**Indice**

1. Il quesito 2

2. Considerazioni generali 2

3. Gradi di libertà residui del quadrilatero articolato 5

4. Analisi cinematica del primo ordine con le coordinate assolute 5

5. Analisi cinematica del secondo ordine con le coordinate assolute 7

6. Algoritmo risolutivo 8

7. Esecuzione del programma per coordinate assolute 9

8. Analisi cinematica del primo ordine per coordinate naturali 11

9. Analisi cinematica del secondo ordine con coordinate naturali 12

10. Algoritmo risolutivo 13

11. Esecuzione del programma per coordinate naturali 13

12. Metodo dei diagrammi polari 15

13. Codice principale per la soluzione con coordinate assolute 19

14. Modulo per la soluzione con coordinate assolute 23

15. Codice principale per la soluzione con coordinate naturali 27

16. Modulo per la soluzione con coordinate naturali 30

**Quadrilatero articolato, cinematica**

**1. Il quesito.** E’ assegnato il quadrilatero articolato le cui aste sono definite dalle seguenti misure

1.1)

Sia inoltre assegnata la rotazione della manovella, la sua velocità angolare e la sua accelerazione angolare, secondo quanto segue

1.2)

Si richiede di effettuare l’analisi cinematica del meccanismo attraverso il metodo delle equazioni di vincolo, e in particolare di ricavare

1.3) velocità angolare e accelerazione angolare delle aste 3,4;

1.4) velocità e accelerazione dei punti di mezzeria , .

**2. Considerazioni generali.** In questo paragrafo richiamo la teoria sull’analisi cinematica dei meccanismi attraverso l’uso delle equazioni di vincolo. Dato un meccanismo caratterizzato dai seguenti parametri

2.1)

siano

2.2)

le **coordinate lagrangiane sovrabbondanti**, e siano

2.3)

le **equazioni di vincolo**. E’ opportuno ricondursi a equazioni di vincolo che siano **indipendenti** fra loro, ovvero tali che ognuna di esse imponga una condizione diversa alle coordinate lagrangiane; la condizione di indipendenza fra le equazioni di vincolo coincide con la condizione di rango massimo per la matrice

2.4)

Infatti la riga *i-ma* della matrice **2.4** è costituita dalle componenti di un vettore normale alla superficie dello spazio n-dimensionale, nel punto ; dunque imporre il rango massimo alla matrice **2.4** coincide con la condizione che le *p* superfici **2.3** abbiano tutte diverso piano tangente nel punto .

Ciò posto si può stabilire per i parametri in **2.1** la seguente relazione

2.5)

Infatti il numero di gradi di libertà residui (che possono essere calcolati ad esempio con la formula di Kutzbach) sono una proprietà del meccanismo, indipendente dalla scelta delle coordinate

lagrangiane sovrabbondanti; se poi abbiamo *p* equazioni di vincolo indipendenti fra loro, allora posso esprimere (a meno di configurazioni critiche) *p* coordinate in funzione delle altre *n-p*, ovvero ottengo un meccanismo con *n-p* gradi di libertà; ma dovendo essere *F* il numero di gradi di libertà, si trova la **2.5**.

Si effettua allora una **partizione delle coordinate** lagrangiane sovrabbondanti, evidenziando coordinate che si assumono come indipendenti -che indichiamo - mentre le restanti coordinate sono indicate . Si ottiene così il vettore

2.6)

Ebbene il **teorema di Ulisse Dini** (o delle funzioni implicite) permette di affermare che il sistema **2.3** può essere esplicitato rispetto alle coordinate nel punto , qualora risulti non nullo il determinante della matrice quadrata

2.7)

nel punto stesso. Nella pratica poi, ottenuta la garanzia da parte del teorema di Dini della possibilità di esplicitare le **2.3** rispetto le , si deve spesso ricorrere a **metodi numerici** per pervenire al risultato, essendo le equazioni **2.3** potenzialmente equazioni trascendenti. Uno di questi metodi è quello di Newton-Raphson la cui *i-ma* iterata si scrive

2.8)

Per l’**analisi cinematica del primo ordine** si derivano nel tempo le equazioni di vincolo **2.3** ottenendo

2.9)

Utilizzando la partizione delle coordinate abbiamo anche

2.10)

che in forma matriciale si scrive

2.11)

Introdotta allora la matrice

2.12)

la **2.11** si scrive

2.13)

da cui si ricavano le derivate prime delle variabili dipendenti

2.14)

Per l’**analisi cinematica del secondo ordine** si procede derivando nel tempo la **2.13** ricordando che per la derivata del prodotto di matrici valgono le stesse regole note per il prodotto di funzioni (la dimostrazione è immediata); si ha

Esplicitando rispetto al vettore delle derivate seconde delle variabili dipendenti si conclude quindi

2.15)

**3. Gradi di libertà residui del quadrilatero articolato.** Prima di procedere con l’analisi cinematica del meccanismo in parola dimostro che i gradi di libertà residui sono pari a uno. A tale scopo si osserva che l’applicazione della **formula di Grübler** nel piano porge

3.1)

dove si ricorda che è il numero di gradi di libertà del corpo non vincolato (6 nello spazio e 3 nel piano), è il numero di membri, è il numero di coppie inferiori (da due gradi di vincolo ciascuna) e è quello di coppie superiori (da un grado di vincolo). Detto allora il numero di coordinate sovrabbondanti, il numero di equazioni di vincolo indipendenti sarà necessariamente

3.2)

**4. Analisi cinematica del primo ordine con le coordinate assolute.** Il vettore delle coordinate lagrangiane sovrabbondanti è

4.1)

il quale viene sottoposto alla partizione

4.2)

Essendo , la **3.2** predice che le equazioni di vincolo indipendenti sono . Le equazioni di vincolo, immediatamente deducibili dalla figura sono

4.3)

Lo jacobiano relativo alle coordinate dipendenti si scrive

4.4)

e a conti fatti si ha

4.5)

Per avere il vettore delle coordinate dipendenti in funzione della coordinata indipendente assegnata si applica il metodo numerico **2.8** il quale nel caso presente si scrive

4.6)

Per l’analisi cinematica del primo ordine, osservato che la matrice **2.12** si scrive nel presente caso

4.7)

la espressione **2.14** porge

4.8)

**5. Analisi cinematica del secondo ordine con le coordinate assolute.** Per l’analisi cinematica del secondo ordine è necessario derivare i jacobiani **4.5**, **4.7** ottenendo le matrici

5.1)

5.2)

Sostituendo le **5.1**, **5.2** nella **2.15** abbiamo in fine

5.3)

Si osserva che il quesito **1.3** trova risposta direttamente attraverso le **4.8**, **5.3** che forniscono in particolare .

**6. Algoritmo risolutivo.** Per effettuare l’analisi cinematica nel caso delle coordinate assolute sono state scritte in Fortran le seguenti unità:

* l’unità chiamante *main\_ese\_13\_a*, la quale si occupa del calcolo delle derivate prime e seconde delle variabili dipendenti;
* il modulo *mod\_ese\_13\_a*, il quale contiene le costanti geometriche **1.1**, **1.2** del problema, oltre alle seguenti procedure di modulo:
* subroutine *inverse* (scaricata dal sito della ODU University) la quale si occupa di invertire la matrice jacobiana **4.5**;
* subroutine *parallelogramma* la quale permette di rappresentare nel piano la configurazione del parallelogramma alla generica iterazione del metodo Newton-Raphson;
* subroutine *diagramma\_resto* la quale esegue il grafico della norma del vettore resto in funzione del numero di iterazione.

Il diagramma di flusso complessivo è indicato in figura, dove non si entra nel dettaglio della applicazione del metodo Newton-Raphson per il calcolo della configurazione congruente, essendo questa parte già stata discussa ampiamente in altra sede.

Inizio

Calcola una delle due configurazioni congruenti con l’angolo di manovella assegnato; quale delle due dipende dall’*initial guess* imposta. Per effettuare questo calcolo sia avvale del metodo numerico di Newton-Raphson.

Calcola la matrice jacobiana della variabile indipendente.

Calcola le derivate prime delle variabili dipendenti attraverso la **2.14**.

Calcola le derivate seconde delle variabili dipendenti attraverso la **2.15**.

Calcola le funzioni cinematiche dei punti

fine

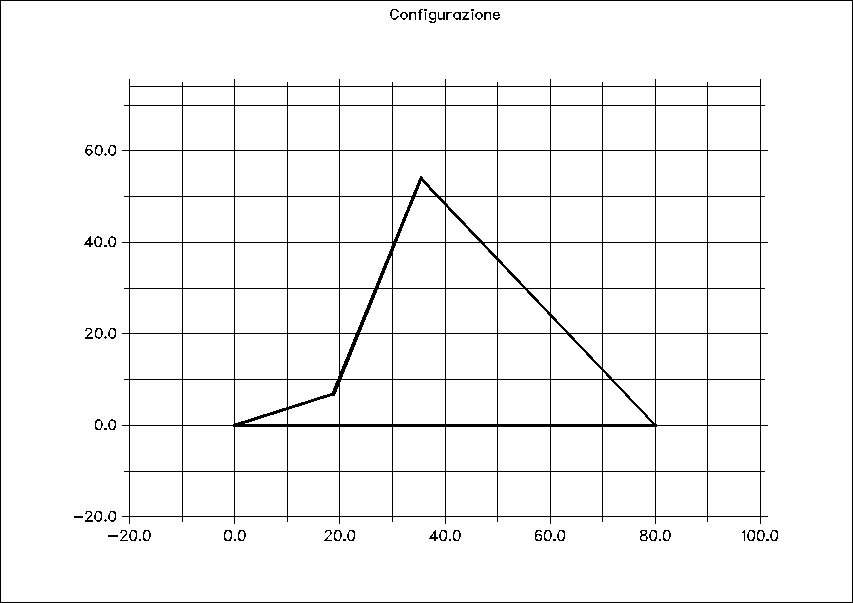
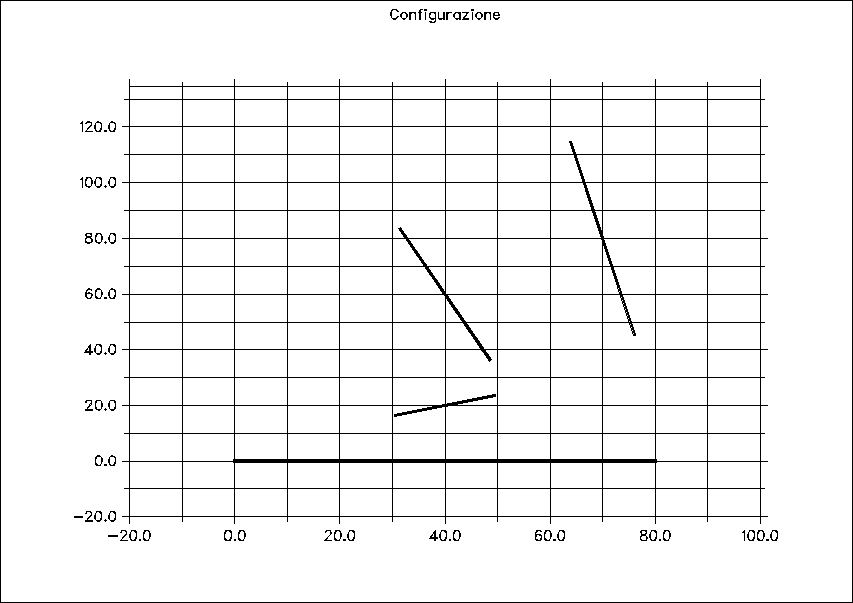
**7. Esecuzione del programma per le coordinate assolute.** Il programma utilizza come *initial guess* la seguente configurazione

7.1)

e perviene alla configurazione congruente con l’angolo di manovella, data da

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
| 9.396926 | 3.4202013 | 70.55076 | 27.118143 | 30.413828 | -50.46595 | 57.72122 | 26.993628 |

Riporto la rappresentazione della configurazione assegnata come *initial guess* e di quella congruente ottenuta:



Riporto direttamente dal Prompt l’output dell’esecuzione del programma che segue l’analisi cinematica attraverso le coordinate assolute.

La velocita' angolare (rad/s) dell'asta 3 e' -18.425283

La velocita' angolare (rad/s) dell'asta 4 e' -10.783358

La accelerazione angolare (rad/s^2) dell'asta 3 e' -259.65527

La accelerazione angolare (rad/s^2) dell'asta 4 e' 584.71075

La velocita' del punto M3 ha modulo 650.86505

Le sue componenti x,y sono 147.81686 633.8576

La velocita' del punto M4 ha modulo 377.41754

Le sue componenti x,y sono 291.08194 240.24008

La accelerazione del punto M3 ha modulo 37044.555

Le sue componenti x,y sono -29680.682 -22166.557

La accelerazione del punto M4 ha modulo 20865.635

Le sue componenti x,y sono -13192.869 -16165.484

Dunque le risposte ai quesiti **1.3**, **1.4** sono rispettivamente

7.2)

7.3)

**8. Analisi cinematica del primo ordine con le coordinate naturali.** Per coordinate naturali si intendono quelle delle cerniere , B. In questo caso allora il vettore delle coordinate lagrangiane sovrabbondanti è dato da

8.1)

il quale viene sottoposto alla partizione

8.2)

Essendo , la **3.2** predice che le equazioni di vincolo indipendenti sono . Le equazioni di vincolo, immediatamente deducibili dalla figura sono

8.3)

Lo jacobiano relativo alle coordinate dipendenti si scrive

8.4)

Per avere il vettore delle coordinate dipendenti in funzione della coordinata indipendente assegnata si applica il metodo numerico **2.8** il quale nel caso presente si scrive

8.5)

Per l’analisi cinematica del primo ordine, osservato che la matrice **2.12** si scrive nel presente caso

8.6)

la espressione **2.14** porge

8.7)

**9. Analisi cinematica del secondo ordine con le coordinate naturali.** Per l’analisi cinematica del secondo ordine è necessario derivare i jacobiani **8.4**, **8.6** ottenendo le matrici

9.1)

9.2)

Sostituendo le **9.1**, **9.2** nella **2.15** abbiamo in fine

9.3)

Si osserva che se la variabile indipendente è , essendo assegnate velocità, accelerazione e valore di , sarà necessario considerare che

9.4)

Per ottenere la risposta al quesito **1.4** si osserva che

9.5)

e quindi derivando si ha

9.6)

9.7)

Per quanto concerne il quesito **1.3** si consideri invece che

Confrontando quindi con le **9.6** abbiamo trovato che

9.8)

da cui, in particolare, segue

9.9)

Sostituendo le **9.5** nelle **9.9** si ha

9.10)

Derivando ulteriormente si ha

9.11)

**10. Algoritmo risolutivo.** Per effettuare l’analisi cinematica nel caso delle coordinate naturali si procede come nel caso delle coordinate assolute. Tuttavia utilizzando la configurazione congruente già calcolata precedentemente si può saltare la parte della ricerca della configurazione. In definitiva sono state scritte in Fortran le seguenti unità:

* l’unità chiamante *main\_ese\_13\_b*, la quale si occupa del calcolo delle derivate prime e seconde delle variabili dipendenti;
* il modulo *mod\_ese\_13\_b*, il quale contiene le costanti geometriche **1.1**, **1.2** del problema, oltre alle seguenti procedure di modulo:
* subroutine *inverse* (scaricata dal sito della ODU University) la quale si occupa di invertire la matrice jacobiana **7.4**;
* subroutine *parallelogramma* la quale permette di rappresentare nel piano la configurazione del parallelogramma alla generica iterazione del metodo Newton-Raphson.

**11. Esecuzione del programma per le coordinate naturali.** Per assegnare la configurazione iniziale si considerino le seguenti trasformazioni dalle coordinate assolute alle coordinate naturali:

11.1)

Quindi si usi la configurazione congruente ricavata con la esecuzione precedente (vedi paragrafo 6), ovvero

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
| 9.396926 | 3.4202013 | 70.55076 | 27.118143 | 30.413828 | -50.46595 | 57.72122 | 26.993628 |

che va sostituita nelle **11.1**. Allora il programma *main\_ese13\_b* fornisce i seguenti risulatati, presi direttamente dal *prompt:*

La velocita' angolare (rad/s) dell'asta 3 e' -18.425283

La velocita' angolare (rad/s) dell'asta 4 e' -10.783358

La accelerazione angolare (rad/s^2) dell'asta 3 e' -259.6553

La accelerazione angolare (rad/s^2) dell'asta 4 e' 584.7109

La velocita' del punto M3 ha modulo 6.5086503

Le sue componenti x,y sono 1.4781685 6.3385763

La velocita' del punto M4 ha modulo 3.7741752

Le sue componenti x,y sono 2.910819 2.402401

La accelerazione del punto M3 ha modulo 370.4456

Le sue componenti x,y sono -296.80682 -221.66559

La accelerazione del punto M4 ha modulo 208.65636

Le sue componenti x,y sono -131.9287 -161.65486

Ovvero si ottengono i risultati seguenti, coincidenti con quelli calcolati utilizzando le coordinate assolute e riportati in **8.1**, **3.2**.

11.2)

11.3)

**12. Metodo dei diagrammi polari.** Per applicare questo metodo si determina la configurazione del meccanismo per via geometrica. Si consideri poi la equazione vettoriale

12.1)

per la quale si costruisce la tabella

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

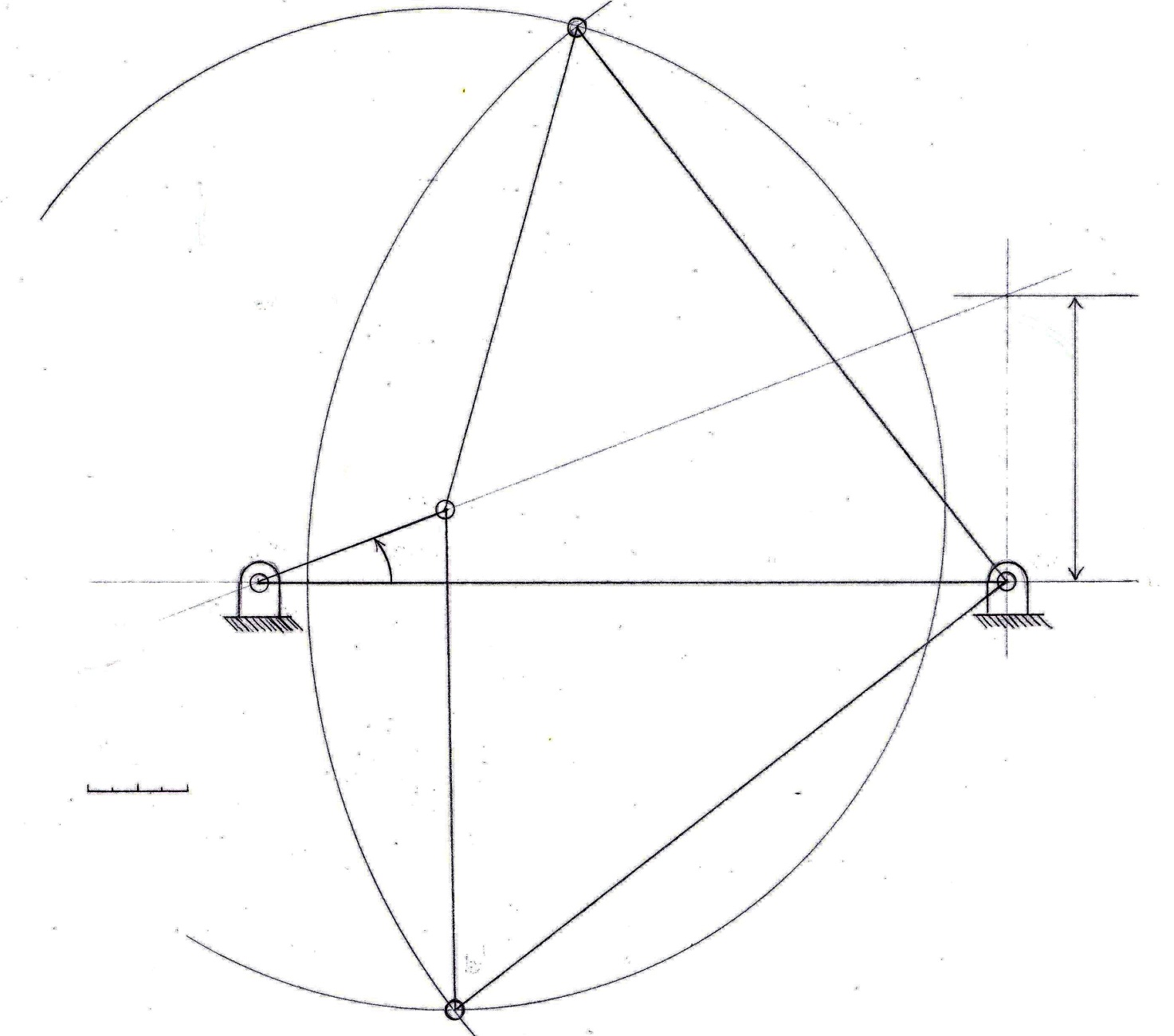
da cui si verifica la presenza di due sole incognite scalari; quindi l’equazione è risolvibile. Per il vettore noto si ha

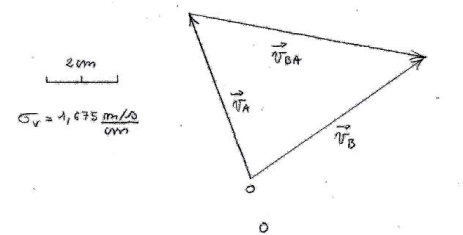
12.3)

Adottando una scala di rappresentazione per le velocità data da

12.4)

si perviene al diagramma polare indicato in figura.



Dalla lettura del diagramma polare si ricava

12.5)

da cui le velocità angolari delle aste 3,4:

12.6)

Le velocità dei punti di mezzeria si ricavano -in modulo- nel modo seguente

12.7)

Per quanto riguarda l’analisi cinematica del secondo ordine si consideri che per l’asta 3 si può scrivere l’equazione

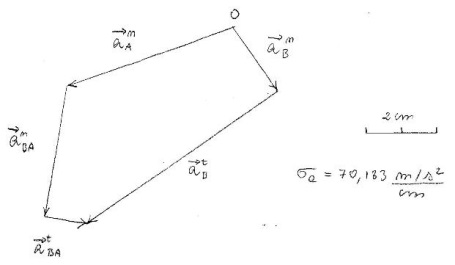
12.8)

che si traduce nella tabella seguente

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| ? |  |  | ? |  |
|  |  |  |  |  |

Per i moduli noti delle accelerazioni si ha

12.9)



Fissando la scala di rappresentazione

12.10)

si ottiene il diagramma polare in figura, da cui si ricava

12.11)

e quindi

12.12)

Per l’accelerazione del punto si ha

12.13)

Per l’accelerazione del punto si ha

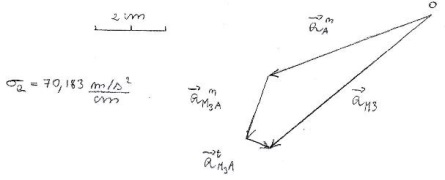
|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| ? |  |  |  |
|  |  |  |  |

Per i moduli noti delle accelerazioni si ha

12.14)

Utilizzando la scala di rappresentazione **11.10** si costruisce il diagramma polare in figura, da cui si deduce che

12.15)



**13. Codice principale per la soluzione con coordinate assolute.** Questo codice rappresenta il programma principale che calcola i risultati **7.1**, **7.3**.

PROGRAM main\_ese\_13\_a

USE DISLIN !libreria grafica

USE mod\_ese\_13\_a

!sezione dichiarativa

IMPLICIT NONE

!dichiaro il vettore delle coordinate dipendenti

REAL,DIMENSION(8,100)::u

!dichiaro il vettore delle velocità delle coordinate dipendenti

REAL,DIMENSION(8)::u\_p

!dichiaro il vettore delle accelerazioni delle coordinate dipendenti

REAL,DIMENSION(8)::u\_p\_p

!dichiaro dei vettori che servono al calcolo di u\_p\_p

REAL,DIMENSION(8,8)::u\_p\_p\_1

REAL,DIMENSION(8)::u\_p\_p\_2,u\_p\_p\_3

!dichiaro la funzione delle equazioni di vincolo

REAL,DIMENSION(8,100)::psi

!dichiaro la matrice jacobiana delle u, la sua derivata e la sua inversa

REAL,DIMENSION (8,8,100)::J

REAL,DIMENSION (8,8)::j\_p

REAL(KIND=8),DIMENSION (8,8)::invj

!dichiaro la matrice jacobiana delle v e la sua derivata

REAL,DIMENSION (8)::j\_v

REAL,DIMENSION (8)::j\_v\_p

!dichiaro il vettore dei resti

REAL,DIMENSION(8,100)::resto

!dichiaro la norma del vettore dei resti

REAL,DIMENSION(100):: norma

!dichiaro i valori di innesco delle coordinate

REAL::x2 = 40.

REAL::y2 = 20.

REAL::theta3 = 110\*(3.1415927/180)

REAL::x3 = 40.

REAL::y3 = 60.

REAL::theta4= -80.\*(3.1415927/180)

REAL::x4= 70.

REAL::y4= 80.

!dichiaro le derivate prime e seconde delle coordinate

REAL::x2\_p, x2\_p\_p

REAL::y2\_p, y2\_p\_p

REAL::theta3\_p, theta3\_p\_p

REAL::x3\_p, x3\_p\_p

REAL::y3\_p, y3\_p\_p

REAL::theta4\_p, theta4\_p\_p

REAL::x4\_p, x4\_p\_p

REAL::y4\_p, y4\_p\_p

!dichiaro le funzioni cinematiche relative ai punti M3, M4

REAL::vxM3, vyM3, vM3

REAL::vxM4, vyM4, vM4

REAL::axM3, ayM3, aM3

REAL::axM4, ayM4, aM4

!altre variabili di lavoro

REAL(KIND=8),DIMENSION(8,8)::matrix

INTEGER::i !indici dei cicli

CHARACTER(len=10):: chiusura !serve per chiudere il programma

!sezione esecutiva

!fisso risoluzione dell'immagine in formato .bmp

CALL BMPMOD (300,'inch','resolution')

!inizializzo i valori d'innesco di u

u(1,1) = x2

u(2,1) = y2

u(3,1) = theta3

u(4,1) = x3

u(5,1) = y3

u(6,1) = theta4

u(7,1) = x4

u(8,1) = y4

!inizializzo la matrice jacobiana con i valori costanti

j(:,:,:)=0.

j(1,1,:) = 1.

j(2,2,:) = 1.

j(3,1,:) = -1.

j(3,4,:) = 1.

j(4,2,:) = -1.

j(4,5,:) = 1.

j(5,4,:) = -1.

j(5,7,:) = 1.

j(6,5,:) = -1.

j(6,8,:) = 1.

j(7,7,:) = 1.

j(8,8,:) = 1.

!inizializzo il vettore resto

resto=0.

!ciclo del metodo Newton-Raphson

i=0.

ciclo: DO

i=i+1

x2 = u(1,i)

y2 = u(2,i)

theta3 = u(3,i)

x3 = u(4,i)

y3 = u(5,i)

theta4 = u(6,i)

x4 = u(7,i)

y4 = u(8,i)

!calcolo la matrice jacobiana

j(3,3,i) = (0.5)\*l3\*SIN(theta3)

j(4,3,i) = -(0.5)\*l3\*COS(theta3)

j(5,3,i) = (0.5)\*l3\*SIN(theta3)

j(5,6,i) = (0.5)\*l4\*SIN(theta4)

j(6,3,i) = -(0.5)\*l3\*cos(theta3)

j(6,6,i) = -(0.5)\*l4\*cos(theta4)

j(7,6,i) = -(0.5)\*l4\*sin(theta4)

j(8,6,i) = (0.5)\*l4\*cos(theta4)

!calcolo l'inversa della matrice jacobiana

matrix(:,:)=j(:,:,i)

CALL inverse(matrix,invj,8)

!calcolo la funzione psi

psi(1,i) = x2 - ((0.5)\*l2\*COS(theta2))

psi(2,i) = y2 - (0.5)\*l2\*SIN(theta2)

psi(3,i) = x3 -(l3\*0.5)\*COS(theta3) - x2 - (l2\*0.5)\*COS(theta2)

psi(4,i) = y3 -(l3/2.)\*SIN(theta3) - y2 - (l2\*0.5)\*SIN(theta2)

psi(5,i) = x4 -x3 -(0.5)\*l3\*COS(theta3) -(0.5)\*l4\*COS(theta4)

psi(6,i) = y4 -y3 -(0.5)\*l3\*SIN(theta3) -(0.5)\*l4\*SIN(theta4)

psi(7,i) = x4 - l1 + (0.5)\*l4\*COS(theta4)

psi(8,i) = y4 + (0.5)\*l4\*sin(theta4)

!calcola il vettore resto

resto(:,i)=MATMUL(invj,psi(:,i))

!calcola la norma del resto

norma(i)=SQRT(DOT\_PRODUCT(resto(:,i),resto(:,i)))

!disegno il parallelogramma alla iterata i

CALL parallelogramma(x2,y2,x3,y3,theta3,x4,y4,theta4,i)

!verifico il valore della norma e in caso esco

IF (norma(i)<=0.01) EXIT ciclo

!calcola le coordinate della iterata successiva

u(1:8,i+1)=u(1:8,i)-resto(1:8,i)

END DO ciclo

iter\_i = i

iter\_r = REAL(iter\_i)

!Traccio il diagramma del resto

CALL diagramma\_resto (norma)

WRITE(\*,\*) "inversa dello jacobiano delle variabili dipendenti"

WRITE(\*,\*) invj(1,:)

WRITE(\*,\*) invj(2,:)

WRITE(\*,\*) invj(3,:)

WRITE(\*,\*) invj(4,:)

WRITE(\*,\*) invj(5,:)

WRITE(\*,\*) invj(6,:)

WRITE(\*,\*) invj(7,:)

WRITE(\*,\*) invj(8,:)

!dichiaro la matrice jacobiana j\_v

j\_v(1) = (0.5)\*l2\*sin(theta2)

j\_v(2) = -(0.5)\*l2\*cos(theta2)

j\_v(3) = (0.5)\*l2\*sin(theta2)

j\_v(4) = -(0.5)\*l2\*cos(theta2)

j\_v(5) = 0.

j\_v(6) = 0.

j\_v(7) = 0.

j\_v(8) = 0.

WRITE(\*,\*) "jacobiano della variabile indipendente"

WRITE(\*,\*) j\_v(1)

WRITE(\*,\*) j\_v(2)

WRITE(\*,\*) j\_v(3)

WRITE(\*,\*) j\_v(4)

WRITE(\*,\*) j\_v(5)

WRITE(\*,\*) j\_v(6)

WRITE(\*,\*) j\_v(7)

WRITE(\*,\*) j\_v(8)

!calcolo le derivate delle coordinate dipendenti

u\_p = (-1)\*MATMUL(invj,j\_v)\*theta2\_p

WRITE(\*,\*) "derivata delle variabili dipendenti"

WRITE(\*,\*) u\_p(1)

WRITE(\*,\*) u\_p(2)

WRITE(\*,\*) u\_p(3)

WRITE(\*,\*) u\_p(4)

WRITE(\*,\*) u\_p(5)

WRITE(\*,\*) u\_p(6)

WRITE(\*,\*) u\_p(8)

!assegno il valore delle derivate prime a variabili omonime

x2\_p = u\_p(1)

y2\_p = u\_p(2)

theta3\_p = u\_p(3)

x3\_p = u\_p(4)

y3\_p = u\_p(5)

theta4\_p = u\_p(6)

x4\_p = u\_p(7)

y4\_p = u\_p(8)

!dichiaro la derivata delle matrice jacobiana j

j\_p = 0.

j\_p(3,3) = (0.5)\*l3\*cos(theta3)\*theta3\_p

j\_p(4,3) = (0.5)\*l3\*sin(theta3)\*theta3\_p

j\_p(5,3) = (0.5)\*l3\*cos(theta3)\*theta3\_p

j\_p(5,6) = (0.5)\*l4\*cos(theta4)\*theta4\_p

j\_p(6,3) = (0.5)\*l3\*sin(theta3)\*theta3\_p

j\_p(6,6) = (0.5)\*l4\*sin(theta4)\*theta4\_p

j\_p(7,6) = -(0.5)\*l4\*cos(theta4)\*theta4\_p

j\_p(8,6) = -(0.5)\*l4\*sin(theta4)\*theta4\_p

WRITE(\*,\*) "derivata della matrice jacobiana delle u"

WRITE(\*,\*) j\_p(1,:)

WRITE(\*,\*) j\_p(2,:)

WRITE(\*,\*) j\_p(3,:)

WRITE(\*,\*) j\_p(4,:)

WRITE(\*,\*) j\_p(5,:)

WRITE(\*,\*) j\_p(6,:)

WRITE(\*,\*) j\_p(8,:)

!dichiaro la derivata della matrice jacobiana j\_v

j\_v\_p = 0.

j\_v\_p(1) = (0.5)\*l2\*cos(theta2)\*theta2\_p

j\_v\_p(2) = (0.5)\*l2\*sin(theta2)\*theta2\_p

j\_v\_p(3) = (0.5)\*l2\*cos(theta2)\*theta2\_p

j\_v\_p(4) = (0.5)\*l2\*sin(theta2)\*theta2\_p

!calcolo le derivate seconde delle coordinate dipendenti

u\_p\_p\_1 = MATMUL(invj,j\_p)

u\_p\_p\_2 = MATMUL(invj,j\_v\_p)

u\_p\_p\_3 = MATMUL(invj,j\_v)

u\_p\_p = (-1)\*( MATMUL(u\_p\_p\_1,u\_p) + (u\_p\_p\_2\*theta2\_p) + (u\_p\_p\_3\*theta2\_p\_p) )

!assegno il valore delle derivate seconde

x2\_p\_p = u\_p\_p(1)

y2\_p\_p = u\_p\_p(2)

theta3\_p\_p = u\_p\_p(3)

x3\_p\_p = u\_p\_p(4)

y3\_p\_p = u\_p\_p(5)

theta4\_p\_p = u\_p\_p(6)

x4\_p\_p = u\_p\_p(7)

y4\_p\_p = u\_p\_p(8)

!calcolo le velocità del punto M3

vxM3 = x3\_p

vyM3 = y3\_p

vM3 = SQRT( (vxM3\*\*2) + (vyM3\*\*2) )

!calcolo la velocità di M4

vxM4 = x4\_p

vyM4 = y4\_p

vM4 = SQRT( (vxM4\*\*2) + (vyM4\*\*2) )

!calcolo l'accelerazione di M3

axM3 = x3\_p\_p

ayM3 = y3\_p\_p

aM3 = SQRT( (axM3\*\*2) + (ayM3\*\*2) )

!calcolo l'accelerazione di M4

axM4 = x4\_p\_p

ayM4 = y4\_p\_p

aM4 = SQRT( (axM4\*\*2) + (ayM4\*\*2) )

WRITE (\*,\*)" "

WRITE (\*,\*)"La velocita' angolare (rad/s) dell'asta 3 e'", theta3\_p

WRITE (\*,\*)" "

WRITE (\*,\*)"La velocita' angolare (rad/s) dell'asta 4 e'", theta4\_p

WRITE (\*,\*)" "

WRITE (\*,\*)"La accelerazione angolare (rad/s^2) dell'asta 3 e'", theta3\_p\_p

WRITE (\*,\*)" "

WRITE (\*,\*)"La accelerazione angolare (rad/s^2) dell'asta 4 e'", theta4\_p\_p

WRITE (\*,\*)" "

WRITE (\*,\*)"La velocita' del punto M3 ha modulo", vM3

WRITE (\*,\*)" "

WRITE (\*,\*)"Le sue componenti x,y sono", vxM3, vyM3

WRITE (\*,\*)" "

WRITE (\*,\*)"La velocita' del punto M4 ha modulo", vM4

WRITE (\*,\*)" "

WRITE (\*,\*)"Le sue componenti x,y sono", vxM4, vyM4

WRITE (\*,\*)" "

WRITE (\*,\*)"La accelerazione del punto M3 ha modulo", aM3

WRITE (\*,\*)" "

WRITE (\*,\*)"Le sue componenti x,y sono", axM3, ayM3

WRITE (\*,\*)" "

WRITE (\*,\*)"La accelerazione del punto M4 ha modulo", aM4

WRITE (\*,\*)" "

WRITE (\*,\*)"Le sue componenti x,y sono", axM4, ayM4

WRITE (\*,\*)" "

WRITE (\*,\*)"Per chiudere il programma premi una lettera qualunque."

WRITE (\*,\*)"Tutti i dati andranno persi."

READ (\*,\*) chiusura

STOP

END PROGRAM main\_ese\_13\_a

**14. Modulo per la soluzione con coordinate assolute.** Segue il modulo relativo al programma principale indicato nel precedente paragrafo.

MODULE mod\_ese\_13\_a

!sezione dichiarativa

IMPLICIT NONE

!dichiaro le costanti geometriche del quadrilatero

REAL, PARAMETER:: l2=.20

REAL, PARAMETER:: l3=.50

REAL, PARAMETER:: l4=.70

REAL, PARAMETER:: l1=.80

REAL, PARAMETER:: theta2 = 20.\*(3.1415927/180)

REAL, PARAMETER:: theta2\_p = 400.\*2\*3.1415927/60

REAL, PARAMETER:: theta2\_p\_p = 0

!dichiaro il numero di iterazioni

INTEGER:: iter\_i !intero

REAL:: iter\_r !relae

!fisso il formato di output per i diagrammi

CHARACTER(len=4):: formato ='bmp '

!scrivo le subroutine

CONTAINS

!---------------------------------------------------------------

subroutine inverse(a,c,n)

!============================================================

! Inverse matrix

! Method: Based on Doolittle LU factorization for Ax=b

! Alex G. December 2009

!-----------------------------------------------------------

! input ...

! a(n,n) - array of coefficients for matrix A

! n - dimension

! output ...

! c(n,n) - inverse matrix of A

! comments ...

! the original matrix a(n,n) will be destroyed

! during the calculation

!===========================================================

implicit none

integer n

double precision a(n,n), c(n,n)

double precision L(n,n), U(n,n), b(n), d(n), x(n)

double precision coeff

integer i, j, k

! step 0: initialization for matrices L and U and b

! Fortran 90/95 aloows such operations on matrices

L=0.0

U=0.0

b=0.0

! step 1: forward elimination

do k=1, n-1

do i=k+1,n

coeff=a(i,k)/a(k,k)

L(i,k) = coeff

do j=k+1,n

a(i,j) = a(i,j)-coeff\*a(k,j)

end do

end do

end do

! Step 2: prepare L and U matrices

! L matrix is a matrix of the elimination coefficient

! + the diagonal elements are 1.0

do i=1,n

L(i,i) = 1.0

end do

! U matrix is the upper triangular part of A

do j=1,n

do i=1,j

U(i,j) = a(i,j)

end do

end do

! Step 3: compute columns of the inverse matrix C

do k=1,n

b(k)=1.0

d(1) = b(1)

! Step 3a: Solve Ld=b using the forward substitution

do i=2,n

d(i)=b(i)

do j=1,i-1

d(i) = d(i) - L(i,j)\*d(j)

end do

end do

! Step 3b: Solve Ux=d using the back substitution

x(n)=d(n)/U(n,n)

do i = n-1,1,-1

x(i) = d(i)

do j=n,i+1,-1

x(i)=x(i)-U(i,j)\*x(j)

end do

x(i) = x(i)/u(i,i)

end do

! Step 3c: fill the solutions x(n) into column k of C

do i=1,n

c(i,k) = x(i)

end do

b(k)=0.0

end do

end subroutine inverse

!------------------------------------------------------------------

SUBROUTINE parallelogramma (x2,y2,x3,y3,theta3,x4,y4,theta4,iterata)

!sezione dichiarativa

!dichiaro gli argomenti fittizi

!dichiaro le coordinate

REAL,INTENT(IN)::x2,y2,x3,y3,theta3,x4,y4,theta4

!dichiaro il numero di iterata

INTEGER,INTENT(IN)::iterata

!dichiaro le variabili locali

!dichiaro e assegno gli estremi dei lati l2

REAL::l2A0x

REAL::l2A0y

REAL::l2Ax

REAL::l2Ay

!dichiaro e assegno gli estremi del lato l3

REAL::l3Ax

REAL::l3Ay

REAL::l3Bx

REAL::l3By

!dichiaro e assegno gli estremi del lato l4

REAL::l4Bx

REAL::l4By

REAL::l4B0x

REAL::l4B0y

!dichiaro e assegno gli estremi del lato l1

REAL::l1B0x

REAL::l1B0y

REAL::l1A0x

REAL::l1A0y

!dichiaro il minimo e ilmassimo delle ascisse e delle ordinate

REAL:: xmin

REAL:: xmax

REAL:: ymin

REAL:: ymax

!sezione esecutiva

!assegno gli estremi dei lati l2

l2A0x = x2 - (0.5\*l2\*COS(theta2))

l2A0y = y2 - (0.5\*l2\*SIN(theta2))

l2Ax = x2 + (0.5\*l2\*COS(theta2))

l2Ay = y2 + (0.5\*l2\*SIN(theta2))

!assegno gli estremi del lato l3

l3Ax = x3 - 0.5\*l3\*COS(theta3)

l3Ay = y3 - 0.5\*l3\*SIN(theta3)

l3Bx = x3 + 0.5\*l3\*COS(theta3)

l3By = y3 + 0.5\*l3\*SIN(theta3)

!assegno gli estremi del lato l4

l4Bx = x4 - 0.5\*l4\*COS(theta4)

l4By = y4 - 0.5\*l4\*SIN(theta4)

l4B0x = x4 + 0.5\*l4\*COS(theta4)

l4B0y = y4 + 0.5\*l4\*SIN(theta4)

!assegno gli estremi del lato l1

l1B0x = 80.

l1B0y = 0

l1A0x = 0

l1A0y = 0

!calcolo l'asscissa minima e quella massima

xmin = MIN(l2A0x,l2Ax, l3Ax, l3Bx, l4Bx, l4B0x, l1B0x, l1A0x)

xmax = MAX(l2A0x,l2Ax, l3Ax, l3Bx, l4Bx, l4B0x, l1B0x, l1A0x)

!calcolo l'ascissa minima e quella massima

ymin = MIN(l2A0y,l2Ay, l3Ay, l3By, l4By, l4B0y, l1B0y, l1A0y)

ymax = MAX(l2A0y,l2Ay, l3Ay, l3By, l4By, l4B0y, l1B0y, l1A0y)

!alcune istruzioni grafiche

CALL METAFL (formato) !indico il formato dell'output

CALL SCRMOD ('revers') !scritta nera su fondo bianco

CALL DISINI !richaima alcune impostazioni di default

CALL PAGERA !traccio un bordo per il piano xy

CALL DUPLX !font a doppio spessore

CALL AXSPOS (450,1800) !coordinate angolo basso sinistra

CALL AXSLEN (2200,1500)!lunghezza dei due assi in pixel

CALL NAME (' ','x') !nome delle ascisse

CALL NAME (' ','y') !nome delle ordinate

CALL TITLIN ("Configurazione",1) !prima riga del titolo

CALL GRAF (xmin-20.,xmax+20.,xmin-20,20.,ymin-20.,ymax+20.,ymin-20.,20.)

CALL GRID (2,2) !impone una griglia sul piano coordinato

CALL TITLE !stampa il titolo di cui sopra

CALL DASH !tratteggio per gli assi coordinati

CALL XAXGIT !traccio la retta y=0

CALL YAxGIT !traccio la retta x=0

CALL MYLINE (1,1) !impone una linea continua

CALL LINWID (8) !spessore della linea

!disegno ciascuno dei lati indicando gli estremi

CALL RLINE (l2A0x, l2A0y, l2Ax, l2Ay)

CALL RLINE (l3Ax, l3Ay, l3Bx, l3By)

CALL RLINE (l4Bx, l4By, l4B0x, l4B0y)

CALL RLINE (l1B0x, l1B0y, l1A0x, l1A0y)

CALL DISFIN

END SUBROUTINE parallelogramma

!------------------------------------------------------------------

SUBROUTINE diagramma\_resto (norma)

!sezione dichiarativa

!dichiaro gli argomenti fittizi

REAL,INTENT(IN),DIMENSION(iter\_i):: norma

!dichiaro le variabili locali

INTEGER::i !indice del ciclo

REAL,DIMENSION(iter\_i)::x !qui metto le ascisse

REAL:: max !il massimo della norma

REAL:: min !il minimo della norma

!sezione esecutiva

!inizializzo l'array delle ascisse

x(1)=1.

ciclo\_ascisse: DO i=2,iter\_i,1

x(i)=x(i-1)+1.

END DO ciclo\_ascisse

WRITE(\*,\*) x

CALL METAFL (formato) !indico il formato dell'output

CALL SCRMOD ('revers') !scritta nera su fondo bianco

CALL DISINI

CALL PAGERA !traccio un bordo per il piano xy

CALL DUPLX !font a doppio spessore

CALL AXSPOS (450,1800) !coordinate angolo basso sinistra

CALL AXSLEN (2200,1500)!lunghezza dei due assi in pixel

CALL NAME ('iterazioni','x') !nome delle ascisse

CALL NAME ('norma del vettore dei resti','y') !nome delle ordinate

CALL TITLIN ("La norma del vettore dei resti deve annullarsi",1) !prima riga del titolo

max=MAXVAL(norma) !il massimo della funzione

min=MINVAL(norma) !il minimo valore della funzione

WRITE(\*,\*) "il minimo della norma del vettoredei resti vale", min

WRITE(\*,\*)"il massimo della norma del vettore dei resti vale",max

CALL GRAF (0.,x(iter\_i),1.,1.0,min,max,0.,20.)

CALL GRID (2,2) !impone una griglia sul piano coordinato

CALL TITLE !stampa il titolo di cui sopra

CALL CURVE (x,norma,iter\_i) !plotto il diagramma del resto

CALL DASH !tratteggio per gli assi coordinati

CALL XAXGIT !traccio la retta y=0

CALL YAxGIT !traccio la retta x=0

CALL DISFIN

END SUBROUTINE diagramma\_resto

!--------------------------------------------------------------------

END MODULE mod\_ese\_13\_a

**15. Codice principale per la soluzione con coordinate naturali.** Segue il programma principale per la soluzione con coordinate naturali.

PROGRAM main\_ese\_13\_b

USE DISLIN !libreria grafica

USE mod\_ese\_13\_b

!sezione dichiarativa

IMPLICIT NONE

!dichiaro il vettore delle coordinate dipendenti

REAL,DIMENSION(3,100)::u

!dichiaro il vettore delle velocità delle coordinate dipendenti

REAL,DIMENSION(3)::u\_p

!dichiaro il vettore delle accelerazioni delle coordinate dipendenti

REAL,DIMENSION(3)::u\_p\_p

!dichiaro dei vettori che servono al calcolo di u\_p\_p

REAL,DIMENSION(3,3)::u\_p\_p\_1

REAL,DIMENSION(3)::u\_p\_p\_2,u\_p\_p\_3

!dichiaro la funzione delle equazioni di vincolo

REAL,DIMENSION(3,100)::psi

!dichiaro la matrice jacobiana delle u, la sua derivata e la sua inversa

REAL(kind=8),DIMENSION (3,3)::j

REAL,DIMENSION (3,3)::j\_p

REAL(kind=8),DIMENSION (3,3)::invj

!dichiaro la matrice jacobiana delle v e la sua derivata

REAL,DIMENSION (3)::j\_v

REAL,DIMENSION (3)::j\_v\_p

!dichiaro i valori della configurazione congruente nota

REAL::x2 = 9.396926

REAL::y2 = 3.4202013

REAL::theta3 = 70.55076\*(3.1415927/180)

REAL::x3 = 27.118143

REAL::y3 = 30.413828

REAL::theta4 = -50.46595\*(3.1415927/180)

REAL::x4 = 57.72122

REAL::y4 = 26.993628

!dichiaro le derivate degli angoli

REAL::theta3\_p, theta3\_p\_p, theta4\_p, theta4\_p\_p

!dichiaro le coordinate naturali

REAL::xA, yA, xB, yB

!dichiaro le derivate prime e seconde delle coordinate naturali

REAL::xA\_p, xA\_p\_p

REAL::yA\_p, yA\_p\_p

REAL::xB\_p, xB\_p\_p

REAL::yB\_p, yB\_p\_p

!dichiaro le funzioni cinematiche relative ai punti M3, M4

REAL::vxM3, vyM3, vM3

REAL::vxM4, vyM4, vM4

REAL::axM3, ayM3, aM3

REAL::axM4, ayM4, aM4

!altre variabili di lavoro

REAL(KIND=8),DIMENSION(3,3)::matrix

INTEGER::i !indici dei cicli

CHARACTER(len=10):: chiusura !serve per chiudere il programma

!sezione esecutiva

!fisso risoluzione dell'immagine in formato .bmp

CALL BMPMOD (300,'inch','resolution')

!disegno la configurazione congruente del parallelogramma

CALL parallelogramma(x2,y2,x3,y3,theta3,x4,y4,theta4,1)

!inizializzo i valori d'innesco delle coordinate dipendenti

xA = l2\*COS(theta2)

yA = l2\*SIN(theta2)

xB = l3\*COS(theta3) + l2\*COS(theta2)

yB = l3\*SIN(theta3) + l2\*SIN(theta2)

!inizializzo le derivate della variabile indipendente

xA\_p = -l2\*theta2\_p\*sin(theta2)

xA\_p\_p = -l2\*( theta2\_p\_p\*sin(theta2) + (theta2\_p\*\*2)\*cos(theta2) )

!inizializzo il vettore u con il valore d'innesco

u(1,1) = yA

u(2,1) = xB

u(3,1) = yB

!calcolo lo jacobiano delle variabili dipendenti

j = 0.

j(1,1) = 2\*yA

j(2,1) = -2\*(yB-yA)

j(2,2) = 2\*(xB-xA)

j(2,3) = 2\*(yB-yA)

j(3,2) = -2\*(l1-xB)

j(3,3) = 2\*yB

!dichiaro la matrice jacobiana j\_v

j\_v(1) = 2\*xA

j\_v(2) = -2\*(xB-xA)

j\_v(3) = 0.

!calcolo l'inversa della matrice jacobiana delle coordinate naturali

CALL inverse(j,invj,3)

!calcolo le derivate delle coordinate dipendenti

u\_p = (-1)\*MATMUL(invj,j\_v)\*xA\_p

WRITE(\*,\*) "derivata delle variabili dipendenti"

WRITE(\*,\*) u\_p(1)

WRITE(\*,\*) u\_p(2)

WRITE(\*,\*) u\_p(3)

!assegno il valore delle derivate prime a variabili omonime

yA\_p = u\_p(1)

xB\_p = u\_p(2)

yB\_p = u\_p(3)

!dichiaro la derivata della matrice jacobiana j

j\_p = 0.

j\_p(1,1) = 2.\*yA\_p

j\_p(2,1) = -2.\*(yB\_p-yA\_p)

j\_p(2,2) = 2.\*(xB\_p-xA\_p)

j\_p(2,3) = 2.\*(yB\_p-yA\_p)

j\_p(3,2) = -2.\*(-xB\_p)

j\_p(3,3) = 2.\*yB\_p

WRITE(\*,\*) "derivata della matrice jacobiana delle u"

WRITE(\*,\*) j\_p(1,:)

WRITE(\*,\*) j\_p(2,:)

WRITE(\*,\*) j\_p(3,:)

!dichiaro la derivata della matrice jacobiana j\_v

j\_v\_p(1) = 2.\*xA\_p

j\_v\_p(2) = -2.\*(xB\_p-xA\_p)

j\_v\_p(3) = 0.

!calcolo le derivate seconde delle coordinate dipendenti

u\_p\_p\_1 = MATMUL(invj,j\_p)

u\_p\_p\_2 = MATMUL(invj,j\_v\_p)

u\_p\_p\_3 = MATMUL(invj,j\_v)

u\_p\_p = (-1)\*( MATMUL(u\_p\_p\_1,u\_p) + (u\_p\_p\_2\*xA\_p) + (u\_p\_p\_3\*xA\_p\_p) )

!assegno il valore delle derivate seconde

yA\_p\_p = u\_p\_p(1)

xB\_p\_p = u\_p\_p(2)

yB\_p\_p = u\_p\_p(3)

!calcolo le derivate degli angoli delle aste

theta3\_p = ( yB\_p - yA\_p )/( xB - xA )

theta3\_p\_p = ( (yB\_p\_p - yA\_p\_p)\*(xB - xA) - (yB\_p - yA\_p)\*(xB\_p - xA\_p) )/(xB - xA)\*\*2

theta4\_p = -xB\_p/yB

theta4\_p\_p = (xB\_p\*yB\_p - XB\_p\_p\*yB)/yB\*\*2

!calcolo le velocità del punto M3

vxM3 = ( xA\_p + xB\_p )/2.

vyM3 = ( yA\_p + yB\_p )/2.

vM3 = SQRT( (vxM3\*\*2) + (vyM3\*\*2) )

!calcolo la velocità di M4

vxM4 = xB\_p/2.

vyM4 = yB\_p/2.

vM4 = SQRT( (vxM4\*\*2) + (vyM4\*\*2) )

!calcolo l'accelerazione di M3

axM3 = ( xA\_p\_p + xB\_p\_p )/2.

ayM3 = ( yA\_p\_p + yB\_p\_p )/2.

aM3 = SQRT( (axM3\*\*2) + (ayM3\*\*2) )

!calcolo l'accelerazione di M4

axM4 = xB\_p\_p/2.

ayM4 = yB\_p\_p/2.

aM4 = SQRT( (axM4\*\*2) + (ayM4\*\*2) )

WRITE (\*,\*)" "

WRITE (\*,\*)"La velocita' angolare (rad/s) dell'asta 3 e'", theta3\_p

WRITE (\*,\*)" "

WRITE (\*,\*)"La velocita' angolare (rad/s) dell'asta 4 e'", theta4\_p

WRITE (\*,\*)" "

WRITE (\*,\*)"La accelerazione angolare (rad/s^2) dell'asta 3 e'", theta3\_p\_p

WRITE (\*,\*)" "

WRITE (\*,\*)"La accelerazione angolare (rad/s^2) dell'asta 4 e'", theta4\_p\_p

WRITE (\*,\*)" "

WRITE (\*,\*)"La velocita' del punto M3 ha modulo", vM3

WRITE (\*,\*)" "

WRITE (\*,\*)"Le sue componenti x,y sono", vxM3, vyM3

WRITE (\*,\*)" "

WRITE (\*,\*)"La velocita' del punto M4 ha modulo", vM4

WRITE (\*,\*)" "

WRITE (\*,\*)"Le sue componenti x,y sono", vxM4, vyM4

WRITE (\*,\*)" "

WRITE (\*,\*)"La accelerazione del punto M3 ha modulo", aM3

WRITE (\*,\*)" "

WRITE (\*,\*)"Le sue componenti x,y sono", axM3, ayM3

WRITE (\*,\*)" "

WRITE (\*,\*)"La accelerazione del punto M4 ha modulo", aM4

WRITE (\*,\*)" "

WRITE (\*,\*)"Le sue componenti x,y sono", axM4, ayM4

WRITE (\*,\*)" "

WRITE (\*,\*)"Per chiudere il programma premi una lettera qualunque."

WRITE (\*,\*)"Tutti i dati andranno persi."

READ (\*,\*) chiusura

STOP

END PROGRAM main\_ese\_13\_b

**16. Modulo per la soluzione con coordinate naturali.** Il modulo per il caso delle coordinate naturali è lo stesso di quello del caso delle coordinate assolute.