UNIVERSITÀ DEGLI STUDI DI PADOVA

UNIVERSITA' DEGLI STUDI DI PADOVA

Dipartimento di Ingegneria Industriale DII

Corso di Laurea in Ingegneria Meccanica

SINTESI OTTIMALE DEL MECCANISMO SOSPENSIVO POSTERIORE CON IL METODO DELLE COORDINATE NATURALI

Relatore: Ch.mo Prof. Vittore Cossalter
Correlatore: Ing. Matteo Massaro

Laureando: Nicolò Ceranto
Matr. 1078895

Anno Accademico 2015/2016

“Da dove vieni è irrilevante,

è ciò che fai del dono della vita

che stabilisce chi sei”

INDICE

SOMMARIO

Il presente lavoro tratta lo sviluppo di un algoritmo per la descrizione e ottimizzazione della sospensione posteriore motociclistica. Nella storia si sono sviluppate diverse tipologie di sistemi sospensivi e in questa sede ci si intende concentrare nello schema a quadrilatero biella-telaio, data la capacità di generare curve di rigidezza molto variegate. A partire dallo studio cinematico del quadrilatero articolato si passa in seguito all'analisi dell'intera sospensione. Il parametro su cui si realizzano le ottimizzazioni è il rapporto tra la velocità di compressione della molla e la velocità di sollevamento verticale dello pneumatico posteriore. Le procedure di ottimizzazione avvengono per via numerica: nel software di calcolo $Matlab$ viene implementata l'intera cinematica e i relativi algoritmi di ottimizzazione: al termine delle iterazioni vengono presentate le dimensioni geometriche ottimali degli elementi facenti parte del sistema sospensivo. Per esplorare il numero più elevato possibile di soluzioni si fa riferimento a diversi metodi matematici basati su altrettanto diverse strutture risolutrici. Vengono infine presentati degli esempi di ottimizzazione per sospensioni a comportamento lineare, progressivo e regressivo.

1. SOSPENSIONE MOTOCICLISTICA POSTERIORE

1.1. INTRODUZIONE

Poiché spesso le strade cittadine risultano avere un manto irregolare, la guida di un motociclo privo di sospensioni risulta essere particolarmente difficoltosa, a causa del disagio avvertito dal pilota a seguito della perdita di aderenza sul piano stradale. Gli pneumatici sono sufficienti per assorbire piccole asperità, ma necessitano di un supporto per affrontare quelle di maggiore entità: di qui la necessità di ricorrere a sistemi sospensivi.

Nel dettaglio, le sospensioni devono contemporaneamente pervenire ai seguenti scopi:

- garantire la corretta aderenza degli pneumatici al manto stradale, al fine di trasferire nel modo ottimale le forze di trazione e frenata imposte dal veicolo;
- assicurare l'assetto desiderato della motocicletta nelle diverse condizioni di utilizzo dello stesso;
- eseguire un corretto isolamento delle masse sospese dalle vibrazioni risultanti dall'interazione degli pneumatici con il terreno: in questo senso le sospensioni devono conseguire il duplice obiettivo di permettere alle ruote di seguire il profilo della strada e garantire un ottimale comfort di guida al pilota.

La soddisfazione contemporanea di questi aspetti risulta essere quasi impossibile: per massimizzare il comfort sono infatti necessarie delle molle molto soffici, le quali tuttavia presentano una notevole escursione nelle fasi di accelerazione e frenata con un conseguente difficile mantenimento di assetto e stabilità desiderati [1]. L'importanza di queste tre funzioni viene quindi variata in base all'utilizzo del veicolo in questione: le moto da Gran Premio privilegiano assetto e aderenza, al fine di massimizzare la prestazione in gara, e riservano al comfort un'importanza secondaria; d'altro canto le moto da turismo devono assicurare un elevato comfort di guida e quindi relegano la prestazione ad un ruolo minore.

Un altro aspetto fondamentale è legato al fatto che un particolare assetto è in grado di soddisfare al meglio una particolare condizione di guida, ad esempio con la strada asciutta, mentre con terreno bagnato può rispondere in modo totalmente diverso; in egual misura la variazione dei carichi sul motoveicolo stesso (diversa corporatura dei piloti, presenza di un passeggero, etc. …) comporta, a parità di condizioni stradali, una variazione di risposta del sistema-veicolo.

Per soddisfare tutte queste necessità sono state sviluppate diverse tipologie di sospensione posteriore motociclistica, caratterizzate dal proprio comportamento e da vantaggi/svantaggi che verranno in seguito esposti.

1.2. FORCELLONE OSCILLANTE

Tradizionalmente la sospensione posteriore classica è composta da un forcellone composto da due bracci oscillanti ognuno dei quali dotato di una propria unità molla-smorzatore.

Figura 1: schema della sospensione a forcellone oscillante tradizionale e applicazione su Guzzi California

Gli angoli particolari con cui vengono realizzati tali schemi consentono di gestire la risposta del sistema e di ottenere una lieve progressività. La semplicità costruttiva di tale apparato ne ha permesso una vasta distribuzione ed utilizzo da parte dei principali costruttori a tal punto da risultare lo schema sospensivo più diffuso; in aggiunta a ciò si evidenziano una facile dissipazione del calore prodotto dagli ammortizzatori, delle basse forze reattive trasmesse al telaio attraverso la coppia rotoidale ed una grande ampiezza di movimento del gruppo molla-smorzatore, la quale costringe gli ammortizzatori a lavorare ad alte velocità di compressione ed estensione e rende più agevole il controllo dello smorzamento. Per contro questo sistema presenta una limitata oscillazione verticale della ruota, un comportamento non molto progressivo e la possibilità di ottenere degli sforzi torcenti sul forcellone, causati da variazioni nei precarichi delle due molle o nelle caratteristiche degli ammortizzatori.

1.3. FORCELLONE CANTILEVER

Per contrastare i principali difetti del forcellone classico si ricorre quindi al sistema "cantilever": esso è composto da un forcellone a bracci uniti con un'unica unità molla-ammortizzatore.

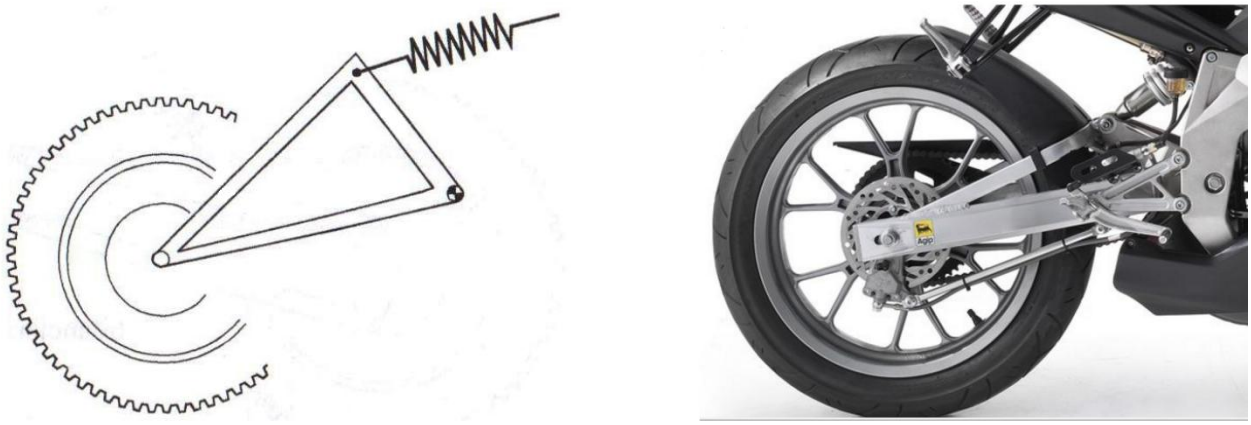


Figura 2: schema della sospensione "cantilever" e applicazione su Aprilia RSV4 Factory

Questa soluzione riduce le masse sospese in gioco, realizza elevate rigidezze torsionali e flessionali, permette una notevole escursione verticale della ruota posteriore, riduce notevolmente il rischio di sforzi torsionali sul forcellone e consente un facile aggiustamento o taratura dell'unità, in quanto è presente un singolo ammortizzatore. La problematica principale è legata alla posizione dell'ammortizzatore stesso che, posto sopra o dietro al motore, può provocare problemi di dissipazione del calore; oltre a ciò, al pari del forcellone tradizionale, la sospensione cantilever non consente di ottenere una caratteristica forza-spostamento progressiva. Una possibile variante del sistema sospensivo, con caratteristiche di funzionamento simili, può prevedere il posizionamento laterale del gruppo molla-ammortizzatore.

Figura 3: Aprilia SMV 750 Dorsoduro con applicazione del meccanismo sospensivo "Cantilever" laterale

1.4. MECCANISMI

La necessità di ottenere delle particolari curve di rigidezza desiderate (e quindi un comportamento progressivo) porta all'introduzione di meccanismi nello schema della sospensione: generalmente essi sono basati sul quadrilatero. Le varie case costruttrici hanno risposto a queste esigenze ricorrendo a diverse soluzioni che si differenziano principalmente per il punto di collegamento del gruppo molla-ammortizzatore. Esso si può quindi realizzare tra il telaio e il bilancere (Kawasaki-Unitrak), tra la biella e il telaio (ProLink-Honda) o tra il forcellone e il bilancere (Full Floater-Suzuki).

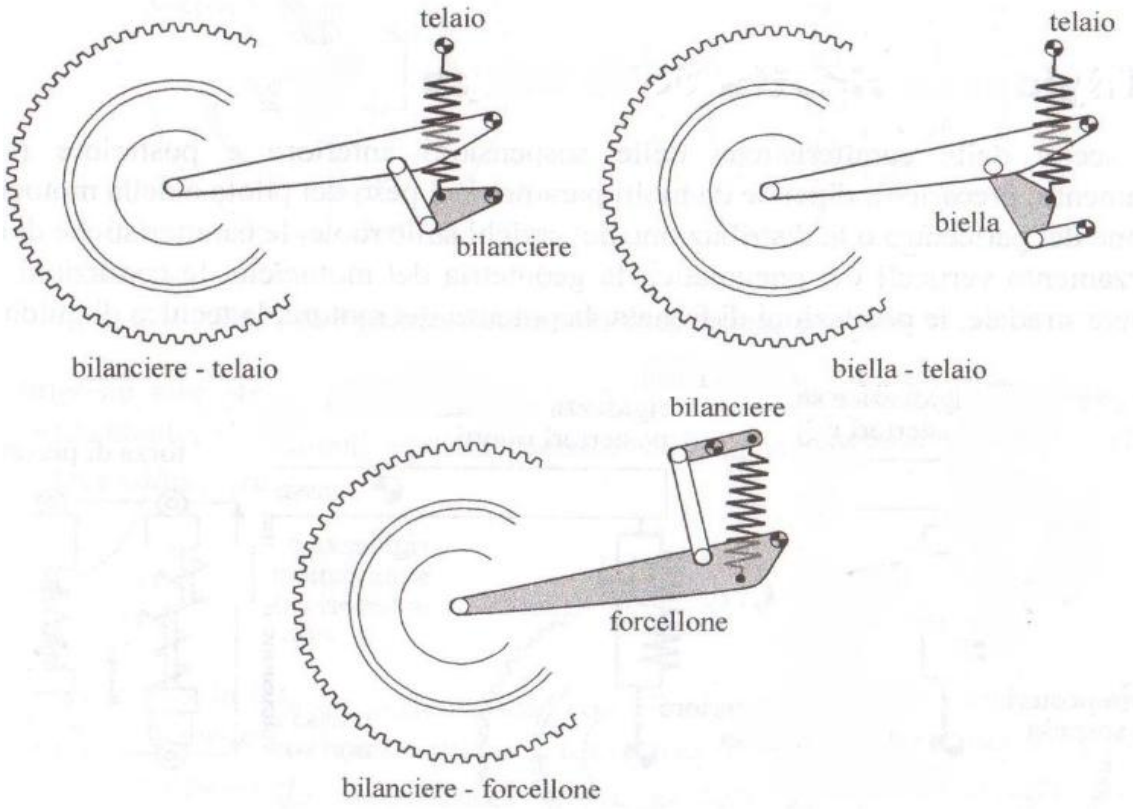


Figura 4: principali schemi di sospensioni con meccanismi a quadrilatero

Questi schemi consentono inoltre ottenere elevate escursioni della ruota e ridotte masse non sospese, ma si registrano anche elevate forze reattive scambiate tra i vari membri. La presenza del quadrilatero sta alla base anche delle sospensioni utilizzate per la trasmissione ad albero con giunti cardanici (in sostituzione della tradizionale trasmissione a catena). In tal caso la ruota è collegata alla biella del quadrilatero e il suo centro di rotazione è dato dall'intersezione dei due bilanceri (e quindi dipende dagli angoli di inclinazione degli stessi).

Un ulteriore schema sospensivo è costituito dal sistema ad esalatero realizzato da Morbidelli: dal punto di visto teorico esso consente di ottenere delle curve di rigidezza molto particolari ed articolate, ma tale vantaggio non è giustificato dalla notevole complessità geometrica costruttiva.

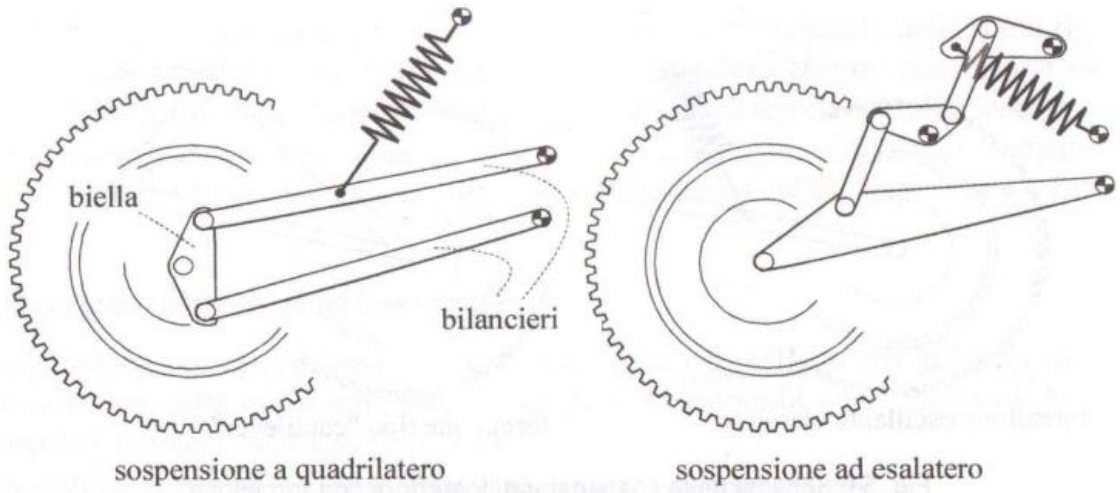


Figura 5: schema sospensivo a quadrilatero per trasmissione ad albero con giunti cardanici (sx) e sistema ad esalatero Morbidelli (dx)

1.5. RIGIDEZZA RIDOTTA DELLA SOSPENSIONE

Al fine di confrontare le rigidezze delle diverse tipologie di sospensioni sopraelencate risulta utile ricorrere ad una metodologia particolare: il calcolo della rigidezza ridotta. Procedendo in tal senso è possibile realizzare un modello semplificato del sistema

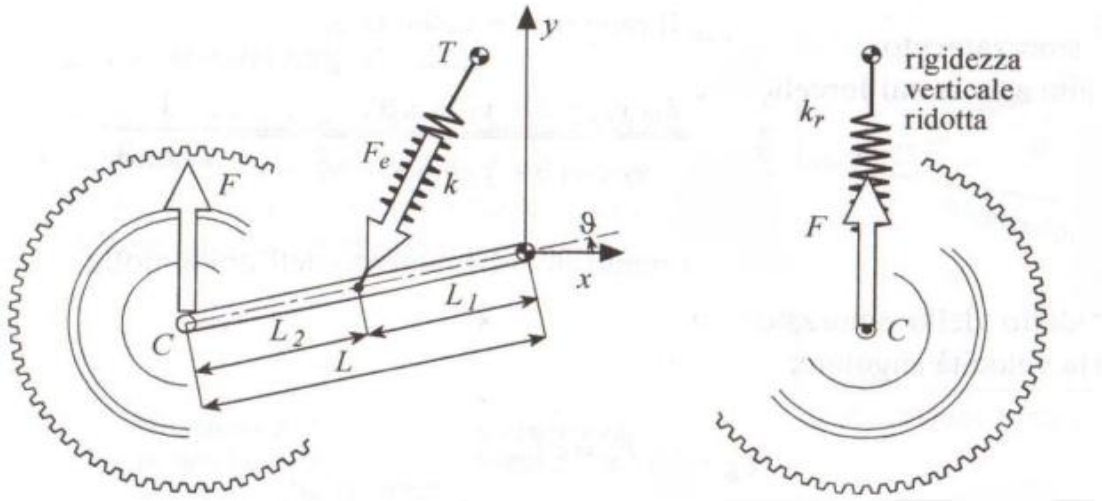


Figura 6: rigidezza ridotta della sospensione posteriore

sospensivo, a partire dalla molla reale di rigidezza k (*spring rate*).

Il modello scelto per il presente lavoro si basa sulla riduzione della sospensione ad una molla verticale di rigidezza k_r (*wheel rate*) collegata direttamente al perno ruota. Da ciò si deduce che lo *spring rate* corrisponde alla forza necessaria per ottenere una compressione unitaria dell'ammortizzatore, mentre il *wheel rate* è legata alla forza necessaria per sollevare verticalmente la ruota della stessa quantità [2]. Il legame tra questi due spostamenti, che dipende dalla tipologia di meccanismo sospensivo, viene espresso dal rapporto di velocità τ_{m,y_C}. Dal punto di vista matematico è possibile esprimere la rigidezza ridotta a partire dalla forza elastica F_e della molla:

$$F_e = k\,(L_m - L_{m_0}) \tag{1}$$

in cui L_{m_0} è la lunghezza iniziale mentre L_m la lunghezza della molla deformata. A partire dalla definizione del rapporto tra la velocità di deformazione della molla e la velocità verticale della ruota

$$\tau_{m,y_C} = \frac{\dot{L_m}}{\dot{y_C}} \tag{2}$$

si può ricavare la forza elastica ridotta F:

$$F = F_e\,\tau_{m,y_C} \tag{3}$$

Derivando quest'ultima espressione rispetto allo spostamento verticale della ruota si ottiene la rigidezza ridotta

$$k_r = \frac{\partial F}{\partial y_C} = k\,{\tau_{m,y_C}}^2 + k(L_m - L_{m_0})\frac{\partial \tau_{m,y_C}}{\partial y_C} \tag{4}$$

Dal punto di vista strettamente numerico si evince che il 2° addendo è meno importante del primo (assume un valore inferiore al 10% rispetto al primo termine), perciò in prima approssimazione si può trascurare e procedere al troncamento.

Per quanto concerne le sospensioni a quadrilatero il rapporto di velocità τ_{m,y_C} varia tra 0.25 e 0.5: ciò implica che la rigidezza reale del gruppo molla-ammortizzatore deve essere 4 volte più grande rispetto alla rigidezza ridotta del relativo schema.

1.6. CURVA DELLA RIGIDEZZA RIDOTTA

Per valutare il comportamento della sospensione risulta utile fare riferimento alla curva di rigidezza: si procede quindi a graficare, in funzione dell'escursione verticale della ruota y_C, la forza elastica e la rigidezza ridotta.

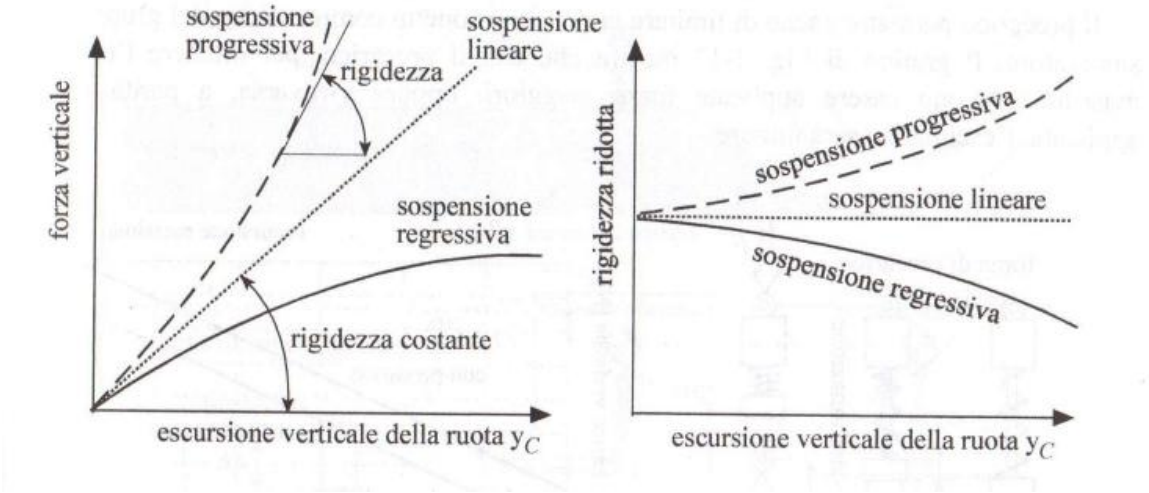


Figura 7: forza elastica e rigidezza ridotta della sospensione in funzione dell'escursione verticale dello pneumatico posteriore

In tal modo si può evidenziare l'andamento lineare, progressivo o regressivo della sospensione: è bene ricordare che nella figura nella destra tali termini si riferiscono alla rigidezza ridotta del sistema, la quale può essere crescente/costante/decrescente mentre la forza elastica risulta essere sempre crescente ma con pendenza dipendente da k_r. Storicamente il comportamento lineare della sospensione è stato il primo introdotto, sia per la semplicità della sua gestione sia per la particolare disposizione del gruppo molla-ammortizzatore nel sistema a forcellone oscillante. Col progredire dell'evoluzione tuttavia

si è passati allo studio di sospensioni progressive, le quali offrono dei particolari vantaggi di primaria importanza: come già affermato in precedenza, la motocicletta deve assolvere alle funzioni di aderenza, comfort e assetto. La scelta di molle rigide permette di avere un assetto costante al variare delle condizioni di utilizzo, ma pregiudica notevolmente gli altri due aspetti: le vibrazioni dovute alle asperità stradali vengono infatti interamente trasmesse al veicolo e quindi al pilota, compromettendone il comfort di guida. Oltre a questo aspetto, ne risulta penalizzata anche la tenuta di strada, poiché le sospensioni non permettono agli pneumatici di seguire gli avvallamenti del manto. Viceversa, l'utilizzo di molle molto morbide migliora notevolmente il comfort e l'aderenza ma genera notevoli escursioni della sospensione, con conseguente variazione dell'assetto specialmente nelle fasi di sterzata, accelerazione e frenata (quando cioè si evidenzia il trasferimento di carico). Queste considerazioni portano alla conseguente necessità di adottare uno schema sospensivo a comportamento progressivo: in tal modo i piccoli disturbi della carreggiata vengono facilmente assorbiti ma si evitano escursioni elevate della molla (aumentando appunto progressivamente la rigidezza delle sospensioni), a beneficio del comfort e della tenuta. Dal punto di vista delle vibrazioni questa soluzione permette inoltre di mantenere costanti le frequenze dei modi di vibrare del veicolo, al variare della massa sul veicolo (peso del pilota, presenza di un passeggero o di bagagli, etc…).

L'utilizzo di sospensioni di tipo regressivo invece ricopre una ristretta nicchia di mercato, per lo più legata a veicoli progettati per il fuoristrada: esse infatti permettono al pilota di ottimizzare il controllo su fondi stradali caratterizzati da una bassa aderenza.

2. METODO DELLE COORDINATE NATURALI

2.1. INTRODUZIONE

Lo sviluppo delle tecniche multibody e di sofisticati algoritmi di calcolo ha permesso, in tempi relativamente recenti, lo studio di meccanismi e sistemi sempre più complessi; i primi metodi utilizzati per analisi e sintesi si basano sulle coordinate angolari per descrivere posizione e moto dei corpi, al fine di ridurre al minimo il numero di equazioni [3]. Dal punto di vista numerico tuttavia questo rappresenta un ostacolo importante, in quanto le grandezze trigonometriche richiedono elevati tempi di calcolo. Si perviene quindi, nei primi anni ’60, allo sviluppo delle coordinate naturali: questo approccio permette di determinare la posizione di un meccanismo facendo ricorso esclusivamente alle coordinate cartesiane, i *punti base,* ed eventuali coseni direttori di versori solidali al corpo. Si osserva sin da subito che il numero di variabili richieste per lo studio del sistema non corrisponde al minimo assoluto: un corpo infatti necessita di 4 coordinate per essere descritto nel piano (invece di 3) e di 12 nello spazio (anziché 6). Le coordinate naturali sono quindi ridondanti e l’utilizzo di equazioni di congruenza risulta necessario per preservare la rigidità del corpo [4]. Questo svantaggio viene tuttavia ampiamente bilanciato dal fatto che le equazioni in questione sono lineari o quadratiche, consentendo

una notevole riduzione in termini di tempi di calcolo per analisi e sintesi. Quando inoltre si studiano sistemi di corpi, il numero di coordinate naturali viene notevolmente ridotto tramite i punti condivisi in prossimità delle coppie cinematiche [5]. Una coppia rotoidale necessita infatti di un singolo punto base per essere descritta, appartenente ad entrambi i corpi, e per soddisfare implicitamente le condizioni di vincolo; una coppia prismatica invece fa riferimento a due punti base,uno per ciascuno dei corpi vincolati, ed un versore, il quale definisce la retta d'azione della coppia stessa. Le relazioni che invece descrivono la congruenza nel piano possono essere di tre tipi:

- La distanza tra due punti solidali ad un corpo deve mantenersi costante, descritta dalla nota formula della distanza;
- L'angolo tra vettori solidali ad un corpo deve mantenersi costante, espresso dal prodotto scalare/vettoriale dei vettori stessi;
- La proiezione di un vettore in una direzione assegnata deve mantenersi costante, delineata matematicamente come nel caso precedente (condizione utilizzata per esempio nelle coppie prismatiche).

2.2. MODELLIZZAZIONE GENERALE DEI MECCANISMI

Attraverso alcune considerazioni cinematiche è possibile definire un metodo generale per la descrizione matematica di meccanismi, al fine di studiarne particolari caratteristiche o applicazioni [6]. Nel caso presente si faccia riferimento ad un generico meccanismo con un grado di libertà, m coordinate naturali $\alpha_1, \ldots, \alpha_m$ e n dimensioni geometriche essenziali $l_1, \ldots, l_n$ dei corpi. Dalla teoria della meccanica applicata si ricava che devono esistere $m-1$ equazioni di congruenza derivanti dalle condizioni di rigidità del sistema:

$$\begin{cases} \varnothing_1(\alpha_1, \ldots, \alpha_m;\ l_1, \ldots, l_n) = 0 \\ \quad\quad . \\ \quad\quad . \\ \quad\quad . \\ \varnothing_{m-1}(\alpha_1, \ldots, \alpha_m;\ l_1, \ldots, l_n) = 0 \end{cases} \tag{5}$$

Nelle fasi relative all'ottimizzazione del meccanismo, l'algoritmo impostato cercherà il valore ottimale delle lunghezze $l_1, \dots, l_n$ che permettano al meccanismo di seguire una particolare funzione desiderata. Tipicamente vi sono tre diverse tipologie di sintesi di meccanismi: generazione di funzione, generazione di traiettorie e guida di corpo rigido, con o senza correlazione. La tecnica di ottimizzazione in esame permette di trattarle con lo stesso metodo, imponendo le opportune equazioni e minimizzando l'errore strutturale.
A titolo di esempio, nel caso della sintesi con generazione di traiettorie senza correlazione si considera un particolare punto (x, y) appartenente al corpo e le relative posizioni desiderate (x_d, y_d) parametrizzate nella coordinata continua s. È quindi possibile definire il sistema di funzioni obiettivo associato al meccanismo:

$$\begin{cases} x = x_d(s) \\ y = y_d(s) \end{cases} \tag{6}$$

Nel caso invece si intenda effettuare una sintesi di traiettoria con correlazione è sufficiente inserire le equazioni relative all'angolo desiderato $\theta_d(s)$ e ad un'eventuale parametro addizionale θ_0:

$$\begin{cases} x = x_d(s) \\ y = y_d(s) \\ \frac{x_1 - x_0}{l_2} = \cos\left(\theta_d(s) + \theta_0\right) \\ \frac{y_1 - y_0}{l_2} = \operatorname{sen}\left(\theta_d(s) + \theta_0\right) \end{cases} \tag{7}$$

dove (x_0, y_0), (x_1, y_1) e l_2 sono grandezze relative ad al corpo in esame.
Lo studio della sintesi per guida di corpo rigido e generazione di funzione si articola con la stessa struttura matematica: procedendo in questo modo è possibile inserire tutte le condizioni k desiderate ed ottenere il sistema di funzioni obiettivo:

$$\begin{cases} f_1(\alpha_1, \dots, \alpha_m, s) = 0 \\ \quad . \\ \quad . \\ \quad . \\ f_k(\alpha_1, \dots, \alpha_m, s) = 0 \end{cases} \tag{8}$$

Oltre alle funzioni obiettivo relative alla movimentazione desiderata del meccanismo, vi sono altre condizioni di natura geometrica (lunghezza massima di elementi, ostacoli, …), operativa (angoli di trasmissione, …) e dinamica (forze, deformazioni, …) che possono essere richieste al sistema: molte di esse possono essere descritte da diseguaglianze e, sebbene esuli dagli intenti del presente lavoro, è comunque possibile raggiungere tali obiettivi con specifiche funzioni penalità.

2.3. ANALISI CINEMATICA

Lo studio dell'analisi cinematica del meccanismo in esame svolge un ruolo fondamentale nel processo di ottimizzazione, in quanto permette di valutarne la prestazione. Nello sviluppo dell'algoritmo infatti le dimensioni $l_1, \dots, l_n$ sono temporaneamente fissate e si valuta la performance facendo riferimento al sistema di funzioni obiettivo (8). Risulta evidente che in questa fase il numero di equazioni $(m + k - 1)$ è maggiore del numero di variabili indipendenti $\alpha_1, \dots, \alpha_m$, quindi il problema è sovravincolato: ciò implica che qualunque combinazione delle coordinate indipendenti non sarà in grado di soddisfare esattamente tutte le funzioni obiettivo.

Si procede pertanto alla configurazione $\alpha_1, \dots, \alpha_m$ che soddisfa al meglio le funzioni obiettivo (8) e che soddisfa esattamente solo le equazioni di congruenza (5) (la mancata soddisfazione di queste ultime porterebbe infatti alla non assemblabilità del meccanismo); in tal modo si "muove" il meccanismo il più vicino possibile alle posizioni richieste in input.

Per valutare il grado di ottimizzazione raggiunto è conveniente ricorrere ad un criterio di performance e quindi si calcola la "distanza" tra le configurazioni attuale e obiettivo introducendo la norma euclidea, basata sui residui delle eq. (8):

$$p(\alpha_1, \dots, \alpha_m, s) = f_1^2 + \ \dots + f_k^2 \qquad (9)$$

Tale parametro è indicativo della prestazione puntuale del meccanismo, pertanto si procede all'integrazione numerica in tutto l'intervallo $[\mathrm{s}_1\ \mathrm{s}_2]$ all'interno del quale si devono soddisfare le funzioni obiettivo:

$$I = \int_{s_1}^{s_2} p(\alpha_1, \dots, \alpha_m, s)ds \tag{10}$$

Questo indice di performance viene calcolato parallelamente all'esecuzione dell'analisi cinematica e permette quindi di valutarne il livello di ottimizzazione. Il sistema in esame quindi rientra nella categoria dei problemi ad ottimizzazione vincolata; è pertanto conveniente ricorrere ai moltiplicatori di Lagrange per ottenere un problema ad ottimizzazione non vincolata.

La funzione integranda di (10) viene quindi sostituita dal lagrangiano:

$$L(\alpha_1, \dots, \alpha_m, \lambda_1, \dots, \lambda_{m-1}, s) = p(\alpha_1, \dots, \alpha_m, s) + \sum_{i=1}^{m-1} \lambda_i \emptyset_i(\alpha_1, \dots, \alpha_m) \tag{11}$$

in cui i valori $\lambda_1, \dots, \lambda_{\mathrm{m}-1}$ sono i moltiplicatori di Lagrange.

Applicando la teoria del calcolo delle variazioni è possibile derivare un set di $2\mathrm{m} - 1$ equazioni per le $2\mathrm{m} - 1$ funzioni incognite $\alpha_1, \dots, \alpha_{\mathrm{m}}, \lambda_1, \dots, \lambda_{\mathrm{m}-1}$:

$$\begin{cases} \frac{\partial L}{\partial \alpha_i} = \frac{\partial p}{\partial \alpha_i} + \sum_{k=1}^{m-1} \lambda_i \frac{\partial}{\partial \alpha_i} \emptyset_k = 0 & i = 1, m \\ \frac{\partial L}{\partial \lambda_j} = \emptyset_j = 0 & j = 1, m-1 \end{cases} \tag{12}$$

Si osserva che le ultime $\mathrm{m} - 1$ equazioni sono relative alla condizione di congruenza (5) mentre le prime m richiedono il gradiente di p e lo Jacobiano delle funzioni $\emptyset_{\mathrm{k}}$. Questo sistema rappresenta per intero l'analisi cinematica del meccanismo, tuttavia è composto da equazioni algebriche non lineari che richiedono un costo computazionale elevato per la ricerca della loro soluzione.

Per ridurre i tempi di calcolo e facilitare il raggiungimento delle soluzioni è opportuno trasformare le (12) in un sistema ordinario del primo ordine di equazioni differenziali nella variabile s: tale metodo viene utilizzato di frequente per le equazioni algebriche nei

sistemi dinamici multi-body [7]. Una generica funzione algebrica $f(q_i, s)$ può essere sostituita dall'equazione differenziale

$$\frac{d}{ds} f(q_i, s) + \mu f(q_i, s) = 0 \tag{13}$$

in cui il parametro μ è una costante di tempo.

Applicando questa trasformazione al suddetto sistema, sostituendo q_i con $\alpha_i(s)$ e $\lambda_i(s)$, si ottiene:

$$\begin{cases} \frac{d}{ds}\frac{\partial L}{\partial \alpha_i} + \mu \frac{\partial L}{\partial \alpha_i} = 0 & i = 1, m \\ \frac{d}{ds}\emptyset_j + \mu \emptyset_j = 0 & j = 1, m-1 \end{cases} \tag{14}$$

Attraverso tali sostituzioni le equazioni sviluppano un sistema di ordinario del primo ordine:

$$\begin{bmatrix} \frac{\partial^2 L}{\partial \alpha_i \partial \alpha_k} & \frac{\partial^2 L}{\partial \alpha_i \partial \lambda_h} \\ \frac{\partial \emptyset_j}{\partial \alpha_k} & \frac{\partial \emptyset_j}{\partial \lambda_h} \end{bmatrix} \begin{Bmatrix} \frac{d}{ds}\alpha_k \\ \frac{d}{ds}\lambda_h \end{Bmatrix} + \frac{\partial}{\partial s}\begin{Bmatrix} \frac{\partial L}{\partial \alpha_i} \\ \emptyset_j \end{Bmatrix} + \mu \begin{Bmatrix} \frac{d}{ds}\alpha_k \\ \emptyset_j \end{Bmatrix} = 0 \quad \begin{matrix} i = 1, m & k = 1, m \\ j = 1, m-1 & h = 1, m-1 \end{matrix} \tag{15}$$

Si osserva che l'analisi cinematica è stata ricondotta ad un problema ai valori iniziali; nei capitoli successivi verrà evidenziato il metodo per la scelta dei valori iniziali di $\alpha_i(s)$ e $\lambda_i(s)$. Analizzando la forma del lagrangiano L (11) è possibile semplificare ulteriormente il sistema (15) per ottenere una formulazione più vantaggiosa dal punto di vista computazionale:

$$\begin{bmatrix} \frac{\partial^2 L}{\partial \alpha_i \partial \alpha_k} & \frac{\partial \emptyset_h}{\partial \alpha_i} \\ \frac{\partial \emptyset_j}{\partial \alpha_k} & 0 \end{bmatrix} \begin{Bmatrix} \frac{d}{ds}\alpha_k \\ \frac{d}{ds}\lambda_h \end{Bmatrix} + \frac{\partial}{\partial s}\begin{Bmatrix} \frac{\partial L}{\partial \alpha_i} \\ 0 \end{Bmatrix} + \mu \begin{Bmatrix} \frac{\partial L}{\partial \alpha_i} \\ \emptyset_j \end{Bmatrix} = 0 \quad \begin{matrix} i = 1, m & k = 1, m \\ j = 1, m-1 & h = 1, m-1 \end{matrix} \tag{16}$$

I risultati ottenuti si possono presentare in forma ulteriormente più compatta nella seguente forma:

$$[\boldsymbol{A}]\frac{d}{ds}\begin{Bmatrix}\alpha_k\\ \lambda_h\end{Bmatrix}+\{\boldsymbol{B}\}+\mu\{\boldsymbol{C}\}=0 \qquad (17)$$

A seconda del meccanismo in esame i componenti A, B e C avranno una particolare descrizione matematica: nel seguente capitolo verrà riportato l'esempio dello studio dell'analisi cinematica del quadrilatero articolato.

3. ANALISI DEL QUADRILATERO ARTICOLATO

3.1. EQUAZIONI DEL MECCANISMO

Per conseguire lo studio delle sospensioni composte da meccanismi è utile, in prima analisi, procedere allo studio del quadrilatero articolato, per poi implementare la sospensione completa. In primo luogo è necessario ricavare le equazioni che descrivono il moto del meccanismo, facendo riferimento a due diverse formule: la distanza tra due punti viene utilizzata per esprimere la lunghezza dei membri L1, L2 e L3 mentre le grandezza a e b sono valutate in termini di proiezioni di vettori nelle due direzioni cartesiane.

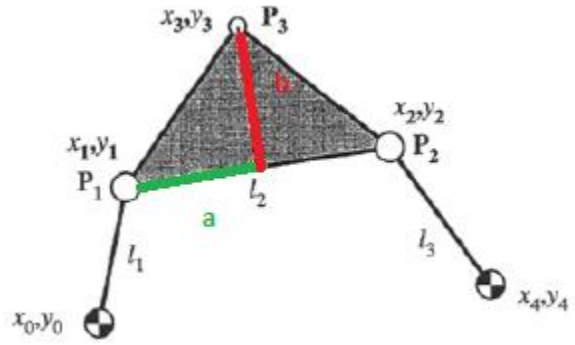


Figura 8: schematizzazione base del quadrilatero articolato

Si osserva facilmente dalla figura che i punti 0, 1, 2 e 3 sono posti in corrispondenza delle coppie rotoidali del sistema, mentre il punto 3 è quello che genera la traiettoria che si intende controllare ed è solidale alla biella [6].

Considerando fisse le coordinate $(\mathrm{x}_0, \mathrm{y}_0, \mathrm{x}_4, \mathrm{y}_4)$ le uniche variabili indipendenti sono $(\mathrm{x}_1, \mathrm{y}_1, \mathrm{x}_2, \mathrm{y}_2, \mathrm{x}_3, \mathrm{y}_3)$: il sistema che quindi descrive le equazioni di vincolo per il meccanismo è il seguente:

$$\begin{cases} (x_1 - x_0)^2 + (y_1 - y_0)^2 = l_1^2 \\ (x_1 - x_2)^2 + (y_1 - y_2)^2 = l_2^2 \\ (x_2 - x_4)^2 + (y_2 - y_4)^2 = l_3^2 \\ a = l_2 \frac{(x_1 - x_2)(x_3 - x_1) - (y_1 - y_3)(y_2 - y_1)}{(x_1 - x_2)^2 + (y_1 - y_2)^2} \\ b = l_2 \frac{(x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1)}{(x_1 - x_2)^2 + (y_1 - y_2)^2} \end{cases} \tag{18}$$

Tale sistema corrisponde alle $\mathrm{m} - 1$ equazioni (5) introdotte in precedenza.

La valutazione della prestazione del meccanismo viene effettuata prendendo a riferimento il punto $(\mathrm{x}_3, \mathrm{y}_3)$ rispetto all'andamento imposto $(\mathrm{x}_\mathrm{d}, \mathrm{y}_\mathrm{d})$, parametrizzato dalla coordinata curvilinea s che viene fatta variare da 0 a 1. Si calcola quindi il criterio di performance

$$I = \int_0^1 \{ [\, x_3 - x_d(s))^2 + [\, y_3 - y_d(s)\,]^2 \, \} ds \tag{19}$$

Per ricavare le altre m equazioni si introducono i moltiplicatori di Lagrange ($\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5$), si calcola la funzione Lagrangiana e le corrispondenti 6 derivate spaziali. A questo punto è possibile operare in maniera analoga al capitolo precedente e ricavare le matrici A, B e C del sistema ordinario (17) che vengono riportate in Appendice A. E' quindi possibile realizzare l'analisi cinematica completa dei quadrilateri articolati.

Gli studi dei vari meccanismi sono stati eseguiti con il software Matlab e, per realizzare l'analisi cinematica, si è fatto riferimento al risolutore ode45 il quale offre un buon compromesso tra affidabilità e velocità di esecuzione.

3.2. VALORI INIZIALI

Ricordando che l'obiettivo finale è l'esecuzione di un'ottimizzazione in grado di variare le lunghezze dei membri $l_1, \dots, l_n$, è opportuno eseguire alcune valutazioni sui dati di input da inserire. Una prima opzione riguarda l'inserimento dei valori iniziali delle grandezze geometriche in esame e le coordinate cartesiane dei punti fissi: si tratterebbe quindi di una base formata dai nove parametri $(x_0, y_0, x_4, y_4, l_1, l_2, l_3, a, b)$, ovvero lo *spazio delle lunghezze iniziali*. La problematica principale di questa base è legata al fatto che la posizione iniziale del meccanismo non è nota e ciò richiederebbe la risoluzione di un sistema non lineare; oltre all'aumento del carico computazionale sono da tenere in considerazione rischi circa la non assemblabilità del sistema e la possibile esistenza di più configurazioni per delle date lunghezze. In ultima analisi è da tenere il considerazione la possibilità che una piccola variazione sul valore iniziale di una lunghezza possa portare ad una soluzione molto diversa rispetto al caso precedente. Per tutte queste motivazioni è utile ricorrere allo spazio dei valori iniziali delle coordinate naturali: i parametri in input sono quindi i dieci valori $(x_0, y_0, x_4, y_4, x_1(0), y_1(0), x_2(0), y_2(0), x_3(0), y_3(0))$. E' necessario pertanto accettare un aumento nelle dimensioni del problema per mitigare gli svantaggi sopraelencati: la posizione iniziale del meccanismo è infatti nota (eliminando i problemi di eventuale non assemblabilità), le lunghezze iniziali dei membri possono essere facilmente calcolate ed una piccola variazione dei valori iniziali porta circa alla stessa soluzione.

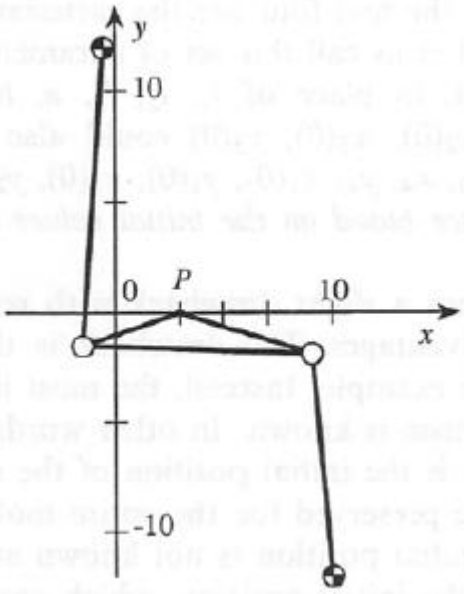
Per la risoluzione analitica del sistema ordinario (17) è necessario introdurre anche i valori iniziali dei moltiplicatori di Lagrange; poiché le equazioni di congruenza devono essere soddisfatte in maniera esatta, le restanti in generale non lo sono (poiché, come descritto in precedenza, il sistema è sovravincolato). Pertanto, per soddisfare al meglio le prime m equazioni del sistema si ricorre al metodo dei minimi quadrati; tale tecnica non è particolarmente onerosa in quanto le equazioni sono lineari rispetto ai moltiplicatori λ_i. Il sistema di ottimizzazione in seguito provvederà a ridurre quanto possibile il residuo di tali equazioni minimizzando quindi l'indice di performance.

3.3. ESEMPI

Per valutare il funzionamento dell'analisi cinematica in forma differenziale sono stati riprodotti tre esempi di quadrilatero articolato presenti in letteratura; per un'ulteriore conferma si è deciso di eseguire anche un'analisi cinematica algebrica, imponendo la variazione della coordinata y_3 e per poter quindi confrontare la risposta dei meccanismi.

- *Esempio 1*

Con riferimento al lavoro svolto in [6] si riporta lo studio di un quadrilatero generatore di una traiettoria rettilinea senza correlazione dal punto (3,0) al punto (7,0).



coordinate iniziali		geometria	
x0	-0,3824	l1	13,424
y0	11,856	l2	9,980
x1	-1,298	l3	10,130
y1	-1,537	a	4,237
x2	8,675	b	1,697
y2	-1,911		
x3	3		
y3	0		
x4	9,591		
y4	-12		

Figura 9: Esempio 1 di studio del quadrilatero articolato e parametri iniziali

La traiettoria è descritta dalle equazioni

$$\begin{cases} x_d(s) = 3 + 4s \\ y_d(s) = 0 \end{cases} \qquad s = [0, 1] \qquad (20)$$

Di seguito sono riportati due grafici relativi al movimento del meccanismo con i due diversi sistema di analisi, i quali permettono di verificare che effettivamente la traiettoria eseguita sia la stessa; più in basso invece si evidenzia l'errore di traiettoria, il quale assume un valore massimo di ben quattro ordini di grandezza inferiore al millimetro.

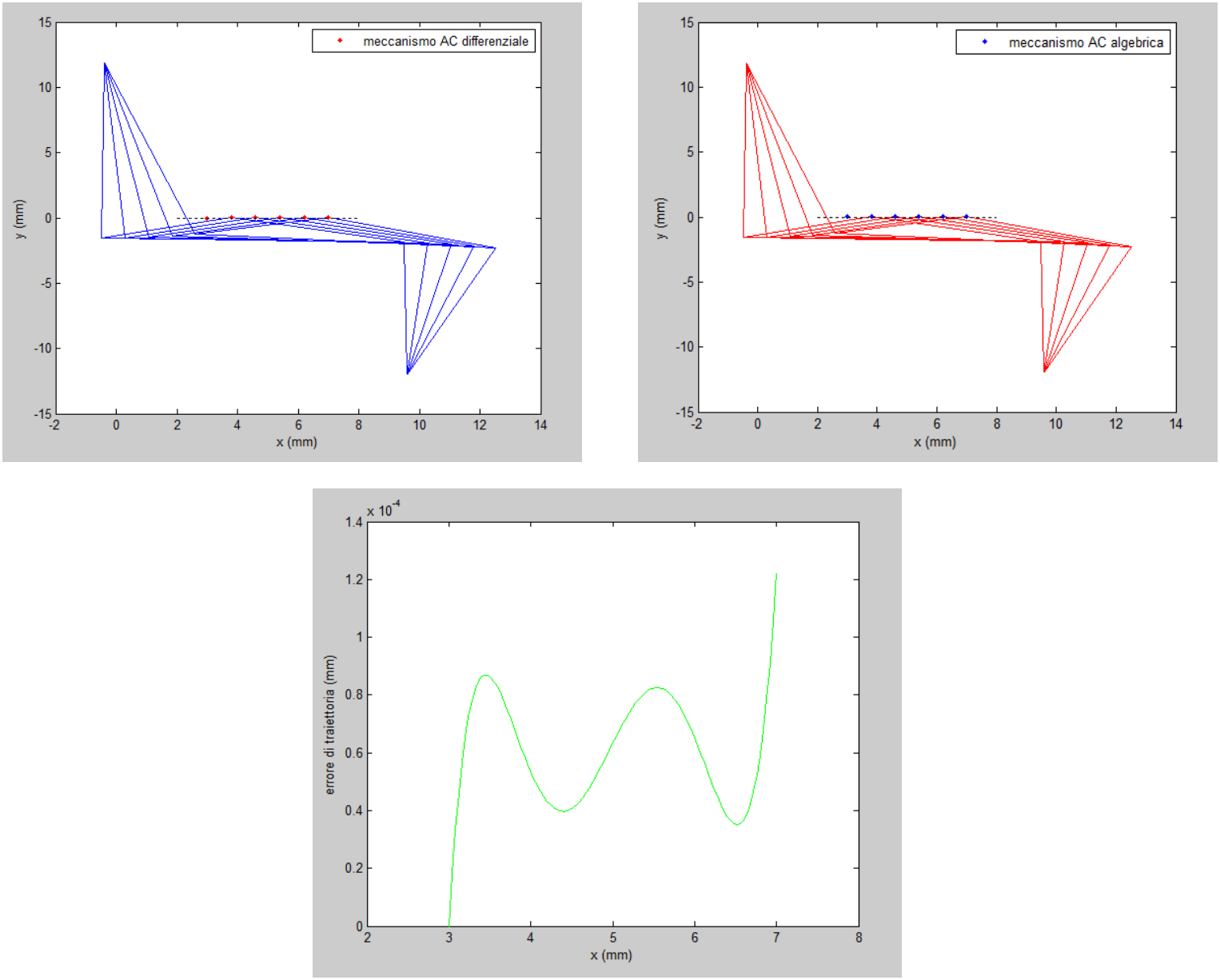
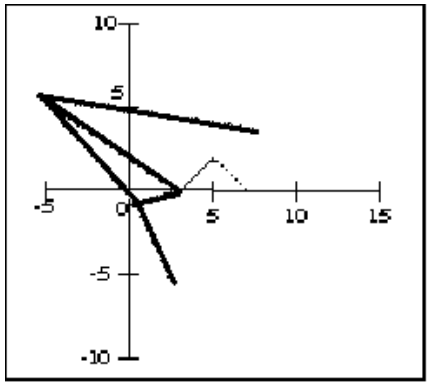


Figura 10: Analisi cinematica differenziale e algebrica del quadrilatero 1 ed errore di traiettoria

- *Esempio 2*

Il secondo caso di studio si ricava da [4] ed affronta una problematica più complessa: il punto (x_3, y_3) deve infatti eseguire una traiettoria formata da due segmenti di linea retta.



coordinate iniziali		geometria	
x0	2,654	l1	5,081
y0	-5,371	l2	8,738
x1	0,502	l3	13,000
y1	-0,768	a	1,128
x2	-5,423	b	2,358
y2	5,655		
x3	3		
y3	0,001		
x4	7,423		
y4	3,656		

Figura 11: Esempio 2 di studio del quadrilatero articolato e parametri iniziali

Le equazioni che descrivono il target da seguire sono:

$$\begin{cases} x_d(s) = 3 + 4s \\ y_d(s) = |4s - 2| - 2 \end{cases} \qquad s = [0, 1] \qquad (21)$$

Analogamente al caso precedente si può osservare nella pagina a fianco che il sistema differenziale e quello algebrico offrono le medesime soluzioni; valutando i due grafici relativi alla traiettoria è possibile notare che, come prevedibile, il punto più penalizzato della traiettoria è il valore mediano (5,2) cui è assegnato un errore di 5 centesimi di millimetro.

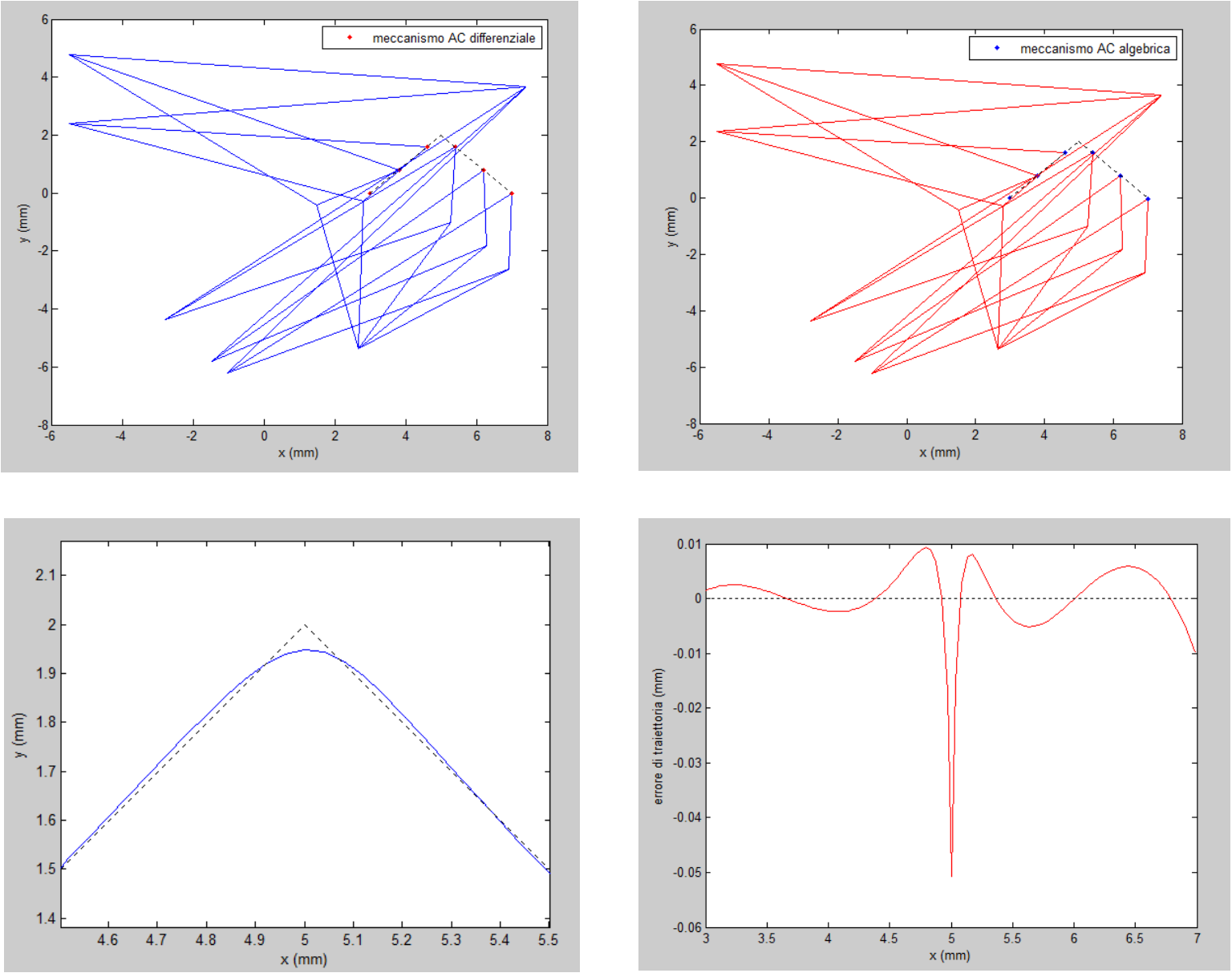
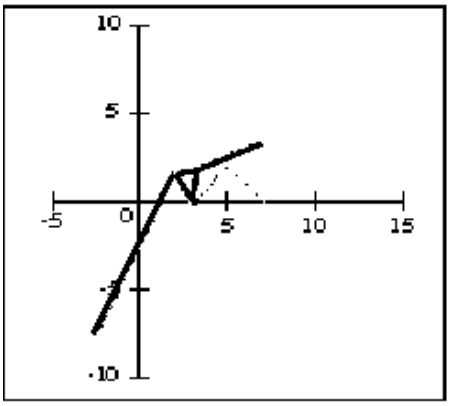


Figura 12: Analisi cinematica differenziale e algebrica del quadrilatero 2, traiettoria del *tracer point* ed errore di traiettoria

- *Esempio 3*

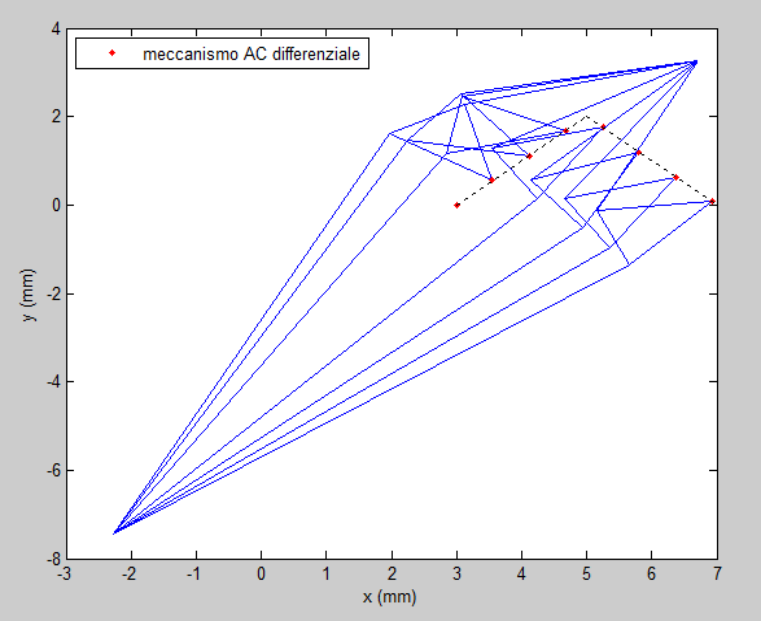
Il terzo caso è presente in [4] e richiede al sistema di eseguire la medesima traiettoria del caso precedente, utilizzando comunque un meccanismo meno ingombrante in termini di spazio di manovra occupato.

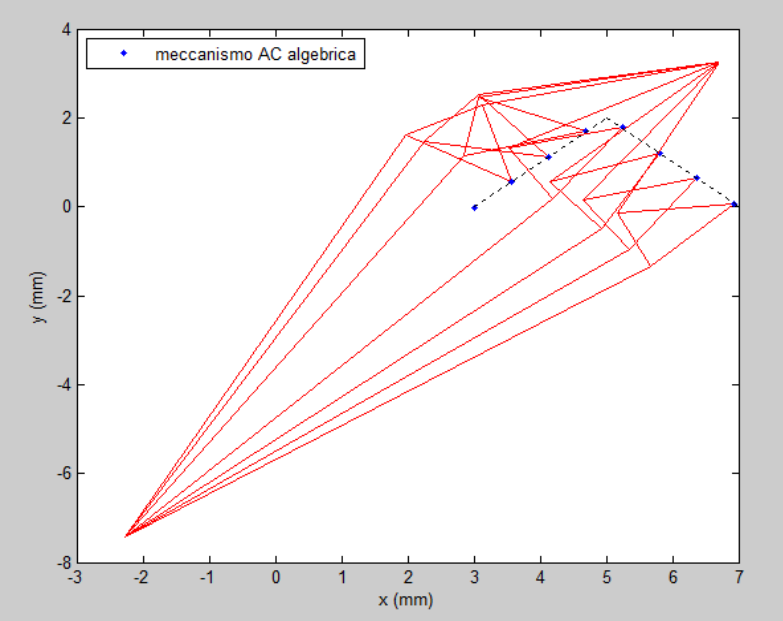


coordinate iniziali		geometria	
x0	-2,272	l1	10,000
y0	-7,437	l2	1,325
x1	2,001	l3	3,725
y1	1,603	a	0,846
x2	3,321	b	1,704
y2	1,724		
x3	3		
y3	-0,016		
x4	6,716		
y4	3,257		

Figura 13: Esempio 3 di studio del quadrilatero articolato e parametri iniziali

Con riferimento alle equazioni target (21) si esegue lo studio del quadrilatero; dai grafici seguenti si può osservare che la compattezza del sistema si paga in termini di errore maggiore nel punto più critico (6 centesimi di millimetro).





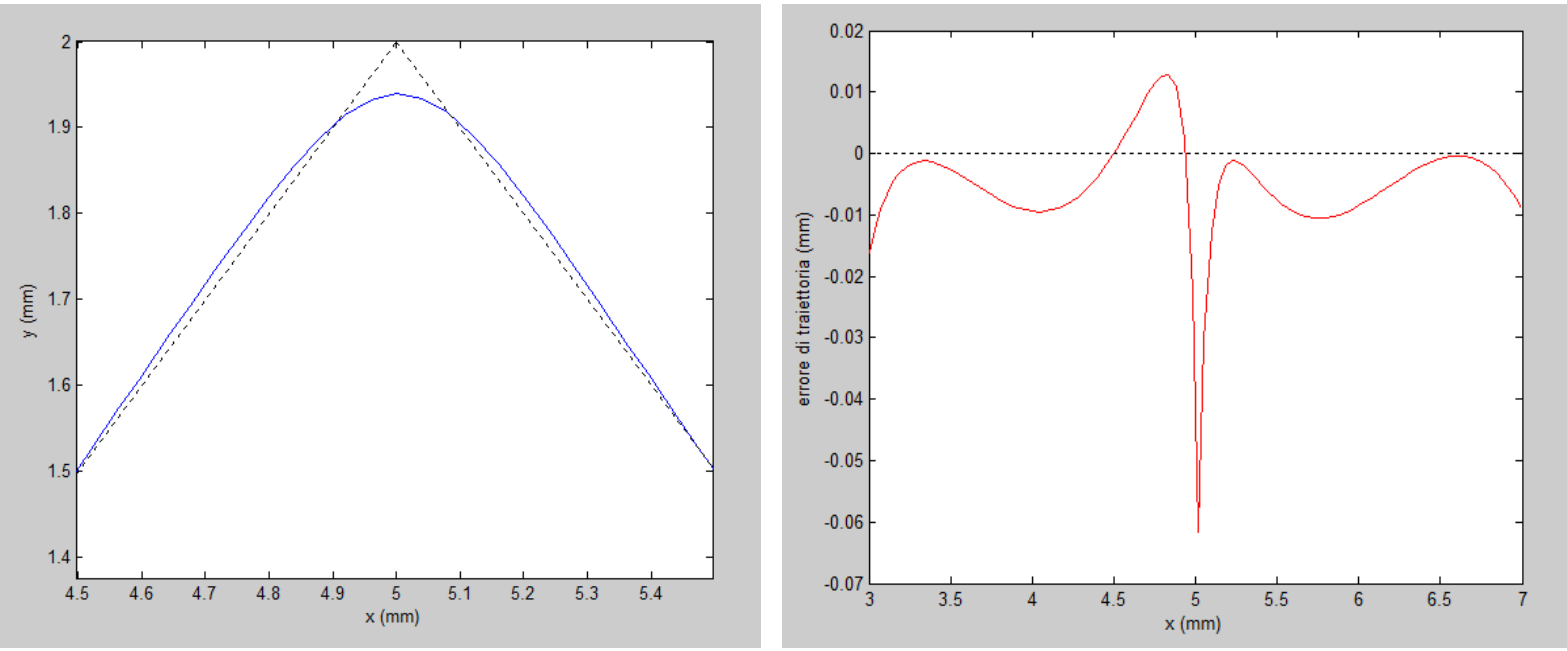


Figura 14: Analisi cinematica differenziale e algebrica del quadrilatero 3, traiettoria del *tracer point* ed errore di traiettoria.

4. IMPLEMENTAZIONE DEL MODELLO SOSPENSIONE

4.1. DESCRIZIONE DEL SISTEMA

Nei capitoli precedenti è stato studiato il comportamento del modello quadrilatero articolato e se ne sono eseguite diverse prove di analisi cinematica. Nel presente capitolo invece si intende estendere lo studio del quadrilatero all'intera sospensione per poi integrare diversi algoritmi di ottimizzazione sulla base degli obiettivi da raggiungere.

Poiché esistono tre diverse tipologie di sospensioni a quadrilatero (bilancere-forcellone, biella-telaio e bilancere-telaio) si è deciso di restringere il campo ad un singolo caso, ovvero al meccanismo biella-telaio: come già evidenziato nei capitoli precedenti, questa soluzione permette di generare delle curve di rigidezza molto varie e quindi si presta molto al ruolo di "caso generale" di studio. Oltre a ciò, per utilizzare delle dimensioni compatibili con quelle dell'industria motociclistica, si è fatto riferimento allo schema sospensivo della moto in fase di progettazione nel gruppo *Motostudent Unipd*, anch'essa basata sul meccanismo biella-telaio. Nel caso si volesse far vertere lo studio su un tipo di sospensione diversa è comunque sufficiente addentrarsi nel codice di calcolo e modificare le equazioni riportate in Appendice B. Per gentile concessione del gruppo *Motostudent*, nel corso di alcune simulazioni, si sono quindi utilizzati i dati relativi alle posizioni fisse dei

perni; è stato inoltre possibile eseguire un confronto tra la soluzione originale e quelle ottimizzate dal presente algoritmo.

Viene riportata di seguito l'immagine che funge da riferimento per la scrittura delle equazioni che governano il moto della sospensione:

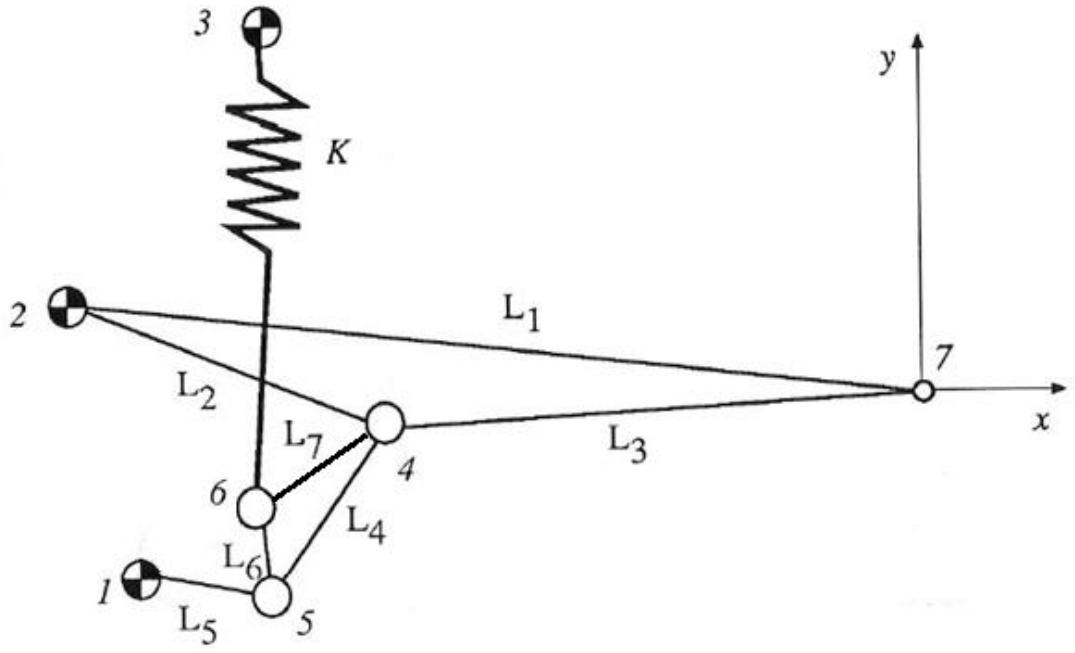


Figura 15: Schema di riferimento per la sospensione posteriore biella - telaio

Il Punto 7 ricopre il ruolo di origine degli assi cartesiani e, a partire da esso, sono descritte tutte le altre coordinate; è possibile riconoscere lo schema del quadrilatero articolato nei membri di lunghezza L2, L4 e L5.

I punti 1, 2 e 3 si mantengono fissi nello spazio sia nel moto che durante il processo di ottimizzazione, mentre gli altri sono liberi di muoversi senza alcun vincolo; la simulazione prevede un sollevamento verticale del perno di attacco della ruota (punto 7) di una quota pari a $100\,mm$. Da ciò se ne ricava che il sistema ha $m = 8$ variabili $(x_4, y_4, x_5, y_5, x_6, y_6, x_7, y_7)$ e le $m - 1$ equazioni che ne descrivono il moto sono:

$$\begin{cases} (x_1 - x_5)^2 + (y_1 - y_5)^2 = l_5^2 \\ (x_2 - x_4)^2 + (y_2 - y_4)^2 = l_2^2 \\ (x_2 - x_7)^2 + (y_2 - y_7)^2 = l_1^2 \\ (x_5 - x_4)^2 + (y_5 - y_4)^2 = l_4^2 \\ (x_4 - x_7)^2 + (y_4 - y_7)^2 = l_3^2 \\ a = l_4 \frac{(x_5 - x_4)(x_6 - x_5) - (y_5 - y_6)(y_4 - y_5)}{(x_5 - x_4)^2 + (y_5 - y_4)^2} \\ b = l_4 \frac{(x_4 - x_5)(y_6 - y_5) - (x_6 - x_5)(y_4 - y_5)}{(x_5 - x_4)^2 + (y_5 - y_4)^2} \end{cases} \quad (22)$$

Analogamente al caso del quadrilatero sono state utilizzate rispettivamente la formula della lunghezza tra due punti e la proiezione di un vettore nelle due direzioni cartesiane.
Al fine di perseguire l'ottimizzazione del meccanismo è necessario introdurre un valore target di una particolare grandezza: nel presente lavoro la scelta è ricaduta sul rapporto di velocità τ tra la velocità di deformazione della molla e la velocità verticale della ruota, ovvero lo stesso parametro calcolato per ricavare la rigidezza ridotta delle sospensioni. Viene quindi calcolata la funzione errore a partire dal confronto tra il valore obiettivo τ_{target} e il valore attuale τ_{optim}:

$$err = \int_0^1 (\tau_{optim} - \tau_{target})^2 ds \quad (22)$$

Si osserva che per evitare problematiche legate alla compensazione delle aree nella funzione errore si è scelto di operare l'ottimizzazione sul valore di errore quadratico. Nulla vieta, in un secondo momento, di variare la grandezza su cui si intende ottimizzare il sistema: il presente algoritmo infatti calcola anche la forza elastica del forcellone e la rigidezza ridotta della sospensione.

4.2. DATI DI INPUT

La configurazione di partenza è un requisito di fondamentale importanza per l'esecuzione dell'ottimizzazione: se infatti la prima iterata (τ_{guess}) è molto distante dal

target i tempi di calcolo aumentano notevolmente ed è da tenere in considerazione il rischio di non poter arrivare a convergenza. Riguardo a ciò è bene evidenziare che la problematica principale non è data dalla "distanza" in termini di valore dell'errore ma piuttosto in termini di andamento del parametro: se infatti si desidera un rapporto di velocità crescente 0.5-0.7 e il guess è crescente 0.3-0.5 si raggiunge la convergenza in tempi relativamente ragionevoli; viceversa se il guess equivale ad un valore costante 0.4 o decrescente, il costo computazionale cresce in modo deciso in quanto l'algoritmo dovrà necessariamente correggere di molto le posizioni iniziali delle coordinate naturali.

Una volta scelto il meccanismo di partenza è quindi conveniente eseguire una preliminare analisi cinematica del sistema, per valutare la prestazione guess: in seguito, in base ad essa si sceglie se inizializzare il processo di ottimizzazione o variare ulteriormente i parametri iniziali.

Un secondo parametro di input corrisponde a τ_{target}: poiché la simulazione prevede un sollevamento della ruota posteriore di 100 mm si è scelto di discretizzare questa lunghezza in dieci parti e per ognuna di esse si inserisce il rapporto di velocità desiderato. Ciò richiede anche l'inserimento del grado di regressione che si intende utilizzare per eseguire il fit della funzione τ_{target} impostata: per valori costanti o linearmente crescenti/decrescenti è utile selezionare la regressione lineare, mentre per valori parabolici conviene inserire la regressione con polinomio di secondo grado. Questi passaggi sono eseguiti nel codice $Matlab$ attraverso la funzione $polyfit$: l'equazione generata è di fondamentale importanza poiché a partire da essa verranno successivamente calcolati tutti i valori della funzione errore. Anche in questo caso è quindi consigliabile eseguire l'analisi cinematica preliminare, per verificare di aver inserito il target desiderato.

Questi dati devono quindi essere inseriti nel file excel "$\text{InputDataCoordinates.xls}$".

Vi sono notevoli metodologie per eseguire lo stop delle ottimizzazioni, molte delle quali basate sulla variazione di errore massimo accettabile tra due iterate successive: nel presente lavoro non è stato possibile inserire questi criteri per i motivi di seguito elencati. In primo luogo il particolare meccanismo non assicura una decisa diminuzione dell'errore al variare delle singole coordinate naturali, soprattutto se la configurazione dà una

funzione errore in prossimità di un punto di flesso a tangente orizzontale (o punto di sella nello spazio tridimensionale); inoltre una variazione significativa di una coordinata non assicura un'altrettanta significativa variazione dell'errore. Un'altra motivazione è data dalla meccanica degli algoritmi scelti: uno di essi infatti procede con iterate realizzate per piccoli incrementi/decrementi delle variabili indipendenti e quindi si corre il rischio di ottenere una diminuzione dell'errore quadratico inferiore alla tolleranza richiesta; un altro algoritmo invece non fa ricorso alle derivate della funzione obiettivo, e quindi può alternare delle elevate variazioni nei valori dell'errore quadratico a variazioni molto piccole. Per tutti questi motivi si è scelto di eseguire lo stop dei processi di ottimizzazione imponendo il valore massimo di errore accettabile: ad ogni iterata, tramite l'utilizzo di una $\mathrm{Output\ function}$, si confronta il valore di errore attuale con quello massimo. Qualora si sia giunti ad un valore inferiore rispetto al target l'algoritmo esegue lo stop e stampa i risultati, in caso contrario prosegue l'ottimizzazione del meccanismo.

Per una corretta simulazione delle prestazioni sospensive può risultare utile controllare i valori di forza elastica sulla molla e di rigidezza ridotta: a tal fine viene richiesto, in input, anche il valore di rigidezza del forcellone in $\mathrm{N/mm}$; nel caso in cui non sia ancora nota la scelta della molla, è sufficiente inserire un valore unitario (il quale permette anche di avere un'idea sulla compressione effettiva che il gruppo molla-smorzatore dovrà subire).

4.3. ESECUZIONE DEGLI ALGORITMI

Completata la fase di inserimento dati si procede all'esecuzione vera e propria dell'algoritmo; come si può osservare dalla figura 16 è possibile scegliere tra quattro diverse alternative: analisi cinematica, ottimizzazione *fminsearch*, ottimizzazione *lsqnonlin* e ottimizzazione *fminunc*.

L'$analisi\ cinematica$ si limita ad eseguire un singolo sollevamento della ruota posteriore e a fornire i risultati associati: è quindi possibile avere delle informazioni circa il comportamento e la risposta di una particolare configurazione del sistema sospensivo. Poiché l'aumento della coordinata y_7 è gestito da un'equazione che non viene soddisfatta

in forma esatta (in quanto non è un'equazione di congruenza), talvolta accade che il sollevamento simulato sia superiore ai $100\ \mathrm{mm}$ desiderati: questa eventualità inoltre può non presentarsi nelle prime iterate dei cicli di ottimizzazione ma venir fuori in seguito. Per evitare questa problematica (che potrebbe aumentare notevolmente i tempi di calcolo a causa del raggiungimento di configurazioni singolari) si fa nuovamente ricorso ad una $Output\ function$: nel processo di analisi cinematica, che richiede la risoluzione di un sistema di equazioni differenziali, viene costantemente osservato il valore della coordinata libera y_7. Quando infatti essa raggiunge il valore di $100\ \mathrm{mm}$ l'esecuzione dell'analisi viene fermata e si procede con la presentazione dei risultati della particolare configurazione geometrica.

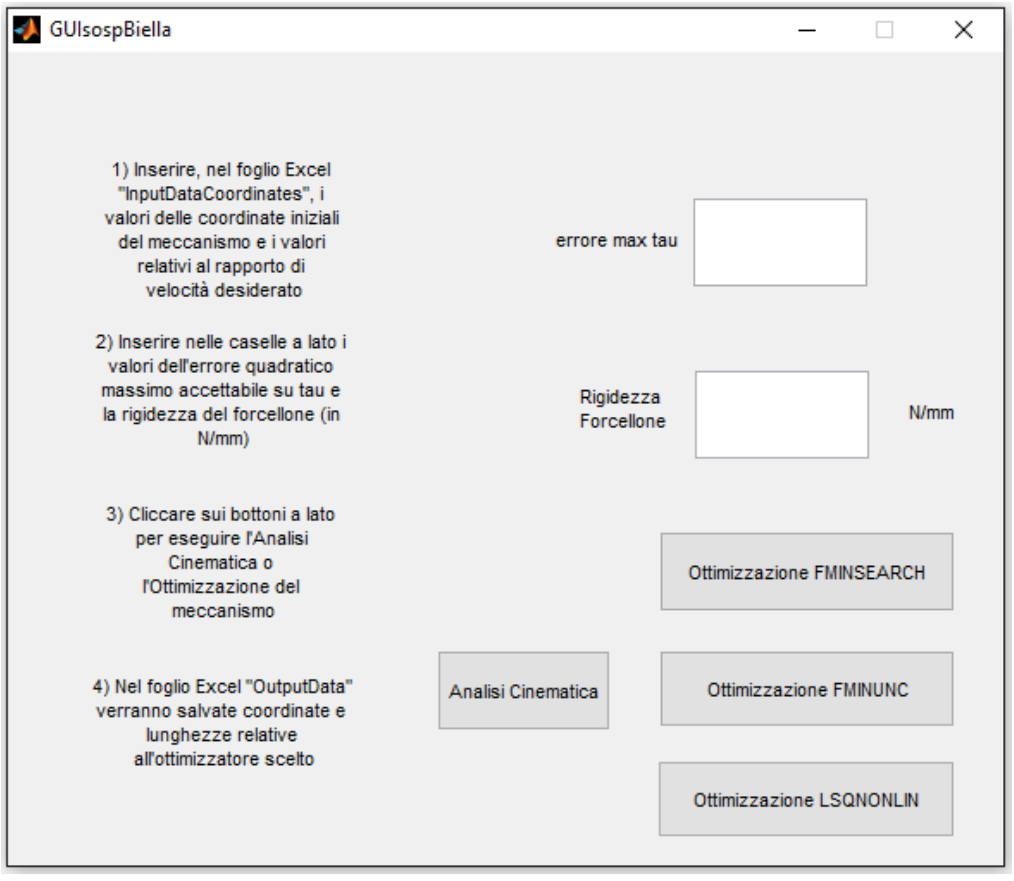


Figura 16: GUI di input per gli algoritmi di analisi e ottimizzazione

L'$ottimizzazione\ Fminsearch$ è legata all'algoritmo di Nelder-Mead e rientra nei metodi diretti di ricerca, in quanto non fa uso delle derivate della funzione obiettivo. Questo particolare ottimizzatore euristico fa riferimento al concetto di simplesso [9]: inizialmente la funzione obiettivo viene valutata in tutti i vertici del simplesso stesso (ovvero i valori iniziali delle coordinate naturali), dopodiché uno di essi viene sostituito da un altro punto (generalmente il centroide dei restanti vertici) in cui viene nuovamente valutata la funzione obiettivo. Se il nuovo punto assume un valore inferiore al precedente si procede alla sostituzione e si continua a cercare il minimo in questa direzione, altrimenti si passa ad un altro vertice ripetendo l'iter. La meccanica dell'algoritmo quindi suggerisce che le iterate immediatamente successive non necessariamente comportando ad una costante diminuzione del valore della funzione obiettivo, perciò le valutazioni vanno fatte considerando un numero più ampio di iterazioni. Questo metodo risulta essere ottimo per problemi a basso numero di variabili, in genere inferiore a 100; poiché il metodo è non derivativo, superato tale numero compare il cosiddetto "effetto dimensionalità" che di fatto lo rende poco efficace. Un'altra problematica di rilievo è il punto di partenza, che se è troppo lontano dal target può richiedere un numero elevatissimo di iterazioni con rischio di mancanza di convergenza. Tuttavia, qualora il vettore iniziale fosse vicino alla configurazione ottima, le simulazioni numeriche hanno rivelato che questo algoritmo, pur essendo il meno "robusto" dei tre, consente di arrivare a convergenza in tempi più rapidi.

L'$ottimizzazione\ Lsqnonlin$ fa riferimento alla teoria di Levenberg-Marquardt, noto anche come metodo dei minimi quadrati "smorzato". Dal punto di vista matematico esso risulta essere una combinazione di altri due metodi: il primo di essi è il "*gradient descendt*", il quale ottimizza i parametri in funzione della direzione del gradiente della funzione obiettivo ed è generalmente utilizzato quando si è molto lontani dal valore target; il secondo è del tipo "Gauss-Newton" e perciò esegue le ottimizzazioni assumendo che localmente la funzione obiettivo sia di tipo quadratico e ricerca il minimo attraverso le derivate direzionali associate. Poiché l'algoritmo necessita del calcolo della matrice Jacobiana è facile prevedere che richiederà dei tempi computazionali decisamente elevati e le simulazioni stesse lo confermano; pur essendo il metodo più lento tuttavia risulta

essere il più affidabile, in quanto ogni iterata successiva garantisce di minimizzare il valore della funzione errore e quindi di avvicinarsi al target imposto.

Anche l'$ottimizzazione\ Fminunc$ risulta essere di tipo derivativo in quanto è legata ad un algoritmo "quasi-Newton": pertanto anche essa prevede l'approssimazione della funzione obiettivo in una forma quadratica, ma ricorre alle derivate prime e seconde per ricercare i punti stazionari. Inizialmente questi metodi richiedevano il calcolo di tutta la matrice Hessiana, mentre al giorno d'oggi necessitano soltanto di un'approssimazione di una parte di essa, la quale non deve essere nemmeno invertita. Si può quindi affermare che questo algoritmo sia una via di mezzo tra i due precedentemente esposti, e quindi da preferire come primo tentativo di ottimizzazione.

4.4. STAMPA DEI RISULTATI

Al termine di ogni analisi cinematica e per ogni passo degli ottimizzatori viene realizzata una figura rappresentante quattro grafici dai quali si possono ottenere delle importanti informazioni preliminari (figura 17). Il primo di essi rappresenta, nelle coordinate spaziali, il moto della sospensione completa: viene generalmente rappresentata una "fotografia" del meccanismo ogni 5/10 iterate, evidenziando nel colore rosso la traiettoria del punto di attacco del gruppo molla-smorzatore. Questo grafico permette quindi di avere una visuale completa sul comportamento e il movimento degli elementi dell'apparato sospensivo con le grandezze in esame in quella specifica iterata. Il secondo grafico nella parte alta rappresenta l'andamento del rapporto di velocità τ al variare della coordinata y_7: con il colore verde viene riportato il τ_{target} che si intende raggiungere con la presente ottimizzazione, in blu è rappresentato il valore di partenza τ_{guess} da cui l'algoritmo deve partire per raggiungere l'ottimo mentre la linea rossa è indice di τ_{optim} ovvero il valore del rapporto di velocità dell'attuale configurazione. Ciò permette di avere un controllo visivo circa il raggiungimento del target previsto per il meccanismo in esame e il relativo andamento.

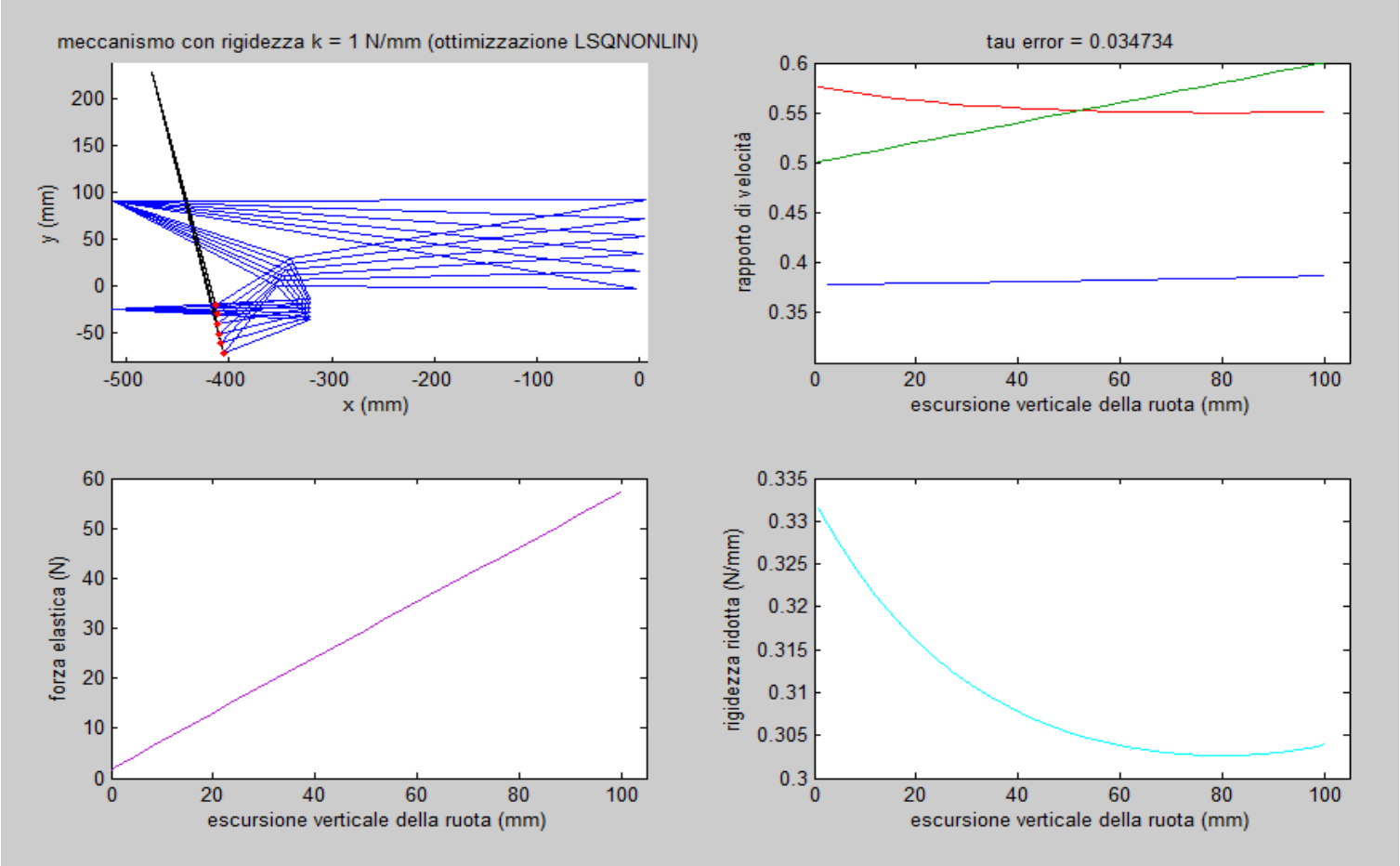


Figura 17: Esempio di output per ogni iterata degli algoritmi di ottimizzazione

Nella parte alta del grafico viene inoltre riportato il valore attuale dell'errore quadratico su τ che, lo ricordiamo, viene costantemente confrontato con l'errore massimo ammissibile inserito dall'operatore in fase di input.

Nel terzo riquadro viene rappresentato l'andamento della forza elastica di compressione nella molla in funzione del sollevamento della ruota posteriore: per ricavare tale dato si è fatto riferimento allo spostamento del punto di attacco del forcellone e moltiplicato per la rigidezza della molla inserita all'inizio della GUI. Ciò permette di fare un prima valutazione circa la progressività della sospensione in esame. (N.B.: qualora si inserisse un valore di rigidezza unitario è possibile avere un riscontro numerico circa l'ampiezza di compressione del gruppo molla-ammortizzatore).

Il quarto grafico riporta infine l'andamento della rigidezza ridotta, anch'essa in funzione del sollevamento della ruota posteriore. Matematicamente questo valore è dato dal quadrato del rapporto di velocità moltiplicato dalla rigidezza della molla inserita (in realtà questo valore non è esattamente veritiero in quanto frutto di un'approssimazione comunemente accettata in ambito accademico e motociclistico). Queste ultime due curve di rigidezza, oltre a consentire delle valutazioni circa la progressività della sospensione, possono essere spunto di ulteriori ottimizzazioni: imponendo infatti un valore target di una di esse è possibile imporre l'ottimizzazione del meccanismo su di esse anziché sul rapporto di velocità.

Oltre a queste rappresentazioni, al termine dell'ottimizzazione la GUI genera in automatico un'altra figura in cui vengono confrontate la configurazione iniziale di input e la configurazione finale ottimizzata, in modo da poter verificare a colpo d'occhio quali sono i parametri maggiormente modificati e dove si sono spostate i vari punti di interesse. Al fine di poter rendere utilizzabili alcuni dati anche dopo la chiusura del codice, si è scelto di imporre all'algoritmo di salvare tre diverse variabili: due di esse sono necessarie per la rappresentazione del rapporto di velocità in funzione di y_7 mentre la terza contiene le coordinate iniziali e ottimizzate per poter facilmente riprodurre le rispettive geometrie del meccanismo.

Infine, così come per i dati di input, per una comodità legata alla visualizzazione i risultati vengono anche salvati in un apposito foglio excel "OutputData.xls": a seconda dell'algoritmo di ottimizzazione scelto essi verranno posizionati nelle rispettive caselle, evidenziando sia le coordinate cartesiane dei punti che le lunghezze dei membri della sospensione nelle configurazioni di input e ottimizzata.

5. SIMULAZIONI E ANALISI DEI DATI

Con riferimento all'algoritmo descritto nel capitolo precedente sono state eseguite diverse simulazioni al fine di ottimizzare i meccanismi. Il parametro su cui si è deciso di incentrare il problema è il rapporto di velocità tra la velocità di compressione della molla del forcellone e la velocità di sollevamento della ruota posteriore della motocicletta; il sollevamento imposto è pari a $100\ mm$.

Per la scelta dei parametri geometrici di input si è fatto ricorso, nelle prime due simulazioni, ai dati gentilmente offerti dal gruppo $Motostudent\ Unipd$: i punti fissi corrispondono esattamente alla geometria della motocicletta sperimentale mentre per le coordinate variabili si è scelto di iniziare con un certo scostamento per valutare gli algoritmi di ottimizzazione. Nella terza simulazione tali coordinate erano troppo lontane dal punto ottimo, perciò si è fatto riferimento ai valori presenti nell'elaborato [10] con gli stessi criteri sopra descritti. Al fine di vagliare tutte le possibilità di input, per la prima e la terza simulazione si è imposto un rapporto di velocità target con andamento lineare, mentre nel secondo caso si è fatto ricorso ad una funzione parabolica.

In tutte le tre casistiche sono stati applicati tutti gli algoritmi di ottimizzazione a disposizione nella GUI i cui risultati vengono di seguito rappresentati singolarmente e poi confrontati; inoltre si evidenzia che è sempre stata utilizzata una rigidezza della molla pari a $1\ N/mm$.

5.1. RAPPORTO DI VELOCITA' COSTANTE 0.5

La prima casistica di studio riguarda una sospensione che mantiene il rapporto di velocità costante al valore 0.5 per l'intera escursione dello pneumatico posteriore: la scelta di tale valore è da imputare al fatto che la sospensione del gruppo $Motostudent\ Unipd$ è stata realizzata con lo stesso criterio, perciò è possibile eseguire un confronto diretto tra la soluzione reale, quella target e quelle ottimizzate. Per questo primo caso si è scelto un errore quadratico massimo pari a 0.0005, ovvero un ordine di grandezza più basso rispetto alle seguenti simulazioni: tale scelta è da imputare al fatto che il meccanismo $Motostudent\ Unipd$, essendo già una buona soluzione, fornisce un errore pari a circa 0.002, e quindi si è imposto un errore inferiore di un ordine di grandezza.

- *Ottimizzazione Fminsearch*

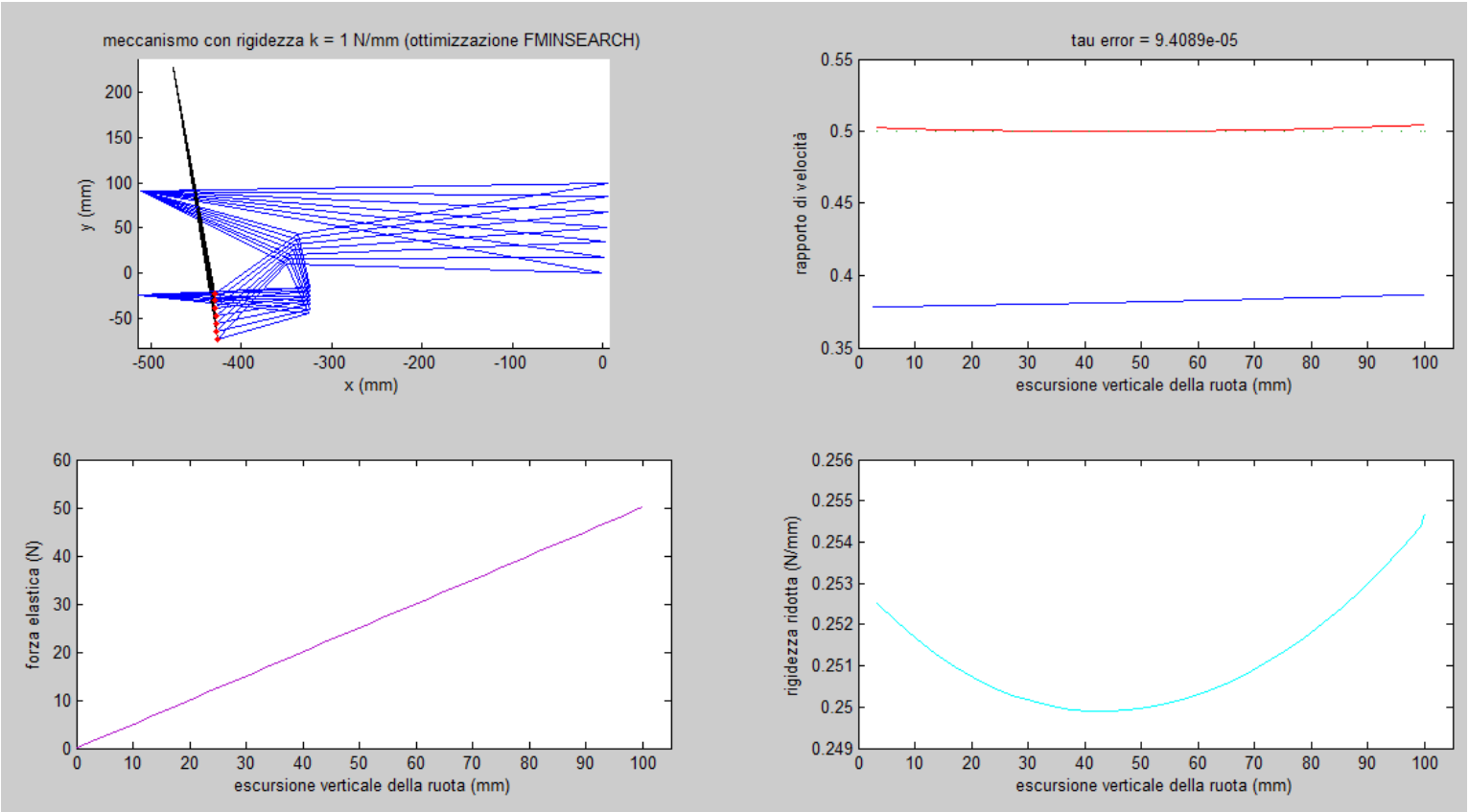


Figura 18: Grafici di meccanismo, rapporto di velocità, forza elastica e rigidezza ridotta per l'algoritmo *Fminsearch* con τ costante

Viene presentato il grafico relativo al primo algoritmo di ottimizzazione: balza subito all'occhio che questa particolare tecnica di ottimizzazione ha fornito una soluzione con un errore quadratico di addirittura un ordine di grandezza inferiore a quello massimo; ciò non deve stupire per quanto già esposto nel paragrafo precedente circa le meccaniche dell'algoritmo di Nelder-Mead. La durata della simulazione è stata di circa 18 minuti.

Si può osservare che la curva del rapporto di velocità ottenuta si avvicina molto al valore target e fornisce un comportamento della sospensione prevalentemente lineare: quanto affermato si può facilmente osservare dalla linearità conclamata della forza elastica. Anche il quarto grafico conferma un valore prevalentemente costante della rigidezza ridotta (non tragga in inganno il valore della scala dell'asse delle ordinate, il quale risulta di due ordini di grandezza inferiore rispetto ai corrispondenti grafici delle casistiche successive).

- *Ottimizzazione Lsqnonlin*

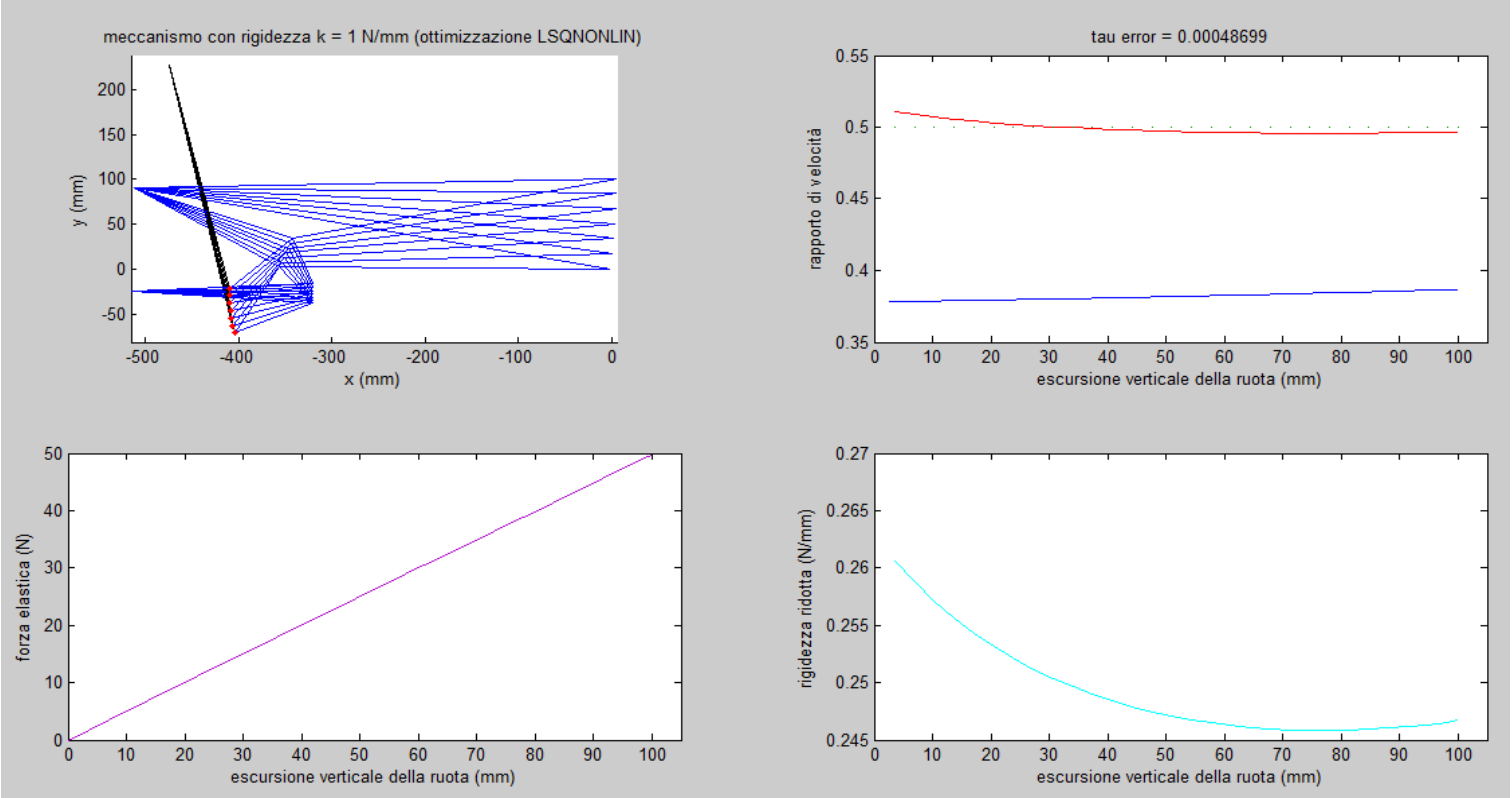


Figura 19: Grafici di meccanismo, rapporto di velocità, forza elastica e rigidezza ridotta per l'algoritmo *Lsqnonlin* con τ costante

A partire dagli stessi dati di input dell'algoritmo precedente si è realizzata l'ottimizzazione con "lsqnonlin"; l'errore ottenuto è in linea con quanto atteso in fase di input per un tempo totale di calcolo di 23 minuti.

La soluzione trovata presenta un andamento del rapporto di velocità prevalentemente decrescente per poi aumentare di valore nel tratto finale: esso parte da 0.51 per poi finire a 0.497. Tutto ciò si riflette in una sospensione leggermente regressiva ma facilmente approssimabile con una a comportamento lineare (anche in questo caso la scala nel grafico della rigidezza ridotta è diversa rispetto agli studi successivi, in quanto presenta un ordine di grandezza inferiore al confronto con essi).

- *Ottimizzazione Fminunc*

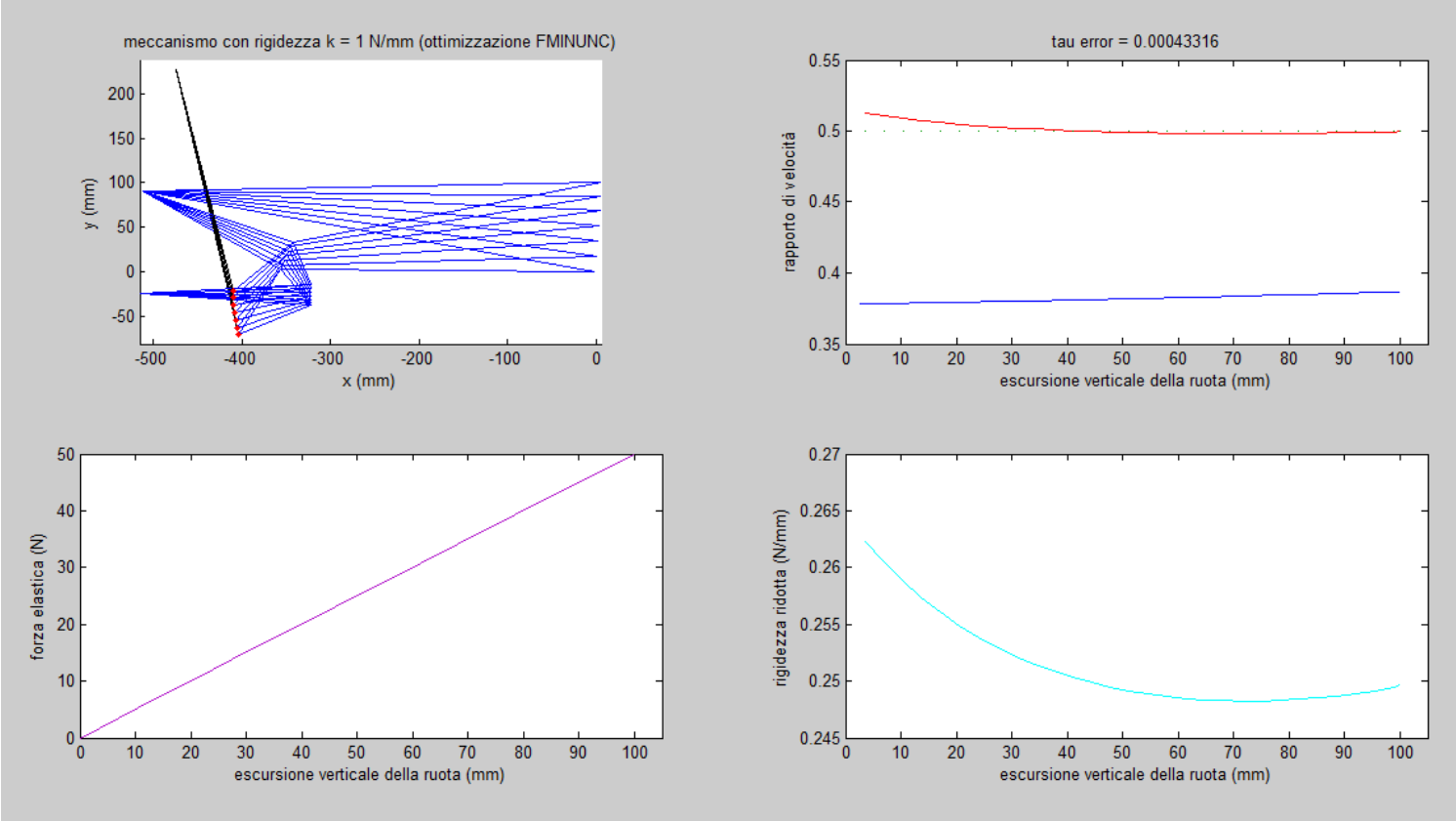


Figura 20: Grafici di meccanismo, rapporto di velocità, forza elastica e rigidezza ridotta per l'algoritmo *Fminunc* con τ costante

Anche il terzo algoritmo parte dagli stessi dati e completa l'esecuzione in un tempo approssimabile a 20 minuti.

Dai grafici riportati nella pagina precedente si può subito notare una marcata analogia con il precedente ottimizzatore: ciò è da imputare alla metodologia di ottimizzazione che risulta essere molto più affine a "*lsqnonlin*" piuttosto che a "*fminsearch*", dato che quest'ultimo algoritmo è non derivativo. Anche in questo caso infatti il rapporto di velocità è leggermente decrescente, partendo da 0.513 e concludendosi a 0.5. L'errore quadratico è quindi leggermente inferiore al caso precedente, mentre valgono le stesse considerazioni circa le curve di rigidezza.

Per poter valutare le diverse soluzioni si è scelto di riportare tutti i risultati nei medesimi grafici; sono inoltre state incluse anche le elaborazioni relative all'analisi cinematica della sospensione del gruppo $Motostudent\ Unipd$.

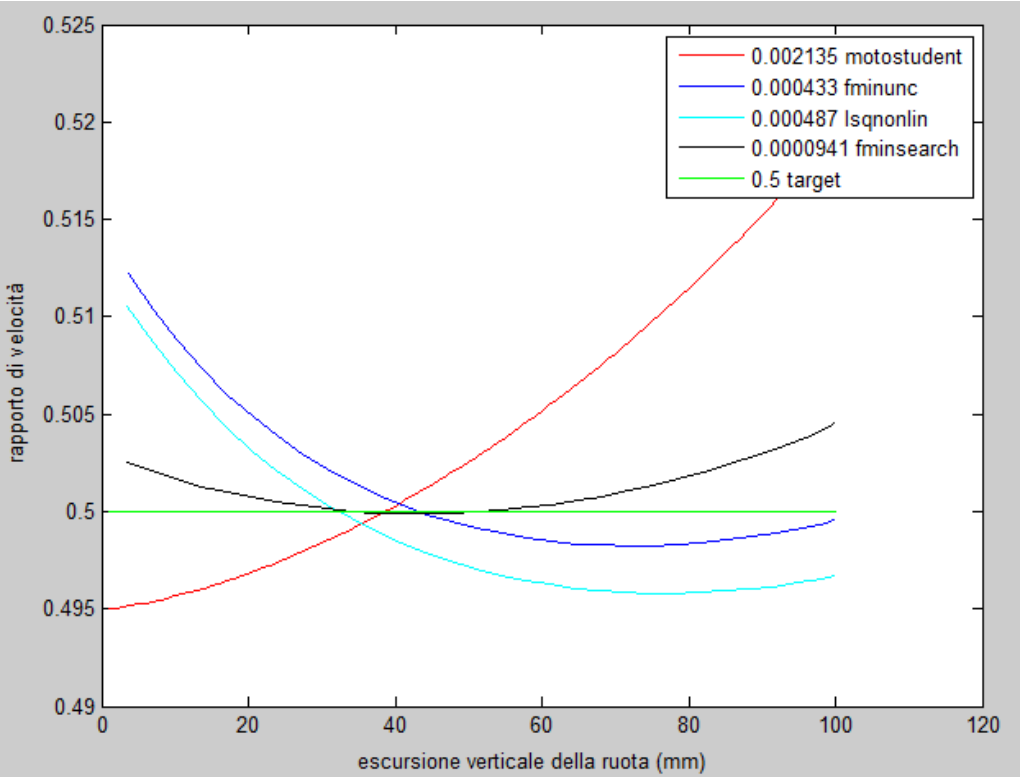


Figura 21: confronto dei rapporti di velocità ottenuti dalle soluzioni per τ costante

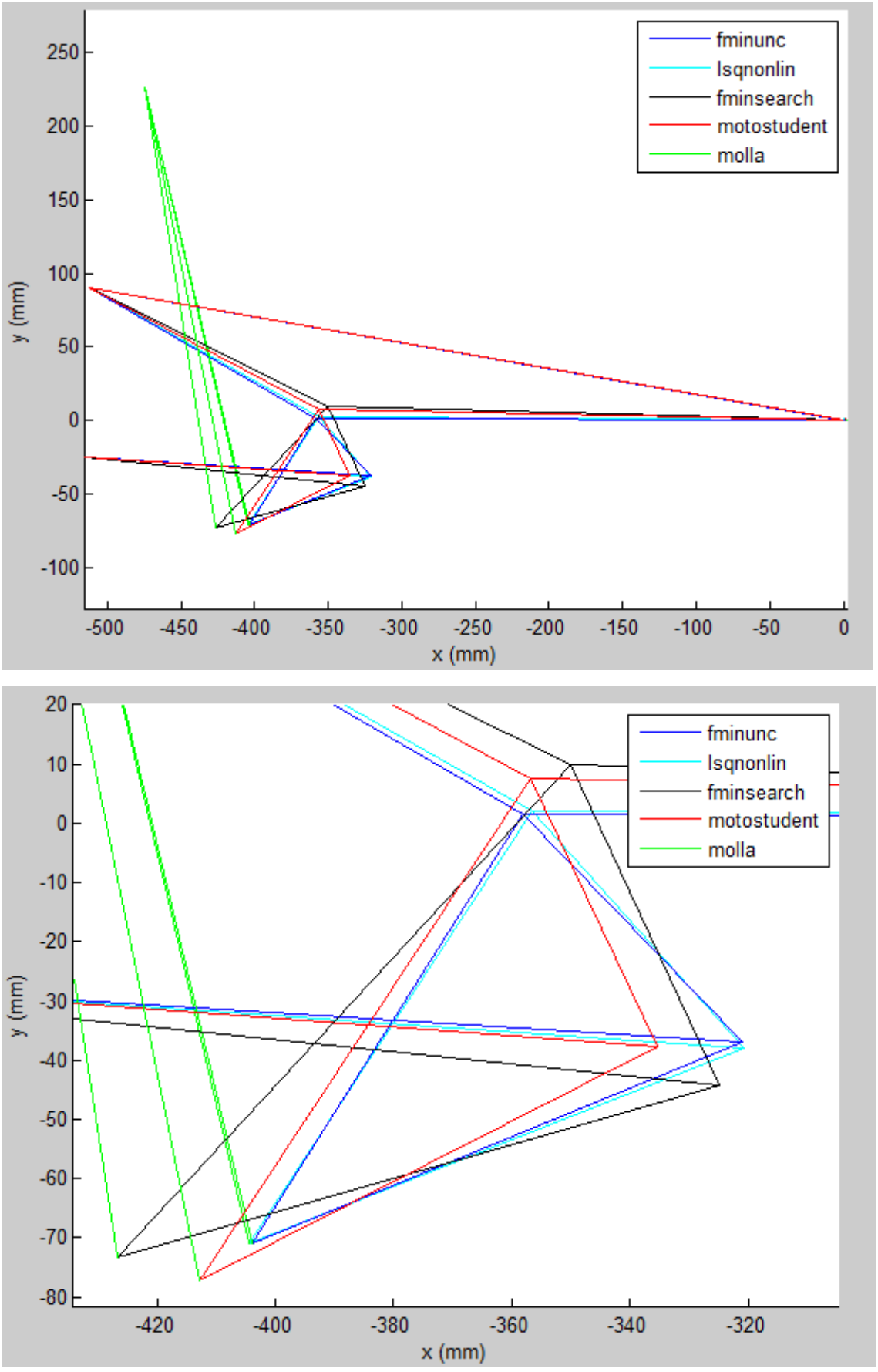


Figura 22: confronto tra i meccanismi ricavati dagli algoritmi (in alto) e Zoom nel punto di attacco del forcellone (in basso) per τ costante

	coordinate naturali e lunghezze				
	iniziali	motostudent	fminunc	lsqnonlin	fminsearch
x1	-514,98	-514,98	-514,98	-514,98	-514,98
y1	-24,7	-24,7	-24,7	-24,7	-24,7
x3	-474,76	-474,76	-474,76	-474,76	-474,76
y3	227,47	227,47	227,47	227,47	227,47
x2	-512,1	-512,1	-512,1	-512,1	-512,1
y2	90,3	90,3	90,3	90,3	90,3
x7	0	0	-1,4174	-1,6013	0
y7	0	0	0,2011	0,17258	0
x5	-320	-335,44	-321,84	-320,74	-324,883
y5	-45	-37,59	-37,477	-38,037	-44,2776
x4	-365	-356,7	-356,67	-356,35	-350,0554
y4	10	7,67	2,0906	2,2453	9,9282
x6	-400	-412,84	-403,77	-404,3	-426,6044
y6	-70	-77,31	-71,087	-70,892	-73,2847
L1	520,00	520,00	518,57	518,39	520,00
L2	167,59	176,00	178,72	178,92	180,88
L3	365,14	356,78	355,26	354,75	350,20
L4	71,06	50,00	52,71	53,77	59,77
L5	196,03	180,00	193,56	194,70	191,10
L6	83,82	87,00	88,56	89,79	105,78
L7	87,32	101,85	87,03	87,45	113,07
Lm0	306,72	311,01	306,88	306,57	304,59

Figura 23: Tabella con le coordinate naturali e lunghezze iniziali, del gruppo Motostudent e ottimizzate dagli algoritmi per τ costante

Il grafico relativo al rapporto di velocità permette di osservare che la soluzione scelta dal gruppo $Motostudent\ Unipd$ risulta essere già valida, partendo da 0.495 e crescendo sino a 0.52; i due algoritmi derivativi hanno fornito una soluzione molto simile sia in termini di andamento del rapporto di velocità sia in termini di coordinate, portando di fatto ad una configurazione di sospensione molto simile. L'algoritmo "*Fminsearch*" ha invece fornito un risultato marcatamente diverso, pur risultando il migliore in termini di τ_{optim}. Questi risultati confermano che la soluzione ottimale per un dato target non è univoca e ciò spiega , in parte, anche le notevoli differenze adottate nello schema sospensivo dalle varie

case motociclistiche per giungere agli stessi obiettivi in termini di comfort, assetto e aderenza.

5.2. RAPPORTO DI VELOCITA' CRESCENTE 0.5-0.7

Il secondo gruppo di simulazioni numeriche prevede come target un rapporto di velocità crescente in forma parabolica da 0.5 a 0.7: a tal scopo si sono imposti dei rapporti di velocità parziali in modo da rispettare l'equazione $\tau = 0.000008\, y_7^2 + 0.0012\, y_7 + 0.5$. Al fine di ottenere un meccanismo con un grado di ottimizzazione simile alla sospensione progettata dal gruppo $Motostudent\ Unipd$ si è scelto di utilizzare un errore quadratico massimo pari a 0.005. Si ricorda che la rigidezza del gruppo molla-ammortizzatore viene imposta con valore unitario; le tempistiche di calcolo degli algoritmi sono in linea con quelle relative al paragrafo precedente. Considerando i valori target del rapporto di velocità ci si aspetta di ottenere dei meccanismi dal comportamento spiccatamente progressivo, utile per permettere alle motociclette di assorbire le piccole asperità del manto stradale e contemporaneamente aumentare la rigidezza in presenza di elevate escursioni del forcellone.

- *Ottimizzazione Fminsearch*

A differenza del caso precedente, in questa occasione l'algoritmo ha proposto una soluzione in linea con i valori di errore massimo imposto. Nuovamente esso si dimostra essere il più rapido in termini di tempi di convergenza; vengono presentate nella pagina successiva le caratteristiche del meccanismo finale.

Si può facilmente osservare che la soluzione proposta risulta essere piuttosto simile al target imposto: il valore di partenza è infatti 0.5 e il valore finale circa 0.69, con un errore massimo che in generale non supera 0.02. E' da rilevare inoltre che la soluzione ottimizzata è molto diversa rispetto ai valori di primo tentativo, a dimostrazione dell'ottima capacità di convergenza del solutore. Dai due grafici in basso (in particolare dal

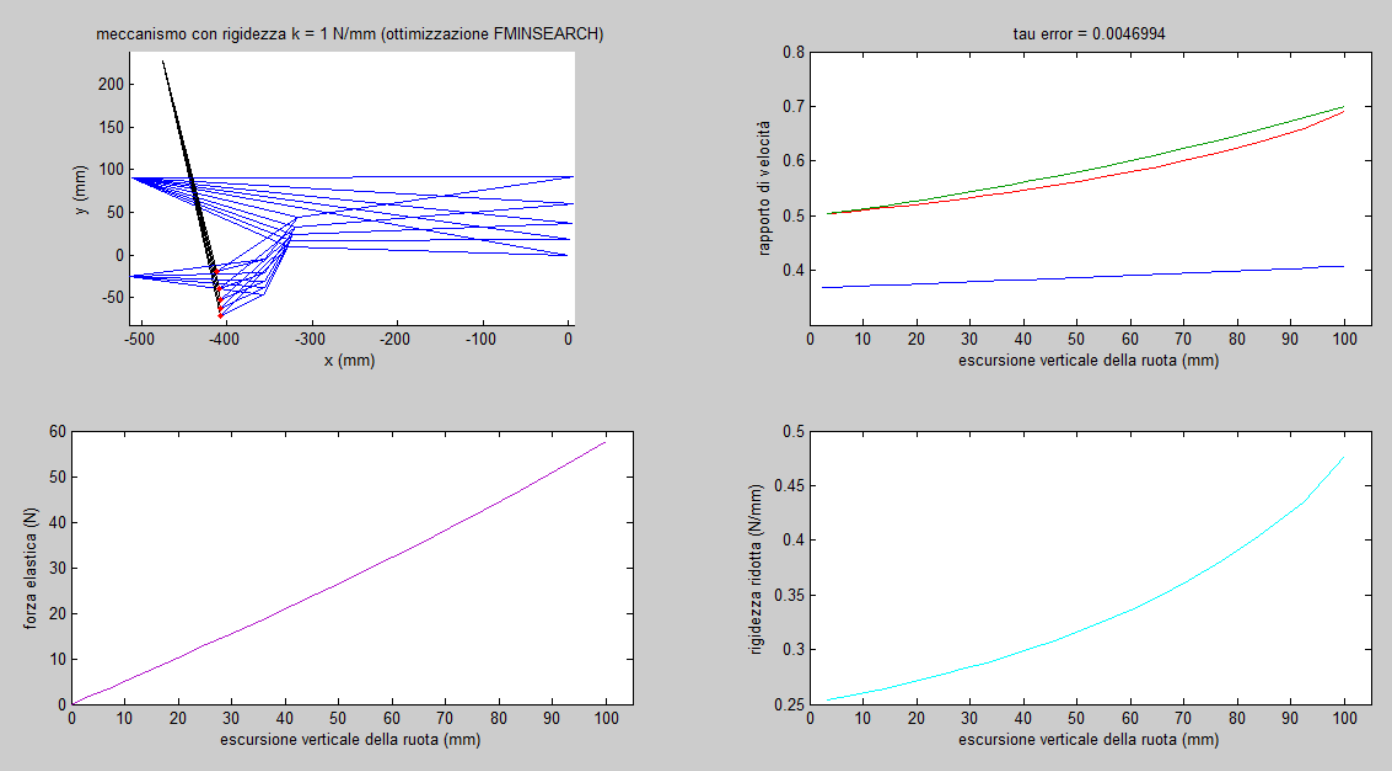


Figura 24: Grafici di meccanismo, rapporto di velocità, forza elastica e rigidezza ridotta per l'algoritmo *Fminsearch* con τ crescente

secondo) è possibile notare il previsto comportamento progressivo del gruppo sospensivo; la scelta di un valore unitario per la rigidezza della molla consente inoltre di valutare che, in corrispondenza del sollevamento pari a $100\,mm$ dello pneumatico posteriore, il gruppo molla-ammortizzatore subirà una compressione di circa $60\,mm$.

- *Ottimizzazione Lsqnonlin*

L'analisi con il metodo dei minimi quadrati ha offerto un errore leggermente superiore rispetto a quella non derivativa dell'algoritmo "*Fminsearch*" ma comunque dello stesso ordine di grandezza.

A differenza del caso precedente si può osservare che il rapporto di velocità ottimizzato assume all'inizio un valore superiore al target (0.57) mentre nella parte finale esso è inferiore (0.69). Le due curve di rigidezza sono invece molto simili alla prima soluzione e

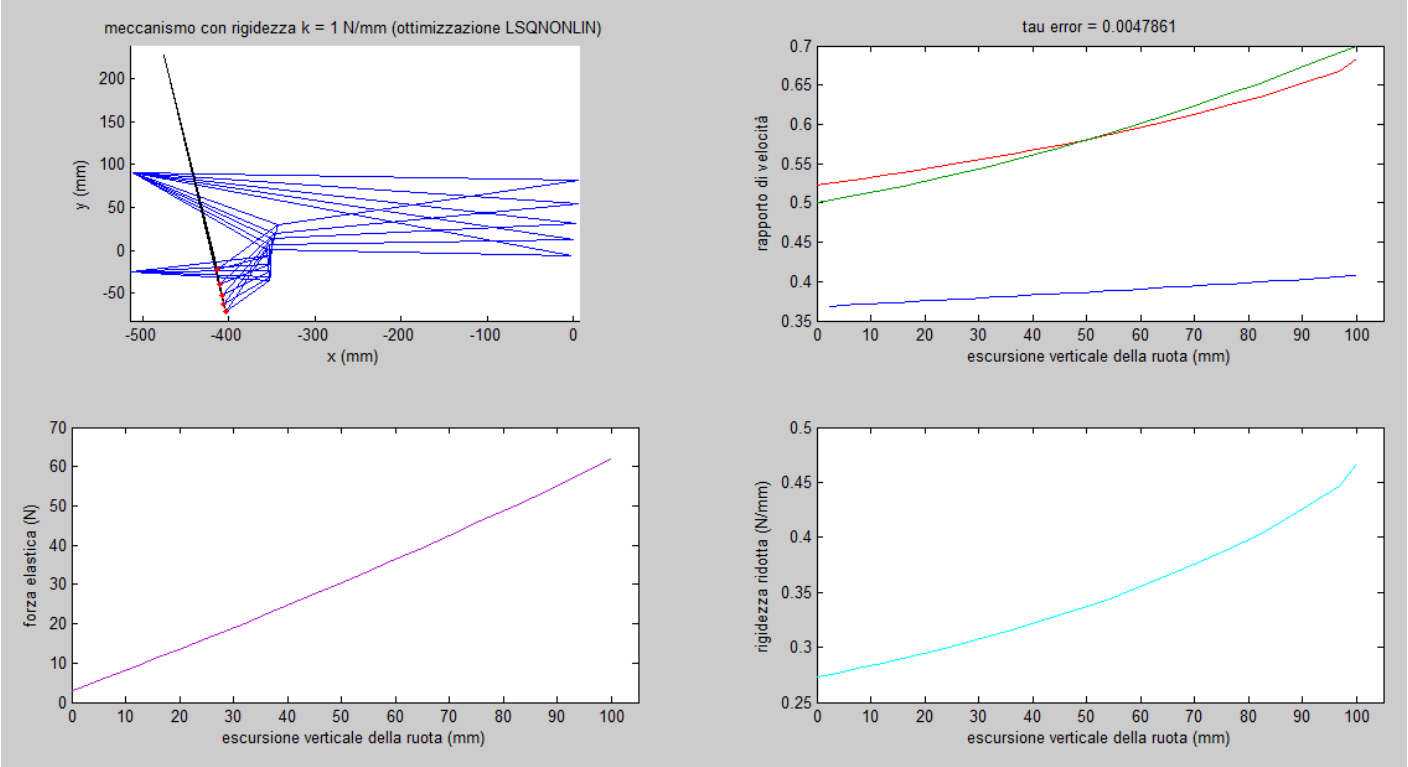


Figura 25: Grafici di meccanismo, rapporto di velocità, forza elastica e rigidezza ridotta per l'algoritmo *Lsqnonlin* con τ crescente

quindi, anche in questo caso, è possibile evidenziare il comportamento prettamente progressivo del meccanismo in esame.

- *Ottimizzazione Fminunc*

Nella casistica a rapporto di velocità crescente questo algoritmo offre la miglior soluzione, in quanto si ha un errore a regime che è la metà di quello massimo accettato.

Dal grafico dei vari andamenti di τ si evince già a colpo d'occhio che la soluzione ottimale si adatta davvero molto bene al target imposto, con scostamenti molto ridotti: i valori iniziale e finale sono infatti rispettivamente 0.51 e 0.69. Le due curve di rigidezza sono compatibili con quelle precedenti, simboleggianti ancora una volta la progressività del sistema.

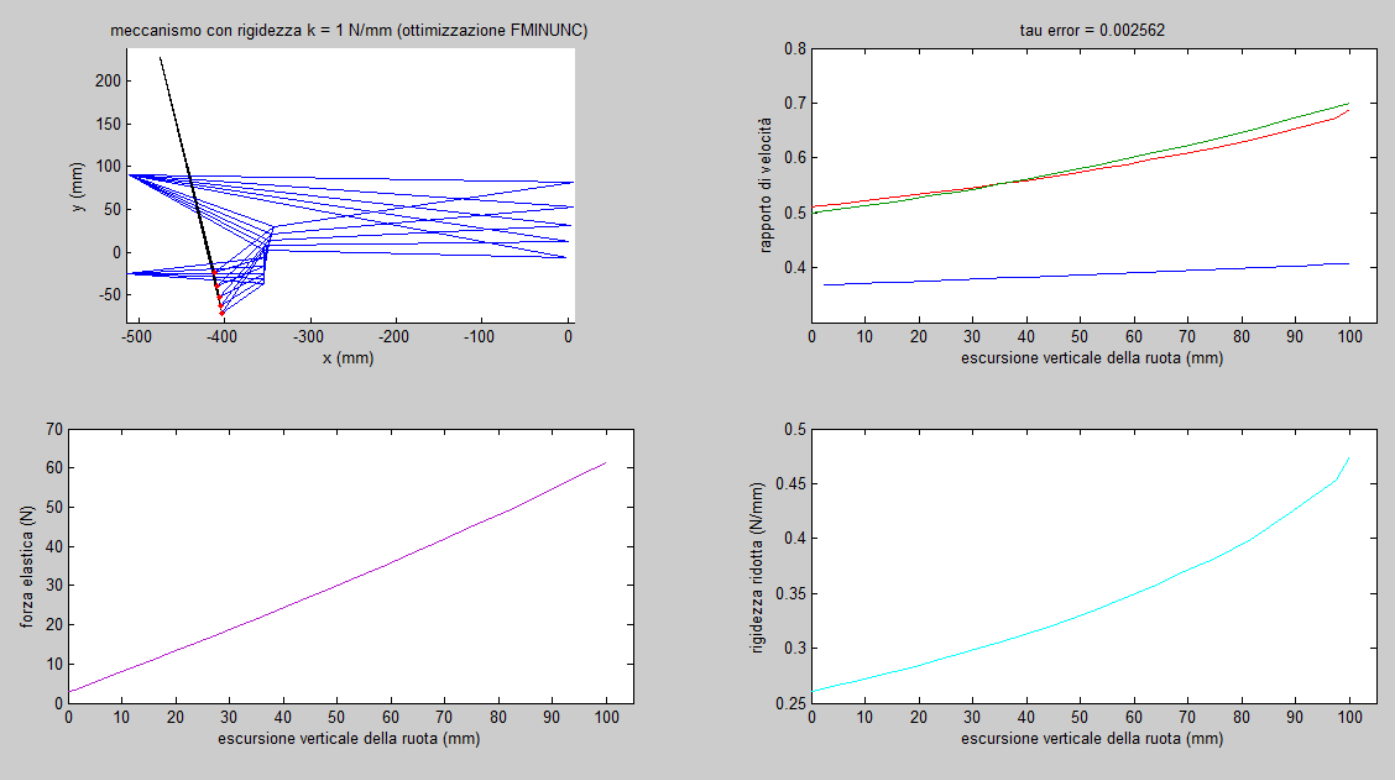


Figura 26: Grafici di meccanismo, rapporto di velocità, forza elastica e rigidezza ridotta per l'algoritmo *Fminunc* con τ crescente

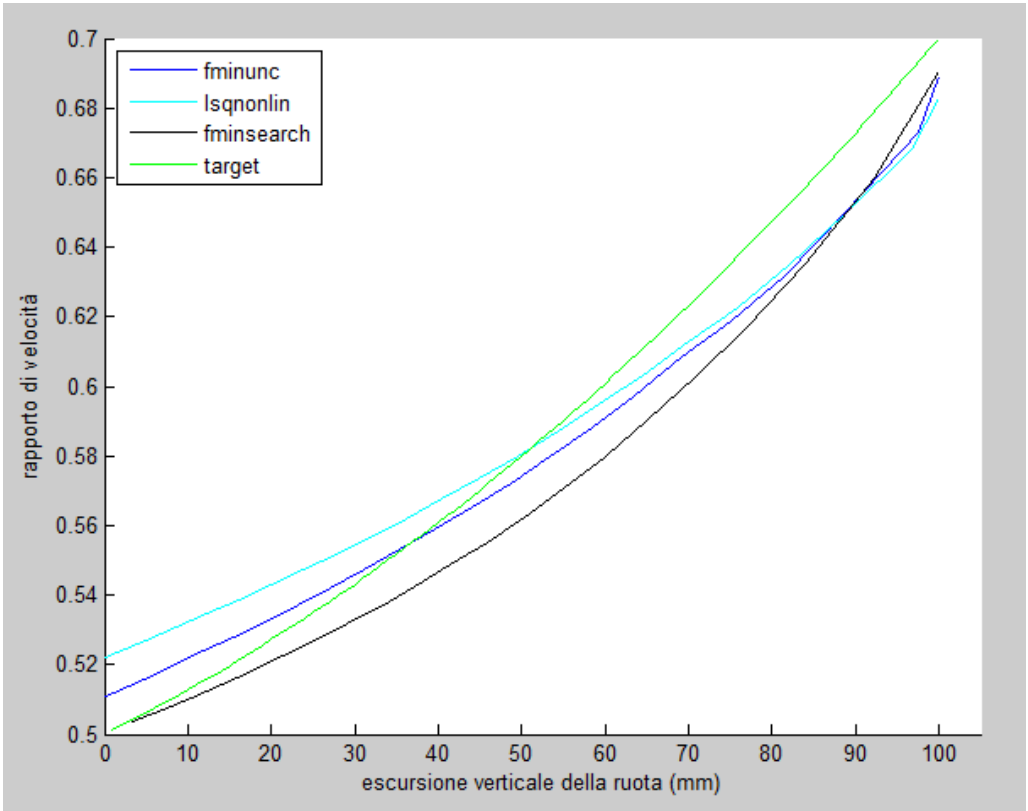


Figura 27: confronto dei rapporti di velocità ottenuti dalle soluzioni per τ crescente

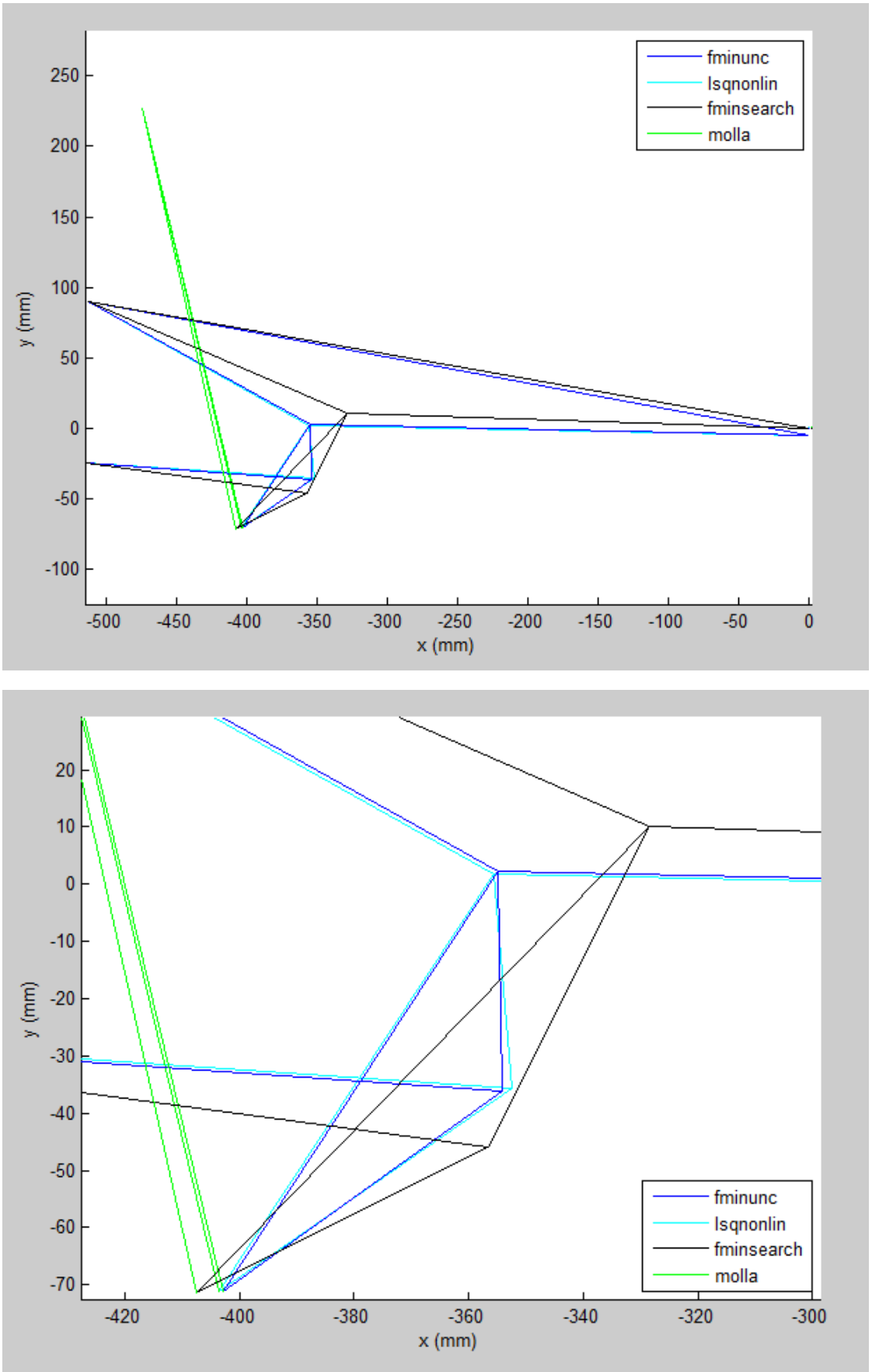


Figura 28: confronto tra i meccanismi ricavati dagli algoritmi (in alto) e Zoom nel punto di attacco del forcellone (in basso) per τ crescente

Il confronto dei risultati questa volta non si è potuto corredare con un caso realmente studiato ma, con riferimento alla casistica a rapporto di velocità costante, si può facilmente prevedere che le soluzioni delle case motociclistiche si avvicinino molto ai risultati numerici.

Il grafico di confronto tra i risultati del rapporto di velocità mostra che le soluzioni sono molto simili in termini di prestazioni; la particolarità di questa casistica è che le configurazioni geometriche associate sono molto diverse.

	coordinate naturali e lunghezze			
	iniziali	**fminunc**	**lsqnonlin**	**fminsearch**
x1	-514,98	-514,98	-514,98	-514,98
y1	-24,7	-24,7	-24,7	-24,7
x3	-474,76	-474,76	-474,76	-474,76
y3	227,47	227,47	227,47	227,47
x2	-512,1	-512,1	-512,1	-512,1
y2	90,3	90,3	90,3	90,3
x7	0	-0,78097	-0,90085	0
y7	0	-5,5899	-5,4774	0
x5	-350	-354,19	-352,41	-356,56
y5	-45	-36,2	-35,67	-45,844
x4	-365	-355	-355,7	-328,5
y4	10	2,3691	1,7079	10,188
x6	-400	-402,85	-403,65	-407,5
y6	-70	-71,226	-71,193	-71,313
L1	520,00	520,23	520,09	520,00
L2	167,59	180,03	179,75	200,32
L3	365,14	354,31	354,87	328,66
L4	57,01	38,58	37,52	62,67
L5	166,22	161,20	162,94	159,82
L6	55,90	59,96	62,35	56,95
L7	87,32	87,78	87,26	113,51
Lm0	306,72	307,23	307,01	306,26

Figura 29: Tabella con le coordinate naturali e lunghezze iniziali e ottimizzate dagli algoritmi per τ crescente

I due algoritmi derivativi danno coordinate naturali e lunghezze dei membri molto simili con biella posta verticalmente nella posizione a riposo e ciò è da imputare principalmente alla filosofia similare dei solutori; la tecnica di Nelder-Mead invece presenta una configurazione molto diversa: la biella infatti risulta essere di lunghezza quasi doppia rispetto al caso precedente e a riposo assume una posizione inclinata. Per contro il punto di attacco del forcellone rimane circa lo stesso e ciò è in accordo con le curve di forza elastica dei tre solutori che sono molto simili tra loro.

5.3. RAPPORTO DI VELOCITA' DECRESCENTE 0.5-0.35

La terza raccolta di simulazioni fa riferimento ad un rapporto di velocità decrescente: si è scelto di ricorrere ad un'interpolazione lineare per valori decrescenti da 0.5 a 0.35. Le condizioni iniziali utilizzate nei due precedenti casi risultano essere molto lontane dall'ottimo e tutti e tre gli algoritmi hanno mostrato grossi problemi circa la convergenza: come già anticipato in precedenza è infatti necessario che i valori di input non siano troppo distanti dal target prefissato. Analizzando altri risultati presenti nella letteratura è stato possibile ricavare una configurazione iniziale soddisfacente, in grado cioè di fornire dei risultati con tempi computazionali accettabili. Il punto di attacco del forcellone è stato perciò posizionato molto più in alto rispetto alle due casistiche precedenti. Poiché il range di variazione del rapporto di velocità è stato ridotto si è scelto di diminuire l'errore quadratico massimo, fissato a 0.002. Ci si aspetta di ottenere in tutti e tre i casi un comportamento sospensivo prettamente regressivo.

- *Ottimizzazione Fminsearch*

I tempi di calcolo sono ancora in linea con il primo caso di studio, perciò per giungere alla soluzione ci si attesta su circa 19 minuti.; l'errore quadratico corrisponde in termini di ordine di grandezza a quella imposto come target.

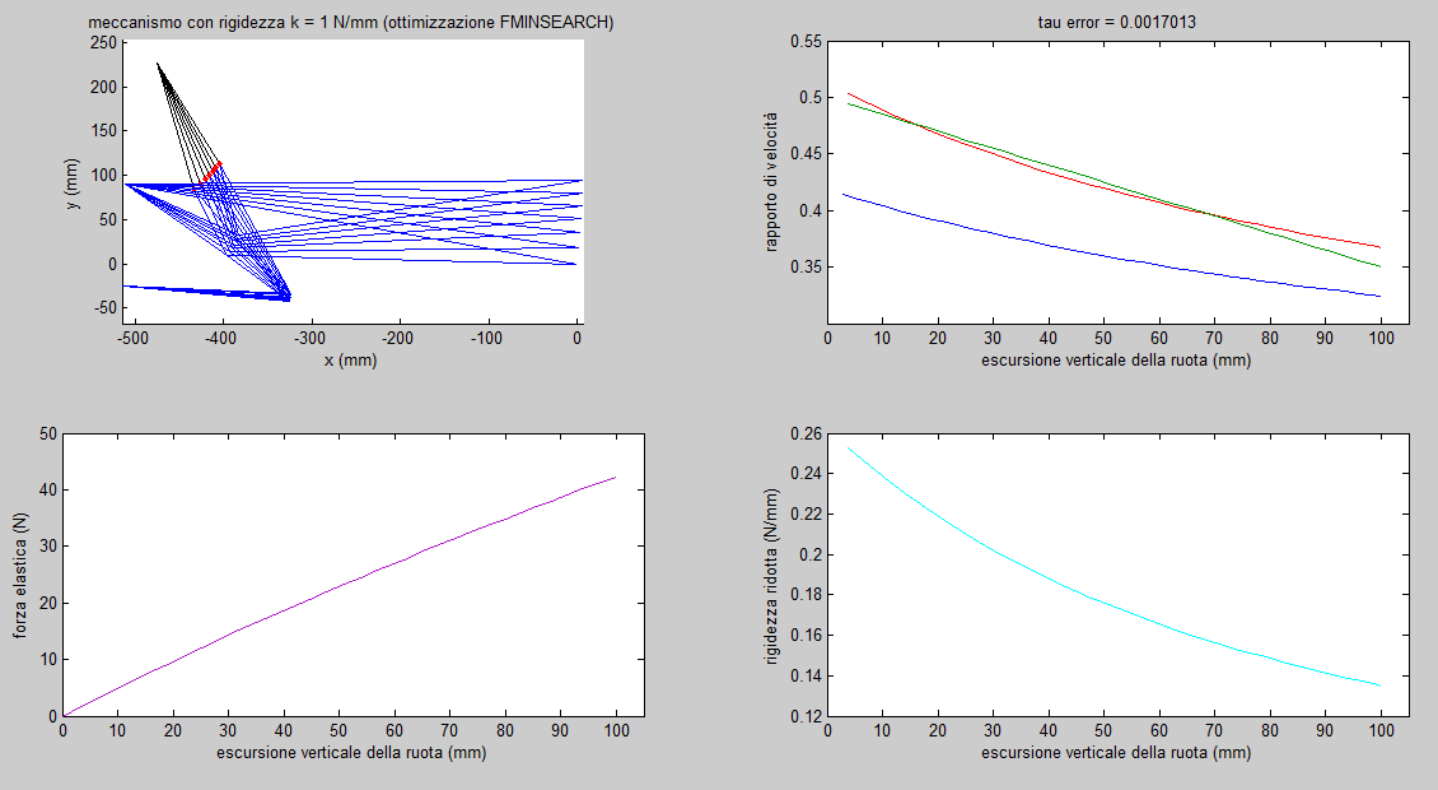


Figura 30: Grafici di meccanismo, rapporto di velocità, forza elastica e rigidezza ridotta per l'algoritmo *Fminsearch* con τ decrescente

Si può osservare che il rapporto di velocità ottimizzato si approssima molto bene a quello imposto come target, salvo poi distanziarsene nel tratto finale ma di una quota pari a 0.01 e quindi accettabile. L'escursione della molla è pari a circa 40 mm e, analizzando le due curve di rigidezza, se ne ricava che il meccanismo risponde alle previsioni di regressività iniziali, passando da una rigidezza ridotta di 0.25 N/mm ad un valore pari a 0.13 N/mm.

- *Ottimizzazione Lsqnonlin*

L'ottimizzazione con la tecnica "*Lsqnonlin*" richiede, al pari dei casi precedenti, un tempo di calcolo superiore: esso infatti corrisponde a circa 25 minuti ed è dovuto alla peculiarità della ricerca del punto di minimo del metodo stesso. Per quanto concerne l'errore ci si attesta su 0.0019 e quindi in linea con il target massimo.

Similmente al precedente solutore, anche in questo caso si ottiene un rapporto di velocità ottimale che fitta molto bene il modello lineare imposto come input. Gli scostamenti

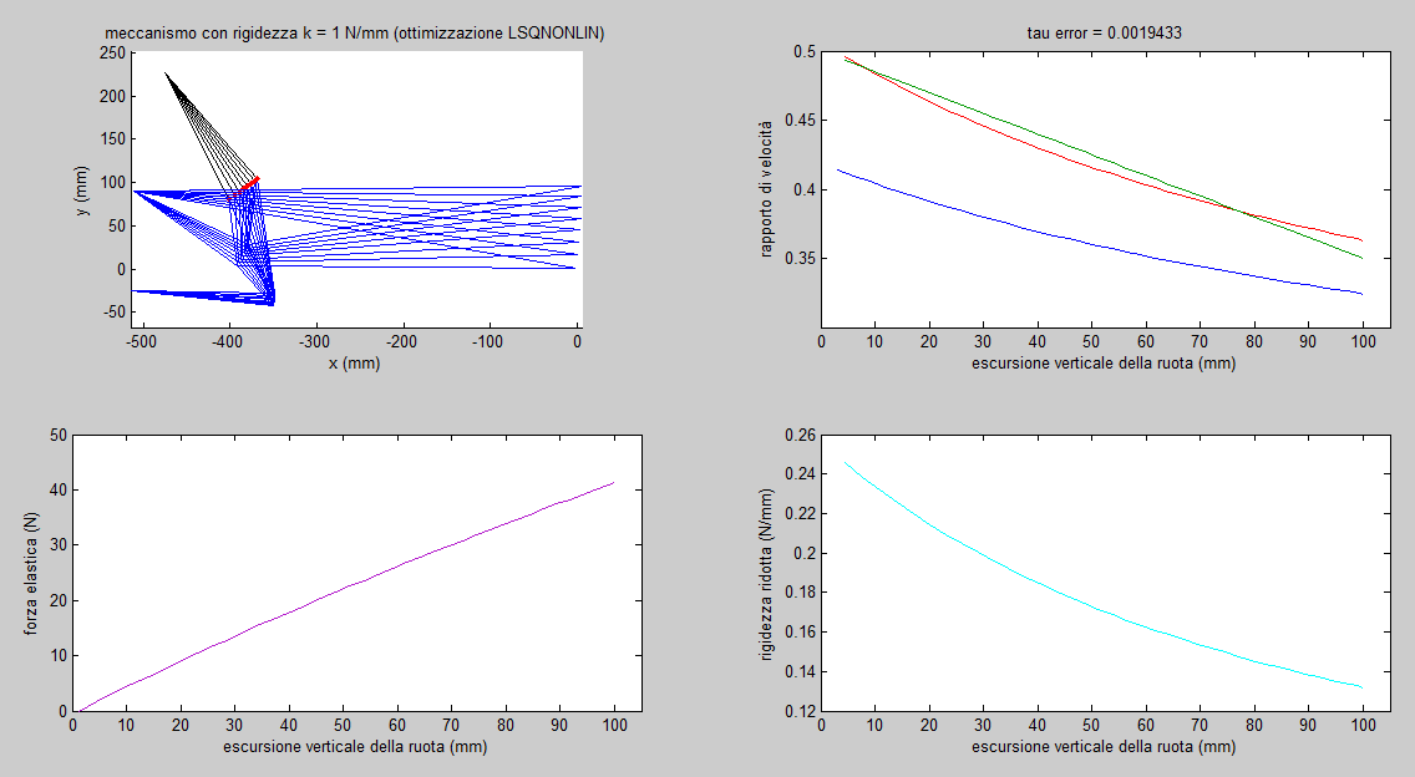


Figura 31: Grafici di meccanismo, rapporto di velocità, forza elastica e rigidezza ridotta per l'algoritmo *Lsqnonlin* con τ decrescente

massimi corrispondono ai valori finale e centrale di sollevamento del pneumatico posteriore ma non sono oltre lo 0.01.

Nonostante le diverse meccaniche di ottimizzazione si riscontra che le curve di rigidezza sono molto simili a quelle ottenute per l'algoritmo non derivativo; anche quivi infatti si osserva un comportamento spiccatamente regressivo.

- *Ottimizzazione Fminunc*

Il terzo solutore, basato sul metodo quasi-Newton, realizza l'ottimizzazione in 21 minuti ed offre l'errore quadratico minore: esso equivale infatti a 0.0014.

Il rapporto di velocità τ_{optim} ricalca le due precedenti raccolte di risultati: lo scostamento massimo si ha infatti nel punto finale in cui raggiunge il valore di 0.36 (anziché 0.35). I grafici di forza elastica del gruppo molla-ammortizzatore e di rigidezza ridotta evidenziano l'andamento regressivo del sistema sospensivo.

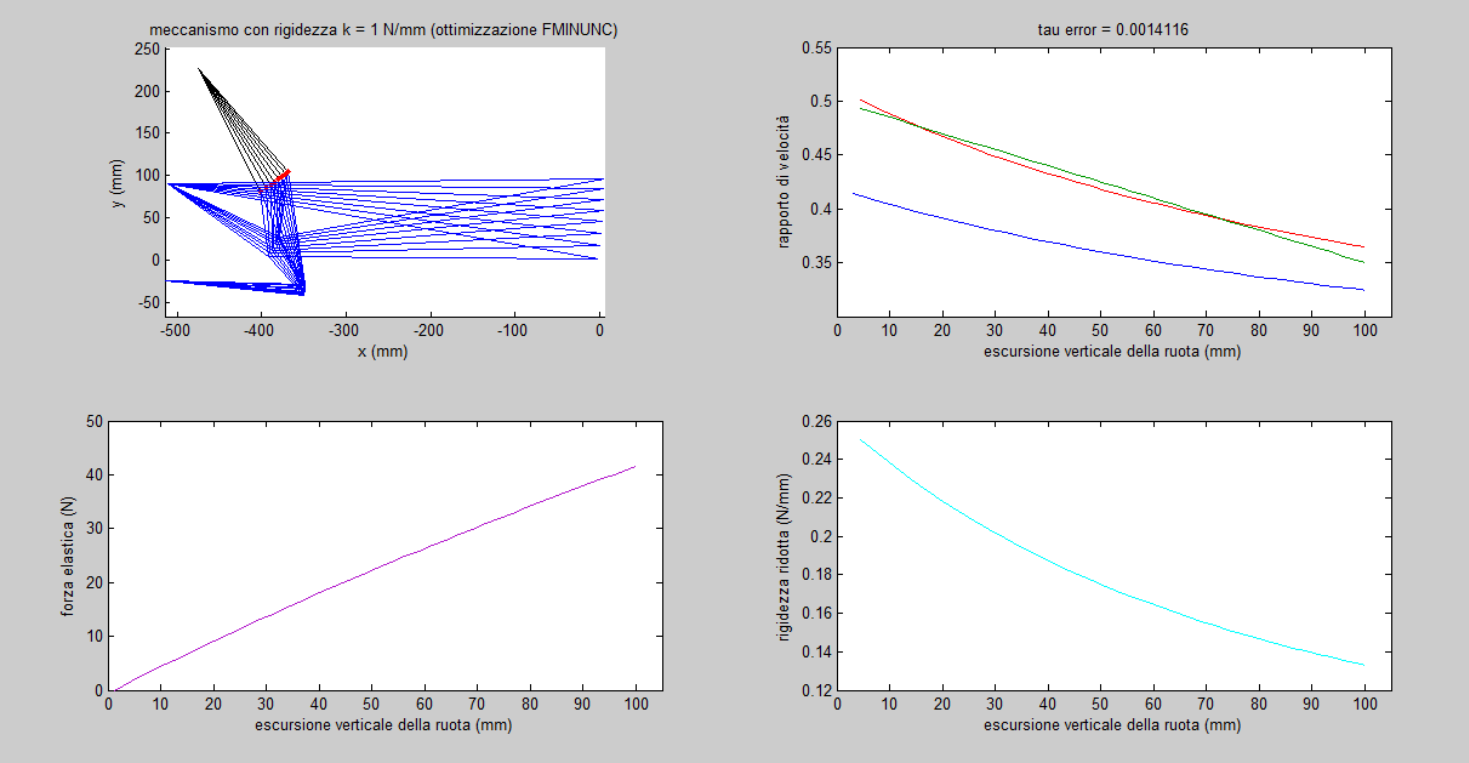


Figura 33: Grafici di meccanismo, rapporto di velocità, forza elastica e rigidezza ridotta per l'algoritmo *Fminunc* con τ decrescente

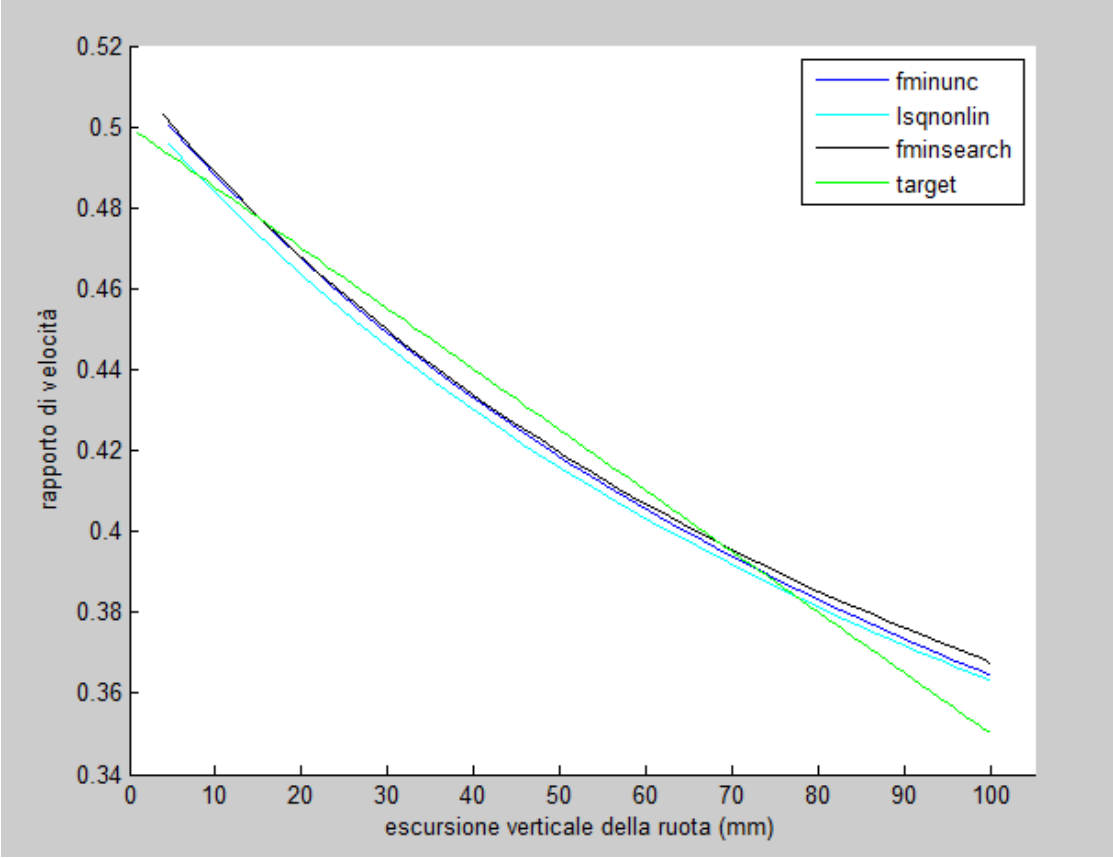


Figura 32: confronto dei rapporti di velocità ottenuti dalle soluzioni per τ decrescente

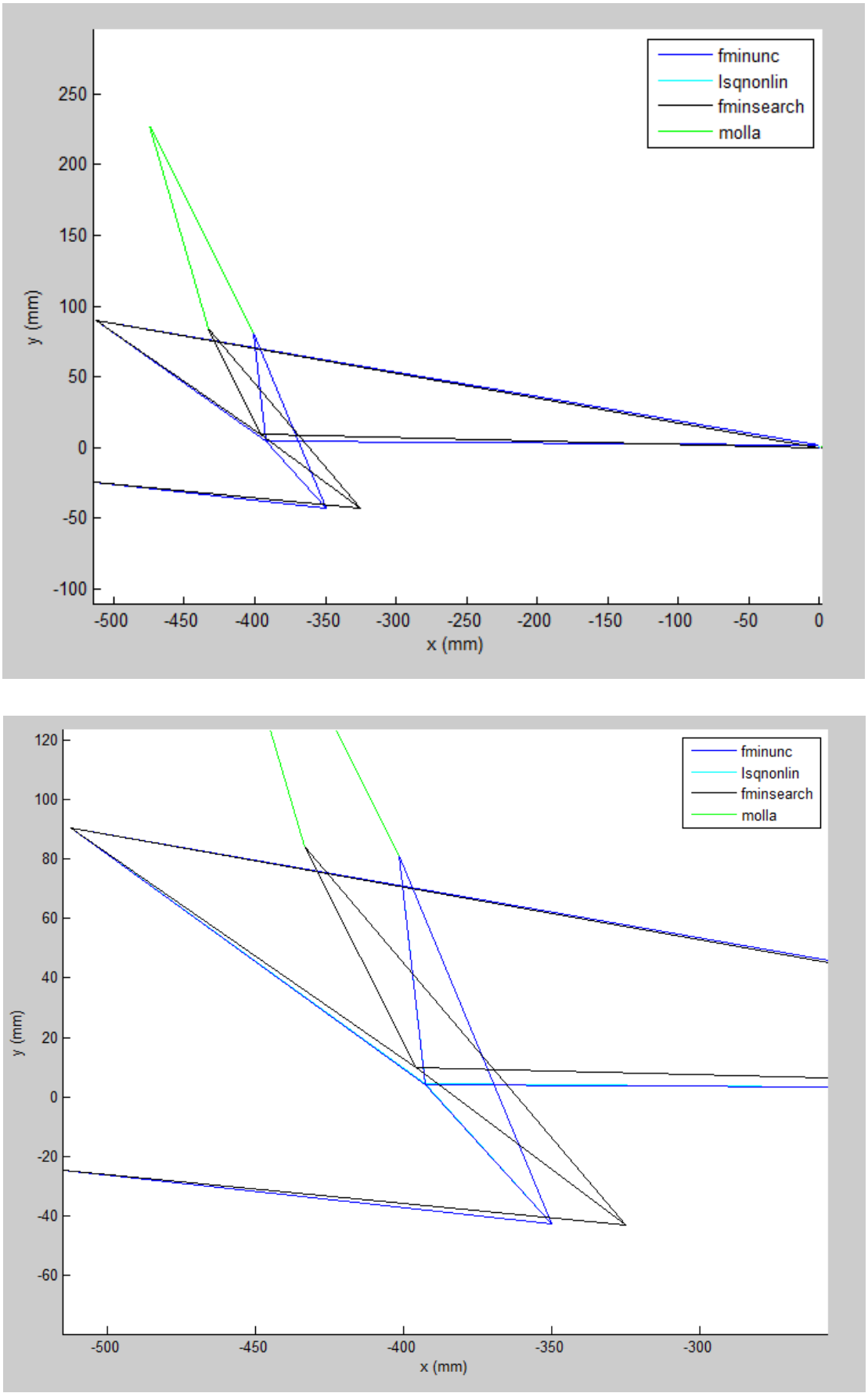


Figura 34: confronto tra i meccanismi ricavati dagli algoritmi (in alto) e Zoom nel punto di attacco del forcellone (in basso) per τ decrescente

Anche in questo caso è utile confrontare i risultati dei vari solutori per effettuare delle importanti valutazioni sui risultati.

	coordinate naturali e lunghezze			
	iniziali	**fminunc**	**lsqnonlin**	**fminsearch**
x1	-514,98	-514,98	-514,98	-514,98
y1	-24,7	-24,7	-24,7	-24,7
x3	-474,76	-474,76	-474,76	-474,76
y3	227,47	227,47	227,47	227,47
x2	-512,1	-512,1	-512,1	-512,1
y2	90,3	90,3	90,3	90,3
x7	0	-0,61755	-0,61789	0
y7	0	1,2824	1,2872	0
x5	-350	-349,99	-350,03	-325,26
y5	-45	-42,603	-42,771	-42,982
x4	-395	-392,7	-392,61	-395,86
y4	10	4,1732	4,4532	9,6528
x6	-400	-401,43	-401,43	-433,44
y6	80	81,353	81,294	84,389
L1	520,00	519,17	519,17	520,00
L2	141,99	147,22	147,13	141,48
L3	395,13	392,09	392,00	395,98
L4	71,06	63,34	63,59	88,06
L5	166,22	165,96	165,94	190,60
L6	134,63	134,21	134,29	167,11
L7	70,18	77,67	77,35	83,65
Lm0	165,34	163,49	163,54	148,93

Figura 35: Tabella con le coordinate naturali e lunghezze iniziali e ottimizzate dagli algoritmi per τ decrescente

Come già anticipato i risultati dell'ottimizzazione offrono circa la stessa soluzione in termini di rapporto di velocità, con un leggero vantaggio del terzo sistema utilizzato. Ciò che più colpisce è la marcata differenza nella configurazione dei meccanismi ottimizzati: come nel caso crescente, anche qui i due algoritmi derivativi offrono due soluzioni praticamente sovrapponibili. Il primo solutore invece si discosta da molto da tali risultati: la biella infatti assume una lunghezza nuovamente maggiore delle precedenti e pure il

punto di attacco del forcellone è più a sinistra di ben $30\ mm$; è bene notare che ciò non impedisce di giungere alle stesse soluzioni in termini di regressività ed escursione della molla.

6. CONCLUSIONI

Nel presente studio si è analizzata la sospensione posteriore motociclistica dotata di quadrilatero con forcellone fissato alla biella e al telaio. A partire dalla sintesi del meccanismo è stato realizzato un algoritmo di ottimizzazione basata sul rapporto tra la velocità di compressione del gruppo molla-ammortizzatore e la velocità di sollevamento dello pneumatico posteriore. Impostato un range di valori di tale rapporto per l'intero campo di moto e la configurazione geometrica iniziale, il codice di calcolo permette di effettuare l'analisi cinematica e tre diverse tipologie di ottimizzazione, in relazione all'errore quadratico massimo accettabile imposto e alla rigidezza della molla. Si è evidenziata l'importanza dell'inserimento di valori iniziali prossimi alla soluzione ottima, al fine di non ottenere tempi di calcolo troppo elevati per i quali non è nemmeno assicurata la convergenza. Le simulazioni hanno permesso di confrontare tra loro le diverse tecniche di ottimizzazione e le soluzioni offerte: gli algoritmi derivativi hanno presentato delle soluzioni affini mentre quello non derivativo ha offerto valori diversi ma comunque soddisfacenti le condizioni di input; ciò può essere spunto di riflessione per la realizzazione di schemi sospensivi geometricamente differenti in funzione dei vincoli di spazio della motocicletta. Infine un confronto con una soluzione in via di realizzazione ha permesso di verificare l'affidabilità degli algoritmi e ha offerto una soluzione migliorativa della stessa.

APPENDICE A

Si riporta di seguito la formulazione matematica delle matrici e dei vettori utili per lo studio dell'analisi cinematica differenziale del quadrilatero articolato.

$$[\mathbf{B}] = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ -2\frac{\partial x_d}{\partial s} \\ -2\frac{\partial y_d}{\partial s} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ (x_3 - x_d)^2 + (y_3 - y_d)^2 \end{bmatrix} ; \qquad [\mathbf{C}] = \begin{bmatrix} -2\lambda_1(x_0 - x_1) + 2\lambda_3(x_1 - x_2) - \lambda_4(a/l_2 - 1) - \lambda_5(b/l_2) \\ 2\lambda_1(y_0 - y_1) + 2\lambda_3(y_1 - y_2) + \lambda_4(b/l_2) - \lambda_5(a/l_2 - 1) \\ 2\lambda_2(x_2 - x_4) + 2\lambda_3(x_1 - x_2) + \lambda_4(a/l_2) + \lambda_5(b/l_2) \\ 2\lambda_2(y_2 - y_4) + 2\lambda_3(y_1 - y_2) - \lambda_4(b/l_2) + \lambda_5(a/l_2) \\ -\lambda_4 + 2(x_3 - x_d) \\ -\lambda_5 + 2(y_3 - y_d) \\ (x_0 - x_1)^2 + (y_0 - y_1)^2 - l_1^2 \\ (x_2 - x_4)^2 + (y_2 - y_4)^2 - l_3^2 \\ (x_1 - x_2)^2 + (y_1 - y_2)^2 - l_2^2 \\ x_3 - x_1 + a(x_1 - x_2)/l_2 - b(y_1 - y_2)/l_2 \\ y_3 - y_1 + a(y_1 - y_2)/l_2 + b(x_1 - x_2)/l_2 \\ 0 \end{bmatrix}$$

$$[\mathbf{A}]=\begin{bmatrix}
2(\lambda_1+\lambda_3) & 0 & -2\lambda_3 & 0 & 0 & 0 & 2(x_1-x_0) & 0 & 2(x_1-x_2) & 1-a/l_2 & -b/l_2 & 0\\
0 & 2(\lambda_1+\lambda_3) & 0 & -2\lambda_3 & 0 & 0 & 2(y_1-y_0) & 0 & 2(y_1-y_2) & b/l_2 & 1-a/l_2 & 0\\
-2\lambda_3 & 0 & 2(\lambda_2+\lambda_3) & 0 & 0 & 0 & 0 & 2(x_2-x_4) & 2(x_2-x_1) & a/l_2 & b/l_2 & 0\\
0 & -2\lambda_3 & 0 & 2(\lambda_2+\lambda_3) & 0 & 0 & 0 & 2(y_2-y_4) & 2(y_2-y_1) & -b/l_2 & a/l_2 & 0\\
0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & -1 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & -1 & 0\\
2(x_1-x_0) & 2(y_1-y_0) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 2(x_2-x_4) & 2(y_2-y_4) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
2(x_1-x_2) & 2(y_1-y_2) & 2(x_2-x_1) & 2(y_2-y_1) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
1-a/l_2 & b/l_2 & a/l_2 & -b/l_2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\
-b/l_2 & 1-a/l_2 & b/l_2 & a/l_2 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}$$

APPENDICE B

Vengono di seguito riportate alcuni files dell'algoritmo relativo alla GUI Matlab per la sospensione posteriore. Si omettono i files "Anakin2.mat" e "Anakin3.mat" poiché del tutto analoghi al codice riportato "Anakin1.mat". Corrispondentemente si omette anche il file "StopOptim3.mat" in quanto simile a quello riportato "StopOptim12".

- ***Codice principale "GUISospBiella.mat"***

```
function varargout = GUIsospBiella(varargin)
% GUISOSPBIELLA MATLAB code for GUIsospBiella.fig
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @GUIsospBiella_OpeningFcn, ...
                   'gui_OutputFcn',  @GUIsospBiella_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
function GUIsospBiella_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;
guidata(hObject, handles);
function varargout = GUIsospBiella_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
function tautarget_Callback(hObject, eventdata, handles)
num1 = str2double(get(hObject,'String'));
if isnan(num1)
    num1 = 0;
    set(hObject,'String',num1);
```

```
    errordlg('Input must be a number', 'Error')
end
handles.tautarget = num1;
guidata(hObject,handles)
function tautarget_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function err_max_Callback(hObject, eventdata, handles)
num2 = str2double(get(hObject,'String'));
if isnan(num2)
    num2 = 0;
    set(hObject,'String',num2);
    errordlg('Input must be a number', 'Error')
end
handles.err_max = num2;
guidata(hObject,handles)
function err_max_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function kforc_Callback(hObject, eventdata, handles)
num3 = str2double(get(hObject,'String'));
if isnan(num3)
    num3 = 0;
    set(hObject,'String',num3);
    errordlg('Input must be a number', 'Error')
end
handles.kforc = num3;
guidata(hObject,handles)
function kforc_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function AnalisiKinematic_Callback(hObject, eventdata, handles)

%ANALISI CINEMATICA

k_spring = handles.kforc;
% INSERIMENTO VALORI INIZIALI DELLE COORDINATE NATURALI
filename = 'InputDataCoordinates.xlsx';
var = xlsread(filename,'C3:C16');
input_tau=xlsread(filename,'G4:G15');
x70= var(1);
y70= var(2);
x10= var(3);
y10= var(4);
x30= var(5);
y30= var(6);
x20= var(7);
y20= var(8);
x50= var(9);
y50= var(10);
x40= var(11);
y40= var(12);
```

```
x60= var(13);
y60= var(14);
coordiniziali=[x50,y50,x40,y40,x60,y60,x70,y70];
x2=x20; y2=y20; x1=x10; y1=y10; x5=x50; y5=y50;
x4=x40; y4=y40; x6=x60; y6=y60; x7=x70; y7=y70;
syms s
%point 7 path
xd=0;
yd=200.*s;
s=0; yd=eval(yd);
coordiniz=[x50,y50,x40,y40,x60,y60,x70,y70];
% Calcolo delle lunghezze iniziali
l5=sqrt((x5-x10).^2 + (y5-y10).^2);
l4=sqrt((x5-x4).^2 + (y5-y4).^2);
l2=sqrt((x4-x20).^2 + (y4-y20).^2);
l1=sqrt((x7-x20).^2 + (y7-y20).^2);
l3=sqrt((x4-x7).^2 + (y4-y7).^2);
a0=(l4.*((x5-x4).*(x6-x5)+(y5-y6).*(y4-y5))./((y4-y5).^2.+(x4-x5).^2));
b0=(l4.*((y6-y5).*(x4-x5)-(y4-y5).*(x6-x5))./((y4-y5).^2.+(x4-x5).^2));
a=-a0; b=b0;
%% CALCOLO VALORE INIZIALE DI t0
 % equazioni di congruenza
 syms x5 y5 x4 y4 x6 y6 x7 y7;
 psi1=(x5-x1).^2.+(y5-y1).^2.-l5.^2;              %dL/dlambda1
 psi2=(x4-x2).^2.+(y4-y2).^2.-l2.^2;              %dL/dlambda2
 psi3=(x5-x4).^2.+(y5-y4).^2.-l4.^2;              %dL/dlambda3
 psi4=x6-x5-a.*(x4-x5)./l4+b.*(y4-y5)./l4;        %dL/dlambda4
 psi5=y6-y5-a.*(y4-y5)./l4-b.*(x4-x5)./l4;        %dL/dlambda5
 psi6=(x7-x2).^2 + (y7-y2).^2-l1.^2;              %dL/dlambda6
 psi7=(x4-x7).^2 + (y4-y7).^2-l3.^2;              %dL/dlambda7
 psi1=-psi1; psi2=-psi2; psi3=-psi3; psi6=-psi6; psi7=-psi7;
% funzione penalità del punto 7
 pen=(x7-xd).^2.+(y7-yd).^2;
% lagrangiano
 syms lambda1 lambda2 lambda3 lambda4 lambda5 lambda6 lambda7;
lagr=pen+lambda1.*psi1+lambda2.*psi2+lambda3.*psi3+lambda4.*psi4+lambda5.
*psi5+lambda6.*psi6+lambda7.*psi7;
 dlx5=diff(lagr, x5);                             %dL/dx5
 dly5=diff(lagr, y5);                             %dL/dy5
 dlx4=diff(lagr, x4);                             %dL/dx4
 dly4=diff(lagr, y4);                             %dL/dy4
 dlx6=diff(lagr, x6);                             %dL/dx6
 dly6=diff(lagr, y6);                             %dL/dy6
 dlx7=diff(lagr, x7);                             %dL/dx7
 dly7=diff(lagr, y7);                             %dL/dy7
x5=x50; y5=y50; x4=x40; y4=y40; x6=x60; y6=y60; x7=x70; y7=y70;
%valuto le derivate spaziali del lagrangiano
dlx5k=eval(dlx5); dly5k=eval(dly5);
dlx4k=eval(dlx4); dly4k=eval(dly4);
dlx6k=eval(dlx6); dly6k=eval(dly6);
dlx7k=eval(dlx7); dly7k=eval(dly7);
%valuto la funzione indice di performance
t=dlx5k.^2+dly5k.^2+dlx4k.^2+dly4k.^2+dlx6k.^2+dly6k.^2+dlx7k.^2+dly7k.^2
;
%% CALCOLO MOLTIPLICATORI DI LAGRANGE INIZIALI
%ricavo i moltiplicatori di Lagrange iniziali
[H, c]=equationsToMatrix([dlx5k==0, dly5k==0, dlx4k==0, dly4k==0,
dlx6k==0, dly6k==0, dlx7k==0, dly7k==0],...
```

```
    [lambda1, lambda2, lambda3, lambda4, lambda5, lambda6, lambda7]);
mltpliniz=eval(H)\eval(c);  %moltiplicatori di lagr.iniziali
mltpliniz=mltpliniz';
valinizivp=[coordiniz,mltpliniz];
lambda1=mltpliniz(1);
lambda2=mltpliniz(2);
lambda3=mltpliniz(3);
lambda4=mltpliniz(4);
lambda5=mltpliniz(5);
lambda6=mltpliniz(6);
lambda7=mltpliniz(7);
t0=eval(t);
valinizivp=[valinizivp,t0];
%% RISOLVO LA ANALISI CINEMATICA CON SIST. EQ. DIFF.
options=odeset('Events',@odeStopEvent);
kk=0:0.02:1; % integration interval of variable s
  sistma=@(s,m) systm(s,m,l5,l4,l2,l1,l3,a,b,x10,y10,x20,y20,x60,y60,t0);
[T,Y,te,ye,ie]=ode45(sistma,kk,valinizivp,options);
Ywh=Y(:,8); %consider only mechanism move with y7<100
Ywheelreal=Ywh(2:end);
%% GRAFICI
%mechanism
 figure('position', [200, 50, 1000, 600]), clf
    if true
        subplot(2,2,1)
        for is=1:5:size(Y,1)
            hold on
            line([Y(is,1),Y(is,5)],[Y(is,2),Y(is,6)])
            line([Y(is,3),Y(is,5)],[Y(is,4),Y(is,6)])
            line([Y(is,1),Y(is,3)],[Y(is,2),Y(is,4)])
            line([x10,Y(is,1)],[y10,Y(is,2)])
            line([Y(is,3),x20],[Y(is,4),y20])
            line([Y(is,3),Y(is,7)],[Y(is,4),Y(is,8)])
            line([x20,Y(is,7)],[y20,Y(is,8)])
            line([x30,Y(is,5)],[y30,Y(is,6)],'Color','k')
            plot(Y(is,5),Y(is,6),'r.') , title(['meccanismo con rigidezza
k = ',num2str(k_spring),' N/mm'])   %point out (x6,y6)
            axis equal
            xlabel('x (mm)')
            ylabel('y (mm)')
            hold off
        end
    end
%tau trend in graphic form
    XX6=abs(diff(Y(:,5))); YY6=abs(diff(Y(:,6))); YY7=diff(Y(:,8));
    XY6=sqrt(XX6.^2+YY6.^2);
    Ytau_r=[];
        for is=1:1:size(XX6)
            tau_r=XY6(is)./YY7(is);
            Ytau_r=[Ytau_r;tau_r];
        end
    Xtau=[0;10;20;30;40;50;60;70;80;90;100];
            if input_tau(12)==1
                pol=polyfit(Xtau,input_tau(1:11),1);
                Ytau_t=pol(1).*Ywheelreal+pol(2);
            else
                pol=polyfit(Xtau,input_tau(1:11),2);
                Ytau_t=pol(1).*(Ywheelreal).^2+pol(2).*Ywheelreal+pol(3);
```

```
            end
    error_tau=trapz((Ytau_r-Ytau_t).^2); %calculate error
  subplot(2,2,2)
  plot(Ywheelreal,Ytau_r,'r'),title(['tau error = ',num2str(error_tau)])
  hold on
  plot(Ywheelreal,Ytau_t,'color',[0 0.6 0])
  hold off
  xlabel('escursione verticale della ruota (mm)')
  ylabel('rapporto di velocità')
  xlim([0 105])
 % progressiveness suspension reduced stiffness graphic
    subplot(2,2,4)
    plot(Ywheelreal,k_spring.*(Ytau_r.^2),'c')
    xlabel('escursione verticale della ruota (mm)')
    ylabel('rigidezza ridotta (N/mm)')
    xlim([0 105])
%progressiveness suspension elastic force graphic
    X6i=Y(1,5); X6i=X6i.*ones(size(Y(:,5)));
    Y6i=Y(1,6); Y6i=Y6i.*ones(size(Y(:,5)));
    XY6=sqrt((Y(:,5)-X6i).^2+(Y(:,6)-Y6i).^2);
    subplot(2,2,3)
    plot(Y(:,8),XY6.*k_spring,'color',[0.7 0.1 0.8])
    xlabel('escursione verticale della ruota (mm)')
    ylabel('forza elastica (N)')
    xlim([0 105])
function Optimization1_Callback(hObject, eventdata, handles)

%ALGORITMO FMINSEARCH

errmaxtau = handles.err_max;
k_spring = handles.kforc;
% INITIAL VALUES OF NATURAL COORDINATES
filename = 'InputDataCoordinates.xlsx';
var = xlsread(filename,'C3:C16');
input_tau=xlsread(filename,'G4:G15');
x70= var(1);
y70= var(2);
x10= var(3);
y10= var(4);
x30= var(5);
y30= var(6);
x20= var(7);
y20= var(8);
x50= var(9);
y50= var(10);
x40= var(11);
y40= var(12);
x60= var(13);
y60= var(14);
coordiniziali=[x50,y50,x40,y40,x60,y60,x70,y70];
x2=x20; y2=y20; x1=x10; y1=y10; x5=x50; y5=y50;
x4=x40; y4=y40; x6=x60; y6=y60; x7=x70; y7=y70;
%% OPTIMIZATION1
options = optimset('OutputFcn', @(x, optimValues, state) stopoptim12(x,
optimValues, state,errmaxtau));
integr=@(n)
anakin1(n,x10,y10,x20,y20,x30,y30,x60,y60,coordiniziali,input_tau,k_sprin
g);
```

```
[X,fval,exitflag,output]=fminsearch(integr,coordiniziali,options);
output=output
guess=coordiniziali';
input_coord=[x10;y10;x30;y30;x20;y20;guess];
optim=X';
optim_coord=[x10;y10;x30;y30;x20;y20;optim];
indice={'x5';'y5';'x4';'y4';'x6';'y6';'x7';'y7'};
Tres=table(guess,optim,'RowNames',indice)
%optmized variables
x5=X(1);
y5=X(2);
x4=X(3);
y4=X(4);
x6=X(5);
y6=X(6);
x7=X(7);
y7=X(8);
%input initial lenghts
l50=sqrt((x50-x10).^2 + (y50-y10).^2);
l40=sqrt((x50-x40).^2 + (y50-y40).^2);
l20=sqrt((x40-x20).^2 + (y40-y20).^2);
l10=sqrt((x70-x20).^2 + (y70-y20).^2);
l30=sqrt((x40-x70).^2 + (y40-y70).^2);
l60=sqrt((x50-x60).^2 + (y50-y60).^2);
l70=sqrt((x60-x10).^2 + (y60-y10).^2);
lm00=sqrt((x30-x60).^2 + (y30-y60).^2);
input_lenghts=[l10;l20;l30;l40;l50;l60;l70;lm00];
%optimized final lenghts
l5=sqrt((x5-x10).^2 + (y5-y10).^2);
l4=sqrt((x5-x4).^2 + (y5-y4).^2);
l2=sqrt((x4-x20).^2 + (y4-y20).^2);
l1=sqrt((x7-x20).^2 + (y7-y20).^2);
l3=sqrt((x4-x7).^2 + (y4-y7).^2);
l6=sqrt((x5-x6).^2 + (y5-y6).^2);
l7=sqrt((x6-x10).^2 + (y6-y10).^2);
lm0=sqrt((x30-x6).^2 + (y30-y6).^2);
optim_lenghts=[l1;l2;l3;l4;l5;l6;l7;lm0];
%Save data
xlswrite('OutputData',input_coord,'Output','C6')
xlswrite('OutputData',input_lenghts,'Output','C21')
xlswrite('OutputData',optim_coord,'Output','D6')
xlswrite('OutputData',optim_lenghts,'Output','D21')
%compare initial and optimized mechanism
        figure(150),set(gcf,'name','confronto meccanismi'),clf
        plot(x60,y60,'b')
        hold on
        plot(x6,y6,'r')
        legend('meccanismo iniziale','meccanismo optim')
            line([x10,x5],[y10,y5],'Color','r')
            line([x5,x4],[y5,y4],'Color','r')
            line([x4,x20],[y4,y20],'Color','r')
            line([x5,x6],[y5,y6],'Color','r')
            line([x4,x6],[y4,y6],'Color','r')
            line([x20,x7],[y20,y7],'Color','r')
            line([x4,x7],[y4,y7],'Color','r')
            line([x6,x30],[y6,y30],'Color','g')
        line([x10,x50],[y10,y50],'Color','b')
        line([x50,x40],[y50,y40],'Color','b')
```

```
        line([x40,x20],[y40,y20],'Color','b')
        line([x50,x60],[y50,y60],'Color','b')
        line([x40,x60],[y40,y60],'Color','b')
        line([x20,x70],[y20,y70],'Color','b')
        line([x40,x70],[y40,y70],'Color','b')
        line([x60,x30],[y60,y30],'Color','g')
        axis equal
        xlabel('x (mm)')
        ylabel('y (mm)')
function optimization2_Callback(hObject, eventdata, handles)
%ALGORITMO FMINUNC
errmaxtau = handles.err_max;
k_spring = handles.kforc;
% INITIAL VALUES OF NATURAL COORDINATES
filename = 'InputDataCoordinates.xlsx';
var = xlsread(filename,'C3:C16');
input_tau=xlsread(filename,'G4:G15');
x70= var(1);
y70= var(2);
x10= var(3);
y10= var(4);
x30= var(5);
y30= var(6);
x20= var(7);
y20= var(8);
x50= var(9);
y50= var(10);
x40= var(11);
y40= var(12);
x60= var(13);
y60= var(14);
coordiniziali=[x50,y50,x40,y40,x60,y60,x70,y70];
x2=x20; y2=y20; x1=x10; y1=y10; x5=x50; y5=y50;
x4=x40; y4=y40; x6=x60; y6=y60; x7=x70; y7=y70;
%% OPTIMIZATION1
options=optimoptions(@fminunc,'Algorithm','quasi-Newton','OutputFcn',
@(x, optimValues, state)stopoptim12(x, optimValues, state,errmaxtau));
integr=@(n)
anakin2(n,x10,y10,x20,y20,x30,y30,x60,y60,coordiniziali,input_tau,k_sprin
g);
[X,fval,exitflag,output]=fminunc(integr,coordiniziali,options);
 output=output
guess=coordiniziali';
input_coord=[x10;y10;x30;y30;x20;y20;guess];
optim=X';
optim_coord=[x10;y10;x30;y30;x20;y20;optim];
indice={'x5';'y5';'x4';'y4';'x6';'y6';'x7';'y7'};
Tres=table(guess,optim,'RowNames',indice)
%optmized variables
x5=X(1);
y5=X(2);
x4=X(3);
y4=X(4);
x6=X(5);
y6=X(6);
x7=X(7);
y7=X(8);
%input initial lenghts
```

```
l50=sqrt((x50-x10).^2 + (y50-y10).^2);
l40=sqrt((x50-x40).^2 + (y50-y40).^2);
l20=sqrt((x40-x20).^2 + (y40-y20).^2);
l10=sqrt((x70-x20).^2 + (y70-y20).^2);
l30=sqrt((x40-x70).^2 + (y40-y70).^2);
l60=sqrt((x50-x60).^2 + (y50-y60).^2);
l70=sqrt((x60-x10).^2 + (y60-y10).^2);
lm00=sqrt((x30-x60).^2 + (y30-y60).^2);
input_lenghts=[l10;l20;l30;l40;l50;l60;l70;lm00];
%optimized final lenghts
l5=sqrt((x5-x10).^2 + (y5-y10).^2);
l4=sqrt((x5-x4).^2 + (y5-y4).^2);
l2=sqrt((x4-x20).^2 + (y4-y20).^2);
l1=sqrt((x7-x20).^2 + (y7-y20).^2);
l3=sqrt((x4-x7).^2 + (y4-y7).^2);
l6=sqrt((x5-x6).^2 + (y5-y6).^2);
l7=sqrt((x6-x10).^2 + (y6-y10).^2);
lm0=sqrt((x30-x6).^2 + (y30-y6).^2);
optim_lenghts=[l1;l2;l3;l4;l5;l6;l7;lm0];
%Save data
xlswrite('OutputData',input_coord,'Output','K6')
xlswrite('OutputData',input_lenghts,'Output','K21')
xlswrite('OutputData',optim_coord,'Output','L6')
xlswrite('OutputData',optim_lenghts,'Output','L21')
%compare initial and optimized mechanism
        figure(150),set(gcf,'name','confronto meccanismi'),clf
        plot(x60,y60,'b')
        hold on
        plot(x6,y6,'r')
        legend('meccanismo iniziale','meccanismo optim')
            line([x10,x5],[y10,y5],'Color','r')
            line([x5,x4],[y5,y4],'Color','r')
            line([x4,x20],[y4,y20],'Color','r')
            line([x5,x6],[y5,y6],'Color','r')
            line([x4,x6],[y4,y6],'Color','r')
            line([x20,x7],[y20,y7],'Color','r')
            line([x4,x7],[y4,y7],'Color','r')
            line([x6,x30],[y6,y30],'Color','g')
        line([x10,x50],[y10,y50],'Color','b')
        line([x50,x40],[y50,y40],'Color','b')
        line([x40,x20],[y40,y20],'Color','b')
        line([x50,x60],[y50,y60],'Color','b')
        line([x40,x60],[y40,y60],'Color','b')
        line([x20,x70],[y20,y70],'Color','b')
        line([x40,x70],[y40,y70],'Color','b')
        line([x60,x30],[y60,y30],'Color','g')
        axis equal
        xlabel('x (mm)')
        ylabel('y (mm)')
function optimization3_Callback(hObject, eventdata, handles)

%ALGORITMO LSQNONLIN

errmaxtau = handles.err_max;
k_spring = handles.kforc;
% INITIAL VALUES OF NATURAL COORDINATES
filename = 'InputDataCoordinates.xlsx';
var = xlsread(filename,'C3:C16');
```

```
input_tau=xlsread(filename,'G4:G15');
x70= var(1);
y70= var(2);
x10= var(3);
y10= var(4);
x30= var(5);
y30= var(6);
x20= var(7);
y20= var(8);
x50= var(9);
y50= var(10);
x40= var(11);
y40= var(12);
x60= var(13);
y60= var(14);
coordiniziali=[x50,y50,x40,y40,x60,y60,x70,y70];
x2=x20; y2=y20; x1=x10; y1=y10; x5=x50; y5=y50;
x4=x40; y4=y40; x6=x60; y6=y60; x7=x70; y7=y70;
%% OPTIMIZATION1
options=optimoptions(@lsqnonlin,'Algorithm','levenberg-
marquardt','OutputFcn', @(x, optimValues, state)stopoptim3(x,
optimValues, state,errmaxtau));
integr=@(n)
anakin3(n,x10,y10,x20,y20,x30,y30,x60,y60,coordiniziali,input_tau,k_sprin
g);
[X,fval,exitflag,output]=lsqnonlin(integr,coordiniziali,[],[],options);
output=output
guess=coordiniziali';
input_coord=[x10;y10;x30;y30;x20;y20;guess];
optim=X;
optim_coord=[x10;y10;x30;y30;x20;y20;optim];
indice={'x5';'y5';'x4';'y4';'x6';'y6';'x7';'y7'};
Tres=table(guess,optim,'RowNames',indice)
%optmized variables
x5=X(1);
y5=X(2);
x4=X(3);
y4=X(4);
x6=X(5);
y6=X(6);
x7=X(7);
y7=X(8);
%input initial lenghts
l50=sqrt((x50-x10).^2 + (y50-y10).^2);
l40=sqrt((x50-x40).^2 + (y50-y40).^2);
l20=sqrt((x40-x20).^2 + (y40-y20).^2);
l10=sqrt((x70-x20).^2 + (y70-y20).^2);
l30=sqrt((x40-x70).^2 + (y40-y70).^2);
l60=sqrt((x50-x60).^2 + (y50-y60).^2);
l70=sqrt((x60-x10).^2 + (y60-y10).^2);
lm00=sqrt((x30-x60).^2 + (y30-y60).^2);
input_lenghts=[l10;l20;l30;l40;l50;l60;l70;lm00];
%optimized final lenghts
l5=sqrt((x5-x10).^2 + (y5-y10).^2);
l4=sqrt((x5-x4).^2 + (y5-y4).^2);
l2=sqrt((x4-x20).^2 + (y4-y20).^2);
l1=sqrt((x7-x20).^2 + (y7-y20).^2);
l3=sqrt((x4-x7).^2 + (y4-y7).^2);
```

```
l6=sqrt((x5-x6).^2 + (y5-y6).^2);
l7=sqrt((x6-x10).^2 + (y6-y10).^2);
lm0=sqrt((x30-x6).^2 + (y30-y6).^2);
optim_lenghts=[l1;l2;l3;l4;l5;l6;l7;lm0];
%Save data
xlswrite('OutputData',input_coord,'Output','G6')
xlswrite('OutputData',input_lenghts,'Output','G21')
xlswrite('OutputData',optim_coord,'Output','H6')
xlswrite('OutputData',optim_lenghts,'Output','H21')
%compare initial and optimized mechanism
        figure(150),set(gcf,'name','confronto meccanismi'),clf
        plot(x60,y60,'b')
        hold on
        plot(x6,y6,'r')
        legend('meccanismo iniziale','meccanismo optim')
            line([x10,x5],[y10,y5],'Color','r')
            line([x5,x4],[y5,y4],'Color','r')
            line([x4,x20],[y4,y20],'Color','r')
            line([x5,x6],[y5,y6],'Color','r')
            line([x4,x6],[y4,y6],'Color','r')
            line([x20,x7],[y20,y7],'Color','r')
            line([x4,x7],[y4,y7],'Color','r')
            line([x6,x30],[y6,y30],'Color','g')
        line([x10,x50],[y10,y50],'Color','b')
        line([x50,x40],[y50,y40],'Color','b')
        line([x40,x20],[y40,y20],'Color','b')
        line([x50,x60],[y50,y60],'Color','b')
        line([x40,x60],[y40,y60],'Color','b')
        line([x20,x70],[y20,y70],'Color','b')
        line([x40,x70],[y40,y70],'Color','b')
        line([x60,x30],[y60,y30],'Color','g')
        axis equal
        xlabel('x (mm)')
        ylabel('y (mm)')
```

- ***Function "Anakin1.mat"***

```
function
f=anakin1(n,x10,y10,x20,y20,x30,y30,x60,y60,coordiniziali,input_tau,k_spr
ing);
x5=n(1);   %specific natural coordinates as variables
y5=n(2);
x4=n(3);
y4=n(4);
x6=n(5);
y6=n(6);
x7=n(7);
y7=n(8);
x2=x20; y2=y20; x1=x10; y1=y10; x3=x30; y3=y30;
coordiniz=[x5,y5,x4,y4,x6,y6,x7,y7];
syms s
%point 7 objective path
xd=0;
```

```
yd=350.*s;
s=0; yd=eval(yd);
% initial lenghts
l5=sqrt((x5-x10).^2 + (y5-y10).^2);
l4=sqrt((x5-x4).^2 + (y5-y4).^2);
l2=sqrt((x4-x20).^2 + (y4-y20).^2);
l1=sqrt((x7-x20).^2 + (y7-y20).^2);
l3=sqrt((x4-x7).^2 + (y4-y7).^2);
 a0=(l4.*((x5-x4).*(x6-x5)+(y5-y6).*(y4-y5))./((y4-y5).^2.+(x4-x5).^2));
b0=(l4.*((y6-y5).*(x4-x5)-(y4-y5).*(x6-x5))./((y4-y5).^2.+(x4-x5).^2));
a=-a0; b=b0;
%% CALCULATE INITIAL VALUE OF t0
 % constraint equation
 syms x5 y5 x4 y4 x6 y6 x7 y7;
 psi1=(x5-x1).^2.+(y5-y1).^2.-l5.^2;              %dL/dlambda1
 psi2=(x4-x2).^2.+(y4-y2).^2.-l2.^2;              %dL/dlambda2
 psi3=(x5-x4).^2.+(y5-y4).^2.-l4.^2;              %dL/dlambda3
 psi4=x6-x5-a.*(x4-x5)./l4+b.*(y4-y5)./l4;        %dL/dlambda4
 psi5=y6-y5-a.*(y4-y5)./l4-b.*(x4-x5)./l4;        %dL/dlambda5
 psi6=(x7-x2).^2 + (y7-y2).^2-l1.^2;              %dL/dlambda6
 psi7=(x4-x7).^2 + (y4-y7).^2-l3.^2;              %dL/dlambda7
 psi1=-psi1; psi2=-psi2; psi3=-psi3; psi6=-psi6; psi7=-psi7;
% point 7 penalty
 pen=(x7-xd).^2.+(y7-yd).^2;
% lagrangian
 syms lambda1 lambda2 lambda3 lambda4 lambda5 lambda6 lambda7;
lagr=pen+lambda1.*psi1+lambda2.*psi2+lambda3.*psi3+lambda4.*psi4+lambda5.
*psi5+lambda6.*psi6+lambda7.*psi7;
 dlx5=diff(lagr, x5);                             %dL/dx5
 dly5=diff(lagr, y5);                             %dL/dy5
 dlx4=diff(lagr, x4);                             %dL/dx4
 dly4=diff(lagr, y4);                             %dL/dy4
 dlx6=diff(lagr, x6);                             %dL/dx6
 dly6=diff(lagr, y6);                             %dL/dy6
 dlx7=diff(lagr, x7);                             %dL/dx7
 dly7=diff(lagr, y7);                             %dL/dy7
x5=n(1);
y5=n(2);
x4=n(3);
y4=n(4);
x6=n(5);
y6=n(6);
x7=n(7);
y7=n(8);
%spatial derivative lagrangian evaluation
dlx5k=eval(dlx5); dly5k=eval(dly5);
dlx4k=eval(dlx4); dly4k=eval(dly4);
dlx6k=eval(dlx6); dly6k=eval(dly6);
dlx7k=eval(dlx7); dly7k=eval(dly7);
%index of performance evaluation
t=dlx5k.^2+dly5k.^2+dlx4k.^2+dly4k.^2+dlx6k.^2+dly6k.^2+dlx7k.^2+dly7k.^2
;
%% CALCULATE INITIAL LAGRANGE MULTIPLIERS
[H, c]=equationsToMatrix([dlx5k==0, dly5k==0, dlx4k==0, dly4k==0,
dlx6k==0, dly6k==0, dlx7k==0, dly7k==0],...
    [lambda1, lambda2, lambda3, lambda4, lambda5, lambda6, lambda7]);
mltpliniz=eval(H)\eval(c);
mltpliniz=mltpliniz';
```

```
valinizivp=[coordiniz,mltpliniz];
lambda1=mltpliniz(1);
lambda2=mltpliniz(2);
lambda3=mltpliniz(3);
lambda4=mltpliniz(4);
lambda5=mltpliniz(5);
lambda6=mltpliniz(6);
lambda7=mltpliniz(7);
t0=eval(t);
valinizivp=[valinizivp,t0];
%% KINEMATIC ANALYSIS
options=odeset('Events',@odeStopEvent);
kk=0:0.02:1; % integration interval of variable s
  sistma=@(s,m) systm(s,m,l5,l4,l2,l1,l3,a,b,x10,y10,x20,y20,x60,y60,t0);
  [T,Y,te,ye,ie]=ode45(sistma,kk,valinizivp,options);
Ywh=Y(:,8); %consider only mechanism move with y7<100
Ywheelreal=Ywh(2:end);
%% GRAPHICS
%tau trend in graphic form
    XX6=abs(diff(Y(:,5))); YY6=abs(diff(Y(:,6))); YY7=diff(Y(:,8));
    XY6=sqrt(XX6.^2+YY6.^2);
    Ytau_r=[];
        for is=1:1:size(XX6)
            tau_r=XY6(is)./YY7(is);
            Ytau_r=[Ytau_r;tau_r];
        end
    Xtau=[0;10;20;30;40;50;60;70;80;90;100];

        if coordiniz-coordiniziali==zeros(1,8)
            global Ytau_g Ywheelrealg pol
            if input_tau(12)==1
                pol=polyfit(Xtau,input_tau(1:11),1);
            else
                pol=polyfit(Xtau,input_tau(1:11),2);
            end
            Ytau_g=Ytau_r;
            Ywheelrealg=Ywheelreal;
        end
    global Ytau_g Ywheelrealg pol
            if input_tau(12)==1
                 Ytau_t=pol(1).*Ywheelreal+pol(2);
            else
                Ytau_t=pol(1).*(Ywheelreal).^2+pol(2).*Ywheelreal+pol(3);
            end
    figure(100)
    figure(100), set(100,'position', [200, 50, 1000, 600]), clf
    subplot(2,2,2)
    error_tau=trapz((Ytau_r-Ytau_t).^2); %calculate error
    plot(Ywheelreal,Ytau_r,'r'),title(['tau error =
',num2str(error_tau)])
    hold on
    plot(Ywheelreal,Ytau_t,'color',[0 0.6 0])
    plot(Ywheelrealg,Ytau_g,'b')
    hold off
    xlabel('escursione verticale della ruota (mm)')
    ylabel('rapporto di velocità')
    xlim([0 105])
        fminsearchtau=Ytau_r;
```

```
        fminsearchYwh=Ywheelreal;
        save('fminsearchtau.mat','fminsearchtau');
        save('fminsearchYwh.mat','fminsearchYwh');
%mechanism
    if true
        subplot(2,2,1)
        for is=1:5:size(Y,1)
            hold on
            line([Y(is,1),Y(is,5)],[Y(is,2),Y(is,6)])
            line([Y(is,3),Y(is,5)],[Y(is,4),Y(is,6)])
            line([Y(is,1),Y(is,3)],[Y(is,2),Y(is,4)])
            line([x10,Y(is,1)],[y10,Y(is,2)])
            line([Y(is,3),x20],[Y(is,4),y20])
            line([Y(is,3),Y(is,7)],[Y(is,4),Y(is,8)])
            line([x20,Y(is,7)],[y20,Y(is,8)])
            line([x30,Y(is,5)],[y30,Y(is,6)],'Color','k')
            plot(Y(is,5),Y(is,6),'r.') ,title(['meccanismo con rigidezza
k = ',num2str(k_spring),' N/mm (ottimizzazione FMINSEARCH)'])   %point
out (x6,y6)
            axis equal
            xlabel('x (mm)')
            ylabel('y (mm)')
            hold off
        end
    end
            guess=coordiniziali';
            optim=n';
            indice={'x5';'y5';'x4';'y4';'x6';'y6';'x7';'y7'};
            fminsearchRes=table(guess,optim,'RowNames',indice);
            save('fminsearchRes.mat','fminsearchRes');
% progressiveness suspension reduced stiffness graphic
    subplot(2,2,4)
    plot(Ywheelreal,k_spring.*(Ytau_r.^2),'c')
    xlabel('escursione verticale della ruota (mm)')
    ylabel('rigidezza ridotta (N/mm)')
    xlim([0 105])
%progressiveness suspension elastic force graphic
    X6i=Y(1,5); X6i=X6i.*ones(size(Y(:,5)));
    Y6i=Y(1,6); Y6i=Y6i.*ones(size(Y(:,5)));
    XY6=sqrt((Y(:,5)-X6i).^2+(Y(:,6)-Y6i).^2);
    subplot(2,2,3)
    plot(Y(:,8),XY6.*k_spring,'color',[0.7 0.1 0.8])
    xlabel('escursione verticale della ruota (mm)')
    ylabel('forza elastica (N)')
    xlim([0 105])
f=error_tau
```

- ***Function "systm.mat"***

```
function dydt=systm(s,m,l5,l4,l2,l1,l3,a,b,x10,y10,x20,y20,x60,y60,t0)
%GENERATION OF ODE SYSTEM FROM MATRIXES
x5=m(1);   %specific natural coordinates,lagrange multipliers and t0 as
variables
y5=m(2);
```

```
x4=m(3);
y4=m(4);
x6=m(5);
y6=m(6);
x7=m(7);
y7=m(8);
lambda1=m(9);
lambda2=m(10);
lambda3=m(11);
lambda4=m(12);
lambda5=m(13);
lambda6=m(14);
lambda7=m(15);
t0=m(16);
[A,B,C]=genmatrix(s,m,l5,l4,l2,l1,l3,a,b,x10,y10,x20,y20,x60,y60,t0);
mun=0.1; %time constant
Ai=inv(A);
syst=Ai*(B+mun*C);
dydt(1)=syst(1);  %differential equations vector
dydt(2)=syst(2);
dydt(3)=syst(3);
dydt(4)=syst(4);
dydt(5)=syst(5);
dydt(6)=syst(6);
dydt(7)=syst(7);
dydt(8)=syst(8);
dydt(9)=syst(9);
dydt(10)=syst(10);
dydt(11)=syst(11);
dydt(12)=syst(12);
dydt(13)=syst(13);
dydt(14)=syst(14);
dydt(15)=syst(15);
dydt(16)=syst(16);
dydt=[dydt(1); dydt(2); dydt(3); dydt(4); dydt(5); dydt(6); dydt(7);
dydt(8); dydt(9); dydt(10); dydt(11); dydt(12); dydt(13); dydt(14);
dydt(15); dydt(16)];
```

- ***Function "genmatrix.mat"***

```
function
[A,B,C]=genmatrix(s,m,l5,l4,l2,l1,l3,a,b,x10,y10,x20,y20,x60,y60,t0)
x1=x10; y1=y10; x2=x20; y2=y20;
mun=0.1;
x5=m(1);
y5=m(2);
x4=m(3);
y4=m(4);
x6=m(5);
y6=m(6);
x7=m(7);
y7=m(8);
lambda1=m(9);
lambda2=m(10);
```

```
lambda3=m(11);
lambda4=m(12);
lambda5=m(13);
lambda6=m(14);
lambda7=m(15);
syms x5 y5 x4 y4 x6 y6 x7 y7 lambda1 lambda2 lambda3 lambda4 lambda5
lambda6 lambda7 k
%point 7 objective path
xd=0;
yd=350.*k;
xd1=0;
yd1=350;
%point 6 objective path
x6d=x60;
y6d=y60+48.*k;
pen2=(x6-x6d).^2.+(y6-y6d).^2;
%% CALCULATE MATRIXES
 psi1=(x5-x1).^2.+(y5-y1).^2.-l5.^2;                %dL/dlambda1
 psi2=(x4-x2).^2.+(y4-y2).^2.-l2.^2;                %dL/dlambda2
 psi3=(x5-x4).^2.+(y5-y4).^2.-l4.^2;                %dL/dlambda3
 psi4=x6-x5-a.*(x4-x5)./l4+b.*(y4-y5)./l4;          %dL/dlambda4
 psi5=y6-y5-a.*(y4-y5)./l4-b.*(x4-x5)./l4;          %dL/dlambda5
 psi6=(x7-x2).^2 + (y7-y2).^2-l1.^2;                %dL/dlambda6
 psi7=(x4-x7).^2 + (y4-y7).^2-l3.^2;                %dL/dlambda7
psi4=-psi4; psi5=-psi5;
pen=(x7-xd).^2.+(y7-yd).^2;
lagr=pen+lambda1.*psi1+lambda2.*psi2+lambda3.*psi3+lambda4.*psi4+lambda5.
*psi5+lambda6.*psi6+lambda7.*psi7;
 dlx5=diff(lagr, x5);
 dly5=diff(lagr, y5);
 dlx4=diff(lagr, x4);
 dly4=diff(lagr, y4);
 dlx6=diff(lagr, x6);
 dly6=diff(lagr, y6);
 dlx7=diff(lagr, x7);
 dly7=diff(lagr, y7);
vectvar=[x5; y5; x4; y4; x6; y6; x7; y7; lambda1; lambda2; lambda3;
lambda4; lambda5; lambda6; lambda7];
lagr=-lagr;
A=hessian(lagr,vectvar);
A=[A, zeros(15,1)]; A=[A; 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1];
C=[dlx5;dly5;dlx4;dly4;dlx6;dly6;dlx7;dly7;psi1;psi2;psi3;-psi4;-
psi5;psi6;psi7];
C=[C;0];
 C=-C;
B=[0;0;0;0; -2.*xd1; -2.*yd1;0;0;0;0;0;0;0;0;0;pen2];
%%
x5=m(1);
y5=m(2);
x4=m(3);
y4=m(4);
x6=m(5);
y6=m(6);
x7=m(7);
y7=m(8);
lambda1=m(9);
lambda2=m(10);
lambda3=m(11);
```

```
lambda4=m(12);
lambda5=m(13);
lambda6=m(14);
lambda7=m(15);
x1=x10; y1=y10; x2=x20; y2=y20;
k=s;
A=eval(A); B=eval(B); C=eval(C);
```

- ***Function "OdeStopEvent.mat"***

```
function [value,isterminal,direction] = odeStopEvent(T,Y)
value=1;
if Y(8)>100
    value=0;
end
isterminal = 1;  % Halt integration
direction = 0;   % The zero can be approached from either direction
```

- ***Function "StopOptim12.mat"***

```
function stop = stopoptim12(x, optimValues, state,errmaxtau)
stop = false;
% Check if objective function is less than errmaxtau.
if optimValues.fval < errmaxtau
    stop = true;
end
```

BIBLIOGRAFIA

[1] V. Cossalter, *"Motorcycle dynamics",* Ed. italiana, Padova, 2008

[2] W. F. Milliken, D. L. Milliken, "Race Car Veichle Dynamics", SAE, 1995.

[3] I. Birò, *"Synthesis of some mechanisms using natural coordinates"*, Interdisciplinary Description of Complex Systems 12(3), 255-262, 2014

[4] V. Cossalter, M. Da Lio, R. Lot, *"Sull'uso delle coordinate naturali nella sintesi di meccanismi con metodi di ottimizzazione"*, XIII Congresso Nazionale Aimeta, Siena, 1997

[5] V. Lorenzi, R. Riva, M. Camposaragna, R. Strada, B. Zappa, *"Metodi numerici per lo studio di sistemi multicorpo",* Bergamo, dispense del corso

[6] V. Cossalter, M. Da Lio, R. Lot, *"On the use of natural coordinates in optimal synthesis of mechanisms",* Mechanism and Machine Theory 35, 1367-1389, 2000.

[7] J.C. de Jalòn, E. Bayo, *"Kinematic and dynamic simulation of Multibody Systems",* Springer-Verlag, New York, 1994

[8] P. Marchand, O. T. Holland, *"Graphics and GUIs with Matlab",* Chapman & Hall/CRC, Washington D.C., 2003

[9] Desmond J. Higham, Nicholas J. Higham, "Matlab Guide", Second Edition, 2005.

[10] D. Prandin, *"Ottimizzazione cinematica di sospensioni motociclistiche",* Tesi di laurea, Padova, 2013