Movimenti

Movimenti sul piano

Il linguaggio GDL prevede rotazioni e traslazioni.

Sul piano, in 2D, si hanno i seguenti comandi:

ADD2 x, y

MUL2 a, b

ROT2 an

Il primo comando esprime la traslazione.
Se un punto aveva coordinate iniziali (x1,y1), per effetto di ADD2 x, y esso verrà traslato nella nuova posizione con coordinate (x2,y2) :

x2=x + x1
y2=y + y1

Il secondo comando esprime una dilatazione lungo gli assi.
Se un punto aveva coordinate iniziali (x1,y1), per effetto di MUL2 a, b esso verrà portato nella nuova posizione con coordinate (a*x1,b*y1). Un oggetto verrà dunque ingrandito del fattore a lungo l'asse x e del fattore b nel senso delle y.

Il terzo comando esprime la rotazione antioraria dell'angolo an.
Se un punto aveva coordinate iniziali (x1,y1), per effetto di ROT2 an esso verrà ruotato in una nuova posizione con coordinate (x2,y2) :

x2 = x1*cos(an)-y2*sin(an)
y2 = x1*sin(an)+ y2*cos(an)

La difficoltà nell'effettuare i movimenti è che ogni volta viene trascinato, assieme all'oggetto, il sistema di riferimento in una nuova posizione, così che il nuovo movimento viene calcolato a partire da quest'ultima. Per le sole traslazioni non ci sono problemi, ma quando si effettuano rotazioni, queste cambiano anche i valori delle traslazioni; inoltre ogni rotazione viene sempre effettuata attorno all'origine degli assi, riferita all'ultima posizione relativa, indicata con la lettera L, ma non a quella globale iniziale G.

Per chiarire come funzionano le cose considero inizialmente il solo movimento su piano.

Un vettore (x,y) può sempre essere scritto nella forma:
sqr(x^2+ x^2)*(cos(al), sin(al)),
dove sqr(x^2+ x^2) è il modulo o lunghezza del vettore, mentre al è l'angolo che il vettore forma con l'asse x.

Valgono le formule:
al=(sgn(y)+ 1-sgn(abs(y)))*acs(x/sqr(x^2+ y^2))
oppure:
al=asn(y/sqr(x^2+ y^2)) se x>0, al=180-asn(y/sqr(x^2+ y^2)) se x<0, al=sgn(y)*90 se x=0
o ancora, per -90<al<90: al=atn(y/x)

Infatti:
x= sqr(x^2+ y^2)*cos(al), y= sqr(x^2+ y^2)*sin(al), y=x*tan(al)

Esempio:

Il segmento in 2D si esprime con il comando **line2**, cui vanno fatte seguire le coordinate dei punti iniziale e finale, separati da virgole. Nello stendere lo script , i vari comandi vanno iniziati su righe distinte. Se una riga termina con la virgola, si intende che i valori da assegnare al comando proseguono nella riga sottostante.
Per inserire frasi di riferimento, che non vengano lette dal programma, occorre farle precedere da un punto esclamativo.

Lo script sottostante:

```
rot2 be
line2 x1,y1, x2,y2
del 1  !cancella l'ultimo movimento
```

equivale a:

```
xa= x1*cos(be)-y1*sin(be)
ya= x1*sin(be)+ y1*cos(be)
xb= x2*cos(be)-y2*sin(be)
yb= x2*sin(be)+ y2*cos(be)
line2 xa,ya, xb,yb
```

Vogliamo ora ricavare una formula per spostare un oggetto da un punto A1=(xa1,ya1) a un punto A2=(xa2,ya2), facendolo poi ruotare di un angolo be .
Nel linguaggio GDL:

```
add2 xa2-xa1, ya2-ya1
rot2 be
line2 x1,y1, x2,y2
del 2              !cancella gli ultimi 2 movimenti
```

L'oggetto ruoterà attorno a quel punto che inizialmente costituiva l'origine e che attualmente ha coordinate assolute xa2-xa1, ya2-ya1. L'ordine dei movimenti non può essere scambiato.
Altrimenti occorrerebbe scrivere:

```
rot2 be
(xa2-xa1)*cos(-be)- (ya2-ya1)*sin(-be) , (xa2-xa1)*cos(-be)+ (ya2-ya1)*sin(-be)
```

Questo perché, avendo effettuato la rotazione, il vettore spostamento viene ora visto ruotato di −be .

Le trasformazioni di coordinate equivalgono a:

```
xa= x1*cos(be) - y1*sin(be) + xa2-xa1
ya= x1*sin(be) + y1*cos(be) + ya2-ya1
xb= x2*cos(be) - y2*sin(be) + xa2-xa1
yb= x2*sin(be) + y2*cos(be) + ya2-ya1
line2 xa,ya, xb,yb
```

Più complesso è far ruotare un oggetto di un angolo be non attorno all'origine, ma attorno a un determinato punto C di cooordinate xc,yc. Sia P1≡(x1,y1) il punto iniziale e P2≡(x2,y2) la posizione ruotata, mentre be sia l'angolo antiorario di rotazione attorno a C.

Poniamo   a=xc−x1 , b=yc−y1

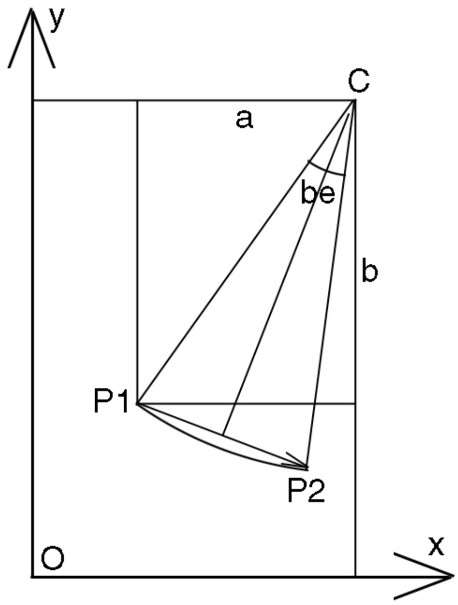
Un modo per risolvere la questione è il considerare le coordinate relative di P1 rispetto C, queste sono: (−a,−b).

La rotazione di quest'ultimo punto si esprime come:
(−a*cos(be)+ b*sin(be), −a*sin(be)+ b*cos(be))

Occorre poi aggiungere il vettore OC≡(xc,yc), per cui infine otteniamo:

x1=xc − (xc−x1)*cos(be) + (yc−y1)*sin(be)
y1=yc − (xc−x1)*sin(be) − (yc−y1)*cos(be)

Oppure, applicando rot2 be , si considera ancora che, il perno attorno a cui avviene la rotazione non sarà più, rispetto l'oggetto, il punto C, ma piuttosto l'origine O.



Effettuata la rotazione, occorrerà pertanto traslare il tutto secondo il vettore OC.
A causa della rotazione degli assi, il vettore spostamento non sarà più (xc,yc) , ma piuttosto:
xc*cos(−be) − yc*sin(−be) , xc*sin(−be) + yc*cos(−be)

Pertanto nel linguaggio GDL, occorrerà scrivere:
rot2 be
add2 −xc + xc*cos(be)+ yc*sin(be) , −yc − xc*sin(be) + yc*cos(be)

L'espressione equivale alla seguente:
add2 xc*(1−cos(be)) + yc*sin(be) , yc*(1−cos(be)) − xc*sin(be)
rot2 be

Infatti ruotare P1 di be equivale a ruotare dello stesso valore tutto il piano, compresa l'origine O. Nel caso di dell'origine, la sua posizione relativa rispetto C risulta essere: O≡(-xc, -yc), a seguito della rotazione verrà spostata in O2≡(−xc*cos(be)+ yc*sin(be), −xc*sin(be)−yc*cos(be)), pertanto il vettore spostamento sarà:
O2-O≡(−xc*cos(be)+ yc*sin(be)−xc, −xc*sin(be)−yc*cos(be)−yc)

Dimostrazione: il comando add2 porta l'origine O nel nuovo punto O2 e successivamente il comando rot2 opera una rotazione del piano attorno a questo punto, che quindi non si sposta. Dunque, nell'insieme delle due operazioni, il piano viene ruotato di be, in modo che O vada a cadere su O2.

Un altro modo per ottenere il vettore spostamento dell'origine è calcolarne direttamente modulo e versore.
Il versore ha la direzione normale alla bisettrice dell'angolo O-C-O2, la quale forma con l'asse x l'angolo:
al=atn(yc/xc)+ be/2
Pertanto tale versore avrà coordinate: ux= sin(al), uy=−cos(al)
La lunghezza della corda vale   cd=2*sqr(xc^2+ yc^2)*sin(be/2)

In conclusione otteniamo una ulteriore formula equivalente:
add2 cd * ux, cd * uy
rot2 be

Vogliamo ancora osservare che la formula diretta di rotazione attorno al perno (xc,yc) può essere riscritta attraverso una matrice di trasformazione:

```
a11=  cos(be)
a21= -sin(be)
a31= xc*( 1-cos(be) ) +  yc*sin(be)
a12= sin(be)
a22= cos(be)
a32= yc*( 1-cos(be) ) - xc*sin(be)
x2= a11*x1+ a21*y1+ a31
y2= a12*x1+ a22*y1+ a32    !rotazione attorno a C ≡ (xc,yc)
```

Il versore colonna (a11,a12) ≡ (cos(be),sin(be)) esprime la nuova posizione assunta dal versore cartesiano delle ascisse: (1,0); il versore colonna ad esso ortogonale (a21,a22) ≡ (-sin(be),cos(be)) descrive la nuova posizione assunta dal versore cartesiano delle ordinate: (0,1); l'ultimo vettore colonna (a31,a32) rappresenta il vettore spostamento.
Ricordiamo che utilizzando i comandi add e rot in successione, va sempre inserito prima quello dello spostamento, poi quello di rotazione.

I movimenti rigidi del piano, anche se composti di rotazione e traslazione, possono essere ridotti ad una unica rotazione, considerando una eventuale traslazione come una rotazione con centro all'infinito.

Per verificare la validità delle formule con un oggetto GDL non troppo elementare, vogliamo determinare il centro di rotazione di un versore che si sposta dalla posizione A1 alla posizione A2, ruotando di be.

Le coordinate del versore iniziale sono:
(cos(ga),sin(ga))
con punto di applicazione: A1=(xa1,ya1),
le coordinate del versore finale sono:
(cos(ga+ be),sin(ga+ be))
con punto di applicazione: A2=(xa2,ya2).
Il centro di rotazione C si trova sull'asse del segmento A1-A2, il cui punto medio M ha coordinate:
xm=(xa1+ xa2)/2
ym=(ya1+ ya2)/2
Costruendo il triangolo isoscele A1-A2-C è evidente che l'angolo A1-C-A2 ha ampiezza be, se infatti per ipotesi il versore iniziale fosse ortogonale a C-A1, cioè tangente al cerchio con centro C per A1, il versore finale sarebbe ancora tangente al cerchio, essendo poi ruotato di be, l'angolo al centro deve valere appunto be. Posto : d=A1-A2, risulta:
d=sqr((xa2-xa1)^2+ (ya2-ya1)^2)
detta h l'altezza del triangolo A1-A2-C, risulta: d/2=h*tan(be/2)
h=d/(2* tan(be/2))
Il raggio del cerchio, cioè il lato del triangolo C-A1:
r=sqr(d^2/4+ h^2)
L'inclinazione del vettore A1-A2 è data dall'angolo al:
al=acs((xa2-xa1)/d)
Poiché il centro C si trova sull'asse di A1-A2 a distanza h, le sue coordinate sono:
xc=xm-h*sin(al)
yc=ym+ h*cos(al)

Di fatto abbiamo determinato il centro xc,yc dell'arco A1-A2, dato l'angolo al centro be :

```
xa1=
ya1=
xa2=
ya2=
be=    !fine parametri
xm=(xa1+ xa2)/2
ym=(ya1+ ya2)/2
d=sqr((xa2-xa1)^2+ (ya2-ya1)^2)
h=d/(2* tan(be/2))
al=acs(  (xa2-xa1)/d )
r=sqr(d^2/4+ h^2)  !raggio
xc=xm-h*sin(al)
yc=ym+ h*cos(al)
```

Oltre al comando line2 , già considerato, nel testo 2D si utilizzano i comandi:

```
arc2 xc,yc,r,an1,an2    !xc,yc=coordinate centro, r=raggio, an1=angolo iniziale, an2 angolo finale
```

```
circle2 xc,yc,r             !xc,yc=coordinate del centro, r=raggio
```

```
rect2 x1,y1,x2,y2
!x1,y1 coordinate vertice in basso a sinistra, x2,y2 coordinate vertice in alto a destra.
```

Il primo comando fornisce un arco e necessita di specificare gli angoli iniziale e finale rispetto l'asse x.
Volendo tracciarne i raggi occorre scrivere:

```
line2 xc,yc, xc+ r*cos(an1),yc+ r*sin(an1)
line2 xc,yc, xc+ r*cos(an2),yc+ r*sin(an2)
arc2 xc,yc,r,an1,an2       !settore circolare
```

Il secondo origina un cerchio. Questi due comandi generano effettive curve e non spezzate composte di un numero finito di segmenti. Se esportati con un programma vettoriale presenteranno i nodi su cui intervenire.

Il comando rect2 fornisce un rettangolo parallelo agli assi, volendo tracciarne le diagonali si scriverà:

```
line2 x1,y1,x2,y2
line2 x1,y2,x2,y1
```

volendo tracciarne invece le mediane si scriverà:

```
line2 (x1+ x2)/2,y1,(x1+ x2)/2,y2
line2 x1,(y1+ y2)/2,x2,(y1+ y2)/2
```

Un altro modo per ottenere il rettangolo è lo script:

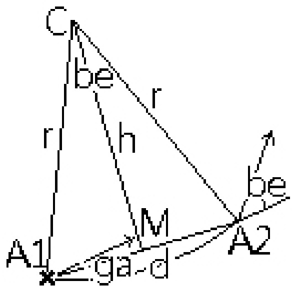
```
line2 x1,y1,x2,y1
line2 x2,y1,x2,y2
line2 x2,y2,x1,y2
line2 x1,y2,x1,y1
```

Come esercizio costruiamo un oggetto dove si verifica la formula che determina il centro di rotazione di un movimento sul piano, che porta un vettore dalla posizione A1 alla posizione A2, facendone cambiare l'inclinazione dall'angolo ga all'angolo be. All'oggetto è assegnato il nome:

rotazioni 2D.gsm

Per generarne l'icona, una volta costruito l'oggetto, occorre copiarne l'immagine e ridurla a un quadrato di 128x128 pixel. Nel caso 3D si può salvare direttamente dalla **Vista 3D**, nel caso 2D occorre ritagliarla sul **2D parametrico** con lo strumento forbici di Windows, oppure riportarla sul piano di Archicad, dopo aver richiamato l'oggetto. Si aprirà poi la casella con **immag.di anteprima** e la si incollerà, dopo averla eventualmente elaborata con un programma di fotoritocco.

Lo script 2D è il seguente :




```
xa1=
ya1=
xa2=
ya2=
ga=
be=
!fine parametri
xm=(xa1+xa2)/2
ym=(ya1+ya2)/2
d=sqr((xa2-xa1)^2+(ya2-ya1)^2)
al=acs(  (xa2-xa1)/d )
h=d/(2* tan(be/2))
r=sqr(d^2/4+h^2)  !raggio
xc=xm-h*sin(al)
yc=ym+h*cos(al)
pen 21
line2 xa1,ya1, xa2,ya2
line2 xc,yc, xm,ym
line2 xc,yc, xa2,ya2
line2 xa1,ya1, xc,yc
arc2 xc,yc,r,-90+al-be/2,-90+al+be/2
pen 20
line2 xa1,ya1,xa1+cos(ga),ya1+sin(ga)
line2 xa2,ya2,xa2+cos(ga),ya2+sin(ga)
line2 xa2,ya2,xa2+cos(ga+be),ya2+sin(ga+be) !vettore ruotato e traslato
pen 6
rot2 be
add2 -xc +xc*cos(be)+  yc*sin(be) , -yc+  yc*cos(be)-xc*sin(be)
line2 xa1,ya1,xa1+cos(ga),ya1+sin(ga)            !vettore ruotato e traslato, verifica 1
del 2
pen 4
xa= (xa1-xc)*cos(be) - (ya1-yc)*sin(be) + xc
ya= (xa1-xc)*sin(be) +(ya1-yc)*cos(be) + yc
xb= (xa1+cos(ga)-xc)*cos(be) -(ya1+sin(ga)-yc)*sin(be) + xc
yb= (xa1+cos(ga)-xc)*sin(be) +(ya1+sin(ga)-yc)*cos(be) + yc
line2 xa,ya, xb,yb                 !vettore ruotato e traslato, verifica 2
```