POLITECNICO DI TORINO

Corso di Laurea Magistrale in
INGEGNERIA MECCANICA

A.a. 2023/2024
Sessione di Laurea LUGLIO 2024

Sviluppo di interfaccia gestuale per il controllo di robot collaborativi con l'utilizzo di sensori inerziali

Relatore
Prof. Stefano Paolo Pastorelli

Candidato
Francesco Crivellari

Correlatori
Ing. Valerio Cornagliotto
Ing. Michele Polito

Abstract

Con l'avvento di Industria 4.0, i robot si sono diffusi notevolmente nelle aziende, rendendo l'interfacciamento con queste macchine comune per molte figure professionali.

Siamo ora all'inizio di una nuova fase che prende il nome di Industria 5.0. Questa mira a ottimizzare i cicli produttivi rafforzando la sinergia tra uomo e robot. L'obiettivo è creare un ambiente di lavoro più sicuro e confortevole affiancando i robot collaborativi (cobot) agli operatori per supportarli durante le attività. Per fare ciò ci si avvale di strumenti quali: intelligenza artificiale, sistemi di visione, sistemi di prossimità e altri sistemi di sensoristica per la percezione della presenza e del movimento degli esseri umani.

Ci sono ancora diversi limiti tecnici che ostacolano questi obiettivi. Il metodo di programmazione e interazione con i robot, ad esempio, è spesso contro-intuitivo, e costringe l'utente a operare in un ambiente ottimizzato per la macchina e non viceversa. Questo causa stress cognitivo per l'operatore, con ripercussioni negative su salute, sicurezza, comfort e produttività.

Con lo scopo di migliorare l'interfacciamento dell'operatore con i robot, l'obiettivo di questo lavoro di tesi è sviluppare una modalità di comunicazione più user-friendly, che preveda l'interpretazione di schemi motori naturali e intuitivi con lo scopo di manovrare un robot collaborativo senza l'utilizzo di interfacce di comando esterne, come joystick o tastiere, ma attraverso il riconoscimento dei gesti dell'operatore. L'acquisizione dei movimenti avviene mediante l'utilizzo di sensori inerziali, i quali non richiedono l'integrazione di tecnologie complesse e costose nello spazio di lavoro e garantiscono flessibilità operativa e rapidità di comunicazione con i robot.

Particolare attenzione è stata rivolta allo sviluppo di un controllo proporzionale e continuo del robot, per consentire all'operatore maggiore flessibilità nei movimenti della macchina e ampliare le possibilità di utilizzo.

La prima parte della tesi si concentra sull'approfondimento delle tecnologie software e hardware già esistenti, con una particolare attenzione alla tipologia di sensore inerziale da utilizzare, andando a confrontare diverse soluzioni al fine di individuare la migliore per lo specifico scopo. Questa ricerca è stata condotta anche attraverso la creazione di un protocollo di test ripetibile, eseguito con braccio cobot, il cui obiettivo è delineare una metodologia di valutazione degli strumenti, implementabile e utilizzabile anche in futuro per testare nuove soluzioni e tecnologie.

La seconda parte è stata invece incentrata sullo sviluppo di un dispositivo indossabile composto da due sensori inerziali da posizionare su braccio e avanbraccio, l'obiettivo è consentire all'operatore di controllare il robot senza rinunciare

alla flessibilità nei movimenti.

Durante la progettazione i supporti indossabili sono stati disegnati con il software di modellazione CAD 3D Solidworks e poi prodotti con tecnologie di stampa 3D. L'implementazione software, invece, è stata eseguita mediante lo sviluppo di un'architettura di controllo in grado di far interagire i sensori con il robot utilizzando il set di librerie software ROS (Robot Operating System). L'ultimo step è stato l'assemblaggio di tutta l'architettura prototipale con successivo test sperimentale in cui i dati delle due IMU sono stati interpretati per controllare orientazione e velocità lineare dell'End-effector del robot, nonché la modalità di utilizzo.

Indice

Indice delle immagini

1 Introduzione

A partire dagli inizi dagli anni '50, i robot hanno occupato sempre più spazio all'interno del contesto industriale e, se da un lato sono diventati sempre più complessi a livello costruttivo e di sviluppo, dall'altro hanno fatto passi da gigante nella sicurezza e nella facilità di utilizzo [1]. Nell'ultimo periodo la nascita di Industria 5.0, che si pone come obiettivo la valorizzazione delle capacità umane all'interno del contesto industriale, attraverso la creazione di un ambiente su misura, ha imposto un'ulteriore accelerazione al processo evolutivo, concentrando un'attenzione sempre maggiore a quella categoria della robotica che si occupa proprio della collaborazione tra uomo e robot [2].

Nel campo dell'automazione, la robotica collaborativa ha l'obiettivo di facilitare l'operatore umano, migliorando efficienza e sicurezza sul posto di lavoro, grazie all'utilizzo di macchine in grado di interagire con l'uomo in modo sicuro e flessibile.

A differenza dei tradizionali robot industriali, che operano in spazi separati dagli esseri umani e sono circondati da barriere di sicurezza, i robot collaborativi, anche detti *cobot*, sono progettati per lavorare a stretto contatto con le persone, senza il ricorso a barriere fisiche di sicurezza come recinzioni. Essi hanno inoltre ridotte dimensioni e capacità di carico, dovute a limitazioni legislative come la norma ISO/TS15066, che indica forza e pressione massime applicabili dal robot su un essere umano, il limite è imposto in base alla parte del corpo coinvolta (esempio: 140N e 1Mpa per il contatto con la testa). Si cerca così di ridurre il rischio introdotto da applicazioni di collaborazione tra uomo e robot.

Esistono diversi livelli di interazione che possono essere classificati in base al grado di contatto che questi ultimi hanno con l'uomo [3].

1. **Robot recintato:** sono i classici robot industriali separati fisicamente dagli operatori umani

2. **Coesistenza:** robot che operano senza barriere fisiche ma non nello stesso spazio di lavoro di operatori umani. Nello spazio di lavoro dei robot sono introdotti sistemi di sicurezza come sensori di prossimità, sistemi di visione o sistemi ad impulsi di luce laser (*Lidar*), che permettono di rilevare la presenza di persone e rallentare o arrestare il robot in caso sia necessario. La maggior parte dei cobot utilizzati attualmente in ambito industriale appartengono a questa tipologia.

3. **Collaborazione sequenziale:** robot in grado di lavorare in uno spazio di lavoro condiviso con un operatore umano ma senza avere un'interferenza temporale. In questo caso il cobot rileva la presenza di una persona nel proprio spazio di lavoro e aspetta che quest'ultima si allontani per avviare le operazioni.

4. **Cooperazione parallela:** robot in grado di operare nello stesso spazio di lavoro, contemporaneamente a un operatore umano. In questo caso, i movimenti dell'operatore sono costantemente monitorati da parte del cobot, affinché quest'ultimo non entri in diretto contatto con la persona.

5. **Collaborazione:** è la massima espressione di collaborazione che può essere richiesta a un cobot. Il robot e l'operatore umano lavorano simultaneamente, interfacciandosi con lo stesso oggetto. La più classica di queste operazioni è quella di *hand-over* [4], in cui il robot consegna un determinato oggetto ad un operatore, In questo caso i sistemi di rilevamento umano montati sul robot non servono a evitare contatti con l'operatore umano, come nelle tipologie precedentemente descritte, ma servono a interagire direttamente con esso in maniera sicura, efficiente e funzionale. Naturalmente le applicazioni con questo livello di collaborazione necessitano di un sistema di percezione dell'essere umano molto accurato. A tale scopo, sono spesso utilizzati sistemi di visione che, attraverso l'utilizzo di una o più telecamere, permettono al robot di conoscere con precisione la posizione di un operatore nelle vicinanze. La complessità delle operazioni e questo tipo di sensori presentano diverse problematiche, tra cui la difficoltà di installazione in ambienti industriali o la possibilità che ci sia occlusione nel campo visivo delle telecamere. In aggiunta a ciò, l'elaborazione immagine richiede anche un notevole tempo computazionale che rende lento il processo di percezione dell'operatore, questo è poco efficace per applicazioni real time o per percepire movimenti improvvisi dell'operatore. Servono dati diretti del movimento dell'operatore e che richiedano meno tempo computazionale per l'elaborazione [5] [6].

Questo lavoro di tesi si inserisce nel contesto della robotica collaborativa, e si pone come obiettivo lo sviluppo di un dispositivo indossabile da utilizzare come interfaccia di comando robot. Questo, attraverso una serie di sensori inerziali denominati IMU (*Inertial Measurement Unit*), i quali permettono all'operatore di avere un controllo real time sul robot collaborativo, grazie alla loro rapidità di elaborazione dati. È stato scelto un modello di comunicazione che sfrutti schemi motori intuitivi e che di conseguenza vada a migliorare la sicurezza, il comfort e la produttività dell'operatore stesso. Questo tipo di soluzione rappresenta un vantaggio dal punto di vista della facilità di controllo del robot e dunque un passo avanti nell'ambito della collaborazione. Il sistema si presta anche a integrazioni con altre soluzioni esistenti, dal momento che potrebbe anche essere affiancato a un sistema di visione, per ottenere informazioni complementari.

Il fine è quello di approfondire la fattibilità e l'efficacia di questi dispositivi indossabili, individuandone limiti e punti di forza, nonché di implementare una possibile applicazione reale che possa a tutti gli effetti essere sfruttata in un contesto industriale.

L' attività svolta durante il progetto di tesi è suddivisa in due parti principali: una prima parte orientata allo studio delle tecnologie esistenti e una seconda parte centrata sullo sviluppo di un dispositivo funzionante per un'applicazione di telecontrollo.

Nella prima parte si è fatta una valutazione generale sullo strumento IMU, andando a studiare inizialmente cosa sono e come si sono evolute, per poi approfondire il loro utilizzo nell'ambito della robotica. In seguito, è stato necessario addentrarsi nel dettaglio delle diverse tipologie di IMU attualmente in commercio, con l'obiettivo di classificarle e selezionare i modelli più adatti all'utilizzo previsto, ovvero la robotica industriale. Sono state, pertanto, considerate caratteristiche come numero di assi, dimensioni e software compatibili, tenendo conto di aspetti complementari, come ad esempio la documentazione e gli algoritmi a supporto dei modelli. Infine, è stato predisposto lo studio di un protocollo di prova ripetibile per la verifica delle caratteristiche delle IMU selezionate, al fine di confrontare le prestazioni delle une con le altre. Il protocollo sviluppato consiste nell'utilizzo di un braccio robotico collaborativo per eccitare il sensore con traiettorie note. Durante il test sono stati previsti sei diversi tipi di movimenti volti a valutare sia la precisione dei sensori interni alla scheda, sia l'algoritmo di data-fusion per la stima dell'orientazione.

Nella seconda parte, è stata eseguita la progettazione di un'architettura, sia hardware che software, volta alla realizzazione di un prototipo indossabile e utilizzabile nel controllo dei movimenti di un robot attraverso l'interpretazione dei movimenti del braccio. Inizialmente, l'attività ha previsto lo studio di un'architettura preesistente, in cui veniva utilizzata una IMU commerciale per un task di telecontrollo. Successivamente è stato necessario procedere alla modifica della suddetta architettura, sostituendo il sensore inerziale commerciale con una scheda di sviluppo, per poi aggiornare l'architettura inserendo nuove modalità di controllo unicamente basate sulla lettura dei movimenti del sensore indossato. Infine, l'attività ha riguardato la progettazione del prototipo definitivo, sia hardware sia software. La progettazione hardware è passata attraverso la definizione del setup, la creazione dei collegamenti elettrici e la realizzazione dei supporti indossabili, progettati con il software Solidworks e successivamente stampati in 3D. Lo sviluppo informatico è stato basato sulla programmazione dei nodi ROS, software utilizzato per la comunicazione tra IMU e robot.

Una volta assemblato, il prototipo è stato sottoposto a un test da banco, finalizzato a verificare il corretto funzionamento delle modalità di utilizzo previste, ovvero il controllo di posizione e orientazione del robot nello spazio. L'intera procedura, sia di movimento del robot che di cambio modalità è stata effettuata unicamente con l'ausilio di un'interfaccia gestuale.

2 Sensori inerziali

2.1 Definizione e Utilizzi

Le IMU, sono dispositivi elettronici che fanno parte della più grande famiglia dei MEMS (micro electro-mechanical systems). Sono costituite da una serie di sensori quali accelerometro, giroscopio e, in molti casi, anche magnetometro che permettono la stima della velocità angolare e dell'accelerazione che caratterizzano il movimento di un corpo nello spazio, da cui si può valutare la sua posizione e orientazione. Con il loro sviluppo, nel tempo sono stati prodotti strumenti sempre più piccoli, precisi ed economici, fattori grazie ai quali le IMU sono diventate indispensabili per moltissimi utilizzi in cui la portabilità risulta essere fattore determinante [7].

Le possibili applicazioni di questo tipo di sensori sono molteplici. Allo stato attuale l'ambito principale di utilizzo è quello degli smartphone, ma anche quello nautico e aerospaziale, i quali vedono l'utilizzo di questo tipi di strumenti da moltissimi anni, come si può vedere nella Figura 1, per via della necessità di conoscere l'orientazione esatta dei velivoli [8]. Le IMU, inoltre, trovano ampia applicazione anche in ambito medico-sanitario e in particolare in quelle attività in cui risulta necessario un monitoraggio costante del movimento [9]. Proprio quest'ultima casistica, infatti, ha recentemente visto una diffusione applicativa anche in altri ambiti non strettamente riabilitativi. In ambito sportivo, ad esempio, le IMU vengono utilizzate per la valutazione delle prestazioni fisiche degli atleti, ma anche nel campo della robotica industriale per via della loro versatilità e precisione [10].

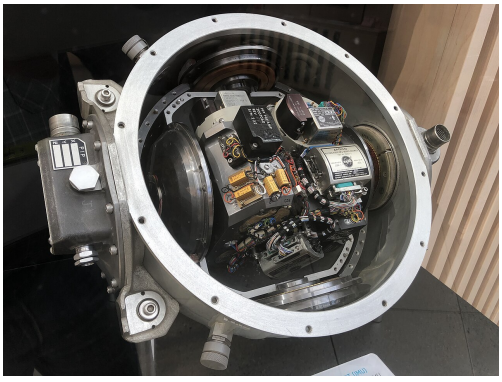


Fig. 1: IMU usata nel programma Apollo

Nell'ambito della robotica, c'è spesso la necessità di registrare dati di accelerazione lineare o di orientazione nello spazio, ad esempio per attività di monitoraggio dello slittamento in un robot dotato di ruote, oppure per la stima della posizio-

ne delle gambe in un robot quadrupede, per verificarne la stabilità. In queste applicazioni, le IMU risultano uno strumento fondamentale di acquisizione dati [11].

Nell'ambito più specifico della robotica collaborativa, questi dispositivi, per via delle loro potenzialità già dimostrate in altri ambiti, potrebbero trovare particolare utilizzo nei casi in cui vi sia la necessità di mappare i movimenti dell'essere umano e per alcuni versi si tratta di un utilizzo non dissimile dall'ambito medico, come già sottolineato, seppur con finalità completamente diverse [12].

Il concetto consiste nel creare dei dispositivi indossabili che permettano innanzitutto di semplificare lo schema di un corpo umano, equiparandolo a un robot con una serie di giunti e gradi di libertà. Successivamente è possibile trasferire questa informazione al robot, programmandolo per reagire a determinati movimenti.

Un esempio di applicazione, recentemente sviluppata e presentata in [13], è il *co-lifting* di un oggetto ingombrante, in cui un operatore dà il comando di avvio al robot attraverso una *gesture* e contestualmente afferra un oggetto ingombrante da un lato, affinché il robot possa sostenere l'oggetto dal lato opposto, facilitando così l'operatore nello spostamento.

I sensori IMU attualmente in commercio possono essere acquistati sotto forma di schede di sviluppo o come soluzioni commerciali *embedded*, ovvero sistemi chiusi e pronti all'uso, ottimizzati per specifiche attività da aziende specializzate nel settore.

Le schede di sviluppo consistono in piccole schede con sensore integrato che vanno manualmente collegate a sistemi in grado di decodificare i segnali forniti in output. Questo tipo di prodotti viene generalmente utilizzato a scopo di ricerca, per sperimentazione o prototipazione.

Queste schede offrono un vantaggio di personalizzazione, sia dal punto di vista del software, sia dal punto di vista dell'hardware. Si può scegliere ad esempio la forma e la dimensione del supporto, al fine di ottimizzare il dispositivo.

Le soluzioni commerciali embedded, invece, forniscono un'architettura pronta all'uso e ottimizzata per specifiche attività. Pertanto, utilizzando soluzioni già ingegnerizzate, si rinuncia alla possibilità di personalizzare l'architettura a favore di una maggiore semplicità di utilizzo. Questa soluzione è principalmente adottata dalle aziende che hanno bisogno di questo tipo di prodotti per utilizzi pratici e scopi specifici, che non prevedano la sperimentazione.

Tra le soluzioni commerciali embedded, nella categoria dei *Wearable Sensors* ovvero sensori indossabili per il tracciamento del movimento umano, utilizzati principalmente in ambito medico e sportivo, possono essere citati i sensori prodotti dall'azienda *APDM* o quelli prodotti dall'azienda *Xsens*.

2.2 Funzionamento

Una IMU fa riferimento a una terna di assi XYZ ed è composta principalmente di 3 strumenti di misurazione:

- Un accelerometro, in grado di misurare l'accelerazione lineare lungo i tre assi, nella maggior parte dei casi espressa in g ovvero *9,81* m/s^2.
- Un giroscopio, in grado di misurare la velocità angolare attorno ai tre assi, spesso espressa in rad/s.
- Un magnetometro, in grado di allinearsi al campo magnetico terrestre, fornendo ulteriori informazioni relative alle rotazioni attorno agli assi. Quest'ultimo strumento, a differenza dei due precedenti, non è sempre presente nella IMU.

Le informazioni acquisite vengono successivamente rielaborate da algoritmi di *Data-Fusion* il cui funzionamento differisce da sistema a sistema in base al software installato, influendo sulla capacità della IMU di tracciare l'orientazione. Generalmente questi software sono dotati di filtri per la regolazione del rumore dei segnali, come ad esempio il *filtro di Kalman*. Quando si lavora, inoltre, in ambienti soggetti a disturbi elettromagnetici, il magnetometro potrebbe essere influenzato con un conseguente rischio di errata stima dell'orientazione e per questo motivo molti software permettono la sua disattivazione.

La distinzione principale nelle IMU avviene in base ai DOF (*Degree Of Freedom*) ovvero al numero di parametri indipendenti del sistema. Esistono IMU che vanno da 2 a 9 DOF, con alcune eccezioni con un numero ancora maggiore di gradi di libertà. Tuttavia, la maggior parte delle schede in commercio si divide in due tipologie:

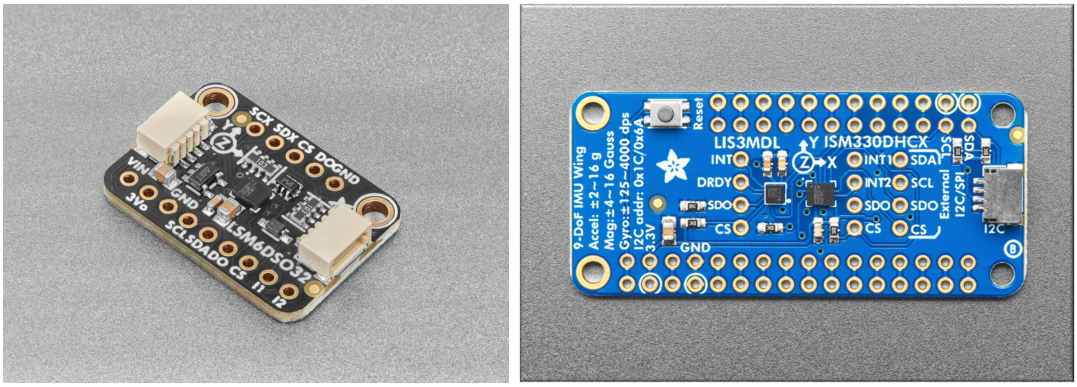
- Strumenti in grado di identificare l'accelerazione lineare su 3 assi con l'accelerometro e le rotazioni attorno agli assi con il giroscopio, identificate, appunto, a 6 DOF. Un esempio è mostrato in Figura 2a.
- Strumenti che prevedono la presenza di un magnetometro e che quindi sono definite a 9 DOF, (Figura 2b). Questi ultimi, naturalmente, hanno costi di produzione e di vendita più alti, risultando però più precisi e, spesso, venendo progettati come strumenti di fascia più alta, hanno performance migliori.

Altri aspetti che vanno considerati nella classificazione delle IMU sono il range di misura dei sensori e in numero dei bit di digitalizzazione. La prima caratteristica indica il massimo valore misurabile di un sensore mentre la seconda influenza la risoluzione dello strumento. Infatti, più i bit di digitalizzazione sono alti e più uno

strumento ha un'alta risoluzione, in cambio si ha una maggiore mole di dati da gestire e di conseguenza sono richieste maggiori capacità di calcolo.

Un'ulteriore distinzione tra questo tipo di sensori si basa sui diversi protocolli di comunicazione utilizzati per inviare dati. Nel presente lavoro è stato deciso di limitarsi al protocollo I2C, vista la sua ampia diffusione e la possibilità, non sempre presente in altri protocolli, di poter facilmente collegare più IMU in serie, caratteristica fondamentale per l'applicazione studiata.

Nel confrontare i dispositivi si è scelto valutare anche altre caratteristiche di interesse per l'applicazione, quali dimensioni, frequenza di campionamento e, soprattutto, accuratezza dei dati. La scelta di una IMU, infatti, non può basarsi su concetti assoluti e gerarchici, ma deve essere operata in base all'utilizzo finale ed effettivo del dispositivo [11].



(a) IMU 6 assi (b) IMU 9 assi

Fig. 2: Schede di sviluppo con sensori IMU integrati

2.3 Schede di sviluppo selezionate

Per l'applicazione nel presente lavoro di tesi è stata scelta una IMU MPU-6050 della TDK, integrata sulla scheda di sviluppo GY-521 mostrata in Figura 3. Trattandosi di un modello molto diffuso, è disponibile molta documentazione in merito, utile per la risoluzione di eventuali problemi o per l'implementazione di nuove funzionalità, partendo da algoritmi già elaborati in precedenza.

In ogni caso, è stato necessario procedere a un approfondimento sui sensori inerziali presenti in commercio al fine di individuare un possibile alternativa dello strumento attualmente utilizzato, in grado di migliorare le prestazioni generali dell'intera architettura.

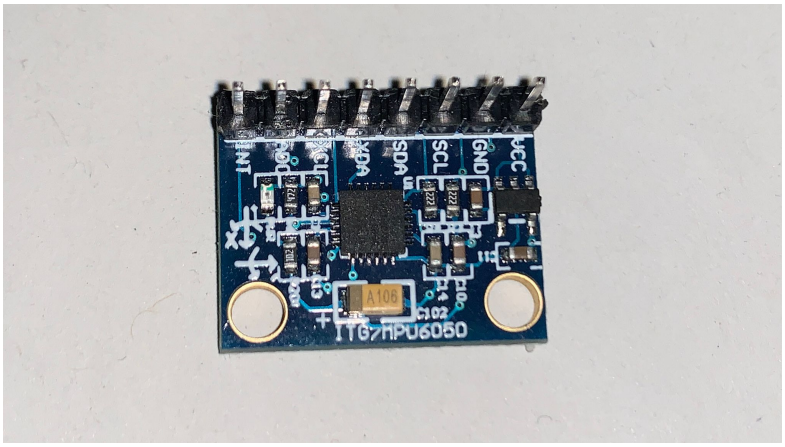


Fig. 3: Scheda GY-521 con sensore MPU6050

Sono stati analizzati 10 modelli presenti nei cataloghi delle principali case produttrici di IMU, ovvero *TDK*, *Stmicroelectronics* e *Bosch* al fine di selezionarne le tre opzioni migliori da acquistare e testare. I criteri di valutazione della scheda più adatta sono stati i seguenti:

- **L'azienda produttrice:** la scelta è ricaduta su un modello per ogni azienda produttrice, al fine di poter analizzare un più ampio spettro di approcci produttivi.
- **La libreria Data-Fusion disponibile:** molte aziende hanno librerie proprietarie, ma al tempo stesso sono state implementate molte altre librerie open-source di libero utilizzo, si è così preferito acquistare i sensori che garantivano una maggior disponibilità di librerie, oppure una migliore documentazione.
- **Il numero di assi:** la scelta è ricaduta preferibilmente su sistemi a 9 assi, con la presenza dunque di un magnetometro eventualmente disattivabile, in grado di assicurare la massima precisione possibile.
- **L'obsolescenza:** si è scelto di dare la precedenza ai modelli più recenti.

Tabella 1: Schede di sviluppo considerate, case produttrici e algoritmi

	Sensore	Data-Fusion
STmicroelectronics	LSM6DSOX	MotionFX
	LSM6DS3TR-C	MotionFX
	ISM330DHCX	MotionFX
	LSM9DS1	MotionFX
TDK	MPU-6050	DMP
	ICM20600	\
	MPU-9250	DMP
	ICM20948	DMP
Bosch	BMI088	BSXlite
	BNO055	BSXlite

Tabella 2: Schede di sviluppo considerate, caratteristiche tecniche

Sensore	Assi	Range accelerometro [g]	Range giroscopio [dps]	Range magnetometro [µT]
LSM6DSOX	6	±2, ±4, ±8, ±16	±125, ±250, ±500, ±1000, ±2000	/
LSM6DS3TR-C	6	±2, ±4, ±8, ±16	±125, ±245, ±500, ±1000,±2000	/
ISM330DHCX	6	±2, ±4, ±8, ±16	±125, ±250, ±500, ±1000, ±2000, ±4000	/
LSM9DS1	9	±2, ±4, ±8, ±16	±245, ±500, ±2000	±4000, ±8000, ±12000, ±16000
MPU-6050	6	±2, ±4, ±8, ±16	±250, ±500, ±1000, ±2000	/
ICM20600	6	±2, ±4, ±8, ±16	±125, ±250, ±500, ±1000, ±2000	/
MPU-9250	9	±2, ±4, ±8, ±16	±250, ±500, ±1000, ±2000	±4800
ICM20948	9	±2, ±4, ±8, ±16	±250, ±500,±1000, ±2000	±4900
BMI088	6	±3, ±6, ±12, ±24	±125, ±250, ±500, ±1000, ±2000	/
BNO055	9	±2, ±4, ±8, ±16	±125, ±250, ±500, ±1000, ±2000	±1300, ±2500

Infine, come evidenziato in rosso nella tabella 2, sono state selezionate le IMU ISM330DHCX della STmicroelectronics, BNO055 della Bosch e ICM-20948 della TDK. Per tutte e tre sono state scelte schede di sviluppo realizzate dalla casa *Adafruit* che si è occupata dell'hardware su cui sono integrate.

2.3.1 ISM330DHCX

Il modello ISM330DHCX mostrato in Figura 4, di dimensione 15x20mm, monta due connettori di tipo *STEMMA* che permettono di agevolare il collegamento e dunque la comunicazione di tipo I2C attraverso un aggancio più comodo ed efficiente. L'ISM330DHCX è a 6 assi e non possiede un magnetometro. Il principale algoritmo di data-fusion installabile è il *MotionFX* sviluppato dalla casa madre STmicroelectronics. Non è compatibile con sistemi Arduino e necessita quindi di un'apposita *Father Board* apposita di tipo *STM32* sviluppata dalla Adafruit [14].

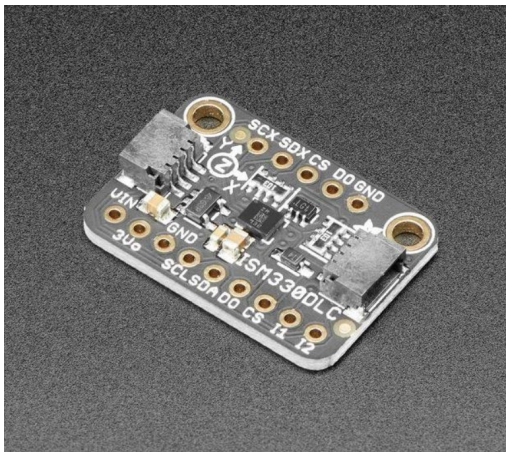


Fig. 4: Scheda Adafruit con sensore ISM330DHCX

2.3.2 BNO055

Anche il modello BNO055, mostrato in Figura 5, monta il sistema *STEMMA*, per agevolare la comunicazione I2C e le sue dimensioni sono paragonabili al modello precedente, ovvero 15x20mm. Le differenze consistono nella presenza del magnetometro che rende il sensore a 9 assi e nell'algoritmo di data-fusion, *BSX*, sviluppato da Bosch. La scheda è compatibile con sistema Arduino, risulta essere una delle più utilizzate per via della sua comprovata affidabilità e dispone della documentazione più esaustiva tra le schede selezionate.

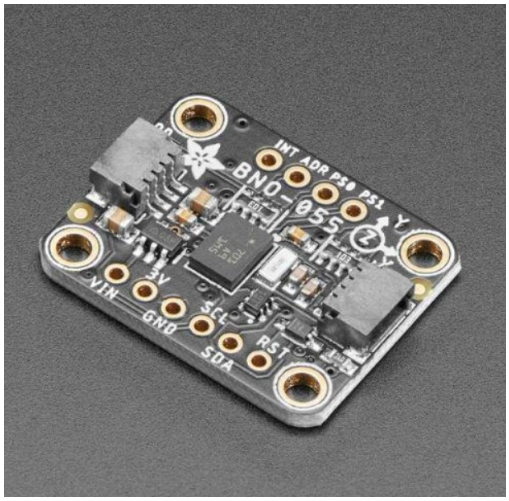


Fig. 5: Scheda Adafruit con sensore BNO055

2.3.3 ICM-20948

Il sensore IMU ICM-20948, mostrato in Figura 6, sviluppato da TDK InvenSense, rappresenta l'upgrade del modello MPU6050, utilizzato nel lavoro di tesi, dispone dello stesso algoritmo di sensor-fusion, il DMP (*Digital Motion Processor*) ampiamente testato e consente anche l'utilizzo di una serie di algoritmi secondari open-source sviluppati dagli utenti. Possiede un magnetometro e, come nei modelli analizzati in precedenza, la scheda di sviluppo, sempre di dimensioni 15x20mm, monta il sistema *STEMMA* per il collegamento I2C.

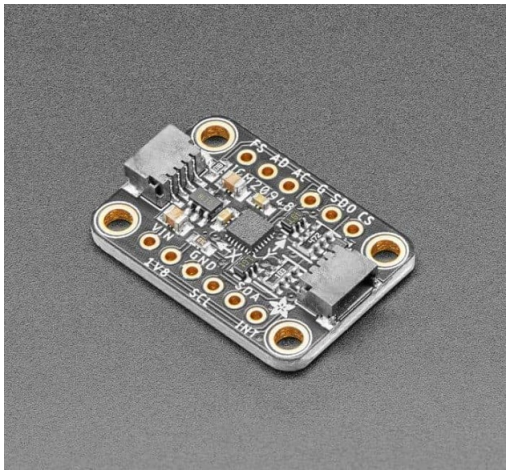


Fig. 6: Scheda Adafruit con sensore ICM-20948

3 Protocollo di Test sensori IMU

Al fine di quantificare le performance di differenti IMU e di confrontare l'accuratezza degli algoritmi di data fusion per la stima dell'orientazione, è stato implementato un protocollo che prevede una serie di prove finalizzate a testare le caratteristiche della IMU in modo ripetibile.

I test che vengono svolti si avvalgono normalmente dell'utilizzo di motori elettrici che azionano piattaforme rotanti calibrate, le quali riescono a garantire un elevato livello di precisione e ripetibilità della prova, con il limite di poter verificare solo un grado di libertà alla volta [15]. Negli ultimi anni un nuovo approccio a questo ambito di ricerca ha visto, in sostituzione delle piattaforme rotanti, l'utilizzo di bracci robotici industriali, che consentono di avere una regolazione più dinamica delle pose assunte dalla IMU, con la conseguenza di poter fare una valutazione più completa delle prestazioni, seppur a discapito della precisione [16]. A tal proposito è importante infatti precisare che quest'ultima metodologia non è in grado di fornire una valutazione con la stessa accuratezza della precedente, in quanto il robot presenta dei valori di precisione e ripetibilità inferiori. Il robot stesso, inoltre, ha a disposizione solo i dati fornitigli dagli *encoder* presenti nei motori, che rappresentano quindi solo la rotazione e la velocità angolare dei singoli giunti, tutte le altre grandezze vengono successivamente derivate da queste ultime, per cui potrebbero verificarsi delle imprecisioni dovute a eventuali approssimazioni in fase di elaborazione dei dati.

Il protocollo studiato, non avendo l'obiettivo di una meticolosa valutazione di precisione in senso assoluto, ma piuttosto di una comparazione sulle prestazioni globali di varie IMU e dei relativi algoritmi di Sensor-Fusion, ha previsto l'utilizzo di un bracco robotico con sensore montato sull'End Effector, come descritto in Figura 7. Tuttavia, per avere una valutazione completa e affidabile si è cercato comunque di simulare le prove su piattaforma, andando, dove possibile, a utilizzare un solo giunto per il movimento.

I risultati delle varie prove sono il frutto del confronto tra i dati registrati dalla IMU e quelli ottenuti dal robot.

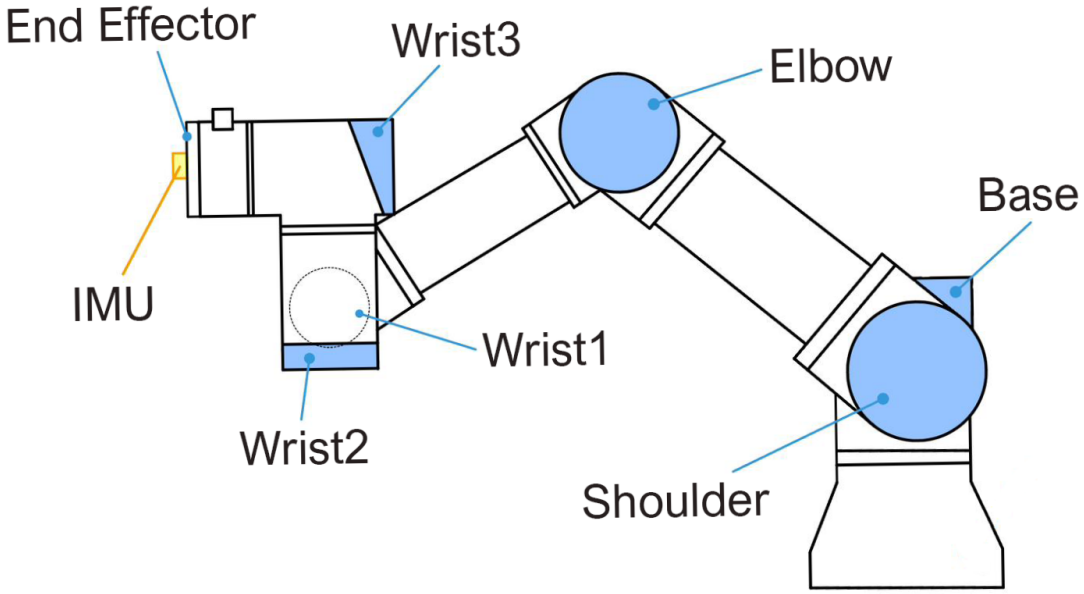


Fig. 7: Rappresentazione schematica del robot UR3 (Universal Robots) con il dispositivo IMU montato sull'End Effector

3.1 Obiettivo dei test

Il protocollo prevede sei diverse tipologie di prove che a loro volta si dividono in una serie di sottocategorie.

1. **Test dell'accelerazione lineare**

   Questo test permette di eccitare l'accelerometro lungo le tre direzioni principali del sensore mediante movimenti lineari dell'end-effector

2. **Test della velocità angolare**

   Questo movimento permette di eccitare il giroscopio mediante rotazioni di singoli giunti, sono state fette le seguenti prove:

   - Rotazione del giunto *Shoulder*.
   - Rotazione del giunto *Elbow*.
   - rotazione del giunto *Wrist 1*.

3. **Test sull'orientazione al variare della dinamica dei movimenti**

   Questo test valuta l'accumulo di errore all'alternarsi di condizioni statiche e dinamiche, sono state fatte le seguenti prove:

   - Rotazione di angoli *piccoli*.
   - Rotazione di angoli *grandi*.

4. **Test sull'orientazione al variare della velocità**

   Questo test valuta l'errore di orientazione percepito dal sensore al variare della sola velocità, sono state fatte le seguenti prove:

   - Movimento *lento*.
   - Movimento *veloce*.

5. **Test sull'orientazione al variare del tempo.**

   Questo test valuta l'accumulo di errore da parte del sensore all'aumentare del tempo di acquisizione, sono state fatte le seguenti prove:

   - *5 cicli* di rotazione.
   - *10 cicli* di rotazione.
   - *20 cicli* di rotazione.

6. **Test sull'orientazione in un percorso con rotazioni intorno ad assi misti.**

   Questo test valuta la capacità del sensore di tracciare correttamente l'andamento dell'orientazione quando è sottoposto a movimenti complessi.

   - Rotazione massima di *45°*.
   - Rotazione massima di *90°*.
   - Rotazione massima di *110°*.
   - Rotazione massima di *180°*.
   - Rotazione massima di *200°*.

I primi due test sono volti alla valutazione dello strumento IMU senza l'interferenza di algoritmi di elaborazione dati, vengono quindi testati i dati grezzi di accelerazione lineare e velocità angolare. I quattro test rimanenti invece sono volti alla valutazione della precisione dell'orientazione stimata dall'algoritmo di SensorFusion utilizzato, quando l'IMU è sottoposto a varie tipologie di movimento. Gli output di queste quattro prove sono quindi matrici omogenee di orientazione.

Come si vede in Figura 8, tutti i test, con eccezione dell'ultimo, sono stati condotti con tre configurazioni diverse in modo da poter testare singolarmente i tre assi propri della IMU e valutarne eventuali differenze.

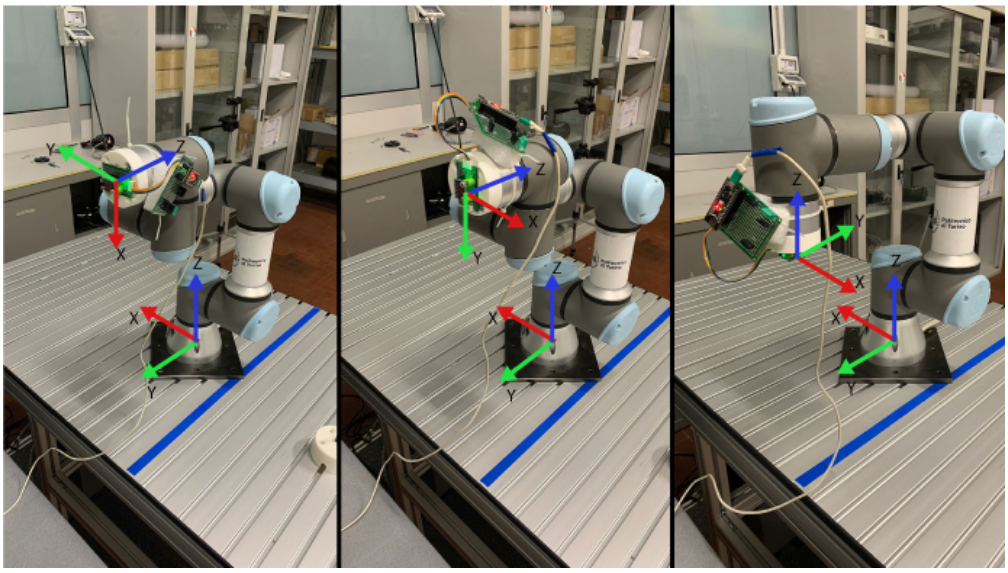


Fig. 8: Esempio di configurazioni dei tre assi, l'asse da testare è orientato perpendicolarmente al terreno

3.2 Setup

Per la prova è stata usata inizialmente solo una scheda GY-521 che monta il sensore IMU MPU6050, quest'ultimo comunica con il PC attraverso un Arduino Nano. Il braccio robotico utilizzato è il cobot UR3 della Universal Robots. Per la prova è stata disegnata, attraverso il software SolidWorks, un'interfaccia, visibile in Figura 9, che permetta allo strumento di rimanere fissato alla flangia del robot, muovendosi in maniera solidale con quest'ultima, eseguendo così una traiettoria nota.

Dal lato software è stato utilizzato il programma MATLAB, in grado di mettere in relazione tre set di dati, indipendenti fra loro:

- **I dati registrati dal robot:** sono tutte le misurazioni in termini di posizione velocità e orientazione dell'EE ottenute come output del controller del robot. Tali dati sono utilizzati come valori di confronto per ciascuna prova.
- **I dati registrati dalla IMU:** sono i valori registrati dallo strumento testato in quel momento e inviati al PC tramite il microcontrollore, per poi esse-

re successivamente elaborati in un apposito script MATLAB. Anche questi valori variano in base al tipo di prova che si sta effettuando.

- **Il tempo misurato dal PC:** è ottenuto da una funzione MATLAB, è unico e associato a ognuna delle misurazioni, sia del robot sia della IMU. In questo modo si ottengono i dati sincronizzati e possono essere individuati gli eventuali ritardi nell'acquisizione della IMU, dovuti allo strumento stesso.

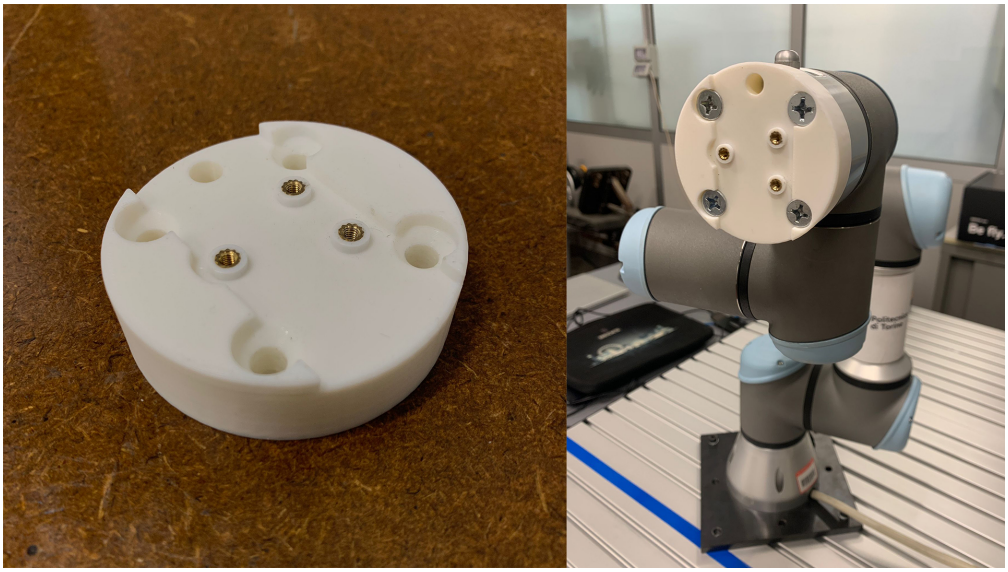


Fig. 9: Supporto del dispositivo

3.3 Prove e risultati attesi

3.3.1 Test dell'accelerazione lineare

In questa prova si vanno a testare i 3 accelerometri presenti sulla IMU. Come primo passaggio è stato creato uno script MATLAB in grado di generare una traiettoria rettilinea. In particolare, è stato studiato un profilo di velocità in spazio cartesiano, visibile in Figura 10, in cui compaiono tutte le componenti nulle, eccetto una componente di velocità lineare, modellata con un profilo a triangolo in cui vengono massimizzati i tempi di accelerazione e decelerazione, oggetto della ricerca. Nel caso specifico è stato programmato un percorso rettilineo di 35 cm con velocità massima pari a 0.35 m/s raggiunta a metà percorso.

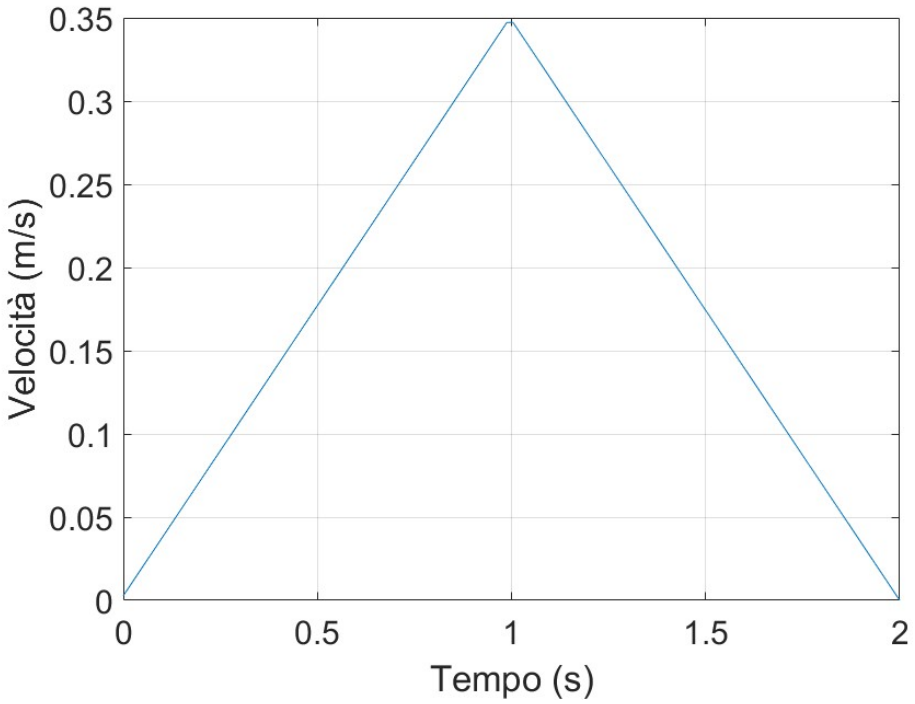


Fig. 10: Profilo di velocità in spazio cartesiano

Successivamente tramite l'utilizzo della matrice Jacobiana inversa, calcolata volta per volta sulla base delle varie configurazioni assunte dall'UR3, è stato individuato il profilo di velocità equivalente in spazio giunti, Figura 11, dal quale è stato ricavato l'andamento delle rotazioni e delle accelerazioni angolari, Figura 12, integrando e derivando rispettivamente il dato di velocità.

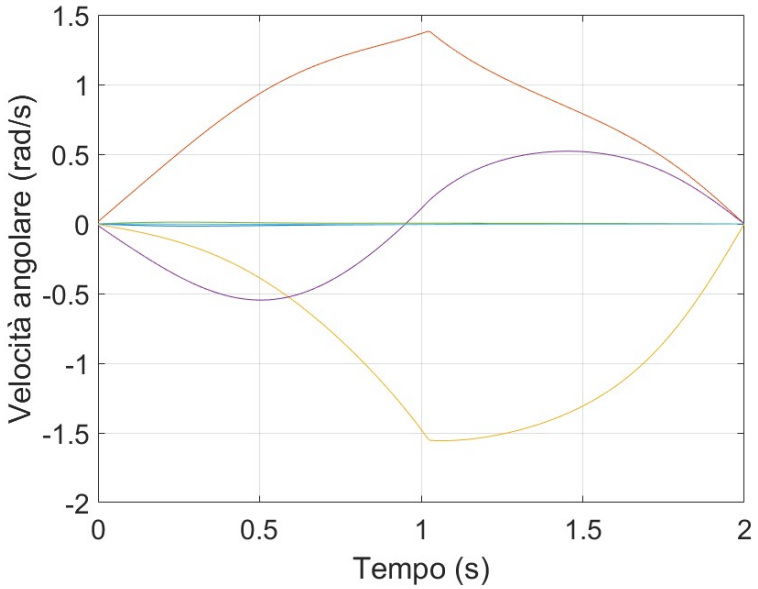


Fig. 11: Profilo di velocità in spazio giunti

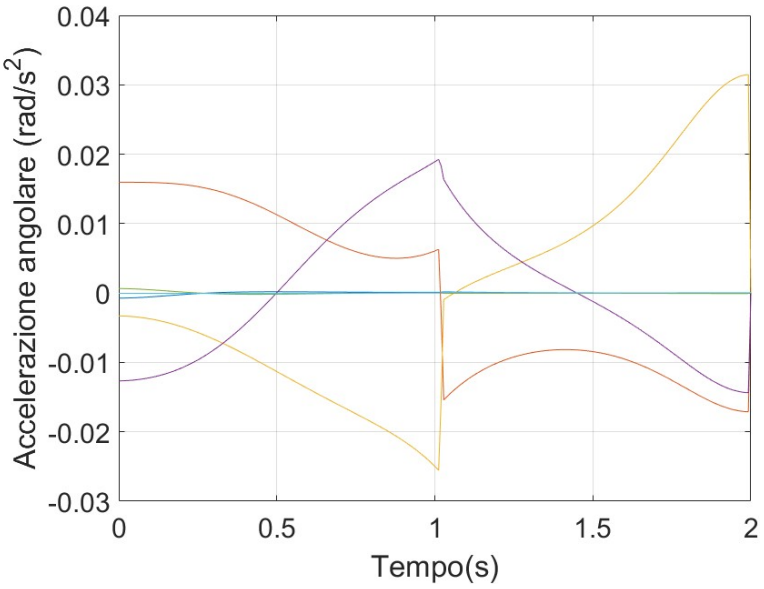


Fig. 12: Profilo di accelerazione in spazio giunti

Dopo aver eseguito la pianificazione della traiettoria è stato possibile far eseguire il movimento al robot il quale si muove seguendo una linea retta nello spazio, parallela a uno degli assi del sistema di riferimento *Base* del robot, indicato in

Figura 8. Si dispone quindi il robot in una configurazione tale da allineare uno degli assi propri della IMU alla retta di movimento e si effettua la prova, come si può vedere in Figura 13.

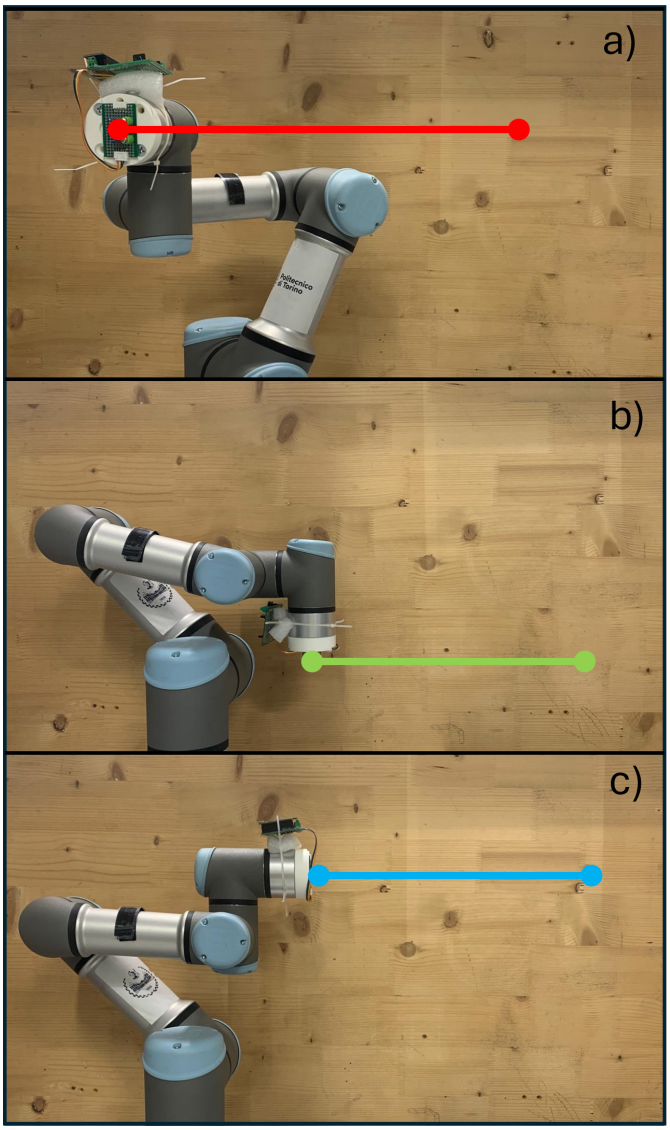


Fig. 13: Traiettorie lineari sugli assi X (a), Y (b) e Z (c) del sensore

In questa prova al robot è richiesto di registrare la velocità lineare dell'TCP in m/s, (Figura 14), la quale verrà in seguito derivata per ottenere l'accelerazione. La IMU invece misura i dati grezzi registrati dall'accelerometro. I dati registrati sono quindi convertiti in post processing da bit a m/s^2 effettuata successivamente, si ottenendo così il dato di accelerazione sui tre assi, (Figura 15). Si noti come

l'asse che punta verso il basso (che non è quello preso in esame dalla prova) registri un'accelerazione negativa 9.81 m/s^2, questo perché orientato in verso opposto all'accelerazione di gravità.

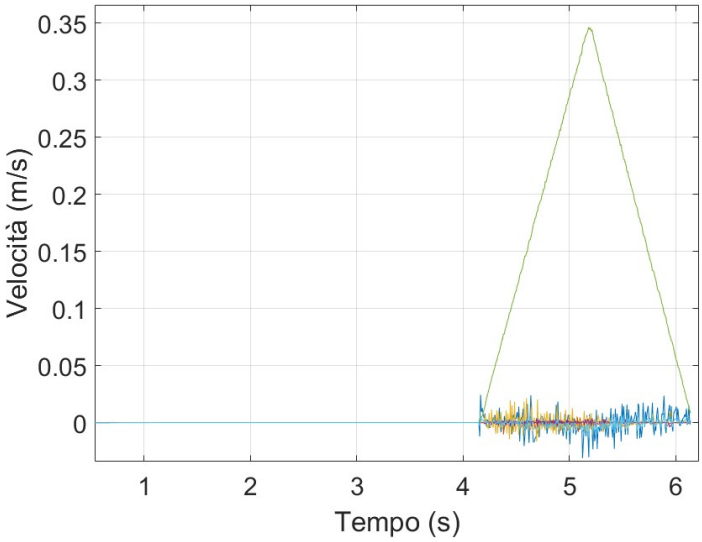


Fig. 14: Acquisizioni del robot

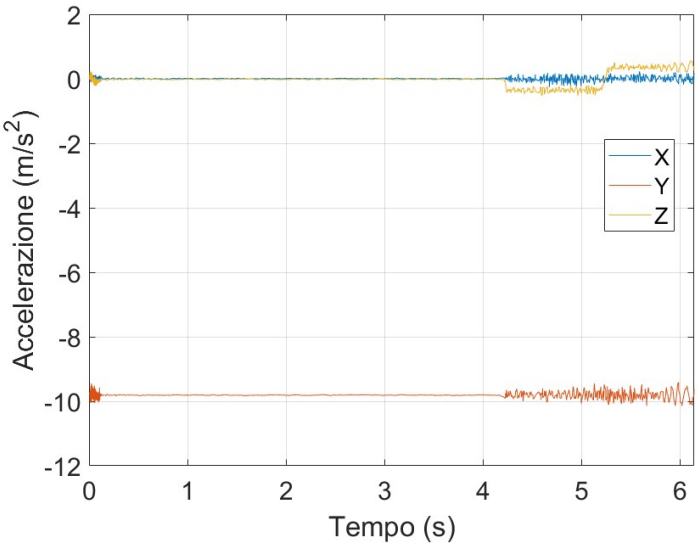


Fig. 15: Acquisizioni della IMU, accelerazione dell'asse Z del sensore

Inoltre, a causa della rumorosità dei dati, è stato applicato un filtro *Butterworth* passa-basso del 2° ordine con una frequenza di taglio di 20 Hz, che, per i dati campionati a 125 Hz. Infine, i dati filtrati del robot e quelli acquisiti mediante IMU vengono confrontati per calcolare l'errore medio ed eventuali ritardi di acquisizione come si mostrato nelle Figure 16, 17 e 18.

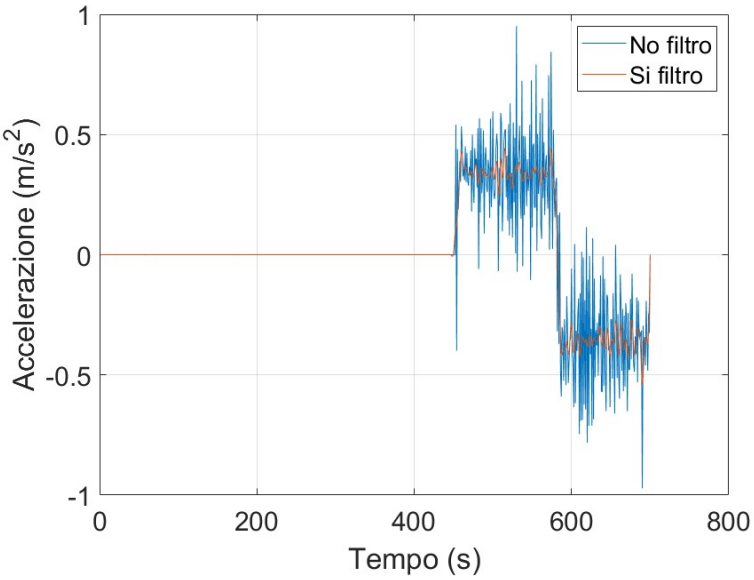


Fig. 16: Dati di accelerazione del robot con applicazione del filtro, asse Z del sensore

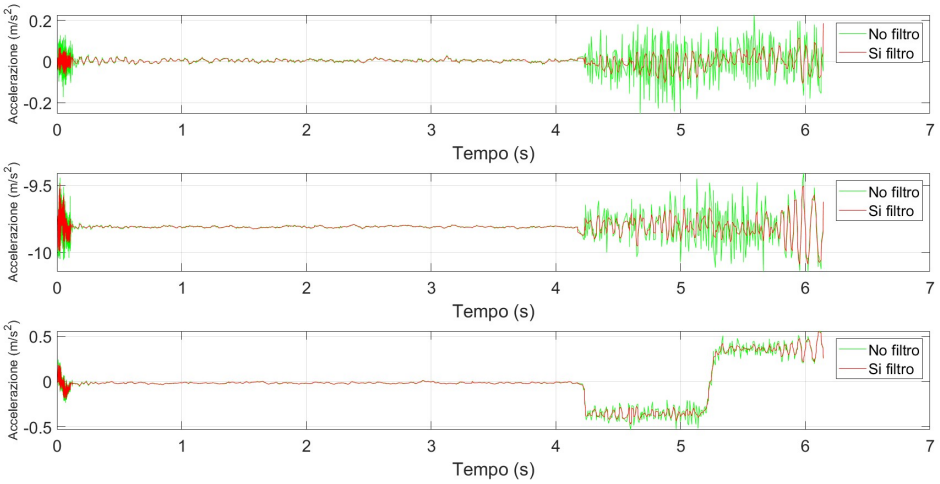


Fig. 17: Dati di accelerazione della IMU con applicazione del filtro

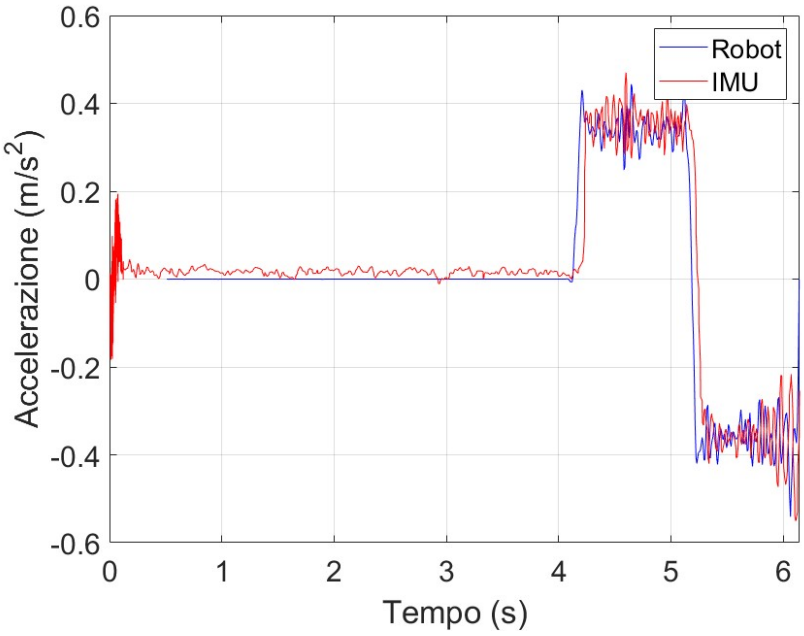


Fig. 18: Confronto tra accelerazioni IMU e robot filtrate, asse Z del sensore

Dalle prove ottenute si può notare una discreta sovrapponibilità tra le acquisizioni, che risente tuttavia di un minimo rumore di fondo e della presenza di un ritardo nell'acquisizione dei dati IMU. Quest'ultimo potrebbe essere imputato sia a un ritardo effettivo di acquisizione da parte dello strumento sia ad un ritardo nella registrazione dati da parte del PC. Si è scelto comunque di tenerne conto nella fase di calcolo dell'errore poiché questo potrebbe essere presente anche in un'applicazione reale, visto che l'hardware utilizzato sarebbe il medesimo.

Per valutare più nello specifico l'entità dell'errore i dati sono stati ulteriormente elaborati, andando a separare la fase di accelerazione da quella di decelerazione (Figura 19) in modo da poter osservare eventuali differenze di errore tra le due fasi. Per individuare il punto di divisione tra accelerazione e decelerazione in maniera univoca è stato preso in considerazione l'istante temporale in cui il robot ha rilevato il cambio di segno dell'accelerazione durante il movimento. Sono così considerati tra gli errori anche i ritardi di acquisizione delle IMU, i quali andranno a pesare sugli errori medi.

In figura 19.b è osservabile questo concetto. Nell'istante in cui inizia la decelerazione (tratto verde), infatti, la IMU non percepisce immediatamente il cambio di segno e restituisce comunque un valore positivo.

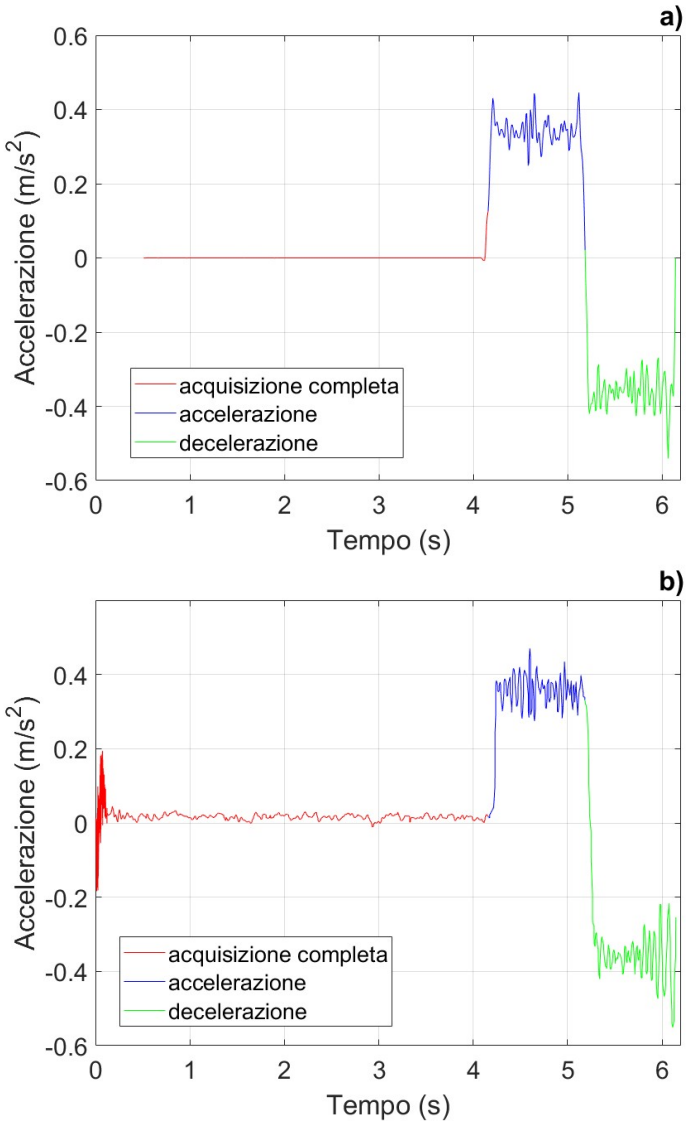


Fig. 19: Divisione tra accelerazione e decelerazione, caso robot (a) e caso IMU (b), asse Z del sensore

Successivamente è stata calcolata la differenza dei due valori acquisiti per ogni intervallo di tempo, andando poi a trovare il valore assoluto è stato possibile creare il grafico dell'andamento dell'errore nelle due fasi, dal quale a sua volta è stato calcolato un errore medio di acquisizione. La procedura è stata ripetuta anche per i test sugli altri due assi, i grafici sono illustrati nelle Figure 20 e 21.

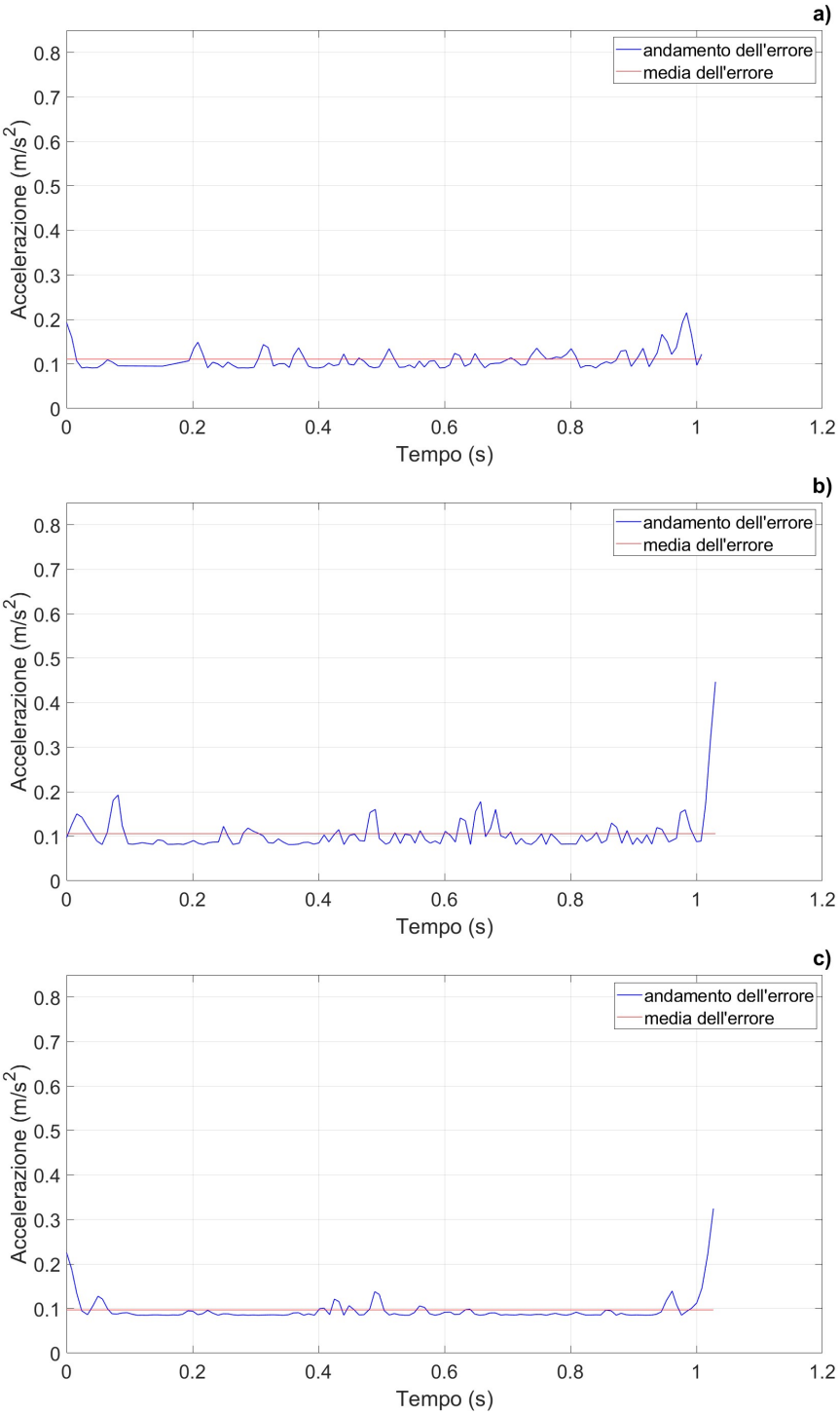


Fig. 20: Errore in fase di accelerazione lungo gli assi X (a), Y (b) e Z (c) del sensore

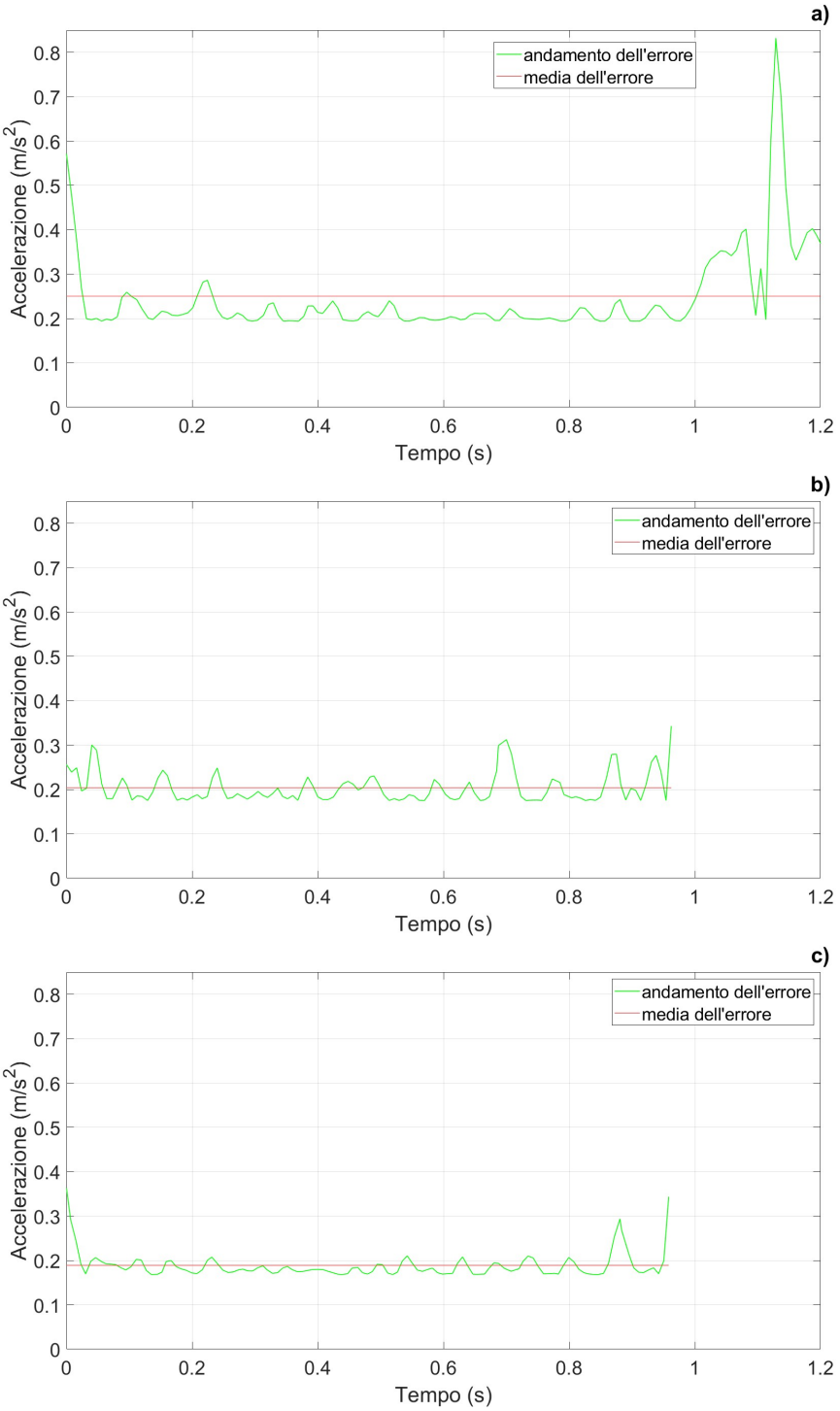


Fig. 21: Errore in fase di decelerazione lungo gli assi X (a), Y (b) e Z (c)

Vengono di seguito riportati i risultati delle prove di accelerazione, decelerazione e totali.

Tabella 3: Risultati della prova di accelerazione lineare

	errore sull'accelerazione [m/sˆ2]	errore sulla decelerazione[m/sˆ2]	errore totale [m/sˆ2]
Asse X	0.1108	0.2505	0.1807
Asse Y	0.1059	0.2039	0.1549
Asse Z	0.0971	0.1891	0.1431

Dai dati emerge un errore costante sugli assi X e Y, superiore a quello presente sull'asse Z. Relativamente alla fase di decelerazione, è possibile rilevare che in tutti i casi l'errore risulta maggiore rispetto all'accelerazione. Come detto, tale condizione è imputabile al ritardo di qualche decimo di secondo nell'acquisizione dei dati da parte della IMU, si possono infatti notare dei picchi più o meno accentuati in corrispondenza di inizio e fine di tutti i grafici. Come detto in precedenza, si è scelto di mantenere questi picchi poiché potrebbero essere riscontrati anche nelle applicazioni reali.

3.3.2 Test della velocità angolare

In questa prova si vanno a testare i 3 giroscopi presenti sulla IMU. Per farlo si mette in rotazione un giunto del robot, in modo che faccia ruotare lo strumento attorno a uno dei suoi propri assi. È stato pianificato infatti un profilo di velocità in spazio giunti a trapezio con la parte costante molto accentuata per avere un range maggiore di misurazione, Figura 22.

La prova è stata effettuata facendo ruotare tre diversi giunti: *Shoulder*, *Elbow* e *Wrist 1*, con l'obiettivo di indagare se la distanza dall'asse di rotazione influisce sulle prestazioni dei giroscopi.

Le tre prove consistono in una rotazione di 100° del giunto designato in 3 secondi, implicando una velocità del tratto costante di 0.582 rad/s, il movimento è descritto in Figura 23.

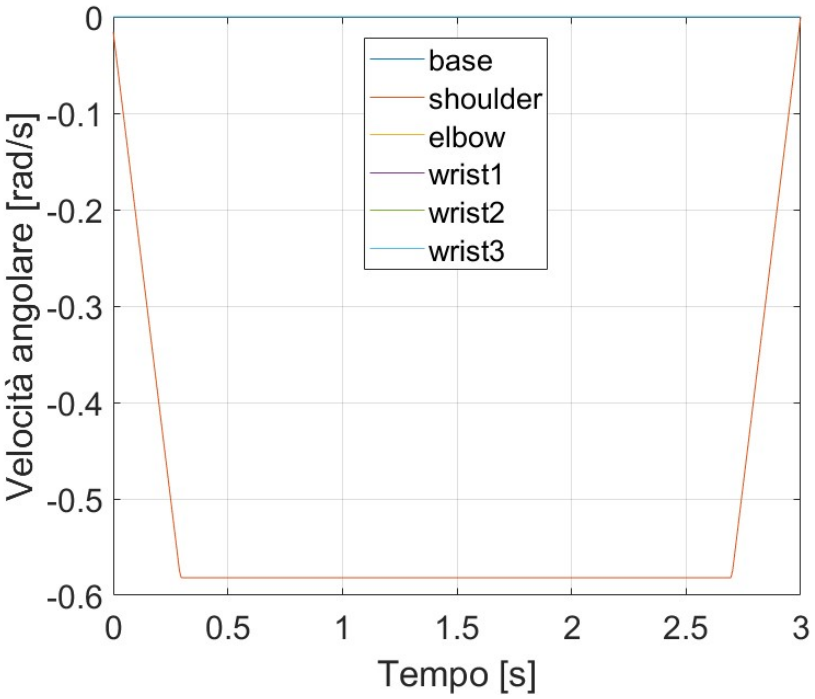


Fig. 22: Profilo di velocità spazio giunti

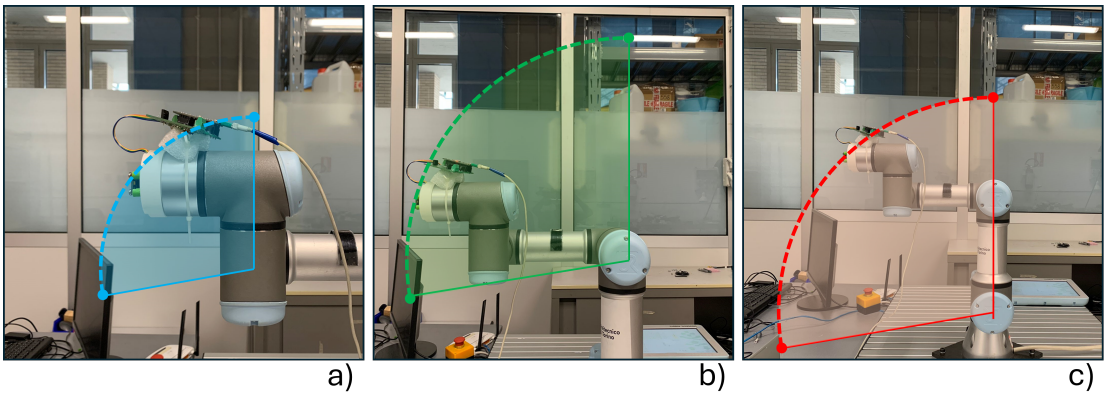


Fig. 23: Rotazione dei giunti Wrist 1 (a), Elbow (b) e Shoulder (c)

I dati di velocità angolare del singolo giunto acquisiti dal robot vengono confrontati con in dati di velocità angolare dei giroscopi della IMU, acquisiti in bit ed elaborati in una seconda fase per ottenere rad/s, Figura 24. Dalla sovrapposizione dei dati si possono confrontare i valori medi per trovare di conseguenza l'errore e anche valutare eventuali ritardi di acquisizione.

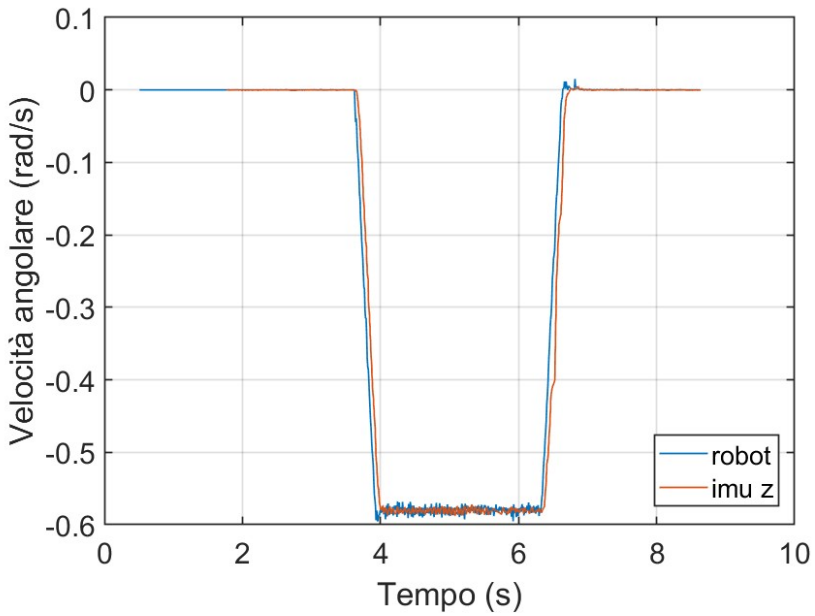


Fig. 24: Confronto tra la velocità angolari IMU e robot, asse Z del sensore, rotazione del giunto Elbow

In questa prova è possibile notare una prestazione migliore rispetto alla prova sull'accelerazione, con un rumore di fondo notevolmente inferiore, per il quale non è stato necessario l'utilizzo di filtri. A differenza della prova precedente, risultano al tempo stesso molto più precisi anche i dati acquisiti dal robot e tale vantaggio è dovuto alla natura stessa della prova, dal momento che essa necessita della rotazione di un solo giunto del robot alla volta, con una riduzione generale di vibrazioni.

Per ampliare lo spettro dei dati a disposizione, è stato successivamente svolto uno studio più approfondito, nel quale è stato isolato il tratto di movimento a velocità costante ed è stato calcolato l'errore medio in queste sezioni, con lo scopo di valutare l'effettiva entità dell'errore nelle rilevazioni, escludendo le fasi di transitorio in cui il giunto accelera o decelera. Poiché le prove di *Shoulder*, *Elbow* e *Wrist1* hanno tempi di inizio e fine acquisizione differenti tra loro, non è possibile individuare un istante unico in cui inizia e finisce l'acquisizione. Il tratto è stato dunque isolato manualmente per ognuna delle prove, i tratti presi in considerazione sono visibili nelle Figure 25, 26 e 27.

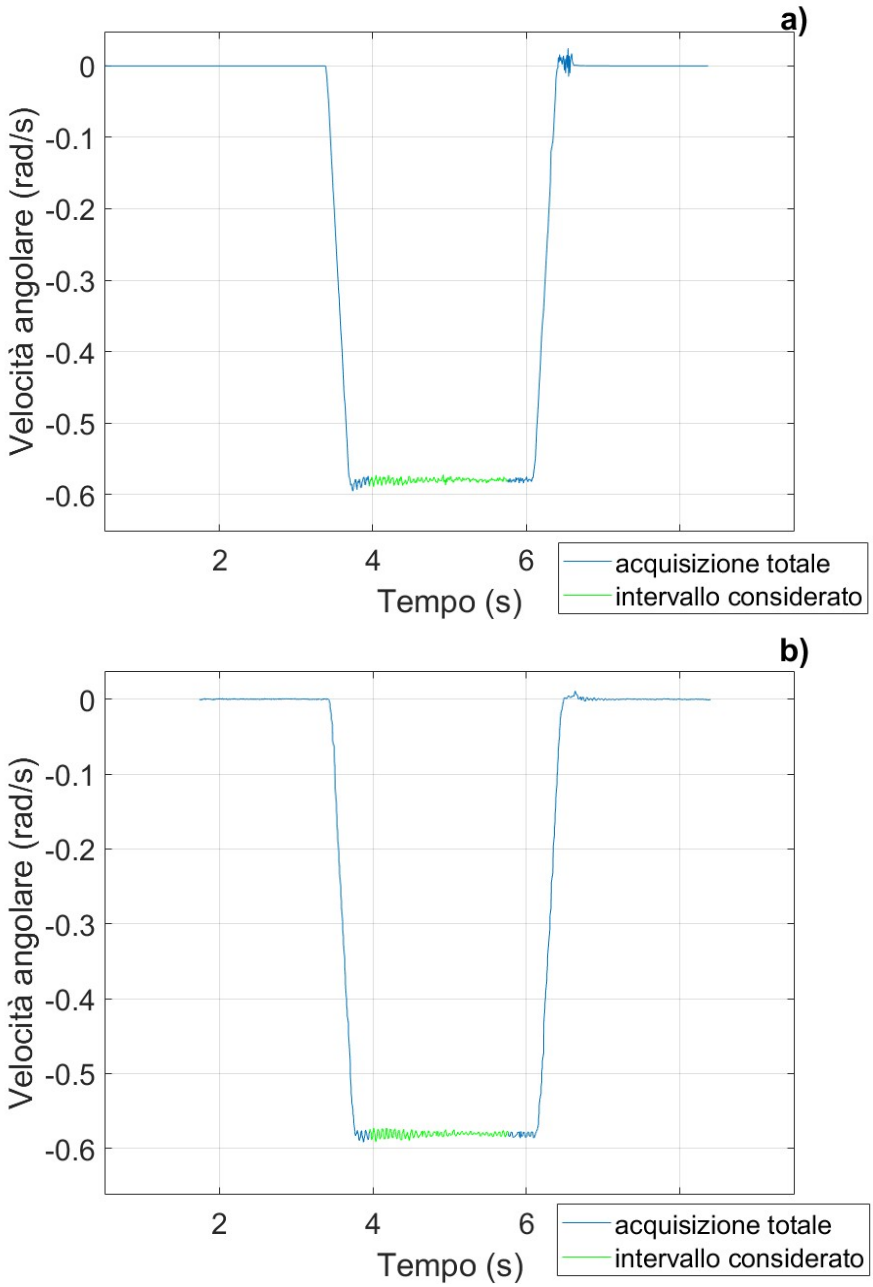


Fig. 25: Tratti isolati sulle acquisizioni di robot (a) e IMU (b), rotazione attorno al giunto "Shoulder", asse Z del sensore

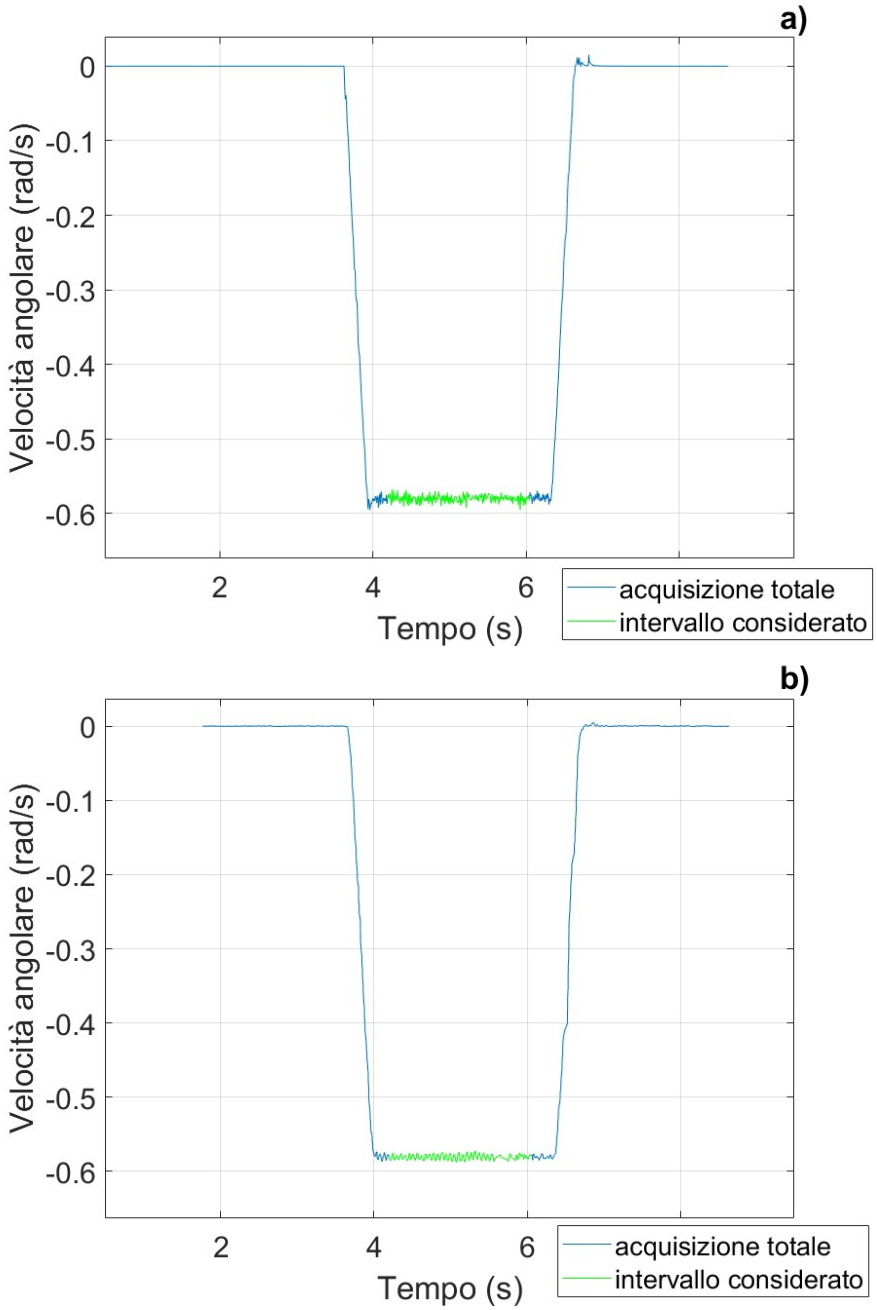


Fig. 26: Tratti isolati sulle acquisizioni di robot (a) e IMU (b), rotazione attorno al giunto "Elbow", asse Z

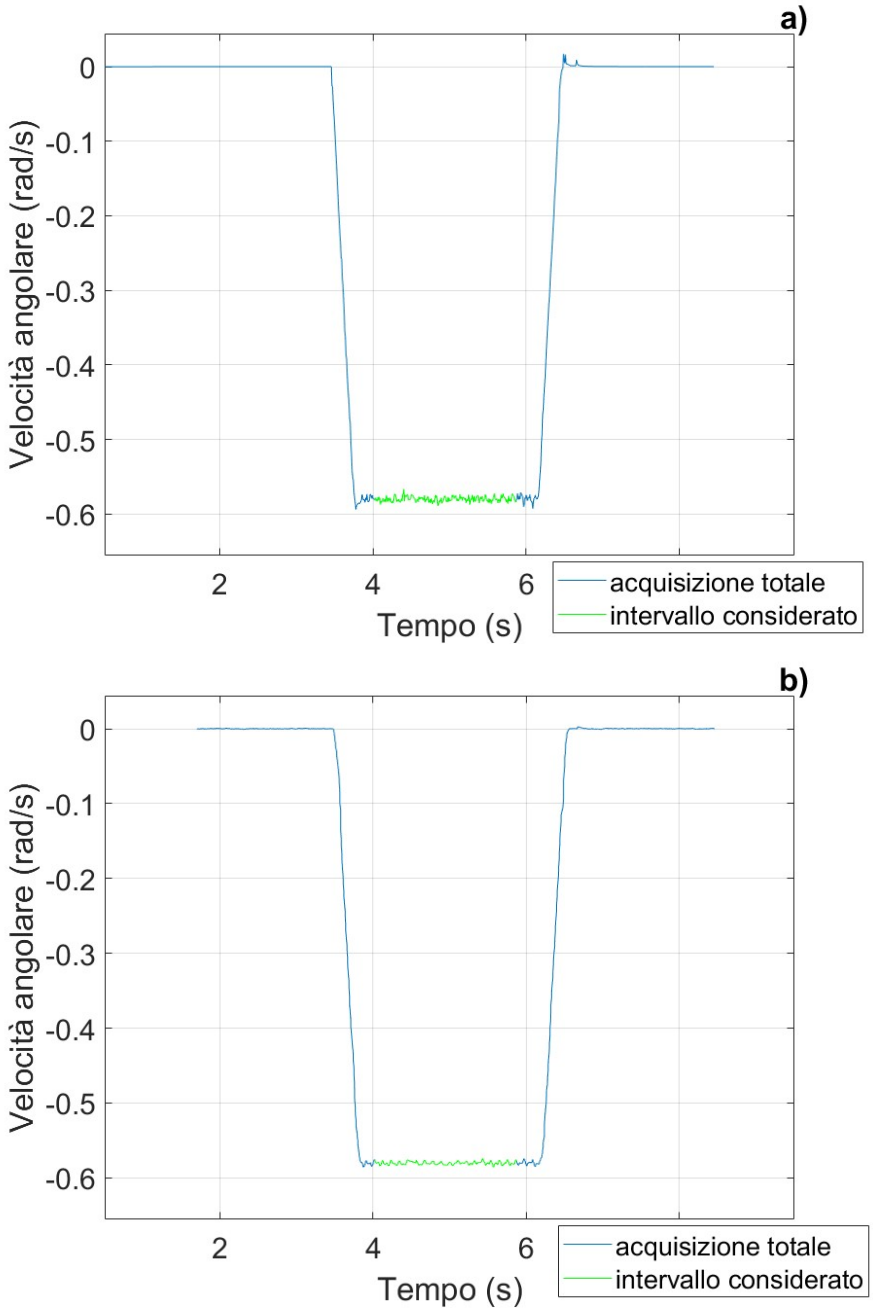


Fig. 27: Tratti isolati sulle acquisizioni di robot (a) e IMU (b), rotazione attorno al giunto "Wrist 1", asse Z del sensore

La prova è stata eseguita ruotando attorno ai tre assi propri della IMU: X, Y e Z e calcolando il valore medio dell'errore per tutte le prove. Di seguito sono riportati i risultati ottenuti.

Tabella 4: Risultati della prova di velocità angolare

	Asse X	Asse Y	Asse Z
Errore medio sulla rotazione del giunto "Shoulder" [rad/s]	$1.7 \cdot 10^{-3}$	$7.2 \cdot 10^{-3}$	$9.4 \cdot 10^{-4}$
Errore medio sulla rotazione del giunto "Elbow" [rad/s]	$2.0 \cdot 10^{-3}$	$6.2 \cdot 10^{-3}$	$5.7 \cdot 10^{-4}$
Errore medio sulla rotazione del giunto "Wrist 1" [rad/s]	$1.9 \cdot 10^{-3}$	$6.3 \cdot 10^{-3}$	$4.5 \cdot 10^{-4}$

Come già evidente dai grafici, l'errore risulta trascurabile. Si nota inoltre che non vi è particolare differenza tra le rilevazioni al variare della distanza della IMU dal centro di rotazione e ciò implica che quest'ultimo non influisce sulle prestazioni delle acquisizioni. Come nel caso dell'accelerazione lineare, si può infine notare che l'asse Z ha prestazioni generalmente migliori rispetto agli altri due assi.

3.3.3 Introduzione ai test sull'orientazione

In tutte le prove seguenti, vengono analizzati i dati ricavati dalla IMU ed elaborati tramite l'algoritmo di data-fusion open-source *Rcmags imuFilter* per sistema Arduino che implementa una variante del filtro di Kalman. Questo programma permette di ottenere le informazioni riguardo l'orientazione nello spazio della IMU sotto forma di quaternioni, andando ad interpretare i dati raccolti da accelerometro e giroscopio ad una frequenza di 100Hz. I dati di confronto saranno le orientazioni del TCP ottenute dal robot partendo dalla posizione dei giunti ed applicando la cinematica diretta [17].

Per elaborare i dati ottenuti da IMU e robot è stato sviluppato un programma MATLAB che compie la seguente serie di operazioni:

1. **Load dei dati:** il programma chiede in input un vettore 4xn rappresentante le acquisizioni della IMU in quaternioni, un vettore 1xn rappresentante gli istanti a cui le acquisizioni sono state fatte, un vettore 6xm in cui sono contenute tutte le configurazioni dei giunti del robot e un vettore 1xm in cui ci sono gli istanti di tempo relativi alle configurazioni. È importante sottolineare che, poiché la frequenza di acquisizione del robot è maggiore di quella della IMU, la variabile m risulta sempre maggiore della variabile n.

2. **Inizializzazione dei dati:** partendo dai dati grezzi di IMU e robot si ottengono matrici di rotazione, tempi sincronizzati e valori in Roll Pitch Yaw per rendere i dati di più facile lettura.

3. **Sovrapposizione delle matrici:** l'obiettivo è quello di uniformare il sistema di riferimento delle due serie di matrici di rotazione, ottenendo così due matrici sovrapposte nel primo istante di acquisizione.

4. **Interpolazione dei dati:** viene fatta un'interpolazione lineare sulle acquisizioni del robot in modo da sincronizzare i tempi con quelle della IMU e poter fare il confronto.

5. **Calcolo dell'errore:** si tagliano le acquisizioni precedenti all'inizio del movimento e si confrontano i quaternioni acquisiti dalla IMU con quelli ottenuti dai dati del robot relativi allo stesso istante attraverso il comando MATLAB *dist*. Si ottiene quindi in output un valore di errore per ogni istante di tempo espresso in radianti, che rappresenta la differenza tra l'orientazione del robot e quella della IMU ipotizzando una singola rotazione attorno a un asse generico.

L'errore, espresso come rotazione attorno ad un asse generico, è calcolato per ogni acquisizione istante per istante. Non è quindi possibile definire con certezza una direzione verso la quale l'errore è più accentuato. È invece possibile, attraverso il calcolo del valore medio degli errori, individuare un cono di incertezza attorno ai tre assi propri (Figura 28).

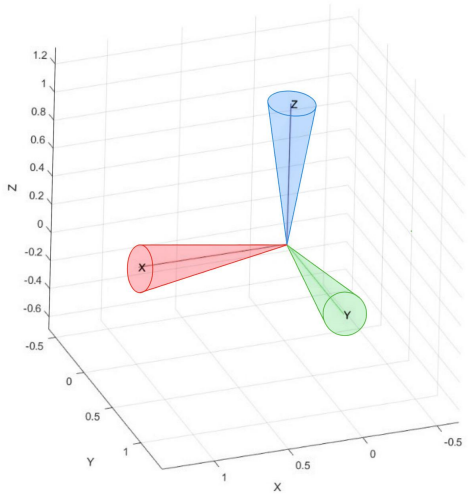


Fig. 28: Ipotesi di terna di orientazione con coni di incertezza

3.3.4 Test sull'orientazione al variare della dinamica dei movimenti

Nel primo test, sono state fatte diverse prove in cui il robot ha assunto varie posizioni nello spazio cartesiano. Ogni prova consiste in una rotazione della base del robot in un arco con pause in orientazioni specifiche per permettere alla IMU di registrarle, Figura 29.

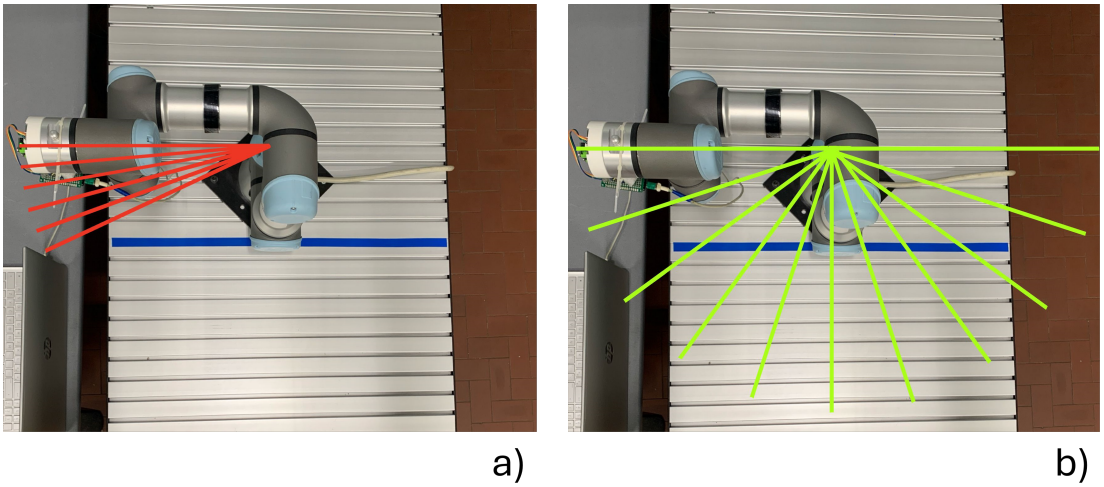


Fig. 29: Angoli di rotazione da 5° (a) e 18° (b)

Il test è diviso in due prove differenti, nella prima si raggiungono 5 pose a una distanza di 5° l'una dall'altra, coprendo così complessivamente un range di 25°. Nella seconda si raggiungono 10 pose a una distanza di 18° l'una dall'altra coprendo così complessivamente un range di 180°. Per entrambe le prove l'acquisizione è continua e prevede gli stessi intervalli di tempo: una fase cambio orientazione di 3 secondi e una fase statica di 4 secondi in cui il robot rimane fermo. Ne consegue che le velocità di movimento saranno diverse in base alla prova: minori per angoli da 5° e maggiori per angoli da 18°. Lo scopo della prova è quello di indagare la divergenza, in termini di errore di orientazione, tra le acquisizioni statiche e quelle dinamiche, in più si può fare una valutazione preventiva su quanto l'entità della rotazione in combinazione con la velocità influisce sulla capacità dell'algoritmo di stimare correttamente l'orientazione, i risultati sono visibili nelle Figure 30 e 31.

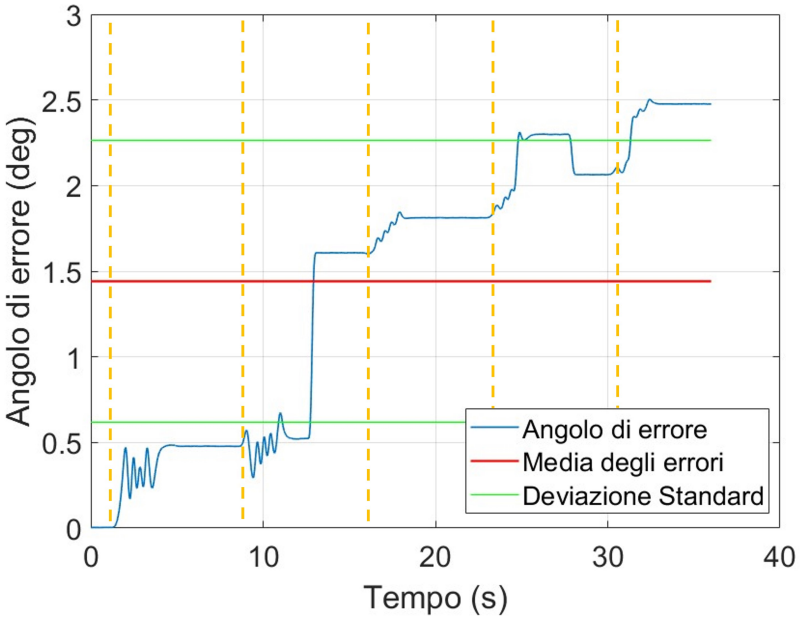


Fig. 30: Angoli di rotazione 5°, rotazione attorno all'asse Z del sensore

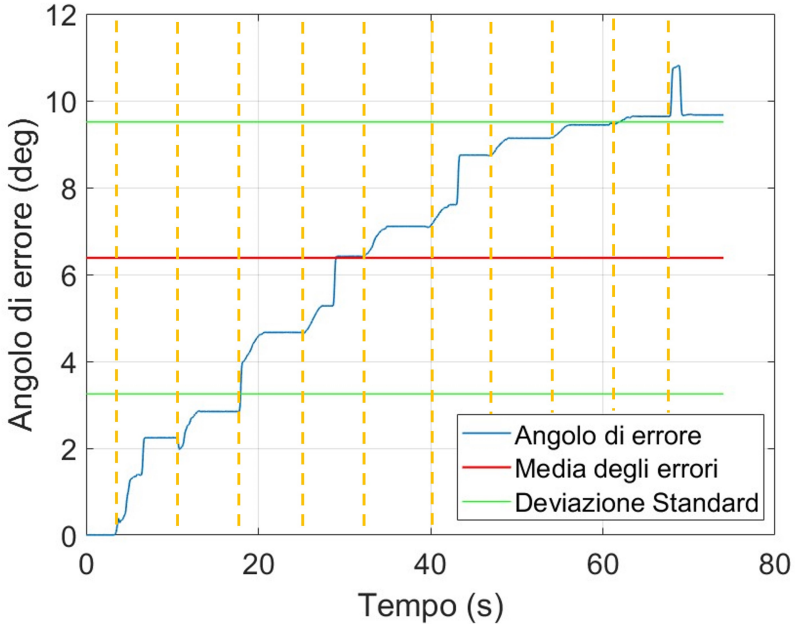


Fig. 31: Angoli di rotazione 18°, rotazione attorno all'asse Z del sensore

Di seguito vengono riportati i valori di angolo medio di errore e deviazione standard ricavati da ogni prova.

Tabella 5: Risultati della prova di orientazione degli assi

		Angolo medio di errore [deg]	Deviazione standard [deg]
5°	X	0.6033	0.3774
	Y	0.6682	0.3971
	Z	1.3940	0.8696
18°	X	3.0895	1.6409
	Y	5.7635	2.9998
	Z	6.4247	3.0562

La prima osservazione è che in entrambe le prove sono presenti delle fasi con errore costante in corrispondenza del momento di staticità, da ciò si può evincere che l'errore viene accumulato solo nella fase di movimento e che non prosegue con un drift quando il sensore si ferma. Si può inoltre notare che nella seconda prova si ha un notevole aumento dell'angolo di errore, sia per quanto riguarda il picco massimo sia per quanto riguarda il valore medio. Ciò è però bilanciato dall'aumento del range di movimento. Prendendo ad esempio le rotazioni attorno all'asse Z, di cui sono stati riportati anche i grafici, nel primo caso si ha un errore medio di 1.4° su un range di 25°, ovvero circa il 5,6% mentre nel secondo si ha un errore medio di 6.4° su un range di 180°, ovvero circa il 3,6%. In questa fase tuttavia non è ancora possibile comprendere in che percentuale l'aumento dell'errore sia dovuto al cambio di velocità e in che percentuale sia dovuto all'ampiezza dell'angolo percorso.

3.3.5 Test sull'orientazione al variare della velocità

In questa prova la IMU viene fatta ruotare di 90° con due diverse velocità, in modo da valutare quanto influisce la rapidità dei movimenti sulla precisione di tracciamento dell'algoritmo. Nello specifico, il robot impiega 10 secondi a compiere la rotazione nella prova denominata "movimento *Lento*" e 2 secondi per quella denominata "movimento *Veloce*". Dai risultati, visibili dalle Figure 32 e 33, si può osservare la differenza dell'angolo di errore nelle due prove confrontate.

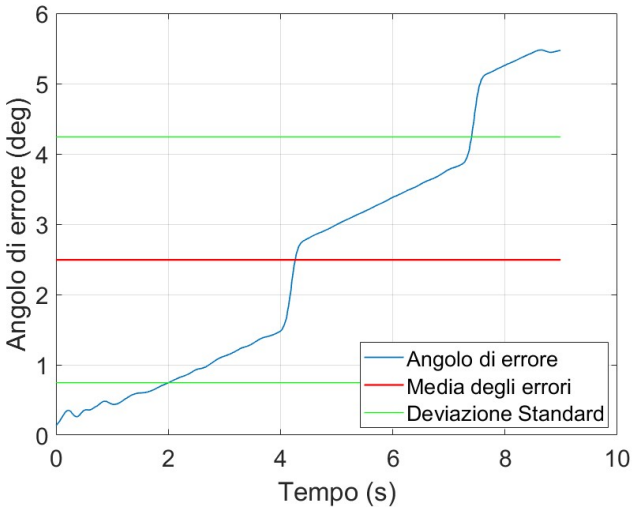


Fig. 32: Errore sul movimento lento, rotazione attorno all'asse Z del sensore

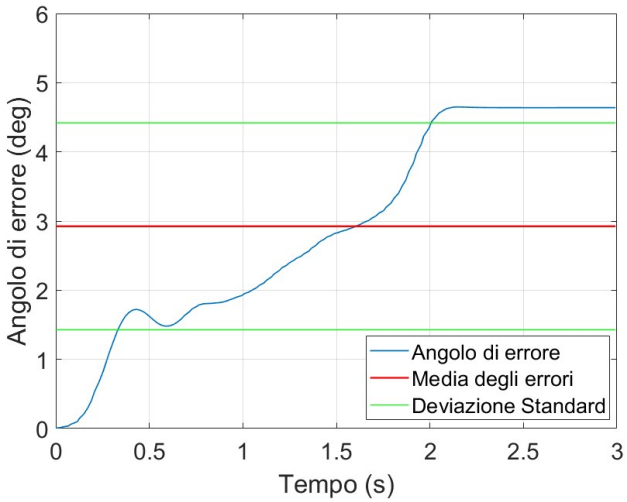


Fig. 33: Errore sul movimento veloce, rotazione attorno all'asse Z del sensore

La prima osservazione che si può fare guardando i grafici è che alla fine delle due prove si raggiunge un errore massimo accumulato simile. Si può inoltre notare dalla tabella 6 che, rimanendo in un range velocità di movimento simile a quelle umane, l'influenza di quest'ultima sulla precisione della IMU è trascurabile, in quanto le due terne di dati non hanno una variazione eccessiva, sia in termini di angolo medio, sia in termini di deviazione standard. Questa prova dimostra quindi che, nell'accumulo di errore, un maggiore impatto e dovuto all'angolo di rotazione

e solo una piccola percentuale è invece dovuto alla velocità con la quale questa avviene.

Di seguito vengono riportati i valori di angolo medio di errore e deviazione standard ricavati da ogni prova.

Tabella 6: Risultati della prova al variare della velocità

		Angolo medio di errore [deg]	Deviazione standard [deg]
Prova lenta	X	0.8304	0.4836
	Y	1.9489	1.5092
	Z	2.2574	1.8150
Prova veloce	X	0.8089	0.7549
	Y	1.7024	1.6078
	Z	2.7926	2.2734

3.3.6 Test sull'orientazione al variare del tempo

In questo test viene valutata la capacità dell'algoritmo di mantenere la precisione all'aumentare del tempo trascorso. Per farlo sono state predisposte tre tipologie di prove in cui viene fatto sempre lo stesso tipo di pianificazione: una funzione periodica in cui un ciclo dura 10 secondi ed è composto da due trapezi di velocità opposti (Figura 34). Il movimento che ne consegue è una rotazione di 90° in senso orario e una rotazione di 90° in senso antiorario. I dati delle varie prove sono stati acquisiti variando il numero di volte in cui questo loop è ripetuto. In particolare, sono stati svolti test con 5, 10 e 20 cicli, documentati dalle Figure 35, 36 e 37.

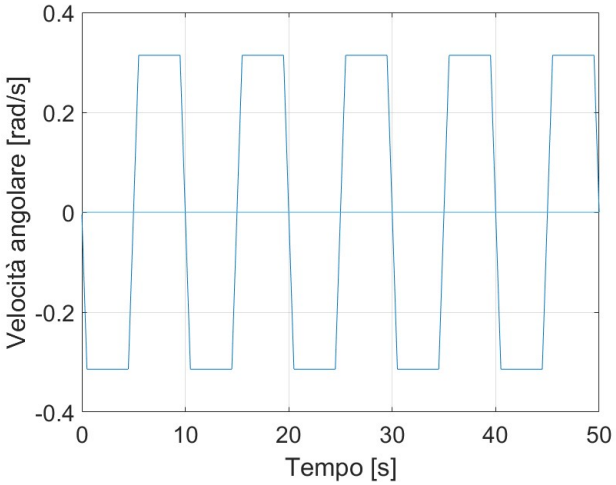


Fig. 34: Andamento della velocità del giunto Base nella prova da 5 cicli

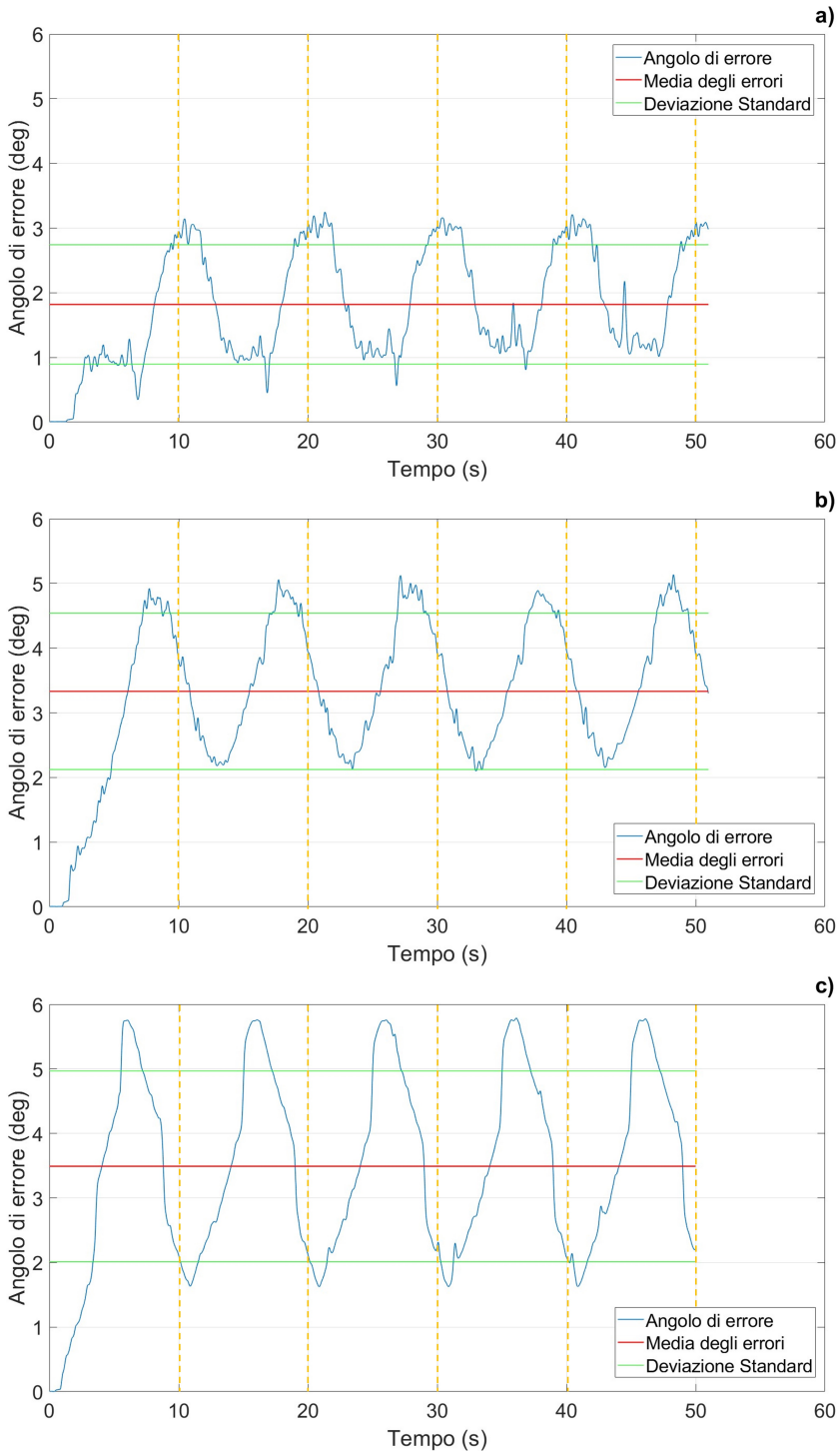


Fig. 35: Errore su 5 cicli, rotazione rispettivamente attorno agli assi X (a), Y (b), Z (c) del sensore

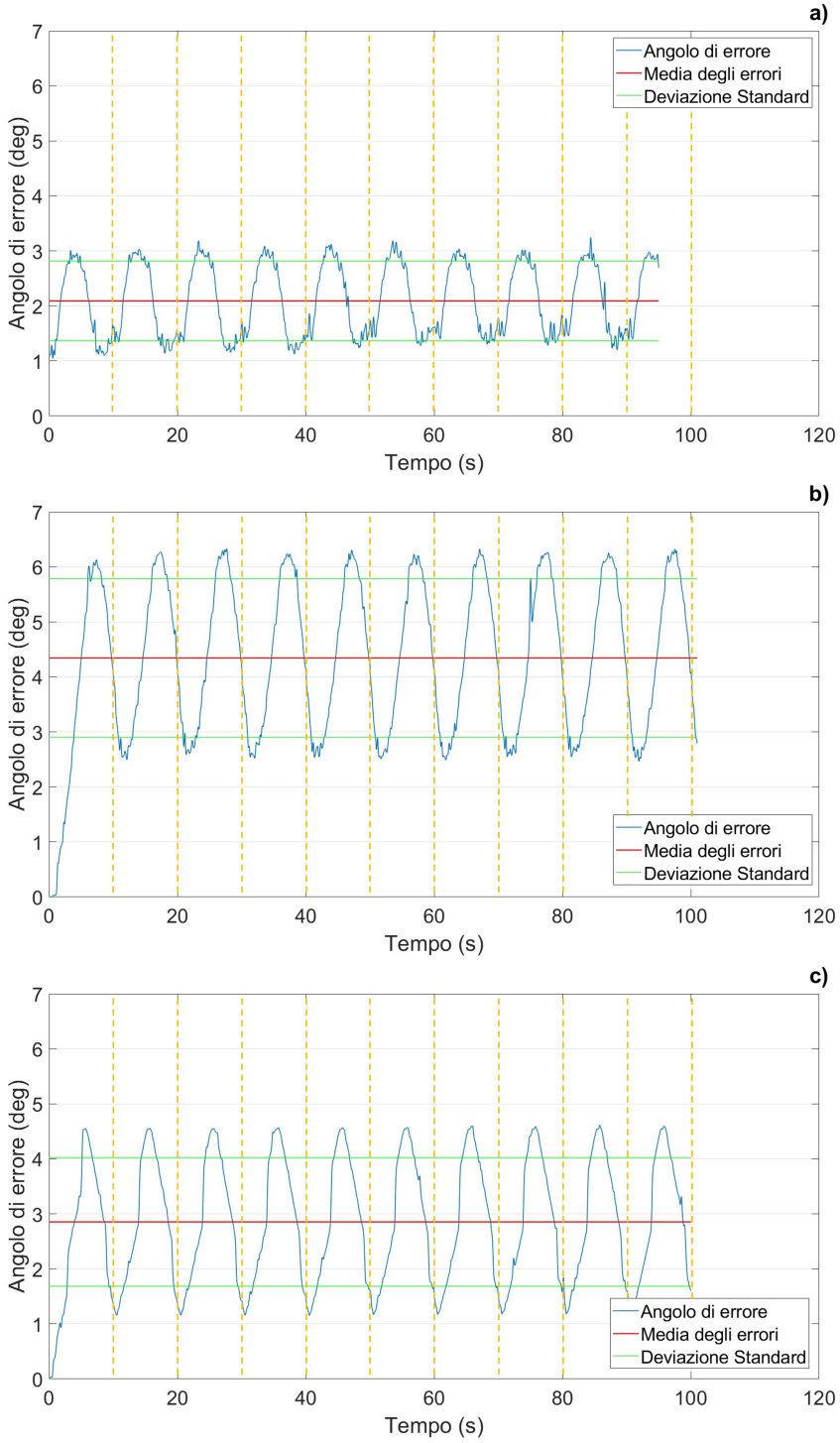


Fig. 36: Errore su 10 cicli, rotazione rispettivamente attorno agli assi X (a), Y (b), Z (c) del sensore

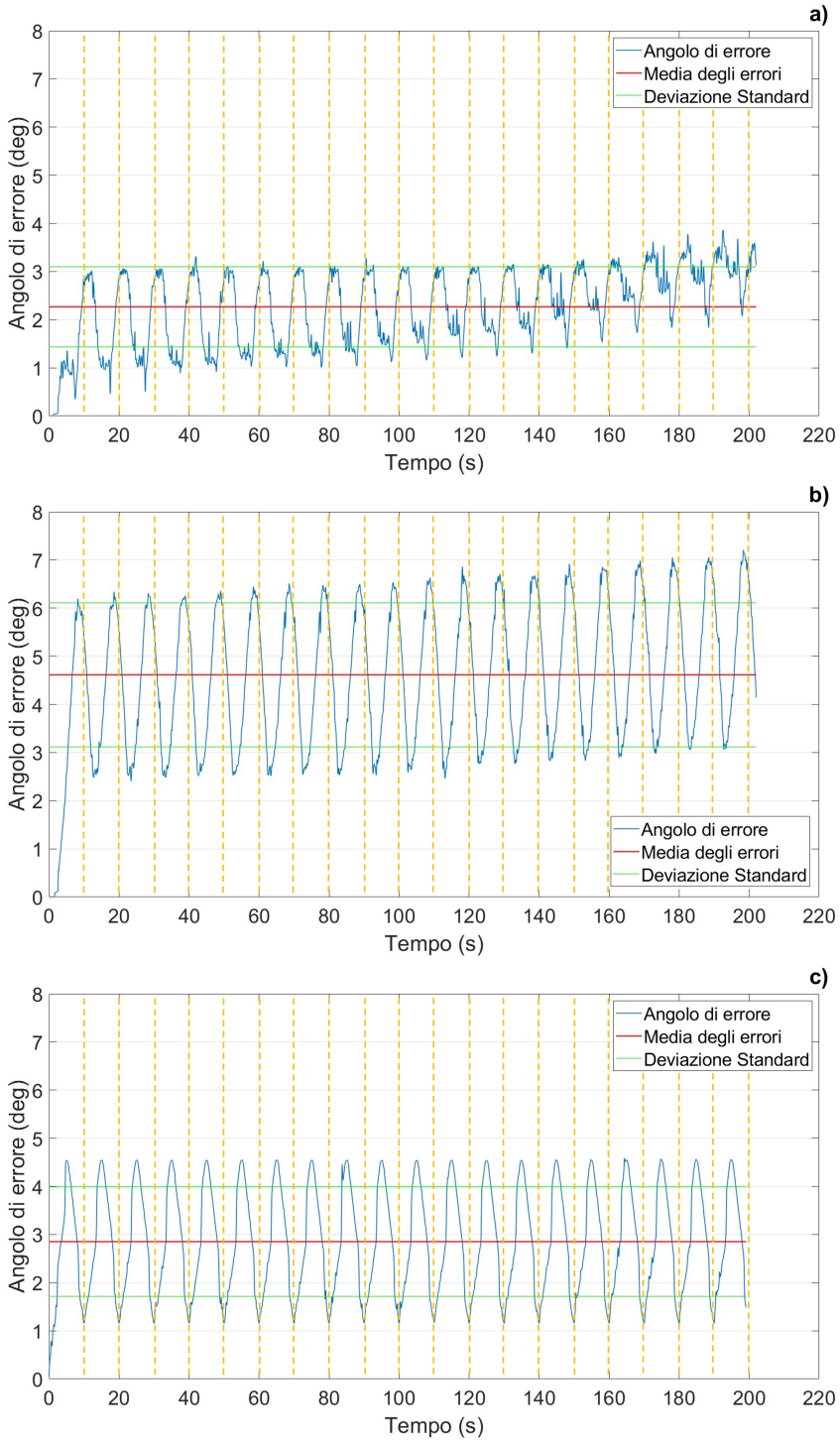


Fig. 37: Errore su 20 cicli, rotazione rispettivamente attorno agli assi X (a), Y (b), Z (c) del sensore

Una prima considerazione può scaturire dall'osservazione della periodicità dell'errore associata alla periodicità del movimento, in altre parole l'errore non solo smette di crescere appena il robot inizia a rallentare, come accadeva nella prova al paragrafo 3.3.4, ma tende anche ad abbassarsi non appena il robot cambia direzione di rotazione, ciò lascia suggerire che in una condizione di movimento libero, in cui non vi è una direzione di movimento preferenziale, l'errore si compensa nelle due direzioni di moto.

In questa prova, inoltre, è interessante notare come i tre assi si comportino in maniera differente in relazione alla prova. Si nota infatti che l'asse Z non risente quasi per nulla del passare del tempo, a differenza degli assi X e Y che perdono precisione man mano che il movimento si prolunga, anche se, nonostante l'accumulo di errore, l'asse X presenta comunque un errore medio inferiore.

Di seguito vengono riportati i valori di angolo medio di errore e deviazione standard ricavati da ogni prova.

Tabella 7: Risultati della prova al variare del tempo di lavoro

		Angolo medio di errore [deg]	Deviazione standard [deg]
5 cicli	X	1.8192	0.9238
	Y	3.3311	1.2089
	Z	3.4911	1.4762
10 cicli	X	1.9237	0.8065
	Y	4.3432	1.4417
	Z	2.8031	1.2114
20 cicli	X	2.2787	0.8197
	Y	4.6375	1.4634
	Z	2.8392	1.1497

Analizzando i dati è possibile avanzare una considerazione, in base alla quale affermare che per utilizzi prolungati sarebbe meglio tenere come asse maggiormente sollecitato l'asse Z, in quanto non perde di precisione, se invece si prevede un utilizzo di questa IMU per applicazioni con tempi limitati, sarebbe preferibile sfruttare maggiormente l'asse X, in quanto globalmente più preciso.

3.3.7 Test sull'orientazione in un percorso con rotazioni intorno ad assi misti

L'ultimo test previsto dal protocollo prevede l'acquisizione prolungata di dati di orientazione da parte della IMU, mentre il robot compie in maniera continuata dei movimenti che prevedono la rotazione non più attorno agli assi propri del dispositivo, bensì attorno ad assi misti che quindi coinvolgono contemporaneamente

più assi propri del sensore, costringendolo a un maggior sforzo computazionale per poter ricavare l'orientazione.

Il percorso studiato per questo test prevede una serie di movimenti del robot alternati a pause in orientazioni ben specifiche che vengono raggiunte più volte durante il percorso, al fine di verificare la perdita di precisione nel tempo, dovuta a eventuali drift dinamici. Il test prevede più varianti dello stesso percorso, dal momento che vi sono due specifiche rotazioni che vengono cambiate da variante a variante, successivamente definite *rotazioni arbitrarie*, indicate da bande rosse nelle Figure 38, 39, 40 e 41. In particolare, queste ultime sono, a seconda della prova, da 45°, 90°, 110°, 180° e 200°. Il fine è osservare il comportamento della IMU sottoposta dinamicamente a rotazioni più o meno ampie, con l'obiettivo di confrontare le condizioni e valutare quelle in cui vengono restituiti risultati migliori. Alle fine del test viene calcolato l'andamento dell'errore nel tempo, con media e deviazione standard.

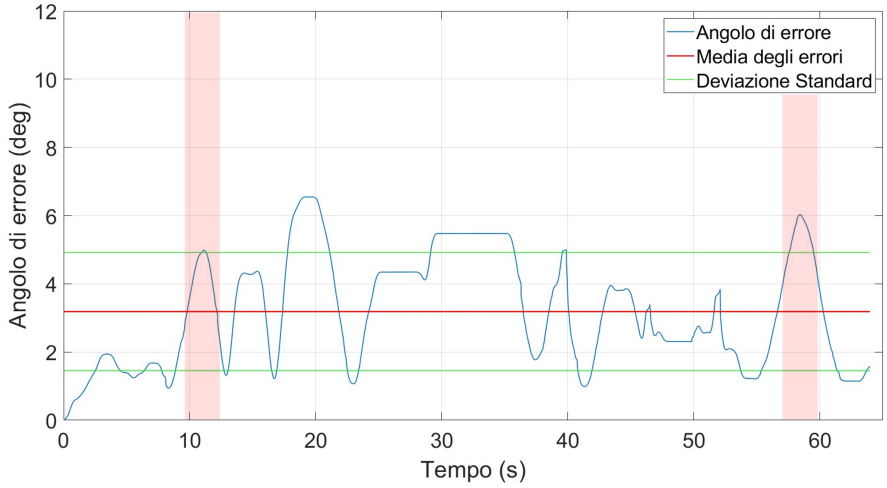


Fig. 38: Andamento dell'errore in test con rotazioni di 45°

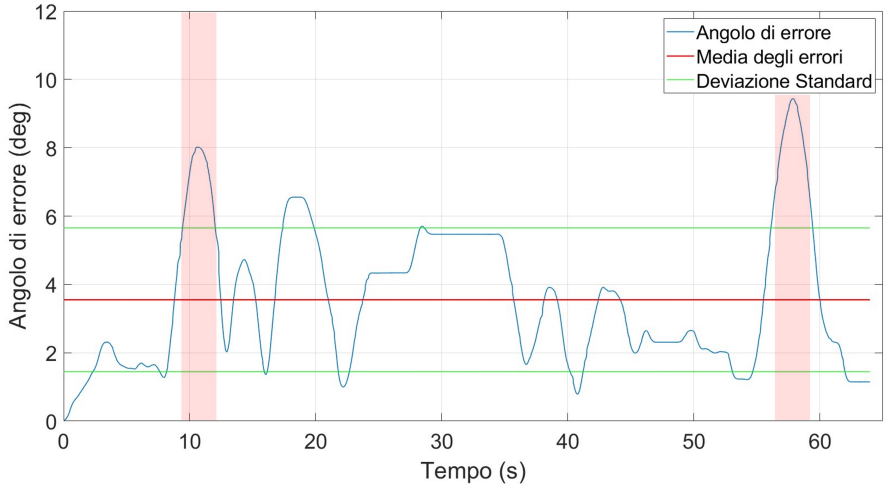


Fig. 39: Andamento dell'errore in test con rotazioni di 90°

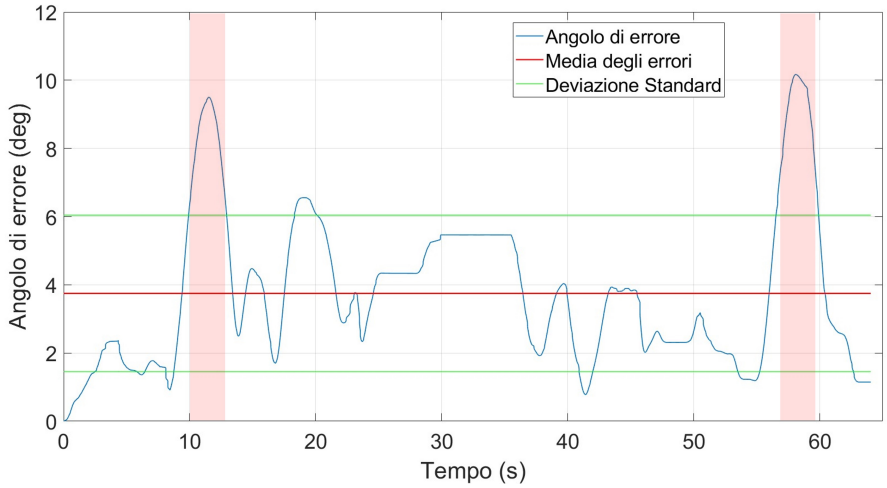


Fig. 40: Andamento dell'errore in test con rotazioni di 180°

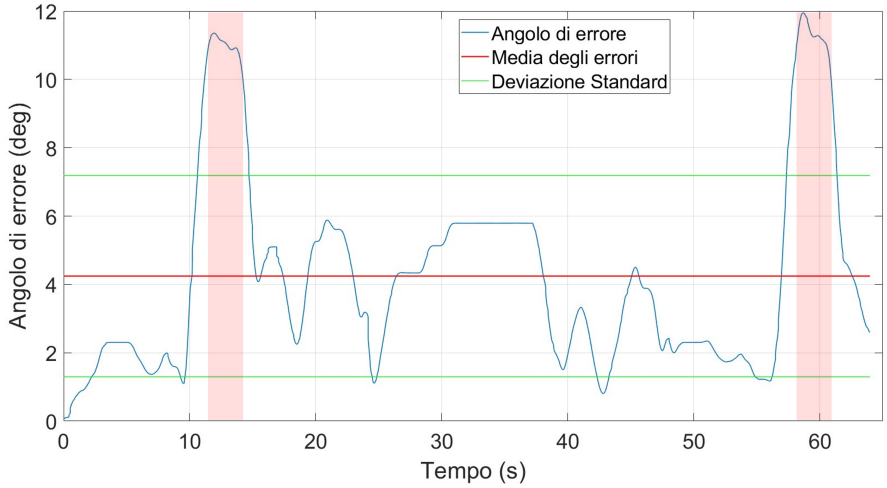


Fig. 41: Andamento dell'errore in test con rotazioni di 200°

Dai grafici si possono notare dei picchi a 13s e 58s circa, in corrispondenza delle rotazioni arbitrarie. Minori sono queste rotazioni e più bassi sono i picchi e viceversa. In generale vengono riconfermati i comportamenti osservati nelle prove su assi propri, si ha ad esempio una precisione media maggiore quando vengono imposte rotazioni di range minore, oppure il fenomeno per cui rotazioni uguali ma in versi opposti tendono a compensare l'errore accumulato. Viene confermata anche la tendenza dei tratti con errore costante in corrispondenza delle fasi in cui il robot è fermo. Tale dato indica che, in condizioni statiche tra un movimento e l'altro, la IMU non è soggetta a drift, anche con schemi di movimento più complessi. Le condizioni dinamiche, infine, uniche responsabili dell'accumulo di errore, inficiano in maniera minima sulle acquisizioni nei momenti successivi, si può notare infatti che nelle quattro prove svolte si ha un errore crescente solo in corrispondenza delle rotazioni variabili, mentre nei tratti in cui i movimenti sono gli stessi si ha un andamento dell'errore sovrapponibile.

Di seguito vengono riportati tutti i risultati delle acquisizioni fatte.

Tabella 8: Risultati della prova di orientazione con assi misti

	Angolo di errore medio [deg]	Deviazione standard [deg]
ROM 45°	3.04	1.80
ROM 90°	3.42	2.17
ROM 110°	3.52	2.36
ROM 180°	3.92	2.82
ROM 200°	3.98	2.99

3.4 Conclusioni

Il protocollo appena descritto, come specificato all'inizio del capitolo, non si prefigge l'obiettivo di fare una valutazione assoluta delle prestazioni dei dispositivi IMU, scopo per il quale esistono metodi più affidabili. Il fine è quello di fornire uno strumento di valutazione delle prestazioni di varie IMU in relazione l'una all'altra e per testare gli algoritmi di sensor fusion, utilizzabile in qualsiasi momento, in caso sia necessario fare un upgrade di hardware o una verifica periodica di corretto funzionamento. In questo caso, in assenza di termini di paragone, la procedura è comunque servita alla definizione dello standard attuale di accuratezza, definito dello strumento usato. La prima attività da compiere in futuro è il confronto con altri modelli per poter creare, nel tempo, una classificazione generale.

Vi sono inoltre vari modi per rendere il protocollo più completo ed efficace, ad esempio potrebbe essere aggiunta una valutazione di IMU su più ampia scala che preveda un maggior numero di ripetizioni delle varie prove, in modo da avere un campione statistico di maggiore rilevanza e dunque poter fare una valutazione più accurata.

Un altro possibile ampliamento del protocollo analizzato prevede l'aggiunta di una variante dell'ultima prova in cui si vanno a valutare le prestazioni della IMU associate a rotazioni attorno ad assi non propri dello strumento per un lungo periodo di tempo, andando così a verificare la precisione di quest'ultima, associata all'algoritmo di Sensor-Fusion, in un contesto in cui più assi sono sollecitati per periodi medio-lunghi.

4 Sviluppo del prototipo di interfaccia e applicazioni

4.1 Strumentazione Software e Hardware

Nel capitolo viene illustrato il funzionamento dei software che sono stati utilizzati per compiere i test successivi, nonché il processo di progettazione e realizzazione dei supporti fisici che compongono il prototipo prototipo di interfaccia, con lo scopo di definire una panoramica generale di tutti gli strumenti utilizzati, a cui far riferimento durante la spiegazione delle prove effettuate.

4.1.1 Robot Operating System - ROS

ROS è definito come *robotic middleware*, ovvero un software open-source che ha alcuni elementi in comune con i sistemi operativi e non può funzionare da solo, in quanto necessita di essere installato su un altro sistema operativo per funzionare. Il software è stato pensato per agevolare il lavoro dei programmatori di robot, con l'obiettivo di mettere in collegamento diversi elementi, chiamati *Nodi*, attraverso uno scambio di messaggi, che avviene attraverso appositi canali chiamati *Topics*. I nodi possono inserire dati all'interno di un topic, in questo caso vengono chiamati *Publisher*. In alternativa, è possibile prendere informazioni da un topic e in tal caso vengono chiamati *Subscriber* [18].

Per la creazione di un nodo occorre costruire un *workspace*, ovvero una cartella che contenga tutti gli elementi necessari al corretto funzionamento del programma. Il processo di costruzione parte dalla creazione di una cartella con il nome desiderato e seppur non obbligatorio, ROS suggerisce comunque di seguire una propria procedura convenzionale. All'interno della cartella sarà presente una sotto-cartella chiamata obbligatoriamente *src*, contrazione del termine "source", nella quale andranno scritti i codici.

Una volta assemblato in tal modo il nostro workspace, ROS mette a disposizione un comando di inizializzazione, da attivare tramite terminale, che varia in base alla versione utilizzata: *catkin_make* in caso di ROS1, *colcon build* in caso di ROS2. Questo comando inserisce automaticamente tutti gli elementi indispensabili per la creazione di uno o più nodi, dal momento che un workspace può infatti contenere al proprio interno più codici che creano nodi.

Dopo aver seguito i passaggi precedenti, è possibile lavorare all'interno del workspace, indicando i pacchetti esterni che saranno condivisi da tutti i nodi, per poi aggiungere i codici di creazione nodo all'interno della cartella src. Completate le procedure, è necessario utilizzare di nuovo il comando di inizializzazione per settare automaticamente tutti i file della cartella.

In questa fase i codici sarebbero già utilizzabili entrando nella cartella src ed eseguendo manualmente il programma, tuttavia, è preferibile attuare un ulteriore step per permettere a ROS di riconoscere il codice come creatore di un nodo, al fine di consentire l'avvio da terminale in qualsiasi momento. Questo passaggio varia in base alla versione di ROS, ma consiste nell'aggiunta del nome del codice, con relativo percorso, all'interno della sezione ROS adibita al *setup*. Nella Figura 42 si può vedere il file da modificare in ROS2 per effettuare questo collegamento. Una volta eseguito questo passaggio sarà sempre possibile eseguire i codici scelti da terminale, utilizzando gli appositi comandi: *rosun* per ROS1 e *ros2 run* per ROS2 [19].

Nel presente lavoro di tesi ROS è stato utilizzato per creare un sistema di comunicazione in grado di collegare il robot UR3, un PC, e il microcontrollore utilizzato per la lettura dei segnali delle IMU.

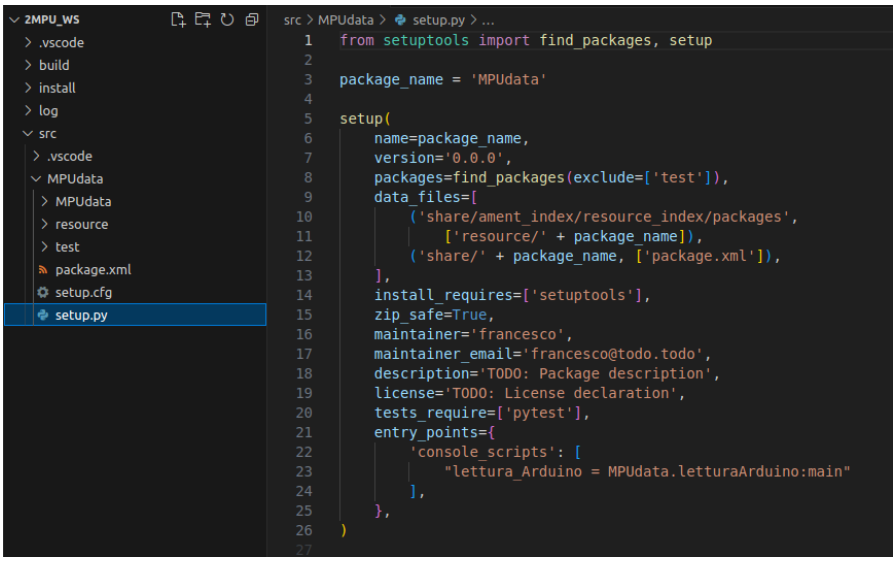


Fig. 42: Setup di collegamento in ROS2

Come richiamato in precedenza, esistono due versioni principali del software: ROS e ROS2, che lavorano sullo stesso principio di base, ma cambiano per alcuni aspetti di sviluppo. ROS, ad esempio, è disponibile solo per ambiente Linux, mentre ROS2 può essere installato anche su sistemi Windows o Macintosh. Ci sono poi altre differenze nell'ottimizzazione, nei di pacchetti disponibili e nel *Build System*, un esempio è il nodo *roscore*, necessario per l'avvio di tutti gli altri nodi ROS, a differenza di ROS2.

Nelle applicazioni del presente lavoro sono stati utilizzati entrambi i sistemi, al fine di ampliare la compatibilità con altri software con i quali sono entrati in comunicazione. Nello specifico, per le attività descritte nel capitolo 3 e nel

capitolo 4.2 è stato usato ROS con la distribuzione *Noetic Ninjemys*, mentre per lo sviluppo del prototipo descritto nel capitolo 4.3 è stato utilizzato ROS2 con la distribuzione *Humble Hawksbill*.

4.1.2 Componenti elettronici

Le componenti elettroniche utilizzate sono, come già anticipato nel paragrafo 3.2, due schede GY-521 con sensore IMU MPU6050 e un Arduino Nano. Nella fase di testing e nella prima applicazione viene usato un solo sensore e il collegamento tra IMU e Arduino avviene secondo lo schema descritto in Figura 43, in cui VCC è il pin adibito all'alimentazione, GND alla messa a terra, mentre SCL e SDA sono i pin attraverso i quali vengono trasferiti rispettivamente le informazioni di tempo e i dati acquisiti. Per il collegamento con il computer, infine, è stato utilizzato un cavo Micro-USB.

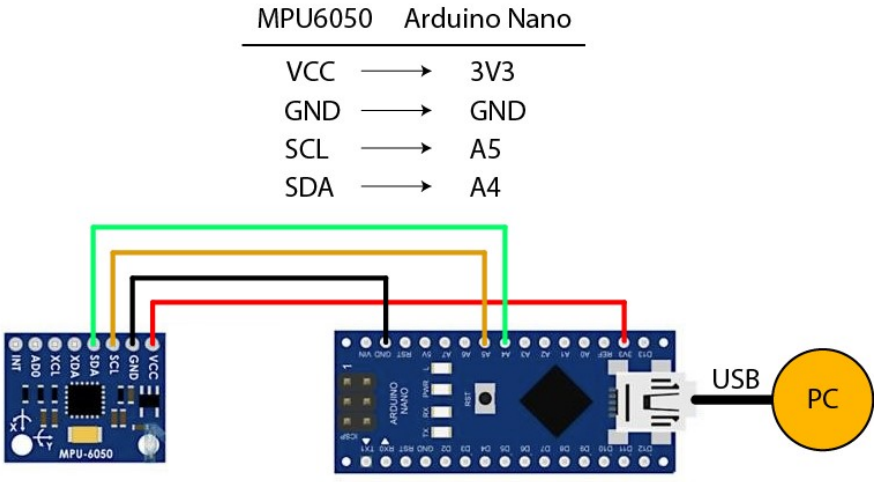


Fig. 43: Schema di collegamento tra MPU6050 e Arduino Nano

Il tipo di collegamento è chiamato I2C, un protocollo di comunicazione che permette di collegare più periferiche utilizzando solo due linee, questo permette di creare architetture più complesse con un minore ingombro.

Il collegamento è stato prima testato con una *breadboard*, come mostrato nella Figura 44, ovvero uno strumento che consente di realizzare collegamenti cablati senza l'ausilio di saldature, con il vantaggio di poter testare i circuiti elettrici prima dell'effettivo assemblaggio finale di un dispositivo. Su questo strumento è stato installato l'Arduino e sono stati effettuati tutti i collegamenti con la IMU, a eccezione dell'ultimo che necessitava di muoversi per la natura stessa della prova.

Con tale composizione è stato effettuato il primo test da banco e parte del secondo, prima di procedere alle verifiche con un hardware più strutturato [20].

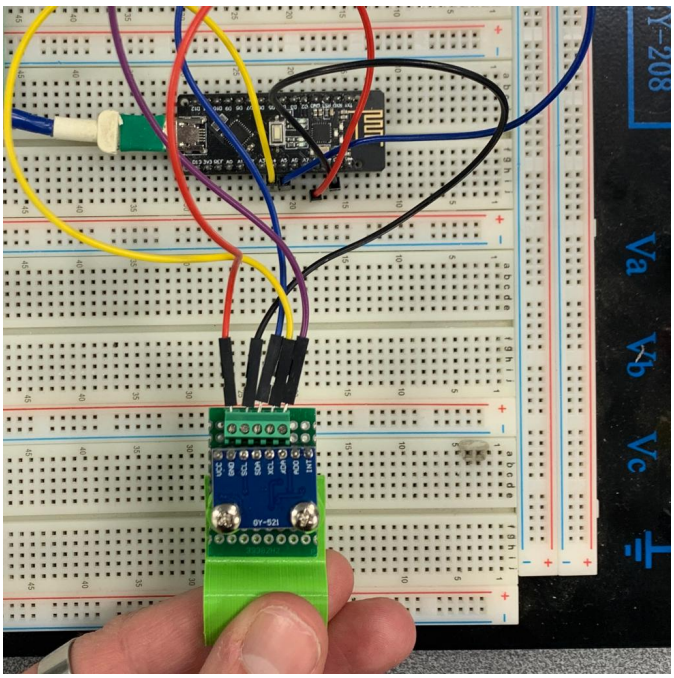


Fig. 44: Collegamento su breadboard

In una fase successiva, durante la realizzazione del supporto per l'applicazione di telecontrollo descritta nel capitolo seguente, i collegamenti sono stati fatti, sulla base dello schema, con saldatura a stagno su una scheda di prototipazione millefori, supporto adibito alla creazione di elementi cablati fissi, tali da poter essere inseriti in un'apposita sede del pezzo stampato in 3D. Come si può notare nella Figura 45, le terminazioni dei collegamenti sono stati realizzate con dei connettori JST che permettono un facile distacco della scheda dal resto della struttura e conferiscono modularità al prototipo.

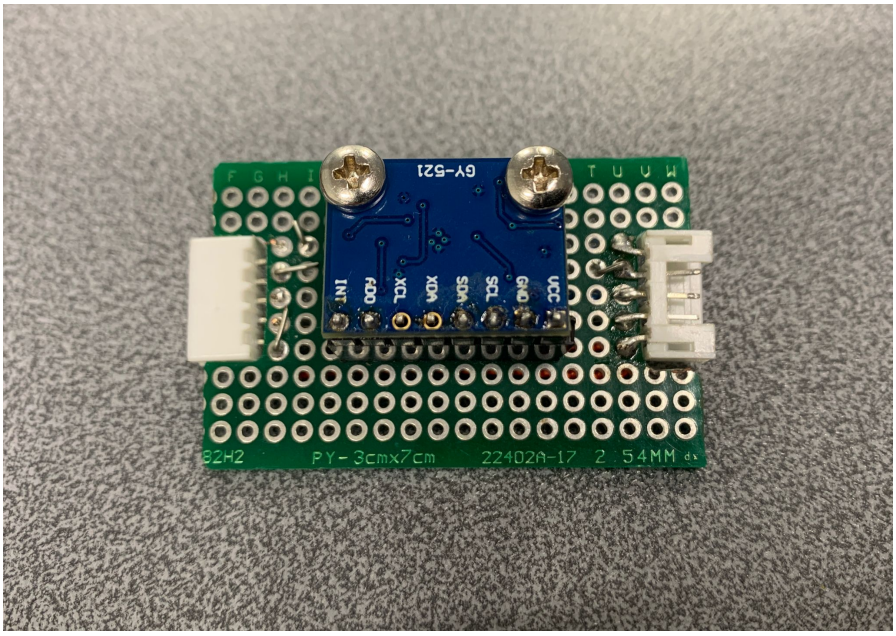


Fig. 45: Millefori prototipale con IMU

Per la seconda applicazione relativa allo sviluppo del prototipo trattato nel capitolo 4.3 è stata creata una variante del collegamento descritto in precedenza, cercando però di associare due sensori in parallelo allo stesso Arduino.I pin da collegare sono gli stessi, sia sulle IMU sia sulla scheda Arduino, la differenza consiste solamente nel collegamento in serie dei pin GRD, SCL e SDA, oltre che nell'utilizzo di AD0 come alimentazione al posto di VCC per una delle due schede IMU, il collegamento è visibile nella Figura 46.

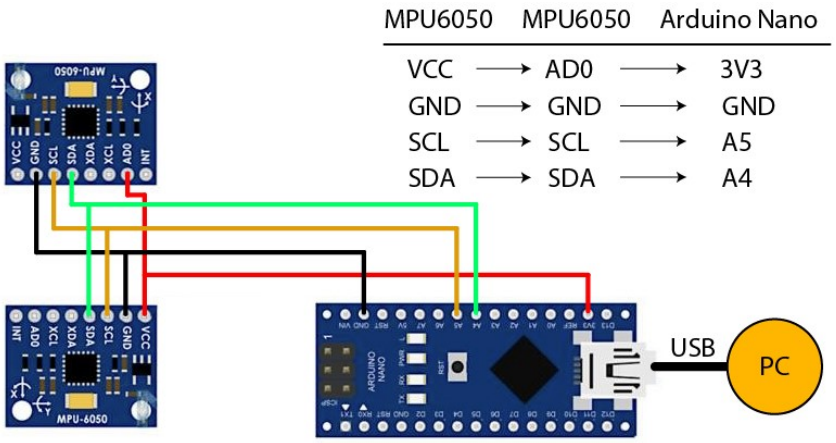
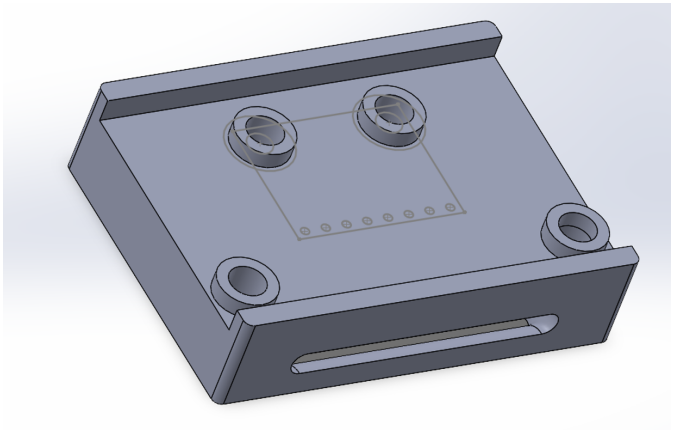


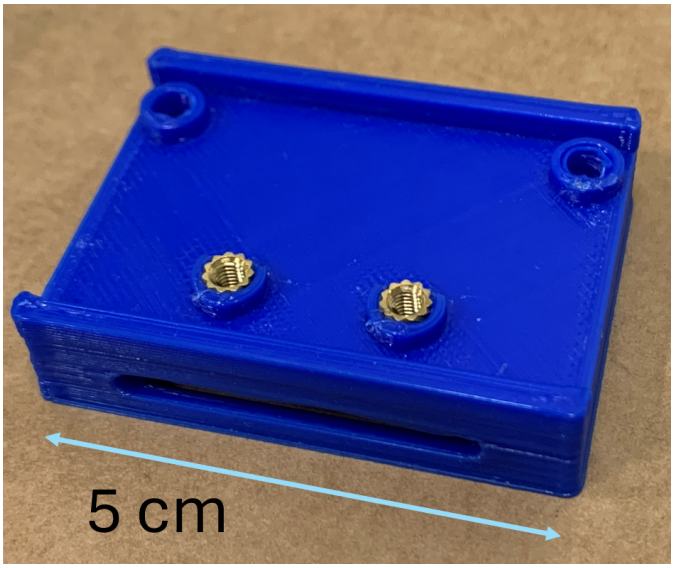
Fig. 46: Schema di collegamento tra due MPU6050 e Arduino Nano

4.1.3 Supporti 3D

Per lo sviluppo dell'architettura del prototipo, è stato fondamentale partire da una fase di progettazione di supporti indossabili che consentissero all'operatore di usufruirne senza penalizzare il comfort e la praticità dei movimenti. La scelta è così ricaduta su due bracciali da stringere attorno al palmo della mano e attorno al braccio, con l'ausilio di un cinturino con chiusura in velcro.



(a)



(b)

Fig. 47: Modello CAD (a) e prima stampa (b)

I supporti, mostrati in Figura 47, sono stati progettati con il software di modellazione CAD Solidworks e successivamente stampati in 3D, hanno la forma di

un parallelepipedo rettangolo di dimensioni 48x35x13mm, con due asole sul lato lungo da 30x4mm per il passaggio del cinturino. Sul lato superiore è inoltre presente una scanalatura, per contenere la scheda millefori, all'interno della quale sono presenti 4 fori con sedi per viti M3, due principali per il fissaggio della IMU e altri due per assicurare ulteriormente l'altro lato della scheda millefori. Si tratta comunque di una scelta di precisione, poiché i due fori principali sarebbero stati comunque in grado di assicurare da soli un buon livello di fissaggio. L'andamento delle due asole, visibile nella Figura 48, è stato concepito per agevolare la procedura di stampa 3D, dal momento che rivolgendosi verso il basso, diminuisce la quantità necessaria di elementi aggiuntivi di supporto alla stampa, i quali devono essere successivamente eliminati, azione che, viste le dimensioni ridotte del pezzo, implica il rischio di rottura di quest'ultimo.

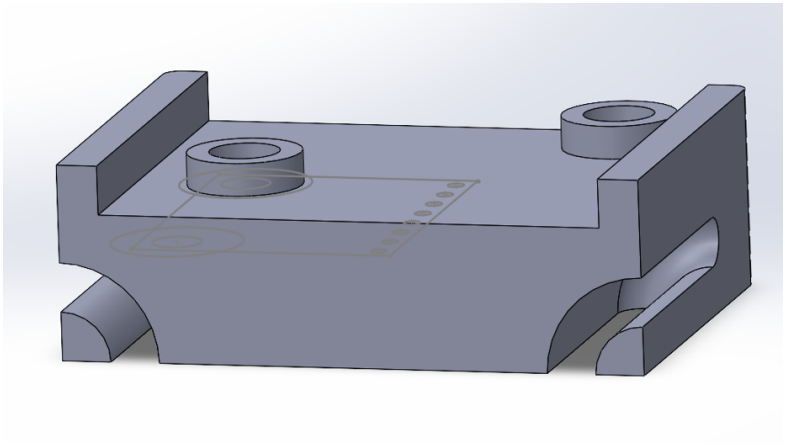


Fig. 48: Andamento interno del foro per l'asola

4.2 Prima versione dell'architettura di controllo a comando remoto

4.2.1 Funzionamento

Il codice di partenza consente il telecontrollo di un robot UR3 per l'esecuzione di un task di pick and place. Gli strumenti di controllo sono: la tastiera, dalla quale si decide la modalita‘ di utilizzo e un sensore inerziale di tipo commerciale, OPAL, che esegue l'effettivo controllo a distanza. Per stabilire un collegamento tra questi strumenti stato utilizzato il software ROS.

Sono disponibili 5 modalità di controllo:

- **Placement:** Inclinando la IMU si fa muovere il TCP del robot con una velocità fissa lungo il piano X-Y.

- **Orientation Control:** L'inclinazione della IMU corrisponde all'inclinazione del TCP.
- **Gripper rotation:** Con l'inclinazione della IMU si ruota il gripper con una velocità fissa attorno all'asse Z del TCP ad una velocità fissa.
- **Approach to object:** Con l'inclinazione della IMU si trasla il gripper con una velocità fissa lungo all'asse Z del TCP.

Sono stati impostati quattro Nodi principali: Opal, kayboard, data_elaboration e robot_control. Questi nodi si comunicano tra loro, come descritto nella Figura 49, attraverso tre topics: IMU_data, control_mode e robot_command.

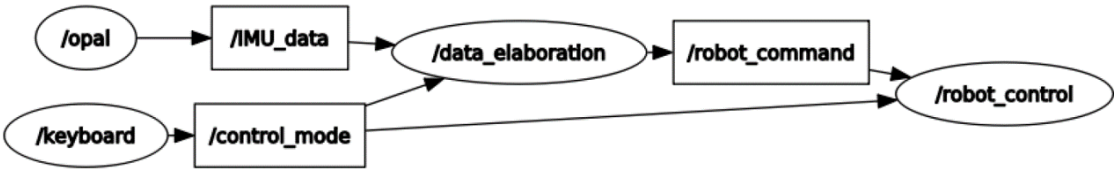


Fig. 49: Grafico con i collegamenti ROS

Vengono in seguito illustrate più nel dettaglio le funzioni dei vari nodi e le loro modalità di comunicazione attraverso i topics:

- **Opal:** stabilisce la connessione con i sensori e inizia lo streaming dei dati, il. suo scopo è ricevere le informazioni della IMU e trasferirle al computer. Questo nodo si comporta esclusivamente da Publisher nei confronti del topic *IMU_data*, nel quale pubblica con una frequenza di 200 Hz. Il messaggio pubblicato è di tipo *imu*, un particolare tipo di dato contenuto nel pacchetto Python per ROS *sensor_msg*, la cui struttura è composta da 10 *float64* associati rispettivamente a: quattro valori per i quaternioni che forniscono l'informazione di orientazione; tre valori per le velocità lineari e tre per le velocità angolari, questi ultimi sei riferiti agli assi propri dello strumento.
- **Keyboard:** implementa un'interfaccia su terminale che consente di passare tra le diverse modalità di controllo tramite input da tastiera. Come il nodo precedente, si comporta unicamente da Publisher, in questo caso nei confronti del topic *control_mode*. Il tipo di dato pubblicato è un *Int8* contenuto nel pacchetto Python per ROS *std_msg*, contenente il numero intero scelto dall'operatore che sta a indicare la modalità di utilizzo.

- **data_elaboration:** è il nodo più importante della sequenza, ha lo scopo di elaborare i dati ricevuti dai nodi precedenti e di trasferirli a quelli successivi in un formato leggibile dal robot. Come prima fase c'è la lettura del messaggio inviato dalla tastiera, che definisce la modalità con la quale verranno convertiti i dati ottenuti dalle IMU. Successivamente viene presa l'informazione di orientazione sotto forma di quaternioni e convertita in *Roll-Pitch-Yaw* che verrà a sua volta tradotta in un vettore di posizione (eq.1) con sei componenti, tre di spostamento lungo gli assi e tre di rotazione.

  $$P = [x\ y\ z\ r_x\ r_y\ r_z] \tag{1}$$

  Questo nodo si comporta da Subscriber nei confronti dei topic *IMU_data* e *control_mode* e da Publisher nei confronti del topic *robot_command* dove pubblica il vettore di posa calcolato. La tipologia di dato utilizzata per la comunicazione è *Float32MultiArray* contenuto nel pacchetto Python per ROS *std_msg*.

- **robot_control:** gestisce le comunicazioni con l'ambiente ROS e include un thread per il controllo del robot. Implementa un'interfaccia RTDE per inviare dati al programma Polyscope installato sul robot UR3. La sua logica operativa prevede la creazione di un'interfaccia RTDE con il robot tramite comunicazione TCP/IP. Il nodo agisce da Subscriber al topic *robot_command* dal quale prende i dati che verranno successivamente inviati al robot [21].

4.2.2 Modifiche hardware e software

L'architettura appena descritta è stata sottoposta al cambiamento del dispositivo IMU e all'introduzione di una nuova interfaccia gestuale per la selezione della modalità. La funzione di queste modifiche è quella di rappresentare un caso studio per la creazione del prototipo successivo.

Da questa seconda versione del software, il cui grafico è visibile nella Figura 50, è stato sviluppato lo scheletro dell'architettura prototipale, oggetto di analisi nel capitolo 4.3.

Le variazioni che sono state prese in considerazione servono a rendere più modificabile l'intera struttura del prototipo, sia dal lato hardware, sia dal lato software. A tale fine sono state effettuate le seguenti operazioni.

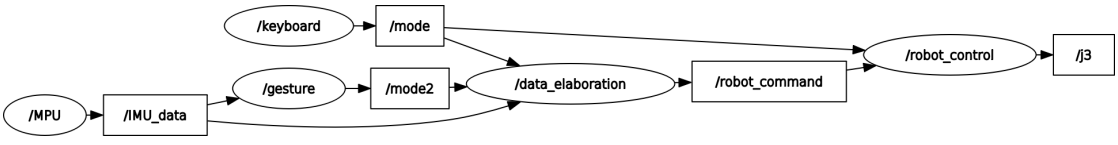


Fig. 50: Grafico con i collegamenti ROS aggiornato

Sostituzione IMU e modularità
Come illustrato in precedenza, si è scelto di utilizzare schede di sviluppo IMU al posto di sistemi embedded, soprattutto per avere maggiori possibilità di aggiornamento hardware in futuro. I sensori Opal sono quindi stati sostituiti con il sistema descritto nel paragrafo 4.1.2, a singola IMU. Dal punto di vista del software questa sostituzione ha implicato la modifica del nodo *opal*, il quale non avrebbe dovuto più trasmettere i dati nella forma precedente. Più nello specifico, sono stati usati due codici diversi:

- il primo nell'Arduino Nano per il riconoscimento dei dati della IMU, questo codice sfrutta l'algoritmo di data-fusion descritto nel paragrafo 3.3.3 per restituire, con una frequenza di 100Hz, l'orientazione dello strumento sotto forma di quaternioni;

- il secondo è un codice Python per creare un nodo ROS denominato *MPU* che stabilisce una connessione con l'Arduino, ne riceve i dati, converte i quaternioni in angoli di Eulero e li pubblica nel topic *IMU_data*. La forma del messaggio rimane invariata rispetto alla versione precedente, per cui non vi è necessità di cambiare la struttura del topic e le dinamiche di collegamento Pubilsher-Subscriber che lo coinvolgono.

Come sottolineato nel paragrafo 4.1.1 la costruzione di un'architettura ROS prevede la creazione di un workspace all'interno di una cartella nella quale ci saranno tutti gli elementi necessari per il funzionamento del software. Per questa applicazione è stato deciso di creare due workspace diversi, come si può vedere nella Figura 51. Un workspace è stato riservato per il nodo relativo alla comunicazione con la IMU e un altro per tutti i restanti nodi di elaborazione dati e controllo del robot. In tal modo, sempre nell'ottica del futuro aggiornamento hardware, si creano compartimenti stagni sostituibili indipendentemente. Se, ad esempio, si ritenesse opportuno cambiare la scheda di sviluppo, lo si potrebbe fare in qualsiasi momento, andando semplicemente a creare una nuova cartella workspace con codici appositamente sviluppati. Con la stessa procedura, sulla base di valutazioni di efficienza, si potrebbe tornare a utilizzare la scheda GY-521 semplicemente avviando la vecchia cartella.

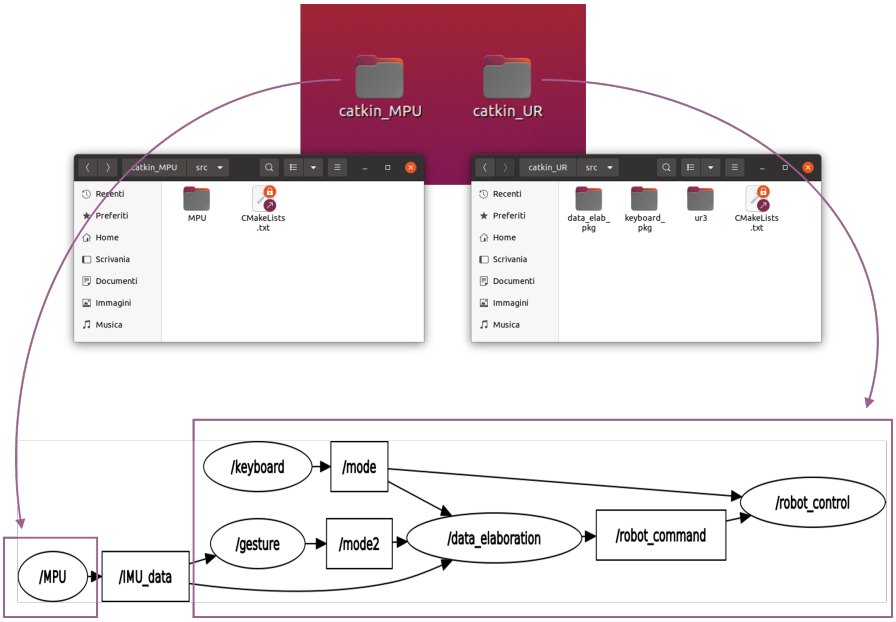


Fig. 51: Divisione dei workspace

Interfaccia gestuale

Nel codice precedentemente analizzato il cambio tra una modalità e l'altra era gestito interamente da tastiera, attraverso il nodo *keyboard* e tale condizione rappresentava certamente un limite, dal momento in cui, in una eventuale applicazione industriale, l'operatore sarebbe stato costretto a rimanere sempre nei pressi del computer per avere la possibilità di variare la modalità di controllo del robot. Per superare tale limite è stato necessario apportare una modifica che consentisse di legare ai comandi gestuali il controllo del robot e al tempo stesso lo switch tra le modalità. Per raggiungere l'obiettivo è stato creato un nuovo nodo chiamato *gesture* ed è stato variato il funzionamento del topic *control_mode*, diviso in due topic diversi: *mode* e *mode2*.

Il nuovo nodo creato riceve i dati dal topic *IMU_data*. Ha il compito di monitorare la rotazione attorno all'asse Y e di pubblicare di continuo sul topic *mode2* un numero intero che funge da flag. Se l'asse Y della IMU rimane inclinato di un valore minore di 45° *gesture* pubblica il numero 0 e *data_elaboration* prende l'informazione sulla modalità da usare dal topic *mode*, controllato dalla tastiera. Se Y si inclina di un valore maggiore di 45° *gesture* pubblica il numero 1 e *data_elaboration* ignora la modalità imposta dalla tastiera e si porta in modalità 2: Orientation Control, dopo aver effettuato il cambiamento il programma torna a dare priorità al topic *mode* per i futuri switch.

L'implementazione di questa modalità non ha un effettivo riscontro pratico sull'esperienza di utilizzo del sistema, ma come detto in precedenza serve come

caso studio per verificare la fattibilità di un modello di comunicazione interamente concepito per il controllo a distanza di un robot, senza l'ausilio di altre interfacce esterne, come una tastiera.

4.2.3 Test da banco e conclusioni

Una volta concluso lo sviluppo del codice è stato condotto un test per verificare il funzionamento effettivo del sistema. Come prima cosa è stato collegato al simulatore di robot di fabbricazione della Universal Robots (URSim), al fine di testare in condizioni di maggiore sicurezza che le velocità impostate fossero effettivamente compatibili con il movimento del cobot, Figura 52.

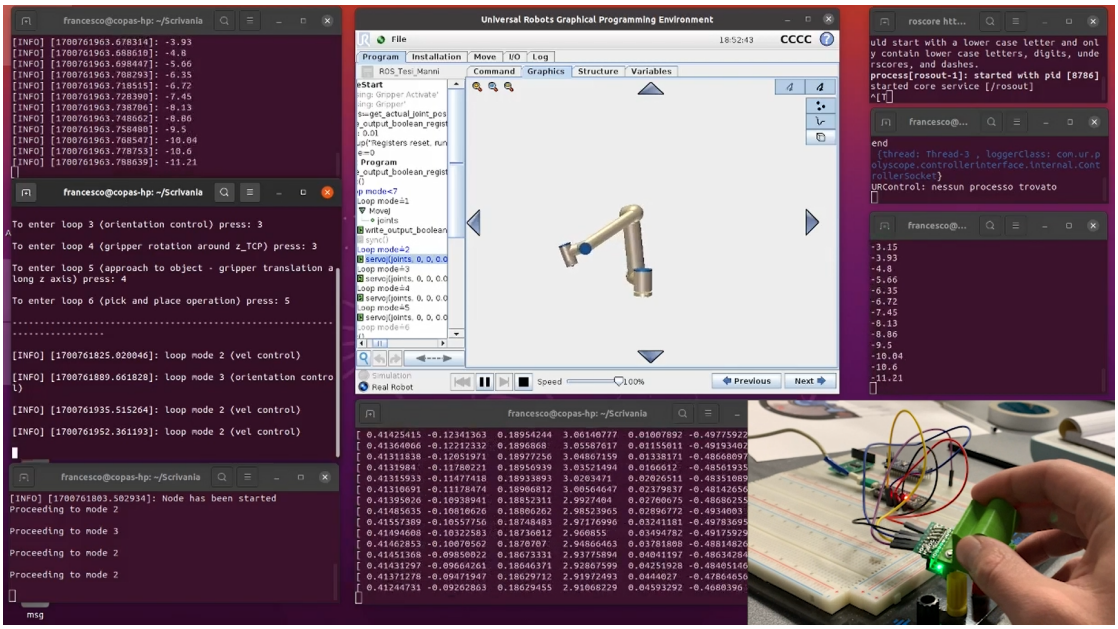


Fig. 52: Screenshot della schermata del PC durante la prova con simulatore

Nello step successivo è stata prevista la prova sul robot reale, che ha confermato il funzionamento corretto di tutta l'architettura, dimostrando fattibile la creazione di un prototipo che possa integrare delle schede di sviluppo e compiere azioni di teleoperazione con il robot tramite un modello di comunicazione gestuale.

Tuttavia, anche questo tipo di comunicazione ha rivelato i propri limiti, dal momento che l'utilizzo di una singola IMU mette a disposizione una quantità di informazioni limitata con la quale controllare il robot.

In sostanza, qualora si dovessero associare movimenti al cambio di modalità, quegli stessi movimenti potrebbero essere compiuti involontariamente durante la fase di controllo e l'operatore potrebbe essere così impossibilitato a far compiere al

robot un'azione il cui comando corrisponde al comando di cambio modalità. Con tale consapevolezza, nella successiva fase di sviluppo di una nuova architettura prototipale è stato previsto l'utilizzo di due sensori, da sistemare su polso e braccio, in modo da incrementare i possibili movimenti percepiti e quindi le potenzialità di utilizzo.

4.3 Seconda versione dell'architettura di controllo

Nel seguente paragrafo è illustrata la progettazione di un dispositivo indossabile su braccio e polso che consente il telecontrollo di un robot collaborativo attraverso i movimenti. L'idea è quella di creare un'interfaccia intuitiva che permetta all'operatore di imparare facilmente a controllare il robot e che allo stesso tempo risulti funzionale dal punto di vista ergonomico. Una schematizzazione del funzionamento di tale prototipo è proposta nella Figura 53. A tale scopo le informazioni della IMU posizionata sul braccio ($IMU_{braccio}$) combinate con quelle della IMU posizionata sul polso (IMU_{polso}) vengono interpretate per far sì che a un determinato movimento del polso siano associate diverse operazioni svolte dal robot a seconda dell'orientazione del braccio.

È previsto anche un tipo di movimento che inibisca il controllo del robot, affinché l'operatore possa utilizzare in qualsiasi momento le braccia per svolgere altre attività, senza la necessità di togliere o spegnere il dispositivo.

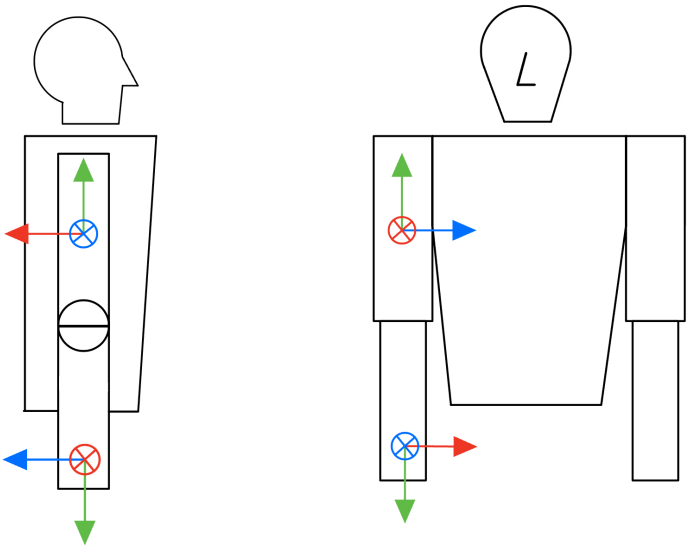


Fig. 53: Schematizzazione del posizionamento dell'architettura prototipale

4.3.1 Logica di funzionamento

La struttura di comunicazione prevede il controllo del robot attraverso l'invio di un vettore contenente 6 informazioni di velocità in spazio cartesiano (eq.2), tre velocità lineari e tre velocità angolari, che verranno poi applicate al TCP.

$$V = [v_x \ v_y \ v_z \ \omega_x \ \omega_y \ \omega_z] \tag{2}$$

Sono previste tre diverse modalità di funzionamento: *Controllo della posizione*, *Controllo dell'orientazione* e *Inibizione*. Nella prima modalità i movimenti dell'operatore vanno a influenzare le velocità lineari del robot, facendo variare solo i primi tre termini del vettore mentre gli altri sono fissati a 0; nella seconda, al contrario, si permette il riorientamento del TCP del robot variando solo gli ultimi tre termini del vettore. Nella modalità Inibizione, infine, vengono fissati a 0 tutti gli elementi del vettore, sospendendo di fatto la comunicazione tra il dispositivo e il robot.

Tutte le modalità sono concepite immaginando un operatore di fronte al robot, perpendicolare al verso positivo dell'asse Y della base del robot, come illustrato nella Figura 54.

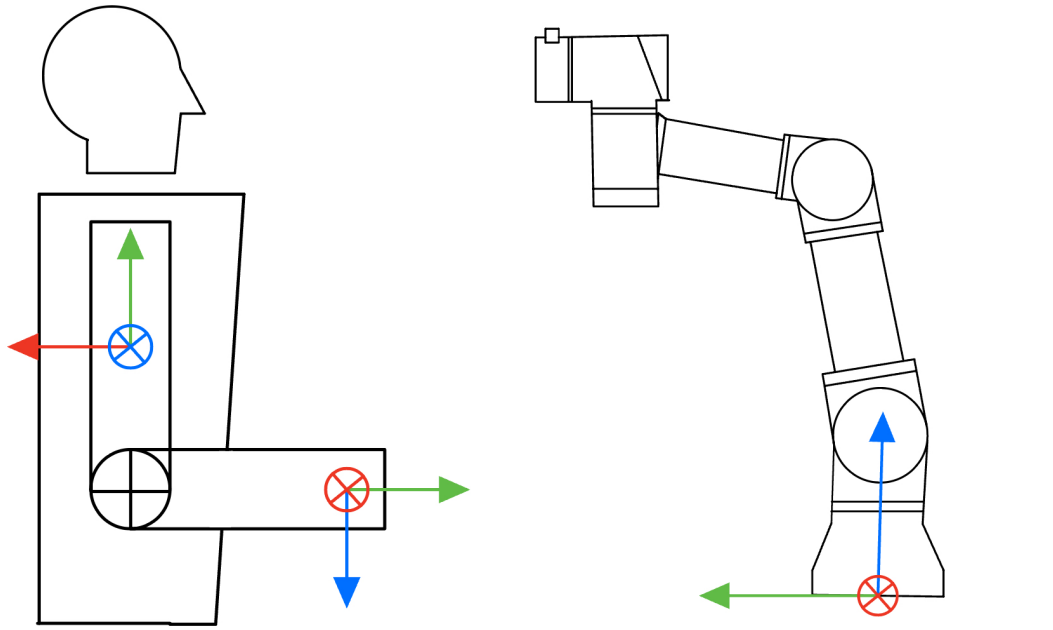


Fig. 54: Rappresentazione schematica del posizionamento dell'operatore rispetto al robot

- **Controllo della posizione**

  Tale modalità, da considerare come principale, prevede come posizione iniziale quella descritta nella Figura 55, ovvero braccio lungo il fianco, gomito ruotato a 90° davanti all'operatore e polso in posizione neutra. La mano risulterà con il pollice rivolto verso l'alto e con tale configurazione non viene impartito al robot nessun comando. Gli spostamenti lungo gli assi vengono controllati in maniera indipendente l'uno dall'altro attraverso i movimenti:

  - **Movimento lungo l'asse X:** si ottiene orientando verso destra o sinistra il polso andando quindi a ruotare la IMU_{polso} attorno all'asse Z.
  - **Movimento lungo l'asse Y:** si ottiene portando in avanti o indietro l'avambraccio, ruotando così attorno all'asse Z della $IMU_{braccio}$
  - **Movimento lungo l'asse Z:** si ottiene con la flessione del gomito che ha l'effetto di far traslare il polso perpendicolarmente al terreno, creando una rotazione attorno all'asse X della IMU_{polso}

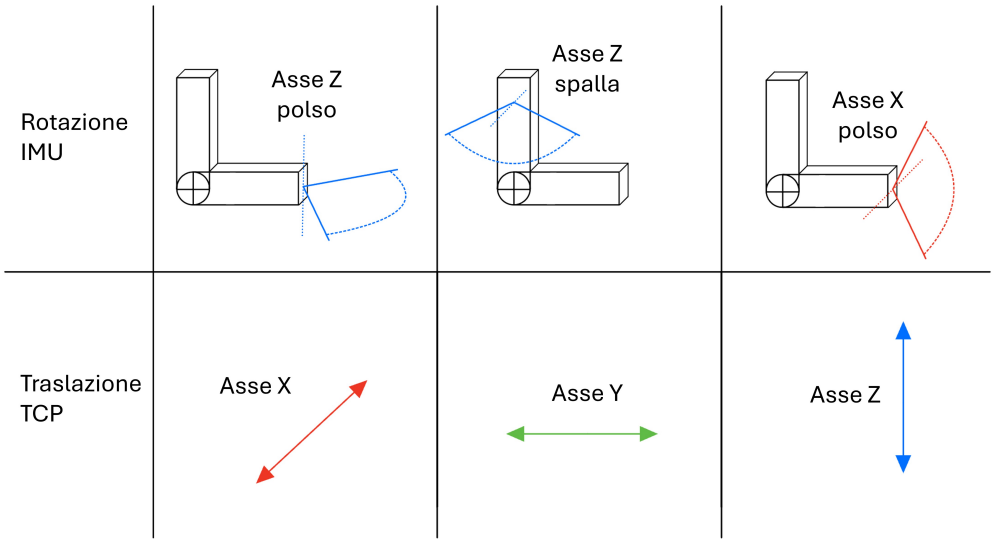


Fig. 55: Utilizzo del dispositivo in modalità Controllo della posizione

Le velocità lineari verranno aumentate o diminuite in modo direttamente proporzionale all'inclinazione dei relativi assi che li controllano. Come si può notare dalla Figura 56, sono state predisposte un'inclinazione massima di ±45° e una minima di ±10° dei tre assi. Se si supera la prima inclinazione la velocità lungo quell'asse viene saturata a un valore prestabilito, se

si inclina l'IMU di un valore inferiore a quello minimo non viene trasmesso nessun valore di velocità. La configurazione è in grado di fornire un range di sicurezza che tenga conto della naturale imprecisione dell'operatore nel tornare alla posizione di partenza.

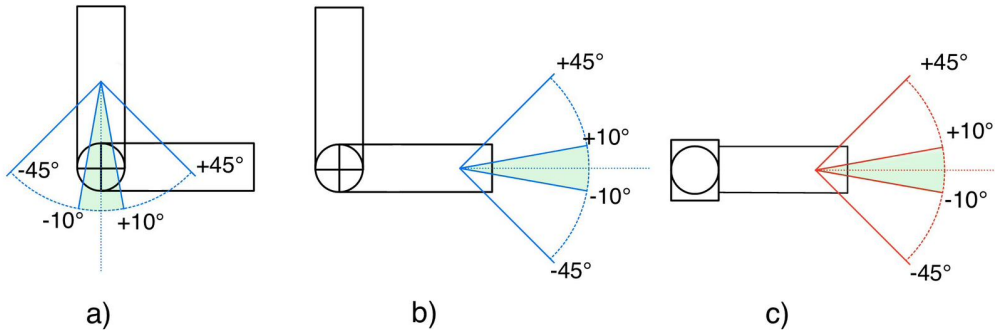


Fig. 56: Range di utilizzo dei tre assi del dispositivo in modalità Controllo della posizione: asse Z spalla (a), asse Z polso (b), asse X polso (c)

- **Controllo dell'orientazione**

  Questa modalità si attiva con un'abduzione della spalla in modo da allontanare il gomito dal fianco, come mostrato in Figura 57, ruotando l'asse X della $IMU_{braccio}$ di un valore uguale o superiore ai 45°.

  Come accade anche nella modalità precedente, c'è un rapporto di proporzionalità tra l'inclinazione dello strumento e la velocità di rotazione del TCP rispetto alla base. La logica di comando è studiata per far sì che, nell'ipotesi della configurazione descritta nella Figura 54, la direzione di rotazione della IMU_{polso} corrisponda alla direzione di rotazione del TCP.

  Anche in questa modalità sono presenti una inclinazione massima di ±60°, oltre la quale la velocità angolare smette di aumentare raggiungendo un plateau; un'inclinazione minima di ±10°, al di sotto della quale la velocità angolare è imposta uguale a 0.

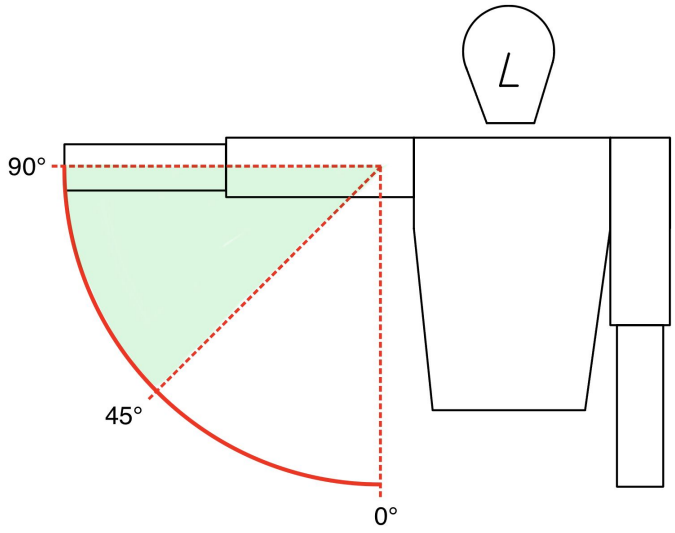


Fig. 57: Range di attivazione (verde) della modalità Controllo dell'orientazione

- **Inibizione**

  Questa è la modalità che permette di interrompere momentaneamente il controllo del robot e dispone del più semplice dei meccanismi di attivazione, basta portare il braccio lungo il fianco e ruotare il polso di 90° facendogli assumere una posizione neutra, portando l'asse X della $IMU_{braccio}$ a una inclinazione inferiore ai 45° e l'asse Y della IMU_{polso} a una inclinazione di 90° o superiore. Così facendo viene trasmesso un vettore velocità in cui tutti i termini sono nulli e il robot smetterà di muoversi, consentendo all'operatore di svolgere altre attività in sicurezza.

  È importante sottolineare come il meccanismo di attivazione di questa modalità può essere soggetto a variazioni, in base al tipo di attività richiesta all'operatore. Nel presente caso, ad esempio, è stata ipotizzata un'attività che preveda una fase di manovra del robot, alternata con una fase di pausa con il polso posizionato in modo neutrale, per cui è stato scelto uno schema motorio adeguato all'attivazione di questa modalità. Se l'attività dell'operatore dovesse essere di altra natura, come manutenzione o cambio pezzo, si dovrebbe pensare a uno schema motorio più adatto alla specifica esigenza.

4.3.2 Implementazione del codice

Per lo sviluppo del prototipo è stata usata una variante del codice Arduino sviluppato e descritto nel capitolo 3.3.3, in combinazione con il software ROS2, de-

scritto nel paragrafo 4.1.1. Più nello specifico sono stati creati tre nodi: *arduino_data_node*, *azzeratore* e *event_servo*. Questi comunicano attraverso l'utilizzo di due topics: *IMU_data* e *Dati_calibrati* secondo lo schema descritto nella Figura 58. Di seguito vengono descritti tutti i passaggi attraverso i quali viene elaborata l'informazione.

Fig. 58: Collegamenti ROS2

1. **Codice Arduino modificato:** i dati vengono acquisiti dalle IMU e letti attraverso un Arduino Nano che ha installato l'algoritmo di data-fusion descritto nel paragrafo 3.3.3 al quale è stata applicata una modifica per permettergli di leggere contemporaneamente e in maniera distinta i dati provenienti dalle due IMU. Per farlo è stata sfruttata una peculiarità del collegamento di tipo I2C, ovvero la possibilità di cambiare l'indirizzo di comunicazione attraverso l'attivazione o meno del pin AD0.

2. **Lettura Arduino:** il PC riceve una stringa di dati dall'Arduino Nano e li interpreta come un vettore di otto valori, ovvero i quattro quaternioni relativi alle due IMU collegate. I quaternioni vengono separati e convertiti in angoli di Eulero, successivamente vengono pubblicati come vettore *1x6* nel topic *IMU_data*. Il tipo di messaggio utilizzato è *Float32MultiArray*, un particolare tipo di dati float contenuti nel pacchetto Python per ROS2 *std_msgs*.

3. **Azzeratore:** è un nodo intermedio che nasce per inizializzare i valori delle IMU all'accensione. Poiché la tipologia di IMU utilizzata tende a perdere l'orientazione dopo lo spegnimento, è frequente dopo un riavvio trovare una orientazione differente rispetto all'acquisizione precedente, essendo cambiata la posizione del sistema di riferimento. Questo nodo è adibito alla correzione di questo possibile errore. Vengono presi i dati dal topic *IMU_data*, viene fatta la media sui primi 300 risultati e i valori ricavati vengono usati come sistema di riferimento di partenza dal quale iniziare a monitorare le inclinazioni. Una volta analizzati i dati precedenti i valori corretti vengono publicati nel topic*Dati_calibrati*, sotto forma di *Float32MultiArray*.

4. **Elaborazione dati:** L'ultimo nodo programmato riceve i dati dal topic *Dati_calibrati* e li elabora secondo la logica descritta nel paragrafo 4.3.1, fino ad ottenere un vettore velocità *1x6*, che viene pubblicato nel topic *event_servo*. Il messaggio pubblicato è di tipo Twist, un particolare tipo di dati float

contenuti nel pacchetto Python per ROS2 *geometry_msgs*, necessario per la corretta comunicazione successiva con il robot.

5. **UR Driver:** è un driver esterno sviluppato per il controllo di robot dalla Universal Robots che permette di acquisire dati pubblicati da *event_servo* e di interpretarli in modo da fornire un comando al robot. Questo driver mette a disposizione degli utenti una serie di tipi di controllo diversi, come il controllo in posizione o la possibilità di fornire al cobot una traiettoria da seguire. Nel caso in esame è stato ritenuto opportuno l'utilizzo del controllo in velocità il quale, grazie al software *MoveIt!*, utilizza i dati del vettore velocità fornito per manovrare il robot. In questa modalità è inoltre permessa la definizione del target da muovere e del sistema di riferimento rispetto al quale farlo, per questa applicazione si è scelto di muovere il TCP del robot rispetto alla base [22].

4.3.3 Test da banco e conclusioni

una volta configurata tutta l'architettura sono stati predisposti due test, attraverso i quali valutare l'effettivo funzionamento complessivo del prototipo finale. Il primo test è stato fatto sul software di simulazione *RViz*, programma per la visualizzazione di robot compatibile con ROS, la cui interfaccia è visibile nella Figura 59. L'obiettivo era quello di testare l'effettivo funzionamento di tutte le parti del codice, nonché avere un ambiente sicuro per verificare, ed eventualmente correggere, alcuni parametri preimpostati, come angoli minimi e massimi acquisibili dai sensori e velocità massima raggiungibile.

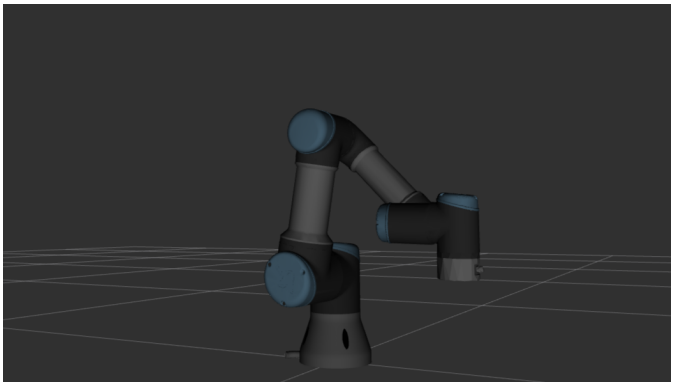


Fig. 59: Interfaccia RViz per il controllo del robot

Il secondo test è avvenuto utilizzando il robot reale, sono state sottoposte a verifica tutte le modalità implementate nel codice e illustrate nel paragrafo 4.3.1,

cercando di valutare la sensibilità del robot, la sua effettiva manovrabilità e facilità d'uso. Di seguito la configurazione finale del prototipo con il quale è stato effettuato il test (Figura 60).

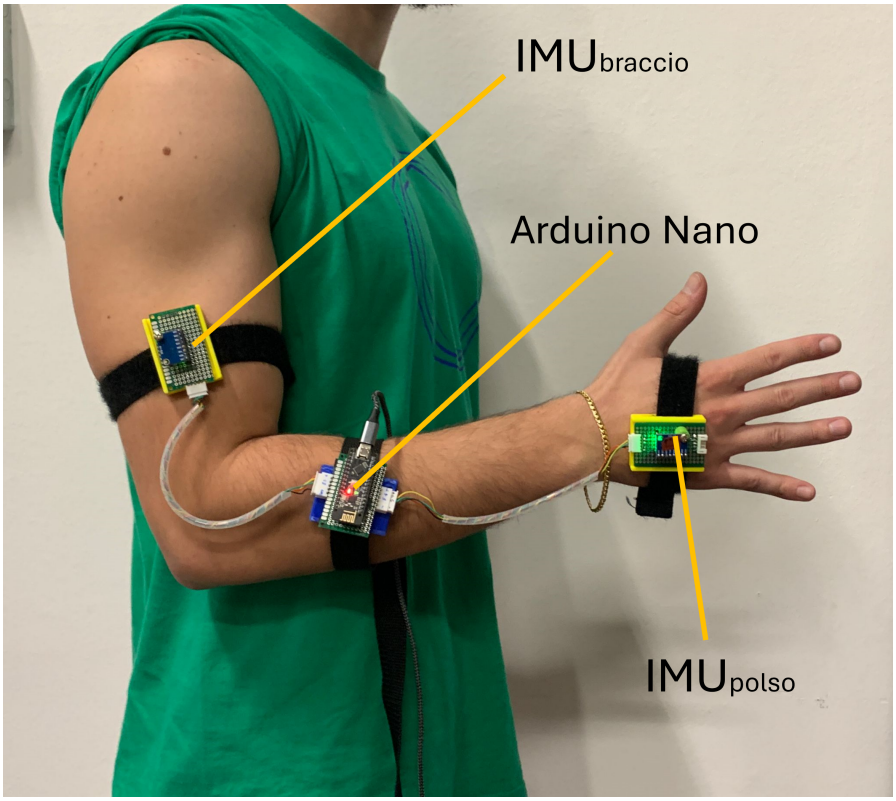


Fig. 60: Configurazione del prototipo in fase di test

Nello specifico sono state testate singolarmente le modalità di Controllo della posizione e di Controllo dell'orientazione, attraverso l'esecuzione di semplici movimenti che implicano il cambiamento di un solo valore del vettore velocità descritto in eq.2. In entrambe le prove sono impostati dati valori di inclinazione variabili per poter verificare l'effettivo rapporto di proporzionalità tra questi ultimi e la velocità di spostamento o rotazione. Di seguito sono visibili due frame del filmato girato durante l'esecuzione dei test (Figura 61).

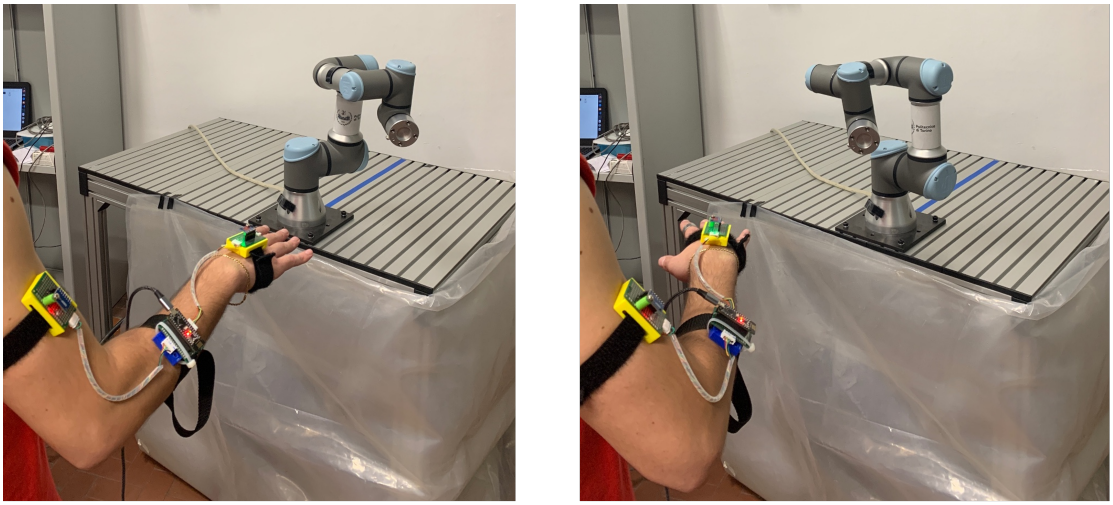


Fig. 61: Spostamenti del sull'asse X del robot in entrambe le direzioni, con relative inclinazioni di IMU_{polso}

Durante le prove sono stati acquisiti sia i dati di inclinazione delle IMU sia i dati di velocità del TCP registrati dal robot, in modo da avere un riscontro numerico riguardo al comportamento effettivo del prototipo.

Nella seguente coppia di grafici è possibile vedere l'andamento delle inclinazioni delle IMU (Figura 62.a) confrontato con l'andamento delle velocità lineare del TCP (Figura 62.b). Gli intervalli sono stati separati in tre fasce diverse in base al movimento eseguito dal robot: azzurro per le traslazioni sull'asse X, giallo per le traslazioni sull'asse Y e rosso per quelle sull'asse Z. Le acquisizioni delle velocità del TCP, seppur affette da rumore, descrivono andamenti sovrapponibili a quelli delle acquisizioni IMU.

Dall'analisi dei grafici emerge, inoltre, la corretta imposizione del range di non utilizzo, presente in tutti i movimenti e descritta nel paragrafo 4.3.1. Le velocità lineari, infatti, assumono un valore significativo soltanto dopo che la relativa inclinazione supera $\pm 10°$, come graficamente descritto anche nella Figura 63, dove l'intervallo di funzionamento viene evidenziato da bande gialle.

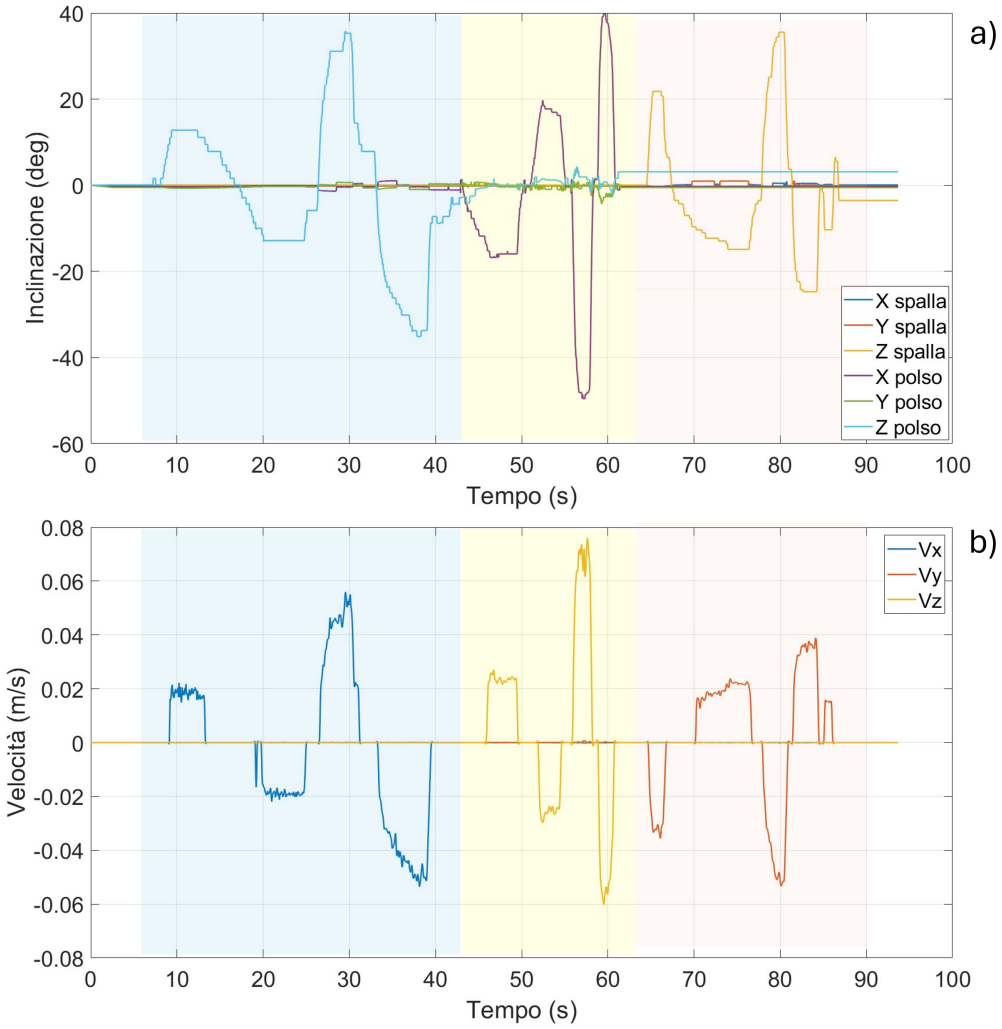


Fig. 62: Acquisizione dati del test da banco sul Controllo della posizione

Di seguito viene riportato il confronto tra i due grafici precedenti in un intervallo ridotto, ovvero solo quello relativo alla fascia azzurra, nella quale ci sono due inclinazioni di circa $\pm 14°$ della IMU_{polso} attorno al suo asse Z, che corrispondo a uno spostamento prima positivo e poi negativo del TCP lungo l'asse X del robot. Quest'azione viene ripetuta inclinando di nuovo il sensore di $\pm 35°$ circa e ottenendo di conseguenza uno spostamento con una velocità superiore.

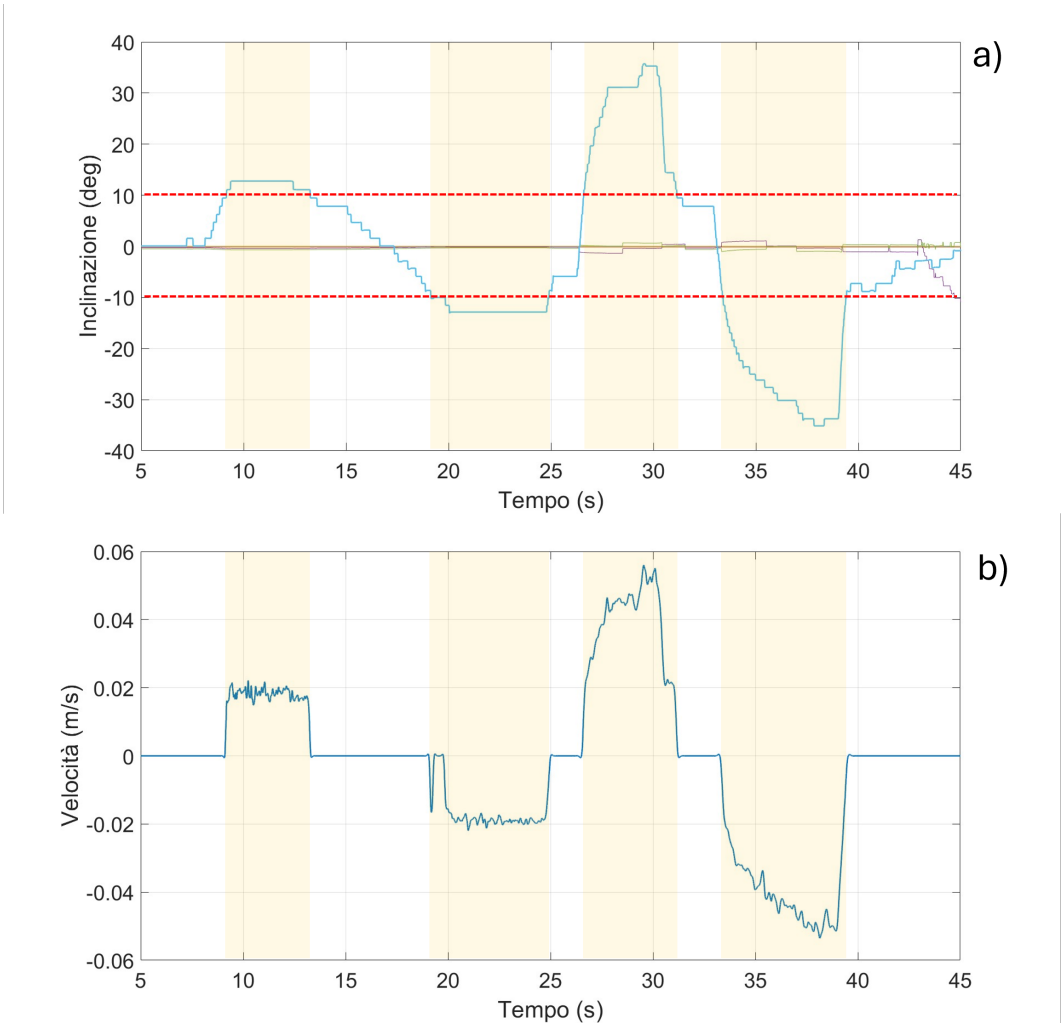


Fig. 63: Acquisizione dati del movimento di traslazione lungo l'asse X della base del robot

Successivamente questa tipologia di acquisizione è stato fatto anche con il Controllo dell'orientazione, con due intervalli di movimento osservabili, divisi da bande colorate (Figura 64): azzurro per le rotazioni attorno all'asse X del TCP e rosso per la inclinazioni attorno all'asse Y del TCP. Come per la prova precedente i risultati hanno mostrato degli andamenti della velocità di rotazione del TCP sovrapponibili con le inclinazioni imposte alla IMU_{polso}.

Per evitare che venisse influenzata da involontari cambi di modalità, inoltre, la prova è stata eseguita con il braccio appoggiato su di un supporto, garantendo la completa abduzione della spalla e il valore dell'inclinazione dell'asse X della $IMU_{braccio}$ fisso a 90°.

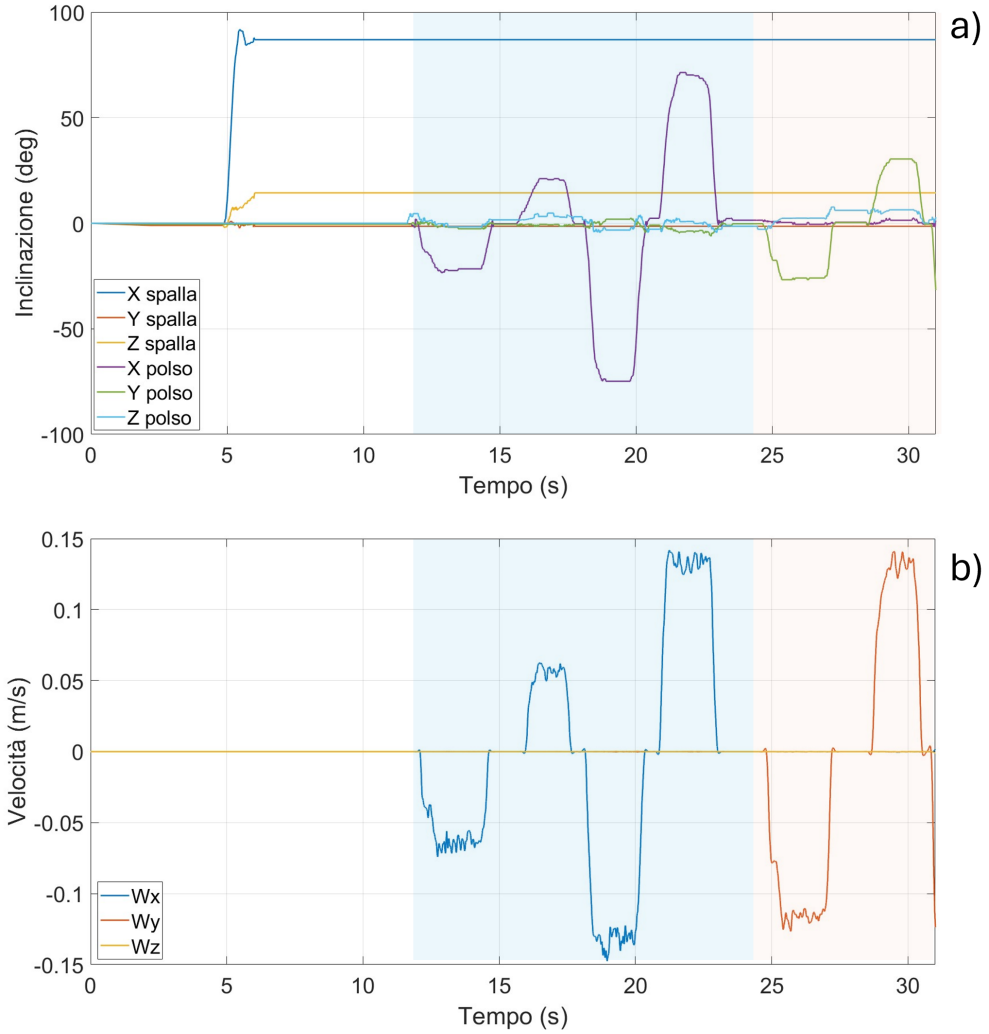


Fig. 64: Acquisizione dati del test da banco sul Controllo dell'orientazione

Anche per questo tipo di controllo è stato confermato il corretto funzionamento del range di non funzionamento imposto a $\pm 10°$, osservabile con maggior dettaglio in Figura 65, dove l'intervallo di funzionamento è stato evidenziato da bande azzurre.

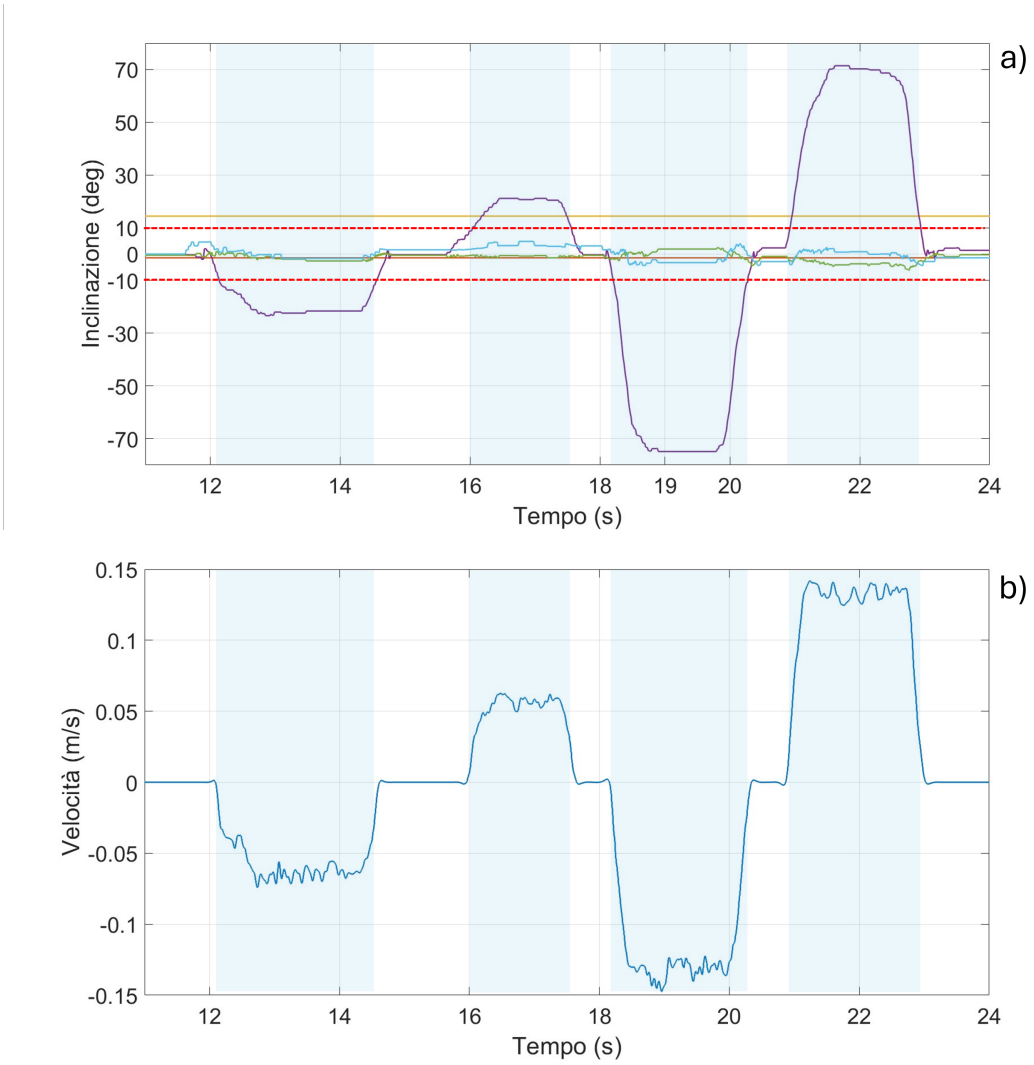


Fig. 65: Acquisizione dati del movimento di rotazione attorno all'asse X del TCP

In tutti i test il prototipo ha restituito i risultati attesi in fase di programmazione, poiché la manovra del robot risulta semplice e intuitiva, per quanto l'impossibilità di poter controllare contemporaneamente sia la posizione che l'orientazione del TCP rappresenta certamente una limitazione. Al fine di avere una maggiore possibilità di controllo, un possibile sviluppo futuro potrebbe riguardare l'implementazione di una terza IMU, possibilmente in combinazione con l'uso di sensori bluetooth con microcontrollori integrati. Tale soluzione permetterebbe l'aggiunta di nuovi dispositivi in maniera più versatile e andrebbe a eliminare limitanti cablazioni, rendendo ancora più pratico e confortevole l'utilizzo di questi strumenti.

5 Conclusioni

La sfida, lanciata da Industria 5.0, di creare un ambiente di lavoro più sicuro e confortevole, attraverso il miglioramento della sinergia tra uomo e macchina rappresenta un obiettivo estremamente complicato da raggiungere.

Nell'ambito della robotica collaborativa questo si traduce nel riuscire a mappare e interpretare la complessità del movimento dell'uomo, per fare in modo che l'ambiente attorno a lui si adatti per rispecchiare completamente le sue esigenze. Questo traguardo non solo è ambizioso, ma è anche estremamente utile per lo sviluppo della società del futuro, aprendo nuove possibilità per migliorare la qualità della vita lavorativa e aumentare l'efficienza e la produttività industriale.

In questo lavoro di tesi si è cercato di aggiungere un piccolo, ma significativo tassello al percorso di questa ricerca innovativa. È stato dimostrato come l'utilizzo di sensori inerziali per il controllo dei robot collaborativi possa rappresentare una soluzione valida ed efficace per superare i limiti delle attuali modalità di interfacciamento tra uomo e robot. L'adozione di tali tecnologie può migliorare significativamente la comunicazione con la macchina, aprendo la strada a nuove applicazioni e opportunità nell'ambito industriale. I sensori inerziali, grazie alla loro capacità di rilevare e misurare in maniera affidabile movimenti e orientazioni, offrono una soluzione robusta e flessibile per il controllo dei robot collaborativi.

La definizione dei vari tipi di sensori inerziali, con una successiva selezione di alcuni di essi, ha rappresentato un punto di partenza fondamentale nello studio e nella classificazione di questi strumenti. Questo processo ha permesso di identificare i sensori più adatti per le specifiche applicazioni di robotica collaborativa, tenendo conto di fattori come la precisione, la velocità di risposta e la robustezza. L'adozione di un protocollo di test ideato appositamente per il confronto tra i vari sensori ha ulteriormente rafforzato l'efficacia dello studio, permettendo di valutare le prestazioni di ciascun sensore in varie condizioni operative. Questa metodologia di valutazione del sistema può conferire al prototipo creato un notevole margine di miglioramento e ottimizzazione, ponendolo come base di partenza per lo sviluppo di dispositivi futuri.

I risultati sperimentali ottenuti hanno dimostrato in primo luogo l'efficacia del telecontrollo, evidenziando la sua capacità di soddisfare i requisiti necessari richiesti per un'interfaccia uomo-robot efficace, come l'esecuzione dei comandi in tempo reale e la facilità di utilizzo. Secondariamente si è visto come l'utilizzo di più sensori distribuiti su diverse parti del corpo permetta di leggere uno schema motorio più articolato e di conseguenza permetta alla comunicazione di diventare più intuitiva e naturale, riducendo significativamente la difficoltà di apprendimento del meccanismo di manovra. Tramite il controllo in velocità basato sull'orientazione delle IMU indossate, usando quindi intuitivamente il braccio come fosse un joy-

stick, è possibile ridurre lo stress cognitivo dell'operatore durante l'utilizzo e, di conseguenza, si migliora il rendimento generale del sistema uomo-macchina.

Visti i promettenti risultati ottenuti, è auspicabile continuare nello sviluppo, ci sono infatti ancora ampi margini di miglioramento e le future ricerche da poter svolgere. Il perfezionamento dei dispositivi indossabili in termini di comunicazione, efficienza e ingombro, potrebbero concentrarsi proprio sull'ottimizzazione degli algoritmi di riconoscimento dei gesti, per rendere ancora più accurata e affidabile l'interazione tra l'operatore e il robot collaborativo.

Un altro importante filone di ricerca potrebbe riguardare l'integrazione con altre tecnologie emergenti, come i sistemi di visione e il Machine Learning, che potrebbero ulteriormente potenziare le capacità dei robot collaborativi e migliorare la loro adattabilità alle esigenze specifiche degli operatori nei diversi ambiti d'intervento. L'integrazione di un prototipo simile in un sistema di visione, ad esempio, potrebbe fornire a quest'ultimo dei dati in tempo reale complementari a quelli già acquisiti, andando a migliorare la sicurezza e l'efficacia delle operazioni svolte. Allo stesso modo, l'utilizzo di algoritmi di Machine Learning potrebbe consentire al robot di apprendere dai movimenti e dai comportamenti dell'operatore, adattandosi in maniera dinamica e continua alle sue esigenze e preferenze.

Bibliografia

[1] Alessandro Gasparetto, Lorenzo Scalera et al. "A brief history of industrial robotics in the 20th century". In: *Advances in Historical Studies* 8 (2019), pp. 24–35.

[2] Saeid Nahavandi. "Industry 5.0—A human-centric solution". In: *Sustainability* 11.16 (2019), p. 4371.

[3] Iina Aaltonen, Timo Salmi e Ilari Marstio. "Refining levels of collaboration to support the design and evaluation of human-robot interaction in the manufacturing industry". In: *Procedia CIRP* 72 (2018). 51st CIRP Conference on Manufacturing Systems, pp. 93–98. ISSN: 2212-8271. DOI: https://doi.org/10.1016/j.procir.2018.03.214. URL: https://www.sciencedirect.com/science/article/pii/S2212827118303743.

[4] Leonardo Sabatino Scimmi et al. "Experimental real-time setup for vision driven hand-over with a collaborative robot". In: *2019 International Conference on Control, Automation and Diagnosis (ICCAD)*. IEEE. 2019, pp. 1–5.

[5] F. Sherwani, Muhammad Mujtaba Asad e B.S.K.K. Ibrahim. "Collaborative Robots and Industrial Revolution 4.0 (IR 4.0)". In: *2020 International Conference on Emerging Trends in Smart Technologies (ICETST)*. 2020, pp. 1–5. DOI: 10.1109/ICETST49965.2020.9080724.

[6] Federico Vicentini. "Collaborative robotics: a survey". In: *Journal of Mechanical Design* 143.4 (2021), p. 040802.

[7] Bin Fang et al. "3D human gesture capturing and recognition by the IMMU-based data glove". In: *Neurocomputing* 277 (2018). Hierarchical Extreme Learning Machines, pp. 198–207. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2017.02.101. URL: https://www.sciencedirect.com/science/article/pii/S0925231217314054.

[8] Yuanyuan Wang et al. "Measurement Quality Control Aided Multisensor System for Improved Vehicle Navigation in Urban Areas". In: *IEEE transactions on industrial electronics* 71.6 (2024-6), pp. 6407–6417. ISSN: 0278-0046.

[9] Marco Iosa et al. "Wearable inertial sensors for human movement analysis". In: *Expert review of medical devices* 13.7 (2016), pp. 641–659.

[10] Kimi D Dahl et al. "Wearable sensor validation of sports-related movements for the lower extremity and trunk". In: *Medical Engineering & Physics* 84 (2020), pp. 144–150.

[11] Norhafizan Ahmad et al. “Reviews on various inertial measurement unit (IMU) sensor applications”. In: *International Journal of Signal Processing Systems* 1.2 (2013), pp. 256–262.

[12] Emil Škultéty, Elena Pivarčiová e Ladislav Karrach. “Design of an Inertial Measuring Unit for Control of Robotic Devices”. In: *Materials Science Forum*. Vol. 952. Trans Tech Publ. 2019, pp. 313–322.

[13] Gizem Ateş e Erik Kyrkjebø. “Human-robot cooperative lifting using IMUs and human gestures”. In: *Annual Conference Towards Autonomous Robotic Systems*. Springer. 2021, pp. 88–99.

[14] *GitHub - stm32duino/MotionFX: Sensor Fusion Library for ST sensors and microcontrollers — github.com*. `https://github.com/stm32duino/MotionFX`.

[15] Leah Taylor, Emily Miller e Kenton R Kaufman. “Static and dynamic validation of inertial measurement units”. In: *Gait & posture* 57 (2017), pp. 80–84.

[16] Martyna Białecka et al. “Shoulder Range of Motion Measurement Using Inertial Measurement Unit—Validation with a Robot Arm”. In: *Sensors* 23.12 (2023), p. 5364.

[17] *GitHub - RCmags/imuFilter: Arduino library for an IMU filter based on a quaternion — github.com*. `https://github.com/RCmags/imuFilter`.

[18] Kumar Bipin. *Robot Operating System Cookbook: Over 70 recipes to help you master advanced ROS concepts*. Packt Publishing Ltd, 2018.

[19] *it/ROS/Tutorials - ROS Wiki — wiki.ros.org*. `http://wiki.ros.org/it/ROS/Tutorials`.

[20] Sepideh Zolfaghari et al. “Speed Classification of Upper Limb Movements through EEG Signal for BCI Application”. In: *IEEE Access* PP (ago. 2021), pp. 1–1. DOI: `10.1109/ACCESS.2021.3102183`.

[21] Giulio Manni. “Sviluppo di interfaccia uomo-macchina per telecontrollo di cobot attraverso sensori inerziali”. Tesi di Laurea Magistrale. Politecnico di Torino, 2022.

[22] *GitHub - UniversalRobots/Universal_Robots_ROS2_Driver: Universal Robots ROS2 driver supporting CB3 and e-Series — github.com*. `https://github.com/UniversalRobots/Universal_Robots_ROS2_Driver`.

Ringraziamenti

Reputo innanzitutto doveroso esprimere un sentito ringraziamento e sincera gratitudine al mio relatore, Prof. Stefano Paolo Pastorelli, per la disponibilità, per gli insostituibili spunti e per avermi sapientemente indirizzato nei momenti più complicati. Desidero, inoltre, ringraziare gli Ingg. Valerio Cornagliotto e Michele Polito per i preziosi consigli con cui hanno affiancato pazientemente il mio lavoro in tanti mesi di sperimentazione, creando all'interno del Laboratorio di Meccanica un clima di serenità e collaborazione, indispensabile per raggiungere qualsiasi obiettivo.