Politecnico di Torino

Corso di Laurea Magistrale in Ingegneria Meccanica

A.a. 2020/2021

Sessione di Laurea Ottobre 2021

Studio prototipale e simulazione multibody di un dispositivo indossabile IoT per il monitoraggio in ambiente di lavoro

RELATORE:

Prof. Aurelio Somà

CANDIDATO:

Simona Civallero

CO-RELATORI:

Prof. Gianpiero Mastinu
Ing. Caterina Russo

A Nonno Beppe,
che mi ha insegnato a sorridere alla vita,
sempre.

Abstract

Negli ultimi decenni, al fine di garantire l'incolumità dei lavoratori e del personale, il tema della sicurezza sul lavoro ha assunto un'importanza rilevante. Anche l'introduzione di nuove regolamentazioni a tal riguardo ha fatto sì che i datori di lavoro adottassero nuove misure e azioni interne ed esterne all'azienda volte a preservare la salute dei dipendenti. La progettazione di dispositivi innovativi fondati sulle nuove tecnologie dell'Industria 4.0 pone le basi a un monitoraggio continuo dei lavoratori, che garantisca un pronto intervento quando necessario ed eviti l'insorgere di situazioni pericolose.

L'obiettivo dello studio è quello di progettare un dispositivo indossabile sensorizzato, che permetta di monitorare costantemente i lavoratori e in questo modo garantire la loro sicurezza sul luogo di lavoro. Si tratta di un casco di protezione in grado di monitorare le condizioni ambientali, lo stato di salute e di attività dell'operatore e il corretto utilizzo del dispositivo di protezione. Parallelamente, per permettere la simulazione delle principali attività svolte sui luoghi di lavoro, il proposito è di realizzare un modello multibody con androide che permetta di predire i parametri monitorati dal casco sensorizzato.

Per la realizzazione del dispositivo indossabile IoT, si utilizzano due schede Arduino con sensori integrati e dotate di Bluetooth e/o Wi-Fi per la trasmissione dei dati. Mediante la loro opportuna programmazione, è possibile trasferire i dati ad un calcolatore centrale e permettere l'analisi degli stessi. Inoltre, si propone l'integrazione del dispositivo con un sistema di avviso di prossimità, che permetta di tenere sotto controllo le distanze relative uomo – macchina. Per quanto concerne l'attività multibody, invece, questa è condotta sul software Adams/View facendo uso di un modello androide già esistente ed apportando le opportune modifiche. Il modello è poi validato sperimentalmente per permettere la correlazione tra lo stesso e i parametri registrati dal dispositivo IoT.

Il lavoro di tesi permette in questo modo di avere un modello multibody in grado di simulare adeguatamente alcune attività caratteristiche del luogo di lavoro, e dunque predire i parametri relativi allo stato di attività registrati dal casco sensorizzato progettato. A tal proposito, si presenta un indice di mobilità che può essere indicativo dello stato di stanchezza degli operatori a fine giornata, e che si possa calcolare sia dai dati monitorati dal dispositivo IoT che dalla simulazione multibody. In futuro sarebbe opportuno integrare il dispositivo realizzato con la possibilità di monitorare la frequenza cardiaca, in maniera tale da poter calcolare direttamente un indice di affaticamento dei dipendenti a fine giornata.

Indice

Introduzione

In questi ultimi anni il tema della sicurezza, in particolare la sicurezza sul lavoro, ha assunto un'importanza rilevante. Nella maggior parte dei Paesi del mondo, infatti, sono state introdotte regolamentazioni per la prevenzione dei rischi lavorativi e la salute sul lavoro. In Italia, l'Inail è l'ente che si occupa della prevenzione e sicurezza sul lavoro, fornendo corsi formativi, consulenze e assistenza alle aziende a tal riguardo. Queste ultime hanno dunque dovuto adottare sistemi di gestione della salute e sicurezza sul lavoro volti ad individuare i possibili rischi presenti sul luogo di lavoro e le modalità con le quali prevenirli. L'attenzione delle diverse aziende verso dispositivi per il settore safety è quindi aumentata esponenzialmente.

Nell'era dell'industria 4.0, sono stati condotti numerosi studi su come applicare le tecnologie digitali emergenti all'ambito della sicurezza e prevenzione degli incidenti sul lavoro. In particolare, sono stati realizzati alcuni prototipi di dispositivi di sicurezza indossabili sensorizzati che permettano di monitorare costantemente l'operatore, individuando situazioni che potrebbero diventare dannose, permettendo un intervento immediato in caso di condizioni critiche e facilitando l'individuazione dei cambiamenti da effettuare per migliorare la qualità degli ambienti di lavoro. Si è così fatto strada, insieme a quello di Industria 4.0, il concetto di Healthy Operator 4.0: l'utilizzo di dispositivi indossabili di tracciamento delle condizioni di stato degli operatori diventa fondamentale per avere una visione unificata delle condizioni di salute del lavoratore, per preservare e proteggere il suo benessere e il suo stato di salute e facilitare i suoi compiti. A tal proposito, sono stati proposti diversi sistemi di monitoraggio delle condizioni ambientali e di salute dell'operatore.

Il presente lavoro di tesi ha come obiettivo la modellazione e prototipazione di un dispositivo indossabile sensorizzato di protezione personale sul luogo del lavoro. Monitorando le condizioni di salute dell'operatore e ciò che avviene nell'ambiente circostante, è possibile ridurre il rischio di incidenti sul lavoro e permettere un tempestivo intervento in caso di incidente avvenuto. Con un dispositivo

intelligente di questo tipo, risulta infatti possibile monitorare le condizioni ambientali, le condizioni di salute dell'operatore, ed eventualmente analizzare l'interazione uomo – macchina. Partendo dall'analisi dei numerosi prototipi di dispositivi sensorizzati presenti in letteratura, si è dunque progettato un casco intelligente in grado di monitorare lo stato dell'operatore e rilevare eventuali situazioni di pericolo.

Il dispositivo sensorizzato è anche utile per la valutazione dello stato di attività dell'operatore sul luogo di lavoro. Al fine di simulare alcune delle comuni attività lavorative, si procederà con la realizzazione di un modello multibody con androide. L'analisi multibody permette infatti di stimare preventivamente i valori di accelerazione e velocità angolare che verranno letti dai sensori IMU (Inertial Measurement Unit) presenti sul dispositivo. È in questo modo possibile relazionare attività sperimentali di utilizzo del casco intelligente con l'attività multibody, che permette di ottenere risultati più rapidamente e con costi inferiori.

Il monitoraggio dei parametri ambientali e dello stato di salute può anche essere utile per analizzare come essi influenzino la stanchezza del lavoratore alla fine della giornata. Si è per questo motivo pensato di introdurre, per la prima volta, il concetto di indice di mobilità ed affaticamento. Il primo si basa sulla sola acquisizione dei segnali indicanti lo stato di attività dell'operatore (accelerazioni e velocità angolari). L'indice di affaticamento si ottiene invece integrando quello di mobilità con la variabilità di alcuni dei parametri monitorati, quali temperatura corporea e frequenza cardiaca.

Nel seguito si tratterà quanto appena esposto. In particolare, il Capitolo 1 è dedicato alla descrizione del dispositivo di sicurezza sensorizzato progettato, ossia dell'apparecchiatura necessaria, della raccolta e trasferimento dati, dell'analisi dati, della possibile integrazione di un sistema di avviso di prossimità e della domanda energetica del sistema. Nel Capitolo 2 si tratterà invece l'analisi multibody, con la descrizione dettagliata del modello e dei movimenti analizzati e con l'illustrazione della procedura di validazione del modello utilizzata. Infine, il Capitolo 3 ha lo scopo di definire un indice di mobilità e affaticamento che permetta di stimare il livello di stanchezza del lavoratore a fine giornata, sulla base dei parametri rilevati dal casco sensorizzato.

Capitolo 1

Realizzazione del dispositivo

Il dispositivo che si vuole realizzare, come anticipato, serve per ridurre i rischi sul lavoro ed intervenire tempestivamente in caso di incidente avvenuto. Le fonti di rischio in ambiente lavorativo sono numerose, tra queste:

+ Movimenti indesiderati dell'operatore (cadute, impatti, ecc...);
+ Eccessiva vicinanza del lavoratore a macchinari pericolosi;
+ Scarsa qualità dell'aria circostante;
+ Problemi di salute dell'operatore (stanchezza, febbre, ecc...);
+ Temperatura e/o umidità ambientali incontrollate;
+ Mancanza di adeguata luminosità dell'ambiente;
+ Incendi;
+ Inutilizzo dei dispositivi di sicurezza.

Per prevenire tali situazioni di rischio per l'operatore, è possibile utilizzare dei dispositivi sensorizzati che monitorino alcuni parametri significativi. Tra questi la rilevazione della temperatura, umidità e luminosità dell'ambiente e delle sostanze presenti nell'aria permettono di valutare il comfort all'interno dell'ambiente lavorativo. Monitorando invece la temperatura corporea, la frequenza cardiaca e il tasso alcolemico dell'operatore è possibile prevenire situazioni rischiose e aggravi delle condizioni di salute del lavoratore. Il continuo rilevamento dei dati di accelerazione legati al movimento dell'individuo permette inoltre di valutare la presenza di impatti e/o cadute dello stesso. Per prevenire la nascita e propagazione di incendi sul luogo di lavoro, è utile misurare la presenza di fumo o gas infiammabili nell'aria. L'aggiunta di opportuni sensori, quali infrarossi, può poi essere utile per valutare il corretto utilizzo dei dispositivi di sicurezza, e in caso contrario mandare un'allerta all'operatore o al suo superiore. Infine, la localizzazione del lavoratore all'interno dello spazio di lavoro può essere utile per allertare l'operatore o fermare il macchinario in caso di distanze relative uomo – macchina inferiori a quella di sicurezza. [1 – 19]

La realizzazione di un sistema di questo tipo, che monitori diversi parametri e li mandi in tempo reale ad una stazione di controllo, porta con sé diverse problematiche. Tra queste, trattandosi di un dispositivo indossabile, esso deve essere leggero, compatto e adattabile, in modo che non crei una sensazione di fastidio sull'operatore, che deve utilizzarlo per intere giornate lavorative. A tal proposito, è quindi anche necessario che il consumo energetico sia ridotto, in modo che il dispositivo sia in grado di lavorare in autonomia almeno per una intera giornata di lavoro. Inoltre, l'apparecchio deve poter lavorare in diversi climi e ambienti (anche in presenza di pioggia nel caso di lavoratori su cantieri all'esterno) senza usurarsi. Bisogna poi tenere presente il fatto che la trasmissione e il processamento delle informazioni e degli eventi deve avvenire in tempi ristretti, per permettere il monitoraggio delle condizioni dell'operatore in tempo reale da una stazione di controllo. Tutte queste problematiche, infine, sono accompagnate dalla necessità che il costo del dispositivo sia contenuto, poiché si vuole realizzare un dispositivo il più accessibile possibile. [1 – 19]

Per progettare un dispositivo che soddisfi tutte le richieste appena illustrate e allo stesso tempo permetta di ridurre le fonti di rischio sui luoghi di lavoro, è necessaria una attenta analisi e selezione delle tecnologie attualmente presenti sul mercato.

Nel seguito, dopo una analisi dello stato dell'arte dei dispositivi indossabili sensorizzati per applicazioni safety e dei prototipi precedentemente realizzati, si descriverà il casco sensorizzato modellato. Verrà dunque illustrata l'apparecchiatura necessaria alla costruzione del sistema, il metodo di comunicazione tra i diversi dispositivi e il meccanismo di trasferimento e analisi dei dati. Inoltre, si proporrà l'integrazione di un sistema di avviso di prossimità al dispositivo realizzato, in modo da permettere l'analisi delle distanze relative uomo - macchina all'interno dello spazio di lavoro. Infine, spazio verrà dedicato all'analisi della domanda energetica del sistema, utile a valutare la sua applicabilità durante un turno di lavoro giornaliero.

1.1 Stato dell'arte

Negli ultimi anni, grazie alle tecnologie digitali emergenti, sono stati condotti numerosi studi e sono stati realizzati alcuni prototipi di dispositivi di sicurezza indossabili. Si tratta di dispositivi di dimensione ridotta che possono essere integrati ad indumenti o dispositivi di protezione indossabili (quali caschi, cinghie, ecc..), dotati di una serie di sensori che permettono il continuo monitoraggio dell'operatore e la rilevazione di condizioni che potrebbero risultare dannose, fornendo un allarme preventivo al lavoratore stesso e permettendo un intervento immediato da parte dei soccorsi in caso di condizioni critiche.

I primi studi condotti a tal riguardo concernono applicazioni safety in ambiente automobilistico e del ciclismo: dispositivi per le automobili, caschi per la moto e caschi per la bicicletta. Si tratta di sistemi volti principalmente alla rilevazione di impatti e incidenti mediante accelerometro e/o sensori di vibrazione per permettere un intervento tempestivo dei soccorsi mediante sistemi di posizionamento GPS e di comunicazione GSM. Alcuni di questi dispositivi incorporano anche ulteriori sensori, come ad esempio un Limit Switch o un sensore alternativo per rilevare il corretto allacciamento del casco ([1], [4], [15], [11], [13]), o un sensore di rilevamento dell'alcol, che monitora la quantità di alcol presente nel respiro dell'operatore ([6], [4], [2], [13], [11]).

Tra questi, nel 2012, Bhumkar S. P. et al. [6] hanno presentato un dispositivo da collocare all'interno dell'auto in grado di investigare eccesso di alcol in corpo mediante un sensore MQ-3 altamente sensibile all'alcol ed eccessiva stanchezza del guidatore mediante un sensore che rileva i battiti di palpebre (se essi diminuiscono significa che il guidatore si sta addormentando, per cui viene attivato un allarme sonoro). Inoltre, il dispositivo integra un sensore di impatto (per rilevare la presenza di urti e nel caso notificarli mediante un sistema GSM), di qualità dell'aria e di temperatura.

Un altro esempio è quello del casco da moto progettato da Shravya K. et al. nel 2019 [13]. Si tratta di un dispositivo dotato di sensore di forza per controllare che il casco sia indossato (in caso contrario inibisce l'accensione della moto), di sensore MQ-3 per la quantità di alcol nel respiro, di accelerometro per registrare le cadute e di sensore di vibrazione per valutare la presenza di incidenti. Inoltre, il casco è dotato di due piccoli pannelli solari per fornire apporto energetico, insieme alla batteria, all'intero dispositivo.

Con il tempo, l'applicazione di tali dispositivi è stata ampliata anche al campo industriale, per ridurre gli incidenti sul lavoro e migliorare la qualità degli ambienti lavorativi. Nel seguito si analizzano alcuni dei dispositivi sensorizzati che sono stati progettati negli ultimi anni. Si tratta di apparecchi, principalmente caschi di protezione, volti alla salvaguardia dell'operatore in ambiente lavorativo.

Tra i primi, Fyffe D., Langenderfer C. e Johns C. [17] realizzarono, nel 2017, un prototipo di casco di protezione da utilizzare in ambiente lavorativo destinato alla determinazione di sforzi eccessivi da parte dell'operatore. Si tratta di un casco dotato di un sensore di temperatura corporea, un sensore di battito cardiaco e un accelerometro per la determinazione di eccessive forze sul dispositivo o cadute. Per i diversi parametri monitorati, gli autori hanno stabilito dei range al di fuori dei quali le condizioni risultano dannose per il lavoratore: in tal caso si allerta lo stesso mediante un segnale luminoso e la stazione di controllo generale mediante sms. La tecnologia impiegata per la trasmissione del segnale è quindi GSM. Si tratta di un dispositivo il cui controllo è affidato ad una scheda Arduino Uno.

Per monitorare l'ambiente lavorativo industriale in tempo reale, Mangala Nandhini V. et al. [7] hanno proposto, nel 2018, un sistema di sensori collocati sul casco di protezione volti alla rilevazione della qualità dell'aria e del corretto utilizzo del dispositivo. In particolare, si usa il sensore MQ-7 per monitorare il livello di gas dannosi presenti nell'aria (CO e altri gas combustibili), un sensore di temperatura ambiente e umidità e un Limit Switch che rileva la rimozione dell'elmetto. Per il trasferimento dei dati viene in tal caso utilizzato il Wi-Fi e le informazioni vengono gestite ed elaborate mediante la piattaforma ThingSpeak.

Un dispositivo molto simile è stato proposto lo stesso anno specificatamente per l'ambiente di miniera [14]. Si tratta di un casco di protezione con sensore MQ-4 (sensore di metano e gas naturale) e MQ-7 per rilevare gas presenti nell'atmosfera, e sensore DHT11 per la rilevazione di temperatura e umidità ambientale. Questo dispositivo impiega Arduino come scheda elettronica e ZigBee come modalità di trasmissione del segnale.

Sempre nel 2018, anche Altamura A. et al. [16] hanno presentato un loro prototipo di casco di protezione per l'ambiente di lavoro, con un sensore MQ-2 per la rilevazione di fumo e gas infiammabili nell'aria, un sensore TMP36 per la temperatura corporea e un PulseSensor per monitorare il battito cardiaco. Tale dispositivo si basa su una scheda Arduino Nano ed utilizza il Wi-Fi per la trasmissione dei segnali e delle informazioni.

In tutti e tre i dispositivi appena illustrati, in caso di valori rilevati oltre una soglia specificatamente definita si attiva sul casco di protezione un segnale di allarme sia luminoso che sonoro.

Con il passare degli anni, sono stati proposti dispositivi IoT indossabili con un crescente numero di sensori. Questo è stato permesso dal miglioramento delle tecnologie: i Big Data, insieme al Machine Learning, consentono di analizzare, estrapolare e correlare una enorme quantità di dati eterogenei, sia strutturati che non, e di determinare i legami tra diversi parametri. La gestione di un maggior numero di informazioni ha quindi permesso la raccolta di più parametri da parte dei diversi sensori.

Nel 2018, oltre a quelli presentati in precedenza, sono stati progettati ulteriori due apparecchi, più complessi. Il primo è un dispositivo adattabile a diversi caschi di protezione e rimovibile, da utilizzare in ambiente di lavoro [19]. Si tratta di un dispositivo dotato di videocamera, IMU, sistema GPS, sensore di luminosità, sensore barometrico, sensore Reed per il controllo del corretto utilizzo dell'elmetto e sensore di temperatura ambientale. Tale sistema impiega una trasmissione del segnale mediante ZigBee. Il secondo dispositivo, presentato da P. Borkar S. e B. Baru V. [18], è invece un casco per minatori con: un sensore MQ-6 per rilevare la quantità di metano, LPG, idrogeno, CO, alcol e fumo presenti nell'aria, un sensore LM35 per monitorare la temperatura, un sensore DHT11 per l'umidità

ambientale, un sensore LDR per ispezionare la luminosità ambientale e un sensore infrarosso per determinare il corretto utilizzo del casco di protezione. Tale sistema utilizza Raspberry Pi 3 come board di controllo e ThingSpeak come piattaforma IoT online di raccolta ed elaborazione dati. Rispetto a quelli presentati in precedenza, si tratta di sistemi con un numero di sensori decisamente maggiore.

Inoltre, nel 2020 sono stati sviluppati, entrambi in Europa, due interessanti progetti di caschi di protezione per l'ambiente lavorativo. Il primo è stato presentato ad aprile 2020 ed è un casco da lavoro basato sulla scheda Raspberry Pi con integrati un sensore di temperatura, di umidità, di rumore, di battito cardiaco e di pressione sanguigna [10]. Il dispositivo integra anche un sistema IPS (Indoor Positioning System) per la rilevazione delle posizioni relative tra operatori e macchinari e un tool di modellazione avanzata dei dati utile alla creazione di nuova conoscenza per la piattaforma di analisi delle informazioni (Machine Learning). A novembre, Campero-Jurado I. et al. [8] hanno invece realizzato un prototipo di casco di protezione da impiegare in ambiente lavorativo dotato di un sensore di temperatura, un sensore di qualità dell'aria (che rileva i gas presenti nell'atmosfera), un sensore di pressione, un sensore di luminosità, un sensore di shock e un accelerometro. Tale sistema utilizza il Wi-Fi per la trasmissione del segnale e ThingBoard come piattaforma IoT di elaborazione dati.

Quelle appena illustrate sono solo alcune delle soluzioni proposte in questi ultimi anni. In Tabella 1 si riporta una sintesi dei principali studi e prototipi che sono stati esaminati per la definizione del nuovo casco di protezione sensorizzato.

Tabella 1 – Analisi bibliografica e soluzioni presenti in letteratura

	Luogo, Anno	**Applicazione**	**Tecnologie Hardware**	**Trasmissione del segnale**	**Sensori impiegati**
[1]	India, 2017	Casco per la moto	Arduino	Bluetooth + GSM	Limit Switch, accelerometro, GPS.
[2]	India, 2015	Dispositivo per l'automobile	Atmel AT89S52	GSM	Sensore di vibrazioni, monitoraggio quantità di alcol nel fiato, GPS.
[3]	Pakistan, 2014	Dispositivo per l'automobile	Atmel AT89S52	GSM	Sensore di vibrazioni, sensore ultrasuono (controllo ostacoli), GPS.
[4]	India, 2014	Casco per la moto	PIC 16F73	Frequenze Radio	Sensore rilevamento del lobo dell'orecchio, monitoraggio quantità di alcol nel fiato.
[5]	Sud Africa, 2016	Casco per l'industria mineraria	Atmel ATZB-24-A2	ZigBee	Sensore infrarosso, accelerometro, sensore di qualità dell'aria.

[6]	India, 2012	Dispositivo per l'automobile	ARM processor	GSM	Sensore di impatto, di qualità dell'aria e temperatura. Monitoraggio quantità di alcol nel fiato e battito delle palpebre.
[7]	India, 2018	Casco per ambiente di lavoro	Arduino	Wi-Fi	Limit Switch, sensore di qualità dell'aria, temperatura, umidità.
[8]	Europa, 2020	Casco per ambiente di lavoro	ESP-WROOM-32 module	Wi-Fi	Accelerometro, sensore di shock, qualità dell'aria, pressione, luminosità e temperatura.
[9]	India, 2020	Casco per ambiente di lavoro	Raspberry	Wi-Fi	Sensore di pressione, videocamera, GPS.
[10]	Europa, 2020	Casco per ambiente di lavoro	Raspberry	Bluetooth	Sensore di temperatura, umidità, rumore, monitoraggio battito cardiaco e pressione sanguigna.
[11]	India, 2020	Casco per la bicicletta	Arduino	Frequenze Radio	Limit Switch, sensore di vibrazioni, monitoraggio quantità di alcol nel fiato, GPS.
[12]	Corea, 2018	Casco per l'industria delle costruzioni	Arduino	Bluetooth	Accelerometro.
[13]	India, 2019	Casco per la moto	Arduino	Frequenze Radio	Force Sensing Sensor, accelerometro, sensore di vibrazioni, monitoraggio quantità di alcol nel fiato, GPS.
[14]	India, 2018	Casco per l'industria mineraria	Arduino	ZigBee	Sensore di qualità dell'aria, umidità e temperatura.
[15]	India, 2018	Casco per la bicicletta	Arduino	Bluetooth	Sensore infrarosso, accelerometro.
[16]	Europa, 2018	Casco per ambiente di lavoro	Arduino	Wi-Fi	Sensore di qualità dell'aria, monitoraggio del battito cardiaco e temperatura corporea.
[17]	USA, 2016	Casco per ambiente di lavoro	Arduino	GSM	Accelerometro, monitoraggio del battito cardiaco e della temperatura corporea.

[18]	India, 2018	Casco per l'industria mineraria	Raspberry	Wi-Fi	Sensore infrarosso, sensore di qualità dell'aria, temperatura, umidità e luminosità.
[19]	Corea, 2018	Casco per ambiente di lavoro	STM32L152 (ARM Cortex-M3)	ZigBee	Sensore Reed, IMU, sensore di temperatura e luminosità. Videocamera, sensore barometrico (rileva l'altezza cui si trova l'operatore), GPS.

1.2 Apparecchiatura necessaria

Dopo un'analisi approfondita delle soluzioni precedentemente presentate sul mercato di dispositivi indossabili sensorizzati da impiegare in ambito safety, si è passati alla progettazione del nuovo dispositivo. Una volta definiti i principali parametri da monitorare in modo da prevenire determinati rischi sul lavoro, si sono selezionati i dispositivi in grado di realizzare il sistema desiderato.

Primariamente, il casco sensorizzato si vuole che monitori i seguenti parametri:

+ Accelerazione e velocità angolare del capo. È in questo modo possibile avere una stima dello stato di attività dell'operatore, nonché avere i dati necessari per il rilevamento di cadute e impatti.
+ Temperatura ambientale. Si tratta di un parametro da tenere sotto controllo per il comfort degli operatori sul luogo di lavoro, in quanto può influenzare la stanchezza e affaticamento dei lavoratori stessi.
+ Temperatura corporea. La misura della temperatura corporea dell'operatore è utile per valutare lo stato di salute dello stesso.
+ Corretto utilizzo del casco da lavoro. Controllando che tutti gli operatori indossino correttamente tale dispositivo di sicurezza indossabile è possibile prevenire incidenti e situazioni indesiderate sul luogo di lavoro.

Definite le principali grandezze da rilevare, sono stati selezionati i dispositivi elettronici idonei.

Come piattaforma hardware, si è deciso di lavorare con le schede Arduino. Si tratta di schede elettroniche dotate di un microcontrollore che permettono la prototipazione rapida. Infatti, con Arduino è possibile realizzare una vasta quantità di progetti in maniera piuttosto semplice e rapida: il microcontrollore si programma tramite un software gratuito, Arduino IDE, dotato di numerosi esempi e schemi circuitali ed utilizzando il linguaggio di programmazione "Wiring", derivato dal C e dal C++ e piuttosto intuitivo. Inoltre, sono presenti sul mercato diverse schede Arduino con sensori e moduli Wi-Fi/Bluetooth già

integrati, per cui risultano pronte all'utilizzo. In particolare, per il dispositivo in analisi, si è deciso di lavorare con due di queste schede Arduino: Arduino Nano 33 BLE Sense e Arduino Nano 33 IoT. Si tratta di schede dotate di numerosi sensori ed entrambe di dimensioni ridotte (45 x 18 mm), dunque adeguate all'applicazione in analisi.

La scheda Arduino Nano 33 BLE Sense è ampiamente utilizzata per applicazioni di AI (Artificial Intelligence) quali il Machine Learning, essendo dotata di una elevata potenza e compattezza. Le principali caratteristiche della scheda in analisi sono il supporto per il Bluetooth Low Energy[1] (BLE) e la presenza di numerosi sensori già integrati nella scheda stessa. Per quanto concerne il BLE, questa tecnologia permette il trasferimento di dati wireless con un consumo energetico ridotto rispetto alle classiche tecnologie Bluetooth. I sensori integrati nella scheda Arduino Nano 33 BLE Sense sono i seguenti:

+ Sensore inerziale a nove assi, dotato quindi di accelerometro, magnetometro e giroscopio. Questo rende la scheda elettronica particolarmente adatta per dispositivi IoT indossabili;
+ Sensore di umidità e temperatura, utile alla rilevazione di misure accurate delle condizioni ambientali;
+ Sensore barometrico;
+ Microfono, per monitorare e analizzare suoni in tempo reale;
+ Sensore di riconoscimento gesti, sensore di prossimità, sensore di riconoscimento del colore e sensore di intensità luminosa, utili ad un monitoraggio più completo delle condizioni ambientali e degli eventi circostanti.

Infine, per quanto concerne il processore, questa scheda presenta una elevata efficienza computazionale e riduce gli assorbimenti di potenza rispetto alle schede precedentemente realizzate (come Arduino Nano). Infatti, il processore nRF52840 impiegato presenta una maggiore disponibilità di risorse di memoria sia flash che SRAM, consentendo l'implementazione di algoritmi e applicazioni di elevata complessità, tipici delle applicazioni AI. Il core ARM Cortex-M4, includendo

[1] Le schede Arduino che sono provviste di questa tecnologia sono dotate di Bluetooth 4.0, il quale integra sia il Bluetooth tradizionale che il Bluetooth Low Energy (BLE). Tale tecnologia viene utilizzata per il trasferimento dei dati tra uno o più "peripheral device" e uno o più "central device". I peripheral device sono quelli che hanno diretto accesso alla lettura dei dati dai sensori, per cui poi trasferiscono tali dati ai central device. Un peripheral device è composto da uno o più servizi, che sono visualizzabili dai singoli central device, con all'interno delle caratteristiche, che sono i "contenitori" in cui vengono immagazzinati i diversi dati dei sensori. I servizi sono identificati mediante numeri unici detti UUID. Le caratteristiche mandate dal peripheral al central possono essere di tre tipi: BLERead (il central è abilitato alla sola lettura della caratteristica in esame), BLEWrite (il valore della caratteristica può essere cambiato sia dal peripheral che dal central), BLENotify (il central viene avvisato quando i dati cambiano). Quest'ultima modalità è impiegata tipicamente per dati streaming, come dati di accelerometri o di sensori. [20]

un'unità di elaborazione in virgola mobile a singola precisione, permette infine di incrementare ulteriormente l'efficienza e le performance della scheda elettronica. [20]

La scheda Arduino Nano 33 IoT è primariamente destinata all'utilizzo per progetti IoT e applicazioni di pico-network. Si tratta di una scheda molto semplice ed economica, dotata sia di tecnologia Bluetooth 4.0 che di Wi-Fi. Il principale processore della scheda Arm Cortex-MO a 32 bit SAMD21 è a bassa potenza, mentre il Microchip ECC608 crypto garantisce una comunicazione sicura ed affidabile. Come accennato, la scheda elettronica è provvista di modulo NINA-W10, che garantisce sia la connessione Bluetooth che quella Wi-Fi sulla medesima scheda. Ciò nonostante, le due tipologie di connessione non possono operare contemporaneamente. Il Wi-Fi permette di collegare alla scheda per la ricezione e invio dati qualunque dispositivo dotato di Wi-Fi, ma consente anche la connessione con servizi Cloud online, quali il Cloud IoT di Arduino o ThingSpeak. La tecnologia Bluetooth 4.0 permette invece l'utilizzo sia della tecnologia BLE a basso consumo energetico che della tecnologia Bluetooth standard. Infine, la scheda è dotata di un'unità IMU a sei assi, con giroscopio e accelerometro, permettendo il monitoraggio di accelerazioni e velocità angolari. Per le caratteristiche appena definite, l'Arduino Nano 33 IoT risulta dunque adeguato alla creazione di progetti IoT con la necessità di dimensioni ridotte. [20]

Con le due schede Arduino appena illustrate si è in grado di monitorare tutti i parametri desiderati fuorché la temperatura corporea. A tal fine, si è deciso di integrare un sensore di temperatura Grove alla scheda Arduino Nano 33 IoT. Si tratta di un modulo sensorizzato che impiega un termistore per leggere la temperatura in termini di resistenza. La resistenza del termistore aumenta quando la temperatura diminuisce e viceversa. Viene dunque restituito un valore analogico proporzionale alla temperatura, e convertibile in essa riferendosi allo schema che si trova sulla scheda tecnica del sensore, riportato in Figura 1 [21].

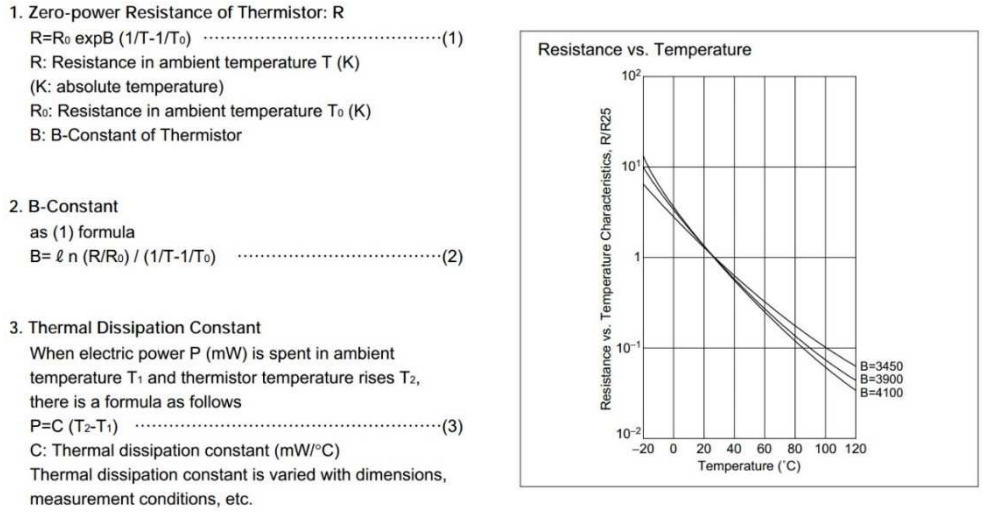


Figura 1 - Legame resistenza / temperatura per Grove Temperature Sensor

In particolare, presi $R_0 = 100$ kΩ la resistenza del sensore a temperatura ambiente di riferimento $T_0 = 25$ °C, B = 4275 la costante del termistore e R l'attuale resistenza del termistore, la temperatura è calcolabile come segue.

$$T = \frac{1}{\frac{log\left(\frac{R}{R_0}\right)}{B} + \frac{1}{298.15}} - 273.15 \quad [°C]$$

Dove R si ottiene con la seguente formula a partire dal valore a del pin di ingresso analogico:

$$R^* = \frac{1023}{a} - 1$$

$$R = R^* \cdot R_0$$

Il range di funzionamento del sensore è compreso tra -40 °C e 125°C, mentre la precisione è di 1.5 °C. L'intenzione è quella di usare il sensore a contatto con la pelle dell'operatore per misurare la temperatura della superficie corporea, correlabile con una misura indicativa della temperatura corporea.

Con l'apparecchiatura appena introdotta, è ora possibile monitorare tutti i parametri desiderati. In particolare, si procede come segue:

+ L'Arduino Nano 33 BLE Sense viene utilizzato per monitorare la temperatura, pressione e umidità ambientale. Inoltre, per avere indicazione sul corretto utilizzo del casco di protezione, si fa riferimento alla misura del sensore di prossimità: questo viene posizionato orientato verso il capo dell'operatore, in maniera tale da rilevare qualora il casco si trovi sulla testa o meno.
+ L'Arduino Nano 33 IoT viene impiegato per la misura di accelerazioni lineari e velocità angolari, fornisce dunque le indicazioni sui movimenti dell'operatore.
+ Infine, direttamente collegato all'Arduino Nano 33 IoT, viene utilizzato il sensore di temperatura Grove per fornire indicazioni sulla temperatura della superficie corporea del lavoratore.

Per avere un quadro affidabile delle condizioni in cui si trova ad operare l'individuo, risulta opportuno utilizzare una frequenza di campionamento di 50 Hz. Tutti i parametri vengono dunque raccolti ogni 20 ms. Questo viene effettuato principalmente per avere dei dati attendibili per il monitoraggio dello stato di attività dei lavoratori e che possano essere utilizzati per la rilevazione di impatti e/o cadute. Inoltre, tale frequenza di campionamento risulta utile per valutare eventuali discontinuità e anomalie negli andamenti dei parametri ambientali e di salute dell'operatore.

A tal punto, quello che manca per la completa realizzazione del sistema è il dispositivo che si occupa della raccolta, salvataggio ed elaborazione dati. A questo proposito, è necessario utilizzare un computer in grado di interfacciarsi con le

schede Arduino e scambiarsi i dati letti dai diversi sensori. Questo può essere effettuato utilizzando uno dei due seguenti software:

+ MATLAB: permette la lettura dei dati scritti da una delle due schede Arduino sulla porta seriale del computer.
+ Processing: consente la ricezione tramite Wi-Fi dei dati registrati da una delle due schede Arduino.

In entrambi i casi, è possibile importare, analizzare e salvare i dati letti dai sensori collocati sulle schede Arduino. Tali dati devono però essere inviati al calcolatore con una ben precisa formattazione. Sia per MATLAB che per Processing è sufficiente realizzare un opportuno script che permetta di leggere i dati in arrivo dalla scheda Arduino, salvarli ed effettuare una prima analisi, volta all'individuazione di anomalie e allerte. Nel caso di Processing, è possibile il collegamento "scheda Arduino – calcolatore" tramite Wi-Fi e i dati vengono salvati all'interno di un file .csv da elaborare in un secondo momento, ad esempio mediante MATLAB, per permettere la loro rappresentazione grafica ed analisi. MATLAB consente invece la sola comunicazione tramite porta seriale con la scheda Arduino; dunque, risulta adeguato in una prima fase di modellazione e prototipazione ma non per l'utilizzo quotidiano del sistema. Per questo motivo si sceglie di operare con Processing, che permette la connessione Wi-Fi, ed utilizzare MATLAB per la successiva elaborazione e rappresentazione grafica dei dati.

1.3 Modalità di comunicazione

Una volta definita l'apparecchiatura necessaria alla realizzazione del dispositivo indossabile sensorizzato, è necessario definire come avviene la comunicazione tra i diversi componenti del sistema.

Come precedentemente anticipato, l'Arduino Nano 33 BLE Sense è dotato di Bluetooth 4.0, mentre l'Arduino Nano 33 IoT è provvisto sia di Bluetooth 4.0 che di Wi-Fi. Per facilitare la comunicazione con il computer di controllo, è utile che tutti i dati vengano inviati ad esso in una sola volta: non è quindi conveniente che il computer si debba collegare ad entrambi i dispositivi. Per questo motivo, si è pensato di realizzare una rete tale per cui i dati dei sensori vengono inviati tutti ad una sola scheda Arduino, che poi li invia alla stazione di controllo.

A tal fine, è dunque necessario instaurare la comunicazione e il trasferimento dei dati tra le due schede Arduino. Essendo entrambe dotate di Bluetooth, la comunicazione tra le due schede sfrutta questa tipologia di connessione. Nello specifico, è la scheda Arduino Nano 33 BLE Sense che invia i dati letti dai sensori integrati alla scheda Arduino Nano 33 IoT. Dunque, la prima leggerà i valori di temperatura, pressione, umidità e prossimità registrati dai suoi relativi sensori e li invierà tramite Bluetooth alla seconda.

I dati devono poi essere trasferiti al computer centrale di controllo. Questa comunicazione avviene tra il computer e l'Arduino Nano 33 IoT mediante Wi-Fi. Rispetto al Bluetooth, infatti, la tecnologia Wi-Fi permette la comunicazione su distanze nettamente maggiori e offre una maggior ampiezza di banda. Dunque, l'Arduino Nano 33 IoT, una volta raccolti i dati di tutti i sensori (quelli a sé integrati e quelli integrati all'Arduino Nano 33 BLE Sense), li invia mediante Wi-Fi ad un computer dotato di Processing per la ricezione e MATLAB per l'analisi.

In sostanza, quindi, l'Arduino Nano 33 IoT opera come nodo nel trasferimento di dati: raccoglie i dati di tutti i sensori del dispositivo indossabile IoT e li invia ad un server centrale che si occupa della loro analisi e raccolta. La Figura 2Figura 2 - Schema dell'apparecchiatura costituente il dispositivo IoT e della tipologia di comunicazione riporta uno schema della comunicazione tra i diversi apparecchi costituenti il casco sensorizzato.

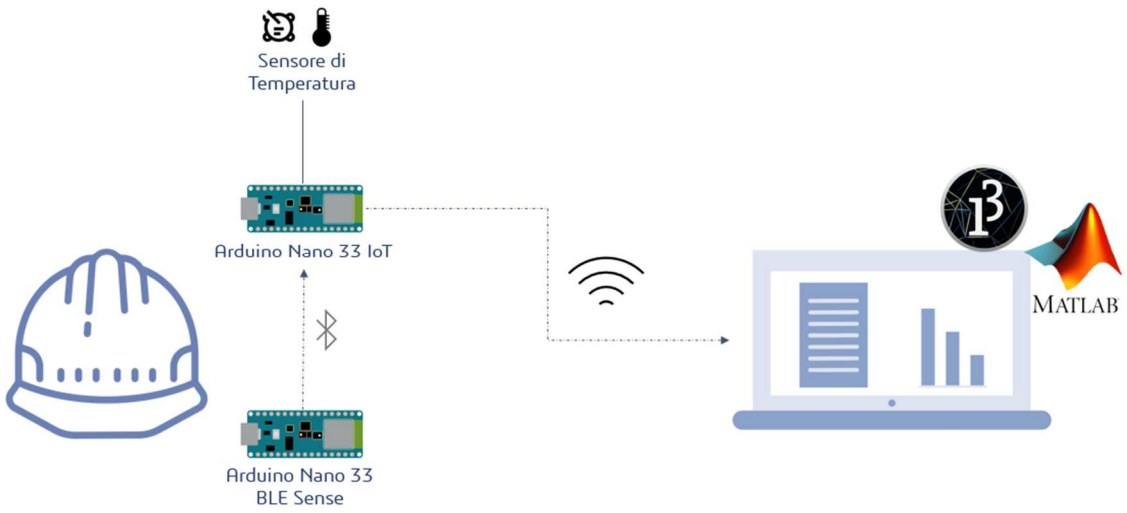


Figura 2 - Schema dell'apparecchiatura costituente il dispositivo IoT e della tipologia di comunicazione

Per la comunicazione, si sfrutta quindi il modulo NINA-W10 della scheda Arduino Nano 33 IoT, che garantisce la duplice modalità di connessione Bluetooth e Wi-Fi. Tuttavia, le due tecnologie non sono in grado di operare contemporaneamente. Per questo motivo è necessario, durante il funzionamento della scheda, passare continuamente dalla modalità Bluetooth a quella Wi-Fi. In particolare, per la maggior parte del tempo l'Arduino Nano 33 IoT è collegato via Wi-Fi al computer, per inviare i dati in tempo reale di accelerazione lineare e velocità angolare, caratterizzati da maggiore variabilità. Dopodiché, ogni N minuti, si scollega il Wi-Fi e si attiva il Bluetooth: viene a tal punto effettuata la ricerca del dispositivo Arduino Nano 33 BLE Sense e le due schede si collegano via Bluetooth. Vengono così inviati i dati mediati sugli N minuti raccolti dall'Arduino Nano 33 BLE Sense all'Arduino Nano 33 IoT. Una volta effettuato l'invio, i due dispositivi si disconnettono, sull'Arduino Nano 33 IoT viene nuovamente attivato il Wi-Fi e riparte l'invio dei dati al computer, compresi quelli mediati appena raccolti dall'Arduino Nano 33 BLE Sense.

Il sistema così realizzato ha però l'obiettivo di operare con tutti i lavoratori presenti all'interno del luogo di lavoro. Dunque, vi saranno più operatori, ciascuno con un casco sensorizzato che monitora condizioni ambientali, stato di attività e temperatura della superficie corporea dell'operatore. Tutti questi dispositivi devono inviare i dati rilevati al computer centrale di controllo; quindi, vi saranno più caschi sensorizzati che comunicano contemporaneamente via Wi-Fi con il server centrale. Si può allora far riferimento ad un sistema del tipo Figura 3Figura 3 - Schema di connessione multipla tra dispositivi sensorizzati e calcolatore centrale, in cui il computer riceve i dati relativi a diversi operatori, li processa, raccoglie ed elabora tutti nello stesso momento.

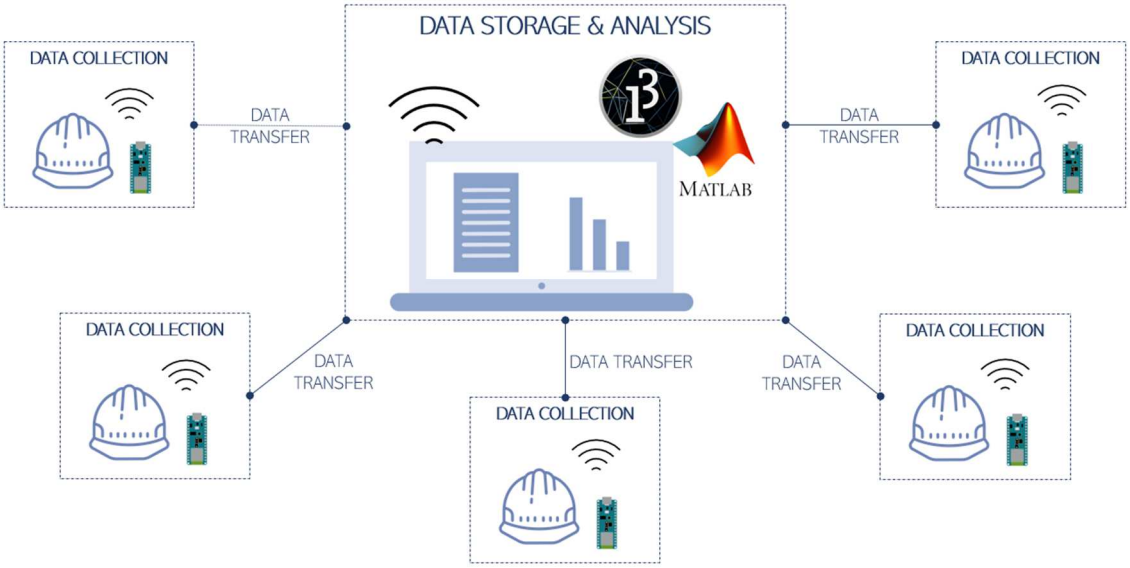


Figura 3 - Schema di connessione multipla tra dispositivi sensorizzati e calcolatore centrale

1.4 Trasferimento e analisi dati

Per quanto concerne la raccolta dei dati, come precedentemente anticipato, si è deciso di utilizzare il microcontrollore Arduino Nano 33 IoT per monitorare accelerazione e velocità angolare, e la scheda Arduino Nano 33 BLE Sense per la rilevazione delle condizioni ambientali. Inoltre, la frequenza di campionamento adatta ad una rilevazione affidabile dei parametri da monitorare è risultata essere 50 Hz. Oltre alla fase di raccolta dei dati, è però importante definire anche le modalità con le quali avviene il trasferimento e l'analisi dei dati stessi. Si procede diversamente per i parametri rilevati dall'Arduino IoT e quelli raccolti dall'Arduino BLE Sense.

Per quanto riguarda i dati di accelerazione lineare e velocità angolare, che hanno una variabilità elevata, si effettua il campionamento ogni 20 ms, si calcola il valore medio su un secondo e infine si manda tale valore al calcolatore per la raccolta ed analisi dati. A tal fine si utilizza l'Arduino Nano 33 IoT, che viene collegato

direttamente al calcolatore tramite Wi-Fi. Lo stesso procedimento si segue anche per la rilevazione della temperatura della superficie corporea, essendo il relativo sensore integrato alla scheda in analisi. Dunque, ogni secondo si calcola la media dei valori rilevati (accelerazione, velocità angolare, temperatura) e la si manda al calcolatore per le successive analisi.

I dati di temperatura, umidità, pressione ambientale e prossimità, invece, avendo una variabilità decisamente inferiore, vengono raccolti dall'Arduino Nano 33 BLE Sense. Il campionamento è effettuato ancora ogni 20 ms, ma si mandano i valori medi via Bluetooth all'Arduino IoT ogni N minuti. In questo caso, quindi, i valori vengono mediati non più sul secondo ma su N minuti, poiché si tratta di parametri con una minore variabilità rispetto ad accelerazioni lineari e velocità angolari. I dati in arrivo dall'Arduino BLE Sense via Bluetooth ogni N minuti vengono poi subito inviati al calcolatore tramite Wi-Fi dalla scheda Arduino Nano 33 IoT.

Per la maggior parte del tempo, quindi, come anticipato in precedenza, l'Arduino IoT è connesso al computer via Wi-Fi e manda tutti i dati monitorati ogni secondo. Ogni N minuti, poi, la connessione Wi-Fi viene interrotta, si instaura la connessione Bluetooth tra le due schede Arduino e avviene il trasferimento dei dati relativi alle condizioni ambientali. Terminato tale trasferimento, la connessione tra Arduino IoT e computer via Wi-Fi viene restaurata permettendo nuovamente il trasferimento delle informazioni ogni secondo.

Una volta effettuato il trasferimento dei dati, è necessario procedere con la loro elaborazione, analisi e salvataggio. Questo viene effettuato in parte sul calcolatore e in parte direttamente sulla scheda Arduino Nano 33 IoT. L'analisi dati include principalmente l'utilizzo dei valori di accelerazione lineare per la rilevazione di eventuali impatti e/o cadute dell'operatore mediante opportuni algoritmi. In questo modo, infatti, è possibile allertare i soccorsi non appena un incidente è avvenuto, garantendo un pronto intervento. Inoltre, è utile andare a valutare la variabilità degli altri parametri monitorati, per rilevare qualora vi siano sbalzi di temperatura, vi sia la rimozione del casco di sicurezza, ecc. Questo può ad esempio essere effettuato mediante la rilevazione automatica in tempo reale di forti irregolarità negli andamenti dei differenti parametri.

1.4.1 Algoritmo di Fall Detection

In letteratura sono presenti numerosi studi riguardanti la messa a punto di algoritmi in grado di rilevare la presenza di cadute a partire dai segnali di accelerazione misurati mediante sensori indossabili. In particolare, esistono principalmente due metodi per individuare la presenza di cadute: l'utilizzo di soglie ("threshold") o metodi di machine learning. Nel metodo delle soglie si usano dei sensori, come un accelerometro a tre assi, per calcolare dati che vengono confrontati con una soglia opportunamente definita. Si può utilizzare un metodo

a singola soglia (con alta sensitività e bassa specificità) o a multiple soglie (maggiore specificità, minore sensitività). Il metodo che impiega invece il machine learning include l'algoritmo SVM, distribuzioni gaussiane, alberi decisionali o modelli di Markov nascosti (HMM). Questo metodo è più sofisticato, permette di rilevare la presenza di cadute con una maggiore accuratezza, ma è più difficile da implementare. Per questo motivo nel presente lavoro si farà utilizzo del metodo delle soglie, che fornisce una risposta estremamente rapida con un minor consumo di risorse. [22, 23]

Per quanto riguarda gli algoritmi di rilevazione cadute threshold-based, la letteratura offre numerosi e differenti spunti. Alcuni studi si basano infatti sull'utilizzo di una sola soglia, altri sull'utilizzo dei soli segnali di accelerazione, alcuni sull'utilizzo di entrambi i segnali di accelerometro e giroscopio [24, 25]. In realtà, l'impiego delle sole accelerazioni risulta adeguato al fine di determinare la presenza di cadute, poiché permette di fornire direttamente informazioni sul movimento sia lineare che angolare. L'utilizzo di una sola soglia potrebbe invece condurre a stime errate poiché permette di distinguere movimenti di alta intensità senza però differenziare una caduta da altri movimenti normali, come il salto, che possono ugualmente produrre valori di picco. Per queste ragioni l'algoritmo selezionato di rilevazione della caduta si baserà sui soli segnali dell'accelerometro e farà riferimento a più valori di soglia. [22, 26]

I diversi studi presenti in letteratura concordano sulle principali fasi in cui è possibile suddividere una caduta. A ciascuna di queste fasi è associabile un valore tipico di modulo di accelerazione lineare. Quest'ultimo è calcolabile come:

$$|a| = \sqrt{a_x^2 + a_y^2 + a_z^2}$$

Le fasi che costituiscono tipicamente un evento di caduta sono quattro, ossia:

1. Nel momento in cui la caduta inizia, il corpo sperimenta una prima fase di caduta libera. In questa fase il modulo dell'accelerazione deve scendere al di sotto del cosiddetto "free fall threshold", pari a 0.6/0.75 g, dove g è l'accelerazione di gravità (= 9.80665 m/s^2).
2. Avvenuta la caduta libera, è necessario rilevare la presenza di un impatto, nel momento in cui il corpo dell'individuo impatta a terra. Tale fase è caratterizzata da un valore di modulo dell'accelerazione al di sopra di 2g.
3. A seguito della rilevazione dell'impatto si ha il "recovery state", durante il quale l'individuo giace a terra, e pian piano si riprende dalla caduta. In questa fase il soggetto non si muove, per cui si ha $|a| \approx g$. Infatti, in posizione di quiete, le accelerazioni lungo due direzioni sono nulle, mentre l'accelerazione lungo la direzione verticale risulta all'incirca pari a g.

4. Infine, è possibile che l'individuo, a seguito della caduta, abbia perso conoscenza. In tal caso la fase di "recovery state" risulta prolungata, dato che l'individuo non è in grado di riprendersi, e per più tempo $|a| \approx g$.

Dalla precedente caratterizzazione delle diverse fasi della caduta, è possibile individuare le tre soglie che è necessario verificare per la rilevazione di una caduta: quella relativa alla fase di caduta libera, quella relativa all'impatto e quella relativa alla fase di recovery. Tuttavia, in un algoritmo di rilevazione cadute risulta anche fondamentale tenere conto dello scorrere del tempo. Infatti, dal momento in cui si registra la caduta libera è necessario che non passi troppo tempo fino al rilevamento dell'impatto: l'intervallo temporale tra una caduta libera e il riconoscimento di un impatto non deve essere superiore a 800 ms. Se non si riconosce nessun impatto in questo periodo, allora nessuna caduta è rilevata e l'algoritmo si resetta per riconoscere ulteriori cadute libere. Dopodiché, constatato l'impatto, vi è la fase di "recovery". Questa deve durare almeno 1 s per essere riconosciuta come tale, poiché l'operatore non riesce istantaneamente ad alzarsi dopo l'impatto. Anche in questo caso, se viene riconosciuta $|a| \approx g$ per una durata inferiore a 1 s, l'algoritmo si resetta, pronto a riconoscere altre cadute libere. Il riconoscimento di una fase di "recovery" è quindi essenziale per la fall detection, ma non indica una possibile perdita di conoscenza. Quest'ultima viene individuata durante la fase critica che segue la "recovery phase": se nessun movimento viene riconosciuto per ulteriori 5 s, allora una perdita di conoscenza può essere assunta. Dunque, è possibile che vi sia stata perdita di conoscenza se dopo la fase di "recovery", per ulteriori 5 s si registra $|a| \approx g$. [26, 27, 28]

In definitiva, l'algoritmo selezionato per la rilevazione di cadute è quello rappresentato in Figura 4 - Algoritmo di fall detection utilizzato. Una volta calcolato il modulo dell'accelerazione |a|, la fase di caduta libera si registra in caso di $|a| < 0.6$ g. Dopodiché, entro 800 ms dalla rilevazione della fase di caduta, si accerta l'impatto se risulta $|a| > 2g$. Si passa poi all'individuazione della fase di "recovery" se per almeno un secondo $0.95 \text{ g} < |a| < 1.065 \text{ g}$ ($|a| \approx g$). Infine, nel caso in cui tale condizione sull'accelerazione permanga per ulteriori 5 s, è possibile che l'operatore abbia perso conoscenza. A tutte le fasi (caduta libera, impatto, "recovery") se entrambe le condizioni sulle accelerazioni e sui tempi non sono rispettate, l'algoritmo si resetta, pronto al riconoscimento di una nuova fase di caduta libera.

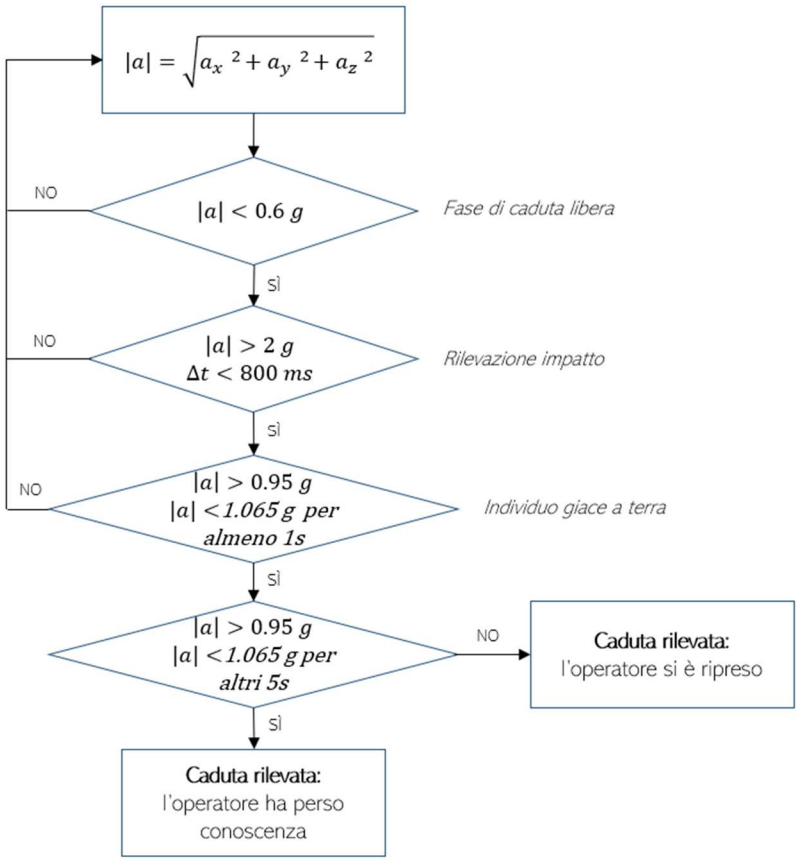


Figura 4 - Algoritmo di fall detection utilizzato

1.4.2 Algoritmo di Impact Detection

È stato dimostrato che le lesioni cerebrali possono derivare da eventi di concussione, da colpi diretti o indiretti al capo, ma anche dall'accumulo di impatti più lievi [29, 30]. È quindi necessario monitorare l'eventuale presenza di impatti al capo in maniera tale da procedere con un pronto intervento e il soccorso della persona quando necessario.

Per quanto concerne l'individuazione di impatti, così come per la rilevazione di cadute, in letteratura sono presenti numerosi studi volti alla determinazione di un algoritmo in grado di identificarli. In questo caso, tuttavia, essendo l'impatto un fenomeno istantaneo, vengono utilizzati solamente algoritmi a singola soglia basati sui dati dell'accelerometro. Ci sono però alcune divergenze sui valori di soglia individuati dai differenti autori. Inoltre, bisogna considerare il fatto che il limite soglia per le lesioni è diverso per ciascuna persona, data la natura multifattoriale delle lesioni.

In generale, gli impatti vengono identificati come tali nel caso in cui si registri un modulo dell'accelerazione lineare maggiore a 10 g. Infatti, impatti con una accelerazione inferiore ai 10 g sono considerati trascurabili in quanto la loro relazione con traumi alla testa rende difficile distinguere tra impatti alla testa e

movimenti della testa volontari [29]. In alternativa, come soglia si può identificare 14.4 g, meno utilizzata della precedente perché permette di individuare un numero minore di impatti [29, 30]. Infatti, aumentando il valore limite da 10 g a 15 g, il numero di impatti rilevati si riduce del 45%, e l'81% del totale degli impatti identificabili viene rimosso. Per questo gli studi con valori minori di soglia registrano un maggior numero di impatti e potrebbero anche mostrare minori ampiezze medie di impatto rispetto agli studi con valori di soglia maggiori [30]. Dunque, siccome non esiste un criterio prestabilito per la rendicontazione degli impatti alla testa e la maggior parte degli studi ha riportato la soglia di accelerazione lineare risultante di 10 g, si è deciso di utilizzare questo valore come soglia di riferimento per l'individuazione di impatti al capo.

Infine, una interessante estensione dell'algoritmo di rilevazione impatti sarebbe quella di calcolare l' "Head Injury Criterion" (HIC). Questo è infatti uno dei parametri più importanti in termini di sopravvivenza umana in quanto fornisce una indicazione delle lesioni cerebrali dovute all'impatto al capo. Questo indice si può stimare integrando l'accelerazione risultante della testa (misurata nel suo centro di gravità) in una finestra temporale secondo la seguente definizione:

$$HIC(\Delta t_{max}) = \max_{t_1,t_2}\left[\left(\frac{1}{t_2-t_1}\int_{t_1}^{t_2} a(t)dt\right)^{2.5}(t_2-t_1)\right] \qquad con\ t_2-t_1 \leq \Delta t_{max}$$

Dove a(t) è il modulo dell'accelerazione, espresso in g, t_1 l'istante iniziale, t_2 l'istante finale, e Δt_{max} è l'intervallo massimo, tipicamente pari a 15 ms o 36 ms. L'HIC non deve superare il valore di 1000, poiché ad esso corrisponde una probabilità del 18% di trauma cranico grave, una probabilità del 55% di trauma serio e una probabilità del 90% di trauma cranico moderato per l'adulto medio [5]. La gravità di una lesione si può anche relazionare all'Abbreviated Injury Scale (AIS), che va da AIS 0, che significa "nessuna lesione", a AIS 6, che significa "virtualmente lesione cui non è possibile sopravvivere". L'HIC opera come indice di lesione ed è correlabile al rischio di AIS maggiore o uguale a 2 per la frattura del cranio e a AIS maggiore o uguale a 4 per i danni al cervello. La conversione da HIC a AIS può essere effettuata facendo riferimento alle curve di Prasad/Mertz [31, 32].

1.4.3 Rete di trasferimento e analisi dati

Nel seguito vengono analizzati più nel dettaglio i ruoli dei tre soggetti coinvolti nella rete di rilevazione, trasferimento, elaborazione e salvataggio dei dati.

La scheda Arduino Nano 33 BLE Sense viene utilizzata per il monitoraggio delle condizioni ambientali, nello specifico di temperatura, umidità e pressione. Inoltre,

essendo la scheda dotata del sensore di prossimità APDS9960, si utilizza tale sensore per rilevare il corretto posizionamento del casco da lavoro. In particolare, tale sensore fornisce in output un valore intero compreso tra 0 e 255, dove 0 significa che è rilevato un soggetto molto prossimo al sensore e 255 significa che non vi sono ostacoli in prossimità del sensore. Nel caso in cui il casco sia correttamente posizionato, dunque, ci si aspetta un valore di prossimità tendente a zero.
Nella rete BLE, avendo il ruolo di peripheral, tale microcontrollore ha sempre il Bluetooth attivo e, quando viene connesso al central (l'Arduino Nano 33 IoT), calcola le medie di tutti i valori registrati durante il periodo di non connessione e le scrive sul servizio che verrà poi letto dal dispositivo central. In questo modo avviene il trasferimento dei dati relativi alle condizioni ambientali ogni N minuti.

L'Arduino Nano 33 IoT viene invece impiegato per rilevare i dati di giroscopio e accelerometro integrati sulla scheda stessa, leggere i dati del sensore di temperatura Grove aggiunto, ricevere le informazioni sulle condizioni ambientali dalla scheda Arduino Nano 33 BLE Sense via Bluetooth e mandare tutti i valori al calcolatore mediante Wi-Fi.
I dati relativi all'accelerazione lineare, alla velocità angolare e alla temperatura della superficie corporea vengono mediati su un secondo prima di essere trasferiti via Wi-Fi al calcolatore dotato di Processing. Invece, quelli relativi alle condizioni ambientali, si ottengono via Bluetooth dalla scheda Arduino Nano 33 BLE Sense ogni N minuti. L'Arduino Nano 33 IoT lavora come central nella comunicazione, per cui è il soggetto in grado di attivare e inibire la connessione Bluetooth.
Il trasferimento Wi-Fi dei dati al calcolatore avviene mediante la scrittura degli stessi su un messaggio da inviare al server (Processing): questi devono essere scritti con una determinata formattazione, ossia con tutti i valori separati dallo slash " / ".

Infine, la scheda Arduino Nano 33 IoT ha un'ulteriore funzione. Per quanto concerne la rilevazione di impatti e cadute, poiché l'algoritmo selezionato si basa sull'individuazione di massimi e minimi della funzione accelerazione, la sua implementazione su Processing non risulta possibile. Infatti, sul calcolatore, operando con medie sul secondo, si perdono i picchi della funzione per cui non è più possibile differenziare le diverse fasi della caduta. Per questo motivo, l'algoritmo di fall e impact detection è implementato direttamente nel codice Arduino IDE della scheda Arduino Nano 33 IoT, che ha a disposizione i valori di accelerazione ogni 20 ms. A tal punto, a seconda di quanto viene rilevato dall'algoritmo, si manda al calcolatore la variabile "check", contenente uno dei seguenti valori:

+ 0 nel caso in cui non venga rilevato nulla;
+ 1 nel caso di fasi interne all'algoritmo di rilevazione caduta rilevate;
+ 2 in caso di caduta con possibile perdita di conoscenza rilevata;

+ 3 in caso di caduta rilevata;
+ 5 in caso di impatto rilevato.

Questo valore viene poi letto da Processing e trasformato in un allarme da visualizzare sull'interfaccia grafica a seconda del tipo di evento rilevato.

In conclusione, i dati trasferiti da Arduino Nano 33 IoT a Processing sono:

+ Accelerazioni lineari mediate sul secondo lungo X, Y e Z, ottenute direttamente sulla scheda Arduino IoT;
+ Velocità angolari mediate sul secondo attorno X, Y e Z, ottenute direttamente sulla scheda Arduino IoT;
+ Temperatura della superficie corporea mediata sul secondo, ottenuta dal sensore di temperatura Grove aggiunto alla scheda Arduino IoT;
+ Variabile "check", ottenuta dall'algoritmo di fall e impact detection implementato nel programma dell'Arduino Nano IoT;
+ Temperatura ambientale, valore medio in arrivo dall'Arduino Nano 33 BLE Sense ogni N minuti;
+ Umidità ambientale, valore medio in arrivo dall'Arduino Nano 33 BLE Sense ogni N minuti;
+ Pressione ambientale, valore medio in arrivo dall'Arduino Nano 33 BLE Sense ogni N minuti;
+ Valore medio rilevato dal sensore di prossimità, in arrivo dall'Arduino Nano 33 BLE Sense ogni N minuti.

Sul calcolatore, come precedentemente anticipato, si può effettuare la ricezione dei dati rilevati dalle schede Arduino mediante i software MATLAB o Processing. Tuttavia, solamente il secondo permette la connessione Wi-Fi con la scheda Arduino, risultando dunque adeguato all'applicazione. In particolare, per la connessione Wi-Fi, si utilizza la tecnologia web del WebSocket, che permette lo scambio di dati basato su protocollo TCP tra un server e un client. Il server è rappresentato dal calcolatore con Processing, e deve essere sempre attivo, mentre il client è la scheda Arduino Nano 33 IoT, che decide quando attivare il Wi-Fi e quando connettersi al server. Questa tecnologia permette l'invio di messaggi dal client (Arduino) al server (Processing). Tali messaggi contengono i diversi dati in arrivo dalla scheda Arduino IoT separati da " / ", in questo modo il software è in grado di riconoscerli e registrarli all'interno di variabili. Il salvataggio dei dati è permesso mediante la scrittura degli stessi all'interno di una tabella, che viene al termine della connessione salvata in un file .csv. Tale file dovrà essere successivamente elaborato per permettere l'analisi e la rappresentazione grafica dei dati: questo viene eseguito su MATLAB. Una prima elaborazione dei dati è però anche effettuata su Processing: si tratta della visualizzazione di allarmi sulla Console Area del software nel caso in cui si individuino cadute, cadute con possibile perdita di conoscenza o impatti. Queste allerte sono possibili grazie all'analisi della variabile check in arrivo dalla scheda Arduino. Infine, il

programma Processing è in grado di rilevare automaticamente forti variazioni dei parametri ambientali e della temperatura corporea e riconoscere qualora il casco non sia indossato (valore di prossimità tendente a 255). Anche in questi casi viene mostrato un allarme sulla console del programma, indicante il tipo di anomalia riscontrata. L'analisi dati condotta sul computer permette quindi ai supervisori di tenere costantemente sotto controllo le condizioni di tutti gli operatori, ed essere aggiornati istantaneamente in caso di situazioni anormali o rischiose.

In definitiva, la rete di trasferimento dei dati risulta quella mostra in Figura 5 - Rete di trasferimento dei dati. La comunicazione tra Arduino Nano 33 BLE Sense e Arduino Nano 33 IoT non è costante, ma attiva ogni N minuti, e il trasferimento dei dati tra le due schede avviene mediante Bluetooth LE. La comunicazione, invece, tra Arduino Nano 33 IoT e il calcolatore (con Processing per la ricezione e una prima analisi dati e MATLAB per la rappresentazione grafica e la successiva elaborazione dati) è continuativa, e ogni secondo le medie dei dati vengono trasferite dal microcontrollore al calcolatore tramite Wi-Fi.

Figura 5 - Rete di trasferimento dei dati

1.4.4 Esempio applicativo

In questo paragrafo viene mostrato un esempio applicativo del dispositivo realizzato. Esso è stato ottenuto con un tempo di raccolta dei dati di cinque minuti e considerando la connessione Bluetooth tra Arduino IoT e Arduino BLE Sense ogni minuto. In totale giungono dunque al software 300 valori per ciascuna variabile, uno per ogni secondo. L'invio dei dati dalla scheda Arduino Nano 33 IoT al calcolatore viene effettuato tramite Wi-Fi utilizzando il software Processing. Su esso viene anche effettuata una prima analisi dati con l'individuazione di allarmi e anomalie. La rappresentazione grafica dei dati viene invece effettuata in un secondo momento su MATLAB, partendo dal file .csv generato da Processing.

I dati di accelerazione lineare e velocità angolare variano ogni secondo, poiché costantemente aggiornati dall'Arduino Nano 33 IoT. Infatti, come definito al paragrafo 1.4, i dati raccolti da tale scheda vengono mediati sul secondo e mandati istantaneamente al calcolatore centrale tramite Wi-Fi. Dunque, ci si aspetta la stessa frequenza di variazione anche per la temperatura della superficie corporea, essendo il relativo sensore integrato all'Arduino Nano 33 IoT. Nelle Figura 6 e

Figura 7 sono riportati gli andamenti realizzati su MATLAB dell'accelerazione lineare e della velocità angolare nel tempo.

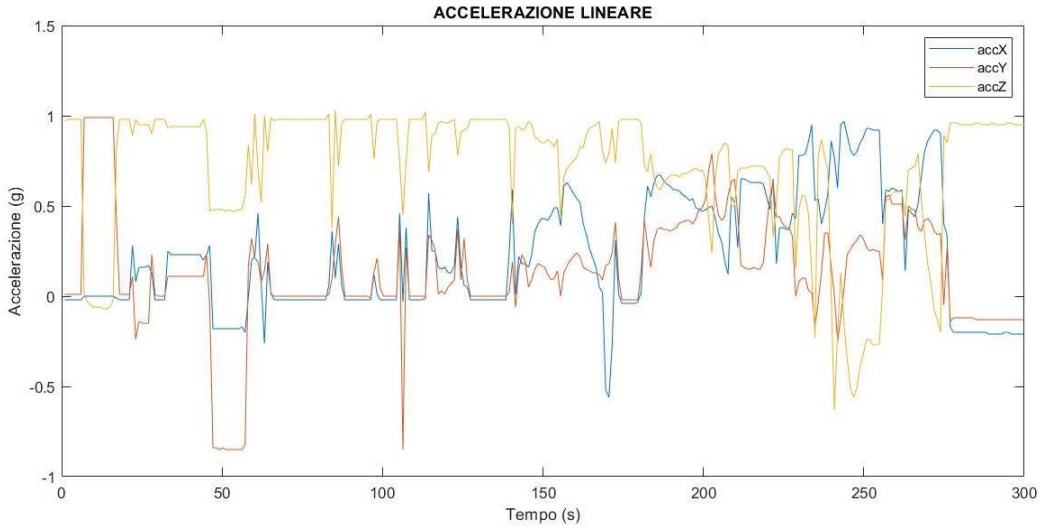


Figura 6 - Prova sperimentale: accelerazione lineare nel tempo

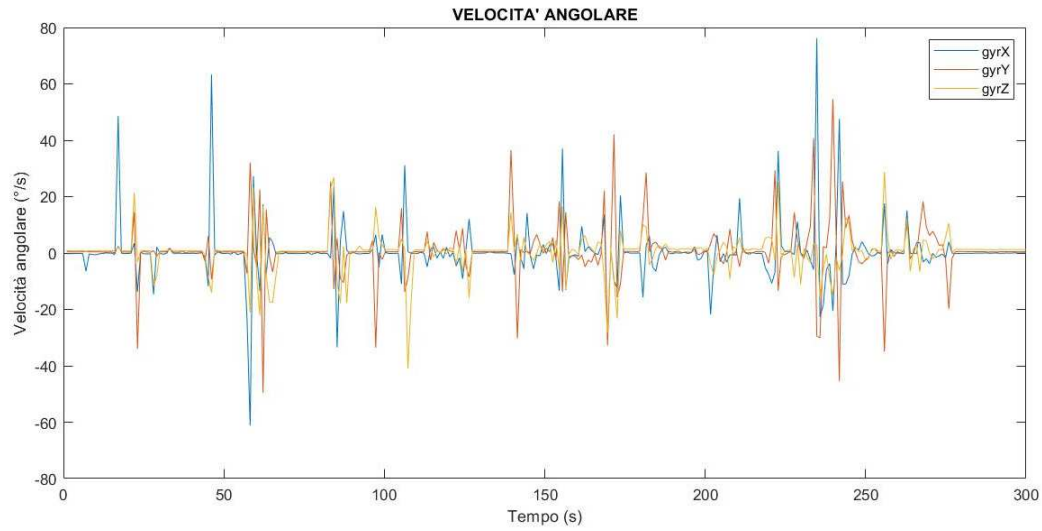


Figura 7 - Prova sperimentale: velocità angolare nel tempo

Come si nota, l'accelerazione lineare è calcolata in g, ossia in funzione del valore di accelerazione di gravità g pari a 9.80665 m/s^2. La velocità angolare è invece registrata in °/s.

Per controllare il corretto funzionamento dell'algoritmo di fall detection, si simulano alcune cadute. In particolare, attorno ai 50 s, si simula una caduta del dispositivo e poi lo si lascia fermo per più di 5 secondi. In effetti, in corrispondenza di tale istante temporale, sulla console di Processing appare il seguente allarme:

ATTENZIONE: caduta con possibile perdita di conoscenza rilevata!

Dunque, la caduta con possibile perdita di conoscenza è stata rilevata correttamente. Analogamente, durante la prova, si lascia cadere il dispositivo altre

tre volte, movimentandolo però ora subito dopo. In effetti, dopo circa 100 s, viene visualizzato il seguente allarme:

ATTENZIONE: caduta rilevata!

Che si ripete, in maniera analoga, anche attorno ai 170 s.

Dall'andamento dell'accelerazione lineare nel tempo è possibile riconoscere gli istanti in cui il sensore è lasciato cadere: in corrispondenza di tali situazioni si verificano delle forti variazioni di accelerazione. Tuttavia, anche poco prima i 250 s si è simulata una caduta, ma il dispositivo non ha segnalato nulla: ciò è dovuto al fatto che il sensore è stato movimento subito, non appena si è verificato l'impatto, per cui non è stato possibile individuare la fase di recovery caratteristica dell'algoritmo di fall detection (si rimanda al paragrafo 1.4.1 per ulteriori informazioni a riguardo).

Al termine di tale prova, risulta dunque verificato il funzionamento dell'algoritmo di fall detection, che ha identificato correttamente tutte le cadute che si sono simulate sperimentalmente.

In maniera analoga, si potrebbe testare il funzionamento dell'algoritmo di impact detection. Ciò nonostante, esso è tarato per riconoscere impatti di elevata intensità, dunque difficili da simulare. Si sono effettuate delle prove abbassando la soglia, per verificare che esso operasse correttamente, ma non è stato possibile sperimentare un vero impatto caratterizzato da modulo di accelerazione lineare superiore a 10 g, che risulta estremamente elevato.

Oltre ai dati registrati da accelerometro e giroscopio, tuttavia, vi sono anche quelli monitorati dagli altri sensori integrati sul dispositivo, ossia: temperatura della superficie corporea, temperatura ambientale, pressione ambientale, umidità ambientale e prossimità. Gli andamenti di tali parametri nel tempo, insieme ai valori della variabile "check" indicante anomalie nel movimento (impatti/cadute), sono riportati in Figura 8. Questi grafici si sono ottenuti su MATLAB, importando il file .csv automaticamente generato da Processing. Fuorché la temperatura della superficie corporea, che come precedentemente riportato viene mediata sul secondo, tutti gli altri valori possono variare soltanto ogni 60 s, poiché è stata impostata la connessione Bluetooth tra le due schede Arduino ogni minuto. Dunque, tali dati vengono comunque raccolti con una frequenza di campionamento di 50 Hz, ma poi mediati sul minuto.

Per valutare il funzionamento del sensore di prossimità, poco dopo l'inizio della prova si è coperto il sensore stesso con la mano. Effettivamente, da questo momento è stato rilevato un valore di prossimità nullo, equivalente alla massima vicinanza con il soggetto.

Attorno ai 125 s, sulla console di Processing è comparso il seguente allarme:

ANOMALIA: variazione repentina di umidità ambientale superiore al 10%

Ciò è probabilmente imputabile all'avvicinamento della mano al sensore, che ha fatto sì che la temperatura ambientale registrata aumentasse e con essa anche l'umidità.

Dopodiché, durante la seconda metà della prova, il sensore di prossimità è di nuovo lasciato libero, allontanando la mano da esso. Il valore di prossimità è effettivamente tornato prossimo al massimo, 255, e sulla console di Processing sono comparsi gli allarmi:

ALLERTA: casco di protezione rimosso!

ANOMALIA: variazione repentina di umidità ambientale superiore al 10%

Il primo allarme è legato al fatto che, essendo passato il valore di prossimità da 0 a 250 circa, il casco di protezione è stato rimosso. Il secondo allarme è invece legato alla diminuzione repentina di umidità ambientale, legata all'allontanamento della mano dal sensore, cui corrisponde anche una conseguente diminuzione di temperatura ambientale. Tali valori variano in maniera discontinua poiché, come detto, si riferiscono alle medie effettuate sul minuto.

Infine, poco prima del termine della prova, compare sullo schermo un nuovo allarme:

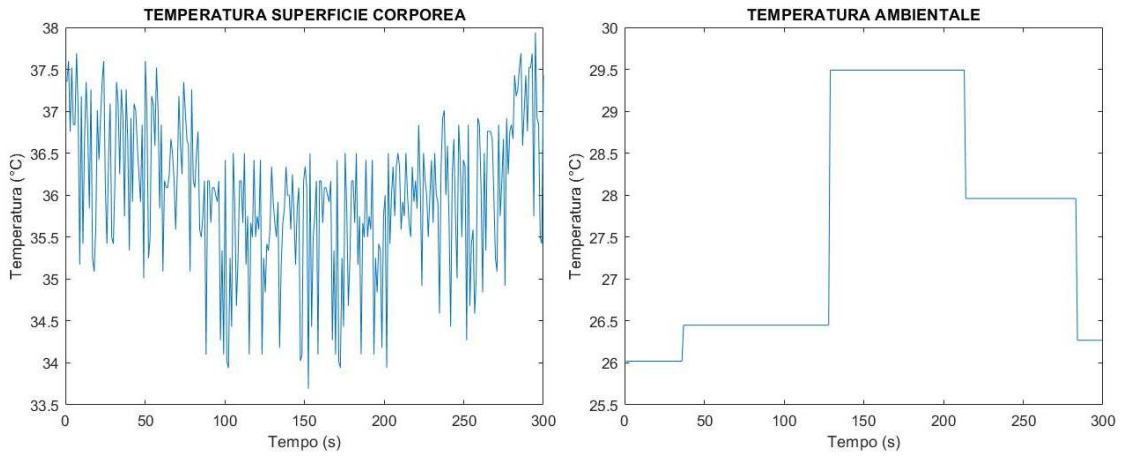
ALLERTA: casco di protezione non indossato!

Si tratta dell'avviso relativo al casco di protezione che, essendo il valore prossimità ancora prossimo a 250, non risulta ancora indossato.

Gli allarmi di anomalia relativi a forti variazioni dei parametri ambientali o della temperatura della superficie corporea vengono visualizzati nel caso di:

+ Variazione di temperatura della superficie corporea superiore a 5 °C;
+ Variazione di temperatura ambientale superiore a 5 °C;
+ Variazione di umidità ambientale superiore a 10 %;
+ Variazione di pressione ambientale superiore a 0.5 kPa.

Infine, l'allarme relativo al casco di protezione non correttamente indossato viene visualizzato per valori di prossimità superiori a 50.



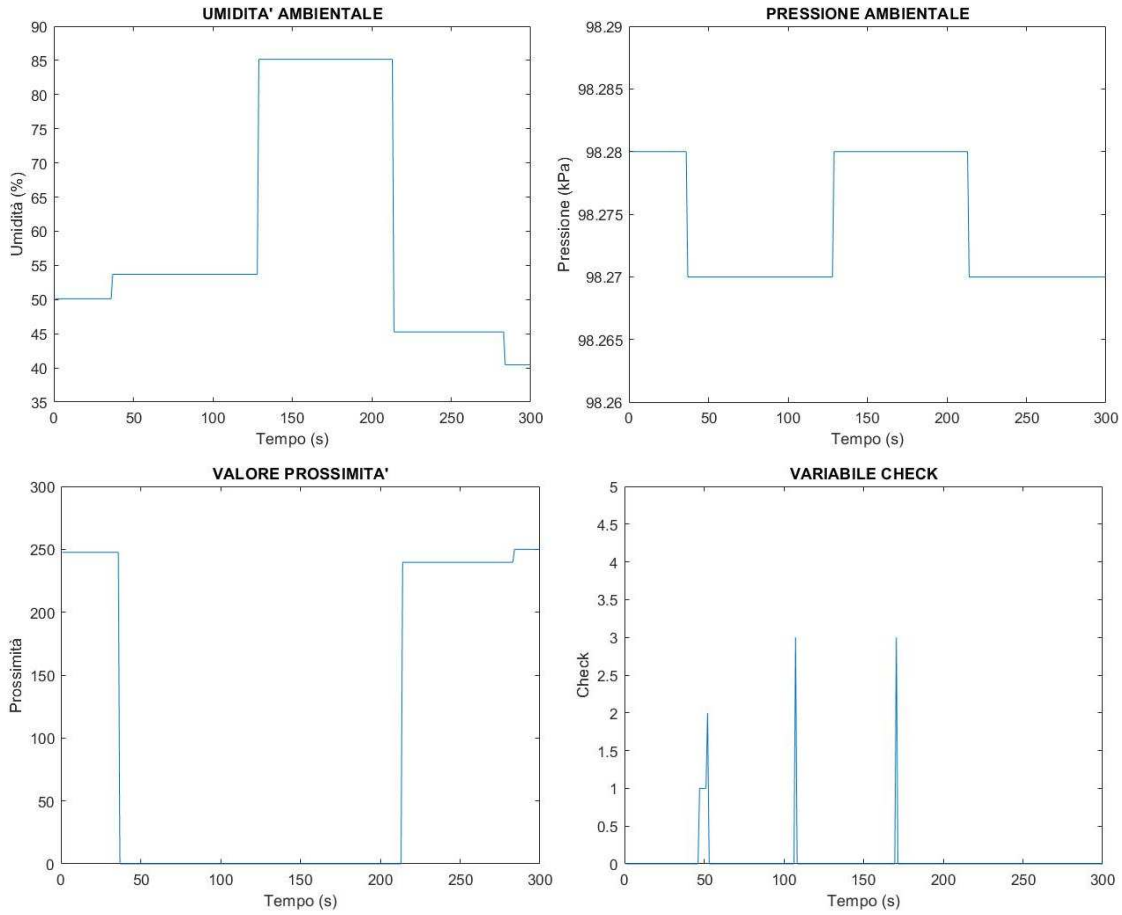


Figura 8 - Prova sperimentale: andamenti di temperatura della superficie corporea, temperatura ambientale, umidità, pressione, prossimità e variabile check nel tempo

L'esempio appena mostrato è stato utile per testare il funzionamento della rete di sensori realizzata in caso di trasmissione dei dati mediante Wi-Fi. È stato in tal modo verificato sia il corretto upload dei dati da Arduino Nano 33 IoT a Processing tramite Wi-Fi, sia la corretta comunicazione Bluetooth tra le due schede Arduino ogni minuto.

1.5 Sistema di avviso di prossimità

Per arricchire ulteriormente il dispositivo realizzato, è possibile integrare un sistema di avviso di prossimità. Si tratta di un sistema in grado di monitorare le distanze relative uomo – macchina e inviare allarmi in caso di operatori all'interno della zona di sicurezza del macchinario. [33]

Il sistema di avviso di prossimità è implementabile mantenendo il casco sensorizzato così come precedentemente definito, e dotando i macchinari in movimento nell'ambiente di lavoro di una scheda Arduino Nano 33 BLE Sense. Infatti, utilizzando la sola potenza del segnale Bluetooth emessa dalle schede Arduino Nano 33 BLE Sense situate sui caschi di protezione, è possibile avere una stima della distanza a cui si trova l'operatore rispetto al macchinario. Tale sistema

si basa sull'analisi dell'indicatore di potenza del segnale ricevuto (RSSI), che diminuisce all'aumentare della distanza tra i due dispositivi, e viceversa. È quindi possibile correlare l'RSSI con la distanza relativa tra i due soggetti. Questo è effettuabile mediante la seguente formula presente in letteratura [34]:

$$D = 10^{\left(\frac{RSSI_{1m} - RSSI}{10 \cdot N}\right)}$$

Dove D è la distanza stimata tra i due soggetti, $RSSI_{1m}$ è l'RSSI calcolato quando i due dispositivi sono a distanza relativa pari a 1 m, RSSI è l'attuale potenza del segnale misurata e N è una costante che dipende dal fattore ambientale, variabile tra 2 e 4. A causa di fattori esterni che influenzano le onde radio (come l'assorbimento, l'interferenza o la diffrazione), l'RSSI tende a fluttuare. In particolare, più sono distanti i dispositivi tra loro più è instabile il valore della potenza del segnale. Per questo motivo si otterrà una *stima* della distanza relativa uomo – macchina.

Dato che il calcolo della distanza a partire dalla potenza del segnale dipende dall'$RSSI_{1m}$ e dalla costante N dell'ambiente, è dapprima necessaria la taratura dell'equazione. Per quanto riguarda la costante N, è possibile determinare il suo valore nei diversi ambienti del luogo di lavoro, e poi farla variare a seconda del luogo in cui si trova il macchinario. Si specifica comunque che N varia poco all'interno di locali chiusi. Per il calcolo dell'$RSSI_{1m}$, bisogna invece considerare il fatto che esso dipende dal dispositivo impiegato, per cui varia a seconda dell'operatore. Si procede allora con una prima fase in cui si posizionano tutti i dispositivi a distanza pari 1 m dal macchinario e la scheda Arduino Nano 33 BLE Sense posizionata sulla macchina automaticamente rileva tali valori di $RSSI_{1m}$ di ciascun operatore. Dopodiché, questi ultimi diventano una costante del dispositivo cui si riferiscono ed è possibile iniziare ad utilizzare il sistema di avviso di prossimità.

Nel sistema, rappresentato schematicamente in Figura 9, vengono dunque coinvolte le schede Arduino Nano 33 BLE Sense presenti sui caschi sensorizzati e quelle situate sui macchinari in movimento. Le prime, come precedentemente illustrato, operano nella connessione Bluetooth come "peripheral": hanno quindi sempre il Bluetooth attivo e possono essere continuamente visualizzate da dispositivi "central" (come la scheda Arduino Nano 33 IoT collocata sul casco). Per l'integrazione del sistema di avviso di prossimità, esse non necessitano di alcun intervento, neanche dal punto di vista della programmazione del microcontrollore. Le schede presenti sui macchinari operano invece come "central devices": esse, mediante l'opportuna programmazione del microcontrollore, sono in grado di effettuare continuamente la scansione di tutti i dispositivi che si trovano nel loro intorno e, ogni volta che rilevano un dispositivo, domandarne la potenza del segnale. In questo modo l'RSSI viene trasformato nella distanza uomo – macchina,

e si invia un allarme nel caso in cui essa sia inferiore alla distanza di sicurezza del macchinario.

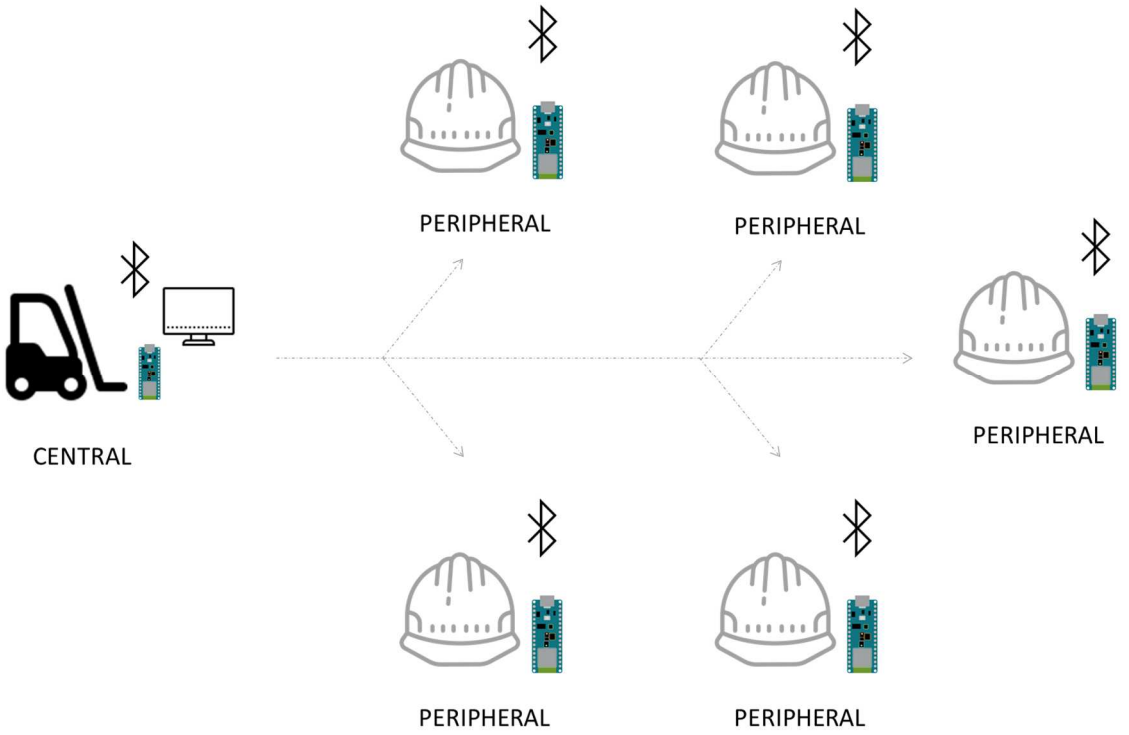


Figura 9 - Sistema di avviso di prossimità, soggetti coinvolti

Affinché il sistema non venga disturbato da tutti gli ulteriori dispositivi Bluetooth che possono trovarsi all'interno del luogo di lavoro, la scheda Arduino presente sul macchinario effettua la scansione dei dispositivi indossabili per nome. È dunque dapprima necessario inserire nello script Arduino IDE che programma la scheda il nome di tutti i dispositivi sensorizzati indossabili che si trovano nel luogo di lavoro. Solo in questo modo è permessa la scansione continua dei caschi sensorizzati presenti nell'ambiente.

Infine, per come attualmente realizzato il programma, tutte le volte che viene calcolata una distanza a partire dall'RSSI, il microcontrollore verifica che essa sia superiore ad una distanza di sicurezza precedentemente impostata e, in caso contrario, fornisce l'allarme:

ATTENZIONE: zona di sicurezza superata dall'operatore XX

Dove XX è il nome del dispositivo indossato dall'operatore che ha oltrepassato la zona di sicurezza del macchinario.

In futuro, dotando i macchinari di un display, questo sistema permetterebbe di mostrare in tempo reale tutti gli operatori che si trovano nell'intorno del macchinario con la loro posizione relativa rispetto allo stesso (si veda la Figura 10). In questo modo, sarebbe poi possibile implementare l'arresto del macchinario

o l'attivazione di una cicalina di allarme nel caso in cui la minima distanza di sicurezza sia oltrepassata. Per come pensato, tale sistema di avviso di prossimità risulta piuttosto semplice, poiché non necessita di modifiche al dispositivo sensorizzato indossabile né a livello di apparecchiatura necessaria né a livello di programmazione delle schede. Tuttavia, la misura della distanza a partire dall'RSSI è una misura approssimata, per cui fornisce una sola stima della distanza relativa, e non permette di conoscere l'effettivo posizionamento nello spazio degli operatori (per il quale sarebbe necessario un sistema più complesso di IPS – Indoor Positioning System).

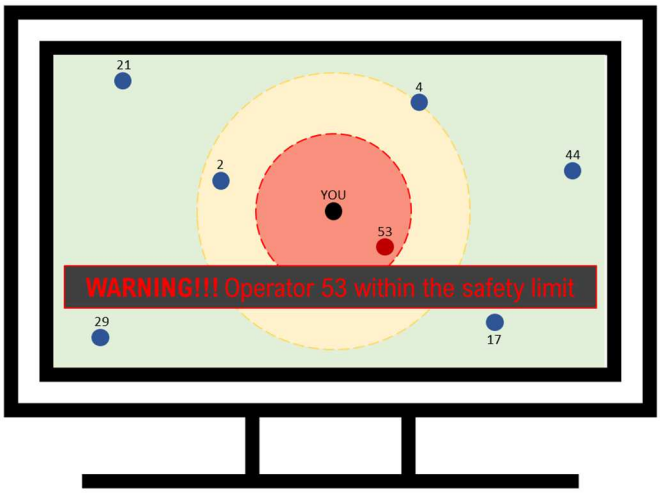


Figura 10 - Futura possibile integrazione del sistema di avviso di prossimità

1.6 Domanda energetica del sistema

Per ultimo, è necessario effettuare una valutazione pre-ottimizzazione della richiesta energetica del dispositivo di sicurezza indossabile IoT durante la fase di acquisizione e trasmissione dei dati. L'intenzione è quella di utilizzare una batteria per alimentare il dispositivo, per cui è dapprima necessario identificare la potenza assorbita durante il suo funzionamento per valutare se ciò sia possibile o meno. Tale analisi è stata effettuata alimentando una scheda Arduino alla volta e valutando l'entità della corrente assorbita da ciascun dispositivo ad una tensione prestabilita.

Per quanto concerne la scheda Arduino Nano 33 BLE Sense, questa è stata alimentata con una tensione pari a 4.5 V, sfruttando dunque il convertitore di tensione presente sulla scheda e necessario a convertire le tensioni in input in 3.3 V, tensione di funzionamento della scheda. Effettuata la prova, tramite l'alimentatore utilizzato è stato possibile valutare la corrente assorbita dalla scheda: si tratta di una corrente costante e pari a 0.18 A. Dunque, la potenza assorbita risulta:

$$P = V \cdot i = 4.5\,[V] \cdot 0.18\,[A] = 0.81\,[W]$$

Tale potenza è costante nel tempo poiché la scheda Arduino Nano 33 BLE Sense ha costantemente il Bluetooth acceso, poiché continuamente disponibile al collegamento con la scheda Arduino Nano 33 IoT. L'andamento della potenza assorbita dalla scheda in analisi nel tempo è mostrato in Figura 11a.

La scheda Arduino Nano 33 IoT è invece stata alimentata con una tensione di 5 V, sempre sfruttando il convertitore di tensione integrato nella scheda stessa. In questo caso, tuttavia, ci si aspetta per la potenza un andamento "a scalini", poiché tale scheda rimane per la maggior parte del tempo collegata al Wi-Fi per l'invio dei dati al calcolatore, assorbendo una certa potenza, e ogni n minuti si attiva invece il Bluetooth per la connessione con l'altra scheda Arduino, assorbendo una potenza differente. In particolare, la corrente richiesta al funzionamento dell'Arduino Nano 33 IoT è risultata essere:

+ 0.14 A durante il collegamento Wi-Fi con il calcolatore;
+ 0.10 A durante il collegamento Bluetooth con la scheda Arduino Nano 33 BLE Sense.

La potenza assorbita risultante avrà dunque andamento "a scalini" con il massimo pari a $P = 5\,[V] \cdot 0.14\,[A] = 0.7\,[W]$ e il minimo pari a $P = 5\,[V] \cdot 0.1\,[A] = 0.5\,[W]$. L'andamento che ne consegue è riportato in Figura 11b.

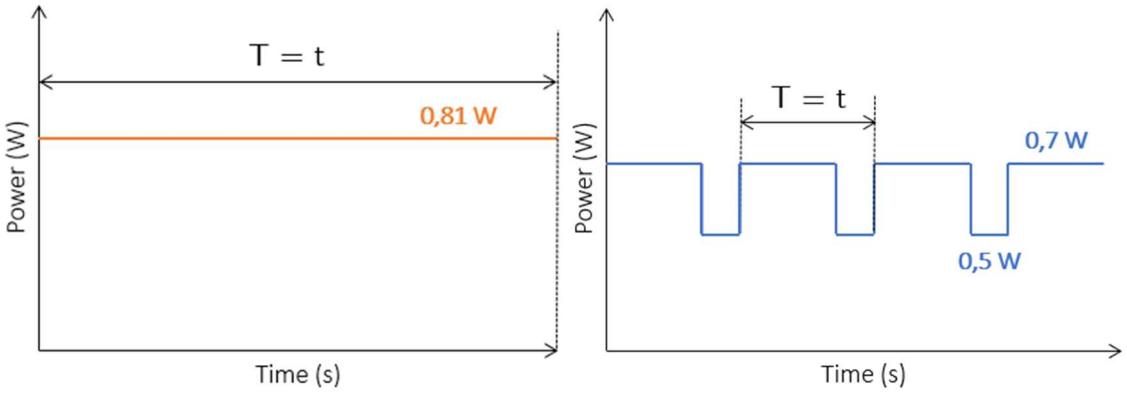


Figura 11 - Andamento della potenza assorbita nel tempo dalla scheda: (a) Arduino Nano 33 BLE Sense, (b) Arduino Nano 33 IoT

Come si nota dai grafici riportati in Figura 11, attualmente il tempo di acquisizione e trasferimento dei dati t risulta pari al tempo di lavoro complessivo T. Infatti, gli andamenti che si sono ottenuti mostrano una richiesta di potenza costante nel tempo e sempre pari a quella massima necessaria per il funzionamento. Per una maggiore durata di utilizzo del dispositivo in autonomia, è però necessario che il rapporto t/T risulti inferiore all'unità. L'attuale domanda energetica del sistema, essendo $t \approx T$, risulta invece piuttosto elevata. Nel futuro, sarebbe quindi opportuno modificare il sistema in maniera tale da ottenere degli andamenti del tipo quelli mostrati in Figura 12. Infatti, per l'Arduino Nano 33 BLE Sense (Figura 12a) sarebbe ottimale ottenere un andamento di potenza con dei massimi pari a

0.81 W ogni n minuti di durata limitata alla durata della connessione Bluetooth tra le due schede Arduino. Per la scheda Arduino Nano 33 IoT, invece, sarebbe adeguato ottenere l'andamento riportato in Figura 12b, con dei massimi pari a 0.7 W di durata pari alla durata di connessione Wi-Fi con il calcolatore e dei massimi pari a 0.5 W per la connessione Bluetooth con l'altra scheda Arduino. Dunque, l'ottimizzazione del sistema richiederebbe una connessione non più costante tra Arduino Nano IoT e calcolatore centrale, ma discontinua ogni N secondi.

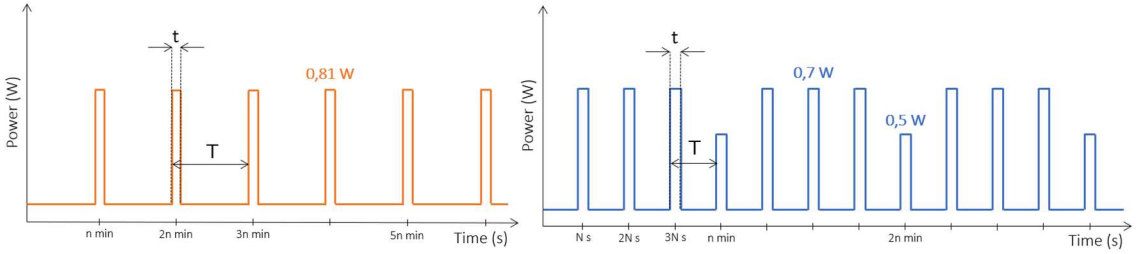


Figura 12 - Andamento desiderato della potenza assorbita nel tempo dalla scheda: (a) Arduino Nano 33 BLE Sense, (b) Arduino Nano 33 IoT

In questo modo, come mostrato in Figura 13, il rapporto t/T si ridurrebbe significativamente, e con esso anche l'energia complessivamente richiesta al dispositivo per il suo funzionamento (che corrisponde all'area sottesa al grafico).

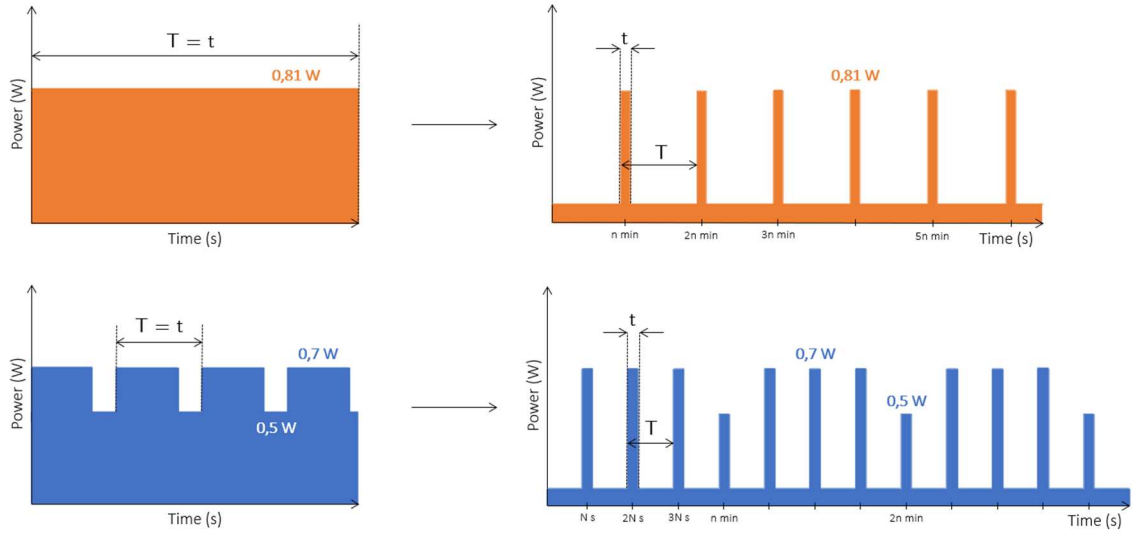


Figura 13 – Andamento attuale e desiderato della potenza assorbita nel tempo dalla scheda: (a) Arduino Nano 33 BLE Sense, (b) Arduino Nano 33 IoT

Infine, per quanto riguarda l'alimentazione del dispositivo indossabile sensorizzato mediante batteria, si procede come segue. Prendendo come riferimento una batteria da 9 V della Duracell, è possibile trovare online la relativa scheda tecnica [35]. Su essa sono riportati dei grafici della durata della batteria in funzione della tensione a corrente, potenza, resistenza o temperatura costanti. Nel caso in esame, si può far riferimento al grafico tensione – durata per una corrente costante, riportato in Figura 14. Per quanto concerne l'Arduino Nano 33 BLE Sense, esso

necessita attualmente, ad una tensione di 4.5 V, 0.18 A per il funzionamento. Dunque, dal grafico seguente, è possibile stimare la durata della batteria di alimentazione di tale scheda: all'incirca quattro ore. La scheda Arduino Nano 33 IoT, invece, non opera a corrente costante, richiedendo a 5 V una corrente di 0.14 A per il collegamento Wi-Fi e una corrente di 0.10 A per il collegamento Bluetooth. Tuttavia, per maggiore semplicità e per effettuare un calcolo a favore della sicurezza, ipotizziamo che la corrente sia costante e pari a quella massima richiesta, 0.14 A. Con tali dati (5 V e 0.14 A) è possibile entrare nel grafico riportato in Figura 14 e stimare la durata della batteria che alimenta l'Arduino IoT, che risulta essere all'incirca pari a sei ore.

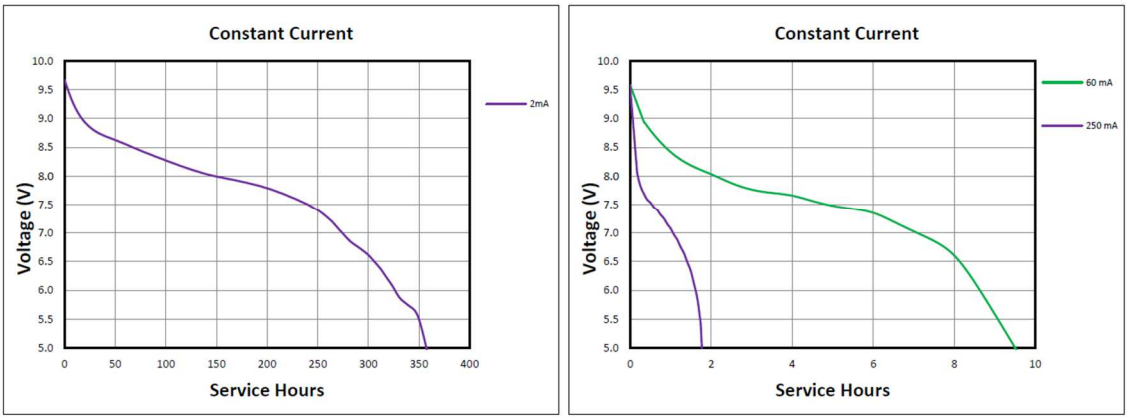


Figura 14 - Tensione che può essere fornita dalla batteria 9V in funzione della durata a corrente costante

Le stime attuali della durata della batteria di alimentazione del dispositivo risultano quindi piuttosto limitate se comparate alla durata di un turno lavorativo giornaliero di 8 ore.

Per un funzionamento più duraturo ed autonomo del casco di protezione sensorizzato, è dunque necessario ricorrere alle modifiche illustrate in precedenza che permettono di ridurre considerevolmente l'energia complessivamente richiesta dal sistema. Inoltre, per l'applicazione in analisi, sarebbe utile l'impiego di batterie ricaricabili, che possano essere dimensionate in funzione del dispositivo da alimentare (in modo da permettere il funzionamento in autonomia per le intere 8 ore lavorative) e che poi vengano ricaricate durante la notte.

Capitolo 2

Analisi multibody

Realizzato il casco sensorizzato IoT, si è proseguito con la realizzazione di un modello multibody con androide per simulazioni in ambiente di lavoro. Come specificato nel Capitolo 1, il dispositivo progettato è in grado di monitorare, oltre alle condizioni ambientali e di salute dell'operatore, il suo stato di attività. Quest'ultimo si può replicare servendosi di software per l'analisi multibody. Realizzando dunque un modello multibody che simuli alcune attività comunemente effettuate sul luogo di lavoro, è possibile valutare anticipatamente, con maggiore semplicità e una minore spesa di tempo e denaro, i valori di accelerazione lineare e velocità angolare che verranno registrati dal dispositivo realizzato durante specifiche operazioni.

Un "simulatore multibody" è un software estremamente utilizzato nell'ingegneria, che permette di simulare sistemi meccanici di vario tipo, quali parti meccaniche rigide o flessibili, parti meccaniche collegate da giunti o che effettuano spostamenti, ecc... In generale, il sistema multibody è composto da una serie di corpi rigidi in movimento relativo tra loro grazie al collegamento effettuato mediante coppie cinematiche o giunti. I simulatori multibody sono in grado, una volta definiti tutti gli elementi che costituiscono il sistema, di analizzarne la cinematica e la dinamica. Per effettuare l'analisi multibody si fa uso del software più conosciuto ed utilizzato al mondo per la dinamica multibody: MSC Adams, e in particolare l'estensione Adams/View.

Prima di poter analizzare la cinematica e la dinamica del sistema multibody, è necessario definire tutti gli elementi che lo costituiscono, ossia:

+ Corpi (link): si tratta generalmente di parti rigide;
+ Vincoli o coppie cinematiche: effettuano il collegamento tra i link, vincolano alcuni gradi di libertà consentendo il moto relativo in una o più direzioni;
+ Forze esterne: si tratta delle forze applicate sul sistema e agenti dall'esterno oppure scambiate con corpi appartenenti all'ambiente circostante (es. suolo).

Come anticipato, è possibile inserire all'interno del modello corpi rigidi o flessibili. Per semplicità di trattazione, tuttavia, si preferisce sempre impiegare corpi rigidi, e simulare quelli flessibili mediante sistemi di corpi rigidi collegati da vincoli, con elasticità e smorzamenti concentrati. Gli elementi necessari alla completa definizione di un corpo rigido del sistema multibody sono:

+ Dimensioni;
+ Massa;
+ Sistema di riferimento solidale al corpo, detto locale, con origine nel suo centro di massa e assi generalmente orientati come i suoi assi principali d'inerzia;
+ Tensore d'inerzia del corpo rispetto al sistema di riferimento locale;
+ Eventuali sistemi di riferimento ausiliari, utili alla definizione dei vincoli o delle forze.

Per quanto riguardano i connettori, invece, questi si occupano di garantire il moto relativo tra due o più corpi in specifiche direzioni. I connettori possono essere di vario tipo:

+ Giunti: tra questi vi sono cerniere, giunti fissi, prismatici, rotoidali, ecc...
+ Primitive;
+ Accoppiamenti;

Per ultime, le forze da integrare nel sistema multibody possono essere:

+ Forza applicate: si tratta di forze concentrate o distribuite, caratterizzate da un punto di applicazione, una direzione e un modulo (che può anche variare nel tempo con una predefinita legge temporale);
+ Connessioni flessibili: tra questi il bushing[2], la molla, la molla torsionale, ecc...
+ Forze speciali: utilizzate per definire le forze di contatto tra corpi, la forza di gravità, le forze modali, ecc..

Infine, per la simulazione di alcuni sistemi meccanici è necessario fornire anche la legge del moto con cui si muovono i corpi. In tal caso, si possono utilizzare moti dei giunti (definiscono il moto relativo dei due o più link collegati dal giunto in esame) o moti generici (direttamente applicati ai corpi, si riferiscono al moto globale dello stesso).

[2] Il bushing è un elemento molto utilizzato perché permette di collegare due link mediante delle forze e momenti. Inoltre, tramite questo elemento è possibile introdurre smorzamento e rigidezza al modello senza iper-vincolarlo.
Nel definire il bushing è necessario stabilire le rigidezze e i coefficienti di smorzamento nelle tre direzioni. Maggiore è la rigidezza in una direzione, più l'elemento sarà rigido in tale direzione, e viceversa.

Nel seguito verrà descritto il modello multibody utilizzato con le sue principali caratteristiche, i movimenti simulati con le relative leggi del moto e il processo di validazione del modello seguito.

2.1 Descrizione del modello

Come modello multibody si è fatto uso di quello precedentemente realizzato da Russo, C., Mocera, F. e Somà, A. per l'analisi del Nordic Walking [36, 37], apportando le dovute modifiche e correzioni. Infatti, anche per l'analisi dell'attività del Nordic Walking era stato necessario realizzare il modello multibody di un umanoide, per cui tale modello è stato poi adattato all'applicazione safety in esame.

È dapprima necessario definire il sistema di riferimento globale utilizzato per l'intera trattazione:

+ X è la direzione frontale, in cui avviene la camminata, con verso positivo quella della camminata stessa;
+ Y è la direzione verticale, con verso positivo verso l'alto;
+ Z è la direzione ortogonale al piano sagittale X – Y, con verso positivo definito dalla regola della mano destra.

Il modello multibody di partenza, riportato in Figura 15, è costituito da soli corpi rigidi, realizzati utilizzando le diverse geometrie di solidi disponibili su Adams/View. In particolare, i corpi rigidi complessivamente impiegati sono venti e includono: capo + collo, braccia, avambracci, mani, bastoncini, core, busto, bacino, cosce, gambe, retropiedi e avampiedi. Tutti i corpi sono caratterizzati da un sistema di riferimento locale centrato nel centro di massa e uno o più sistemi di riferimento addizionali, utili per la successiva definizione dei vincoli e connettori. Infine, di ciascun corpo sono definiti la massa, il tensore d'inerzia e le dimensioni caratterizzanti.

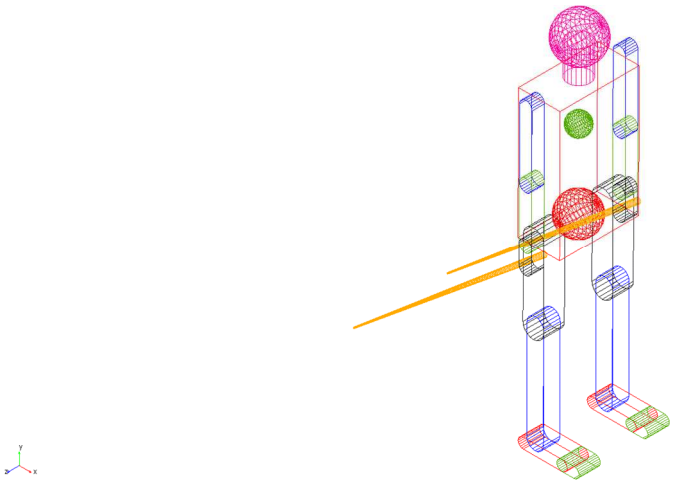


Figura 15 - Modello multibody relativo al Nordic Walking

Per ultimo, è stato inserito il corpo Ground, un parallelepipedo di dimensioni opportune, per permettere la simulazione del terreno durante le attività in analisi e le forze di reazione scambiate con esso.

Una volta definiti i corpi rigidi, è necessario passare all'individuazione dei connettori. Questi servono per vincolare i diversi link in maniera tale da simulare le articolazioni del corpo umano. I connettori sono definiti di seguito, nella Tabella 2.

Tabella 2 – Definizione dei connettori del sistema multibody

Articolazione di riferimento	**Tipologia di giunto**	**Gradi di libertà**
Metatarso	Giunto rotante	1 (rotazione attorno a Z)
Ginocchio	Giunto rotante	1 (rotazione attorno a Z)
Bacino	Giunto sferico	3 (rotazione attorno a X, Y, Z)
Colonna	Giunto sferico	3 (rotazione attorno a X, Y, Z)
Core	Primitiva	
Collo	Giunto fisso	0
Spalla	Giunto sferico	3 (rotazione attorno a X, Y, Z)
Gomito	Giunto rotante	1 (rotazione attorno a Z)

La primitiva utilizzata per vincolare il core impone che esso si muova mantenendo in ogni istante i suoi assi paralleli a quelli del corpo Ground.

Alcuni gradi di libertà delle diverse articolazioni mostrati in Tabella 2, sono poi stati trascurati durante la definizione delle leggi del moto. Infatti, poiché il movimento dell'operatore avviene principalmente sul piano sagittale X – Y, è possibile considerare le sole rotazioni su tale piano, senza commettere errori di approssimazione eccessivamente grossolani.

Con i connettori appena illustrati, ancora non sono state definite tutte le articolazioni necessarie a realizzare un androide in grado di simulare il corpo umano. Questo perché, come anticipato in precedenza, è possibile anche collegare due link mediante forze e momenti tramite i bushing. Si precisano quindi ora quali sono le forze agenti all'interno e sul sistema, in maniera da aver poi completamente definito il sistema multibody in tutti i suoi elementi. Le forze introdotte sono mostrate in Tabella 3.

Tabella 3 - Definizione delle forze del sistema multibody

Articolazione/corpi di riferimento	**Tipo di forza**
Spalla	Molla torsionale (attorno a Z)
Metatarso	Molla torsionale (attorno a Z)
Caviglia	Bushing
Anca	Bushing
Polso	Bushing
Impugnatura	Bushing
Avampiede/terreno	Forza di contatto (impact function model)
Retropiede/terreno	Forza di contatto (impact function model)
Bastoncino/terreno	Forza di contatto (impact function model)

Infine, è necessario menzionare il fatto che l'intero modello risulta parametrico rispetto all'altezza e il peso dell'umanoide. In questo modo, variando massa e/o altezza dell'individuo in analisi, il modello di androide viene scalato proporzionalmente in maniera automatica.

Il modello utilizzato per il Nordic Walking appena descritto necessita di opportune modifiche per poter essere utilizzato per simulare attività tipiche degli ambienti di lavoro. Primariamente, è necessario rimuovere i bastoncini da Nordic Walking. Con Adams/View è quindi sufficiente cancellare i corpi associati. Inoltre, insieme ai link, è anche necessario rimuovere tutte le forze scambiate con i bastoncini, dunque:

+ Rimozione del bushing all'impugnatura, tra bastoncini e mano dell'androide;
+ Rimozione della forza di contatto tra bastoncino e suolo.

Inoltre, a seconda del movimento analizzato sarà necessario apportare ulteriori modifiche locali e specifiche al modello. Le ulteriori variazioni da effettuare coinvolgono le leggi del moto fornite ai diversi connettori, i valori di rigidezze e coefficienti di smorzamento e i parametri utilizzati per il calcolo delle forze di contatto. Tuttavia, questi aspetti verranno trattati nel seguito.

Rimossi i bastoncini e le forze ad essi collegate, dunque, si ottiene il modello multibody di partenza per la simulazione di alcune delle più comuni attività lavorative. Il sistema multibody risultante è mostrato in Figura 16.

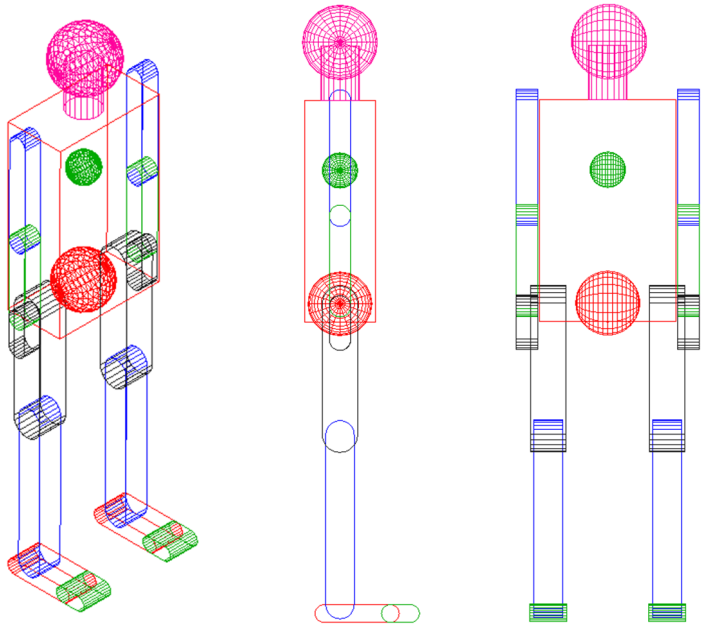


Figura 16 - Modello multibody di partenza

Infine, nell'insieme di immagini riportate nel seguito (Figura 17), sono illustrate:

+ Nell'immagine a sinistra, in arancio, sono riportati tutti i connettori del modello: core, collo, metatarso, gomito, spalla, colonna, bacino, ginocchio;
+ L'immagine centrale rappresenta, mediante le frecce grigie, i moti che verranno imposti ai diversi giunti e connettori;
+ Nell'immagine a destra, infine, si riportano tutte le forze agenti sul sistema multibody: bushing caviglia, bushing anca, bushing polso, molla torsionale metatarso, molla torsionale spalla, contatto avampiede/suolo, contatto retropiede/suolo e forza di gravità.

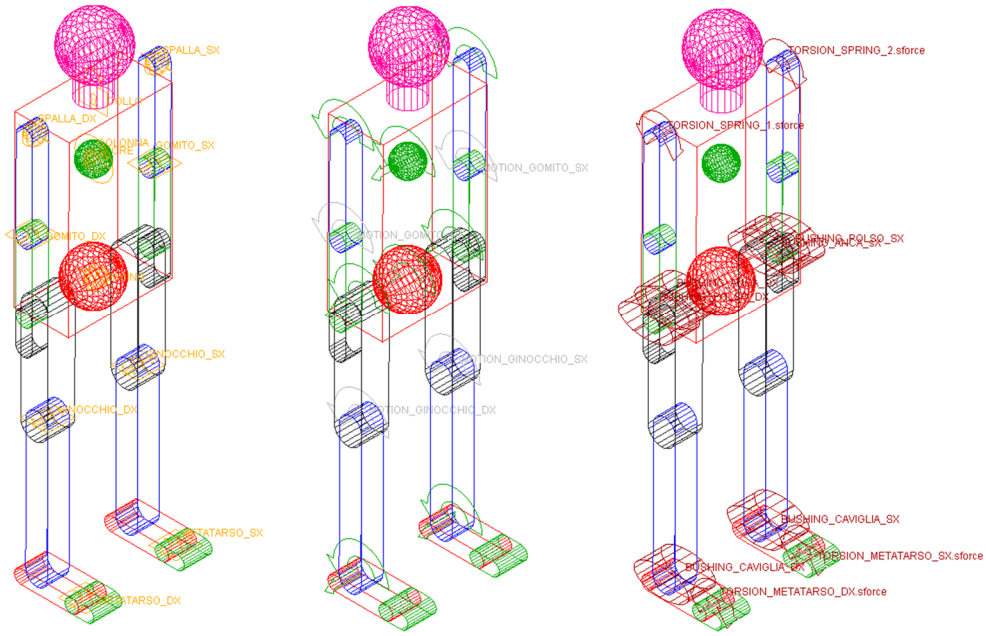


Figura 17 - Focus su connettori, moti imposti e forze del modello multibody di partenza

2.2 Movimenti analizzati

L'obiettivo dell'analisi multibody è quello di simulare alcuni movimenti ed attività tipicamente svolte sul luogo di lavoro. A tal proposito, si è deciso di studiare tre di questi movimenti, ossia:

+ POSA 1: camminata;
+ POSA 2: operatore che si china a terra, a raccogliere qualcosa;
+ POSA 3: operatore che effettua una torsione del busto, per spostare un oggetto da destra a sinistra e viceversa.

Per entrambe le pose 2 e 3 i movimenti si sono principalmente simulati "a vuoto", ossia senza alcun peso afferrato dall'umanoide. Alcune prove sono poi state effettuate inserendo una massa da far movimentare all'operatore, per controllare che il modello risponda nella maniera desiderata.

Per quanto riguarda la posa 2, si è fatta distinzione tra una posa ergonomica e una non. Infatti, nel chinarsi a terra, per ridurre al minimo gli sforzi e i sovraccarichi, l'operatore dovrebbe solamente piegare le gambe, mantenendo sempre il busto in posizione eretta. Tuttavia, la maggior parte degli individui si china piegando principalmente il busto, e dunque sovraccaricando la schiena. Per questo motivo si sono analizzati entrambi i movimenti, che verranno denominati con posa 2a e posa 2b.

Nel seguito si procederà dapprima definendo le leggi del moto da fornire in input ai diversi giunti e poi validando il modello multibody così realizzato.

2.2.1 Modifica del modello necessaria per la Posa 3

Il terzo movimento che si analizza è quello di un operatore che ruota il busto da sinistra a destra, ad esempio spostando oggetti da un piano di lavoro ad un altro. Per la simulazione di tale posa è dunque fondamentale che sia permessa la torsione del busto. Tuttavia, il modello attuale non lo permette: per la posa 3 è necessaria una modifica ulteriore al sistema multibody di partenza ottenuto al paragrafo 2.

Per come definito il modello, l'umanoide è costituito da una serie di corpi rigidi connessi tra loro mediante giunti e bushing. Tuttavia, il busto è realizzato mediante un unico corpo rigido, un parallelepipedo, collegato con le cosce nella porzione inferiore e con il collo e le spalle nella sua parte superiore. Non è dunque permessa la rotazione del busto, poiché esso risulta rigidamente collegato a gambe, braccia e collo nella rotazione attorno alla direzione verticale Y. Per risolvere tale problema, si è pensato di dividere il busto in due corpi rigidi, legati con i medesimi precedenti giunti a collo, braccia e gambe, e uniti tra loro mediante un giunto sferico. Il giunto sferico risulta infatti quello approssimante meglio la reale mobilità del busto negli umani.

Il busto di partenza, dunque, è stato diviso in due parallelepipedi di altezza dimezzata rispetto a quella originale. Di conseguenza, è stato anche necessario dimezzare la massa del busto di partenza e ricalcolare il tensore d'inerzia. Per permettere di replicare i collegamenti originari con collo, spalle e anche, nuovi sistemi di riferimento aggiuntivi sono stati introdotti, facendo attenzione a collocarli nelle posizioni corrette. Infine, dato che il movimento da analizzare coinvolge la sola torsione del busto, si imporrà un moto allo stesso tale da vincolare la rotazione nelle direzioni trasversali e permettere solamente quella attorno alla verticale.

Al termine della modifica, il modello di umanoide risulta corretto come mostrato nella Figura 18, in cui è mostrato il modello multibody di partenza e quello modificato con il busto diviso in due parti. Per ultimo, nell'immagine a destra è riportato il nuovo modello con in grigio i markers necessari alle connessioni delle due parti di busto e in azzurro il giunto sferico introdotto tra le due porzioni di esso.

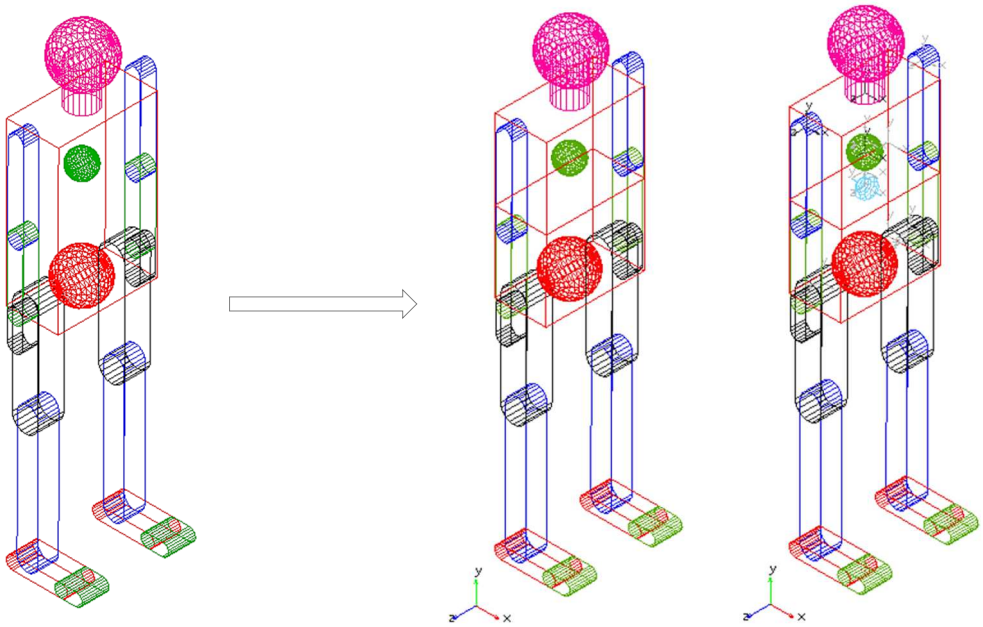


Figura 18 - Modifica apportata al modello multibody per la posa 3

2.3 Definizione delle equazioni del moto

L'obiettivo della analisi multibody è quello di studiare la cinematica e la dinamica del modello. Per le diverse simulazioni, è noto il movimento che si vuole analizzare, mentre le incognite risultano le forze e coppie agenti sul sistema e le velocità e accelerazioni di alcune parti del corpo. Per risolvere il problema si utilizza una logica a dinamica inversa. Si procede dunque imponendo la cinematica al modello, ossia le equazioni del moto che permettono la corretta movimentazione dell'androide. In particolare, le leggi del moto si ottengono dai grafici di variazione

angolare delle articolazioni in funzione del tempo. In questo modo, è poi possibile ricavare le grandezze dinamiche coinvolte e le grandezze cinematiche incognite, che derivano dal movimento dell'umanoide.

Per la determinazione degli andamenti degli angoli delle articolazioni nel tempo, si è fatto uso del software Tecnomatix Jack. Dopodiché, per la definizione delle leggi del moto da fornire in input ai diversi giunti, si impiega MATLAB.

2.3.1 Definizione degli angoli ai giunti mediante Tecnomatix Jack

Tecnomatix Jack è un toolkit ergonomico per la simulazione umana sviluppato dalla Siemens. Il software è finalizzato alla simulazione e modellazione umana per l'analisi ergonomica delle attività effettuate dai lavoratori negli impianti di produzione. Esso permette l'analisi virtuale dei movimenti degli operatori sul luogo del lavoro, in modo da valutare l'ergonomia e la sicurezza delle differenti attività lavorative. Risulta ad esempio possibile valutare una fase di montaggio, includendo nella simulazione tutti gli oggetti che potrebbero interferire con l'operatore, e di conseguenza realizzare un report ergonomico. Una volta realizzata e simulata l'attività in esame sul software, è possibile scaricare:

+ Time report;
+ Video dell'attività simulata;
+ Report temporale degli angoli, coppie e forze medie ai giunti nel tempo;
+ Caricamento cumulativo, confrontato con i limiti ammessi dall'ergonomia;
+ TSB ergonomic report, con indicazione dei tempi stimati per il recovery e la storia temporale dello sforzo muscolare;
+ LBA report, l'analisi della parte bassa della schiena con le relative forze, momenti e tensioni muscolari;
+ Sintesi dell'analisi metabolica, con il costo metabolico totale e la sua suddivisione nelle diverse sotto-attività.

Per la definizione delle leggi del moto delle articolazioni è necessario scaricare il report contenente gli angoli dei giunti nel tempo. Quest'ultimo è un file .csv contenente gli angoli dei differenti giunti nel tempo. È quindi sufficiente elaborare tale file e la variabilità angolare delle articolazioni in funzione del tempo è nota.

Una volta aperta la pagina principale di Tecnomatix Jack ed inserito un umanoide (è possibile selezionare sia uomo che donna), mediante il menù in alto si può aprire la finestra mostrata in Figura 19, che permette di definire nel dettaglio la taglia dell'operatore. È dunque possibile selezionare l'altezza, il peso, e il tipo di proporzionamento da impiegare per la statura e il peso dell'individuo. Una volta impostati tali parametri, si procede aprendo il TSB (Task Simulation Builder – icona cerchiata in Figura 19) di Jack, nel quale si definiscono i movimenti che si vogliono simulare.

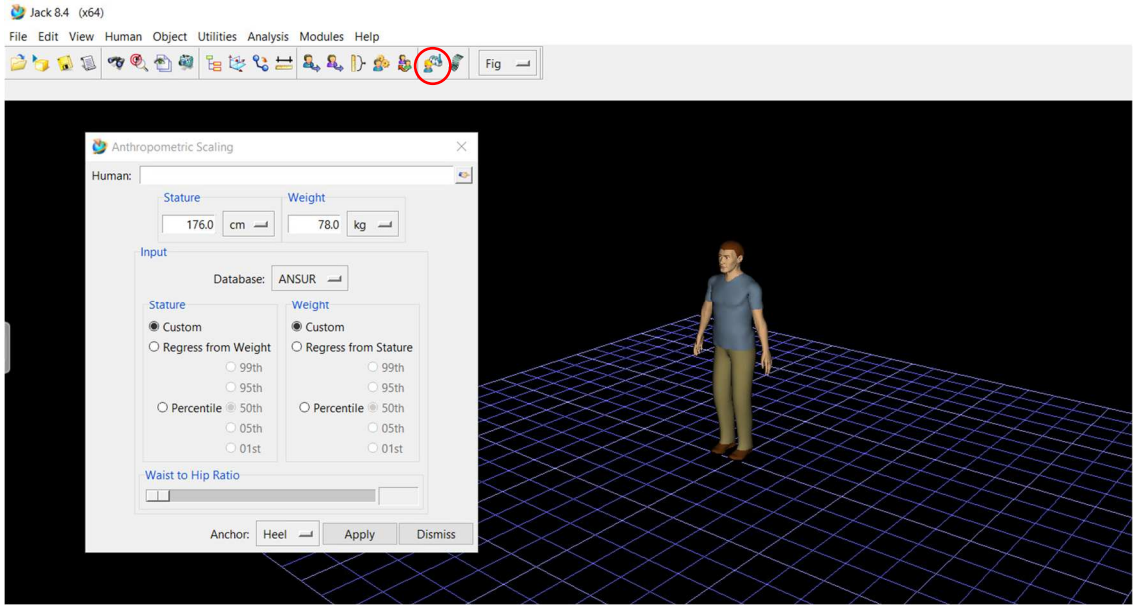


Figura 19 - Tecnomatix Jack: definizione della taglia dell'operatore

Quella riportata di seguito, in Figura 20, è l'interfaccia del TSB di Jack. Come si può notare, sulla sinistra è presente un menù dove selezionare le pose e i movimenti da far effettuare all'umanoide. Una volta scelta l'attività, è necessario selezionare la persona alla quale si vuole far fare il movimento (poiché si può lavorare anche con più operatori contemporaneamente), la durata dell'attività ed ulteriori parametri dipendenti dal tipo di attività. Ad esempio, per la task "go", che simula la camminata, è necessario specificare il luogo dove si vuole far andare l'individuo, dopodiché il software si calcola automaticamente il percorso oppure glielo si fornisce manualmente. Dunque, molti movimenti sono automaticamente realizzati e impostati dal software, è solamente necessario selezionarli.

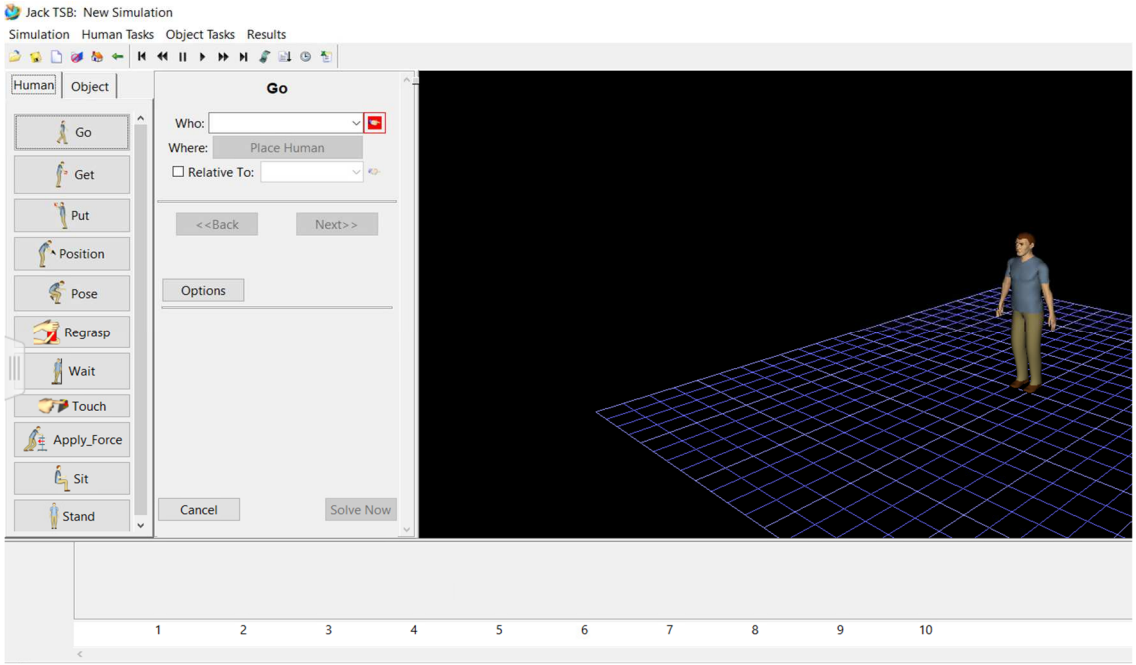


Figura 20 - Tecnomatix Jack: interfaccia del Task Simulation Builder (TSB)

Nella sezione “Posa”, illustrata in Figura 21, invece, è possibile aprire la finestra “set posture” per definire manualmente la postura da far assumere all’umanoide. In questa pagina l’utente può impostare a mano la posa (regolando gli angoli ai giunti – come mostrato in Figura 21) oppure scegliere una posa tra quelle predefinite presenti nel software (cliccando su “predefined postures”). Una volta definita la posa, è necessario impostare la durata temporale richiesta per passare dalla posa precedente (di default la “stand position”, con l’operatore in posizione eretta) a quella appena settata. Dopodiché, è il software che in automatico si calcolerà le pose intermedie da far assumere all’operatore per passare da quella iniziale a quella finale, e di conseguenza realizzerà i report, tra cui quello di nostro interesse che riporta gli andamenti degli angoli ai giunti nel tempo.

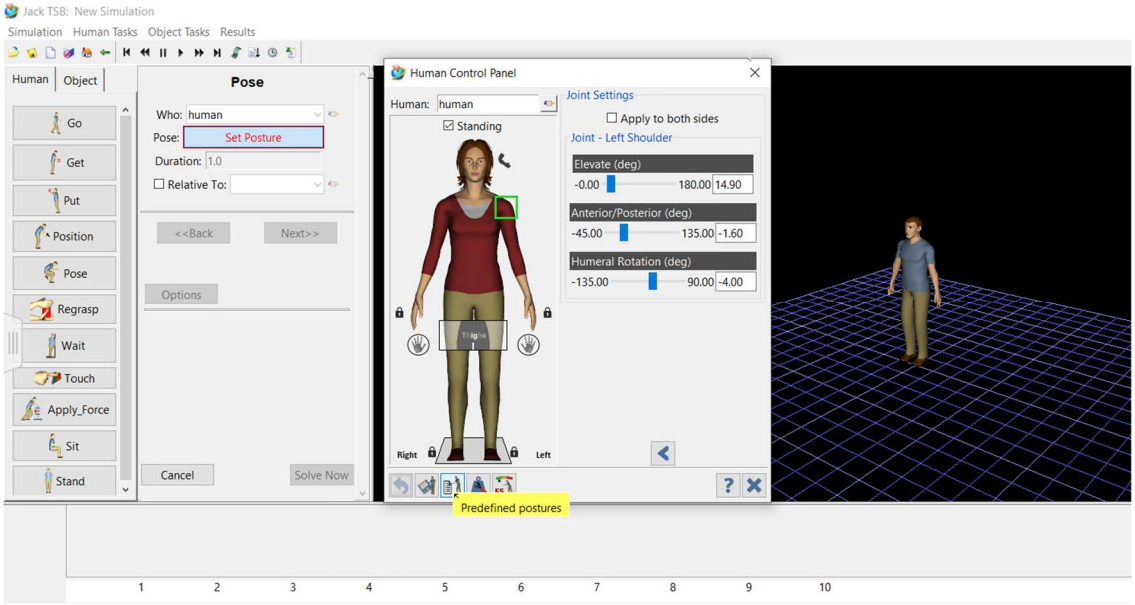


Figura 21 - Tecnomatix Jack: definizione delle pose

È infine possibile far fare più movimenti in sequenza all’umanoide, impostando i tempi di durata di ciascuna attività e dei tempi di pausa tra una attività e la successiva.

Una volta definito l’intero movimento che si vuole analizzare, è possibile scaricare il report TSB di interesse, che riporta il vettore tempo con associata la variazione degli angoli ai diversi giunti. In particolare, vengono considerate: le rotazioni attorno ai tre assi ortogonali per il polso, la spalla e il busto e la rotazione nel piano sagittale per gomito, anca, ginocchio e caviglia.

2.3.2 Definizione delle leggi del moto mediante MATLAB

Una volta ottenute dal software Tecnomatix Jack le variazioni angolari nel tempo di ciascun giunto, è possibile procedere con la definizione delle leggi del moto. A tal fine, per ottenere delle leggi continue e periodiche, si sono interpolati i dati estratti da Jack mediante delle serie di Fourier. Secondo la definizione delle serie

di Fourier, tutte le equazioni del moto hanno la stessa struttura, indipendentemente dal giunto che si sta considerando, ossia:

$$f(time) = a_0 + \sum_{i=1}^{n} a_i \cdot \cos(i \cdot \omega \cdot time) + b_i \cdot \sin(i \cdot \omega \cdot time)$$

Dove i = 0, ..., n è il numero dell'armonica, a_i e b_i sono i coefficienti reali della serie di Fourier e ω è l'armonica fondamentale.

Questo è stato effettuato sul software MATLAB dove, mediante il comando:

```
f = fit ( time, data, 'fourierN' )
```

è possibile ottenere i coefficienti dell'espansione di Fourier di ordine N, a partire dal vettore tempo *time* e dai dati con le variazioni angolari dei giunti *data*.

Dunque, in MATLAB è necessario importare:

+ Il vettore tempo *time*, che viene fornito da Tecnomatix Jack;
+ I vettori degli angoli ai giunti *data*, anch'essi forniti da Jack, che variano con il tempo ed hanno la medesima dimensione del vettore *time*.

Dopodiché, noti i coefficienti dell'espansione di Fourier, risultano anche determinate le leggi del moto da fornire in input alle diverse articolazioni.

Mediante l'utilizzo, congiunto, dei software Tecnomatix Jack e MATLAB, si è dunque in grado di definire le leggi del moto dei giunti da fornire in input al modello multibody. Nel seguito si riportano le equazioni del moto in tal modo ottenute per i tre movimenti analizzati.

2.3.3 Equazioni del moto: Posa 1

Come definito in precedenza, la posa 1 è rappresentata dalla camminata. Si tratta di una delle attività più comuni in ambiente lavorativo, e quindi di notevole interesse. La camminata è uno dei movimenti predefiniti su Tecnomatix Jack, realizzabile mediante il comando "go" esposto in precedenza. È dunque stato sufficiente selezionare l'individuo, il punto di partenza e il punto di arrivo e il software ha automaticamente calcolato i via points realizzando la camminata desiderata. In questo modo, è poi stato possibile esportare l'andamento degli angoli ai giunti nel tempo, utili per ottenere le leggi del moto da fornire in input alle articolazioni su Adams.

Tuttavia, nel caso della camminata il passo temporale con cui sono registrati gli angoli ai giunti da Tecnomatix Jack risulta piuttosto elevato: questo fa sì che gli andamenti ottenuti siano molto disturbati, con numerose discontinuità. È stato quindi dapprima necessario effettuare uno smooth delle curve ottenute da Jack.

Essendo stata, la camminata, ampiamente trattata in letteratura, l'operazione di smooth delle curve è stata effettuata al fine di ottenere degli andamenti di angoli

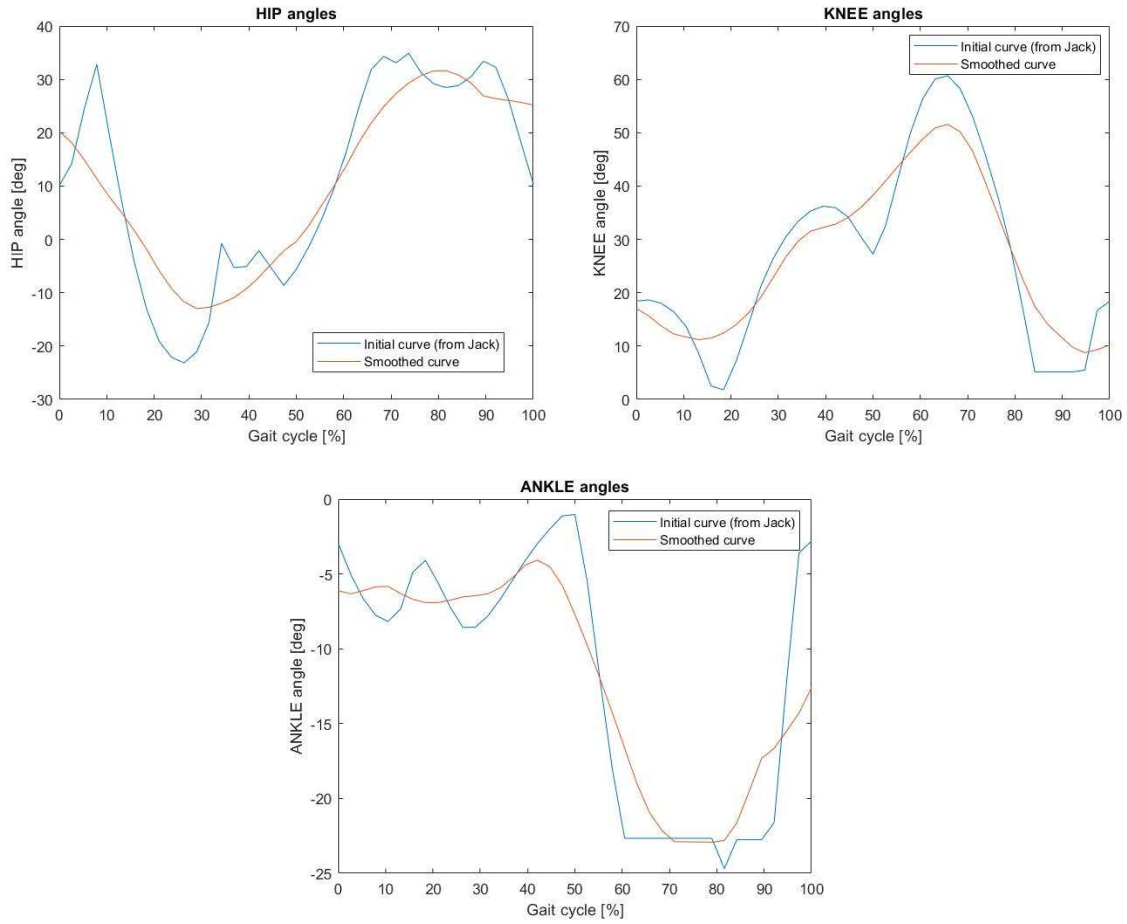
ai giunti nel tempo che si potessero ricondurre a quelli presenti in letteratura [36, 38]. Questa operazione è stata realizzata su MATLAB, importando i dati di Jack ed utilizzando la seguente funzione:

```
smoothdata(function, method, window);
```

Tale funzione permette di effettuare lo smooth della funzione *function*, mediante un predefinito metodo di smoothing ed utilizzando la lunghezza della finestra *window* per il metodo di livellamento. In output viene in questo modo fornita la nuova funzione con la lunghezza della finestra utilizzata. Il numero di finestre utilizzate è stato scelto in modo da avere una buona rappresentazione dei dati presenti in letteratura, in particolare:

+ 10 per l'anca;
+ 9 per il ginocchio, la caviglia, il gomito e la spalla.

Nella Figura 22 si riportano le funzioni così ottenute, in confronto con la funzione di partenza ricavata da Jack. In particolare, in blu è riportata la curva di partenza, esportata da Jack, mentre in rosso è rappresentata la curva "smoothed", ottenuta mediante la funzione MATLAB dinanzi illustrata.



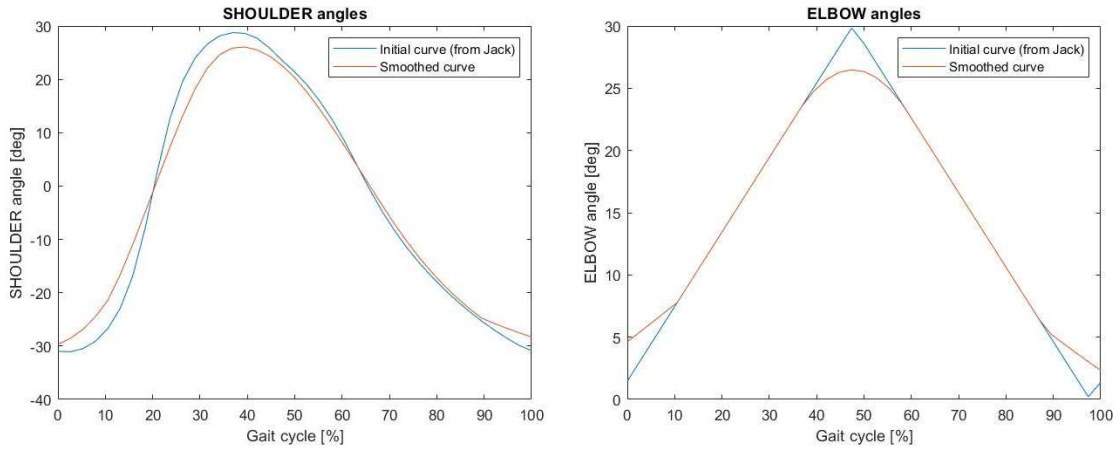


Figura 22 - Smooth dei dati di Jack per la posa 1

Le leggi del moto devono quindi essere fornite in input ai giunti di: anca, ginocchio, caviglia, spalla e gomito. Per quanto riguarda la spalla, si fa riferimento alla sola rotazione sul piano sagittale, coinvolgendo la camminata principalmente movimenti su tale piano. Come si può notare, specialmente per i giunti delle gambe, la curva smooth presenta molte meno discontinuità rispetto alla curva iniziale, caratterizzata invece da numerose irregolarità e cambi repentini di andamento. Una volta ottenute queste curve sulla base di quelle riportate in letteratura, è stato possibile procedere e ricavare le leggi del moto per i giunti.

Come verrà effettuato anche per i movimenti seguenti, per ottenere le leggi del moto si interpolano gli andamenti degli angoli ai giunti nel tempo mediante delle serie di Fourier. Infatti, con questo tipo di funzioni è possibile gestire il grado della serie in funzione della precisione di approssimazione della curva. Presa la forma generale di una serie di Fourier nel tempo:

$$f(time) = a_0 + \sum_{i=1}^{n} a_i \cdot \cos(i \cdot \omega \cdot time) + b_i \cdot \sin(i \cdot \omega \cdot time)$$

Nel caso della camminata è conveniente riscrivere tale equazione nella seguente forma:

$$f(time) = a_0 + \sum_{i=1}^{n} a_i \cdot \cos\left((time + K) \cdot \frac{2\pi}{\nu} + sfas\right) + b_i \cdot \sin\left((time + K) \cdot \frac{2\pi}{\nu} + sfas\right)$$

Dove:

$$\nu = \frac{velocità\ di\ camminata}{2 \cdot ampiezza\ del\ passo}$$

Inoltre: $\frac{2\pi}{\nu}$ è il periodo della funzione, parametrizzato con il ciclo del cammino, $time + K$ è lo sfasamento temporale delle leggi cinematiche dei giunti (ogni giunto ha un proprio sfasamento), $K = (\%cycle \cdot \nu/100)$ è la percentuale di ciclo di

camminata e $sfas$ è uno shift della curva, utile per spostare l'inizio della camminata contemporaneamente in tutti i giunti.

In questo modo, mediante il comando "fit" di MATLAB, è possibile ottenere i coefficienti a_i e b_i delle espansioni di Fourier dei giunti di anca, ginocchio, caviglia, spalla e gomito, riportati in Tabella 4.

Tabella 4 - Coefficienti di Fourier per posa 1

	ANCA	GINOCCHIO	CAVIGLIA	SPALLA	GOMITO
a_0	11.58	25.18	-11.59	-3.956	13.31
a_1	19.63	-2.863	-7.322	-2.831	-9.693
b_1	8.029	-17.25	4.304	-27.3	-6.736
a_2	-1.241	-6.702	-1.184	-1.783	0.1273
b_2	-3.166	-0.628	3.539	4.478	1.056
a_3	1.216	-4.853	0.5857	0.14	-0.09585
b_3	-1.092	-0.3745	-0.5242	-1.308	-0.3184
a_4	-0.02398	0.5845	-0.1881		
b_4	-0.5633	0.6761	-0.2777		
a_5	0.3922	0.3616	0.2659		
b_5	0.3756	-0.05148	0.09213		
a_6		0.4674	0.1487		
b_6		0.3043	0.1201		
a_7		-0.1204	0.06517		
b_7		-0.1629	0.4378		

Inoltre, per il busto, si è impostata una rotazione attorno a Z (direzione ortogonale al piano sagittale che descrive la camminata), costante e pari a – 0.064 radianti. Infatti, durante il cammino, il busto rimane pressoché fermo ma leggermente ruotato in avanti per permettere l'equilibrio dell'operatore.

Nella pagina seguente sono rappresentate tali serie Fourier approssimanti le curve iniziali. I punti blu sono quelli appartenenti alle curve smooth ricavate in precedenza, mentre le curve rosse sono le serie di Fourier utilizzate per interpolare tali punti. Come si può notare anche dalla tabella precedente, si sono impiegati gradi differenti per le serie di Fourier dei diversi giunti: questo è stato fatto con il fine di ottenere sempre una buona approssimazione ed interpolazione dei dati di partenza mantenendo al minimo il costo computazionale.

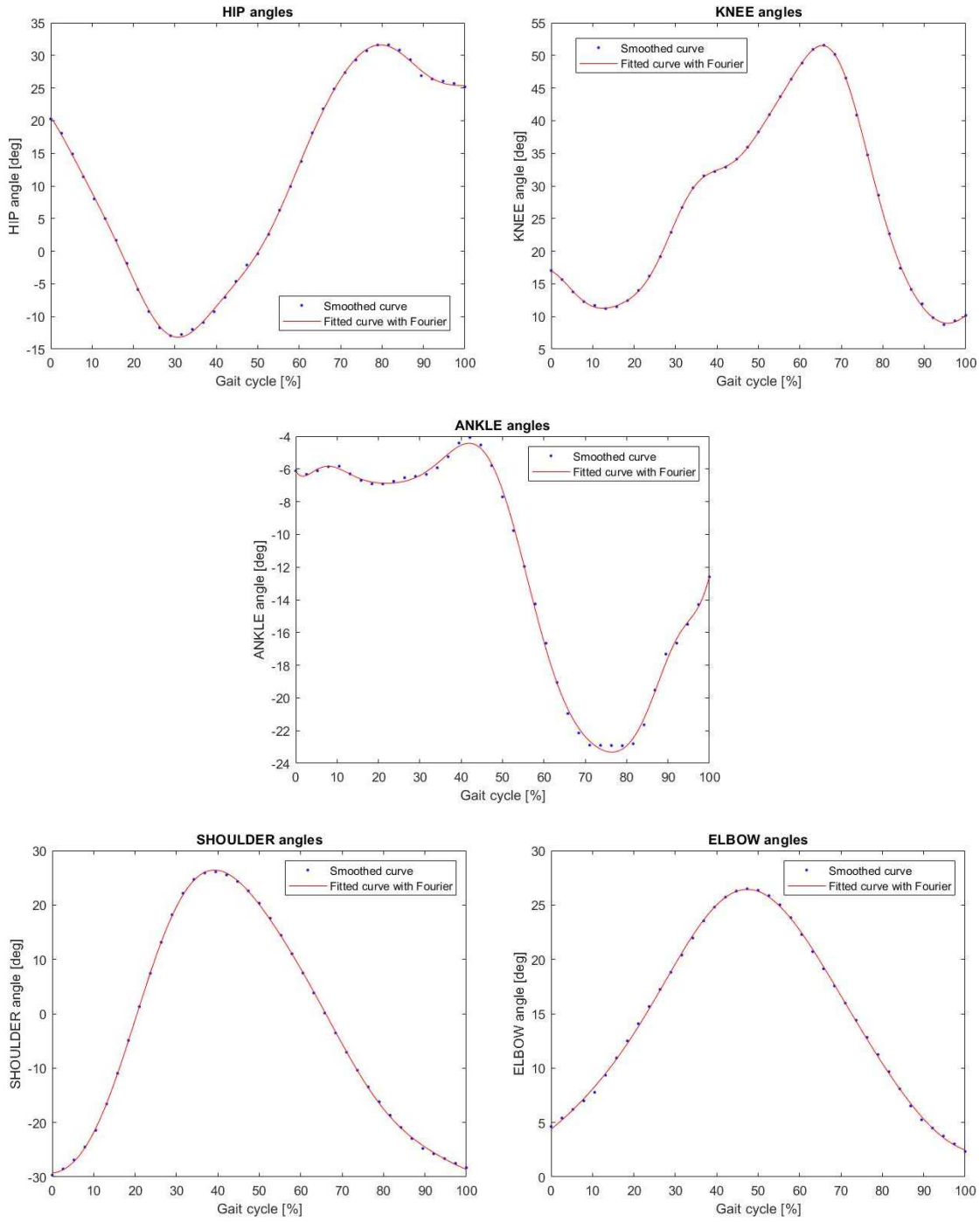


Figura 23 - Serie di Fourier utilizzate per interpolare gli andamenti angolari dei giunti per la posa 1

Una volta ottenute le serie di Fourier, queste sono state fornite come leggi del moto ai diversi giunti. È però stato dapprima necessario apportare una modifica: le leggi così costruite esprimono gli angoli in gradi, mentre su Adams/View è necessario esprimerle in radianti. Anziché la funzione f(time), si è dunque fornita al software la funzione g(time), così ottenuta:

$$g(time) = \frac{\pi}{180} \cdot f(time)$$

Ad esempio, la legge del moto per l'anca destra è la seguente:

$$g_{ANCA\_DX}(time) = -\frac{\pi}{180} \cdot (11.58 + 19.63 \cdot \cos((\text{time} + \text{adx}) \cdot (2\pi/\text{v}) + \text{sfas}) + 8.029 \cdot \sin((\text{time} + \text{adx}) \cdot (2\pi/\text{v}) + \text{sfas}) - 1.241 \cdot \cos(2 \cdot (\text{time} + \text{adx}) \cdot (2\pi/\text{v}) + \text{sfas}) - 3.166 \cdot \sin(2 \cdot (\text{time} + \text{adx}) \cdot (2\pi/\text{v}) + \text{sfas}) + 1.216 \cdot \cos(3 \cdot (\text{time} + \text{adx}) \cdot (2\pi/\text{v}) + \text{sfas}) - 1.092 \cdot \sin(3 \cdot (\text{time} + \text{adx}) \cdot (2\pi/\text{v}) + \text{sfas}) - 0.02398 \cdot \cos(4 \cdot (\text{time} + \text{adx}) \cdot (2\pi/\text{v}) + \text{sfas}) - 0.5633 \cdot \sin(4 \cdot (\text{time} + \text{adx}) \cdot (2\pi/\text{v}) + \text{sfas}) + 0.3922 \cdot \cos(5 \cdot (\text{time} + \text{adx}) \cdot (2\pi/\text{v}) + \text{sfas}) + 0.3756 \cdot \sin(5 \cdot (\text{time} + \text{adx}) \cdot (2\pi/\text{v}) + \text{sfas}))$$

E in maniera analoga si ottengono tutte le altre leggi del moto. I coefficienti K, che esprimono la percentuale di ciclo di camminata, utili per lo sfasamento temporale delle leggi cinematiche dei diversi giunti, risultano:

+ $gomdx = spdx = asx = 28 \cdot v/100$;
+ $gomsx = spsx = adx = 78 \cdot v/100$;
+ $cdx = 95 \cdot v/100$;
+ $csx = 45 \cdot v/100$;
+ $gdx = 75 \cdot v/100$;
+ $gsx = 25 \cdot v/100$.

Dove gomdx e gomsx si riferiscono al gomito, spdx e spsx alla spalla, adx e asx all'anca, cdx e csx alla caviglia e gdx e gsx al ginocchio. Inoltre, si fa distinzione tra i coefficienti relativi alla parte destra (dx) e quelli riferiti alla parte sinistra (sx) dell'umanoide.

Definite le leggi del moto per i giunti del sistema multibody, risulta definita la cinematica del modello: è quindi già possibile visualizzare il movimento in analisi nell'ambiente multibody. A tal proposito, si riportano in Figura 24 e Figura 25 le pose successivamente assunte dall'operatore durante due passi di camminata.

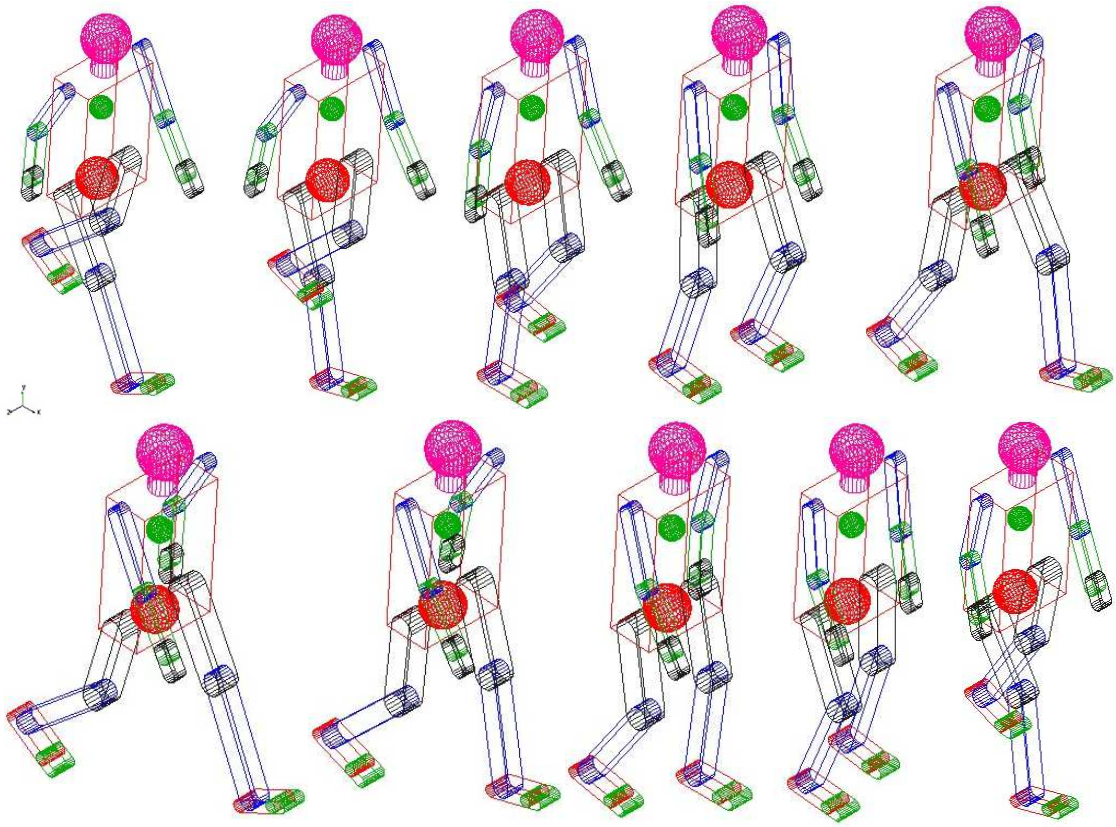


Figura 24 - Fasi della camminata - vista assonometrica

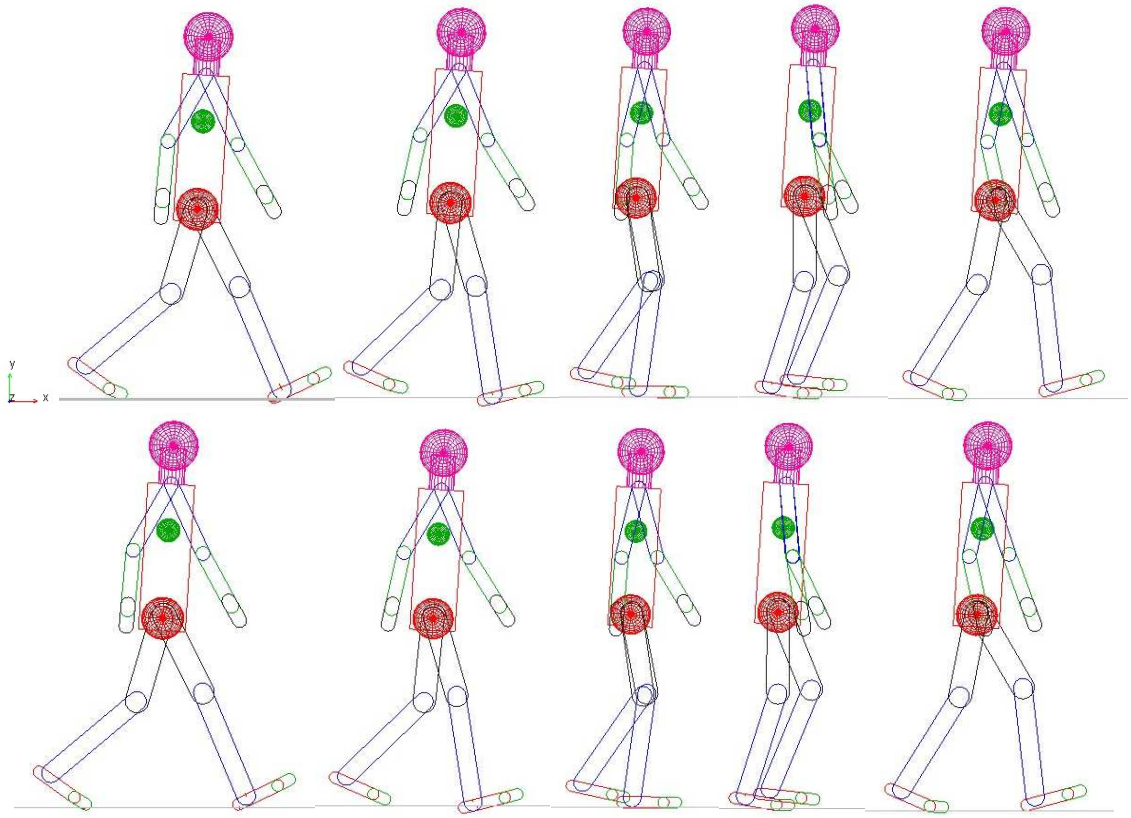


Figura 25 - Fasi della camminata - vista laterale

2.3.4 Equazioni del moto: Posa 2a

Il secondo movimento che si simula è quello di un operatore che si china verso terra, a prendere un oggetto. Si tratta, anche in questo caso, di un movimento che viene effettuato quotidianamente sul luogo di lavoro. Tuttavia, la maggior parte degli individui si china piegando solamente la schiena, il che risulta poco ergonomico. Per questo si fa differenza tra la posa 2a (l'operatore si china verso terra in maniera ergonomica, effettuando un importante piegamento sulle gambe e di conseguenza un minimo sforzo della schiena) e la posa 2b (l'operatore si china verso terra piegando principalmente il busto e minimamente le gambe).

Nel caso della posa 2a, l'operatore si china a terra facendo un piegamento sulle gambe e mantenendo il busto in posizione tale da garantire l'equilibrio. Come per gli altri movimenti, si è dapprima simulata la posa mediante Tecnomatix Jack, in modo da determinare la variazione degli angoli dei giunti nel tempo. Rispettando la posa in analisi le regole dell'ergonomia, essa si trova, sul software, all'interno delle pose predefinite. È quindi semplicemente necessario selezionare tale movimento dal database delle pose, definire la durata temporale, impostare come posa iniziale e finale quella eretta (posa "stand", anch'essa predefinita su Jack), ed effettuare la simulazione. Dal report fornito in output risultano così noti i valori degli angoli ai giunti nel tempo, utili per la determinazione delle leggi del moto.

Effettuata la simulazione su Tecnomatix Jack, è poi necessario passare all'analisi del file .csv fornito in output dal software e contenente i dati delle variazioni angolari dei giunti nel tempo. Questo viene effettuato su MATLAB, che permette di interpolare tali dati mediante delle serie di Fourier (che verranno poi date come equazioni del moto ai diversi connettori su Adams/View). Rispetto alla posa 1, non è ora più necessario effettuare lo smooth dei dati, poiché gli andamenti degli angoli nel tempo risultano già adeguati alla successiva interpolazione con serie di Fourier.

Nell'immagine seguente, Figura 26, sono riportati gli andamenti degli angoli dei giunti nel tempo. In particolare, i punti blu sono i dati esportati da Jack, mentre la curva rossa rappresenta la serie di Fourier con cui si sono interpolati tali dati su MATLAB. A seconda del giunto in analisi, è stato necessario impiegare delle serie di Fourier di grado differente, in modo da ottenere sempre una buona interpolazione dei valori in output da Tecnomatix Jack.

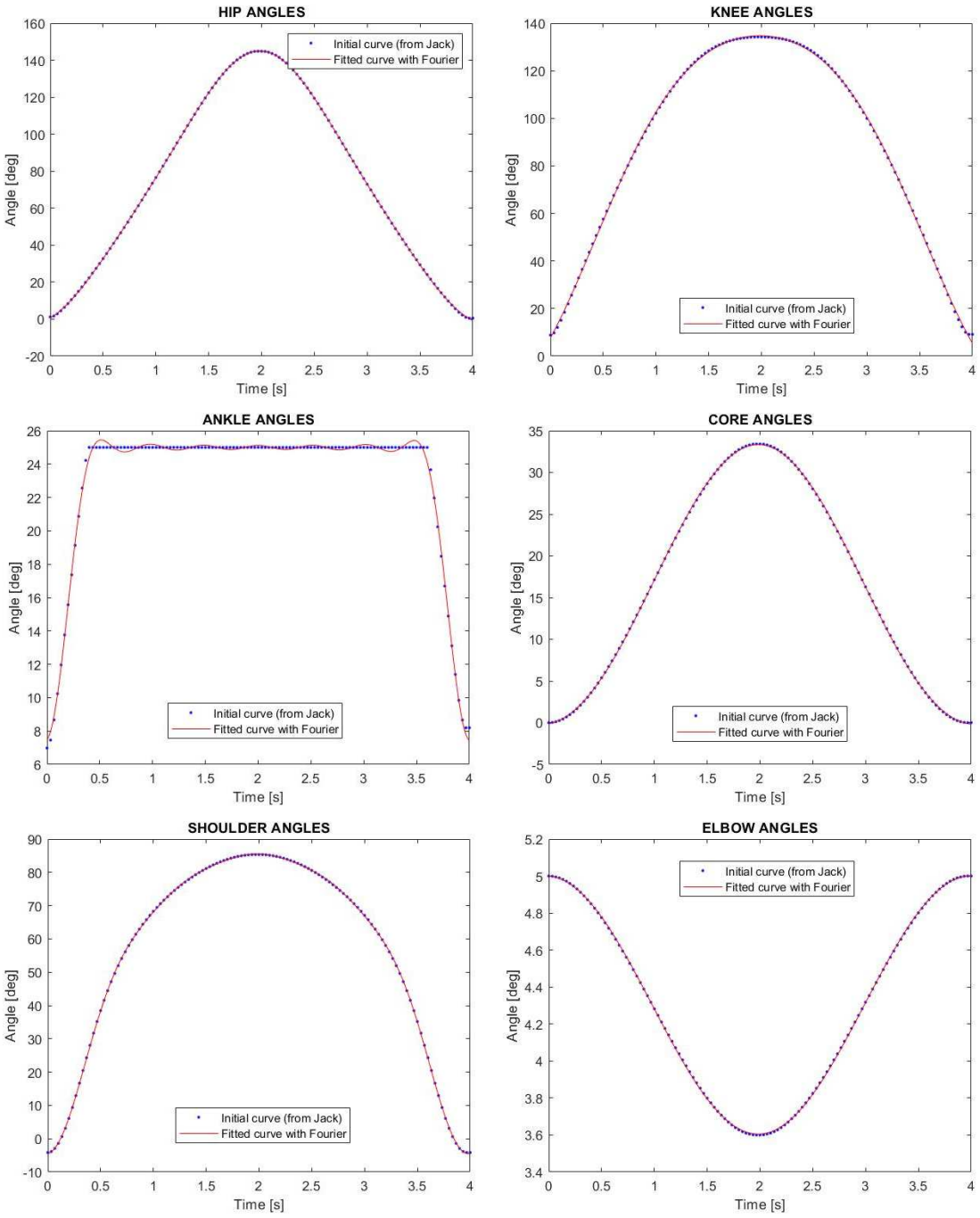


Figura 26 - Serie di Fourier utilizzate per interpolare gli andamenti angolari dei giunti per la posa 2a

Come si può notare dai grafici precedenti, le serie di Fourier selezionate approssimano in maniera accurata i dati ottenuti da Jack. Solamente per la caviglia si ha una accuratezza minore: tuttavia, anche con serie di grado ulteriormente maggiore la precisione non migliora sensibilmente, per cui si è deciso di fermarsi all'ordine 7 per minimizzare il costo computazionale.

Le leggi del moto sono quindi state estrapolate per i giunti di anca, ginocchio, caviglia, gomito e spalla. Inoltre, rispetto alla camminata, l'operazione che si sta analizzando coinvolge anche un movimento nel tempo del busto. È quindi

necessario fornire al busto (vincolo *core* su Adams) una legge di variazione degli angoli nel tempo.

Preso il modello generale di una serie di Fourier nel tempo:

$$f(time) = a_0 + \sum_{i=1}^{n} a_i \cdot \cos(i \cdot \omega \cdot time) + b_i \cdot \sin(i \cdot \omega \cdot time)$$

Per i diversi giunti, i coefficienti a_i e b_i impiegati, con i relativi pesi ω, sono riportati nella Tabella 5.

Tabella 5 - Coefficienti di Fourier per posa 2a

	ANCA	GINOCCHIO	CAVIGLIA	BUSTO	SPALLA	GOMITO
a_0	68.95	69.8	22.99	12.46	56.46	4.303
a_1	-68.34	-59.53	-3.882	-1.2	-38.05	0.697
b_1	19.61	45.82	0.1127	9.006	0.8339	-0.02596
a_2	1.92	-2.612	-3.5	-11.4	-13.49	
b_2	-1.017	9.834	0.2031	-3.061	0.5919	
a_3	-1.895		-2.928		-6.153	
b_3	2.101		0.2546		0.4054	
a_4			-2.251		-2.417	
b_4			0.2599		0.2128	
a_5			-1.561		-0.7868	
b_5			0.2229		0.08712	
a_6			-0.9405			
b_6			0.1567			
a_7			-0.4502			
b_7			0.07955			
ω	1.445	1.252	1.56	0.8582	1.573	1.565

Le leggi del moto f(t), così ottenute, esprimono l'andamento degli angoli nel tempo in gradi. Per poterle utilizzare su Adams/View, a partire dalla f(t) degli angoli nel tempo in gradi, è dunque essenziale dapprima ottenere la funzione g(t) degli angoli nel tempo in radianti, con la seguente formula:

$$g(time) = \frac{\pi}{180} \cdot f(time)$$

È poi la funzione g(t) ad essere inserita in Adams come equazione del moto dei diversi giunti.

Ad esempio, la legge del moto per l'anca è la seguente:

$$g_{ANCA}(time) = -\frac{\pi}{180} \cdot \Big(68.95 - 68.34 \cdot cos(\omega_{ANCA} \cdot time) + 19.61 \cdot sin(\omega_{ANCA} \cdot time) + 1.92 \cdot cos\big(2 \cdot (\omega_{ANCA} \cdot time)\big) - 1.017 \cdot sin\big(2 \cdot (\omega_{ANCA} \cdot time)\big) - 1.895 \cdot cos\big(3 \cdot (\omega_{ANCA} \cdot time)\big) + 2.101 \cdot sin\big(3 \cdot (\omega_{ANCA} \cdot time)\big)\Big)$$

E in maniera analoga si ottengono tutte le altre. Ovviamente, a differenza della posa 1, ora le equazioni per la porzione destra e sinistra dell'umanoide sono analoghe.

Definite le equazioni del moto da fornire in input ai diversi giunti del modello, è già possibile visualizzare l'umanoide che si muove all'interno dell'ambiente multibody. In particolare, nelle Figura 27 e Figura 28, si riportano istanti successivi del movimento che permette di realizzare la posa 2a.

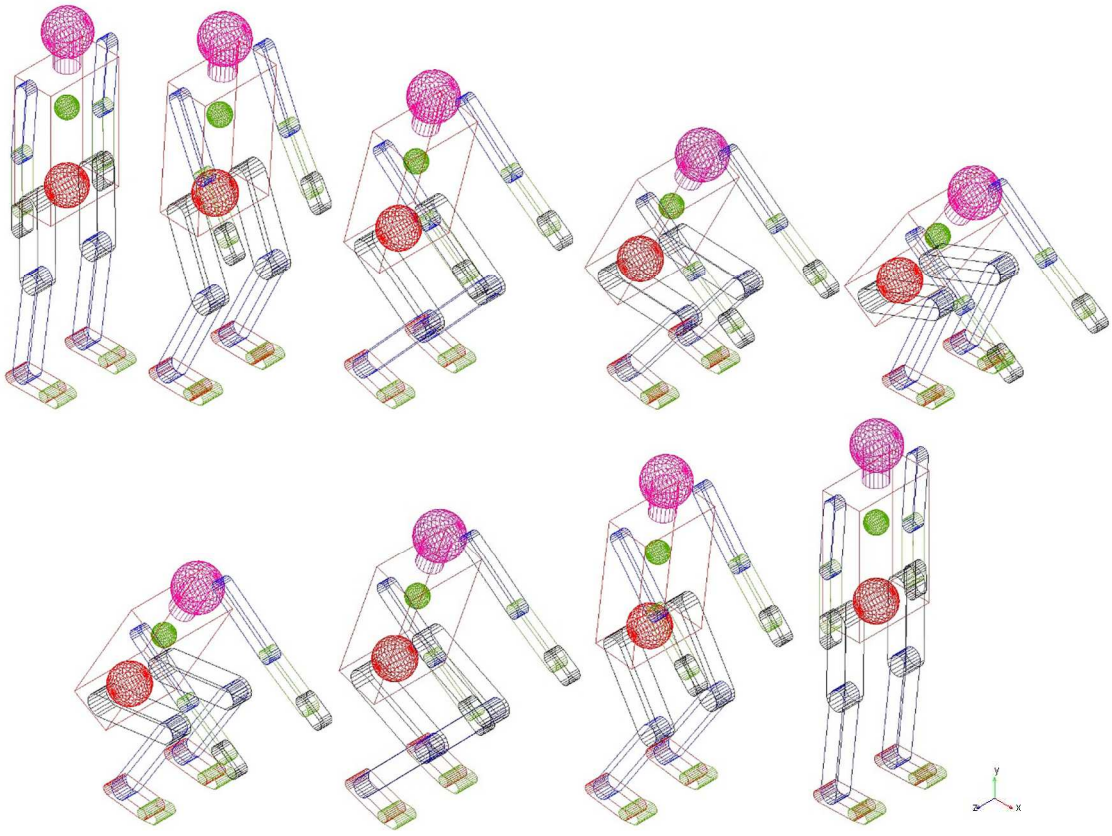


Figura 27 - Istanti della posa 2a - vista assonometrica

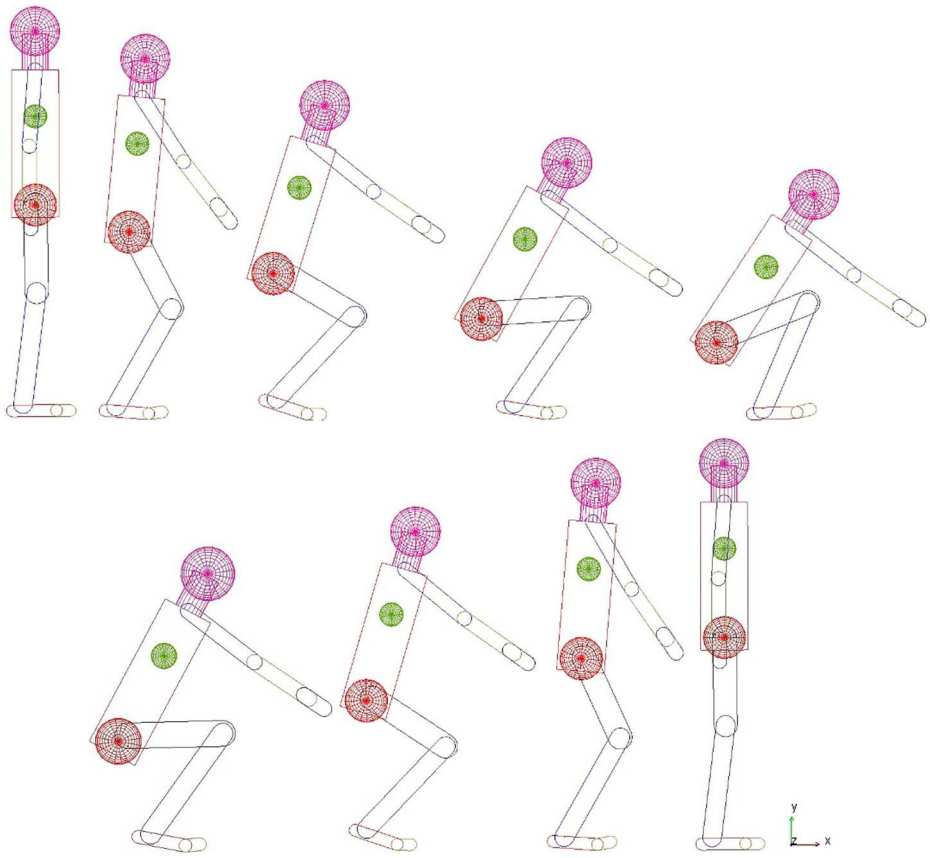


Figura 28 - Istanti della posa 2a - vista laterale

2.3.5 Equazioni del moto: Posa 2b

La posa 2b riguarda un movimento meno ergonomico, con l'operatore che si china verso terra facendo praticamente tutto lo sforzo richiesto con la schiena. In questo caso la posa, non rispettando le regole dell'ergonomia, è stata settata a mano su Tecnomatix Jack, poiché non presente nel database delle pose predefinite. È quindi stato necessario utilizzare la finestra illustrata in Figura 21 - Tecnomatix Jack: definizione delle pose, in cui a mano si definiscono gli angoli ai giunti che meglio rappresentano la posa desiderata. Dopodiché, definita la posa da far assumere all'operatore, le pose iniziale e finale (semplice postura eretta) e la durata del movimento, il software si è calcolato la variazione angolare dei giunti nel tempo. Tali dati sono successivamente stati elaborati su MATLAB, in modo da ottenere le serie di Fourier che meglio li approssimano.

In maniera analoga al caso 2a, le leggi del moto sono state ottenute per i giunti di anca, ginocchio, caviglia, gomito e spalla e per il busto (core). Preso il modello generale di una serie di Fourier nel tempo:

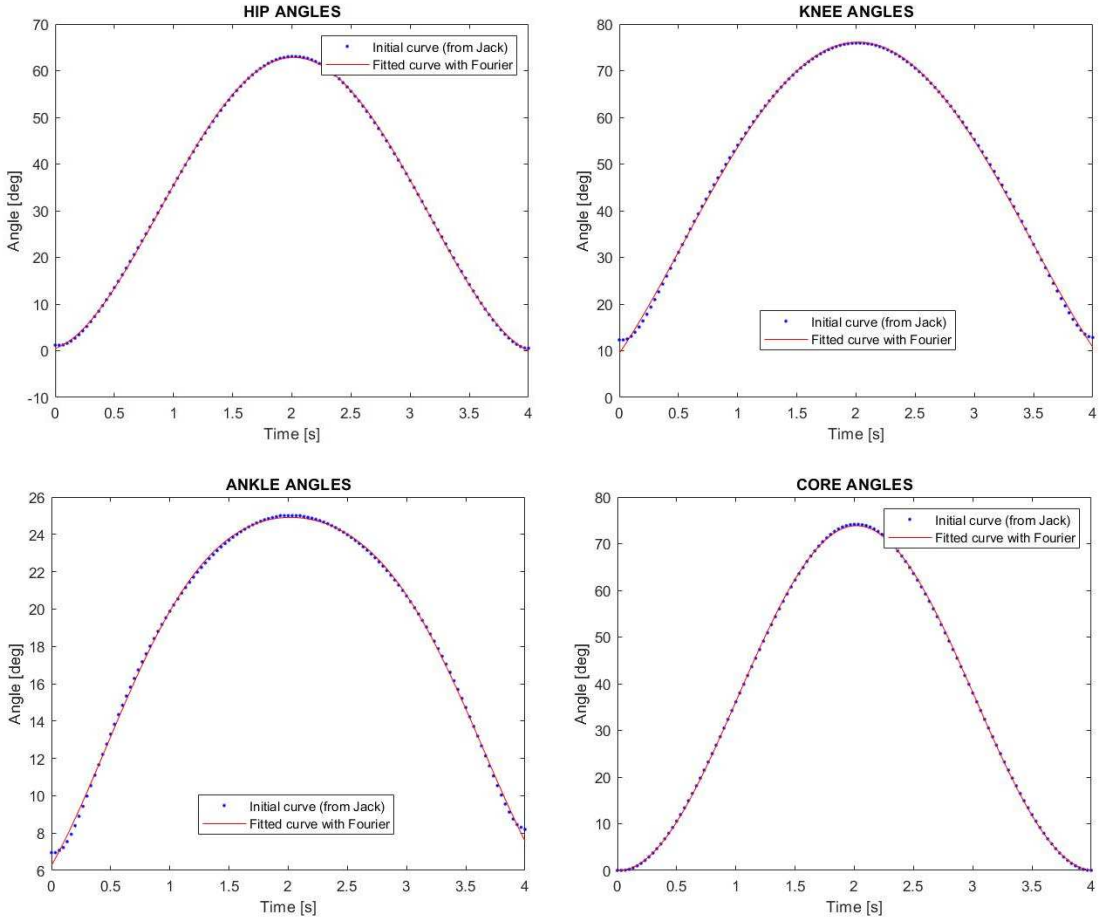
$$f(time) = a_0 + \sum_{i=1}^{n} a_i \cdot \cos(i \cdot \omega \cdot time) + b_i \cdot \sin(i \cdot \omega \cdot time)$$

Per i diversi giunti i coefficienti a_i e b_i impiegati e i relativi pesi ω forniti dalla funzione "fit" di MATLAB sono riportati nella Tabella 6.

Tabella 6 - Coefficienti di Fourier per posa 2b

	ANCA	GINOCCHIO	CAVIGLIA	BUSTO	SPALLA	GOMITO
a_0	33.12	30.94	15.82	36.92	47.35	2.509
a_1	-30.76	-22.17	-8.775	-36.92	-41.71	2.49
b_1	2.725	11.82	5.834	-0.5283	0.5813	-0.03569
a_2	-1.354	-1.192	-0.6632		-8.408	
b_2	0.3778	1.671	1.304		0.2344	
a_3	-0.5016				-1.98	
b_3	0.1857				0.0828	
ω	1.519	1.315	1.248	1.565	1.551	1.565

Nella figura seguente, Figura 29, si riporta invece l'andamento nel tempo degli angoli ai giunti ricavati da Jack (i punti blu) e le relative serie di Fourier interpolanti (le curve rosse). Come si può notare, mediante delle serie di Fourier di ordine massimo pari a tre è stato possibile ottenere una funzione interpolante i dati di partenza con una buona approssimazione.



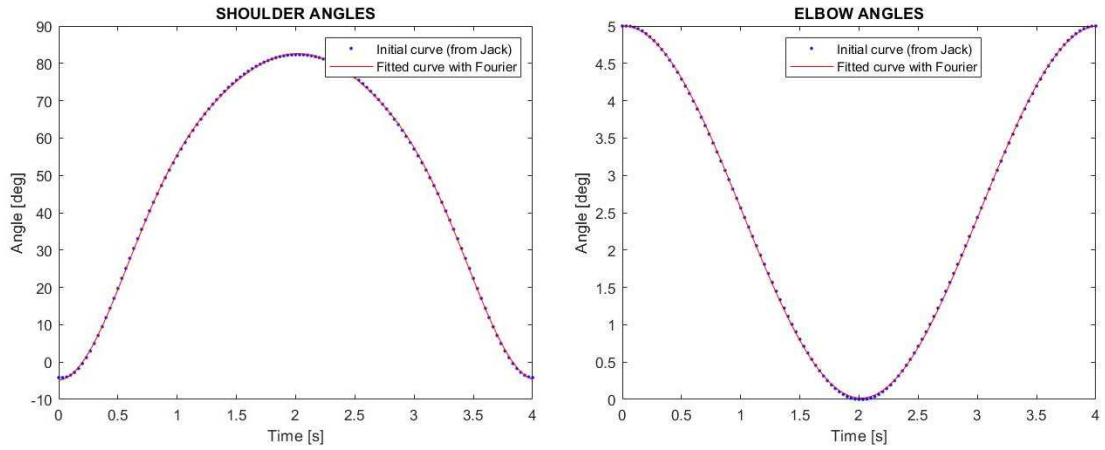


Figura 29 - Serie di Fourier utilizzate per interpolare gli andamenti angolari dei giunti per la posa 2b

È a tal punto nuovamente necessario passare dalla funzione f(t) che esprime gli angoli in gradi alla funzione g(t), relativa agli angoli in radianti:

$$g(time) = \frac{\pi}{180} \cdot f(time)$$

Come input di legge di moto ai giunti si utilizza quindi g(t) all'interno di Adams/View.

Ad esempio, la legge del moto per l'anca (uguale per la parte destra e sinistra del corpo dell'androide) è la seguente:

$$\begin{aligned} g_{ANCA}(time) = \frac{\pi}{180} \cdot 2 \\ \cdot \Big(33.12 - 30.76 \cdot cos(\omega_{ANCA} \cdot time) + 2.725 \cdot sin(\omega_{ANCA} \cdot time) \\ - 1.354 \cdot cos\big(2 \cdot (\omega_{ANCA} \cdot time)\big) + 0.3778 \cdot sin\big(2 \cdot (\omega_{ANCA} \cdot time)\big) \\ - 0.5016 \cdot cos\big(3 \cdot (\omega_{ANCA} \cdot time)\big) + 1.671 \cdot sin\big(3 \cdot (\omega_{ANCA} \cdot time)\big)\Big) \end{aligned}$$

E in maniera analoga si ottengono le equazioni del moto degli altri giunti.

Definite le leggi del moto da fornire in input ai diversi giunti su Adams/View, risulta completamente definito il movimento che si desidera simulare. È quindi già possibile effettuare una simulazione per la visualizzazione della posa, con l'operatore che si china verso terra con un limitato piegamento delle gambe. Alcuni istanti caratteristici di tale movimento sono riportati in Figura 30 e Figura 31.

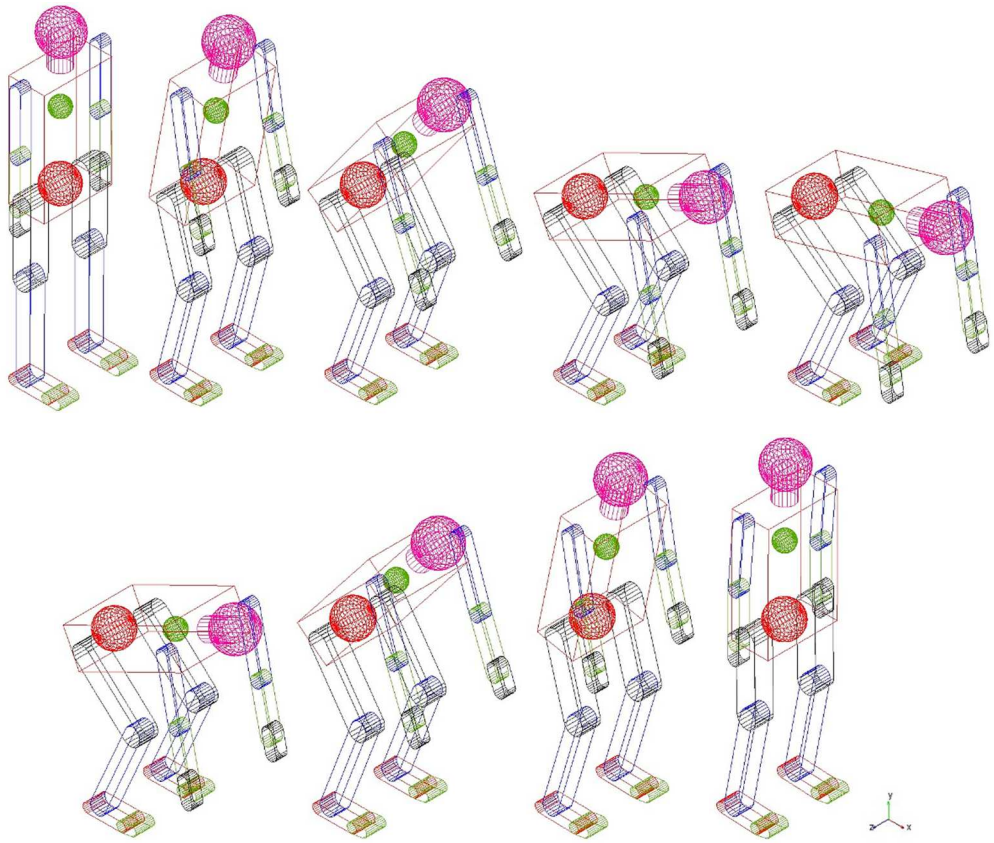


Figura 30 - Istanti della posa 2a - vista assonometrica

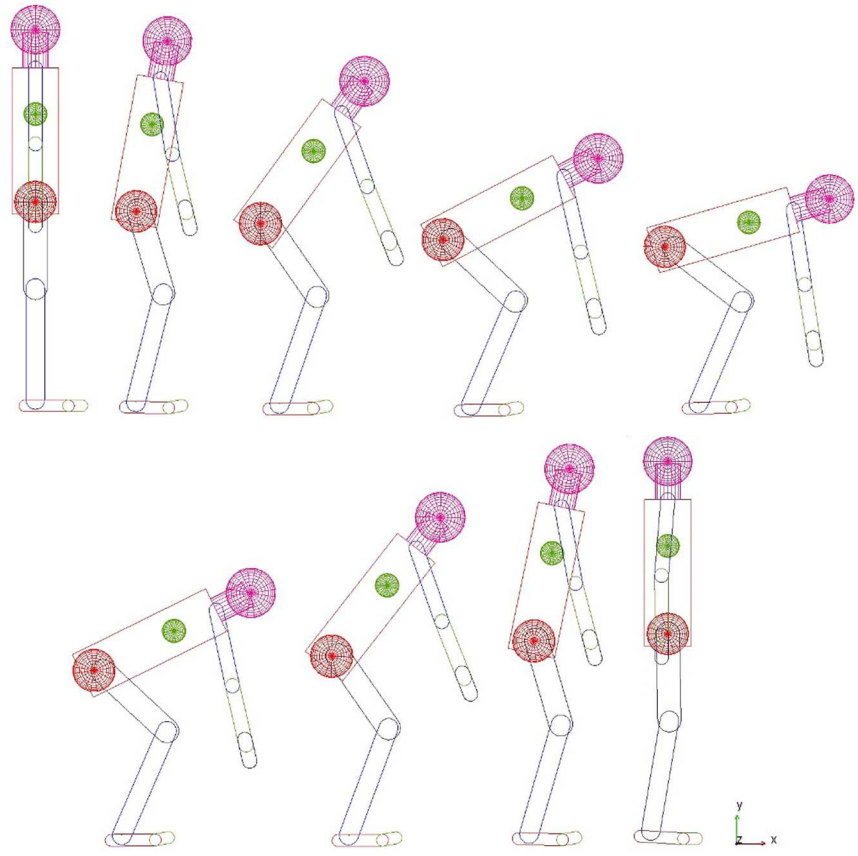


Figura 31 - Istanti della posa 2a - vista laterale

2.3.6 Equazioni del moto: Posa 3

L'ultimo movimento che si va a simulare è la rotazione del busto da sinistra a destra come per spostare degli oggetti da un piano di lavoro ad un altro. Per come definita, questa attività è caratterizzata dagli angoli ai giunti di anca, ginocchio e caviglia costanti nel tempo. Variano invece gli angoli di spalla e gomito nel tempo (per poter permettere la presa di oggetti) e la rotazione del busto attorno all'asse verticale. È già stata discussa in precedenza, nel paragrafo 2.2.1, la necessità di modificare il modello per permettere la torsione del busto. Si passa dunque al procedimento seguito per la definizione delle leggi del moto.

Come per i movimenti precedenti, si è partiti dal simulare l'attività su Tecnomatix Jack. Non essendo presente, questo movimento, tra quelli predefiniti, è stato necessario impostare a mano gli angoli dei diversi giunti, come per la posa 2b. Dopodiché, è il software che calcola gli angoli nel passaggio da una posa alla successiva. Dunque, come output si hanno nuovamente gli andamenti degli angoli ai giunti nel tempo, che vengono successivamente trasferiti su MATLAB per permettere l'interpolazione dei dati mediante serie di Fourier. Nella Figura 32 sono riportati i grafici con i dati ottenuti da Jack (i punti blu) e le serie di Fourier utilizzate per interpolarli (curve rosse).

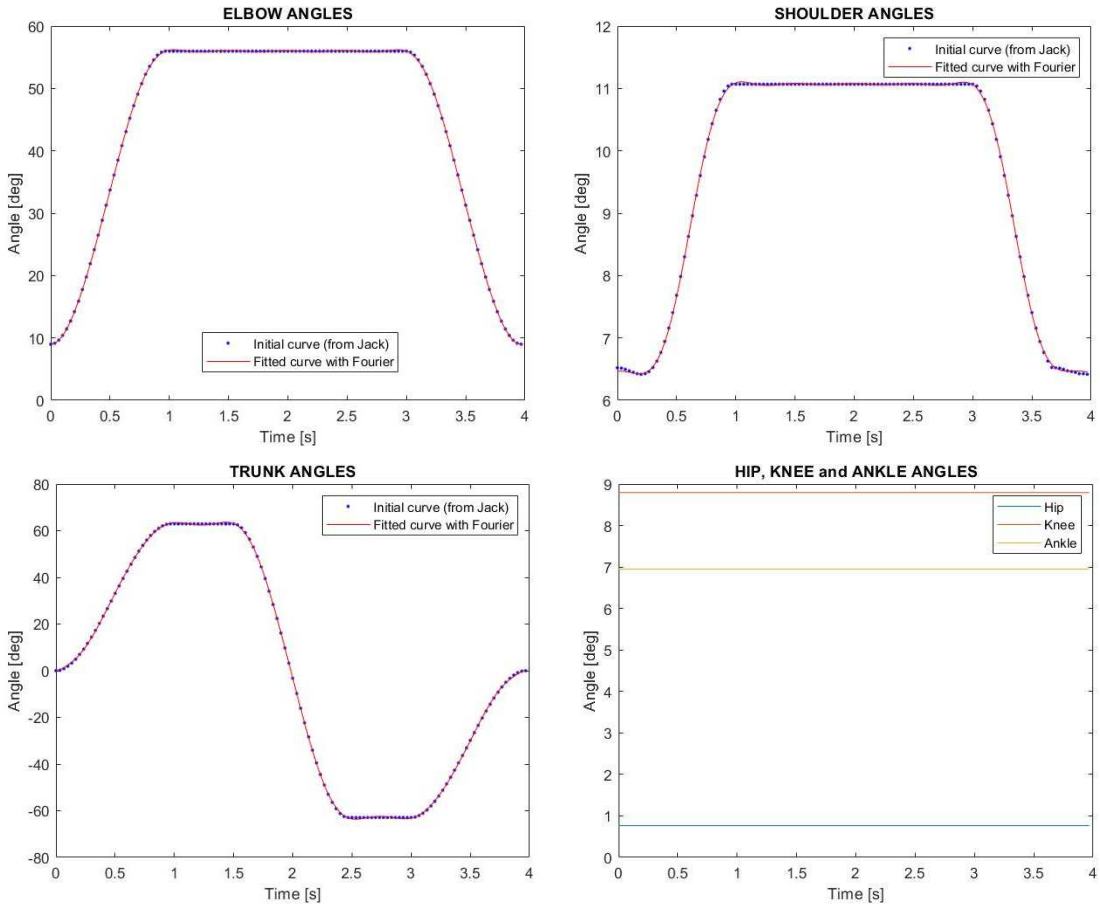


Figura 32 - Serie di Fourier utilizzate per interpolare gli andamenti angolari dei giunti per la posa 3

Come anticipato, gli angoli ai giunti di anca, ginocchio e caviglia sono costanti nel tempo: per essi si sono presi come riferimento quelli caratterizzanti la semplice posa eretta ("stand" su Tecnomatix Jack).

Poiché vi sono, in questo caso, dei tratti di curva caratterizzati da angoli costanti, per poter interpolare al meglio i dati ottenuti da Jack è stato necessario ricorrere a serie di Fourier tutte di settimo grado. Per quanto concernono i giunti delle gambe, invece, essendo gli angoli costanti nel tempo essi non sono stati, ovviamente, interpolati con alcuna serie di Fourier.

In Adams/View sarà dunque necessario fornire delle leggi del moto sotto forma di serie di Fourier ai giunti di gomito, spalla e al tronco (corpo colonna). Ai giunti di anca, ginocchio e caviglia viene invece fornita in input una rotazione costante nel tempo attorno all'asse Z, e pari a quella rappresentata in Figura 32.

Preso dunque il modello generale di una serie di Fourier nel tempo:

$$f(time) = a_0 + \sum_{i=1}^{n} a_i \cdot \cos(i \cdot \omega \cdot time) + b_i \cdot \sin(i \cdot \omega \cdot time)$$

Per i giunti di spalla, gomito e tronco i coefficienti a_i e b_i impiegati, con i relativi pesi ω, sono riportati nella Tabella 7.

Tabella 7 - Coefficienti di Fourier per posa 3

	GOMITO	SPALLA	BUSTO
a_0	24.72	9.717	-1.326E-06
a_1	-10.88	-2.255	0.9061
b_1	0.02196	-0.1607	-32.17
a_2	-6.515	-1.222	-0.3981
b_2	0.02631	-0.1762	7.18
a_3	-2.321	-0.2547	0.1182
b_3	0.01406	-0.05858	-1.419
a_4	-0.09725	0.2106	-0.3514
b_4	0.0007882	0.05631	3.159
a_5	0.2668	0.2149	0.01056
b_5	-0.00269	0.07385	-0.07577
a_6	-0.03681	0.06729	0.01283
b_6	0.000447	0.02494	-0.07642
a_7	-0.1468	-0.008331	-0.06582

b_7	0.002076	-0.009608	0.3352
ω	1.583	1.619	1.598

Anche in questo caso, le leggi del modo devono essere fornite ad Adams/View in radianti, non in gradi, per cui è necessario effettuare la trasformazione da gradi a radianti come segue:

$$g(time) = \frac{\pi}{180} \cdot f(time)$$

Si fornisce quindi in input la legge del moto in radianti g(time) anziché quella in gradi f(time).

Ad esempio, la legge del moto per il gomito è la seguente:

$$\begin{aligned} g_{GOMITO}(time) &= -\frac{\pi}{180} \cdot 1.8 \\ &\cdot \Big(24.72 - 10.88 \cdot cos(\omega_{GOMITO} \cdot time) + 0.02196 \\ &\cdot sin(\omega_{GOMITO} \cdot time) - 6.515 \cdot cos\big(2 \cdot (\omega_{GOMITO} \cdot time)\big) + 0.02631 \\ &\cdot sin\big(2 \cdot (\omega_{GOMITO} \cdot time)\big) - 2.321 \cdot cos\big(3 \cdot (\omega_{GOMITO} \cdot time)\big) \\ &+ 0.01406 \cdot sin\big(3 \cdot (\omega_{GOMITO} \cdot time)\big) - 0.09725 \\ &\cdot \cos\big(4 \cdot (\omega_{GOMITO} \cdot time)\big) + 0.0007882 \cdot \sin\big(4 \cdot (\omega_{GOMITO} \cdot time)\big) \\ &+ 0.2668 \cdot cos\big(5 \cdot (\omega_{GOMITO} \cdot time)\big) - 0.00269 \\ &\cdot sin\big(5 \cdot (\omega_{GOMITO} \cdot time)\big) - 0.03681 \cdot cos\big(6 \cdot (\omega_{GOMITO} \cdot time)\big) \\ &+ 0.000447 \cdot sin\big(6 \cdot (\omega_{GOMITO} \cdot time)\big) - 0.1468 \\ &\cdot cos\big(7 \cdot (\omega_{GOMITO} \cdot time)\big) + 0.002076 \cdot sin\big(7 \cdot (\omega_{GOMITO} \cdot time)\big)\Big) \end{aligned}$$

E in maniera analoga sono ottenute quelle di spalla e busto. La legge del moto per il busto è fornita al corpo "colonna", che fa parte della porzione di busto superiore. Per permettere la rotazione attorno all'asse verticale, poi, è lasciata libera la rotazione attorno a Y (asse verticale in Adams) per il giunto sferico che collega parte superiore e inferiore del busto, mentre sono bloccate le rotazioni attorno ai due assi trasversali.

Infine, le rotazioni per i giunti delle gambe sono impostate come segue (si fa l'esempio per l'anca, ma in maniera analoga si procede per ginocchio e caviglia):

$$g_{ANCA}(time) = -\frac{\pi}{180} \cdot 0.766$$

Poiché anche in tal caso è necessario passare dagli angoli in gradi agli angoli in radianti prima di fornire le leggi ad Adams/View.

In conclusione, nelle Figura 33 e Figura 34, è riportata la sequenza delle pose caratterizzanti il movimento in analisi. Si può in tal modo notare che l'operazione coinvolge le sole rotazioni di spalla e gomito e la torsione del busto, mentre le gambe restano fisse nella posizione iniziale.

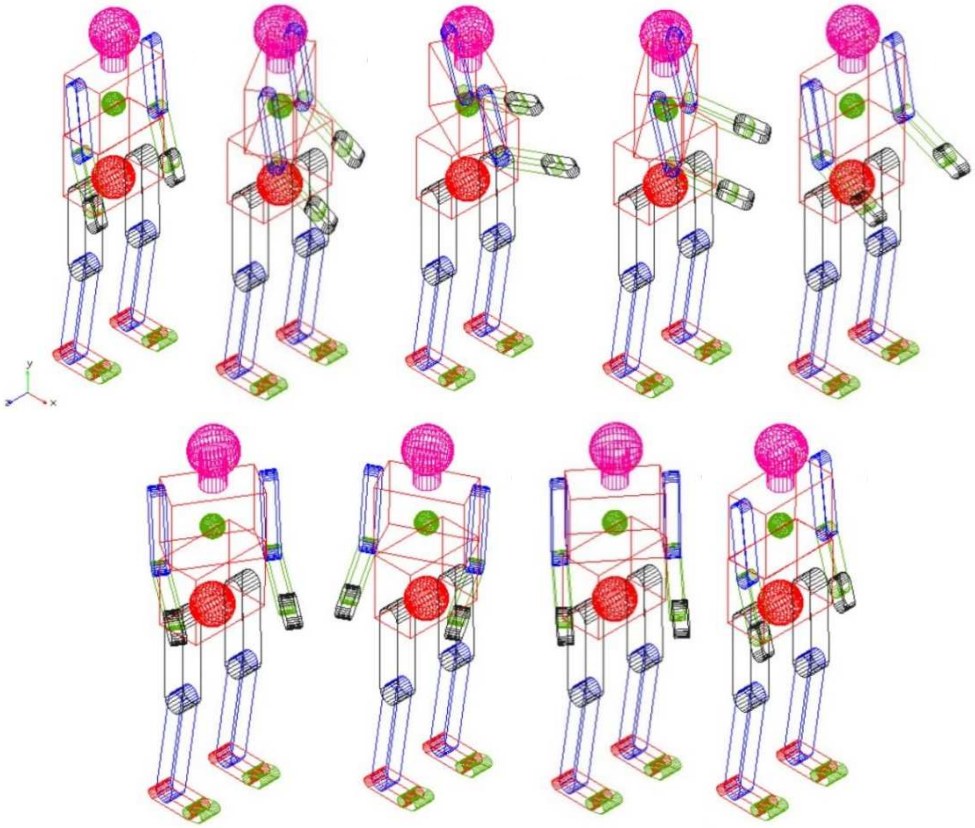


Figura 33 - Istanti della posa 3 - vista assonometrica

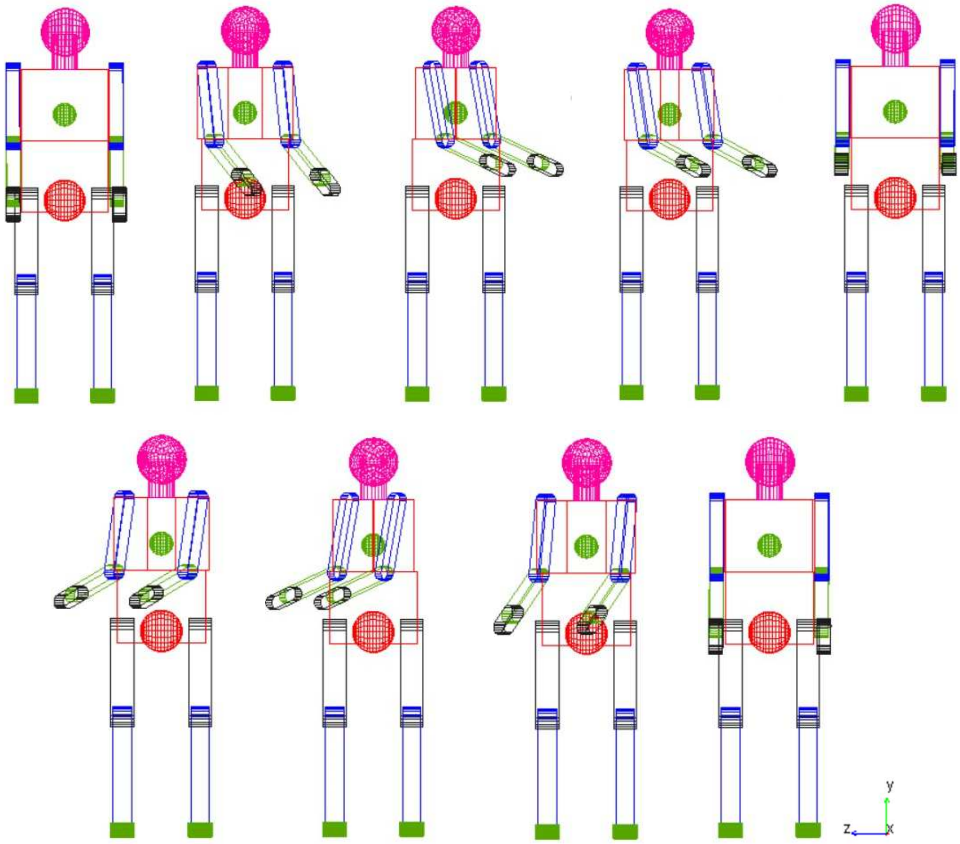


Figura 34 - Istanti della posa 3 - vista frontale

2.4 Validazione del modello

Affinché il modello multibody realizzato possa essere utilizzato per simulare su calcolatore attività caratteristiche dei luoghi di lavoro e prevedere i dati raccolti dal dispositivo sensorizzato IoT, è necessario procedere con la sua validazione. Per ciascuna posa, si effettueranno diverse tipologie di verifiche, in parte anche rispetto a dati raccolti sperimentalmente.

Per quanto concernono i dati sperimentali, questi sono stati raccolti utilizzando un caschetto da lavoro con un MetaMotionS (MMS) del MbientLab[3]. Si tratta di un apparecchio dotato di numerosi sensori che permette il continuo monitoraggio dei movimenti e dei parametri ambientali. Il dispositivo presenta dimensioni ridotte (27 x 27 x 4 mm), elevata potenza, alimentazione a batteria e risulta indossabile. La scheda MMS è una IMU a 10 assi con sensore di monitoraggio dell'ambiente. In particolare, è dotata di:

+ Giroscopio, accelerometro, magnetometro;
+ Barometro, sensore di temperatura e di luminosità ambientale;
+ Batteria, memoria, Bluetooth 4.0;
+ Micro USB per la ricarica.

Dunque, per le prove sperimentali, si sono simulati i movimenti di interesse utilizzando un dispositivo sensorizzato costituito da casco da lavoro + MMS + casing, come mostrato in Figura 35. Il casing mostrato in Figura 35 permette di alloggiare il dispositivo MMS ed è stato agganciato alla porzione posteriore della bandatura interna del casco.
In tale fase, non si è ancora utilizzato il dispositivo indossabile progettato al Capitolo 1 poiché si è solamente interessati al raccoglimento dei dati della IMU e a tal fine il dispositivo MMS risulta più pratico e di più semplice utilizzo rispetto al dispositivo realizzato con le due schede Arduino. Anche per la raccolta dei dati sperimentali si è impiegata una frequenza di acquisizione di 50 Hz, che si era vista essere la più adeguata nel Capitolo 1.

Figura 35 - Dispositivo utilizzato per la raccolta dei dati sperimentali

Nel seguito si analizzerà dunque la validazione del modello multibody per le tre pose precedentemente illustrate.

[3] Maggiori info su tali dispositivi si possono consultare al sito https://mbientlab.com/.

2.4.1 Validazione del modello: Posa 1

Nel caso della camminata, affinché il modello multibody risulti validato è necessario effettuare le seguenti verifiche:

+ Forza di contatto piede – terreno e affondamento del piede;
+ Coppie ai giunti;
+ Accelerazione del capo.

Si procede di seguito con tali verifiche.

Verifica della forza di contatto piede – terreno e dell'affondamento del piede.

Per poter considerare il modello validato, è dapprima necessario prestare attenzione alla forza di contatto scambiata tra piede e suolo e all'affondamento del piede. In Adams/View si modella il contatto avampiede – ground e retropiede – ground utilizzando il modello di contatto Hertziano non lineare, basato sulla seguente formula:

$$F_N = k \cdot d^e \qquad [N]$$

Dove k è la rigidezza del contatto, e è l'esponente (con valore tipico di 1-1.2) e d è la distanza oltre la quale applicare il coefficiente di smorzamento c. La forza di contatto dipende poi anche dall'attrito, e quindi dal coefficiente di attrito statico, dal coefficiente di attrito dinamico e dalle velocità di transizione di attrito. Il problema dipende dunque da numerosi parametri.

Per approcciare il problema, si è deciso di sviluppare una analisi di sensitività, volta alla definizione della coppia rigidezza di contatto – coefficiente di smorzamento che permetta di ottenere i risultati migliori. Si analizzano quindi i soli effetti di rigidezza k e smorzamento c sugli output del problema: la forza di reazione del terreno (GRF - Ground Reaction Force) e l'affondamento del piede nel terreno.

Per quanto riguarda la GRF, questa è definita come la forza di contatto tra piede e ground (ossia la somma della forza di contatto tra avampiede e ground e quella tra retropiede e ground). Il valore massimo teorico per la GRF durante la camminata è riportato in letteratura [36] e pari a:

$$GRF_{th} = 1.4 \cdot massa \cdot g$$

Mediante questo valore di GRF_{th} e quello della GRF massima ottenuta dalla simulazione e riportata dal post-processor di Adams, è possibile definire la deviazione, ossia l'errore relativo sulla GRF, che si desidera essere la minore possibile:

$$\%deviazione = \left(\frac{GRF_{th} - GRF}{GRF_{th}}\right)$$

Dove:

$$GRF = \left(GRF_{avampiede} + GRF_{retropiede}\right)_{MAX}$$

Durante la camminata, il massimo della forza di contatto piede – terreno si verifica nel momento in cui il tallone tocca terra, all'inizio della fase d'appoggio. In questo istante solamente il retropiede tocca terra, mentre l'avampiede è ancora alzato: la GRF del retropiede sarà quindi ivi massima, mentre quella dell'avampiede nulla. La forza di contatto piede – terreno da inserire all'interno della formula della %deviazione corrisponderà dunque alla massima GRF scambiata tra retropiede e terreno.

Per quanto concerne l'affondamento del piede, invece, questo è calcolato come la distanza lungo la verticale del centro di massa del retropiede dal suolo. Essendo il piede, nel modello, alto 5 cm, è poi necessario andare a sottrarre a tale valore una quantità pari a 2.5 cm, per avere il valore di affondamento effettivo della base del piede:

$$affondamento = distanza_{CMretropiede-suolo} - 2.5cm$$

Dopodiché, ipotizzando che durante la camminata sia ammessa una deformazione complessiva di piede e suola della scarpa pari a 2.5 cm al massimo, è possibile definire il valore limite di affondamento di -2.5 cm. Per cui:

$$affondamento \geq -2.5\ cm$$

Da cui:

$$distanza_{CMretropiede-suolo} \geq 0\ cm$$

Caratterizzati gli output del problema, è possibile passare all'analisi di sensitività. Essa è stata condotta come segue:

1. Valutazione dell'errore relativo sulla GRF e dell'affondamento del piede al variare della rigidezza, mantenendo costante lo smorzamento;
2. Valutazione dell'errore relativo sulla GRF e dell'affondamento del piede al variare dello smorzamento, mantenendo costante la rigidezza;
3. Sulla base delle analisi 1. e 2., scelta di cinque coppie rigidezza/smorzamento che garantiscano un affondamento del piede entro il valore limite e un errore relativo sulla GRF limitato. Analisi di tali coppie al variare di peso/altezza dell'individuo, e selezione della coppia rigidezza/smorzamento adatta ad una più ampia varietà di individui.

Per quanto riguarda la valutazione di %deviazione e affondamento del piede al variare della rigidezza, con smorzamento costante e pari a 1300 Nm/s, i valori che si sono ottenuti sono mostrati nei grafici riportati in Figura 36 e Figura 37. In particolare, si nota che con valori di rigidezza superiori a 3.6 E+04 N/m risulta soddisfatta la condizione sull'affondamento massimo[4] del piede. All'aumentare

[4] Si fa qui riferimento all'affondamento totale del piede, ossia alla quantità $distanza_{CMretropiede-suolo} - 2.5cm$, già sottratta della metà altezza del piede. Il valore di soglia risulta dunque pari a -2.5 cm.

della rigidezza, tuttavia, se da una parte lo sprofondamento massimo diminuisce, dall'altra si ha un aumento, in modulo, dell'errore sulla forza di contatto: sarà dunque necessario trovare un compromesso tra i due.

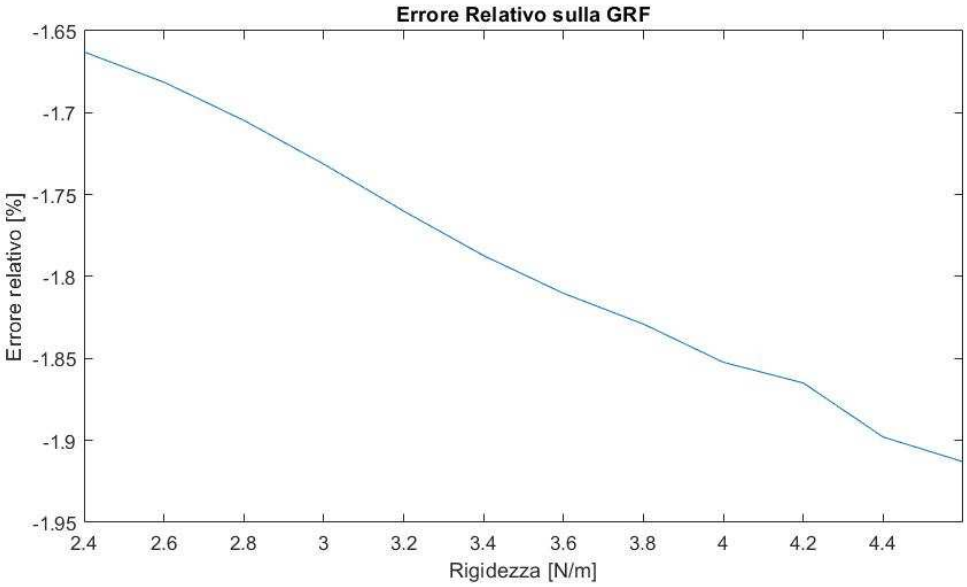


Figura 36 - Errore relativo sulla GRF al variare della rigidezza

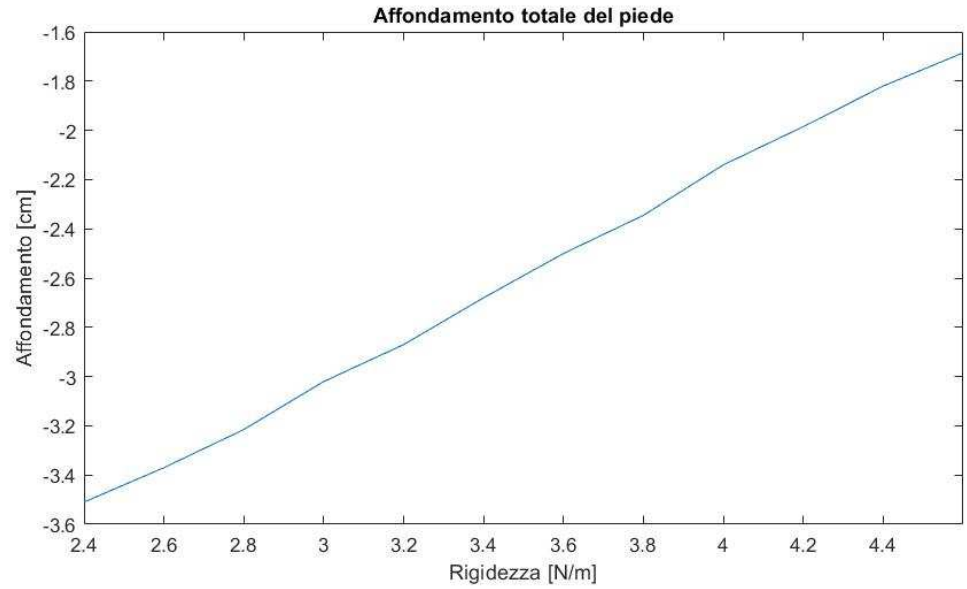


Figura 37 - Affondamento totale del piede al variare della rigidezza

La stessa analisi è poi stata condotta anche al variare dello smorzamento, mantenendo la rigidezza costante al valore di 4.0 E+04 N/m. I risultati sono riportati nei grafici delle Figura 38 e Figura 39. In tal caso, è necessario uno smorzamento superiore a 1000 Nm/s per soddisfare la condizione sull'affondamento massimo del piede. Analogamente alla rigidezza, anche aumentando lo smorzamento si ha un aumento del modulo dell'errore relativo sulla GRF e una diminuzione dell'affondamento complessivo del piede, per cui sarà necessario scendere a un compromesso tra i due.

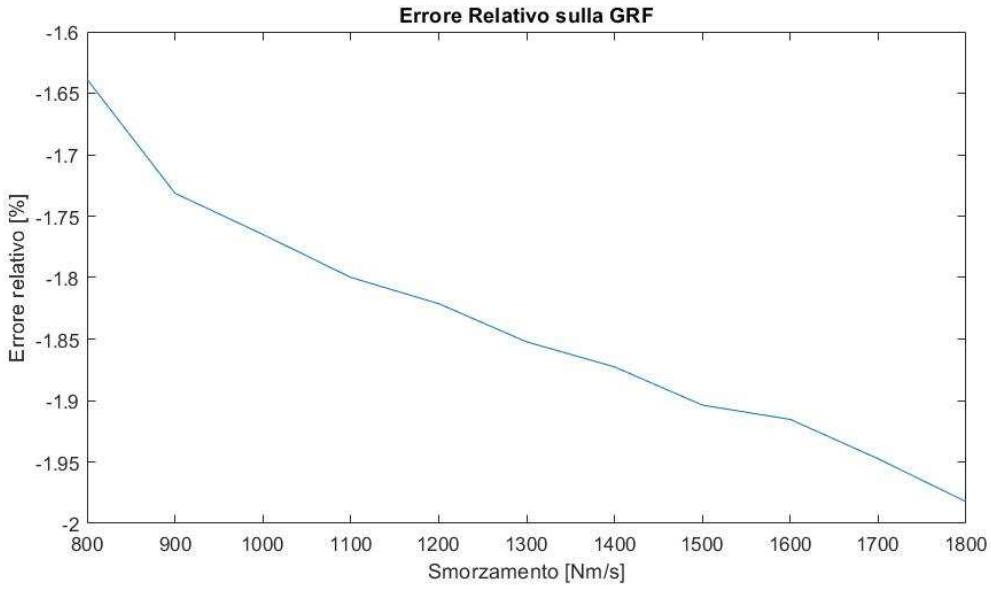


Figura 38 - Errore relativo sulla GRF al variare dello smorzamento

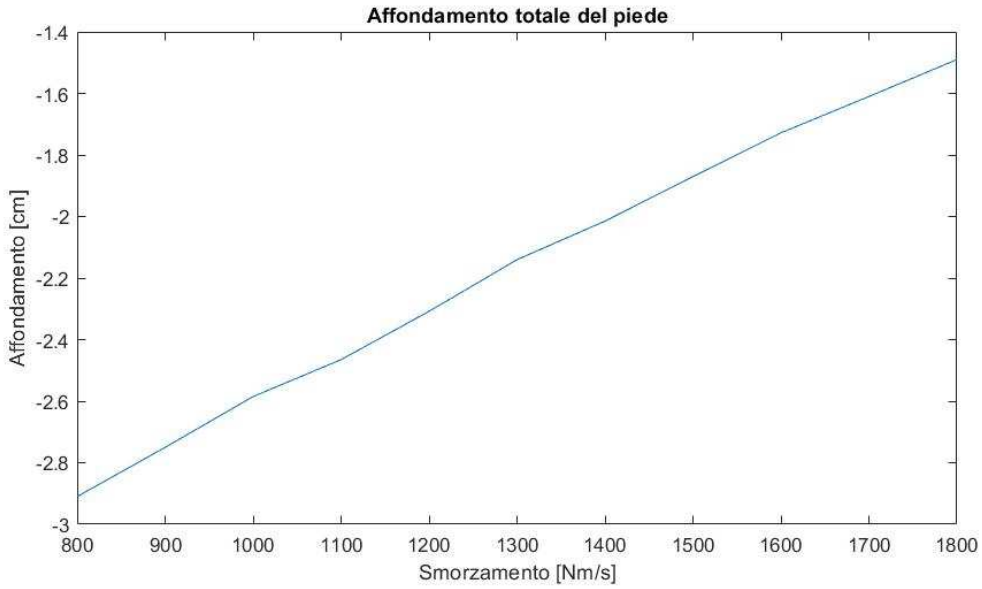


Figura 39 - Affondamento totale del piede al variare dello smorzamento

A questo punto, sulla base delle analisi condotte in precedenza, sono state scelte le seguenti cinque coppie di rigidezza/smorzamento, da analizzare al variare di peso e altezza dell'individuo:

1. K = 3.6 E+04 N/m, c = 1300 Nm/s;
2. K = 3.7 E+04 N/m, c = 1400 Nm/s;
3. K = 3.8 E+04 N/m, c = 1200 Nm/s;
4. K = 4.0 E+04 N/m, c = 1300 Nm/s;
5. K = 4.2 E+04 N/m, c = 1400 Nm/s.

Tale analisi di sensitività è volta alla determinazione di una coppia rigidezza/smorzamento che possa essere il più universale possibile, e quindi

adattabile a diverse taglie di individuo. Durante il processo, le taglie dell'individuo vengono fatte variare in maniera "proporzionale", ad esempio massa di 60 kg e altezza di 160 cm, o massa di 75 kg e altezza di 175 cm. Infatti, da analisi condotte in precedenza [36], si è visto che queste coppie di altezza – massa sono quelle cui corrispondono i migliori valori di scostamento della GRF. Si è dunque analizzata la variabilità dell'errore relativo sulla GRF e dell'affondamento del piede in funzione delle coppie smorzamento/rigidezza appena definite e delle seguenti taglie di individuo:

1. Massa = 60 kg, altezza = 160 cm;
2. Massa = 65 kg, altezza = 165 cm;
3. Massa = 70 kg, altezza = 170 cm;
4. Massa = 75 kg, altezza = 175 cm;
5. Massa = 80 kg, altezza = 180 cm;
6. Massa = 85 kg, altezza = 185 cm.

I risultati ottenuti sono riportati nei seguenti grafici, in Figura 40 e Figura 41.

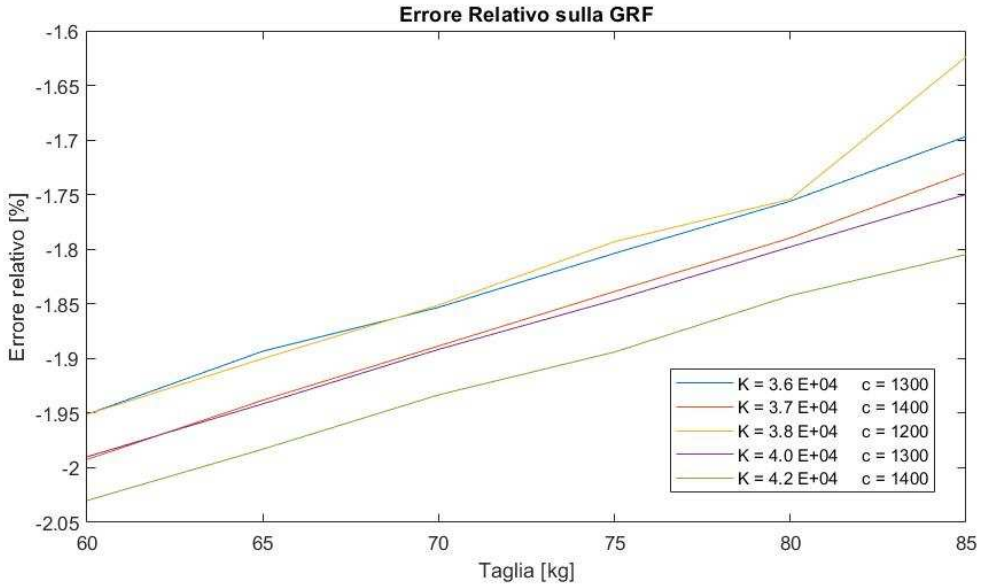


Figura 40 - Errore relativo sulla GRF al variare della taglia dell'individuo

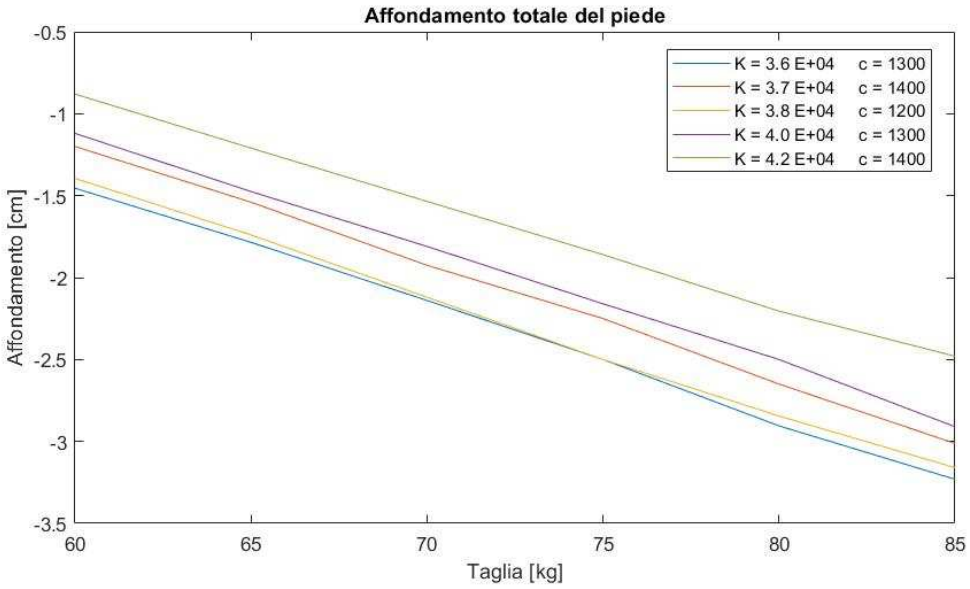


Figura 41 - Affondamento totale del piede al variare della taglia dell'individuo

Come si può notare, tra l'affondamento totale e l'errore sulla GRF il criterio che risulta essere più restrittivo è il primo. In particolare, l'unica coppia rigidezza/smorzamento che consente di soddisfare il valore soglia per l'affondamento del piede per tutte le taglie di individuo è K = 4.2 E+04 N/m e c = 1400 Nm/s, che viene dunque selezionata per il modello.

L'analisi di sensitività è quindi stata utile per definire i valori di rigidezza di contatto e coefficiente di smorzamento da inserire nel calcolo della forza di contatto in modo da garantire:

+ Il minore errore relativo possibile sulla GRF;
+ Un affondamento massimo del piede inferiore a 2.5 cm;
+ Adattabilità del modello a diverse taglie di individuo.

Verifica delle coppie ai giunti degli arti inferiori.

Essendo stata la camminata ampiamente analizzata da numerosi studiosi, molti risultati legati all'analisi cinematica e dinamica sono già presenti in letteratura: è dunque possibile validare il modello anche confrontando i risultati ottenuti con quelli di studi precedenti. Infatti, il modello finora ottenuto dà origine a coppie nei giunti di anca, ginocchio e caviglia molto più elevate rispetto a quelle riportate in letteratura. Questo è da imputarsi al fatto che si tratta di un modello estremamente rigido e poco smorzato rispetto alla realtà, con alcuni giunti (quali quello del ginocchio) infinitamente rigidi. Si procede dunque modificando ulteriormente il modello al fine di ottenere coppie ai giunti comparabili con quelle riportate in letteratura. In particolare, le modifiche hanno coinvolto la variazione

di rigidezza e smorzamento rotazionale attorno all'asse Z[5] per i giunti di anca e caviglia, e l'introduzione di smorzamento per il ginocchio. Nel dettaglio:

+ Per quanto concerne l'anca, al fine di validare il modello sulla base delle coppie ai giunti si sono modificati rigidezza e smorzamento rotazionali del bushing presente tra coscia e busto:
  - RIGIDEZZA ROTAZIONALE attorno a Z: 1 Nm/°;
  - SMORZAMENTO ROTAZIONALE attorno a Z: 0.3 Nms/°.
+ Al ginocchio è stata aggiunta una molla torsionale, in modo da introdurre uno smorzamento rotazionale al giunto in analisi:
  - COEFFICIENTE di SMORZAMENTO ROTAZIONALE attorno a Z: 0.2.
+ Infine, per la caviglia, si è proceduto in maniera analoga all'anca, modificando rigidezza e smorzamento rotazionali del bushing presente tra gamba e retropiede:
  - RIGIDEZZA ROTAZIONALE attorno a Z: 45 Nm/°;
  - SMORZAMENTO ROTAZIONALE attorno a Z: 0.2 Nms/°.

Nelle immagini seguenti, Figura 42, Figura 43 e Figura 44, si riportano i grafici con il confronto tra le coppie presenti in letteratura [39, 40], in blu, e quelle ottenute dalla simulazione su Adams/View apportando le modifiche appena definite, in rosso.

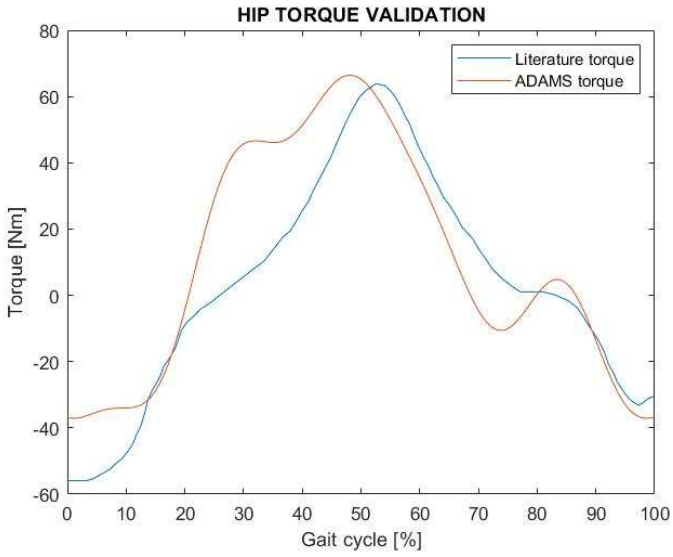


Figura 42 - Validazione delle coppie all'anca con quelle riportate in letteratura

[5] L'asse Z nel modello costruito su Adams è l'asse che descrive le rotazioni del piano sagittale.

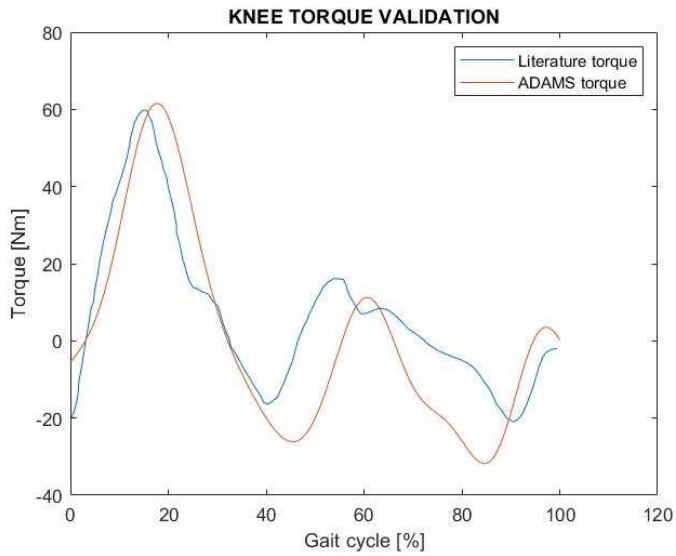


Figura 43 - Validazione delle coppie al ginocchio con quelle riportate in letteratura

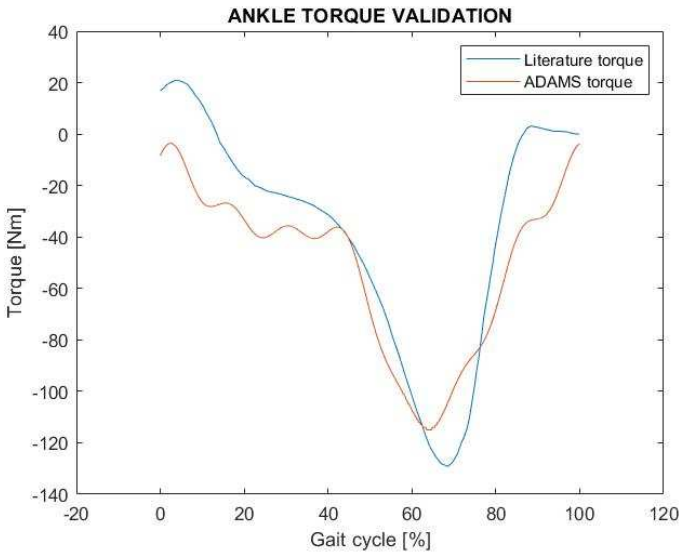


Figura 44 - Validazione delle coppie alla caviglia con quelle riportate in letteratura

Come si può notare, con i valori di rigidezza e smorzamento in precedenza illustrati si ha una buona corrispondenza per tutti e tre i giunti sia in termini di valore della coppia al giunto sia in termini di andamento in funzione della percentuale di ciclo della camminata. Per quanto riguardano le coppie ai giunti degli arti inferiori, si può dunque considerare validato il modello multibody.

Verifica dell'accelerazione del capo.

Dopo aver validato il modello multibody realizzato in funzione della forza di contatto con il suolo teorica, dell'affondamento del piede e delle coppie ai giunti riportate in letteratura, è necessario procedere con il confronto con i risultati sperimentali. Essendo il dispositivo progettato al Capitolo 1 un casco sensorizzato, si fa riferimento all'accelerazione del capo, poiché l'unica in grado di rilevare il

suddetto dispositivo. Come anticipato in precedenza, tuttavia, per semplicità i risultati sperimentali sono stati raccolti utilizzando il sensore MMS, dotato anche di accelerometro, analogamente al dispositivo realizzato.

Per la camminata, sono state effettuate diverse prove sperimentali costituite da nove passi ciascuna, così come il modello multibody. Al fine di effettuare passi sempre della stessa lunghezza, sono state collocate delle tacche sul pavimento, che permettessero una maggiore regolarità di camminata. Per quanto riguarda la durata dei passi, invece, si è cercato di mantenere sempre la stessa andatura, il più simile possibile a quella impostata su Adams/View. Ciò nonostante, al momento del confronto è stato necessario scalare leggermente i grafici in funzione del tempo, per far sì che tutte le prove avessero esattamente la stessa durata e fossero dunque comparabili e sovrapponibili.

I risultati sperimentali, essendo molto disturbati, hanno dapprima necessitato l'interpolazione. In particolare, si è effettuata sia l'interpolazione mediante serie di Fourier che mediante polinomi. La prima metodologia è stata selezionata, in quanto in tal caso maggiormente rappresentativa dei risultati sperimentali. È stato quindi necessario interpolare i dati sperimentali con delle serie di Fourier: nella maggior parte dei casi sono risultate adeguate serie di ordine 4.

Una preventiva elaborazione è poi stata necessaria anche per i risultati estrapolati dal post - processor di Adams/View. Per prima cosa, essi sono stati interpolati con delle serie di Fourier, come i risultati sperimentali, in maniera tale da rimuovere alcuni picchi locali nell'andamento dovuti alle approssimazioni del modello multibody. Inoltre, essendo il sistema multibody molto rigido rispetto a quello di un umano, è stato necessario apportare un'ulteriore correzione poiché le accelerazioni risultavano molto più elevate di quelle registrate sperimentalmente. Si sono effettuate diverse prove aggiungendo differenti valori di smorzamento localizzati al modello, ma le accelerazioni risultavano comunque molto più elevate rispetto a quelle sperimentali. Questo è dovuto al fatto che il modello è realizzato mediante un'unione di corpi rigidi, per cui anche aggiungendo smorzamenti localizzati non è possibile simulare lo smorzamento distribuito caratteristico del corpo umano. Si sono allora scalati i grafici rispetto ai valori sperimentali massimi, validando il modello solamente sulla base dell'andamento delle accelerazioni lungo X e Y. Ad esempio, per le accelerazioni lungo x si è proceduto come segue:

$$Acc_x = Acc_x \frac{MAX_{Acc_{x\_sperimentale}}}{MAX_{Acc_{x\_Adams}}}$$

E analogamente per le accelerazioni lungo Y. I valori corretti si potranno poi ottenere sul simulatore andando ad aggiungere l'opportuno smorzamento nelle diverse parti del corpo.

Coinvolgendo la camminata principalmente movimenti sul piano sagittale, per come fornite le leggi del moto su Adams/View, il modello non riporta movimenti lungo la direzione Z, ortogonale al piano sagittale. I risultati sperimentali sono invece caratterizzati da rumore in direzione Z, ma l'accelerazione lungo tale asse risulta comunque tendente a zero, e quindi trascurabile.

Nelle immagini seguenti si riporta il confronto effettuato tra le accelerazioni del capo estrapolate dal post - processor di Adams/View e quelle di tre prove sperimentali. Nello specifico, in Figura 45 è riportato il confronto delle accelerazioni lungo X per nove passi, in Figura 46 il medesimo paragone su un passo solo, in Figura 47 la comparazione delle accelerazioni lungo Y per nove passi e in Figura 48 il confronto per un solo passo.

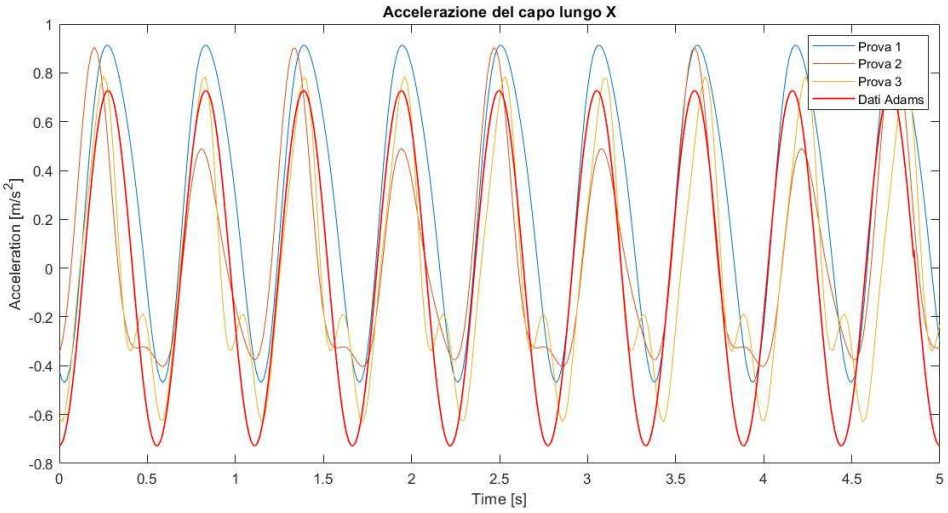


Figura 45 - Accelerazione del capo lungo X - confronto sui 9 passi

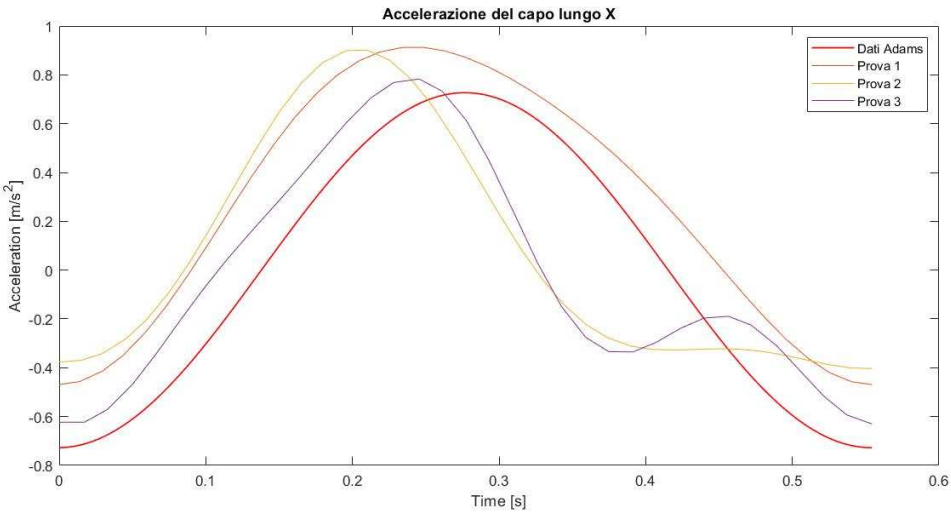


Figura 46 - Accelerazione del capo lungo X - confronto sul singolo passo

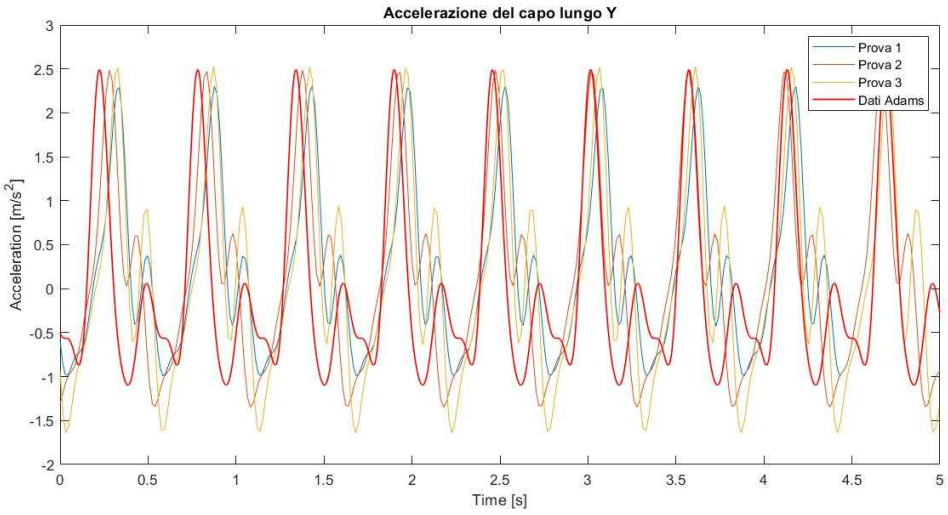


Figura 47 - Accelerazione del capo lungo Y - confronto sui 9 passi

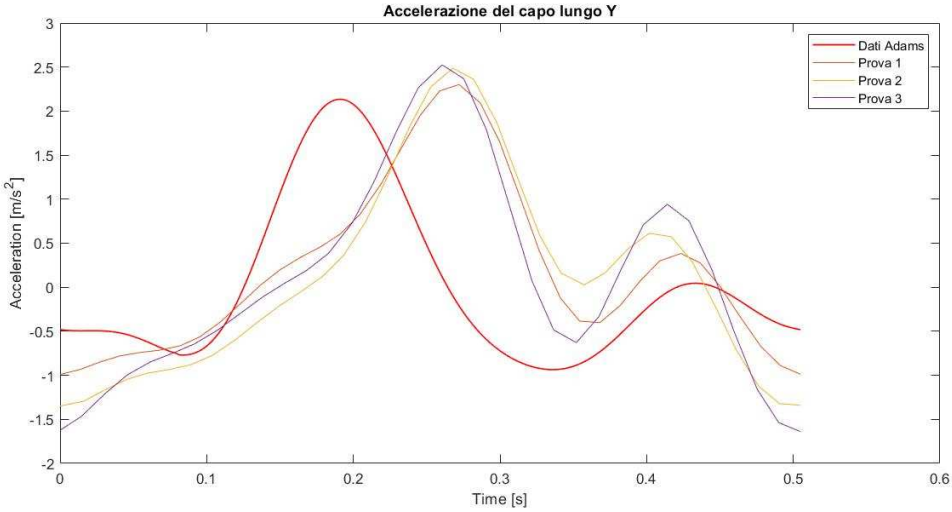


Figura 48 - Accelerazione del capo lungo Y - confronto sul singolo passo

Come si può notare dai grafici precedenti, gli andamenti delle accelerazioni di Adams/View sono molto simili a quelli sperimentali. Si ha un primo picco di accelerazione nel momento in cui il tallone tocca terra, e poi un secondo picco, minore, quando la restante parte di piede raggiunge il suolo. Il secondo picco è principalmente registrato dall'accelerazione lungo Y, mentre quella lungo X non sempre lo rileva. Come si può notare dalla Figura 46, l'accelerazione lungo X del modello Adams è molto simile a quella raccolta sperimentalmente, in particolare alla prova 2. Le altre due prove si discostano invece leggermente perché presentano la porzione con un secondo picco poco marcato. Una maggiore differenza si rileva invece nell'andamento della accelerazione Y, poiché i due picchi risultano più distanti tra loro nei risultati del simulatore rispetto quelli sperimentali: questo è probabilmente dovuto alla geometria del piede del modello multibody, diviso in piede e avampiede.

Oltre al loro andamento nel tempo, si è però anche interessati al valore efficacie (Root Mean Square – RMS) delle accelerazioni. Si vuole infatti che il modello Adams sia perlomeno in grado di simulare correttamente il valore RMS della accelerazione sperimentale. Quest'ultimo è calcolabile come segue:

$$RMS_x = \sqrt{\frac{\sum_{i=1}^{N} Ax_i^2}{N}}$$

$$RMS_y = \sqrt{\frac{\sum_{i=1}^{N} Ay_i^2}{N}}$$

Elaborando opportunamente i risultati sperimentali e del simulatore, i rispettivi valori efficaci sono riportati in Tabella 8. Come si nota, sia per l'accelerazione lungo X che per l'accelerazione lungo Y si ha ottima corrispondenza tra i valori RMS dei risultati sperimentali e quello estrapolato da Adams/View.

Tabella 8 - Valori RMS dell'accelerazione al capo per la posa 1

Valore efficacie dell'accelerazione lungo X			
Prova 1	Prova 2	Prova 3	Dati Adams/View
0.557	0.4104	0.4434	0.5141
Valore efficacie dell'accelerazione lungo Y			
Prova 1	Prova 2	Prova 3	Dati Adams/View
0.9684	1.1089	1.445	1.0426

Per ultimo, si verifica che non vengano commessi errori troppo grossolani nel trascurare le accelerazioni minime lungo Z che possono nascere durante la simulazione della camminata. A tal proposito, si confronta il valore efficacie del modulo dell'accelerazione del capo di Adams/View con quello calcolato dai dati sperimentali. Questo è calcolabile come segue:

$$RMS_{modulo} = \sqrt{\frac{\sum_{i=1}^{N} a_{modulo\_i}^2}{N}}$$

Dove, per i risultati di Adams/View:

$$a_{modulo} = \sqrt{{a_x}^2 + {a_y}^2}$$

Mentre per i risultati sperimentali vale:

$$a_{modulo} = \sqrt{{a_x}^2 + {a_y}^2 + {a_z}^2}$$

Il confronto dei valori RMS nel caso in cui si consideri anche l'eventuale movimento in direzione Z è riportato in Tabella 9. Come riferimento di prova

sperimentale si è presa la prova 1, ma analogamente si sarebbe potuto procedere con le altre prove. Come mostrato, il valore efficacie del modulo dell'accelerazione del modello multibody rappresenta correttamente i risultati sperimentali, per cui l'approssimazione di trascurare l'accelerazione Z risulta adeguata.

Tabella 9 – Confronto del valore efficacie dell'accelerazione attorno diversi assi per posa 1

	RMS ω_x	RMS ω_y	RMS ω_z	RMS ω_{modulo}
Dati Adams/View	0.5038	1.0426	-	1.1579
Prova sperimentale	0.557	0.9685	0.3822	1.1808

A tal punto, il modello multibody per la posa 1 risulta correttamente validato, e quindi utilizzabile per la simulazione dell'attività.

2.4.2 Validazione del modello: Posa 2a

Per effettuare la validazione del modello per la posa 2a, è necessario verificare:

+ Forza di contatto piede – terreno;
+ Velocità angolare del capo.

Rispetto alla posa 1, dunque, non si effettua la verifica dell'affondamento del piede poiché esso, non movimentandosi rispetto al terreno, è caratterizzato da un affondamento sempre maggiore a zero. Inoltre, la verifica delle coppie ai giunti degli arti inferiori non è possibile poiché non sono presenti dati in letteratura. Infine, essendo il movimento più stazionario rispetto alla posa 1, si effettua ora la validazione sulla base della velocità angolare del capo, non più a seguito dell'accelerazione lineare dello stesso.

Nel seguito si procede dunque con le due verifiche appena definite.

<u>Verifica della forza di contatto piede – terreno.</u>

Al fine di validare il modello, è necessario controllare che la forza di contatto con il suolo sia all'incirca pari a $m \cdot g$. Infatti, rispetto alla camminata, l'individuo ha ora i piedi sempre poggiati a terra, per cui la forza di contatto totale scaricata al pavimento deve essere all'incirca pari alla forza peso dell'individuo. Nel caso di operatore di 75 kg, quindi:

$$GRF_{tot} = m \cdot g = 75[kg] \cdot 9.80665[m/s^2] = 735.499N$$

Questa è la forza complessiva scaricata dai due piedi dell'individuo. Di conseguenza, ciascuno di essi dovrà scaricare una forza pari a:

$$GRF_{sing\_piede} = GRF_{tot}/2 = 367.749\ N$$

GRF_{sing_piede} è la forza che deve essere scaricata al suolo da ciascun piede. Essendo il piede diviso in retropiede e avampiede, è dunque necessario che la somma delle forze scaricate da ciascuna parte del piede fornisca il valore di GRF_{sing_piede}.

Realizzando il grafico della forza di contatto di retropiede e avampiede nel tempo fornita dalla simulazione e dal post-processor di Adams, si ottiene la Figura 49. Per semplicità di lettura dei dati, è anche stata rappresentata la curva della forza totale di contatto piede – suolo e la curva indicante la GRF teorica. Quest'ultima è la GRF_{sing_piede} di riferimento, pari a 367.749 N. Come si può notare, la forza di contatto totale non risulta costante nel tempo: questo è probabilmente dovuto alla movimentazione dell'individuo, che fa anche variare l'entità delle forze di contatto tra avampiede e suolo e tra retropiede e suolo nel tempo. Ciò nonostante, la GRF complessiva ha un valore variabile ma sempre prossimo a quello di riferimento, per cui si può considerare validato il modello.

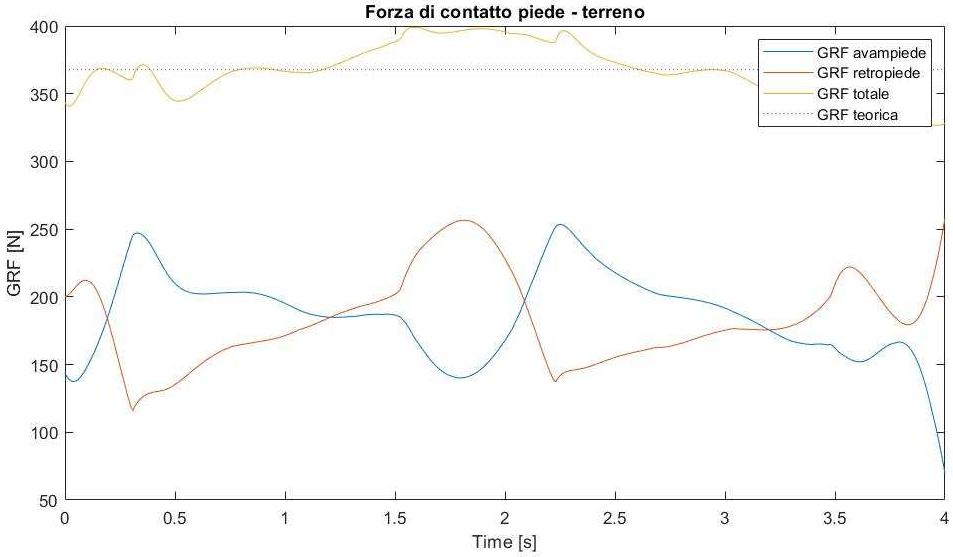


Figura 49 - Forza di contatto piede - terreno per posa 2a

Infine, per verificare che il simulatore percepisca correttamente l'aggiunta di un eventuale peso nel caso di operatore che si china a terra e raccoglie un oggetto, si procede come segue. Il peso viene simulato su Adams/View come una forza applicata sulle mani che passa istantaneamente da 0 a $N = M \cdot g$ (dove M è il peso, in kg, dell'oggetto) nel momento in cui l'umanoide è chinato a terra. Tale forza è diretta verticalmente verso il basso, come l'accelerazione di gravità. Per verificare che il peso sia correttamente trattato dal software, poi, si verifica nel post-processor che la forza di contatto totale con il terreno, pari a $m \cdot g$, veda una discontinuità nel momento in cui la forza passa da 0 a N. Tale discontinuità dovrà essere pari a $N = M \cdot g$.

Dunque, nel caso di massa dell'oggetto pari a 20 kg e di istante di presa dell'oggetto pari a 2s, la forza di contatto teorica con il suolo risulta:

$$\begin{cases} GRF_{sing\_piede} = (m_{INDIVIDUO} \cdot g)/2 = 367.749\ N & se\ t \leq 2s \\ GRF_{sing\_piede} = (m_{INDIVIDUO} + m_{OGGETTO}) \cdot g/2 = 465.816\ N & se\ t > 2s \end{cases}$$

Per ottenere questo, è dunque necessario impostare una forza concentrata con punto di applicazione nelle mani che passi istantaneamente da 0 a N. Questo si può ottenere mediante la funzione step, ossia fornendo alla forza applicata una legge del tipo:

```
step(x, x0, h0, x1, h1)
```

Dove x è il tempo ("time"), x0 è l'ultimo istante in cui la forza è pari al primo valore h0 e x1 è il primo istante in cui la forza è pari al secondo valore h1. Dunque, nel caso in esame, x0 = 2s, h0 = 0, x1 = 2.001s e h1 = N/2. Si imposta N/2 poiché si assume che la forza sia perfettamente equiparata tra le due mani, per cui ciascuna mano debba sopportare una forza dimezzata.

In conclusione, si vede dalla Figura 50 che la condizione ricercata di forza con salto in prossimità dei 2s risulta verificata: la GRF totale oscilla infatti sempre nell'intorno della linea tratteggiata, rappresentante la GRF teorica. È quindi possibile la simulazione della presa di un oggetto da terra per la posa 2a.

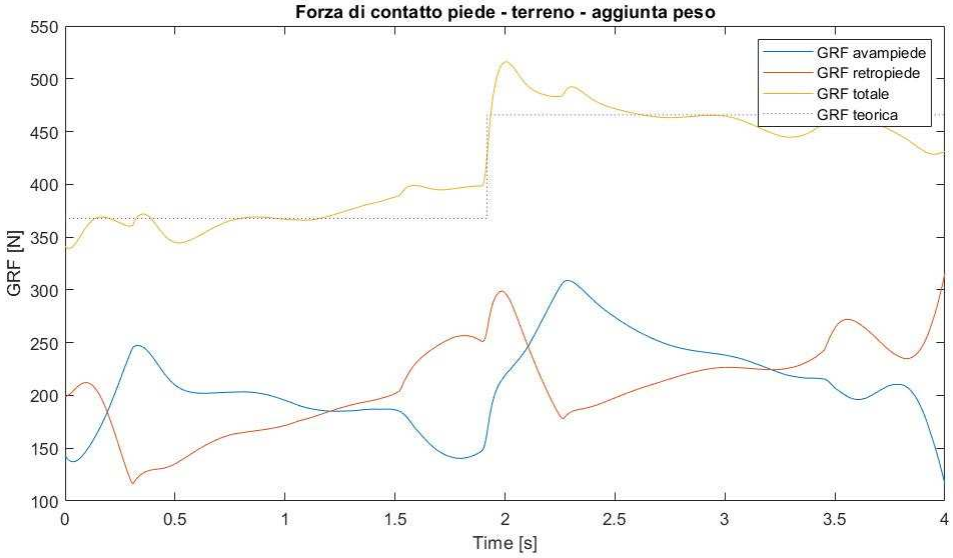


Figura 50 - Forza di contatto piede - terreno per posa 2a con peso

Verifica della velocità angolare del capo.

Per poter utilizzare il modello multibody per predire i dati rilevati dai sensori di movimento localizzati sul casco intelligente progettato al Capitolo 1, è anche necessaria la validazione in termini di velocità angolare del capo. Si verifica dunque che la velocità angolare del capo fornita dal simulatore sia comparabile con quella registrata sperimentalmente. A differenza della posa 1, si opera ora con

le velocità angolari e non con le accelerazioni lineari: la scelta è dettata dal fatto che si tratta di una posa più stazionaria, per cui le accelerazioni rilevate durante tale movimento risultano molto basse. È quindi in tal caso più difficile riuscire ad isolare il disturbo del segnale legato alle misurazioni sperimentali e individuare una tendenza per le accelerazioni che si ripeta per tutte le prove. Per tale motivo, lo stesso procedimento si seguirà anche per le pose 2b e 3.

Nel caso della posa 2a, è stato necessario porre l'attenzione sulla sola velocità angolare attorno all'asse Z. Infatti, il movimento in analisi si svolge interamente sul piano sagittale, piano X – Y, per cui la rotazione del capo avviene in tale piano ed è dunque una rotazione attorno all'asse Z, ortogonale al piano X - Y. Le velocità angolari attorno a X e a Y, a causa del disturbo dei risultati sperimentali, non risultano esattamente nulle, ma oscillanti attorno allo zero. Si tratta in ogni caso di valori contenuti rispetto a quelli delle velocità angolari attorno a Z, e dunque trascurabili.

Come nel caso precedente, si sono effettuate più prove sperimentali. In particolare, nel seguito si farà riferimento a cinque differenti prove sperimentali, che verranno confrontate con il risultato fornito dal post-processor di Adams/View. Anche in questo caso, è dapprima stato necessario interpolare i dati sperimentali, in maniera tale da rimuovere il rumore caratteristico della sperimentazione. Si è dunque effettuata l'interpolazione mediante il comando *fit* di MATLAB, sperimentando sia l'utilizzo delle serie di Fourier che dei polinomi. In questo caso, sono risultati maggiormente adeguati i polinomi, poiché leggermente più precisi nell'interpolazione delle serie di Fourier. In particolare, per ottenere una buona approssimazione di interpolazione dei dati, a seconda della prova sperimentale è stato necessario ricorrere a polinomi di diverso grado, ossia:

+ Polinomio di terzo grado per le prove 1 e 3;
+ Polinomio di quarto grado per la prova 5;
+ Polinomio di quinto grado per la prova 2;
+ Polinomio di ottavo grado per la prova 4.

Poiché non tutte le prove sperimentali avevano durata di esattamente 4s, è stato poi necessario scalare leggermente i grafici in maniera tale da ottenere delle curve sovrapponibili e dunque più facilmente confrontabili.

In Figura 51 si riporta il grafico così ottenuto. In rosso è rappresentata la prova effettuata su Adams/View, ossia i risultati forniti dal post-processor del software, mentre le altre curve sono le prove sperimentali, interpolate con polinomi di vario grado. Come si nota, si ha una buona rappresentazione dei dati sperimentali mediante la tendenza fornita da Adams/View: è dunque possibile utilizzare questo modello multibody per fare stime relative ai dati raccolti dal dispositivo indossabile durante movimenti analoghi alla posa in analisi.

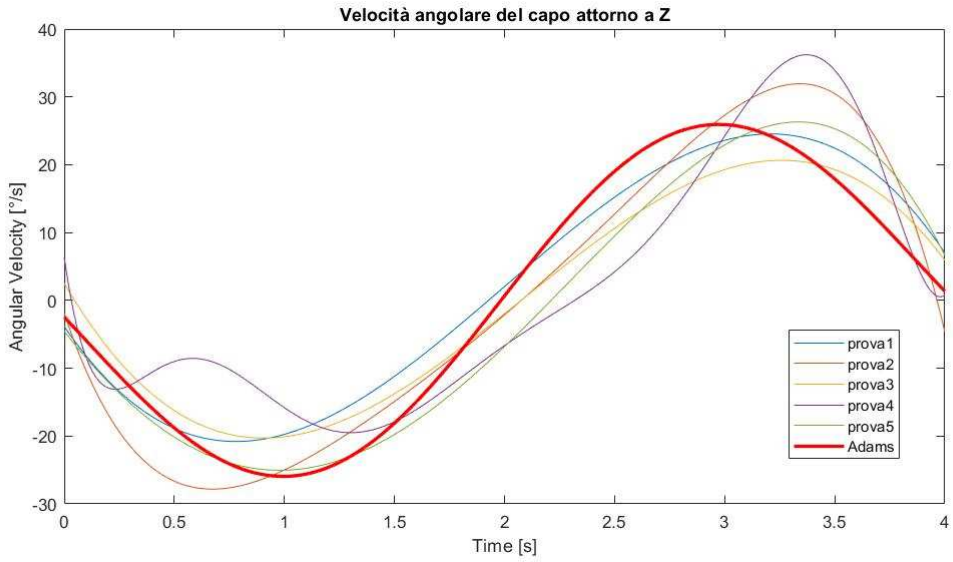


Figura 51 - Velocità angolare del capo attorno a Z – posa 2a

Dunque, per l'analisi della posa 2a si impiegheranno i valori di velocità angolare rilevati dal giroscopio, più rappresentativi dei dati dell'accelerometro. Dato che tali parametri vengono raccolti ogni 20 ms, per molte analisi, come quella di cui si tratterà al Capitolo 3, è necessario operare con i valori efficaci. A tal fine, si effettua quindi anche un controllo che i valori RMS legati alle prove sperimentali siano comparabili a quello ottenibile dai risultati di Adams/View. Il valore efficacie della velocità angolare del capo attorno all'asse Z è calcolabile come segue:

$$RMS_z = \sqrt{\frac{\sum_{i=1}^{N} \omega_{zi}^2}{N}}$$

Dove ω è la velocità angolare. In Tabella 10 sono riportati tali valori efficaci: come si può notare, si ha una corrispondenza piuttosto buona tra risultati sperimentali e risultati di simulazione. A seconda della prova sperimentale di riferimento, si vede che il valore RMS varia anche di alcune unità. Ciò nonostante, si tratta sempre di valori nell'intorno del valore efficacie ottenibile dal simulatore Adams/View.

Tabella 10 – Valori efficaci di velocità angolare attorno a Z del capo per posa 2a

Prova 1	Prova 2	Prova 3	Prova 4	Prova 5	Dati Adams/View
16.7456	20.7040	14.7986	17.4401	18.7519	18.4527

Infine, per verificare che non si commettano errori significativi nel trascurare le velocità angolari rilevate dalla IMU attorno agli assi X e Y, si effettua un confronto del valore efficacie del modulo della velocità angolare. In particolare,

per i dati di Adams/View, essendo sul simulatore le velocità angolari rilevate attorno a X e a Y perfettamente nulle, si ha:

$$RMS_{modulo} = RMS_z$$

Per i risultati sperimentali, avendo del disturbo anche sugli assi X e Y, è necessario calcolare il valore efficacie del modulo delle velocità angolari come segue:

$$RMS_{modulo} = \sqrt{\frac{\sum_{i=1}^{N} \omega_{modulo\_i}^2}{N}}$$

Dove:

$$\omega_{modulo} = \sqrt{{\omega_x}^2 + {\omega_y}^2 + {\omega_z}^2}$$

Da tale analisi, si ottengono i valori riportati in Tabella 11. Come prova sperimentale si è presa di riferimento la prova 5, ma i risultati ottenibili sono analoghi anche per le altre prove effettuate.

Tabella 11 – Confronto del valore efficacie della velocità angolare attorno diversi assi per posa 2a

	RMS ω_x	RMS ω_y	RMS ω_z	RMS ω_{modulo}
Dati Adams/View	-	-	18.4498	18.4498
Prova sperimentale	1.6161	1.4291	18.7519	18.8757

L'approccio utilizzato sul modello multibody di simulare la sola velocità angolare attorno a Z risulta dunque adeguato, poiché il valore efficacie del modulo della velocità angolare è comparabile con quello legato a prove sperimentali. È a tal punto possibile considerare validato il modello multibody per la posa 2a.

2.4.3 Validazione del modello: Posa 2b

Si procede a tal punto con la validazione del modello multibody utilizzato per la simulazione della posa 2b. Come per la posa 2a, è necessario verificare:

+ Forza di contatto piede – terreno;
+ Velocità angolare del capo.

Verifica della forza di contatto piede - terreno.

Analogamente ai movimenti precedenti, è anche in tal caso necessario effettuare un controllo della forza di contatto dei piedi dell'umanoide con il suolo. Infatti, affinché si possa considerare validato il modello, è necessario che la reazione vincolare complessiva del terreno sia all'incirca pari a $m \cdot g$.

Ipotizzando, nuovamente, di avere a che fare con un individuo di 75 kg, quindi:

$$GRF_{tot} = m \cdot g = 75[kg] \cdot 9.80665[m/s^2] = 735.499N$$

Da cui la forza di contatto di ciascun piede con il pavimento (pari alla somma della forza di contatto dell'avampiede e quella del retropiede) vale:

$$GRF_{sing\_piede} = GRF_{tot}/2 = 367.749\ N$$

Come si può vedere dal grafico riportato in Figura 52, nonostante la ripartizione delle forze tra avampiede e retropiede vari nel tempo (a seconda delle diverse fasi del movimento in esame), la somma delle due fornisce approssimativamente la GRF totale desiderata. Le variazioni di forza complessivamente scaricata nel tempo, già presenti nel precedente caso, possono essere dovute alle approssimazioni del modello e al movimento cui è soggetto l'individuo nel tempo. Ciò nonostante, si ottiene una buona approssimazione del risultato desiderato, per cui si può considerare validato il modello.

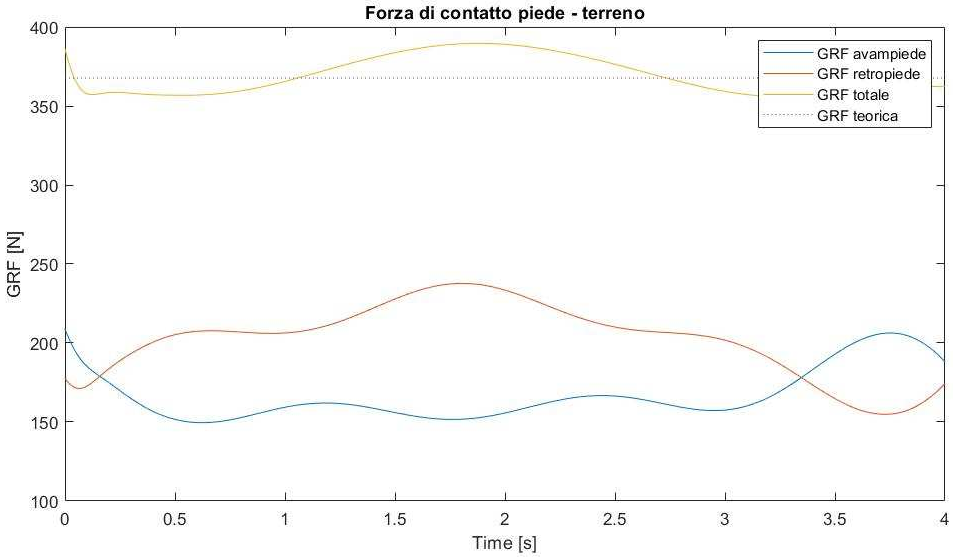


Figura 52 - Forza di contatto piede - terreno per posa 2b

Analogamente al caso riportato al paragrafo 2.4.2, è importante verificare che il modello sia adeguato alla simulazione non solo del movimento in esame ma anche del movimento con presa di oggetti da terra. A tal proposito, è quindi necessario introdurre una forza in direzione verticale verso il basso alle mani, che sia nulla nella prima metà del movimento e pari a $N = M \cdot g$ nella seconda metà dello stesso (dove M è il peso, in kg, dell'oggetto e N è la forza peso associata). Si verifica in seguito che il post – processor di Adams/View percepisca correttamente tale forza controllando che il nuovo grafico della forza di contatto piede - suolo veda un salto nel momento in cui si afferra l'oggetto, discontinuità pari a $N = M \cdot g$.

Come per la posa 2a, in caso di oggetto di massa pari a 20 kg e di presa dell'oggetto dopo 2s dall'inizio del movimento, la forza di contatto con il suolo dovrebbe essere pari a:

$$\begin{cases} GRF_{sing\_piede} = (m_{INDIVIDUO} \cdot g)/2 = 367.749\ N & se\ t \leq 2s \\ GRF_{sing\_piede} = (m_{INDIVIDUO} + m_{OGGETTO}) \cdot g/2 = 465.816\ N & se\ t > 2s \end{cases}$$

Analogamente al caso precedente, questo è ottenibile applicando a ciascuna mano una forza concentrata così definita:

```
step(x, x0, h0, x1, h1)
```

Dove x è il tempo, x0 l'ultimo istante in cui la forza è pari al primo valore h0 e x1 il primo istante in cui la forza è pari al secondo valore h1. Dunque, nello specifico, x0 = 2s, h0 = 0, x1 = 2.001s e h1 = N/2. Nuovamente, il peso dell'oggetto si ripartisce equamente tra le due mani, per questo h1 = N/2.

Si può notare dal grafico riportato alla Figura 53 che effettivamente, nell'intorno dei 2s, si ha la discontinuità nella forza di contatto piede – terreno ricercata. Tale discontinuità risulta all'incirca pari a 98 N, come desiderato. Infatti, la curva gialla della GRF totale varia continuamente nell'intorno della curva tratteggiata rappresentante la GRF teorica. Il massimo della curva che si verifica subito dopo l'applicazione della forza è probabilmente dovuto ad incertezze ed approssimazioni del modello multibody. Ciò nonostante, risulta verificata la possibilità di simulare la presa di un oggetto di massa M da terra anche per la posa 2b.

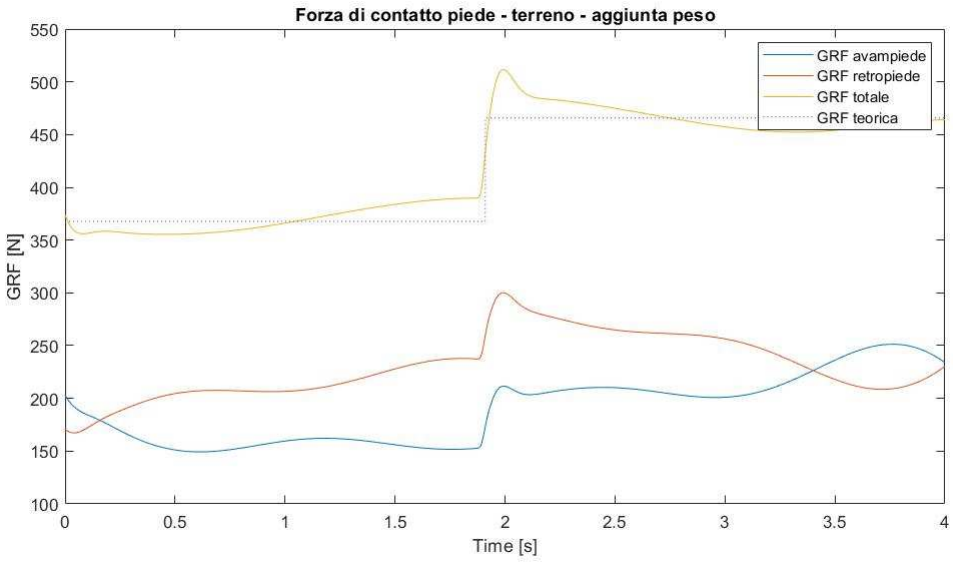


Figura 53 - Forza di contatto piede - terreno per posa 2b con peso

Verifica della velocità angolare del capo.

Per poter utilizzare il modello multibody per predire le indicazioni sul movimento dell'operatore rilevate dal sensore IMU situato sul casco sensorizzato, è necessario dapprima validare lo stesso sulla base delle velocità angolari del capo.

Una volta effettuata la procedura di raccolta dei dati sperimentali utilizzando il dispositivo descritto al paragrafo 642.4 e alla Figura 35 - Dispositivo utilizzato per la raccolta dei dati sperimentali, si procede con l'elaborazione di tali dati e la loro comparazione con i valori di Adams/View. Infatti, è dapprima necessario interpolare i risultati sperimentali con delle serie di Fourier o dei polinomi in maniera tale da rimuovere il rumore caratteristico della sperimentazione. Anche in questo caso, come per la posa 2a, si è optato per l'utilizzo dei polinomi e il numero di prove sperimentali analizzate è pari a cinque. In particolare, per quanto riguarda il grado del polinomio richiesto per una adeguata rappresentazione dei risultati sperimentali, tutte le prove sono state interpolate con polinomi di quinto grado, fuorché la prova 4 per la quale è risultato sufficiente il terzo grado di polinomio. Oltre all'interpolazione, è poi stata necessaria un'ulteriore elaborazione, ossia una leggera scalatura delle curve. Infatti, tutte le prove sono caratterizzate da una durata nell'intorno dei 4s ma, per avere le curve perfettamente sovrapposte, è stato necessario scalare proporzionalmente i grafici in maniera da avere una durata di tutte le prove esattamente pari a 4s. Se, ad esempio, una prova ha una durata pari a X secondi, l'asse dei tempi è scalato come segue:

$$tempo = tempo \cdot \frac{4\ s}{X\ s}$$

Mentre l'asse delle ordinate, ossia l'asse delle velocità angolari, nella maniera seguente:

$$vel\_angolare = vel\_angolare \cdot \frac{X\ s}{4\ s}$$

Tale operazione è necessaria perché se si aumenta il tempo la velocità angolare diminuisce e viceversa. La scalatura dei grafici è stata così effettuata anche per le altre pose.

Rispetto alla posa 2a, ci si aspetta ora di avere velocità angolari maggiori poiché, nello stesso tempo, il capo subisce una rotazione maggiore. Infatti, durante il movimento 2a si piegano principalmente le gambe, per cui il busto, e insieme ad esso il capo, subisce una rotazione ridotta, mentre durante la posa 2b è il busto ad effettuare la rotazione maggiore, e di conseguenza anche il capo (che si considera fisso rispetto al busto).

Anche per la posa 2b, si considera la sola rotazione attorno all'asse Z, ossia quella contenuta nel piano sagittale X – Y. Nel seguito verrà poi dimostrato che la scelta di trascurare le velocità angolari attorno a X e Y risulta comunque adeguata in termini di valori efficaci delle velocità angolari, grandezze cui si è maggiormente interessati per l'analisi dei movimenti caratteristici sui luoghi di lavoro.

Di seguito, in Figura 54, si riporta il confronto tra le velocità angolari registrate durante le cinque prove sperimentali e quella fornita dal post – processor di

Adams/View, conseguenza delle equazioni del moto fornite in input e definite al paragrafo 2.3.5. Come si nota, l'andamento fornito dal simulatore è ben rappresentativo delle prove sperimentali, tutte caratterizzate da tale andamento simil – sinusoidale. Il modello multibody è quindi settato correttamente al fine di rappresentare la velocità angolare del capo nel tempo.

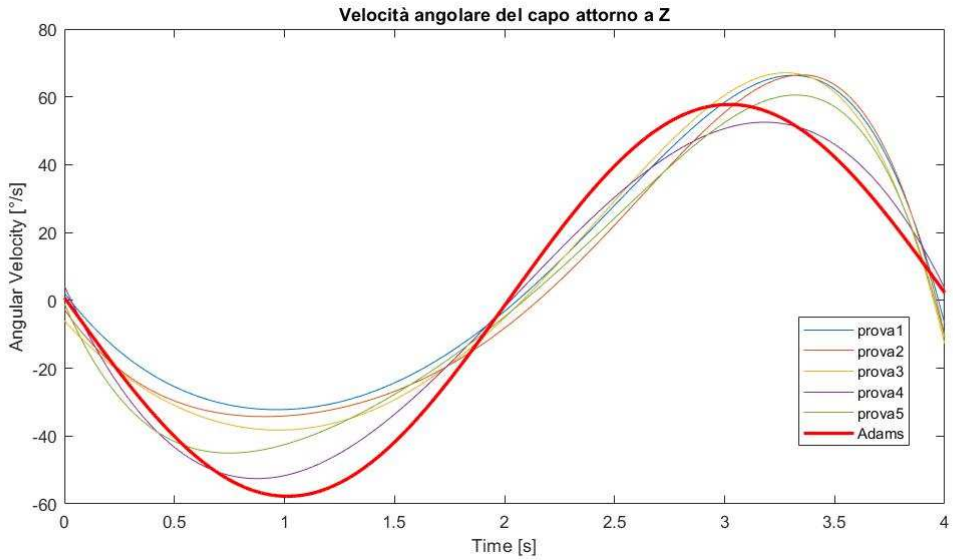


Figura 54 - Velocità angolare del capo attorno a Z - posa 2b

Come precedentemente esposto, nell'analisi dei segnali di accelerazione lineare e velocità angolare si è interessati ai valori efficaci delle diverse grandezze. Affinché si possa dunque correttamente impiegare il modello multibody, non è sufficiente avere gli andamenti delle velocità angolari nel tempo corrispondenti a quelli sperimentali: è anche importante il controllo dei valori RMS. Per la posa 2b, si considera la sola rotazione attorno l'asse Z, per cui l'unico valore efficacie da considerare sarà:

$$RMS_z = \sqrt{\frac{\sum_{i=1}^{N} \omega_{zi}^2}{N}}$$

Dove ω è la velocità angolare. I differenti valori efficaci associati alle prove sperimentali e al modello multibody sono riportati in Tabella 12. Il valore RMS legato all'analisi multibody ben caratterizza quelli sperimentali, nonostante esso sia leggermente maggiore. Questo è legato al fatto che il modello multibody è tale da avere il capo dell'umanoide rigidamente connesso al busto, mentre sperimentalmente quando si piega di molto il busto il capo lo si ruota leggermente meno. Ciò nonostante, la differenza risulta limitata, per cui il modello si può considerare rappresentativo della realtà.

Tabella 12 - Valori efficaci di velocità angolare attorno a Z del capo per posa 2b

Prova 1	Prova 2	Prova 3	Prova 4	Prova 5	Dati Adams/View
35.9040	36.1024	37.9634	37.5272	36.9282	40.9259

Per ultimo, è necessario verificare che gli errori commessi nel trascurare le velocità angolari attorno a X e Y siano ridotti. A tal fine, si effettua il confronto del valore efficacie del modulo della velocità angolare. Come per la posa 2a, i dati di Adams/View sono tali per cui:

$$RMS_{modulo} = RMS_z$$

Invece, per i risultati sperimentali, è necessario considerare anche le velocità angolari attorno a X e Y (se pur ridotte rispetto quelle attorno a Z). Il valore efficacie si calcola in tal caso sulla base del modulo della velocità angolare come segue:

$$RMS_{modulo} = \sqrt{\frac{\sum_{i=1}^{N} \omega_{modulo\_i}^2}{N}}$$

Dove il modulo della velocità angolare si valuta come:

$$\omega_{modulo} = \sqrt{{\omega_x}^2 + {\omega_y}^2 + {\omega_z}^2}$$

I valori che si ottengono sono riportati in Tabella 13. La prova sperimentale che si prende come riferimento è la 4, ma il discorso è analogo anche per le altre prove.

Tabella 13 – Confronto del valore efficacie della velocità angolare attorno diversi assi per posa 2b

	RMS ω_x	RMS ω_y	RMS ω_z	RMS ω_{modulo}
Dati Adams/View	-	-	40.9259	40.9259
Prova sperimentale	2.4565	2.7831	37.5272	37.7104

Come si nota, il valore efficacie del modulo della velocità angolare relativo alla prova sperimentale si scosta di poco dal valore efficacie della sola velocità angolare attorno a Z. L'approccio utilizzato di trascurare le rotazioni attorno agli assi X e Y risulta dunque opportuno. Effettuata quest'ultima verifica, si può considerare validato il modello multibody per la posa 2b anche dal punto di vista della velocità angolare al capo.

2.4.4 Validazione del modello: Posa 3

Per ultimo, si procede con la validazione del modello multibody relativo alla posa 3. Come per le pose 2a e 2b, è essenziale verificare il modello sulla base di:

+ Forza di contatto piede – terreno;
+ Velocità angolare del capo.

Verifica della forza di contatto piede – terreno.

In maniera analoga ai movimenti analizzati in precedenza, è necessario validare il modello controllando che la forza di contatto con il pavimento risulti pari a $m \cdot g$. Si tratta infatti di un movimento piuttosto statico, caratterizzato da piedi fissi rispetto al ground, per cui la GRF dovrà risultare pari alla forza peso. Rispetto ai casi precedenti, però, ci si aspettano degli andamenti più costanti, poiché il peso viene ripartito tra avampiede e retropiede sempre allo stesso modo con lo scorrere del tempo (il baricentro ora non cambia posizione nel tempo, il movimento dell'individuo è molto meno marcato). Nel caso in cui l'individuo pesi 75 kg, la forza totale scaricata al suolo (pari alla reazione vincolare del pavimento stesso) è:

$$GRF_{tot} = m \cdot g = 75[kg] \cdot 9.80665[m/s^2] = 735.499N$$

La quale deve essere ripartita tra i due piedi, per cui:

$$GRF_{sing\_piede} = GRF_{tot}/2 = 367.749\ N$$

Il piede è però diviso in avampiede e retropiede: la somma delle forze scaricate da ciascuna porzione dovrà dunque essere pari a GRF_{sing_piede}. In effetti, se si va a rappresentare l'andamento della forza scaricata dall'avampiede (in blu), della forza scaricata dal retropiede (in rosso), della somma delle due (in giallo) e della GRF_{sing_piede} teorica (linea tratteggiata), si ottiene il grafico in Figura 55.

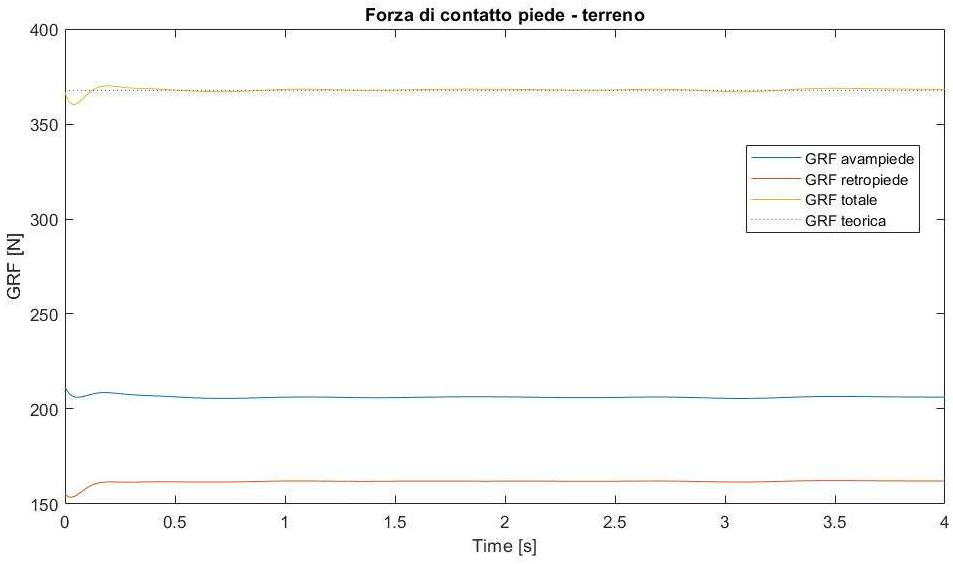


Figura 55 - Forza di contatto piede - terreno per posa 3

La porzione iniziale del grafico, caratterizzata da un'oscillazione nell'andamento, è probabilmente dovuta ad errori insiti nel software legati all'inizio del movimento. Ciò nonostante, come si nota, la somma delle due forze scaricate a terra è molto prossima alla linea tratteggiata della forza di reazione teorica, per cui si può ritenere validato il modello.

Come per le due precedenti pose, si analizza poi il caso in cui l'operatore debba movimentare un peso. In tale situazione, per come definito il movimento, si immagina che l'umanoide ruoti il busto verso sinistra, prenda l'oggetto, ruoti il busto verso destra per spostare il peso su un pianale alla sua destra, posi l'oggetto, e torni nella posizione di partenza. Vi saranno quindi tre fasi:

1. L'operatore dalla posizione frontale ruota il busto verso sinistra;
2. L'individuo afferra l'oggetto, lo sposta da sinistra a destra ruotando il busto, e poi lascia il peso;
3. L'operatore ruota il busto per tornare nella posizione di partenza.

Affinché il modello multibody sia adeguatamente costruito, è necessario verificare che il simulatore gestisca opportunamente l'aggiunta del peso appena definita. A tal proposito, si introduce una forza diretta verticalmente verso il basso alle mani, per tutta la durata della fase 2. Tale forza ha modulo pari a $N = M \cdot g$ (dove M è il peso dell'oggetto, in kg, e N la sua forza peso). Per validare il modello bisogna verificare che nel post – processor di Adams/View la forza di contatto piede – suolo percepisca tale variazione di forza peso totale.

Nello specifico, preso un oggetto di 20 kg, ci si aspetta di ottenere una forza di contatto piede - terreno che abbia il seguente andamento variabile nel tempo:

$$\begin{cases} GRF_{sing\_piede} = (m_{INDIVIDUO} \cdot g)/2 = 367.749\ N & se\ t < 1.2\ s \\ GRF_{sing\_piede} = (m_{INDIVIDUO} + m_{OGGETTO}) \cdot g/2 = 465.816\ N & se\ 1.2\ s \leq t \leq 2.8\ s \\ GRF_{sing\_piede} = (m_{INDIVIDUO} \cdot g)/2 = 367.749\ N & se\ t < 2.8\ s \end{cases}$$

Dove 1.2s è l'istante in cui si ipotizza di prendere il peso e 2.8s l'istante in cui esso viene posato.

Come per le pose precedenti, questa forza variabile nel tempo è stata ottenuta mediante l'utilizzo della funzione step. In questo caso, tuttavia, poiché si ha una doppia variazione di forza, si sono utilizzate due funzioni step nel modo seguente:

```
  step(time, 1.2, 0, 1.201, N/2)+
step(time, 2.8, 0, 2.801,    -N/2)
```

La prima funzione imposta dunque il valore della forza a N/2 all'istante 1.2 s, mentre la seconda annulla la forza applicata all'istante 2.8 s. Questa forza agisce su entrambe le mani, diretta verso il basso, per cui complessivamente la forza applicata sarà pari a $N = M \cdot g$.

Nella Figura 56 è riportato l'andamento della forza di contatto piede – terreno in questo modo ottenuto. Si nota la discontinuità desiderata agli istanti 1.2 s e 2.8 s, pari, all'incirca, a 98.0665 N ($= M \cdot g \,/2 = 10 \cdot 9.80665$). All'inizio di ogni fase, vi sono delle oscillazioni nelle curve: si tratta probabilmente di incertezze e approssimazione del modello multibody legate alla variazione delle condizioni operative del modello. Ciò nonostante, per la maggior parte della durata del movimento, la GRF complessiva (pari alla somma della GRF dell'avampiede e della GRF del retropiede) risulta molto prossima alla forza di contatto piede – terreno teorica (linea tratteggiata). Il modello si può dunque considerare validato e adeguato alla simulazione dello spostamento di oggetti da un piano di lavoro ad un altro.

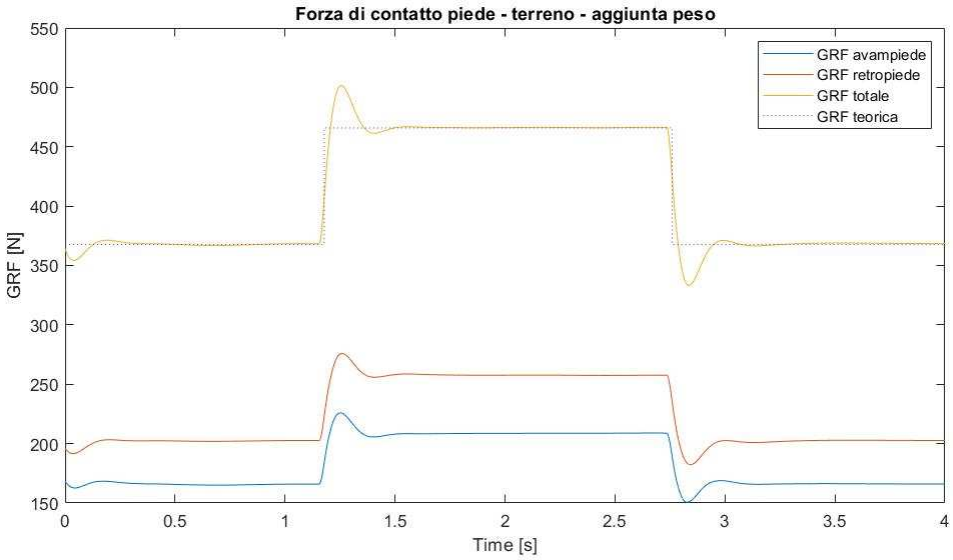


Figura 56 - Forza di contatto piede - terreno per posa 3 con peso

Verifica della velocità angolare del capo.

Come per le pose 2a e 2b, si caratterizza il movimento mediante la velocità angolare del capo, e non l'accelerazione lineare. In questo caso, a maggior ragione, l'utilizzo delle accelerazioni risulta inadeguato poiché sul modello multibody, realizzando la sola torsione del busto, il capo è fisso nello spazio per cui caratterizzato da accelerazioni lineari nulle in tutte e tre le direzioni. È quindi necessario verificare che le velocità angolari caratteristiche ottenute dal simulatore siano comparabili con quelle ottenute sperimentalmente.

Per la posa 3, si fa riferimento alla sola velocità angolare del capo attorno all'asse verticale Y. Infatti, le leggi del moto inserite all'interno del modello multibody sono tali da permettere la sola torsione del busto, per cui le velocità angolari attorno a X e Z risultano necessariamente nulle sul simulatore. Si verificherà in seguito che questa approssimazione risulta adeguata anche nel caso dei risultati sperimentali.

Come per le pose precedentemente analizzate, è dapprima necessario effettuare l'elaborazione dei dati sperimentali. Una volta registrati i dati del giroscopio relativi a più prove sperimentali, si procede dapprima con la loro interpolazione. Questa azione è essenziale per rimuovere il rumore caratteristico dei dati sperimentali ed individuare una tendenza comune a tutte le curve. Come per la posa 1, in tal caso si utilizzano le serie di Fourier per l'interpolazione, più adeguate agli andamenti che si devono ottenere rispetto ai polinomi. Per tutte e sei le prove sperimentali analizzate, in particolare, il sesto grado per la serie di Fourier risulta essere il più adeguato ad una interpolazione accurata e precisa. Infine, come per le altre pose, è necessario scalare leggermente i grafici, in modo da avere tutte le prove sperimentali che durano esattamente 4 s come la prova Adams/View, in maniera tale da poter comparare più facilmente le diverse curve. Le curve vengono scalate proporzionalmente come illustrato al paragrafo 2.4.3: riducendo i tempi aumentano le velocità e viceversa.

Per come definita la posa 3, ci si aspetta una velocità angolare dapprima positiva, poi negativa, e infine nuovamente positiva. Infatti, il verso positivo dell'asse Y è verso l'alto per cui sul piano X – Z le rotazioni positive sono in verso antiorario e quelle negative in verso orario. Nella fase 1 in cui l'operatore ruota il busto verso sinistra, la rotazione è antioraria, dunque la velocità angolare sarà positiva. Durante la fase 2, invece, l'individuo torce il busto da sinistra a destra, in verso orario, per cui la velocità angolare è negativa. Infine, nella terza ed ultima fase, l'operatore torna in posizione frontale mediante una rotazione antioraria, da cui, nuovamente, velocità angolare maggiore a zero.

In Figura 57 sono riportate le curve rappresentanti le velocità angolari nel tempo. La curva rossa è quella ottenuta dal post – processor di Adams, mentre le altre sono quelle relative alle prove sperimentali.

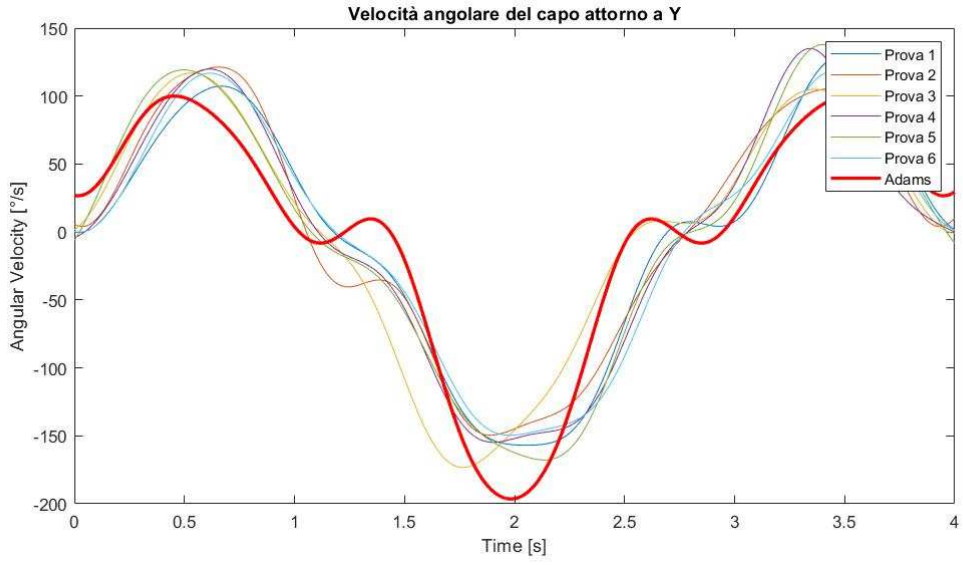


Figura 57 - Velocità angolare del capo attorno a Y - posa 3

Come si può notare, gli andamenti dei risultati sperimentali sono tutti analoghi, e ben rappresentati da quello ottenuto mediante il simulatore. Il modello multibody per la posa 3 riproduce dunque correttamente gli andamenti sperimentali della velocità angolare del capo attorno a Y nel tempo.

Oltre agli andamenti, è però anche importante il confronto dei valori efficaci delle velocità angolari del capo nel tempo. Tali quantità sono infatti fondamentali per analizzare la variazione dei segnali di velocità angolare nel tempo, per poi trarne delle conclusioni, come verrà mostrato al Capitolo 3. È per questo motivo importante confrontare i valori RMS dei risultati sperimentali con quello ottenuto dalla simulazione su Adams/View. Considerando la sola rotazione attorno l'asse verticale Y, il valore efficacie di interesse è calcolabile come:

$$RMS_y = \sqrt{\frac{\sum_{i=1}^{N} \omega_{yi}^2}{N}}$$

Dove ω è la velocità angolare. In Tabella 14 sono riportati i valori efficaci della velocità angolare attorno a Y per le diverse prove sperimentali e per la simulazione su Adams/View. La corrispondenza tra i valori RMS delle prove sperimentali e quello relativo alla simulazione è piuttosto buona. A seconda delle prove, si hanno valori efficaci che variano di anche 8 °/s: ciò è legato all'impossibilità di ruotare sperimentalmente il busto sempre alla medesima velocità (come viene invece effettuato su Adams). Malgrado ciò, si ha una buona affinità tra prove sperimentali e analisi multibody, per cui il modello si può considerare rappresentativo della realtà.

Tabella 14 - Valori efficaci di velocità angolare attorno a Z del capo per posa 3

Prova 1	Prova 2	Prova 3	Prova 4	Prova 5	Prova 6	Dati Adams/View
86.4859	85.0097	86.7465	89.1469	93.3427	85.4267	85.6079

Infine, si verifica la validità dell'ipotesi di modello multibody caratterizzato dalla sola velocità angolare attorno l'asse verticale Y. A tal proposito, si confronta il valore efficacie del modulo della velocità angolare tra simulazione e sperimentazione. Su Adams/View, il modello è costruito in maniera tale da ammettere le sole rotazioni attorno all'asse Y, per cui:

$$RMS_{modulo} = RMS_y$$

I risultati sperimentali sono invece caratterizzati dalla preponderante velocità angolare attorno a Y e da velocità angolari minori, molto disturbate, attorno a X e Z. In questo caso, il valore efficacie si calcola utilizzando il modulo della velocità angolare ω_{modulo} come mostrato nella pagina seguente.

$$RMS_{modulo} = \sqrt{\frac{\sum_{i=1}^{N} \omega_{modulo\_i}^2}{N}}$$

Dove:

$$\omega_{modulo} = \sqrt{{\omega_x}^2 + {\omega_y}^2 + {\omega_z}^2}$$

Nella Tabella 15, rappresentata nel seguito, sono riportati i valori ottenuti da Adams/View e dalla sperimentazione. In questo caso si è presa come riferimento la prova sperimentale 3, ma l'analogo si sarebbe potuto ottenere con le altre prove. Si può notare che il valore efficacie relativo alla prova sperimentale del modulo della velocità angolare permane molto prossimo a quello della sola velocità angolare attorno Y. Entrambi i valori RMS sono poi confrontabili con quello ottenuto dal simulatore. L'ipotesi di trascurare le rotazioni attorno agli assi X e Z che possono nascere durante il movimento risulta quindi adeguata.

Tabella 15 - Confronto del valore efficacie della velocità angolare attorno diversi assi per posa 3

	RMS ω_x	RMS ω_y	RMS ω_z	RMS ω_{modulo}
Dati Adams/View	-	-	85.6079	85.6079
Prova sperimentale	17.8416	86.7465	7.5353	88.8823

Con questa ultima verifica, il modello multibody per la posa 3 risulta validato, e quindi utilizzabile per predire i parametri rilevati dai sensori di movimentazione collocati sul dispositivo sensorizzato IoT.

A questo punto, completata la validazione, il modello multibody realizzato risulta rappresentativo della realtà ed utilizzabile per effettuare simulazioni in ambiente di lavoro. In particolare, nel seguente capitolo (Capitolo 3) si evidenzierà come i risultati ottenibili dalla simulazione multibody possono essere utilizzati per predire la stanchezza e affaticamento degli operatori al termine della giornata lavorativa.

Capitolo 3

Proposta di un indice di mobilità

In quest'ultimo capitolo viene studiato un indice di mobilità e affaticamento che permetta di valutare, sulla base dei dati raccolti dal dispositivo indossabile IoT progettato al Capitolo 1 e della simulazione multibody di cui al Capitolo 2, le condizioni operative e di lavoro degli operatori in un arco temporale di una giornata.

Si tratta di una stima relativa della quale possono essere effettuate iniziali valutazioni comparative.

In assenza di indicazioni di normative specifiche, la proposta dell'indice di mobilità prende spunto dagli studi e dalle normative relative allo studio del comfort. In questo ambito sono molto più frequenti gli studi in ambito ergonomico.

Al fine di individuare parametri sintetici, come base di partenza di una proposta valutativa da approfondire in successivi studi sistematici, nel presente capitolo è inizialmente valutata la distinzione tra un indice di mobilità e un indice di affaticamento. Infatti, per valutare l'effettiva stanchezza dell'operatore non si può prescindere dalla misura della frequenza cardiaca (non possibile con il casco sensorizzato progettato). Si monitora dunque il livello di attività fisica dei lavoratori sulla sola base dei dati raccolti dal giroscopio, e in tal modo si calcola un indice di mobilità. Questo è utile per una prima stima della stanchezza dell'operatore, poiché a maggiore movimento corrisponde maggiore affaticamento e viceversa. Tuttavia, per avere una stima più affidabile e realistica della stanchezza dei lavoratori alla fine del turno di lavoro, si propone una estensione al lavoro effettuato, con l'indicazione di una possibile metodologia di calcolo di un indice di affaticamento e i sensori aggiuntivi necessari.

3.1 Indice di mobilità

Utilizzando i dati raccolti dal dispositivo progettato al Capitolo 1 è possibile definire un indice di mobilità che sia rappresentativo dello stato di attività dell'operatore durante la giornata lavorativa, e quindi correlabile direttamente alla sua stanchezza a fine giornata. Si tratta di un indice calcolabile sulla base dei soli segnali registrati dal giroscopio. Dunque, risulta possibile anche calcolare tale indice sulla base delle simulazioni multibody effettuate con il modello di cui al Capitolo 2.

Come verificato in precedenza, i modelli multibody realizzati sono rappresentativi della realtà, e dunque i valori di velocità angolare/accelerazione lineare ottenibili dalla simulazione sono riconducibili a quelli registrati sperimentalmente. In particolare, la posa 1 (camminata) è stata validata sulla base delle accelerazioni lineari, mentre le altre pose in funzione della velocità angolare. In generale, sembra dunque essere più significativa la velocità angolare per la caratterizzazione del movimento, motivo per il quale essa è stata scelta per il calcolo dell'indice di mobilità. Nella Tabella 16, per le diverse pose sono riportati i valori RMS della velocità angolare ottenuti sia sperimentalmente che dalla simulazione. Il valore efficacie relativo alla fase sperimentale è riferito al modulo della velocità angolare, che considera tutte le componenti lungo le tre direzioni X, Y e Z. Il valore RMS ottenuto dalla simulazione considera invece la velocità angolare attorno ad un solo asse, quello rappresentativo della posa in analisi (si rimanda a tal proposito ai Paragrafi 2.4.2, 2.4.3, 2.4.4 per maggiore chiarezza).

Tabella 16 - Valori RMS della velocità angolare per le diverse pose analizzate

	RMS simulazione (°/s)	RMS sperimentale (°/s)
POSA 1	N.C. [6]	14.694
POSA 2A	18.45	18.879
POSA 2B	40.926	37.71
POSA 3	85.608	88.882

A questo punto, si definisce l'indice di mobilità (I.M.) come segue:

$$I.M. = \frac{\sum_{i=1}^{n} e_i \cdot t_i}{t_{TOT}}$$

[6] Per come realizzato il modello multibody, non potendo simulare i muscoli del collo e lo smorzamento distribuito presente nel corpo umano, nella posa 1 il capo non risulta ruotare. Il valore RMS della velocità angolare fornito dal simulatore è dunque nullo. Ciò è però legato a limiti di realizzazione del modello multibody, che non rappresenta a tal proposito adeguatamente la realtà. Per questo motivo, per la sola posa 1, si farà riferimento ai soli risultati sperimentali.

Dove t_i è la durata temporale dell'azione i-esima, t_{TOT} è la durata totale che si considera (tipicamente la giornata lavorativa di 8 ore), ed e_i si calcola come segue sulla base dei valori di velocità angolare:

$$e_i = \frac{\int_i \sqrt{(\omega_x{}^2 + \omega_y{}^2 + \omega_z{}^2)}(t)dt}{RMS_{|\omega|}} = \frac{\int_i |\omega(t)|dt}{RMS_{|\omega|}}$$

Dunque, e_i si calcola rapportando l'area sottesa al grafico del modulo della velocità angolare nell'intervallo i-esimo per il valore RMS dello stesso nel medesimo intervallo.

Per come definito, l'indice di mobilità risulta adimensionale. Inoltre, si considerano le durate temporali t_i e t_{TOT} per fare in modo che l'indice aumenti se l'azione che si sta analizzando ha una durata maggiore. Infatti, in caso di azione continuativa si accumula una stanchezza maggiore rispetto alla stessa azione ma intervallata da momenti di riposo.

Nelle seguenti tabelle sono riportati i valori e_i relativi alle diverse pose analizzate. Per tutte, fuorché la posa 1 per il problema illustrato in precedenza, sono mostrati i valori ottenibili sia dalla simulazione multibody che dalla sperimentazione. Come si nota, i valori e_i estrapolati dalla simulazione sono comparabili a quelli ottenuti sperimentalmente: ancora una volta, dunque, si è verificata la rappresentatività del modello multibody. I seguenti valori di e_i si riferiscono ad una singola azione di posa, della durata riportata nelle rispettive tabelle.

Tabella 17 - Calcolo e_i per posa 1

POSA 1 – sperimentale		
	1 passo (0.556 s)	10 passi (5.556 s)
RMS velocità angolare X	4.7327	4.242
RMS velocità angolare Y	4.7064	4.2958
RMS velocità angolare Z	8.438	8.3813
RMS modulo velocità angolare	10.7586	10.3293
Integrale modulo velocità angolare	5.8602	54.6219
e_i	0.5447	5.2881

Tabella 18 – Calcolo e_i per posa 2a

POSA 2A – durata di riferimento della singola azione pari a 4 s		
	Simulazione	Sperimentale
RMS velocità angolare X	-	1.6161

RMS velocità angolare Y	-	1.4291
RMS velocità angolare Z	18.4498	18.752
RMS modulo velocità angolare	18.4498	18.8757
Integrale modulo velocità angolare	66.7493	69.7281
e_i	3.6179	3.6941

Tabella 19 – Calcolo e_i per posa 2b

POSA 2B – durata di riferimento della singola azione pari a 4 s		
	Simulazione	Sperimentale
RMS velocità angolare X	-	2.4565
RMS velocità angolare Y	-	2.7831
RMS velocità angolare Z	40.9259	37.5272
RMS modulo velocità angolare	40.9259	37.7104
Integrale modulo velocità angolare	147.6723	137.7087
e_i	3.3083	3.6517

Tabella 20 – Calcolo e_i per posa 3

POSA 3 – durata di riferimento della singola azione pari a 4 s		
	Simulazione	Sperimentale
RMS velocità angolare X	-	17.8416
RMS velocità angolare Y	85.6079	86.7465
RMS velocità angolare Z	-	7.5353
RMS modulo velocità angolare	85.6079	88.8823
Integrale modulo velocità angolare	260.3805	290.7409
e_i	3.0415	3.2711

Dalla Tabella 17, essendo mostrato il calcolo di e_i sia nel caso di un singolo passo che nel caso di 10 passi, si evidenzia la necessità di considerare anche le durate temporali nel calcolo dell'indice di mobilità. Infatti, se così non fosse, l'indice avrebbe lo stesso valore sia che si facciano 10 passi continuati sia che si facciano 5 passi e poi ulteriori 5 passi dopo essersi fermati a riposare. Infatti, in tal caso si avrebbe:

$$e_{i\ -\ 10\ passi\ continuativi} \cong 10 \cdot e_{i\ -\ singolo\ passo} = 5.447$$

$$e_{i\,-\,5\,passi\,+\,5\,passi} \cong 5 \cdot e_{i\,-\,singolo\,passo} + 5 \cdot e_{i\,-\,singolo\,passo}$$
$$= 10 \cdot e_{i\,-\,singolo\,passo} = 5.447$$

Se si considerano invece, nel calcolo dell'indice, anche le durate relative di ogni azione, è possibile differenziare i due casi appena illustrati, e dunque quantificare la maggiore stanchezza derivante da una azione continuativa rispetto ad una azione frazionata. Infatti:

$$I.M._{10\,passi\,continuativi} \cong 10 \cdot e_{i\,-\,singolo\,passo} \cdot \frac{5.556\,s}{5.556\,s} = 5.447$$

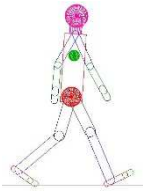
$$I.M._{5\,passi\,+\,5\,passi} \cong 5 \cdot e_{i\,-\,singolo\,passo} \cdot \frac{2.778\,s}{5.556\,s} + 5 \cdot e_{i\,-\,singolo\,passo} \cdot \frac{2.778\,s}{5.556\,s}$$
$$= 5 \cdot e_{i\,-\,singolo\,passo} = 2.7235$$

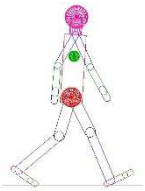
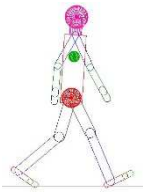
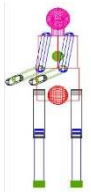
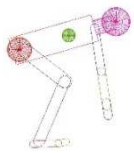
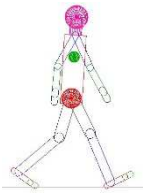
L'indice di mobilità risulta in tal modo non solo rappresentativo dello stato di attività dell'operatore ma anche riconducibile all'entità della stanchezza accumulata dallo stesso dopo una giornata di lavoro.

In Tabella 21 è riportato un esempio di calcolo dell'indice di mobilità nel caso di una giornata lavorativa di 8 ore composta dalle attività ivi illustrate. Si tratta di un semplice esempio, utile a capire l'ordine di grandezza dell'indice di mobilità che ci si può aspettare da un turno di lavoro giornaliero. Per il calcolo dell'I.M. di ogni attività si è proceduto come segue:

1. Determinazione della durata di ciascuna attività.
2. Calcolo del numero di "pose singole" effettuate durante l'intervallo temporale in analisi.
3. Determinazione dell'I.M. come prodotto del numero di singole pose effettuate, del valore di e_i relativo alla singola posa, e del rapporto t_i/t_{TOT}. In tal caso, t_i è la durata dell'attività determinata al punto 1. e t_{TOT} è pari a 8 h (= 480 min).

Tabella 21 - Esempio di calcolo dell'indice di mobilità per una giornata lavorativa di 8 ore

ATTIVITÀ	DURATA	CALCOLO	VALORE I.M.
1. POSA 1 	1 h	1h = 3600 s → 6480 passi $6480 \cdot e_{i\,-\,sing\,passo} \cdot \frac{1\,h}{8\,h}$ $= 6480 \cdot \frac{0.5447}{8}$	441.207
2. RIPOSO	40 min	-	0

3. POSA 2a 	20 min	20 min = 1200 s → 300 pose $300 \cdot e_{i\,-\,posa2a} \cdot \frac{20\ min}{480\ min} = 300 \cdot 3.6179 \cdot \frac{20}{480}$	45.223
4. POSA 1 	30 min	30 min = 1800 s → 3240 passi $3240 \cdot e_{i\,-\,sing\ passo} \cdot \frac{0.5\ h}{8\ h} = 3240 \cdot 0.5447 \cdot \frac{0.5}{8}$	110.302
5. RIPOSO	45 min	-	0
6. POSA 1 	30 min	30 min = 1800 s → 3240 passi $3240 \cdot e_{i\,-\,sing\ passo} \cdot \frac{0.5\ h}{8\ h} = 3240 \cdot 0.5447 \cdot \frac{0.5}{8}$	110.302
7. RIPOSO	45 min	-	0
8. POSA 3 	1 h	1 h = 3600 s → 900 pose $900 \cdot e_{i\,-\,posa3} \cdot \frac{1\ h}{8\ h} = 900 \cdot 3.0415 \cdot \frac{1}{8}$	342.169
9. POSA 2b 	30 min	30 min = 1800 s → 450 pose $300 \cdot e_{i\,-\,posa2b} \cdot \frac{30\ min}{480\ min} = 450 \cdot 3.3083 \cdot \frac{30}{480}$	93.046
10. RIPOSO	1 h	-	0
11. POSA 1 	1 h	1h = 3600 s → 6480 passi $6480 \cdot e_{i\,-\,sing\ passo} \cdot \frac{1\ h}{8\ h} = 6480 \cdot \frac{0.5447}{8}$	441.207
TOTALE	8 h	$I.M._1 + I.M._2 + \ldots + I.M._{11}$	1583.456

Dunque, dopo una giornata lavorativa di 8 h in cui più della metà del tempo è passata a svolgere attività di vario tipo, l'indice di mobilità risultante è di 1600 circa.

Sarebbe a tal punto opportuno analizzare differenti tipologie di giornate lavorative reali, in maniera tale da individuare dei range di variazione dell'indice di mobilità a cui associare un livello basso – medio – alto di movimento, e dunque stanchezza, al termine del turno di lavoro.

3.2 Indice di affaticamento

Per come definito, l'indice di mobilità fornisce indicazione dello stato di attività dell'operatore durante la giornata di lavoro, ma non è direttamente correlabile al suo stato di stanchezza al termine di essa. Per questo motivo si propone la definizione di un indice di affaticamento, che tenga effettivamente conto della fatica effettuata dal lavoratore durante il turno di lavoro.

L'analisi dell'affaticamento di un individuo non può però prescindere dall'analisi della sua frequenza cardiaca e della temperatura della superficie corporea. Infatti, nel momento in cui si compiono sforzi fisici sia la frequenza cardiaca che la temperatura della superficie corporea tendono ad aumentare. Per poter calcolare l'indice di affaticamento a fine giornata, è quindi necessario poter monitorare continuamente anche tali due parametri. Per la frequenza cardiaca è necessario integrare al dispositivo sensorizzato di cui al Capitolo 1 un opportuno sensore di monitoraggio. Alcuni spunti su sensori di monitoraggio della frequenza cardiaca sono riportati nel seguito, al paragrafo 3.2.1. Il monitoraggio della temperatura della superficie corporea è invece permesso dal sensore di temperatura Grove già integrato alla scheda Arduino Nano 33 IoT del casco di protezione sensorizzato progettato.

Sulla base di ciò, è possibile definire l'indice di affaticamento (I.A.) come segue:

$$I.A. = \sum_{i=1}^{n} e_i \cdot \frac{\Delta T}{T_{REF}} \cdot j \cdot \frac{\Delta f}{f_{REF}}$$

Dove:

+ e_i è il rapporto calcolabile sulla base delle velocità angolari e illustrato al Paragrafo 3.1, ossia:

$$e_i = \frac{\int_i \sqrt{\left(\omega_x{}^2 + \omega_y{}^2 + \omega_z{}^2\right)}(t)dt}{RMS_{|\omega|}} = \frac{\int_i |\omega(t)|dt}{RMS_{|\omega|}}$$

+ ΔT è la variazione di temperatura della superficie corporea registrata nell'intervallo i-esimo;

+ T_{REF} è una temperatura di riferimento della superficie corporea, rappresentativa dello stato di quiete dell'operatore. Essa è dell'ordine dei 35/37 °C, ma deve essere tarata in base all'operatore, a riposo;
+ j è un indice di massa corporea, dipendente dal tipo di individuo in analisi, che tiene conto della diversità di risposta alla fatica di ciascun individuo;
+ Δf è la variazione di frequenza cardiaca registrata nell'intervallo i-esimo;
+ f_{REF} è una frequenza cardiaca di riferimento, rappresentativa dello stato di quiete dell'operatore. Essa deve essere tarata in base all'individuo, a riposo.

Dunque, per come definito l'indice di affaticamento, attività alle quali sono associate variazioni nulle di frequenza cardiaca o temperatura della superficie corporea non contribuiscono all'aggravamento della condizione di stanchezza dell'operatore. Invece, attività durante le quali si registra una variazione non nulla di frequenza cardiaca o di temperatura superficiale del corpo fanno variare l'indice in maniera proporzionale sia all'entità del movimento registrato, sia all'entità di variazione di frequenza cardiaca e temperatura misurate. In particolare, nel caso di Δf o ΔT minori a zero il contributo all'indice di affaticamento è negativo, poiché variazioni negative di frequenza cardiaca o temperatura della superficie corporea indicano una diminuzione di affaticamento del lavoratore. Viceversa, Δf o ΔT positivi, indicando un aumento di stanchezza dell'individuo, forniscono un contributo positivo all'indice di affaticamento, facendolo aumentare.

3.2.1 Sensore di monitoraggio della frequenza cardiaca

Per permettere il monitoraggio della frequenza cardiaca dell'operatore, è necessaria l'aggiunta di un opportuno sensore. I dispositivi che permettono la rilevazione del battito cardiaco attualmente presenti sul mercato sono di vario tipo. Nel seguito si mostrano tre tipologie di dispositivi per il monitoraggio della frequenza cardiaca che si potrebbero integrare al casco sensorizzato realizzato al Capitolo 1 per permettere la stima di un indice di affaticamento. Queste sono:

+ Fascia da braccio/polso. Si tratta di dispositivi molto utilizzati quotidianamente, poiché economici e quindi facilmente integrabili in ulteriori dispositivi quali smartwatch. Risultano confortevoli, poco ingombranti ed economici (Figura 58a). Per il funzionamento, tali dispositivi impiegano sensori ottici che, tramite delle luci LED, analizzano il volume del sangue di passaggio nelle vene, variabile a ogni pulsazione. Tuttavia, la misura del battito cardiaco dal polso risulta avere una accuratezza limitata [41, 42].
+ Fascia per il capo. Si tratta di dispositivi di nuova generazione, che promettono di monitorare la frequenza cardiaca con un'accuratezza comparabile a quella dell'elettrocardiogramma. Essi sono costituiti da una

fascia indossabile sul capo con all'interno due sensori ottici, analoghi a quelli delle fasce da braccio/polso, che devono essere posizionati in prossimità delle tempie (Figura 58b). Rispetto al polso, tuttavia, le tempie sembrano essere punti migliori per il monitoraggio del battito cardiaco basato sull'utilizzo di luci LED [41].

+ Fascia toracica. Questo dispositivo, a differenza dei precedenti, opera reagendo agli impulsi elettrici del cuore, come un vero elettrocardiogramma. Si tratta quindi di uno strumento molto preciso e accurato, oltre che confortevole (Figura 58c). Rispetto alle fasce da braccio/polso e per il capo, presenta una maggiore accuratezza poiché la misurazione dei sensori ottici è fortemente influenzata da numerose variabili, quali peli, tatuaggi, ecc.. [42].

Figura 58 - Dispositivi per il monitoraggio della frequenza cardiaca. (a) Fascia da braccio/polso, (b) fascia per il capo, (c) fascia toracica.

Ciascuno di questi dispositivi potrebbe essere integrato al sistema illustrato al Capitolo 1 facendolo comunicare, per l'invio dei dati, con una delle due schede Arduino tramite Bluetooth. In questo modo, il valore della frequenza cardiaca registrato verrebbe in un secondo momento inviato al computer di controllo mediante Wi-Fi.

Conclusioni

Il presente lavoro di tesi ha permesso la progettazione di un prototipo di dispositivo indossabile IoT per il monitoraggio in ambiente di lavoro. Questo è il punto di partenza di un progetto più ampio che ha l'obiettivo di realizzare nuovi dispositivi di sicurezza indossabili da utilizzare costantemente sui luoghi di lavoro e che permettano di preservare la sicurezza dei lavoratori.

L'utilizzo delle schede Arduino per la raccolta e trasferimento dei dati dei sensori ad un calcolatore centrale ha permesso la realizzazione di un casco sensorizzato accessibile, economico. Ovviamente, ciò si rispecchia nella limitata robustezza del sistema, che alle volte si interrompe non riuscendo a ristabilire la connessione Bluetooth tra le due schede Arduino. Ciò nonostante, il dispositivo rappresenta un'ottima base di partenza per ulteriori sviluppi, poiché in grado di monitorare numerosi parametri, relativi sia all'ambiente che allo stato di attività e salute dell'operatore, e trasmettere tutte le informazioni ad un calcolatore centrale per le successive analisi ed elaborazioni. Inoltre, la potenza del casco di protezione sensorizzato progettato sta nella sua capacità di rilevare impatti al capo e cadute, anomalie negli andamenti dei parametri registrati e il corretto posizionamento del dispositivo di protezione, permettendo la visualizzazione di allarmi in tempo reale in caso di incidenti o situazioni rischiose/anomale. Infine, l'integrazione del sistema di avviso di prossimità permette di arricchire ulteriormente il dispositivo, permettendogli il monitoraggio delle distanze relative uomo – macchina e il controllo del rispetto delle minime distanze di sicurezza tra i due.

Lo studio prototipale del dispositivo indossabile IoT è ulteriormente rafforzato dall'analisi multibody che l'ha accompagnato. Infatti, essa ha permesso la realizzazione di un modello androide per la simulazione di attività in ambiente di lavoro. Tale modello è stato opportunamente validato in funzione di risultati sperimentali, di risultati riportati in letteratura e della forza di contatto con il suolo, in modo da renderlo rappresentativo della realtà. In questo modo, è resa possibile la simulazione di alcune delle attività che sono effettuate dai lavoratori, la quale risulta molto meno dispendiosa in termini di tempo e denaro della sperimentazione.

Infine, uno dei maggiori punti di forza dell'attuale di lavoro di tesi è l'introduzione di un indice che si possa relazionare alla stanchezza dei lavoratori alla fine della giornata. In relazione ai parametri monitorati dal dispositivo di sicurezza IoT o di quelli ricavabili dalla simulazione multibody, è infatti possibile calcolare un indice basato sull'entità del movimento effettuato dall'operatore durante la giornata lavorativa e dunque rappresentativo del suo livello di stanchezza. Maggiori studi sarebbero necessari per poter determinare l'effettiva correlazione tra il valore di indice di mobilità e la stanchezza dei lavoratori, ma quanto effettuato rappresenta un buono spunto per prossimi lavori e studi.

In futuro, sarebbe opportuno irrobustire il dispositivo di sicurezza realizzato e migliorarlo sotto numerosi punti di vista. Primariamente, è importante apportare delle modifiche al sistema che permettano di ridurre la domanda energetica complessivamente richiesta dallo stesso per il suo funzionamento. Inoltre, è necessario potenziare la connessione Bluetooth tra le due schede Arduino, che attualmente presenta alcuni problemi nel momento in cui le due schede si devono riconnettere. Il casco di protezione sensorizzato potrebbe anche essere migliorato aggiungendo un sensore di monitoraggio della frequenza cardiaca, utile per il successivo computo di un indice di affaticamento, che permetta di quantificare la stanchezza del lavoratore alla fine della giornata in relazione alle attività portate a termine. Infine, sarebbe utile irrobustire il sistema di avviso di prossimità e implementare la possibilità di visualizzare graficamente la posizione relativa degli operatori rispetto al macchinario in movimento per una migliore efficienza di funzionamento.
Per quanto concerne il modello multibody realizzato, nonostante esso abbia permesso l'ottenimento di importanti risultati, potrebbe anch'esso in futuro essere migliorato. Infatti, sarebbe opportuno riuscire a modellare lo smorzamento distribuito tipico del corpo umano e dei muscoli, che permetterebbe una migliore rappresentatività dei risultati sperimentali.

Nonostante i possibili sviluppi futuri, gli obiettivi iniziali del lavoro di tesi sono stati raggiunti. È infatti stato realizzato il casco di protezione sensorizzato IoT, che permette di monitorare costantemente lo stato di attività e salute dei lavoratori e le condizioni ambientali. Inoltre, si è portata a termine l'analisi multibody delle principali attività in ambiente di lavoro ed è stato presentato un indice possibilmente correlabile all'entità della stanchezza del lavoratore al termine della giornata lavorativa.

Appendice A

Codici di programmazione

Nel presente capitolo si riportano gli script di programmazione utilizzati per la realizzazione del dispositivo indossabile sensorizzato IoT descritto nel dettaglio nel Capitolo 1.

A.1 Arduino Nano 33 BLE Sense

Il seguente paragrafo riporta il codice di programmazione della scheda Arduino Nano 33 BLE Sense collocata sul casco di protezione sensorizzato. Tale script risulta anche adeguato all'eventuale integrazione di un sistema di avviso di prossimità.

```
// richiamo delle librerie necessarie al funzionamento del progetto
#include <Arduino_LPS22HB.h>
#include <Arduino_APDS9960.h>
#include <ArduinoBLE.h>
#include <Arduino_HTS221.h>

// definizione degli UUID del servizio e delle caratteristiche
Bluetooth su cui mandare i dati all'Arduino IoT
#define BLE_UUID_SENSORS_SERVICE      "9A48ECBA-2E92-082F-C079-
9E75AAE428B1"
#define BLE_UUID_TEMPERATURE_VALUE    "C8F88594-2217-0CA6-8F06-
A4270B675D69"
#define BLE_UUID_HUMIDITY_VALUE       "C8F88595-2217-0CA6-8F06-
A4270B675D69"
#define BLE_UUID_PRESSURE_VALUE       "C8F88596-2217-0CA6-8F06-
A4270B675D69"
#define BLE_UUID_PROXIMITY_VALUE      "C8F88597-2217-0CA6-8F06-
A4270B675D69"

BLEService sensorService( BLE_UUID_SENSORS_SERVICE ); // BLE SENSORS
Service
```

```
BLEFloatCharacteristic temperatureLevel( BLE_UUID_TEMPERATURE_VALUE,
BLERead | BLENotify );
BLEFloatCharacteristic humidityLevel( BLE_UUID_HUMIDITY_VALUE ,BLERead
| BLENotify);
BLEFloatCharacteristic pressureLevel( BLE_UUID_PRESSURE_VALUE ,BLERead
| BLENotify);
BLEFloatCharacteristic proximityLevel( BLE_UUID_PROXIMITY_VALUE
,BLERead | BLENotify);

// definizione delle variabili utili nello script
float temperature = 0.0;
float temperatureTOT = 0.0;
float humidity = 0.0;
float humidityTOT = 0.0;
float pressure = 0.0;
float pressureTOT = 0.0;
int p = 0;
float proximityTOT = 0.0;
float proximity = 0.0;
int check = 0;
int count = 0;

void setup() {
  Serial.begin(9600);
  while (!Serial);

  if (!HTS.begin()) {
    Serial.println("Failed to initialize humidity temperature
sensor!");
  }
  if (!BARO.begin()) {
    Serial.println("Failed to initialize pressure sensor!");
  }
  if (!APDS.begin()) {
    Serial.println("Error initializing APDS9960 sensor!");
  }

  BLE.begin();
  // set advertised local name and service
  BLE.setDeviceName( "Arduino Nano 33 BLE" );
  BLE.setLocalName( "SENSORS" );
  BLE.setAdvertisedService( sensorService );
  // BLE add characteristics
  sensorService.addCharacteristic( temperatureLevel );
  sensorService.addCharacteristic( humidityLevel );
  sensorService.addCharacteristic( pressureLevel );
  sensorService.addCharacteristic( proximityLevel );
  // add service
  BLE.addService( sensorService );
  // set the initial value for characeristics
```

```
  temperatureLevel.writeValue( temperature );
  humidityLevel.writeValue( humidity );
  pressureLevel.writeValue( pressure );
  proximityLevel.writeValue( proximity );
  BLE.advertise();
}

void loop() {
  temperature = HTS.readTemperature();
  temperature = temperature - 6;
  humidity = HTS.readHumidity();
  pressure = BARO.readPressure();
  if (APDS.proximityAvailable()){
    p = APDS.readProximity();
    proximity = (float)p;
  }

  temperatureTOT = temperatureTOT + temperature;
  humidityTOT = humidityTOT + humidity;
  pressureTOT = pressureTOT + pressure;
  proximityTOT = proximityTOT + proximity;

  count++;
  // listen for BLE peripherals to connect:
  BLEDevice central = BLE.central();

  if ( central ) {
    temperature = temperatureTOT / count;
    humidity = humidityTOT / count;
    pressure = pressureTOT / count;
    proximity = proximityTOT / count;

    while (central.connected() ){
      temperatureLevel.writeValue( temperature );
      humidityLevel.writeValue( humidity );
      pressureLevel.writeValue( pressure );
      proximityLevel.writeValue( proximity );
    }
    Serial.println(temperature);
    Serial.println(humidity);
    Serial.println(pressure);
    Serial.println(proximity);
    count = 0;
    temperatureTOT = 0;
    humidityTOT = 0;
    pressureTOT = 0;
    proximityTOT = 0;
  }
  delay(20);
}
```

A.2 Arduino Nano 33 IoT

Nel seguito viene riportato lo script di programmazione della scheda Arduino Nano 33 IoT, che permette la ricezione dei dati via Bluetooth dall'Arduino Nano 33 BLE Sense e l'invio dei dati al calcolatore, con Processing, mediante Wi-Fi. Nel codice sono inoltre riportati gli algoritmi di fall e impact detection.

```
// richiamo delle librerie necessarie al funzionamento del progetto
#include <Arduino_LSM6DS3.h>
#include <ArduinoBLE.h>
#include <ArduinoHttpClient.h>
#include <WiFiNINA.h>
#include "arduino_secrets.h"

char ssid[] = SECRET_SSID;
char pass[] = SECRET_PASS;
char serverAddress[] = "192.168.1.2";  // server address WIFI
int port = 8080;
WiFiClient wificlient;
WebSocketClient client = WebSocketClient(wificlient, serverAddress,
port);
int count = 0;

// definizione degli UUID del servizio e delle caratteristiche
Bluetooth da cui prendere i dati dell'Arduino BLE Sense
#define BLE_UUID_SENSORS_SERVICE              "9A48ECBA-2E92-082F-
C079-9E75AAE428B1"
#define BLE_UUID_TEMPERATURE_VALUE               "C8F88594-2217-0CA6-
8F06-A4270B675D69"
#define BLE_UUID_HUMIDITY_VALUE                "C8F88595-2217-0CA6-
8F06-A4270B675D69"
#define BLE_UUID_PRESSURE_VALUE                "C8F88596-2217-0CA6-
8F06-A4270B675D69"
#define BLE_UUID_PROXIMITY_VALUE                "C8F88597-2217-0CA6-
8F06-A4270B675D69"

// definizione delle variabili utili nello script
float Ax, Ay, Az;
float Gx, Gy, Gz;
float AxTOT, AyTOT, AzTOT;
float GxTOT, GyTOT, GzTOT;
float Atot;
int flag, check;
int numRipetizioni = 50;
float temperature = 0;
float humidity = 0;
float pressure = 0;
float proximity = 0;
```

```
unsigned long previousTime;
unsigned long interval = 20;
unsigned long previousCheck;
unsigned long lastUpdate = -61000;
unsigned long attesa = 60000;
int voltage = 0;
int B = 4275;               // B value of the thermistor
int R0 = 100000;            // R0 = 100k
int pinTempSensor = A0;     // Grove - Temperature Sensor connect to A0

void setup() {
  Serial.begin(9600);

  if (!IMU.begin()) {
    Serial.println("Failed to initialize IMU!");
    while (1);
  }
}

void loop() {
  // ogni volta che passa un'intervallo pari ad attesa si effettua il
collegamento Bluetooth tra le due schede Arduino
  if (millis() - lastUpdate > attesa){
    BLE.begin();
    Serial.println("BLE Central - SENSORS control");
    delay(500);
    BLE.scanForUuid( BLE_UUID_SENSORS_SERVICE );
    delay(500);

    BLEDevice peripheral = BLE.available();
    if ( peripheral ) {
      if ( peripheral.localName() != "SENSORS" ) {
        return;
      }
      BLE.stopScan();
      Serial.print("Found ");

      controlSensors( peripheral );
      lastUpdate = millis();
    }
  }

  //START WI-FI
  if ( WiFi.status() != WL_CONNECTED ){
    Serial.print("Attempting to connect to Network named: ");
    Serial.println(ssid);

    while( WiFi.status() != WL_CONNECTED ) {
      // Connect to WPA/WPA2 network:
      WiFi.begin(ssid, pass);
```

```
      Serial.print(".");
      delay(5000);
    }
    Serial.println("\nConnected");
    Serial.println("starting WebSocket client");
    client.begin();
  }

  while ( client.connected() && ((millis() - lastUpdate) < attesa) ) {
    // inizializzazione delle variabili a zero
    previousTime = millis();
    AxTOT = 0;
    AyTOT = 0;
    AzTOT = 0;
    GxTOT = 0;
    GyTOT = 0;
    GzTOT = 0;

    // monitoraggio dei parametri rilevati dai sensori integrati
(rilevazione ogni 20 ms e media sul secondo)
    for (int i = 0; i < numRipetizioni; i++) {
      // dati giroscopio e accelerometro
      if (IMU.accelerationAvailable() && IMU.gyroscopeAvailable()) {
        IMU.readAcceleration(Ax, Ay, Az);
        IMU.readGyroscope(Gx, Gy, Gz);

        AxTOT = AxTOT + Ax;
        AyTOT = AyTOT + Ay;
        AzTOT = AzTOT + Az;
        GxTOT = GxTOT + Gx;
        GyTOT = GyTOT + Gy;
        GzTOT = GzTOT + Gz;

        // Fall detection algorithm
        Atot = sqrt(Ax*Ax+Ay*Ay+Az*Az);
        if (Atot < 0.6 && flag == 0){
          flag = 1;
          previousCheck = millis();
        }
        if(Atot > 2 && flag == 1 && (millis() - previousCheck) < 800){
          flag = 2;
          previousCheck = millis();
        } else if (flag == 1 && (millis() - previousCheck) > 800){
          flag = 0;
          check = 0;
        }
        if (Atot > 0.95 && Atot < 1.065 && flag == 2 && (millis() -
previousCheck) >= 1000) {
          flag = 3;
          check = 1;
```

```
        } else if (flag == 2 && (Atot < 0.95 || Atot > 1.065) &&
(millis() - previousCheck) >= 400 && (millis() - previousCheck) <=
1000){
          flag = 0;
          check = 0;
        }
        if (Atot > 0.95 && Atot < 1.065 && flag == 3 && (millis() -
previousCheck) >= 6000) {
          flag = 4;
          check = 2;
        } else if (flag == 3 && (Atot < 0.95 || Atot > 1.065) &&
(millis() - previousCheck) <= 6000){
          flag = 5;
          check = 3;
        }
        // Impact detection algorithm
        if (Atot > 10){
          check = 5;
        }
      }
      if (millis() - previousTime < interval) {
        delay( interval - ( millis() - previousTime) );
      }
      previousTime = millis();
    }

    Ax = AxTOT / numRipetizioni;
    Ay = AyTOT / numRipetizioni;
    Az = AzTOT / numRipetizioni;
    Gx = GxTOT / numRipetizioni;
    Gy = GyTOT / numRipetizioni;
    Gz = GzTOT / numRipetizioni;

    voltage = analogRead(pinTempSensor);
    float R = 1023.0/voltage-1.0;
    R = R0*R;
    float bodyTemperature = 1.0/(log(R/R0)/B+1/298.15)-273.15; //
convert to temperature via datasheet

    // Invio messagio al WebSocket server
    client.beginMessage(TYPE_TEXT);
    client.print(Ax);
    client.print("/");
    client.print(Ay);
    client.print("/");
    client.print(Az);
    client.print("/");
    client.print(Gx);
    client.print("/");
    client.print(Gy);
```

```
    client.print("/");
    client.print(Gz);
    client.print("/");
    client.print(check);
    client.print("/");
    client.print(bodyTemperature);
    client.print("/");
    client.print(temperature);
    client.print("/");
    client.print(humidity);
    client.print("/");
    client.print(pressure);
    client.print("/");
    client.print(proximity);
    client.println();
    client.endMessage();

    if (check == 2 || check == 3){
      check = 0;
      flag = 0;
    }
    if (check == 5){
      check = 0;
    }
  }

  WiFi.end();
}

// funzione che si occupa della connessione Bluetooth tra le due
schede Arduino
void controlSensors( BLEDevice peripheral )
{
  if ( !peripheral.connect() )  {
    Serial.println("error_Connection");
    return;
  }
  if ( !peripheral.discoverAttributes() )  {
    Serial.println("error_Attributes");
    peripheral.disconnect();
    return;
  }

  BLECharacteristic temperatureLevel = peripheral.characteristic(
BLE_UUID_TEMPERATURE_VALUE );
  BLECharacteristic humidityLevel = peripheral.characteristic(
BLE_UUID_HUMIDITY_VALUE );
  BLECharacteristic pressureLevel = peripheral.characteristic(
BLE_UUID_PRESSURE_VALUE );
```

```
  BLECharacteristic proximityLevel = peripheral.characteristic(
BLE_UUID_PROXIMITY_VALUE );

  if ( !temperatureLevel || !humidityLevel || !pressureLevel ||
!proximityLevel )  {
    Serial.println("error_Characteristics");
    peripheral.disconnect();
    return;
  }
  if ( !temperatureLevel.canSubscribe() ||
!humidityLevel.canSubscribe() || !pressureLevel.canSubscribe() ||
!proximityLevel.canSubscribe() )  {
    Serial.println("error_canSubscribe");
    peripheral.disconnect();
    return;
  }
  if ( !temperatureLevel.subscribe() || !humidityLevel.subscribe() ||
!pressureLevel.subscribe() || !proximityLevel.subscribe() )  {
    Serial.println("error_Subscribe");
    peripheral.disconnect();
    return;
  }

  int aggiorno = 0; //una volta che sono tutti aggiornati si
interrompe la connessione Bluetooth

  while ( peripheral.connected() && aggiorno != 4 )
  {
    if ( temperatureLevel.valueUpdated() )  {
      temperatureLevel.readValue( &temperature, 4 );
      temperature = temperature;
      aggiorno = aggiorno + 1;
    }
    if ( humidityLevel.valueUpdated() )  {
      humidityLevel.readValue( &humidity, 4 );
      aggiorno = aggiorno + 1;
    }
    if ( pressureLevel.valueUpdated() )  {
      pressureLevel.readValue( &pressure, 4 );
      aggiorno = aggiorno + 1;
    }
    proximityLevel.readValue( &proximity, 5 );
    aggiorno = aggiorno + 1;
  }

  peripheral.disconnect();
  return;
}
```

A.3 Processing

Il seguente codice di programmazione del software Processing deve essere utilizzato per permettere la ricezione dei dati via Wi-Fi da parte del calcolatore. Tale script permette di visualizzare allarmi e anomalie in tempo reale e di salvare i dati in una tabella su un file .csv.

```
import websockets.*;

WebsocketServer ws;
int now;
int i=0;
float Ax,Ay,Az,Gx,Gy,Gz,bodyT,t,h,p,proximity;
float bodyT0, t0, h0, p0;
float proximity0 = 1000;
int check;
String data="";
float roll, pitch;
Table table;

void setup(){
  ws= new WebsocketServer(this,8080,"");
  now=millis();

  table = new Table();
  table.addColumn("Ax");
  table.addColumn("Ay");
  table.addColumn("Az");
  table.addColumn("Gx");
  table.addColumn("Gy");
  table.addColumn("Gz");
  table.addColumn("Check");
  table.addColumn("BodyTemperature");
  table.addColumn("Temperature");
  table.addColumn("Humidity");
  table.addColumn("Pressure");
  table.addColumn("Proximity");
}

void webSocketServerEvent(String msg){
 data = trim(msg);
    String items[] = split(data, '/');

    if (items.length > 1) {
      Ax = float(items[0]);
      Ay = float(items[1]);
      Az = float(items[2]);
      Gx = float(items[3]);
```

```
      Gy = float(items[4]);
      Gz = float(items[5]);
      check = int(items[6]);
      bodyT = float(items[7]);
      t = float(items[8]);
      h = float(items[9]);
      p = float(items[10]);
      proximity = float(items[11]);
    }

 TableRow newRow = table.addRow();
 newRow.setFloat("Ax", Ax);
 newRow.setFloat("Ay", Ay);
 newRow.setFloat("Az", Az);
 newRow.setFloat("Gx", Gx);
 newRow.setFloat("Gy", Gy);
 newRow.setFloat("Gz", Gz);
 newRow.setFloat("Check", check);
 newRow.setFloat("BodyTemperature", bodyT);
 newRow.setFloat("Temperature", t);
 newRow.setFloat("Humidity", h);
 newRow.setFloat("Pressure", p);
 newRow.setFloat("Proximity", proximity);

 saveTable(table, "data/new.csv");

 if (check == 3) {
   println("ATTENZIONE: caduta rilevata!");
 } else if (check == 2) {
   println("ATTENZIONE: caduta con possibile perdita di conoscenza
rilevata!");
 } else if (check == 5) {
   println("ATTENSIONE: impatto rilevato!");
 }

 if (proximity0 != 1000){
   if (proximity != proximity0){
     if (proximity > 50 && proximity0 > 50){
       println("ALLERTA: casco di protezione non indossato!");
     } else if (proximity > 50 && proximity0 < 50){
       println("ALLERTA: casco di protezione rimosso");
     }
   }
 }

 if (proximity0 != 1000){
   if (bodyT != bodyT0){
     if ((bodyT > bodyT0 + 5) || (bodyT < bodyT0 - 5)){
       println("ANOMALIA: variazione repentina di temperatura della
superficie corporea superiore a 5 °C");
```

```
      }
    }
    if (t != t0){
      if ((t > t0 + 5) || (t < t0 - 5)){
        println("ANOMALIA: variazione repentina di temperatura
ambientale superiore a 5 °C");
      }
    }
    if (h != h0){
      if ((h > h0 + 10) || (h < h0 - 10)){
        println("ANOMALIA: variazione repentina di umidità ambientale
superiore al 10%");
      }
    }
    if (p != p0){
      if ((p > p0 + 0.5) || (p < p0 - 0.5)){
        println("ANOMALIA: variazione repentina di pressione ambientale
superiore a 0.5 kPa");
      }
    }
  }

  proximity0 = proximity;
  t0 = t;
  h0 = h;
  p0 = p;
  bodyT0 = bodyT;

}
```

A.4 Sistema di Avviso di Prossimità

Per ultimo, si riporta lo script di programmazione della scheda Arduino Nano 33 BLE Sense di cui dotare i macchinari in movimento nell'ambiente di lavoro per permettere la realizzazione di un sistema di avviso di prossimità. Dato che la ricerca dei dispositivi avviene per nome, è necessario modificare il codice con il nome dei dispositivi che si possono trovare nell'area interessata.

```
#include <ArduinoBLE.h>

int i = 0;
int taratura = 0;
int valore, exponent1;
float exponent2, distance;
int RSSImetawear, RSSImiband, RSSIarduino;
int N = 20;
int fatto = 0;
int fatto1 = 0;
```

```
int fatto2 = 0;

void setup() {
  Serial.begin(9600);
  while (!Serial);
  // begin initialization
  if (!BLE.begin()) {
    Serial.println("starting BLE failed!");
    while (1);
  }
  Serial.println("BLE Central scan");
}

void loop() {
  if (taratura == 0) {
     Serial.println("COLLOCARE DISPOSITIVI A DISTANZA = 1 m DAL
MACCHINARIO");
    while(fatto == 0){
      BLE.scanForName( "Nome dispositivo 1" );
      delay(500);
      BLEDevice peripheral = BLE.available();
      if (peripheral && i == 0) {
        Serial.print("Local Name: ");
        Serial.println(peripheral.localName());
        Serial.print("RSSI: ");
        Serial.println(peripheral.rssi());
        i = 1;
        RSSImetawear = peripheral.rssi();
        fatto = 1;
      }
    }
    while(fatto1 == 0){
      BLE.scanForName( "Nome dispositivo 2" );
      delay(500);
      BLEDevice peripheral1 = BLE.available();
      if (peripheral1 && i == 1) {
        Serial.print("Local Name: ");
        Serial.println(peripheral1.localName());
        Serial.print("RSSI: ");
        Serial.println(peripheral1.rssi());
        i = 2;
        RSSImiband = peripheral1.rssi();
        fatto1 = 1;
      }
    }
    // CONTINUARE COSI' CON TUTTI GLI ALTRI DISPOSITIVI DA MONITORARE

    taratura = 1;
  }
```

```
  BLE.scanForName( "Nome dispositivo 1" );
  delay(500);
  BLEDevice peripheral = BLE.available();
  if (peripheral && i == 0) {
    Serial.print("Local Name: ");
    Serial.println(peripheral.localName());
    Serial.print("RSSI: ");
    Serial.println(peripheral.rssi());
    i = 1;
    valore = peripheral.rssi();
    exponent1 = ( RSSImetawear - valore );
    exponent2 = float( exponent1 ) / N;
    distance = pow(10, exponent2);
    if (distance < 1){
      Serial.println("ATTENZIONE: zona di sicurezza di 1m superata!");
    }
  }

  BLE.scanForName( "Nome dispositivo 2" );
  delay(500);
  BLEDevice peripheral1 = BLE.available();
  if (peripheral1 && i == 1) {
    Serial.print("Local Name: ");
    Serial.println(peripheral1.localName());
    Serial.print("RSSI: ");
    Serial.println(peripheral1.rssi());
    i = 2;
    valore = peripheral1.rssi();
    exponent1 = ( RSSImiband - valore );
    exponent2 = float( exponent1 ) / N;
    distance = pow(10, exponent2);
    if (distance < 1){
      Serial.println("ATTENZIONE: zona di sicurezza di 1m superata!");
    }
  }

  // CONTINUARE COSI' CON TUTTI GLI ALTRI DISPOSITIVI DA MONITORARE
}
```

Ringraziamenti

Al termine di questo lavoro di tesi e del mio percorso di laurea desidero dedicare alcune righe alle persone che, con il loro supporto, mi sono state vicine in questi cinque anni.

Per primo, desidero ringraziare il professore Aurelio Somà per avermi dato fiducia e avermi permesso di lavorare su questo progetto, che rientrava in un ambito a me completamente sconosciuto. Inoltre, ringrazio Caterina Russo, che mi ha seguita costantemente durante tutta la realizzazione del lavoro di tesi, per i suoi preziosi consigli e insegnamenti. Grazie a voi e alla vostra disponibilità ho potuto accrescere le mie conoscenze e competenze.

Ringrazio di cuore i miei genitori, che mi hanno sempre sostenuto in ogni mia scelta, sono stati sempre dalla mia parte, gioendo con me dei miei traguardi. Grazie per avermi permesso di arrivare fin qui e di aver fatto sì che credessi un po' di più in me stessa. Ringrazio mia sorella Alessia, che è stata un esempio per me ed è sempre stata pronta a consigliarmi ed aiutarmi quando c'era bisogno. Un ringraziamento speciale va al mio fidanzato Manuel, che in questi anni ha creduto in me più di quanto lo potessi fare io, ha saputo aspettare quando necessario e mi ha supportato, e sopportato, sempre.

Un ringraziamento va poi ai miei nonni e alla mia famiglia, che in questi anni hanno sempre fatto il tifo per me e sono sempre stati al mio fianco.

E grazie a tutti i miei amici, quelli di una vita, quelli che ho conosciuto in questi ultimi anni e i miei compagni di università: mi avete regalato dei bei momenti di spensieratezza e avete reso questo mio percorso indimenticabile!

Bibliografia

[1] Varade, A.; Gajbhiye, N.; Mousam; *Smart Helmet Using GSM and GPS*. International Research Journal of Engineering and Technology (IRJET), Volume: 04 Issue: 03 | Mar - 2017, Nagpur, India.

[2] Kumar, D.; Gupta, S.; Kumar, S.; Srivastava, S; *Accident Detection and Reporting System using GPS and GSM Module*. Journal of Emerging Technologies and Innovative Research (JETIR), May 2015, Volume 2, Issue 5, India.

[3] Nazir, R., Tariq, A., Murawwat, S. and Rabbani, S. (2014) *Accident Prevention and Reporting System Using GSM* (SIM 900D) and GPS (NMEA 0183). Int. J. Communications, Network and System Sciences, 7, 286-293.

[4] Vijayan, S.; T Govind, V.; Mathews, M.; Surendran, S.; Sabah M E, M.; *Alcohol Detection Using Smart Helmet System*. International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE), ISSN: 0976-1353, Volume 8, Issue 1 – April 2014. Kannur, India.

[5] Behr, C. J.; Kumar, A.; Hancke, G. P.; *A Smart Helmet for Air Quality and Hazardous Event Detection for the Mining Industry*. Conference Paper · March 2016, Pretoria, South Africa.

[6] Bhumkar, S.P.; Deotare, V. V.; Babar, R. V.; *Accident Avoidance and Detection on Highways*. International Journal of Engineering Trends and Technology – Volume 3, Issue 2, 2012, Pune, India.

[7] Mangala Nandhini, V.; Padma Priya, G.V; Nandhini, S.; Dinesh, K.; *IoT based Smart Helmet for Ensuring Safety in Industries*. International Journal of Engineering Research & Technology (IJERT), ISSN: 2278-0181, ETEDM - 2018 Conference Proceedings, Perundurai, Erode.

[8] Campero-Jurado, I.; Márquez-Sánchez, S.; Quintanar-Gómez, J.; Rodríguez, S.; M. Corchado, J.; *Smart Helmet 5.0 for Industrial Internet of Things Using Artificial Intelligence*. Article in "Sensors", 1 November 2020.

[9] Impana, H. C.; Hamsaveni, M.; Chethana, H. T.; *A Review on Smart Helmet for Accident Detection using IOT*. EAI Endorsed Transactions on Internet of Things, 14 May 2020, Mysore, India.

[10] Sun, S.; Zheng, X.; Gong, B.; Paredes, J. G.; Ordieres-Meré, J.; *Healthy Operator 4.0: A Human Cyber–Physical System Architecture for Smart Workplaces*. Article in "Sensors", 3 April 2020.

[11] Santhanakrishnan, C.; Sharma, D.; Vashistha, A.; *Smart Helmet for Rider (SHR) and Accident Detection using IOT*. International Journal of Advanced Science and Technology, Vol. 29, No. 6s, (2020), pp. 50-57, Chennai – India.

[12] Hun Kim, S.; Wang, C.; Dong Min, S.; Hyun Lee, S.; *Safety Helmet Wearing Management System for Construction Workers Using Three-Axis Accelerometer Sensor*. Article in "Applied sciences", 26 November 2018.

[13] Shravya, K.; Mandapati, Y.; Keerthi, D.; Harika, K.; K. Senapati, R.; *Smart Helmet for Safe Driving*. VNR VignanaJyothi Institute of Engineering and Technology, Bachupally, Hyderabad, Telangana, India, 2019.

[14] Sharma, M.; Maity, T.; *Low Cost Low Power Smart Helmet for Real-Time Remote Underground Mine Environment Monitoring*. Department of Mining Machinery Engineering, IIT (ISM), Dhanbad, India, 2018.

[15] Kumar Kar, S.; A Anshuman, D.; Raj, H.; Pall Singh, P.; *New Design and Fabrication of Smart Helmet*. 2018 IOP Conf. Ser.: Mater. Sci. Eng. 402 012055. Tamil Nadu.

[16] Altamura, A; Inchingolo, F.; Mevoli, G.; Boccadoro, P.; *SAFE: Smart helmet for Advanced Factory Environment*. Department of Electrical and Information Engineering (DEI), Politecnico di Bari, Bari, Italy, June 2018.

[17] Fyffe, D.; Langenderfer, C.; Johns, C.; *The Smart Hard Hat*. 2016. Honors Research Projects. 267.

[18] P. Borkar, S.; B. Baru, V.; *IoT Based Smart Helmet for Underground Mines*. International Journal of Research in Engineering, Science and Management Volume 1, Issue 9, September 2018.

[19] Byeon, J.; Jang, M.-S.; Choi, S.-W.; Dong Yoo, H.; Lee, E.-H.; *A Study on Smart Helmet to Efficiently Cope with the Operation and Safety of Workers in Industrial Settings*. International Journal of Control and Automation, Vol. 11, No. 3 (2018), pp.169-178.

[20] https://store.arduino.cc/

[21] https://wiki.seeedstudio.com/Grove-Temperature_Sensor_V1.2

[22] Wu, F.; Zhao, H.; Zhao, Y.; Zhong, H, *Development of a Wearable-Sensor-Based Fall Detection System*. Hindawi Publishing Corporation, International Journal of Telemedicine and Applications. Volume 2015, Article ID 576364.

[23] Lim, D.; Park, C.; Ho Kim, N.; Kim, S.; Seop Yu, Y., *Fall-Detection Algorithm Using 3-Axis Acceleration: Combination with Simple Threshold and Hidden Markov Model*.

Hindawi Publishing Corporation, Journal of Applied Mathematics. Volume 2014, Article ID 896030.

[24] Rakhman, A. Z.; Nugroho, L. E.; Widyawan, Kurnianingsih., *Fall Detection System Using Accelerometer and Gyroscope Based on Smartphone.* 20 14 I st International Conference on Infonnation Technology, Computer and Electrical Engineering (ICIT ACEE).

[25] Bagalà, F.; Becker, C.; Cappello, A.; Chiari, L.; Amnian, K.; Hausdorff, J. M.; Zijlstra, W.; Klenk, J., *Evaluation of Accelerometer-Based Fall Detection Algorithms on Real-World Falls.* May 2012, PLoS ONE 7(5):e37062.

[26] Fudickar, S.; Lindemann, A.; Schnor, B., *Threshold-based Fall Detection on Smartphones.* Department of Computer Science, University of Potsdam, August-Bebel Str. 89, Potsdam, Germany. 2014.

[27] Silva, M.; Teixeira, P.; Abrantes, F.; Sousa, F., *Design and Evaluation of a Fall Detection Algorithm on Mobile Phone Platform.* AMBI-SYS 2011, LNICST 70, pp. 28–35, 2011. Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2011.

[28] Ge, Y.; Xu, B., *Detecting Falls Using Accelerometers by Adaptive Thresholds in Mobile Devices.* Journal Of Computers, Vol. 9, No. 7, July 2014.

[29] King, D.; Hume, P.; Giddane, C.; Brughelli, M.; Clark, T. , *The Influence of Head Impact Threshold for Reporting Data in Contact and Collision Sports: Systematic Review and Original Data Analysis.* Article in Sports Medicine · February 2016.

[30] O'Connor, K. L.; Rowson, S.; Duma, S. M.; Broglio, S. P., *Head-Impact Measurement Devices: A Systematic Review.* Article *in* Journal of Athletic Training · March 2017.

[31] Virzi Mariotti, G.; Golfo, S.; Nigrelli, V.; Carollo, F., *Head Injury Criterion: Mini Review.* Am J Biomed Sci & Res. 2019 - 5(5). AJBSR.MS.ID.000957. DOI: 10.34297/AJBSR.2019.05.000957.

[32] Gao, D. and Wampler, C. W., *On the Use of the Head Injury Criterion (HIC) to Assess the Danger of Robot Impacts.* General Motors R&D Center, Warren, MI 48090, USA. March 17, 2009.

[33] Kim, Y, Baek, J., and Choi, Y., *Smart Helmet-Based Personnel Proximity Warning System for Improving Underground Mine Safety.* Applied sciences, 2021, 11, 4342.

[34] https://iotandelectronics.wordpress.com/2016/10/07/how-to-calculate-distance-from-the-rssi-value-of-the-ble-beacon/

[35] Scheda tecnica Duracell – duralock power preserve - Plus Power. MN1604, 9V (6LR61). Alkaline-Manganese Dioxide Battery.

[36] Russo, C.; Somà, A., *Studio di un modello multi-body antropomorfo per la caratterizzazione dell'attività del Nordic Walking*. Tesi di Laurea Magistrale in Ingegneria Meccanica. Politecnico di Torino, Aprile 2019.

[37] Russo, C.; Mocera, F.; Somà, A, *Nordic walking multibody analysis and experimental identification*. Journal of SPORTS ENGINEERING AND TECHNOLOGY, March 2020.

[38] Dziuba, A., K.; Zurek, G.; Garrard, I.; Wierzbicka-Damska, I., *Biomechanical parameters in lower limbs during natural walking and Nordic walking at different speeds*. Acta of Bioengineering and Biomechanics Original paper, vol. 17, n. 1, pp. 97-101, 2015.

[39] Mooney, L., M.; Herr, H., M., *Biomechanical walking mechanisms underlying the metabolic reduction caused by an autonomous exoskeleton*. Mooney and Herr Journal of NeuroEngineering and Rehabilitation (2016) 13:4. DOI 10.1186/s12984-016-0111-3.

[40] http://www.clinicalgaitanalysis.com/

[41] https://www.theverge.com/circuitbreaker/2016/10/19/13329086/moov-hr-heart-rate-monitoring-wearable-fitness-sweatband

[42] https://www.garmin.com/it-IT/blog/frequenza-cardiaca-fascia-cardio-sensore-ottico/