigliato Windows XP per evitare problemi di compatibilità.
3. Un adattatore Bluetooth (esterno o integrato) compatibile con lo *stack* Microsoft o con lo *stack* Windcomm.
4. Le librerie *WiimoteLib.dll* (versione modificata) e *WiiLAB.dll.*
5. Il software *Matlab* (consigliata versione 2008 o superiore per la compatibilità con *WiiLAB*) e la classe Matlab *"Wiimote Class*" per la comunicazione con la libreria WiiLAB.
6. Il software *Labview* (consigliata versione 2008 o superiore) ed i *VI* realizzati per l'acquisizione dei dati dal Wiimote.

Per avere un sistema di acquisizione funzionante si devono infine effettuare alcune semplici operazioni, in modo tale che il Wiimote venga correttamente identificato dal computer ed associato (*pairing*):

1. Avviare il software per la ricerca dei dispositivi Bluetooth.
2. Premere e tenere premuti entrambi i pulsanti "1" e "2" sul Wiimote. Questa operazione forza il Wiimote in modalità *"Discovery"* e lo rende rilevabile dal PC (grazie al *Service Discovery Protocol*). Durante questa fase si vedranno i led lampeggiare.
3. Avviare la ricerca dei dispositivi Bluetooth raggiungibili, sempre tenendo premuti i pulsanti sul Wiimote.
4. Il Wiimote dovrebbe comparire nella lista delle periferiche come *"Nintendo RVL-CNT-01"*, scegliere di connettersi al dispositivo SENZA inserire il PIN, se richiesto il PIN lasciare il campo vuoto. Normalmente è sufficiente cliccare con il tasto destro sulla periferica e scegliere la voce "connetti". Se fosse richiesto quale servizio usare per la periferica si deve scegliere il servizio *tastiera/mouse/HID*.
5. Terminata la connessione si possono rilasciare i pulsanti del Wiimote. Se tutto è andato a buon fine i led continueranno a lampeggiare.

Ora il Wiimote è associato al PC, ma per poterlo utilizzare in *Matlab* o *Labview* bisogna predisporre la directory di lavoro, il cui contenuto deve essere:

- I files *WiimoteLib.dll* e *WiiLab.dll*;
- I files **.m* che costituiscono la classe *Wiimote* di *Matlab* per la comunicazione con il Wiimote e le varie funzioni globali;
- Eventuali altri files *.m che definiscono le funzioni personalizzate create dall'utente.

Per poterlo utilizzare in *Labview* il cui contenuto della directory deve essere:

- I files *WiimoteLib.dll* e *WiiLab.dll*;

- I files **.vi* realizzati per effettuare l'acquisizione dei dati dal Wiimote;
- Gli altri files *.vi creati dall'utente per la realizzazione dell'applicazione.

Dopo queste operazioni il sistema è pronto per essere utilizzato.
Il sistema di acquisizione usato per le misure proposte in questo scritto (figura 4.25) è composto da un Tablet PC *HP tc-1100* dotato di processore Intel Centrino M, stack Bluetooth integrato HP compatibile Windcomm, sistema operativo Windows XP, Matlab R2009b (Versione 7.9.0), Labview 2009 (versione 9.0).

Figura 4.25: Il sistema di acquisizione usato

Capitolo 5

ESPERIMENTI DI FISICA E LABVIEW

Obiettivo di questa sezione è la realizzazione di semplici programmi in Labview che mostrano delle possibili applicazioni del controller Wiimote a fini didattici. Nella realizzazione dei programmi si focalizzerà l'attenzione sulla creazione di un'interfaccia grafica *user-friendly*, chiara ed intuitiva, in modo da consentire allo studente di eseguire l'esperimento seguendo alcuni semplici *step* e visualizzare quindi i risultati conseguiti, i quali devono essere rappresentati in modo tale da garantire un'immediata comprensione.
Per raggiungere lo scopo i programmi sono stati tutti realizzati con la stessa struttura:

- Titolo e descrizione sulle modalità di esecuzione dell'esperimento.
- *TabControl* organizzato in vari TABS numerati in base agli step da eseguire per il completamento dell'esperimento. Il primo TAB è sempre dedicato al test di comunicazione con il Wiimote, in questo modo è possibile verificare che la ricezione dei dati dal dispositivo avvenga correttamente prima di procedere.
- TAB dedicato alla configurazione dei parametri utili all'esperimento che si sta eseguendo.
- TAB per l'effettiva esecuzione dell'esperimento.
- TAB per la visualizzazione dei risultati numerici e grafici.

I principali motivi che hanno portato alla scelta di Labview al posto di Matlab per la realizzazione dei programmi sono due:

1. Maggior semplicità nella realizzazione delle interfaccie grafiche. Labview è pensato infatti proprio per la realizzazione di pannelli grafici ben strutturati e ricchi di controlli, pulsanti ed indicatori di vario genere. Il linguaggio di programmazione usato è anch'esso grafico, questo potrebbe a prima vista risultare caotico, ma in realtà risulta immediato quando si tratta di realizzare applicativi dedicati all'acquisizione e analisi di dati, controllo processi e automazione in genere.
2. Possibilità di distribuire gli applicativi senza che gli utenti finali debbano sostenere costi aggiuntivi. Gli eseguibili generati da Labview per funzionare richiedono che sul computer sia installato il *Labview Runtime Engine*, software interprete per gli applicativi realizzati. La *Runtime Engine* è gratuita e scaricabile liberamente da internet quindi è possibile usare i programmi realizzati senza acquistare alcun software.

Verranno ora proposti gli esperimenti realizzati, descrivendoli singolarmente in modo approfondito.

5.1 WIIMOTE IN CADUTA LIBERA

Questo primo esperimento permette di analizzare la caduta di un corpo a terra per effetto dell'accelerazione gravitazionale. L'oggetto viene rilasciato da una certa altezza (altezza iniziale Z_0) all'istante iniziale $t_0 = 0$, quindi a causa dell'azione dell'accelerazione di gravità g risente di una forza $F = m \cdot g$ che lo spinge verso terra con velocità crescente. Al momento dell'impatto con il terreno si ha un urto, seguito da una fase di assestamento dovuta al rimbalzo del corpo, infine il corpo si ferma (velocità nulla). Nel nostro caso il corpo è sostituito dal Wiimote, che viene fatto cadere da altezze di qualche decina di centimetri e lasciato schiantare su un morbido cuscino. Si deve evitare assolutamente un urto con una superficie rigida per non danneggiare irreparabilmente il delicato dispositivo.

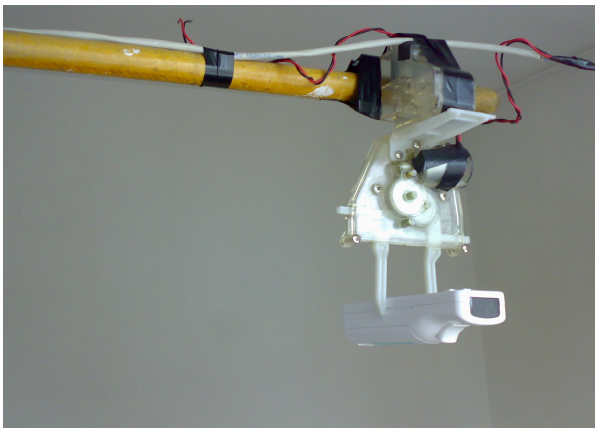


Figura 5.1: Esempio: una pinza robotica tiene il Wiimote sollevato da terra

Lo scopo principale del programma realizzato è determinare il tempo di caduta del controller basandosi sui valori di accelerazione trasmessi durante l'esecuzione dell'esperimento; con tale dato è possibile quindi risalire all'altezza iniziale ed alla velocità finale all'istante dell'urto. Grazie all'applicazione è possibile anche emulare la caduta di un oggetto in condizioni particolari, come ad esempio una diversa accelerazione gravitazionale oppure una velocità iniziale non nulla.
Prima di addentrarsi nell'esplorazione delle caratteristiche del programma e nella discussione dei problemi affrontati nella sua realizzazione sono necessari alcuni richiami di fisica per capire meglio cosa avviene nell'accelerometro quando il Wiimote viene lasciato cadere.

5.1.1 Richiami teorici

La caduta di un corpo può essere studiata attraverso le formule del moto uniformemente accelerato (si veda il [33]). Per convenzione si esprimeranno le relazioni matematiche in riferimento ad un asse z orientato nello stesso verso del moto, come mostrato in figura 5.2.

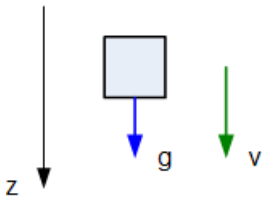


Figura 5.2: Caduta di un corpo e asse z orientato verso il basso

Si definisce l'accelerazione istantanea come la derivata della velocità istantanea $v(t)$ rispetto al tempo:

$$a(t) = \frac{dv(t)}{dt} \tag{5.1.1}$$

Viceversa, nota l'accelerazione istantanea è possibile, integrando, valutare la velocità del corpo:

$$v(t) = \int_{t_0}^{t} a(t) \cdot dt + v(t_0) \tag{5.1.2}$$

dove $v(t_0)$, di seguito indicata semplicemente con V_0, è la velocità istantanea all'istante $t = t_0$.
Nel nostro caso $a = g = cost \simeq 9.81 m/s^2$ e $t_0 = 0$ (si fa coincidere l'istante iniziale con l'origine dell'asse dei tempi), ne risulta la nota formula per il calcolo della velocità all'istante t per un moto uniformemente accelerato:

$$v(t) = V_0 + g \cdot t \tag{5.1.3}$$

Se il corpo arriva a terra in un tempo t_{cad}, la velocità finale al momento dell'urto risulta quindi essere data da:

$$V_f = V_0 + g \cdot t_{cad} \tag{5.1.4}$$

Si definisce la velocità istantanea come la derivata della posizione istantanea $z(t)$ rispetto al tempo:

$$v(t) = \frac{dz(t)}{dt} \tag{5.1.5}$$

A partire dalla velocità istantanea è possibile, integrando, ricavare la posizione:

$$z(t) = \int_{t_0}^{t} v(t) \cdot dt + z(t_0) \tag{5.1.6}$$

dove $z(t_0)$ è la posizione iniziale all'istante $t = t_0$. Nel seguito per semplicità si indicherà $z(t_0) = Z_0$.
Sostituendo l'espressione di $v(t)$ ricavata in 5.1.3 nell'equazione 5.1.6 si ottiene la formula per il calcolo della posizione istantanea per un moto uniformemente accelerato:

$$z(t) = Z_0 + V_0 \cdot t + \frac{1}{2} \cdot g \cdot t^2 \tag{5.1.7}$$

Sia t_{cad} il tempo impiegato dal corpo per raggiungere il terreno, ovvero la posizione $z(t_{cad}) = 0$. Si può ottenere in questo modo l'altezza iniziale Z_0 a cui si trovava il corpo all'istante iniziale t_0 mediante:

$$Z_0 = -\left(V_0 \cdot t_{cad} + \frac{1}{2} \cdot g \cdot t_{cad}^2 \right) \tag{5.1.8}$$

Orientando l'asse z verso l'alto, velocità ed accelerazione sono negative in quanto dirette verso il basso, quindi l'altezza Z_0 ottenuta sarà positiva, come rappresentato in figura 5.3.

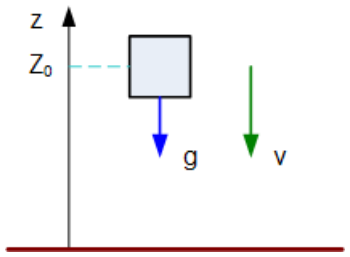


Figura 5.3: Caduta di un corpo e asse z orientato verso l'alto con origine al suolo

Dopo la caduta (per $t > t_{cad}$) il corpo subisce un urto. Nel caso reale della caduta del Wiimote l'urto causa rimbalzi del dispositivo con possibili rotolamenti, dopodichè l'oggetto si ferma ($a = 0$, $v = 0$). L'accelerometro è sottoposto ad un gradino di accelerazione, infatti un istante prima dell'urto il dispositivo misura $a = 0$ e l'istante dopo $a = g$; si hanno quindi oscillazioni smorzate della massa interna all'accelerometro, in accordo con la descrizione del secondo ordine tipica dei sistemi Massa-Molla-Smorzatore (questo fenomeno è molto evidente nei grafici che mostrano l'andamento dell'accelerazione durante l'evento). Se l'accelerazione è nulla, non è possibile sapere se il corpo è in quiete o in moto uniforme. Poichè siamo consapevoli che dopo l'urto il corpo arresta il suo moto possiamo affermare che anche $v = 0$, se ci basassimo solo sulla misura dell'accelerometro potremmo solo dire che la sorgente che causava l'accelerazione è stata tolta, di conseguenza è logico pensare che il corpo continui il suo moto a velocità costante. Si vedrà nel programma realizzato che l'andamento della velocità segue quest'ultima descrizione in quanto calcolata per integrazione del segnale corrispondente all'accelerazione del Wiimote.
Prima di procedere all'analisi dei risultati è bene studiare il comportamento di un accelerometro quando è sottoposto a moto uniformemente accelerato. Come già noto un accelerometro può essere modellato come sistema Massa-Molla-Smorzatore, lo studio di questo tipo di sistema risulta abbastanza semplice, tuttavia vanno fatte delle considerazioni nel caso in cui l'accelerometro non sia fermo ma si muova di moto uniformemente accelerato, in quanto il valore misurato dal dispositivo risulta riferito ad un sistema di riferimento non inerziale. Il Wiimote che contiene l'accelerometro può decelerare o accelerare, come nel caso della caduta libera, il sistema di riferimento è solidale al Wiimote e quindi si muove con esso. L'effetto dell'accelerazione impressa al sistema di riferimento si ripercuote sull'accelerometro: il valore misurato non corrisponde alla reale accelerazione del Wiimote. cosa accade in dettaglio analizzando prima il caso in cui il Wiimote risulta fermo e poi il caso in cui il Wiimote cada sotto l'azione dell'accelerazione gravitazionale.

1. WIIMOTE FERMO

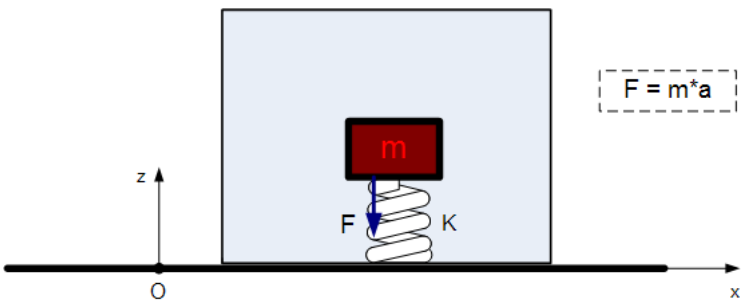


Figura 5.4: Sistema Inerziale: Wiimote fermo

Il Wiimote è rappresentato in figura 5.4 dal rettangolo grande, al suo interno l'accelerometro (asse z) è rappresentato dalla massa m e dalla molla k (lo smorzatore è omesso). La massa è

soggetta all'accelerazione di gravità quindi $a = g \simeq 9.81 m/s^2$, di conseguenza essa risente di una forza $F = m \cdot g$ orientata verso il centro della terra. All'equilibrio la molla risulta compressa della quantità:

$$z_c = \frac{m \cdot g}{k} \tag{5.1.9}$$

L'accelerometro misura quindi l'accelerazione gravitazionale g. Si noti che il sistema di riferimento O è inerziale in quanto è in quiete. Gli stessi risultati si avrebbero anche se il sistema di riferimento fosse in moto rettilineo uniforme lungo l'asse z.

2. WIIMOTE IN CADUTA

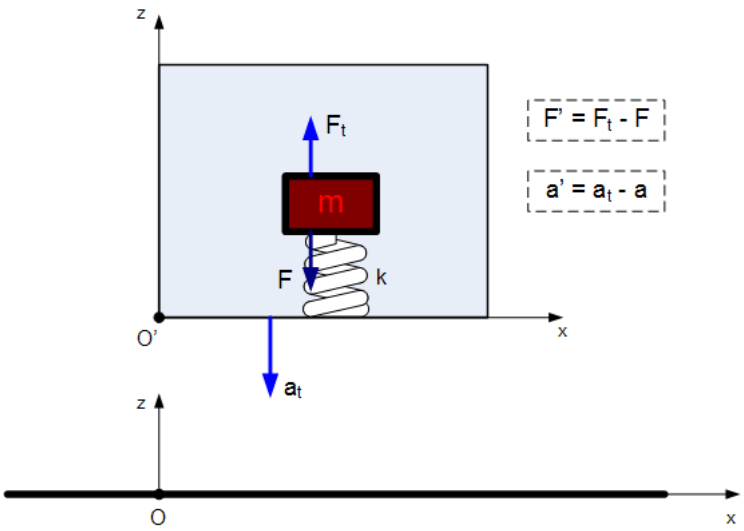


Figura 5.5: Sistema Non Inerziale: Wiimote in caduta

Nel caso in cui il Wiimote venga fatto cadere (figura 5.5) l'accelerazione a cui è soggetta la massa risulta essere la differenza tra l'accelerazione gravitazionale g (nel sistema inerziale O) e l'accelerazione a cui è sottoposto il sistema di riferimento O', anch'essa uguale a g:

$$a' = a_t - a = g - g = 0 \tag{5.1.10}$$

quindi l'accelerazione misurata dal dispositivo (che è riferita al sistema di riferimento O') è nulla (la massa m sperimenta la cosiddetta "assenza di peso"). Questo ovviamente non è dovuto alla scomparsa dell'attrazione terrestre ma al fatto che se tutto il sistema sta scendendo con la stessa accelerazione dei corpi che ad esso si riferiscono non c'è più accelerazione relativa (si pensi ad esempio alle sensazioni percepite in un ascensore in discesa con accelerazione costante).
Un osservatore in O' vedrebbe, nell'istante in cui al sistema viene applicata l'accelerazione g, la molla espandersi, come se la massa fosse soggetta ad una forza $F_t = m \cdot a_t$. All'equilibrio, essendo l'accelerazione nulla, risulta:

$$z'_c = \frac{F'}{k} = \frac{m \cdot a'}{k} = 0 \tag{5.1.11}$$

L'accelerazione che interessa è quella applicata al Wiimote, ovvero quella vista dal sistema inerziale O: essa si può ottenere grazie alla relazione inversa:

$$a = a' + a_t = a_{mis} + g \tag{5.1.12}$$

dove a è orientata nel verso di caduta, ovvero verso il basso.
Se il Wiimote si muovesse di moto uniforme oppure fosse in quiete l'accelerometro misurerebbe $a_{mis} = g$ e sarebbe $a = 0$, non si avrebbe modo di sapere in quale dei due casi, moto o quiete, ci si trova. Queste considerazioni sono fondamentali nello sviluppo dell'esperimento e nell'elaborazione dei dati acquisiti mediante Wiimote.

5.1.2 Interfaccia Grafica

Si analizzeranno ora gli elementi principali che costituiscono l'interfaccia grafica del programma, chiarendo i compiti dei vari controlli ed indicatori presenti e definendo i passi da compiere per portare a termine l'esperimento ed analizzare i risultati.
All'apertura del programma appare la finestra riportata in figura 5.6:

Figura 5.6: Finestra all'apertura del programma (STEP 1)

Per poter eseguire l'esperimento è necessario far cadere il Wiimote tenendolo con i tasti rivolti verso l'alto (asse z del Wiimote orientato verso l'alto e stessa direzione dell'asse su cui agisce la forza di gravità). Il Wiimote deve essere già in posizione al momento dell'avvio dell'esperimento, quindi deve essere rilasciato. Per un funzionamento corretto del programma è importante che il dispositivo cada mantenendo l'orientazione iniziale (tasti verso l'alto). È bene non toccare il Wiimote prima del termine dell'acquisizione dati. È necessario inoltre configurare tutti i parametri richiesti per l'esecuzione dell'esperimento, che verranno tra poco discussi. Lo svolgimento dell'esperimento è organizzato in 4 passi che vanno eseguiti con ordine, ad ogni passo è associato un *TAB*. All'apertura del programma automaticamente viene visualizzato il TAB dello *STEP 1: Test Connessione Al Wiimote* (vedi figura 5.6).

Il primo passo è controllare che la ricezione dei dati dal Wiimote avvenga correttamente. Dopo aver connesso il controller al computer si può cliccare sul pulsante *"Connetti Wiimote"*, se tutto funziona a dovere il grafico sottostante mostra in *real-time* i dati acquisiti dal dispositivo. Si possono controllare i dati provenienti da tutti e tre gli assi del controller. Per terminare la connessione con il Wiimote è sufficiente premere sul pulsante *"Disconnetti Wiimote"*.
Il periodo di campionamento impostato sarà quello usato per l'acquisizione e per la successiva elaborazione dei dati dell'esperimento. Il valore di default è il minimo valore permesso, pari a $10ms$.
Terminato il primpo *STEP* si può quindi passare al successivo *STEP 2: Parametri Per L'esperimento*. Ciò che viene visualizzato è riportato in figura 5.7:

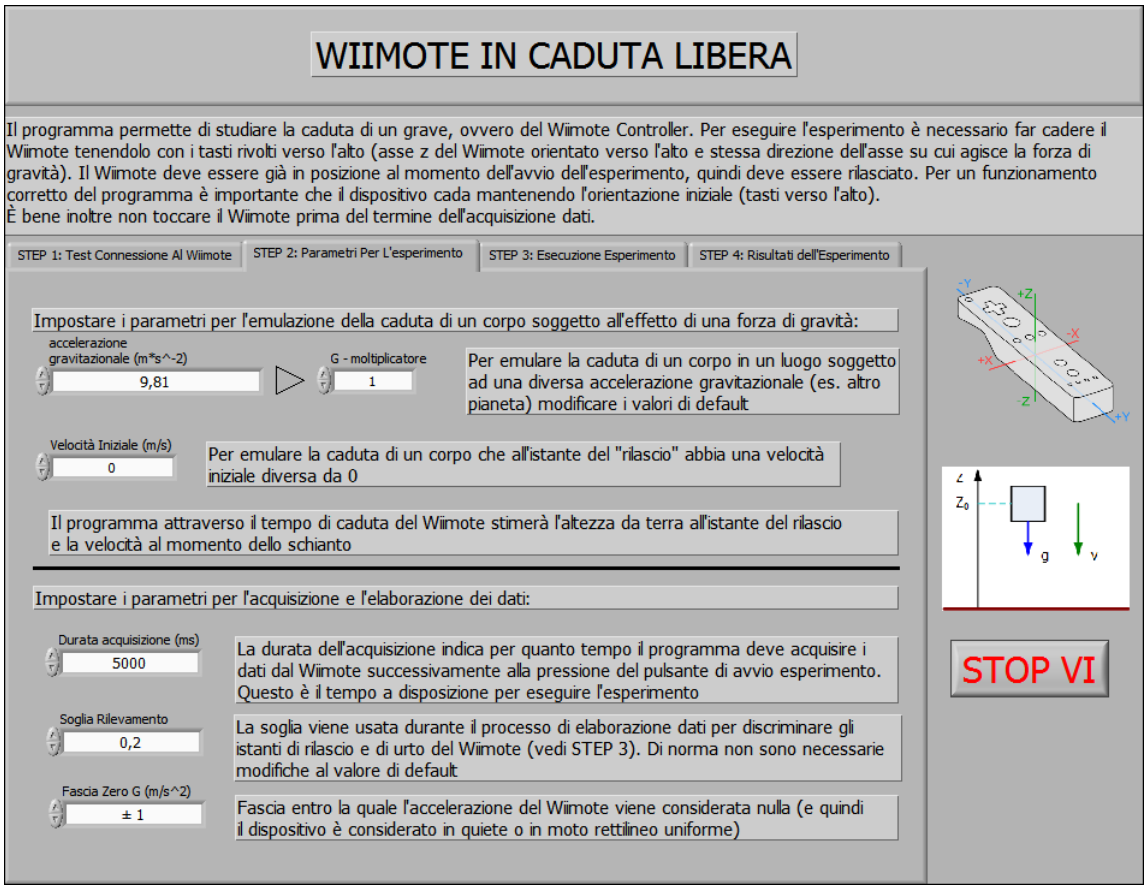


Figura 5.7: STEP 2: Parametri Per L'esperimento

In questo passo si chiede all'utente di modificare eventualmente i parametri per l'emulazione della caduta del corpo e per l'elaborazione dei dati. Si nota subito la possibilità di emulare una gravità differente da quella terrestre (impostata di default), questo si può fare sia modificando il valore dell'accelerazione gravitazionale sia modificando il valore del moltiplicatore. Un altro valore impostabile è la velocità iniziale al momento del rilascio del corpo.
Nella parte inferiore della finestra si possono modificare alcuni parametri legati all'acquisizione ed all'elaborazione dei dati: la *durata dell'acquisizione* indica per quanto tempo il programma dovrà acquisire i dati dal Wiimote, l'esperimento della caduta deve essere eseguito all'interno di questa finestra temporale secondo le modalità descritte. La *soglia rilevamento* viene utilizzata per rilevare gli istanti di rilascio e di urto del Wiimote durante la fase di elaborazione dei dati acquisiti (la sua funzione verrà chiarita durante l'analisi del codice, sottosezione 5.1.3).
Il valore del controllo *"Fascia Zero G"* viene utilizzato in fase di elaborazione, la sua funzione è legata alla generazione dei risultati grafici: valori di accelerazione del Wiimote a_{wii} (quella riferita al

sistema inerziale) compresi nell'intervallo $-V \leq a_{wii} \leq +V$, dove V è il valore di soglia impostato nel campo numerico *"Fascia Zero G"*, vengono interpretati come accelerazione nulla, ovvero come se fosse $a_{wii} = 0$. Questo garantisce una corretta rappresentazione dei risultati in quanto vengono eliminate le oscillazioni iniziali del controller prima del rilascio (dovute ad esempio ad una mano non ferma) e gli effetti dovuti ad una leggera inclinazione del controller al momento dell'urto. Nella idealità infatti il Wimote dovrebbe arrestarsi con il suo asse z rivolto verso l'alto e perpendicolare al piano d'appoggio. Un'eccessiva inclinazione del Wiimote causa la generazione di grafici sbagliati. Di norma l'utilizzatore del programma non ha motivo di cambiare i parametri di configurazione per l'elaborazione dei dati, tuttavia potrebbe essere necessario impostare un diverso valore di soglia, ad esempio nel caso in cui venga simulata una diversa gravità.
Dopo l'impostazione dei parametri di configurazione il controllo passa allo *STEP 3: Esecuzione Esperimento*, che costituisce il passo principale per l'emulazione della caduta del corpo. Il relativo TAB viene riportato in figura 5.8.

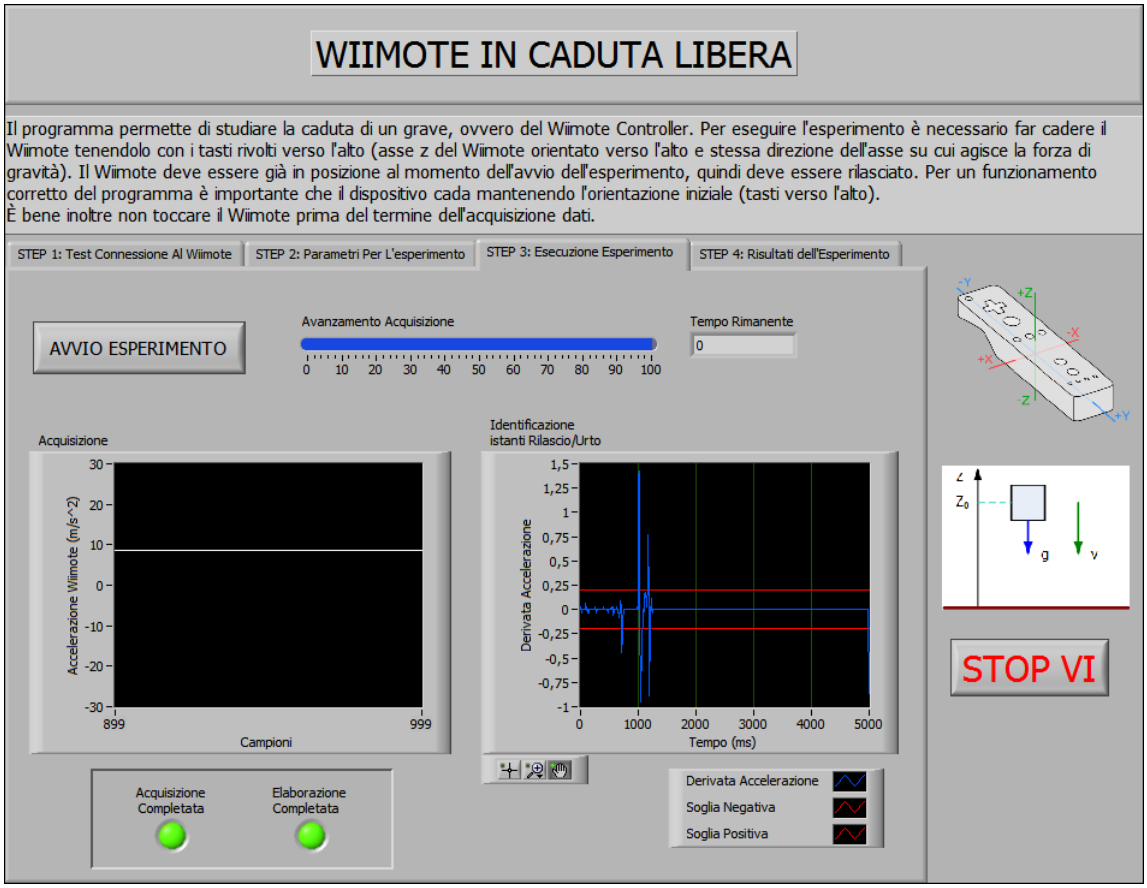


Figura 5.8: STEP 3: Esecuzione Esperimento

In questo passo si eseguirà l'esperimento facendo cadere il Wiimote su un cuscino. Le operazioni da eseguire sono:

1. Tenere il Wiimote sospeso in aria, orientato con i tasti verso l'alto, cercando di mantenerlo il più possibile fermo ed evitando di inclinarlo.

2. Cliccare sul pulsante *"Avvia Esperimento"* per avviare l'acquisizione, la barra di completamento inizierà ad avanzare ed il tempo a scorrere.

3. Rilasciare il Wiimote.

4. Attendere il completamento dell'acquisizione senza toccare il Wiimote (si accendono le spie verdi in basso a sinistra).

5. Se il Wiimote ha raggiunto il terreno con i tasti rivolti verso l'alto (a meno di una leggera inclinazione) si può procedere alla visioni dei risultati, alternativamente è necessario ripetere l'esperimento eseguendo le operazioni a partire dalla numero 1.

Il grafico *"Acquisizione"* sulla sinistra del pannello frontale mostra in real-time il valore di accelerazione misurato e quindi ottenuto interpolando i campioni ricevuti dal Wiimote. Nel grafico *"identificazione istanti rilascio/urto"* invece si può osservare l'esito della procedura di calcolo del tempo di caduta del controller a partire dal segnale di accelerazione acquisito. Sono ben evidenziate le soglie il cui valore (in modulo) era stato definito nello *STEP 2*.
L'analisi dei risultati è riportata nello *STEP 4: Risultati dell'Esperimento*. I risultati sono sia numerici che grafici: i singoli grafici sono separati e organizzati in un *TAB Control*. Lo studente che utilizza il programma può facilmente passare da un grafico all'altro cliccando sui TAB. Il TAB aperto di default è quello contenente i risultati numerici dell'esperimento, in figura 5.9 è riportata la schermata corrispondente.

Figura 5.9: STEP 4: Risultati dell'Esperimento (Numerici)

I risultati dell'esperimento sono il tempo di caduta del Wiimote che corrisponde al tempo di caduta del corpo (t_{cad}) nell'emulazione. Tale valore è indipendente dalla gravità scelta o dalla velocità iniziale specificata, in quanto corrispondente al reale tempo di caduta del dispositivo, in condizioni di $g \simeq 9.81 m/s^2$ e $V_0 = 0$.
Viene poi visualizzata la velocità finale del corpo al momento dell'urto, che dipende dalla velocità

iniziale, dall'accelerazione gravitazionale e dal tempo di caduta secondo le relazioni fisiche dimostrate in precedenza.
L'ultimo valore considerato è l'altezza del corpo al momento del rilascio, ovvero la posizione di partenza nel caso in cui il sistema di riferimento inerziale sia un asse orientato verso l'alto e con origine al livello del terreno. Nell'esempio riportato in figura 5.9 il tempo di caduta misurato è $290ms$, con i parametri per l'esperimento indicati in figura 5.7 si ottengono una velocità al momento dell'urto pari a $2.8m/s^2$ ed un'altezza iniziale del corpo pari a $41cm$.
I grafici infine mostrano l'andamento di accelerazione, velocità e posizione del Wiimote nel tempo. Ogni grafico è dotato di appositi controlli che permettono l'analisi approfondita dei segnali rappresentati: è possibile eseguire zoom, creare cursori, modificare il colore, il tipo di interpolazione usata, ecc. Cliccando sul grafico con il tasto destro del mouse è possibile anche esportare l'immagine (opzione *"Export Simplified Image"*).
In figura 5.10 è riportato, a titolo di esempio, il grafico accelerazione-tempo all'interno del TAB *STEP 4*. Nel grafico è mostrato il segnale di accelerazione acquisito dal Wiimote durante l'esperimento, emulando le condizioni di gravità specificate nello *STEP 2*.

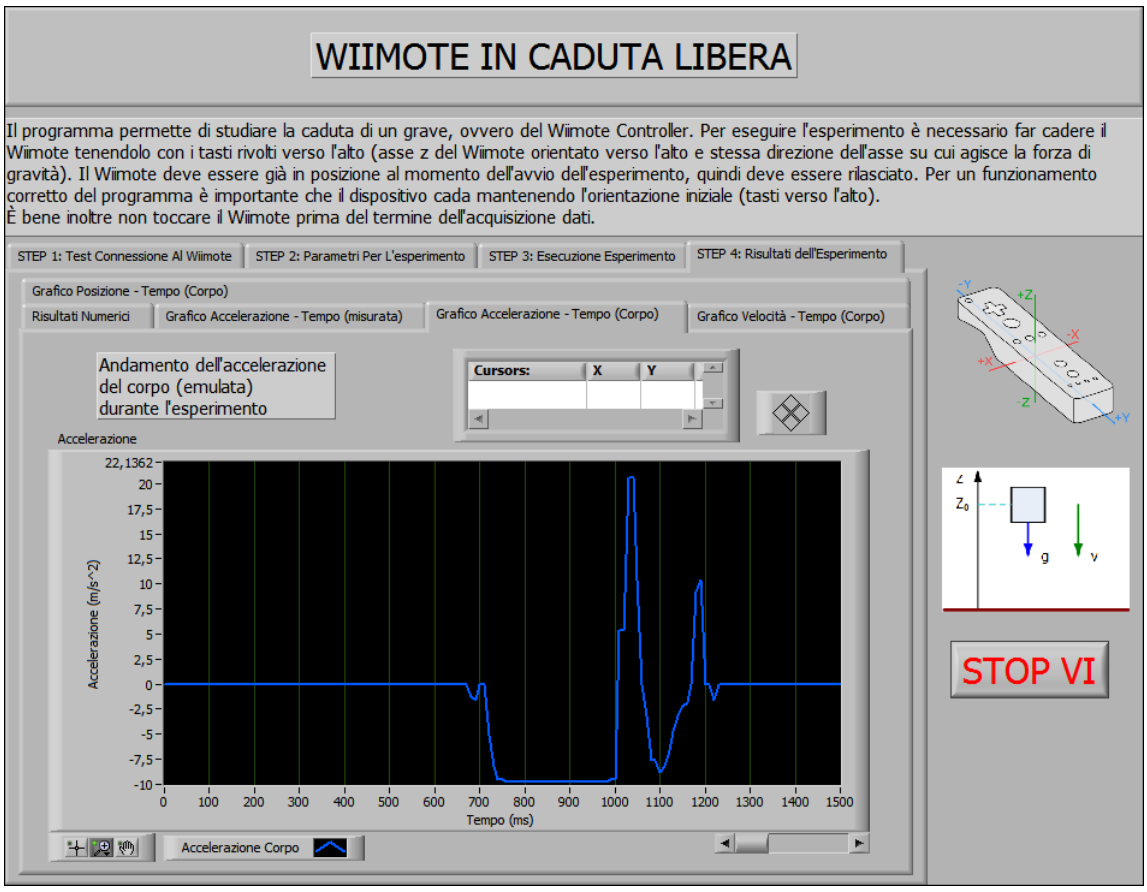


Figura 5.10: STEP 4: Risultati dell'Esperimento (Grafici)

5.1.3 Analisi del Codice e dei Risultati

In questa sezione viene riportato il codice sorgente sviluppato, descrivendone le parti di maggior interesse e motivando con idee e ragionamenti le scelte fatte ai fini di risolvere i problemi ed ottenere un'applicazione in grado di funzionare nel modo voluto.
Al *TAB Control* principale (selezione *STEP*) è associata una struttura *CASE*, attraverso la quale è possibile eseguire solamente il codice di pertinenza del passo selezionato. Gli unici *STEP* nei quali

vengono eseguite operazioni sono il primo, per la visualizzazione dei dati ricevuti dal Wiimote, ed il terzo, dove avviene l'acquisizione e l'elaborazione dei dati dell'esperimento. Il codice associato allo *STEP 1* è riportato in figura 5.11.

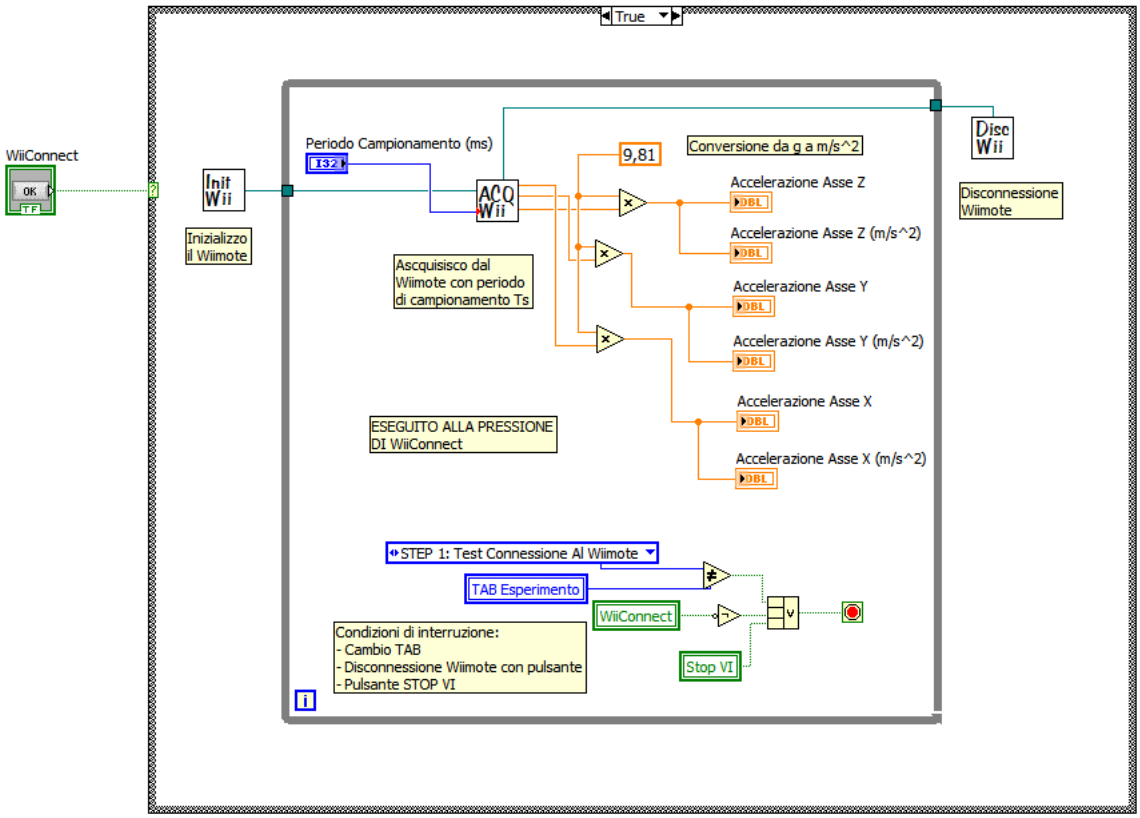


Figura 5.11: Codice STEP 1 per la visualizzazione dei dati trasmessi dal Wiimote

Come si può vedere la pressione del tasto *WiiConnect* (nell'interfaccia grafica *"Connetti Wiimote"*) avvia un ciclo *WHILE* che si occupa dell'acquisizione continua dei dati dal controller con il periodo di campionamento impostato.
Il valori di accelerazione restituiti dal Sub-VI *ACQWii*, che si occupa della lettura dei dati dal Wiimote, sono espressi in unità di misura g. Prima di essere usati questi dati devono venire convertiti nell'unità del Sistema Internazionale m/s^2 tramite moltiplicazione per $g = 9.81 m/s^2$. È sufficiente un valore approssimato di g in quanto i valori di accelerazione ottenuti dal dispositivo hanno un'incertezza estesa $Ug \simeq 0.015g \simeq 0.15 m/s^2$ (si veda la sezione 4.2.4). L'accelerazione viene poi visualizzata in real-time sui *Chart* presenti nell'interfaccia grafica. La terminazione del ciclo avviene alla pressione del pulsante *WiiConnect* (nell'interfaccia grafica *"Disconnetti Wiimote"*), al passaggio ad un TAB diverso dallo *STEP 1*, oppure alla pressione del pulsante di terminazione VI *"Stop VI"*.
Il codice associato allo *STEP 3*, riportato in figura 5.12, costituisce il cuore dell'applicazione: qui viene eseguita l'acquisizione e l'elaborazione dei dati in base ai parametri impostati negli *STEP 1* e *2*. La pressione del pulsante *Start* (nell'interfaccia grafica *"Avvia Esperimento"*) avvia tutta la procedura, i dati vengono acquisiti dal Wiimote attraverso un ciclo *FOR* (la durata dell'acquisizione è finita) e successivamente elaborati e rappresentati su grafici. Nella fase di elaborazione vengono chiamati alcuni Sub-VI realizzati appositamente per svolgere determinate funzioni: il calcolo del tempo di caduta del Wiimote (Sub-VI *time*) ed il calcolo dei vettori contenenti i campioni rappresentanti velocità e posizione emulate del corpo (Sub-VI *Vel&Space*).

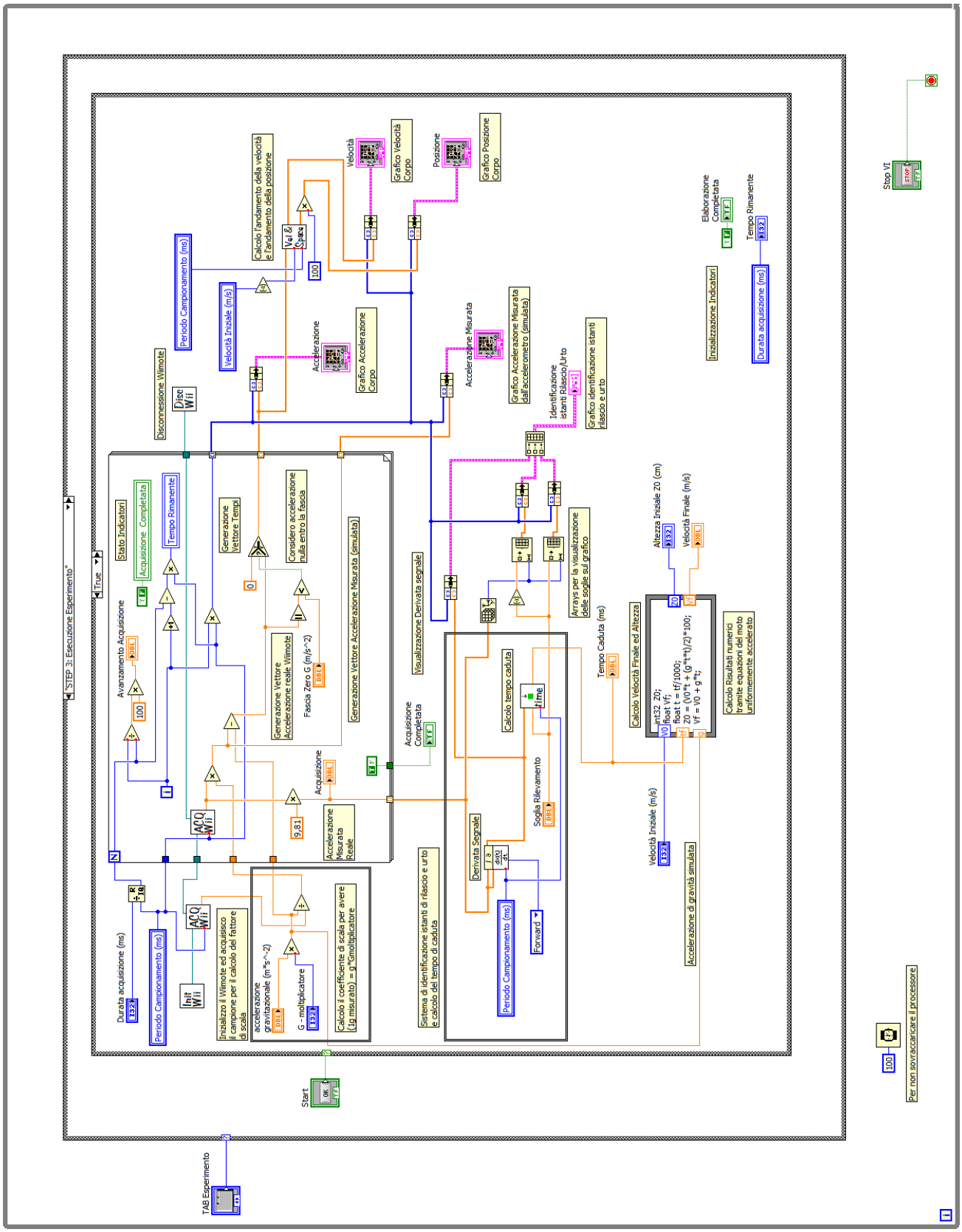


Figura 5.12: Codice STEP 3: acquisizione, elaborazione, visualizzazione dei dati

Il codice risulta essere abbastanza esteso, in alto a sinistra si può osservare la connessione con il Wiimote, la generazione del fattore di scala per l'emulazione di diverse accelerazioni gravitazionali ed il ciclo *FOR* che si occupa dell'acquisizione e della configurazione iniziale dei dati per l'elaborazione, che avviene successivamente alla terminazione del ciclo. In alto a destra vi è il codice che si occupa del calcolo e della rappresentazione della velocità e della posizione del corpo nel tempo. nel centro del *block diagram* è riportato il codice legato al calcolo del tempo di caduta. Immediatamente sotto invece si ha il calcolo dei risultati numerici mediante le formule del moto uniformemente accelerato (*FormulaNode*).

Calcolo del tempo di caduta e risultati numerici
Il calcolo del tempo di caduta costituisce l'operazione più importante eseguita in fase di elaborazione: permette di ottenere i risultati numerici riportati nel pannello frontale. Le operazioni eseguite sono:

- Calcolo della derivata (discreta) del segnale riferito all'accelerazione reale del Wiimote.
  Osservando ciò che avviene nell'esperimento di caduta del Wiimote si nota immediatamente che nell'istante del rilascio si ha una brusca variazione dell'accelerazione misurata, a_{mis} scende a gradino, questo nel segnale derivato si traduce in un impulso negativo in prossimità dell'istante del rilascio. Allo stesso modo si nota che al momento dell'urto si ha ancora una brusca variazione di a_{mis}, che sale a gradino, dando origine ad un picco positivo nel segnale derivato.

- Identificazione degli istanti in cui il segnale derivato presenta il primo picco negativo ed il primo picco positivo.
  Per discriminare i picchi si usa una soglia (*"Soglia Rilevamento"* nei parametri dell'esperimento), per risalire agli istanti temporali nei quali avvengono rilascio e urto quindi si analizzano i campioni del vettore contenente la derivata del segnale di accelerazione partendo dal primo: si cerca come prima cosa il primo campione minore della soglia (negativa), situato all'istante $t_0 = 0$ corrispondente al rilascio. Successivamente si procede, allo stesso modo, all'individuazione del primo campione maggiore della soglia (positiva), situato all'istante t_1 corrispondente all'arrivo a terra del Wiimote. Soglia positiva e negativa sono uguali in modulo. In figura 5.13 è riportato il grafico che viene visualizzato nello *STEP 3* al termine dell'acquisizione, utile per capire il funzionamento del processo di identificazione degli istanti di rilascio/urto: viene mostrato il grafico dell'accelerazione misurata dal Wiimote (in blu nell'immagine) con sovrapposte le soglie descritte (in rosso nell'immagine). È possibile identificare immediatamente il primo picco negativo, individuato dall'intersezione della soglia negativa con il grafico dell'accelerazione, ed il primo picco positivo, individuato dall'intersezione della soglia positiva con il grafico dell'accelerazione.

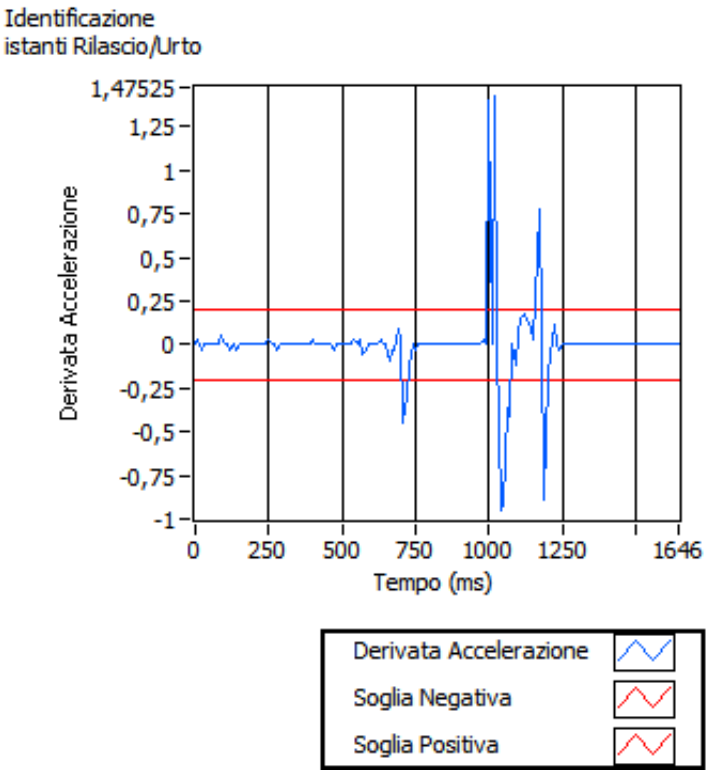


Figura 5.13: Identificazione Istanti Rilascio/Urto

- Il tempo di caduta è dato dalla differenza $t_{cad} = t_1 - t_0 = t_1$

Il calcolo della derivata dell'accelerazione misurata avviene nel VI principale, mentre l'identificazione degli istanti di rilascio e di urto ed il calcolo del tempo di caduta avvengono nel Sub-VI *time*. I risultati numerici dello *STEP 4* vengono calcolati a partire dai parametri dell'esperimento e dal tempo di caduta attraverso un *formula node*, nel quale vengono applicate le equazioni 5.1.8 ed 5.1.4. Le variabili in uscita dal *nodo* vengono infine visualizzate negli indicatori numerici del TAB *"Risultati Numerici"*. In figura 5.14 è riportato il frammento di codice che evidenzia le operazioni descritte.

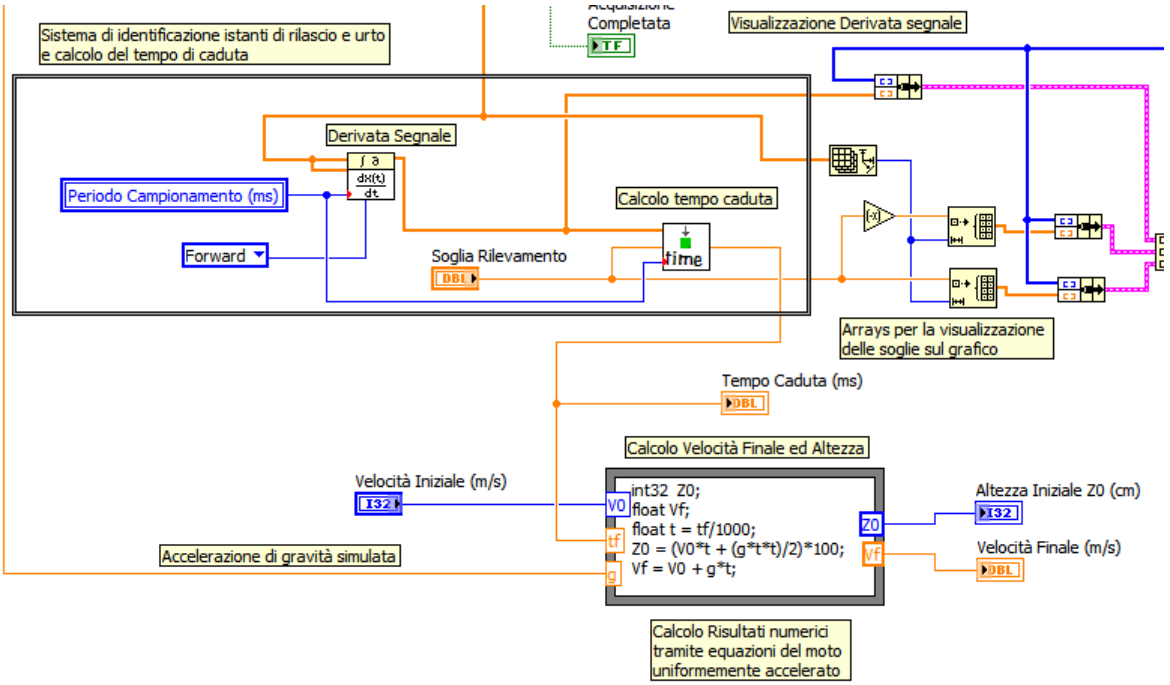


Figura 5.14: Calcolo tempo di caduta e risultati numerici

Il Sub-VI *time* riceve in ingresso il vettore contenente la derivata del segnale di accelerazione, il valore della *soglia di rilevamento* ed il periodo di campionamento. In uscita viene fornito il valore in millisecondi del tempo di volo t_{cad}. Il calcolo di t_{cad} avviene attraverso due cicli *FOR* eseguiti consecutivamente: il primo ciclo si occupa di rilevare il primo campione del vettore in ingresso minore della soglia negativa, corrispondente all'istante di rilascio. Quando tale campione viene identificato il primo ciclo termina e viene eseguito il secondo. Il secondo ciclo opera sui campioni del vettore rimanenti, che rappresentano la "storia" dell'esperimento successivamente al rilascio del Wiimote. Quando viene individuato il primo campione maggiore della soglia positiva, corrispondente all'urto, il ciclo termina. Il numero di campioni esaminati da quest'ultimo ciclo rappresenta la distanza in campioni N dall'istante di rilascio all'istante di urto. Il tempo di caduta (in millisecondi) è dato infine da $T_{cad} = N \cdot T_s$, dove T_s è il periodo di campionamento (in millisecondi). Il codice del Sub-VI *time* è riportato in figura 5.15.

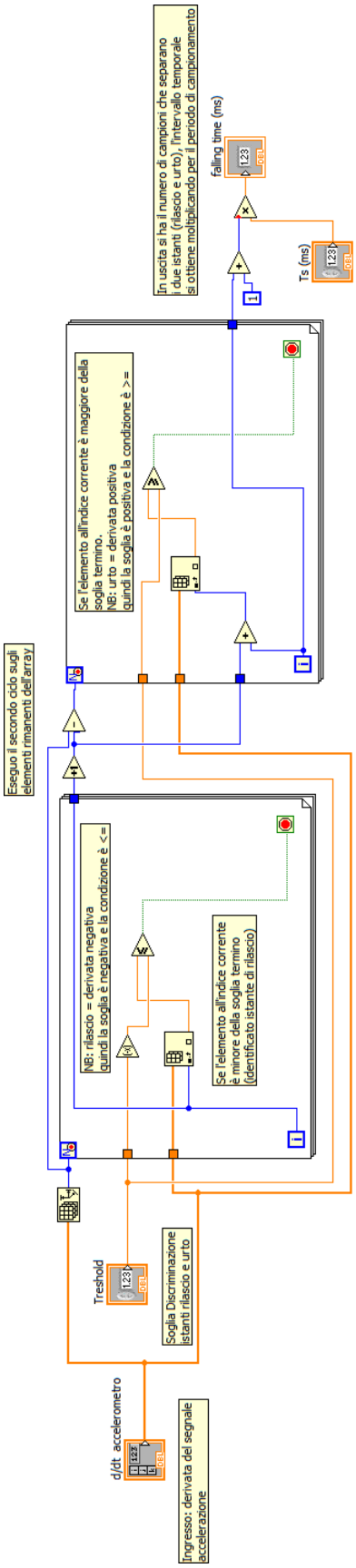


Figura 5.15: Sub-VI *time*

Calcolo dei vettori velocità e posizione, risultati grafici
I risultati grafici dell'esperimento sono divisi nei seguenti *TAB*:

- Grafico *Accelerazione Misurata - tempo*.
  Il grafico mostra l'andamento dell'accelerazione misurata, emulando le condizioni di gravità volute (a_{emu}), vedi paragrafo *"osservazioni aggiuntive: fattore di scala e vettori accelerazione"*. Un esempio ottenuto con gravità terrestre $9.81 m/s^2$ è mostrato in figura 5.16.

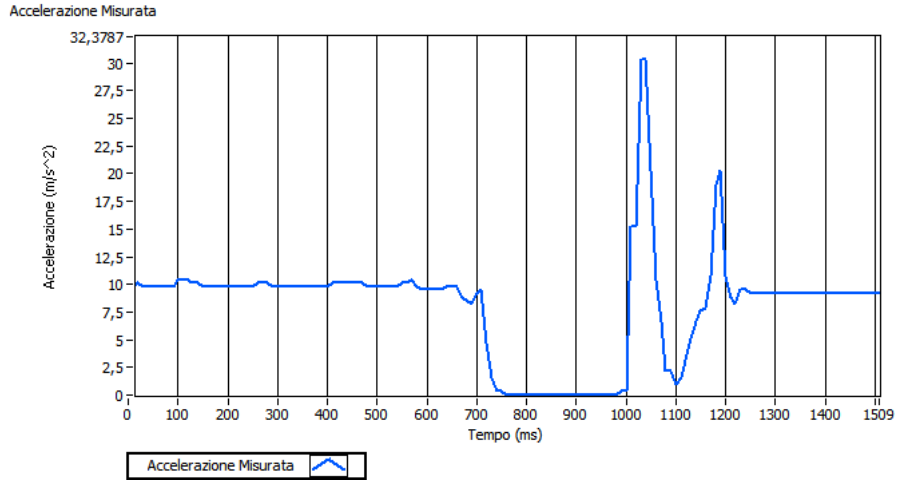


Figura 5.16: Accelerazione Misurata

- Grafico *Accelerazione del corpo durante l'esperimento - tempo*.
  Il grafico mostra l'andamento dell'accelerazione del corpo a_{corpo} (considerando l'approssimazione a 0 entro la *fascia Zero G*). Un esempio ottenuto con gravità terrestre e con caduta corretta del Wiimote è mostrato in figura 5.17. L'accelerazione durante la caduta è negativa in quanto viene usato un asse di riferimento orientato verso l'alto.

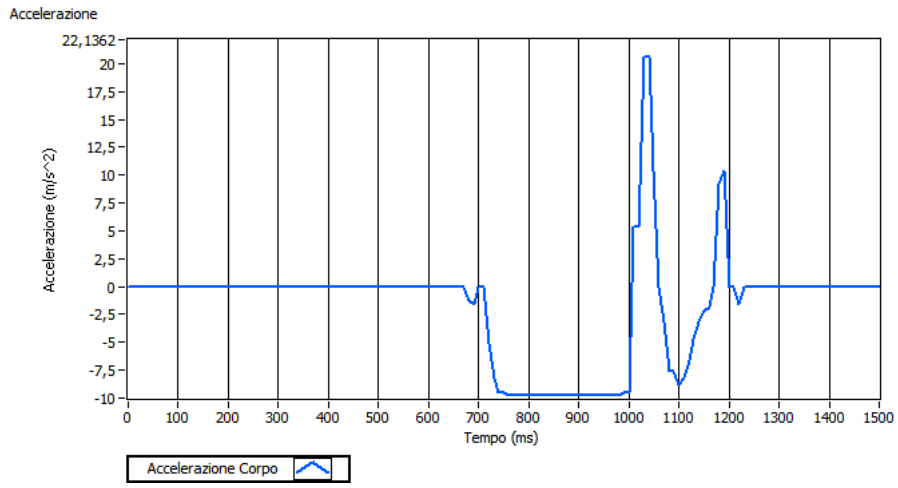


Figura 5.17: Accelerazione Corpo

- Grafico *Velocità del corpo durante l'esperimento - tempo*.
  Il grafico mostra l'andamento della velocità durante l'esecuzione dell'esperimento. La velocità

risulta anch'essa negativa in quanto viene usato un asse di riferimento orientato verso l'alto. Il vettore contenente le misure di velocità è ottenuto eseguendo l'integrazione discreta delle misure di accelerazione; questa operazione è eseguita all'interno del Sub-VI *Vel&Space*. Se il Wiimote cade correttamente, quindi dritto, con i tasti verso l'alto e poco inclinato viene generato un grafico come quello riportato in figura 5.18.

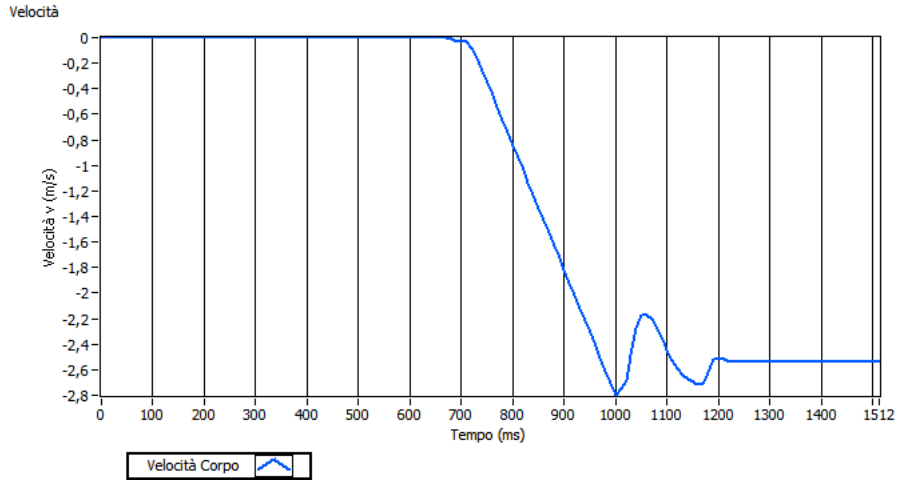
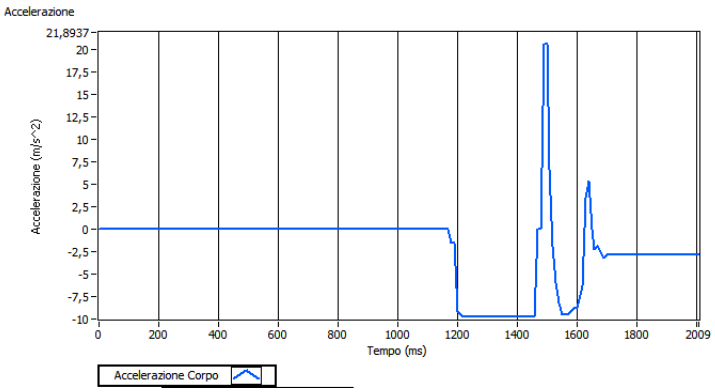
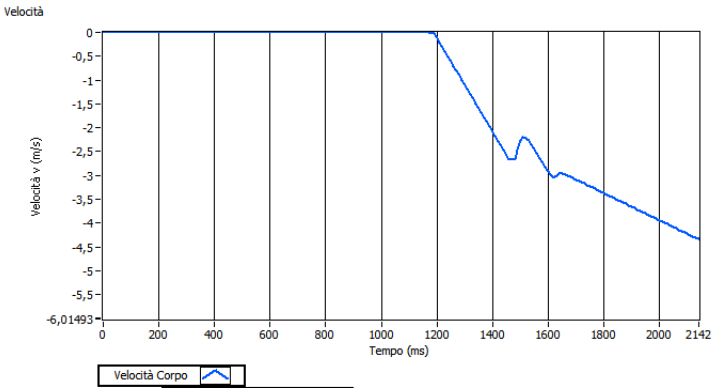


Figura 5.18: Velocità Corpo (caduta corretta)

Successivamente all'urto il corpo procede con velocità costante in quanto non vi è più accelerazione. Nella realtà il corpo risulta fermo, il comportamento visualizzato nel grafico è dovuto alla mancanza di informazione dell'accelerazione di cui si è parlato precedentemente.
L'introduzione della *"fascia Zero G"* fa si che la velocità non aumenti successivamente all'urto, nel caso in cui il Wiimote atterri leggermente inclinato: un'inclinazione del dispositivo infatti causerebbe un'accelerazione del corpo diversa da zero successivamente all'urto, $a_{corpo} = a_{emu} - g < 0$, in quanto la componente dell'accelerazione gravitazionale registrata dall'accelerometro sull'asse z sarebbe $a_{emu} < g$. Essendo la velocità ottenuta per integrazione dell'accelerazione si avrebbe erroneamente un comportamento come quello riportato in figura 5.19.



(a) Accelerazione Corpo (Wiimote inclinato dopo l'urto)



(b) Velocità Corpo (Wiimote inclinato dopo l'urto)

Figura 5.19: Effetti dovuti all'inclinazione del Wiimote dopo l'urto

- Grafico *Posizione corpo durante l'esperimento - tempo.*
  Il grafico mostra l'andamento della posizione del corpo durante l'esecuzione dell'esperimento, si veda la figura 5.20. La posizione è riferita ad un asse orientato verso l'alto e con origine nella posizione iniziale del corpo (posizione prima del rilascio), risulta quindi essere negativa (corpo in caduta). La posizione è ottenuta per integrazione discreta della velocità, il calcolo viene eseguito nel Sub-VI *Vel&Space* come nel caso precedente. Anche l'andamento della posizione nel tempo risente della mancanza di informazione dell'accelerazione, successivamente all'urto lo spazio percorso continua ad aumentare in quanto la velocità risulta costante. Come nel caso della velocità vi sono gli effetti indesiderati dovuti all'inclinazione del controller.

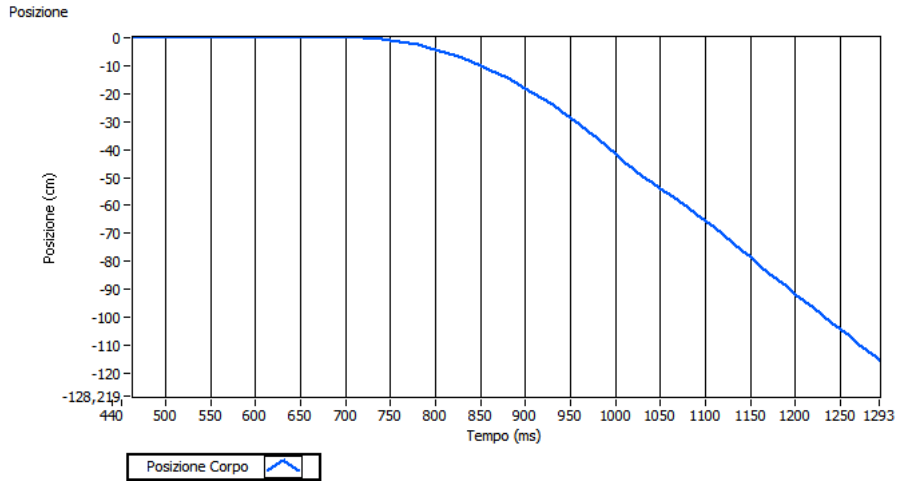


Figura 5.20: Posizione Corpo (caduta corretta) - Dettaglio durante l'intervallo di caduta

L'integrazione per segnali a tempo discreto si ottiene come segue:

$$\int_0^t a(t)\cdot dt \implies T_s \cdot \sum_{k=0}^{K} a(k\cdot T_s) \tag{5.1.13}$$

Tale operazione di somma può essere implementata attraverso un ciclo *FOR* ed all'utilizzo di *Shift register* (o di *feedback node*). Il calcolo di velocità e posizione eseguito in *Vel&Space* è mostrato in figura 5.21. L'operazione di integrazione viene implementata mediante un ciclo *FOR*, sebbene esistano già delle funzioni adatte allo scopo in Labview (*"Integral x(t)"* nella *Mathematic Functions Palette*).

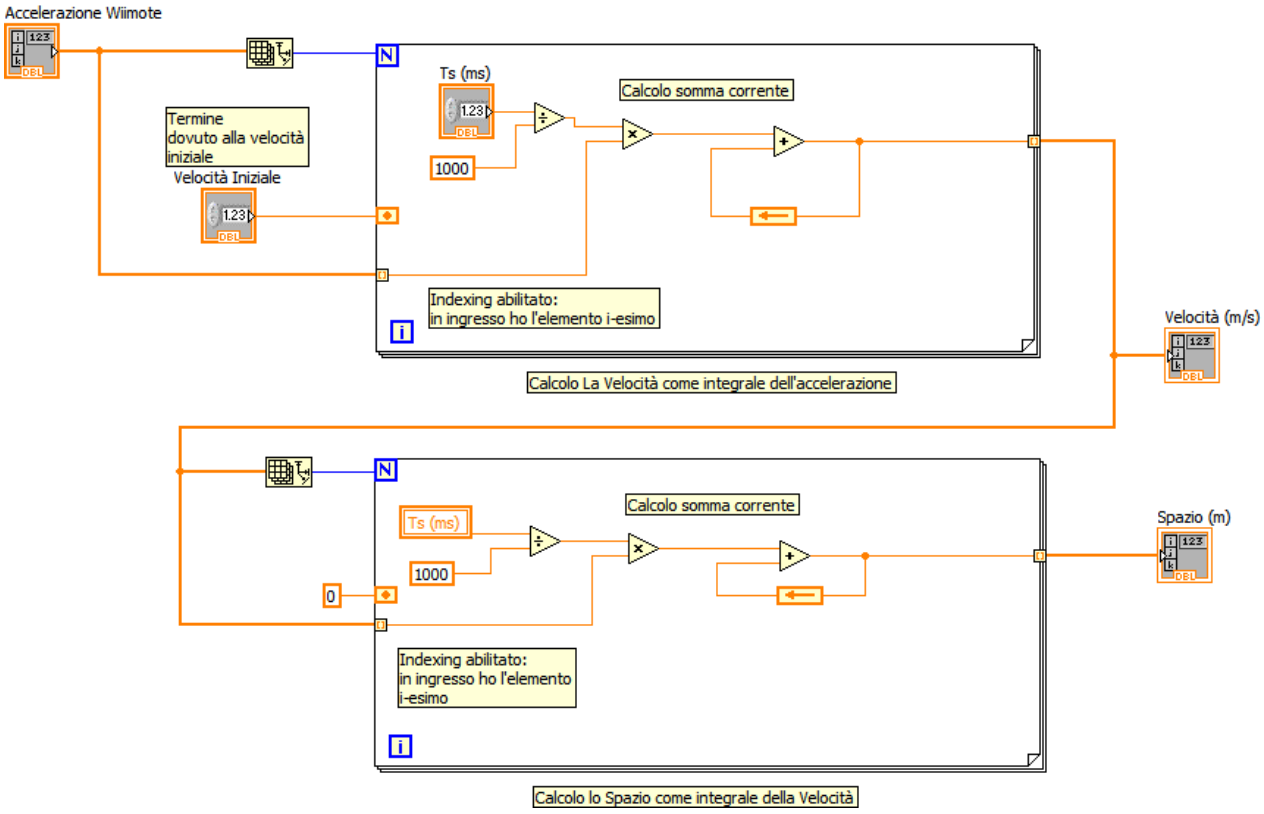


Figura 5.21: Sub-VI *Vel&Space*

Osservazioni aggiuntive: Fattore di scala e ettori accelerazione
All'avvio dell'esperimento viene acquisito un campione dal Wiimote, se il controller è tenuto con i tasti rivolti verso l'alto il valore misurato sarà pari all'accelerazione gravitazionale terrestre (espressa in g). Il fattore di scala permette di associare a questo valore l'accelerazione gravitazionale emulata specificata nei parametri di configurazione dell'esperimento. Secondo la proporzione:

$$a_{1g} : g = 1 : \alpha \tag{5.1.14}$$

dove a_{1g} è il campione acquisito rappresentante la misura dell'accelerometro in condizioni di gravità terrestre, $g = gravità \cdot moltiplicatore$ è la gravità simulata ed α è il fattore di scala da applicare alle successive misure di accelerazione. Infatti risulta:

$$a_{emu} = a_{mis} \cdot \alpha = \frac{a_{mis} \cdot g}{a_{1g}} \tag{5.1.15}$$

dove a_{emu} è l'accelerazione misurata dall'accelerometro nel caso emulato, ovvero come se fosse soggetto all'accelerazione gravitazionale g.
Il ciclo *FOR* genera un primo vettore contenente il segnale dell'accelerazione misurata dal Wiimote (a_{mis}), questo viene usato esclusivamente per il calcolo del tempo di caduta, che non è emulato ma reale. Un secondo vettore generato contiene invece l'accelerazione emulata (a_{emu}), ottenuta applicando il fattore di scala α. Infine un ultimo vettore contiene il segnale dell'accelerazione emulata del corpo, ottenuta dall'accelerazione emulata secondo la relazione:

$$a_{corpo} = a_{emu} - g \tag{5.1.16}$$

L'accelerazione misurata all'inizio dell'esperimento è pari all'accelerazione gravitazionale g (positiva), ma il corpo è tenuto sospeso fermo, quindi la sua accelerazione, riferita al sistema inerziale deve

essere nulla. Quando il corpo è in caduta l'accelerazione misurata è pari a zero, ma l'accelerazione dell'oggetto riferita al sistema inerziale deve essere pari a g. Per risolvere alcuni problemi legati al calcolo dei risultati grafici relativi alla velocità ed alla posizione del corpo si considera inoltre nulla un'accelerazione a_{corpo} compresa nell'intervallo $(-V\,,V)$, con V specificato nei parametri dell'esperimento e corrispondente al valore denominato *"Fascia Zero G"*. Il ciclo *FOR* si occupa anche della creazione del vettore dei tempi usato per la generazione dei grafici. La figura 5.22 mostra nel dettaglio la parte di codice che implementa le operazioni descritte.

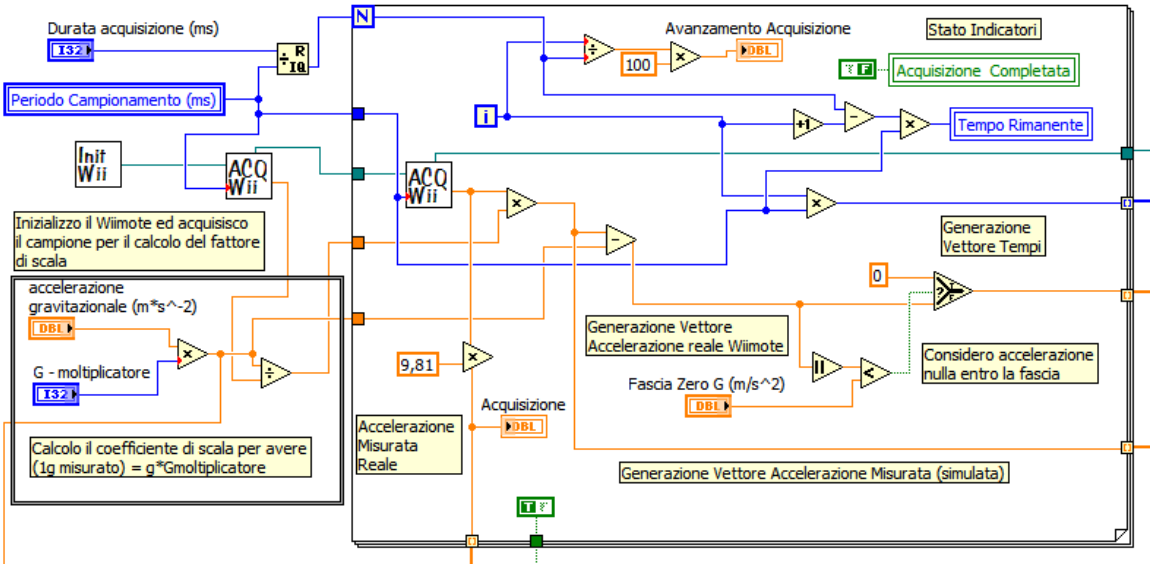


Figura 5.22: Acquisizione Dati e generazione vettori accelerazione

5.1.4 Considerazioni sulle incertezze dei risultati e conclusioni

Il Wiimote non è pensato per effettuare misure precise ed accurate di accelerazione, inoltre il programma creato è molto semplice, vengono fatte numerose approssimazioni che causano un'elevata incertezza sui risultati, difficilmente stimabile. I principali contributi all'incertezza di misura sono:

- Incertezza intrinseca del controller.
  Il dispositivo presenta un'incertezza sulle misure di accelerazione dovuta all'accelerometro (processo di trasduzione accelerazione-capacità-tensione) ed alla conversione analogica-digitale. In secondo luogo il Wiimote non è uno strumento pensato per eseguire misure accurate e precise, ma per la *gesture recognition* (riconoscimento di particolari combinazioni di movimenti, si veda la [30]), quindi non è sottoposto ad adeguate operazioni di taratura e calibrazione, questo incide sulla qualità della misura.

- Incertezza dovuta ai ritardi nella comunicazione tra controller e computer.
  Il Wiimote invia i dati al computer attraverso una connessione Bluetooth e l'uso di una libreria driver. Quando il programma effettua l'acquisizione di un campione viene inviata una richiesta alla libreria, successivamente quest'ultima restituisce alla funzione chiamante gli ultimi dati disponibili inviati precedentemente dal dispositivo. La libreria interroga periodicamente il controller, il periodo di interrogazione è valutabile in modo approssimato (come già visto), mentre non si hanno informazioni sulla quantità di tempo che intercorre tra l'interrogazione del Wiimote da parte della libreria e l'effettiva ricezione dei nuovi dati.
  La conseguenza di quanto appena detto è un'incertezza sul periodo di campionamento e sul-

le misure temporali (come ad esempio il tempo di caduta). Ad incrementare l'incertezza contribuisce anche la modalità di acquisizione basata su un ciclo di polling.

- Incertezza dovuta alle approssimazioni introdotte per semplificare i calcoli nel programma realizzato.
  Si ha un'incertezza sugli istanti di rilascio e di urto introdotta dal sistema "derivata + soglie" adottato. Il contributo maggiore di incertezza è associato al calcolo del tempo di caduta.

- Incertezza dovuta alla modalità di esecuzione dell'esperimento.
  Il Wiimote non è chiaramente un corpo che si presta bene ad esperimenti di caduta, la sua forma e la distribuzione non uniforme della sua massa (a causa della presenza delle batterie) fanno si che sia impossibile il verificarsi di una caduta senza inclinazioni e rotazioni del dispositivo. Questo incide sia sulla misura di accelerazione, sia, di conseguenza, sulla misura del tempo di caduta.

Una stima grossolana dell'incertezza associata alla misura potrebbe essere fatta, ad esempio, mediante prove ripetute (incertezza di tipo A), tuttavia con questo sistema non si avrebbero informazioni sull'errore sistematico eventualmente presente nelle misure.

5.2 WIIMOTE INCLINOMETRO

Il secondo applicativo realizzato ha come obiettivo quello di trasformare il Wiimote in un misuratore di inclinazione (figura 5.23). Il sensore così realizzato potrà essere posto su un piano inclinato per valutarne l'inclinazione rispetto alla posizione orizzontale (ovvero la posizione perpendicolare alla forza di gravità), in un range compreso tra $-90°$ (pendenza negativa) e $+90°$ (pendenza positiva).

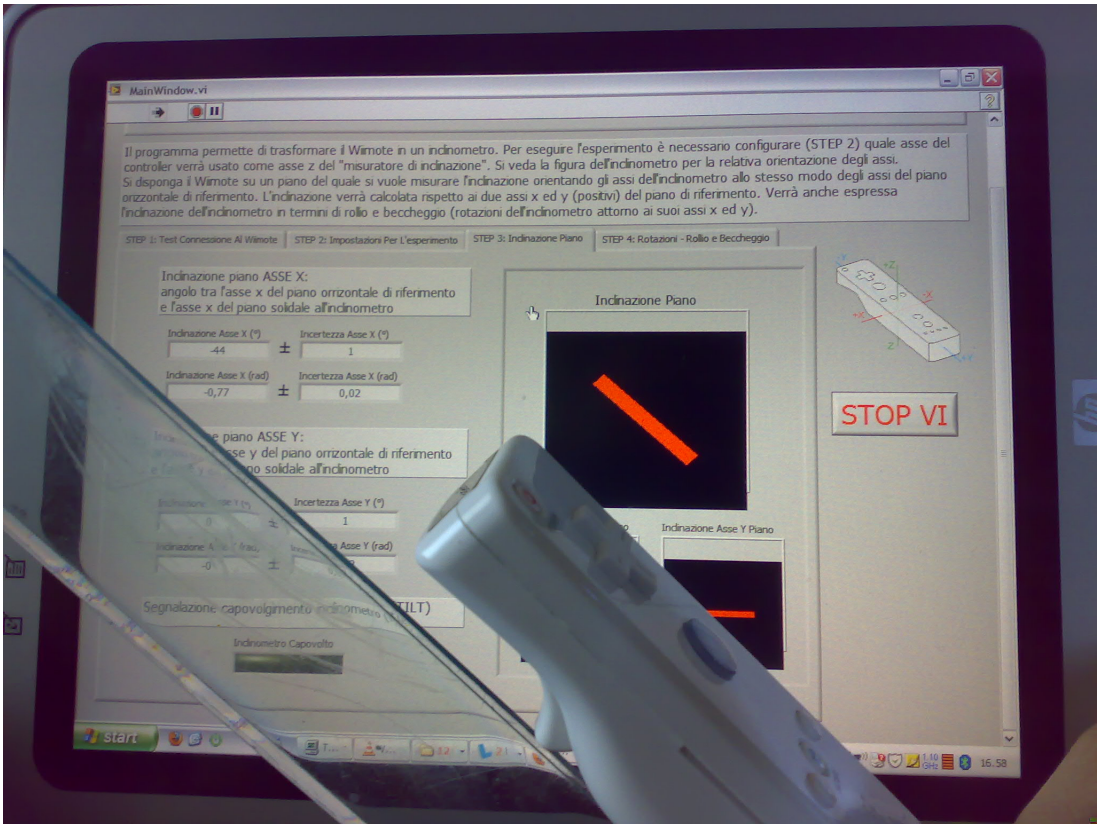


Figura 5.23: Il Wiimote usato come inclinometro. Il dispositivo è tenuto inclinato a 45° per mezzo di una squadra da disegno, sullo sfondo l'applicazione realizzata visualizza i risultati della misura

Il controller misura le componenti dell'accelerazione gravitazionale g_x, g_y, g_z rispetto ai suoi tre assi X, Y, Z. Ruotando il dispositivo (o inclinandolo) le componenti cambiano in quanto l'accelerazione gravitazionale è sempre rivolta nella stessa direzione (verso il basso) mentre il sistema di riferimento, che è solidale al dispositivo, ruota con esso. Grazie alla misura delle componenti è possibile risalire all'angolo di rotazione/inclinazione.
Tenendo il Wiimote con i tasti rivolti verso l'alto, quindi con l'asse +Z rivolto verso l'alto, il valore di accelerazione fornito dal dispositivo risulta, sull'asse Z, positivo e pari a $1g$. Considerando però che con tale orientazione dell'asse la componente su Z deve essere negativa, in quanto orientata verso il basso, è necessario negare il valore restituito dal Wiimote (per avere $-1g$). In modo analogo si devono negare i valori restituiti associati agli assi X ed Y, ottenendo il giusto segno per le componenti in riferimento agli assi associati al controller. In figura 5.24 sono evidenziati gli assi del Wiimote e l'accelerazione di gravità di cui risente il dispositivo.

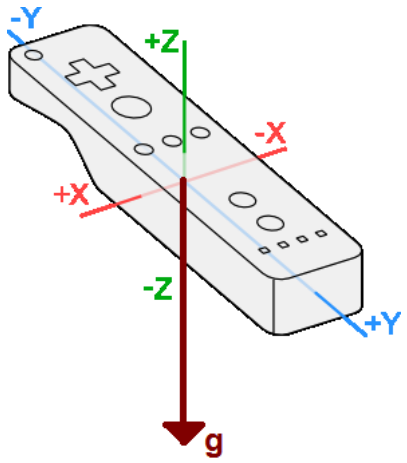


Figura 5.24: Assi di riferimento solidali al Wiimote ed accelerazione gravitazionale

Nella realizzazione del programma si è scelto di astrarre gli assi associati al Wiimote e di lavorare invece con nuovo oggetto, denominato "*inclinometro*", costituito dal Wiimote orientato nel modo evidenziato in figura 5.25 sul quale è stato definito un sistema di riferimento di default. L'asse z dell'*inclinometro* può essere associato agli assi +Z, -Z, +X, -X del Wiimote, in questo modo è possibile scegliere quale lato del controller appoggiare sul piano inclinato. Le orientazioni degli assi x ed y non vengono modificate dalla scelta dell'asse z in quanto definite in riferimneto alle posizioni della *IR Camera* e del connettore per le espansioni del controller: l'applicazione esegue in modo automatico le associazioni tra gli assi del dispositivo (figura 5.24) e gli assi definiti per l'*inclinometro*. Il programma permette, tuttavia, di modificare le orientazioni di default degli assi x ed y dell'*inclinometro* ed anche di scambiarne i ruoli (x diventa y e viceversa), queste operazioni incidono solo sul modo di interpretare i risultati e non sul funzionamento base dell'applicazione (vi è solo un cambiamento di segno sugli angoli calcolati).

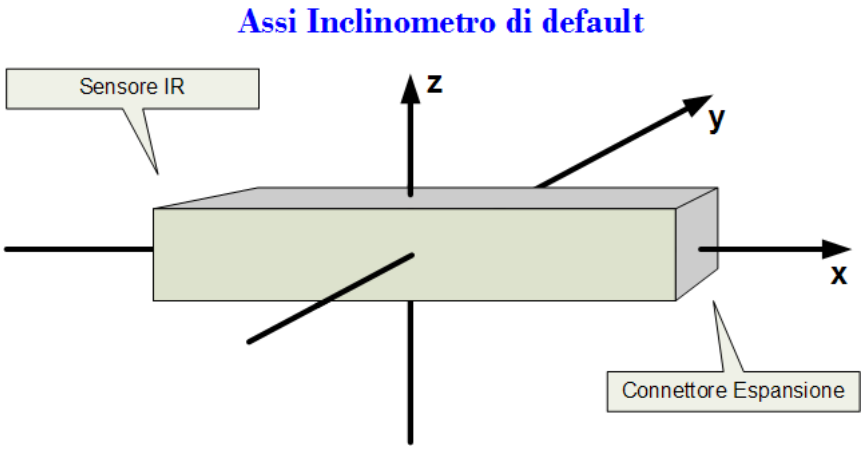


Figura 5.25: *Inclinometro* e relativi assi di riferimento (di default)

La misura dell'inclinazione di un piano avviene nel modo seguente:

1. Si considera un piano inclinato, si definisce per tale piano un sistema di riferimento costituito da tre assi ortogonali tra loro x_{piano}, y_{piano}, z_{piano}, dove z_{piano} è uscente dal piano (verso l'alto).
2. Si definisce un piano orizzontale di riferimento i cui assi x_{ref}, y_{ref}, z_{ref}, abbiano gli stessi versi di quelli del piano inclinato (figura 5.26).

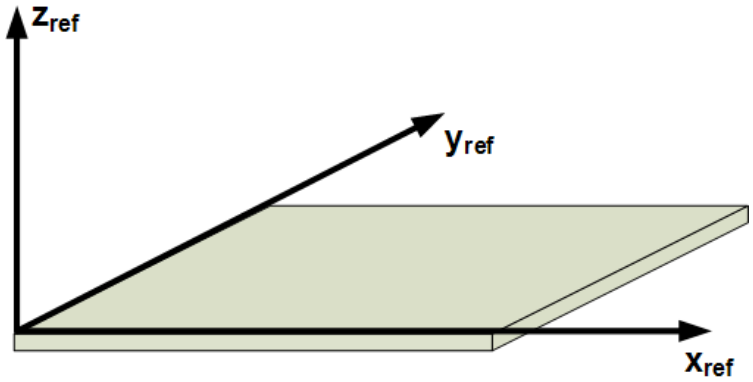


Figura 5.26: Piano orizzontale di riferimento

3. Si colloca l'*inclinometro* sul piano inclinato facendone coincidere i sistemi di riferimento, come mostrato in figura 5.27.

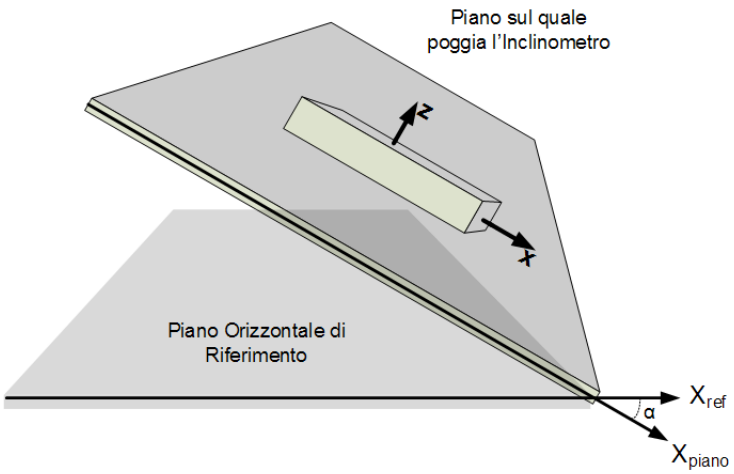


Figura 5.27: Misura dell'inclinazione di un piano attraverso l'*inclinometro*

4. Si esegue l'applicazione, si configurano eventualmente le orientazioni degli assi, se diverse da quelle di default, si procede all'avvio dell'esperimento ed infine si analizzano i risultati.

5. Gli angoli forniti come risultato dell'esperimento sono l'angolo tra l'asse x_{piano} e l'asse x_{ref} (inclinazione rispetto all'asse x_{ref}), e l'angolo tra l'asse y_{piano} e l'asse y_{ref} (inclinazione rispetto all'asse y_{ref}). Gli angoli sono definiti in senso antiorario, ad esempio nel caso di figura 5.27 l'angolo α rappresentato risulta essere negativo (α è definito da x_{piano} a x_{ref}) e si parlerà di piano inclinato negativamente (guardando nel verso di x_{ref} si vede un piano in discesa). L'applicazione limita gli angoli misurabili al range $[-90°, +90°]$, in quanto si prevede di misurare l'inclinazione di un normale piano, senza contemplare il verificarsi di un capovolgimento dello stesso durante l'esecuzione dell'esperimento.

5.2.1 Richiami teorici

Prima di analizzare il funzionamento del programma è necessario chiarire dal punto di vista teorico come sia possibile risalire agli angoli di inclinazione a partire dalle componenti dell'accelerazione

gravitazionale associate agli assi di riferimento dell'*inclinometro* [33, 35].
Prendiamo in considerazione il caso elementare in cui l'*inclinometro* viene ruotato solo attorno all'asse y, ovvero viene inclinato rispetto all'asse x_{ref}, ma non rispetto all'asse y_{ref}. In figura 5.28 è mostrato il caso particolare in cui l'*inclinometro* è inclinato negativamente (l'angolo di inclinazione α risulta essere negativo).

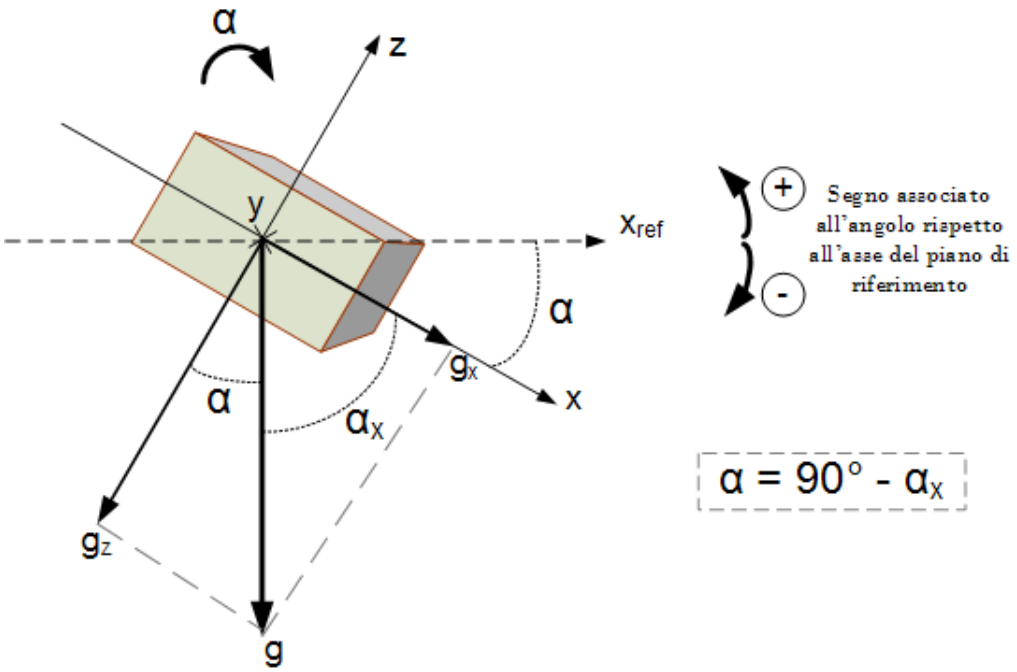


Figura 5.28: Componenti accelerazione gravitazionale nel caso di inclinazione rispetto all'asse x (l'asse y è perpendicolare al foglio, con verso entrante)

La somma vettoriale delle due componenti $\overrightarrow{g_z}$ e $\overrightarrow{g_x}$ da come risultato il vettore $\overrightarrow{g}$ dell'accelerazione gravitazionale, sempre rivolto verso il basso. Con gli assi di riferimento adottati i vettori possono essere riscritti come:

$$\overrightarrow{g_z} = -g_z \cdot \overrightarrow{u_z}$$
$$\overrightarrow{g_x} = g_x \cdot \overrightarrow{u_x}$$
$$\overrightarrow{g} = g \cdot \overrightarrow{u_g}$$

dove $\overrightarrow{u_z}$, $\overrightarrow{u_x}$ sono i versori associati agli assi di riferimento x ed y, g_z e g_x sono i moduli delle componenti di accelerazione in z ed in x, $\overrightarrow{u_g}$ è il versore associato al vettore risultante $\overrightarrow{g}$ mentre g è il suo modulo.
Si verifica sempre la condizione $g = \sqrt{g_x^2 + g_z^2} = cost = 1g$. La componente $\overrightarrow{g_y}$ (non indicata) è nulla in quanto l'inclinometro non risulta inclinato rispetto a quest'asse.
L'angolo che si vuole determinare è α, ovvero l'angolo compreso tra l'asse x dell'*inclinometro* e l'asse x_{ref} di riferimento. Si osserva che α è anche l'angolo compreso tra $\overrightarrow{g}$ e $\overrightarrow{g_z}$. Per determinare l'angolo α si può procedere come segue, dove viene evidenziata la relazione tra l'angolo stesso e le componenti dell'accelerazione gravitazionale, rispettivamente l'ungo l'asse z e l'asse x:

$$g_z = -g \cdot \cos(\alpha) \tag{5.2.1}$$

$$g_x = g \cdot \sin(\alpha) \tag{5.2.2}$$

infine dalle due componenti dell'accelerazione è possibile misurare l'angolo di inclinazione:

$$\alpha = \arctan\left(\frac{g_x}{-g_z}\right) \tag{5.2.3}$$

La formula 5.2.3 restituisce l'angolo con il segno voluto, infatti l'argomento dell'arcotangente è negativo. Nel caso di inclinazione positiva (l'*inclinometro* di figura 5.28 verrebbe ruotato verso sinistra) risulterebbe $\overrightarrow{g_x} = -g_x \cdot \overrightarrow{u_x}$ e quindi:

$$\alpha = \arctan\left(\frac{g_x}{g_z}\right) \tag{5.2.4}$$

con α positivo. Il valori restituiti dal Wiimote includono già i segni relativi alle orientazioni delle componenti, quindi le formule 5.2.3 e 5.2.4 possono essere riassunte scrivendo:

$$\alpha = \arctan\left(\frac{\hat{g_x}}{\hat{g_z}}\right) \tag{5.2.5}$$

dove $\hat{g_x}$ e $\hat{g_z}$ sono i valori (con segno) restituiti dal Wiimote *"inclinometro"* con gli assi di riferimento orientati come in figura 5.28. Risultano $\hat{g_z} < 0$ ed $\hat{g_x} > 0$ nel caso di inclinazione negativa, $\hat{g_z} < 0$ ed $\hat{g_x} < 0$ nel caso di inclinazione positiva.
In figura 5.29 è mostrato cosa accade nel caso in cui l'*inclinometro* venga inclinato rispetto all'asse y_{ref}, ma non rispetto all'asse x_{ref}. La situazione è analoga al caso appena discusso, cambiano solo i nomi delle variabili in gioco. L'angolo da determinare è quello compreso tra l'asse y e l'asse y_{ref} che risulta essere:

$$\beta = \arctan\left(\frac{\hat{g_y}}{\hat{g_z}}\right) \tag{5.2.6}$$

dove $\hat{g_y}$ e $\hat{g_z}$ sono i valori (con segno) restituiti dal Wiimote *"inclinometro"* con gli assi di riferimento orientati come in figura 5.29. Risultano $\hat{g_z} < 0$ ed $\hat{g_y} > 0$ nel caso di inclinazione negativa, $\hat{g_z} < 0$ ed $\hat{g_y} < 0$ nel caso di inclinazione positiva.

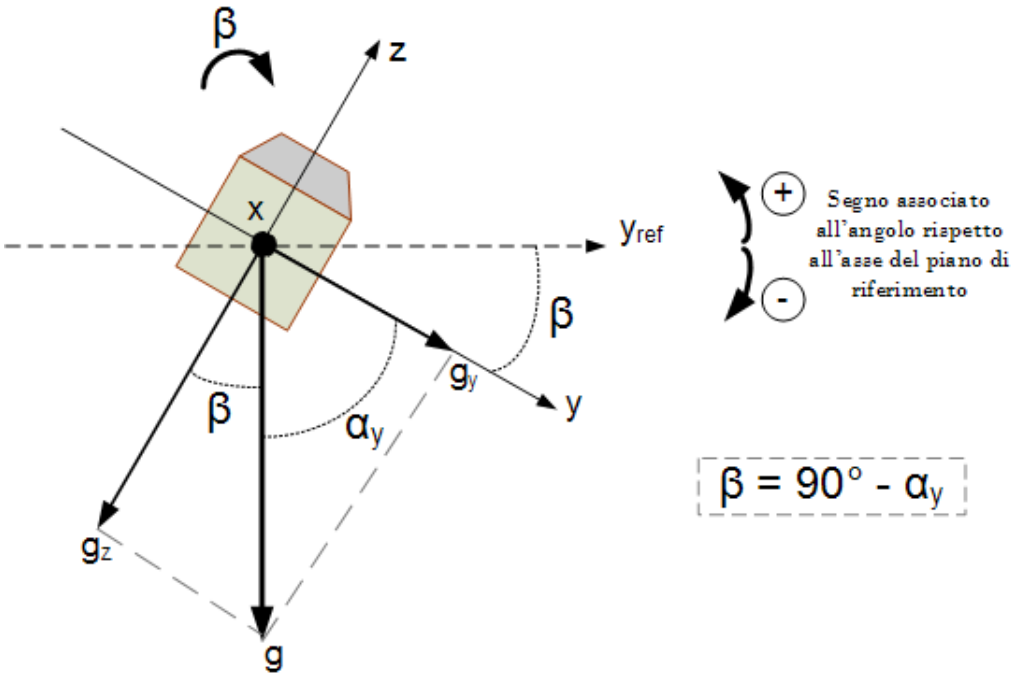
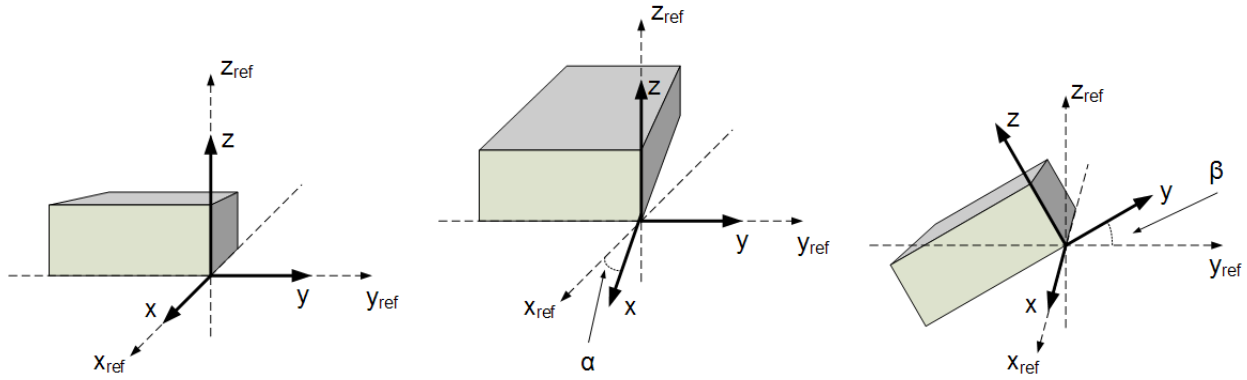


Figura 5.29: Componenti accelerazione gravitazionale nel caso di inclinazione rispetto all'asse y (l'asse x è perpendicolare al foglio, con verso uscente)

Nel caso in cui l'*inclinometro* ruoti attorno ad entrambi gli assi x ed y, ovvero venga inclinato sia rispetto a x_{ref} che rispetto a y_{ref} le cose si complicano: si deve tener conto di tutte e tre le componenti dell'accelerazione gravitazionale nel calcolo degli angoli. Nelle figure 5.30b ed 5.30c si possono osservare in rappresentazione tridimensionale i casi già visti di inclinazione rispetto ad uno solo degli assi di riferimento, mentre in figura 5.31 si ha il caso generale in cui l'inclinazione avviene contemporaneamente rispetto ad entrambi gli assi (per approfondimenti si vedano gli *angoli di Eulero* [34]).



(a) Condizioni iniziali (nessuna inclinazione/rotazione)

(b) Inclinazione rispetto ad x_{ref}

(c) Inclinazione rispetto ad y_{ref}

Figura 5.30: Inclinazione rispetto ad uno solo degli assi di riferimento

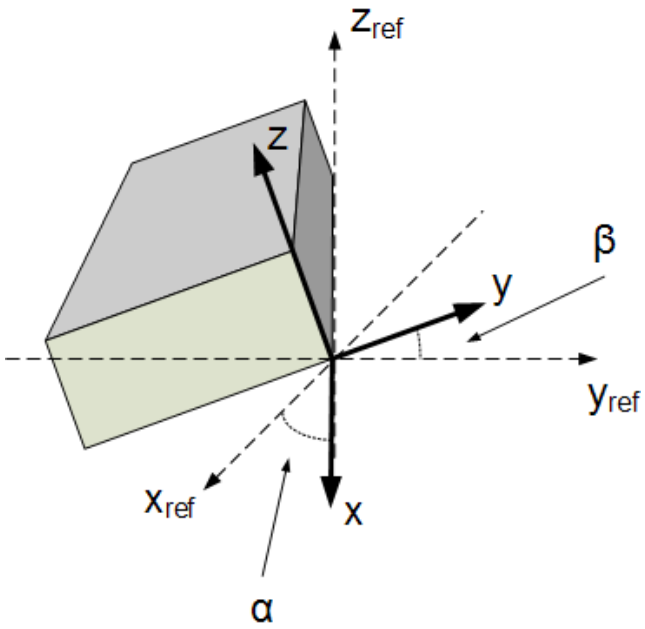


Figura 5.31: Inclinazione rispetto ad entrambi gli assi di riferimento

Nel caso generale, riportato per completezza, è possibile scrivere le seguenti relazioni (si veda la [35]):

$$\alpha = -\arctan\left(\frac{\hat{g_x}}{\sqrt{\hat{g_y}^2 + \hat{g_z}^2}}\right) \tag{5.2.7}$$

$$\beta = -\arctan\left(\frac{\hat{g_y}}{\sqrt{\hat{g_x}^2 + \hat{g_z}^2}}\right) \tag{5.2.8}$$

dove $\hat{g_x}$, $\hat{g_y}$, $\hat{g_z}$ sono i valori (con segno) delle componenti di accelerazione gravitazionale rispetto agli assi x, y e z dell'*inclinometro*. Anche in questo caso i segni associati agli angoli risultano corretti: nel caso di α, essendo il denominatore una quantità sempre positiva, $\hat{g_x}$ positivo implica un'inclinazione negativa rispetto a x_{ref}, mentre $\hat{g_x}$ negativo implica un'inclinazione positiva. Allo stesso modo varia il segno di β in relazione al segno di $\hat{g_y}$.
Vale inoltre sempre la relazione:

$$g = \sqrt{\hat{g_x}^2 + \hat{g_y}^2 + \hat{g_z}^2} \tag{5.2.9}$$

dove g è il valore in modulo dell'accelerazione gravitazionale ($g \simeq 9.81 m/s^2$).
Per concludere si è detto che non viene considerato il caso in cui l'*inclinometro* subisce inclinazioni maggiori in modulo di 90°. In queste condizioni il dispositivo risulta capovolto, ovvero l'asse di riferimento z risulta orientato verso il basso e $\hat{g_z}$ risulta quindi positivo (figura 5.32). Identificare il capovolgimento dell'*inclinometro* è quindi semplice: si deve solo imporre una condizione sul segno di $\hat{g_z}$ per discriminare le due situazioni.

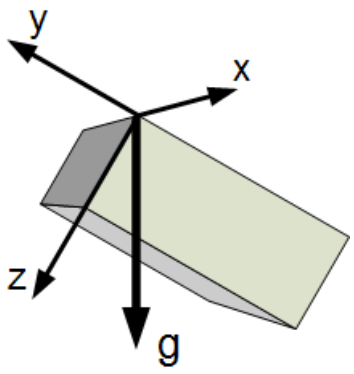


Figura 5.32: Capovolgimento dell'*inclinometro*

5.2.2 Interfaccia Grafica

L'interfaccia grafica dell'applicazione è composta da un'unica finestra nella quale attraverso un *TAB Control* l'utente può eseguire l'esperimento, seguendo con ordine quattro semplici passi, identificati con i nomi *STEP* 1, *STEP* 2, *STEP* 3 e *STEP* 4.
Il primo passo (*STEP* 1) consiste nell'usuale test di connessione con il Wiimote, in questo modo è possibile verificare se vi sono problemi di comunicazione con il dispositivo prima di procedere. La figura 5.33 mostra la finestra visualizzata all'avvio del programma, lo *STEP* evidenziato di default è il primo.

Figura 5.33: Interfaccia grafica *STEP* 1: Test Connessione al Wiimote

Il periodo di campionamento impostato in questo passaggio verrà utilizzato anche durante l'esperimento per l'effettiva acquisizione dei dati.
Nello *STEP* 2 vengono definiti gli assi di riferimento per l'*inclinometro*. La schermata visualizzata è riportata in figura 5.34. L'utente può selezionare quale asse del Wiimote associare all'asse z dell'*inclinometro*, la scelta è legata esclusivamente al modo in cui il controller viene appoggiato sul piano inclinato. Gli assi di riferimento x ed y dell'*inclinometro* possono essere modificati grazie ai *Check Box* "Inverti Asse X", "Inverti Asse Y" e "Scambia Assi X e Y". La funzione dei primi due è quella di invertire l'orientazione rispettivamente dell'asse x e dell'asse y, successivamente la terza opzione permette di scambiare i ruoli dei due assi, definendo y quello che di default è chiamato x e viceversa. Se l'utente non effettua modifiche ai parametri per l'esperimento gli assi di riferimento dell'*inclinometro* sono quelli indicati in figura 5.25, riportata anche nell'interfaccia grafica.

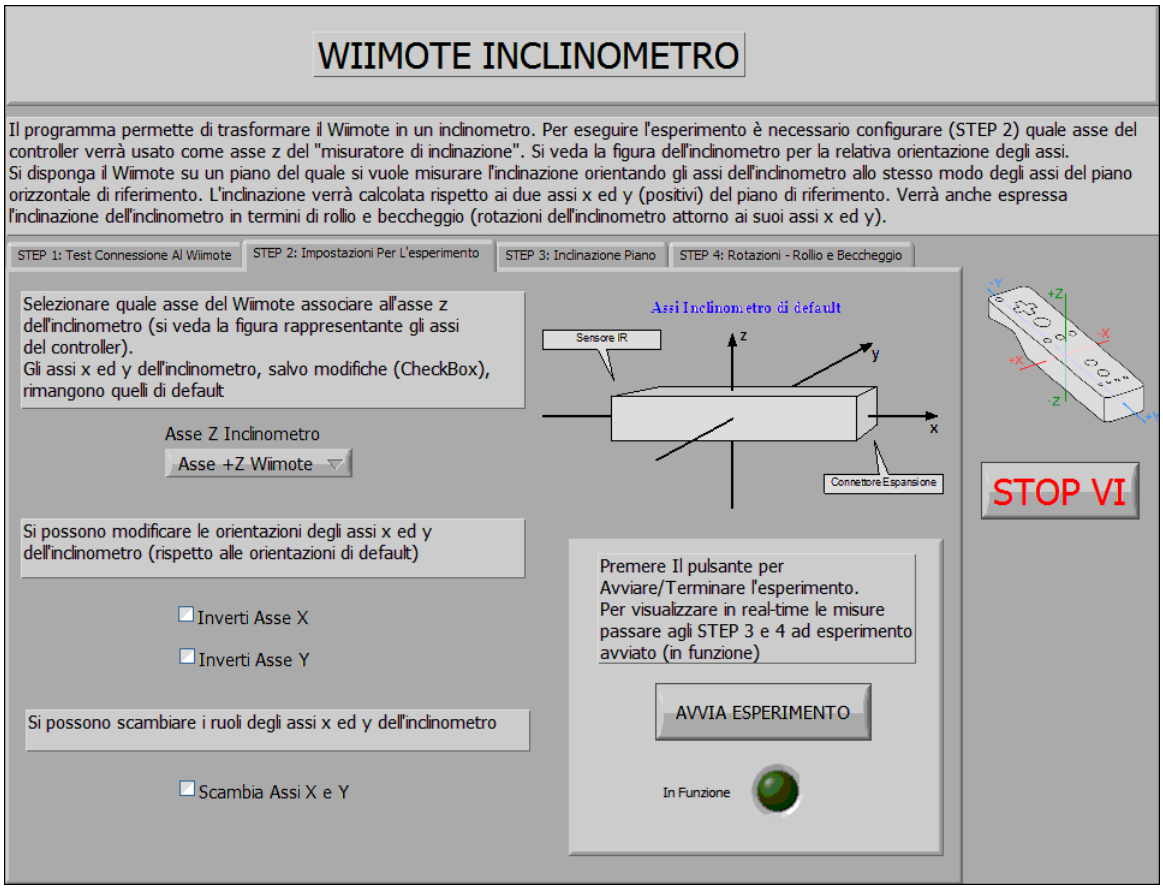


Figura 5.34: Interfaccia grafica *STEP* 2: Impostazioni per l'Esperimento

Per avviare l'esperimento è sufficiente premere il pulsante "Avvia Esperimento". Se il programma sta ricevendo correttamente i dati dal dispositivo, il led identificato con l'etichetta "In Funzione" inizia a lampeggiare. Si può passare agli *STEP* 3 e 4 per visualizzare rispettivamente gli angoli di inclinazione del piano o gli angoli di rotazione dell'inclinometro. Per interrompere l'esperimento si può tornare allo *STEP* 2 e cliccare sul pulsante "Termina Esperimento" oppure terminare direttamente il programma grazie al pulsante "STOP VI".
Nello *STEP* 3 l'inclinazione stimata del piano è riportata sia numericamente che in modo grafico. I risultati numerici riportano gli angoli di inclinazione del piano rispetto ad entrambi gli assi, sia in radianti che in gradi. Ai valori misurati è associata un'incertezza, della quale si parlerà in seguito. Tra i risultati grafici trova posto la rappresentazione tridimensionale del piano inclinato, la cui inclinazione varia in real-time in relazione all'inclinazione del dispositivo. Sono mostrate con lo stesso sistema anche le inclinazioni dei singoli assi x ed y separatamente. Infine il led "Inclinometro Capovolto" permette di segnalare il capovolgimento del dispositivo (angoli superiori a $\pm 90°$). In figura 5.35 è mostrata la schermata associata allo *STEP* 3.

Figura 5.35: Interfaccia grafica *STEP* 3: Inclinazione Piano

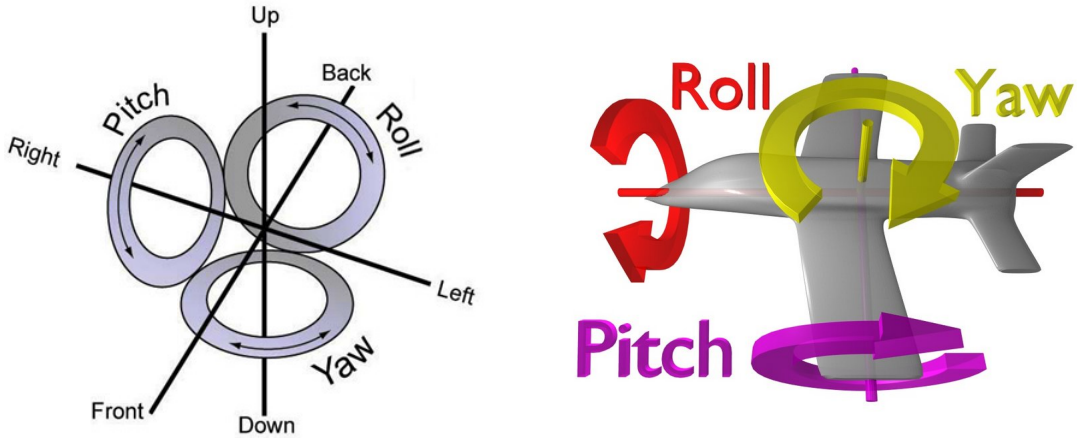
Nello *STEP 4* gli angoli misurati vengono descritti attraverso i termini "nautici" di rollio e beccheggio: in questa rappresentazione gli angoli riportati nei risultati dell'esperimento si riferiscono a rotazioni dell'*inclinometro* rispetto ai suoi assi. Quella che prima era un'inclinazione rispetto all'asse x_{ref} viene ora rappresentata come rotazione attorno all'asse y, allo stesso modo un'inclinazione rispetto ad y_{ref} diventa una rotazione attorno ad x.
Gli assi x ed y di default usati nell'*inclinometro* vengono invertiti in modo da verificare le definizioni di rollio [36] e beccheggio [37].

Rollio Il rollio (in inglese ***roll***) si definisce come l'oscillazione di un veicolo (ad esempio un'imbarcazione, un aereo...) intorno al proprio asse longitudinale. Si definisce per convenzione l'asse longitudinale x orientato da *poppa* (o la parte posteriore del veicolo) a *prua* (o la parte anteriore del veicolo). Nel caso dell'*inclinometro* (con gli assi predefiniti) la parte posteriore è costituita dal connettore per le espansioni, la parte anteriore invece è identificata dalla *IR Camera*.
L'angolo di rollio è definito come l'angolo di rotazione intorno all'asse di rollio, è positivo per rotazioni orarie e negativo per rotazioni antiorarie ed è compreso nell'intervallo $[-180°, +180°]$. Si noti che nel caso di un velivolo, a differenza di un'imbarcazione, sono ammesse rotazioni maggiori di $\pm 90°$ (volo capovolto). Nel caso particolare del programma realizzato la sua escursione è limitata all'intervallo $[-90°, +90°]$.

Beccheggio Il beccheggio (in inglese ***pitch***) si definisce come l'oscillazione di un veicolo (ad esempio un'imbarcazione, un aereo...) intorno al proprio asse trasversale. Si definisce per convenzione l'asse trasversale y orientato da *dritta* (o la parte destra del veicolo) a *sinistra* (o la parte sinistra del veicolo).

L'angolo di beccheggio è definito come l'angolo di rotazione intorno all'asse di beccheggio, come nel caso del rollio è positivo per rotazioni orarie e negativo per rotazioni antiorarie. L'angolo di beccheggio è compreso nell'intervallo $[-90°, +90°]$.

Per concludere, anche la rotazione attorno all'asse verticale ha un corrispondente termine nautico, **l'imbardata** (in inglese ***yaw***), che definisce l'oscillazione attorno all'asse z orientato dal basso verso l'alto [38]. Nel caso dell'*inclinometro* posto con l'asse z perfettamente verticale l'imbardata non altera le componenti dell'accelerazione gravitazionale, quindi l'angolo di imbardata non risulta misurabile. Diverso è il caso in cui l'*inclinometro* risulta già inclinato, ad ogni modo questa rotazione non verrà presa in considerazione. In figura 5.36a è schematizzato il significato di rollio e beccheggio, in figura 5.36b si possono osservare le definizioni applicate al caso di un aereo.



(a) Rappresentazione schematica delle definizioni

(b) Definizioni applicate al caso di un aereo

Figura 5.36: Rollio Beccheggio e Imbardata

Nello *STEP 4* Gli angoli di rollio e beccheggio sono rappresentati sia in forma numerica che in forma grafica: una rappresentazione tridimensionale dell'*inclinometro* visto secondo la direzione dell'asse longitudinale mostra in real-time le rotazioni applicate al dispositivo. Rollio e beccheggio sono rappresentati anche separatamente attraverso gli indicatori *Gauge* (per il rollio) e *Tank* (per il beccheggio). In figura 5.37 è mostrata la schermata associata a questo *STEP*.

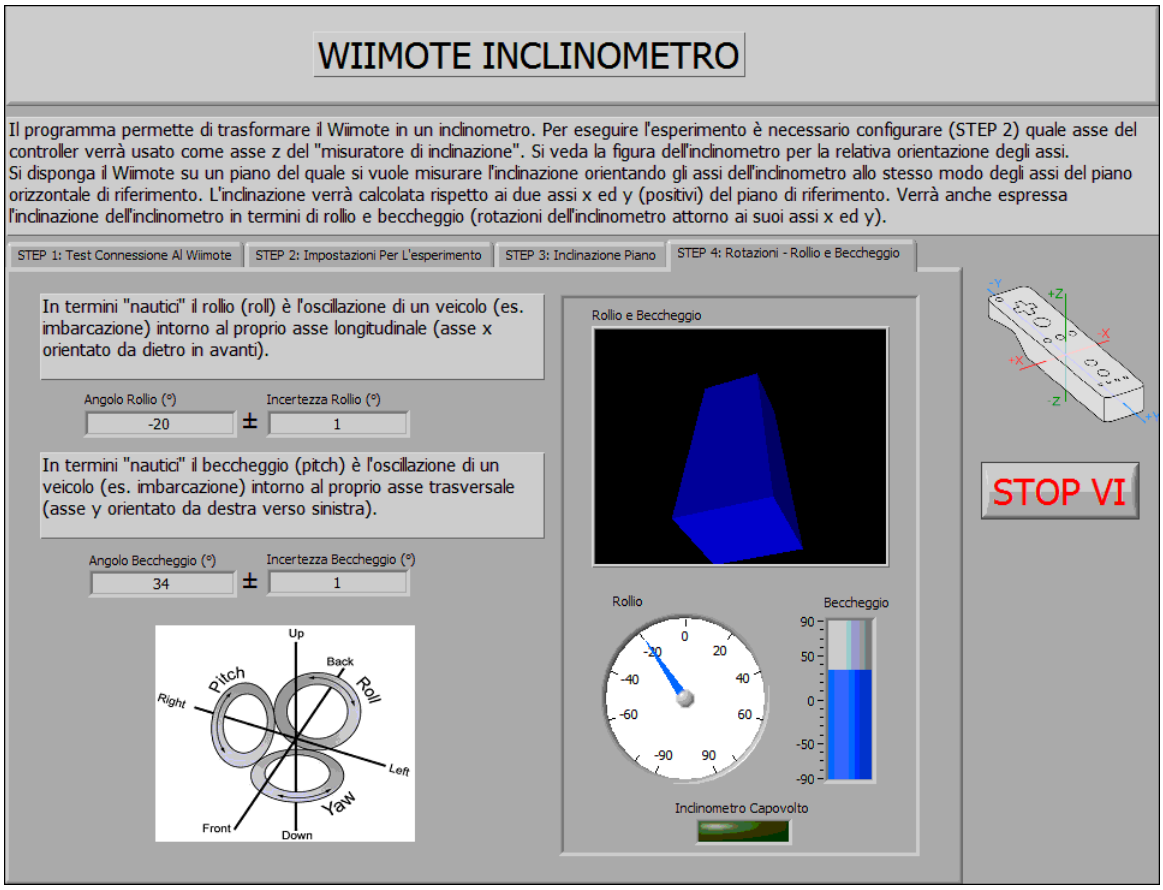


Figura 5.37: Interfaccia grafica *STEP* 4: Rotazioni - Rollio e Beccheggio

5.2.3 Analisi del Codice

Ad ogni passo del *TAB Control* corrisponde un *caso* della struttura *CASE* ad esso associata. Il codice associato allo *STEP* 1 è lo stesso usato nell'esperimento della caduta del corpo (sezione 5.1.3) ed è riportato nella figura 5.11. La sua funzione è quella di acquisire e visualizzare in *real-time* i dati provenienti dal Wiimote. Agli *STEP* 3 e 4 non sono associate operazioni in quanto utilizzati solo per la visione dei risultati dell'esperimento. Il codice principale del programma è contenuto completamente nel *caso* corrispondente allo *STEP* 2, vedi figura 5.38.

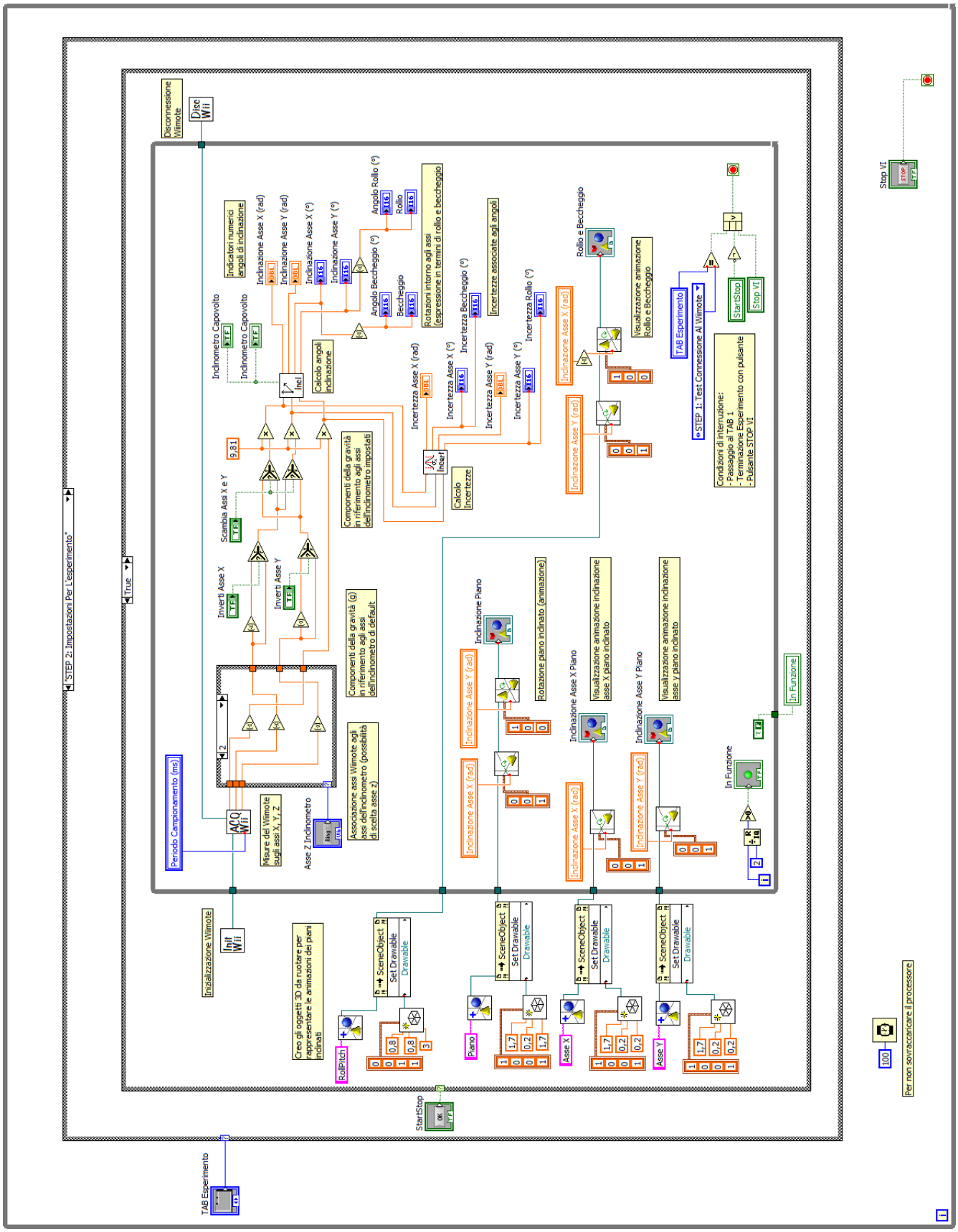


Figura 5.38: Codice principale del programma (*STEP* 2)

Il codice può essere diviso in tre parti logicamente separate:

- Associazione degli assi del Wiimote agli assi dell'*inclinometro* in base ai parametri dell'esperimento.
  Attraverso una struttura *CASE,* associata al controllo *Ring* per la scelta dell'asse del Wiimote da utilizzare come asse z dell'*inclinometro*, si distinguono i quattro casi possibili (+Z, -Z, +X, -X). Per ognuno di essi si effettuano le giuste associazioni tra i valori di accelerazione riferiti agli assi del controller (in ingresso al *CASE*) ed i valori di accelerazione riferiti agli assi di default dell'*inclinometro* (in uscita dal *CASE*). Successivamente, in base alle opzioni impostate di "inversione asse x", "inversione asse y" e "scambio assi", le accelerazioni associate agli assi di default dell'*inclinometro* vengono riferite ai nuovi assi impostati. Queste rappresentano le componenti dell'accelerazione gravitazionale che vengono usate per la determinazione degli angoli di inclinazione e per il calcolo dell'incertezza ad essi associata. In figura 5.39 sono mostrati i passaggi descritti.

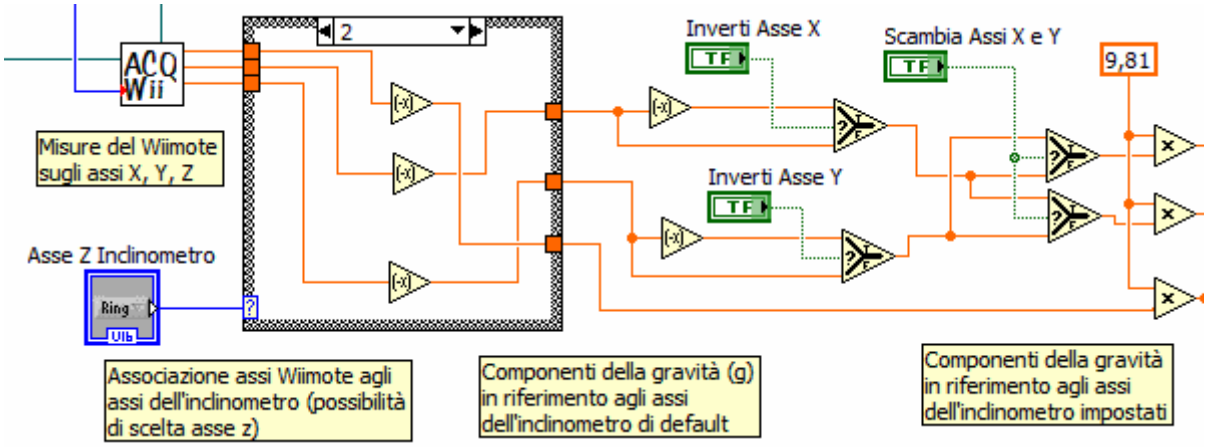
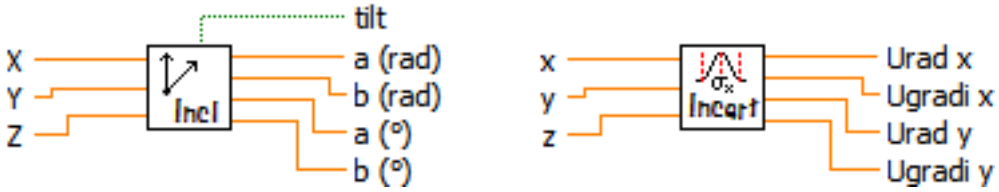


Figura 5.39: Codice che esegue l'associazione tra assi del Wiimote ed assi dell'inclinometro

- Calcolo degli angoli di inclinazione e relative incertezze.
  Le operazioni avvengono attraverso due Sub-VI. Il primo, denominato *Incl* (figura 5.40a), si occupa del calcolo degli angoli di inclinazione (in radianti ed in gradi) a partire dalle componenti di accelerazione gravitazionale X, Y e Z riferite agli assi impostati dell'*inclinometro* (nella sottosezione 5.2.1 tali componenti erano denominate rispettivamente $\hat{g}_x$, $\hat{g}_y$, $\hat{g}_z$). Il secondo, denominato *Incert* (figura 5.40b), si occupa del calcolo delle incertezze associate agli angoli di inclinazione per entrambe le unità di misura, sempre a partire dalle componenti X, Y e Z.



(a) Sub-VI *Incl* (b) Sub-VI *Incert*

Figura 5.40: Sub-VI per il calcolo degli angoli di inclinazione e relative incertezze

- Visualizzazione dei risultati numerici e grafici (animazioni tridimensionali).
  I valori in uscita da *Incl* ed *Incert* vengono visualizzati attraverso gli indicatori numerici visibili negli STEP 2 e 3. I valori in radianti degli angoli calcolati vengono inoltre utilizzati per applicare le rotazioni ai solidi 3D usati nelle rappresentazioni grafiche (*Scene Node*).

Verranno analizzati ora in dettaglio i SUB-VI *Incl* ed *Incert*, fulcro dell'applicazione realizzata.

Calcolo degli angoli di inclinazione

Il calcolo degli angoli di inclinazione è effettuato dal SUB-VI *Incl*, il cui codice è riportato in figura 5.41.

Come si può vedere tutte le operazioni di calcolo sono eseguite da un *Formula Node*. Le variabili in ingresso alla struttura sono le componenti dell'accelerazione gravitazionale sugli assi dell'*inclinometro* (X, Y, Z), quelle in uscita (a, b, tilt) sono rispettivamente i valori degli angoli di inclinazione in radianti (poi convertiti anche in gradi) ed il flag indicatore dello stato di capovolgimento del dispositivo. Il codice "C-Like" contenuto nel *Formula Node* implementa la teoria discussa nella sottosezione 5.2.1, in particolare le formule 5.2.7 e 5.2.8. Si osservi che in caso di *inclinometro* orientato verso il basso ($Z > 0$) la variabile *tilt* viene posta a 1 e gli angoli di inclinazione vengono fissati ai valori estremi, $\pm 90°$ e 0 a seconda dei valori delle componenti X ed Y (questo è stato fatto principalmente per avere una migliore rappresentazione grafica dei risultati nel caso in cui il controller venga ruotato piu di 90°).

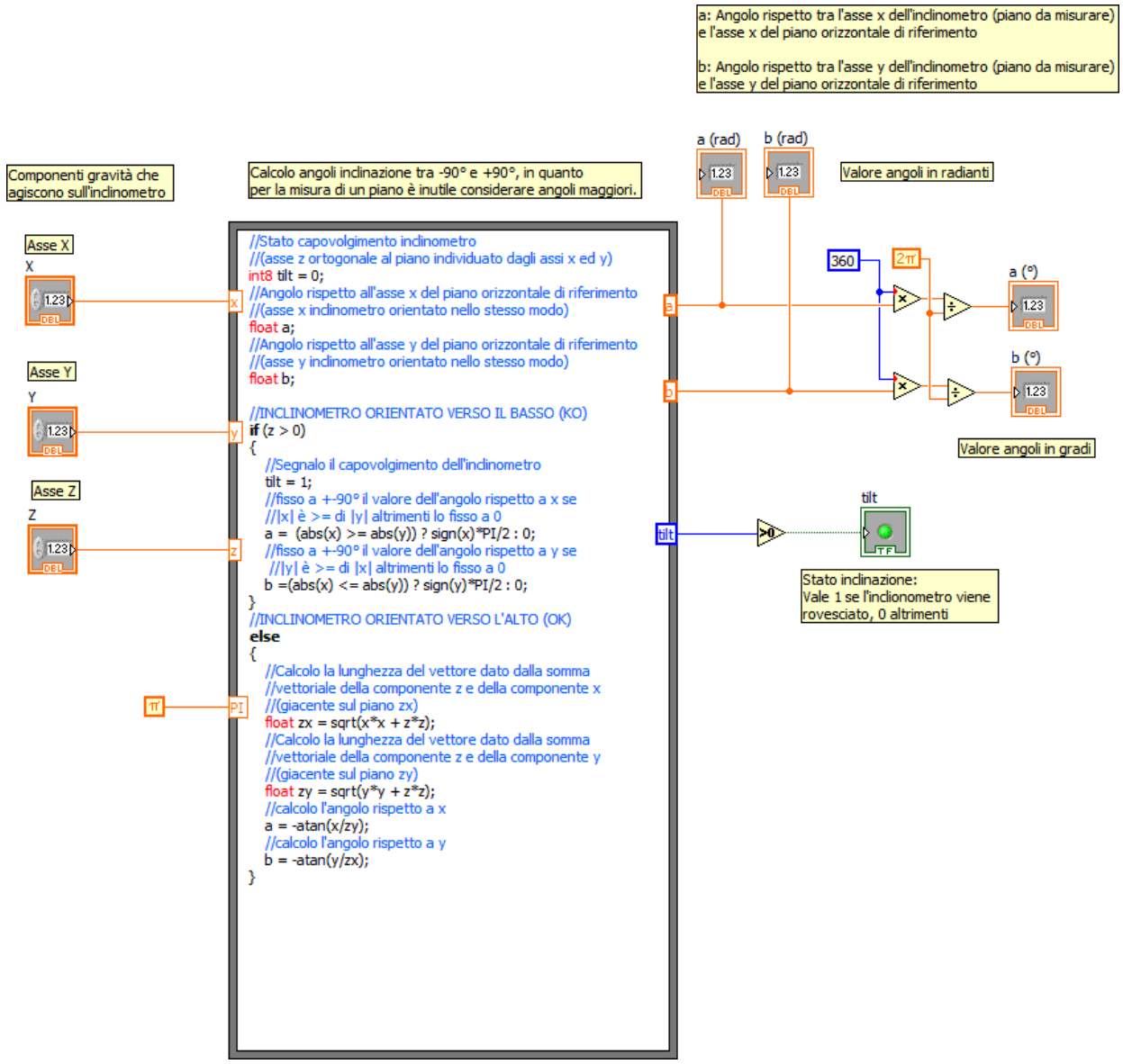


Figura 5.41: Codice del Sub-VI *Incl*

Calcolo Incertezze

Per il calcolo delle incertezze si considera solamente l'incertezza dovuta al processo di quantizzazione effettuato durante la conversione analogico-digitale dei segnali in tensione provenienti dall'accelerometro interno al Wiimote.

L'incertezza estesa associata alla misura di accelerazione su un singolo asse è uguale per tutti e tre gli assi e vale, in riferimento alla formula 4.2.2(sottosezione 4.2.4):

$$U_{acc} = \frac{u_{acc}}{k} = 0.15 m/s^2 \tag{5.2.10}$$

dove $k = \sqrt{3}$ è il fattore di copertura (distribuzione di probabilità uniforme) e u_{acc} è l'incertezza standard associata alla misura di accelerazione.

Le incertezze estese U_α e U_β (in radianti) sugli angoli α e β (in radianti) misurati, vengono calcolate applicando la legge di propagazione delle incertezze (ovvero calcolando le incertezze combinate) come descritto nella GUM (*Guide to the expression of uncertainty in measurament*, si veda la [29]).

Le incertezze standard per gli angoli α e β sono definite come le deviazioni standard, rispettivamente

u_α ed u_β, associate alle misure. Si possono calcolare le varianze associate alle misure di α e di β attraverso le espressioni:

$$u_\alpha^2 = \left(\frac{\partial\alpha}{\partial\hat{g}_x}\right)^2 \cdot u_{acc}^2 + \left(\frac{\partial\alpha}{\partial R_{YZ}}\right)^2 \cdot u_{yz}^2 = u_{acc}^2 \cdot \left[\left(\frac{\partial\alpha}{\partial\hat{g}_x}\right)^2 + \left(\frac{\partial\alpha}{\partial R_{YZ}}\right)^2\right] \tag{5.2.11}$$

$$u_\beta^2 = \left(\frac{\partial\beta}{\partial\hat{g}_y}\right)^2 \cdot u_{acc}^2 + \left(\frac{\partial\beta}{\partial R_{XZ}}\right)^2 \cdot u_{xz}^2 = u_{acc}^2 \cdot \left[\left(\frac{\partial\beta}{\partial\hat{g}_y}\right)^2 + \left(\frac{\partial\beta}{\partial R_{XZ}}\right)^2\right] \tag{5.2.12}$$

dove $R_{XZ} = \sqrt{\hat{g}_x{}^2 + \hat{g}_z{}^2}$ è il denominatore nell'espressione 5.2.8, mentre $R_{YZ} = \sqrt{\hat{g}_y{}^2 + \hat{g}_z{}^2}$ è il denominatore nell'espressione 5.2.7
Come prima cosa vengono determinate le incertezze associate ad R_{XZ} ed R_{YZ}. Le varianze associate alle misure sono date dalle relazioni:

$$u_{xz}^2 = \left(\frac{\partial R_{XZ}}{\partial\hat{g}_x}\right)^2 \cdot u_{acc}^2 + \left(\frac{\partial R_{XZ}}{\partial\hat{g}_z}\right)^2 \cdot u_{acc}^2 = u_{acc}^2 \cdot \left[\left(\frac{\partial R_{XZ}}{\partial\hat{g}_x}\right)^2 + \left(\frac{\partial R_{XZ}}{\partial\hat{g}_z}\right)^2\right] \tag{5.2.13}$$

$$u_{yz}^2 = \left(\frac{\partial R_{YZ}}{\partial\hat{g}_y}\right)^2 \cdot u_{acc}^2 + \left(\frac{\partial R_{YZ}}{\partial\hat{g}_z}\right)^2 \cdot u_{acc}^2 = u_{acc}^2 \cdot \left[\left(\frac{\partial R_{YZ}}{\partial\hat{g}_y}\right)^2 + \left(\frac{\partial R_{YZ}}{\partial\hat{g}_z}\right)^2\right] \tag{5.2.14}$$

eseguendo le derivate parziali e risolvendo i quadrati si ottiene:

$$u_{xz}^2 = u_{acc}^2 \cdot \left[\frac{\hat{g}_x{}^2}{\hat{g}_x{}^2 + \hat{g}_z{}^2} + \frac{\hat{g}_z{}^2}{\hat{g}_x{}^2 + \hat{g}_z{}^2}\right] = u_{acc}^2 \tag{5.2.15}$$

$$u_{yz}^2 = u_{acc}^2 \cdot \left[\frac{\hat{g}_y{}^2}{\hat{g}_y{}^2 + \hat{g}_z{}^2} + \frac{\hat{g}_z{}^2}{\hat{g}_y{}^2 + \hat{g}_z{}^2}\right] = u_{acc}^2 \tag{5.2.16}$$

Le incertezze standard associate a R_{XZ} ed R_{YZ} risultano quindi essere $u_{xz} = u_{yz} = u_{acc}$.
Ricordando le espressioni di α e β, riportate nelle formule 5.2.7 e 5.2.8 rispettivamente, eseguendo le derivate parziali e risolvendo i quadrati si ottiene:

$$u_\alpha^2 = u_{acc}^2 \cdot \left[\frac{R_{YZ}^2}{\left(\hat{g}_x{}^2 + R_{YZ}^2\right)^2} + \frac{\hat{g}_x{}^2}{\left(\hat{g}_x{}^2 + R_{YZ}^2\right)^2}\right] = \frac{u_{acc}^2}{X^2 + R_{YZ}^2} = \frac{u_{acc}^2}{\hat{g}_x{}^2 + \hat{g}_y{}^2 + \hat{g}_z{}^2} \tag{5.2.17}$$

$$u_\beta^2 = u_{acc}^2 \cdot \left[\frac{R_{XZ}^2}{\left(\hat{g}_y{}^2 + R_{XZ}^2\right)^2} + \frac{\hat{g}_y{}^2}{\left(\hat{g}_y{}^2 + R_{XZ}^2\right)^2}\right] = \frac{u_{acc}^2}{\hat{g}_y{}^2 + R_{XZ}^2} = \frac{u_{acc}^2}{\hat{g}_x{}^2 + \hat{g}_y{}^2 + \hat{g}_z{}^2} \tag{5.2.18}$$

Le incertezze estese associate alle misure degli angoli di inclinazione risultano quindi essere:

$$U_\beta = U_\alpha = \frac{U_{acc}}{\sqrt{\hat{g}_x{}^2 + \hat{g}_y{}^2 + \hat{g}_z{}^2}} = \frac{U_{acc}}{g} \tag{5.2.19}$$

Considerando che $g \simeq 10 m/s^2$ l'incertezza sull'angolo è circa 10 volte inferiore a quella relativa alla misura di accelerazione. Considerando infine che $U_{acc} \simeq 0.15 m/s^2$ risulta che $U_\alpha = U_\beta \simeq 0.015 rad \simeq$

0.9°.
La 5.2.19 viene calcolata nel Sub-VI *Incert*, il cui codice è mostrato in figura 5.42. Le operazioni vengono eseguite all'interno di un *Formula Node* come nel caso del Sub-VI per il calcolo degli angoli di inclinazione. Le variabili in ingresso alla struttura sono le componenti X, Y e Z dell'accelerazione gravitazionale sugli assi dell'*inclinometro*, le variabili in uscita sono le incertezze estese associate agli angoli calcolati α e β, sia per la rappresentazione in radianti che per la rappresentazione in gradi.

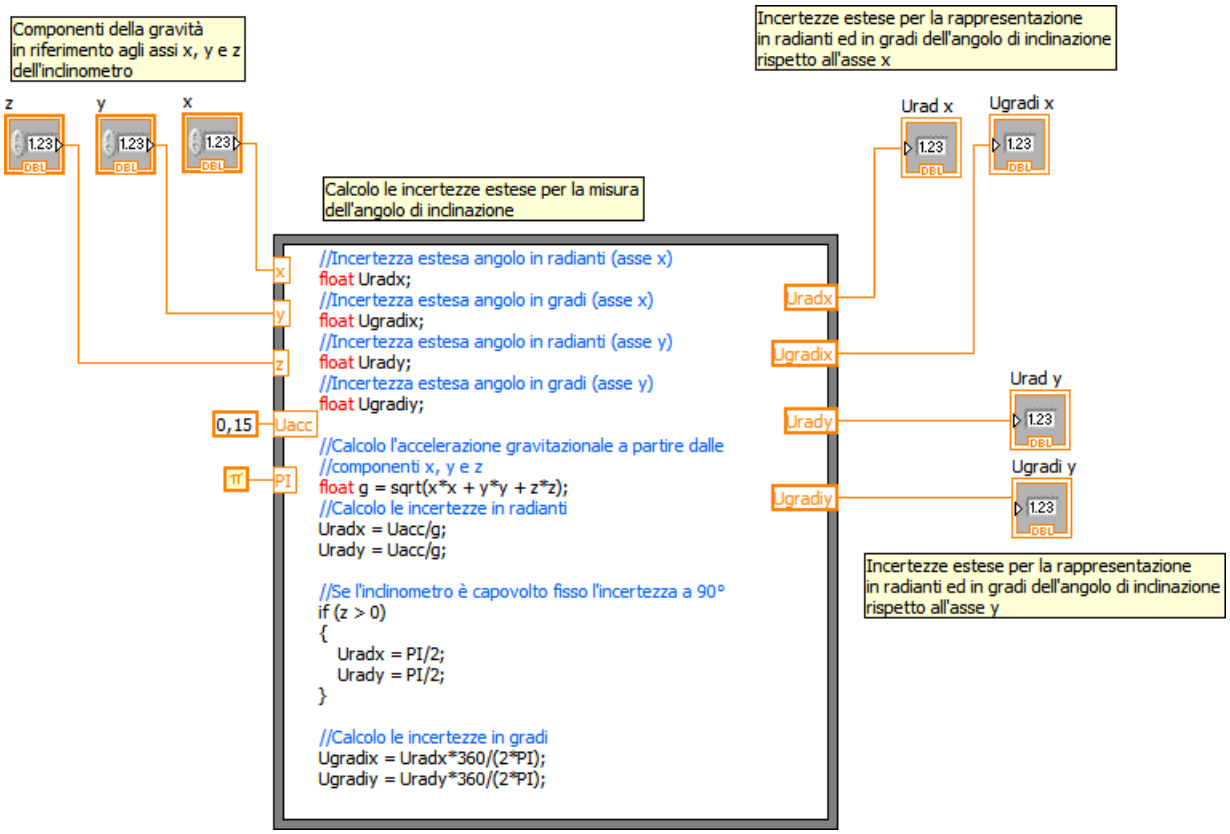


Figura 5.42: Codice Sub-VI *Incert*

È importante ricordare che vi è anche un contributo di incertezza associato al sensore, sebbene sia stato per semplicità trascurato. La misura dell'inclinazione di un piano è inoltre afflitta da un errore sistematico dovuto all'allineamento del *package* (si veda la sezione 2.4) ed all'allineamento del PCB all'interno del Wiimote: viene misurata un'inclinazione non nulla anche se il controller è tenuto perfettamente orizzontale.

5.3 WIIMOTE FREQUENZIMETRO

L'applicazione realizzata permette di misurare la frequenza di un segnale di accelerazione periodico acquisito tramite Wiimote. Il segnale viene generato applicando un movimento meccanico periodico durante l'esecuzione dell'esperimento, questo può essere fatto ad esempio:

- Attraverso un movimento oscillatorio manuale, ad esempio muovendo il Wiimote prima verso sinistra e poi verso destra ciclicamente e ritmicamente.
- Fissando il Wiimote al tergicristallo di un'automobile, caso preso in esame in questa tesi, si veda figura 5.43.
- Eccetera.

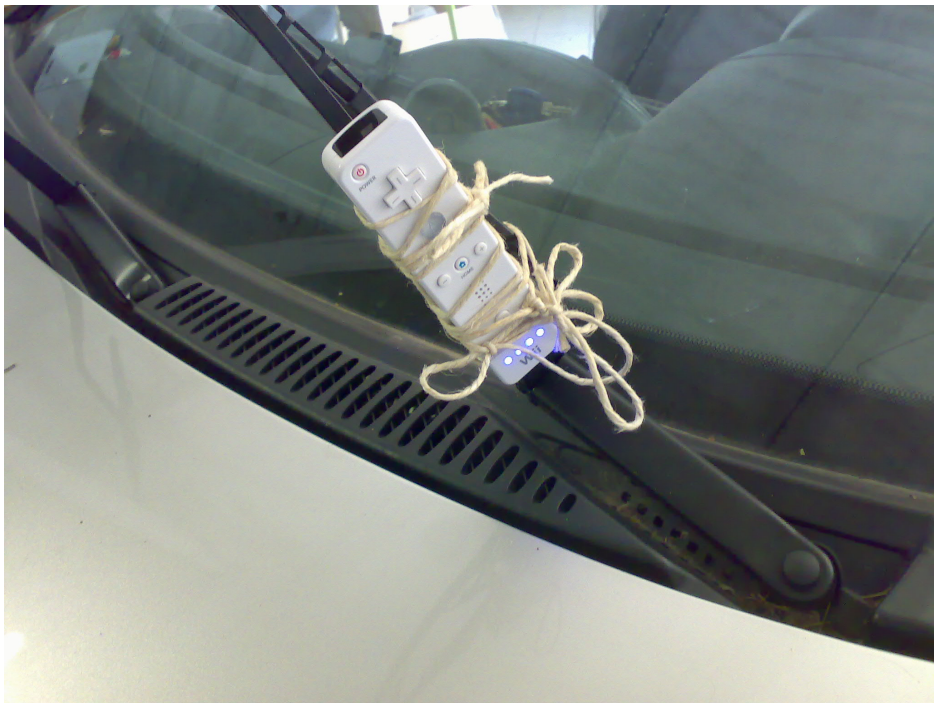


Figura 5.43: Il Wiimote fissato al tergicristallo di un'automobile

Il programma è in grado di misurare frequenze in un range compreso tra 0 (componente continua) e qualche decina di Hertz attraverso l'uso della DFT *(Discrete Fourier Trasform*), il range misurabile è quindi idoneo per l'analisi dei segnali generabili con i metodi sopra descritti (frequenze di norma comprese tra $0.2Hz$ e $5Hz$). L'utente è in grado di selezionare quale asse del controller utilizzare per l'acquisizione, in questo modo è possibile operare sull'asse che risente maggiormente del movimento applicato al dispositivo. Il segnale acquisito viene elaborato ed all'utente vengono proposti risultati sia numerici che grafici: è possibile visualizzare lo spettro del segnale in un grafico, l'ampiezza stimata della componente continua, ampiezza e frequenza stimate della componente fondamentale. Quest'ultimo risultato è caratterizzante per l'applicazione realizzata.
Per analizzare meglio il funzionamento del programma ed il ruolo dei parametri per l'esperimento impostabili dall'utente sono necessari alcuni richiami teorici sulla DFT.

5.3.1 Richiami Teorici

La trasformata di Fourier per segnali continui è definita dall'equazione [39]:

$$X(f) = \mathscr{F}(x(t)) = \int_{-\infty}^{+\infty} x(t) \cdot e^{-j2\pi ft} dt \tag{5.3.1}$$

dove $x(t)$ è il segnale nel dominio del tempo, mentre $X(f)$ è il segnale nel dominio della frequenza. Similmente alla trasformata per segnali continui, la trasformata di Fourier nel caso discreto è data dall'equazione [1]:

$$X_k = \mathscr{F}(x_k) = \frac{1}{N} \cdot \sum_{i=0}^{N-1} x_i \cdot e^{-j2\pi ik/N} \quad k = 0, 1, 2, \ldots, N-1 \tag{5.3.2}$$

dove x_k è la sequenza di campioni del segnale acquisito, X_k è la sequenza di campioni rappresentante lo spettro del segnale ed N è il numero di capioni acquisiti. N costituisce la dimesione di entrambe le sequenze x_k (dominio del tempo) ed X_k (dominio della frequenza). La sequenza X_k è composta da valori complessi, può quindi essere rappresentata distinguendo parti reale ed immaginaria oppure distinguento modulo e fase:

$$X_k = Re(X_k) + j \cdot Im(X_k) = X_{Re} + j \cdot X_{Im} \tag{5.3.3}$$

$$X_k = |X_k| \cdot e^{j\varphi_{X_k}} \tag{5.3.4}$$

dove X_{Re} è la parte reale, X_{Im} è la parte immaginaria, $|X_k|$ è il modulo e φ_{X_k} è la fase (sono queste ultime ad essere maggiormente utilizzate nella rappresentazione grafica di X_k). Vale inoltre la relazione, nel caso in cui x è usa sequenza composta da valori reali:

$$X_{N-i} = X_{-i} = X_i^* \tag{5.3.5}$$

ovvero la metà superiore della sequenza X_k rappresenta le frequenze negative, trascurabili in quanto per segnali reali vale la simmetria hermitiana. Si possono quindi rappresentare solo i primi $M = \left\lfloor \frac{N+1}{2} \right\rfloor$ valori della sequenza.
I campioni della sequenza x_k sono ottenuti campionando il segnale continuo $x(t)$, il numero di campioni N acquisiti dipende dalla finestra di acquisizione (durata dell'acquisizione) T_W e dal periodo di campionamento $T_s = 1/f_s$, dove f_s è la frequenza di campionamento, secondo la relazione:

$$N = \frac{T_W}{T_s} = T_W \cdot f_s \tag{5.3.6}$$

T_s rappresenta la risoluzione della scala dei tempi del segnale acquisito, ovvero i campioni della sequenza x_k sono equispaziati temporalmente della quantità T_s (la scala dei tempi per la rappresentazione del segnale va da 0 a Tw ed è quantizzata con quanto T_s). La risoluzione in frequenza è data dalla relazione:

$$F = \frac{1}{N \cdot T_s} = \frac{1}{T_W} \tag{5.3.7}$$

F rappresenta la spaziatura sull'asse delle frequenze tra i valori che compongono la sequenza X_k. Rappresentando solo i primi M campioni, l'asse delle frequenze va da 0 a $F_{ny} = M \cdot F = F_s/2$, dove F_{ny} è chiamata *frequenza di Nyquist*. Per avere una risoluzione elevata, quindi F piccola, è necessario un valore T_W elevato, quindi un'acquisizione molto lunga. Fissato T_W il numero di campioni acquisiti dipende solo dal periodo di campionamento, che deve essere scelto in modo da garantire che sia verificato il teorema del campionamento di Nyquist:

$$F_s \geq 2 \cdot B \tag{5.3.8}$$

dove B è la banda del segnale acquisito. Un vincolo sulla massima frequenza di campionamento impone un vincolo sulla massima banda del segnale, nel caso limite deve essere:

$$B_{max} = F_s/2 = F_{ny} \tag{5.3.9}$$

La massima frequenza (teorica) rappresentabile nel modo corretto (assenza fenomeni di *aliasing*) è pari a F_{ny}. Nel caso del Wiimote il minimo periodo di campionamento è pari a 10ms, la massima frequenza di campionamento è pari a 100Hz, la massima frequenza teorica che può avere il segnale di accelerazione è quindi pari a 50Hz (nella pratica è molto al di sotto di tale valore). In figura 5.44 è riportato un esempio di *aliasing* causato da un sottocampionamento del segnale.

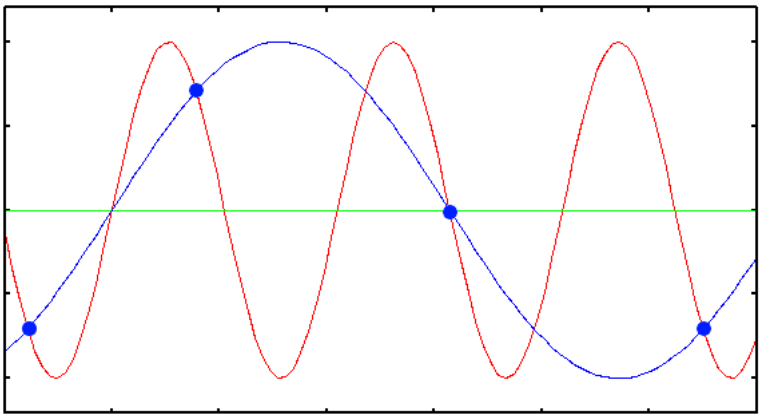


Figura 5.44: La frequenza della sinusoide campionata (in blu) risulta erroneamente minore della frequenza della sinusoide originaria (in rosso) a causa del sottocampionamento (fenomeno di aliasing)

Nelle figure 5.45 e 5.46 viene riportato un esempio illustrativo per chiarire quanto discusso: nella prima figura viene illustrato il processo di acquisizione su un segnale sinusoidale di esempio, nella seconda figura è rappresentata la DFT del segnale acquisito (sequenza X_k).

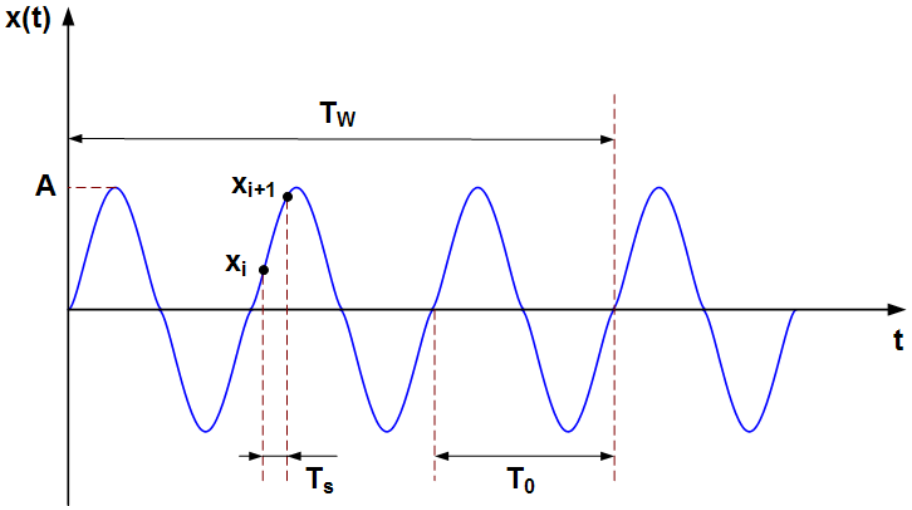


Figura 5.45: Acquisizione segnale (dominio del tempo)

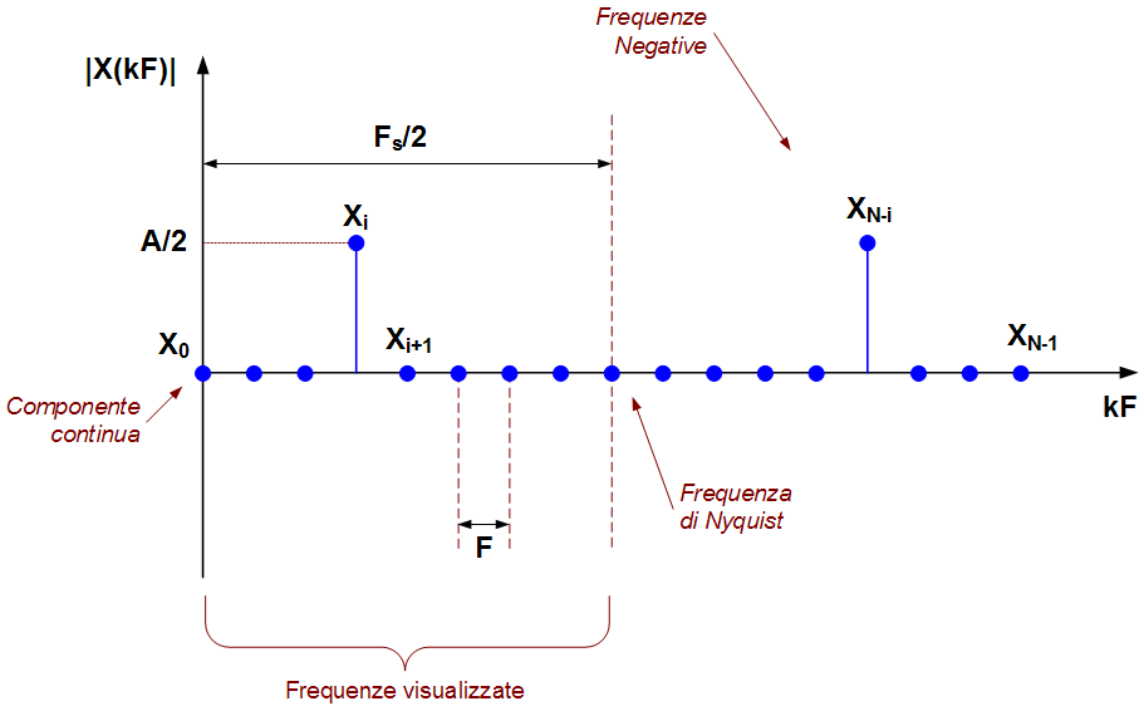


Figura 5.46: DFT del segnale (dominio della frequenza)

A causa della risoluzione finita in frequenza (in quanto la DFT è calcolata su un numero finito di campioni) le frequenze rappresentabili senza errori sono quelle multiple del quanto in frequenza F, ovvero deve valere la relazione:

$$f_0 = c \cdot F \tag{5.3.10}$$

dove $f_0 = 1/T_0$ è la frequenza del segnale acquisito (supposto per semplicità sinusoidale) e c è un numero intero. In questo caso la frequenza del segnale è riportata correttamente sulla scala delle frequenze, come mostrato in figura 5.46. Anche l'ampiezza della componente risulta essere corretta. La 5.3.10 corrisponde al caso in cui nella finestra di acquisizione T_W viene acquisito un numero intero di periodi del segnale $x(t)$ (*campionamento coerente*), ovvero se:

$$\frac{T_W}{T_0} = c \tag{5.3.11}$$

Nel caso in cui la 5.3.11 non sia verificata, come illustrato in figura 5.47, si verifica una dispersione spettrale (*spectral leakage*), la frequenza e l'ampiezza misurate del segnale acquisito risultano errate, come mostrato in figura 5.48 (si vedano [1] e [41]). L'incertezza sul valore di frequenza misurato risulta essere $\Delta F = F/2$. Gli errori dovuti alle code nel caso di dispersione spettrale non vengono considerati.

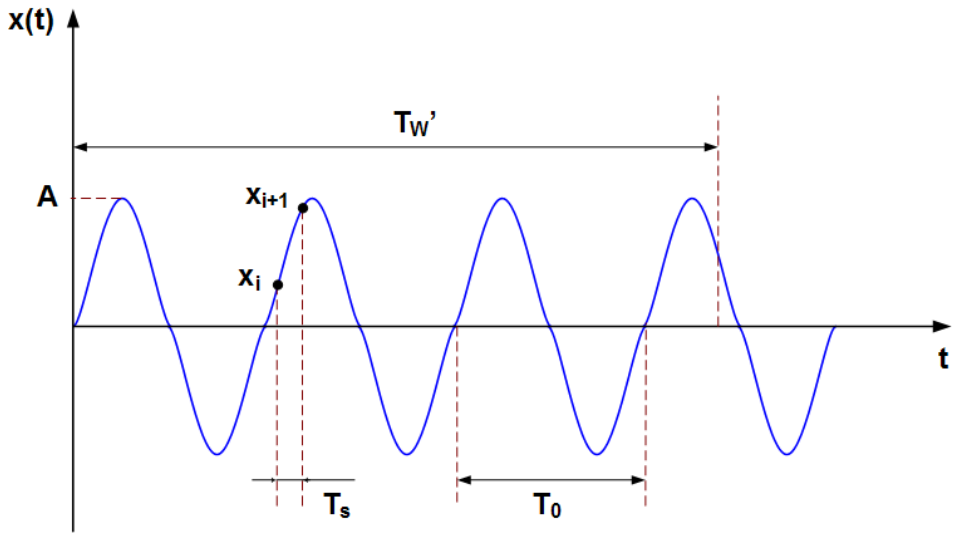


Figura 5.47: Acquisizione segnale con campionamento non coerente (dominio del tempo)

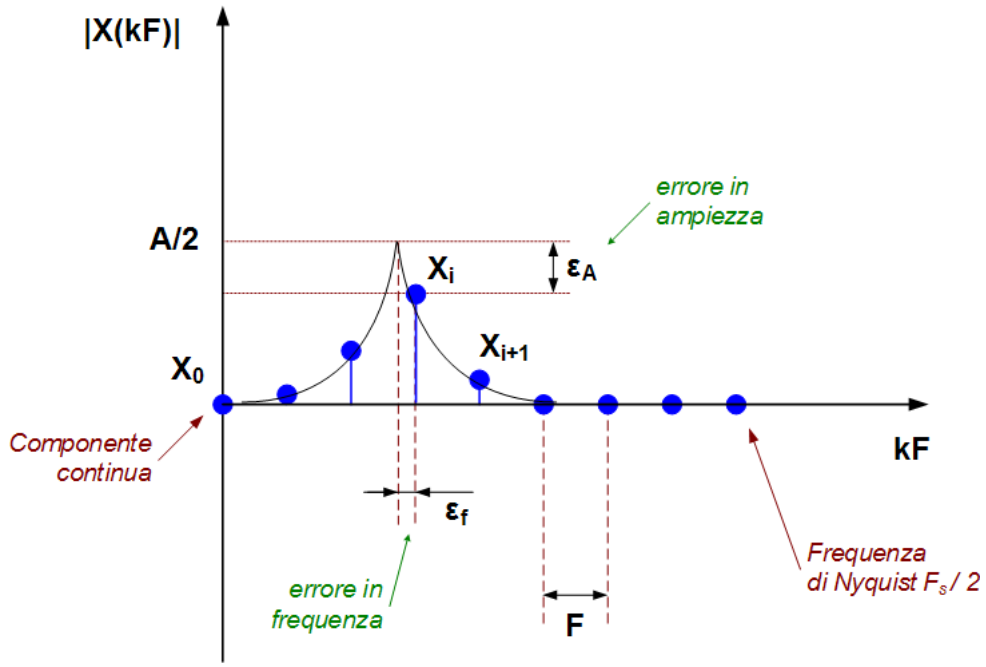


Figura 5.48: DFT del segnale (dominio della frequenza): errori in frequenza ed in ampiezza

La frequenza della componente fondamentale del segnale $x(t)$ può essere determinata semplicemente individuando l'indice k_{max} per il quale si ha il valore massimo (picco) all'interno della sequenza $|X_k|$: la frequenza stimata risulta $\hat{f}_0 = F \cdot k_{max}$ e l'ampiezza stimata risulta $\hat{A} = 2 \cdot |X_{k_{max}}|$. L'ampiezza stimata della componente continua $|\hat{X_0}|$ è data dal valore all'indice $k = 0$ della sequenza, ovvero $|\hat{X_0}| = |X_0|$.

5.3.2 Interfaccia Grafica

L'interfaccia grafica dell'applicazione è composta da un'unica finestra nella quale attraverso un *TAB Control* l'utente può eseguire l'esperimento, seguendo con ordine quattro semplici passi, identificati con i nomi *STEP* 1, *STEP* 2, *STEP* 3 e *STEP* 4.
Il primo passo (*STEP* 1) consiste nell'usuale test di connessione con il Wiimote, in questo modo è

possibile verificare se vi sono problemi di comunicazione con il dispositivo prima di procedere. La figura 5.49 mostra la finestra visualizzata all'avvio del programma, lo *STEP* evidenziato di default è il primo.

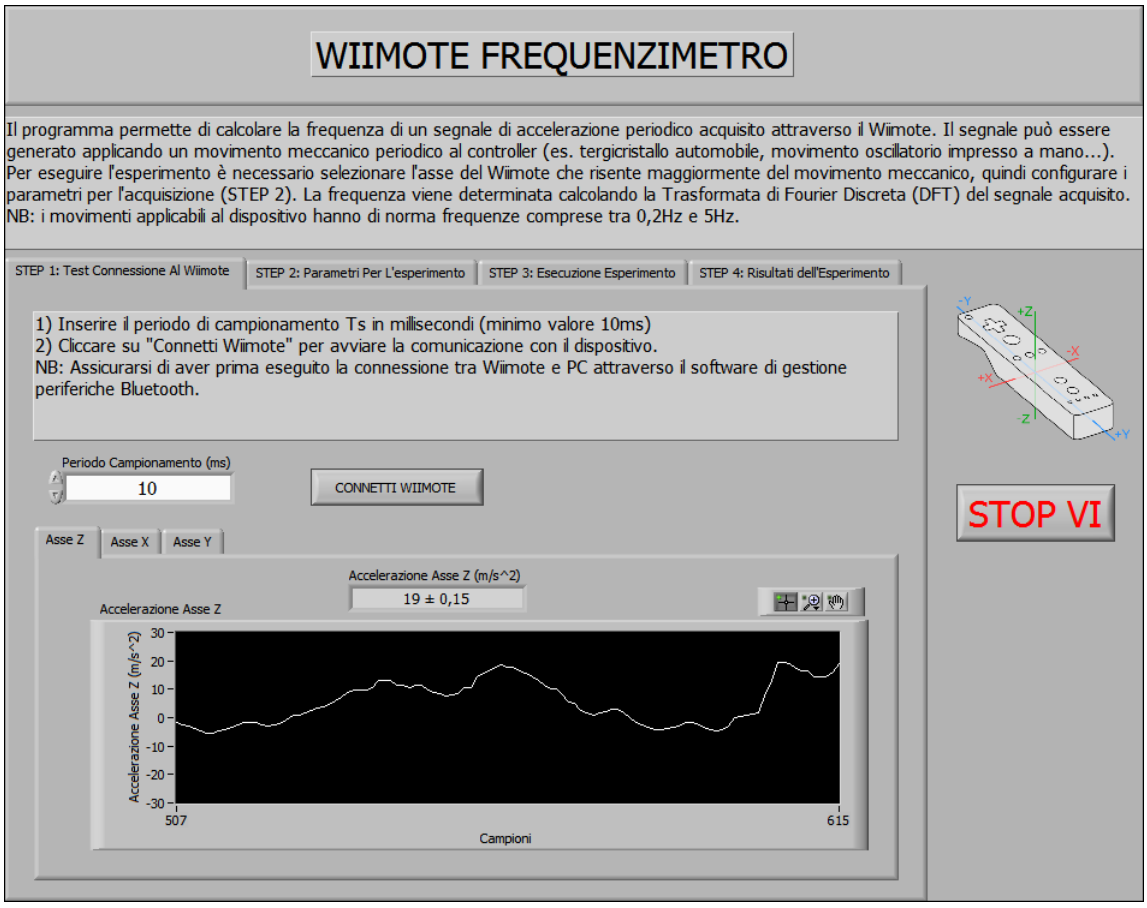


Figura 5.49: Interfaccia grafica *STEP* 1: Test Connessione al Wiimote

Nello *STEP* successivo vengono impostati i parametri per l'esperimento, la schermata visualizzata viene riportata in figura 5.50. I parametri impostabili si dividono in due categorie, quelli legati alla sorgente del segnale (Wiimote) e quelli legati all'acquisizione ed all'elaborazione del segnale:

- *Parametri di configurazione Wiimote:*
  - *"Asse acquisizione":* permette all'utente di selezionare l'asse del controller da utilizzare per l'acquisizione del segnale di accelerazione. Per un corretto funzionamento del programma è bene scegliere l'asse che risente maggiormente del movimento applicato al dispositivo.
  - *"Inverti orientazione asse":* la sua funzione è quella di invertire l'orientazione dell'asse selezionato rispetto all'orientazione di default (riportata in figura 1.8).
- *Parametri di configurazione per l'acquisizione e l'elaborazione del segnale:*
  - *"Ts (ms)"*: è il periodo di campionamento usato per l'acquisizione (in millisecondi). A differenza dei precedenti esperimenti si è scelto di inserire il periodo di campionamento tra i parametri dell'esperimento, in quanto la sua funzione svolge un ruolo di primaria importanza. Il suo valore determina, infatti, la frequenza massima teorica misurabile, in

base al teorema di Nyquist (vedi formula 5.3.8). Accanto al periodo di campionamento, nell'indicatore numerico *"Fs (Hz)*", è mostrata la frequenza di campionamento ad esso associata .

- *"Tw (s)"*: è la finestra temporale dell'acquisizione, ovvero la durata dell'acquisizione (in secondi). Questo parametro determina la risoluzione in frequenza della DFT del segnale, secondo la relazione 5.3.7. Accanto al controllo *"Tw (s)"* viene visualizzato il numero di campioni (indicatore *"N campioni"*) che è necessario acquisire con i parametri impostati, tale numero dipende dalla finestra temporale *Tw* e dal periodo di campionamento *Ts* secondo la relazione 5.3.6.
- Gli indicatori *"Frequenza Massima Teorica (Hz)"* e *"Risoluzione in frequenza (Hz)"* visualizzano gli omonimi valori (formule 5.3.9 e 5.3.7). Tutti gli indicatori numerici vengono aggiornati ad ogni modifica dei parametri da parte dell'utente.
- *"Elimina componente DC per il calcolo della frequenza"*: permette di escludere la componente DC[1] durante l'elaborazione dei dati. L'identificazione della frequenza del segnale acquisito si basa sulla ricerca del picco nel relativo spettro, se la componente continua è dominante rispetto alla componente periodica da individuare il picco rilevato sarà, erroneamente, quello relativo alla componente continua. Escludendo la componente continua dall'elaborazione si evita questo possibile errore[2]. L'opzione è attivata di default.

[1] Si farà uso, per semplicità, del termine *componente DC* sebbene l'acronimo DC (*Direct Current*) venga usualmente associato a segnali di tipo elettrico (in questa sezione si trattano invece segnali di accelerazione).

[2] L'opzione *Elimina componente DC* è stata introdotta solamente a scopo didattico, il calcolo della frequenza poteva essere eseguito direttamente sul segnale con componente media azzerata in modo trasparente all'utente.

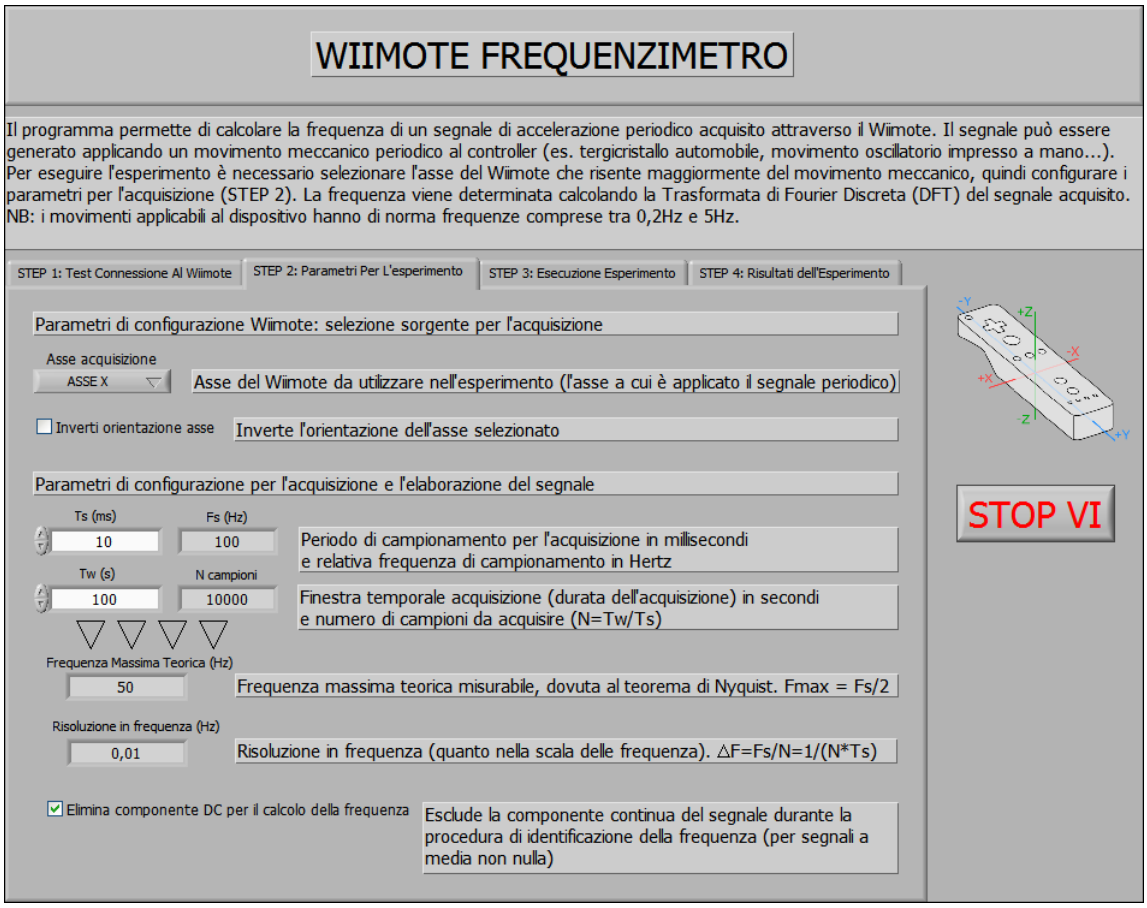


Figura 5.50: Interfaccia grafica *STEP* 2: Parametri per L'esperimento

Attraverso lo *STEP 3* è possibile avviare l'esperimento e monitorare lo stato della sua esecuzione. La schermata relativa a questo *STEP* è riportata in figura 5.51. Attraverso la pressione del pulsante *"Avvia Esperimento"* l'acquisizione dei dati viene avviata: la barra *"Avanzamento Acquisizione"* mostra la percentuale di completamento dell'esperimento, il cronometro *"Tempo"* mostra il tempo trascorso dall'inizio dell'acquisizione e l'indicatore *"Campioni Acquisiti"* visualizza il numero di campioni salvati in memoria. Nel grafico *"Acquisizione"* viene mostrato in *real-time* il segnale proveniente dal Wiimote (asse selezionato). Il termine dell'acquisizione è indicato dall'accensione del led *"Acquisizione Completata"* e dal raggiungimento del valore di fondo scala negli altri indicatori. Quando anche l'elaborazione successiva all'acquisizione ha termine si accende il secondo led *"Elaborazione Completata"*. È possibile ora procedere alla visione dei risultati dell'esperimento contenuti nello *STEP* 4.

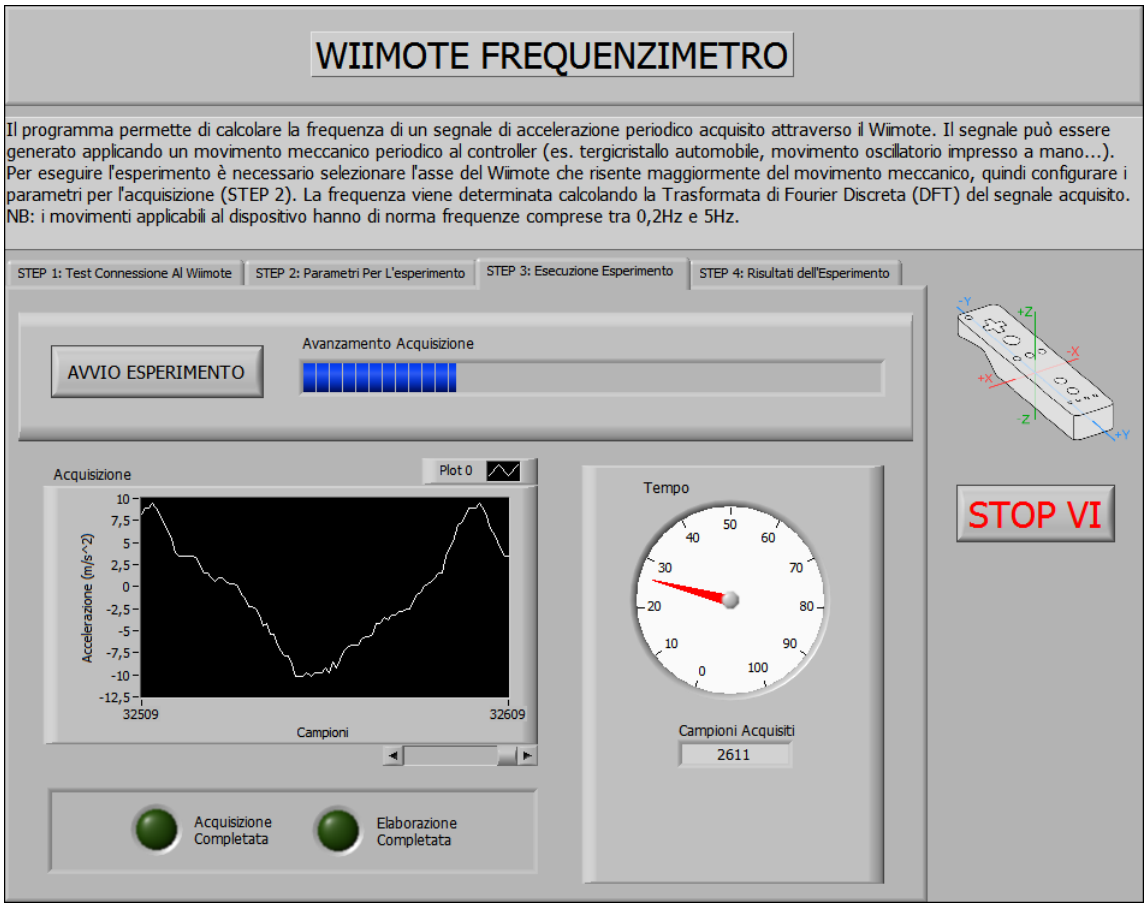


Figura 5.51: Interfaccia grafica *STEP* 3: Esecuzione Esperimento

Nello *STEP 4* vengono proposti i risultati dell'esperimento, divisi tra "numerici" e "grafici". I risultati numerici vengono mostrati nel TAB *"Risultati Numerici"* e sono:

- *"Frequenza F (Hz)"*: è la frequenza stimata del segnale di accelerazione (della componente fondamentale) dovuto al movimento del Wiimote (indicata nella sottosezione 5.3.1 con il simbolo $\hat{f}_0$). Questo risultato è il più importante e caratterizza l'applicazione realizzata.
- *"Frequenza F (periodi/min)"*: è la frequenza stimata espressa in unità $periodi/minuto$, rappresenta il numero di periodi del segnale contenuti in un minuto. Il valore è ottenuto moltiplicando la frequenza in Hz per il numero di secondi contenuti in un minuto, ovvero 60.
- *"Periodo T (ms)"*: è il periodo (in millisecondi) del segnale.
- *"Ampiezza componente a frequenza F (m/s^2)"*: è l'ampiezza stimata della componente fondamentale (frequenza $\hat{f}_0$) del segnale di accelerazione (indicata nella sottosezione 5.3.1 con il simbolo $\hat{A}$).
- *"Ampiezza della componente DC (m/s^2)"*: è l'ampiezza stimata della componente continua del segnale (indicata nella sottosezione 5.3.1 con il simbolo $|\hat{X}_0|$).

La schermata corrispondente al TAB *"Risultati Numerici"* è riportata in figura 5.52.

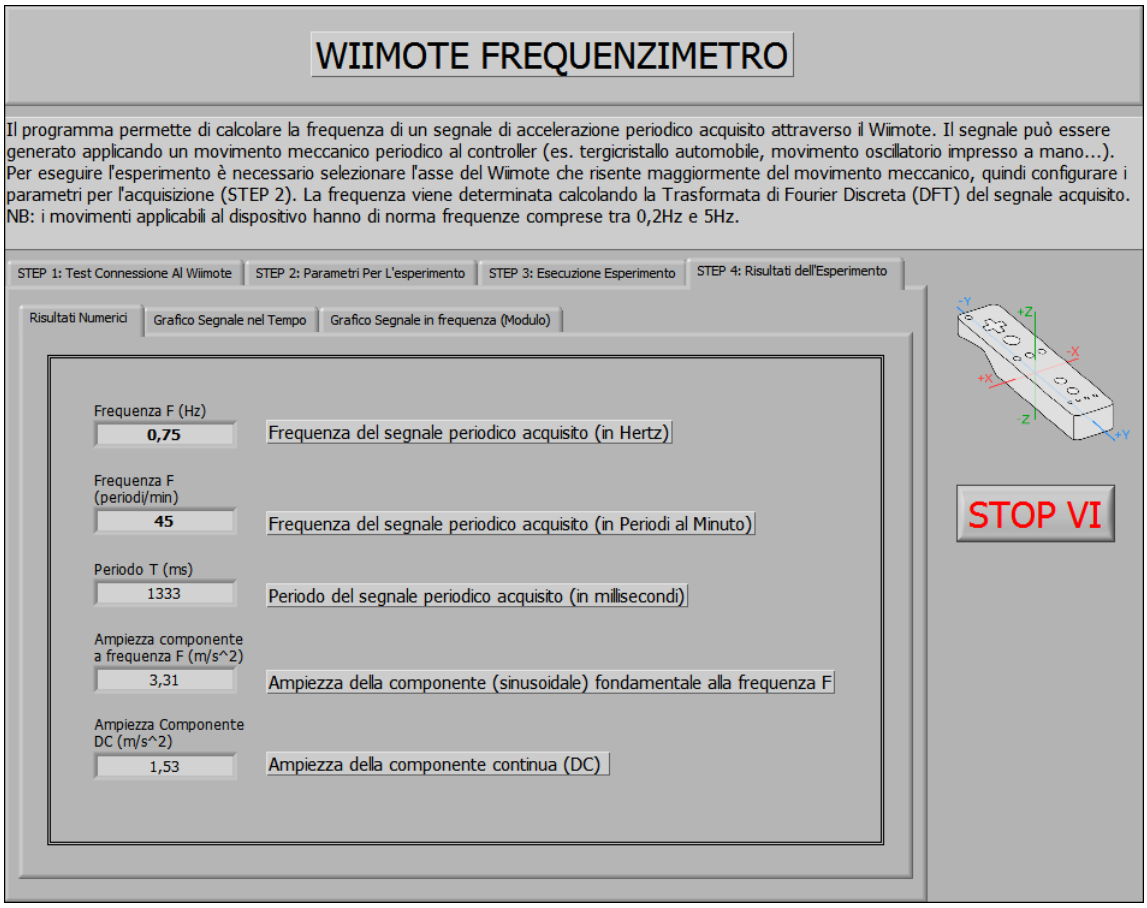


Figura 5.52: Interfaccia grafica *STEP* 4: Risultati dell'Esperimento, TAB *"Risultati Numerici"*

I risultati grafici, che vengono mostrati nei TAB rimanenti, sono:

- *"Grafico Segnale nel Tempo"*: mostra il segnale acquisito nel dominio del tempo, è riportato per permettere all'operatore di avere una più chiara visione del tipo di segnale osservato.
- *"Grafico Segnale in Frequenza (modulo)"*: rappresenta il modulo del segnale nel dominio della frequenza, ovvero la sequenza $|X_k|$ ottenuta a seguito del calcolo della DFT. Le frequenze mostrate vanno da 0 (componente continua) a $F_{ny} = F_s/2$ (frequenza di Nyquist).

La schermata corrispondente al TAB *"Grafico Segnale in Frequenza (modulo)"* è riportata in figura 5.53.

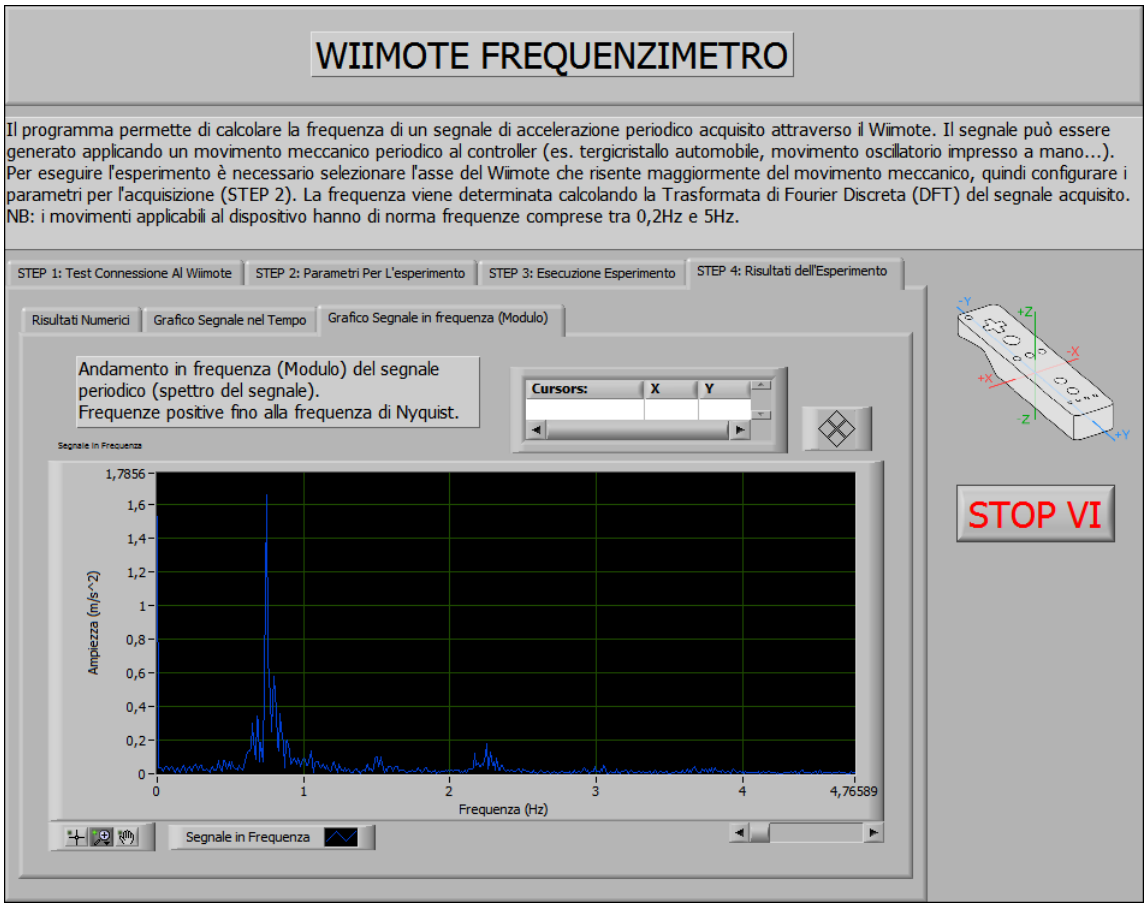


Figura 5.53: Interfaccia grafica *STEP* 4: Risultati dell'Esperimento, TAB *"Grafico Segnale in Frequenza (modulo)"*

5.3.3 Analisi del Codice

Come per le precedenti applicazioni viene utilizzata una struttura *CASE* per suddividere il codice in funzione dello *STEP* al quale è associato. Ad ogni passo del *TAB Control* corrisponde un *caso* della struttura: il codice associato allo STEP *1* rimane lo stesso usato negli esperimenti *"Wiimote in Caduta Libera"* e *"Wiimote Inclinometro"* ed è riportato nella figura 5.11. Il primo *STEP* è, come già detto, l'unica parte comune tra i programmi realizzati, la sua funzione è puramente diagnostica: l'utente può verificare il corretto funzionamento del controller visualizzando in *real-time* i dati provenienti dal dispositivo. Allo *STEP 2* è associato il codice riportato in figura 5.54, esso si occupa di aggiornare i valori degli indicatori numerici presenti nel relativo TAB, in questo modo se l'utente agisce sui parametri per l'acquisizione vengono aggiornate anche le altre informazioni associate (frequenza di campionamento, numero di campioni, frequenza massima teorica, risoluzione in frequenza).

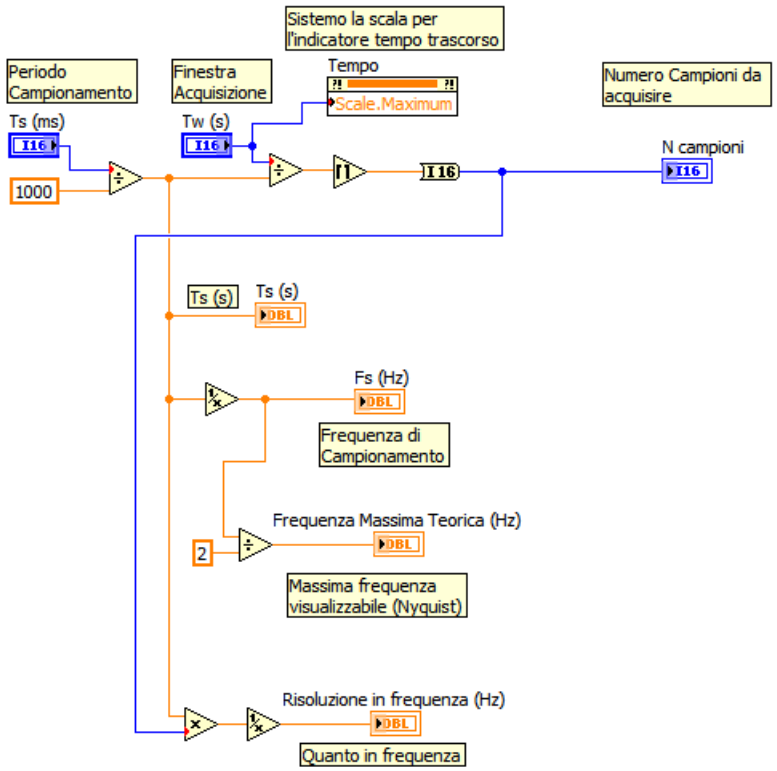


Figura 5.54: Codice associato allo *STEP 2*

Il *caso* della struttura associato allo *STEP 3* contiene la parte principale del codice dell'applicazione: esso si occupa dell'acquisizione dei dati, del calcolo della DFT sui dati acquisiti e del calcolo dei risultati. Gli indicatori che mostrano i risultati (numerici e grafici) sono contenuti nel TAB *"STEP 4: Risultati dell'Esperimento"* e non sono visibili all'utente finchè questo *STEP* non si è concluso. Il codice principale è riportato in figura 5.55. Allo *STEP 4* non è associato codice.

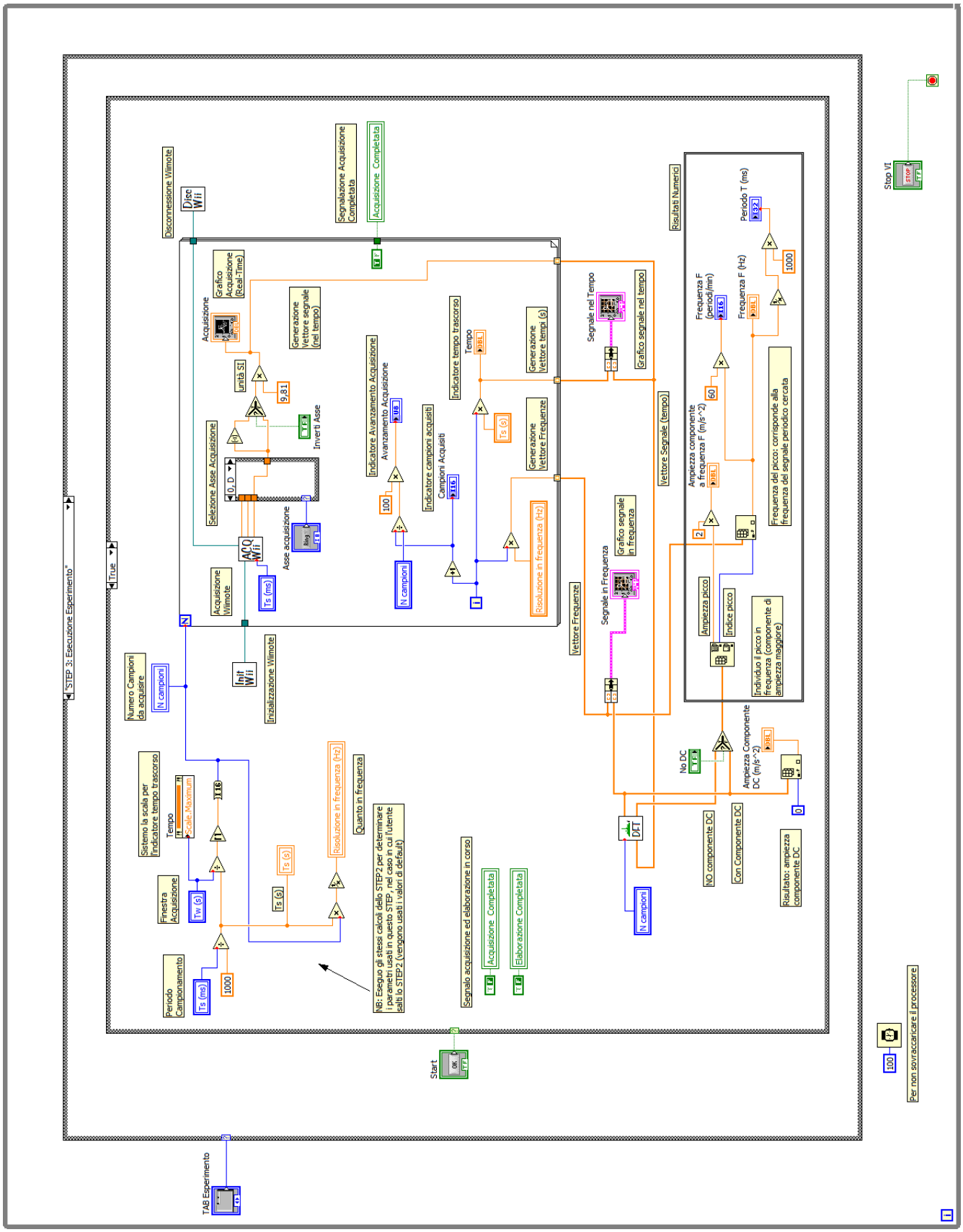


Figura 5.55: Codice associato allo *STEP 3*

Il codice principale può essere analizzato dividendolo in tre "parti fondamentali": acquisizione dati e generazione vettori tempi e frequenze, calcolo della DFT, calcolo dei risultati. Si riassumeranno di seguito le operazioni svolte da ogni "parte".

Acquisizione dati e generazione vettori

L'acquisizione dei dati consiste nell'acquisizione di N campioni con un periodo di campionamento T_s, dove N è definito dall'equazione 5.3.6, l'operazione è effettuata attraverso un ciclo *FOR* temporizzato. In uscita dalla struttura *FOR* si hanno 3 vettori di dati, generati internamente al ciclo:

1. Il vettore contenente il segnale acquisito (discreto) ottenuto per campionamento del segnale di accelerazione periodico dovuto al movimento del Wiimote. La sequenza di campioni rappresenta l'andamento dell'accelerazione durante il tempo di acquisizione T_W, viene visualizzata sul grafico *"Segnale Nel Tempo"* con la giusta scala dei tempi. La DFT viene calcolata su tale sequenza (Ncampioni).

2. Il vettore contenente i valori della scala dei tempi, viene utilizzato per la generazione del grafico *"Segnale Nel Tempo"*. La sequenza è composta da N valori del tipo $t_i = i \cdot T_s$, con $i = 0, 1, 2, \ldots N-1$ e T_s periodo di campionamento.

3. Il vettore contenente i valori della scala delle frequenze, viene utilizzato per la generazione del grafico *"Segnale in Frequenza"* e per la determinazione della frequenza del segnale. La sequenza è composta da N valori del tipo $t_i = i \cdot F$, con $i = 0, 1, 2, \ldots N-1$ ed F risoluzione in frequenza (vedere formula 5.3.7).

In figura 5.56 è riportata la porzione di codice che esegue le operazioni descritte.

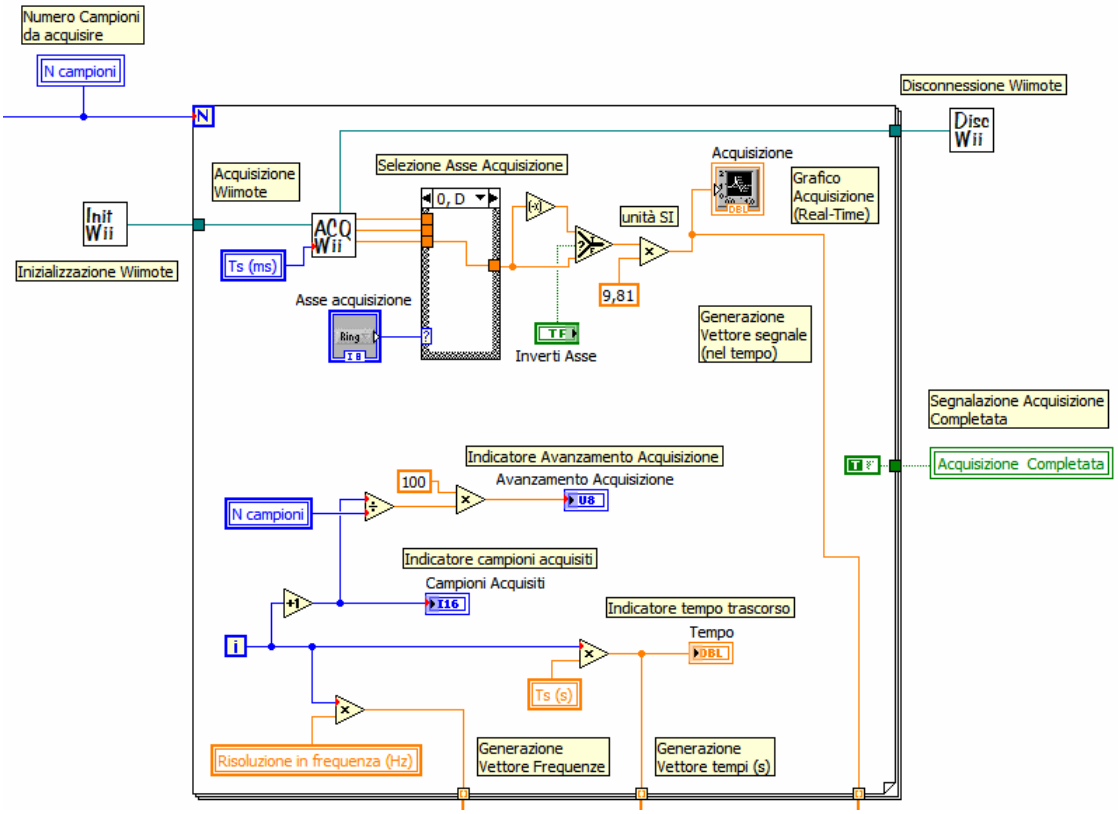


Figura 5.56: Acquisizione dei dati e generazione dei vettori per le scale dei tempi e delle frequenze

Calcolo della *Discrete Fourier Transform*

Il calcolo della DFT avviene attraverso un Sub-VI appositamente realizzato, identificato con il nome *dftVI* e raffigurato nel *block diagram* con l'icona:  . Per il calcolo della trasformata di Fourier viene utilizzata una funzione di Labview *FFT*, disponibile nella *Signal Processing Palette:*  . Per maggiori dettagli sulla *FFT* e Labview si vedano [40] e[27]. La funzione implementa l'algoritmo per il calcolo veloce della trasformata di Fourier, *Fast Fourier Trasform* (*FFT*). Una diretta implementazione della formula 5.3.12 darebbe origine ad un algoritmo con complessità temporale $O(n^2)$, l'algoritmo FFT invece permette il calcolo della DFT in un tempo $O(n \cdot \log_2(n))$. Il vettore in uscita dalla funzione è la sequenza di Ncampioni:

$$X'_k = \sum_{i=0}^{N-1} x_i \cdot e^{-j2\pi ik/N} \quad k = 0, 1, 2, \ldots, N-1 \tag{5.3.12}$$

dove x_k è la sequenza in ingresso, ovvero il vettore contenente il segnale acquisito. La 5.3.12 differisce dalla 5.3.2 per il fattore moltiplicativo $1/N$, la sequenza di valori in uscita dalla *FFT* deve essere normalizzata affinchè lo spettro del segnale venga rappresentato con la giusta scala. Si deve eseguire l'operazione:

$$X_k = \frac{X'_k}{N} \tag{5.3.13}$$

la sequenza X_k viene troncata all'M-esimo valore, con $M = \left\lfloor \frac{N+1}{2} \right\rfloor$, vengono ovvero considerate solo le frequenze positive da 0 a $F_{ny} = F_s/2$. La sequenza troncata verrà indicata come $\widetilde{X_k}$. Per l'applicazione realizzata interessa solo il modulo di $\widetilde{X_k}$, ovvero $|\widetilde{X_k}|$.
In figura 5.57 è riportato il codice del Sub-VI *dftVI*. Le operazioni eseguite dal codice sono quelle appena descritte, in uscita vengono forniti due vettori, uno contenente $|\widetilde{X_k}|$ ed uno contenente la stessa sequenza ma con il valore della componente continua azzerato, ovvero $|\widetilde{X_0}| = 0$. Quest'ultima sequenza verrà indicata con $|\widetilde{X_k}|'$ Quest'ultimo vettore verrà utilizzato nella successiva fase dell'elaborazione, nel caso in sia abilitata l'opzione *"Elimina componente DC per il calcolo della frequenza"* nei parametri per l'esperimento.

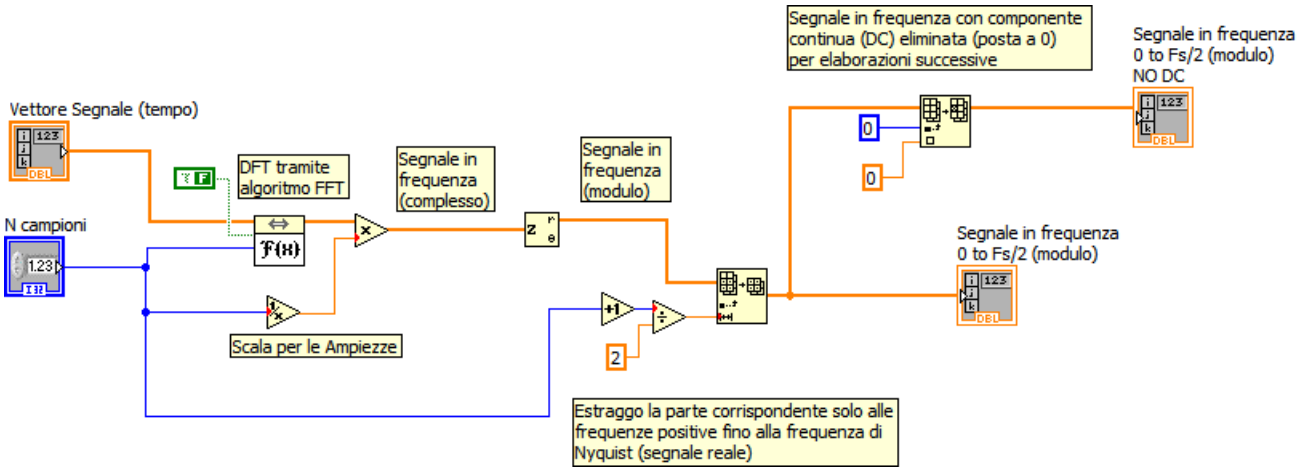


Figura 5.57: Codice SubVI *dftVI*

Calcolo dei risultati

Il frammento di codice che si occupa del calcolo dei risultati e della loro visualizzazione è riportato

in figura 5.58. Il vettore su cui operare per il calcolo della frequenza viene determinato dal parametro *"Elimina componente DC per il calcolo della frequenza":* attraverso la funzione *select* viene selezionata la sequenza $|\widetilde{X}_k|'$, se l'opzione è attivata, oppure $|\widetilde{X}_k|$ in caso contrario. Le operazioni eseguite sono:

1. Individuazione del picco in frequenza $|X_{k_{max}}|$, ovvero del valore massimo nel vettore, e dell'indice K_{max} in cui si trova tale massimo. L'ampiezza stimata della componente vale $\hat{A} = 2 \cdot |X_{k_{max}}|$.

2. Individuazione della frequenza alla quale si trova $|X_{k_{max}}|$ attraverso il vettore delle frequenze. La frequenza stimata risulta essere il valore contenuto all'indice k_{max} di tale vettore, ovvero $f_0 = k_{max} \cdot F$.

3. Conversione unità di misura e visualizzazione su indicatori numerici.

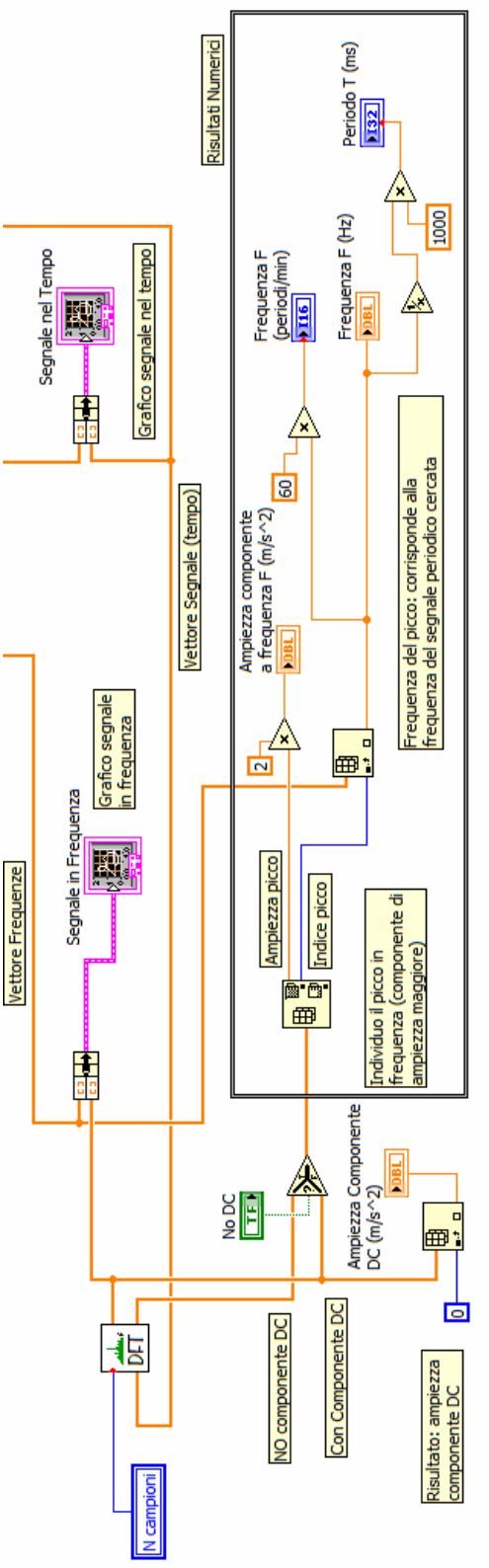


Figura 5.58: Calcolo e visualizzazione dei risultati

5.3.4 Analisi dei risultati

Sono state eseguite delle prove con il programma realizzato, sfruttando il movimento periodico del tergicristallo di un'automobile. Il Wiimote è stato accuratamente fissato al tergicristallo come mostrato in figura 5.43, in questo caso l'asse del controller che risente maggiormente dell'accelerazione risulta essere l'asse X. Il tergicristallo può funzionare in tre modalità con "velocità" crescenti, indicate in seguito con i termini *"Intermittente"*, *"Continuo Lento"* e *"Continuo Veloce"*, obiettivo sarà misurare la frequenza di lavoro del dispositivo.
Il parametri dell'esperimento usati per tutte e tre le "velocità" sono:

- *Asse acquisizione*: ASSE X;
- *Inverti orientazione asse*: NO;
- *Periodo di campionamento Ts (s)*: $10ms$;
- *Finestra acquisizione Tw (s)*: $100s$;
- *Elimina componente DC per il calcolo della frequenza*: SI;

Con questi parametri l'acquisizione ha una durata abbastanza lunga, pari ad 1 minuto e 40 secondi, tuttavia risulta, come visto, una risoluzione in frequenza elevata, pari a $0.01Hz$. La frequenza massima misurabile teorica risulta essere $50Hz$.

Tergicristallo "Intermittente"
Il tergicristallo esegue un ciclo dopodichè si ferma per un intervallo di tempo prima di eseguire un secondo ciclo. Il segnale acquisito, nel dominio del tempo, è riportato in figura 5.59. Come si può osservare il segnale è composto da triangoli equispaziati, ad ogni triangolo corrisponde un ciclo del tergicristallo. In figura 5.60 è riportato lo spettro del segnale.

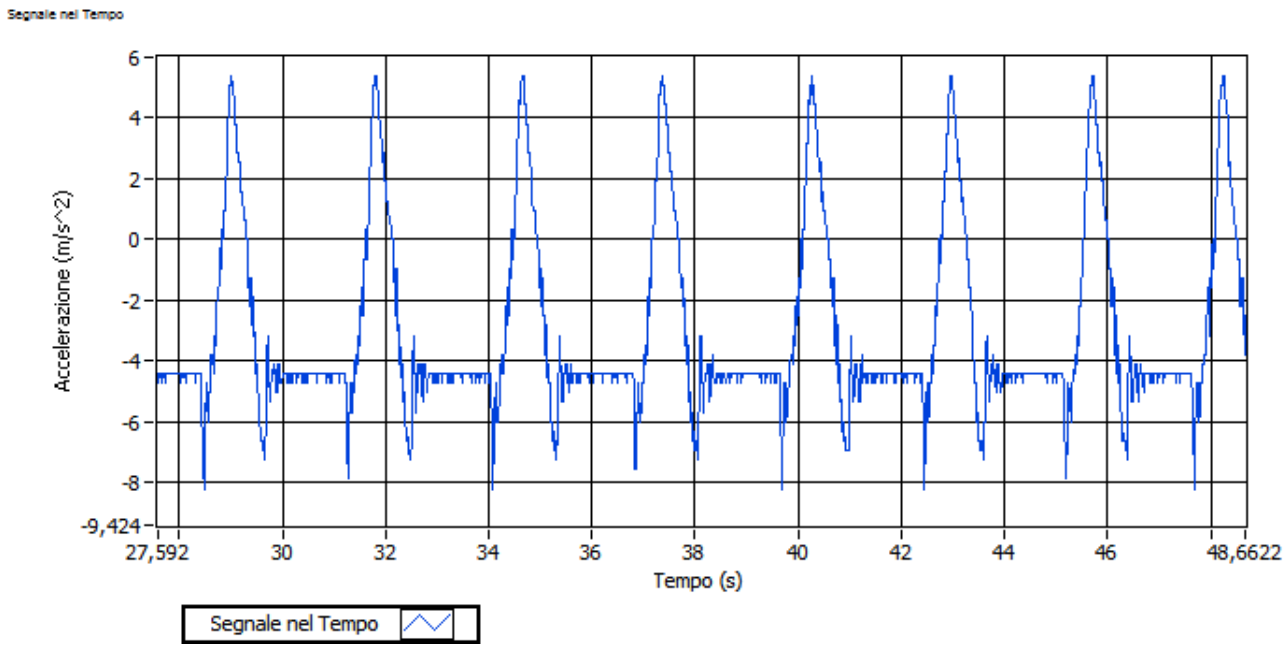


Figura 5.59: Funzionamento *"Intermittente"* - Segnale nel tempo

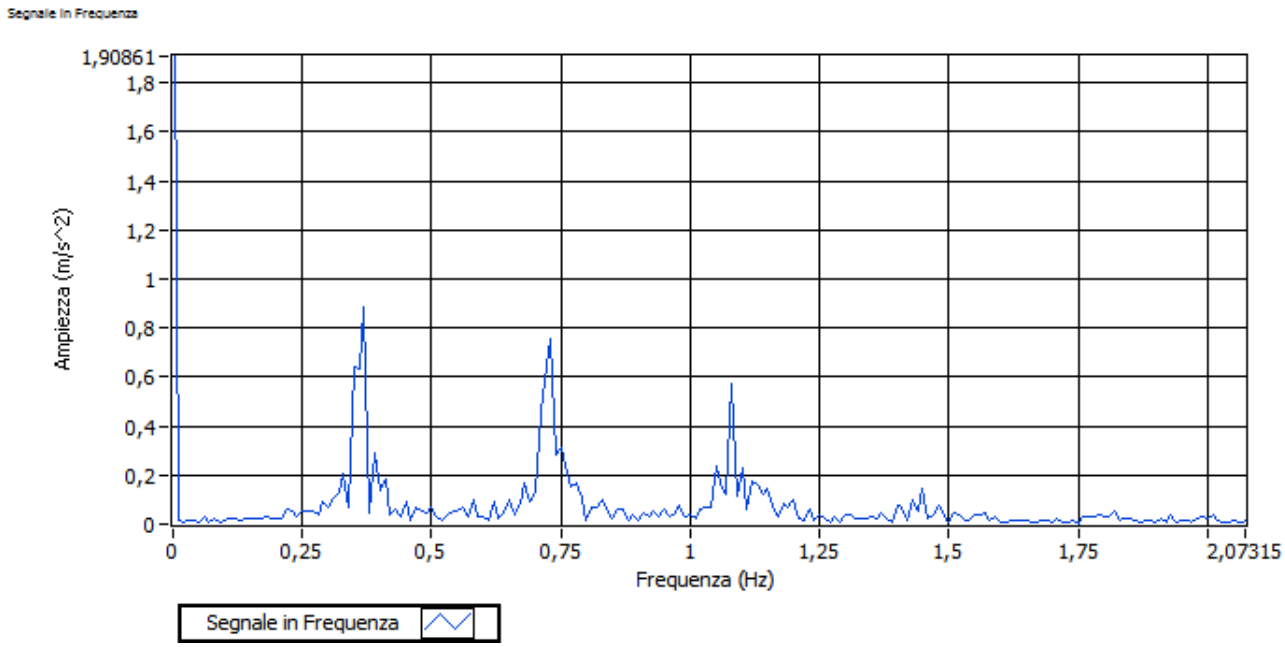


Figura 5.60: Funzionamento *"Intermittente"* - Segnale in frequenza

È interessante notare che vi sono evidenti componenti armoniche ai multipli della frequenza $\hat{f}_o$ della fondamentale, lo spettro ricorda molto quello di un segnale a *dente di sega*. I risultati numerici forniti dall'applicazione sono:

- Frequenza (Hz): $0.37Hz$;
- Frequenza (periodi/minuto): 22;
- Periodo (ms): $2703ms$;
- Ampiezza componente fondamentale (m/s^2): $1.76m/s^2$;
- Ampiezza componente DC (m/s^2): $3.13m/s^2$;

Il tergicristallo esegue un ciclo approsimativamente ogni $2.7s$, in un minuto vengono eseguiti circa 22 cicli.

Tergicristallo "Continuo Lento"
Il tergicristallo funziona in modo continuo senza pausa tra un ciclo e l'altro. Il segnale acquisito, nel dominio del tempo, è riportato in figura 5.61. Come si può osservare il segnale ottenuto è triangolare. In figura 5.62 è riportato lo spettro del segnale.

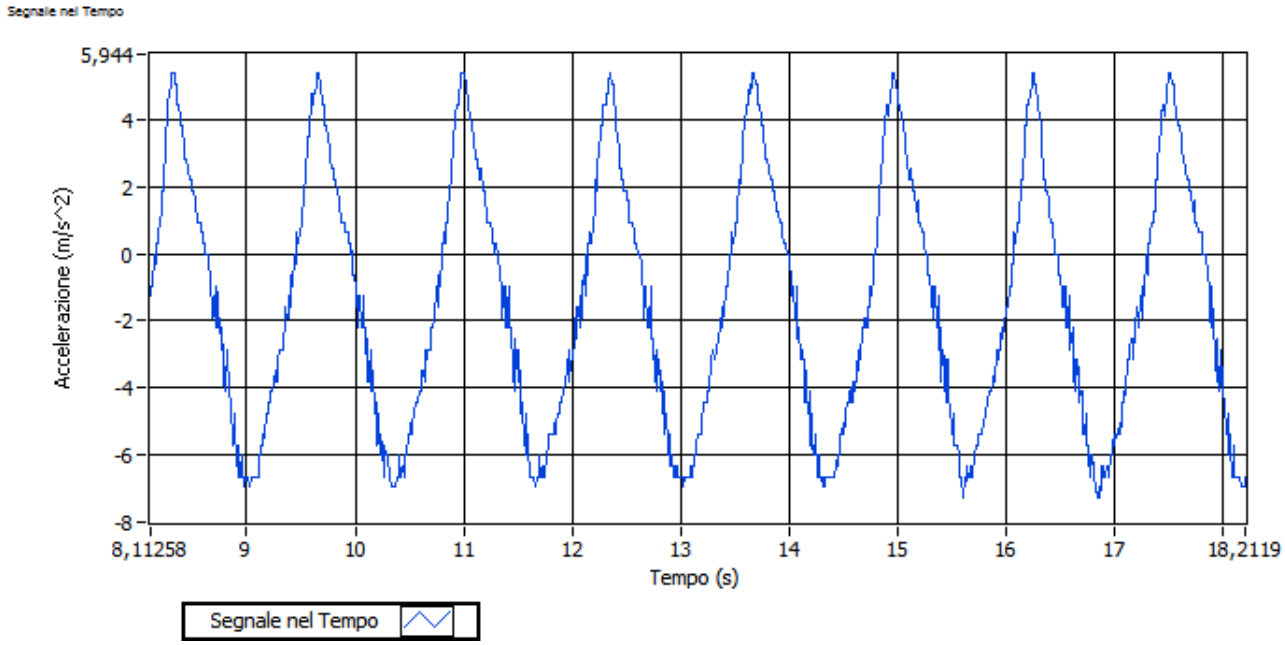


Figura 5.61: Funzionamento *"Continuo Lento"* - Segnale nel tempo

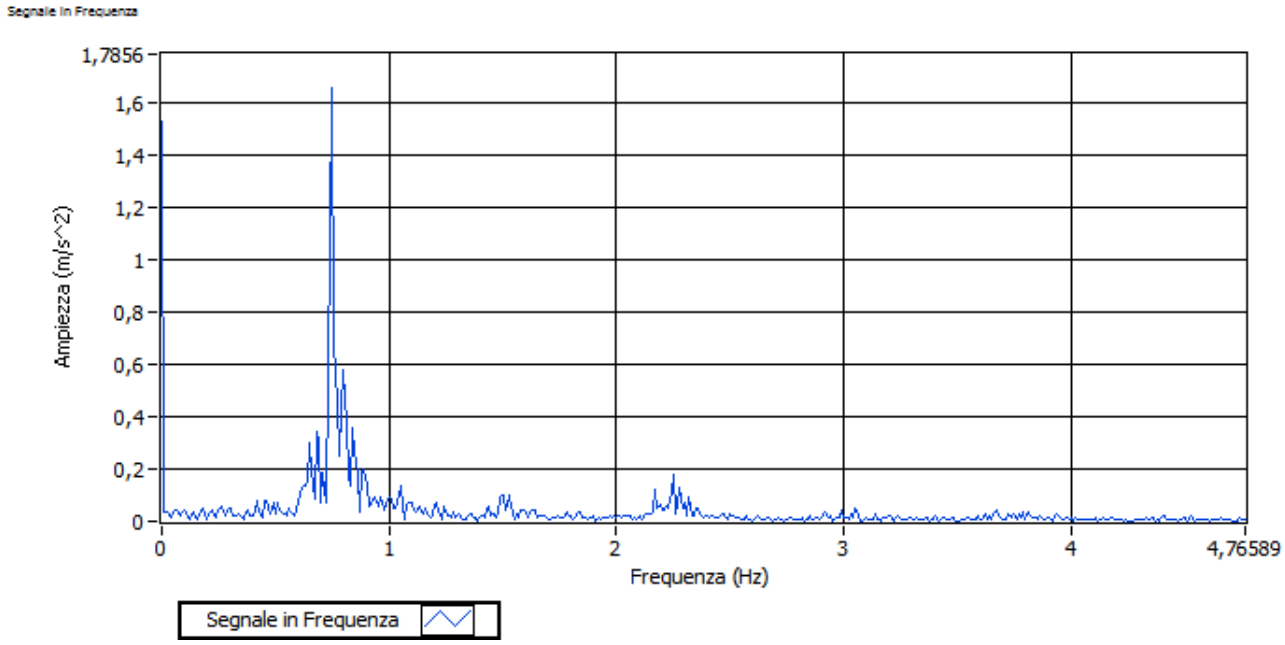


Figura 5.62: Funzionamento *"Continuo Lento"* - Segnale in frequenza

È chiaramente visibile la componente fondamentale a frequenza $\hat{f}_o$, rappresentante la frequenza di lavoro del tergicristallo. Essendo il segnale triangolare sono presenti armoniche ai multipli dispari della frequenza $\hat{f}_o$, le armoniche pari invece sono nulle. Il grafico X rispecchia quanto detto. I risultati numerici forniti dall'applicazione sono:

- Frequenza (Hz): $0.75Hz$;
- Frequenza (periodi/minuto): 45;
- Periodo (ms): $1333ms$;

- Ampiezza componente fondamentale (m/s^2): $3.31 m/s^2$;
- Ampiezza componente DC (m/s^2): $1.53 m/s^2$;

Il tergicristallo esegue un ciclo approsimativamente ogni $1.3 s$, in un minuto vengono eseguiti circa 45 cicli.

Tergicristallo "Continuo Veloce"
Per completezza vengono riportati i grafici anche per il modo di funzionamento *"Continuo Veloce"*. Il funzionamento è analogo al modo *"Continuo Lento"*, ma il tergicristallo si muove con una velocità maggiore. Il segnale ottenuto è triangolare, l'andamento nel tempo è riportato in figura 5.63, l'andamento in frequenza è riportato in figura 5.64 (le componenti armoniche dispari sono meglio visibili in questo grafico rispetto a quello di figura 5.62).

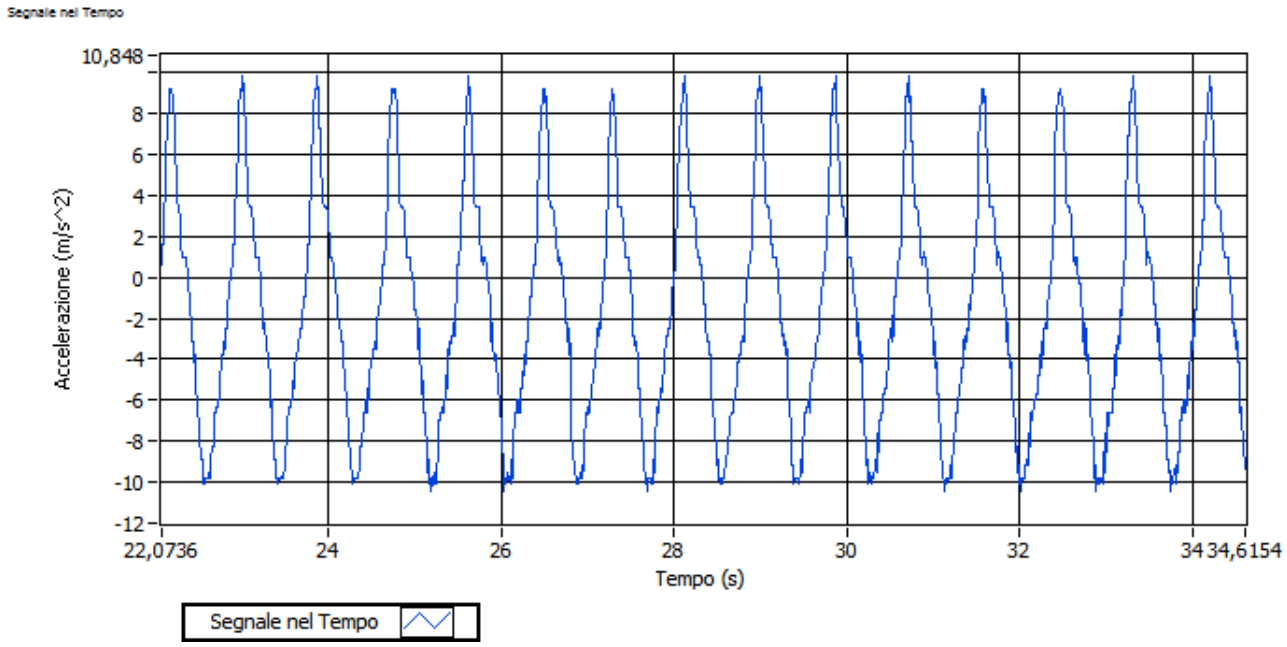


Figura 5.63: Funzionamento *"Continuo Veloce"* - Segnale nel tempo

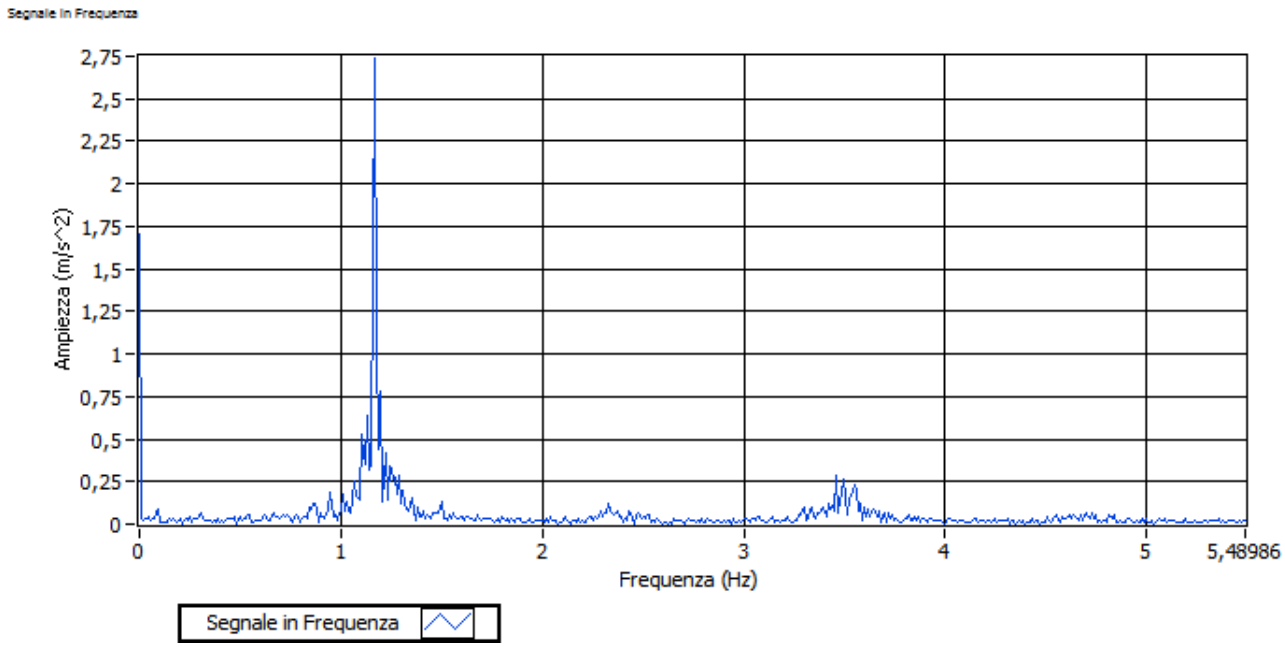


Figura 5.64: Funzionamento *"Continuo Veloce"* - Segnale in frequenza

I risultati numerici forniti dall'applicazione sono:

- Frequenza (Hz): $1.17Hz$;
- Frequenza (periodi/minuto): 70;
- Periodo (ms): $855ms$;
- Ampiezza componente fondamentale (m/s^2): $5.49m/s^2$;
- Ampiezza componente DC (m/s^2): $1.71m/s^2$;

Il tergicristallo esegue un ciclo approsimativamente ogni $0.9s$, in un minuto vengono eseguiti circa 70 cicli.

CONCLUSIONI

I sistemi di misura fanno parte della vita di tutti i giorni, spesso teniamo in mano un comune oggetto elettronico e non ci rendiamo conto di quello che c'è dentro, non immaginiamo neanche il lavoro ingegneristico che c'è dietro, non ci soffermiamo a pensare a quanto studio e quanta ricerca è stata fatta per garantire il suo funzionamento.

In questo lavoro di tesi è stata presentata una possibile applicazione di carattere scientifico di un controller commerciale ampiamente utilizzato nel mondo dei videogiochi. La scelta di un tale dispositivo è motivata dalla sua versatilità, dal ridotto costo e soprattutto dalla presenza al suo interno di numerosi sensori già inseriti in sistemi di misura completamente funzionanti e facilmente interrogabili da remoto attraverso un normale computer dotato di interfaccia Bluetooth. La prima parte di questo lavoro di tesi è stata infatti rivolta allo studio delle funzionalità dell'hardware contenuto nel controller e della modalità di comunicazione tra tale dispositivo e un altro dispositivo di interrogazione (ad esempio un computer) attraverso le librerie disponibili. La seconda parte invece ha visto lo sviluppo di alcuni semplici applicativi a scopo di esempio basati su una post-elaborazione al calcolatore dei dati acquisiti dal controller Wiimote.
Gli applicativi realizzati risultano essere un valido strumento per uno studente che si trova ad affrontare per la prima volta i concetti fisici e matematici discussi nel capitolo 5. Egli infatti può, ad esempio, fare uso dei programmi per giustificare le nozioni teoriche apprese a lezione, dando un senso alle formule studiate sui libri in maniera autonoma, grazie agli esperimenti guidati ed al Wiimote. Le interfacce grafiche create permettono una rapida esecuzione dell'esperimento, lo studente viene guidato passo passo nei singoli *STEP* che lo compongono. Il Wiimote è un dispositivo economico e di facile reperibilità, questo rende il pacchetto di applicazioni sviluppato alla portata di tutti.

Bibliografia

[1] L. Benetazzo, C. Narduzzi, G. Giorgi, *Dispense di misure per l'automazione e la produzione industriale*, Padova, 2008

[2] Wikimedia Foundation, Wikipedia: l'enciclopedia libera, *Wii*,
URL (luglio 2010): http://it.wikipedia.org/wiki/Wii

[3] Wikimedia Foundation, Wikipedia: l'enciclopedia libera, *Human Interface Device Class*,
URL (luglio 2010): http://en.wikipedia.org/wiki/Human_interface_device

[4] USB Implementers Forum, *Device Class Definition for Human Interface Devices (HID)*, Version 1.11,
URL (luglio 2010): http://www.usb.org/developers/devclass_docs/HID1_11.pdf

[5] Wikimedia Foundation, Wikipedia: l'enciclopedia libera, *Bluetooth Profile*,
URL (luglio 2010): http://en.wikipedia.org/wiki/Bluetooth_profile

[6] Wikimedia Foundation, Wikipedia: l'enciclopedia libera, *Accelerometro*,
URL (luglio 2010): http://it.wikipedia.org/wiki/Accelerometro

[7] Wikimedia Foundation, Wikipedia: l'enciclopedia libera, *Micro Electro Mechanical Systems (MEMS)*,
URL (luglio 2010): http://en.wikipedia.org/wiki/Microelectromechanical_systems

[8] WiiBrew Wiki, *Wiimote*
URL (luglio 2010): http://wiibrew.org/wiki/Wiimote/

[9] WiiBrew Wiki, *Library*
URL (luglio 2010): http://wiibrew.org/wiki/Wiimote/Library

[10] Nathan Seidle, SparkFun electronics, *Wii-mote Guts Tutorial*,
URL (luglio 2010): http://www.sparkfun.com/commerce/tutorial_info.php?tutorials_id=43

[11] Broadcom Corporation, *BCM2042 Product Brief*, 2005,
URL (luglio 2010): http://www.datasheetcatalog.com/

[12] Oliver Kreylos, Oliver kreylos' research and development homepage, *IR Camera 3D tracking*,
URL (luglio 2010): http://idav.ucdavis.edu/∼okreylos/ResDev/Wiimote/

[13] MEMS Universe, *MEMS Accelerometers*
URL (luglio 2010): http://www.memsuniverse.com/1548

[14] Michael Kraft, *Closed loop digital accelerometer employing oversampling conversion*, Ph.D Thesis, Coventry University, School of Engineering, UK, 1997, capitoli 1, 2, 3 e 5,
URL (luglio 2010): http://users.ecs.soton.ac.uk/mk1/

[15] Analog Devices, *iMEMS ADXL330 Datasheet*, revision A, 2007,
URL (luglio 2010): http://www.analog.com/static/imported-files/data_sheets/ADXL330.pdf

[16] Analog Devices, Techincal Documentation, *Accelerometer Specifications Definitions*,
URL (luglio 2010):
http://www.analog.com/en/sensors/inertial-sensors/adxl330/products/technical-documentation/td_Accelerometer_Specifications_Definitions/resources/fca.html

[17] Lisa Zyga, Physorg, *Hacking the wii remote for physics class*,
URL (luglio 2010): http://www.physorg.com/news104502773.html

[18] Università di Firenze, *OpenLab - laboratori aperti*,
URL (luglio 2010): http://www.openlab.unifi.it/

[19] Johnny Chung Lee, Johnny Chung Lee Homepage, *Wii Remote Projects*,
URL (luglio 2010): http://johnnylee.net/projects/wii/

[20] Cynergy Labs, *Project maestro*,
URL (luglio 2010): http://labs.cynergysystems.com/Silverlight.html

[21] Carl Kenner, *GlovePIE Home Page*,
URL (luglio 2010): http://glovepie.org/

[22] Deirdre Walsh, National Instrument Comunity, *Labview interface to a Wii Remote (through WiimoteLib)*, dicembre 2009,
URL (luglio 2010): http://decibel.ni.com/content/docs/DOC-1353

[23] Brian Peek, BrianPeek.com, *WiimoteLib - .NET Managed Library for the Nintendo Wii Remote*,
URL (luglio 2010): http://www.brianpeek.com/blog/pages/wiimotelib.aspx

[24] Brian Peek, MSDN Blogs - Coding4Fun, *Managed Library for Nintendo's Wiimote*, 2007,
URL (luglio 2010): http://blogs.msdn.com/b/coding4fun/archive/2007/03/14/1879033.aspx

[25] Michael Laforest, *Wiiuse - a library written in C that connects with several Nintendo Wii remotes*,
URL (luglio 2010): http://www.wiiuse.net/

[26] National Instrument, *Labview Fundamentals*, agosto 2005,
URL (luglio 2010): http://www.ni.com/pdf/manuals/374029a.pdf

[27] National Instrument, *Labview 2009 Help*, giugno 2009,
URL (luglio 2010): http://zone.ni.com/reference/en-XX/help/371361F-01/

[28] The MathWorks, *Matlab Documentation*,
URL (luglio 2010): http://www.mathworks.com/access/helpdesk/help/techdoc/

[29] Joint Committee for Guides in Metrology (JCGM), *Guides to the expression of uncertainty in measurement (GUM)*, settembre 2008,
URL (luglio 2010): http://www.bipm.org/utils/common/documents/jcgm/JCGM_100_2008_E.pdf

[30] Thomas Schlömer, Benjamin Poppinga, Niel Henze, Susanne Boll, *Gesture Recognition with a Wii Controller*, University of Oldenburg and OFFIS Institute for Information Technology, Bonn, Germany, 2008,

URL (luglio 2010):
http://www.benjaminpoppinga.de/downloads/gesture_recognition_with_a_wii_controller-schloemer_poppinga_henze_boll.pdf

[31] Jordan Brindza, Jessica Szweda, Qi Liao, Yingxin Jiang, Aaron Striegel, *WiiLab: Bringing Togheter the Nintendo Wiimote and MATLAB*, Department of Computer Science and Engineering, University of Notre Dame (USA), 39^{th} ASEE/IEEE Frontiers in Education Conference, URL (luglio 2010): http://fie-conference.org/fie2009/papers/1347.pdf

[32] University of Notre Dame, *WiiLAB Wiki Page and Class Documentation*, URL (luglio 2010): http://netscale.cse.nd.edu/twiki/bin/view/Edu/WiiMote

[33] P. Mazzoldi, M. Nigro, C. Voci, *Elementi di fisica meccanica-termodinamica*, EdiSES, Napoli, 2006

[34] Wikimedia Foundation, Wikipedia: l'enciclopedia libera, *Euler angles*, URL (luglio 2010): http://en.wikipedia.org/wiki/Euler_angles

[35] Freescale Semiconductor, *Application Note AN3461 - Tilt Sensing Using Linear Accelerometers*, revision 2, giugno 2007, URL (luglio 2010): http://cache.freescale.com/files/sensors/doc/app_note/AN3461.pdf

[36] Wikimedia Foundation, Wikipedia: l'enciclopedia libera, *Rollio*, URL (luglio 2010): http://it.wikipedia.org/wiki/Rollio

[37] Wikimedia Foundation, Wikipedia: l'enciclopedia libera, *Beccheggio*, URL (luglio 2010): http://it.wikipedia.org/wiki/Beccheggio

[38] Wikimedia Foundation, Wikipedia: l'enciclopedia libera, *Imbardata*, URL (luglio 2010): http://it.wikipedia.org/wiki/Imbardata

[39] G. Cariolaro, G. Pierobon, G. Calvagno, *Segnali e sistemi*, McGraw- Hill, Milano, 2004

[40] National Instrument, *Application Note 040 - Fast Fourier Transforms and Power Spectra in LabVIEW*, febbraio 1993, pubblicato nella NI Developer Zone nel settembre 2006, URL (luglio 2010): http://zone.ni.com/devzone/cda/tut/p/id/4541

[41] National Instrument, *Application Note 041 - The Fundamentals of FFT-Based Signal Analysis and Measurement*, luglio 2000, pubblicato nella NI Developer Zone nel giugno 2009, URL (luglio 2010): http://zone.ni.com/devzone/cda/tut/p/id/4278