

UNIVERSITÀ
DEGLI STUDI
DI PADOVA



DIPARTIMENTO DI TECNICA E GESTIONE DEI SISTEMI INDUSTRIALI
CORSO DI LAUREA IN INGEGNERIA MECCATRONICA

TESI IN:

LEG MECHANISM

LAUREANDI:

- Anass Berradi
- Luca Dalle Fusine
- Gregorio Lofoco
- Arlind Zheli

RELATORE:

Prof Ing. Paolo Boscariorl

CORRELATORE:

Prof Ing. Iacopo Tamellin

ANNO ACCADEMICO 2022/2023

Indice

1	INTRODUZIONE	5
1.1	STRANDBEEST	7
1.2	TROTBOT	7
1.3	STRIDER	8
1.4	KLANN	8
2	OBIETTIVI	9
3	AMBIENTI DI SIMULAZIONE	11
4	PROBLEMA COMPARABILITÀ	13
5	ALTEZZA PASSO	15
6	VELOCITÀ E VARIANZA DEL MECCANISMO	17
7	STABILITÀ CENTRO DI MASSA	21
8	COSTO ENERGETICO	25
9	ANALISI DI POSIZIONE	33
9.1	TROTBOT	34
9.2	STRIDER	39
9.3	KLANN	45
9.4	STRANDBEEST	50
10	CONCLUSIONI E CONFRONTO FINALE	55
11	SITOGRAFIA	57

Capitolo 1

INTRODUZIONE

LEG MECHANISM

I **leg mechanism** sono particolari sistemi meccanici progettati per permettere il moto planare, basandosi su un contatto intermittente che genera una forza propulsiva di avanzamento.

Nel mondo della robotica esistono altri sistemi meccanici sicuramente a noi più familiari, come ad esempio le ruote e i cingoli, che però garantiscono il moto mediante un contatto continuo. In entrambi i casi, è fondamentale l'attrito.

Rispetto alle ruote, i leg mechanism hanno la possibilità di muoversi su terreni sconnessi, il che lo rende il più grande vantaggio.

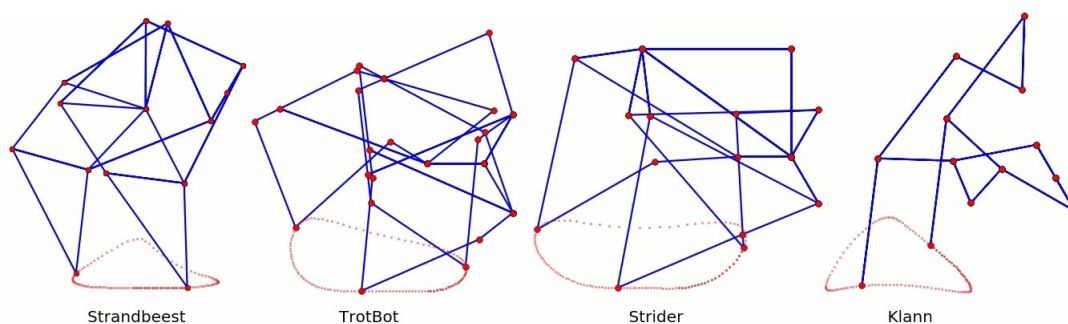


Nel nostro caso andremo ad analizzare dei progetti molto semplici, dove i meccanismi sono composti da un **telaio**, un **motore**, e da **travi** (ovvero strutture portanti elementari dove la dimensione longitudinale prevale sulle altre due dimensioni) connesse mediante **coppie rotoïdali**.

DIY WALKERS

Il primo progetto di meccanismo a camminata nasce a fine del XIX secolo grazie al matematico russo Pafnuty Chebyshev, ma ad oggi il numero di progetti non è vastissimo. Quelli più interessanti sono stati raccolti dalla **DIY Walkers** (team di ricercatori specializzati nella costruzione di tali meccanismi mediante l'uso dei LEGO), e più in dettaglio sono:

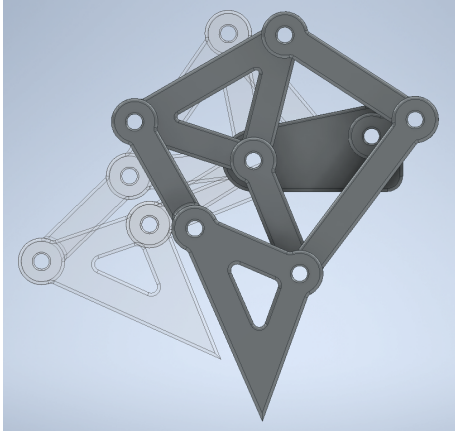
- Strandbeest
- Trotbot
- Strider
- Klann



Come notiamo dall'immagine sopra riportata (raffigurante solo le singole gambe), le traiettoria disegnata dal piede di ciascuno è completamente differente, anche se i meccanismi presentano diversi aspetti in comune, e ciò è dovuto alle diverse configurazioni. Ma quali sono questi aspetti? Innanzitutto i 4 meccanismi hanno **movimenti limitati** solo in avanti e all'indietro, per via della presenza di un solo motore che comanda tutte le gambe contemporaneamente (per avere la sterzata è necessario che le gambe ruotino a velocità diverse).

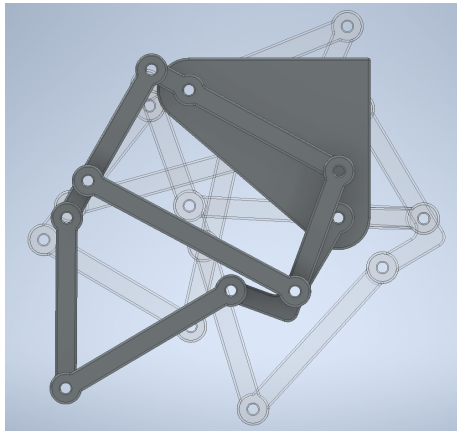
Altro aspetto riguarda invece la collocazione di tale motore, che noi definiremo come **centro di massa**, situata sempre all'interno del meccanismo, rendendo di fatto l'insieme piuttosto stabile. Infine se andiamo ad analizzare la traiettoria, non a caso noteremo che la **velocità nella fase di appoggio** è inferiore e poco variabile, mentre quando avviene lo stacco il piede assume una velocità superiore.

1.1 STRANDBEEST



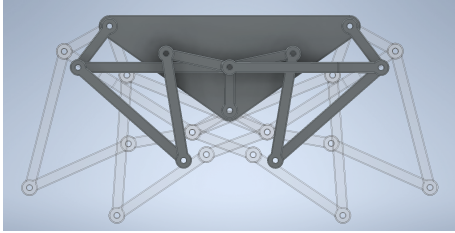
Il nome del meccanismo significa letteralmente "animali da spiaggia", e in realtà va ad identificare tutta una famiglia di progetti artistico-ingegneristici create dall'olandese Theo Jansen. Il meccanismo che noi analizzeremo deriva da un processo evolutivo simulato mediante algoritmi su computer, con l'obiettivo di andare a creare il meccanismo migliore per la designazione di una traiettoria a triangolo, e per tale motivo ha meritato di portare il nome di tutta la famiglia. Creato inizialmente con il fine di farlo muovere con la sola spinta del vento, la DIY Walkers lo adatta al mondo meccanico rendendolo di fatto un meccanismo di locomozione. Lo Strandbeest presenta 4 gambe che ruotano a coppie sfasate di 180° , ed ogni gamba presenta 11 travi, 8 giunti e 4 maglie.

1.2 TROTBOT



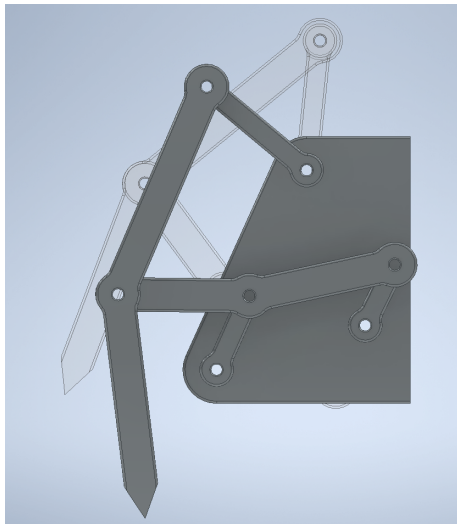
Noi andremo ad analizzare la cinematica della prima versione, infatti ne esistono di più recenti con la semplice aggiunta di tallone per un miglior meccanismo. Prende il nome da trotto, visto la somiglianza con l'andatura del cavallo, ed è stato completamente progettato dalla DIY Walkers. Il trotbot presenta 6 gambe che ruotano a coppie sfasate di 120° , ed ogni gamba presenta 8 travi, 9 giunti e 2 maglie.

1.3 STRIDER



Anche questo meccanismo è frutto del lavoro di DIY Walkers, e noi ne analizzeremo solo la prima versione. Tra i 4 meccanismi è sicuramente il più semplice dal punto di vista realizzativo, ha una velocità più costante ma sopporta carichi minori. Lo strider presenta 6 gambe che ruotano a coppie sfasate di 120° , ed ogni gamba presenta 9 travi, 10 giunti e 4 maglie, ma da notare che la gamba ha due appoggi rendendola così di fatto doppia.

1.4 KLANN



Progetto risalente al 1994, prende il nome dal suo creatore Joe Klann, e va a simulare l'andatura del ragno. Tale meccanismo è risultato dell'espansione delle curve Burmester. Il Klann presenta 4 gambe che ruotano a coppie sfasate di 180° , ed ogni gamba presenta 5 travi, 7 giunti e 2 maglie, ma da notare che la gamba ha due appoggi rendendola così di fatto doppia.

Capitolo 2

OBIETTIVI

L'obiettivo di tale tesi è quella di andare a comparare i quattro meccanismi sopra descritti, andando ad analizzare i parametri per noi più significativi, ovvero:

- Altezza passo
- Velocità meccanismo
- Varianza velocità
- Costo energetico
- Stabilità centro di massa

Ottenuti dei valori numerici, sarà poi nostro compito andare a rappresentarli qualitativamente su un **grafico radar**, con l'obiettivo di andare ad individuarne uno migliore rispetto agli altri.

FASI DI LAVORO

Inizialmente è stata affrontata la **fase di orientamento**, andando ad affrontare tutto ciò che è la ricerca bibliografica, e cercando di inquadrare le qualità e le caratteristiche di tali meccanismi (siti internet, video youtube, ricerche universitarie e brevetti da google scholar).

In un secondo momento abbiamo intrapreso la **fase centrale**. Prima di tutto è stata fatta un'analisi e una scelta dei mezzi (programmi informatici) più adatti ad affrontare tale studio, dopodichè siamo andati a simulare via codice e via costruzione virtuale i meccanismi. Tale fase si conclude con l'estrapolazione dei dati da tali mezzi e la loro elaborazione mediante foglio di calcolo.

Nella **fase finale**, raccolti i diversi dati di nostro interesse (alcuni per estrapolazione diretta dai programmi e altri per elaborazione numerica) li abbiamo raccolti in un'unica tabella e rappresentati nel grafico come da nostro obiettivo. Si conclude il tutto con la stesura della tesi e il commento dei risultati.

Capitolo 3

AMBIENTI DI SIMULAZIONE

Per il raggiungimento del nostro obiettivo, come abbiamo specificato nel paragrafo "fasi di lavoro", abbiamo dovuto inizialmente scegliere i programmi più adatti in questa ricerca.

PROGRAMMI

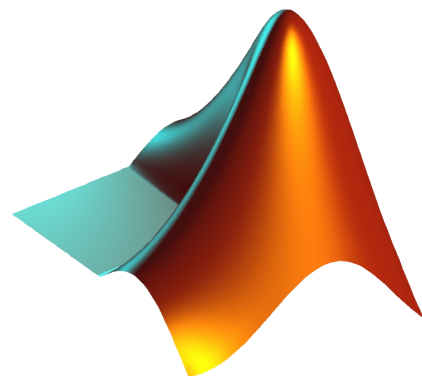
Il problema iniziale è stato come simulare la cinematica di un meccanismo, quindi creare l'assieme e muoverlo, e allo stesso tempo poter estrapolare le misurazioni necessarie (come angoli e velocità) con affidabilità e precisione; la soluzione migliore a tale problema è stato il mondo dei software per la progettazione 3D attraverso il disegno computerizzato (CAD), tra i quali a nostra scelta INVENTOR E SOLIDWORKS.

Inizialmente abbiamo approfondito l'utilizzo di questi due software, andando a creare due meccanismi su uno e due meccanismi sull'altro, ma l'estrema similitudine delle dinamiche di lavoro ci ha spinto sulla scelta del solo utilizzo di **INVENTOR**, a nostro avviso più veloce e intuitivo.

Il secondo problema sorto è come poter verificare la validità dei dati ottenuti: era nostro interesse cercare un secondo supplemento, che magari avesse una dinamica diversa dalla progettazione 3D, ma che allo stesso tempo ci desse gli stessi risultati. A tal proposito abbiamo fatto affidamento al software **MATLAB**, ambiente di calcolo numerico con dinamica completamente diversa.



(a) Logo INVENTOR



(b) Logo MATLAB

INVENTOR

È un software di progettazione meccanica 3D, dotato di modulo per la **creazione di parti** (che abbiamo usato per la creazione virtuale di ciascuna trave) e di un modulo per la **creazione di assemblaggi** (che abbiamo usato per creare gli accoppiamenti rotoidali e di fatto l'assieme). In oltre il software permette la simulazione del movimento mediante l'inserimento di un motore virtuale, il tracciamento di grafici in funzione del tempo, e l'acquisizione di dati come posizioni, angoli e velocità.

Durante la simulazione, vista la periodicità con cui si ripete il movimento del meccanismo, i dati sono stati acquisiti soltanto durante un giro di manovella (corrispondente ad un passo completo), per un totale di 1000 campioni (ogni campione dotato di 5 cifre dopo la virgola).

MATLAB

È una piattaforma di programmazione e calcolo numerico, e permette la creazione di modelli, l'analisi di dati e il tracciamento di grafici.

Su questo software abbiamo creato i meccanismi come fossero semplicemente un **insieme di vettori** che uniscono due punti, e imponendo poi il movimento di tali punti siamo riusciti a simulare la cinematica di ogni meccanismo. A tal proposito è stato fondamentale fare un'analisi di posizione, imporre delle condizioni iniziali e simulare il movimento con dei cicli (infatti non avevamo un movimento continuo, ma l'unione di più configurazioni).

Il numero di acquisizione dei dati su un giro di manovella corrisponde al numero di scatti, e visto che abbiamo considerato il tempo di rotazione pari a 2π , per una **periodo di campionamento** pari a $T = 0.01s$, abbiamo ottenuto un totale di 628 campioni.

NOTA: aumentare la frequenza di campionamento comportava tempi di simulazione molto lunghi, ma precisione dei dati finali irrilevante al nostro scopo.

INTERSCAMBIABILITÀ

Durante tutta la analisi abbiamo effettuato le stesse misurazioni su entrambi i programmi, così da poterli compararli e avere l'esattezza dei risultati ottenuti. Per avere tale sicurezza abbiamo inizialmente tracciato grafici di angoli e posizioni casuali, li abbiamo sovrapposti e abbiamo verificato che fossero identici.

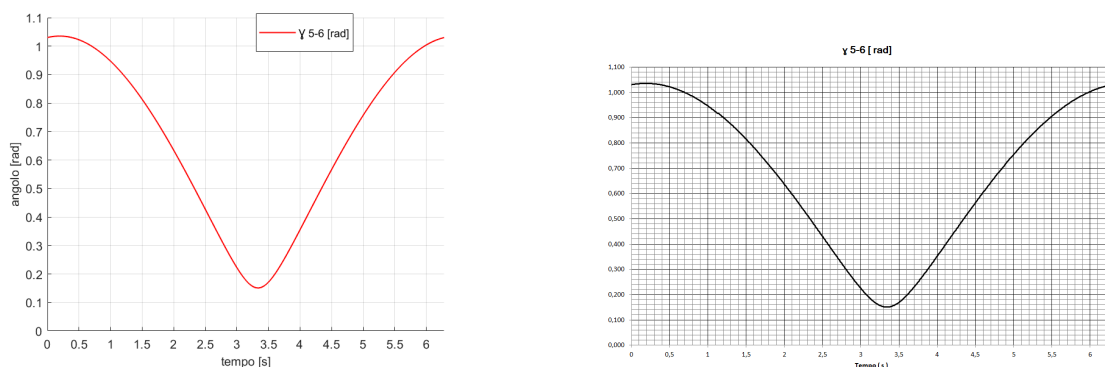


Figura 3.2: Entrambi i grafici riportano l'andamento della posizione angolare della stessa coppia rotoidale

Capitolo 4

PROBLEMA COMPARABILITÀ

Nello studio dei 4 meccanismi sono state assunte come **dimensioni** quelle **standard**. Come visibile da figura sottostante sorge un problema di comparabilità, ha senso paragonare la velocità di un ragno alla velocità di un cavallo? Per esaminare i vari parametri è stato quindi necessario relazionare i parametri dipendenti dalla grandezza del meccanismo al proprio ingombro.

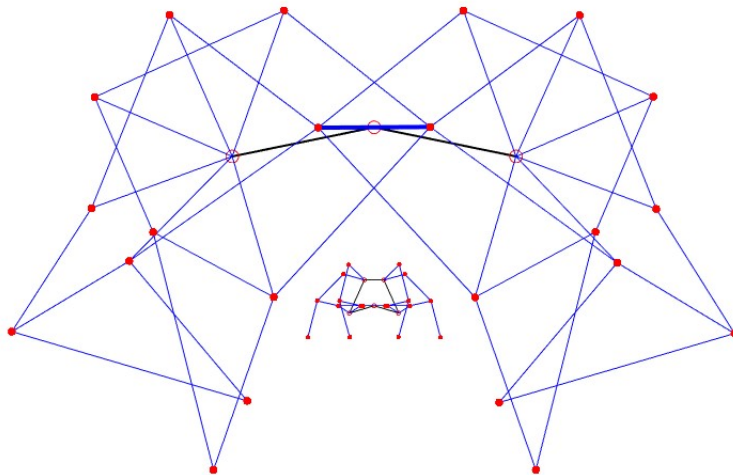
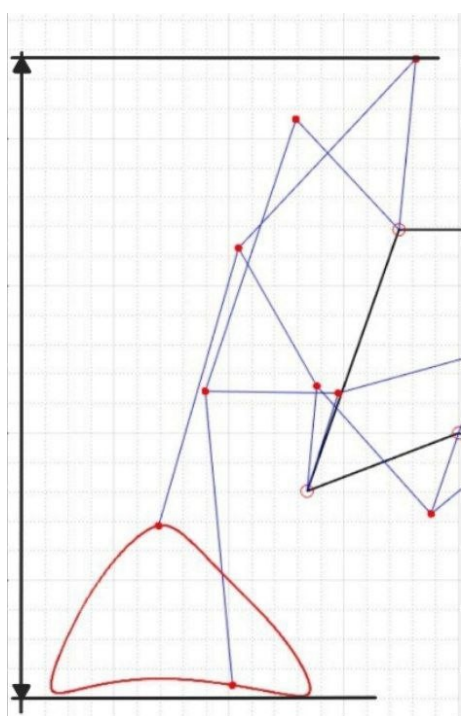


Figura 4.1: Confronto tra Strendbeest e klann

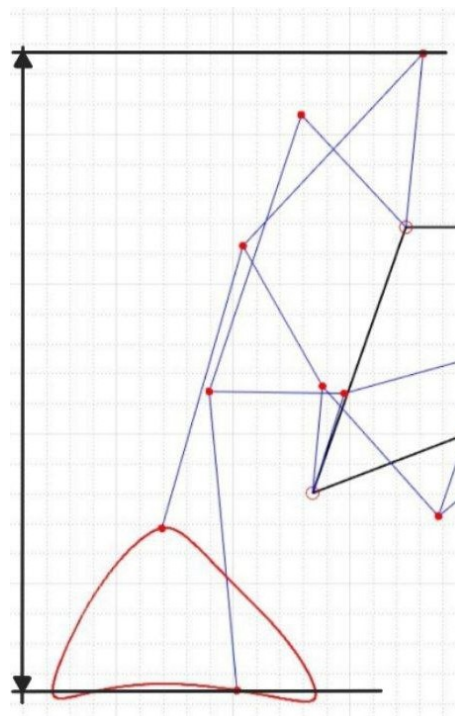
Come **ingombro** abbiamo considerato la dimensione del meccanismo **in altezza**.

VERSIONI DI CALCOLO DELL'INGOMBRO

Inizialmente l'ingombro è stato calcolato tramite la differenza tra le coordinate in y rispettive del punto massimo che il meccanismo è in grado di raggiungere e il punto più basso del percorso tracciato dal piede Fig.(a). Abbiamo notato però che effettuando questo calcolo stavamo andando in contro ad un'impresione in quanto i due punti non venivano raggiunti nello stesso istante. Tramite Matlab è stato quindi calcolato, ad ogni periodo di campionamento, la differenza tra la coordinata in y del punto massimo del meccanismo (che in alcuni casi potrebbe coincidere con un punto a telaio) e il punto più basso in quel stesso istante (che corrisponderà al piede appoggiato a terra) Fig.(b).



(a) Versione 1



(b) Versione 2

Sono di seguito riportati i valori dell'ingombro ottenuti per ciascun meccanismo con il metodo descritto precedentemente:

	Klann	Strandbeest	Strider	Trotbot
INGOMBRO	21.4194	125.559	18.1133	18.9525

Tabella 4.1: Valori degli ingombri calcolati

Capitolo 5

ALTEZZA PASSO

L'altezza del piede è da intendere come la **massima altezza di superamento ostacolo** che i meccanismi sono in grado di raggiungere.

VERSIONI DI CALCOLO

Valutando le varie modalità di misurazione di tale parametro si sono prese in considerazione le seguenti versioni:

1. La misurazione è stata effettuata facendo la differenza tra l'altezza massima e minima della traiettoria tracciata dal piede lungo tutto il suo percorso.
2. La misurazione è stata effettuata facendo la differenza tra il punto massimo del percorso e il punto più alto che si raggiunge durante il periodo in cui il piede tocca terra.
3. La misurazione è stata effettuata facendo istante per istante, per tutto il periodo, la differenza tra la posizione del piede che sta toccando terra e il piede che ne entrerà in contatto successivamente. Tra questa serie di valori è stato considerato il valore maggiore.

La prima versione di calcolo è stata scartata perchè restituisce un valore superiore a quello reale, mentre la seconda versione perchè, pur essendo accettabile ai fini dello studio, restituisce un valore meno accurato rispetto a quello che possiamo ottenere con la terza opzione.

Quest'ultima ci restituisce il valore effettivo di superamento ostacolo, ed è perciò quella applicata allo studio.

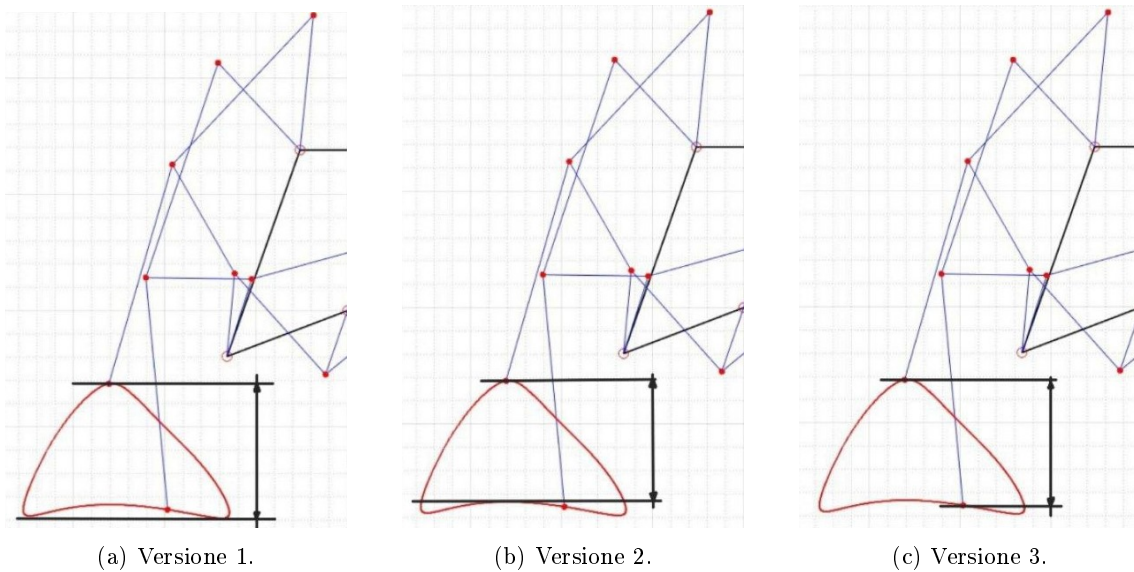


Figura 5.1: Modalità di misurazione dell'altezza del piede

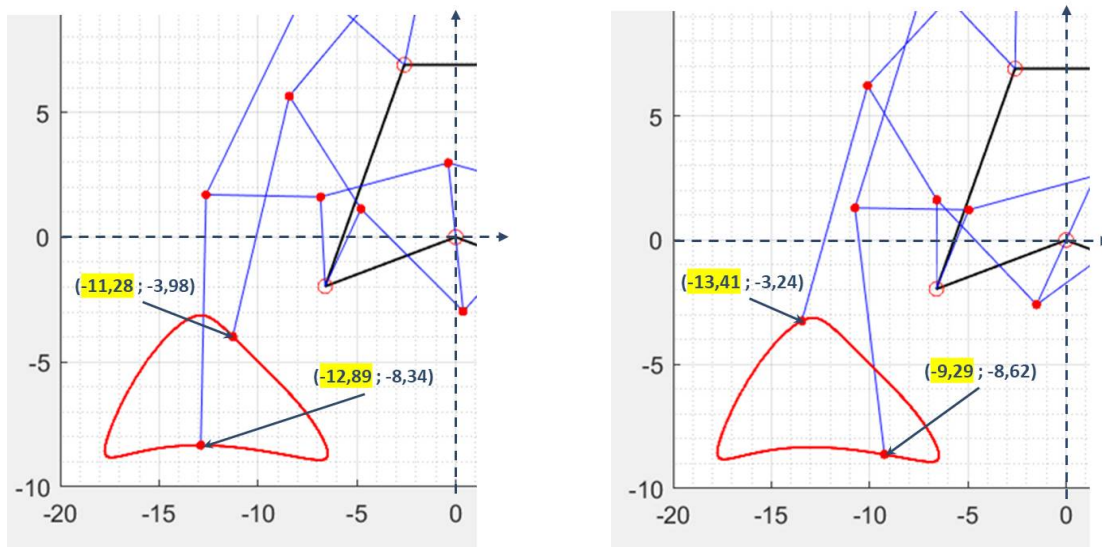
Di seguito sono riportati i valori dell'altezza del piede, rapportati all'ingombro, per ciascun meccanismo, con il metodo descritto precedentemente:

	Klann	Strandbeest	Strider	Trotbot
$\frac{\text{altezzapiede}}{\text{ingombro}}$	25.26%	17.84%	29.75%	26.86%

Tabella 5.1: Raccolta dati altezza del passo

VERIFICA SUPERAMENTO OSTACOLI

È necessario verificare che, nell'istante di misurazione dell'altezza del passo, il piede a contatto con il terreno non ostruisca il superamento dell'ostacolo da parte del piede più alto, ovvero verificando che la posizione sull'asse x del piede a contatto con il terreno sia più interna rispetto all'altro piede interessato alla misurazione. Nel caso rappresentato il robot sta avanzando verso sinistra, si verifica quindi che il piede a terra sia più a destra rispetto all'altro.



Capitolo 6

VELOCITÀ E VARIANZA DEL MECCANISMO

Si è andati a studiare la velocità di ciascun **meccanismo**, per riuscire a comprendere quale, tra i quattro, fosse il **più veloce**. Per il confronto si è rapportata la velocità del meccanismo, assunta pari a quella del piede a contatto con la terra, con la velocità di rotazione della manovella. Nella simulazione le velocità delle manovelle sono state imposte pari a 1 rad/s.

METODO DI CALCOLO DELLA VELOCITÀ

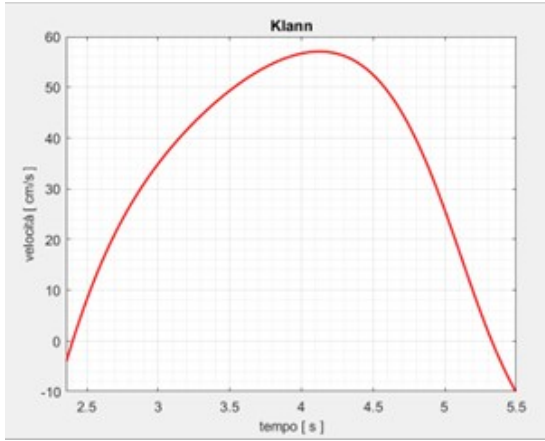
La velocità di ciascun meccanismo è stata calcolata sul piede solamente nei periodi in cui quest'ultimo è a contatto con il terreno. Il calcolo della velocità è stato effettuato tramite un rapporto tra la variazione di posizione in x del piede in due istanti successivi e il periodo di campionamento.

Replicando questo calcolo per l'intero periodo di movimento del meccanismo otteniamo il grafico della velocità nel loro relativo periodo di contatto con il terreno. Con l'utilizzo di Matlab abbiamo definito il periodo di calcolo per tale parametro, ovvero siamo andati a vincolare il calcolo della velocità solo quando il piede preso in considerazione avesse una coordinata in y inferiore rispetto agli altri piedi.

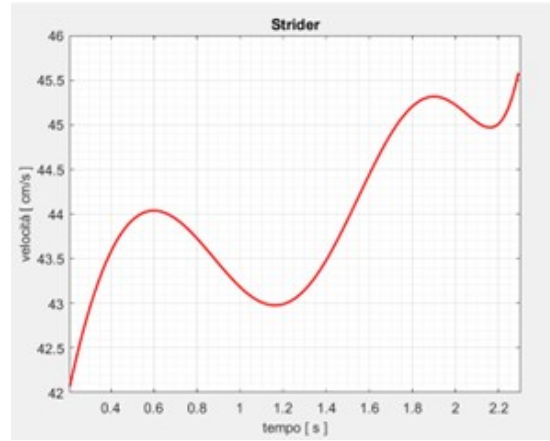
Considerando che il periodo di movimento dei nostri meccanismi è pari a 2π , possiamo verificare che questi periodi di calcolo siano congrui con ciò che ci aspettiamo. Ciò è stato fatto considerando che, nel caso in cui abbiamo tre gambe per lato, il periodo di contatto dovrebbe essere pari a $\frac{2}{3}\pi$, mentre nel caso in cui siano due allora il nostro periodo dovrebbe essere pari a π .

CODICE MATLAB

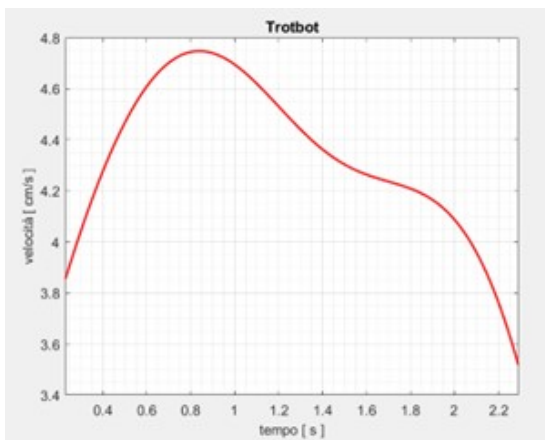
```
1 G_1_xP = G_1(1); % assegnazione del valore in x del piede nell'  
   istante precedente  
2  
3 if G_1(2) < G_2(2) % periodo di calcolo  
4  
5     vel_piede = (G_1_xP - G_1(1)) / (1 / frequenza_campionamento);  
6     vel_piede_ar = [vel_piede_ar, vel_piede];  
7     n = n + 1; % numero di campioni presi mentre il piede tocca  
   terra  
8  
9 end
```



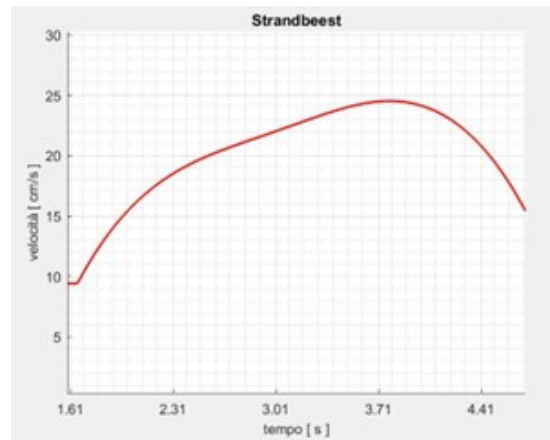
(a) Klann



(b) Strider



(c) Trotbot



(d) Strandbeest

Figura 6.1: Raccolta grafici Matlab delle velocità

Di seguito sono riportati i valori della velocità, rapportati all'ingombro, ottenuti per ciascun meccanismo con il metodo descritto precedentemente:

	Klann	Strandbeest	Strider	Trotbot
$\frac{velocità}{ingombro}$	16.66%	16.14%	24.30%	22.82%

Tabella 6.1: Raccolta dati velocità

VARIANZA

La varianza è una **misura statistica** che indica la dispersione o l'estensione dei dati rispetto alla loro media. Una varianza elevata suggerisce che i dati sono più dispersi, mentre una varianza bassa indica che i dati sono più vicini alla media. Tenendo conto di queste supposizioni nel nostro caso specifico viene valutata rispetto alla velocità. La formula della varianza utilizzata è la seguente:

$$\text{Varianza} = \frac{\sum_{i=1}^N (X_i - \bar{X})^2}{N} \quad (6.1)$$

Dove N indica il numero di campioni, X_i rappresenta la velocità istantanea e la \bar{X} indica la velocità media.

CODICE MATLAB

```
1 valore_medio_vel = mean(vel_piede_ar);
2 i = 1;
3 numeratore = 0;
4
5 while i <= n
6
7     vel_piede_i = vel_piede_ar(i);
8     numeratore = numeratore + (vel_piede_i - valore_medio_vel)^2;
9     i = i + 1;
10
11 end
12
13 varianza = numeratore / n;
```

Di seguito sono riportati i valori della varianza, rapportata all'ingombro, ottenuti per ciascun meccanismo con il metodo descritto precedentemente:

	Klann	Strandbeest	Strider	Trotbot
$\frac{\text{varianza}}{\text{ingombro}}$	17.85%	10.85%	0.04%	0.47%

Tabella 6.2: Raccolta dati varianza

Capitolo 7

STABILITÀ CENTRO DI MASSA

Abbiamo considerato per tutto lo studio il centro di massa posto nel giunto di collegamento della manovella con il telaio. La stabilità del centro di massa che siamo andati a valutare è riferita al solo **movimento verticale**, siccome quella sull'asse orizzontale si può eventualmente dedurre dalla stabilità della velocità del meccanismo. Per valutare la stabilità del CM abbiamo inizialmente considerato l'andamento del contatto a terra del meccanismo sul solo lato sinistro. Grazie alle capacità di calcolo di inventor e matlab, ci siamo fatti restituire dalla simulazione un grafico che mostri il movimento verticale nel tempo del piede più basso tra i presenti in ogni istante, per un giro completo della manovella. Abbiamo assunto che il movimento del CM corrisponda al movimento del piede a contatto appena ricavato. Dopo una rivalutazione, considerando il meccanismo nel suo insieme, si è optato per una diversa misurazione. Ciò è dovuto al fatto che nel movimento, le gambe dei lati sinistro e destro non si muovono in modo sincrono. Conoscendo l'andamento della totalità dei piedi nell'intero periodo ci è stato possibile ricavare la **distanza tra CM e terreno** per ogni istante di misurazione, seguendo il seguente procedimento:

1. Valutazione piede a contatto con terra lato SX (Punto A) e DX (Punto B)
2. Calcolo coefficiente angolare retta passante per i punti A e B: $m = \frac{y_A - y_B}{x_A - x_B}$
3. Equazione completa retta passante per A e B: $q = y_A - mx_A \implies y = mx + q$
4. Calcolo coefficiente angolare retta perpendicolare e conseguente equazione: $y_{perp} = -\frac{1}{m}x$
5. Calcolo coordinate x e y del punto O: $y_0 = \frac{1}{1+m^2}$ $x_0 = \frac{y_0}{m}$
6. Calcolo distanza tra O e CM: $\overline{O.CM} = \sqrt{x_0^2 + y_0^2}$

Effettuando questi passaggi istante per istante, ci è stato possibile valutare il **movimento del CM nel tempo**, in un giro completo della manovella. Per valutare la stabilità siamo andati a calcolare la **variazione tra punto massimo e minimo** nell'intero percorso. Lo studio per i meccanismi Strandbeest e Strider è stato effettuato su Inventor, mentre per i meccanismi Klann e TrotBot su Matlab.

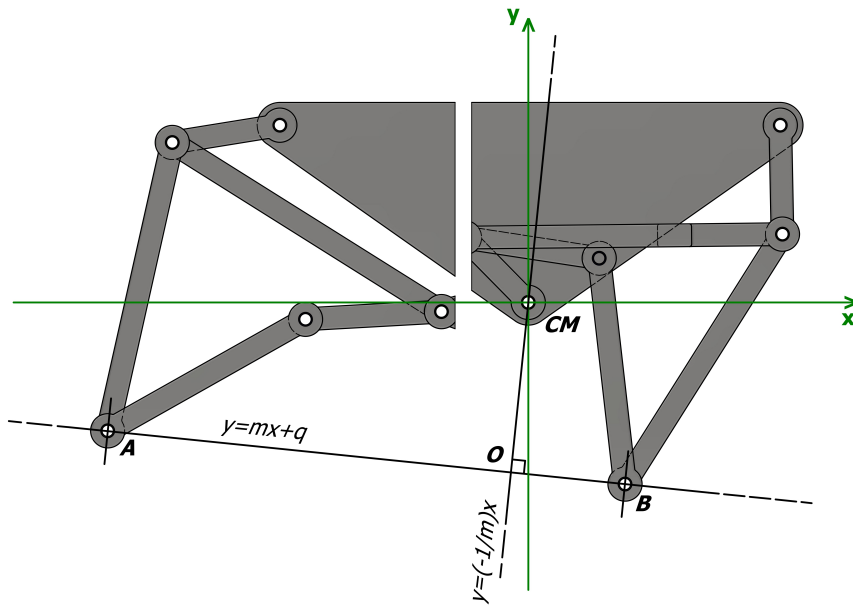


Figura 7.1: Rappresentazione piano di appoggio reale

CODICE MATLAB

```

1  distanza_0-CM_ar = [];
2
3  if G_1(2) < G_2(2)
4      if P_1(2) <= P_2(2)
5          delta_Y = G_1(2) - P_1(2);
6          delta_X = G_1(1) - P_1(1);
7      else
8          delta_Y = G_1(2) - P_2(2);
9          delta_X = G_1(1) - P_2(1);
10     end
11     m = delta_Y/delta_X; % coefficiente angolare
12     q = G_1(2) - m * G_1(1); % offset
13
14     m_perp = -1/m ;
15     y_0 = q / (1 + m^2) ;
16     x_0 = y_0 / m_perp;
17
18     distanza_0-CM = sqrt(x_0^2 + y_0^2);
19     distanza_0-CM_ar = [distanza_0-CM_ar,distanza_0-CM];
20
21 end
22
23 h_max_cm = max(distanza_cm_ar);
24 h_min_cm = min(distanza_cm_ar);
25
26 variazione_cm = h_max_cm - h_min_cm;

```

GRAFICI OTTENUTI

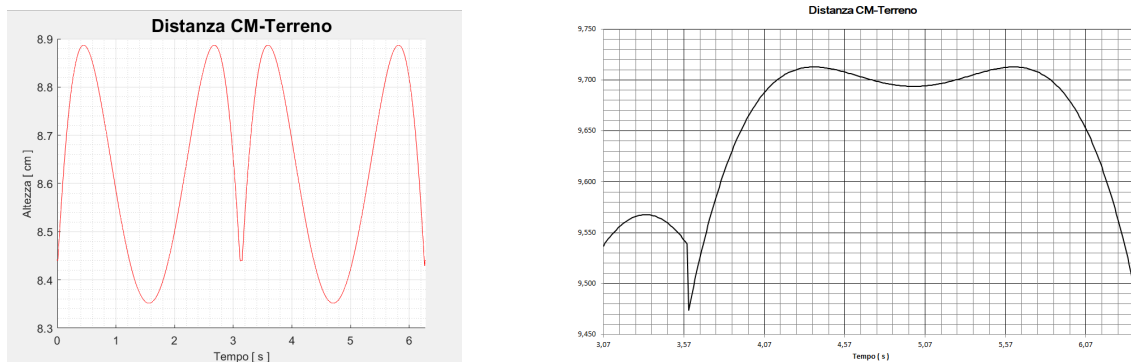


Figura 7.2: A sinistra grafico Klann (MATLAB), a destra grafico Strider (INVENTOR)

Per il confronto dei 4 meccanismi il valore appena calcolato non ci restituisce un confronto veritiero a causa delle diverse dimensioni. È stato quindi necessario riportare la variazione del CM con il relativo ingombro. I risultati ottenuti sono i seguenti:

	Klann	Strandbeest	Strider	Trotbot
$\frac{\text{VariazioneCM}}{\text{ingombro}}$	2.50%	1.12%	1.32%	1.81%

Tabella 7.1: Raccolta valori stabilità centro di massa.

Capitolo 8

COSTO ENERGETICO

Per l'accoppiamento tra articolazioni è stato ipotizzato l'utilizzo di **cuscinetti a rulli conici**, data la loro efficacia nel consentire la rotazione relativa tra componenti non perfettamente allineati, e al contempo garantire un minor **coefficiente d'attrito**.
Nella scelta di tali cuscinetti è fondamentale soddisfare determinati requisiti, quali: velocità, precisione, condizioni di operatività e tolleranza dei carichi.

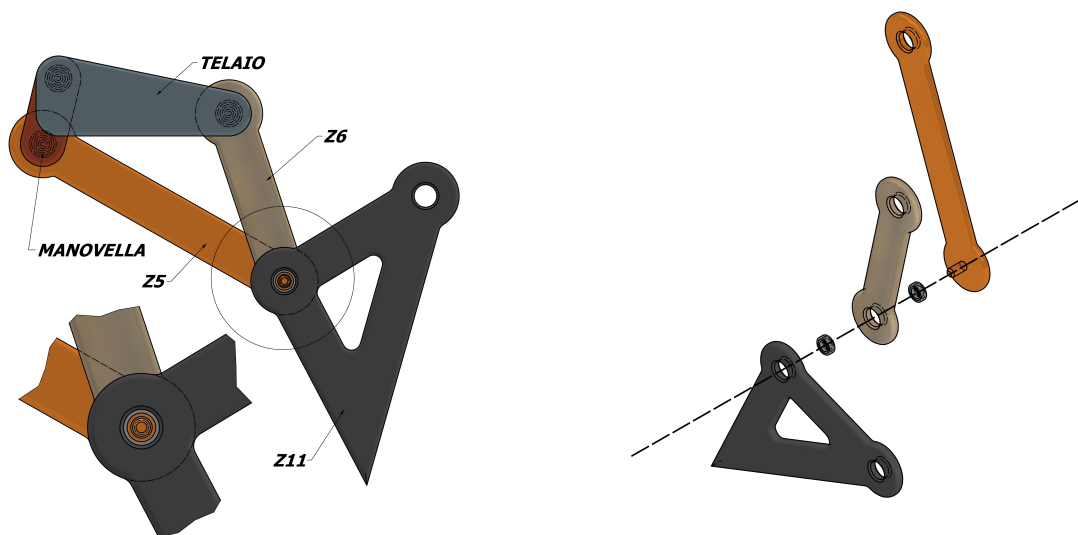


Figura 8.1: Esploso di un accoppiamento rotoidale

Aspetto caratterizzante dei cuscinetti a rulli conici (e di tutti i cuscinetti in generale) è la presenza di **attrito viscoso**, dovuto allo strisciamento tra aste meccaniche, ma che spesso viene diminuito grazie all'uso di lubrificazione meccanica.

La resistenza d'attrito influisce sulle prestazioni del cuscinetto, contribuendo alla **perdita di potenza** (dovuto alla coppia resistente) e alla **dissipazione di energia**.

La coppia d'attrito oltre che a dipendere dalla lubrificazione (coeff. d'attrito viscoso) dipende anche dalla velocità di rotazione, secondo una relazione di proporzionalità diretta.

Essa può essere calcolata come:

$$\tau = \eta * \omega \quad (8.1)$$

Dove τ rappresenta la coppia di attrito viscoso [Nm], η rappresenta il coefficiente di viscosità e ω rappresenta la velocità angolare del giunto [rad/s].

Per il calcolo della potenza dissipata, è stata considerata la formula della potenza generata da una coppia.

Nel contesto dei sistemi rotanti, la potenza è data dal prodotto della coppia stessa per la velocità angolare :

$$P = \tau * \omega \quad (8.2)$$

Dove P è la potenza in Watt.

Per valutare la potenza dissipata, nell'equazione 8.2 la τ rappresenta la coppia di attrito viscoso, ω la velocità angolare del giunto.

Combinando quindi le due formule 8.1 e 8.2 otterremo:

$$P = \eta * \omega^2 \quad (8.3)$$

La 8.3 è pertanto la relazione finale usata per il calcolo della potenza dissipata su tutti i vari cuscinetti.

Siccome la velocità angolare del giunto è variabile nel tempo, è stato necessario considerare la coppia di attrito viscoso punto per punto. È stato perciò suddiviso il movimento in piccoli intervalli di tempo, per poi misurare la velocità angolare tra le piste del cuscinetto.

Queste piste costituiscono le superfici di contatto all'interno del cuscinetto, dove i rulli si muovono per consentire il movimento rotatorio tra le parti collegate.

Le velocità angolare è definita come il tasso di variazione dell'angolo rispetto al tempo :

$$\omega = \frac{\partial \gamma}{\partial t} = \frac{\Delta \gamma}{\Delta t} \quad (8.4)$$

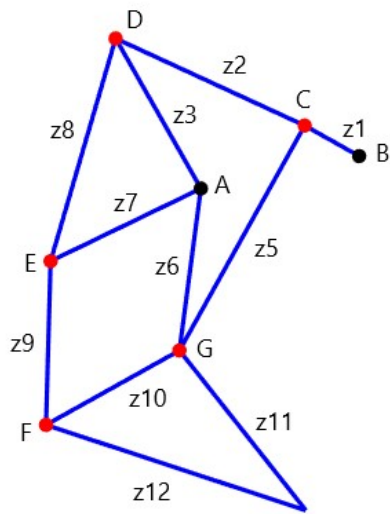
Per il movimento del meccanismo, con una velocità angolare costante della manovella pari ad 1 rad/s, sono stati considerati degli scatti costanti di 0,01 rad. Avremmo perciò una frequenza di campionamento pari a 100 campioni per ogni radiante percorso, con un periodo di campionamento, cioè il tempo che trascorre tra due campioni, pari a 0,01 s. Ciò che è stato fatto consiste nell'utilizzo di codice implementato su Matlab al fine di valutare la variazione dell'angolo interno tra le aste connesse attraverso il cuscinetto in due istanti di tempo successivi.

Tale procedura prevede il calcolo della differenza angolare tra i suddetti istanti, al fine di determinare la velocità angolare relativa tra le articolazioni vincolate dal giunto. Questa velocità angolare è ottenuta mediante la divisione della differenza angolare per il periodo di campionamento.

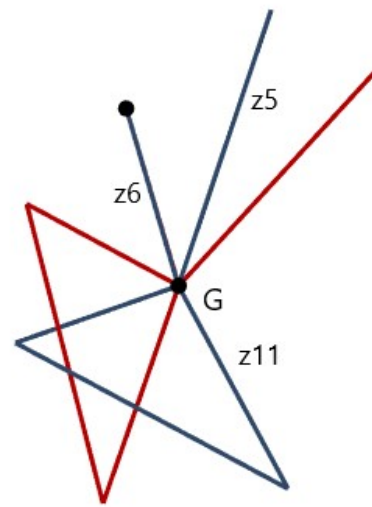
Per valutare il movimento angolare relativo è stata definita un'articolazione principale, nella quale sono stati montati cuscinetti che consentono la rotazione con le altre articolazioni.

Ai fini della valutazione, abbiamo studiato il movimento angolare relativo su una singola gamba. Prendiamo ad esempio in considerazione il meccanismo Strandbeest 8.2a.

Se si considera per esempio il accoppiamento posto in G, si ha che z5 è l'asta principale, alla quale vengono associate le aste z6 ed il corpo rigido z11. Sul particolare riferito allo stesso cuscinetto 8.2b si possono osservare le configurazioni delle aste associate ad esso in due istanti



(a)



(b)

differenti. Per valutare il movimento angolare "infinitesimo", è stata considerata la configurazione delle articolazioni in due campionamenti successivi.

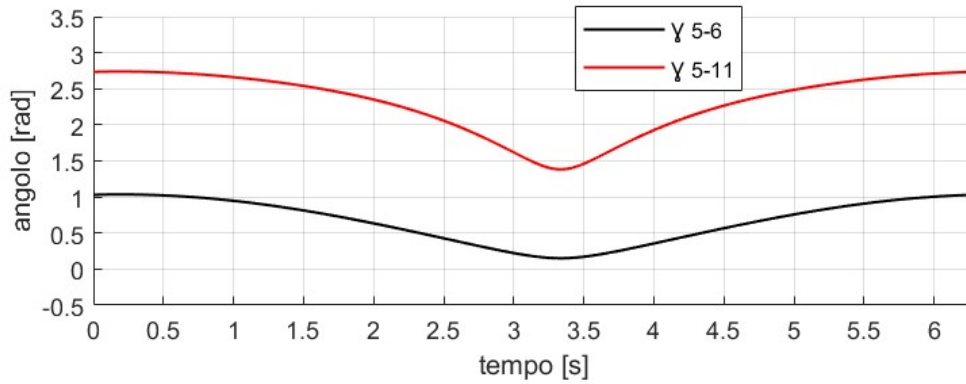
CODICE MATLAB

```

1  zx = sqrt((C(1)-A(1))^2 + (C(2)-A(2))^2);
2  gamma5_6 = acos((z5^2 + z6^2 - zx^2)/(2*z5*z6));
3  zc = sqrt((H(1)-C(1))^2 + (H(2)-C(2))^2);
4  gamma5_11 = acos((z5^2 + z11^2 - zc^2)/(2*z5*z11));
5  w = 1; % velocità angolare manovella
6  frequenza_campionamento = 100; %campioni presi in un secondo
7  dt = 1/frequenza_campionamento;
8  tempo_simulazione = 2*pi %durata percorso completo
9  %all'interno del ciclo while ...
10 gammap_5_6 = gamma5_6;
11 gammap_5_11 = gamma5_11;
12 %ricalcò degli angoli interni, come in 1-4 riga ...
13 %differenza tra angoli interni
14 gammay_5_6 = gamma5_6 - gammap_5_6;
15 gammav_5_11 = gamma5_11 - gammap_5_11 ;

```

Valutando il movimento angolare per un periodo completo di rotazione pari 2π radianti percorsi, in 2π s, è risultato il seguente grafico :



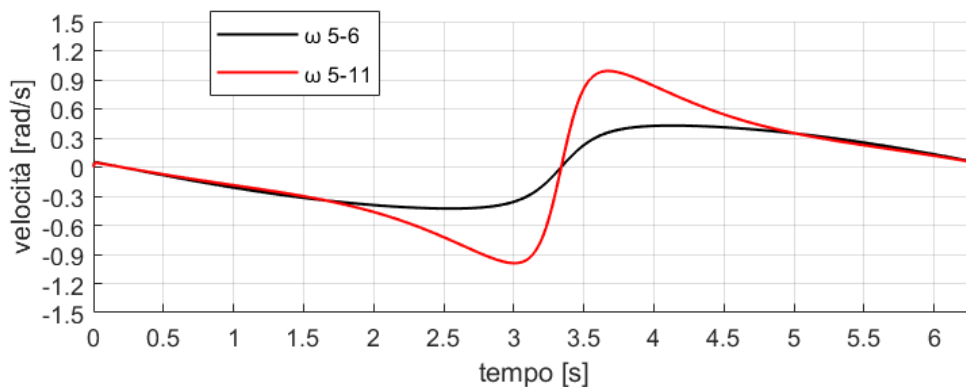
Una volta trovata la variazione infinitesimale tra le aste, è stata valutata la velocità angolare relativa tra queste nello stesso periodo, richiamando la definizione che afferma che la velocità angolare è definita come il tasso di variazione dell'angolo rispetto al tempo (formula 7.4), e ricordando che la velocità relativa tra le aste coincide con il movimento rotatorio subito dalle piste del cuscinetto :

```

1 %all'interno del ciclo while ...
2 %dt = 0.01
3 speed_5_6 = gammav_5_6/dt;
4 speed_5_11 = gammav_5_11/dt;

```

Trovata ora la velocità angolare relativa tra le articolazioni fra due campionamenti successivi, e valutandola nel periodo completo di rotazione, ne è risultato il seguente grafico :



Ricordando la 8.3, che mette in relazione direttamente proporzionale la potenza con la velocità angolare, e trascurando il coefficiente di attrito viscoso, che sarà solamente un fattore di scala, si può ora ottenere la potenza infinitesima persa per attrito nei vari giunti.

Considerando sempre il l'accoppiamento in G, la potenza dissipata per attrito viscoso si può ricavare come :

```
1 %all'interno del ciclo while ...
2 potG = speed_5_6^2 + speed_5_11^2;
```

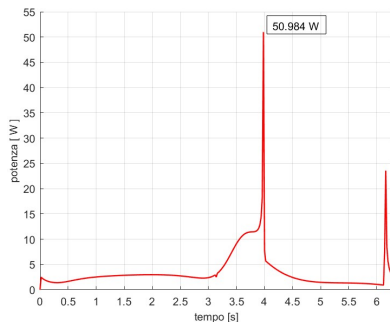
Ripetendo gli stessi ragionamenti e calcoli per ogni singolo cuscinetto, è possibile valutare la potenza dissipata istantaneamente per attrito viscoso nella singola gamba, sommando le potenze dissipate :

```
1 potenza_ar = [];
2 %all'interno del ciclo while ...
3 potC = speed_1_2^2 + speed_1_5^2;
4 potD = speed_2_3^2;
5 potE = speed_3_9^2;
6 potF = speed_9_11^2;
7 potG = speed_5_6^2 + speed_5_11^2;
8 %potenza dissipata su tutta la gamba
9 potenza = potC + potD + potE + potF + potG;
10 potenza_ar = [potenza_ar, potenza];
```

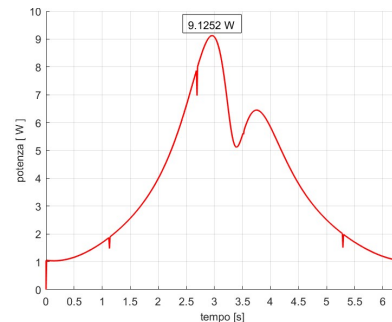
Lo stesso studio è stato ripetuto per tutto il periodo di rotazione della manovella, pari a 2π s, e per tutti e 4 i meccanismi analizzati.

Lo studio è stato svolto diversamente per il meccanismo Strider, in quanto, essendo asimmetrico, è stato necessario studiare la coppia di gambe che comprende lato destro e lato sinistro.

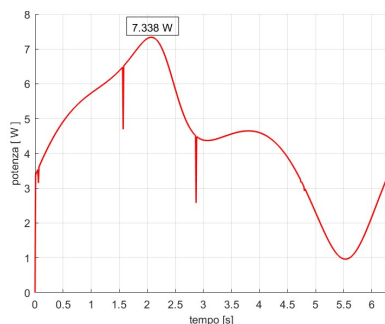
A meno del coefficiente di attrito viscoso, ciò che ne è risultato sono i seguenti grafici :



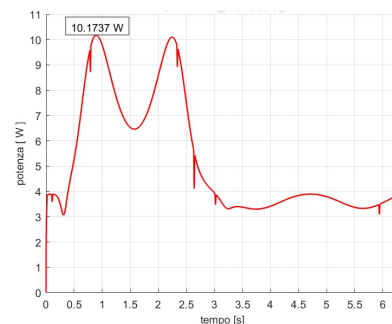
(a) Klann



(b) Strandbeest



(c) Trotbot



(d) Strider

Figura 8.3: Tracciamento dei grafici della potenza in funzione del tempo

Ai fini del dimensionamento di un ipotetico motore in grado di azionare il meccanismo, abbiamo preso in considerazione la massima potenza dissipata all'interno dei vari cuscinetti, ricordando che ciò viene richiesto per il mantenimento della sincronia nel meccanismo è che la potenza erogata dal motore sia almeno pari alla potenza dissipata a causa dell'attrito.

A meno del coefficiente di attrito, e rammentando il numero di gambe da cui sono composti i vari meccanismi di locomozione, ciò che ne risulta è:

	Klann(4 gambe)	Strandbeest (4 gambe)	Strider(6 gambe)	Trotbot (6 gambe)
$P_{DISSIPATA}[W]$	$36.501 * \eta$	$44.028 * \eta$	$30.521 * \eta$	$203.936 * \eta$

Tabella 8.1: Raccolta dati potenza dissipata

Per un confronto più accurato tra i 4 meccanismi analizzati, si è deciso di valutare l'energia dissipata in un percorso completo di 2π s, ricordando che l'energia è definita come l'integrale della potenza nel tempo.

Avendo definito la potenza ad intervalli di tempo di 0.01 s, ne risulterà quindi un integrale a tratti, dove l'area totale sottesa alla curva di interesse sarà data dalla somma delle aree definite ad incrementi infinitesimi :

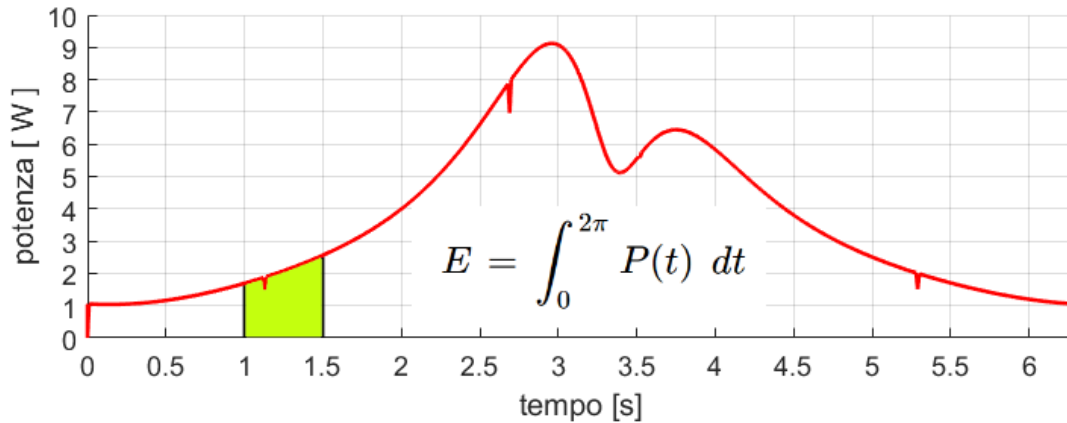


Figura 8.4: Area sottesa dal grafico della potenza

```

1 tempo = 0:dt:tempo_simulazione; %creo un vettore di tempi
2 energia = trapz(tempo, potenza_ar); %trovo l'energia, integrale
   potenza

```

Abbiamo ottenuto infine questi risultati :

	Klann(4 gambe)	Strandbeest (4 gambe)	Strider(6 gambe)	Trotbot (6 gambe)
$E_{DISSIPATA}[J/giro]$	$74.92 * \eta$	$92.08 * \eta$	$97.62 * \eta$	$165.15 * \eta$

Tabella 8.2: Raccolta dati energia dissipata

Da tutti i vari grafici della potenza, si possono notare dei cali improvvisi in alcuni tratti. Se si analizza il movimento tra due articolazioni collegate, si può facilmente notare che queste perdite si hanno generalmente in corrispondenza del cambio di direzione del moto relativo tra le piste del cuscinetto.

Se prendiamo sempre in esempio l'accoppiamento posto in G del meccanismo Strandbeest, e valutiamo solo la potenza dissipata dovuta al movimento rotatorio tra l'asta z_5 e z_6 , e tra l'asta z_5 ed il corpo rigido z_{11} , si ha :

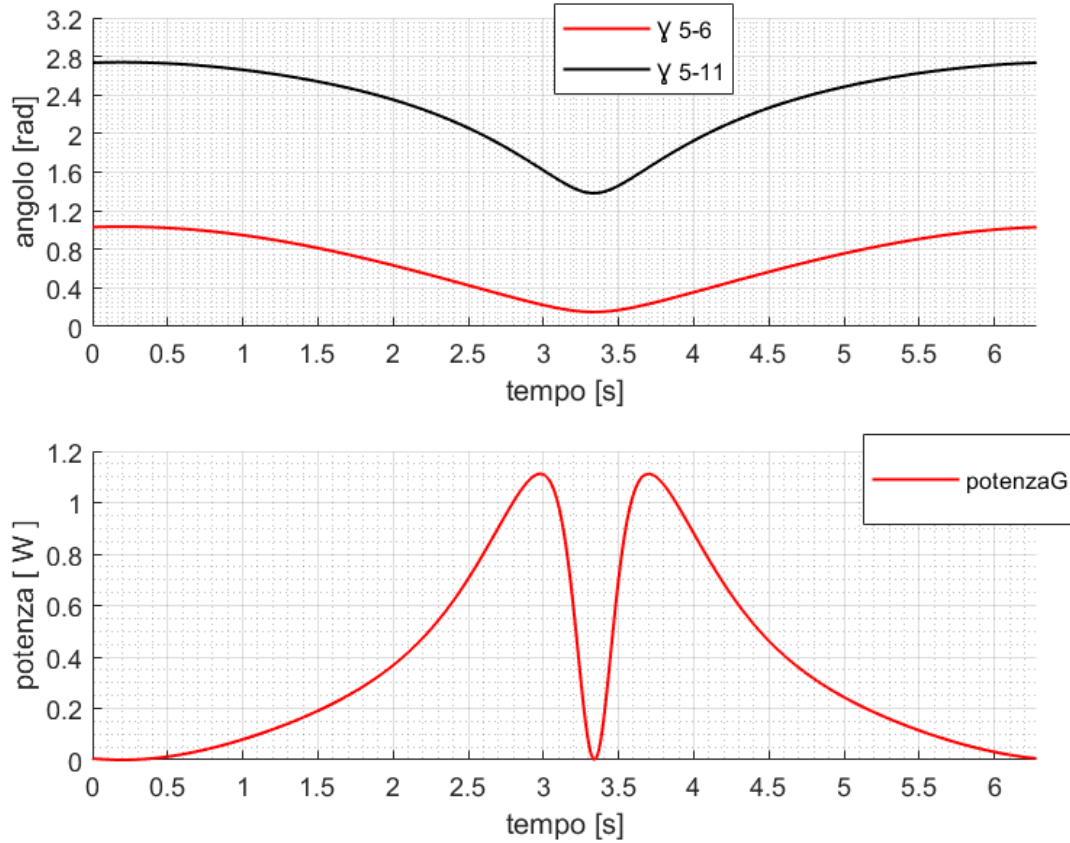


Figura 8.5: Esempio andamento potenza in funzione della rotazione angolare

La perdita improvvisa di potenza quando il moto cambia da senso orario ad anti-orario è dovuta al fatto che in quel punto specifico c'è una transizione di direzione del moto. Quando, infatti, il movimento cambia direzione, ci sono dei cambiamenti istantanei di velocità angolare, portando a picchi di carico, causando "perdite" di potenza. Questo è particolarmente evidente in un tratto infinitesimo, perchè il cambiamento di direzione è istantaneo. Infatti nei punti prossimi al cambio di direzione, la velocità angolare decresce notevolmente, mentre nell'istante preciso del cambio di direzione, la derivata dell'angolazione relativa, ossia la velocità relativa, è nulla. In termini matematici, si verificano salti discontinui della derivata della potenza in quel punto.

Capitolo 9

ANALISI DI POSIZIONE

In questo capitolo andremo a raccogliere tutto lo studio preliminare fatto in merito alla posizione che può assumere istante per istante ogni meccanismo. Ogni trave verrà rappresentata graficamente come vettore.

METODO DELLE DIADI RRR

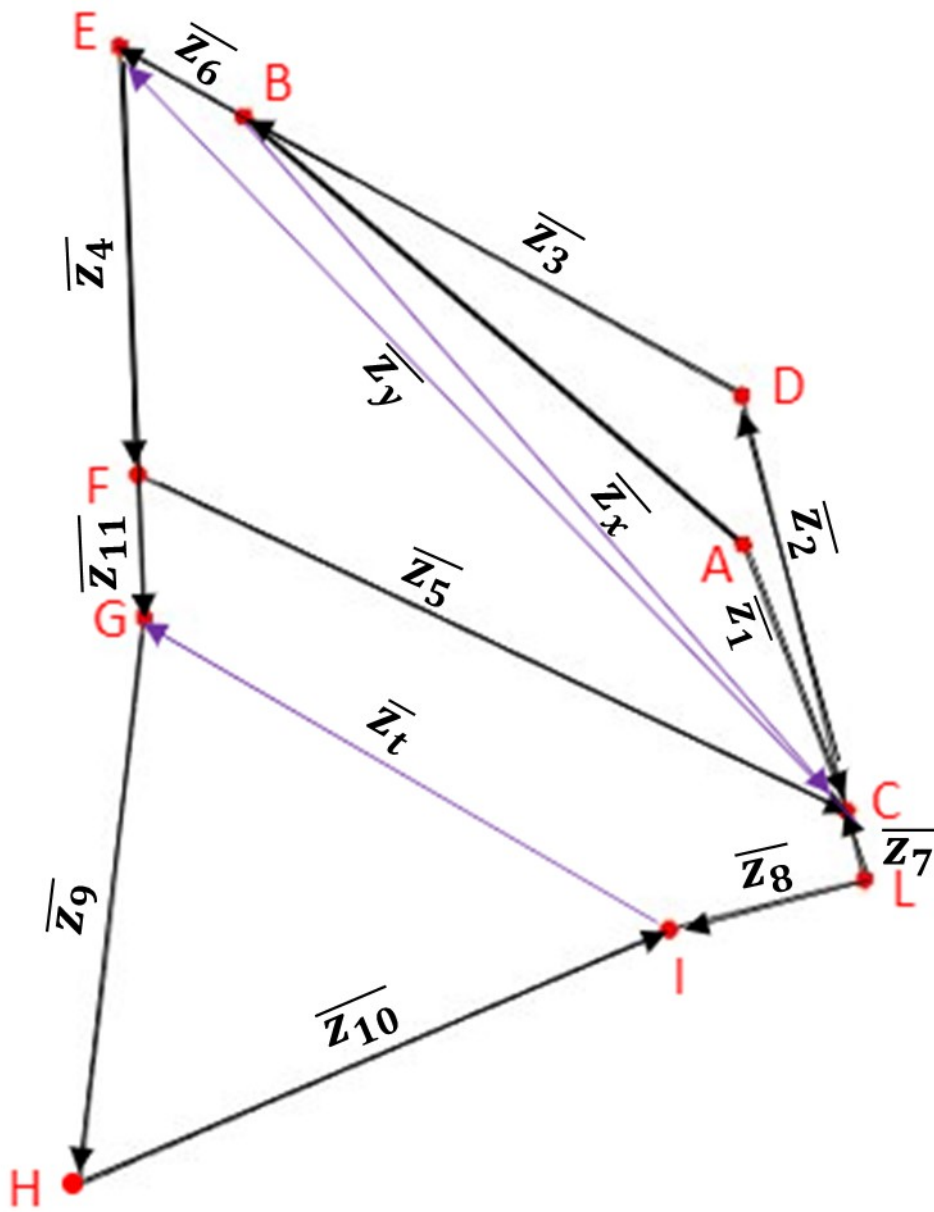
Analizzare per intero un meccanismo con le semplici equazioni di chiusura risulta molto difficile. Il metodo che useremo per semplificare l'analisi consiste semplicemente nel vedere il meccanismo come insieme di tanti meccanismi più piccoli, chiamati appunto **DIADI**.

Ogni diade sarà formata da tre travi formanti un'equazione di chiusura, e sarà sempre di tipo RRR (essendo tutte le travi congiunte mediante soli accoppiamenti rotoidali). Spesso risulta impossibile scegliere tre travi che si chiudono formando un triangolo, per questo spesso andremo ad inserire dei vettori immaginari.

La comodità di tale metodo sta nel fatto che ci troveremo a lavorare sempre con dei triangoli, e conoscendo la lunghezza delle travi, ci basterà conoscere la posizione e l'inclinazione di una sola tra essi per ricavare quelle di tutte le altre, mediante l'utilizzo dei **TEOREMI DEL SENO E DEL COSENO**.

NOTA: tutte le formule saranno scritte in funzione della posizione angolare della manovella (coordinata libera).

9.1 TROTBOT



MAGLIA CDB

$$PUNTOC : \begin{cases} C_x = A_x + B_1 \cos(\varphi_1) \\ C_y = A_y + B_1 \sin(\varphi_1) \end{cases}$$

$$z_x = \sqrt{(C_x - B_x)^2 + (C_y - B_y)^2}$$

$$\varphi_x = \arctan\left(\frac{C_y - B_y}{C_x - B_x}\right)$$

$$\gamma_{2x} = \arccos\left(\frac{z_2^2 + z_x^2 - z_3^2}{2z_2z_x}\right)$$

$$\gamma_{3x} = \arccos\left(\frac{z_3^2 + z_x^2 - z_2^2}{2z_3z_x}\right)$$

$$\gamma_{23} = \pi - \gamma_{2x} - \gamma_{3x}$$

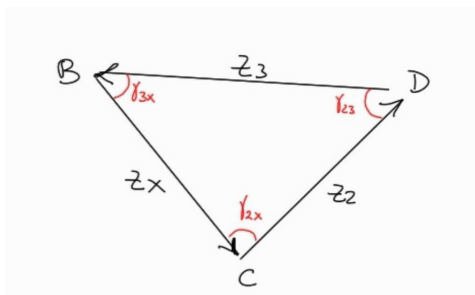
$$\varphi_2 = \varphi_x + \pi - \gamma_{2x}$$

$$\varphi_3 = \varphi_x + \pi + \gamma_{3x}$$

$$\varphi_8 = \varphi_2 + \frac{\pi}{2}$$

$$PUNTOD : \begin{cases} D_x = C_x + z_2 \cos(\varphi_2) \\ D_y = C_y + z_2 \sin(\varphi_2) \end{cases}$$

$$PUNTOE : \begin{cases} E_x = B_x + z_6 \cos(\varphi_3) \\ E_y = B_y + z_6 \sin(\varphi_3) \end{cases}$$



$$PUNTOL : \begin{cases} L_x = C_x + z_7 \cos(\varphi_2) \\ L_y = C_y + z_7 \sin(\varphi_2) \end{cases}$$

$$PUNTOI : \begin{cases} I_x = L_x + z_8 \cos(\varphi_8) \\ I_y = L_y + z_8 \sin(\varphi_8) \end{cases}$$

MAGLIA CEF

$$z_y = \sqrt{(E_x - C_x)^2 + (E_y - C_y)^2}$$

$$\varphi_y = \arctan\left(\frac{E_y - C_y}{E_x - C_x}\right) + \pi$$

$$\gamma_{4y} = \arccos\left(\frac{z_4^2 + z_y^2 - z_5^2}{2z_4z_y}\right)$$

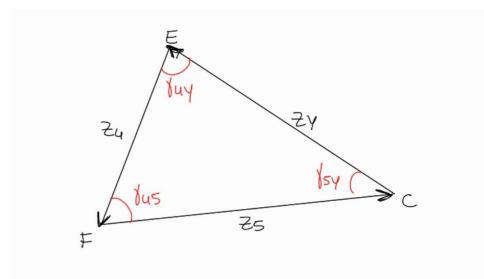
$$\gamma_{5y} = \arccos\left(\frac{z_5^2 + z_y^2 - z_4^2}{2z_5z_y}\right)$$

$$\gamma_{54} = \pi - \gamma_{5y} - \gamma_{4y}$$

$$\varphi_4 = \varphi_y + \pi - \gamma_{4y}$$

$$\varphi_5 = \varphi_y + \pi + \gamma_{5y}$$

$$PUNTOG : \begin{cases} G_x = E_x + (z_4 + z_{11}) \cos(\varphi_8) \\ G_y = E_y + (z_4 + z_{11}) \sin(\varphi_8) \end{cases}$$



MAGLIA GHI

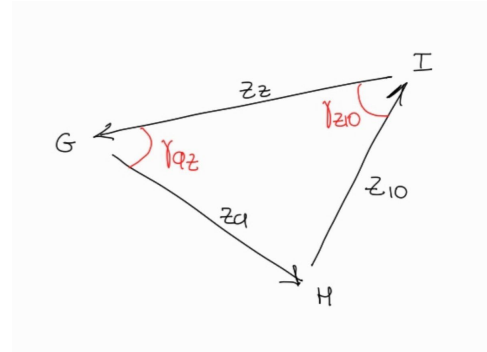
$$z_z = \sqrt{(G_x - I_x)^2 + (G_y - I_y)^2}$$

$$\varphi_z = \arctan\left(\frac{G_y - I_y}{G_x - I_x}\right) + \pi$$

$$\gamma_{9z} = \arccos\left(\frac{z_9^2 + z_z^2 - z_{10}^2}{2z_9z_z}\right)$$

$$\varphi_9 = \varphi_z + \pi - \gamma_{9z}$$

$$PUNTOH : \begin{cases} H_x = G_x + z_9 \cos(\varphi_2) \\ H_y = G_y + z_9 \sin(\varphi_2) \end{cases}$$



ANALISI DI POSIZIONE SU MATLAB

```

1  %TROTBOT
2  f1 = figure('Name','Assieme','NumberTitle','off') ;
3  xlim([-20,15])
4  ylim([0, 20])
5  title('TROTBOT');
6  hold on;
7  axis equal;
8  %definisco i punti a telaio
9  A = [0, 10];
10 B = [-7, 16];
11 %linea tra i punti a telaio
12 z_telaio = plot([A(1), B(1)], [A(2), B(2)], 'k-', 'LineWidth', 1 );
13 %definisco i punti iniziali
14 C = [3.99980000166666, 10.03999993333367];
15 D = [0.961265741939805, 15.2137126567115];
16 E = [-8.99031643548495, 16.1965718358221];
17 F = [-6.98881920697114, 10.5402471594752];
18 G = [-6.32165346413320, 8.65480560069285];
19 H = [-3.92001172046644, 1.02380848792461];
20 I = [2.06595426076268, 7.74453845360303];
21 L = [4.50622237828780, 9.17771377944086];
22 %lunghezze dei vari membri
23 z1 = 4;
24 z2 = 6;
25 z3 = 8;
26 z4 = 6;
27 z5 = 11;
28 z6 = 2;
29 z7 = 1;
30 z8 = 2.83;
31 z9 = 8;
32 z10 = 9;

```

```

33 z11 = 2;
34 %imposto il tempo a 0
35 t = 0;
36 %definisco l'angolo iniziale della manovella
37 theta1 = 0;
38 % parametri per il movimento della manovella
39 w = 1; % vel angolare costante
40 tempo_simulazione = 2*pi;
41 frequenza_campionamento = 100;
42 %definisco le varie articolazioni
43 z1_line = plot([A(1),C(1)],[A(2),C(2)], 'b-');
44 z2_line = plot([C(1),D(1)],[C(2),D(2)], 'b-');
45 z3_line = plot([D(1),B(1)],[D(2),B(2)], 'b-');
46 z4_line = plot([E(1),F(1)],[E(2),F(2)], 'b-');
47 z5_line = plot([F(1),C(1)],[F(2),C(2)], 'b-');
48 z6_line = plot([B(1),E(1)],[B(2),E(2)], 'b-');
49 z7_line = plot([L(1),C(1)],[L(2),C(2)], 'b-');
50 z8_line = plot([L(1),I(1)],[L(2),I(2)], 'b-');
51 z9_line = plot([G(1),H(1)],[G(2),H(2)], 'b-');
52 z10_line = plot([H(1),I(1)],[H(2),I(2)], 'b-');
53 z11_line = plot([F(1),G(1)],[F(2),G(2)], 'b-');
54 giuntoA = plot(A(1), A(2), 'ro');
55 giuntoB = plot(B(1), B(2), 'ro');
56 giuntoC = plot(C(1), C(2), 'r.', 'MarkerSize', 10);
57 giuntoD = plot(D(1), D(2), 'r.', 'MarkerSize', 10);
58 giuntoE = plot(E(1), E(2), 'r.', 'MarkerSize', 10);
59 giuntoF = plot(F(1), F(2), 'r.', 'MarkerSize', 10);
60 giuntoG = plot(G(1), G(2), 'r.', 'MarkerSize', 10);
61 giuntoI = plot(I(1), I(2), 'r.', 'MarkerSize', 10);
62 giuntoL = plot(L(1), L(2), 'r.', 'MarkerSize', 10);
63 %inizio col far variare l'angolo della manovella
64 while t < tempo_simulazione
65     theta1 = theta1 + w/frequenza_campionamento;
66     C = [A(1)+z1*cos(theta1), A(2)+z1*sin(theta1)];
67     %risoluzione maglia CDB
68     z12 = sqrt((B(1)-C(1))^2 + (B(2)-C(2))^2);
69     theta12 = atan2((C(2)-B(2)),(C(1)-B(1)));
70     gamma2_3 = acos((z2^2+z3^2-z12^2)/(2*z2*z3));
71     gamma2_12 = acos((z2^2+z12^2-z3^2)/(2*z2*z12));
72     gamma3_12 = acos((z3^2+z12^2-z2^2)/(2*z3*z12));
73     theta2 = theta12 + pi - gamma2_12;
74     theta3 = theta12 + pi + gamma3_12;
75     D = [C(1)+z2*cos(theta2), C(2)+z2*sin(theta2)];
76     %arriviamo al punto E
77     E = [B(1)+z6*cos(theta3), B(2)+z6*sin(theta3)];
78     %arriviamo al punto L
79     L = [C(1)-z7*cos(theta2), C(2)-z7*sin(theta2)];
80     %arriviamo al punto I
81     theta8 = theta2 + pi/2;
82     I = [L(1)+z8*cos(theta8), L(2)+z8*sin(theta8)];
83     %risoluzione maglia EFC
84     z13 = sqrt((E(1)-C(1))^2 + (E(2)-C(2))^2);
85     theta13 = atan2((E(2)-C(2)),(E(1)-C(1)));
86     gamma4_13 = acos((z4^2+z13^2-z5^2)/(2*z4*z13));
87     gamma4_5 = acos((z4^2+z5^2-z13^2)/(2*z4*z5));
88     gamma5_13 = pi - gamma4_13 - gamma4_5;

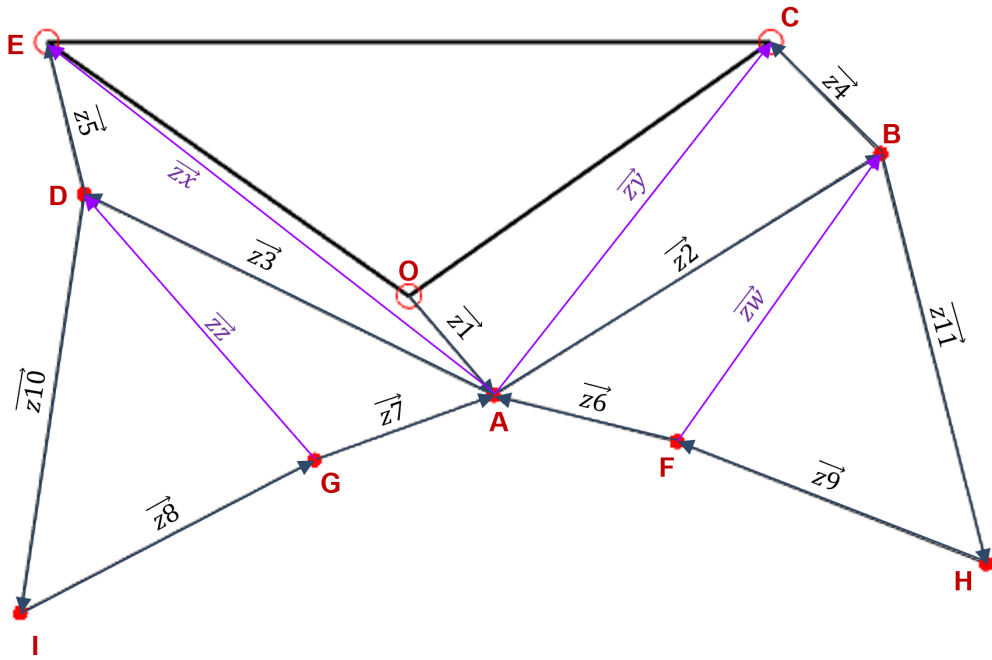
```

```

89     theta4 = theta13 + pi - gamma4_13;
90     theta5 = theta4 - pi - gamma4_5;
91     F = [E(1)+z4*cos(theta4), E(2)+z4*sin(theta4)];
92     %arriviamo al punto G
93     G = [F(1)+z11*cos(theta4), F(2)+z11*sin(theta4)];
94     %risoluzione maglia GHI
95     z14 = sqrt((I(1)-G(1))^2 + (I(2)-G(2))^2);
96     theta14 = atan2((G(2)-I(2)),(G(1)-I(1)));
97     gamma9_10 = acos((z9^2+z10^2-z14^2)/(2*z9*z10));
98     gamma9_14 = acos((z9^2+z14^2-z10^2)/(2*z9*z14));
99     theta9 = theta14 + pi - gamma9_14;
100    H = [G(1)+z9*cos(theta9), G(2)+z9*sin(theta9)];
101
102    set(z1_line, 'XData', [A(1),C(1)], 'YData', [A(2),C(2)]);
103    set(z2_line, 'XData', [C(1),D(1)], 'YData', [C(2),D(2)]);
104    set(z3_line, 'XData', [D(1),B(1)], 'YData', [D(2),B(2)]);
105    set(z4_line, 'XData', [E(1),F(1)], 'YData', [E(2),F(2)]);
106    set(z5_line, 'XData', [F(1),C(1)], 'YData', [F(2),C(2)]);
107    set(z6_line, 'XData', [B(1),E(1)], 'YData', [B(2),E(2)]);
108    set(z7_line, 'XData', [L(1),C(1)], 'YData', [L(2),C(2)]);
109    set(z8_line, 'XData', [L(1),I(1)], 'YData', [L(2),I(2)]);
110    set(z9_line, 'XData', [G(1),H(1)], 'YData', [G(2),H(2)]);
111    set(z10_line, 'XData', [H(1),I(1)], 'YData', [H(2),I(2)]);
112    set(z11_line, 'XData', [F(1),G(1)], 'YData', [F(2),G(2)]);
113    set(giuntoA, 'XData', A(1), 'YData', A(2));
114    set(giuntoB, 'XData', B(1), 'YData', B(2));
115    set(giuntoC, 'XData', C(1), 'YData', C(2));
116    set(giuntoD, 'XData', D(1), 'YData', D(2));
117    set(giuntoE, 'XData', E(1), 'YData', E(2));
118    set(giuntoF, 'XData', F(1), 'YData', F(2));
119    set(giuntoG, 'XData', G(1), 'YData', G(2));
120    set(giuntoI, 'XData', I(1), 'YData', I(2));
121    set(giuntoL, 'XData', L(1), 'YData', L(2));
122    plot(H(1), H(2), 'r. '); %percorso piede
123
124    drawnow ;
125
126    t = t + 1 / frequenza_campionamento;
127    pause(1 / frequenza_campionamento);
128 end

```

9.2 STRIDER



MAGLIA ADE

$$PUNTOA : \begin{cases} A_x = z_1 \cos(\varphi_1) \\ A_y = z_1 \sin(\varphi_1) \end{cases}$$

$$z_x = \sqrt{(E_x - A_x)^2 + (E_y - A_y)^2}$$

$$\varphi_x = \arctan\left(\frac{E_y - A_y}{E_x - A_x}\right)$$

$$\gamma_{35} = \arccos\left(\frac{z_3^2 + z_5^2 - z_x^2}{2z_3z_5}\right)$$

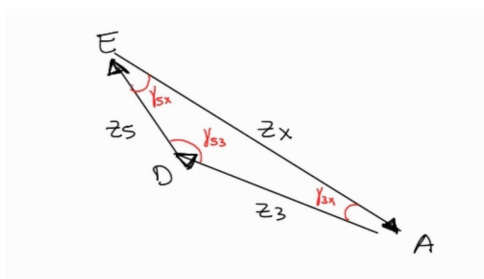
$$\gamma_{3x} = \arccos\left(\frac{z_3^2 + z_x^2 - z_5^2}{2z_3z_x}\right)$$

$$\gamma_{5x} = \pi - \gamma_{3x} - \gamma_{35}$$

$$\varphi_3 = \varphi_x + \pi - \gamma_{3x}$$

$$\varphi_5 = \varphi_x + \pi + \gamma_{5x}$$

$$\varphi_6 = \varphi_3 + \frac{\pi}{15}$$



MAGLIA ABC

$$z_y = \sqrt{(C_x - A_x)^2 + (C_y - A_y)^2}$$

$$\varphi_y = \arctan\left(\frac{C_y - A_y}{C_x - A_x}\right) + \pi$$

$$\gamma_{24} = \arccos\left(\frac{z_4^2 + z_2^2 - z_y^2}{2z_4z_2}\right)$$

$$\gamma_{2y} = \arccos\left(\frac{z_4^2 + z_2^2 - z_4^2}{2z_2z_y}\right)$$

$$\gamma_{y4} = \pi - \gamma_{24} - \gamma_{2y}$$

$$\varphi_2 = \varphi_y + \pi - \gamma_{2y}$$

$$\varphi_4 = \varphi_y + \pi + \gamma_{4y}$$

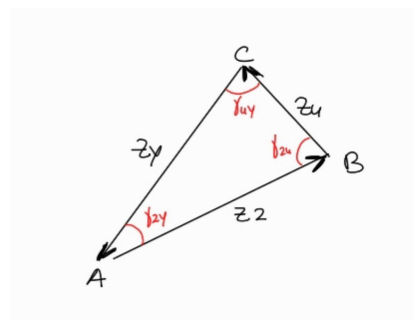
$$\varphi_7 = \varphi_2 - \frac{\pi}{15}$$

$$PUNTO D : \begin{cases} D_x = z_1 \cos(\varphi_1) + z_3 \cos(\varphi_3) \\ D_y = z_1 \sin(\varphi_1) + z_3 \sin(\varphi_3) \end{cases}$$

$$PUNTO G : \begin{cases} G_x = z_1 \cos(\varphi_1) - z_7 \cos(\varphi_7) \\ G_y = z_1 \sin(\varphi_1) - z_7 \sin(\varphi_7) \end{cases}$$

$$PUNTO B : \begin{cases} B_x = z_1 \cos(\varphi_1) + z_2 \cos(\varphi_2) \\ B_y = z_1 \sin(\varphi_1) + z_2 \sin(\varphi_2) \end{cases}$$

$$PUNTO F : \begin{cases} F_x = z_1 \cos(\varphi_1) - z_6 \cos(\varphi_6) \\ F_y = z_1 \sin(\varphi_1) - z_6 \sin(\varphi_6) \end{cases}$$



MAGLIA DGI

$$z_z = \sqrt{(D_x - G_x)^2 + (D_y - G_y)^2}$$

$$\varphi_z = \arctan\left(\frac{D_y - G_y}{D_x - G_x}\right) + \pi$$

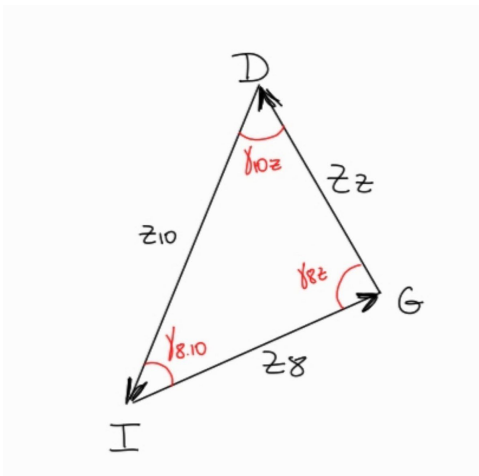
$$\gamma_{10.8} = \arccos\left(\frac{z_8^2 + z_{10}^2 - z_z^2}{2z_8z_{10}}\right)$$

$$\gamma_{8.z} = \arccos\left(\frac{z_8^2 + z_z^2 - z_{10}^2}{2z_8z_z}\right)$$

$$\gamma_{10.z} = \pi - \gamma_{10.8} - \gamma_{8.z}$$

$$\varphi_8 = \varphi_z + \pi + \gamma_{8.z}$$

$$\varphi_{10} = \varphi_z + \pi - \gamma_{10.z}$$



MAGLIA BFH

$$z_w = \sqrt{(B_x - F_x)^2 + (B_y - F_y)^2}$$

$$\varphi_w = \arctan\left(\frac{B_y - F_y}{B_x - F_x}\right) + \pi$$

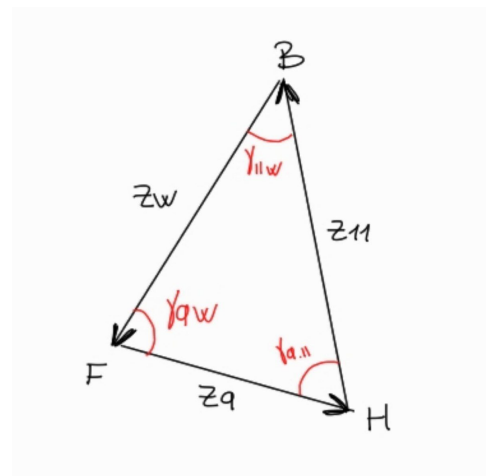
$$\gamma_{9.11} = \arccos\left(\frac{z_9^2 + z_{11}^2 - z_w^2}{2z_9z_{11}}\right)$$

$$\gamma_{9.w} = \arccos\left(\frac{z_9^2 + z_w^2 - z_{11}^2}{2z_9z_w}\right)$$

$$\gamma_{11.w} = \pi - \gamma_{9.11} - \gamma_{9.w}$$

$$\varphi_9 = \varphi_w + \pi - \gamma_{9.z}$$

$$\varphi_{11} = \varphi_w + \pi + \gamma_{11.w}$$



ANALISI DI POSIZIONE SU MATLAB

```

1 xlim ([-20,20])
2 % coordinate dei punti
3 0 = [ 0 , 0 ];
4 A = [ 3.999390780625565 , 0.069809625749134 ];
5 B = [ 15.888146791144774 , 7.462879388868063 ];
6 C = [ 11.1 , 7.8 ];
7 D = [ -9.639859678620823 , 3.227474413206339 ];

```

```

8 E = [ -11.1 , 7.8 ];
9 F = [ 9.798440216283574 , -0.035193435132622 ];
10 G = [ -1.455043452420048 , -1.902282460686010 ];
11 H = [ 18.558744893786294 , -5.259852462985439 ];
12 I = [ -8.058757293423735 , -9.676017953107676 ];
13
14 % lunghezze dei collegamenti
15 z1 = 4;
16 z2 = 14;
17 z3 = 14;
18 z4 = 4.8;
19 z5 = 4.8;
20 z6 = 5.8;
21 z7 = 5.8;
22 z8 = 10.2;
23 z9 = 10.2;
24 z10 = 13;
25 z11 = 13;
26
27 % inizializzazione variabili
28 t = 0;
29 theta1 = 0;
30
31 % parametri per il movimento della manovella
32 velocita_angular = 1; % vel angolare costante
33 tempo_simulazione = 2*pi; % definito in secondi
34 frequenza_campionamento = 100;
35
36 % linee tra i punti a telaio
37 hold on;
38 plot ([C(1),0(1),E(1),C(1)],[C(2),0(2),E(2),C(2)], 'k-','LineWidth'
, 1);
39 axis equal; % verifica che i dati siano proporzionali
40 plot([C(1),0(1),E(1),C(1)],[C(2),0(2),E(2),C(2)], 'ro');
41
42 z1_line = plot([0(1),A(1)],[0(2),A(2)], 'k-');
43 z2_line = plot([A(1),B(1)],[A(2),B(2)], 'k-');
44 z3_line = plot([A(1),D(1)],[A(2),D(2)], 'k-');
45 z4_line = plot([B(1),C(1)],[B(2),C(2)], 'k-');
46 z5_line = plot([D(1),E(1)],[D(2),E(2)], 'k-');
47 z6_line = plot([F(1),A(1)],[F(2),A(2)], 'k-');
48 z7_line = plot([G(1),A(1)],[G(2),A(2)], 'k-');
49 z8_line = plot([I(1),G(1)],[I(2),G(2)], 'k-');
50 z9_line = plot([H(1),F(1)],[H(2),F(2)], 'k-');
51 z10_line = plot([D(1),I(1)],[D(2),I(2)], 'k-');
52 z11_line = plot([B(1),H(1)],[B(2),H(2)], 'k-');
53
54 A_aggiornamento = plot ([A(1)],[A(2)], 'r.',MarkerSize=10) ;
55 B_aggiornamento = plot ([B(1)],[B(2)], 'r.',MarkerSize=10) ;
56 D_aggiornamento = plot ([D(1)],[D(2)], 'r.',MarkerSize=10) ;
57 F_aggiornamento = plot ([F(1)],[F(2)], 'r.',MarkerSize=10) ;
58 G_aggiornamento = plot ([G(1)],[G(2)], 'r.',MarkerSize=10) ;
59 H_aggiornamento = plot ([H(1)],[H(2)], 'r.',MarkerSize=10) ;
60 I_aggiornamento = plot ([I(1)],[I(2)], 'r.',MarkerSize=10) ;
61
62 % percorso_I = plot(I(1), I(2), 'r.');
```

```

63
64 while theta1 < tempo_simulazione
65
66     theta1 = theta1 + (velocita_angular/frequenza_campionamento);
67
68     % risoluzione diade A-D-E
69
70     A = [ 0(1) + z1*cos(theta1) , 0(2) + z1*sin(theta1) ];
71
72     z12 = sqrt ( ( E(2)-A(2) )^2 + ( E(1)-A(1) )^2 );
73     theta12 = atan2 ( ( A(2)-E(2) ) , ( A(1)-E(1) ) );
74
75     gamma_3_5 = acos ( ( z3^2 + z5^2 - z12^2 ) / ( 2*z3*z5 ) );
76     gamma_12_3 = acos ( ( z3^2 + z12^2 - z5^2 ) / ( 2*z3*z12 ) );
77     gamma_12_5 = pi - gamma_12_3 - gamma_3_5 ;
78
79     theta3 = theta12 + pi + gamma_12_3;
80     theta5 = theta12 + pi - gamma_12_5;
81     theta6 = theta3 + 0.2094 ;
82
83     % risoluzione diade A-B-C
84
85     z13 = sqrt ( ( C(2)-A(2) )^2 + ( C(1)-A(1) )^2 );
86     theta13 = atan2 ( ( A(2)-C(2) ) , ( A(1)-C(1) ) );
87
88     gamma_2_4 = acos ( ( z2^2 + z4^2 - z13^2 ) / ( 2*z2*z4 ) );
89     gamma_13_2 = acos ( ( z13^2 + z2^2 - z4^2 ) / ( 2*z13*z2 ) );
90     gamma_13_4 = pi - gamma_2_4 - gamma_13_2 ;
91
92     theta2 = theta13 - pi - gamma_13_2 ;
93     theta4 = theta13 - pi + gamma_13_4 ;
94     theta7 = theta2 - 0.2094 ;
95
96     % risoluzione diade D-G-I
97
98     D = [ z1*cos(theta1) + z3*cos(theta3) , z1*sin(theta1) + z3*sin
          (theta3) ] ;
99     G = [ z1*cos(theta1) - z7*cos(theta7) , z1*sin(theta1) - z7*sin
          (theta7) ] ;
100
101     z14 = sqrt ( ( D(2)-G(2) )^2 + ( D(1)-G(1) )^2 );
102     theta14 = atan2 ( ( D(2)-G(2) ) , ( D(1)-G(1) ) );
103
104     gamma_10_8 = acos ( ( z8^2 + z10^2 - z14^2 ) / ( 2*z8*z10 ) );
105     gamma_8_14 = acos ( ( z8^2 + z14^2 - z10^2 ) / ( 2*z8*z14 ) );
106     gamma_10_14 = pi - gamma_8_14 - gamma_10_8 ;
107
108     theta8 = theta14 - pi + gamma_8_14 ;
109     theta10 = theta14 + pi - gamma_10_14 ;
110
111     % risoluzione diade F-B-H
112
113     B = [ z1*cos(theta1) + z2*cos(theta2) , z1*sin(theta1) + z2*sin
          (theta2) ] ;
114     F = [ z1*cos(theta1) - z6*cos(theta6) , z1*sin(theta1) - z6*sin
          (theta6) ] ;

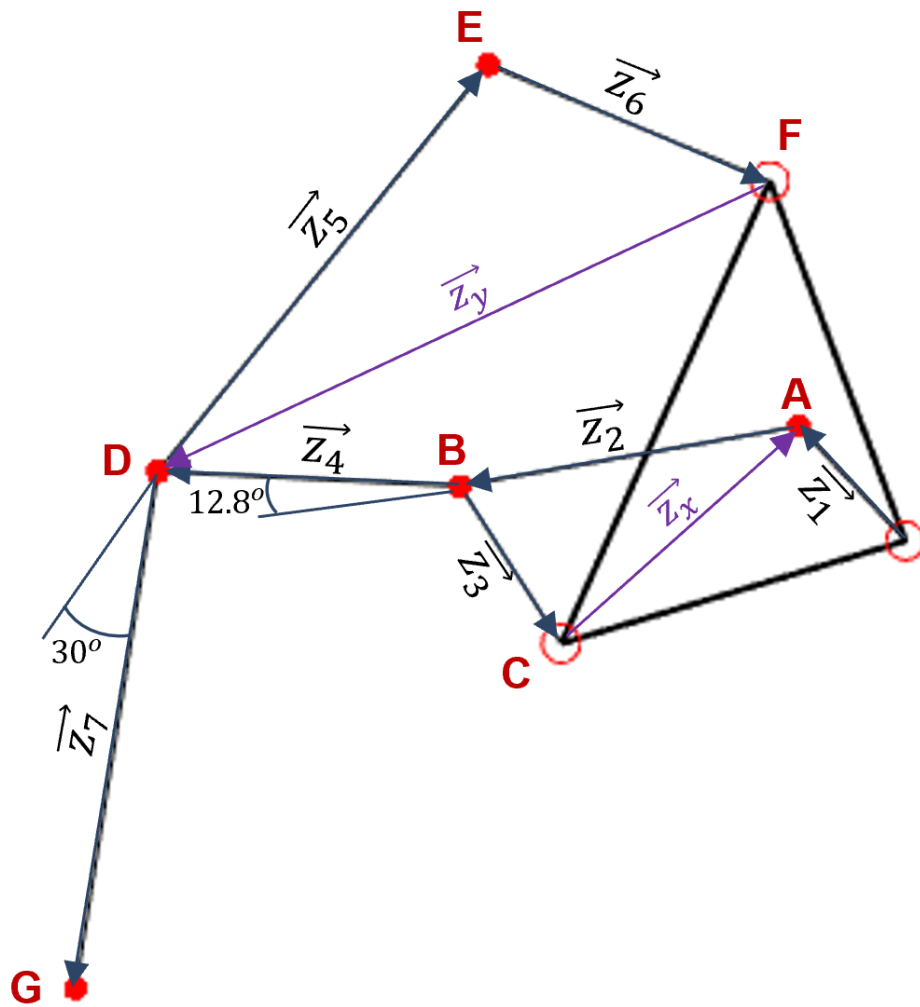
```

```

115
116 z15 = sqrt ( ( B(2)-F(2) )^2 + ( B(1)-F(1) )^2 );
117 theta15 = atan2( ( B(2)-F(2) ), ( B(1)-F(1) ) );
118
119 gamma_9_11 = acos ( ( z9^2 + z11^2 - z15^2 ) / ( 2*z9*z11 ) );
120 gamma_9_15 = acos ( ( z9^2 + z15^2 - z11^2 ) / ( 2*z9*z15 ) );
121 gamma_11_15 = pi - gamma_9_11 - gamma_9_15 ;
122
123 theta11 = theta15 + pi + gamma_11_15 ;
124 theta9 = theta15 + pi - gamma_9_15 ;
125
126 % aggiornamento delle posizioni dei giunti
127 A = [ 0(1) + z1*cos(theta1) , 0(2) + z1*sin(theta1) ] ;
128 B = [ A(1) + z2*cos(theta2) , A(2) + z2*sin(theta2) ] ;
129 D = [ A(1) + z3*cos(theta3) , A(2) + z3*sin(theta3) ] ;
130 C = [ B(1) + z4*cos(theta4) , B(2) + z4*sin(theta4) ] ;
131 E = [ D(1) + z5*cos(theta5) , D(2) + z4*sin(theta5) ] ;
132 F = [ A(1) - z6*cos(theta6) , A(2) - z6*sin(theta6) ] ;
133 G = [ I(1) + z8*cos(theta8) , I(2) + z8*sin(theta8) ] ;
134 H = [ F(1) - z9*cos(theta9) , F(2) - z9*sin(theta9) ] ;
135 I = [ D(1) + z10*cos(theta10) , D(2) + z10*sin(theta10) ] ;
136
137 % aggiornamento delle braccia del meccanismo
138 set(z1_line, 'XData', [0(1),A(1)] , 'YData', [0(2),A(2)]);
139 set(z2_line, 'XData', [A(1),B(1)] , 'YData', [A(2),B(2)]);
140 set(z3_line, 'XData', [A(1),D(1)] , 'YData', [A(2),D(2)]);
141 set(z4_line, 'XData', [B(1),C(1)] , 'YData', [B(2),C(2)]);
142 set(z5_line, 'XData', [D(1),E(1)] , 'YData', [D(2),E(2)]);
143 set(z6_line, 'XData', [F(1),A(1)] , 'YData', [F(2),A(2)]);
144 set(z7_line, 'XData', [G(1),A(1)] , 'YData', [G(2),A(2)]);
145 set(z9_line, 'XData', [H(1),F(1)] , 'YData', [H(2),F(2)]);
146 set(z8_line, 'XData', [I(1),G(1)] , 'YData', [I(2),G(2)]);
147 set(z10_line, 'XData', [D(1),I(1)] , 'YData', [D(2),I(2)]);
148 set(z11_line, 'XData', [B(1),H(1)] , 'YData', [B(2),H(2)]);
149
150 set(A_aggiornamento, 'XData', [A(1)] , 'YData', [A(2)]);
151 set(B_aggiornamento, 'XData', [B(1)] , 'YData', [B(2)]);
152 set(D_aggiornamento, 'XData', [D(1)] , 'YData', [D(2)]);
153 set(F_aggiornamento, 'XData', [F(1)] , 'YData', [F(2)]);
154 set(G_aggiornamento, 'XData', [G(1)] , 'YData', [G(2)]);
155 set(H_aggiornamento, 'XData', [H(1)] , 'YData', [H(2)]);
156 set(I_aggiornamento, 'XData', [I(1)] , 'YData', [I(2)]);
157
158 drawnow ;
159
160 t = t + 1 / frequenza_campionamento;
161
162 % set(percorso_I, 'XData', [get(percorso_I, 'XData'), I(1)], '
      YData', [get(percorso_I, 'YData'), I(2)]);
163
164 end

```

9.3 KLANN



MAGLIA ABC

$$z_x = \sqrt{(C_x - A_x)^2 + (C_y - A_y)^2}$$

$$\varphi_x = \arctan\left(\frac{C_y - A_y}{C_x - A_x}\right) + \pi$$

$$\gamma_{23} = \arccos\left(\frac{z_3^2 + z_3^2 - z_x^2}{2z_2z_3}\right)$$

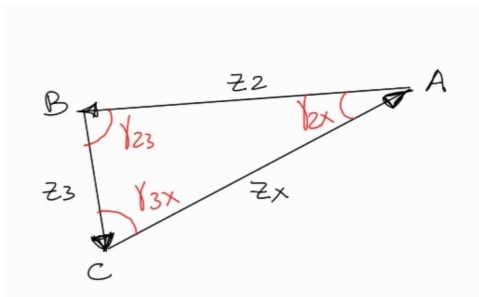
$$\gamma_{2x} = \arccos\left(\frac{z_2^2 + z_x^2 - z_3^2}{2z_2z_x}\right)$$

$$\gamma_{3x} = \pi - \gamma_{2x} - \gamma_{23}$$

$$\varphi_2 = \varphi_x + \pi - \gamma_{2x}$$

$$\varphi_3 = \varphi_x + \pi + \gamma_{3x}$$

$$\varphi_4 = \varphi_2 - \frac{12.8\pi}{180}$$



MAGLIA DEF

$$z_y = \sqrt{(D_x - F_x)^2 + (D_y - F_y)^2}$$

$$\varphi_y = \arctan\left(\frac{D_y - F_y}{D_x - F_x}\right) + \pi$$

$$\gamma_{56} = \arccos\left(\frac{z_5^2 + z_6^2 - z_y^2}{2z_5z_6}\right)$$

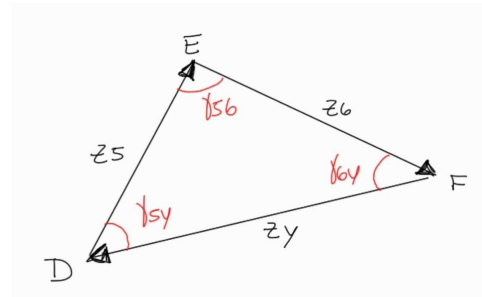
$$\gamma_{6y} = \arccos\left(\frac{z_6^2 + z_y^2 - z_5^2}{2z_6z_y}\right)$$

$$\gamma_{5y} = \pi - \gamma_{6y} - \gamma_{56}$$

$$\varphi_5 = \varphi_y + \pi + \gamma_{5y}$$

$$\varphi_6 = \varphi_y + \pi - \gamma_{6y}$$

$$\varphi_7 = \varphi_5 + \pi + \frac{\pi}{6}$$



ANALISI DI POSIZIONE SU MATLAB

```

1 xlim ([-20,20])
2 ylim ([0,20])
3
4 % coordinate
5 O = [10 , 10];
6 A = [12.999543085469174,10.052357219311851];
7 B = [6.399733431436177,10.002232070605318];
8 C = [3.4 , 8.03];
9 D = [0.744036600655769,11.242068091335660];
10 E = [3.215000641302922,20.973251817369647];

```

```

11 F = [7.4,16.9];
12 G = [3.469722663945678,1.579137092700353];
13
14 % lunghezze dei collegamenti
15 z1 = 3;
16 z2 = 6.6;
17 z3 = 3.59;
18 z4 = 5.79;
19 z5 = 5.84;
20 z6 = 10.04;
21 z7 = 10.04;
22
23 % inizializzazione variabili
24
25 t = 0;
26 theta1 = 0;
27
28 % parametri per il movimento della manovella
29 velocita_angular = 1; % vel angolare costante
30 tempo_simulazione = 2*pi; % definito in secondi
31 frequenza_campionamento = 50;
32
33 % linee tra i punti a telaio
34 hold on;
35 plot ([C(1),0(1),F(1),C(1)],[C(2),0(2),F(2),C(2)], 'k-', 'LineWidth
    ', 1);
36 axis equal; % verifica che i dati siano proporzionali
37 plot([C(1),0(1),F(1),C(1)],[C(2),0(2),F(2),C(2)], 'ro');
38
39 z1_line = plot([0(1),A(1)],[0(2),A(2)], 'k-');
40 z2_line = plot([A(1),B(1)],[A(2),B(2)], 'k-');
41 z3_line = plot([B(1),C(1)],[B(2),C(2)], 'k-');
42 z4_line = plot([B(1),D(1)],[B(2),D(2)], 'k-');
43 z6_line = plot([D(1),E(1)],[D(2),E(2)], 'k-');
44 z5_line = plot([E(1),F(1)],[E(2),F(2)], 'k-');
45 z7_line = plot([D(1),G(1)],[D(2),G(2)], 'k-');
46
47 A_aggiornamento = plot ([A(1)],[A(2)], 'r.',MarkerSize=10) ;
48 B_aggiornamento = plot ([B(1)],[B(2)], 'r.',MarkerSize=10) ;
49 D_aggiornamento = plot ([D(1)],[D(2)], 'r.',MarkerSize=10) ;
50 E_aggiornamento = plot ([E(1)],[E(2)], 'r.',MarkerSize=10) ;
51 G_aggiornamento = plot ([G(1)],[G(2)], 'r.',MarkerSize=10) ;
52
53 % percorso_G = plot(G(1), G(2), 'r. ');
54
55 % calcolo angoli interni primo ciclo
56
57 z20 = sqrt( ( (B(1)-0(1))^2 ) + ( (B(2)-0(2))^2 ) );
58 gamma_1_2 = acos ( ( z1^2 + z2^2 - z20^2 ) / ( 2*z1*z2 ) );
59
60 while t < tempo_simulazione
61
62     theta1 = theta1 + velocita_angular/frequenza_campionamento;
63
64     % risoluzione diade A-B-C
65

```

```

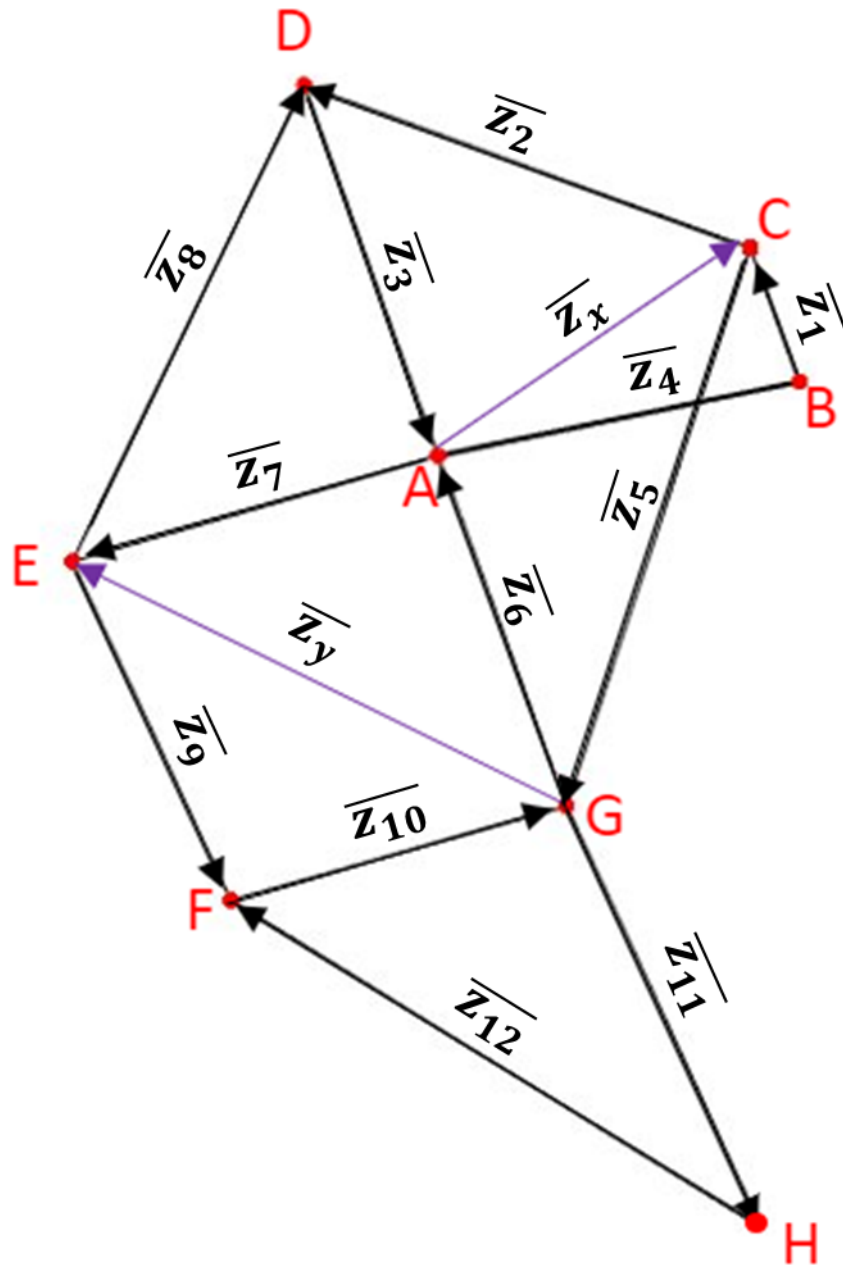
66     A = [ 0(1) + z1*cos(theta1) , 0(2) + z1*sin(theta1) ];
67
68     z9 = sqrt( (A(1)-C(1))^2 + (A(2)-C(2))^2 );
69     theta9 = atan2 ( (A(2)-C(2)) , (A(1)-C(1)) ) ;
70
71     gamma_2_3 = acos ( (z2^2+z3^2-z9^2) / (2*z2*z3) ) ;
72     gamma_2_9 = acos ( (z2^2+z9^2-z3^2) / (2*z2*z9) ) ;
73     gamma_3_9 = pi - gamma_2_3 - gamma_2_9 ;
74
75     theta2 = theta9 + pi - gamma_2_9 ;
76     theta3 = theta9 + pi + gamma_3_9 ;
77     theta4 = theta2 - 0.2234;
78
79     % risoluzione diade F-E-D
80
81     D = [ A(1)+z2*cos(theta2)+z4*cos(theta4) , A(2)+z2*sin(theta2)+
           z4*sin(theta4) ];
82
83     z8 = sqrt( (F(1)-D(1))^2 + (F(2)-D(2))^2 ) ;
84     theta8 = atan2 ( (D(2)-F(2)) , (D(1)- F(1)) ) ;
85
86     gamma_5_6 = acos( (z5^2+z6^2-z8^2) / (2*z5*z6) ) ;
87     gamma_5_8 = acos( (z5^2+z8^2-z6^2) / (2*z5*z8) ) ;
88     gamma_6_8 = pi - gamma_5_6 - gamma_5_8;
89
90     theta5 = theta8 + pi - gamma_5_8;
91     theta6 = theta8 - pi + gamma_6_8;
92     theta7 = theta6 + pi + 0.5236;
93
94     A = [ 0(1) + z1*cos(theta1), 0(2) + z1*sin(theta1)];
95     B = [ A(1) + z2*cos(theta2), A(2) + z2*sin(theta2)];
96     C = [ B(1) + z3*cos(theta3), B(2) + z3*sin(theta3)];
97     D = [ B(1) + z4*cos(theta4), B(2) + z4*sin(theta4)];
98     E = [ D(1) + z6*cos(theta6), D(2) + z6*sin(theta6)];
99     F = [ E(1) + z5*cos(theta5), E(2) + z5*sin(theta5)];
100    G = [ D(1) + z7*cos(theta7), D(2) + z7*sin(theta7)];
101
102    set(z1_line, 'XData' , [0(1),A(1)] , 'YData' , [0(2),A(2)]);
103    set(z2_line, 'XData' , [A(1),B(1)] , 'YData' , [A(2),B(2)]);
104    set(z3_line, 'XData' , [B(1),C(1)] , 'YData' , [B(2),C(2)]);
105    set(z4_line, 'XData' , [B(1),D(1)] , 'YData' , [B(2),D(2)]);
106    set(z6_line, 'XData' , [D(1),E(1)] , 'YData' , [D(2),E(2)]);
107    set(z5_line, 'XData' , [E(1),F(1)] , 'YData' , [E(2),F(2)]);
108    set(z7_line, 'XData' , [D(1),G(1)] , 'YData' , [D(2),G(2)]);
109
110    set(A_aggiornamento, 'XData' , [A(1)] , 'YData' , [A(2)]);
111    set(B_aggiornamento, 'XData' , [B(1)] , 'YData' , [B(2)]);
112    set(D_aggiornamento, 'XData' , [D(1)] , 'YData' , [D(2)]);
113    set(E_aggiornamento, 'XData' , [E(1)] , 'YData' , [E(2)]);
114    set(G_aggiornamento, 'XData' , [G(1)] , 'YData' , [G(2)]);
115
116    drawnow ;
117
118    t = t + 1 / frequenza_campionamento;
119    pause(1 / frequenza_campionamento);
120

```



```
121 % set(percorso_G, 'XData', [get(percorso_G, 'XData'), G(1)], '
    YData', [get(percorso_G, 'YData'), G(2)]);
122
123 end
```

9.4 STRANDBEEST



MAGLIA ABCD

$$z_4 = \sqrt{(38)^2 + (7.8)^2} \simeq 38.8$$

$$\varphi_4 = \arctan\left(\frac{7.8}{38}\right) \simeq 0.2025$$

$$z_x = \sqrt{(38 + z_1 \cos \varphi_1)^2 + (7.8 + z_1 \sin \varphi_1)^2}$$

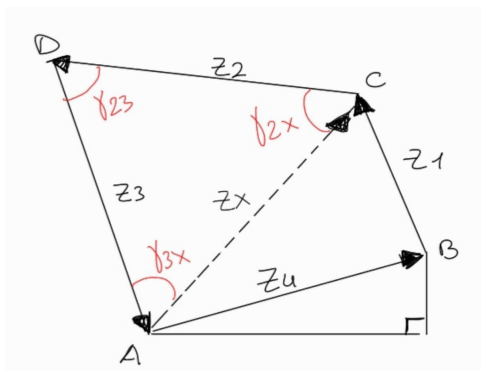
$$\varphi_x = \arctan\left(\frac{7.8 + z_1 \sin \varphi_1}{38 + z_1 \cos \varphi_1}\right)$$

$$\gamma_{23} = \arccos\left(\frac{z_3^2 + z_3^2 - z_x^2}{2z_2z_3}\right)$$

$$\gamma_{2x} = \arccos\left(\frac{z_2^2 + z_x^2 - z_3^2}{2z_2z_x}\right)$$

$$\varphi_2 = \varphi_x + \pi - \gamma_{2x}$$

$$\varphi_3 = \varphi_x + \pi - \gamma_{3x}$$



MAGLIA ACG

$$\gamma_{56} = \arccos\left(\frac{z_5^2 + z_6^2 - z_x^2}{2z_5z_6}\right)$$

$$\gamma_{5x} = \arccos\left(\frac{z_5^2 + z_x^2 - z_6^2}{2z_5z_x}\right)$$

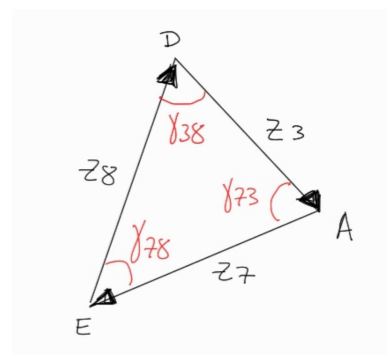
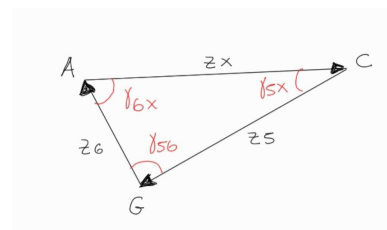
$$\gamma_{6x} = \pi - \gamma_{5x} - \gamma_{56}$$

$$\varphi_6 = \varphi_x + \pi - \gamma_{6x}$$

MAGLIA ADE

$$\gamma_{73} = \arccos\left(\frac{z_7^2 + z_3^2 - z_8^2}{2z_7z_3}\right)$$

$$\varphi_7 = \varphi_3 + \pi - \gamma_{73}$$



MAGLIA EFG

$$z_y = \sqrt{(E_x - G_x)^2 + (E_y - G_y)^2}$$

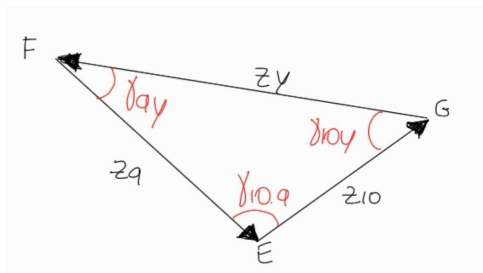
$$\varphi_y = \arctan\left(\frac{E_y - G_y}{E_x - G_x}\right) + \pi$$

$$\gamma_{10,y} = \arccos\left(\frac{z_{10}^2 + z_y^2 - z_9^2}{2z_{10}z_y}\right)$$

$$\gamma_{9y} = \arccos\left(\frac{z_y^2 + z_9^2 - z_{10}^2}{2z_9z_y}\right)$$

$$\varphi_9 = \varphi_y + \pi - \gamma_{9y}$$

$$\varphi_{10} = \varphi_y + \pi + \gamma_{10,y}$$



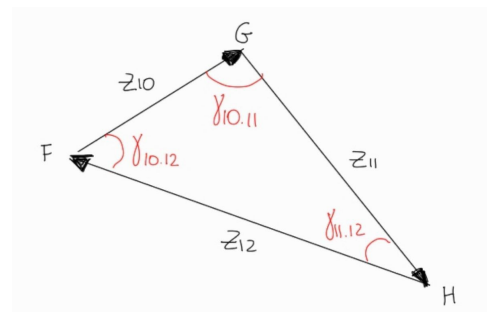
MAGLIA FGH

$$\gamma_{10,11} = \arccos\left(\frac{z_{10}^2 + z_{11}^2 - z_{12}^2}{2z_{10}z_{11}}\right)$$

$$\gamma_{10,12} = \arccos\left(\frac{z_{10}^2 + z_{12}^2 - z_{11}^2}{2z_{10}z_{12}}\right)$$

$$\varphi_{11} = \varphi_{10} + \pi + \gamma_{10,11}$$

$$\varphi_{12} = \varphi_{10} + \pi - \gamma_{10,12}$$



ANALISI DI POSIZIONE SU MATLAB

```

1 %STRANDBEEST
2 f1 = figure('Name','Assieme','NumberTitle','off') ;
3 xlim([-100,100])
4 ylim([-110,60])
5 title('STRANDBEEST');
6 %definisco le coordinate dei punti a telaio
7 A = [0, 0];
8 B = [38, 7.8];
9 %definisco i punti iniziali
10 C = [30.5, 20.79];
11 D = [-15.95, 39.32];
12 E = [-36.23, -17.18];
13 F = [-30.44, -56.15];
14 G = [2.11, -39.24];
15 H = [31.28, -78.61];
16 %lunghezze dei vari membri

```

```

17 z1 = 15;
18 z2 = 50;
19 z3 = 41.5;
20 z4 = 38.8; theta4 = 0.2025; %telaio
21 z5 = 61.9;
22 z6 = 39.3;
23 z7 = 40.1;
24 z8 = 55.8;
25 z9 = 39.4;
26 z10 = 36.7;
27 z11 = 49;
28 z12 = 65.7;
29 %imposto il tempo a 0
30 t = 0;
31 %definisco l'angolo iniziale della manovella
32 theta1 = 0;
33 % parametri per il movimento della manovella
34 w = 1; % vel angolare costante
35 tempo_simulazione = 2*pi;
36 frequenza_campionamento = 100;
37 %linea tra i punti a telaio
38 hold on;
39 z4_line = plot([A(1), B(1)], [A(2), B(2)], 'k-', 'LineWidth', 1 );
40 axis equal;
41 %definisco le articolazioni
42 z1_line = plot([B(1), C(1)], [B(2), C(2)], 'b-');
43 z2_line = plot([C(1), D(1)], [C(2), D(2)], 'b-');
44 z3_line = plot([D(1), A(1)], [D(2), A(2)], 'b-');
45 z5_line = plot([C(1), G(1)], [C(2), G(2)], 'b-');
46 z6_line = plot([G(1), A(1)], [G(2), A(2)], 'b-');
47 z7_line = plot([A(1), E(1)], [A(2), E(2)], 'b-');
48 z8_line = plot([E(1), D(1)], [E(2), D(2)], 'b-');
49 z9_line = plot([E(1), F(1)], [E(2), F(2)], 'b-');
50 z10_line = plot([F(1), G(1)], [F(2), G(2)], 'b-');
51 z11_line = plot([G(1), H(1)], [G(2), H(2)], 'b-');
52 z12_line = plot([H(1), F(1)], [H(2), F(2)], 'b-');
53 giuntoA = plot(A(1), A(2), 'ro');
54 giuntoB = plot(B(1), B(2), 'ro');
55 giuntoC = plot(C(1), C(2), 'r.', 'MarkerSize', 10);
56 giuntoD = plot(D(1), D(2), 'r.', 'MarkerSize', 10);
57 giuntoE = plot(E(1), E(2), 'r.', 'MarkerSize', 10);
58 giuntoF = plot(F(1), F(2), 'r.', 'MarkerSize', 10);
59 giuntoG = plot(G(1), G(2), 'r.', 'MarkerSize', 10);
60 %inizio col far variare l'angolo della manovella
61 while t < tempo_simulazione
62     theta1 = theta1 + w/frequenza_campionamento;
63     %risoluzione BCDA
64     C = [B(1) + z1*cos(theta1), B(2) + z1*sin(theta1)];
65     thetax = atan((B(2)+z1*sin(theta1))/(B(1)+z1*cos(theta1)));
66     zx = sqrt((B(1)+z1*cos(theta1))^2 + (B(2)+z1*sin(theta1))^2);
67     gamma23 = acos((z2^2 + z3^2 - zx^2)/(2*z2*z3));
68     gamma2x = acos((z2^2 + zx^2 - z3^2)/(2*z2*zx));
69     gamma3x = acos((z3^2 + zx^2 - z2^2)/(2*z3*zx));
70     theta2 = thetax + pi - gamma2x;
71     theta3 = theta2 + pi - gamma23;
72     D = [C(1)+z2*cos(theta2), C(2)+z2*sin(theta2)];

```

```

73 %risoluzione CGA
74 gamma56 = acos((z5^2 + z6^2 - zx^2)/(2*z5*z6));
75 gamma5x = acos((z5^2 + zx^2 - z6^2)/(2*z5*zx));
76 gamma6x = acos((z6^2 + zx^2 - z5^2)/(2*zx*z6));
77 theta5 = thetax - pi + gamma5x;
78 theta6 = thetax + pi - gamma6x;
79 G = [C(1)+ z5*cos(theta5),C(2)+ z5*sin(theta5)];
80 %risoluzione AED
81 gamma73 = acos((z7^2+z3^2-z8^2)/(2*z7*z3));
82 gamma83 = acos((z8^2+z3^2-z7^2)/(2*z8*z3));
83 gamma78 = pi - gamma73 - gamma83;
84 theta7 = theta3 - pi + gamma73;
85 theta8 = theta3 + pi - gamma83;
86 E = [A(1)+z7*cos(theta7), A(2)+z7*sin(theta7)];
87 %risoluzione AEEFG
88 zy = sqrt((G(1)-E(1))^2 + (G(2)- E(2))^2);
89 thetay = pi + atan((G(2)-E(2))/(G(1)-E(1)));
90 gamma10y = acos((z10^2+zy^2-z9^2)/(2*z10*zy));
91 gamma9y = acos((z9^2+zy^2-z10^2)/(2*z9*zy));
92 gamma910 = pi - gamma10y - gamma9y;
93 theta9 = thetay + pi - gamma9y;
94 theta10 = thetay - pi + gamma10y;
95 F = [(E(1)+z9*cos(theta9)), (E(2)+z9*sin(theta9))];
96 %risoluzione GHF
97 gamma1011 = acos((z10^2+z11^2-z12^2)/(2*z10*z11));
98 gamma1012 = acos((z10^2+z12^2-z11^2)/(2*z10*z12));
99 gamma1112 = pi - gamma1011 - gamma1012;
100 theta11 = theta10 + pi + gamma1011;
101 theta12 = theta10 + pi - gamma1012;
102 H = [(G(1)+z11*cos(theta11)), (G(2)+z11*sin(theta11))];
103
104 set(z1_line, 'XData', [B(1),C(1)], 'YData', [B(2),C(2)]);
105 set(z2_line, 'XData', [C(1),D(1)], 'YData', [C(2),D(2)]);
106 set(z3_line, 'XData', [D(1),A(1)], 'YData', [D(2),A(2)]);
107 set(z5_line, 'XData', [C(1),G(1)], 'YData', [C(2),G(2)]);
108 set(z6_line, 'XData', [G(1),A(1)], 'YData', [G(2),A(2)]);
109 set(z7_line, 'XData', [A(1),E(1)], 'YData', [A(2),E(2)]);
110 set(z8_line, 'XData', [E(1),D(1)], 'YData', [E(2),D(2)]);
111 set(z9_line, 'XData', [E(1),F(1)], 'YData', [E(2),F(2)]);
112 set(z10_line, 'XData', [F(1),G(1)], 'YData', [F(2),G(2)]);
113 set(z11_line, 'XData', [G(1),H(1)], 'YData', [G(2),H(2)]);
114 set(z12_line, 'XData', [H(1),F(1)], 'YData', [H(2),F(2)]);
115 set(giuntoA, 'XData', A(1), 'YData', A(2));
116 set(giuntoB, 'XData', B(1), 'YData', B(2));
117 set(giuntoC, 'XData', C(1), 'YData', C(2));
118 set(giuntoD, 'XData', D(1), 'YData', D(2));
119 set(giuntoE, 'XData', E(1), 'YData', E(2));
120 set(giuntoF, 'XData', F(1), 'YData', F(2));
121 set(giuntoG, 'XData', G(1), 'YData', G(2));
122 plot(H(1), H(2), 'r. '); %percorso piede
123
124 drawnow ;
125
126 t = t + 1 / frequenza_campionamento;
127 pause(1 / frequenza_campionamento);
128 end

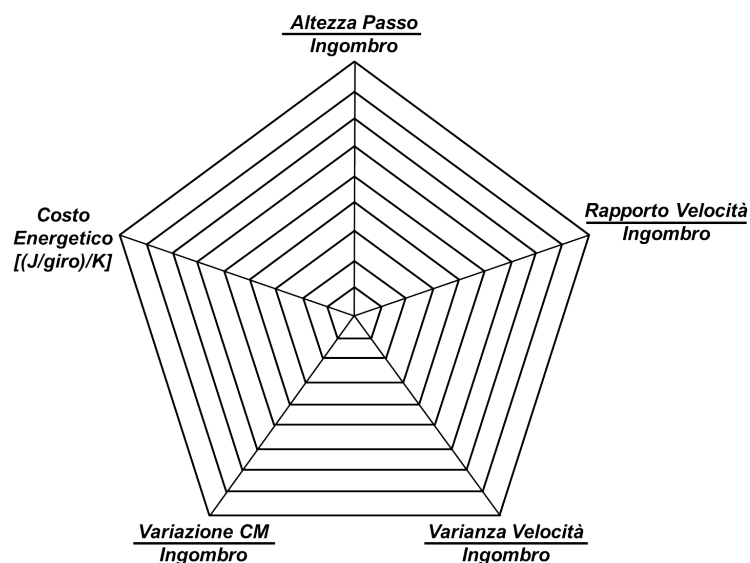
```

Capitolo 10

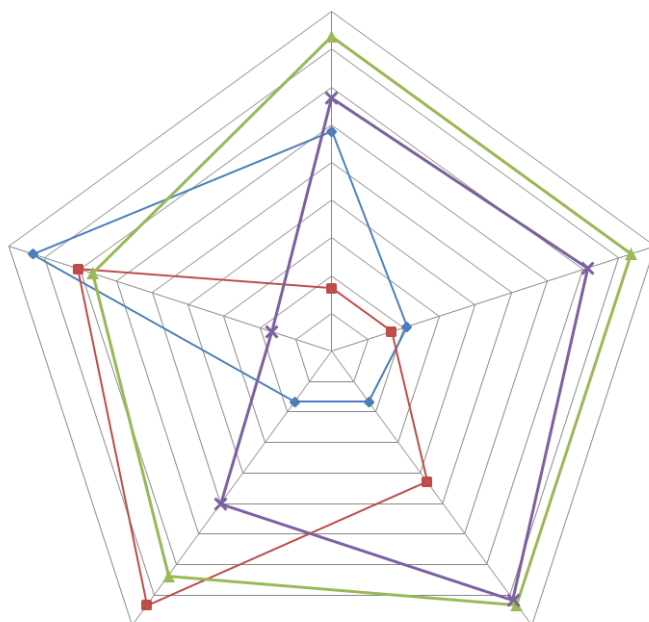
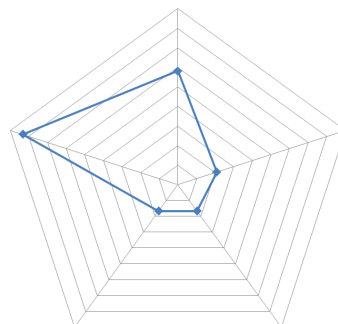
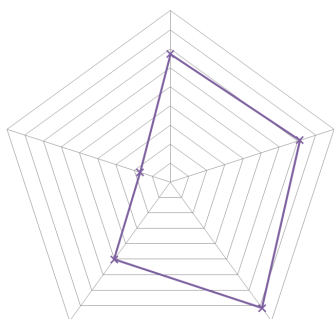
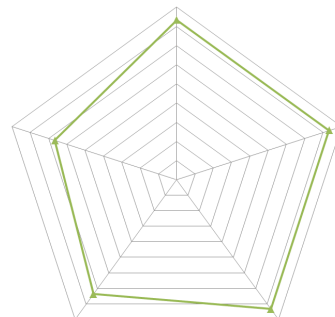
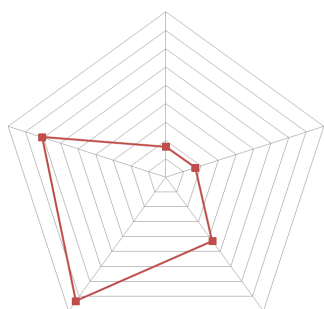
CONCLUSIONI E CONFRONTO FINALE

Grazie alle analisi fatte ci è stato possibile confrontare i 4 meccanismi per i 5 parametri studiati. In base all'utilizzo richiesto dal meccanismo è possibile scegliere la configurazione che offre la miglior prestazione. Per poter confrontare i 4 meccanismi sotto un punto di vista complessivo, in modo da trovare il più equilibrato, ci siamo posti l'obiettivo di creare un grafico radar che mostri visivamente le **potenzialità e carenze** di ognuno. Per avere una visualizzazione ottimale, per ognuno dei 5 parametri, abbiamo considerato la differenza di valore tra la configurazione migliore e la peggiore, e la abbiamo moltiplicato per un fattore di scala di 1,5 per trovare il range di visualizzazione da applicare al grafico.

In questo modo il range rappresentato dipende dai valori stessi dei risultati di ogni parametro e si può visualizzare come si collocano tra loro i 4 meccanismi. Si sono quindi riposizionati i vari risultati degli studi in base al relativo range. Per gli studi nei quali un valore minore rappresenta un risultato migliore è stata effettuata un'inversione dei valori in scala. È stato effettuato questo procedimento siccome non si hanno a disposizione dei valori di riferimento che ci permettono di capire se un meccanismo ha prodotto valori buoni o scadenti, ci è possibile solo confrontarli tra loro valutando il migliore, il peggiore e come gli altri 2 si dispongono di conseguenza.



STRANDBEEST
TROTBOT
STRIDER
KLANN



Si giunge così alla conclusione che il meccanismo migliore nella nostra analisi risulta essere lo **STRIDER.**

Capitolo 11

SITOGRAFIA

- DIYWALKERS-TROTBOT
<https://www.diywalkers.com/trotbot-builds.html>
- DIYWALKERS-STRIDER
<https://www.diywalkers.com/strider-linkage-plans.html>
- DIYWALKERS-STRANDBEEST
<https://www.diywalkers.com/strandbeest.html>
- DIYWALKERS-KLANN
<https://www.diywalkers.com/klann-mechanical-spider---climbing-mod.html>
- WIKIPEDIA-LEG MECHANISM
[https://en.wikipedia.org/wiki/Leg_mechanism#:~:text=A%20leg%20mechanism%20\(walking%20mechanism,frictional%20contact%20with%20the%20ground.](https://en.wikipedia.org/wiki/Leg_mechanism#:~:text=A%20leg%20mechanism%20(walking%20mechanism,frictional%20contact%20with%20the%20ground.)
- WIKIPEDIA-JENSEN.LINKAGE
https://en.wikipedia.org/wiki/Jansen%27s_linkage
- WIKIPEDIA-KLANN LINKAGE
https://en.wikipedia.org/wiki/Klann_linkage
- STRANDBEEST
<https://www.strandbeest.com/>

Ringraziamenti

