

1 introduzione

Nella prima parte di questo capitolo introduttivo, verrà illustrata la richiesta formulata dall'Istituto Montecatone, con particolare attenzione alle caratteristiche che devono essere rispettate nella progettazione di protesi. Si farà riferimento ai requisiti di forza applicata, al design e agli standard normativi. Sarà presentato, in seguito, lo stato dell'arte tramite un'analisi della letteratura e dei dispositivi attualmente esistenti, concentrandosi in particolare sulle differenti tipologie di attuazione. Proseguendo, verranno esplorate le diverse possibilità di sensoristica e controllo impiegate nel settore. Il capitolo si concluderà con una rassegna storica dei progetti della famiglia GLOVE, a partire dai primi prototipi fino ad arrivare al sistema finale, che sarà equipaggiato con l'elettronica necessaria all'attuazione, tema principale di questa ricerca.

1.1 Presentazione del progetto

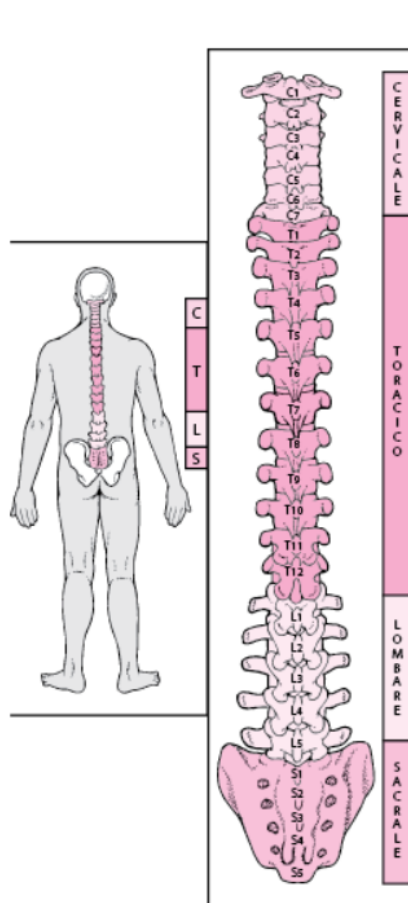
In questa sezione, verrà introdotto il progetto in generale, illustrando l'origine e la natura del prototipo commissionato, descrivendo le problematiche da risolvere e i prerequisiti iniziali. Questa introduzione risulta fondamentale per comprendere il sistema che ospiterà la parte di attuazione.

Il problema da risolvere.

L'incontro con l'istituto di Montecatone

Il Montecatone Rehabilitation Institute [1] è una struttura altamente specializzata del Servizio Sanitario Regionale dell'Emilia-Romagna, nonché un centro di riferimento per la riabilitazione intensiva di persone con lesioni midollari. Rappresenta, inoltre uno dei tre centri di riferimento regionale per le lesioni cerebrali acquisite. L'approccio dell'istituto va oltre l'aspetto clinico, includendo la creazione di programmi personalizzati volti a consentire ai pazienti di recuperare il miglior livello di autonomia possibile nella loro nuova condizione. Questo significa sostenere la ricostruzione dell'esistenza dei pazienti, mantenendo intatte la loro indipendenza e dignità, con

l'obiettivo di garantire una piena inclusione in tutti gli aspetti nella vita sociale. La ricerca rappresenta un elemento strategico nelle attività dell'istituto, fondamentale per la cura e l'assistenza medico-riabilitativa. Per questo motivo, all'università di Bologna è stata richiesta la progettazione di un esoscheletro per la mano, adattabile a diversi pazienti dell'istituto, che hanno subito traumi al midollo spinale, provando l'interruzione delle vie nervose ascendenti e discendenti, con conseguenti alterazioni temporanee, durature o permanenti delle loro funzioni motorie, sensoriali o autonome.



Livello della lesione	Effetto*
Fra C2 e C5	Paralisi di alcuni o tutti i muscoli coinvolti nella respirazione e di tutti i muscoli del braccio e della gamba Solitamente fatale salvo l'utilizzo di un ventilatore
Fra C5 e C6	Paralisi di gambe, tronco, mani e polsi Debolezza dei muscoli che muovono spalla e gomito
Fra C6 e C7	Paralisi di gambe, tronco e parte dei polsi e delle mani Movimento normale di spalle e gomiti
Fra C7 e C8	Paralisi di gambe, tronco e mani
C8 e T1	Paralisi delle gambe e del tronco Debolezza dei muscoli che muovono dita e mani Sindrome di Horner (abbassamento palpebrale, restringimento della pupilla ristretta ridotta sudorazione su un lato del volto) Possibile movimento normale di spalle e gomiti
T2 e T4	Paralisi delle gambe e del tronco Perdita di sensibilità sotto i capezzoli Movimento normale di spalle e gomiti
T5 e T8	Paralisi delle gambe e del tronco inferiore Perdita di sensibilità sotto la gabbia toracica
T9 e T11	Paralisi delle gambe Perdita di sensibilità sotto l'ombelico
T11 e L1	Paralisi e perdita della sensibilità nelle anche e nelle gambe
L2 e S2	Vari modelli di debolezza e intorpidimento della gamba a seconda del livello preciso della lesione
S3 e S5	Intorpidimento del peritoneo

* A qualsiasi livello del midollo spinale, una lesione grave può provocare perdita del controllo della vescica e dell'intestino.

Figura 1-1 Esempi di varie tipologie di lesione al midollo spinale

In particolare, i pazienti oggetto di questo studio hanno riportato una lesione all'altezza delle vertebre C6 e C7, responsabile della paralisi di gambe, tronco, mani e parte dei polsi. Come illustrato nella Figura 1-2, questi pazienti conservano i movimenti della spalla, come la pronazione (rotazione verso l'interno), la flessione e, in alcuni casi, l'abduzione del polso. Inoltre, sono in grado di flettere il gomito tramite l'utilizzo del

bicipite, ma non di estenderlo a causa dell'indebolimento del tricipite.

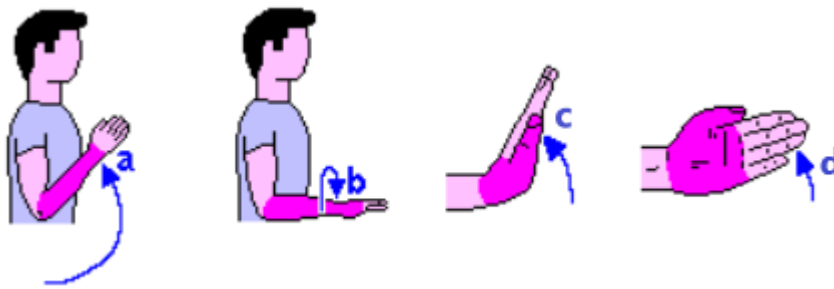


Figura 1-2 Flessione gomito (a), Pronazione polso (b), Flessione polso (c), Abduzione polso (d)

1.1.1 Obbiettivi e requisiti del prototipo finale

1.1.2.1 abilità motoria della mano

La mano è un meccanismo molto complesso e replicarla perfettamente risulta estremamente difficile. L'obiettivo funzionale del prototipo consiste nel realizzare un sistema capace di svolgere semplici funzioni. In una prima approssimazione, si ipotizza un dispositivo semplice ON/OFF, in grado di eseguire un'apertura e chiusura della mano, permettendo all'utente di compiere azioni quotidiane come afferrare una bottiglietta d'acqua, il telefono, un bicchiere, ecc. Il guanto dovrà essere comandato nel modo più diretto possibile, interfacciandosi con l'utente senza ricorrere a stadi intermedi come telecomandi o joystick, che comprometterebbero la naturalezza dell'uso.

1.1.2.2 Forze da generare

Un esoscheletro riabilitativo, secondo l'articolo "Flexo Glove" [2], deve soddisfare determinati requisiti per essere ritenuto idoneo a pazienti con lesioni al midollo spinale. Questi pazienti, costretti a vivere in carrozzina, generalmente non hanno alcuna forza nella mano, eccetto quella derivante dalla promozione del polso. I dispositivi devono pertanto garantire una forza di chiusura di 22N e una forza di presa complessiva di 48N, esercitata su oggetti con un diametro massimo di 90 mm [3].

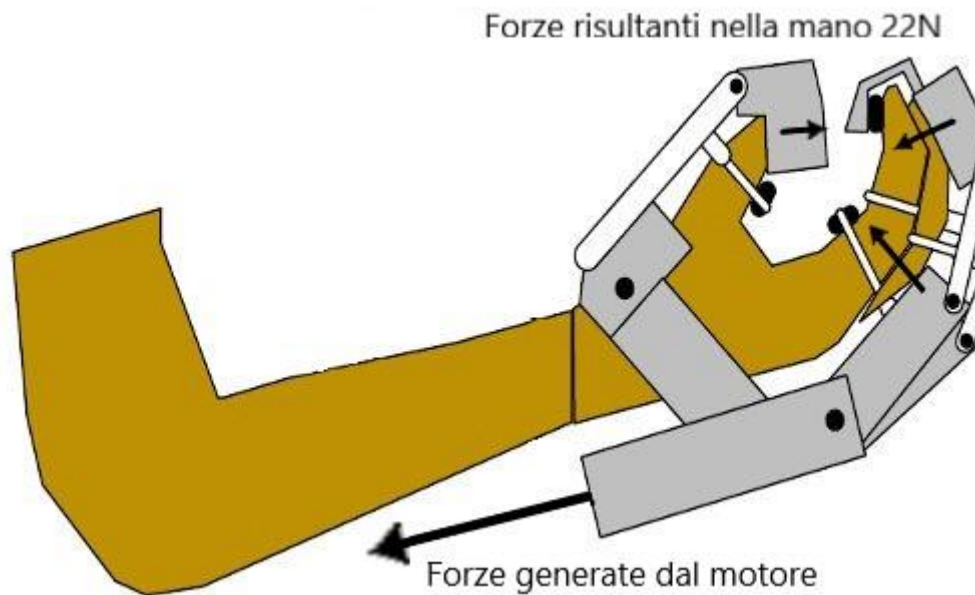


Figura 1-3 Esempio delle forze in gioco in un esempio di esoscheletro

1.1.2 Esoscheletro

Gli esoscheletri sono dispositivi indossabili che forniscono un'interazione fisica tra la tecnologia del meccanismo e l'utente, con lo scopo di migliorare o restituire il movimento, rispettando la struttura ossea e muscolare. Trattandosi di dispositivi indossabili, è essenziale garantire un controllo rigoroso sulla sicurezza, data la vicinanza al corpo. Inoltre, devono essere leggeri e in grado di adattarsi ai movimenti articolari dell'utente senza richiedere regolazioni eccessive o complesse calibrizioni. La progettazione di dispositivi biomedicali è soggetta a una serie di normative nazionali e internazionali che mirano alla prevenzione dei rischi e alla garanzia della qualità per l'utilizzatore.

1.2 Stato dell'arte

Il paragrafo seguente riporta le informazioni acquisite tramite la ricerca bibliografica sui soft glove, focalizzandosi su alcuni interessanti prototipi sperimentali e dispositivi commerciali attualmente disponibili. Successivamente, verranno esaminate le principali tipologie di azionamento e metodi di controllo impiegati nel settore.

1.2.1 Tipologie di esoscheletro per la mano

Attualmente, gli esoscheletri per la mano si dividono in due macrocategorie: morbidi e rigidi. La differenza risiede nella struttura meccanica adottata: nel primo caso è interamente flessibile, nel secondo è costituita da elementi rigidi e solidali alle falangi. Questa differenza potrebbe essere espressa in modo grossolano differenziando le nomenclature in: guanto ed esoscheletro. I dispositivi si distinguono anche per le loro funzionalità. Nel caso dei pazienti considerati, che hanno perso completamente il movimento delle dita, l'esoscheletro richiesto deve essere di tipo assistivo, ovvero in grado di eseguire autonomamente i movimenti desiderati, piuttosto che limitarsi a fornire un incremento di forza o ad accompagnare l'individuo in un percorso di recupero delle capacità motorie.

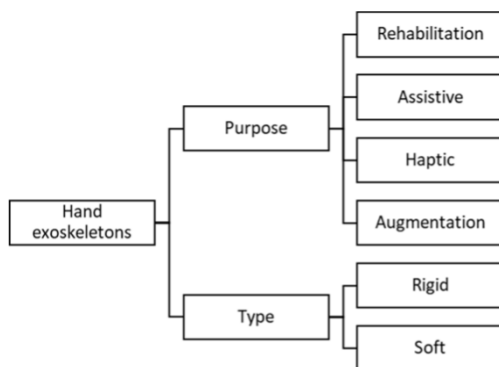


Figura 1.4 Gerarchia dei diversi tipi e scopi dei guanti robotici

1.2.1 Attuazione

Esistono diverse modalità di attuazione per questi dispositivi, spaziando dai motori elastici in serie (Series Elastic Actuators) ai motori elettrici, dai sistemi pneumatici ai materiali a memoria di forma (Shape Memory Alloy, SMA). La trasmissione delle forze può avvenire tramite:

- **Struttura meccanica rigida:** composta da giunti meccanici, ingranaggi o drive diretti.
- **Tramite cavi:** in configurazione tendon-driver o tipo bowden.
- **Attraverso meccanismi flessibili senza giunti** (soft glove): di tipo tendondriven o con muscoli artificiali.

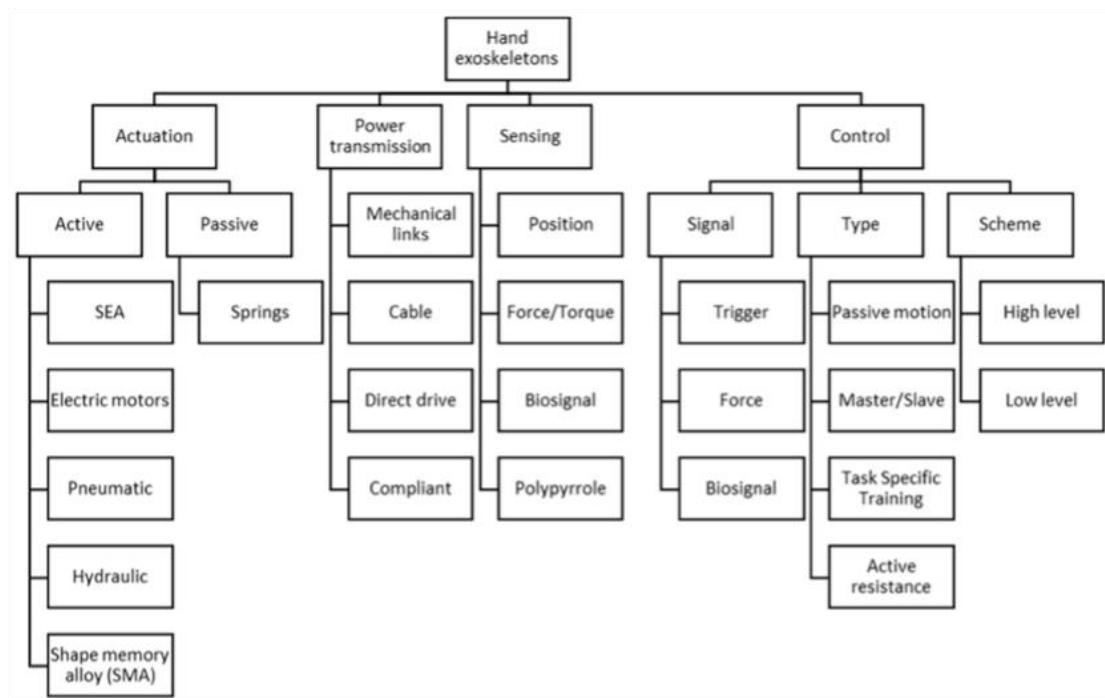
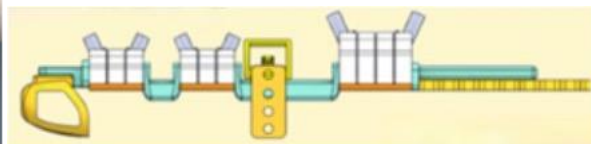
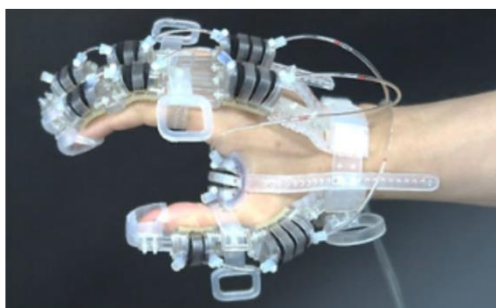


Figura 1-5 Attuazione, trasmissione, sensoristica e controllo degli esoscheletri

1.2.1.1 Attuazione pneumatica

La prima tipologia di attuazione esaminata è quella pneumatica, in cui il trasferimento delle forze avviene tramite l'uso di aria compressa o altri gas pressurizzati. Molti gruppi di ricerca hanno testato questa possibilità, come Yun Kang Cho [4] e Yap Goh Yeow [5]. Il primo gruppo (Yun Kang Cho) ha prodotto l'Exo-Glove PM, il cui funzionamento è reso possibile da un attuatore inserito in una struttura esterna costituita da canali d'aria rigidi collegati a uno strato di gomma e un tubo ricoperto di tessuto. La forza viene generata dall'espansione dello strato di gomma, che tende a flettersi quando viene iniettata aria compressa.



Il secondo dispositivo citato, fa affidamento a reti di canali e camere d'aria inserite negli attuatori, che si gonfiano in seguito a pressurizzazione. Il funzionamento, più in particolare, è dovuto alla presenza di uno strato di tessuto tensione-limitante sul fondo

degli attuatori. In seguito a pressurizzazione, la superficie superiore si espande a causa del rigonfiamento delle camere d'aria incorporate, mentre è assente una deformazione della superficie inferiore. Il risultato è la produzione di una coppia e la flessione delle dita. 7 Per entrambi i guanti citati, i tempi di attuazione sono di circa 2s, non sono così necessari eccessivi picchi di energia per il funzionamento. Tale aspetto, in aggiunta al fatto che il sistema risente di una scarsa usura, vista l'assenza di attrito, rende i glove pneumatici ottimi per usi prolungati, mostrandoli quindi adatti al compimento di task di routine giornaliera. Lo svantaggio di tali dispositivi risiede nel prezzo, elevato rispetto ad altre tecnologie, e la difficile prototipazione, dovuta alla complessa costruzione e gestione delle camere stagne. Esse devono essere progettate in base a particolari modelli matematici simulativi del comportamento delle dita e poco adattabili a mani di diverse dimensioni.

1.2.2.2 Attuazione SMA

Un'interessante soluzione attuativa, da tenere in considerazione, è quella che impiega i materiali SMA sotto forma di cavi che simulino il ruolo di tendini artificiali. Si tratta di materiali "intelligenti" che godono dell'effetto piezoelettrico diretto ed inverso: se sottoposti a sollecitazione meccanica sviluppano una carica elettrica, dualmente se sottoposti a una tensione elettrica presentano una deformazione nel reticolo cristallino. Sfruttando l'effetto inverso, i tendini artificiali SMA sono in grado di ridurre la propria lunghezza, con conseguente possibilità di chiusura del guanto. Tali materiali godono di due importanti proprietà:

- **effettua memoria di forma:** grazie al quale ricordano la configurazione cristallina macroscopica, antecedente la deformazione, la quale può essere riacquisita se sottoposti a calore.
- **Comportamento superelastico:** per cui non subiscono deformazioni residue durante la fase di scarico.

L'attuazione con materiali SMA, sebbene abbastanza recente, è stata sperimentata da diversi gruppi di ricerca come LAI Teh Chiu [6], Tao et Al [7], Liu et Al [8] e Eang et Al [9]. Il prototipo proposto da Yeh et Al, ad esempio, attraverso una Lega SMA di titanio-nichel-rame attivata la temperatura di 55°, garantisce la movimentazione della ditta con una forza di 22N, reputata sufficiente alla riabilitazione della mano. Sebbene dispositivo abbia superato tutti gli standard che normano gli esoscheletri e riabilitativi,

tale tipologia di azionamento presenta notevoli svantaggi. I lunghi tempi di attuazione, di circa 300s, dovuti al raffreddamento del materiale e le elevate correnti elettriche in circolo, limitano le possibilità di applicazione.

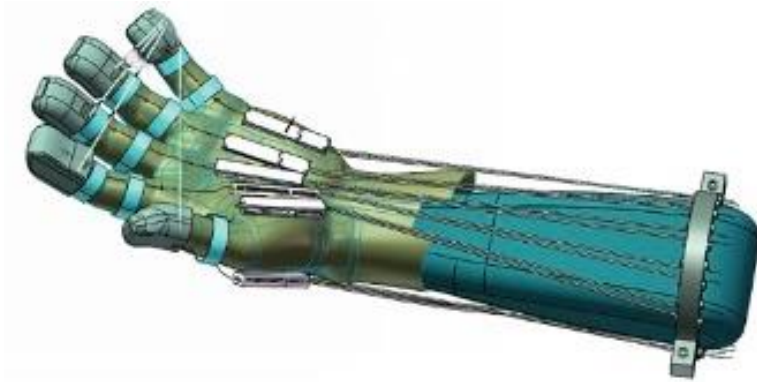


Figura 1-7 Modello di Wang et Al

1.2.1.2 Attuazione con motori DC

Grazie alla recente miniaturizzazione dei motori allo sviluppo della stampa 3D di materiali resistenti, economici e facilmente modellabili, è stato largamente studiato il settore dei guanti attuati da motore DC. La progettazione risulta spesso semplificata e più economica rispetto agli altri metodi di attuazione, portando così tale azionamento ad essere il più sperimentato. La tipologia di trasmissione della forza scelta è tipicamente tendon-driven (Figura 1-8), ovvero sono presenti dei cavi che percorrono le dita, e che attuati da un motore di DC, sono responsabili della loro chiusura ed apertura. Tale caratteristica determinante per poter alleggerire il dispositivo, in quanto risulta possibile scomporre il sistema in un semplice guanto collocabile sulla mano e una struttura di controllo e attuazione posizionabile in remotamente. La trasmissione della forza viene garantita da sistemi di pulegge e motoriduttori, che determinano un'elevata precisione e un controllo di chiusura proporzionale, caratteristica fondamentale ed assente nei sistemi pneumatici

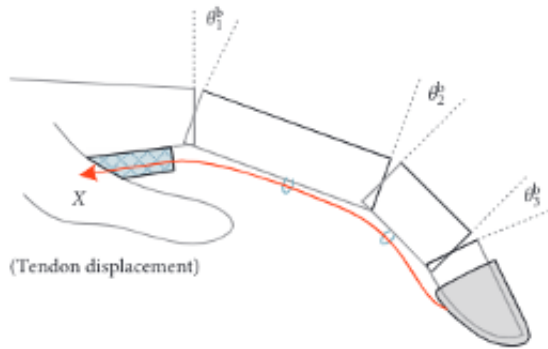


Figura 1-8 Schema di un'attuazione tendon-driven

I sistemi di trasmissione sono in grado di sopportare forze notevoli, risentendo però della problematica dell'attrito, e conseguente, dell'usura per frizione. Tali aspetti ne preferiscono l'impiego in applicazioni in cui sono richieste sporadiche forze elevate, mentre rimangono preferibili sistemi pneumatici per continue attività meno esose.

• **Exo-Glove Poly II**

attraverso un design di soli quattro componenti facilmente scalabili e adattabili a diversi pazienti, Exo-Glove Poly II [10] è realizzato in plastica per essere economico, facilmente lavabile ed impermeabile, grazie anche alla collocazione in remota dei motori e l'elettronica di controllo. Il pollice non è motorizzato, ma è comunque rigido in modo da permettere il movimento di abduzione rendere possibile una presa migliore. I tendini artificiali sono posti ai lati delle dita schermati da guagni di protezione, mentre i cavi utilizzati sono posti in una configurazione adattatore abilitativo ad allentamento duale (*dual slack enabling actuator*).

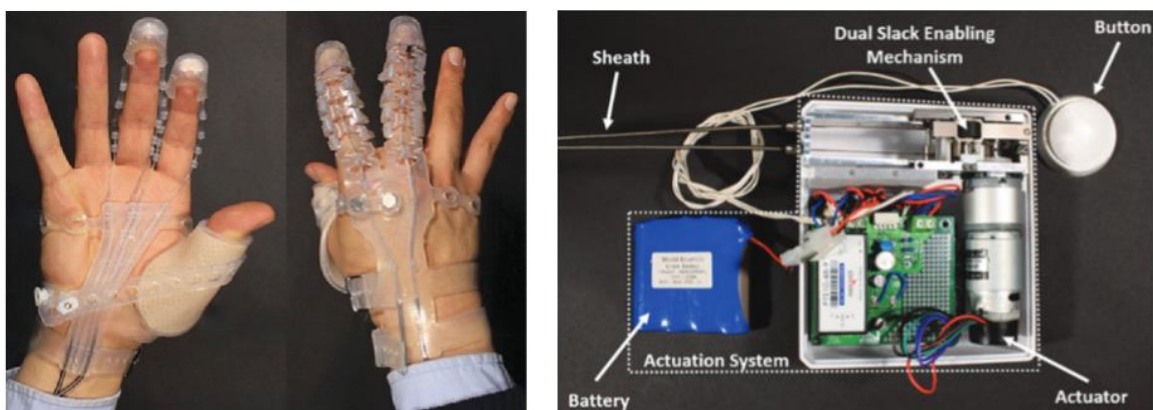


Figura 1-9 Prototipo Exo-Glove Poly II

• Flexo-Glove

Il modello Flexo-Glove [11], propone invece una soluzione innovativa che combina la modularità e customizzazione del design con una notevole portabilità. In quanto garantisce 22N per la presa a pinza e 48N per la presa normale con diametro massimo dell'oggetto di 81mm. La particolarità del dispositivo in questione è la possibilità di essere impiegato in modalità assistiva, con un wireless sensing via EMG che consente di captare le intenzioni dell'utente, e di garantire in modalità riabilitativa un controllo tramite applicazione.

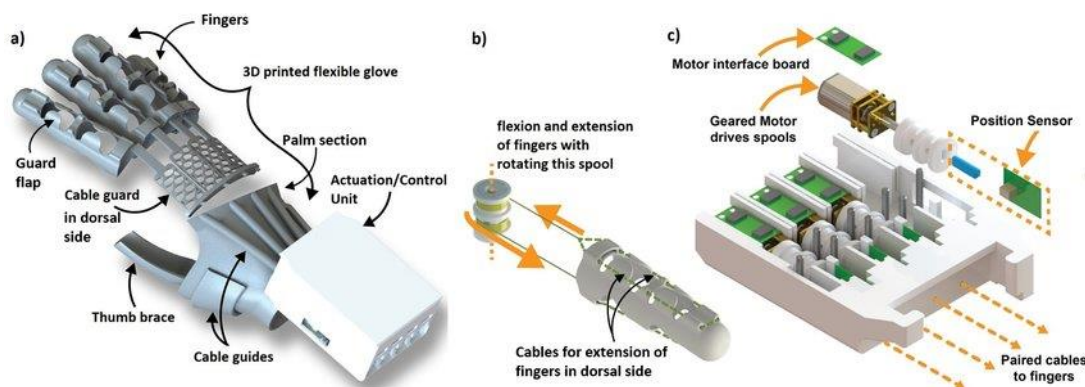


Figura 1-10 Prototipo Flexo-Glove

• Modello di Chen et Al

Un altro dispositivo da tenere in considerazione quello realizzato dal gruppo di ricerca di Chen et Al [12]. La sua particolarità risiede nell'impiego del machine learning per la definizione di algoritmi in grado di replicare nel guanto della mano con disabilità motoria, la forza e le configurazioni spaziali rilevate da un'altro guanto, indossato nella mano sana dello stesso paziente, riuscendo a ricreare 16 posizioni diverse con una correttezza del 93%. Si tratta di un dispositivo riabilitativo che adopera la terapia a specchio.

• Tenoexo

Il modello realizzato all'interno del laboratorio ingegneristico di riabilitazione dell'ETH di Zurigo è il RELab Tenoexo [13], o in versione pediatrico PEXO, guanto robotico ortopedico per la terapia e l'assistenza nelle attività quotidiane di individui con disfunzioni manuali più o meno gravi. Il controllo è garantito da una serie di pulsanti, la voce e sensori posizionati direttamente nel guanto. L'intero dispositivo

comprende un leggero modulo da posizionare sulla mano, degli attachments per il braccio e un modulo, posizionabile comodamente sulla schiena del paziente, che include motori, alimentazione e controllori. Tenoexo si differenzia dai modelli tendon driven in quanto presenta un metodo di trasmissione tramite meccanismo a triplo layer di molle traslanti, grazie al quale è possibile calibrare la forza e cambiare le modalità di utilizzo, adattandosi passivamente agli oggetti per ottimizzare la presa. La forza massima applicabile con la sola falange distale è di 5N, sufficiente per afferrare da alzare la maggior parte degli oggetti quotidiani.



Figura 1-11 Prototipo Tenoexo

- **Emovo Assit**

Emovo Assit [14] è un esoscheletro passivo per mano, ideato per aiutare gli individui con medie parziali e debolezza di tale parte del corpo. Ancora nella sua fase prototipale, ha un controllo simile a Tenoexo, poiché è in grado di chiudere le dita dopo un primo movimento eseguito da parte del paziente, inoltre comandabile tramite pulsante o applicazione. Il dispositivo è composto da l'esoscheletro, dagli attachments per il corpo e dal controllore virgola che in questo prototipo non è vincolato al paziente. Sistema di attuazione prevede dei tendini artificiali che aprono e chiudono le dita con una forza preimpostata, uguale per entrambi i movimenti. L'elettronica, le batterie e i motori sono contenute nel modulo del controllore, posizionato remotamente rispetto al guanto, per alleggerire l'esoscheletro e garantirne l'impermeabilità.



Figura 1-2 Prorotipo Emovo Assitst

1.2.2 Dispositivi commerciali

Esistono molteplici guanti robotici assistivi già disponibili in commercio, si differenziano principalmente per fascia di prezzo e tipologia di impiego. Il mercato spazia tra guanti molto semplici, come quelli a compressione che favoriscono la circolazione del sangue, fino a più complessi e costosi soft glove capaci di aiutare l'utilizzatore nelle attività quotidiane.

1.2.2.1 Nuada

Il soft glove assistivo NUADA [15], venduto al prezzo di 4000\$, grazie al disegno compatto e al sistema di controllo intuitivo, è reputato il modello più interessante attualmente sul mercato. Il dispositivo è composto da un guanto, al cui interno sono cuciti dei tendini artificiali, e da un bracciale, che comprende un sistema di controllo in grado di produrre forze di supporto a ciascun dito. L'interfacciamento con l'utente è completamente naturale e non richiede nessuna azione da parte dell'utilizzatore. Grazie ad una combinazione particolare di sensori, il sistema percepisce autonomamente quando l'individuo richiede la chiusura della mano. Per garantire una presa più sicura, a mano chiusa, i tendini artificiali conservano la configurazione raggiunta finché i sensori non avvertono l'intenzione di riapertura della mano. La peculiarità di questo dispositivo è la possibilità di raccolta dati in tempo reale, i quali vengono trasferiti ad un'applicazione visibile dall'utente e dal centro medico, al fine di poter essere elaborati portando ad un costante miglioramento delle prestazioni.



1.2.2.2 Carbonhand

Il modello Carbonhand [16], reperibile sul mercato al prezzo di 7000\$, realizzato dall'azienda Bioservo, è un soft glove simile a NUADA per funzionamento, ma meno compatto vista la separazione del guanto dal sistema di attuazione e controllo. Il guanto è un dispositivo che aumenta la potenza manuale rafforzando la presa. L'intenzione di chiusura è rilevata da sensori di pressione posizionati sui polpastrelli, che percepiscono la volontà di movimento dell'individuo. La sensibilità di tali sensori può essere calibrata in base al grado di disabilità della mano del paziente. Il funzionamento è garantito da tendini artificiali attuati da motore elettrico DC posizionato a distanza dal guanto. Tecnologia avanzate permettono la scelta di diverse modalità e livelli di integrazione dell'aiuto nella chiusura, rendendo così possibile lo svolgimento di svariate attività. Il dispositivo comprende il guanto collegato ad una power unit indossabile sulla schiena o sul polso per garantire più libertà e leggerezza possibile.



Figura 1-14 Dispositivo Carbonhand

1.2.2.3 Neomano

Il soft glove Neomano[17], realizzato da Neotech ad un prezzo di mercato di soli 2000\$, è attualmente la soluzione più economica. Pensato per utenti che sono in grado di compiere movimentazione di polso e braccio, ma che possiedono la scarsa o quasi assente forza nelle mani, rientra nella categoria degli assistive glove. Il dispositivo è costituito da un guanto che comprende pollice, indice e medio, una power unit e un controllore. Le due dita superiori sono dotate di tendini artificiali cuciti all'interno del guanto, che vengono avvolti da un motore DC posizionato sul lato della mano. Il guanto presenta una barra metallica flessibile connessa al pollice, in modo da poterlo impostare nella posizione desiderata, che varierà in base alla presa da eseguire. Il motore provvede all'apertura e alla chiusura delle due dita mobili quando vi è una pressione del pulsante corrispondente. Vista la semplicità ed economicità del sistema, non è presente un sistema di rilevazione autonoma delle intenzioni dell'utente, inoltre non è possibile eseguire una calibrazione proporzionale della forza applicata.



Figura 1-15 Dispositivo Neomano

È importante notare come i dispositivi attualmente presenti sul mercato siano di natura principalmente riabilitativa, ovvero pazienti per i quali è stato intrapreso un percorso di riacquisizione delle capacità motorie manuali, o che semplicemente necessitano un rafforzamento. Inoltre, nei sistemi autonomi, l'attuazione è quasi sempre attivata da un primo, anche se lieve, movimento effettuato direttamente dalla mano del paziente. Per ciò che concerne i pazienti dell'Istituto Montecatone con lesione alle vertebre C6 e C7, la mobilità manuale è tuttavia assente e persa per sempre. Il dispositivo richiesto deve quindi essere in grado di attuarsi automaticamente tramite segnali neurologici raccolti a monte della mano, dove sono più intensi, e di garantire il comportamento autonomo dell'intero movimento.

1.2.3 Sensoristica e controllo

Nella sezione seguente verranno esposti i principali sensori utilizzati nel settore degli esoscheletri per la mano e le possibili soluzioni per il controllo.

1.2.3.1 Sensori

Tipicamente sono utilizzati sensori collocati direttamente all'interno degli esoscheletri, per fornire un riscontro al sistema di controllo, utilizzabile dall'utente o dal terapeuta, per monitorare gli sviluppi della riabilitazione. Tali dati, di fondamentale importanza, possono essere di varie tipologie e natura; in modo da poter monitorare più grandezza di interesse.

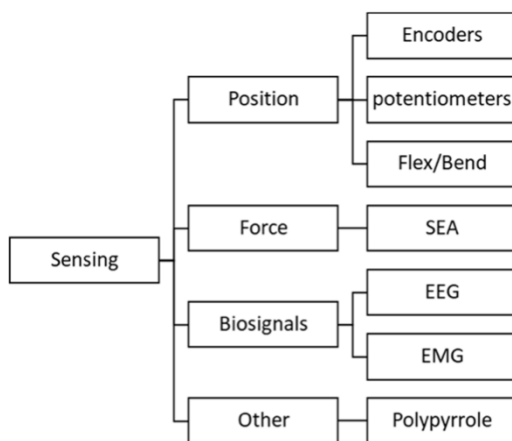


Figura 1-16 Illustrazione dei metodi di sensing per esoscheletri

Per tenere traccia dell'evoluzione degli spostamenti nel tempo, sono necessarie misurazioni di posizione, effettuabili tramite encoder solidali all'albero del motore, se presente. Essi forniscono la misura dell'angolo percorso, da cui volendo si può ricavare anche la velocità.

Un'altra soluzione possibile riguardo l'utilizzo di potenziometri ubicati in prossimità delle dita, ciò rende però il sistema più ingombrante, vista la diretta collocazione sul guanto.

Importante è anche la conoscenza della forza applicabile dalle dita, misurata spesso sul polpastrello tramite *Series Elastic Actuator*.

Uno dei metodi più efficaci per rilevare le intenzioni dell'utente è attraverso il biosegnali, misurando l'attività muscolare elettrica o le funzioni motorie. Ne esistono di due tipologie principali che risultano utili all'analisi, vengono ottenute tramite

elettromiografia superficiale (sEMG) o elettroencefalogramma (EEG). L' sEMG studia l'attività muscolare attraverso il segnale elettrico emanato dal muscolo stesso. Per ciò che riguarda la movimentazione della mano, i muscoli responsabili per la flessione e l'estensione delle dita attraverso i tendini, sono collocati nell'avambraccio.

L'EEG Misura i segnali elettrici che si scatenano nel cervello in corrispondenza dell'attuazione di alcuni muscoli. Tali segnali richiedono un filtraggio prima di essere processati in quanto è presente un elevato rumore di fondo.

1.2.3.2 Controllo

Un'importante caratteristica richiesta dall'Istituto consiste nel rendere l'utilizzo dell'esoscheletro il più semplice e naturale possibile, cercando di ridurre ingombri e componenti intermedi che andrebbero complicare l'esperienza d'uso al paziente. Pulsanti ed applicazioni, devono quindi essere sostituite con tecniche di attuazione automatizzate. Per ottenere ciò, si fa tipicamente affidamento ai biosegnali, raccolti tramite dispositivi posizionati direttamente sul paziente e trasmessi ai componenti elettronici con lo scopo di essere convertiti in segnali utili a comandare l'azionamento.

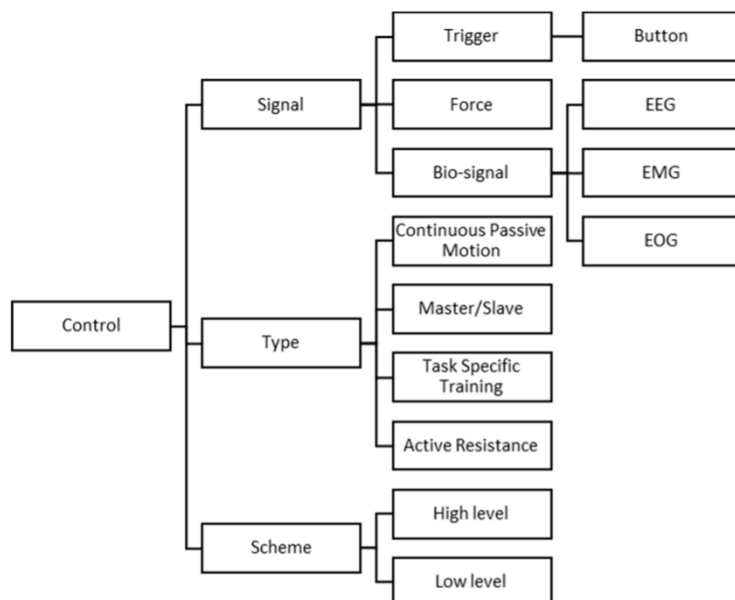


Figura 1-7 Segnali, tipologie e schemi di controllo

- **Controllo muscolare**

L'attività muscolare del corpo umano genera un campo elettrico che raggiunge la superficie cutanea. Più in dettaglio, si tratta di potenziali elettrici causati dalla depolarizzazione elettrica delle fibre muscolari in risposta all'impulso elettrico della sinapsi neuromuscolare. L'sEMG viene ricavato nell'area richiesta dell'avambraccio tramite elettrodi di superficie, i quali vengono posizionati sulla pelle nella direzione della fibra muscolare, al fine di prelevarne il segnale mioelettrico. Gli elettrodi misurano il livello di attività elettrica nel muscolo, segnale che viene poi trasmesso ad un'unità di elaborazione per essere gestito e reso utilizzabile

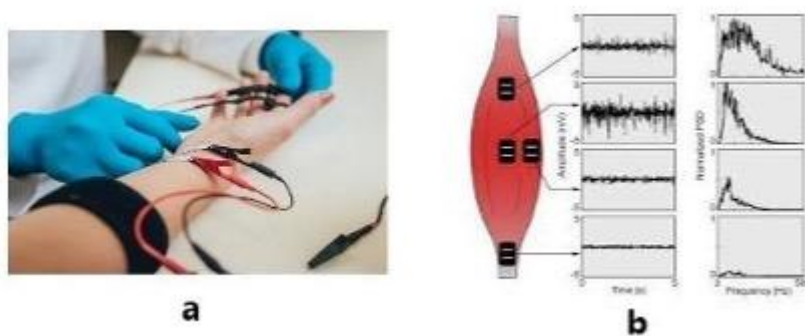


Figura 1-18 esempio di EMG (a), segnali ricavabili dal muscolo (b)

In commercio sono disponibili dispositivi creati appositamente per controllare le più disparate operazioni tecnologiche tramite elettromiografia. Tipicamente i sensori utilizzati sono monouso, impiegati perlopiù in ambito medico. Sono tuttavia presenti anche dispositivi più complessi, realizzati senza trascurare l'estetica e la praticità di indossamento, come nel caso di integrazione in comandi bracciali elastici. Quest'ultima si è dimostrata una soluzione poco invasiva a livello fisico e visivo, inoltre il training necessario abbastanza breve grazie alla semplicità di utilizzo.



Figura 1-19 Dispositivo portatile per EMG (MYO)

- **Controllo mentale**

per poter comunicare con il corpo, l'encefalo manda impulsi elettrici ai muscoli tramite i nervi, che trasportano l'informazione per attivarli. Tali impulsi generano un campo elettrico simile ad una vibrazione, rilevabile sul cuoio capelluto tramite EEG. Questa procedura prevede una cuffia di elettrodi posizionata sulla testa del paziente, in grado di rilevare segnali, che vengono processati da analizzati da un *Brain computer Interface* (BCI) per ricostruire il comportamento cognitivo del paziente.

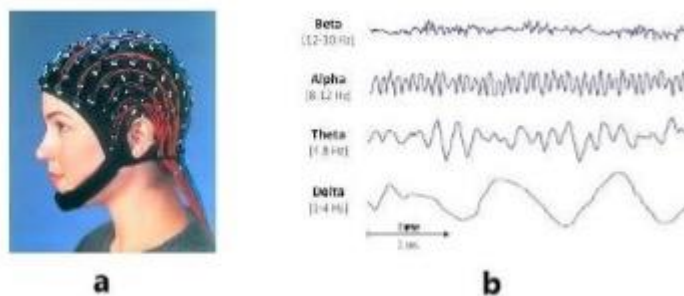


Figura 1-20 casco per EEG (a), segnali ricavati dal cervello (b)

- **Controllo oculare**

Attraverso l'esame della elettroculografia (EOG) è possibile individuare i movimenti degli occhi, grazie alla registrazione la differenza di potenziale tra cornea e retina. La misurazione viene ancora una volta mediante elettrodi. Questo segnale, spesso combinato con quello ottenuto dall' EEG, permette di attuare un dispositivo attraverso i movimenti dell'occhio, codificando opportunamente la relazione che intercorre tra tale movimento e l'intenzione del paziente. Tale tecnica non viene più però impiegata frequentemente poiché, nonostante la semplicità, è un sistema abbastanza invasivo e poco pratico per l'utilizzatore

- **Controllo in pressione**

Esaminando il senso del tatto è possibile realizzare un controllo in pressione. Speciali sensori tattili sono in grado di emulare la sensibilità della pelle, permettendo di mappare la pressione di contatto. Tali sensori, estremamente sottili e flessibili, sono composti da una matrice di righe e colonne di materiale semi-conduttivo che cambia la propria resistenza elettrica se sottoposto ad una forza. La misura della variazione di resistenza in ogni punto permette di quantificare la forza applicata e di individuarne la posizione sulla superficie del sensore. Grazie

all'impiego di tali sensori, il dispositivo è in grado di percepire il contatto con un oggetto. Come metodo di controllo, risulta essere poco invasivo per il paziente, ma è necessaria una movimentazione autonoma per essere utilizzato.

- **Controllo vocale**

i suoni sono vibrazioni che assumono diverse configurazioni in base alla struttura dell'apparato fonetico che le ha prodotte. Tali onde sono accompagnate da modifiche del flusso d'aria, generandone una variazione di pressione, convertibile in segnale elettrico grazie a questa tecnica, è possibile trasformare la pressione sonora dei comandi vocali in segnali elettrici che permettono di eseguire le attività richieste.

1.3 Sistema GLOVE

Il seguente, ed ultimo paragrafo introduttivo, illustrerà le fasi di sviluppo dei prototipi GLOVE, realizzati per soddisfare la richiesta della Clinica di Montecatone. Sono state testate diverse tipologie di attuazione e, una volta appurato che quella più efficiente e semplice da realizzare fosse quella motorizzata, si è deciso di approfondirla costruendo più modelli.

Per testare inizialmente il corretto funzionamento del prototipo UNO (capitolo 1.3.3), è stata utilizzata una acerba versione di controllo, utilizzando motore ed elettronica temporanei, al momento già disponibili in laboratorio, nell'attesa di cogliere da ordinare i componenti finali, ampiamente analizzati nel capitolo 2.1. Il prodotto finale di questo elaborato di tesi sarà invece costituito dal controllo del motore dc utilizzato per attuare il Prototipo DUE (capitolo 1.3.4).

1.3.1 Prototipo SMAtronic

Il primo prototipo, sviluppato da un gruppo di studenti dell'università Alma Mater di Bologna: Ivano de Paola, Anna Fritz, Eleonora Valeri, Mirko Paltrinieri, Xavier Hajjar e Marilyn Gilbert, prevedeva un azionamento tramite tecnologia SMA. Denominato

SMAtronic [18] fa affidamento all'attuazione tendon-driven, ovvero sono presenti cavi che svolgono la funzione di tendini, che, se tirati dagli SMA, permettono la chiusura dell'esoscheletro.

In particolare, in questo primo prototipo è stato imposto l'utilizzo degli SMA come requisito base del progetto, i successivi studi e sviluppi sono volti alla creazione di un sistema che insultasse al meglio le qualità, andando a limitare il più possibile gli svantaggi.

1.3.1.1 Schema del sistema

Si procede ora ad esporre i vari blocchi funzionali dal quale è composto il prototipo:

- **Blocco di alimentazione:** Comprende la batteria, un regolatore e un dispositivo per visualizzare la tensione fornita dalla batteria, utile per la corretta alimentazione del sistema e per visualizzare la carica residua.
- **Blocco di attuazione elettrica:** Comprende i fili SMA alimentati tramite un modulo a ponte doppio ad H controllato attraverso un segnale PWM fornito dal microcontrollore.
- **Blocco di attuazione meccanica:** comprende un componente per lo sblocco manuale del dispositivo, un paranco per sfruttare al meglio il ridotto accorciamento che lo SMA è in grado di esercitare, e un meccanismo di frenatura che sfrutta una camma e un rullo cilindrico per limitarne il consumo di energia
- **Blocco di interfacce esterne:** comprende i moduli di interfaccia tra scheda e utente, quali moduli acustici per suoni di verifica o allarme, i moduli Bluetooth per dare ordini al sistema o collezionare dati e moduli Input/Output come pulsanti e schermi
- **Blocco di design:** comprende le tecniche di collegamento dei cavi al guanto, tramite anelli e cappucci dotate di passanti.

1.3.1.2 Componenti

Il microcontrollore scelto per il controllo del dispositivo è il modello Uno della famiglia Arduino, dispositivo robusto e dal semplice utilizzo. L'alimentazione del dispositivo avviene tramite batterie Roaring Top a polimero di litio con tensione 11,1V capacità 2200mAh.

I cavi SMA sono una lega intermetallica di nichel e titanio, presenti in eguale percentuale, dimensionati a 200 μm (micrometri) consentono una deformazione dell'8.5% e sviluppano una forza massima di 19 N (Newton). Tale valore di forza viene comunque reputato sufficiente in quanto il sistema prevede la sola chiusura di pollice e indice. Inoltre, essi presentano un'ottima resistenza a corrosione, buona duttilità ed elevata biocompatibilità.

Per alimentare i fili SMA viene impiegato il modulo L298, scheda basata sul circuito integrato ST L298N. Si tratta di un doppio ponte ad H in grado di erogare fino a 2° (Ampere). Il dispositivo è munito di dissipatore termico e viene controllato dalla scheda Arduino tramite modulazione PWM.

1.3.1.3 Prototipazione

Il circuito elettrico è costituito dall'Arduino Uno, il modulo L298, un potenziometro, un buzzer e 4 led

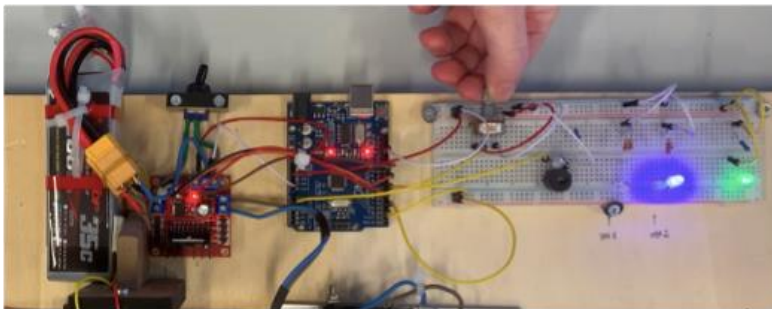
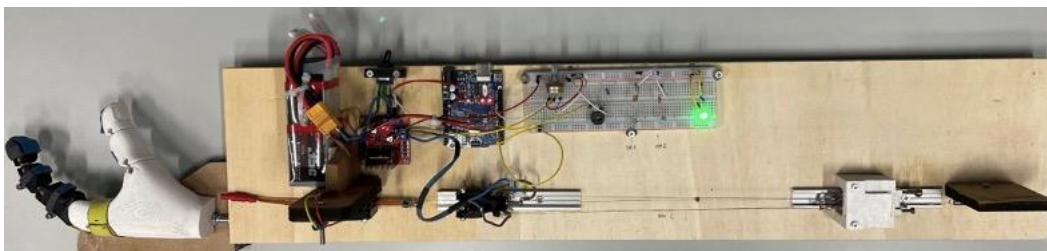


Figura 1-21 Circuito elettrico del prototipo SMA

Il ruolo del potenziometro è quello di ricreare funzionamento del segnale elettromiografico durante un'analisi ON/OFF. Tale segnale, acquisito grazie all'attivazione del muscolo, viene dapprima rettificato e poi sviluppato per essere confrontato con un segnale di riferimento, al fine di determinare la soglia di attivazione del muscolo. Sopra tale valore il muscolo sarà considerato attivo, mentre al di sotto sarà interpretato come a riposo. Inoltre, è necessario filtrare le attivazioni residue, ovvero i movimenti che durano solo pochi istanti non rappresentano la vera intenzione di azionare dispositivo. Per questo motivo, il muscolo sarà considerato attivo solo

quando la soglia sarà superata per un intervallo di tempo superiore ad un secondo. L'aggiunta del buzzer ha lo scopo di allertare, attraverso l'emissione del suono SOS in linguaggio morse, quando è presente un errore nella lettura del segnale, condizione che porterebbe a una continua attivazione degli SMA, con conseguente eccessivo dispendio energetico. L'aggiunta del LED ha il solo scopo di accompagnare il segnale sonoro prodotto dal buzzer. Al fine di evitare situazioni critiche come la caduta di oggetti, la situazione di allarme comporta la ritenuta della mano nella sua configurazione di chiusura fino allo sblocco manuale del dispositivo. Il ruolo degli altri tre LED è di indicare la tensione del sistema e l'attivazione dei cavi SMA. Il design dell'esoscheletro vero e proprio è frutto della progettazione meccanica di anelli per ogni falange di un cappuccio per ogni polpastrello, entrambi dotati di passanti attraverso i quali fa scorrere il filo di collegamento. Per tale scopo è stata utilizzata una semplice lenza da pesca vista l'elevata resistenza meccanica e lo spessore ridotto. La posizione dei passanti è stata individuata grazie ad uno studio del meccanismo della mano e un'opportuna modellazione matematica. I fili SMA, una volta azionati, si accorciano andando a tirare la lenza collocata internamente alla mano, garantendo così la chiusura di quest'ultima. Per la riapertura sono previsti invece delle molle, caratterizzate da specifici coefficienti di elasticità, posizionate sul dorso della mano tra ogni coppia di anelli o cappucci adiacenti



1.3.1.4 Funzionamento

Il filo SMA viene azionato dal controllore con un opportuno segnale PWM quando il segnale del potenziometro supera il valore di soglia per il tempo minimo impostato. Il cavo, riscaldato per effetto Joule, riduce le proprie dimensioni, tirando i cavi collegati agli anelli ed eseguendo così la chiusura della mano. Per la successiva riapertura si attende un ulteriore segnale da parte potenziometro che aziona un altro cavo SMA adibito allo sblocco del sistema di frenatura, lasciando così libera la mano, che viene riportata in posizione di riposo dalle molle di ritorno.

1.3.2 prototipo ZERO

Tale prototipo è stato portato a termine da Mattia Tarsi, studente dell'università Alma Mater di Bologna, nell'elaborato di tesi: “GLOVE: progettazione di un dispositivo elettronico per l'abilitazione della mano di pazienti tetraplegici” [19].

Il prototipo ZERO aveva come obiettivo una maggiore efficienza, affidabilità e sicurezza rispetto alla tecnologia SMA mantenendo però semplicità e costi contenuti, caratteristiche non affini a sistemi pneumatici. Per queste motivazioni è stata scelta la tipologia di attuazione motorizzata, mantenendo la configurazione tendon-driven per la distribuzione delle forze.

1.3.2.1 Schema generale del sistema

Tale prototipo si è soffermato principalmente sulla progettazione dell'elettronica necessaria all'azionamento del sistema. Lo schema sotto riportato (Figura 1-23), mostra concettualmente tutte le funzionalità di interfaccia dell'elettronica da realizzare, nello specifico un qualsiasi diviso in tre parti:

- **Blocco di alimentazione:** contieni l'unità di batteria, il trasformatore, un dispositivo per visualizzare la tensione fornita dalla batteria, utile per la corretta alimentazione del motore e la visualizzazione della carica del dispositivo.
- **Blocco motore:** comprende il motore, azionato tramite un modulo full-bridge, un sensore di temperatura, uno di corrente e uno encoder per avere conoscenza di coppia e posizione del motore.
- **Blocco delle interfacce esterne:** comprende i moduli di interfaccia tra scheda e utente, tra cui segnalatore acustici, modulo Bluetooth, e moduli Input/Output, come pulsanti e schermi.

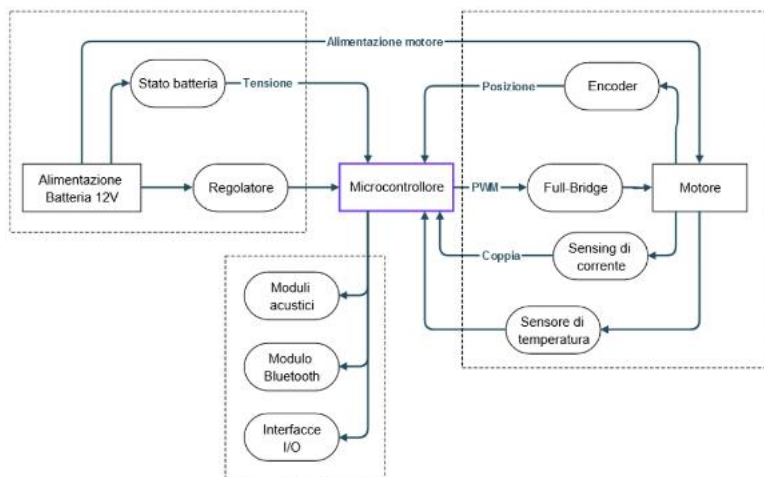


Figura 1-23 Schemi a blocchi concettuale del prototipo ZERO

1.3.2.2 Componenti

I blocchi sopra riportati, sono collegati tra loro tramite un'unità centrale di controllo, il microcontrollore, ossia il cervello del sistema, che elabora e genera segnali di comando da inviare ai vari moduli. Per questo progetto è stato utilizzato un microcontrollore *Nano 33 BLE Sense* della famiglia Arduino.

Per quanto riguarda il motore, è stato selezionato il WALFRONT alimentato a 12V DC (Direct Current). Per ottenere alta precisione e facilità di installazione e controllo; è dotato di encoder integrato per la misura di velocità e posizione. Tale motore è in grado di generare, attraverso una riduzione dei giri e un tamburo di raggio 1cm (centimetro), una coppia di circa 220N/cm (Newton per centimetro). Dalle specifiche del motore, si è optato per il modulo di controllo ANGEERK – Driver full-bridge HG7881 capace di gestire fino a due distinti motori DC. Al fine di misurare la corrente circolante tra gli avvolgimenti del motore, e segnalare eventuali assorbimenti eccessivi, è stato inserito il sensore di corrente ad effetto Hall IHAOSPACE ACS712 5A.

Nonostante la batteria sia la chiave la portabilità e l'autosufficienza, del dispositivo, ancora in fase ampia fase sperimentale, si è deciso di procedere con una più semplice, stabile e sicuro alimentazione via rete.

1.3.2.2 Prototipazione

A valle dello schema a blocchi concettuale, è possibile creare uno schema elettrico del prototipo finale (Figura 1-24), ovvero una rappresentazione semplificata del circuito reale mediante componenti a parametri concentrati.

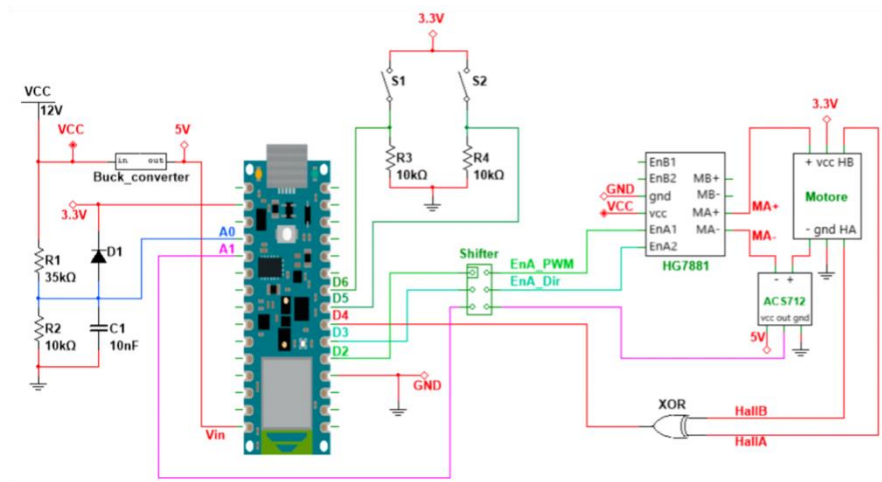


Figura 1-24 Schema elettrico del prototipo ZERO

Nello schema elettrico si possa identificare diversi moduli:

- **Convertitore Buck DC-DC:** metodo efficiente per convertire, da una sorgente di corrente continua, il livello di tensione disponibile in quello che si necessita.
- **Lettura della tensione di alimentazione:** tramite l'utilizzo di un diodo, necessario a limitare la tensione in ingresso ad un valore di soglia fissato, si possono prevenire picchi di corrente anomali e dannosi per il microcontrollore.
- **Shifter logico:** particolare partitore di tensione resistivo, impiegato per scalare la tensione di 5 V (volt), livello logico di alcuni moduli e sensori, al fine di raggiungere il livello logico del microcontrollore, corrispondente a 3.3V
- **Porta XOR:** prende in ingresso i segnali inquadatura provenienti dall' encoder e ne fornisce in uscita la combinazione che risulterà avere frequenza doppia. Aspetto cruciale in quanto gli spostamenti del cavo sulla puleggia sono talmente ridotti da necessitare un'elevata risoluzione sulla lettura della posizione.

- **Sistema pull up pull down:** viene inserita una resistenza tra il pin di ingresso e la tensione di riferimento al fine di forzare il livello logico basso. Ciò viene fatto per non lasciare il pin flottante, prevenendo così che una minima corrente generata dai campi elettrici presenti nell'ambiente generi un cambio di stato.

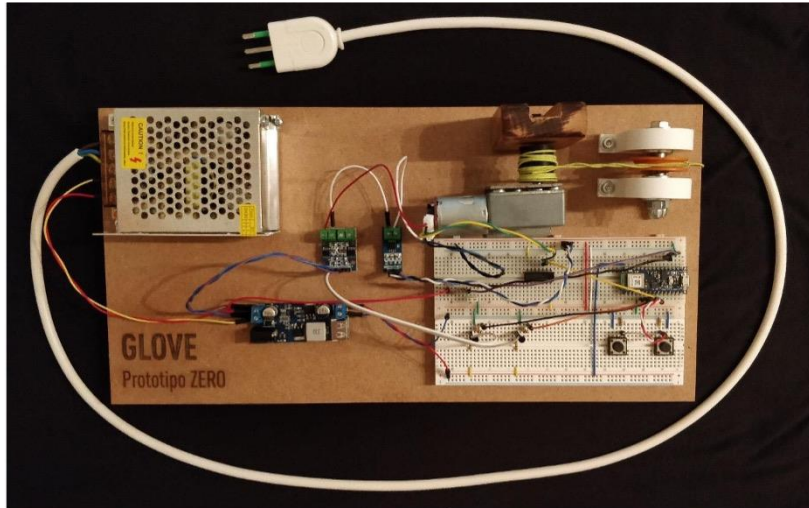


Figura 1-25 Immagine del prototipo ZERO

1.3.2.4 Funzionamento

Il sistema è gestito da due pulsanti: il primo, se premuto, è responsabile della rotazione del motore ad una velocità controllata, è stata preimpostata ad un valore necessario a tirare il tendine artificiale in tempi opportunamente scelti. Il secondo, a singola pressione, inverte il verso di rotazione del motore. Grazie ad un convertitore, il motore viene comandato attraverso due soli segnali che permettono di controllare contemporaneamente velocità e verso di rotazione.

1.3.3 Funzionamento

Il prototipo UNO [20] ha come obiettivo quello di perfezionare la struttura del prototipo ZERO e creare un primo modello di esoscheletro funzionante. Tramite attuazione motorizzata tendon-driven, viene quindi studiato il sistema in trasmissione della forza tra il motore e il guanto. Per semplicità e praticità dei test, l'esoscheletro, è stato montato sul modello semplificato di una mano, stampata 3D.

1.3.3.1 Schema generale del sistema

Il sistema può essere suddiviso in macro-blocchi distinguibili:

- **Blocco di alimentazione:** l'alimentazione viene fornita tramite un trasformatore a 12V (volt) DC collegato direttamente alla rete, le varie schede, richiedono un valore di tensione inferiore, sono alimentate da un convertitore DC-DC che riduce la tensione a 5V (Volt)
- **Blocco di attuazione elettrica:** comprendi i componenti elettronici della prima acerba prototipazione virgola che verranno poi sostituiti con i componenti finali illustrati nel capitolo 2.1
- **Blocco motore:** comprende il sistema composto da motore, riduttore ed encoder temporanei.
- **Blocco di attuazione meccanica:** comprende il sistema di trasmissione della forza costituito da cavi e pulegge.
- **Blocco di design:** comprende la struttura dell'esoscheletro composto da anelli dotati di passanti per il direccionamento dei cavi che permettono l'attuazione
- **Blocco di interfacce esterne:** comprendi i moduli di interfaccia tra schede utente, come sistema acustico per suoni di allarme e virgola il modulo Bluetooth per il collazionamento dei dati, utilizzato poi come segnale di comando
- **Blocco di sensoristica:** comprende il bracciale MYO Armband, capace di captare, prefigurare e di inviare tramite Bluetooth segnali EMG.



Figura 1-26 Schematizzazione del prototipo

1.3.3.2 Componenti

Il circuito elettrico di riferimento è concettualmente simile a quello del prototipo ZERO con l'aggiunta della parte di sensoristica e qualche modifica sull'elettronica di controllo del motore. L'Arduino Nano 33 BLE Sense è stato utilizzato per la ricezione e la gestione dei segnali di EMG provenienti, tramite Bluetooth, al bracciale MYO armband.

Per il controllo del motore, è stato impiegato un più semplice e comune Arduino Uno, connesso ad un Arduino Motor shield Rev 3 per l'amplificazione di potenza, tale scheda poi è stata impiegata anche per la soluzione finale di controllo, verrà più dettagliatamente illustrata nel capitolo successivo. Per quanto riguarda il motore, è stata utilizzata una vecchia versione, ormai fuori produzione, del Faulhaber Series 2233_S, con funzionamento a 12V (volt) DC, assorbimento di correnti nominale 0.43A, coppia di 5.9N*mm e velocità regime di circa 4600 rpm. Tale motore comprende già integrati un encoder a due canali in quadratura e il riduttore Faulhaber Series 22/5 con un rapporto di riduzione di 161:1. Il tendine artificiale viene simulato dal filo della Caperlan BRAID TRESSE UHMPE, costituito da un trecciato di 4 fibre di polietilene ad altissima densità, permettendo di resistere a frattura di un carico di 17.5 Kg. Per la stampa 3D dei giunti cedevoli, è stato utilizzato il materiale ULTRAFUSE TPU BASF shore 85 A, scelto per l'ottima flessibilità basse temperature, buone prestazioni di usura e ottimo comportamento di smorzamento.

1.3.3.3 Prototipazione

L'assemblaggio finale dei componenti richiede la progettazione di un sistema di trasmissione, un nuovo design per gli anelli, giunti elastici da collegare sul dorso dell'esoscheletro, l'integrazione dell'elettronica e qualche perfezionamento sulle connessioni.

Il modulo motore encoder riduttore deve essere fissato saldamente alla base di supporto ed essere allineato al sistema di trasmissione, ciò serve per prevenire la generazione di oscillazione e comportamenti indesiderati. Per ottenere ciò, sono state disegnate e aggiunte delle flange di sostegno per il motore ed un cuscinetto per l'albero rotante, entrambi opportunamente dimensionati e accuratamente fissati alla struttura. Nel sistema di trasmissione vengono impiegate due pulegge per il trasferimento della forza dal motore all'esoscheletro e due serie composte da tre pulegge, posizionate verticalmente, che fungono da guida per il percorso dei tendini. La mano di prova, stampata in 3D, è stata fissata alla struttura utilizzando un supporto a C.

I tendini artificiali scorrono ai lati del dito, superano ed aggirano il polpastrello in modo

da limitare l'ingombro e le interferenze con gli oggetti da afferrare. Negli anelli delle falangi e nel palmo sono presenti dei tubicini flessibili che guidano lo scorrimento dei fili lungo la traiettoria desiderata. Aspetto rilevante del prototipo è l'interfacciamento tra il modulo di sensoristica e quello di controllo del motore; la comunicazione avviene tramite due pin, uno comandato dal modulo di sensoristica che richiede la chiusura o apertura della mano, l'altro generato dal modulo di controllo del motore che restituisce lo stato dell'azionamento: in transitorio o a regime. La sensoristica, tramite elettrodi, compie l'elettromiografia volta alla rilevazione dei segnali, determinando quando è avvenuta l'attivazione del muscolo, che nel nostro caso corrisponde al cambio di stato dell'esoscheletro. Il dispositivo impiegato per l'EMG è il MYO Armband, con opportuno filtraggio dei sensori operato dall'Arduino Nano.

Il controllo del motore avviene tramite l'Arduino Uno e la Motor Shield Rev 3, utilizzata come amplificatore di potenza.

1.3.3.4 Funzionamento

All'accensione, il sistema si riporta una condizione nota grazie ad una fase di roaming, a quale termina con il contatto di un fine corsa fisico. Tale procedura permette al motore di trovarsi una posizione conosciuta, corrispondente alla mano aperta, qualsiasi fosse la posizione iniziale. Al generarsi dell'impulso muscolare dell'utilizzatore, il modulo di sensoristica fornisce il comando di chiusura dell'esoscheletro al motore viene fornito come target da raggiungere il numero di giri corrispondente alla configurazione di mano completamente chiusa qualsiasi comando ricevuto durante i transitori viene ignorato, una volta raggiunta la destinazione il sistema rimane in attesa di gestire il comando successivo.

Il fine corsa è ottenuto con due materiali e conduttori che ha raccontato formano un corto circuito e cambia uno stato di un pin dell'Arduino.

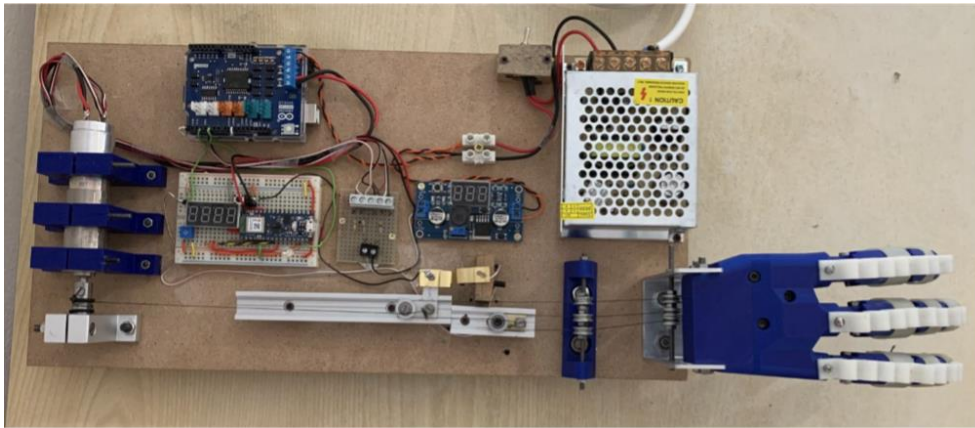


Figura 1-27 Prototipo UNO

1.3.4 Prototipo DUE

Il prototipo DUE [21] (Figura 1-28) ha come scopo il perfezionamento delle tecniche maturate dalla costruzione dei prototipi precedenti e l'obiettivo di rendere l'esoscheletro indossabile su una mano vera. Vengono mantenute nell'attuazione motorizzate a trasmissione della forza in modalità tendon driven virgola che viene però sostanzialmente rivisitata

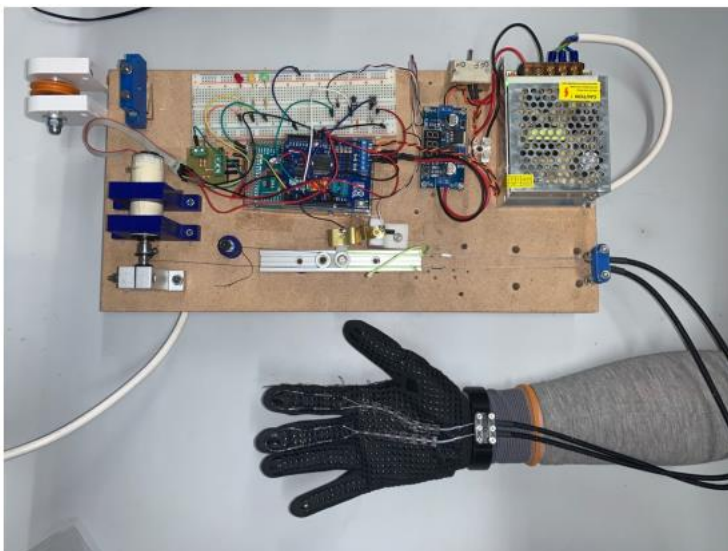


Figura 1-28 Struttura del prototipo DUE

1.3.4.1 Trasmissione delle forze

Viene accantonato il precedente sistema di trasmissione delle forze costituito da paranchi, pulegge e carrucole, vista l'incompatibilità con la portabilità e vestibilità richiesta con l'obiettivo finale.

Gli aspetti critici da risolvere, come fragilità dovuto al cavo scoperto e la mancata portabilità dei componenti fissati, hanno portato alla sperimentazione di soluzioni con cavi di tipologia *Bowden*. Tale nuova soluzione è in grado di fornire maggiore robustezza, semplicità implementativa, leggerezza, portabilità ed immediatezza nella trasmissione della forza.

Il *Bowden Cable*, anche noto come guaina, è una particolare tipologia di cavo flessibile impiegato nella trasmissione di forza meccanica o energia attraverso la movimentazione di un cavo, opportunamente rivestito.



Figura 1-29 Esempio di Bowden Cable con cavo interno libero alle estremità

1.3.4.2 Bowden Cable

Vista la possibilità di rinvio meccanico di un comando, il *Bowden cable*, viene sfruttato nelle applicazioni di freni di biciclette, moto e automobili. La particolare struttura della guaina è formata da molteplici strati (Figura 1-30), ognuno con una funzione caratteristica necessaria al corretto funzionamento del sistema complessivo. Il vero e proprio compito di trasmissione viene affidato ad un cavo metallico rigido, core del sistema, intorno al quale viene avvolta una guaina in PTFE, che consente di ridurre notevolmente l'attrito tra cavo e camicia esterna. Il terzo strato è costituito da una spirale realizzata con una molla a pacco, volta a conferire una forma costante al dispositivo ed a irrigidirlo, senza limitarne però la flessibilità grazie alle caratteristiche

proprie delle molle. È presente, infine, un rivestimento esterno di tipo plastico per isolare e proteggere il sistema da interferenze con l'ambiente esterno



Figura 1-30 Struttura interna di un Bowden Cable

1.3.4.3 Soluzione applicativa

Per il prototipo, in particolare, la guaina deve poter essere bloccata in prossimità del polso, sulla struttura dell'esoscheletro, e in prossimità del motore. Il cavo interno deve partire dall'anello della falangina distale, percorrere il dito e il palmo, entrare nella guaina, scorrere all'interno ed entrare nel blocco di distribuzione della forza. Le strutture di bloccaggio devono ospitare il Bowden cable in una sede che tenga l'estremità solidale con il telaio, permettendo invece al cavo interno di fuoriuscire e proseguire nella sua traiettoria. Per la connessione motore trasmissione è stata mantenuta la soluzione a paranco, facendo scorrere un carrellino su una rotaia è possibile tirare l'estremità del bowden cable e quindi procedere alla chiusura della mano.

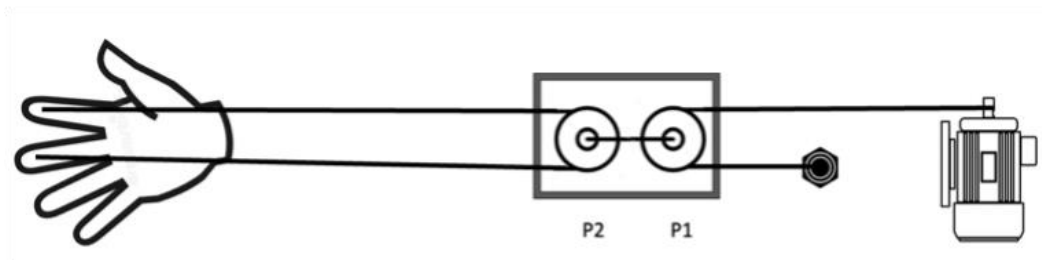


Figura 1-31 Configurazione del paranco

2 Sistema di attuazione elettrica

2.1 Progettazione del sistema

Dopo un'attenta analisi della letteratura esistente e della storia dei prototipi precedentemente realizzati, si giunge al cuore dell'argomento trattato in questo elaborato. Nei paragrafi seguenti, verranno esposti il funzionamento e la struttura del sistema, enumerando le componenti utilizzate e giustificandone la selezione.

2.1.1 Introduzione all'hardware del sistema

Come già accennato nel capitolo precedente, il sistema meccanico che accoglierà il motore e l'elettronica necessaria al suo corretto funzionamento è il prototipo DUE (Capitolo 1.3.4); tuttavia, gran parte della sperimentazione è stata condotta sul precedente prototipo UNO (Capitolo 1.3.3). Le prime prove eseguite su tale prototipo hanno rappresentato un elemento cruciale per la raccolta di dati, informazioni ed esperienze, fondamentali nella scelta dei componenti da utilizzare per il progetto. Per una comprensione più chiara della composizione del sistema finale, è riportato uno schema concettuale (Figura 2-1) che include tutte le funzionalità di interfaccia presenti. Lo schema può essere suddiviso in cinque sottoblocchi:

- **Blocco di alimentazione:** Comprende il trasformatore utilizzato per abbassare la tensione di rete a 12V DC, necessaria per il funzionamento del motore. È richiesta un'ulteriore riduzione della tensione a 9V DC per alimentare i microcontrollori, poiché i regolatori di tensione integrati presenti sulle schede rischiano di surriscaldarsi eccessivamente se sottoposti a tensioni elevate.
- **Blocco di rilevazione comandi:** Include il bracciale MYO e l'elettronica necessaria per il campionamento e l'elaborazione dei segnali EMG rilevati. Tale blocco comunica con il microcontrollore principale, fornendo i comandi per l'apertura e la chiusura della mano.

- **Blocco di amplificazione di potenza:** Include il modulo full Bridge, comandato tramite la tecnica Pulse Width Modulation (PWM) dal microcontrollore, e un modulo di misura della corrente, composto da una resistenza di sensing e un amplificatore.
- **Blocco di lettura encoder:** Comprende l'encoder integrato al motore e un semplice circuito partitore di tensione per rendere possibile la lettura dei due canali in quadratura forniti a un livello logico superiore a quello utilizzato dal microcontrollore.
- **Blocco del motore:** Include il motore DC, il moto riduttore integrato e la meccanica necessaria per il collegamento con il prototipo DUE.

Tutti i blocchi sopra menzionati comunicano tra loro tramite l'unità centrale di controllo, il **microcontrollore**, ossia il fulcro dell'elettronica che elabora i segnali ricevuti e genera quelli necessari al comando e al corretto funzionamento dei moduli.

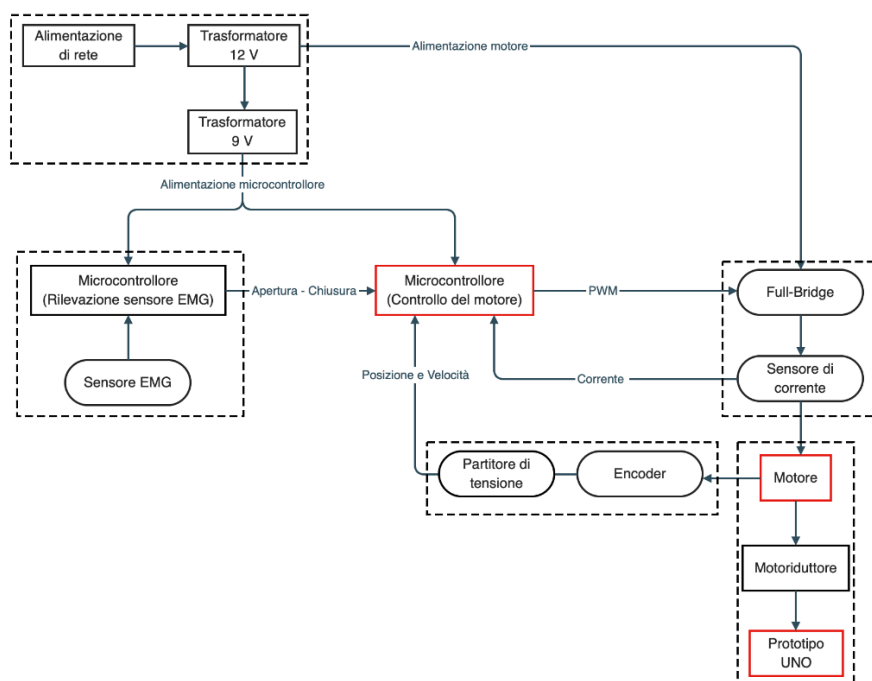


Figura 2-1 Schema a blocchi concettuale del sistema da realizzare

2.1.2 Motore

Attualmente, i motori elettrici trovano impiego in pressoché tutti i settori industriali. Per comprendere l'importanza assunta dai motori elettrici nel corso del tempo, secondo MyDirectIndustry [22], essi rappresentano il 46% del consumo globale di elettricità e vengono impiegati per azionare macchine di ogni tipo. A seconda delle caratteristiche tecnologiche, un motore può eseguire diversi tipi di movimentazione: spostamenti rapidi, precisi, continui, a velocità costante, o persino compiere complesse traiettorie. Le diverse tipologie di movimento vengono soddisfatte da tecnologie costruttive differenti.

2.1.2.1 Principio di funzionamento

Un motore elettrico è una macchina che converte l'energia elettrica in meccanica sfruttando le proprietà del campo magnetico. Esistono diverse tipologie di motori elettrici:

- **Motori in Corrente Continua (DC) o brushed**
- **Motori in Corrente Alternata (AC) o brushless**
- **Motori stepper** (particolare tipologia di motori brushless)

Il funzionamento varia da tipologia a tipologia, ma la composizione generale è comune, si possono dunque individuare due parti fondamentali: lo statore, parte statica, tipicamente fissata e il rotore, parte che viene posta in rotazione generando l'energia meccanica.

Una delle due parti, a seconda della tipologia, è composta da magneti permanenti o campi magnetici generati da un circuito di eccitazione esterno. Quest'ultimo, se presente, permette la variazione del flusso magnetico a scapito di perdite per effetto Joule, causate dal passaggio della corrente negli avvolgimenti. La restante parte è costituita da bobine di rame smaltato, che generano un campo magnetico quando attraversate da corrente.

Il movimento è reso possibile dalla contrapposizione dei campi magnetici di statore e rotore; affinché vi sia movimento, il campo magnetico sugli avvolgimenti deve variare continuamente. Nel caso specifico dei motori elettrici in corrente continua, non

essendo presente un'inversione autonoma della polarità, il campo viene fatto variare grazie a un sistema di "spazzole", che permettono la continua inversione della polarità della tensione applicata alle bobine. Le criticità di questa tecnica includono una maggiore usura dei componenti e la generazione di brusche variazioni di corrente che si manifestano in scintille.

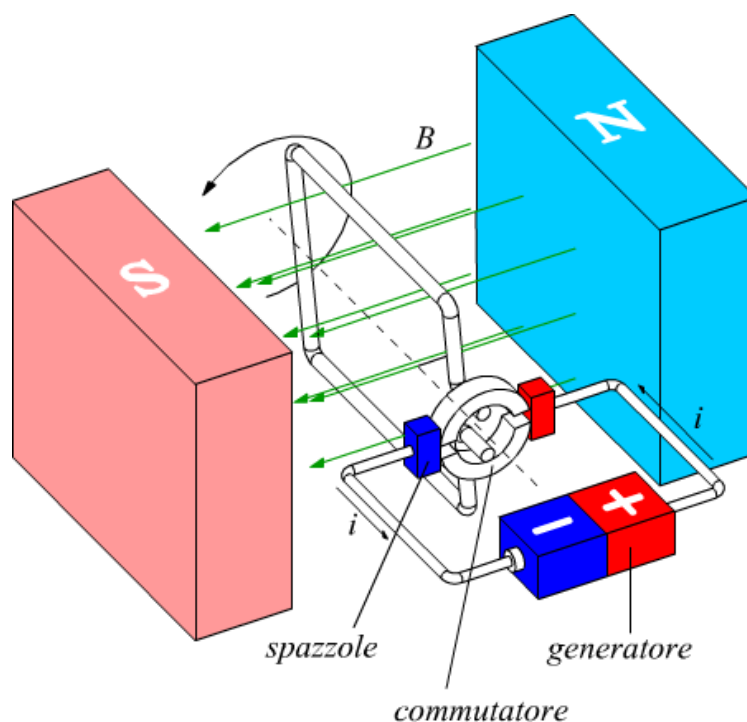


Figura 2-2 Esempio concettuale di un motore elettrico DC

Ogni macchina elettromeccanica può essere descritta attraverso un circuito elettrico equivalente. Nel caso di una macchina operante in corrente continua, il circuito di armatura può essere rappresentato tramite una resistenza r_a e un'induttanza L_a equivalente. Inoltre, qualora il circuito magnetico operi in condizioni di linearità, ossia non in stato di saturazione, l'induttanza permane costante al variare della corrente di armatura. All'interno del circuito di armatura, si manifesta altresì una forza elettromotrice indotta negli avvolgimenti quando il rotore è in movimento, tensione questa indicata con e_a o V_m . Infine, nel caso in cui non si faccia uso di motori a magneti permanenti, il circuito di eccitazione può essere descritto mediante una resistenza r_e ed un coefficiente di autoinduzione L_e , definito come il rapporto tra il flusso di eccitazione concatenato con l'eccitazione stessa e la corrente di eccitazione ($N_e \varphi_e = L_e i_e$).

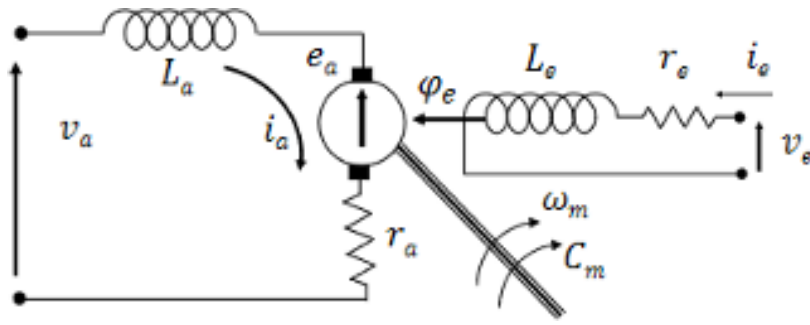


Figura 2-3 Circuito elettrico equivalente di un motore DC

Viene ora riportato il sistema di equazioni differenziali che descrive la macchina elettrica in corrente continua:

- **Dinamica elettrica:**

$$V_A(t) = R_A i_A + L_A \frac{d i_A(t)}{d t} + V_M(t) \quad (2-1)$$

Dove $V_M(t)$ è la forza contro elettromotrice (BEMF)

- **Costante elettro-meccanica**

$$C_M(t) = K_t i_A(t) \quad (2-2)$$

Dove $C_M(t)$ è la coppia generata dal motore e K_t la costante elettro-meccanica

- **Dinamica meccanica:**

$$(J + J_c) \frac{d \omega_m(t)}{d t} = C_m(t) - b C_m(t) - C_R \quad (2-3)$$

Dove $\omega_m(t)$ è la velocità di rotazione del motore, J è il momento di inerzia e b il coefficiente di frizione complessivo. J_c è l'inerzia della carrucola e C_R la coppia resistente che la mano del paziente impone sul sistema

- **Costante meccanico-elettrica**

$$V_m(t) = K_v \omega_m(t) \quad (2-4)$$

Dove $K_v = K_t$ è la costante di tensione del motore

2.1.2.2 Motore utilizzato

Per il progetto è stato scelto il DC-Micromotors Series 2232 012 SR dell'azienda Faulhaber. La differenza sostanziale tra i micromotori DC FAULHABER [23] e i più comuni motori DC risiede nella composizione del rotore. Esso, infatti, non ha un'anima in ferro, ma è costituita da un avvolgimento autoportante in rame avvolto obliquo (Figura 2-4). Tale modifica permette al rotore di essere più leggero e avere così un momento di inerzia estremamente ridotto, oltre al fatto di poter ruotare senza essere sottoposto a *cogging*, forza parassita dovuto alla presenza di “denti” di materiale ferromagnetico che rende meno fluido il movimento. Il risultato è l'ottenimento di un'eccelsa dinamica di rotazione in un formato dalle dimensioni contenute.

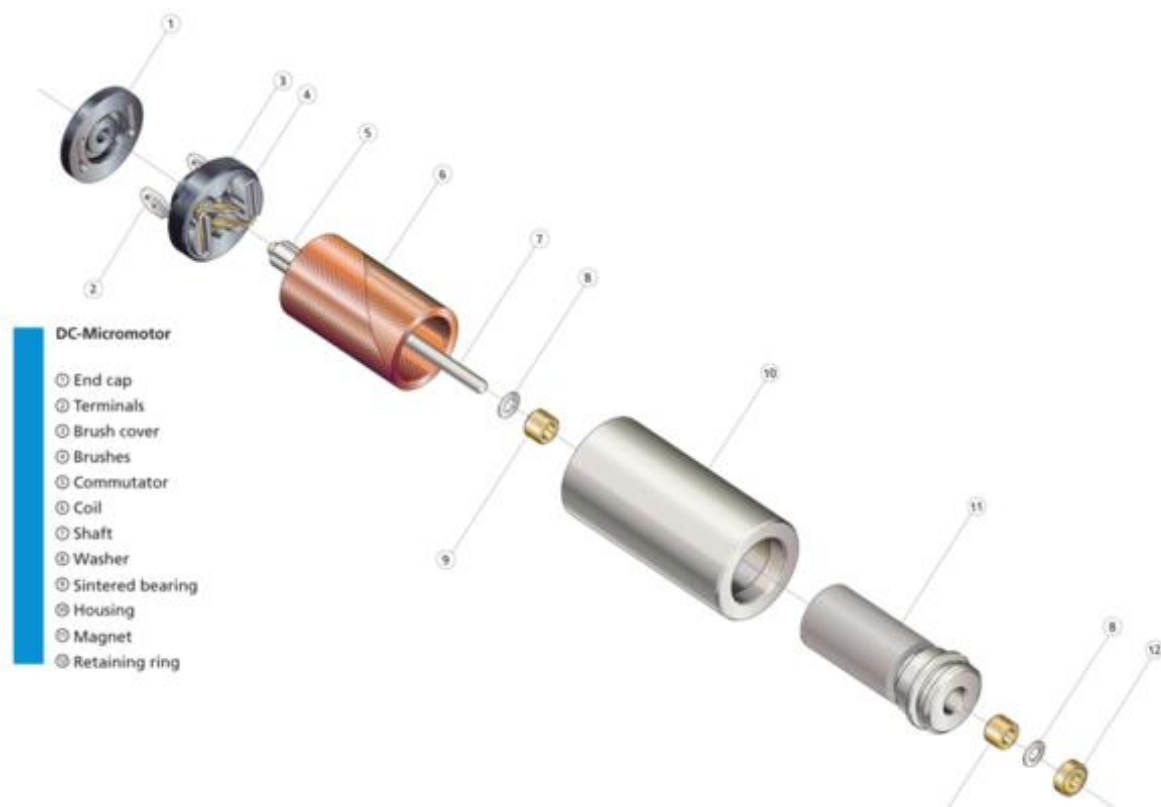


Figura 2-4 Rappresentazione del componimento della serie Micromotors di Faulhaber

In particolare, il modello 012 SR, ha una tensione di alimentazione di 12V (Volt) DC con una velocità a vuoto di 7100 rpm (rotazioni per minuto), ed un limite di corrente di 0.67 A (Ampere), arrivando a generare una coppia massima di 10 mNm (milli Newton per metro)

2.1.2.3 Riduttore utilizzato

Vista l'applicazione di interesse richiedere coppia elevata a velocità ridotte, è necessario ridurre notevolmente i giri al minuto compiuti dal motore. Per fare ciò è stato scelto un riduttore compatibile, in particolare, il Planetary Gearheads Series 22E, con un rapporto di riduzione 152:1, portando così il sistema motoriduttore ad avere una velocità a vuoto di 46.71 rpm una coppia generata di 1.5 Nm. Dividendo tale valore per il raggio della puleggia utilizzata, corrispondente a 0.33 cm, si ottiene la forza generabile, corrispondente a 455N, decisamente superiori ai 55N sufficienti ad avere una corretta forza di presa, come illustrato nel paragrafo 1.1.2.2.

2.1.3 Encoder

In un sistema robotico, i motori sono utilizzati per la movimentazione di parti mobili, siano essi ruote di locomozione, o giunti di bracci meccanici. Essendo l'output finale del motore la rotazione del suo asse, è essenziale avere conoscenza di tale spostamento se lo si vuole controllare. Per poter realizzare tale misurazione, si collega (calettatura) all'asse del motore un dispositivo denominato encoder, il quale traduce la posizione angolare dell'asse in un valore numerico.

Esistono diverse tipologie di encoder, che si differenziano per metodo di rilevazione:

- **Resistivi:** Hanno un comportamento molto simile ad un potenziometro radiale, ovvero sono partitori di tensione, in cui la resistenza varia con la posizione del motore.
- **Ottici:** l'elemento principale è un disco, calettato al motore, con fori equamente distanziati. In corrispondenza dei fori, è presente un foto-accoppiatore, ovvero una coppia led-fotodiodo che rileva il passaggio di un foro grazie al raggio di luce che filtra attraverso.
- **Magnetici:** viene utilizzato un sistema di rilevazione dei segnali basato sulla variazione del flusso magnetico, generato dalla rotazione di un magnete, rispetto a un punto fisso in cui è situato il sensore. Sono particolarmente adatti all'applicazione in ambienti gravosi che richiedono un'elevata robustezza,

velocità e resistenza termica, garantendo al tempo stesso un'affidabilità nella generazione dei segnali.

Tralasciando il caso resistivo, in cui viene fornito in uscita un valore analogico, proporzionale alla posizione del rotore, sono di maggior interesse gli encoder Ottici e Magnetici in cui l'uscita è in formato digitale. Per questa tipologia di sensori, la rotazione del disco genera un treno di impulsi la cui frequenza è proporzionale alla velocità di rotazione. Questi tipi di encoder permettono di determinare:

- La **posizione angolare** contando i fronti di salita del segnale denominati "pulse"
- La **velocità** conteggiando i pulse rilevati in un determinato intervallo di tempo noto.

Tramite un singolo treno di impulsi non è però possibile avere conoscenza del verso di rotazione del motore, per ovviare a questa problematica si utilizza un secondo sensore, sfasato tipicamente di 90 gradi (Figura 2-5). Così facendo, per esempio, al fronte di salita del canale A è possibile determinare il verso di rotazione osservando lo stato del canale B.
(B=0 → senso orario (CW), B=1 → senso antiorario (CCW))

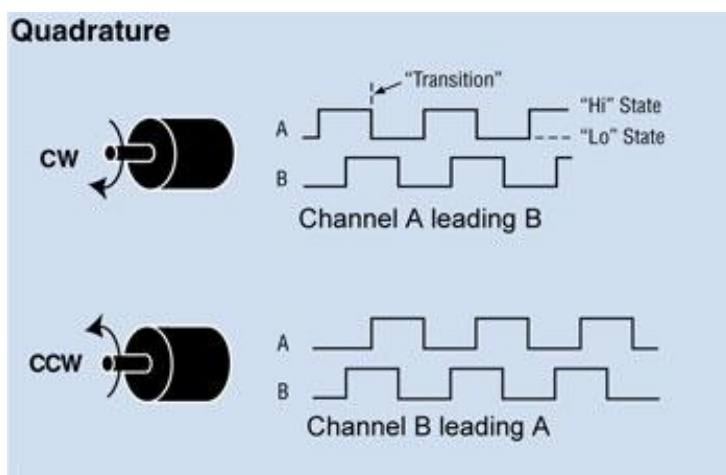


Figura 2-5 Treno di impulsi di un encoder in quadratura

2.1.3.1 Encoder utilizzato

Fra i vari encoder compatibili con il motore, è stato scelto il *magnetic Encoder, digital outputs, 2 channels Series IE2-1024*, con una risoluzione di 1024 PPR (*Pulse Per Revolution*) e una frequenza massima di 300 KHz (Kilo Hertz). Per avere una migliore precisione ed evitare di incappare in problemi di derive dovuti a vibrazioni indesiderate, viene conteggiato il passaggio di un “pulse” ad ogni fronte (salita e discesa) e ciascuno dei due canali, così facendo si ottiene una risoluzione quattro volte maggiore, corrispondente a 4096 fronti totali per ogni giro compiuto.

2.1.4 Microcontrollore

Per semplicità di utilizzo, reperibilità, familiarità con il linguaggio di programmazione e compatibilità con il sistema responsabile del campionamento dei segnali EMG, è stato scelto di utilizzare un prodotto della famiglia Arduino.

2.1.4.1 Introduzione ad Arduino

Arduino è un’azienda italiana che costruisce schede elettroniche open source dotate di microcontrollore e software. È stata ideata e sviluppata nel 2005 da alcuni membri dell’Interaction Design Institute di Ivrea come strumento poco costoso, inferiore ai 50\$, per la prototipazione rapide e per scopi hobbistici, didattici e professionali. La scheda elettronica si basa su un circuito stampato integrato, costituito da un microcontrollore con dei pin connessi alle porte I/O, un regolatore di tensione, e quando necessario, un’interfaccia USB che permette l’interazione con il computer, utilizzato per la programmazione. All’hardware viene affiancato un ambiente di sviluppo integrato (IDE) multiplatforma realizzato in Java, scaricabile per Linux, Apple Machintosh e Windows

2.1.4.2 Arduino DUE

Sul prototipo UNO era stata utilizzata una semplice e comune scheda Arduino UNO, questo microcontrollore è risultato troppo poco potente in termini prestazionali e quindi è stato necessario cambiare piattaforma, rivolgendosi al microcontrollore Arduino DUE. A differenza di modelli più semplici, questa scheda gode di una potenza di calcolo

molto superiore, garantita dal processore Atmel SAM3X8E ARM Cortex-M3, ed è dotata di molte più porte I/O e altri componenti hardware già integrati. Arduino DUE ha un'architettura a 32 bit ed esegue i calcoli ad una frequenza di 84 MHz, dispone di un ampio numero di pin digitali per l'esattezza 54, che si possono utilizzare come ingressi e uscite, di cui 12 possono essere utilizzati come uscite PWM. Inoltre, questa scheda ha 12 ingressi e due uscite analogiche, le quali sono gestite da Digital to Analog Converter (DAC) a 12 bit, che è la stessa risoluzione utilizzata anche dalla Analog to Digital Converter (ADC) connesso ai pin di ingresso analogico. Questo nuovo tipo di architettura segue la filosofia "Low Power" e proprio per questo riduce notevolmente i consumi della scheda, infatti l'intero sistema opera ad una tensione di 3,3V (Volt). Inoltre, viene utilizzato il pin IOREF, che presenta la tensione di riferimento per il microcontrollore, che ribadisco in questo caso essere pari a 3,3V (Volt). In questo modo si possono utilizzare le diffuse shield presenti sul mercato. Le Shield utilizzano tale tensione come indicazione tassativa del voltaggio di funzionamento di comunicazione con il microcontrollore della scheda Arduino a cui si collegano.

Quindi 3,3V è il livello di tensione a cui il processore SAM3X si aspetta di trovare sulle porte come livello logico "1", allo stesso modo le porte ADC e DAC operano da zero a 3,3V. Un'altra logica di funzionamento è quella 5V, nel caso di presenza di un segnale a questo livello di voltaggio si rischierebbe il danneggiamento della scheda per cui bisogna prestare molta attenzione a questa criticità. Le porte digitali in uscita non sono tutte uguali, a seconda del tipo possono fornire 3 o 15 milliampere, per un totale complessivo di 130 milliampere massimi, oppure possono ricevere in caso di porte sink, una corrente di 6 o 9 mA (milliampere).

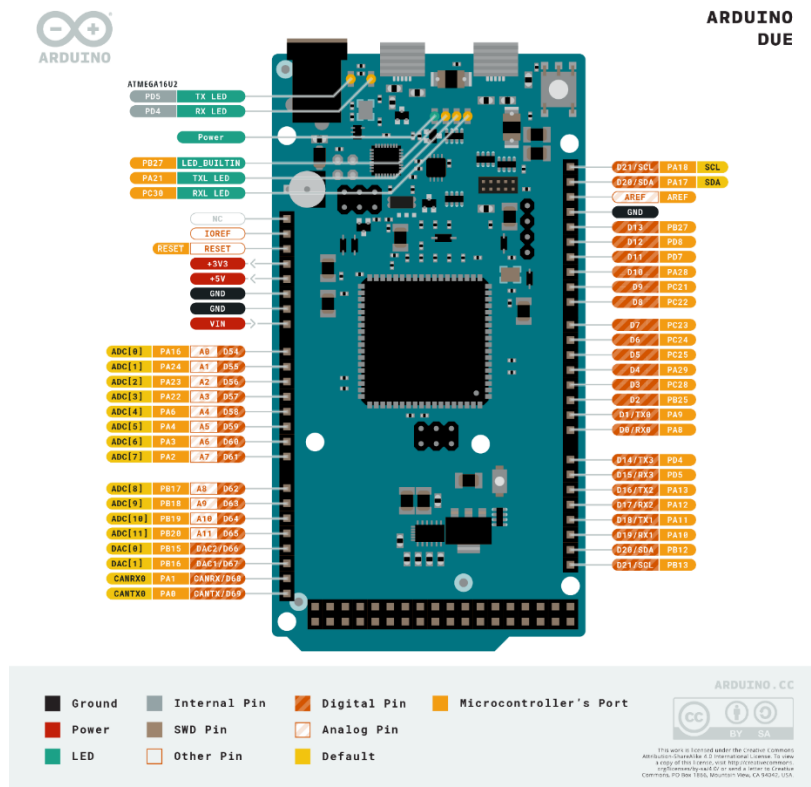


Figura 2-6 Pin-out della scheda Arduino DUE

Il principale motivo di migrazione verso la scheda Arduino due è dovuto alla presenza di due contatori hardware in quadratura già integrati sulla scheda (Figura 2-7), poiché nel caso vengano abilitati, possono essere impostati per funzionare come contatori dei pulse dell'encoder o per restituire direttamente il valore di velocità di rotazione del motore.

Il setup motore ed coder fornisce dati ad una frequenza troppo elevata per essere gestita via software, anche da microcontrollori molto performanti. Per questo si è deciso di utilizzare tale componente. L'encoder ha una risoluzione di 1024 PPR, configurato in modo da aver una precisione moltiplicata per quattro quindi in sostanza ad ogni giro del motore vengono generati $1024 * 4 = 4096$ pulse.

Il motore, a vuoto, ha una velocità di 7100 rpm (rotazioni per minuto), corrispondente a $7100/60=118.33$ rps (Rotazioni per secondo). A questa velocità, vengono generati $118.33 * 4096 = 484679.68$ pulse al secondo che sono all'incirca 500KHz (kilohertz), che è una frequenza troppo elevata per essere correttamente campionata è gestita via software, anche tramite l'utilizzo di procedure di interrupt ottimizzate.



è possibile ricavare il valore medio di uscita, che sarà proporzionale al duty-cycle del segnale di controllo.

Essendo il motore un sistema tipicamente induttivo, è come se venisse letta in ingresso una tensione analogica, facilmente controllabile dalle uscite digitali del microcontrollore.

Materiale di riferimento “principi di elettronica industriale” [24]

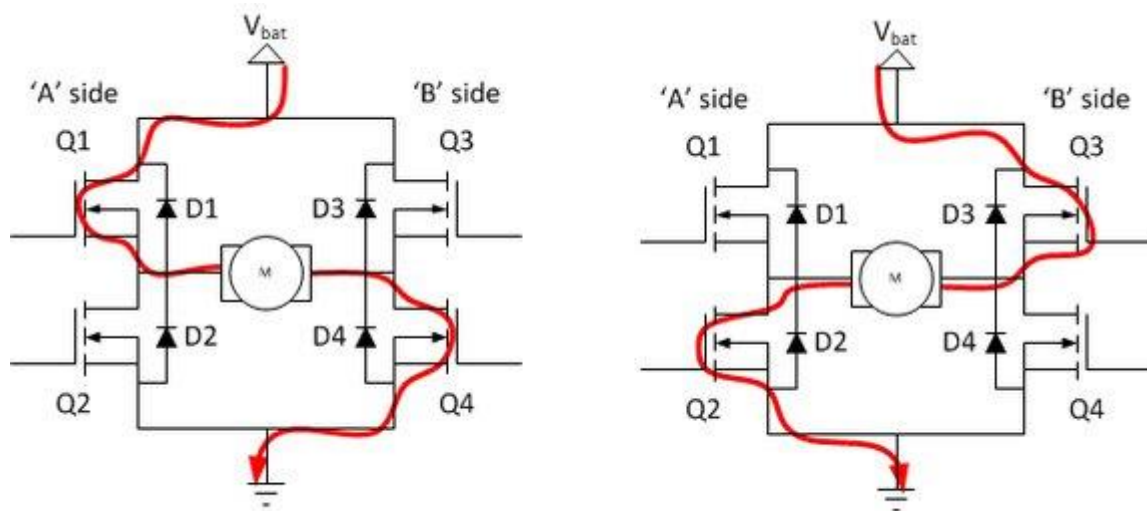


Figura 2-8 Schema circuitale di un full h-bridge

2.1.5.1 Arduino Motor Shield Rev 3

L'Arduino Motor Shield Rev 3 (Figura 2-9) è una scheda costruita da Arduino per l'amplificazione di potenza, compatibile con la scheda Arduino DUE e facile da utilizzare.

Questa board si basa sul chip L298 della STMicroelectronics, un *driver dual full-Bridge* progettato per pilotare carichi induttivi come: relè, solenoidi, motori DC e motori passo passo. Questa shield accoppiata insieme a una scheda Arduino compatibile, permette di controllare fino a due motori in corrente continua, potendone controllare velocità e posizione in modo indipendente.

All'interno di questa scheda sono presenti due ponti H integrati, che supportano elevati voltaggi, fino a 46V, e correnti fino a 2A per ponte, con la possibilità di essere pilotati in *Transistor-transistor logic* (TTL).

È possibile misurare la caduta di tensione generata dal passaggio della corrente che attraversa il motore, attraverso la presenza di una resistenza di shunt. Per ricavare il valore di corrente che il motore sta assorbendo è possibile utilizzare la legge di Ohm

e un'opportuna amplificazione. Il pin IOREF (funzionalità precedentemente illustrata) garantisce la compatibilità con l'Arduino DUE, sebbene la shield non sia stata appositamente progettata per questo tipo di scheda. Inoltre, il valore logico massimo consentito è di 3.3V che viene limitato superiormente dal controllore, che è l'unico valore di uscita generato dalla board ovvero l'unico valore di uscita generato dalla scheda. Il treno di impulsi generato dalla PWM ad un valore alto di 3.3V, è sufficiente a portare in conduzione i MOSFET del ponte, che quindi consentono il corretto funzionamento del sistema.

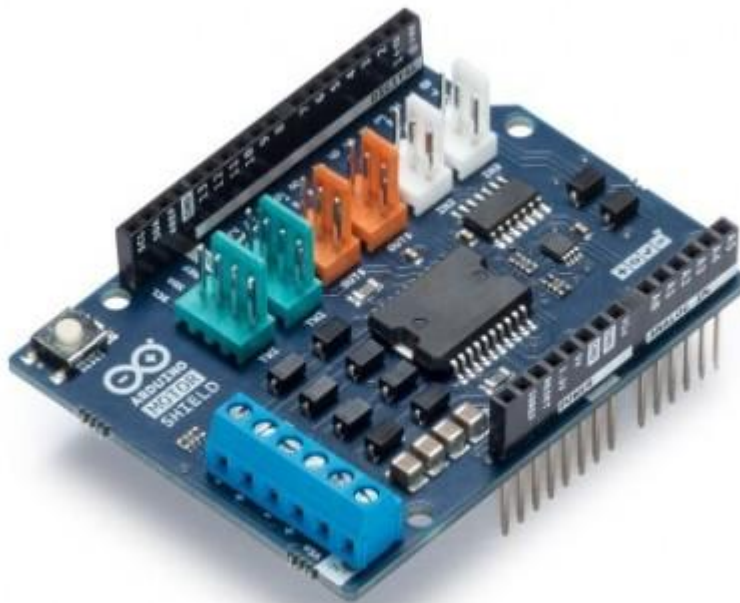


Figura 2-9 Arduino Motor Shield Rev 3

3 Progettazione dell'algoritmo di controllo

Dopo aver illustrato schematicamente il sistema e i suoi componenti viene descritta la tecnica di controllo utilizzata. Essa è l'elemento cardine del controllo poiché, permette al motore, e quindi all' esoscheletro e di conseguenza alla mano, il corretto inseguimento dei riferimenti imposti. La tecnica utilizzata è quella classica per un motore DC dove si implementa una serie di controllori in cascata rispettivamente: un controllore di tipo P per la posizione, un controllore di tipo PI per la velocità ed infine controllore di tipo PI per la corrente o coppia. Infine, viene utilizzata una accortezza ovvero l'anti Wind up in modo da evitare il danneggiamento dei componenti.

3.2.1 Modello di un motore DC

Le equazioni che descrivono il modello di un motore di c sono già state ricavate nel capitolo 2.1. 2.1; per poterle utilizzare implementare nel dominio elettronico è conveniente, vista la linearità delle equazioni, trasportare le equazioni dal dominio del tempo al dominio della trasformata di Laplace (s) ($F(s) = L[F(t)] = \int_0^\infty e^{-st} f(t) dt$), arrivando così ad ottenere il seguente sistema:

- **Dinamica elettrica**

$$V_A(s) = R_A I_A(s) + sL_A I_A(s) + V_M(s) \quad (3-1)$$

- **Costante elettro-meccanica:**

$$C_m(t) = K_m I_A(s) \quad (3-2)$$

- **Dinamica meccanica**

$$s(J + J_c)\omega_m(s) = C_m(s) - b\omega_m(s) - C_r \quad (3-4)$$

- **Costante meccanico-elettrica:**

$$V_M(s) = K_m \omega_m(s) \quad (3-5)$$

Da questo sistema di equazioni è possibile ottenere, dopo qualche calcolo matematico, le due diverse dinamiche del sistema motore DC, espresse dalle seguenti Funzioni Di Trasferimento (FDT):

- **Dinamica elettrica**

$$I_A(s) = \frac{V_A(s) - V_M(s)}{R_A + sL_A}$$

- **Dinamica meccanica**

$$\omega_m(s) = \frac{C_m(s) - C_r}{b + s * (J + J_c)} \quad (3-7)$$

Le FDT ricavate qua sopra possono essere scritte dal seguente diagramma a blocchi in retroazione che è una tipologia di rappresentazione fondamentale e molto utilizzata nel mondo dei sistemi dei controlli automatici

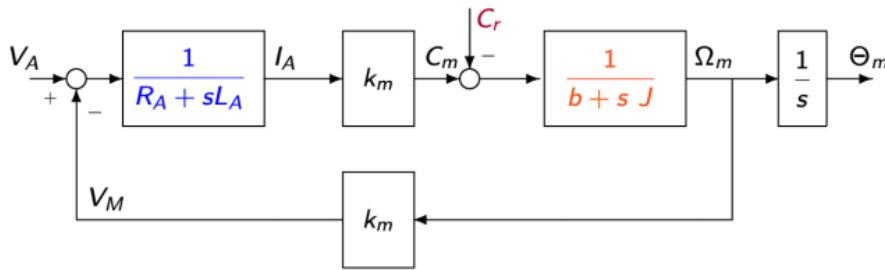


Figura 3-10 Schema a blocchi di un motore DC

Considerando la FDT dell'intero sistema si ottiene:

$$\omega_m(s) = \frac{K_m}{(b + s * (J + J_c) - C_r)(R_A + sL_A) + K_m^2} V_A(s) = G_\omega(s) V_A(s) \quad (3-8)$$

Supponendo che il coefficiente d'attrito b sia trascurabile ($b = 6.33e^{-5} \approx 0$)

$$G_\omega(s) = \frac{\omega_m(s)}{V_A(s)} = \frac{K_m}{(s * (J + J_c) - C_r)(R_A + sL_A) + K_m^2} \quad (3-9)$$

Da quest'ultima formula otteniamo due funzioni di trasferimento, una per la dinamica elettrica e una per la dinamica meccanica, rispettivamente

$$G_{inner} = \frac{K_t}{L_a + sR_m} \quad (3-10)$$

$$G_{outer} = \frac{1}{(J_c + J) * s + b} \quad (3-11)$$

3.3 Controllo in cascata di un motore DC

Il sistema in catena aperta, in generale, non è stabile. Si rende quindi necessario l'utilizzo di schemi di controllo più complessi e, come molto spesso accade nel campo dei controlli automatici, la risposta a questi problemi è la retroazione. Esistono tanti tipi di metodi diversi per sviluppare i controllori che andranno a stabilizzare il sistema del motore DC come, ad esempio, si può sviluppare il progetto del controllore tramite sintesi in frequenza, luogo delle radici oppure tramite controllori PID. In questo progetto ho scelto di utilizzare controllori PID poiché sono i più semplici ed intuitivi da progettare.

3.3.1 Controllo in cascata di un motore DC

In applicazione ad alte prestazioni, il controllo del motore DC può essere naturalmente implementato con una tecnica di controlli avanzati ovvero con una struttura in cascata composta da tre loop (o anelli in retroazione) (Figura 3-2). Per la progettazione si parte dal loop più interno fino ad arrivare al più esterno. Partendo dall'interno si trova:

- Loop di controllo di **corrente** (o controllo di **coppia**)
- Loop di controllo di **velocità**
- Loop di controllo di **posizione**

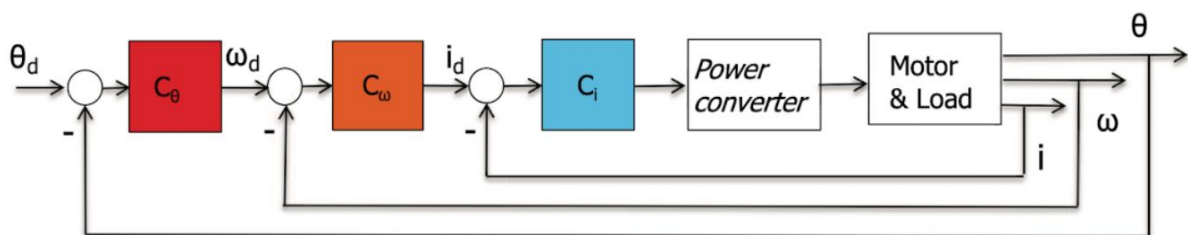


Figura 3-11 Diagramma a blocchi del controllo in cascata

3.3.1.1 Controllo in cascata di un motore DC

La forza controelettromotrice, o **back electromotive force (back-EMF)**, è una tensione che si genera all'interno del motore DC quando esso è in movimento. Questo fenomeno è una conseguenza della **Legge di Faraday** sull'induzione elettromagnetica: ogni volta che un conduttore (in questo caso l'armatura del motore) si muove in un campo magnetico, si induce una tensione che si oppone al movimento stesso.

L'espressione matematica della forza controelettromotrice per un motore DC può essere formulata come:

$$V_{EMF} = K_e \omega$$

Dove K_e è la costante di velocità del motore. Questa tensione si oppone al voltaggio applicato per azionare il motore, riducendo la corrente e, di conseguenza, la coppia generata. Il fenomeno della EMF, quindi, influenza negativamente il comportamento dinamico del motore, soprattutto durante le variazioni di velocità, complicando il controllo accurato della posizione. Nel controllo di un motore DC, la presenza della EMF può causare: errori nel raggiungimento della posizione desiderata, può influenzare la retroazione soprattutto in transitori rapidi, la reazione del motore ai segnali di controllo può risultare attenuata a causa dell'opposizione della EMF, rallentando la risposta complessiva. La compensazione della CEM è essenziale per migliorare le prestazioni del sistema di controllo del motore DC, garantendo un comportamento più stabile e preciso. La compensazione può essere fatta attraverso diverse tecniche, che cercano di neutralizzare l'effetto della CEM agendo sul controllo della corrente o della tensione del motore. Una delle tecniche più comuni è includere un modello della forza controelettromotrice direttamente nel controllore. In questo approccio, si tiene conto della velocità del motore (che genera la CEM) per prevedere e compensare la tensione indotta.

Il controllore può essere arricchito con una compensazione basata su una stima della CEM. Il segnale di controllo V_{con} può essere corretto includendo un termine che neutralizza l'effetto della CEM, come segue:

$$V = V_{con} - K_e \omega$$

In questo caso, V è la tensione calcolata dal controllore di posizione, e $K_e \omega$ è il termine di compensazione della CEM basato sulla velocità stimata o misurata. Nell'immagine sottostante la compensazione della EMF è il segnale evidenziato in blu

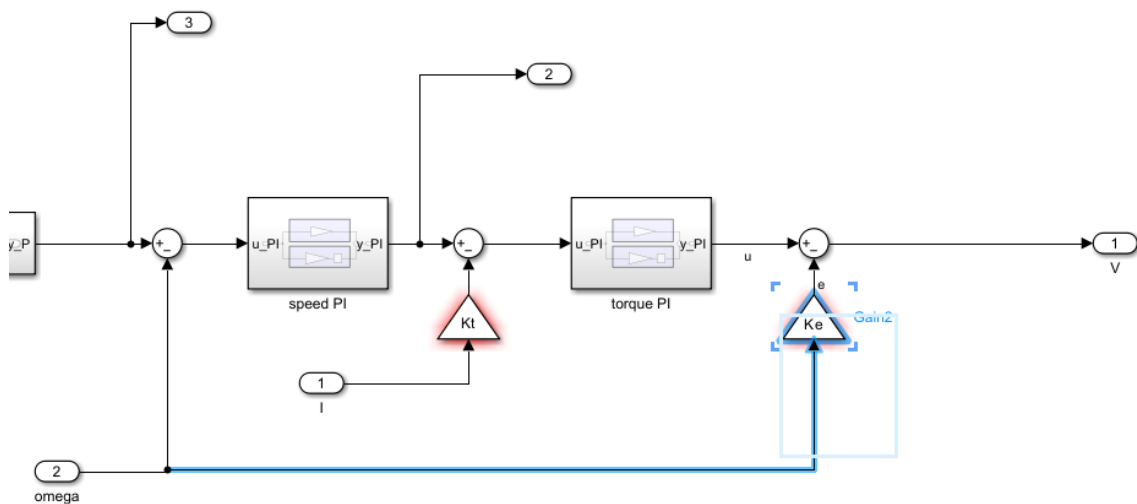


Figura 3-12 BACK EMF

3.3.2 Regolatori PID

Prima di sviluppare i calcoli e la teoria di ciascun regolatore è necessario presentare brevemente come funzionano i regolatori PID.

La sintesi di un controllore PID è un metodo col quale si costruisce la funzione di trasferimento del controllore in un sistema in retroazione in modo che il sistema complessivo abbia le prestazioni volute. Il controllore pid è una combinazione di tre funzioni di controllo, una di tipo Proporzionale, una Integrale e una Differenziale. Le tre parti di questo tipo di controllore intervengono in maniera diversa sull'input del PID ovvero l'errore $e(t)$. Lo schema generale è quello riportato nella (Figura 3-12).

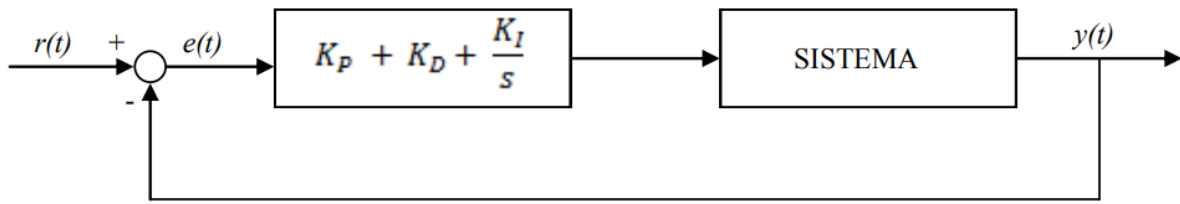


Figura 3-13

Nello schema riportato $e(t) = r(t) - y(t)$, dove $r(t)$ è il segnale di riferimento (ovvero l'andamento che dovrebbe assumere l'uscita), $e(t)$ l'errore di inseguimento e $y(t)$ è chiaramente l'uscita. Il segnale $e(t)$ rappresenta quindi la differenza tra l'andamento che l'uscita dovrebbe avere e quello che effettivamente ha nella realtà: essendo un errore l'obiettivo, quindi, è quello di farlo tendere a zero in qualche maniera. La forma generale della funzione di trasferimento del controllore PID è dunque

(3-12)

$$C(s) = K_P + K_D s + \frac{K_I}{s}$$

L'azione proporzionale (sconta dal primo termine della (3.10)) è la più semplice, i consiste e corregge l'ingresso del sistema controllato $u(t)$ in maniera proporzionale all'errore di inseguimento $e(t)$. L'effetto è quindi:

(3-13)

$$u_p(t) = K_P e(t)$$

L'azione integrale invece incide sul comportamento regime del sistema. Il termine $\frac{K_I}{s}$ svolge un'azione integrativa dell'errore dal momento in cui il sistema è stato avviato all'istante attuale t . In questo modo mantiene memoria di quel che è accaduto nel passato e produce un contributo adeguato all'ingresso di controllo

$$u_I(t) = K_I \int_0^t r(\tau) d\tau \quad (3-14)$$

L'azione derivativa infine interviene sul comportamento transitorio del sistema. Produce un contributo non nullo non appena l'errore inizia a variare, e questo permette una correzione dinamica dell'ingresso al variare dell'errore. Il risultato di questa azione di controllo è

(3-14)

$$u_D(t) = K_D \frac{de}{dt}(t)$$

Questo contributo è quello più delicato, perché può portare facilmente a destabilizzare il sistema. Che ingresso di controllo completo è dunque formato e sommando queste tre azioni ovvero:

$$u(t) = u_p(t) + u_i(t) + u_d(t) = K_P e(t) + K_I \int_0^t e(\tau) d\tau + K_D \frac{de}{dt}(t) \quad (3-15)$$

3.3.3 Anello di corrente (Coppia)

Dato che disponiamo della misura di corrente, che ricaviamo dalla scheda di acquisizione, riusciamo a calcolare la coppia dalla relazione (2-2) che unisce coppia e corrente grazie alla costante del motore K_m . I motori DC sono molto sensibili alla corrente soprattutto per quanto riguarda l'inversione del moto e quando il motore esercita uno sforzo eccessivo; perciò, bisogna prestare molta attenzione ai limiti di saturazione. Nel caso in cui i limiti di corrente venissero infranti, sarebbe molto dannoso per il motore in quanto si andrebbe a danneggiare il materiale isolante che ricopre le bobine degli avvolgimenti di rame del motore, precludendone così il funzionamento. L'immagine seguente mostra il classico schema di un loop in retroazione negativa per il controllo in corrente.

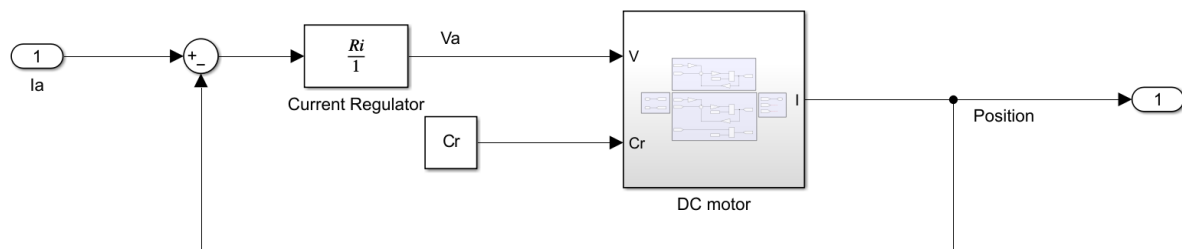


Figura 3-14 Diagramma a blocchi dell'anello di corrente (coppia)

All'interno del blocco Simulink "DC motor" sono contenute le equazioni che descrivono il sistema ovvero le equazioni 3-6 e 3-7.

Per ottenere le specifiche del sistema ovvero un errore a regime nullo a seguito di un ingresso a gradino è stato scelto un regolatore di tipo proporzionale integrale (PI) il quale è sufficiente a soddisfare le specifiche del sistema. Per questo motivo il regolatore ha questa struttura:

$$R_i(s) = K_p \left(1 + \frac{1}{T_i s} \right) = \mu \frac{1 + s\tau_z}{s} = \mu \frac{\left(1 + \frac{L_a}{R_m} s \right)}{s}$$

Scegliendo $\tau_z = \frac{L_a}{R_m}$ e un guadagno μ sufficientemente grande si risolvono i requisiti fondamentali della stabilità del sistema e si è dimostrato come sono stati calcolati i parametri del regolatore di corrente.

3.3.4 Anello di velocità

Dato che lo scenario di controllo è caratterizzato dal controllo di più variabili: impedenza, posizione, velocità e corrente, in questa condizione il progetto è stato scomposto in più fasi dove si progettano gli anelli di controllo separatamente disinteressandosi degli altri, ovvero il regolatore di corrente si interessa solo della dinamica di corrente, stessa cosa per il regolatore di velocità, posizione e corrente, questo metodo è detto controllo in cascata. Si può utilizzare solo se le costanti di tempo dei vari anelli sono 4-5 volte rispetto a quello più esterno. Se queste condizioni sono verificate si può assumere che il sistema fornisca istantaneamente il valore di coppia desiderato.

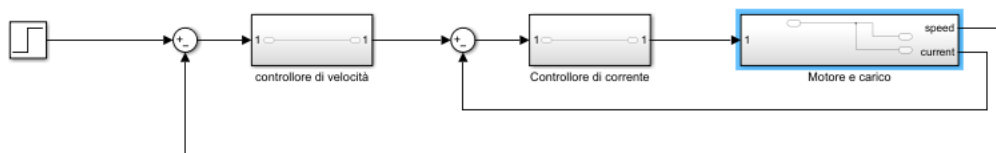


Figura 3-15 Applicazione del principio di separazione delle dinamiche all'anello di corrente

Il diagramma a blocchi del loop di retroazione per il controllo di velocità può essere descritto dalla seguente immagine:

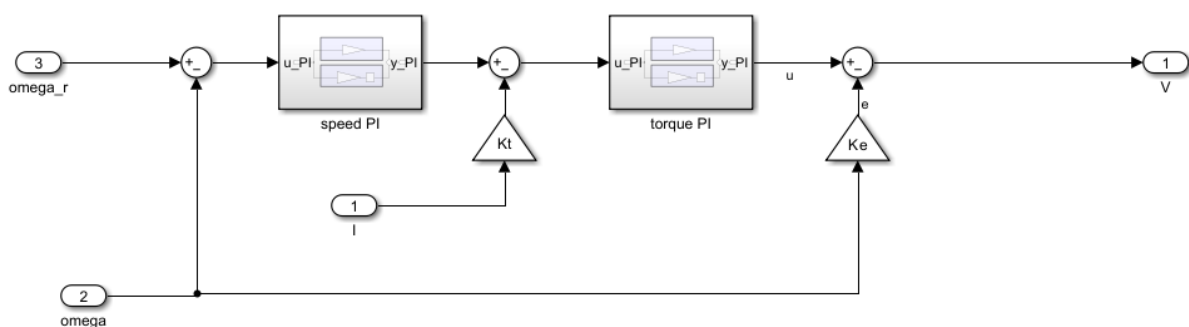


Figura 3-16 Diagramma a blocchi dell'anello di velocità

Dove l'uscita di questo "subsystem" sarà l'ingresso del sistema del motore DC.

Anche per l'anello di velocità è stato scelto un regolatore di tipo PI sfruttando la stessa logica del regolatore di corrente posso ottenere facilmente che

$$R_{\omega} = K_p \left(1 + \frac{1}{T_i s} \right) = \mu \frac{1 + s\tau_z}{s} = \mu \frac{\left(1 + \frac{(J + J_c)}{b} s \right)}{s} \quad (3-17)$$

3.3.5 Anello di posizione

Si può estendere il ragionamento delle dinamiche anche per il loop di posizione, in questo caso sarà l'anello di velocità ad essere considerato come un attuatore ideale

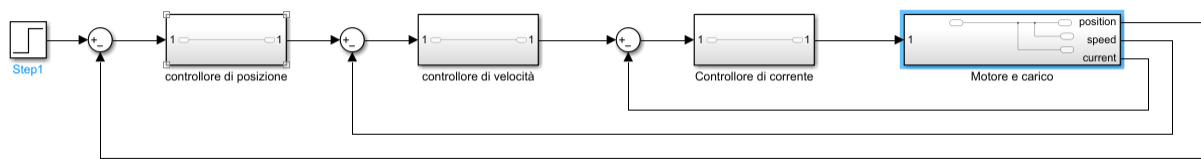


Figura 3-17 Applicazione del principio di separazione delle dinamiche all'anello di velocità

Assumendo che sia possibile applicare il principio di separazione delle dinamiche l'anello di velocità può essere descritto andando a derivare la FDT della velocità, nella trasformata di Laplace ciò vuol dire dividere per s la dalla FDT in retrazione $G_{outer}(s)$ (3-11), espressa dalla formula:

$$G_{pos} = \frac{1}{((J + J_c)s + b)s} \quad (3-18)$$

Per l'anello di posizione è stato scelto un regolatore di Proporzionale (P), poiché sufficiente a garantire la stabilità del sistema selezionando un guadagno pari a 10.

3.3.5 Anello di impedenza

L'obiettivo di questo tipo di controllo è di definire una relazione dinamica tra le velocità e le forze. Modellando un sistema simile ad una massa-molla-smorzatore.

Definendo:

$$\mathbf{M}_d \ddot{x} + \mathbf{D}_d \dot{x} + \mathbf{K}_d x = -F$$

Che con la trasformata di Laplace diventa:

$$sF(s) = -(\mathbf{M}_d s^2 + \mathbf{D}_d s + \mathbf{K}_d)$$

$F(s)$ rappresenta la forza che vogliamo applicare, M_d è l'inerzia del sistema, D_d è lo smorzamento che si vuole applicare, mentre K_d è la rigidità.

Questa equazione definisce il comportamento desiderato dell'esoscheletro assistivo. In questo modo si possono evitare chiusure da parte della mano con forze elevate a causa di incertezze sul riferimento di posizione o dell'ambiente esterno. Si possono adattare le proprietà di rigidezza in base a ciò che si vuole fare. Ed infine, è possibile mimare il comportamento umano. È possibile eseguire il tuning dei valori in base al risultato che si vuole ottenere:

- Alti valori di M_d e piccoli valori di K_d si applicano dove ci sono elevati contatti
- Alti valori di K_d e piccoli valori di M_d dove lo spazio di lavoro non prevede contatti
- Se l'ambiente è rigido, si impostano piccoli valori di K_d
- I valori di D_d sono usati per modificare le fasi transitorie

4 Validazione dei risultati

La parte finale di questo elaborato discuterà prima i risultati simulativi, applicando ciò che è stato annunciato riguardante i concetti teorici.

4.1 Implementazione Matlab e Simulink

Attraverso il software di simulazione Matlab Simulink si è potuto simulare il corretto funzionamento del motore. Tale Software permette una facile e rapida progettazione

dell'algoritmo di controllo grazie all'IDE specializzata nella simulazione di diagrammi a blocchi, questo software è molto all'avanguardia poiché permette di simulare sia sistemi tempo continui che tempo discreti, in modo da replicare il reale ambiente dei controllori digitali

4.1.2 Simulazione del controllo in cascata

Prima di implementare il controllo sul microcontrollore Arduino è importante assicurarsi che i regolatori progettati precedentemente funzionino correttamente e inseguano il riferimento e non si verifichino strane dinamiche. Questa fase è fondamentale per testare il funzionamento limite del sistema e la corretta saturazione delle variabili, dato che essendo in simulazione non si rischia il danneggiamento dei componenti qualora qualcosa sia progettato male.

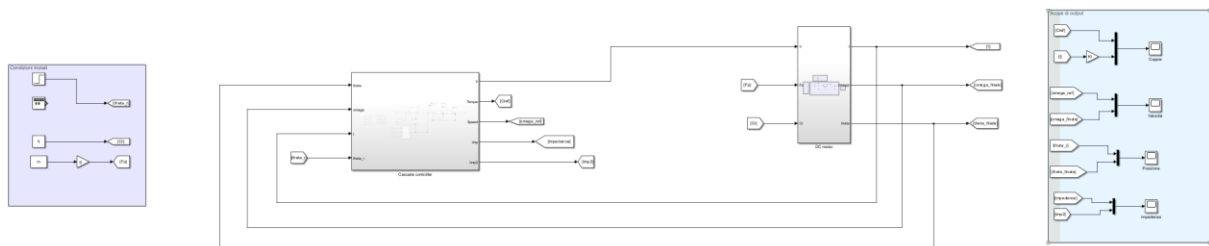


Figura 3-18 Schemi a blocchi del sistema

Il sistema può essere suddiviso in macro-blocchi, per descriverne il funzionamento si può suddividere in:

- **Condizioni iniziali** (area viola): dove è presente il blocco utilizzato per imporre il riferimento al sistema, ovvero il gradino. Al di sotto è presente un blocco di coppia resistente impostata a 0.01 Nm che rappresenterebbe la coppia che la mano del paziente impone sul guanto. Infine la forza peso perché nella simulazione si tiene anche in conto del meccanismo utilizzato per trasmettere la coppia dal motore alla mano e quindi è necessario considerare anche la forza peso.
- **Regolatori più motore DC** (zona centrale) è costituita da due sotto blocchi quello di destra che contiene i regolatori, mentre quello di sinistra contiene le equazioni del motore DC
- **Scope** (Area azzurrina): blocchi utilizzati per osservare l'andamento dei segnali nel tempo

4.1.3 Analisi blocco regolatori

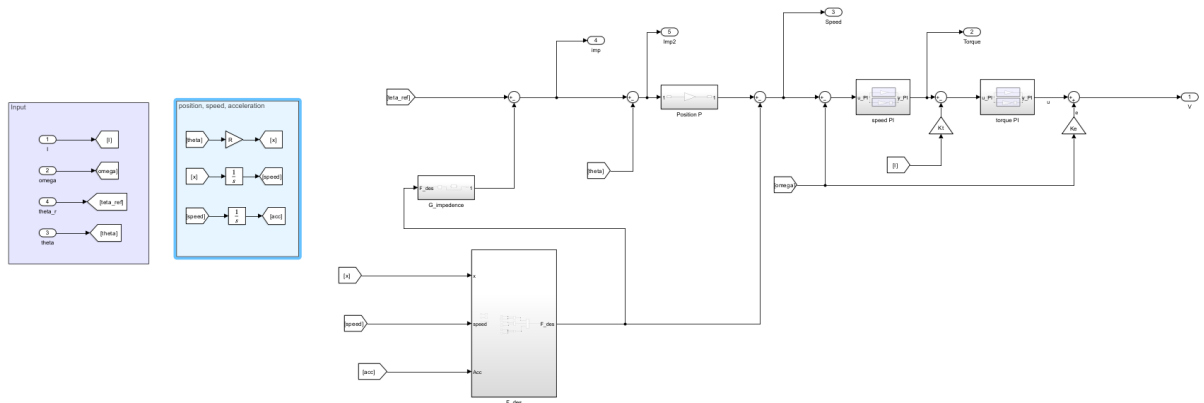


Figura 3-19 Schema a blocchi dei regolatori

Nell'area viola ci sono gli input del sistema ovvero: la corrente, la velocità, la posizione che sono i segnali retroazionati ed il riferimento. Nel blocco azzurrino invece calcolo lo spostamento lineare dato che nel controllo di impedenza ho bisogno di quello lineare e non di quello angolare, utilizzando la formula:

$$x = r\theta \quad (3-19)$$

Successivamente integro la posizione per ottenere la velocità e lo rifaccio ancora una volta per ottenere l'accelerazione.

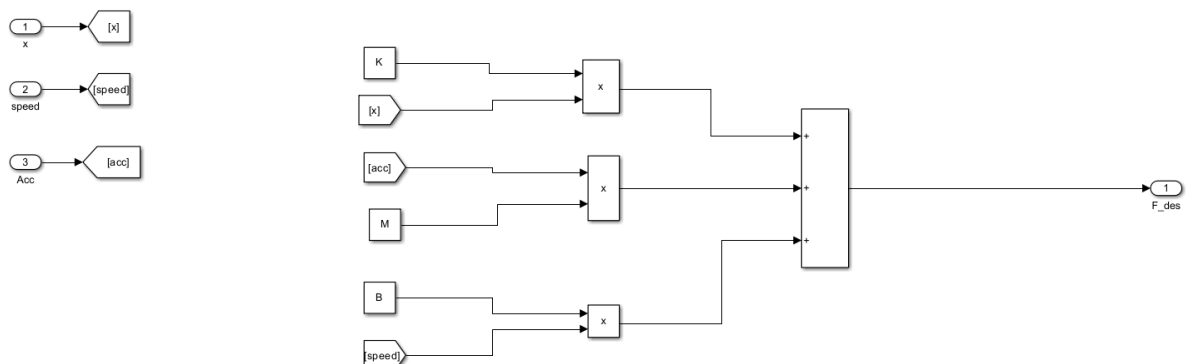


Figura 3-20 schema a blocchi impedenza

Nel blocco inferiore (F_des) è contenuta l'equazione per introdurre la forza che simula massa, molla e smorzatore che viene data in input al PID.

Descrivendo lo schema complessivo si ha che il riferimento di posizione entra nel sistema, dove viene sottratta la forza del controllo di impedenza, la loro differenza genera la posizione con tanto di impedenza. Viene retroazionata la posizione del motore e sottratta al riferimento di posizione, generando così l'errore di posizione che grazie ai regolatori, a regime è nullo.

4.2 Discretizzazione delle FDT degli anelli di controllo

Per poter utilizzare i regolatori calcolati precedentemente in sistemi a tempo discreto quindi su microcontrollori è necessario eseguire la discretizzazione. Una volta ottenuto il regolatore discretizzato, si può implementare su simulink e osservare se ci siano delle anomalie nel sistema, nel caso non ce ne siano allora si può passare al test sul prototipo

4.2.1 Metodo di Tustin (o trasformazione bilineare)

Esistono molteplici tecniche per discretizzare una funzione, in generale se è data una funzione $f: \mathbb{R} \rightarrow \mathbb{R}$ continua vogliamo calcolare l'integrale definito

$$\int_{kT_s}^{(k+1)T_s} f(t) dt \quad (3-20)$$

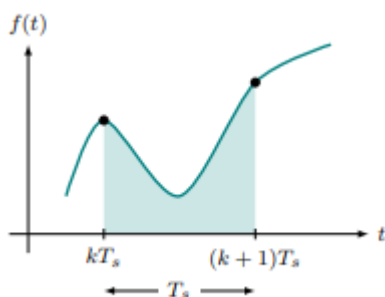


Figura 3-21 Area sottesa alla curva

Graficamente, il valore esatto dell'integrale è l'area sottesa da $f(t)$ per $t \in [kT_s, (k+1)T_s]$.

Questo integrale (3-20) si può approssimare come:

$$\int_{kT_s}^{(k+1)T_s} f(t)dt \approx \left((1 - \alpha)f(kT_s) + \alpha f((k+1)T_s) \right) * T_s \quad (3-21)$$

Dove $\alpha \in [0,1]$ è un parametro.

Se si pone $\alpha = 0$ si usa il metodo: Eulero in avanti, l'integrale (3-21) diventa:

$$\int_{kT_s}^{(k+1)T_s} f(t)dt \approx f(kT_s) * T_s \quad (3-22)$$

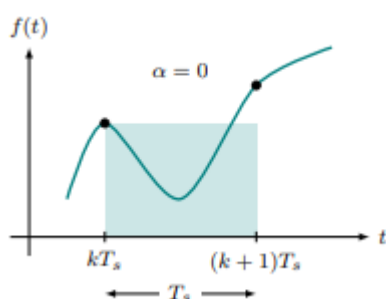


Figura 3-221 Eulero in avanti

Al contrario ponendo $\alpha = 1$ si usa il metodo: Eulero indietro, l'integrale (3-21) diventa:

$$\int_{kT_s}^{(k+1)T_s} f(t)dt \approx f((k+1)T_s) * T_s \quad (3-23)$$

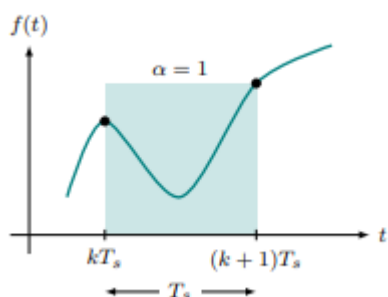


Figura 3-23 Eulero all'indietro

Il “difetto” di questi due metodi è che il primo sovrastima l'area sottesa alla curva, mentre il secondo sottostima l'area sottesa alla curva, come si può vedere dall'immagine successiva, dove si mettono in evidenza i due metodi precedentemente

enunciati. Sull'asse delle ascisse è posto il tempo discretizzato, mentre sull'asse delle ordinate è posto il valore della funzione

Il metodo migliore che si può utilizzare è il metodo di Tustin (o dei trapezi), l'area sottesa alla curva viene approssimata mediante l'utilizzo di trapezi, costruiti raccordando con un segmento i campioni di segnali adiacenti, come mostrato nella figura seguente:

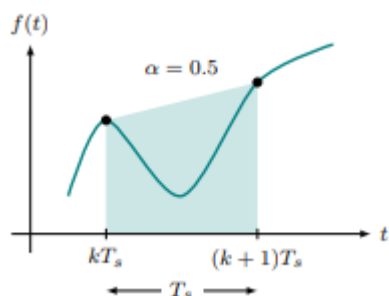


Figura 3-243 Metodo di Tustin

nella prossima figura si possono confrontare i 3 metodi

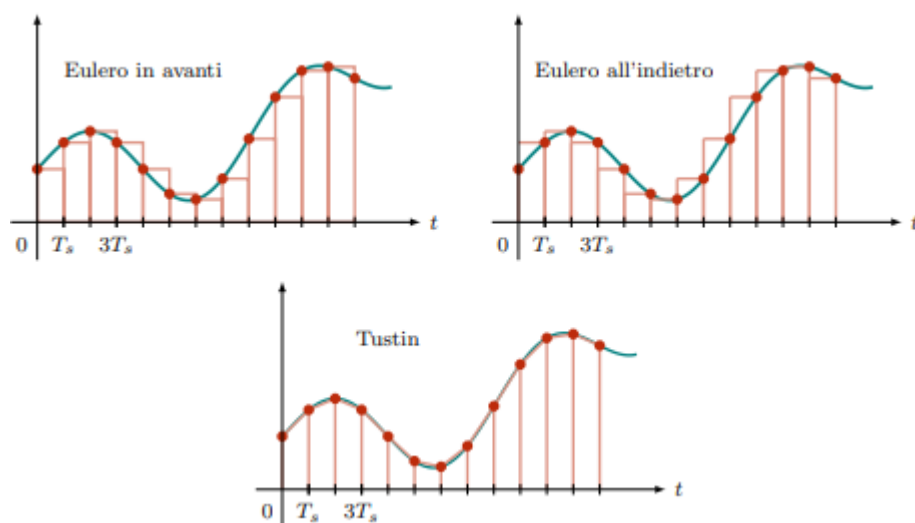


Figura 3-25 Confronto dei 3 metodi

Il metodo di Tustin è stato scelto in quanto, a differenza degli altri metodi, traspone perfettamente il semipiano sinistro del dominio continuo s , nella circonferenza unitaria del dominio discreto z . Tale peculiarità permette di sfruttare tutta la circonferenza unitaria, allo stesso tempo evita che i poli stabili perdano tale fondamentale caratteristica una volta discretizzati (poli instabili = poli esterni alla circonferenza unitaria).

Si osservi che nel dominio a tempo discreto, il termine z^{-1} che moltiplica una variabile rappresenta il valore assunto della variabile nell'istante precedente

4.2.3 Discretizzazione del regolatore PI

Per discretizzare il regolatore PI, utilizzato per l'anello di corrente e di velocità si possono utilizzare diversi metodi, uno dei quali può essere quello di utilizzare alcuni passaggi matematici qui elencati:

$$R(s) = \frac{Y(z)}{X(z)} = K_p + \frac{K_i}{s} \quad \text{con } s = \frac{2}{T_s} \frac{z-1}{z+1} \quad (3-25)$$

Attraverso alcuni passaggi matematici si può ottenere:

$$R(z) = R(s) \Big|_{s=\frac{2}{T_s} \frac{z-1}{z+1}} \quad (3-26)$$

$$= K_p + \frac{K_i}{\frac{2}{T_s} \frac{z-1}{z+1}} = K_p + \frac{T_s(1-\alpha+\alpha z)K_i}{z-1} = \frac{z(K_p + \alpha T_s K_i) + (1-\alpha)T_s K_i - K_p}{z-1} \quad (3-27)$$

Ponendo $\alpha = \frac{1}{2}$

$$R(z) = \frac{z\left(K_p + \frac{1}{2}T_s K_i\right) + \frac{1}{2}T_s K_i - K_p}{z-1} \quad (3-28)$$

T_s indica il tempo di campionamento che per questa applicazione è stato scelto pari a $T_s = 1 * 10^{-5}$

In maniera molto più efficace e rapida si può usare il comando di MATLAB: `sysd=c2d(sysc,Ts,method)`, per `sysd` si intende il nuovo sistema discretizzato, `sysc` è il sistema tempo continuo, `Ts` è il tempo di campionamento, e per `method` si indica il metodo di discretizzazione che si intende utilizzare, in questo caso Tustin. Questo comando calcola direttamente il regolatore discretizzato che poi verrà implementato su Arduino.

Per quanto riguarda il regolatore di corrente si ottiene:

(3-29)

$$R_i(z) = \frac{12.78z - 10.17}{z - 1}$$

Invece per il regolatore di velocità risulta:

$$R_\omega(z) = \frac{3.32 * 10^{-5}z - 3.32 * 10^{-5}}{z - 1} \quad (3-30)$$

Per quanto riguarda il regolatore di posizione invece non è necessario discretizzare poiché si ha solo un guadagno.

Il regolatore di impedenza diventa

$$R_{imp} = \frac{5.1 * 10^6 z^2 - 1.02 * 10^7 z + 5.1 * 10^6}{z^2 - 1} \quad (3-31)$$

4.3.4 Azione di saturazione e anti Windup

Le funzioni scritte qui sopra sono quelle utilizzate nel sistema reale perciò è estremamente necessario limitare la variabile di uscita entro i limiti supportabili dal motore forniti sul data sheet del motore. Nel caso il valore superi i limiti consentiti viene calcolato il valore massimo che consente il motore. Si può implementare questa funzione utilizzando il blocco di switch di Simulink.

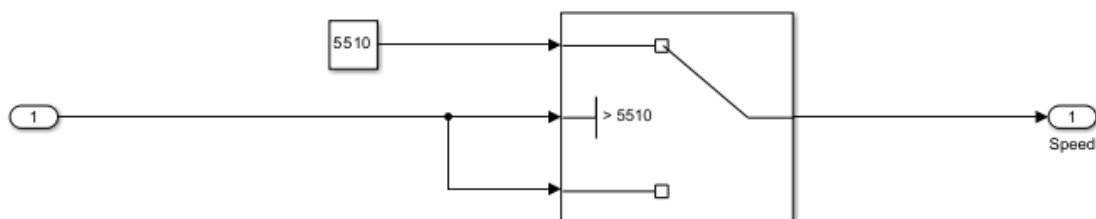


Figura 4-26 Switch per l'azione anti Windup

4.4 Risultati Simulink del controllo in cascata

Verranno riportati i risultati ottenuti in seguito alla simulazione della struttura del controllo in cascata.

4.4.1 Controllo di impedenza

Come prima immagine viene riportata l'uscita del segnale di impedenza. Sull'asse delle ordinate è posto il valore di forza, mentre sulle ascisse il tempo. Il segnale Blu è il riferimento di posizione, mentre quello arancione è il segnale di impedenza. Come si può vedere nell'intorno di $t = 0$ l'impedenza aumenta un pochino, questo è causato dalla forza resistente che il paziente impone sul guanto. All'istante $t = 1$, viene inviato il gradino di posizione pari ad 1 radiante, il sistema per raggiungere questo riferimento di posizione deve imprimere una forza molto più elevata al sistema che è rappresentata dal picco arancione.

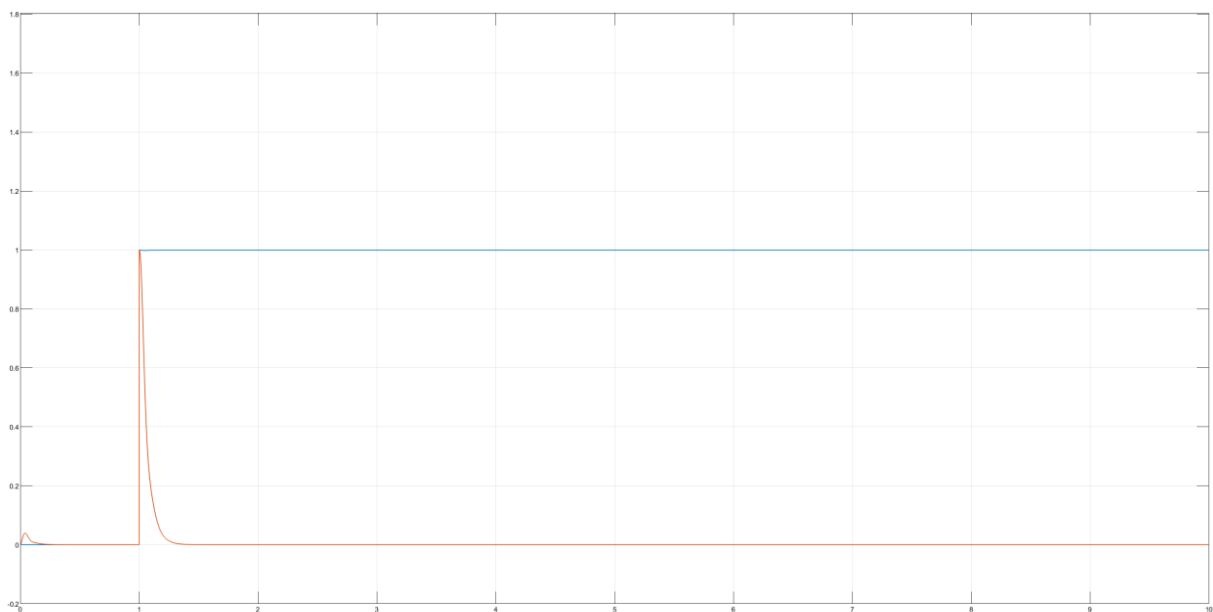


Figura 4-27 risultati simulink controllo di impedenza

4.4.2 Controllo di posizione

I riferimenti sugli assi cartesiani sono gli stessi dell'immagine precedente, ma in questo scope di Simulink si paragona il riferimento di posizione in blu con la posizione del sistema in arancione. Quella leggera deviazione dal riferimento nell'intorno di $t = 0$ è sempre dovuta alla forza che la mano del paziente applica sul guanto e quindi il risultato è uno spostamento nel verso opposto rispetto alla chiusura. Come si può osservare il riferimento viene raggiunto in poco tempo e l'errore a regime è nullo, per questi due motivi si può affermare che le specifiche del sistema sono rispettate

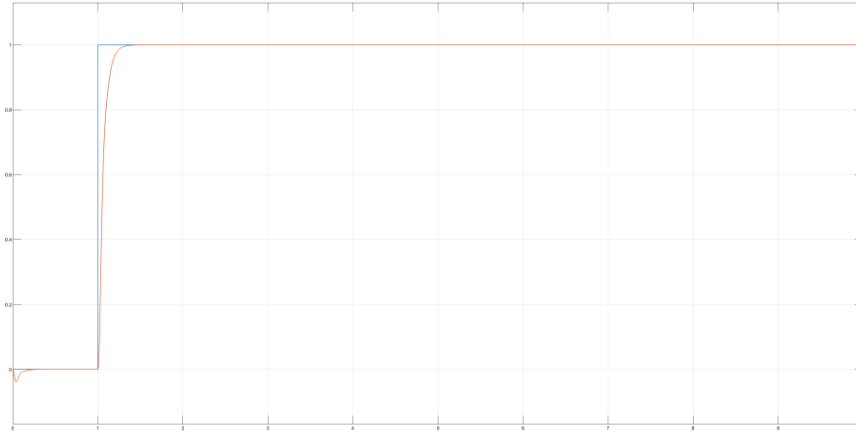


Figura 4-28 risultato Simulink controllo di posizione

4.4.3 Controllo di velocità

Questo invece è il segnale di velocità, in blu si ha il riferimento ovvero l'uscita del regolatore di posizione e in arancione si ha la velocità del motore. Come si può vedere, il motore è molto sensibile alle variazioni di posizione, come nell'intorno di $t = 0$ e $t = 1$. Però a regime il riferimento viene raggiunto correttamente e soprattutto il motore non scatta sul riferimento ovvero riesce ad annullare l'errore senza muoversi in una direzione e poi in quella opposta senza mai raggiungere il riferimento.

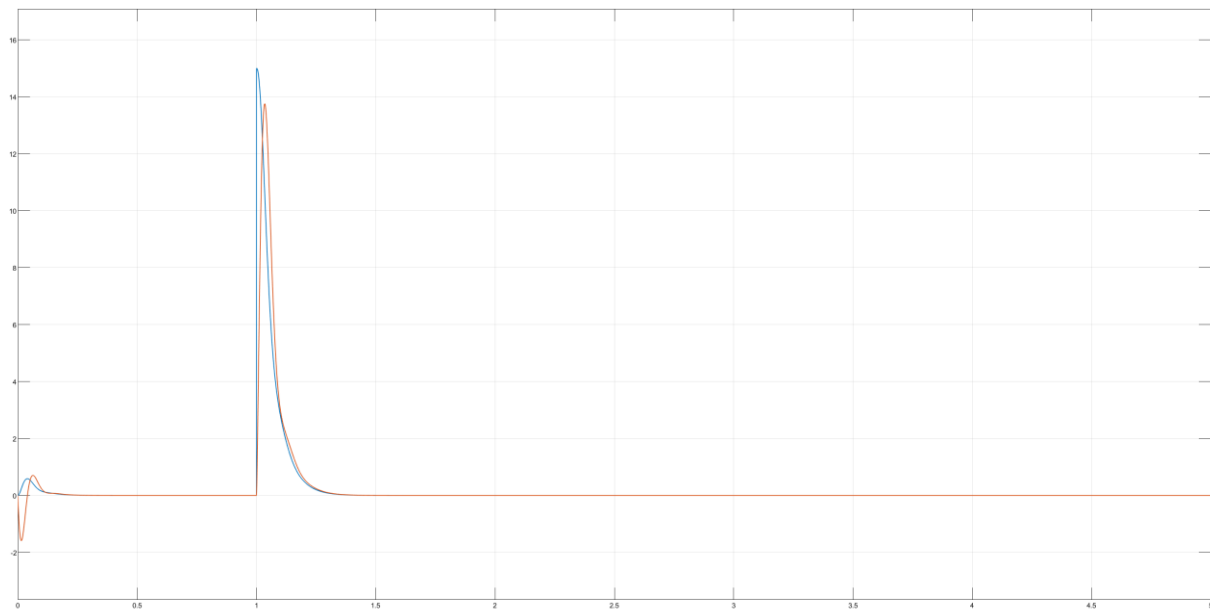


Figura 4-29 Controllo di velocità

4.4.4 Controllo di coppia

Questa è la coppia del sistema, in realtà ci sarebbero due segnali che sono sovrapposti, uno è l'uscita del regolatore di velocità che rappresenterebbe il riferimento da seguire e l'altro segnale è la coppia del motore, il fatto che siano sovrapposti vuol dire che sono corretti. La coppia aumenta all'inizio per andare a contrastare sempre la forza che il paziente imprime sul guanto, il picco è quando viene imposto il riferimento di posizione. La coppia non torna mai al valore nullo perché "rimane sempre in coppia" ovvero contrasta la forza che il paziente applica sul guanto

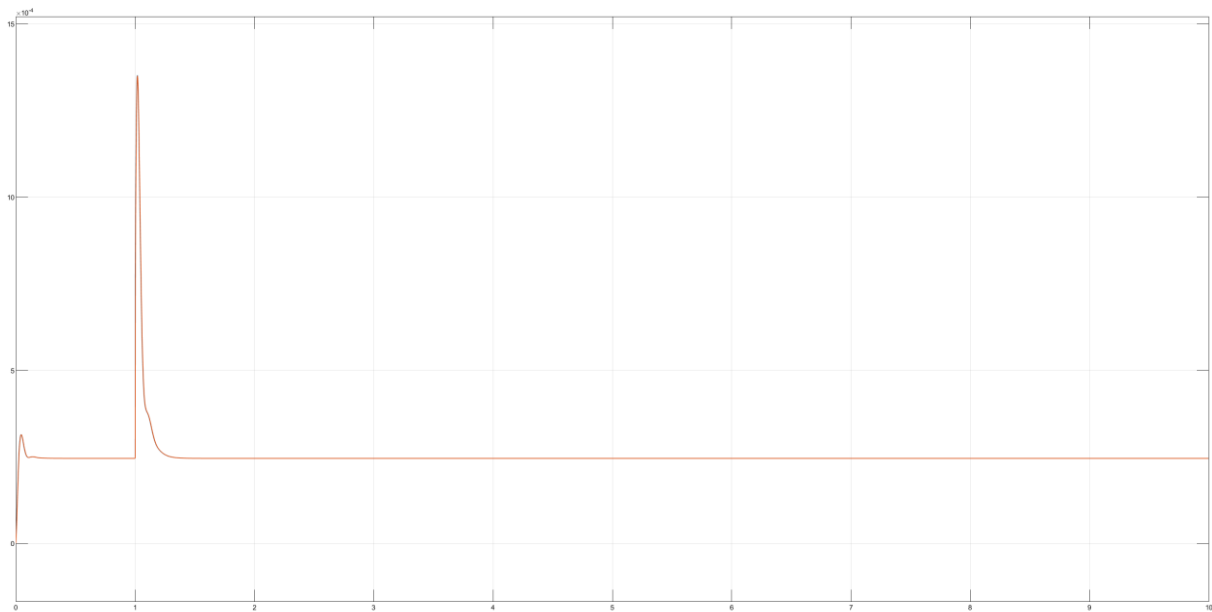


Figura 4-30 Controllo di coppia

4.5 implementazione algoritmo di controllo

La realizzazione dell'algoritmo di controllo si può suddividere in 4 parti fondamentali:

- La realizzazione del segnale PWM
- La sincronizzazione PWM ADC
- Lettura di posizione e velocità dell'encoder
- Implementazione finale del codice

4.5.1 Segnale PWM

La tecnica PWM è volta a ricreare la funzione di commutazione tramite l'intersezione di un segnale modulante con un segnale portante (comunemente un'onda triangolare). Essa permette di ottenere una tensione media variabile dipendente dal rapporto tra la durata dell'impulso positivo e dell'intero periodo (duty cycle). Dato che nei motori DC la velocità è regolata dalla tensione sarà proprio il segnale PWM a regolare la velocità di rotazione del motore DC. Il segnale PWM ha due caratteristiche fondamentali:

- La frequenza del segnale
- Il duty cycle che viene espresso in percentuale rappresenta la frazione di tempo all'interno del periodo dove il segnale è alto. Ad esempio, un segnale PWM con

duty cycle pari al 50% vuol dire che in un periodo il lasso di tempo che il segnale trascorre quando è alto è lo stesso di quando è basso.

4.5.1 Realizzazione del segnale PWM

Per realizzare il segnale PWM sulla piattaforma Arduino esistono diversi metodi, uno dei quali è la gestione ad interrupt che comporta la modifica dei registri e del clock della CPU dell'Arduino DUE in modo da impostare la frequenza del segnale PWM. Il problema di questa soluzione è che non è possibile modificare il valore del duty cycle durante il loop di Arduino. Poiché il clock modifica la frequenza a cui si vuole impostare il segnale, non per quanto tempo si vuole impostare il segnale alto; infatti, il duty cycle è sempre impostato al 50%. La modifica dei registri avviene nel setup e quindi durante tutto il loop del codice non è possibile modificare né i registri né il clock.

Per modificare il duty cycle a piacimento è necessario cambiare metodo, impostando manualmente sia la frequenza a cui il pin deve oscillare, sia il tempo in cui il Pin rimane alto o basso. Tra i due metodi disponibili è stato scelto il secondo poiché era l'unico che rispettava i requisiti del sistema, anche se comporta alcuni svantaggi, perché per impostare il tempo in cui il pin rimane alto e basso sulla scheda Arduino Due si implementa con questa logica: pongo il pin alto e aspetto il tempo necessario affinché rimanga alto e faccio la stessa cosa per porre il pin basso. Lo svantaggio è che questo ritardo non è molto preciso e potrebbe portare a segnali PWM non accurati.

4.5.2 Sincronizzazione PWM – ADC

La sincronizzazione PWM-ADC (Analog to Digital Converter) è una tecnica utilizzata nei sistemi di controllo e acquisizione dati per sincronizzare il funzionamento tra PWM e ADC. Questa tecnica è importante quando si vuole campionare un segnale in modo che la conversione analogica-digitale avvenga in momenti precisi, spesso legati ai cambiamenti del segnale PWM. Quando si utilizza il segnale PWM per controllare dei motori, il segnale di

tensione che si vuole controllare può essere influenzato dal segnale stesso, causando rumore o distorsione. Sincronizzare la lettura dell'ADC con il segnale PWM permette di effettuare le misurazioni nei momenti giusti, ad esempio durante la parte stabile dell'onda PWM (spesso a metà del momento in cui il segnale è attivo), migliorando così la qualità e l'accuratezza della misurazione. È stato scelto di sincronizzare l'ADC con il segnale PWM quando quest'ultimo si trova a metà del fronte positivo. Proprio in quel momento viene misurata la corrente. Se non fosse così ovvero se l'ADC campionasse in modo casuale durante il ciclo PWM, potrebbe rilevare dei transitori o rumore, causando letture imprecise. La sincronizzazione riduce significativamente questo rischio. Si ha una migliore efficienza di controllo perché la sincronizzazione permette un feedback accurato al sistema di controllo, migliorando la risposta dinamica e la stabilità complessiva del sistema. Nei sistemi a PWM, i rapidi cambiamenti tra gli stati ON e OFF possono introdurre rumore nel segnale. Sincronizzare il campionamento a un momento preciso può ridurre l'effetto di questo rumore, migliorando la qualità del segnale acquisito.

4.5.3 Lettura dall'Encoder

Il passo successivo è stato quello di far funzionare l'encoder ovvero di leggere gli impulsi dell'encoder. L'encoder è collegato a 2 pin, in base al verso di rotazione dell'albero motore, prima un pin diventa alto e poi l'altro. Ad esempio, se l'albero ruota in senso orario prima il pin A diventa alto e poi il pin B diventa alto. Se ruota nella direzione opposta viceversa. La logica sviluppata in questo codice è questa: si inizializza un contatore che ad ogni impulso dell'encoder aumenta, l'encoder utilizzato per un giro conta 4096 impulsi, attraverso una conversione: divido il conteggio per 4096 così ottengo i giri del motore e successivamente li divido ulteriormente per il rapporto di riduzione del motoriduttore così calcolo quanti giri il motore ha fatto. Dai giri posso ricavarmi la velocità del motore poiché essa è la derivata della posizione. Calcolando il delta della posizione ovvero posizione attuale sottratto alla posizione precedente e dividendolo per un periodo di tempo infinitesimo allora posso ricavare la velocità:

$$v = \frac{dx}{dt} = \frac{x - x_{precedente}}{dt}$$

4.6 Implementazione Arduino

Una volta appurato che il controllo e la saturazione funzionino correttamente, non resta che provare l'implementazione su piattaforma Arduino, verificando così l'effettivo comportamento reale. Il codice Arduino, oltre alle FDT dei regolatori, contiene un insieme di altre funzioni e condizioni che permettono il corretto azionamento dell'esoscheletro.

4.6.1 Risultati Arduino sincronizzazione PWM-ADC

Durante la fase di sperimentazione e test sul prototipo, è stato necessario osservare l'andamento dei segnali logici nel tempo. Per ottenere ciò è stato fondamentale lo strumento analizzatore logico comprato su un sito online, il quale permette di aver traccia in tempo reale dei valori logici dei pin collegati tramite cavo. Verranno ora riportati i risultati ottenuti eseguendo il codice sulla piattaforma Arduino e tramite il programma Logic 2.0 che permette di visualizzare i valori logici ricevuti in input dall'analizzatore logico. Qua sotto viene riportato uno screenshot ricavato da tal programma dove è stato impostato il segnale PWM il primo in alto (bianco) a 20 KHZ e il duty cycle al 17%. Mentre nel secondo screenshot ho impostato un duty cycle all'85% e una frequenza a 20 KHz. Invece nella seconda riga il segnale arancione è l'ADC, per visualizzarlo sull'analizzatore logico ho impostato un pin in modo che triggerasse all'inizio e alla fine della conversione, in questo modo si capisce quando inizia e quando finisce.

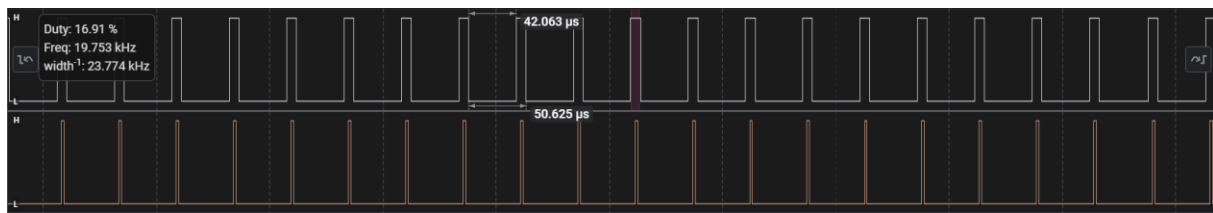


Figura 4-31 segnale PWM con frequenza pari a 20 KHz e duty cycle impostato al 15%

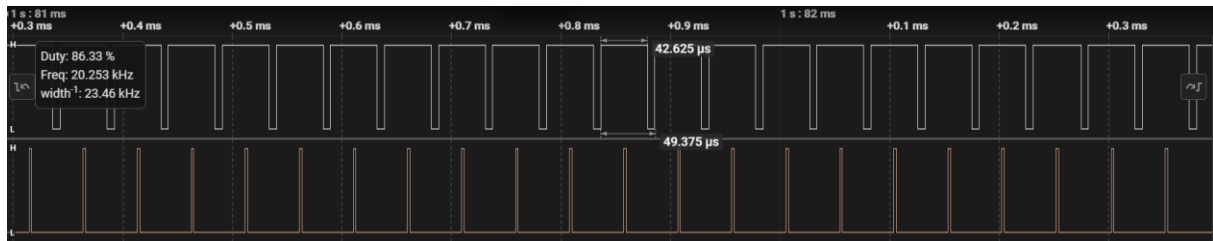


Figura 32 segnale PWM con frequenza pari a 20 KHz e duty cycle impostato al 85%

4.7 funzionamento del motore

Per rendere l'elettronica più pulita a livello visivo abbiamo realizzato in laboratorio, attraverso una scheda mille fori, un circuito stampato con i 4 pulsanti e 3 LED.

- Il pulsante più alto serve per azionare il motore, è il pulsante utilizzato per cambiare i riferimenti da 0 a quello voluto; quindi, è anche il pulsante che cambia la direzione di rotazione dell'albero e di conseguenza l'apertura e la chiusura della mano.
- Il pulsante più piccolo posizionato dalla parte opposta dei Led serve per dare il riferimento. La prima azione da eseguire è premere il pulsante più alto (quello precedentemente descritto) in modo da azionare il motore, la mano si andrà a chiudere. Però non è detto che quando una paziente chiuda la mano abbia sempre bisogno degli stessi gradi di rotazione dell'albero motore. Proprio per questo motivo si può premere il bottone più piccolino in modo da far chiudere la mano sempre nella stessa posizione
- Il pulsante più grande di fianco a quello alto serve per interrompere il funzionamento, è un pulsante di sicurezza che è sempre utile inserire, serve in caso di errori nella chiusura o eventuali danneggiamenti del motore.

- L'ultimo pulsante è lasciato senza funzioni, è stato inserito nel caso si vogliano realizzare sviluppi futuri.

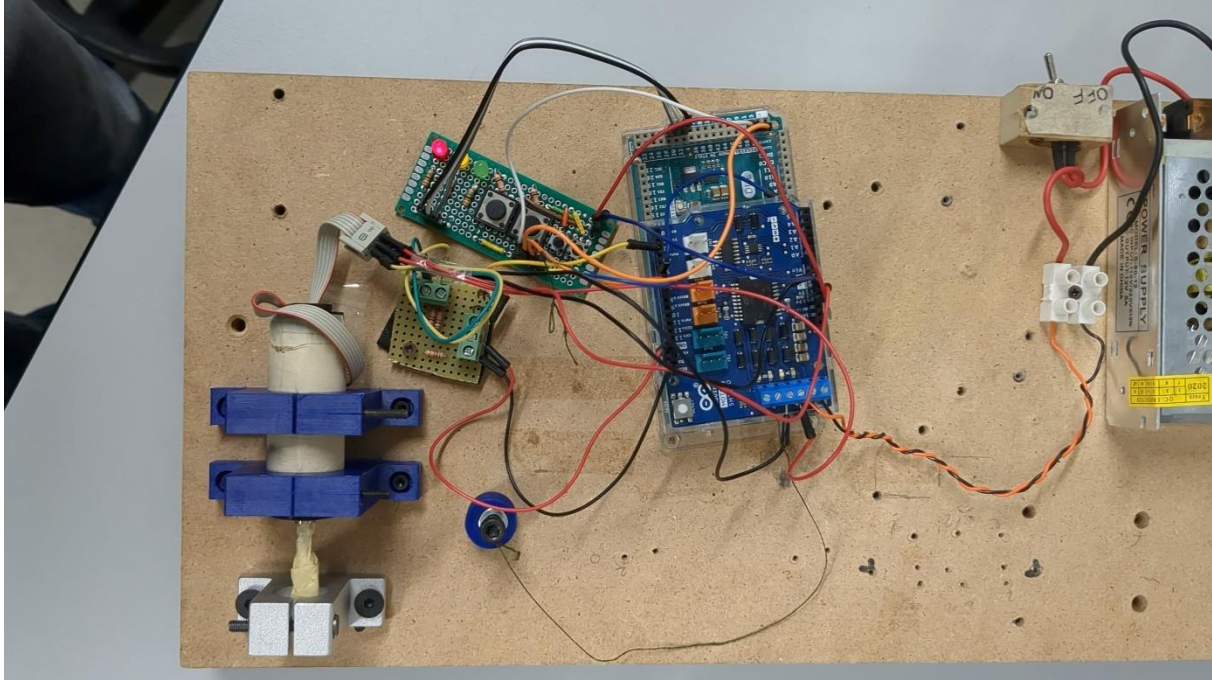


Figura 4-33 Set-up del motore con pulsantiera

5 Codice finale

Una volta appurato che la sincronizzazione PWM – ADC funzionasse e che venissero letti bene i valori dall'encoder il passo successivo è stato quello di inserire i regolatori estratti da Matlab e SIMULINK e progettare il funzionamento logico del motore. Proprio in questa fase si sono riscontrati i primi problemi perché durante la fase di impostazione dell' ADC viene eseguita la lettura di corrente e inoltre vengono anche scritti i regolatori di corrente, velocità, posizione e impedenza. Dato che il compilatore impiega un certo tempo per eseguire i comandi dentro all'ADC e non è istantaneo, si aggiunge un ritardo che sfasa la sincronizzazione tra PWM – ADC vanificando ciò che era stato fatto prima. Data l'urgenza di avere un codice funzionante ho modificato il codice precedentemente caricato andandolo a perfezionare e ultimare per questo tipo

di motore, inserendo i riferimenti a gradino e migliorando la fase di raggiungimento del setpoint. Inoltre, è stata implementata in maniera definitiva la comunicazione con il sensore EMG in modo da eliminare i precedenti azionamenti tramite pulsanti.

5.1 struttura del codice

La struttura del codice, come tipicamente avviene nei sistemi Arduino, si suddivide in due macro-funzioni:

- **Setup:** funzione eseguita una volta sola all'avvio, in cui vengono inizializzate le porte di Input/Output (I/O) e le funzionalità di sistema.
- **Loop:** funzione che viene eseguita continuamente (ciclo While(1)) una volta terminata la fase di setup. In tale ciclo sono presenti le chiamate a funzione, opportunamente condizionate, necessarie allo svolgimento delle azioni richieste.

5.2 Funzionamento del codice

5.2.1 Lettura variabili di interesse

Nella prima parte del loop, con una frequenza di campionamento pari a T_s , avviene la lettura delle variabili di posizione, velocità e corrente.

- **Letture di posizione:** una volta settato opportunamente il contatore hardware nella fase di setup, basterà leggere il valore contenuto nel registro corrispondente per avere il conteggio del pulse di encoder. Bisognerà poi dividere tale valore per la risoluzione dell'encoder e il rapporto di riduzione del riduttore per avere l'effettivo numero di giri compiuti.

- **Lettura di velocità:** la lettura di velocità avviene tramite una funzione dedicata che, tramite il conteggio dei pulse in un intervallo noto, è in grado di ricavare tale valore.

```
// Funzione utilizzata per calcolare il valore della velocità del motore
float SpeedCalc(unsigned long dt, int PulseCnt, int CntPrev) {
    int DeltaCnt = PulseCnt - CntPrev;
    float DeltaThetaMis = DeltaCnt * RadPerCount;

    float SpeedRad = (DeltaThetaMis / (dt) ) * 1000;
    //SpeedRad = ((TWO_PI * DeltaCnt) / (EncoderQuad * dt) * 1000);
    return SpeedRad;
}
```

Figura 34 Funzione utilizzata per calcolare la velocità

- **Lettura di corrente:** il valore analogico fornito dalla Motor Shield viene letto nel momento in cui si sincronizza ADC e PWM in modo che non ci siano disturbi.

5.2.2 Cambio di stato

Per dare inizio alla chiusura, o alle azioni successive si aspetta il segnale proveniente dal sensore EMG, il quale fornisce il numero 1 se la mano si vuole aprire e 2 se si vuole chiudere. Perciò quando il paziente decide che vuole chiudere la mano, il sensore rileverà gli impulsi sul braccio e comunica alla scheda di controllo Arduino DUE l'intenzione che verrà poi eseguita. Nonostante tutto, i pulsanti sono stati comunque inseriti perché se il paziente è stanco, i suoi muscoli non riescono più ad imprimere un impulso sufficiente per essere letto dall'EMG, di conseguenza abbiamo scelto di lasciare i pulsanti ed il loro funzionamento. Il cambio di stato può avvenire solo se ci si trova già a regime (movimentazione precedente conclusa) oppure se il guanto sta afferrando un oggetto, situazione interpretabile dal quantitativo di corrente assorbita. In caso di ostacolo in chiusura e conseguente carico sul motore, la corrente assorbita tenderà ad aumentare fino al valore di saturazione impostato, tale condizione può essere interpretata come corretta presa dell'oggetto.

5.2.3 Dinamiche di moto

Una volta appurato che sia possibile eseguire il cambio di stato, viene generato un riferimento a gradino. Il controllo tenderà ad eseguire l'obiettivo con una dinamica che dipenderà dalle costanti di tempo dei regolatori. La prima fase del moto avviene ad un'elevata dinamica, in modo da avere tempi di attuazione contenuti e poter esercitare la maggior forza di presa possibile. Una volta avvicinati all'obiettivo, tale dinamica viene drasticamente rallentata diminuendo le costanti di tempo dei vari regolatori. Così facendo il sistema sarà in grado di esercitare forze elevate senza però ricadere in andamenti oscillatori una volta raggiunto il riferimento.

5.2.4 Verifica raggiungimento Set-Point

Una volta giunti all'obiettivo posizionale richiesto, il sistema continuerà ad operare nella zona a bassa dinamica fino alla richiesta del prossimo cambio di stato. Tale procedura consente di continuare ad esercitare l'azione di controllo, senza però avere un eccessivo consumo di corrente. Così facendo, qualora il sistema venga sollecitato da una forza esterna, sarà sempre pronto e responsivo a compensarla per rimanere nella posizione precedentemente raggiunta.

5.2.5 Azionamento del motore

Dopo aver controllato il raggiungimento del set-point, e aver avuto un risultato positivo il motore può essere azionato nuovamente se il paziente avvia il comando.

6 Conclusione

6.1 commenti sul lavoro svolto

La redazione dell'elaborato ha seguito fedelmente le fasi effettive della prototipazione del sistema. Inizialmente, è stato fondamentale identificare chiaramente le richieste e gli obiettivi da raggiungere. Successivamente, attraverso una approfondita ricerca bibliografica sullo stato attuale della tecnologia, sono state considerate le possibili difficoltà e criticità che avrebbero potuto emergere lungo il percorso.

Dopo aver esaminato diverse alternative e scenari, si è deciso di adottare l'approccio considerato più adeguato, avviando lo studio dei materiali e dei metodi necessari per individuare le componenti e le procedure più efficaci. Una volta ottenuti i materiali e verificato il corretto funzionamento teorico tramite software di simulazione, sono stati condotti i primi test pratici. Si è riscontrato subito che il passaggio dalla teoria alla pratica spesso comporta imprevisti e comportamenti inattesi.

Tuttavia, grazie all'esperienza acquisita durante le numerose prove effettuate, è stato quasi sempre possibile risolvere i problemi emersi, o almeno individuare soluzioni alternative in caso di difficoltà persistenti. Dopo vari tentativi, è stato possibile raggiungere un risultato soddisfacente, completando con successo la sperimentazione finalizzata al controllo dell'attuatore dell'esoscheletro per la mano, come richiesto dall'Istituto di Montecatone.

6.1 commenti sul lavoro svolto

Si sono incontrate varie difficoltà in questo progetto uno delle più grandi è stata quella di realizzare un segnale PWM alla frequenza giusta che deve essere impostata a 20 KHz, sincronizzare la PWM con l'ADC, ed infine leggere la posizione e la velocità dall'encoder. È stata scelta la strada di utilizzare i ritardi perché l'altra è molto difficile e complessa dato che la gestione ad interrupt comporterebbe la modifica dei registri della scheda Arduino Due. Ovviamente sono stati eseguiti svariati tentativi di eseguire l'interrupt, ma purtroppo nessuno funzionava in maniera ottimale per ciò che bisognava fare.

Questo problema va sicuramente risolto, ci possono essere due soluzioni differenti:

- Comprare e utilizzare una scheda di controllo diversa, ad esempio, una scheda Texas instrument specifica per il controllo di motori di piccola taglia, dove la realizzazione del segnale PWM è molto più semplice da pensare e implementare e vale la stessa cosa per la sincronizzazione PWM – ADC, dall'altra parte però ciò comporterebbe un'ulteriore spesa
- Studiare in maniera approfondita e dettagliata come si possono modificare i registri di Arduino DUE in modo da modificare il clock interno del processore e porlo pari a 20 KHz e sviluppare tutto il codice su questo tipo di frequenza

Una possibile futura implementazione può essere quella di non dover più fare 2 giri a vuoto per impostare il riferimento. Poiché attualmente il funzionamento è questo: si accende il motore con l'interruttore della PSU, successivamente si preme il pulsante lungo che aziona il motore e chiude la mano con un riferimento preimpostato a 4 giri. Una volta che la mano si è chiusa si può ripremere il pulsante lungo per aprire la mano. Infine, si può aprire la mano e dare il riferimento con il pulsante piccolino. Come si può intuire la prima apertura e chiusura della mano è superflua. Sarebbe opportuno eliminare questi due giri a vuoto in modo da rendere più veloce l'impostazione del riferimento.

Appendice

Il codice si divide in due parti il primo con estensione .ino che è il file principale contenente le dichiarazioni delle porte I/O e le macro funzioni principali. Inoltre, è presente un altro codice con estensione .h contenente i parametri del motore, FDT dei regolatori e alcune funzioni accessorie

CodiceFinale.ino

```
#include "Control.h"
#include "src/EmgSensor.hpp"

// Pin per comandare il motore tramite motor shield (modulo A)
#define CurrentPin A0 // non utilizzato per questo codice
#define BrakePin 9 // HIGH -> blocca l'alimentazione al motore
#define PwmPin 3 // 0 -> 255
#define DirectionPin 12 // HIGH -> CW - LOW -> CCW
// Pin utilizzati per leggere l'encoder
#define EncPinA 2
#define EncPinB 13
#define HomingPin A6
// Pin utilizzati per il pulsante di cambio stato
#define ButtonPin 6
#define ButtonDelay 50 // Delay in ms inserito per evitare di campionare le fluttuazioni che seguono la pressione del pulsante
// Pin utilizzati per i finecorsa
#define FineCorsaIn 10
#define FineCorsaOut 5
#define EmergencyPin 50
// Pin utilizzati per accendere i led
#define OpenedLed 31
#define ReachedLed 33
#define ClosedLed 29
#define ButtonStop 52
#define ButtonStopChange 7
```

```

//Utilizzo di maschere per i pin dell'encoder
const unsigned int mask_EncPinA = digitalPinToBitMask(EncPinA);
const unsigned int mask_EncPinB = digitalPinToBitMask(EncPinB);

//Variabili per determinare la posizione di cmano chiusa/aperta
float ManoChiusa =4;
float ManoAperta= 0.01 ;
// Oss non imposto = 0 per problemi con divisione per 0 e per scostarsi dal finecorsa

//Variabile utilizzata per comandare il ponte H tramite Pwm e delimitarne i limiti
int PwmMax = 255;
#define PwmMin 3
#define PwmHoming 10
int PwmValue = 0;
#define ErrorLimit 0.5
#define PwmLimit 5
//bool CheckLimit = true;
#define CurPresa 0.02

//Variabile utilizzate durante le prove per plottare su monitor seriale nuovi valori solo quando il
motore è in movimento
float GiriPrev = 0;

//Variabile utilizzata per decidere se aprire o chiudere la Mano
int HandStateNew = 1; //2 = Mano pronta per essere aperta o mano in apertura
//1 = Mano pronta per essere chiusa o mano in chiusura
int HandStateOld = 1;
bool HandState = false;
//Variabili utilizzate per definire la traiettoria nei transitori di chiusura e apertura
bool TrajClosing = false;
bool TrajOpening = false;

//Variabili utilizzate per temporizzare il codice

```

```
unsigned long t4, t3, t2, t1;
```

```
unsigned long dt = 0;
```

```
// Variabile utilizzata per verificare quando è stato raggiunta la destinazione o si è trovato un ostacolo
```

```
bool Reached = true;
```

```
// //Variabile utilizzata per capire quando viene raggiunta la destinazione per la prima volta (si resetta ogni volta che viene raggiunta la destinazione)
```

```
// bool ReachedFT = true;
```

```
bool TrajFT = true;
```

```
// Variabili utilizzate per la lettura della velocità
```

```
int CntPrev = 0;          // Variabile utilizzata per conteggiare i pulse tra un intervallo e quello successivo
```

```
float SpeedRad, SpeedRpm = 0; // Variabili utilizzate per contenere la misurazione di velocità
```

```
// Variabili utilizzate per la lettura di posizione
```

```
float Error = 0;
```

```
int PulseCnt = 0;
```

```
float Giri = 0;
```

```
float PositionStop=0;
```

```
//Errore accettabile per determinare raggiunta la posizione finale
```

```
#define Eps 0.5
```

```
#define EpsAperta 50
```

```
// Intervalli in ms utilizzati per dare consistenza al codice
```

```
#define PrintTime 100    // Intervallo in ms tra una stampa seriale e la successiva
```

```
//#define ReachedTime 2000 // Intervallo in ms dopo il quale se l'Error è inferiore al valore determinato (Eps) il motore viene spento
```

```
#define TrajectoryTime 50 // Intervallo in ms tra un aumento/diminuzione di Trajectory
```

```
#define ClosingTime 2000 // Tempo richiesto per la chiusura della mano in ms
```

```
#define ClosingDelta (ManoChiusa - ManoAperta) / (ClosingTime / TrajectoryTime)
```



```

#define OpeningDelta ClosingDelta // Delta di incremento/dcremento della traiettoria

EmgSensor Emg(Serial3, 115200);

void setup() {
    Serial.begin(115200);    // Inizializzazione della comunicazione seriale
    Emg.begin();
    analogReadResolution(12); // Determinazione della risoluzione dell'ADC a 12 bit

    // Inizializzazione dei pin per la lettura dell'encoder
    REG_PIOB_PDR = mask_EncPinA; // Attivate peripheral function (disabilitate le funzioni di
    PIO)
    REG_PIOB_ABSR |= mask_EncPinA; // Scelta peripheral option B
    REG_PIOB_PDR = mask_EncPinB; // Attivate peripheral function (disabilitate le funzioni di
    PIO)
    REG_PIOB_ABSR |= mask_EncPinB; // Scelta peripheral option B

    // Attivazione del clock per TC0
    activateCNT_TC0();

    // Dichichiarazione delle porte come Input/Output
    pinMode(CurrentPin, INPUT);
    pinMode(BrakePin, OUTPUT);
    pinMode(PwmPin, OUTPUT);
    pinMode(DirectionPin, OUTPUT);

    pinMode(ButtonPin, INPUT_PULLUP);
    pinMode(ButtonStop, INPUT);
    pinMode(HomingPin, INPUT);
    pinMode(FineCorsaIn, INPUT_PULLUP);
    pinMode(FineCorsaOut, OUTPUT);
    pinMode(ButtonStopChange, INPUT);
    pinMode(EmergencyPin, INPUT);
    //digitalWrite(EmergencyPin, LOW);
    delay(500);
}

```

```

//attachInterrupt(digitalPinToInterrupt(EmergencyPin), Stop, LOW);
attachInterrupt(digitalPinToInterrupt(EmergencyPin), Stop, RISING);
pinMode (OpenedLed, OUTPUT);
pinMode (ReachedLed, OUTPUT);
pinMode (ClosedLed, OUTPUT);

// Imposizione del valore basso per evitare indesiderati reset dell'encoder
digitalWrite(FineCorsaOut, LOW);

//Homing del setup
Setpoint = 4;
}

void loop() {

    dt = (millis() - t1);
    if (dt > (Ts * 1000)) {
        VarReading();
        t1 = millis();
    }
    HandStateNew=Emg.getHandState();
    if ((HandStateNew != HandStateOld) || (digitalRead(ButtonPin) == LOW )) {
        //Serial.println("preChange");
        Change(); // Imposizione dell'obiettivo da raggiungere in base allo stato precedente
        //Serial.println("postChange");
        HandStateOld=HandStateNew;
    }
    if (digitalRead(ButtonStop) == HIGH) {

        StopPosition();
    }
    //if (digitalRead(ButtonStopChange) == HIGH) {
    //StopPosition1();
    //}
}

```

```

if ((millis() - t2) > TrajectoryTime) {
    Setpoint = Trajectory(Setpoint); // Creazione di una traiettoria a trapezio per evitare brusche
risposte a seguito di ingressi a gradino
    t2 = millis();
}

if (RegEnable) {
    if ((millis() - t3) > (Ts * 1000)) {
        LedStatus(HandState, Reached);    // Segnalazione dello stato del sistema tramite led
        CheckReached(Giri, Error, AvgCur); // Verifica di aver raggiunto la destinazione desiderata
        Output = Regulators(Setpoint, Giri, SpeedRad, AvgCur);
        PwmValue = MotorAct(Output);      // Calcolo del valore PWM da applicare al motore
        analogWrite(PwmPin, PwmValue);    // Azionamento del motore
        t3 = millis();
    }
}

//Stampa su monitor seriale (solo se sono passati almeno PrintTime ms e se il motore si è
mosso)
if (((millis() - t4) > PrintTime) && (Giri != GiriPrev)) {
    MonitorSeriale(Setpoint, Giri, Error, Output, PwmValue, Current, AvgCur, SpeedRpm);
    t4 = millis();
    GiriPrev = Giri;
}

/*if (digitalRead(ButtonStopChange) == HIGH) {
    Change();
} */

}

// Funzione che permette la corretta lettura delle variabili di Posizione, velocità e corrente
void VarReading() {
    // Lettura posizione

```

```

PulseCnt = REG_TC0_CV0;          // Lettura dei ppr dell'encoder

Giri = (float)PulseCnt / EncoderQuad; // Conversione da ppr a giri del motore (a valle del
riduttore)

// Lettura velocità
SpeedRad = SpeedCalc(dt, PulseCnt, CntPrev);
SpeedRpm = SpeedRad * RadToRpm;
CntPrev = PulseCnt;

// Lettura della corrente
Current = (((float)analogRead(CurrentPin) / 4095) * 2);

AvgCur = AveregeCurrent(Current); // Valore medio della corrente calcolato per evitare
letture spurie

Error = Setpoint - Giri; // Errore utilizzato per test di funzionamento
}

// Funzione che permette di imporre il setpoint di posizione desiderato
void Change() {
    // Se la mano è chiusa avvio l'apertura
    if (HandState == true) {
        //Serial.println("Apro");
        TrajOpening = true;
    } else { // Se la mano è aperta avvio la chiusura
        //Serial.println("Chiudo");
        TrajClosing = true;
    }
    RegEnable = true; // Azionamento del controllo
    delay(ButtonDelay); // Attesa per evitare di leggere le fluttuazioni del pulsante
    digitalWrite(BrakePin, LOW);
    Reached = false;
    CheckLimit = false;
    LimitFT = true;

    // E' necessario sapere quanto tempo è passato dalla prima richiesta
    /*if (TrajFT) {
        t3 = millis();

```

```

    TrajFT = false;
  }*/
}

// Funzione che permette di creare una traiettoria a trapezio
double Trajectory(double Setpoint) {
  if (TrajOpening) {
    if (Setpoint > (ManoAperta + OpeningDelta)) {
      // Ogni TrajectoryTime ms se il set point != mano aperta, viene decrementato il setpoint
      // attuale
      Setpoint = ManoAperta;
    } else {
      // set point = mano aperta
      Setpoint = ManoAperta;
      TrajOpening = false;
      TrajFT = true;
      CheckLimit = true;
    }
  } else if (TrajClosing) {
    if (Setpoint < (ManoChiusa - ClosingDelta)) {
      // Ogni TrajectoryTime ms se il set point != mano chiusa, viene incrementato il setpoint
      // attuale
      Setpoint = ManoChiusa;
    } else {
      // set point = mano chiusa
      Setpoint = ManoChiusa;
      TrajClosing = false;
      TrajFT = true;
      CheckLimit = true;
    }
  }
  return Setpoint;
}

```

// Funzione che permette di accendere i led per monitorare lo stato del sistema

```

void LedStatus(bool HandState, bool Reached) {

    if(HandState == true) {
        // Mano aperta in chiusura
        digitalWrite(OpenedLed, HIGH);
        digitalWrite(ClosedLed, LOW);
    } else {
        // Mano chiusa o in apertura
        digitalWrite(ClosedLed, HIGH);
        digitalWrite(OpenedLed, LOW);
    }

    if (Reached) {
        digitalWrite(ReachedLed, HIGH);
    } else
    {
        digitalWrite(ReachedLed, LOW);
    }
}

// Funzione che verifica il corretto raggiungimento del setpoint richiesto
void CheckReached(float Giri, float Error, float Current) {
    // Se la mano è in apertura e l'errore è inferiore a epsilon
    if ( (HandState == false && ((-EpsAperta <= (Giri / ManoAperta)) && ((Giri / ManoAperta) <=
EpsAperta)) ) || Current >= CurPres ) {
        //Serial.println("Aperto");
        HandState = true; // Mano pronta per essere chiusa
        Reached = true;
        //RegEnable = false; // Interrompo il calcolo delle variabili del Pid
    }
    // Se la mano è in chiusura e l'errore è inferiore a epsilon
    if ( (HandState == true && ((-Eps <= (Giri / ManoChiusa) - 1) && ((Giri / ManoChiusa) - 1 <=
Eps)) ) || Current >= CurPres ) {
        //Serial.println("Chiuso");
        HandState = false; // Mano pronta per essere aperta
    }
}

```

```

    Reached = true;
    // RegEnable = false; // Interrompo il calcolo delle variabili del Pid
}
}

// Funzione che determina il valore PWM da applicare al motore ricevuta l'uscita del regolatore
di corrente
int MotorAct(double out) {

    // Determinazione della direzione di rotazione
    if (out > 0) {
        digitalWrite(DirectionPin, LOW);
        //Serial.println("Gira1");
    } else {
        digitalWrite(DirectionPin, HIGH);
        //Serial.println("Gira2");
    }

    // Calcolo il valore PWM da fornire al motore
    int Pwm = map(fabs(out), 0, MotorVin, 0, PwmMax);

    // Imposizione della dinamica a bassa velocità se ci trova nei pressi della destinazione
    if ( ( fabs(Error) < ErrorLimit) && CheckLimit ) {
        TauPos = 0.2;
        TauVel = 0.5;
        CurTau = 0.01;
        PwmMax = PwmLimit;
    } else {
        TauPos = 0.2;
        TauVel = 0.002;
        CurTau = 0.0001;
        PwmMax = 255;
    }
}

```

```

if (Pwm < PwmMin) {
    Pwm = PwmMin;
}

```

```

return Pwm;
}

```

// Funzione che arresta immediatamente il motore qual'ora si raggiunga il finecorsa oltre alla posizione di mano completamente chiusa

```

void Stop() {
    Serial.println("666");
    analogWrite(PwmPin, 0);
    digitalWrite(BrakePin, HIGH);
    Serial.println("Emergency stop");
    while (1) {
        delay(1000);
    }
}

```

```

void StopPosition() {

    analogWrite(PwmPin, 0);
    digitalWrite(BrakePin, HIGH);
    Serial.println("Stop Position");
    PositionStop=Giri;
    ManoChiusa=PositionStop;
}

```

```

void StopPosition1() {

    analogWrite(PwmPin, 0);
    digitalWrite(BrakePin, HIGH);
    Serial.println("Stop Position");
    PositionStop=Giri;
    ManoAperta=PositionStop;
}

```



```
}
```

Control.h

```
#ifndef Control_h
```

```
#define Control_h
```

```
bool CheckLimit = true;
```

```
bool LimitFT = false;
```

```
#define TWO_PI 6.283185307179586476925286766559
```

```
// Parametri del motore
```

```
#define La (180*pow(10, -6)) // [H]
```

```
#define Ra 4.09 // [Ohm]
```

```
#define MotorVin 12 // [V]
```

```
#define GearRed 152
```

```
#define EncoderPpr 1024 // Pulse per revolution dell'encoder
```

```
#define MotorPpr (EncoderPpr * GearRed) // Pulse per revolution a valle del riduttore
```

```
#define EncoderQuad (MotorPpr * 4) // L'encoder in quadratura conta 4 fronti per ogni pulse
```

```
#define RotorInertia 3.8 // [gcm^2]
```

```
#define TorqueK 16 // [mNm/A]
```

```
#define MechTimeK 6 // [ms]
```

```
#define J (RotorInertia * pow(10, -7)) // [KgM^2]
```

```
#define Km (TorqueK * pow(10, -3)) // [Nm/A]
```

```
#define TauMech (MechTimeK * pow(10, -3)) // [s]
```

```
#define b (J/TauMech) // [Nm^2/OhmA^2] or [KgM^2/us]
```

```

// Costanti di conversione per le varie unità di misura
#define RadToRpm (60/TWO_PI)
#define RpmToRad (TWO_PI/60)
#define RadToTurn ( (180/(TWO_PI/2)) / 360 )
#define TurnToRad (1/RadToTurn)
#define RadPerCount (TWO_PI/(EncoderPpr*4) )

// Tempo utilizzato per la discretizzazione delle equazioni
#define Ts pow(10, -6)

// Variabili utilizzate per la lettura di corrente
#define ArraySize 50 // Dimensione dell'array per il calcolo del valore medio di corrente
float ArrayCur[ArraySize]; // Definizione dell'array
int i = 0; // Contatore utilizzato per scorrere l'array
float Current = 0;
float AvgCur = 0;

// Struttura che contiene tutti i dati necessari per i regolatori PID
struct Reg {
    float Tau, Kp, Kd, Ti, Sat, In, InPrec, Out, OutPrec;
};

// Costanti di tempo dei regolatori
float TauPos = 0.2;
float TauVel = 0.002;
float CurTau = 0.0001;

// Valori di saturazione delle variabili dei regolatori
#define SpeedSat (6000*RpmToRad)
#define CurSat 0.4
#define VoltSat 11

// Creazione dei vari regolatori PD e PI
struct Reg PosReg = {TauPos, 1/TauPos, 0, 0, SpeedSat, 0, 0, 0, 0};

```

```

struct Reg SpeedReg = {CurTau*5, J/(TauVel*Km), 0, TauMech, CurSat, 0, 0, 0, 0};
struct Reg CurReg = {CurTau, La/CurTau, 0, La/Ra, VoltSat, 0, 0, 0, 0};

// Variabili utilizzate per come riferimento o output funzionamento del controllo
double Setpoint, Output;

// Variabili utilizzate nella fase di test per aver traccia delle richieste dei vari regolatori
double CurRef, SpeedRef = 0;

// Variabile utilizzata per attivare o meno il controllo
bool RegEnable = false;

double Regulators(double SetPoint, float Giri, float OmegaRad, float Current) {

    PosReg.In = (((SetPoint - Giri) * GearRed) * TurnToRad);

    // ANELLO DI POSIZIONE

    PosReg.Out = (PosReg.In * (PosReg.Kp + ((2 * PosReg.Kd) / Ts)) + PosReg.InPrec *
(PosReg.Kp - ((2 * PosReg.Kd) / Ts)) - PosReg.OutPrec);

    if (PosReg.Out > PosReg.Sat) {
        PosReg.Out = PosReg.Sat;
        PosReg.In = ((1 / (2 * PosReg.Kd + PosReg.Kp * Ts)) * (Ts * (PosReg.Out +
PosReg.OutPrec) + 2 * PosReg.Kd * PosReg.InPrec));
    } else if (PosReg.Out < -PosReg.Sat) {
        PosReg.Out = -PosReg.Sat;
        PosReg.In = ((1 / (2 * PosReg.Kd + PosReg.Kp * Ts)) * (Ts * (PosReg.Out +
PosReg.OutPrec) + 2 * PosReg.Kd * PosReg.InPrec));
    }

    PosReg.OutPrec = PosReg.Out;
    PosReg.InPrec = PosReg.In;

    //ANELLO DI VELOCITA'

    SpeedReg.In = (PosReg.Out - OmegaRad);
    SpeedRef = PosReg.Out*RadToRpm;

```

```

    SpeedReg.Out = (SpeedReg.In * SpeedReg.Kp * ((Ts / (2 * SpeedReg.Ti)) + 1) +
    SpeedReg.InPrec * SpeedReg.Kp * ((Ts / (2 * SpeedReg.Ti)) - 1) + SpeedReg.OutPrec);
    if (SpeedReg.Out > SpeedReg.Sat) {
        SpeedReg.Out = SpeedReg.Sat;
        SpeedReg.In = (1 / (SpeedReg.Kp + Ts / (2 * SpeedReg.Ti)) * (SpeedReg.Out -
        SpeedReg.OutPrec - SpeedReg.InPrec * (Ts / (2 * SpeedReg.Ti) - SpeedReg.Kp)));
    } else if (SpeedReg.Out < -SpeedReg.Sat) {
        SpeedReg.Out = -SpeedReg.Sat;
        SpeedReg.In = (1 / (SpeedReg.Kp + Ts / (2 * SpeedReg.Ti)) * (SpeedReg.Out -
        SpeedReg.OutPrec - SpeedReg.InPrec * (Ts / (2 * SpeedReg.Ti) - SpeedReg.Kp)));
    }
    SpeedReg.OutPrec = SpeedReg.Out;
    SpeedReg.InPrec = SpeedReg.In;

```

//ANELLO DI CORRENTE

```
CurReg.In = (SpeedReg.Out - Current);
```

```
CurRef = SpeedReg.Out;
```

```

    CurReg.Out = (CurReg.In * CurReg.Kp * ((Ts / (2 * CurReg.Ti)) + 1) + CurReg.InPrec *
    CurReg.Kp * ((Ts / (2 * CurReg.Ti)) - 1) + CurReg.OutPrec);
    if(LimitFT){
        CurReg.Out = 0;
        LimitFT = false;
    }
    if (CurReg.Out > CurReg.Sat) {
        CurReg.Out = CurReg.Sat;
        CurReg.In = (1 / (CurReg.Kp + Ts / (2 * CurReg.Ti)) * (CurReg.Out - CurReg.OutPrec -
        CurReg.InPrec * (Ts / (2 * CurReg.Ti) - CurReg.Kp)));
    } else if (CurReg.Out < -CurReg.Sat) {
        CurReg.Out = -CurReg.Sat;
        CurReg.In = (1 / (CurReg.Kp + Ts / (2 * CurReg.Ti)) * (CurReg.Out - CurReg.OutPrec -
        CurReg.InPrec * (Ts / (2 * CurReg.Ti) - CurReg.Kp)));
    }
    CurReg.OutPrec = CurReg.Out;
    CurReg.InPrec = CurReg.In;

```

```

    return CurReg.Out;
}

// Funzione che inizializza il contatore in quadratura
void activateCNT_TC0() {

    REG_PMC_PCER0 = (1 << 27); // Attivazione del clock per TC0

    // Selezione di XC0 come fonte di clock e impostazione in captrue mode del contatore
    // Riga di codice che inizializza il contatore ad un fronte di salita del pin A6
    REG_TC0_CMR0 = (1 << 0) | (1 << 2) | (1 << 8) | (1 << 10);
    //REG_TC0_CMR0 = 5; // Riga di codice che non ha il reset del counter

    // Attivazione del contatore dell'encoder in quadratura in position measure mode senza filtri
    REG_TC0_BMR = (1 << 8) | (1 << 9) | (1 << 12) | (1 << 13);

    // Attivazione del clock (CLKEN=1) e reset del counter (SWTRG=1)
    // SWTRG = 1 è necessario per attivare il clock
    REG_TC0_CCR0 = 5;
}

// Funzione utilizzata per calcolare il valore della velocità del motore
float SpeedCalc(unsigned long dt, int PulseCnt, int CntPrev) {
    int DeltaCnt = PulseCnt - CntPrev;
    float DeltaThetaMis = DeltaCnt * RadPerCount;

    float SpeedRad = (DeltaThetaMis / (dt) ) * 1000;
    //SpeedRad = ((TWO_PI * DeltaCnt) / (EncoderQuad * dt) * 1000);
    return SpeedRad;
}

```

// Funzione utilizzata per calcolare il valore medio di corrente assorbita

```
float AveregeCurrent(float Current) {
```

```
    // Riempimento dell'array con i valori di corrente
```

```
    ArrayCur[i] = Current;
```

```
    if (i < (ArraySize - 1)) {
```

```
        i++;
```

```
    } else {
```

```
        i = 0;
```

```
    }
```

```
    float AvgCur = 0;
```

```
    //Somma di tutti i valori presenti nell'array
```

```
    for (int j = 0; j < ArraySize; j++) {
```

```
        AvgCur += ArrayCur[j];
```

```
    }
```

```
    AvgCur /= ArraySize; // Valore medio all'interno dell'array
```

```
    return AvgCur;
```

```
}
```

// Funzione utilizzata per stampare sul monitor e plotter seriale

```
void MonitorSeriale(double Setpoint, float Giri, float Error, double Output, int PwmValue, float Current, float AvgCur, float SpeedRpm) {
```

```
    //Monitor seriale
```

```
    Serial.print("Setpoint:");
```

```
    Serial.print(Setpoint, 3);
```

```
    Serial.print("\tGiri:");
```

```
    Serial.print(Giri, 3);
```

```
    Serial.print("\tError:");
```

```
    Serial.print(Error, 4);
```

```
    Serial.print("\tCheckLimit:");
```

```
Serial.print(CheckLimit);

Serial.print("\tOutput:");
Serial.print(Output);
Serial.print("\tPwmValue:");
Serial.print(PwmValue);
// Serial.print("\tCurrent:");
// Serial.print(Current);
Serial.print("\tAverage current:");
Serial.print(AvgCur, 4);
// Serial.print("\tSpeedRef:");
// Serial.print(SpeedRef, 4);
// Serial.print("\tCurrentRef:");
// Serial.print(CurRef, 4);
Serial.print("\tSpeed in rpm:");
Serial.println(SpeedRpm, 4);
}
#endif
```

Bibliografia

[1] [Online]. Available: <https://www.montecatone.com/> .

[2] A. Mohammadi, J. Lavranos, P. Choong e D. Oetomo, «Flexo-glove: A 3D Printed Soft xoskeleton Robotic Glove for Impaired Hand Rehabilitation and Assistance,» 40th Annual International Conference of the IEEE Engineering in Medicine and Biology , 2018.

[3] T. Triwiyanto, E. Yulianto, M. R. Mak'ruf, D. Titisari, T. Rahmawati, S. Luthfiyah, T. Hamzah, S. Syaifudin e I. D. G. H. Wisana, «A Review on Robotic Hand Exoskeleton Devices: State-of-the-Art Method».

[4] B. B. Kang, S.-S. Yun e K.-J. Cho, «Exo-Glove PM: An Easily Customizable Modularized Pneumatic Assistive Glove. IEEE Robotics and Automation Letters,» vol. 2, n. 3, 2017.

[5] H. K. Yap, J. C. H. Goh e R. C. H. Xeov, «Characterization of a Soft Bending Actuator for Rehabilitation Application.,» 2014.

[6] Y.-J. Lai, L.-J. Yeh e M.-C. Chiu, «An experimental investigation on shape memory alloy dynamic splint for a finger joint application. Sensors and Actuators .,» vol. A, n. 173, pp. 210-218, 20

[7] Z. Yao, C. Linnenberg, A. A.-Wollensen, R. Weidner e J. P. Wulfsberg, «Biomimetic design of an ultra-compact and light-weight soft muscle glove.,» Vol. %1 di %210.1007/s11740-017-0767-y., 2017.

[8] M. Liu, Z. Zhao, W. Zhang e L. Hao, «Reinforcement learning control of a humanoid robotic hand actuated by shape memory alloy.,» vol. 10.1177/0954406220982019, 2021.

[9] Y. Wang, S. Zheng, J. Pang, S. Li e J. Li., «Design and Experiment of a Hand Movement Device Driven by Shape Memory Alloy Wires. Journal of Robotics.,» vol. 2021, 2021.

[10] B. B. Kang e H. C.-J., «Exo-Glove Poly II: A Polymer-Based Soft Wearable Robot for the Hand with a Tendon-Driven Actuation System. Soft Robot.,» vol. 10.1089/soro.2018.0006, 2019.

- [11] Mohammadi e J. L., «Elexo-Glove: A 3D Printed Soft Exoskeleton Robotic Glove for Impaired Hand Rehabilitation and Assistance. Annual International Conference of the IEEE Engineering in Medicine 10.1109/EMBC.2018.8512617, 2018. and Biology Society.,» vol.
- [12] X. Chen, L. Gong, LuWei, S.-C. Yeh, L. D. Xu, L. Zheng e Z. Zou, «A Wearable Hand Rehabilitation System with Soft Gloves.,» IEEE Trans. Ind Informatiss , 2021.
- [13] T. Zürich e R. Tenoexo, «A Robotic Hand Orthosis for Therapy and Assistance in Activities of Daily Living».
- [14] W. Williams, «Fimoxe Assist Hand Exoskeleton. Category: Assistive Robotic Glove, Mobility,» 2022.
- [15] W. Williams, «NUADA Glove. Category: Assistive Robotic Glove, Mobility.,» 2021. 81
- [16] «Bioserwo Strengh for Life. Carbonhand; get a grip.».
- [17] W. Williams, «Neomane, Glove. Category: Assistive Robotic Glove, Mobility.,» 2021.
- [18] E. Valeri, I. D. Paola, M. Paltrinieri, A. Eriz, X. Hajat e M. M. Gilbert, «SMAtronis: guanto robotico per la riabilitazione della mobilità manuale in pazienti tetraplegici attuato tramite tecnologia SMA.,» 2022.
- [19] M. Tarsi, «GLOVE: progettazione di un dispositivo elettrico per la abilitazione della mano di pazienti tetraplegici.,» 2021.
- [20] E. Valeri, «GLOVE: Prototipo di un esoscheletro per la abilitazione della mano di pazienti tetraplegici.,» 2022.
- [21] M. M. Gilbert, «GLOVE: Prototipo di un esoscheletro per la assistenza al movimentodella mano di pazienti tetraplegici,» 2022.
- [22] [Online]. elettrico/. Available: <https://guide.directindustry.com/it/come-scegliere-un-motore>
- [23] [Online]. Available: <https://www.faulhaber.com/it/prodotti/a-spazzole/micromotori-cc/>.

[24] C. Florian, «Convertitori in commutazione Caratteristiche degli interruttori controllabili. Dissipazione di potenza in commutazione.,» 2020.