01173

DIVISION DE ESTUDIOS DE POSGRADO
Facultad De Ingeniería

SIMULACION COMPUTACIONAL DE CADENAS CINEMATICAS ABIERTAS TRIDIMENSIONALES

MIGUEL ANGEL VEGA GONZALEZ

TESIS

PRESENTADA A LA DIVISION DE ESTUDIOS DE POSGRADO DE LA

FACULTAD DE INGENIERIA

DE LA

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

COMO REQUISITO PARA OBTENER EL GRADO DE

MAESTRO EN INGENIERIA
(MECANICA)

Director de Tesis: **Dr. Luis Reyes Avila**

CIUDAD UNIVERSITARIA, AGOSTO 1997.

AGRADECIMIENTOS

A mis padres Miguel y Ma. Soledad, ya que ellos lograron darme las bases necesarias para poder siempre seguir avanzando.

A mi esposa Beatriz y mis dos hijas Blanca Angélica y Claudia Patricia por todo el apoyo que me dieron para iniciar y concluir esta etapa de mi formación profesional.

A todos mis hermanos que siempre me han alentado para seguir adelante en todo lo que me he propuesto.

Al Dr. Luis Reyes Avila que desde un principio logró infundirme confianza en que con el trabajo se pueden lograr grandes cosas.

A todos mis compañeros y maestros que de una u otra forma incidieron para llegar a este punto tan importante de mi vida.

CONTENIDO

INTRODUCCION

En este trabajo se presenta la aplicación del álgebra de Quaterniones para generar las ecuaciones que gobiernan la cinemática de cadenas cinemáticas abiertas, las cuales se utilizan para generar programas en el lenguaje de cálculo formal Mathematica®, que nos permitirán simular el movimiento de los manipuladores que se seleccionaron como ejemplos. El principal objetivo es demostrar la validez del álgebra de Quaterniones para la modelación de la cinemática de, básicamente, cualquier tipo de manipulador robótico con relativa facilidad. Lo anterior se logra al presentar la modelación de juntas rotacionales y prismáticas, que son las dos juntas básicas de la mayoría de los manipuladores, como se podrá ver en la clasificación que se presenta más adelante.

En el primer capítulo se dan algunas definiciones básicas dando una clasificación de los manipuladores robóticos de cadena abierta.

En el segundo capítulo se muestran las propiedades del álgebra de Quaterniones y se proponen dos transformaciones lineales (rotación y reflexión) realizando todas las demostraciones de las propiedades que les corresponden, además de demostrar cual es el significado de los parámetros que las caracterizan; muchas de las demostraciones son realizadas utilizando Mathematica.

En el tercer capítulo se realiza el proceso de modelación de tres manipuladores utilizando rotaciones, uno de tres grados de libertad, otro de cuatro y el último un tipo SCARA; durante el proceso de modelación del primer manipulador, se muestra el efecto de las secuencias de movimientos en el modelo final obtenido, el cual no es afectado, solamente se tienen diferentes puntos intermedios. El manipulador de tres grados es utilizado también para generar un modelo alterno al obtenido con puras rotaciones, dicho modelo se obtiene utilizando una combinación de rotaciones y reflexiones.

En el cuarto capítulo se muestra el código desarrollado en Mathematica para poder realizar las simulaciones computacionales de los tres manipuladores, los programas son acompañados por breves descripciones de las instrucciones utilizadas, estos programas permiten observar la forma que adquieren las ecuaciones cinemáticas explícitas de cada manipulador, ya que son generadas en los mismos.

Finalmente el quinto y último capítulo nos presenta una aplicación de los diferentes modelos que se pueden obtener al utilizar solamente rotaciones, reflexiones o una combinación de ellas. La aplicación es del manipulador de tres grados de libertad del cual se obtuvieron dos modelos cinemáticos diferentes en el capítulo III, y consiste en lograr la evasión de un obstáculo virtual en el proceso de cinemática inversa.

CAPITULO I

1. ALGUNAS DEFINICIONES BASICAS

Un robot manipulador, en principio, es un dispositivo de propósito general; en la práctica los manipuladores son usualmente diseñados pensando en que cubran al menos una clase amplia de aplicaciones, como podría ser, soldar, manejar materiales, ensamblar, entre otras. Estas aplicaciones en gran parte dictan la elección de varios parámetros de diseño del manipulador, incluyendo su estructura cinemática. Por ejemplo, el ensamblado de tarjetas de circuitos es realizada naturalmente por un manipulador tipo SCARA. Los manipuladores pueden ser clasificados por muchos criterios, tales como su geometría o estructura cinemática, el tipo de aplicación para la cual fueron diseñados, la manera en que son controlados, etc.

Muchos manipuladores industriales actuales tienen 6 ó menos grados de libertad. Estos manipuladores son usualmente clasificados cinemáticamente sobre la base del primer brazo o las primeras 3 juntas, describiendo por separado el órgano terminal o muñeca. La mayoría de estos manipuladores caen en uno de los siguientes tipos de geometría, [5].

1.- Articulados.
2.- Esféricos.
3.- Tipo SCARA.
4.- Cilíndricos.
5.- Cartesianos.

Manipuladores Articulados.- Estos también son llamados manipuladores antropomórficos. Un diseño común es el manipulador tipo codo, tal como el PUMA (Programmable Universal Manipulator for Assembly), en éste arreglo el eje de la junta z_3 es paralelo a z_2 y, z_2 y z_3 son perpendiculares a z_1 (Figura 1.1). Esta configuración proporciona, relativamente, una amplia libertad de movimiento en un espacio de trabajo compacto.

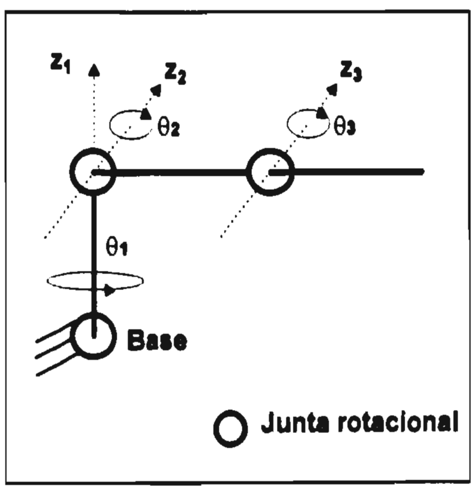


Figura 1.1 *Manipulador articulado*

Manipuladores Esféricos.- Este tipo de manipuladores se obtienen al remplazar la tercera junta de un manipulador antropomórfico por una junta prismática (Figura 1.2).

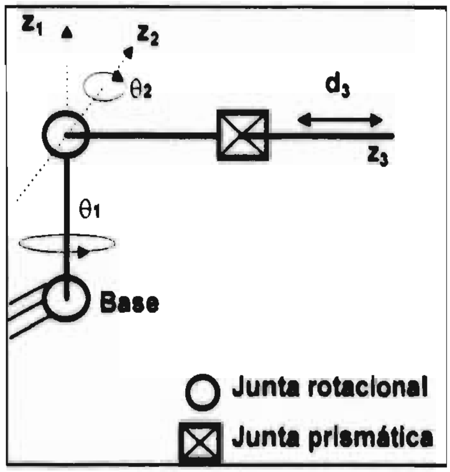


Figura 1.2 *Manipulador esférico*

Manipulador tipo SCARA.- El llamado SCARA (Selective Compliant Articulated Rob:t Assembly), mostrado en la Figura 1.3, es una configuración relativamente reciente que a incrementado su popularidad, el cual como su nombre lo indica está diseñado principalmente para operaciones de ensamble.

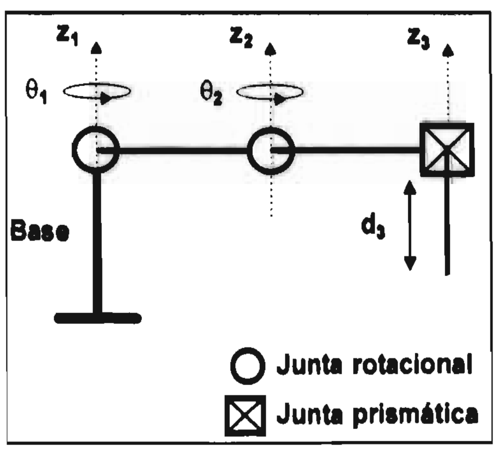


Figura 1.3 *Manipulador tipo SCARA*

Manipulador Cilíndrico.- La configuración de un manipulador cilíndrico se muestra en la Figura 1.4. Como se puede observar la primera junta es rotacional y produce una rotación alrededor de la base, mientras las otras dos juntas son prismáticas.

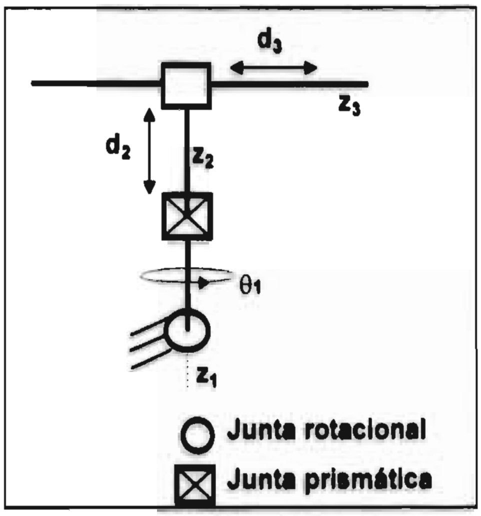


Figura 1.4 *Manipulador cilíndrico.*

Manipulador Cartesiano.- Un manipulador cuyas primeras tres juntas sean prismáticas es conocido como manipulador cartesiano, en la **Figura** 1.5 se muestra dicha configuración.

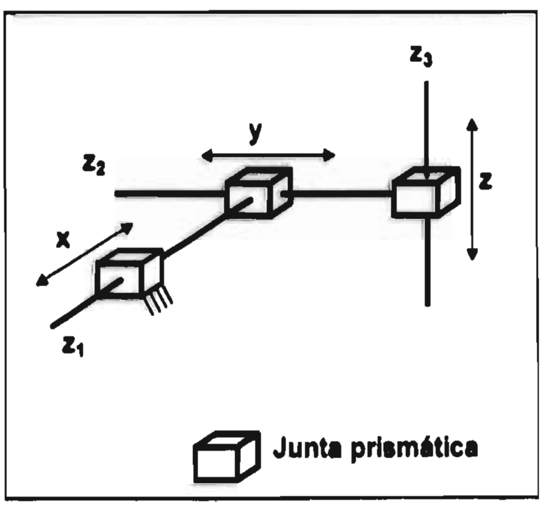


Figura 1.5 *Manipulador cartesiano*

CAPITULO II

2. ANTECEDENTES

2.1. Algebra de Quaterniones.

Los Quaterniones se han utilizado en años recientes, entre otras cosas, para representar la forma paramétrica de rotaciones finitas de cuerpos rigidos, basándose en representaciones puramente geométricas presentándolos como una generalización de los números complejos. Sin embargo en sus aplicaciones no se encuentran las definiciones precisas y apropiadas de sus conceptos. En este trabajo, basándonos en [1],[2], se presentará una estructura algebraica mediante la cual sistematizaremos las aplicaciones estudiadas.

Definamos dos operaciones binarias sobre el conjunto $\mathfrak{R}^4$, una operación aditiva $\oplus:\mathfrak{R}^4\times\mathfrak{R}^4\to\mathfrak{R}^4$ y otra multiplicativa $*:\mathfrak{R}^4\times\mathfrak{R}^4\to\mathfrak{R}^4$ mediante las cuales las parejas $(\mathfrak{R}^4,\oplus)$ y $(\mathfrak{R}^4,*)$ forman, respectivamente, un grupo aditivo conmutativo y un grupo multiplicativo no conmutativo, definiremos además una multiplicación escalar $\bullet$: $\mathfrak{R}\times\mathfrak{R}^4\to\mathfrak{R}^4$ y un producto interno $\langle\bullet,\bullet\rangle$: $\mathfrak{R}^4\times\mathfrak{R}^4\to\mathfrak{R}$ mediante el cual $\mathbf{Q}\equiv(\mathfrak{R}^4,\oplus,\bullet,\langle\bullet,\bullet\rangle)$ es un espacio vectorial provisto de un producto interno. Tal espacio vectorial será llamado espacio vectorial de Quaterniones. Consideremos el conjunto $\mathfrak{R}^4$ y definamos las operaciones siguientes:

I).- $$(\mathbf{a},\mathbf{b},\mathbf{c},\mathbf{d})\oplus(\alpha,\beta,\gamma,\delta)=(\mathbf{a}+\alpha,\ \mathbf{b}+\beta,\ \mathbf{c}+\gamma,\ \mathbf{d}+\delta) \tag{2.1}$$

II).- $$\begin{aligned}(\mathbf{a},\mathbf{b},\mathbf{c},\mathbf{d})*(\alpha,\beta,\gamma,\delta)=(&\mathbf{a}\alpha-\mathbf{b}\beta-\mathbf{c}\gamma-\mathbf{d}\delta,\ \mathbf{a}\beta+\mathbf{b}\alpha+\mathbf{c}\delta-\mathbf{d}\gamma,\\ &\mathbf{a}\gamma-\mathbf{b}\delta+\mathbf{c}\alpha+\mathbf{d}\beta,\mathbf{a}\delta+\mathbf{b}\gamma-\mathbf{c}\beta+\mathbf{d}\alpha),\\ &\forall(\mathbf{a},\mathbf{b},\mathbf{c},\mathbf{d}),(\alpha,\beta,\gamma,\delta)\in\mathfrak{R}^4.\end{aligned} \tag{2.2}$$

La operación $\oplus:\mathfrak{R}^4\times\mathfrak{R}^4\to\mathfrak{R}^4$ es la suma usual en $\mathfrak{R}^4$ y se sabe que el conjunto $(\mathfrak{R}^4,\oplus)$ es un grupo aditivo conmutativo. En [1] se estudiaron las propiedades de las operaciones $\oplus:\mathfrak{R}^4\times\mathfrak{R}^4\to\mathfrak{R}^4$ y $*:\mathfrak{R}^4\times\mathfrak{R}^4\to\mathfrak{R}^4$, realizándose los desarrollos de manera manual. En esta parte concretaremos las demostraciones de manera simbólica mediante Mathematica®, esto permitirá construir nuevas estructuras algebraicas para parametrizar las rotaciones e incluso reflexiones, simplificando los desarrollos algebraicos. Ahora mostraremos que las operaciones anteriores satisfacen las siguientes propiedades.

Teorema 2.1.1: La terna $\mathbf{Q}\equiv(\mathfrak{R}^4,\oplus,*)$ es un campo no conmutativo.

Demostración: Sabemos que $(\mathfrak{R}^4,\oplus)$ es un grupo aditivo conmutativo, por lo que solo falta mostrar que $(\mathfrak{R}^4,*)$ es un grupo multiplicativo no conmutativo. Para ello debe cumplir las propiedades siguientes:

I).- La operación $*:\mathfrak{R}^4\times\mathfrak{R}^4\to\mathfrak{R}^4$ es asociativa. En decir, siendo **p,q,s** $\in$ **Q**, donde p={a,b,c,d}, q={e,f,g,h}, s={w,x,y,z}, tenemos que: **p*(q*s)=(p*q)*s**. (Las demostraciones son realizadas con **Mathematica®** y serán encerradas en un marco como se muestra a continuación).

Demostración: En efecto,

In[1]:= (*Este símbolo significa entrada de datos u operaciones y aparece automaticamente al correr el programa*)

(* Se define el producto de dos Quaterniones ProductoQ[p,q] *) (*comentarios*)

```
ProductoQ[p_,q_]:= {p[[1]] * q[[1]] - p[[2]] * q[[2]] - p[[3]] * q[[3]]  -  p[[4]] * q[[4]],
                    p[[1]] * q[[2]] + p[[2]] * q[[1]] + p[[3]] * q[[4]] - p[[4]] * q[[3]],
                    p[[1]] * q[[3]] - p[[2]] * q[[4]] + p[[3]] * q[[1]] + p[[4]] * q[[2]],
                    p[[1]] * q[[4]] + p[[2]] * q[[3]] - p[[3]] * q[[2]] + p[[4]] * q[[1]]};
```

In[3]:=

(* se asignan componentes a los Quaterniones a utilizar en la demostración, el punto y coma sirve para evitar que se despliegue el resultado de la operación *)

```
p = {a,b,c,d};  q = {e,f,g,h};  s = {w,x,y,z};
```

In[5]:=

(* Se demuestra la asociatividad en la multiplicación, es decir que "**p*(q*s)=(p*q)*s**", se utiliza el signo **===** para comparar las dos expresiones y se utiliza **Expand[]** para que pueda realizarse la comparación *)

```
Expand[ProductoQ[p,ProductoQ[q,s]]] === Expand[ProductoQ[ProductoQ[p,q],s]]
```

Out[6]= (*Este símbolo significa salida de datos o resultados y se genera automaticamente*)

```
True
```

■

Podemos ver que Mathematica® nos da como salida de la comparación la palabra **True,** lo que significa que las expresiones comparadas son iguales, es decir la multiplicación definida si es ***asociativa.***

ii).- El elemento $\underset{\sim}{1}$ =(1,0,0,0) $\in$ **Q es tal que $\underset{\sim}{1}$ *p = p* $\underset{\sim}{1}$ = p,** $\forall$**p** $\in$ Q, y es llamado elemento neutro **para la multiplicación en Q.**

iii).- $\forall$**p** $\in$ **Q, p** $\neq$ **(0,0,0,0),** existe **p'** $\in$ Q, tal que **p*p' =** $\underset{\sim}{1}$. el elemento **p'** es conocido como el inverso multiplicativo de **p**. Si tenemos que **p=(a,b,c,d)** y **p'=(e,f,g,h)** $\in$ **Q** tal que **p*p' =** $\underset{\sim}{1}$, se tiene que:

$$p' = \left\{\frac{a}{a^2+b^2+c^2+d^2}, \frac{-b}{a^2+b^2+c^2+d^2}, \frac{-c}{a^2+b^2+c^2+d^2}, \frac{-d}{a^2+b^2+c^2+d^2}\right\}$$

Demostración: En efecto,

```
In[1]:=
     (* Se definen los Quaterniones p y p' *)

     p={a,b,c,d};  p'={e,f,g,h};
In[3]:=
     (* como p*p' = (1,0,0,0), se tiene un conjunto de 4 ecs*)

     pp'=ProductoQ[p,p']; (* Se calcula el producto p*p' *)

     ec1=pp'[[1]]==1
     ec2=pp'[[2]]==0
     ec3=pp'[[3]]==0
     ec4=pp'[[4]]==0

Out[5]=
     a e - b f - c g - d h == 1
Out[6]=
     b e + a f - d g + c h == 0
Out[7]=
     c e + d f + a g - b h == 0
Out[8]=
     d e - c f + b g + a h == 0

In[9]:=
     (* Se resuelven las ecuaciones utilizando la función Solve[] *)

     Solve[{ec1,ec2,ec3,ec4},{e,f,g,h}]//Simplify

Out[11]=

                 a                                b
{{e -> -----------------, f -> -(-----------------),
         2    2    2    2             2    2    2    2
        a  + b  + c  + d             a  + b  + c  + d

                   c                            d
  g -> -(-----------------), h -> -(-----------------)}}
           2    2    2    2           2    2    2    2
          a  + b  + c  + d           a  + b  + c  + d
```

La solución del sistema de ecuaciones es el que se muestra en la salida *Out[11]*, que corresponde a las componentes de **p'**, es decir:

$$\mathbf{p'} = \left\{\frac{a}{a^2+b^2+c^2+d^2}, \frac{-b}{a^2+b^2+c^2+d^2}, \frac{-c}{a^2+b^2+c^2+d^2}, \frac{-d}{a^2+b^2+c^2+d^2}\right\} \qquad (2.3)$$

■

iv).- La operación $*: \Re^4 \times \Re^4 \to \Re^4$ es no conmutativa, es decir, $\forall \mathbf{p,q} \in \mathbf{Q}$, $\mathbf{p*q} \neq \mathbf{q*p}$.

Demostración: En efecto,

```
In[1]:=
    (* Se define el conjugado de un Quaternion *)

    ConjugadoQ[p_]:={p[[1]],-p[[2]],-p[[3]],-p[[4]]};
In[3]:=
    (* Se definen Quaterniones p y q *)

    p={a,b,c,d};  q={e,f,g,h};
In[5]:=
    (* Se obtienen los producto p*q y q*p *)

    pq=ProductoQ[p,q];
    qp=ProductoQ[q,p];
In[8]:=
    (* Se comparan los productos *)
    pq === qp
Out[9]=
    False
```

De la demostración anterior se determinó que la multiplicación definida es no conmutativa, ya que $\forall$**p,q** $\in$ **Q, p*q ≠ q*p.**

■

v).- Las propiedades siguientes de distributividad deben ser son satisfechas:

$$(p \oplus q)*s = p*s \oplus q*s, \qquad (2.4)$$

$$p*(q \oplus s) = p*q \oplus p*s,$$

Demostración: En efecto,

```
In[1]:=
    (* Se definen los Quaterniones p,q,s *)

    p={a,b,c,d};  q={e,f,g,h};  s={w,x,y,z};

In[3]:=
    (* Se comprueba la igualdad de las propiedades de distributividad *)

    Expand[ProductoQ[(p+q),s]] === Expand[ProductoQ[p,s]+ProductoQ[q,s]]
    Expand[ProductoQ[p,(q+s)]] === Expand[ProductoQ[p,q]+ProductoQ[p,s]]

Out[4]=

    True

Out[5]=

    True
```

■

Es bien conocido que la operación $\bullet : \Re \times Q \to Q$ definida $\forall \alpha \in \Re$, por :

$$\alpha \bullet (a,b,c,d) = (\alpha a, \alpha b, \alpha c, \alpha d), \quad \forall (a,b,c,d) \in \Re^4 \qquad (2.5)$$

es una multiplicación escalar y por consecuencia Q es un espacio vectorial. Consideremos los subespacios siguientes de Q:

$$Q_R = \{(a,0,0,0) : a \in \Re\} \subset Q, \qquad (2.6)$$

$$Q_V = \{(0,b,c,d) : b,c,d \in \Re\} \subset Q, \qquad (2.7)$$

los cuales son isomórficos a $\Re$ y $\Re^3$ respectivamente, puesto que las transformaciones T_R: $Q_R \to \Re$, T_V: $Q_V \to \Re^3$, definidas por:

$$T_R(a,0,0,0) = a, \qquad \forall (a,0,0,0) \in Q_R \qquad (2.8)$$

$$T_V(0,b,c,d) = (b,c,d), \qquad \forall (0,b,c,d) \in Q_V \qquad (2.9)$$

son isomorfismos.

Con la operación escalar definida, podemos representar el inverso multiplicativo de **p=(a,b,c,d)**, obtenido anteriormente como **p'**:

$$p' = \frac{1}{a^2+b^2+c^2+d^2} \bullet \{a, -b, -c, -d\} = \frac{1}{a^2+b^2+c^2+d^2} \bullet \overline{p} \,. \qquad (2.10)$$

Donde $\overline{p} = (a, -b, -c, -d)$ lo llamaremos el **conjugado** de $p = (a, b, c, d)$. Con lo anterior podemos representar el elemento $p \in Q$ de la forma siguiente:

$$p = p_R \oplus p_V, \qquad p_R \in Q_R, \qquad p_V \in Q_V \qquad (2.11)$$

De acuerdo con los isomorfismos T_R: $Q_R \to \Re$, T_V: $Q_V \to \Re^3$, podemos decir que p_R y p_V son respectivamente la **parte real** y la **parte vectorial** de $p \in Q$. Además el conjugado satisface los resultados siguientes.

Teorema 2.1.2: Si $p = (p_0, p_1, p_2, p_3) \in Q$ y $q = (q_0, q_1, q_2, q_3) \in Q$, se tiene que:

i).- $\overline{p \oplus q} = \overline{p} \oplus \overline{q}$,

ii).- $\overline{p * q} = \overline{p} * \overline{q}$,

iii).- $p * \overline{p} = \overline{p} * p \in Q_R$. $\qquad (2.12)$

Demostración: Como lo hicimos anteriormente la demostración se realizará utilizando Mathematica®. En efecto:

```
In[1]:=
      (* Se define el conjugado de un Quaternión *)

      ConjugadoQ[p_]:={p[[1]], - p[[2]], - p[[3]], - p[[4]]};

In[3]:=
      (* Se definen Quaterniones p y q *)

      p = {a,b,c,d};  q = {e,f,g,h};

In[5]:=
      (* inciso i) *)

      ConjugadoQ[p+q] == ConjugadoQ[p]+ConjugadoQ[q]

Out[6]=

      True

In[7]:=
      (* inciso ii) *)

      ConjugadoQ[ProductoQ[p,q]] == ProductoQ[ConjugadoQ[q],ConjugadoQ[p]]

Out[8]=

      True

In[9]:=
      (* inciso iii) *)

      ProductoQ[p,ConjugadoQ[p]] == ProductoQ[ConjugadoQ[p],p]

Out[10]=

      True
```

■

Es bien conocido que la transformación $\langle \bullet,\bullet \rangle$: $\Re^4 \times \Re^4 \to \Re$ definida por $\langle \mathbf{p,q} \rangle \doteq \mathbf{p_0q_0+p_1q_1+p_2q_2+p_3q_3}$ es un producto interno en $\Re^4$ por lo que (Q, $\oplus$,$*$, $\langle \bullet,\bullet \rangle$) es un espacio vectorial con producto interno. Además se tiene que la función $\|\bullet\| : \Re^4 \to \Re$, definida por $\|\mathbf{p}\| \doteq \langle \mathbf{p,p} \rangle^{1/2}$, es la norma para el producto interno, con lo que podemos concluir que (Q, $\oplus$,$*$, $\|\bullet\|$) es un espacio normado. Un resultado importante para nuestra álgebra se presenta en el siguiente teorema.

Teorema 2.1.3: La transformación $\langle \bullet,\bullet \rangle$: $\Re^4 \times \Re^4 \to \Re$ puede representarse de la forma siguiente:

$$\langle \mathbf{p,q} \rangle \doteq \tfrac{1}{2} \mathbf{T_R}(\bar{\mathbf{p}} * \mathbf{q} \oplus \bar{\mathbf{q}} * \mathbf{p}), \quad \forall \mathbf{p,q} \in \mathbf{Q}$$

Demostración: En efecto,

```
In[1]:=
    (* Se define el conjugado de un Quaternion *)

    ConjugadoQ[p_]:={p[[1]],-p[[2]],-p[[3]],-p[[4]]};

In[3]:=
    (* Se definen Quaterniones p y q *)

    p={p0,p1,p2,p3}; q={q0,q1,q2,q3};

In[5]:=
    (* Se determina el producto de la definición (p̄ * q ⊕ q̄ * p)   *)

    ProductoQ[ConjugadoQ[p],q]+ProductoQ[ConjugadoQ[q],p]

Out[6]=

    {2 p0 q0 + 2 p1 q1 + 2 p2 q2 + 2 p3 q3, 0, 0, 0}
```

De lo anterior podemos ver que $T_R(\bar{p} * q \oplus \bar{q} * p) = 2(p_0q_0 + p_1q_1 + p_2q_2 + p_3q_3) \doteq 2\langle p,q\rangle$.

■

2.2. Representación paramétrica de rotaciones.

En esta parte utilizaremos el álgebra de Quaterniones, presentada anteriormente, para presentar de forma paramétrica una rotación de cuerpo rígido. Lo anterior se realizará mediante a definición de una transformación lineal $\rho(p,\bullet)$: $Q \rightarrow Q$. Se mostraran algunas de sus propiedades, además de la matriz asociada a la transformación respecto a la base canónica de $\Re^4$, que se trata de una matriz 4x4, si la transformación presentada se restringe a Q_v, es decir, $\rho(p,\bullet)$: $Q_v \rightarrow Q_v$, se podrá obtener fácilmente la matriz 3x3 que se utiliza comúnmente para realizar las rotaciones de cuerpo rígido. Definamos la siguiente transformación lineal $\rho(p,\bullet)$: $Q \rightarrow Q$, con $p \in Q$ fijo, como:

$$\rho(p,q) = p * q * p^{-1} = \frac{1}{\|p\|^2} \bullet (p * q * \bar{p}), \quad \forall q \in Q \quad , \qquad (2.13)$$

Algunas de las propiedades desarrolladas anteriormente son utilizadas en los teoremas siguientes referentes a la transformación anterior, la cual representa una *rotación*.

Teorema 2.2.1: La transformación $\rho(p,\bullet)$: $Q \rightarrow Q$ es **lineal**, **ortogonal** y $\rho(p,q) \in Q_V$, $\forall q \in Q_V$.

Demostración: Para demostrar al linealidad de la transformación, utilizamos las propiedades de asociatividad y distributividad de la estructura algebraica de **Q**.

$$\rho(\mathbf{p},(\mathbf{q}\oplus\mathbf{s}))=\frac{1}{\|\mathbf{p}\|^{2}}\bullet\{\mathbf{p}*(\mathbf{q}\oplus\mathbf{s})*\overline{\mathbf{p}}\}=\frac{1}{\|\mathbf{p}\|^{2}}\bullet\{\mathbf{p}*(\mathbf{q}*\overline{\mathbf{p}}\oplus\mathbf{s}*\overline{\mathbf{p}})\}=$$
$$\frac{1}{\|\mathbf{p}\|^{2}}\bullet\{\mathbf{p}*\mathbf{q}*\overline{\mathbf{p}}\oplus\mathbf{p}*\mathbf{s}*\overline{\mathbf{p}}\}=\rho(\mathbf{p},\mathbf{q})\oplus\rho(\mathbf{p},\mathbf{s}),\quad\forall\,\mathbf{p},\mathbf{q},\mathbf{s}\in\mathbf{Q}$$

y

$$\rho(\mathbf{p},(\alpha\bullet\mathbf{q}))=\frac{1}{\|\mathbf{p}\|^{2}}\bullet\{\mathbf{p}*(\alpha\bullet\mathbf{q})*\overline{\mathbf{p}}\}=\frac{\alpha}{\|\mathbf{p}\|^{2}}\{\mathbf{p}*\mathbf{q}*\overline{\mathbf{p}}\}=\alpha\bullet\rho(\mathbf{p},\mathbf{q}),\quad\forall\mathbf{p},\mathbf{q}\in\mathbf{Q},\quad\alpha\in\Re$$

Con lo anterior se demuestra la linealidad de la transformación, es decir que $\rho(\mathbf{p},\bullet)\in\mathbf{L}(\mathbf{Q},\mathbf{Q})$. Ahora para la demostración de la ortogonalidad de $\rho(\mathbf{p},\bullet)$ utilizamos el Teorema 2.1.2 y el Teorema 2.1.3. Para que la transformación sea ortogonal debe cumplirse que:

$$\langle\rho(\mathbf{p},\mathbf{q}),\rho(\mathbf{p},\mathbf{s})\rangle=\langle\mathbf{p},\mathbf{s}\rangle\qquad\forall\mathbf{q},\mathbf{s}\in\mathbf{Q}.$$

$$\begin{aligned}\langle\rho(\mathbf{p},\mathbf{q}),\rho(\mathbf{p},\mathbf{s})\rangle&=\frac{1}{2\|\mathbf{p}\|^{4}}\mathrm{T}_{\mathbf{R}}\{(\overline{\mathbf{p}*\mathbf{q}*\overline{\mathbf{p}}})*(\mathbf{p}*\mathbf{s}*\overline{\mathbf{p}})\oplus(\overline{\mathbf{p}*\mathbf{s}*\overline{\mathbf{p}}})*(\mathbf{p}*\mathbf{q}*\overline{\mathbf{p}})\}\\&=\frac{1}{2\|\mathbf{p}\|^{4}}\mathrm{T}_{\mathbf{R}}\{(\mathbf{p}*\overline{\mathbf{q}}*\overline{\mathbf{p}})*(\mathbf{p}*\mathbf{s}*\overline{\mathbf{p}})\oplus(\mathbf{p}*\overline{\mathbf{s}}*\overline{\mathbf{p}})*(\mathbf{p}*\mathbf{q}*\overline{\mathbf{p}})\}\\&=\frac{1}{2\|\mathbf{p}\|^{2}}\mathrm{T}_{\mathbf{R}}\{(\mathbf{p}*\overline{\mathbf{q}}*\mathbf{s}*\overline{\mathbf{p}})\oplus(\mathbf{p}*\overline{\mathbf{s}}*\mathbf{q}*\overline{\mathbf{p}})\}=\frac{1}{2\|\mathbf{p}\|^{2}}\mathrm{T}_{\mathbf{R}}\{(\mathbf{p}*(\overline{\mathbf{q}}*\mathbf{s}\oplus\overline{\mathbf{s}}*\mathbf{q})*\overline{\mathbf{p}})\}\\&=\frac{1}{2\|\mathbf{p}\|^{2}}\mathrm{T}_{\mathbf{R}}\{(\mathbf{p}*2\langle\mathbf{q},\mathbf{s}\rangle*\overline{\mathbf{p}})\}=\frac{2\langle\mathbf{q},\mathbf{s}\rangle}{2\|\mathbf{p}\|^{2}}\mathrm{T}_{\mathbf{R}}\{(\mathbf{p}*\overline{\mathbf{p}})\}=\langle\mathbf{q},\mathbf{s}\rangle\end{aligned}$$

Finalmente observemos que si $\mathbf{q}=\{0,\mathbf{q}_1,\mathbf{q}_2,\mathbf{q}_3\}\in\mathbf{Q}_V$, entonces $\rho(\mathbf{p},\mathbf{q})\in\mathbf{Q}_V$. En Efecto

```
In[1]:=
      (* Se define la transformación lineal ρ(p,q) y utilizando las definiciones hechas anteriormente
         en Mathematica *)

      RotacionQ[p_,q_]:= (1/np2)*ProductoQ[p,ProductoQ[q,ConjugadoQ[p]]];

      (* np2= norma de p al cuadrado *)

In[3]:=
      (* Se definen Quaterniones p y q *)

      p={p0,p1,p2,p3}; q={0,q1,q2,q3};

In[5]:=
      (* Se determina ρ(p,q) *)

      Collect[RotacionQ[p,q],{np2,q1,q2,q3}]
```

Out[6]=

```
        2      2      2      2
    (p0  + p1  - p2  - p3 )q1 + (2 p1 p2 - 2 p0 p3)q2 + (2 p0 p2 + 2 p1 p3)q3
{0, --------------------------------------------------------------------------,
                                   np2

                             2     2      2     2
 (2 p1 p2 + 2 p0 p3) q1 + (p0  - p1  + p2  - p3 ) q2 + (-2 p0 p1 + 2 p2 p3) q3
 ----------------------------------------------------------------------------,
                                   np2

                                                        2      2      2      2
 (-2 p0 p2 + 2 p1 p3) q1 + (2 p0 p1 + 2 p2 p3) q2 + (p0  - p1  - p2  + p3 ) q3
 ----------------------------------------------------------------------------}
                                   np2
```

■

Teorema 2.2.2: La transformación ρ**(p,•)** es una **rotación.**

Demostración: Algunas de las propiedades necesarias para que la transformación sea una rotación se han demostrado ya en el Teorema 2.2.1, por ello solo resta demostrar que el determinante de la matriz asociada a la transformación $M_{\rho(p,\bullet)} \in M_{4\times4}$ es **+1**. En efecto,

In[1]:=

```
(* Se define el Quaternión p *)

p = {p0,p1,p2,p3};

(* se define la base canónica de R^4 *)

e1 = {1,0,0,0}; e2 = {0,1,0,0}; e3 = {0,0,1,0}; e4 = {0,0,0,1};
```

In[5]:=

```
(* Se determina la matriz de la transformación (MRot) *)

(* Se inicializa la lista Mrot donde se almacenará la matriz *)

MRot={{},{},{},{}};

(* Se aplica la rotación a la base canónica *)

e11 = RotacionQ[p,e1];
e21 = RotacionQ[p,e2];
e31 = RotacionQ[p,e3];
e41 = RotacionQ[p,e4];

(* Se determinan los elementos de la matriz *)

MRot = Insert[MRot,e11[[1]],{1,1}];
MRot = Insert[MRot,e11[[2]],{2,1}];
MRot = Insert[MRot,e11[[3]],{3,1}];
MRot = Insert[MRot,e11[[4]],{4,1}];

MRot = Insert[MRot,e21[[1]],{1,2}];
```

```
MRot = Insert[MRot,e21[[2]],{2,2}];
MRot = Insert[MRot,e21[[3]],{3,2}];
MRot = Insert[MRot,e21[[4]],{4,2}];

MRot = Insert[MRot,e31[[1]],{1,3}];
MRot = Insert[MRot,e31[[2]],{2,3}];
MRot = Insert[MRot,e31[[3]],{3,3}];
MRot = Insert[MRot,e31[[4]],{4,3}];

MRot = Insert[MRot,e41[[1]],{1,4}];
MRot = Insert[MRot,e41[[2]],{2,4}];
MRot = Insert[MRot,e41[[3]],{3,4}];
MRot = Insert[MRot,e41[[4]],{4,4}];

(* Se sutituye (p0^2+p1^2+p2^2+p3^2) por np2 *)

MRot = MRot//.{p0^2+p1^2+p2^2+p3^2->np2};

(* Se imprime la Matriz de la Tranformación Rotación *)

MatrixForm[MRot]
```

Out[33]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \dfrac{p0^2 + p1^2 - p2^2 - p3^2}{np2} & \dfrac{2\ p1\ p2 - 2\ p0\ p3}{np2} & \dfrac{2\ p0\ p2 + 2\ p1\ p3}{np2} \\ 0 & \dfrac{2\ p1\ p2 + 2\ p0\ p3}{np2} & \dfrac{p0^2 - p1^2 + p2^2 - p3^2}{np2} & \dfrac{-2\ p0\ p1 + 2\ p2\ p3}{np2} \\ 0 & \dfrac{-2\ p0\ p2 + 2\ p1\ p3}{np2} & \dfrac{2\ p0\ p1 + 2\ p2\ p3}{np2} & \dfrac{p0^2 - p1^2 - p2^2 + p3^2}{np2} \end{pmatrix}$$

In[34]:=

```
(* Se calcula el determinante de la matriz (simplificando)*)

Determinante = Det[MRot]//Simplify
```

Out[35]=

$$\frac{(p0^2 + p1^2 + p2^2 + p3^2)^3}{np2^3}$$

In[36]:=

```
(* Se sustituye (p0^2+p1^2+p2^2+p3^2)^3 por np2^3 en el resultado anterior *)
```

```
Determinante//.{(p0^2+p1^2+p2^2+p3^2)^3->np2^3}
Out[37]=
  1
```

Se observa que efectivamente se cumple que $\mathrm{Det}[M_{\rho(p,\bullet)}] = +1$.

■

2.2.1. Significado de los parámetros de la Transformación Rotación $\rho(p,\bullet)$.

En esta sección se demostrará cual es el significado de los parámetros de la Transformación Rotación ($\rho(p,\bullet)$) analizada hasta ahora. Para ello tenemos lo siguiente:

Sean $\theta \in [0,n\pi]$, $n \in \Re$, $\underline{w} \in \Re^3$, $\|\underline{w}\| = 1$ Siendo $p \in Q$ tal que $\rho(p, \underline{w}) = \underline{w}$

Tenemos también que el espacio característico Q_ε de valor característico $\lambda = 1$ se define por, [1] :

$$Q_\varepsilon = \{ q \in Q : \rho(p,q) = q \} \qquad (2.14)$$

y el complemento ortogonal de Q_ε por:

$$Q_\varepsilon^\perp = \{ p \in Q : \langle p,q \rangle = 0 , \ \forall q \in Q_\varepsilon \} \qquad (2.15)$$

En base a lo anterior podemos obtener los resultados siguientes.

Teorema 2.2.3: Sea $u = \{u_0, u_1, u_2, u_3\} \in Q_\varepsilon^\perp$, $\|u\| = 1$, entonces:

$$\mathrm{Cos}\,\theta = \frac{\langle u,\rho(p,u)\rangle}{\|u\|^2} = \frac{1}{\|p\|^2}\langle u,\ p * u * \overline{p}\rangle = \frac{1}{\|p\|^2}\left(p_0^2 + \|p_v\|^2 (2u_0^2 - 1)\right) \qquad (2.16)$$

$$\mathrm{Sen}^2\,\theta = (\pm)\frac{4\|p_v\|^2}{\|p\|^4}(1 - u_0^2)(p_0^2 + u_0^2\|p_v\|^2) \qquad (2.17)$$

Demostración: En efecto,

```
In[1]:=
      (* Definición de la Transformación Rotación *)

      (* Se define la Transformación de un Quaternión a un vector tridimensional Tv[] *)
      Tv[q_]:={q[[2]],q[[3]],q[[4]]};

      (* Se define la Transformación inversa de Tv[] *)
      TvInv[v_]:={0,v[[1]],v[[2]],v[[3]]};


      (* Se define la Transformación de un Quaternión a Quaternión vectorial *)
      TvQ[q_]:={0,q[[2]],q[[3]],q[[4]]};

      (* Se define la Multiplicación entre dos Quaterniones *)
      ProductoQ[p_,q_]:={p[[1]]*q[[1]] - p[[2]]*q[[2]] - p[[3]]*q[[3]] - p[[4]]*q[[4]],
                          p[[1]]*q[[2]] + p[[2]]*q[[1]] + p[[3]]*q[[4]] - p[[4]]*q[[3]],
                          p[[1]]*q[[3]] - p[[2]]*q[[4]] + p[[3]]*q[[1]] + p[[4]]*q[[2]],
                          p[[1]]*q[[4]] + p[[2]]*q[[3]] - p[[3]]*q[[2]] + p[[4]]*q[[1]]};

      (* Se define el conjugado de un Quaternión *)
      ConjugadoQ[p_]:={p[[1]],-p[[2]],-p[[3]],-p[[4]]};

      (* Se define la Transformación Rotación de un vector tridimensional, considerando norma
         unitaria de "p", si se desea aplicar otra norma debera incluirse el producto 1/np^2 *)

      Rotacion[p_,v_]:= Tv[ProductoQ[p,ProductoQ[TvInv[v],ConjugadoQ[p]]]];

      (* Se define la Transformación Reflexión de un Quaternión, también con norma unitaria
         de "p" *)

      RotacionQ[p_,q_]:= RotacionQ[p_,q_]:= ProductoQ[p,ProductoQ[q,ConjugadoQ[p]]];

In[16]:=

      (* Se limpian variables *)

      ClearAll["p*","u*","n*",T]

      (* Se definen los Quaterniones p y u *)

      p={p0,p1,p2,p3}; u={u0,u1,u2,u3};

      (* Se determina el "Coseno de Theta" = coseno *)

      coseno = u.RotacionQ[p,u]//Expand;

      (* Se aplica la ortogonalidad entre p y u, es decir el producto  interno de sus partes
         vectoriales es igual a cero *)

      coseno = coseno//.4*p1*p2*u1*u2-> - 2*p1^2*u1^2 - 2*p2^2*u2^2 +
                        - 2*p3^2*u3^2 - 4*p1*p3*u1*u3 - 4*p2*p3*u2*u3;
```

```
(* Se usa la norma unitaria de u *)

coseno = Collect[coseno//.{u3^2->1-u0^2-u1^2-u2^2},{u0}];

(* Se sustituye la norma al cuadrado de pv (parte vectorial del Quaternión p) por la
   variable nPv^2*)

coseno = coseno//.{2*p1^2+2*p2^2+2*p3^2->2*nPv^2,-p1^2-p2^2-p3^2->-nPv^2};

(* Como no se trata de norma unitaria de p, se incluye el cuadrado de la norma de p
   como nP^2 *)

coseno = Collect[coseno,{nPv}]/nP^2
Out[29]=

  2        2              2
p0  + nPv   (-1 + 2 u0 )
------------------------
              2
            nP
In[30]:=
(* Ahora se determina el "seno cuadrado de Theta" = seno2 utilizando la identidad de
   Cos^2(T) + Sen^2(T) = 1 *)

seno2 = 1-coseno^2//Expand;

(* Sustituimos nP^2 por nP2 y npv^2 por npv2 *)

seno2 = seno2//.{nP->nP2^(1/2),nPv->nPv2^(1/2)}//Expand;

(* Se aplica la norma de "p", nP^2 = p0^2 + nPv^2 *)

seno2 = (seno2//.{nPv2->nP2-p0^2}//Expand//Factor)*nP2^2;

seno2 = (seno2//.nP2->nPv2+p0^2//Expand//Factor)/nP2^2;

(* Sustituimos nP2 por nP^2 y npv por npv^2 *)

seno2 = seno2//.{nP2->nP^2,nPv2->nPv^2}
Out[38]=

       2                          2      2   2
4 nPv   (1 - u0) (1 + u0) (p0  + nPv  u0 )
-----------------------------------------
                          4
                        nP
```

Se puede observar que las salidas Out[29] y Out[38] son iguales a lo que se quería demostrar en este teorema.

■

Teorema 2.2.4: Sea $u = (0, u_1, u_2, u_3) \in Q_\varepsilon^-$, entonces:

$$u \times R(p,u) = \frac{2p_0\|u\|^2}{\|p\|^2}(0,p_1,p_2,p_3) = 2p_0\frac{\|u\|^2}{\|p\|^2}p_v \qquad (2.18)$$

Demostración: En efecto,

In[1]:=

```
(* Se define la transformación Tv[], que transforma un Quaternión en un vector
   tridimensional *)

Tv[q_]:={q[[2]],q[[3]],q[[4]]};

(* Se define el Producto Cruz entre dos vectores tridimensionales *)

Cruz[a_,b_]:= {a[[2]] b[[3]] - a[[3]] b[[2]],  - a[[1]] b[[3]] + a[[3]] b[[1]],  a[[1]] b[[2]] - a[[2]] b[[1]]}
```

In[4]:=

```
(* Se definen los Quaterniones u y p *)

u={0,u1,u2,u3}; u={0,u1,u2,u3};

(* Se realiza el producto Cruz de la partes vectoriales de u y R(p,u), incluyendose la norma de
   "p", que no es unitaria (1/nP^2) *)

uXRu = Cruz[Tv[u],(1/nP^2)*Tv[RotacionQ[p,u]]]//Expand;

(* Se aplica la definición de norma de u y la propiedad de que el  producto interno de u y p
   es igual a cero por ser ortogonales *)

uXRu = uXRu//.{u3^2->nu^2-u1^2-u2^2,p1*u1->-p3*u3-p2*u2}//Expand;
uXRu = uXRu//.{u3^2->nu^2-u1^2-u2^2,p3*u3->-p1*u1-p2*u2}//Expand;
uXRu = uXRu//.{p2*u2->-p1*u1-p3*u3}//Expand;
uXRu = uXRu//.{u1^2->nu^2-u3^2-u2^2}//Expand
```

Out[12]=

```
         2                2                2
   2 nu   p0 p1     2 nu   p0 p2     2 nu   p0 p3
 {------------,   ------------,   ------------}
          2                2                2
        nP               nP               nP
```

El resultado obtenido en Mathematica® se puede representar también como:

$$\left(\frac{2\|u\|^2 p_0p_1}{\|p\|^2},\frac{2\|u\|^2 p_0p_2}{\|p\|^2},\frac{2\|u\|^2 p_0p_3}{\|p\|^2}\right) = \frac{2\|u\|^2 p_0}{\|p\|^2}(p_1,p_2,p_3) = \frac{2\|u\|^2 p_0}{\|p\|^2}p_v \;=\; u \times R(p,u)$$

Con lo anterior se demuestra la validez de este teorema.

■

Teorema 2.2.5: Sea $\mathbf{u} = \{0, u_1, u_2, u_3\} \in Q_v$, $\|\mathbf{u}\| = 1$, entonces:

$$\mathbf{Cos}\ \theta = \frac{1}{\|\mathbf{p}\|^2}\left(p_0^2 - \|\mathbf{p}_v\|^2\right) \ ; \qquad \mathbf{Sen}\ \theta = (\pm)\frac{2\,\|\mathbf{p}_v\|}{\|\mathbf{p}\|^2}\mathbf{p}_0 \qquad (2.19)$$

$$\mathbf{u} \times \rho(\mathbf{p},\mathbf{u}) = (\pm)\,\mathbf{Sen}\ \theta\ \underline{\mathbf{w}} \qquad (2.20)$$

Donde $\underline{\mathbf{w}} = \frac{1}{\|\mathbf{p}_v\|}\mathbf{p}_v$.

Demostración: En efecto, si tomamos del Teorema 2.2.3 **Cos** θ, **Sen** θ y sustituimos $u_0 = 0$, obtenemos directamente lo mostrado para estas dos variables en el teorema. Para el producto cruz $\mathbf{u}\times\rho(\mathbf{p},\mathbf{u})$, del Teorema 2.2.4 tenemos que:

$$\mathbf{u} \times \rho(\mathbf{p},\mathbf{u}) = \frac{2p_0\|\mathbf{u}\|^2}{\|\mathbf{p}\|^2}(0, p_1, p_2, p_3) = 2p_0\frac{\|\mathbf{u}\|^2}{\|\mathbf{p}\|^2}\mathbf{p}_v$$

de este teorema: $\mathbf{Sen}\ \theta = (\pm)\frac{2\,\|\mathbf{p}_v\|}{\|\mathbf{p}\|^2}p_0 \quad \Rightarrow \quad p_0 = (\pm)\frac{\|\mathbf{p}\|^2}{2\,\|\mathbf{p}_v\|}\mathbf{Sen}\ \theta$

Sustituyendo p_0 y $\|\mathbf{u}\| = 1$ en la expresión del producto $\mathbf{u}\times\rho(\mathbf{p},\mathbf{u})$:

$$\mathbf{u} \times \rho(\mathbf{p},\mathbf{u}) = 2\left((\pm)\frac{\|\mathbf{p}\|^2}{2\|\mathbf{p}_v\|}\mathbf{Sen}\ \theta\right)\frac{\|\mathbf{u}\|^2}{\|\mathbf{p}\|^2}\mathbf{p}_v = (\pm)\frac{\mathbf{p}_v}{\|\mathbf{p}_v\|}\mathbf{Sen}\ \theta$$

Como $\mathbf{u}\times\rho(\mathbf{p},\mathbf{u})$ da como resultado un vector perpendicular al plano formado por **u** y $\rho(\mathbf{p},\mathbf{u})$, que es la dirección del eje de rotación, podemos definir como eje de rotación al vector $\underline{\mathbf{w}} = \frac{\mathbf{p}_v}{\|\mathbf{p}_v\|}$, por lo que:

$$\mathbf{u} \times \rho(\mathbf{p},\mathbf{u}) = (\pm)\mathbf{Sen}\ \theta\ \underline{\mathbf{w}}$$

■

Teorema 2.2.6: Sea $\theta \in [0, n\pi]$, $n \in \Re$, $\underline{\mathbf{w}} \in Q_v$, $\|\underline{\mathbf{w}}\| = 1$, entonces la parte vectorial del Quaternión $\mathbf{p} = \{p_0, p_1, p_2, p_3\} \in Q$ con $p_0 \in \Re - \{0\}$ es:

$$\mathbf{p}_v = \{p_1, p_2, p_3\} = (\pm)\frac{\|\mathbf{p}\|^2}{2p_0}\mathbf{Sen}\ \theta\ \underline{\mathbf{w}}\ ; \qquad (2.21)$$

siendo $\underline{w}$ tal que: $\rho(p, \underline{w}) = \underline{w}$
También $p_0 \in \Re - \{0\}$ satisface la siguiente ecuación:

$$4\,p_0^4 - 4\,\|p\|^2 p_0^2 \operatorname{Cos}\theta - \|p\|^4 \operatorname{Sen}^2\theta = 0 \qquad (2.22)$$

Demostración: Del Teorema 2.2.4 se tiene que:

$$u \times \rho(p,u) = (\pm)\operatorname{Sen}\theta\,\underline{w} = \frac{2p_0}{\|p\|^2}p_v$$

Despejando p_v tenemos que:

$$(\pm)\operatorname{Sen}\theta\,\underline{w} = \frac{2p_0}{\|p\|^2}p_v \qquad \Rightarrow \qquad p_v = (\pm)\frac{\|p\|^2}{2p_0}\operatorname{Sen}\theta\,\underline{w}$$

con esto se demuestra la primera parte del teorema. Para la segunda parte tenemos del Teorema 2.2.5 que:

$$\operatorname{Cos}\theta = \frac{1}{\|p\|^2}(\,p_0^2 - \|p_v\|^2) \;; \qquad \operatorname{Sen}\theta = (\pm)\frac{2\,\|p_v\|}{\|p\|^2}p_0$$

Despejando de **Sen** θ a $\|p_v\|$ y de **Cos** θ a p_0^2, tenemos que:

$$\|p_v\| = (\pm)\,\|p\|^2\,\frac{\operatorname{Sen}\theta}{2\,p_0} \;; \qquad p_0^2 = \|p\|^2\operatorname{Cos}\theta + \|p_v\|^2$$

combinando ambos resultados:

$$p_0^2 = \|p\|^2\operatorname{Cos}\theta + (\,(\pm)\,\|p\|^2\,\frac{\operatorname{Sen}\theta}{2\,p_0}\,)^2 = \|p\|^2\operatorname{Cos}\theta + \|p\|^4\,\frac{\operatorname{Sen}^2\theta}{4\,p_0^2}$$

$$p_0^2 - \|p\|^2\operatorname{Cos}\theta - \|p\|^4\,\frac{\operatorname{Sen}^2\theta}{4\,p_0^2} = 0$$

$$4\,p_0^4 - 4\,\|p\|^2 p_0^2 \operatorname{Cos}\theta - \|p\|^4 \operatorname{Sen}^2\theta = 0$$

Con esto se demuestra la segunda parte del teorema.

■

Teorema 2.2.7: Las raíces reales de la ecuación de cuarto grado son:

$$4\,p_0^4 - 4\,\|p\|^2 p_0^2 \operatorname{Cos}\theta - \|p\|^4 \operatorname{Sen}^2\theta = 0 \qquad (2.23)$$

$$p_{0_1} = \|p\| \operatorname{Cos}\frac{\theta}{2}, \qquad p_{0_2} = -\|p\| \operatorname{Cos}\frac{\theta}{2} \qquad (2.24)$$

Demostración: En efecto,

In[1]:=

(* Se resuelve la ecuación de cuarto grado y se simplifica la solución *)

```
Solve[4*p0^4 - 4*nP^2*p0^2*Cos[T] - nP^4*(Sin[T])^2 == 0,{p0}]//Simplify//PowerExpand
      //Simplify//PowerExpand
```

Out[2]=

```
                  T                     T
{{p0 -> -I nP Sin[-]}, {p0 -> I nP Sin[-]},
                  2                     2

                 T                     T
 {p0 -> -(nP Cos[-])}, {p0 -> nP Cos[-]}}
                 2                     2
```

Se puede observar que las soluciones reales de la ecuación de cuarto grado, son las presentadas en el teorema.

■

Finalmente obtenemos con una combinación de los teoremas anteriores:

$$p_0 = (\pm)\,\|p\| \operatorname{Cos}\frac{\theta}{2} \qquad (2.25)$$

$$p_v = (p_1, p_2, p_3) = (\pm)\frac{\|p\|^2}{2\|p\| \operatorname{Cos}\frac{\theta}{2}} \operatorname{Sen}\theta\, \underline{w} = (\pm)\,\|p\| \operatorname{Sin}\frac{\theta}{2}\, \underline{w} \qquad (2.26)$$

Propiedades adicionales del álgebra de Quaterniones y de la representación paramétrica de rotaciones se encuentran en [1].[2].

2.3. Representación paramétrica de reflexiones.

De la misma forma que en la sección anterior, aquí presentaremos la definición y propiedades de una transformación lineal que representará paramétricamente una reflexión de cuerpo rígido. Entendiéndose como reflexión a una transformación lineal, ortogonal de determinante negativo. Lo

anterior se mostrará detalladamente mas adelante, cuando se presente el significado de los parámetros de la reflexión.
De la misma forma que para las rotaciones, tenemos la transformación lineal **R(p,•): Q→ Q**, con **p** ∈ **Q** fijo, como:

$$\mathbf{R}(\mathbf{p},\mathbf{q}) = \overline{\mathbf{p} * \mathbf{q} * \mathbf{p}^{-1}} = \frac{1}{\|\mathbf{p}\|^2} \bullet \left(\overline{\mathbf{p} * \mathbf{q} * \overline{\mathbf{p}}}\right), \quad \forall \mathbf{q} \in \mathbf{Q}, \tag{2.27}$$

Podemos observar que se trata del conjugado de la transformación *Rotación*. De esta transformación se desprenden los siguientes teoremas.

Teorema 2.3.1: La transformación **R(p,•): Q→ Q es lineal**, **ortogonal** y **R(p,q)** ∈ $\mathbf{Q}_V$, ∀**q** ∈ $\mathbf{Q}_V$.

Demostración: Para demostrar al linealidad de la transformación, utilizamos las propiedades de asociatividad y distributividad de la estructura algebraica de **Q**.

$$\mathbf{R}(\mathbf{p},(\mathbf{q}\oplus\mathbf{s})) = \frac{1}{\|\mathbf{p}\|^2} \bullet \left\{\overline{\mathbf{p} * (\mathbf{q}\oplus\mathbf{s}) * \overline{\mathbf{p}}}\right\} = \frac{1}{\|\mathbf{p}\|^2} \bullet \left\{\overline{\mathbf{p} * (\mathbf{q} * \overline{\mathbf{p}} \oplus \mathbf{s} * \overline{\mathbf{p}})}\right\} =$$

$$\frac{1}{\|\mathbf{p}\|^2} \bullet \left\{\overline{\mathbf{p} * \mathbf{q} * \overline{\mathbf{p}} \oplus \mathbf{p} * \mathbf{s} * \overline{\mathbf{p}}}\right\} = \frac{1}{\|\mathbf{p}\|^2} \bullet \left\{\overline{\mathbf{p} * \mathbf{q} * \overline{\mathbf{p}}} \oplus \overline{\mathbf{p} * \mathbf{s} * \overline{\mathbf{p}}}\right\} =$$

$$\mathbf{R}(\mathbf{p},\mathbf{q}) \oplus \mathbf{R}(\mathbf{p},\mathbf{s}), \quad \forall\, \mathbf{p},\mathbf{q},\mathbf{s} \in \mathbf{Q}$$

y

$$\mathbf{R}(\mathbf{p},(\alpha \bullet \mathbf{q})) = \frac{1}{\|\mathbf{p}\|^2} \bullet \left\{\overline{\mathbf{p} * (\alpha \bullet \mathbf{q}) * \overline{\mathbf{p}}}\right\} = \frac{\alpha}{\|\mathbf{p}\|^2} \left\{\overline{\mathbf{p} * \mathbf{q} * \overline{\mathbf{p}}}\right\} = \alpha \bullet \mathbf{R}(\mathbf{p},\mathbf{q}), \quad \forall \mathbf{p},\mathbf{q} \in \mathbf{Q}, \quad \alpha \in \Re$$

Con lo anterior se demuestra la linealidad de la transformación, es decir que **R(p,•)** ∈ **L(Q, Q)**. Ahora para la demostración de la ortogonalidad de R(p,•) utilizamos el Teorema 2.1.2 y el Teorema 2.1.3. Ahora para que la transformación sea ortogonal debe cumplirse que:

$$\langle \mathbf{R}(\mathbf{p},\mathbf{q}), \mathbf{R}(\mathbf{p},\mathbf{s}) \rangle = \langle \mathbf{p},\mathbf{s} \rangle \qquad \forall \mathbf{q},\mathbf{s} \in \mathbf{Q}.$$

$$
\begin{aligned}
\langle R(p,q), R(p,s)\rangle &= \frac{1}{2\|p\|^4} T_R\{\overline{\overline{(p * q * \bar{p})}} * \overline{(p * s * \bar{p})} \oplus \overline{\overline{(p * s * \bar{p})}} * \overline{(p * q * \bar{p})}\} \\
&= \frac{1}{2\|p\|^4} T_R\{(p * q * \bar{p}) * \overline{(p * s * \bar{p})} \oplus (p * s * \bar{p}) * \overline{(p * q * \bar{p})}\} \\
&= \frac{1}{2\|p\|^4} T_R\{(p * q * \bar{p}) * (p * \bar{s} * \bar{p}) \oplus (p * s * \bar{p}) * (p * \bar{q} * \bar{p})\} \\
&= \frac{1}{2\|p\|^2} T_R\{(p * q * \bar{s} * \bar{p}) \ominus (p * s * \bar{q} * \bar{p})\} = \frac{1}{2\|p\|^2} T_R\{(p * (q * \bar{s} \oplus s * \bar{q}) * \bar{p})\} \\
&= \frac{1}{2\|p\|^2} T_R\{(p * 2\langle q,s\rangle * \bar{p})\} = \frac{2\langle q,s\rangle}{2\|p\|^2} T_R\{(p * \bar{p})\} = \langle q,s\rangle
\end{aligned}
$$

Finalmente observemos que si $\mathbf{q} = \{0, q_1, q_2, q_3\} \in Q_V$, entonces $\mathbf{R(p,q)} \in Q_V$. En efecto

In[1]:=

```
(* Se define la transformación reflexión *)

ReflexionQ[p_,q_]:= (1/np2)*ProductoQ[p,ProductoQ[ConjugadoQ[q],ConjugadoQ[p]]];

(* np2= norma de p al cuadrado *)
```

In[3]:=

```
(* Se definen Quaterniones p y q *)

p={p0,p1,p2,p3};  q={0,q1,q2,q3};
```

In[5]:=

```
(* Se determina R(p,q) *)

Collect[ReflexionQ[p,q],{np2,q1,q2,q3}]//InputForm
```

Out[6]=

```
              2      2      2     2
        (-p0  - p1  + p2  + p3 )q1 +(-2 p1 p2 + 2 p0 p3)q2 +(-2 p0 p2 - 2 p1 p3)q3
{0, ---------------------------------------------------------------------------,
                                       np2

                               2      2      2     2
 (-2 p1 p2 - 2 p0 p3) q1 + (-p0  + p1  - p2  + p3 ) q2 + (2 p0 p1 - 2 p2 p3) q3
 ------------------------------------------------------------------------------,
                                     np2

                                                          2      2      2     2
 (2 p0 p2 - 2 p1 p3) q1 + (-2 p0 p1 - 2 p2 p3) q2 + (-p0  + p1  + p2  - p3 ) q3
 ------------------------------------------------------------------------------}
                                     np2
```

■

Teorema 2.3.2: La transformación **R(p,•)** es una **reflexión.**

Demostración: Algunas de las propiedades necesarias para que la transformación sea una reflexión se han demostrado ya en el Teorema 2.2.1, por ello solo resta demostrar que el determinante de la matriz asociada a la transformación $M_{R(p,\circ)} \in M_{4\times 4}$ es -1. En efecto,

```
In[1]:=
 (* Se define el Quaternión p *)

 p = {p0,p1,p2,p3};

 (* se define la base canónica de R^4 *)

 e1 = {1,0,0,0}; e2 = {0,1,0,0}; e3 = {0,0,1,0}; e4 = {0,0,0,1};

In[5]:=
 (* Se determina la matriz de la transformación (MRef) *)

 (* Se inicializa la lista Mref *)

 MRef={{},{},{},{}};

 (* Se aplica la rotación a la base canónica *)

 e11 = ReflexionQ[p,e1];
 e21 = ReflexionQ[p,e2];
 e31 = ReflexionQ[p,e3];
 e41 = ReflexionQ[p,e4];

 (* Se determinan los elementos de la matriz *)

 MRef = Insert[MRef,e11[[1]],{1,1}];
 MRef = Insert[MRef,e11[[2]],{2,1}];
 MRef = Insert[MRef,e11[[3]],{3,1}];
 MRef = Insert[MRef,e11[[4]],{4,1}];

 MRef = Insert[MRef,e21[[1]],{1,2}];
 MRef = Insert[MRef,e21[[2]],{2,2}];
 MRef = Insert[MRef,e21[[3]],{3,2}];
 MRef = Insert[MRef,e21[[4]],{4,2}];

 MRef = Insert[MRef,e31[[1]],{1,3}];
 MRef = Insert[MRef,e31[[2]],{2,3}];
 MRef = Insert[MRef,e31[[3]],{3,3}];
 MRef = Insert[MRef,e31[[4]],{4,3}];

 MRef = Insert[MRef,e41[[1]],{1,4}];
 MRef = Insert[MRef,e41[[2]],{2,4}];
 MRef = Insert[MRef,e41[[3]],{3,4}];
 MRef = Insert[MRef,e41[[4]],{4,4}];

 (* Se sutituye (p0^2+p1^2+p2^2+p3^2) por np2 *)

 MRef = MRef//.{p0^2+p1^2+p2^2+p3^2->np2};
```

```
(* Se imprime la Matriz de la Tranformación Reflexion *)

MatrixForm[MRef]
```

Out[33]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{-p0^2 - p1^2 + p2^2 + p3^2}{np2} & \frac{-2\,p1\,p2 + 2\,p0\,p3}{np2} & \frac{-2\,p0\,p2 - 2\,p1\,p3}{np2} \\ 0 & \frac{-2\,p1\,p2 - 2\,p0\,p3}{np2} & \frac{-p0^2 + p1^2 - p2^2 + p3^2}{np2} & \frac{2\,p0\,p1 - 2\,p2\,p3}{np2} \\ 0 & \frac{2\,p0\,p2 - 2\,p1\,p3}{np2} & \frac{-2\,p0\,p1 - 2\,p2\,p3}{np2} & \frac{-p0^2 + p1^2 + p2^2 - p3^2}{np2} \end{pmatrix}$$

In[34]:=

(* Se calcula el determinante de la matriz (simplificando)*)

Determinante = Det[MRef]//Simplify

Out[35]=

$$-\left(\frac{(p0^2 + p1^2 + p2^2 + p3^2)^3}{np2^3}\right)$$

In[36]:=

(* Se sustituye (p0^2+p1^2+p2^2+p3^2)^3 por np2^3 en el resultado anterior *)

Determinante//.{(p0^2+p1^2+p2^2+p3^2)^3->np2^3}

Out[37]=

-1

Se observa que efectivamente se cumple que **Det[$M_{R(p,\bullet)}$] = -1.**

■

2.3.1. Significado de los parámetros de la Transformación Reflexión R(p,•).

De la misma forma que se demostró el significado de los parámetros de la Transformación Rotación (ρ(p,•)) en [1], ahora se determinará el significado de los parámetros de la Transformación Reflexión (R(p,•)) mostrada anteriormente. Para ello tenemos lo siguiente:

Sean $\theta \in [0,n\pi]$, $n \in \Re$, $\underline{w} \in \Re^3$, $\|\underline{w}\| = 1$ Siendo $p \in Q$ tal que $R(p, \underline{w}) = -\underline{w}$

Tenemos también que el espacio característico Q_ε de valor característico $\lambda = -1$ se define por, [1]:

$$Q_\varepsilon = \{ q \in Q : R(p,q) = \bar{q} \} \qquad (2.28)$$

y el complemento ortogonal de Q_ε por:

$$Q_\varepsilon^\perp = \{ p \in Q : \langle p,q \rangle = 0 , \quad \forall q \in Q_\varepsilon \} \qquad (2.29)$$

En base a lo anterior podemos obtener los resultados siguientes.

Teorema 2.3.3: Sea $u = (u_0, u_1, u_2, u_3) \in Q_\varepsilon^\perp$, $\|u\| = 1$, entonces:

$$\mathrm{Cos}\,\theta = \frac{\langle u, R(p,u) \rangle}{\|u\|^2} = \frac{1}{\|p\|^2} \langle u, \overline{p * u * \bar{p}} \rangle = \frac{1}{\|p\|^2} (\|p_v\|^2 + p_0^2(2u_0^2 - 1)) \qquad (2.30)$$

$$\mathrm{Sen}^2\,\theta = (\pm) \frac{4p_0^2}{\|p\|^4} (1 - u_0^2)(\|p_v\|^2 + p_0^2 u_0^2) \qquad (2.31)$$

Demostración: En efecto,

```
In[1]:=

(* Definición de la Transformación Reflexión *)

(* Se define la Transformación de un Quaternión a un vector Tv[] *)
Tv[q_]:={q[[2]],q[[3]],q[[4]]};

(* Se define la Transformación inversa de Tv[] *)
TvInv[v_]:={0,v[[1]],v[[2]],v[[3]]};

(* Se define la Transformación de un Quaternión a Quaternión vectorial *)
TvQ[q_]:={0,q[[2]],q[[3]],q[[4]]};

(* Se define la Multiplicación entre dos Quaterniones *)

ProductoQ[p_,q_]:={p[[1]]*q[[1]] - p[[2]]*q[[2]] - p[[3]]*q[[3]] - p[[4]]*q[[4]],
                   p[[1]]*q[[2]] + p[[2]]*q[[1]] + p[[3]]*q[[4]] - p[[4]]*q[[3]],
                   p[[1]]*q[[3]] -  p[[2]]*q[[4]] + p[[3]]*q[[1]] + p[[4]]*q[[2]],
                   p[[1]]*q[[4]] + p[[2]]*q[[3]] - p[[3]]*q[[2]] +  p[[4]]*q[[1]]};

(* Se define el conjugado de un Quaternión *)
ConjugadoQ[p_]:={p[[1]],-p[[2]],-p[[3]],-p[[4]]};

(* Se define la Transformación Reflexión de un vector tridimensional, considerando norma
   unitaria de "p", si se desea aplicar otra norma debera incluirse el producto 1/np^2 *)

Reflexion[p_,q_]:=Tv[ConjugadoQ[ProductoQ[p,ProductoQ[TvInv[q],ConjugadoQ[p]]]] ];
```

```
(* Se define la Transformación Reflexión de un Quaternión, también con norma unitaria
   de "p" *)

ReflexionQ[p_,q_]:= ConjugadoQ[ProductoQ[p,ProductoQ[q,ConjugadoQ[p]]]];
```

In[16]:=

```
(* Se limpian variables *)

ClearAll["p*","u*","n*",T]

(* Se definen los Quaterniones p y u *)

p={p0,p1,p2,p3}; u={u0,u1,u2,u3};

(* Se determina el Coseno de Theta = coseno *)

coseno = u.ReflexionQ[p,u]//Expand;

(* Se aplica la ortogonalidad entre p y u, es decir el producto interno de sus partes
   vectoriales  es igual a cero *)

coseno = coseno//.-4*p1*p2*u1*u2-> 2*p1^2*u1^2 + 2*p2^2*u2^2 +
                  2*p3^2*u3^2 + 4*p1*p3*u1*u3 + 4*p2*p3*u2*u3;

(* Se usa la norma unitaria de u *)

coseno = Collect[coseno//.{u3^2->1-u0^2-u1^2-u2^2},{u0}];

(* Se sustituye la norma al cuadrado de pv (parte vectorial del Quaternión p) por la
   variable nPv^2*)

coseno = coseno//.{p1^2+p2^2+p3^2->nPv^2};

(* Como no se trata de norma unitaria de p, se incluye el cuadrado de la norma de p como
   nP^2 *)

coseno = Collect[coseno,{p0}]/nP^2
```

Out[29]=

```
   2       2             2
nPv  + p0  (-1 + 2 u0 )
------------------------
               2
             nP
```

In[30]:=

```
(* Ahora se determina el "seno cuadrado de Theta" = seno2 utilizando la identidad de
   Cos^2(T) + Sen^2(T) = 1 *)

seno2 = 1-coseno^2//Expand;

(* Sustituimos nP^2 por nP2 y npv^2 por npv2 *)
```

```
seno2 = seno2//.{nP->nP2^(1/2),nPv->nPv2^(1/2)}//Expand;

(* Se aplica la norma de "p", nP^2 = p0^2 + nPv^2 *)

seno2 = (seno2//.{nPv2->nP2-p0^2}//Expand//Factor)*nP2^2;
seno2 = (seno2//.nP2->nPv2+p0^2//Expand//Factor)/nP2^2;

(* Sustituimos nP2 por nP^2 y npv por npv^2 *)

seno2 = seno2//.{nP2->nP^2,nPv2->nPv^2}
```

Out[38]=

```
      2                          2      2    2
4 p0   (1 - u0) (1 + u0) (nPv   + p0   u0 )
----------------------------------------
                      4
                    nP
```

Se puede observar que las salidas Out[29] y Out[38] son iguales a lo que se quería demostrar en este teorema.

■

Teorema 2.3.4: Sea u = {0, u₁, u₂, u₃} $\in Q_\varepsilon^\perp$, entonces:

$$\mathbf{u \times R(p,u) = \frac{-2p_0\|u\|^2}{\|p\|^2}(0,p_1,p_2,p_3) = -2p_0\frac{\|u\|^2}{\|p\|^2}p_v} \qquad (2.32)$$

Demostración: En efecto,

In[1]:=

```
(* Se define la transformación Tv[], que transforma un Quaternión en un vector
   tridimensional *)

Tv[q_]:={q[[2]],q[[3]],q[[4]]};

(* Se define el Producto Cruz entre dos vectores tridimensionales *)

Cruz[a_,b_]:= {a[[2]] b[[3]] - a[[3]] b[[2]],  - a[[1]] b[[3]] + a[[3]] b[[1]],  a[[1]] b[[2]] - a[[2]] b[[1]]}
```

In[4]:=

```
(* Se definen los Quaterniones u y p *)

u={0,u1,u2,u3}; u={0,u1,u2,u3};

(* Se realiza el producto Cruz de la partes vectoriales de u y R(p,u), incluyendose la norma
   de "p", que no es unitaria (1/nP^2) *)

uXRu = Cruz[Tv[u],(1/nP^2)*Tv[ReflexionQ[p,u]]]//Expand;
```

```
(* Se aplica la definición de norma de u y la propiedad de que el producto interno de u y p
   es igual a cero por ser ortogonales *)

uXRu = uXRu//.{u3^2->nu^2-u1^2-u2^2,p1*u1->-p3*u3-p2*u2}//Expand;
uXRu = uXRu//.{u3^2->nu^2-u1^2-u2^2,p3*u3->-p1*u1-p2*u2}//Expand;
uXRu = uXRu//.{p2*u2->-p1*u1-p3*u3}//Expand;
uXRu = uXRu//.{u1^2->nu^2-u3^2-u2^2}//Expand

Out[12]=

        2              2              2
  -2 nu  p0 p1   -2 nu  p0 p2   -2 nu  p0 p3
 {------------, ------------, ------------}
         2              2              2
       nP             nP             nP
```

El resultado obtenido en Mathematica® se puede representar también como:

$$\left(\frac{-2\|u\|^2 p_0 p_1}{\|p\|^2}, \frac{-2\|u\|^2 p_0 p_2}{\|p\|^2}, \frac{-2\|u\|^2 p_0 p_3}{\|p\|^2}\right) = \frac{-2\|u\|^2 p_0}{\|p\|^2}(p_1, p_2, p_3) = \frac{-2\|u\|^2 p_0}{\|p\|^2}\mathbf{p}_v = \mathbf{u} \times \mathbf{R(p,u)}$$

Con lo anterior se demuestra la validez de este teorema.

■

Teorema 2.3.5: Sea $\mathbf{u} = (0, u_1, u_2, u_3) \in Q_v$, $\|\mathbf{u}\| = 1$, entonces:

$$\mathbf{Cos}\,\theta = \frac{1}{\|p\|^2}\left(\|p_v\|^2 - p_0^2\right) \; ; \qquad \mathbf{Sen}\,\theta = (\pm)\frac{2\,\|p_v\|}{\|p\|^2}p_0 \qquad (2.33)$$

$$\mathbf{u} \times \mathbf{R(p,u)} = (\pm)\,\mathbf{Sen}\,\theta\, \underline{\mathbf{w}} \qquad (2.34)$$

Donde $\underline{\mathbf{w}} = \frac{1}{\|p_v\|}\mathbf{p}_v$.

Demostración: En efecto, si tomamos del Teorema 2.3.3 **Cos** θ, **Sen** θ y sustituimos $\mathbf{u_0 = 0}$, obtenemos directamente lo mostrado para estas dos variables en el teorema. Para el producto cruz **u×R(p,u)**, del Teorema 2.3.4 tenemos que:

$$\mathbf{u} \times \mathbf{R(p,u)} = \frac{-2p_0\|u\|^2}{\|p\|^2}(0, p_1, p_2, p_3) = -2p_0\frac{\|u\|^2}{\|p\|^2}\mathbf{p}_v$$

de este teorema: $\mathbf{Sen}\,\theta = (\pm)\frac{2\,\|p_v\|}{\|p\|^2}p_0 \quad \Rightarrow \quad p_0 = (\pm)\frac{\|p\|^2}{2\,\|p_v\|}\mathbf{Sen}\,\theta$

Sustituyendo $\mathbf{p_0}$ y $\|\mathbf{u}\| = 1$ en la expresión del producto $\mathbf{u}\times\mathbf{R(p,u)}$:

$$\mathbf{u} \times \mathbf{R(p,u)} = -2\left((\pm)\frac{\|\mathbf{p}\|^2}{2\|\mathbf{p_v}\|}\,\mathbf{Sen}\,\theta\right)\frac{\|\mathbf{u}\|^2}{\|\mathbf{p}\|^2}\mathbf{p_v} = (\pm)\frac{\mathbf{p_v}}{\|\mathbf{p_v}\|}\,\mathbf{Sen}\,\theta$$

Como $\mathbf{u}\times\mathbf{R(p,u)}$ da como resultado un vector perpendicular al plano formado por **u** y **R(p,u)**, que es la dirección del eje de rotación, podemos definir como eje de rotación el vector $\underline{\mathbf{w}} = \dfrac{\mathbf{p_v}}{\|\mathbf{p_v}\|}$, por lo que:

$$\mathbf{u} \times \mathbf{R(p,u)} = (\pm)\mathbf{Sen}\,\theta\ \underline{\mathbf{w}}$$

■

Teorema 2.3.6: Sea $\theta \in [0,n\pi]$, $n \in \Re$, $\underline{\mathbf{w}} \in \mathbf{Q_v}$, $\|\underline{\mathbf{w}}\| = 1$, entonces la parte vectorial del Quaternión $\mathbf{p} = \{\mathbf{p_0}, \mathbf{p_1}, \mathbf{p_2}, \mathbf{p_3}\} \in \mathbf{Q}$ con $\mathbf{p_0} \in \Re - \{0\}$ es:

$$\mathbf{p_v} = \{\mathbf{p_1},\mathbf{p_2},\mathbf{p_3}\} = (\pm)\frac{\|\mathbf{p}\|^2}{2\mathbf{p_0}}\,\mathbf{Sen}\,\theta\ \underline{\mathbf{w}}\,; \qquad (2.35)$$

siendo $\underline{\mathbf{w}}$ tal que: $\mathbf{R(p,}\ \underline{\mathbf{w}}\,) = -\underline{\mathbf{w}}$

También $\mathbf{p_0} \in \Re - \{0\}$ satisface la siguiente ecuación:

$$4\,\mathbf{p_0^4} + 4\,\|\mathbf{p}\|^2\mathbf{p_0^2}\,\mathbf{Cos}\,\theta - \|\mathbf{p}\|^4\,\mathbf{Sen^2}\theta = 0 \qquad (2.36)$$

Demostración: Del Teorema 2.3.4 se tiene que:

$$\mathbf{u} \times \mathbf{R(p,u)} = (\pm)\,\mathbf{Sen}\,\theta\ \underline{\mathbf{w}} = \frac{-2\mathbf{p_0}}{\|\mathbf{p}\|^2}\mathbf{p_v}$$

Despejando $\mathbf{p_v}$ tenemos que:

$$(\pm)\,\mathbf{Sen}\,\theta\ \underline{\mathbf{w}} = \frac{-2\mathbf{p_0}}{\|\mathbf{p}\|^2}\mathbf{p_v} \qquad \Rightarrow \qquad \mathbf{p_v} = (\pm)\frac{\|\mathbf{p}\|^2}{2\mathbf{p_0}}\,\mathbf{Sen}\,\theta\ \underline{\mathbf{w}}$$

con esto se demuestra la primera parte del teorema. Para la segunda parte tenemos del Teorema 2.3.5 que:

$$\mathbf{Cos}\,\theta = \frac{1}{\|\mathbf{p}\|^2}\left(\|\mathbf{p_v}\|^2 - \mathbf{p_0^2} \right)\ ; \qquad \mathbf{Sen}\,\theta = (\pm)\frac{2\,\|\mathbf{p_v}\|}{\|\mathbf{p}\|^2}\mathbf{p_0}$$

Despejando de $\mathbf{Sen}\,\theta$ a $\|\mathbf{p_v}\|$ y de $\mathbf{Cos}\,\theta$ a $\mathbf{p_0^2}$, tenemos que:

$$\|p_\perp\| = (\pm)\,\|p\|^2 \frac{\text{Sen}\,\theta}{2\,p_0} \;; \qquad p_0^2 = -\|p\|^2 \text{Cos}\,\theta + \|p_\perp\|^2$$

combinando ambos resultados:

$$p_0^2 = -\|p\|^2 \text{Cos}\,\theta + \left((\pm)\,\|p\|^2 \frac{\text{Sen}\,\theta}{2\,p_0} \right)^2 = -\|p\|^2 \text{Cos}\,\theta + \|p\|^4 \frac{\text{Sen}^2\theta}{4\,p_0^2}$$

$$p_0^2 + \|p\|^2 \text{Cos}\,\theta - \|p\|^4 \frac{\text{Sen}^2\theta}{4\,p_0^2} = 0$$

$$4\,p_0^4 + 4\,\|p\|^2 p_0^2 \text{Cos}\,\theta - \|p\|^4 \text{Sen}^2\theta = 0$$

Con esto se demuestra la segunda parte del teorema.

■

Teorema 2.3.7: Las raíces reales de la ecuación de cuarto grado son:

$$4\,p_0^4 + 4\,\|p\|^2 p_0^2 \text{Cos}\,\theta - \|p\|^4 \text{Sen}^2\theta = 0 \qquad (2.37)$$

$$p_{0_1} = \|p\|\,\text{Sen}\,\frac{\theta}{2}\,, \qquad p_{0_2} = -\,\|p\|\,\text{Sen}\,\frac{\theta}{2} \qquad (2.38)$$

Demostración: En efecto,

In[1]:=

```
(* Se resuelve la ecuación de cuarto grado y se simplifica la solución *)

Solve[4*p0^4 + 4*nP^2*p0^2*Cos[T] - nP^4*(Sin[T])^2 == 0,{p0}]//Simplify//PowerExpand
      //Simplify//PowerExpand
```

Out[2]=

```
                      T                          T
{{p0 -> -I nP Cos[-]}, {p0 -> I nP Cos[-]},
                      2                          2

                     T                          T
 {p0 -> -(nP Sin[-])}, {p0 -> nP Sin[-]}}
                     2                          2
```

Se puede observar que las soluciones reales de la ecuación de cuarto grado, son las presentadas en el teorema.

■

Finalmente obtenemos con una combinación de los teoremas anteriores:

$$\mathbf{p_0} = (\pm)\,\|\mathbf{p}\|\,\mathbf{Sen}\frac{\theta}{2} \qquad (2.39)$$

$$\mathbf{p_v} = \{\mathbf{p_1},\mathbf{p_2},\mathbf{p_3}\} = (\pm)\frac{\|\mathbf{p}\|^2}{2\|\mathbf{p}\|\,\mathbf{Sen}\,\frac{\theta}{2}}\mathbf{Sen}\,\theta\,\underline{\mathbf{w}} = (\pm)\,\|\mathbf{p}\|\,\mathbf{Cos}\frac{\theta}{2}\,\underline{\mathbf{w}} \qquad (2.40)$$

CAPITULO III

3. MODELACION CINEMATICA

En este capítulo se presentará la modelación de la cinemática de algunos manipuladores robóticos utilizando el álgebra de Quaterniones. Dicha modelación se realizará utilizando inicialmente solamente rotaciones, después se obtendrán otros modelos diferentes aplicando reflexiones o una combinación de rotaciones y reflexiones según sea el caso. Lo anterior nos permitirá tener dos modelos cinemáticos diferentes para el mismo manipulador, que nos permitirá mover un manipulador, siguiendo una misma trayectoria, pero con una configuración diferente de sus eslabones en el espacio. Esto puede aplicarse, por ejemplo, en el proceso de evasión de obstáculos, ya que el manipulador puede cambiar la configuración de sus eslabones permitiéndole evadir algunos obstáculos al cambiar de configuración sin afectar la trayectoria.

3.1. MODELACION CINEMATICA DE UN MANIPULADOR DE TRES GRADOS DE LIBERTAD, UTILIZANDO ROTACIONES CON DIFERENTES SECUENCIAS DE MOVIMIENTOS.

OBJETIVO: El principal objetivo de esta sección, será presentar las diferencias en el proceso de modelación de la cinemática de un manipulador de tres grados de libertad, cuando se realiza utilizando diferentes secuencias de movimientos de sus eslabones. Se mostrará además, que el modelo final es el mismo, pero con configuraciones intermedias diferentes. Dicho cuerpo se presenta en la Figura 3.1

3.1.1. Configuración no deformada.

Mostraremos primeramente la modelación de la configuración no deformada, que consiste en determinar un sistema de ecuaciones que nos permitan determinar la posición del segundo y tercer eslabón, cuando el primer eslabón se encuentra en una posición inicial de referencia (home), conociendo la posición del elemento terminal del manipulador. Para lo anterior tenemos dos posibles secuencias de movimientos:

Primera Secuencia:

1.- Se mantiene el primer eslabón fijo.
Rotación 2.- Se aplica una rotación al segundo eslabón. (Figura 3.2).
Rotación 3.- Se aplica una rotación al tercer eslabón. (Figura 3.3).

Segunda Secuencia:

1.- Se mantiene el primer eslabón fijo.
Rotación 2.- Se aplica una rotación al tercer eslabón. (Figura 3.4).
Rotación 3.- Se aplica una rotación al segundo eslabón. (Figura 3.5).

Para lo anterior definiremos la configuración no deformada del manipulador, que consiste en definir el vector de posición del elemento terminal para posiciones arbitrarias del segundo y tercer eslabones en un plano dado. Consideremos la base canónica de $\Re^3$, esto es:

$$\mathbf{e}_1 = (1,0,0); \quad \mathbf{e}_2 = (0,1,0); \quad \mathbf{e}_3 = (0,0,1);$$

Ubicamos sistemas de referencia locales en la base de cada eslabón del manipulador como se muestra en la Figura 3.1. Observemos que, en este caso, los sistemas de referencia definidos coinciden con la base canónica la cual sirve como sistema de referencia inercial fijo, es decir:

$$\mathbf{g}_j = \mathbf{f}_j = \mathbf{e}_j \ , \quad j = 1,2,3.$$

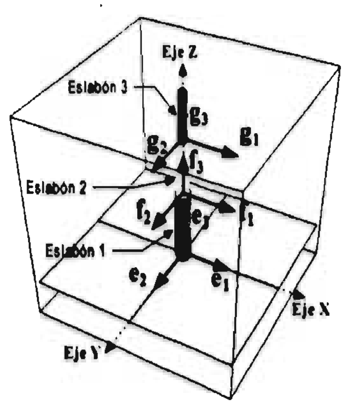


Figura 3.1. *Bases locales en la configuración original para la modelación.*

De la posición mostrada en la Figura 3.1, debemos permitir que los eslabones 2 y 3 giren a una posición arbitraria, considerando que los eslabones tienen los siguientes ejes de rotación:

Eslabón	Eje De Rotación
1	$\mathbf{e}_3$
2	$\mathbf{f}_1 = \mathbf{e}_1$
3	$\mathbf{g}_1 = \mathbf{e}_1$

Con los ejes de rotación anteriores, podemos definir la forma que tendrán inicialmente los Quaterniones que nos permitirán rotar los eslabones 2 y 3.

Eslabón 2: Consideremos el siguiente parámetro que caracterizará a nuestra primera rotación;

$$P = \{P_0, P_1, P_2, P_3\} \Rightarrow \qquad P_v = \{P_1, P_2, P_3\}, \quad \|P\| = 1.$$

En este caso se tiene que siendo $\underline{w} \in \Re^3$ el eje de rotación, entonces, [1]:

$$P_0 = Cos(\varphi_1/2)$$
$$P_v = Sin(\varphi_1/2)^*\underline{w} = Sin(\varphi_1/2)^*\mathbf{f}_1 = Sin(\varphi_1/2)^*\mathbf{e}_1 = Sin(\varphi_1/2)^*\{1,0,0\}$$
$$P_v = \{P_1, 0,0\}$$

De lo anterior se obtiene la forma local del parámetro P, que es: $\mathbf{P = \{P_0, P_1, 0, 0\}}$, Donde:

$P_0 = Cos(\varphi_1/2);$
$P_1 = Sen(\varphi_1/2);$

Siendo φ_1 el ángulo de rotación del eslabón 2.

Eslabón 3: Consideremos ahora el siguiente parámetro que caracterizará la segunda rotación;

$$Q = \{Q_0, Q_1, Q_2, Q_3\} \qquad \Rightarrow \qquad Q_v = \{Q_1, Q_2, Q_3\}, \quad \|Q\| = 1.$$

Siendo $\underline{w} \in \Re^3$ el eje de rotación, tenemos que:

$$Q_v = Sin(\varphi_2/2)^* \underline{w} = Sin(\varphi_2/2)^* g_1 = Sin(\varphi_2/2)^* e_1 = Sin(\varphi_2/2)^*\{1,0,0\}$$
$$Q_v = \{ Q_1, 0, 0\}$$

De lo anterior se obtiene la forma local del parámetro **Q**, que es: $\mathbf{Q = \{Q_0, Q_1, 0, 0\}}$, Donde:

$Q_0 = Cos(\varphi_2/2);$
$Q_1 = Sen(\varphi_2/2);$

Siendo φ_2 el ángulo de rotación del eslabón 3.

Con los Quaterniones anteriores podemos ahora realizar las rotaciones necesarias para determinar la configuración no deformada, se debe observar que si los ejes de rotación cambian su forma, también cambiará la forma de los Quaterniones dados.

Aplicación de la Primera Secuencia en la configuración no deformada.

Rotación 2.- Primeramente rotamos el segundo eslabón con el Quaternión **P**. Esto se realiza rotando la base local $\mathbf{f}_j$, pero además, debemos rotar la base $\mathbf{g}_j$ ya que al rotar el eslabón 2, el eslabón tres también se ve afectado por la misma rotación (Figura 3.2).

$$\mathbf{f}_j^I = \rho(\mathbf{P}, \mathbf{f}_j) \quad \text{(Rotación del segundo eslabón con Quaternión P)} \qquad (3.1)$$

$$\mathbf{g}_j^I = \rho(\mathbf{P}, \mathbf{g}_j) \quad \text{(Rotación del tercer eslabón con Quaternión P)}, \quad j = 1,2,3$$

Como la rotación anterior afectó el eje de rotación del tercer eslabón ($\mathbf{g}_1$), dicho eje de rotación cambia, siendo el nuevo eje $\mathbf{g}^I_1$. Por lo tanto el Quaternión **Q** se modifica de la siguiente forma:

$$Q^I = (Q^I_0, Q^I_1, Q^I_2, Q^I_3), \text{ con } \qquad Q^I_0 = Cos(\varphi_2/2), \qquad Q^I_v = Sin(\varphi_2/2)^*\mathbf{g}^I_1$$

$$Q^I_v = Sin(\varphi_2/2)^*\rho(P, \mathbf{g}_1),$$

Como $Sin(\varphi_2/2) = Cte$, se tiene que: $\quad \mathbf{Q^I_v = \rho(P, Q_v) \Rightarrow Q^I = \rho(P, Q)}$

El Quaternión Q^I caracteriza a la rotación de la barra 3. Esto es, el Quaternión es actualizado, con respecto al sistema inercial fijo, puesto que el eje original de rotación es afectado por las correspondientes rotaciones locales de los eslabones, lo cual modifica con respecto al sistema de referencia global al mismo. Si aplicamos la transformación ρ(P,Q) obtenemos la forma explícita del Quaternión Q^{I} :

$$\mathbf{Q^{I} = \rho(P,\ Q) = (Q_0,\ P_0{*}(-P_1{*}Q_0 + P_0{*}Q_1) + P_1{*}(P_0{*}Q_0 + P_1{*}Q_1),\ 0,\ 0)}$$

$$\mathbf{Q^{I} = (Q_0,\ (P_0^2 + P_1^2){*}Q_1,\ 0,\ 0) = (Q_0,\ Q_1,\ 0,\ 0) = Q}$$

Podemos observar, en este caso, que el Quaternión Q no sufrió ninguna transformación, esto debido a que los ejes de rotación de los eslabones 2 y 3 son paralelos.

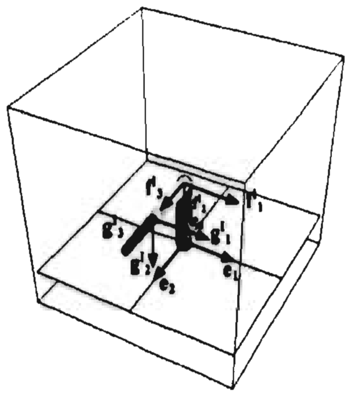


Figura 3.2. *Rotación de los eslabones 2 y 3 en la primera secuencia, para la configuración no deformada.*

Rotación 3.- De la misma forma, ahora se rotará el tercer eslabón utilizando el Quaternión correspondiente $\mathbf{Q^{I}}$, lo que implica que la base $\mathbf{g_j}$ sufre otra transformación como se ilustra a continuación. (Figura 3.3)

$$\mathbf{g_j^{II} = \rho(Q^{I},\ g_j^{I})} \qquad \text{(Rotación del tercer eslabón con Quaternión } Q^{I}\text{)}.$$

Pero como $\mathbf{g_j^{I} = \rho(P,\ g_j\ )}$, y $\mathbf{Q^{I} = \rho(P,\ Q)}$ se tiene que: $\mathbf{g_j^{II} = \rho(\rho(P,\ Q),\ \rho(P,\ g_j\ ))}$

Desarrollando las rotaciones anteriores tenemos que:

$$\mathbf{g_j^{II} = \rho(P * Q * \overline{P}, P * g_j * \overline{P}) = (P * Q * \overline{P} * P * g_j * \overline{P} * \overline{P * Q * \overline{P}})}$$

Pero como: $\mathbf{\overline{P * Q * \overline{P}} = P * \overline{P * Q}}$ y $\mathbf{\overline{P} * P = P * \overline{P} = \{1,0,0,0\}}$, entonces:

$$\mathbf{g_j^{II} = \rho((P{*}Q),\ g_j) = \rho(P,\rho(Q,\ g_j)) = P{*}Q * g_j * \overline{P * Q}} \qquad (3.2)$$

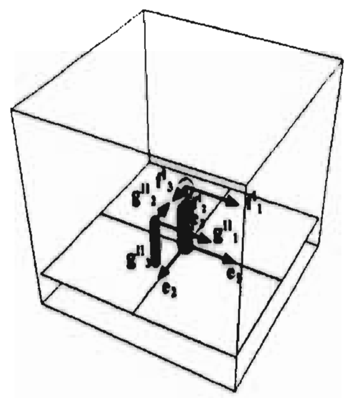


Figura 3.3. *Rotación del eslabón 3 en la primera secuencia, para la configuración no deformada.*

Aplicación de la Segunda Secuencia en la configuración no deformada.

Rotación 1.- Rotamos el tercer eslabón con el Quaternión Q. Esto se realiza rotando solamente la base local $\mathbf{g}_j$, es decir, esta rotación no afecta a ningún otro eslabón y por ello tampoco afecta a ninguna otra base local ni Quaternión. (Figura 3.4)

$$\mathbf{g}_j^{I} = \rho(\mathbf{Q}, \mathbf{g}_j) \quad \text{(Rotación del tercer eslabón con Quaternión Q), } j = 1,2,3$$

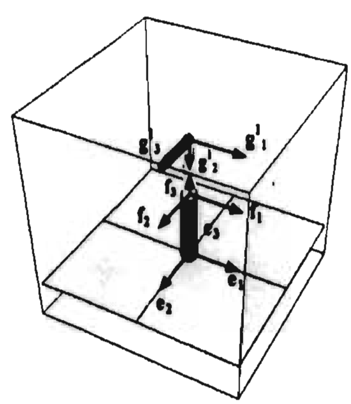


Figura 3.4. *Rotación del tercer eslabón para la configuración no deformada, segunda secuencia.*

Rotación 2.- Se aplica la rotación al segundo eslabón con el Quaternión P, lo que implica que las bases $\mathbf{f}_j$ y $\mathbf{g}_j$ sufren las siguientes transformaciones.(Figura 3.5)

$$\mathbf{f}_j^{I} = \rho(\mathbf{P}, \mathbf{f}_j) \quad \text{(Rotación del segundo eslabón con Quaternión P)} \qquad (3.3)$$

$$\mathbf{g}_j^{II} = \rho(\mathbf{P}, \mathbf{g}_j^{I}) \quad \text{(Rotación del tercer eslabón con Quaternión P).}$$

Como $\mathbf{g}_j^{I} = \rho(\mathbf{Q}, \mathbf{g}_j)$ tenemos que:

$$\mathbf{g}_j^{II} = \rho(\mathbf{P}, \rho(\mathbf{Q}, \mathbf{g}_j)) = \rho((\mathbf{P}^*\mathbf{Q}), \mathbf{g}_j) \tag{3.4}$$

Nota: *En este caso no hubo necesidad de modificar el Quaternión Q, debido a que cuando se aplicó la rotación P, que es la que afecta a Q, la rotación correspondiente al Quaternión Q ya se había aplicado.*

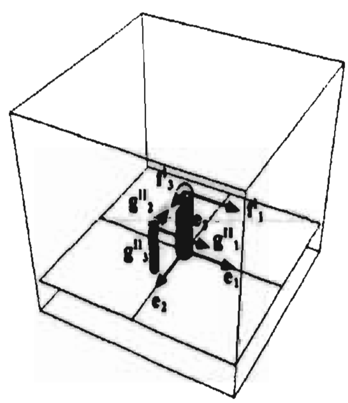


Figura 3.5. *Rotación de los eslabones 2 y 3 en la configuración no deformada, segunda secuencia.*

De lo anterior podemos observar que la forma de las bases locales, después de aplicar las rotaciones en las dos secuencias, es exactamente la misma, o sea:

$$\mathbf{f}_j^{I} = \rho(\mathbf{P}, \mathbf{f}_j) \qquad \text{(base local del segundo eslabón)}$$

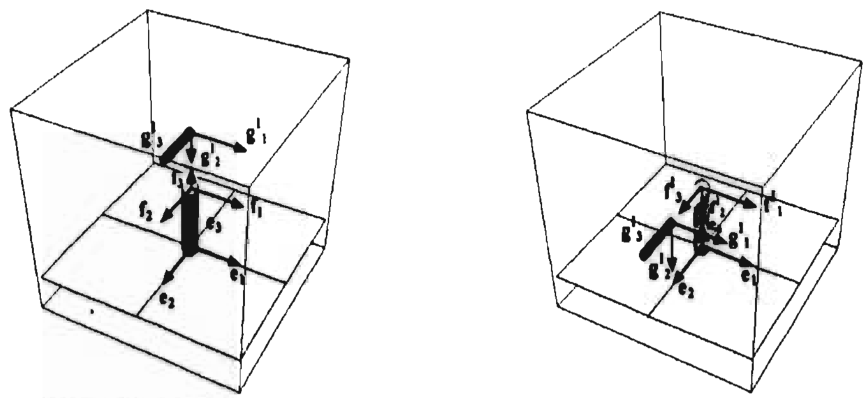
$$\mathbf{g}_j^{II} = \rho(\mathbf{P}, \rho(\mathbf{Q}, \mathbf{g}_j)) = \rho(\mathbf{P}^*\mathbf{Q}, \mathbf{g}_j) \qquad \text{(base local del tercer eslabón)} \tag{3.5}$$

La forma explícita de las bases anteriores es la siguiente:

$$\begin{aligned}
\mathbf{f}_1^{I} &= \{1, 0, 0\} \\
\mathbf{f}_2^{I} &= \{0, -1 + 2^*P_0^2, 2^*P_0^*P_1\} \\
\mathbf{f}_3^{I} &= \{0, -2^*P_0^*P_1, -1 + 2^*P_0^2\}
\end{aligned} \tag{3.6}$$

$$\begin{aligned}
\mathbf{g}_1^{II} &= \{1, 0, 0\} \\
\mathbf{g}_2^{II} &= \{0, -P_0^2 + P_1^2 + 2^*P_0^{2*}Q_0^2 - 2^*P_1^{2*}Q_0^2 - 4^*P_0^*P_1^*Q_0^*Q_1, \\
&\quad 2^*(-(P_0^*P_1) + 2^*P_0^*P_1^*Q_0^2 + P_0^{2*}Q_0^*Q_1 - P_1^{2*}Q_0^*Q_1)\} \\
\mathbf{g}_3^{II} &= \{0, 2^*(P_0^*P_1 - 2^*P_0^*P_1^*Q_0^2 - P_0^{2*}Q_0^*Q_1 + P_1^{2*}Q_0^*Q_1), \\
&\quad -P_0^2 + P_1^2 + 2^*P_0^{2*}Q_0^2 - 2^*P_1^{2*}Q_0^2 - 4^*P_0^*P_1^*Q_0^*Q_1\}
\end{aligned} \tag{3.7}$$

Podemos observar que la forma final de las bases locales, después de aplicadas las dos secuencias de rotaciones, son idénticas, esto quiere decir que la secuencia en la modelación no afecta la forma final del modelo, pero es importante hacer notar que los puntos intermedios que se generan en una secuencia y otra, son diferentes, como se puede observar en las siguientes figuras.



Punto intermedio en la primera secuencia **Punto intermedio en la segunda secuencia**

Figura 3.6. *Puntos intermedios en la configuración no deformada.*

Los puntos intermedios en las dos diferentes secuencias, quedan determinados por las bases locales correspondientes y por la posición del elemento terminal en cada caso, como se muestra a continuación, siendo L_1, L_2 y L_3 las longitudes de los eslabones.

Vectores que definen los eslabones y posición del elemento terminal, para el punto intermedio de la primera secuencia (Figura 3.6).

Eslabón 1.

$$b_1 = L_1 * e_3 = \{0, 0, L_1\};$$

Eslabón 2.

$$b_2 = L_2 * f_3^1 = L_2 * \rho(P, f_3) = L_2 * (P * e_3 * \bar{P}) = \{0, -2*L_2*P_0*P_1, L_2*(2*P_0^2 - 1)\}; \qquad (3.8)$$

Eslabón 3.

$$b_3 = L_3 * g_3^1 = L_3*\rho(P, g_3) = L_3 * (P * e_3 * \bar{P}) = \{0, -2*L_3*P_0*P_1, L_3*(2*P_0^2 - 1)\};$$

Posición del elemento terminal en el punto intermedio de la primera secuencia.

$$R_{p1} = b_1 + b_2 + b_3 = \{0, -2*(L_2 + L_3)*P_0*P_1, L_1 - L_2 - L_3 + 2*(L_2 + L_3)*P_0^2\} \qquad (3.9)$$

Vectores que definen los eslabones y posición del elemento terminal, para el punto intermedio de la segunda secuencia (Figura 3.6).

Eslabón 1.

$$b_1 = L_1 * e_3 = \{0, 0, L_1\};$$

Eslabón 2.

$$b_2 = L_2*f_3 = L_2 * e_3 = \{0, 0, L_2\}; \qquad (3.10)$$

Eslabón 3.

$$b_3 = L_3 * g_3^1 = L_3*\rho(Q, g_3) = L_3 * (Q * e_3 * \bar{Q}) = \{0, -2*L_3*Q_0*Q_1, L_3*(2*Q_0^2 - 1)\};$$

Posición del elemento terminal en el punto intermedio de la segunda secuencia.

$$\mathbf{R}_{p1} = \mathbf{b}_1 + \mathbf{b}_2 + \mathbf{b}_3 = \{0,\ -2{*}L_3{*}Q_0{*}Q_1,\ L_1 + L_2 + L_3{*}(2{*}Q_0^2 - 1)\} \qquad (3.11)$$

3.1.2. Vectores que definen los eslabones y posición del elemento terminal en la configuración no deformada.

De la Figura 3.5 y Figura 3.7 podemos determinar los vectores que definen los eslabones del manipulador y el vector de posición del elemento terminal (punto p). Las longitudes de los eslabones son L_1, L_2, L_3.

$$\begin{aligned} \mathbf{b}_1 &= L_1{*}\,e_3; \quad (\text{Eslabón } 1) \\ \mathbf{b}_2 &= L_2{*}\,f_3^{\,|} = L_2{*}\,\rho(P, f_3) = L_2{*}\,(P * e_3 * \bar{P}); \quad (\text{Eslabón } 2) \\ \mathbf{b}_3 &= L_3{*}\,g_3^{\,||} = L_3{*}\rho((P{*}Q), g_3) = L_3{*}\,((P{*}Q) * e_3 * (\overline{P * Q})); \quad (\text{Eslabón } 3) \end{aligned} \qquad (3.12)$$

$$\mathbf{R}_p = \mathbf{b}_1 + \mathbf{b}_2 + \mathbf{b}_3 \quad (\text{Posición del elemento terminal}) \qquad (3.13)$$

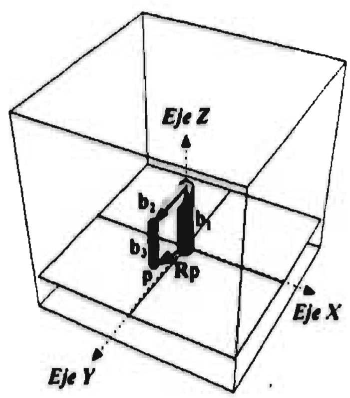


Figura 3.7. *Vector de posición del elemento terminal en la configuración no deformada.*

3.1.3. Ecuaciones explícitas optimizadas para la configuración no deformada.

A continuación se muestra la forma de las ecuaciones optimizadas, correspondientes a la configuración no deformada.

$$\begin{aligned} \mathbf{b}_1 &= \{0,\ 0,\ L_1\}; \\ \mathbf{b}_2 &= \{0,\ -2{*}A1{*}L2,\ -A2{*}L_2\}; \\ \mathbf{b}_3 &= \{0,\ 2{*}(A2{*}A3 + A1{*}A4){*}L_3,\ (A2{*}A4 - 4{*}A1{*}A3){*}L_3\}; \end{aligned} \qquad (3.14)$$

$$\mathbf{R}_p = \{0,\ 2{*}((A2{*}A3 + A1{*}A4){*}L_3 - A1{*}L2),\ L_1 - A2{*}L_2 + (A2{*}A4 - 4{*}A1{*}A3){*}L_3\}; \qquad (3.15)$$

Donde:

$$\begin{aligned} A1 &= P_0{*}P_1; \quad & A2 &= 1 - 2{*}P_0{\wedge}2; \\ A3 &= Q_0{*}Q_1; \quad & A4 &= 1 - 2{*}Q_0{\wedge}2; \end{aligned} \qquad (3.16)$$

Posición del elemento terminal en el punto intermedio de la segunda secuencia.

$$\mathbf{R_{p1} = b_1 + b_2 + b_3 = \{0, -2{*}L_3{*}Q_0{*}Q_1, L_1 + L_2 + L_3{*}(2{*}Q_0^2 - 1)\}} \qquad (3.11)$$

3.1.2. Vectores que definen los eslabones y posición del elemento terminal en la configuración no deformada.

De la Figura 3.5 y Figura 3.7 podemos determinar los vectores que definen los eslabones del manipulador y el vector de posición del elemento terminal (punto p). Las longitudes de los eslabones son $\mathbf{L_1, L_2, L_3}$.

$$\mathbf{b_1 = L_1{*}\,e_3;} \quad (\text{Eslabón } 1)$$
$$\mathbf{b_2 = L_2{*}\,f_3^{I} = L_2{*}\,\rho(P, f_3) = L_2{*}\,(P * e_3 * \bar{P});} \quad (\text{Eslabón } 2) \qquad (3.12)$$
$$\mathbf{b_3 = L_3{*}\,g_3^{II} = L_3{*}\rho((P{*}Q), g_3) = L_3{*}\,((P{*}Q) * e_3 * (\overline{P * Q}));} \quad (\text{Eslabón } 3)$$

$$\mathbf{R_p = b_1 + b_2 + b_3} \quad (\text{Posición del elemento terminal}) \qquad (3.13)$$

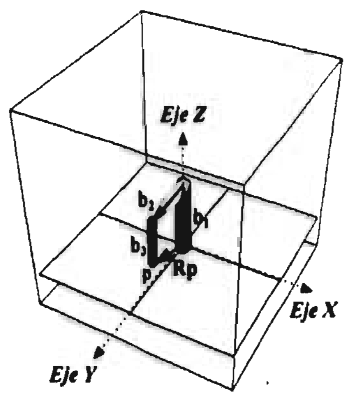


Figura 3.7. *Vector de posición del elemento terminal en la configuración no deformada.*

3.1.3. Ecuaciones explícitas optimizadas para la configuración no deformada.

A continuación se muestra la forma de las ecuaciones optimizadas, correspondientes a la configuración no deformada.

$$\mathbf{b_1 = \{0, 0, L_1\};}$$
$$\mathbf{b_2 = \{0, -2{*}A1{*}L2, -A2{*}L_2\};} \qquad (3.14)$$
$$\mathbf{b_3 = \{0, 2{*}(A2{*}A3 + A1{*}A4){*}L_3, (A2{*}A4 - 4{*}A1{*}A3){*}L_3\};}$$

$$\mathbf{R_p = \{0, 2{*}((A2{*}A3 + A1{*}A4){*}L_3 - A1{*}L2), L_1 - A2{*}L_2 + (A2{*}A4 - 4{*}A1{*}A3){*}L_3\};} \qquad (3.15)$$

Donde:

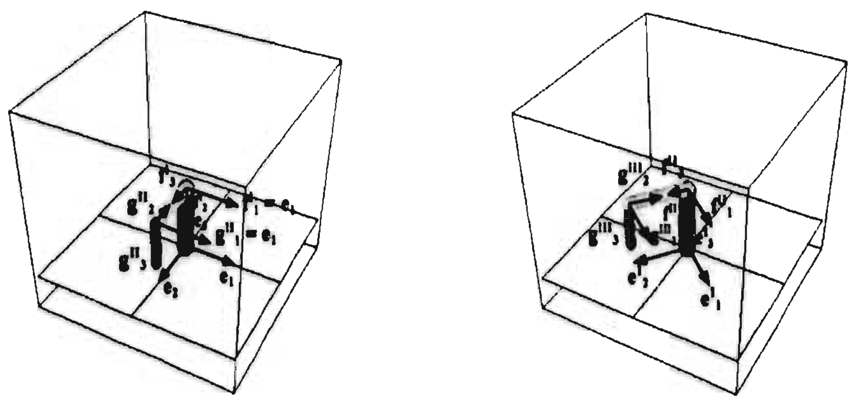
$$\mathbf{A1 = P_0{*}P_1; \quad A2 = 1 - 2{*}P_0\text{^}2;}$$
$$\mathbf{A3 = Q_0{*}Q_1; \quad A4 = 1 - 2{*}Q_0\text{^}2;} \qquad (3.16)$$

3.1.4. Configuración deformada.

De la misma forma que se realizó la modelación de la configuración no deformada, ahora se modelara la configuración deformada, utilizando dos diferentes secuencias de movimientos de los tres eslabones, partiendo de la configuración deformada. Las secuencias que se utilizarán son las siguientes:

Primera Secuencia.

Rotación 1.- rotan los tres eslabones con el Quaternión **r1**, partiendo de la configuración no deformada. (Figura 3.8).



Configuración no deformada **Rotación de los tres eslabones**

Figura 3.8. *Rotación de los tres eslabones en la configuración deformada, primera secuencia.*

Rotación 2.- rotan los eslabones 2 y 3 con el Quaternión **r2**. (Figura 3.9).

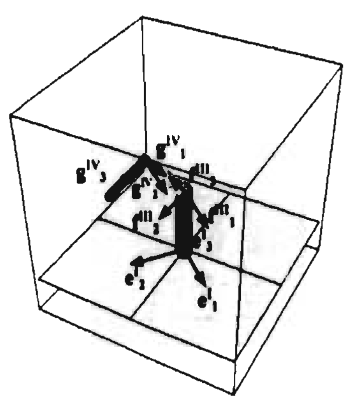


Figura 3.9. *Rotación de los eslabones 2 y 3 en la configuración deformada, primera secuencia.*

Rotación 3.- rota el eslabón 3 con el Quaternión **r3**. (Figura 3.10).

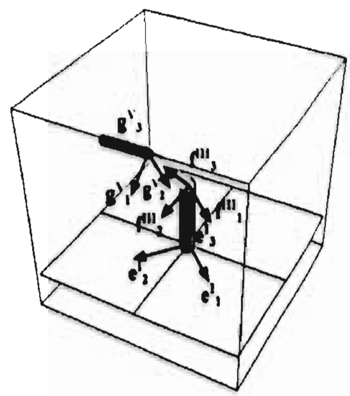
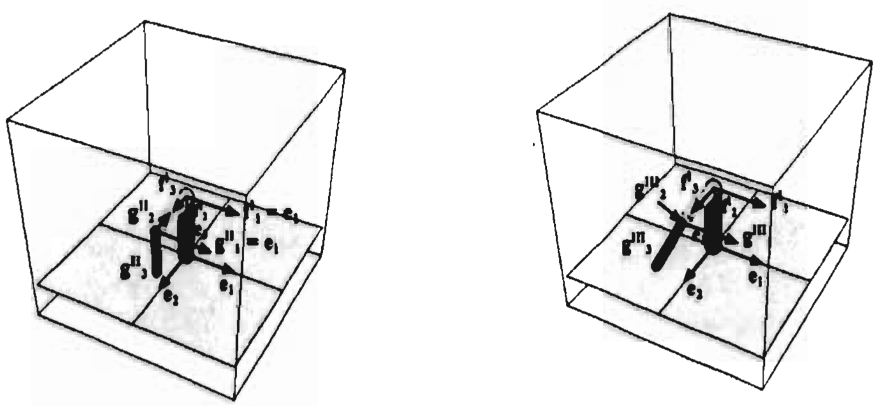


Figura 3.10. *Rotación del eslabón 3 en la configuración deformada, primera secuencia.*

Segunda Secuencia.

Rotación 1.- rota el eslabón 3 con el Quaternión **r3**, partiendo de la configuración no deformada. (Figura 3.11).



Configuración no deformada **Rotación del tercer eslabón**

Figura 3.11. *Rotación del eslabón 3 en la configuración deformada, segunda secuencia.*

Rotación 2.- rotan los eslabones 2 y 3 con el Quaternión **r2**. (Figura 3.12).

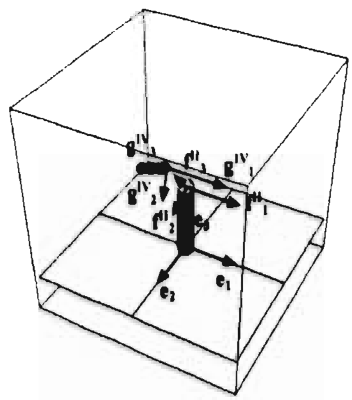


Figura 3.12. *Rotación de los eslabones 2 y 3 en la configuración deformada, segunda secuencia.*

Rotación 3.- rotan los tres eslabones con el Quaternión **r1**. (Figura 3.13).

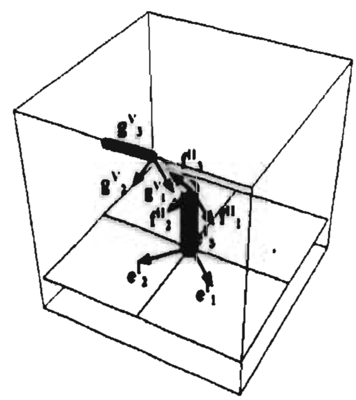


Figura 3.13. *Rotación de los tres eslabones en la configuración deformada, segunda secuencia.*

De la Figura 3.3 podemos ver que los ejes de rotación de los eslabones se conservan como en la configuración original (Figura 3.1). Por lo anterior podemos definir los Quaterniones para rotar a posición deformada como se muestra a continuación.

Eslabón 1: $r1 = \{r1_0, r1_1, r1_2, r1_3\} \Rightarrow r1_v = \{r1_1, r1_2, r1_3\}, \quad \|r1\| = 1.$

La forma local del Quaternión r1 es:

$$r1_v = \mathrm{Sin}(\theta_1/2)^* e_3 = \mathrm{Sin}(\theta_1/2)^*\{0, 0, 1\}$$
$$r1_v = \{0, 0, r1_3\}$$

De lo anterior: $\mathbf{r1 = \{ r1_0, 0, 0, r1_3 \}}$, Donde:

$r1_0 = Cos(\theta_1/2);$
$r1_3 = Sen(\theta_1/2);$

Siendo θ_1 el ángulo de rotación del eslabón 1 en la configuración deformada.

Eslabón 2: $r2 = \{r2_0, r2_1, r2_2, r2_3\} \quad \Rightarrow \quad r2_v = \{r2_1, r2_2, r2_3\}, \quad \|r2\| = 1.$

La forma local del Quaternión r2 es:

$r2_v = Sin(\theta_2/2)^* e_1 = Sin(\theta_2/2)^*\{1, 0, 0\}$
$r2_v = \{ r2_1, 0, 0 \}$

De lo anterior: $\mathbf{r2 = \{ r2_0, r2_1, 0, 0\}}$, Donde:

$r2_0 = Cos(\theta_2/2);$
$r2_1 = Sen(\theta_2/2);$

Siendo θ_2 el ángulo de rotación del eslabón 2 en la configuración deformada.

Eslabón 3: $r3 = \{r3_0, r3_1, r3_2, r3_3\} \quad \Rightarrow \quad r3_v = \{r3_1, r3_2, r3_3\}, \quad \|r3\| = 1.$

La forma local del Quaternión r3 es:

$r3_v = Sin(\theta_3/2)^* e_1 = Sin(\theta_3/2)^*\{1, 0, 0\}$
$r3_v = \{ r3_1, 0, 0 \}$

De lo anterior: $\mathbf{r3 = \{ r3_0, r3_1, 0, 0 \}}$, Donde:

$r3_0 = Cos(\theta_3/2);$
$r3_1 = Sen(\theta_3/2);$

Siendo θ_3 el ángulo de rotación del eslabón 3 en la configuración deformada.

Aplicación de la Primera Secuencia en la configuración deformada.

Rotación 1.- Rotación de los tres eslabones con Quaternión r1:

En la Figura 3.8 podemos observar que todas las bases locales son afectadas por la rotación **r1** (rotación de los tres eslabones), a continuación se muestran dichas bases.

$\mathbf{e_j^{I}} = \rho(\mathbf{r1}, \mathbf{e_j})$ (Rotación del primer eslabón con Quaternión r1)

$\mathbf{f_j^{II}} = \rho(\mathbf{r1}, \mathbf{f_j^{I}})$ (Rotación del segundo eslabón con Quaternión r1) (3.17)

$\mathbf{g_j^{III}} = \rho(\mathbf{r1}, \mathbf{g_j^{II}})$ (Rotación del tercer eslabón con Quaternión r1)

Donde: $\mathbf{j = 1,2,3}$

Los ejes de rotación del segundo y tercer eslabones son afectados por **r1**, es decir los nuevos ejes de rotaciones de los eslabones 2 y 3 son $\mathbf{f}_1^{II}$ y $\mathbf{g}_1^{III}$ respectivamente, por ello los Quaterniones **r2** y **r3** se transforman de la siguiente forma:

$$\mathbf{r2^I = \{r2_0, r2^I{}_v\}; \quad r2^I{}_v = Sin(\theta_2/2)^* f_1^{II} = Sin(\theta_2/2)^*\rho(r1,\ f_1^I)}$$
$$\mathbf{r2^I{}_v = Sin(\theta_2/2)^*\rho(r1, e_1) = \rho(r1, r2_v) \quad \Rightarrow \quad r2^I = \rho(r1, r2)} \tag{3.18}$$

$$\mathbf{r3^I = \{r3_0, r3^I{}_v\}; \quad r3^I{}_v = Sin(\theta_3/2)^*\ g_1^{III} = Sin(\theta_3/2)^*\rho(r1,\ g_1^{II})}$$
$$\mathbf{r3^I{}_v = Sin(\theta_3/2)^*\rho(r1, e_1) = \rho(r1, r3_v) \quad \Rightarrow \quad r3^I = \rho(r1, r3)}$$

Después de aplicada la primera rotación, los Quaterniones que caracterizan la rotaciones de los eslabones 2 y 3 son **r2'** y **r3'** respectivamente, los cuales han sido actualizados con respecto al sistema inercial fijo, y su forma explícita es la siguiente:

$$\mathbf{r2^I = \rho(r1, r2) = \{r2_0, (-1 + 2^*r1_0{}^2)^*r2_1, 2^*r1_0{}^*r1_3{}^*r2_1, 0\},}$$

$$\mathbf{r3^I = \rho(r1, r3) = \{r3_0, (-1 + 2^*r1_0{}^2)^*r3_1, 2^*r1_0{}^*r1_3{}^*r3_1, 0\}.}$$

Rotación 2.- Rotación de los eslabones 2 y 3 con $r2^I$:

Para este caso observamos que las bases locales afectadas por la rotación $r2^I$ son las que corresponden a los eslabones 2 y 3 ($\mathbf{f}_j$ y $\mathbf{g}_j$). (Figura 3.9)

$\mathbf{f}_j^{III} = \rho(\mathbf{r2^I}, \mathbf{f}_j^{II})$ (Rotación del segundo eslabón con Quaternión $r2^I$)

$\mathbf{g}_j^{IV} = \rho(\mathbf{r2^I}, \mathbf{g}_j^{III})$ (Rotación del tercer eslabón con Quaternión $r2^I$) (3.19)

Donde: $\mathbf{j = 1,2,3}$

Con la rotación **r2'** se ve afectado el eje de giro del tercer eslabón, que ahora será $\mathbf{g}_1^{IV}$, por ello el Quaternión **r3'** se transforma de la siguiente forma:

$$\mathbf{r3^{II} = \{r3_0, r3^{II}{}_v\};}$$
$$\mathbf{r3^{II}{}_v = Sin(\theta_3/2)^* g_1^{IV} = Sin(\theta_3/2)^*\rho(r2^I,\ g_1^{III}) = Sin(\theta_3/2)^*\rho(r2^I, \rho(r1,\ g_1^{II}))}$$
$$\mathbf{r3^{II}{}_v = \rho(r2^I, Sin(\theta_3/2)^*\rho(r1, e_1)) = \rho(r2^I, r3^I{}_v) \quad \Rightarrow \quad r3^{II} = \rho(r2^I, r3^I)}$$

$$\mathbf{r3^{II} = \rho(r2^I, r3^I) = \rho(\rho(r1, r2), \rho(r1, r3)) = \rho(r1 * r2 * \overline{r1},\ r1 * r3 * \overline{r1})}$$
$$\mathbf{r3^{II} = r1 * r2 * \overline{r1} * r1 * r3 * \overline{r1} * \overline{r1 * r2 * \overline{r1}}}$$

$$\mathbf{r3^{II} = r1 * r2 * r3 * \overline{r1 * r2} = \rho(r1 * r2, r3) = \rho(r1, \rho(r2, r3))} \tag{3.20}$$

Ahora, después de aplicadas las rotaciones anteriores el Quaternión que caracteriza la rotación del tercer eslabón es **r3''**, cuya forma explícita es:

$$\mathbf{r3^{II} = \rho((r1^*r2), r3) = \{r3_0, (-1 + 2^*r1_0{}^2)^*r3_1, 2^*r1_0{}^*r1_3{}^*r3_1, 0\}.}$$

Rotación 3.- Rotación del tercer eslabón 3 con r3'':

Por último se tiene la rotación del tercer eslabón con el Quaternión **r3''**, con lo que se modifica la ubicación de la base local del eslabón 3. (Figura 3.10).

$$\mathbf{g}_j^{V} = \rho(\mathbf{r3''}, \mathbf{g}_j^{IV}) \quad \text{(Rotación del tercer eslabón con Quaternión r3'')} \qquad (3.21)$$

Donde: $\mathbf{j = 1,2,3}$

Finalmente las bases que representan a cada eslabón, con las rotaciones correspondientes son:

Eslabón 1:

$$\mathbf{e}_j^{I} = \rho(\mathbf{r1}, \mathbf{e}_j) \qquad (3.22)$$

Eslabón 2: Simplificando $\mathbf{f}_j^{III}$.

$$\mathbf{f}_j^{III} = \rho(\mathbf{r2'}, \mathbf{f}_j^{II}) = \rho(\rho(\mathbf{r1}, \mathbf{r2}), \rho(\mathbf{r1}, \mathbf{f}_j^{I})) = \rho(\rho(\mathbf{r1}, \mathbf{r2}), \rho(\mathbf{r1}, \rho(\mathbf{P}, \mathbf{f}_j)))$$

$$\mathbf{f}_j^{III} = \rho(\rho(\mathbf{r1}, \mathbf{r2}), \rho(\mathbf{r1}, \rho(\mathbf{P}, \mathbf{f}_j))) = \rho(\mathbf{r1}*\mathbf{r2}*\overline{\mathbf{r1}}, \rho(\mathbf{r1}, \mathbf{P}*\mathbf{f}_j*\overline{\mathbf{P}})$$

$$\mathbf{f}_j^{III} = \rho(\mathbf{r1}*\mathbf{r2}*\overline{\mathbf{r1}}, \mathbf{r1}*\mathbf{P}*\mathbf{f}_j*\overline{\mathbf{P}}*\overline{\mathbf{r1}}) = \mathbf{r1}*\mathbf{r2}*\overline{\mathbf{r1}}*\mathbf{r1}*\mathbf{P}*\mathbf{f}_j*\overline{\mathbf{P}}*\overline{\mathbf{r1}}*\overline{\mathbf{r1}*\mathbf{r2}*\overline{\mathbf{r1}}}$$

$$\mathbf{f}_j^{III} = \mathbf{r1}*\mathbf{r2}*\mathbf{P}*\mathbf{f}_j*\overline{\mathbf{P}}*\overline{\mathbf{r1}}*\mathbf{r1}*\overline{\mathbf{r1}*\mathbf{r2}} = \mathbf{r1}*\mathbf{r2}*\mathbf{P}*\mathbf{f}_j*\overline{\mathbf{P}}*\overline{\mathbf{r1}*\mathbf{r2}}$$

$$\mathbf{f}_j^{III} = \mathbf{r1}*\mathbf{r2}*\mathbf{P}*\mathbf{f}_j*\overline{\mathbf{r1}*\mathbf{r2}*\mathbf{P}}$$

$$\mathbf{f}_j^{III} = \rho(\mathbf{r1}*\mathbf{r2}*\mathbf{P}, \mathbf{f}_j) = \rho(\mathbf{r1}, \rho(\mathbf{r2}, \rho(\mathbf{P}, \mathbf{f}_j)))$$

$$\mathbf{f}_j^{III} = \rho(\mathbf{r1}*\mathbf{r2}*\mathbf{P}, \mathbf{e}_j) = \rho(\mathbf{r1}, \rho(\mathbf{r2}, \rho(\mathbf{P}, \mathbf{e}_j))) \qquad (3.23)$$

Eslabón 3: Simplificando $\mathbf{g}_j^{V}$.

$$\mathbf{g}_j^{V} = \rho(\mathbf{r3''}, \mathbf{g}_j^{IV})$$

$$\mathbf{r3''} = \mathbf{r1}*\mathbf{r2}*\mathbf{r3}*\overline{\mathbf{r1}*\mathbf{r2}} = \mathbf{r1}*\mathbf{r2}*\mathbf{r3}*\overline{\mathbf{r2}}*\overline{\mathbf{r1}}$$

$$\mathbf{g}_j^{IV} = \rho(\mathbf{r2'}, \mathbf{g}_j^{III}) = \rho(\rho(\mathbf{r1}, \mathbf{r2}), \rho(\mathbf{r1}, \rho(\mathbf{P}, \rho(\mathbf{Q}, \mathbf{g}_j))))$$

$$\mathbf{g}_j^{IV} = \rho(\mathbf{r1}*\mathbf{r2}*\overline{\mathbf{r1}}, \rho(\mathbf{r1}, \rho(\mathbf{P}, \mathbf{Q}*\mathbf{g}_j*\overline{\mathbf{Q}}))) = \rho(\mathbf{r1}*\mathbf{r2}*\overline{\mathbf{r1}}, \rho(\mathbf{r1}, \mathbf{P}*\mathbf{Q}*\mathbf{g}_j*\overline{\mathbf{Q}}*\overline{\mathbf{P}}))$$

$$\mathbf{g}_j^{IV} = \rho(\mathbf{r1}*\mathbf{r2}*\overline{\mathbf{r1}}, \mathbf{r1}*\mathbf{P}*\mathbf{Q}*\mathbf{g}_j*\overline{\mathbf{Q}}*\overline{\mathbf{P}}*\overline{\mathbf{r1}})$$

$$\mathbf{g}_j^{IV} = \mathbf{r1}*\mathbf{r2}*\overline{\mathbf{r1}}*\mathbf{r1}*\mathbf{P}*\mathbf{Q}*\mathbf{g}_j*\overline{\mathbf{Q}}*\overline{\mathbf{P}}*\overline{\mathbf{r1}}*\overline{\mathbf{r1}*\mathbf{r2}*\overline{\mathbf{r1}}}$$

$$\mathbf{g}_j^{IV} = \mathbf{r1}*\mathbf{r2}*\mathbf{P}*\mathbf{Q}*\mathbf{g}_j*\overline{\mathbf{Q}}*\overline{\mathbf{P}}*\overline{\mathbf{r1}}*\mathbf{r1}*\overline{\mathbf{r1}*\mathbf{r2}}$$

$$\mathbf{g}_j^{IV} = \mathbf{r1}*\mathbf{r2}*\mathbf{P}*\mathbf{Q}*\mathbf{g}_j*\overline{\mathbf{Q}}*\overline{\mathbf{P}}*\overline{\mathbf{r2}}*\overline{\mathbf{r1}}$$

Sustituyendo **r3''** y $\mathbf{g}_j^{IV}$ en $\mathbf{g}_j^{V}$, tenemos que:

$$g_j^V = \rho(\, r1 * r2 * r3 * \overline{r2} * \overline{r1}\ ,\ r1 * r2 * P * Q * g_j * \overline{Q} * \overline{P} * \overline{r2} * \overline{r1}\ )$$

$$g_j^V = r1 * r2 * r3 * \overline{r2} * \overline{r1} * r1 * r2 * P * Q * g_j * \overline{Q} * \overline{P} * \overline{r2} * \overline{r1} * \overline{r1 * r2 * r3 * \overline{r2} * \overline{r1}}$$

$$g_j^V = r1 * r2 * r3 * P * Q * g_j * \overline{Q} * \overline{P} * \overline{r2} * \overline{r1} * r1 * r2 * \overline{r1 * r2 * r3}$$

$$g_j^V = r1 * r2 * r3 * P * Q * g_j * \overline{r1 * r2 * r3 * P * Q}$$

$$g_j^V = \rho(r1 * r2 * r3 * P * Q,\ g_j) = \rho(r1,\ \rho(r2,\ \rho(r3,\ \rho(P,\ \rho(Q,\ g_j)\,)\,)\,)\,)$$

$$g_j^V = \rho(r1 * r2 * r3 * P * Q,\ e_j) = \rho(r1,\ \rho(r2,\ \rho(r3,\ \rho(P,\ \rho(Q,\ e_j)\,)\,)\,)\,) \qquad (3.24)$$

Aplicación de la Segunda Secuencia en la configuración deformada.

Rotación 1.- Rotación del tercer eslabón con r3. Esta rotación solamente afecta al eslabón 3. (Figura 3.11).

$g_j^{III} = \rho(r3,\ g_j^{II}\ )$ (Rotación del tercer eslabón con Quaternión r3)

Donde: $j = 1,2,3$ (3.25)

Rotación 2.- Rotación de los eslabones 2 y 3 con r2. Esta rotación afecta al eje de rotación del tercer eslabón, o sea, al Quaternión r3, el cual ya ha sido aplicado a las bases correspondientes, y por ello no es necesario determinar la nueva forma que adquiere. (Figura 3.12).

$f_j^{II} = \rho(r2,\ f_j^{I}\ )$ (Rotación del segundo eslabón con Quaternión r2)

$g_j^{IV} = \rho(\ r2,\ g_j^{III}\ )$ (Rotación del tercer eslabón con Quaternión r2) (3.26)

Donde: $j = 1,2,3$

Rotación 3.- Rotación de los tres eslabones con r1. Como esta es la última rotación no será necesario actualizar ningún eje de rotación. (Figura 3.13).

$e_j^{I} = \rho(r1,\ e_j\ )$ (Rotación del primer eslabón con Quaternión r1)

$f_j^{III} = \rho(r1,\ f_j^{II}\ )$ (Rotación del segundo eslabón con Quaternión r1) (3.27)

$g_j^{V} = \rho(\ r1,\ g_j^{IV}\ )$ (Rotación del tercer eslabón con Quaternión r1)

Donde: $j = 1,2,3$

De la misma forma que en la primera secuencia de movimientos, la forma de las bases locales de cada eslabón es la siguiente:

Eslabón 1.-

$$e_j^{I} = \rho(r1,\ e_j\ ) \qquad (3.28)$$

Eslabón 2.-

$$f_j^{III} = \rho(r1,\ f_j^{II}\ ) = \rho(r1, \rho(r2,\ f_j^{I}\ )) = \rho(r1, \rho(r2, \rho(P, f_j))) = \rho(r1^*r2^*P, f_j)$$

$$f_j^{III} = \rho(r1, \rho(r2, \rho(P, e_j))) = \rho(r1^*r2^*P, e_j) \qquad (3.29)$$

Eslabón 3.-

$$g_j^{V} = \rho(\ r1,\ g_j^{IV}\ ) = \rho(r1, \rho(\ r2,\ g_j^{III}\ )) = \rho(r1, \rho(\ r2, \rho(r3,\ g_j^{II}\ )))$$
$$g_j^{V} = \rho(r1, \rho(r2, \rho(r3, \rho(P, \rho(Q, g_j))))) = \rho(r1^*r2^*r3^*P^*Q, g_j)$$

$$g_j^{V} = \rho(r1, \rho(r2, \rho(r3, \rho(P, \rho(Q, e_j))))) = \rho(r1^*r2^*r3^*P^*Q, e_j) \qquad (3.30)$$

Al igual que en la configuración no deformada, la forma final de las bases locales en las dos secuencias es la misma, cambiando únicamente las posiciones intermedias, las cuales pueden observase en las figuras 3.8 a 3.13. En estas figuras podemos observar que existen dos puntos intermedios en cada secuencia y las ecuaciones que los representan se mostrarán, en su forma no explícita, a continuación:

Puntos Intermedios en la Primera Secuencia.

Punto Intermedio 1. (Figura 3.8)

$b_1 = L_1{}^* e_3^{I} = L_1{}^*\rho(r1, e_3)$; (Eslabón 1)
$b_2 = L_2{}^* f_3^{II} = L_2{}^*\rho(r1, f_3^{I})$; (Eslabón 2) (3.31)
$b_3 = L_3{}^* g_3^{III} = L_3{}^*\rho(r1, g_3^{II})$; (Eslabón 3)
$R_{p1} = b_1 + b_2 + b_3$; (Posición del elemento terminal) (3.32)

Punto Intermedio 2. (Figura 3.9)

$b_1 = L_1{}^* e_3^{I} = L_1{}^*\rho(\ r1, e_3)$; (Eslabón 1)
$b_2 = L_2{}^* f_3^{III} = L_2{}^*\rho(r2^{I},\ f_3^{II}\ ) = L_2{}^*\rho(\rho(r1, r2),\ f_3^{II}\ )$; (Eslabón 2) (3.33)
$b_3 = L_3{}^* g_3^{IV} = L_3{}^*\rho(\ r2^{I}, {}^*\rho(\ r1,\ g_3^{II}\ )) = L_3{}^*\rho(\rho(r1, r2), \rho(\ r1,\ g_3^{II}\ ))$; (Eslabón 3)
$R_{p2} = b_1 + b_2 + b_3$; (Posición del elemento terminal) (3.34)

Puntos Intermedios en la Segunda Secuencia.

Punto Intermedio 3. (Figura 3.11)

$b_1 = L_1{}^* e_3$; (Eslabón 1)
$b_2 = L_2{}^* f_3^{II}$; (Eslabón 2) (3.35)
$b_3 = L_3{}^* g_3^{III} = L_3{}^*\rho(r3,\ g_3^{II}\ )$; (Eslabón 3)
$R_{p3} = b_1 + b_2 + b_3$; (Posición del elemento terminal) (3.36)

Punto Intermedio 4. (Figura 3.12)

$\mathbf{b_1 = L_1 {}^* e_3}$; (Eslabón 1)
$\mathbf{b_2 = L_2 {}^* f_3^{II} = L_2 {}^* \rho(r2,\ f_3^{I})}$; (Eslabón 2) (3.37)
$\mathbf{b_3 = L_3 {}^* g_3^{IV} = L_3 {}^* \rho(r2,\ g_j^{III}) = L_3 {}^* \rho(r2, \rho(r3,\ g_3^{II}))}$; (Eslabón 3)
$\mathbf{R_p = b_1 + b_2 + b_3}$; (Posición del elemento terminal) (3.38)

3.1.5. Vectores que definen los eslabones y posición del elemento terminal en la configuración deformada.

De las figuras Figura 3.13 y Figura 3.14 se determinan los vectores que definen los eslabones del manipulador y el vector del posición del elemento terminal (punto p). Las longitudes de los eslabones son $\mathbf{L_1, L_2, L_3}$.

$\mathbf{b_1 = L_1 {}^* e_3^{I}}$; **(Eslabón 1)**
$\mathbf{b_2 = L_2 {}^* f_3^{III}}$; **(Eslabón 2)** (3.39)
$\mathbf{b_3 = L_3 {}^* g_3^{V}}$; **(Eslabón 3)**

$\mathbf{r_p = b_1 + b_2 + b_3}$ **(Posición del elemento terminal)** (3.40)

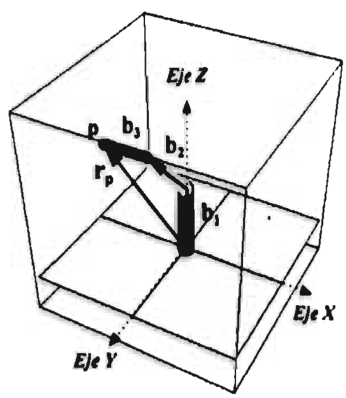


Figura 3.14. *Vector de posición del elemento terminal en la configuración no deformada.*

3.1.6. Ecuaciones explícitas optimizadas para la configuración deformada.

Las expresiones anteriores se explicitaron y optimizaron utilizando Mathematica, y se presentan a continuación.

$\mathbf{b_1 = \{0, 0, L_1\}}$
$\mathbf{b_2 = \{-4{}^*A5{}^*B2{}^*L_2, -2{}^*A6{}^*B2{}^*L_2, B1{}^*L_2\}}$ (3.41)
$\mathbf{b_3 = \{-4{}^*A5{}^*C1{}^*L_3, -2{}^*A6{}^*C1{}^*L_3, C2{}^*L_3\}}$

$$r_p = \{-4*A5*(B2*L_2 + C1*L_3), -2*A6*(B2*L_2 + C1*L_3), L_1 + B1*L_2 + C2*L_3\} \quad (3.42)$$

Donde:

$$\begin{aligned}
&A1 = P_0*P_1; \quad &A2 = 1-2*P_0^2;\\
&A3 = Q_0*Q_1; \quad &A4 = 1-2*Q_0^2;\\
&A5 = r1_0*r1_1; \quad &A6 = 1-2*r1_0^2;\\
&A7 = r2_0*r2_3; \quad &A8 = 1-2*r2_0^2;\\
&A9 = r3_0*r3_3; \quad &A10 = 1-2*r3_0^2;\\
\\
&B1 = A2*A8 - 4*A1*A7; \quad &B2 = A2*A7 + A1*A8;\\
&B3 = A10*A8 - 4*A7*A9; \quad &B4 = A10*A7 + A8*A9;\\
&B5 = A2*A4 - 4*A1*A3; \quad &B6 = A2*A3 + A1*A4;\\
\\
&C1 = B4*B5 + B3*B6; \quad &C2 = B3*B5 - 4*B4*B6;
\end{aligned} \quad (3.43)$$

De las dos secuencias utilizadas en la modelación mostrada, y habiendo realizado otras secuencias que no se presentan aquí, podemos concluir que:

1.- Las secuencias de movimientos, en el proceso de modelación, no afecta la forma final de las ecuaciones resultantes, pero se presentan diferentes configuraciones intermedias en cada secuencia.

2.- Las secuencias de movimientos pueden facilitar o dificultar la obtención de las ecuaciones finales, ya que en unas secuencias se requiere actualizar ejes de rotación en los Quaterniones correspondientes (*Primera Secuencia*), lo que provoca que las ecuaciones resultantes sean más complejas y deban ser simplificadas utilizando propiedades de las operaciones con Quaterniones, mientras que en otras secuencias de movimientos no se requiere actualizar ejes de rotación (*Segunda Secuencia*), lo que permite que las ecuaciones obtenidas ya tengan su forma final simplificada.

3.- La forma más eficiente de modelar la cinemática de cadenas abiertas con Quaterniones es iniciando con el eslabón terminal y finalizar con el eslabón base.

4.- La forma general de la rotación de la base local de un eslabón cualquiera en la configuración deformada de una cadena abierta con juntas rotacionales, es:

$$e^n_j = \rho(r_1 \cdot r_2 * \ldots * r_n, e^x_j) \quad (3.44)$$

Donde:

e^n_j = Base local del eslabón n.

r_1, r_2, r_n = Quaterniones que rotan los eslabones 1, 2, ... n. Siendo el eslabón base el número uno, y el eslabón n, el eslabón analizado.

e^x_j = Base local del eslabón n en la configuración no deformada.

n = Número de eslabón analizado.

La numeración de los eslabones se realiza empezando en el eslabón base, es decir en el que se soporta todo el mecanismo, y se continua secuencialmente hasta finalizar en el eslabón terminal.

3.2. MODELACION CINEMATICA DE UN MANIPULADOR DE TRES GRADOS DE LIBERTAD, UTILIZANDO ROTACIONES Y REFLEXIONES.

OBJETIVO: El objetivo en este caso será determinar un modelo cinemático diferente al obtenido utilizando solamente rotaciones, el cual servirá para ubicar los eslabones del manipulador en una posición alterna en la cinemática inversa, esto con el propósito de evitar un posible obstáculo que pudiese impedir el movimiento del manipulador.

Para realizar la modelación, partiremos de la configuración no deformada utilizada en la modelación con rotaciones (Figura 3.5), aplicaremos una rotación al primer eslabón del manipulador y posteriormente aplicaremos reflexiones a los otros dos eslabones, la actualización de los ejes de rotación se realizará utilizando la transformación correspondiente, solo que tomaremos como eje de rotación el de sentido negativo, que es equivalente a realizar una actualización de eje de rotación solo con reflexiones como se verá más delante; por lo anterior procederemos a obtener directamente la configuración deformada. Debemos observar que la configuración inicial del manipulador es diferente a la que se obtuvo para la modelación con rotaciones (Figura 3.15).

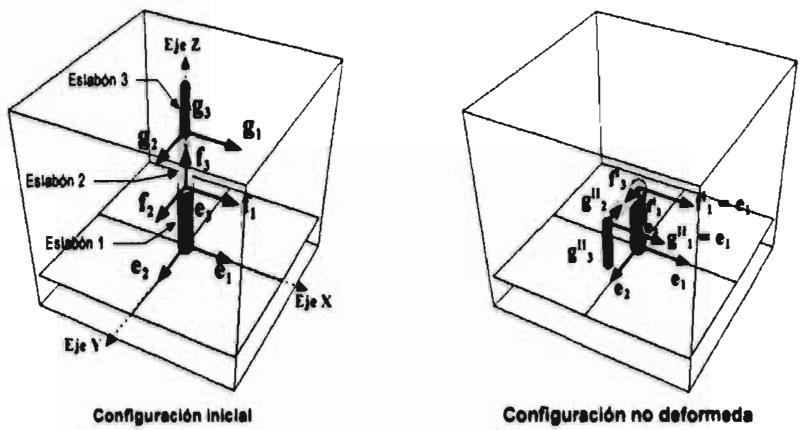


Figura 3.15. *Configuración inicial y no deformada del manipulador de 3 grados de libertad.*

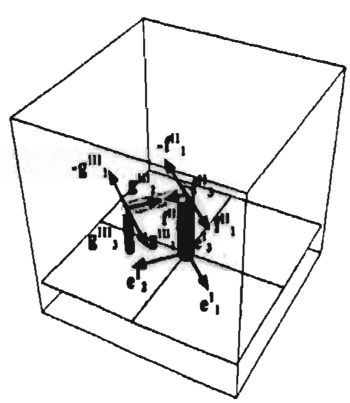


Figura 3.16. *Rotación de los tres eslabones en la configuración deformada.*

3.2.1. Configuración deformada.

Para obtener la configuración deformada utilizaremos la siguiente secuencia.

Rotación 1.- rotan los tres eslabones con el Quaternión **r1**. (Figura 3.16).

Reflexión 1.- se aplican reflexiones a los eslabones 2 y 3 con el Quaternión **r2**. (Figura 3.17).

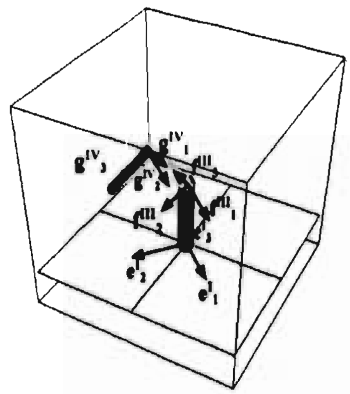


Figura 3.17. *Reflexión de los eslabones 2 y 3 en la configuración deformada.*

Reflexión 2.- se aplica reflexión al eslabón 3 con el Quaternión **r3**. (Figura 3.18).

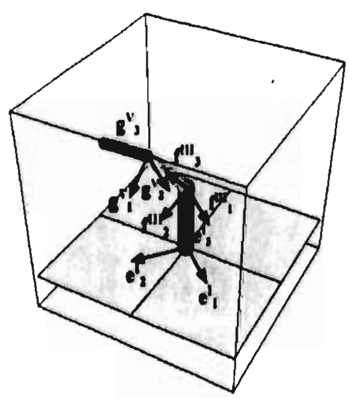


Figura 3.18. *Reflexión del eslabón 3 en la configuración deformada.*

Los Quaterniones que se utilizarán para el giro de los eslabones serán los mismos que los utilizados en el modelo realizado solamente con rotaciones, es decir, la forma local de dichos Quaterniones es::

Eslabón 1: $r1 = \{ r1_0, r1_1, 0, 0\}$, Donde:

$r1_0 = Cos(\theta_1/2);$
$r1_1 = Sen(\theta_1/2);$

Siendo θ_1 el ángulo de rotación del eslabón 1 en la configuración deformada.

Eslabón 2: $r2 = \{ r2_0, 0, 0, r2_3\}$.

$r2_0 = Sen(\theta_2/2);$
$r2_1 = Cos(\theta_2/2);$

Siendo θ_2 el ángulo de rotación del eslabón 2 en la configuración deformada.

Eslabón 3: $r3 = \{ r3_0, 0, 0, r3_3\}$.

$r3_0 = Sen(\theta_3/2);$
$r3_1 = Cos(\theta_3/2);$

Siendo θ_3 el ángulo de rotación del eslabón 3 en la configuración deformada.

Con lo anterior procedemos a la aplicación de las rotaciones y reflexiones mencionadas.

Rotación 1.- Rotación de los tres eslabones con r1:

En la Figura 3.16 podemos observar que todas las bases locales son afectadas por la rotación **r1** (rotación de los tres eslabones), a continuación se muestran dichas bases.

$$\mathbf{e}_j^{I} = \rho(\mathbf{r1}, \mathbf{e}_j) \quad \text{(Rotación del primer eslabón con Quaternión r1)}$$
$$\mathbf{f}_j^{II} = \rho(\mathbf{r1}, \mathbf{f}_j^{I}) \quad \text{(Rotación del segundo eslabón con Quaternión r1)} \qquad (3.45)$$
$$\mathbf{g}_j^{III} = \rho(\mathbf{r1}, \mathbf{g}_j^{II}) \quad \text{(Rotación del tercer eslabón con Quaternión r1)}$$

Donde: $j = 1,2,3$

Los ejes de rotación del segundo y tercer eslabones son afectados por **r1**, es decir los nuevos ejes de rotación serían $\mathbf{f}_1^{II}$ y $\mathbf{g}_1^{III}$ según la modelación realizada con rotaciones, pero en este caso se realizará la variante mencionada anteriormente, es decir, los nuevos ejes de rotación serán $-\mathbf{f}_1^{II}$ y $-\mathbf{g}_1^{III}$.

$$-\mathbf{f}_1^{II} = -\rho(\mathbf{r1}, \mathbf{f}_1^{I}) \quad \text{(Eje de rotación del eslabón 2).}$$
$$-\mathbf{g}_1^{III} = -\rho(\mathbf{r1}, \mathbf{g}_1^{II}) \quad \text{(Eje de rotación del eslabón 3).} \qquad (3.46)$$

Por lo anterior la forma actualizada de los Quaterniones **r2** y **r3** es:

$$r2^I = \{r2_0, r2^I_v\}; \quad r2^I_v = Cos(\theta_2/2)^* (-f_1^{II}) = Cos(\theta_2/2)^*(-\rho(r1, f_1^I))$$

$$r2^I_v = -Cos(\theta_2/2)^* \rho(r1, e_1) = \rho(r1, -r2_v)$$

$$\Rightarrow \quad r2^I = \overline{\rho(r1,r2)} = R(r1,r2) \qquad (3.47)$$

$$r3^I = \{r3_0, r3^I_v\}; \quad r3^I_v = Cos(\theta_3/2)^* (-g_1^{III}) = Cos(\theta_3/2)^*(-\rho(r1, g_1^{II}))$$

$$r3^I_v = -Cos(\theta_3/2)^* \rho(r1, e_1) = \rho(r1, -r3_v)$$

$$\Rightarrow \quad r2^I = \overline{\rho(r1,r3)} = R(r1,r3) \qquad (3.48)$$

Después de aplicada la primera rotación, los Quaterniones correspondientes a los eslabones 2 y 3 son **$r2^I$** y **$r3^I$** respectivamente, los cuales han sido actualizados con respecto al sistema inercial fijo mediante una Reflexión como se mostró, y su forma explícita es la siguiente:

$$r2^I = R(r1, r2) = \{r2_0, (1 - 2^*r1_0^2)^*r2_1, -2^*r1_0^*r1_3^*r2_1, 0\}.$$

$$r3^I = R(r1, r3) = \{r3_0, (1 - 2^*r1_0^2)^*r3_1, -2^*r1_0^*r1_3^*r3_1, 0\}. \qquad (3.49)$$

Reflexión 1.- Reflexión de los eslabones 2 y 3 con Quaternión $r2^I$:

Para este caso observamos que las bases locales afectadas por la reflexión **$r2^I$** son las que corresponden a los eslabones 2 y 3 ($\mathbf{f}_j$ y $\mathbf{g}_j$). (Figura 3.17)

$$\mathbf{f}_j^{III} = R(r2^I, \mathbf{f}_j^{II}) \qquad \text{(Reflexión del segundo eslabón con Quaternión } r2^I \text{)}$$

$$\mathbf{g}_j^{IV} = R(r2^I, \mathbf{g}_j^{III}) \qquad \text{(Reflexión del tercer eslabón con Quaternión } r2^I \text{)} \qquad (3.50)$$

Donde: $j = 1,2,3$

La reflexión anterior afecta el eje de giro del tercer eslabón, que al igual que en el caso anterior, tomaremos el sentido negativo del eje de rotación que es $-\mathbf{g}_1^{IV}$, por ello el Quaternión **$r3^I$** se transforman de la siguiente forma:

$$r3^{II} = \{r3_0, r3^{II}_v\};$$

$$r3^{II}_v = Cos(\theta_3/2)^* -\mathbf{g}_1^{IV} = Cos(\theta_3/2)^* R(r2^I, -\mathbf{g}_1^{III}) = Cos(\theta_3/2)^* R(r2^I, (-\rho(r1, \mathbf{g}_1^{II})))$$

$$r3^{II}_v = R(r2^I, -Cos(\theta_3/2)^*\rho(r1, \mathbf{g}_1^{II})) = R(r2^I, r3^I_v) \qquad \Rightarrow \qquad r3^{II} = R(r2^I, r3^I)$$

$$r3^{II} = R(r2^I, r3^I) = R(\rho(r1, r2), R(r1, r3)) = R(r1 * \overline{r2} * \overline{r1}, r1 * \overline{r3} * \overline{r1})$$

$$r3^{II} = r1 * \overline{r2} * \overline{r1} * \overline{r1 * \overline{r3} * \overline{r1}} * \overline{r1 * \overline{r2} * \overline{r1}} = r1 * \overline{r2} * \overline{r1} * r1 * r3 * \overline{r1} * r1 * r2 * \overline{r1}$$

$$r3^{II} = r1 * \overline{r2} * r3 * r2 * \overline{r1} = r1 * \overline{r2} * r3 * \overline{r1 * \overline{r2}} = \rho((r1 * \overline{r2}), r3) \qquad (3.51)$$

Ahora, después de aplicadas las rotaciones anteriores el Quaternión que caracteriza el giro del tercer eslabón es $r3^{II}$, y su forma explicita es:

$$r3^{II} = R(r2^I, r3^I) = \{r3_0, (-1 + 2^*r1_0^2)^*r3_1, 2^*r1_0^*r1_3^*r3_1, 0\}. \qquad (3.52)$$

Reflexión 2.- Reflexión del tercer eslabón 3 con Quaternión $r3''$:

Por último se tiene la reflexión del tercer eslabón con el Quaternión **r3''**, con lo que se modifica la base local del eslabón 3. (Figura 3.18).

$$g_j^{V} = \rho(r3'', g_j^{IV}) \qquad \text{(Reflexión del tercer eslabón con Quaternión r3'')} \qquad (3.53)$$

Donde: $j = 1,2,3$

Finalmente las bases que representan a cada eslabón son:

Eslabón 1:

$$e_j^{I} = \rho(r1, e_j) \qquad (3.54)$$

Eslabón 2:

$$f_j^{III} = \rho(r2', f_j^{II}) = R(R(r1, r2), \rho(r1, f_j^{I})) = R(r1 \cdot \bar{r2} \cdot \bar{r1}, \rho(r1, \rho(P, f_j)))$$

$$f_j^{III} = R(r1 \cdot \bar{r2} \cdot \bar{r1}, r1 \cdot P \cdot f_j \cdot \bar{P} \cdot \bar{r1}) = r1 \cdot \bar{r2} \cdot \bar{r1} \cdot \overline{r1 \cdot P \cdot f_j \cdot \bar{P} \cdot \bar{r1}} \cdot \overline{r1 \cdot \bar{r2} \cdot \bar{r1}}$$

$$f_j^{III} = r1 \cdot \bar{r2} \cdot \bar{r1} \cdot r1 \cdot P \cdot \bar{f_j} \cdot \bar{P} \cdot \bar{r1} \cdot r1 \cdot r2 \cdot \bar{r1} = r1 \cdot \bar{r2} \cdot P \cdot \bar{f_j} \cdot \bar{P} \cdot r2 \cdot \bar{r1}$$

$$f_j^{III} = r1 \cdot \bar{r2} \cdot P \cdot \bar{f_j} \cdot \bar{P} \cdot r2 \cdot \bar{r1} = r1 \cdot \bar{r2} \cdot P \cdot \bar{f_j} \cdot \overline{r1 \cdot \bar{r2} \cdot P}$$

$$f_j^{III} = R(r1 \cdot \bar{r2} \cdot P, f_j) \qquad (3.55)$$

Eslabón 3:

$$g_j^{V} = R(r3'', g_j^{IV})$$

$$r3'' = r1 \cdot \bar{r2} \cdot r3 \cdot r2 \cdot \bar{r1}$$

$$g_j^{IV} = R(r2', g_j^{III}) = R(r1 \cdot \bar{r2} \cdot \bar{r1}, \rho(r1, \rho(P, \rho(Q, g_j))))$$

$$g_j^{IV} = R(r1 \cdot \bar{r2} \cdot \bar{r1}, r1 \cdot P \cdot Q \cdot g_j \cdot \bar{Q} \cdot \bar{P} \cdot \bar{r1}) = r1 \cdot \bar{r2} \cdot \bar{r1} \cdot \overline{r1 \cdot P \cdot Q \cdot g_j \cdot \bar{Q} \cdot \bar{P} \cdot \bar{r1}} \cdot \overline{r1 \cdot \bar{r2} \cdot \bar{r1}}$$

$$g_j^{IV} = r1 \cdot \bar{r2} \cdot \bar{r1} \cdot r1 \cdot P \cdot Q \cdot \bar{g_j} \cdot \bar{Q} \cdot \bar{P} \cdot \bar{r1} \cdot r1 \cdot r2 \cdot \bar{r1} = r1 \cdot \bar{r2} \cdot P \cdot Q \cdot \bar{g_j} \cdot \bar{Q} \cdot \bar{P} \cdot r2 \cdot \bar{r1}$$

$$g_j^{IV} = r1 \cdot \bar{r2} \cdot P \cdot Q \cdot \bar{g_j} \cdot \bar{Q} \cdot \bar{P} \cdot r2 \cdot \bar{r1}$$

Sustituyendo **r3''** y g_j^{IV} en g_j^{V}, tenemos que:

$$g_j^{V} = \rho(r1 \cdot \bar{r2} \cdot r3 \cdot r2 \cdot \bar{r1}, r1 \cdot \bar{r2} \cdot P \cdot Q \cdot \bar{g_j} \cdot \bar{Q} \cdot \bar{P} \cdot r2 \cdot \bar{r1})$$

$$g_j^{V} = r1 \cdot \bar{r2} \cdot r3 \cdot r2 \cdot \bar{r1} \cdot \overline{r1 \cdot \bar{r2} \cdot P \cdot Q \cdot \bar{g_j} \cdot \bar{Q} \cdot \bar{P} \cdot r2 \cdot \bar{r1}} \cdot \overline{r1 \cdot \bar{r2} \cdot r3 \cdot r2 \cdot \bar{r1}}$$

$$g_j^{V} = r1 \cdot \bar{r2} \cdot r3 \cdot r2 \cdot \bar{r1} \cdot r1 \cdot \bar{r2} \cdot P \cdot Q \cdot g_j \cdot \bar{Q} \cdot \bar{P} \cdot r2 \cdot \bar{r1} \cdot r1 \cdot \bar{r2} \cdot \bar{r3} \cdot r2 \cdot \bar{r1}$$

$$g_j^{V} = r1 * \overline{r2} * r3 * P * Q * g_j * \overline{Q} * \overline{P} * \overline{r3} * r2 * \overline{r1} = r1 * \overline{r2} * r3 * P * Q * g_j * \overline{r1 * \overline{r2} * r3 * P * Q}$$

$$g_j^{V} = \rho(r1 * \overline{r2} * r3 * P * Q, g_j) \qquad (3.56)$$

3.2.2. Vectores que definen los eslabones y posición del elemento terminal en la configuración deformada.

De las figuras Figura 3.18 y Figura 3.19 se determinan los vectores que definen los eslabones del manipulador y el vector del posición del elemento terminal (punto p). Las longitudes de los eslabones son L_1, L_2, L_3.

$$\begin{aligned} b_1 &= L_1 * e_3^{I}; \quad (\text{ Eslabón 1 }) \\ b_2 &= L_2 * f_3^{III}; \quad (\text{ Eslabón 2 }) \\ b_3 &= L_3 * g_3^{V}; \quad (\text{ Eslabón 3 }) \end{aligned} \qquad (3.57)$$

$$r_p = b_1 + b_2 + b_3 \quad (\text{ Posición del elemento terminal }) \qquad (3.58)$$

3.2.3. Ecuaciones explícitas optimizadas para la configuración deformada.

$$\begin{aligned} b_1 &= \{0, 0, L_1\}; \\ b_2 &= \{4*A5*B2*L_2, 2*A6*B2*L_2, -B1*L_2\}; \\ b_3 &= \{4*A5*C1*L_3, 2*A6*C1*L_3, C2*L_3\}; \end{aligned} \qquad (3.59)$$

$$r_p = \{4*A5*(B2*L_2 + C1*L_3), 2*A6*(B2*L_2 + C1*L_3), L_1 - B1*L_2 + C2*L_3\}; \qquad (3.60)$$

Donde:

$$\begin{aligned} &A1 = P_0*P_1; \quad A2 = 1-2*P_0^2; \quad A3 = Q_0*Q_1; \quad A4 = 1-2*Q_0^2; \\ &A5 = r1_0*r1_1; \quad A6 = 1-2*r1_0^2; \quad A7 = r2_0*r2_1; \quad A8 = 1-2*r2_0^2; \\ &A9 = r3_0*r3_1; \quad A10 = 1-2*r3_0^2; \\ &B1 = A2*A8 + 4*A1*A7; \quad B2 = A1*A8 - A2*A7; \\ &B3 = A10*A8 + 4*A7*A9; \quad B4 = A10*A7 - A8*A9; \\ &B5 = A2*A3 + A1*A4; \quad B6 = A2*A4 - 4*A1*A3; \\ &C1 = B4*B6 - B3*B5; \quad C2 = B3*B6 + 4*B4*B5; \end{aligned} \qquad (3.61)$$

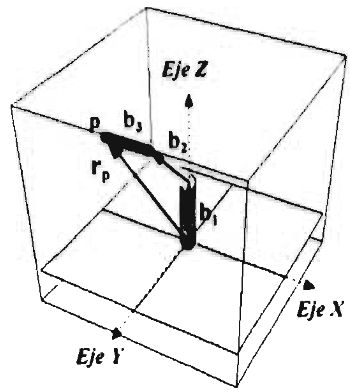


Figura 3.19. *Vector de posición del elemento terminal en la configuración deformada.*

Utilizando los dos modelos cinemáticos anteriores de manera conjunta, se puede hacer que el manipulador siga una trayectoria determinada evitando algunos obstáculos, al utilizar un modelo u otro, dependiendo de la ubicación del obstáculo en el espacio. Esto se verá en las simulaciones cinemáticas que se mostraran más adelante.

3.3. MODELACIÓN CINEMÁTICA DE UN MANIPULADOR DE CUATRO GRADOS DE LIBERTAD, UTILIZANDO ROTACIONES.

En esta sección mostraremos la modelación de un manipulador de cuatro grados de libertad, aplicando la secuencia más sencilla determinada en la modelación del manipulador de tres grados de libertad, tanto para la configuración deformada como para la no deformada..

3.3.1. Configuración no deformada.

Mostraremos primeramente la modelación de la configuración no deformada, que consiste en determinar un sistema de ecuaciones que nos permitan determinar la posición del segundo y tercer eslabón, cuando el primer eslabón se encuentra en una posición inicial de referencia (home), conociendo la posición del extremo del tercer eslabón, que consideraremos para este caso como la posición del elemento terminal del manipulador.

Secuencia de Rotaciones:

1.- Se mantiene fijo el el cuarto eslabón.
Rotación 2.- Se aplica una rotación al tercer eslabón. (Figura 3.21).
Rotación 3.- Se aplica una rotación al segundo eslabón. (Figura 3.22).
4.- Se mantiene fijo el primer eslabón.

Para lo anterior definiremos la configuración no deformada del manipulador, que consiste en definir el vector de posición del elemento terminal para posiciones arbitrarias del segundo y tercer eslabones en un plano dado. Consideremos la base canónica de $\Re^3$, esto es:

$$\mathbf{e}_1 = (1,0,0); \quad \mathbf{e}_2 = (0,1,0); \quad \mathbf{e}_3 = (0,0,1);$$

Ubicamos sistemas de referencia locales en la base de cada eslabón del manipulador como se muestra en la Figura 3.20. Observemos que, en este caso, los sistemas de referencia definidos coinciden con la base canónica la cual sirve como sistema de referencia inercial fijo, es decir:

$$\mathbf{h}_j = \mathbf{g}_j = \mathbf{f}_j = \mathbf{e}_j \ , \quad j = 1,2,3.$$

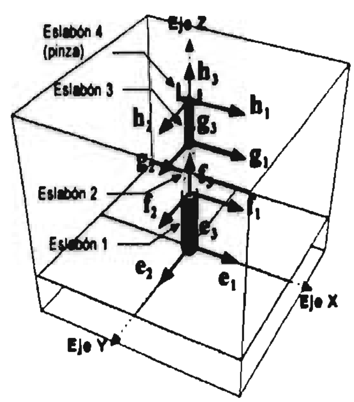


Figura 3.20. *Bases locales en la configuración original para la modelación.*

De la posición mostrada en la Figura 3.20, debemos permitir que los eslabones 2 y 3 giren a una posición arbitraria, considerando que los eslabones tienen los siguientes ejes de rotación:

Eslabón	Eje De Rotación
1	e_3
2	$f_1 = e_1$
3	$g_1 = e_1$
4	$h_3 = e_3$

Con los ejes de rotación anteriores, podemos definir la forma que tendrán inicialmente los Quaterniones que nos permitirán rotar los eslabones 2 y 3.

Eslabón 2: Consideremos el siguiente parámetro que caracterizará a nuestra primera rotación;

$$P = \{P_0, P_1, P_2, P_3\} \quad \Rightarrow \quad P_v = \{P_1, P_2, P_3\}, \quad \|P\| = 1.$$

En este caso se tiene que si $\underline{w} \in \Re^3$ es el eje de rotación, entonces, [1]:

$$P_0 = Cos(\varphi_1/2)$$
$$P_v = Sin(\varphi_1/2)^*\underline{w} = Sin(\varphi_1/2)^*f_1 = Sin(\varphi_1/2)^*e_1 = Sin(\varphi_1/2)^*\{1,0,0\}$$
$$P_v = \{P_1, 0,0\}$$

De lo anterior se obtiene la forma local del parámetro P, que es: $\mathbf{P = \{P_0, P_1, 0, 0\}}$, Donde:

$$P_0 = Cos(\varphi_1/2);$$
$$P_1 = Sen(\varphi_1/2);$$

Siendo φ_1 el ángulo de rotación del eslabón 2.

Eslabón 3: Consideremos ahora el siguiente parámetro que caracterizará la segunda rotación;

$$Q = \{Q_0, Q_1, Q_2, Q_3\} \quad \Rightarrow \quad Q_v = \{Q_1, Q_2, Q_3\}, \quad \|Q\| = 1.$$

Siendo $\underline{w} \in \Re^3$ el eje de rotación, tenemos que:

$$Q_v = Sin(\varphi_2/2)^*\underline{w} = Sin(\varphi_2/2)^*g_1 = Sin(\varphi_2/2)^*e_1 = Sin(\varphi_2/2)^*\{1,0,0\}$$
$$Q_v = \{ Q_1, 0, 0\}$$

De lo anterior se obtiene la forma local del parámetro Q, que es: $\mathbf{Q = \{Q_0, Q_1, 0, 0\}}$, Donde:

$$Q_0 = Cos(\varphi_2/2);$$
$$Q_1 = Sen(\varphi_2/2);$$

Siendo φ_2 el ángulo de rotación del eslabón 3.

Con los Quaterniones anteriores podemos ahora realizar las rotaciones necesarias para determinar la configuración no deformada, se debe observar que si los ejes de rotación cambian su forma, también cambiará la forma de los Quaterniones dados.

Aplicación de la Secuencia de rotaciones para determinar la configuración no deformada.

Rotación 2.- Rotamos el tercer eslabón con el Quaternión **Q**. Esto se realiza rotando las bases locales $\mathbf{g}_j$ y $\mathbf{h}_j$. (Figura 3.21)

$$\mathbf{g}_j^{\prime} = \rho(\mathbf{Q}, \mathbf{g}_j) \quad \text{(Rotación del tercer eslabón con Quaternión Q)},\ j = 1,2,3$$
$$\mathbf{h}_j^{\prime} = \rho(\mathbf{Q}, \mathbf{h}_j) \quad \text{(Rotación del cuarto eslabón con Quaternión Q)},\ j = 1,2,3 \qquad (3.62)$$

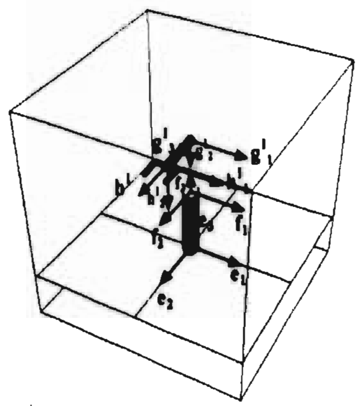


Figura 3.21. *Rotación del tercer eslabón en la configuración no deformada.*

Rotación 3.- Se aplica la rotación al segundo eslabón con el Quaternión P, lo que implica que las bases $\mathbf{f}_j$, $\mathbf{g}_j$ y $\mathbf{h}_j$. sufren las siguientes transformaciones.(Figura 3.22)

$$\mathbf{f}_j^{\prime} = \rho(\mathbf{P}, \mathbf{f}_j) \quad \text{(Rotación del segundo eslabón con Quaternión P)}$$
$$\mathbf{g}_j^{\prime\prime} = \rho(\mathbf{P}, \mathbf{g}_j^{\prime}) \quad \text{(Rotación del tercer eslabón con Quaternión P).} \qquad (3.63)$$
$$\mathbf{h}_j^{\prime\prime} = \rho(\mathbf{Q}, \mathbf{h}_j^{\prime}) \quad \text{(Rotación del cuarto eslabón con Quaternión P)},\ j = 1,2,3$$

Como $\mathbf{g}_j^{\prime} = \rho(\mathbf{Q}, \mathbf{g}_j)$ y $\mathbf{h}_j^{\prime} = \rho(\mathbf{Q}, \mathbf{h}_j)$, tenemos que:

$$\mathbf{g}_j^{\prime\prime} = \rho(\mathbf{P}, \rho(\mathbf{Q}, \mathbf{g}_j)) = \rho((\mathbf{P}^*\mathbf{Q}), \mathbf{g}_j) \qquad (3.64)$$
$$\mathbf{h}_j^{\prime\prime} = \rho(\mathbf{P}, \rho(\mathbf{Q}, \mathbf{h}_j)) = \rho((\mathbf{P}^*\mathbf{Q}), \mathbf{h}_j)$$

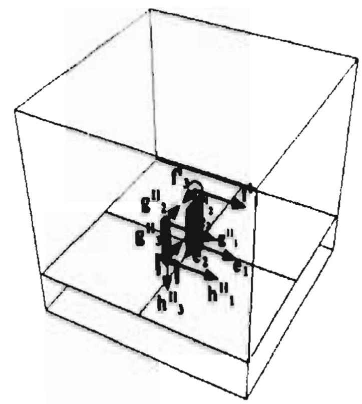


Figura 3.22. *Rotación de los eslabones 2 y 3 en la configuración no deformada.*

La forma explícita de las bases locales, después de aplicadas las rotaciones anteriores, es la siguiente:

$$
\begin{aligned}
f_1^{I} &= \{1, 0, 0\} \\
f_2^{I} &= \{0, -1 + 2*P_0^2, 2*P_0*P_1\} \\
f_3^{I} &= \{0, -2*P_0*P_1, -1 + 2*P_0^2\}
\end{aligned}
\tag{3.65}
$$

$$
\begin{aligned}
g_1^{II} &= \{1, 0, 0\} \\
g_2^{II} &= \{0, -P_0^2 + P_1^2 + 2*P_0^{2}*Q_0^2 - 2*P_1^{2}*Q_0^2 - 4*P_0*P_1*Q_0*Q_1, \\
&\qquad 2*(-(P_0*P_1) + 2*P_0*P_1*Q_0^2 + P_0^{2}*Q_0*Q_1 - P_1^{2}*Q_0*Q_1)\} \\
g_3^{II} &= \{0, 2*(P_0*P_1 - 2*P_0*P_1*Q_0^2 - P_0^{2}*Q_0*Q_1 + P_1^{2}*Q_0*Q_1), \\
&\qquad - P_0^2 + P_1^2 + 2*P_0^{2}*Q_0^2 - 2*P_1^{2}*Q_0^2 - 4*P_0*P_1*Q_0*Q_1\}
\end{aligned}
\tag{3.66}
$$

$$
\begin{aligned}
h_1^{II} &= \{1, 0, 0\} \\
h_2^{II} &= \{0, -P_0^2 + P_1^2 + 2*P_0^{2}*Q_0^2 - 2*P_1^{2}*Q_0^2 - 4*P_0*P_1*Q_0*Q_1, \\
&\qquad 2*(-(P_0*P_1) + 2*P_0*P_1*Q_0^2 + P_0^{2}*Q_0*Q_1 - P_1^{2}*Q_0*Q_1)\} \\
h_3^{II} &= \{0, 2*(P_0*P_1 - 2*P_0*P_1*Q_0^2 - P_0^{2}*Q_0*Q_1 + P_1^{2}*Q_0*Q_1), \\
&\qquad - P_0^2 + P_1^2 + 2*P_0^{2}*Q_0^2 - 2*P_1^{2}*Q_0^2 - 4*P_0*P_1*Q_0*Q_1\}
\end{aligned}
\tag{3.67}
$$

3.3.2. Vectores que definen los eslabones y posición del elemento terminal en la configuración no deformada.

De la Figura 3.22 y Figura 3.24 podemos determinar los vectores que definen los eslabones del manipulador y el vector de posición del elemento terminal (punto p). Siendo $\mathbf{L_1}$, $\mathbf{L_2}$, $\mathbf{L_3}$, $\mathbf{L_4}$ y $\mathbf{L_6}$ las longitudes de los eslabones.

$\mathbf{b}_1 = \mathbf{L}_1{}^* \mathbf{e}_3$; (Eslabón 1)

$\mathbf{b}_2 = \mathbf{L}_2{}^* \mathbf{f}_3^{I} = \mathbf{L}_2{}^* \rho(\mathbf{P}, \mathbf{f}_3) = \mathbf{L}_2{}^* (\mathbf{P} * \mathbf{e}_3 * \bar{\mathbf{P}})$; (Eslabón 2) (3.68)

$\mathbf{b}_3 = \mathbf{L}_3{}^* \mathbf{g}_3^{II} = \mathbf{L}_3{}^* \rho((\mathbf{P}^*\mathbf{Q}), \mathbf{g}_3) = \mathbf{L}_3{}^* ((\mathbf{P}^*\mathbf{Q}) * \mathbf{e}_3 * (\overline{\mathbf{P} * \mathbf{Q}}))$; (Eslabón 3)

$\mathbf{R}_p = \mathbf{b}_1 + \mathbf{b}_2 + \mathbf{b}_3$ (Posición del elemento terminal) (3.69)

De la Figura 3.22 podemos observar que la base local $\mathbf{h}_j^{II}$ es la que define la pinza o eslabón 4 del manipulador, de la misma figura se puede ver que son necesarios cuando menos dos vectores para definir dicha pinza, uno paralelo y otro perpendicular a $\mathbf{b}_3$, lo cual se muestra en la siguiente figura.

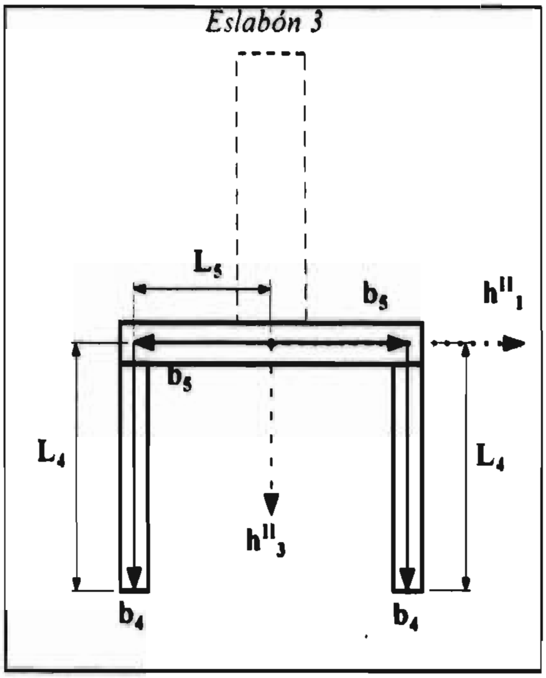


Figura 3.23. *Vectores que definen la pinza del manipulador (cuarto eslabón).*

De lo anterior tenemos que los vectores que definen la pinza son los siguientes.

$\mathbf{b}_4 = \mathbf{L}_4{}^* \mathbf{h}_3^{II} = \mathbf{L}_4{}^* \rho((\mathbf{P}^*\mathbf{Q}), \mathbf{h}_3) = ((\mathbf{P}^*\mathbf{Q}) * \mathbf{e}_3 * (\overline{\mathbf{P} * \mathbf{Q}}))$;

$\mathbf{b}_5 = \mathbf{L}_5{}^* \mathbf{h}_1^{II} = \mathbf{L}_5{}^* \rho((\mathbf{P}^*\mathbf{Q}), \mathbf{h}_1) = ((\mathbf{P}^*\mathbf{Q}) * \mathbf{e}_1 * (\overline{\mathbf{P} * \mathbf{Q}}))$; (3.70)

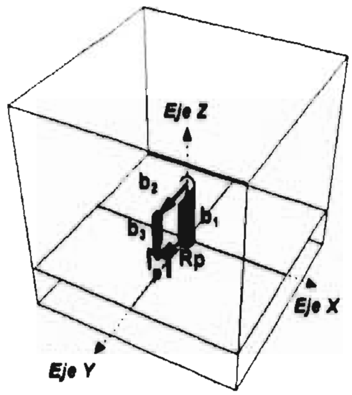


Figura 3.24. *Vector de posición del elemento terminal en la configuración no deformada.*

3.3.3. Ecuaciones explícitas optimizadas para la configuración no deformada.

A continuación se muestra la forma de las ecuaciones optimizadas, correspondientes a la configuración no deformada.

$$\begin{aligned} b_1 &= \{0, 0, L_1\}; \\ b_2 &= \{0, -2^*A1^*L_2, -A2^*L_2\}; \\ b_3 &= \{0, 2^*(A2^*A3 + A1^*A4)^*L_3, (A2^*A4 - 4^*A1^*A3)^*L_3\}; \end{aligned} \tag{3.71}$$

$$R_p = \{0,\ 2^*((A2^*A3 + A1^*A4)^*L_3 - A1^*L_2),\ L_1 - A2^*L_2 + (A2^*A4 - 4^*A1^*A3)^*L_3\}; \tag{3.72}$$

$$\begin{aligned} b_4 &= \{0, 2^*(A2^*A3 + A1^*A4)^*L_4, (A2^*A4 - 4^*A1^*A3)^*L_4\}; \\ b_5 &= \{L_5, 0, 0\}; \end{aligned} \tag{3.73}$$

Donde:

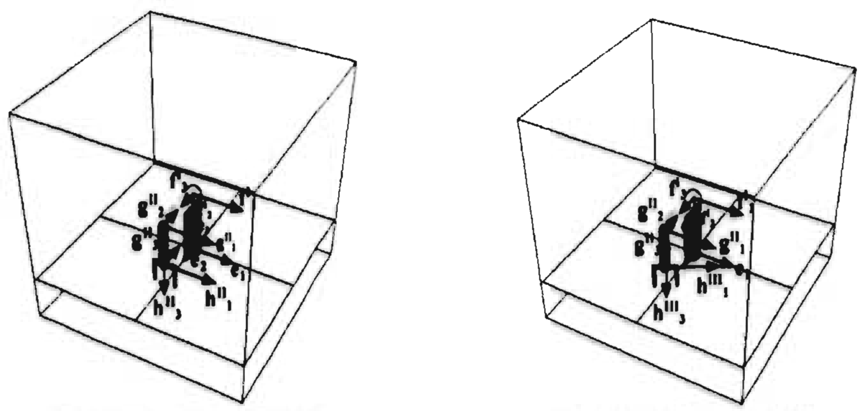
$$\begin{aligned} A1 &= P_0^*P_1; \quad & A2 &= 1 - 2^*P_0^2; \\ A3 &= Q_0^*Q_1; \quad & A4 &= 1 - 2^*Q_0^2; \end{aligned} \tag{3.74}$$

3.3.4. Configuración deformada.

Ahora se modelará la configuración deformada, iniciando las rotaciones en el elemento terminal y finalizando en el primer elemento, ya que es la secuencia de modelación más sencilla.

Secuencia de Rotaciones:

Rotación 1.- Rota el eslabón 4 con el Quaternión **r4**, partiendo de la configuración no deformada. (Figura 3.25).



Configuración no deformada **Rotación del cuarto eslabón (pinza)**

Figura 3.25. *Rotación del cuarto eslabón en la configuración deformada.*

Rotación 2.- Rotan los eslabones 3 y 4 con el Quaternión **r3**. (Figura 3.26).

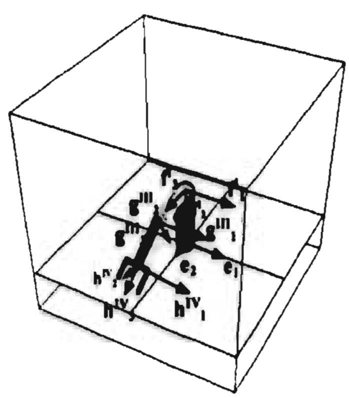


Figura 3.26. *Rotación de los eslabones 3 y 4 en la configuración deformada.*

Rotación 3.- Rotan los eslabones 2, 3 y 4 con el Quaternión **r2**. (Figura 3.28).

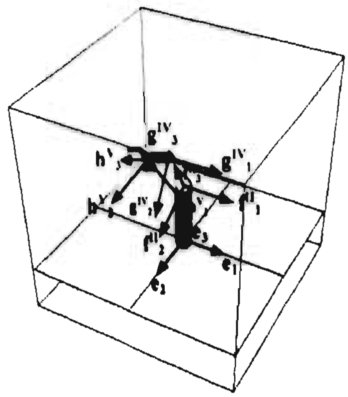


Figura 3.27. *Rotación de los eslabones 2, 3 y 4 en la configuración deformada.*

Rotación 3.- Rotan los cuatro eslabones con el Quaternión **r1**. (Figura 3.28).

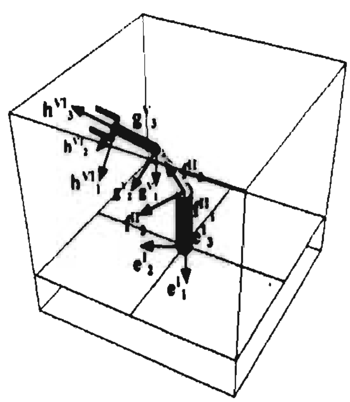


Figura 3.28. *Rotación de los cuatro eslabones en la configuración deformada.*

De la Figura 3.25 podemos ver que los ejes de rotación de los eslabones se conservan como en la configuración original (Figura 3.20), excepto para el cuarto eslabón, cuyo eje de rotación es $\mathbf{h}_3^{II}$. Por lo anterior podemos definir los Quaterniones para rotar a posición deformada como se muestra a continuación.

Eslabón 1: $r1 = \{r1_0, r1_1, r1_2, r1_3\} \quad \Rightarrow \quad r1_v = \{r1_1, r1_2, r1_3\}, \quad \|r1\| = 1.$

La forma local del Quaternión r1 es:

$$r1_v = \mathrm{Sin}(\theta_1/2)^* e_3 = \mathrm{Sin}(\theta_1/2)^*\{0, 0, 1\}$$
$$r1_v = \{0, 0, r1_3\}$$

De lo anterior: $\mathbf{r1 = \{ r1_0, 0, 0, r1_3 \}}$, Donde:

$r1_0 = Cos(\theta_1/2);$
$r1_3 = Sen(\theta_1/2);$

Siendo θ_1 el ángulo de rotación del eslabón 1 en la configuración deformada.

Eslabón 2: $r2 = \{r2_0, r2_1, r2_2, r2_3\} \Rightarrow r2_v = \{r2_1, r2_2, r2_3\}, \quad \|r2\| = 1.$

La forma local del Quaternión r2 es:

$r2_v = Sin(\theta_2/2)^* e_1 = Sin(\theta_2/2)^*\{1, 0, 0\}$
$r2_v = \{ r2_1, 0, 0 \}$

De lo anterior: $\mathbf{r2 = \{ r2_0, r2_1, 0, 0\}}$, Donde:

$r2_0 = Cos(\theta_2/2);$
$r2_1 = Sen(\theta_2/2);$

Siendo θ_2 el ángulo de rotación del eslabón 2 en la configuración deformada.

Eslabón 3: $r3 = \{r3_0, r3_1, r3_2, r3_3\} \Rightarrow r3_v = \{r3_1, r3_2, r3_3\}, \quad \|r3\| = 1.$

La forma local del Quaternión r3 es:

$r3_v = Sin(\theta_3/2)^* e_1 = Sin(\theta_3/2)^*\{1, 0, 0\}$
$r3_v = \{ r3_1, 0, 0 \}$

De lo anterior: $\mathbf{r3 = \{ r3_0, r3_1, 0, 0 \}}$, Donde:

$r3_0 = Cos(\theta_3/2);$
$r3_1 = Sen(\theta_3/2);$

Siendo θ_3 el ángulo de rotación del eslabón 3 en la configuración deformada.

Eslabón 4: $r4 = \{r4_0, r4_1, r4_2, r4_3\} \Rightarrow r4_v = \{r4_1, r4_2, r4_3\}, \quad \|r4\| = 1.$

La forma local del Quaternión r4 en la configuración no deformada es:

$r4_v = Sin(\theta_4/2)^* e_3 = Sin(\theta_4/2)^*\{0, 0, 1\}$
$r4_v = \{0, 0, r4_3 \}$

De lo anterior: $\mathbf{r4 = \{ r4_0, 0, 0, r4_3 \}}$, Donde:

$r4_0 = Cos(\theta_4/2);$
$r4_3 = Sen(\theta_4/2);$

Siendo θ_4 el ángulo de rotación del eslabón 4 en la configuración deformada.

Pero como el eje de rotación del eslabón 4 se ve afectado por las rotaciones aplicadas en la configuración no deformada, siendo ahora $\mathbf{h}_3^{II}$, el Quaternión r4, será:

$r4^{I}_{v} = \mathrm{Sin}(\theta_4/2)^{*}\ \mathbf{h}^{II}_{3} = \mathrm{Sin}(\theta_4/2)^{*}\rho((P^{*}Q), \mathbf{e}_3) = \rho((P^{*}Q), r4_v)$

$\mathbf{r4}^{I} = \rho((\mathbf{P^{*}Q}), \mathbf{r4})$

Aplicación de la Secuencia de rotaciones para determinar la configuración deformada.

Rotación 1.- Rotación del cuarto eslabón con el Quaternión r4. Esta rotación solamente afecta al eslabón 4. (Figura 3.25).

$\mathbf{h}^{III}_{j} = \rho(\mathbf{r4}^{I}, \mathbf{h}^{II}_{j})$ (Rotación del cuarto eslabón con Quaternión r4)

Donde: $j = 1,2,3$ (3.75)

Rotación 2.- Rotación de los eslabones 3 y 4 con el Quaternión r3. Esta rotación afecta las bases g_j y h_j. (Figura 3.26).

$\mathbf{g}^{III}_{j} = \rho(\mathbf{r3}, \mathbf{g}^{II}_{j})$ (Rotación del tercer eslabón con Quaternión r3)

$\mathbf{h}^{IV}_{j} = \rho(\mathbf{r3}, \mathbf{h}^{III}_{j})$ (Rotación del cuarto eslabón con Quaternión r3) (3.76)

Donde: $j = 1,2,3$

Rotación 3.- Rotación de los eslabones 2, 3 y 4 con el Quaternión r2. Esta rotación afecta las bases f_j, g_j y h_j. (Figura 3.27).

$\mathbf{f}^{II}_{j} = \rho(\mathbf{r2}, \mathbf{f}^{I}_{j})$ (Rotación del segundo eslabón con Quaternión r2)

$\mathbf{g}^{IV}_{j} = \rho(\mathbf{r2}, \mathbf{g}^{III}_{j})$ (Rotación del tercer eslabón con Quaternión r2) (3.77)

$\mathbf{h}^{V}_{j} = \rho(\mathbf{r2}, \mathbf{h}^{IV}_{j})$ (Rotación del cuarto eslabón con Quaternión r2)

Donde: $j = 1,2,3$

Rotación 4.- Rotación de los cuatro eslabones con el Quaternión r1. Esta rotación, que es la última, afecta a todas las bases.. (Figura 3.28).

$\mathbf{e}^{I}_{j} = \rho(\mathbf{r1}, \mathbf{e}_{j})$ (Rotación del primer eslabón con Quaternión r1)

$\mathbf{f}^{III}_{j} = \rho(\mathbf{r1}, \mathbf{f}^{II}_{j})$ (Rotación del segundo eslabón con Quaternión r1)

$\mathbf{g}^{V}_{j} = \rho(\mathbf{r1}, \mathbf{g}^{IV}_{j})$ (Rotación del tercer eslabón con Quaternión r1) (3.78)

$\mathbf{h}^{VI}_{j} = \rho(\mathbf{r1}, \mathbf{h}^{V}_{j})$ (Rotación del cuarto eslabón con Quaternión r1)

Donde: $j = 1,2,3$

La forma final resultante de las bases locales de cada eslabón es la siguiente:

Eslabón 1.-

$$e_j^{I} = \rho(r1,\ e_j\ ) \qquad (3.79)$$

Eslabón 2.-

$$f_j^{III} = \rho(r1,\ f_j^{II}\ ) = \rho(r1, \rho(r2,\ f_j^{I}\ )) = \rho(r1, \rho(r2, \rho(P, f_j))) = \rho(r1{*}r2{*}P, f_j)$$

$$f_j^{III} = \rho(r1, \rho(r2, \rho(P, e_j))) = \rho(r1{*}r2{*}P, e_j) \qquad (3.80)$$

Eslabón 3.-

$$g_j^{V} = \rho(\ r1,\ g_j^{IV}\ ) = \rho(r1, \rho(\ r2,\ g_j^{III}\ )) = \rho(r1, \rho(\ r2, \rho(r3,\ g_j^{II}\ )))$$

$$g_j^{V} = \rho(r1, \rho(r2, \rho(r3, \rho(P, \rho(Q, e_j))))) = \rho(r1{*}r2{*}r3{*}P{*}Q, e_j) \qquad (3.81)$$

Eslabón 4.-

$$h_j^{VI} = \rho(\ r1,\ h_j^{V}\ ) = \rho(r1, \rho(\ r2,\ h_j^{IV}\ )) = \rho(r1, \rho(r2, \rho(r3,\ h_j^{III}\ )))$$
$$h_j^{VI} = \rho(r1, \rho(r2, \rho(r3, \rho(r4^{I},\ h_j^{II}\ )))) = \rho(r1, \rho(r2, \rho(r3, \rho(\rho((P{*}Q), r4),\ h_j^{II}\ ))))$$
$$h_j^{VI} = \rho(r1, \rho(r2, \rho(r3, \rho(\rho((P{*}Q), r4), \rho((P{*}Q), h_j)))))$$
$$h_j^{VI} = \rho(r1, \rho(r2, \rho(r3, \rho(P{*}Q{*}r4{*}\overline{P{*}Q}\ , P{*}Q{*}h_j{*}\overline{P{*}Q}\ ))))$$
$$h_j^{VI} = \rho(r1, \rho(r2, \rho(r3, P{*}Q{*}r4{*}\overline{P{*}Q}{*}\ P{*}Q{*}h_j{*}\overline{P{*}Q}{*}P{*}Q{*}\overline{r4}{*}\overline{P{*}Q}\ )))$$
$$h_j^{VI} = \rho(r1, \rho(r2, \rho(r3, P{*}Q{*}r4{*}h_j{*}\overline{P{*}Q{*}r4}\ )))$$

$$h_j^{VI} = \rho(r1, \rho(r2, \rho(r3, \rho(P, \rho(Q, \rho(r4, h_j)))))) = \rho(r1{*}r2{*}r3{*}P{*}Q{*}r4, e_j) \qquad (3.82)$$

3.3.5. Vectores que definen los eslabones y posición del elemento terminal en la configuración deformada.

De la Figura 3.29 y Figura 3.29 se determinan los vectores que definen los eslabones del manipulador y el vector del posición del elemento terminal (punto p), que en este caso, se considera ser un punto central del extremo de la pinza. Siendo L_1, L_2 , L_4 y L_5 las longitudes de los eslabones.

$$\begin{aligned}
b_1 &= L_1{*}e_3^{I}; && (\text{ Eslabón 1 })\\
b_2 &= L_2{*}f_3^{III}; && (\text{ Eslabón 2 }) \qquad (3.83)\\
b_3 &= L_3{*}g_3^{V}; && (\text{ Eslabón 3 })\\
b_4 &= L_4{*}h_3^{VI}\ ; && (\text{ Eslabón 4, pinza })\\
b_5 &= L_5{*}h_1^{VI}\ ; && (\text{ Eslabón 4, pinza })
\end{aligned}$$

$$r_p = b_1 + b_2 + b_3 + b_4 \quad (\text{ Posición del elemento terminal }) \qquad (3.84)$$

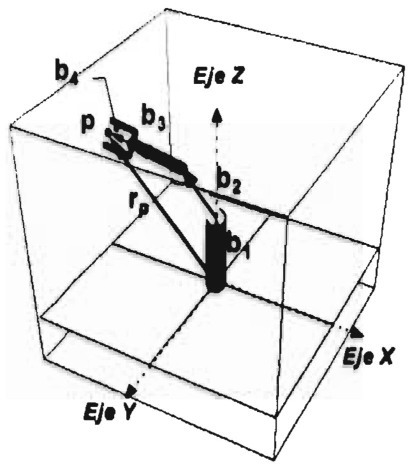


Figura 3.29. *Vector de posición del elemento terminal en la configuración no deformada.*

3.3.6. Ecuaciones explícitas optimizadas para la configuración deformada.

Las expresiones anteriores se explicitaron y optimizaron utilizando Mathematica®, y se presentan a continuación.

$$\begin{aligned}
b_1 &= \{0,\ 0,\ L_1\} \\
b_2 &= \{-4*A5*B2*L_2,\ -2*A6*B2*L_2,\ B1*L_2\} \\
b_3 &= \{-4*A5*C1*L_3,\ -2*A6*C1*L_3,\ C2*L_3\} \\
b_4 &= \{-4*A5*C1*L_4,\ -2*A6*C1*L_4,\ C2*L_4\} \\
b_5 &= \{(A12*A6 - 4*A11*A5*C2)*L_5,\ -2*(A12*A5 + A11*A6*C2)*L_5,\ -4*A11*C1*L_5\};
\end{aligned} \tag{3.85}$$

$$\begin{aligned}
r_p = \{&-4*A5*(B2*L_2 + C1*(L_3+L_4)), \\
&-2*A6*(B2*L_2 + C1*(L_3+L_4)),\ L_1 + B1*L_2 + C2*(L_3+L_4)\}
\end{aligned} \tag{3.86}$$

Donde:

$$\begin{aligned}
A1 &= P_0*P_1; & A2 &= 1-2*P_0^2; \\
A3 &= Q_0*Q_1; & A4 &= 1-2*Q_0^2; \\
A5 &= r1_0*r1_3; & A6 &= 1-2*r1_0^2; \\
A7 &= r2_0*r2_1; & A8 &= 1-2*r2_0^2; \\
A9 &= r3_0*r3_1; & A10 &= 1-2*r3_0^2; \\
A11 &= r4_0*r4_3; & A12 &= 1-2*r4_0^2;
\end{aligned} \tag{3.87}$$

$$\begin{aligned}
B1 &= A2*A8 - 4*A1*A7; & B2 &= A2*A7 + A1*A8; \\
B3 &= A10*A8 - 4*A7*A9; & B4 &= A10*A7 + A8*A9; \\
B5 &= A2*A4 - 4*A1*A3; & B6 &= A2*A3 + A1*A4;
\end{aligned}$$

$$C1 = B4*B5 + B3*B6; \qquad C2 = B3*B5 - 4*B4*B6;$$

3.4. MODELACION CINEMATICA DE UN MANIPULADOR DE CUATRO GRADOS DE LIBERTAD TIPO SCARA, UTILIZANDO ROTACIONES.

En esta sección mostraremos la modelación de un robot tipo SCARA de cuatro grados de libertad, aplicando la secuencia más sencilla determinada en la modelación del manipulador de tres grados de libertad, tanto para la configuración deformada como para la no deformada..

3.4.1. Configuración no deformada.

Mostraremos primeramente la modelación de la configuración no deformada, que consiste en determinar un sistema de ecuaciones que nos permitan determinar la posición del primero, segundo y tercer eslabón, conociendo la posición del elemento terminal del manipulador. Para determinar las ecuaciones necesarias realizamos las siguientes rotaciones y una traslación.

1.- Se mantiene fija la pinza.
Traslación 2.- Se aplica una traslación al tercer eslabón. (Figura 3.31).
Rotación 3.- Se aplica una rotación al segundo eslabón. (Figura 3.32).
Rotación 4.- Se aplica una rotación al primer eslabón. (Figura 3.33).

Para lo anterior definiremos la configuración no deformada del manipulador, que consiste en definir el vector de posición del elemento terminal para posiciones arbitrarias del primero y segundo eslabón. Consideremos la base canónica de $\Re^3$, esto es:

$$\mathbf{e}_1 = \{1,0,0\}; \quad \mathbf{e}_2 = \{0,1,0\}; \quad \mathbf{e}_3 = \{0,0,1\};$$

Ubicamos sistemas de referencia locales en la base de cada eslabón del manipulador como se muestra en la Figura 3.30. Observemos que, en este caso, los sistemas de referencia definidos coinciden con la base canónica la cual sirve como sistema de referencia inercial fijo, es decir:

$$\mathbf{h}_j = \mathbf{g}_j = \mathbf{f}_j = \mathbf{e}_j \;, \quad j = 1,2,3.$$

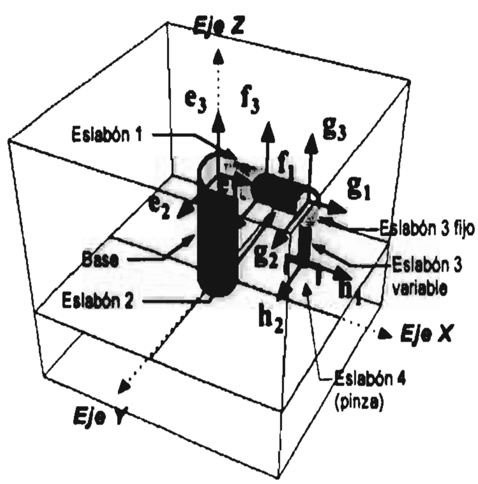


Figura 3.30. *Bases locales en la configuración original para la modelación.*

De la posición mostrada en la Figura 3.30, debemos permitir que los eslabones 1 y 2 giren a una posición arbitraria, considerando que los eslabones tienen los siguientes ejes de rotación o traslación:

Eslabón	Eje De Rotación
1	e_3
2	$f_3 = e_3$
3	$g_3 = e_3$ (Traslación)
4	$h_3 = e_3$

Con los ejes de rotación anteriores, podemos definir la forma que tendrán inicialmente los Quaterniones que nos permitirán rotar los eslabones 1 y 2, y la variable a utilizar para la traslación del tercer eslabón..

Eslabón 1: Consideremos el siguiente parámetro que caracterizará a nuestra primera rotación;

$$P = \{P_0, P_1, P_2, P_3\} \Rightarrow \quad P_v = \{P_1, P_2, P_3\}, \quad \|P\| = 1.$$

En este caso se tiene que siendo $\underline{w} \in \Re^3$ el eje de rotación, entonces, [1]:

$$P_0 = Cos(\varphi_1/2)$$
$$P_v = Sin(\varphi_1/2)^* \underline{w} = Sin(\varphi_1/2)^* e_3 = Sin(\varphi_1/2)^*\{0,0,1\}$$
$$P_v = \{0, 0, P_3\}$$

De lo anterior se obtiene la forma local del parámetro **P**, que es: $\mathbf{P = \{P_0, 0, 0, P_3\}}$, Donde:

$$P_0 = Cos(\varphi_1/2);$$
$$P_3 = Sen(\varphi_1/2);$$

Siendo φ_1 el ángulo de rotación del eslabón 1.

Eslabón 2: Consideremos ahora el siguiente parámetro que caracterizará la segunda rotación;

$$Q = \{Q_0, Q_1, Q_2, Q_3\} \quad \Rightarrow \quad Q_v = \{Q_1, Q_2, Q_3\}, \quad \|Q\| = 1.$$

Siendo $\underline{w} \in \Re^3$ el eje de rotación, tenemos que:

$$Q_v = Sin(\varphi_2/2)^* \underline{w} = Sin(\varphi_2 2)^* f_3 = Sin(\varphi_2/2)^* e_3 = Sin(\varphi_2/2)^*\{0,0,1\}$$
$$Q_v = \{0, 0, Q_3\}$$

De lo anterior se obtiene la forma local del parámetro **Q**, que es: $\mathbf{Q = \{Q_0, 0, 0, Q_3\}}$, Donde:

$$Q_0 = Cos(\varphi_2/2);$$
$$Q_3 = Sen(\varphi_2/2);$$

Siendo φ_2 el ángulo de rotación del eslabón 2.

Eslabón 3: Como este eslabón se traslada exclusivamente sobre el eje e_3, su desplazamiento, en la configuración no deformada, lo definiremos simplemente como la magnitud de la variable l_{3v}.

Con los Quaterniones anteriores podemos ahora realizar las rotaciones necesarias para determinar la configuración no deformada, se debe observar que si los ejes de rotación cambian su forma, también cambiará la forma de los Quaterniones dados.

Aplicación de la Secuencia de rotaciones y traslaciones para determinar la configuración no deformada.

Traslación 2.- La aplicación de la traslación no afecta la forma de las bases locales de los eslabones, como se puede observar en Figura 3.31.

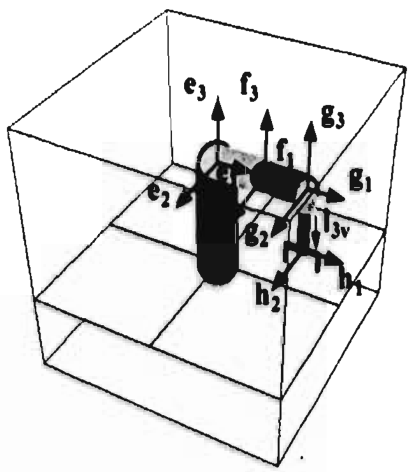


Figura 3.31. *Traslación del eslabón 3* (l_{3v}) *en la configuración no deformada.*

Rotación 3.- Rotamos el segundo eslabón con el Quaternión **Q**. Esto se realiza rotando las bases locales $\mathbf{f}_j$, $\mathbf{g}_j$ y $\mathbf{h}_j$. (Figura 3.32)

$$\mathbf{f}_j^1 = \rho(\mathbf{Q}, \mathbf{f}_j) \quad \text{(Rotación del segundo eslabón con Quaternión Q),}$$
$$\mathbf{g}_j^1 = \rho(\mathbf{Q}, \mathbf{g}_j) \quad \text{(Rotación del tercer eslabón con Quaternión Q),} \qquad (3.88)$$
$$\mathbf{h}_j^1 = \rho(\mathbf{Q}, \mathbf{h}_j) \quad \text{(Rotación del cuarto eslabón con Quaternión Q),}$$
$$j = 1,2,3$$

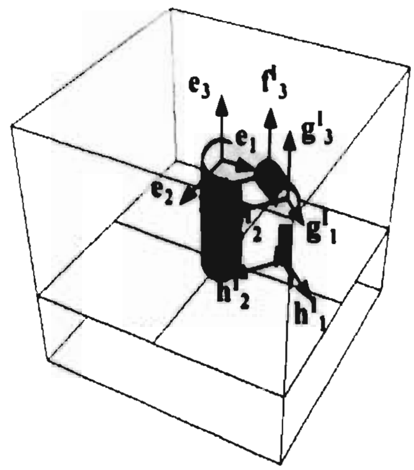


Figura 3.32. *Rotación del segundo eslabón en la configuración no deformada.*

Rotación 4.- Se aplica la rotación al primer eslabón con el Quaternión P, lo que implica que las bases $\mathbf{e}_j$, $\mathbf{f}_j$, $\mathbf{g}_j$ y $\mathbf{h}_j$. sufren las siguientes transformaciones.(Figura 3.33)

$$
\begin{array}{ll}
\mathbf{e}_j^{I} = \rho(\mathbf{P}, \mathbf{e}_j) & \text{(Rotación del primer eslabón con Quaternión P).} \\
\mathbf{f}_j^{II} = \rho(\mathbf{P}, \mathbf{f}_j^{I}) & \text{(Rotación del segundo eslabón con Quaternión P).} \\
\mathbf{g}_j^{II} = \rho(\mathbf{P}, \mathbf{g}_j^{I}) & \text{(Rotación del tercer eslabón con Quaternión P).} \\
\mathbf{h}_j^{II} = \rho(\mathbf{Q}, \mathbf{h}_j^{I}) & \text{(Rotación del cuarto eslabón con Quaternión P).} \\
j = 1,2,3 &
\end{array}
\qquad (3.89)
$$

Como $\mathbf{f}_j^{I} = \rho(\mathbf{Q}, \mathbf{f}_j)$, $\mathbf{g}_j^{I} = \rho(\mathbf{Q}, \mathbf{g}_j)$ y $\mathbf{h}_j^{I} = \rho(\mathbf{Q}, \mathbf{h}_j)$, tenemos que:

$$
\begin{aligned}
\mathbf{f}_j^{II} &= \rho(\mathbf{P}, \rho(\mathbf{Q}, \mathbf{f}_j)) = \rho((\mathbf{P}^*\mathbf{Q}), \mathbf{f}_j) \\
\mathbf{g}_j^{II} &= \rho(\mathbf{P}, \rho(\mathbf{Q}, \mathbf{g}_j)) = \rho((\mathbf{P}^*\mathbf{Q}), \mathbf{g}_j) \\
\mathbf{h}_j^{II} &= \rho(\mathbf{P}, \rho(\mathbf{Q}, \mathbf{h}_j)) = \rho((\mathbf{P}^*\mathbf{Q}), \mathbf{h}_j)
\end{aligned}
\qquad (3.90)
$$

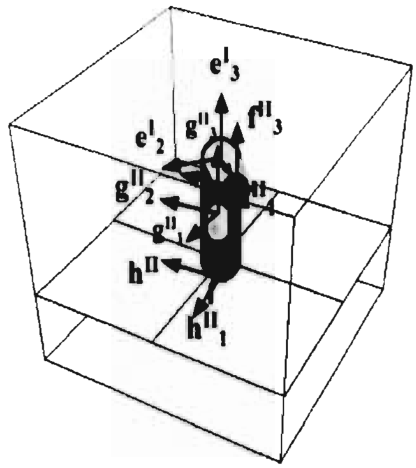


Figura 3.33. *Rotación de los eslabones 2 y 3 en la configuración no deformada.*

La forma explícita de las bases locales, después de aplicadas las rotaciones anteriores, es la siguiente:

$$
\begin{aligned}
e_1^I &= \{-1 + 2{*}P_0^2,\ 2{*}P_0{*}P_3,\ 0\} \\
e_2^I &= \{-2{*}P_0{*}P_3,\ -1 + 2{*}P_0^2,\ 0\} \\
e_3^I &= \{0,\ 0,\ 1\}
\end{aligned}
\qquad (3.91)
$$

$$
\begin{aligned}
f_1^{II} &= \{1 - 2{*}P_0^2 - 2{*}Q_0^2 + 4{*}P_0^2{*}Q_0^2 - 4{*}P_0{*}P_3{*}Q_0{*}Q_3, \\
&\qquad -2{*}P_0{*}P_3 + 4{*}P_0{*}P_3{*}Q_0^2 - 2{*}Q_0{*}Q_3 + 4{*}P_0^2{*}Q_0{*}Q_3,\ 0\} \\
f_2^{II} &= \{2{*}(P_0{*}P_3 - 2{*}P_0{*}P_3{*}Q_0^2 + Q_0{*}Q_3 - 2{*}P_0^2{*}Q_0{*}Q_3), \\
&\qquad 1 - 2{*}P_0^2 - 2{*}Q_0^2 + 4{*}P_0^2{*}Q_0^2 - 4{*}P_0{*}P_3{*}Q_0{*}Q_3,\ 0\} \\
f_3^{II} &= \{0,\ 0,\ 1\}
\end{aligned}
\qquad (3.92)
$$

$$
\begin{aligned}
h_1^{II} &= g_1^{II} = f_1^{II} \\
h_2^{II} &= g_2^{II} = f_2^{II} \\
h_3^{II} &= g_3^{II} = f_3^{II}
\end{aligned}
\qquad (3.93)
$$

3.4.2. Vectores que definen los eslabones y posición del elemento terminal en la configuración no deformada.

De la Figura 3.33 y Figura 3.34 podemos determinar los vectores que definen los eslabones del manipulador y el vector de posición del elemento terminal (punto p). Siendo $\mathbf{L_1}$, $\mathbf{L_2}$ y $\mathbf{L_3}$ las longitudes de los eslabones.

$$\mathbf{b_1 = L_1 * e_1^{I} = L_2 * \rho(P, e_1)} \quad ; \qquad \textbf{(Eslabón 1)}$$

$$\mathbf{b_2 = L_2 * f_1^{II} = L_2 * \rho(P, f_1^{I}) = L_2 * \rho(P, \rho(Q, f_1))}; \qquad \textbf{(Eslabón 2)}$$

$$\mathbf{b_3 = L_3 * (-g_3^{II}) = - L_3 * e_3}; \qquad \textbf{(Eslabón 3 fijo)} \qquad (3.94)$$

$$\mathbf{b_{3v} = l_{3v} * (-g_3^{II}) = - l_{3v} * e_3}; \qquad \textbf{(Eslabón 3 variable)}$$

$$\mathbf{b_4 = L_4 * (-h_3^{II})}; \qquad \textbf{(Pinza)}$$

$$\mathbf{b_5 = L_5 * h_1^{II}}; \qquad \textbf{(Pinza)}$$

$$\mathbf{R_p = b_1 + b_2 + b_3 + b_{3v}} \qquad \textbf{(Posición del elemento terminal)} \qquad (3.95)$$

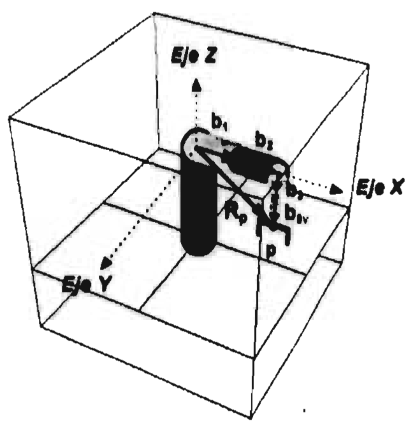


Figura 3.34. *Vector de posición del elemento terminal en la configuración no deformada.*

3.4.3. Ecuaciones explícitas optimizadas para la configuración no deformada.

A continuación se muestra la forma de las ecuaciones optimizadas, correspondientes a la configuración no deformada.

$$\mathbf{b_1 = \{- A2*L_1, 2*A1*L_1, 0\}};$$
$$\mathbf{b_2 = \{L_2*(A2*A4 - 4*A1*A3), - 2*L_2*(A2*A3 + A1*A4), 0\}}; \qquad (3.96)$$
$$\mathbf{b_3 = \{0, 0, - L_3\}};$$
$$\mathbf{b_{3v} = \{0, 0, - l_{3v}\}};$$

$$\mathbf{R_p = \{- A2*L_1 + L_2*(A2*A4 - 4*A1*A3), 2*(A1*L_1 - L_2*(A2*A3 + A1*A4)), - (L_3 + l_{3v})\}}; \qquad (3.97)$$

Donde:

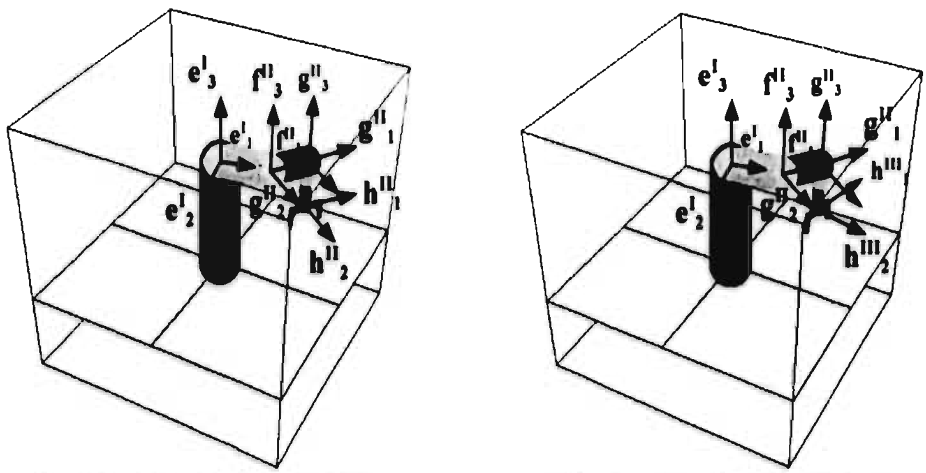
$$\mathbf{A1 = P_0*P_3; \quad A2 = 1 - 2*P_0\text{^}2};$$
$$\mathbf{A3 = Q_0*Q_3; \quad A4 = 1 - 2*Q_0\text{^}2}; \qquad (3.98)$$

3.4.4. Configuración deformada.

Ahora se modelará la configuración deformada, iniciando las rotaciones en el elemento terminal y finalizando en el primer elemento, ya que es la secuencia de modelación más sencilla.

Secuencia de Rotaciones y Traslaciones:

Rotación 1.- Rota el eslabón 4 con el Quaternión **r4**, partiendo de una configuración no deformada arbitraria (Figura 3.35).



Configuración no deformada **Rotación del cuarto eslabón (pinza)**

Figura 3.35. *Rotación del cuarto eslabón en la configuración deformada.*

Traslación 1.- Se trasladan los eslabones 3 y 4 en la dirección e_3 una distancia $\mathbf{L}_{3v}$. (Figura 3.36).

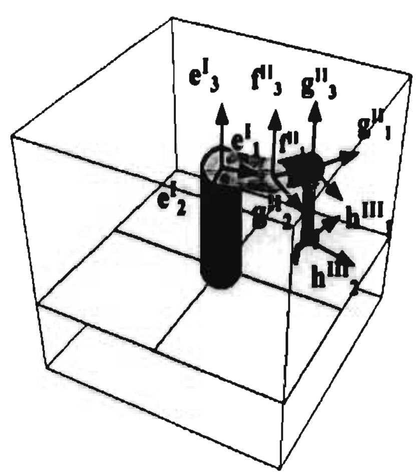


Figura 3.36. *Traslación de los eslabones 3 y 4 en la configuración deformada.*

Rotación 2.- Rotan los eslabones 2, 3 y 4 con el Quaternión **r2**. (Figura 3.37).

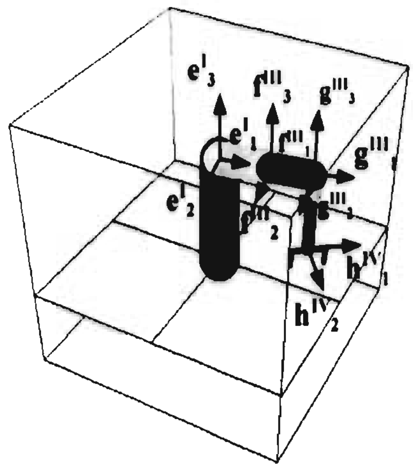


Figura 3.37. *Rotación de los eslabones 2, 3 y 4 en la configuración deformada.*

Rotación 3.- Rotan los cuatro eslabones con el Quaternión **r1**. (Figura 3.38).

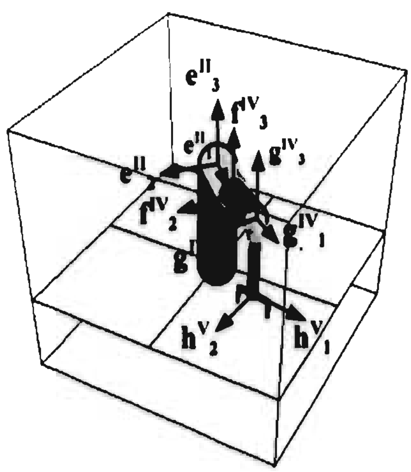


Figura 3.38. *Rotación de los cuatro eslabones en la configuración deformada.*

De la Figura 3.35 podemos ver que los ejes de rotación de los eslabones se conservan como en la configuración original (Figura 3.30). Por lo anterior podemos definir los Quaterniones para rotar a posición deformada como se muestra a continuación.

Eslabón 1: $r1 = \{r1_0, r1_1, r1_2, r1_3\} \quad \Rightarrow \quad r1_v = \{r1_1, r1_2, r1_3\}, \quad \|r1\| = 1.$

La forma local del Quaternión r1 es:

$r1_v = Sin(\theta_1/2)^* e_3 = Sin(\theta_1/2)^*\{0, 0, 1\}$
$r1_v = \{0, 0, r1_3\}$

De lo anterior: $\mathbf{r1 = \{r1_0, 0, 0, r1_3\}}$, Donde:

$r1_0 = Cos(\theta_1/2);$
$r1_3 = Sen(\theta_1/2);$

Siendo θ_1 el ángulo de rotación del eslabón 1 en la configuración deformada.

Eslabón 2: $r2 = \{r2_0, r2_1, r2_2, r2_3\} \Rightarrow r2_v = \{r2_1, r2_2, r2_3\}, \quad \|r2\| = 1.$

La forma local del Quaternión r2 es:

$r2_v = Sin(\theta_2/2)^* e_3 = Sin(\theta_2/2)^*\{0, 0, 1\}$
$r2_v = \{0, 0, r2_3\}$

De lo anterior: $\mathbf{r2 = \{r2_0, 0, 0, r2_3\}}$, Donde:

$r2_0 = Cos(\theta_2/2);$
$r2_3 = Sen(\theta_2/2);$

Siendo θ_2 el ángulo de rotación del eslabón 2 en la configuración deformada.

Eslabón 3: Como el eslabón 3 tiene una junta prismática y no rotacional como los demás, no hay Quaternión asociado a este eslabón, pero en su lugar existe una variable de longitud que caracteriza su traslación, que es $\mathbf{L_{3v}}$.

Eslabón 4 (pinza): $r4 = \{r4_0, r4_1, r4_2, r4_3\} \Rightarrow r4_v = \{r4_1, r4_2, r4_3\}, \quad \|r4\| = 1.$

La forma local del Quaternión r4 es:

$r4_v = Sin(\theta_4/2)^* e_3 = Sin(\theta_4/2)^*\{0, 0, 1\}$
$r4_v = \{0, 0, r4_3\}$

De lo anterior: $\mathbf{r4 = \{r4_0, 0, 0, r4_3\}}$, Donde:

$r4_0 = Cos(\theta_4/2);$
$r4_3 = Sen(\theta_4/2);$

Siendo θ_4 el ángulo de rotación de la pinza (Eslabón 4) en la configuración deformada.

Aplicación de la Secuencia de rotaciones y traslaciones para determinar la configuración deformada.

Rotación 1.- Rotación del cuarto eslabón con el Quaternión r4. Esta rotación solamente afecta al eslabón 4. (Figura 3.35).

$\mathbf{h_j^{III}} = \rho(\mathbf{r4},\ \mathbf{h_j^{II}})$ (Rotación del cuarto eslabón con Quaternión r4)
Donde: $\mathbf{j = 1,2,3}$ (3.99)

Traslación 1.- Se trasladan, una distancia $\mathbf{L_{3\lambda}}$, los eslabone 3 y 4 sobre el eje e_3. Esta traslación no afecta la forma de ninguna base local. (Figura 3.36).

Rotación 2.- Rotación de los eslabones 2, 3 y 4 con el Quaternión r2. Esta rotación afecta las bases f_j, g_j y h_j. (Figura 3.37).

$\mathbf{f_j^{III}} = \rho(\mathbf{r2},\ \mathbf{f_j^{II}})$ (Rotación del segundo eslabón con Quaternión r2)
$\mathbf{g_j^{III}} = \rho(\mathbf{r2},\ \mathbf{g_j^{II}})$ (Rotación del tercer eslabón con Quaternión r2) (3.100)
$\mathbf{h_j^{IV}} = \rho(\mathbf{r2},\ \mathbf{h_j^{III}})$ (Rotación del cuarto eslabón con Quaternión r2)
Donde: $\mathbf{j = 1,2,3}$

Rotación 3.- Rotación de los cuatro eslabones con el Quaternión r1. Esta rotación, que es la última, afecta a todas las bases.. (Figura 3.38).

$\mathbf{e_j^{II}} = \rho(\mathbf{r1},\ \mathbf{e_j^{I}})$ (Rotación del primer eslabón con Quaternión r1)
$\mathbf{f_j^{IV}} = \rho(\mathbf{r1},\ \mathbf{f_j^{III}})$ (Rotación del segundo eslabón con Quaternión r1)
$\mathbf{g_j^{IV}} = \rho(\mathbf{r1},\ \mathbf{g_j^{III}})$ (Rotación del tercer eslabón con Quaternión r1) (3.101)
$\mathbf{h_j^{V}} = \rho(\mathbf{r1},\ \mathbf{h_j^{IV}})$ (Rotación del cuarto eslabón con Quaternión r1)
Donde: $\mathbf{j = 1,2,3}$

La forma final resultante de las bases locales de cada eslabón es la siguiente:

Eslabón 1.-

$$\mathbf{e_j^{II}} = \rho(\mathbf{r1}, \rho(\mathbf{P}, \mathbf{e_j})) = \rho(\mathbf{r1{*}P}, \mathbf{e_j}) \qquad (3.102)$$

Eslabón 2.-

$$\mathbf{f_j^{IV}} = \rho(\mathbf{r1},\ \mathbf{f_j^{III}}) = \rho(\mathbf{r1}, \rho(\mathbf{r2},\ \mathbf{f_j^{II}})) = \rho(\mathbf{r1}, \rho(\mathbf{r2}, \rho((\mathbf{P{*}Q}), \mathbf{f_j}))) = \rho(\mathbf{r1{*}r2{*}P{*}Q}, \mathbf{f_j})$$

$$\mathbf{f_j^{IV}} = \rho(\mathbf{r1}, \rho(\mathbf{r2}, \rho((\mathbf{P{*}Q}), \mathbf{e_j}))) = \rho(\mathbf{r1{*}r2{*}P{*}Q}, \mathbf{e_j}) \qquad (3.103)$$

Eslabón 3.-

$$\mathbf{g_j^{IV}} = \rho(\mathbf{r1},\ \mathbf{g_j^{III}}) = \rho(\mathbf{r1}, \rho(\mathbf{r2},\ \mathbf{g_j^{II}})) = \rho(\mathbf{r1}, \rho(\mathbf{r2}, \rho((\mathbf{P{*}Q}), \mathbf{g_j})))$$

$$\mathbf{g_j^{IV}} = \rho(\mathbf{r1}, \rho(\mathbf{r2}, \rho((\mathbf{P{*}Q}), \mathbf{e_j}))) = \rho(\mathbf{r1{*}r2{*}P{*}Q}, \mathbf{e_j}) \qquad (3.104)$$

Eslabón 4.-

$$h_j^{V} = \rho(r1,\ h_j^{IV}\ )) = \rho(r1,\ \rho(r2,\ h_j^{III}\ )) = \rho(r1,\ \rho(r2,\ \rho(r4,\ \rho((P{*}Q),\ h_j))))$$

$$h_j^{V} = \rho(r1,\ \rho(r2,\ \rho(r4,\ \rho((P{*}Q),\ e_j)))) = \rho(r1{*}r2{*}r4{*}P{*}Q,\ e_j) \qquad (3.105)$$

3.4.5. Vectores que definen los eslabones y posición del elemento terminal en la configuración deformada.

De la Figura 3.38 y Figura 3.39 se determinan los vectores que definen los eslabones del manipulador y el vector del posición del elemento terminal (punto p), que en este caso, se considera ser un punto intermedio de la pinza, definido por el vector b_4. Siendo L_1, L_2, L_{3v} y L_4 las longitudes de los eslabones y l_{3v} la posición del eslabón 3 en la posición no deformada.

$$
\begin{array}{ll}
b_1 = L_1{*}\,e_1^{I}\,; & (\text{Eslabón 1}) \\
b_2 = L_2{*}\,f_1^{III}\,; & (\text{Eslabón 2}) \\
b_{3v} = L_3{*}(-g_3^{V}\ ); & (\text{Eslabón 3 fijo}) \\
b_{3v} = (L_{3v}+l_{3v}){*}(-g_3^{V}\ ); & (\text{Eslabón 3 variable}) \\
b_4 = L_4{*}(-h_3^{VI}\ ); & (\text{Pinza}) \\
b_5 = L_5{*}(-h_1^{VI}\ ); & (\text{Pinza})
\end{array}
\qquad (3.106)
$$

$$r_p = b_1 + b_2 + b_3 + b_{3v} + b_4 \quad (\text{Posición del elemento terminal}) \qquad (3.107)$$

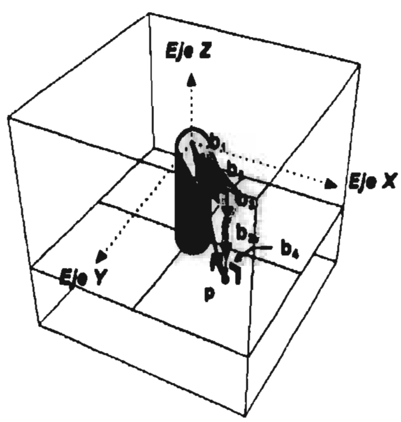


Figura 3.39. *Vector de posición del elemento terminal en la configuración deformada.*

3.4.6. Ecuaciones explícitas optimizadas para la configuración deformada.

Las expresiones anteriores se explicitaron y optimizaron utilizando Mathematica®, y se presentan a continuación.

$$
\begin{aligned}
b_1 &= \{B1*L_1,\ -2*B2*L_1,\ 0\}\\
b_2 &= \{C1*L_2,\ -2*C2*L_2,\ 0\}\\
b_3 &= \{0,\ 0,\ -L_3\}\\
b_{3v} &= \{0,\ 0,\ -(L_{3v}+l_{3v})\}\\
b_4 &= \{0,\ 0,\ -L_4\}\\
b_6 &= \{(4*A9*C2 - A10*C1)*L_6,\ 2*(A9*C1 + A10*C2)*L_6,\ 0\}
\end{aligned}
\qquad (3.108)
$$

$$
r_p = \{B1*L_1 + C1*L_2,\ -2*(B2*L_1 + C2*L_2),\ -(L_3 + L_{3v} + l_{3v} + L_4)\} \qquad (3.109)
$$

Donde:

$$
\begin{aligned}
&A1 = P_0*P_3; && A2 = 1-2*P_0^2;\\
&A3 = Q_0*Q_3; && A4 = 1-2*Q_0^2;\\
&A5 = r1_0*r1_3; && A6 = 1-2*r1_0^2;\\
&A7 = r2_0*r2_3; && A8 = 1-2*r2_0^2;\\
&A9 = r4_0*r4_3; && A10 = 1-2*r4_0^2;
\end{aligned}
\qquad (3.110)
$$

$$
\begin{aligned}
&B1 = A2*A8 - 4*A1*A7; && B2 = A2*A7 + A1*A8;\\
&B3 = A10*A8 - 4*A7*A9; && B4 = A10*A7 + A8*A9;\\
&B5 = A2*A4 - 4*A1*A3; && B6 = A2*A3 + A1*A4;
\end{aligned}
$$

$$
C1 = B4*B5 + B3*B6; \qquad C2 = B3*B5 - 4*B4*B6;
$$

CAPITULO IV

4. SIMULACION COMPUTACIONAL DE LA CINEMATICA.

La cinemática de un robot consiste en describir la relación entre el movimiento de las juntas del manipulador y el movimiento resultante de los cuerpos rígidos que constituyen el manipulador robótico. En este capítulo presentaremos la simulación computacional de la cinemática directa y cinemática inversa de los manipuladores modelados en el capítulo anterior, dicha simulación se realizará utilizando el lenguaje de cálculo formal Mathematica®, que hemos venido utilizando en el desarrollo de este trabajo. La cinemática directa de un robot consiste en determinar la posición, y orientación en su caso, del elemento terminal (pinza o herramienta montada en el extremo del robot) cuando se dan las configuraciones relativas de cada par de eslabones adyacentes del robot. En este trabajo solo trataremos manipuladores de cadena abierta, en los cuales los eslabones forman una cadena simple y cada par de eslabones esta conectado por medio de una junta rotacional o prismática. El problema de la cinemática inversa consiste en que dada la posición, y orientación si es el caso, del elemento terminal, se deben determinar los ángulos que deberán desplazarse las diferentes juntas del manipulador para alcanzar la posición y orientación deseada; lo anterior se realiza planteando un sistema de ecuaciones donde se involucran los parámetros de rotación de cada eslabón, así como también, la posición y orientación del elemento terminal.

4.1 Cinemática de un robot de tres grados de libertad.

En la siguiente figura se muestra la configuración del robot que se simulará en está sección, el cual fue previamente modelado en el capítulo anterior.

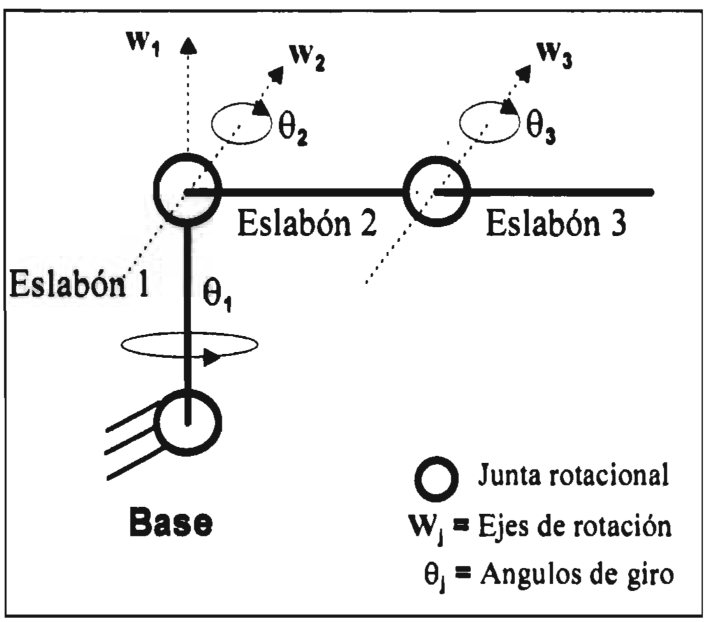


Figura 4.1. *Configuración del manipulador de tres grados de libertad*

4.1.1 Cinemática Directa.

El problema cinemático directo, en este caso, consiste en que dados los desplazamientos angulares θ_1, θ_2 y θ_3, se debe determinar la posición del elemento terminal, así como la configuración resultante de sus eslabones. Para realizar lo anterior hacemos uso de las ecuaciones (3.41), (3.42), (3.43) determinadas en el capitulo anterior, que al evaluarlas se resuelve la cinemática directa de este manipulador. Dichas ecuaciones se muestran a continuación.

$$\begin{aligned} b_1 &= \{0, 0, L_1\} \\ b_2 &= \{-4{*}A5{*}B2{*}L_2, -2{*}A6{*}B2{*}L_2, B1{*}L_2\} \\ b_3 &= \{-4{*}A5{*}C1{*}L_3, -2{*}A6{*}C1{*}L_3, C2{*}L_3\} \end{aligned} \tag{4.1}$$

$$r_p = \{-4{*}A5{*}(B2{*}L_2 + C1{*}L_3), -2{*}A6{*}(B2{*}L_2 + C1{*}L_3), L_1 + B1{*}L_2 + C2{*}L_3\} \tag{4.2}$$

Siendo:

$$\begin{array}{ll} A1 = P_0{*}P_1; & A2 = 1-2{*}P_0^2; \\ A3 = Q_0{*}Q_1; & A4 = 1-2{*}Q_0^2; \\ A5 = r1_0{*}r1_1; & A6 = 1-2{*}r1_0^2; \\ A7 = r2_0{*}r2_1; & A8 = 1-2{*}r2_0^2; \\ A9 = r3_0{*}r3_1; & A10 = 1-2{*}r3_0^2; \\ B1 = A2{*}A8 - 4{*}A1{*}A7; & B2 = A2{*}A7 + A1{*}A8; \\ B3 = A10{*}A8 - 4{*}A7{*}A9; & B4 = A10{*}A7 + A8{*}A9; \\ B5 = A2{*}A4 - 4{*}A1{*}A3; & B6 = A2{*}A3 + A1{*}A4; \\ C1 = B4{*}B5 + B3{*}B6; & C2 = B3{*}B5 - 4{*}B4{*}B6; \end{array} \tag{4.3}$$

Las variables de las ecuaciones anteriores tienen el siguiente significado:

$L_1, L_2, L_3 \rightarrow$ Longitud de los eslabones del manipulador.
$b_1, b_2, b_3 \rightarrow$ Eslabones del manipulador.
$r_p \rightarrow$ Posición del elemento terminal.
$P_0, P_1, Q_0, Q_1 \rightarrow$ Parámetros de rotación en la configuración no deformada.
$r1_0, r1_1, r2_0, r2_1, r3_0, r3_1 \rightarrow$ Parámetros de rotación en la configuración deformada.

Los parámetros de rotación tienen la siguiente relación con las rotaciones de los eslabones del manipulador.

$$\begin{array}{ll} P_0 = \mathrm{Cos}\,(\varphi_1/2); & P_1 = \mathrm{Sen}\,(\varphi_1/2); \\ Q_0 = \mathrm{Cos}\,(\varphi_2/2); & Q_1 = \mathrm{Sen}\,(\varphi_2/2); \end{array} \tag{4.4}$$

$$\begin{array}{ll} r1_0 = \mathrm{Cos}\,(\theta_1/2); & r1_1 = \mathrm{Sen}\,(\theta_1/2); \\ r2_0 = \mathrm{Cos}\,(\theta_2/2); & r2_1 = \mathrm{Sen}\,(\theta_2/2); \\ r3_0 = \mathrm{Cos}\,(\theta_3/2); & r3_1 = \mathrm{Sen}\,(\theta_3/2); \end{array} \tag{4.5}$$

Siendo:

φ_1 el ángulo de rotación del eslabón 2 en la configuración no deformada.
φ_2 el ángulo de rotación del eslabón 3 en la configuración no deformada.

θ_1 el ángulo de rotación del eslabón 1 en la configuración deformada.
θ_2 el ángulo de rotación del eslabón 2 en la configuración deformada.
θ_3 el ángulo de rotación del eslabón 3 en la configuración deformada.

Antes de poder realizar el proceso de cinemática directa del manipulador, debemos determinar la configuración no deformada del robot mediante un proceso de cinemática inversa, que consiste en calcular los parámetros P={P_0,P_1,0,0} y Q={Q_0,Q_1,0,0} dada la posición del elemento terminal R_p (Ecuaciones (3.11), (3.12)), que es el extremo libre del eslabón 3. Es decir, con las siguientes ecuaciones puede plantearse un sistema de ecuaciones, como se muestra.

$$R_p = \{0,\ 2^*((A2^*A3 + A1^*A4)\ ^*L_3 - A1^*L_2),\ L_1 - A2^*L_2 + (A2^*A4 - 4^*A1^*A3)^*L_3\}; \qquad (4.6)$$

Donde:

$$A1 = P_0^*P_1; \quad A2 = 1 - 2^*P_0^2; \qquad (4.7)$$
$$A3 = Q_0^*Q_1; \quad A4 = 1 - 2^*Q_0^2;$$

Con la ecuación para R_p podemos plantear 2 ecuaciones si damos la posición para el elemento terminal como R_p = {0, R_py, R_pz} y con las normas unitarias de los Quaterniones P y Q completamos un sistema de 4 ecuaciones con 4 incógnitas, que al resolverlo obtenemos los parámetros que determinan la configuración no deformada.

$$\begin{gathered} 2^*((A2^*A3 + A1^*A4)\ ^*L_3 - A1^*L_2) = R_p y \\ L_1 - A2^*L_2 + (A2^*A4 - 4^*A1^*A3)^*L_3 = R_p z \\ P_0^2 + P_1^2 = 1 \\ Q_0^2 + Q_1^2 = 1 \end{gathered} \qquad (4.8)$$

Con lo anterior podemos ahora resolver el problema cinemático directo y realizar la correspondiente simulación computacional, que como se mencionó se realizará en Mathematica®. A continuación se presenta el programa desarrollado para este caso, que se mostrará en dos columnas, a la derecha el programa propiamente dicho y a la izquierda una breve explicación del programa. Con el objeto de mostrar la forma de utilizar Mathematica® para obtener las ecuaciones del manipulador, tanto en su configuración no deformada como en la deformada, se incluye el proceso completo para su obtención en esta parte del programa.

En la primera parte del programa se define la transformación rotación de un vector tridimensional y de un Quaternión, y de algunas otras funciones necesarias que se muestran a continuación.	**Definición de la Transformación Rotación.** (* Se apagan algunos mensajes del sistema *) Off[General::spell1] Off[General::spell]
Transformación de un Quaternión en un vector tridimensional.	(* Se define la Transformación de un Quaternión a un vector tridimensional Tv[] *)
q = {q_0, q_1, q_2, q_3} ∈ Q **Tv[q] = {q_1, q_2, q_3}.**	Tv[q_]:={q[[2]],q[[3]],q[[4]]}; (* Se define la Transformación inversa de Tv[] *)
Transformación vectorial inversa.	TvInv[v_]:={0,v[[1]],v[[2]],v[[3]]};

$v = \{v_1, v_2, v_3\} \in \Re^3$ **TvInv[v] = $\{0, v_1, v_2, v_3\}$.** Transformación de un Quaternión en un Quaternión vectorial. **TvQ[q] = $\{0, q_1, q_2, q_3\}$.** Producto de dos Quaterniones. $p = \{p_0, p_1, p_2, p_3\}$, $q = \{q_0, q_1, q_2, q_3\} \in Q$ **ProductoQ[p,q] =** $\{p_0 * q_0 - p_1 * q_1 - p_2 * q_2 - p_3 * q_3,$ $p_0 * q_1 + p_1 * q_0 + p_2 * q_3 - p_3 * q_2,$ $p_0 * q_2 - p_1 * q_3 + p_2 * q_0 + p_3 * q_1,$ $p_0 * q_3 + p_1 * q_2 - p_2 * q_1 + p_3 * q_0\}$ Conjugado de un Quaternión. **ConjugadoQ[p] = $\{p_0, -p_1, -p_2, -p_3\}$** Rotación de un vector $v \in \Re^3$ con un Quaternión **p**, siendo $\lVert p \rVert = 1$. Según la definición (2.13). $\rho(p,q) = \frac{1}{\lVert p \rVert^2} \bullet (p * q * \bar{p}), \quad \forall q \in Q$ **Rotacion[p,v] = Tv[p * TvInv[v] * $\bar{p}$]** Rotación de $q \in Q$ con un Quaternión **p**, siendo $\lVert p \rVert = 1$. De acuerdo con la definición (2.13) y considerando que el primer elemento de **q** no cambia al rotar cuando $\lVert p \rVert = 1$.. **RotacionQ[p,q] = $\{q_0,0,0,0\}$ + TvQ[p*q* $\bar{p}$]**	(* Se define la Transformación de un Quaternión a Quaternión vectorial *) TvQ[q_]:={0,q[[2]],q[[3]],q[[4]]}; (* Se define la Multiplicación entre dos Quaterniones *) ProductoQ[p_,q_]:= {p[[1]]*q[[1]] - p[[2]]*q[[2]] - p[[3]]*q[[3]] - p[[4]]*q[[4]], p[[1]]*q[[2]] + p[[2]]*q[[1]] + p[[3]]*q[[4]] - p[[4]]*q[[3]], p[[1]]*q[[3]] - p[[2]]*q[[4]] + p[[3]]*q[[1]] + p[[4]]*q[[2]], p[[1]]*q[[4]] + p[[2]]*q[[3]] - p[[3]]*q[[2]] + p[[4]]*q[[1]]}; (* Se define el conjugado de un Quaternión *) ConjugadoQ[p_]:={p[[1]],-p[[2]],-p[[3]],-p[[4]]}; (* Se define la Transformación Rotación de un vector tridimensional *) Rotacion[p_,v_]:= Tv[ProductoQ[p, ProductoQ[TvInv[v], ConjugadoQ[p]]]]; (* Se define la Transformación Rotación de un Quaternión *) RotacionQ[p_,q_]:={q[[1]],0,0,0}+ TvQ[ProductoQ[p,ProductoQ[q,ConjugadoQ[p]]]];
Se realizan definiciones iniciales necesarias, como son la base canónica de $\Re^3$ y las bases que servirán como **bases locales** en los eslabones. También se define la forma local de los Quaterniones que nos permitirán rotar, tanto en la posición no deformada como en la deformada.	**Definiciones iniciales** (* Base canónica de R3*) e1 = {1,0,0}; e2 = {0,1,0}; e3 = {0,0,1}; (* Bases locales de los eslabones *) (* Eslabón 1 *) e1 = {1,0,0}; e2 = {0,1,0}; e3 = {0,0,1}; (* Eslabón 2 *) f1 = {1,0,0}; f2 = {0,1,0}; f3 = {0,0,1};

	(* Eslabón 3 *) g1 = {1,0,0}; g2 = {0,1,0}; g3 = {0,0,1}; (*Quaterniones para rotar a posición no deformada*) P = {P0,P1,0,0}; (* Eslabón 2 *) Q = {Q0,Q1,0,0}; (* Eslabón 3 *) (* Quaterniones para rotar a posición deformada *) r1 = {r10, 0, 0, r13}; (* Eslabón 1 *) r2 = {r20, r21, 0, 0}; (* Eslabón 2 *) r3 = {r30, r31, 0, 0}; (* Eslabón 3 *)
En esta sección del programa se realiza la modelación de la configuración no deformada del manipulador. Esta se obtiene rotando con la secuencia 3-2 (primero se rota el tercer eslabón y después el segundo). Se definen las normas unitarias de los Quaterniones P y Q y se almacenan en las variables "normas1" y "normas2", para aplicarlas en las ecuaciones resultantes de las rotaciones. Las variables que se utilizan para las bases locales de los eslabones son: ej = Primer eslabón. fj = Segundo eslabón. gj = Tercer eslabón. j = 1,2,3. Para identificar las bases después de aplicar rotaciones, se les agrega a cada variable un número que identifica la rotación aplicada.	**MODELADO DE LA CONFIGURACION NO DEFORMADA** **Modelo para la configuración no deformada (Secuencia 3-2)** (* Se inicializan variables *) ClearAll[P0,P1,Q0,Q1,L1,L2,L3,l1,l2,l3]; (* Se definen variables para las normas unitarias de los Quaterniones P y Q *) normas1 = {P1->(1-P0^2)^(1/2), Q1->(1-0^2)^(1/2)}; normas2 = {(1-P0^2)^(1/2)->P1, (1-Q0^2)^(1/2)-Q1}; (* Rotación de bases locales y ejes de rotación. Se rota primero el tercer eslabón y luego el segundo *) (* Rotación del tercer eslabón (base "g") con Quaternión Q *) g11 = Rotacion[Q,g1]//.normas1//.normas2; g21 = Rotacion[Q,g2]//.normas1//.normas2; g31 = Rotacion[Q,g3]//.normas1]//.normas2; (* Rotación del segundo eslabón (base "f") con Quaternión P *) f11 = Rotacion[P,f1]//.normas1//.normas2; f21 = Rotacion[P,f2]//.normas1//.normas2; f31 = Rotacion[P,f3]//.normas1//.normas2; (* Rotación del tercer eslabón (base "g") con Quaternión P *) g12 = Expand[Expand[Rotacion[P,g11]] //.normas1]//.normas2; g22 = Expand[Expand[Rotacion[P,g21]] //.normas1]//.normas2; g32 = Expand[Expand[Rotacion[P,g31]] //.normas1]//.normas2;

Los eslabones del manipulador y la posición del elemento terminal son identificados, en la posición no deformada, por las variables siguientes: b1 = Eslabón 1. b2 = Eslabón 2. b3 = Eslabón 3. Rp = b1+b2+b3; Rp = Posición del elemento terminal. Las ecuaciones se simplifican utilizando las siguientes constantes.. A1 = P0*P1; A2 = 1 - 2*P0^2; A3 = Q0*Q1; A4 = 1 - 2*Q0^2;	(* Vectores que definen los eslabones del manipulador *) b1 = L1*e3; (* Eslabón 1 *) b2 = L2*f31; (* Eslabón 2 *) b3 = L3*g32; (* Eslabón 3 *) (* Constantes que se repiten y que servirán para simplificar las ecuaciones y optimizar operaciones *) constantes = {P0*P1->A1, P0^2->(1-A2)/2, Q0*Q1->A3, Q0^2->(1-A4)/2}; b2 = b2//.constantes; b3 = Expand[Expand[b3//.constantes]//.constantes]; (*Vector que define posición del elemento terminal *) Rp = b1+b2+b3;
Es esta parte se muestran las ecuaciones obtenidas en la sección anterior, las cuales se utilizarán para determinar los parámetros P y Q, que definen la configuración no deformada en una posición dada del elemento terminal, que será de donde arranque la simulación de la cinemática directa del manipulador en cuestión. Es decir, antes de generar una simulación computacional de la cinemática directa del manipulador de tres grados de libertad, debemos determinar la configuración no deformada realizando un proceso de cinemática inversa para calcular P y Q para una posición conocida del elemento terminal, la cual debe tener la forma siguiente: Rp = {0,Rpy, Rpz}.	**Ecuaciones optimizadas para la posición no deformada.** ClearAll[P0,P1,Q0,Q1] (* Ecuaciones optimizadas *) b1 = {0, 0, L1}; (* Eslabón 1 *) b2 = {0, -2*A1*L2, -A2*L2}; (* Eslabón 2 *) b3 = {0, 2*(A2*A3 + A1*A4)*L3, (* Eslabón 3 *) (A2*A4 - 4*A1*A3)*L3}; (* Posición el elemento terminal *) Rp = {0, 2*((A2*A3 + A1*A4)*L3 - A1*L2), L1 - A2*L2 + (A2*A4 - 4*A1*A3)*L3}; (* Constantes de las ecuaciones *) A1 = P0*P1; A2 = 1 - 2*P0^2; A3 = Q0*Q1; A4 = 1 - 2*Q0^2; (* Longitud de los eslabones del manipulador *) L1 = 6; L2 = 5; L3 = 4;
Se resuelve el sistema de ecuaciones mostrado en (4.6), para P0,P1,Q0,Q1, utilizando la función FindRoot, cuya sintaxis se puede consulta en el Apéndice, al igual que todas las funciones que se utilizan en este trabajo. Los resultados obtenidos se redondean si	**Cinemática Inversa para determinar la Configuración no Deformada** (* Conociendo la posición del elemento terminal se determinan los Quaterniones "P" y "Q" correspondientes *)

alguno de ellos está muy cercano a cero, finalmente se almacenan en las variables correspondientes (P0, P1, Q0, Q1) para poder ser utilizados posteriormente, además se imprimen dichos valores.	ClearAll[P0,P1,Q0,Q1] (* Se da la posición del elemento terminal *) posicion = {0, 0, 15}; (* Se determinan "P" y "Q" utilizando los siguiente valores iniciales *) iP0 = .7071; iP1 = .7071; iQ0 = .866; iQ1 = .5; (* Se almacena la solución en la variable sol y se utiliza la función FindRoot para determinar la solución del sistema de ecuaciones *) sol = FindRoot[{Rp2 == posicion[[2]],Rp3 == posicion[[3]], P0^2+P1^2 == 1, Q0^2+Q1^2 == 1}, {P0,iP0}, {P1,iP1}, {Q0,iQ0}, {Q1,iQ1}, MaxIterations->50]; (* Se asignan los valores de la solución a las variables correspondientes *) P0=P0/.sol; P1=P1/.sol; Q0=Q0/.sol; Q1=Q1/.sol; (* Se redondean los valores de la solución *) If[Abs[P0]==1,P1=0,If[Abs[P1]==1,P0=0]]; If[Chop[Abs[Q0]-1]==0,Q0=1;Q1=0, If[Chop[Abs[Q1]-1]==0,Q0=0;Q1=1]]; (* Se imprimen los resultados *) Print["P0 = ",P0 ," ", "P1 = ", P1, " Q0 = ", Q0, " ", "Q1 = ", Q1]
De la misma forma que se realizó la modelación de la configuración no deformada, ahora se presenta el desarrollo para programar la generación de la configuración deformada, que básicamente sigue el siguiente procedimiento. Se parte de la forma que adquirieron las bases locales de los eslabones en la configuración no deformada y se les aplican las rotaciones correspondientes a los tres eslabones en la secuencia 3-2-1. También en este caso, se definen las normas unitarias de los Quaterniones utilizados, almacenándolas en las variables "normas3" y "normas4", es de hacer notar que en estas variables se incluyen las normas unitarias de los	**MODELADO DE LA CONFIGURACION DEFORMADA** **Modelo para la configuración deformada (Secuencia 3-2-1)** ClearAll[r10,r13,r20,r21,r30,r31,L1,L2,L3]; (* Posición no deformada conocida *) P = {P0,P1,0,0}; Q = {Q0,Q1,0,0}; (* Quaterniones para rotar en secuencia *) (* 1.- Rotación para el tercer eslabón *) r3 = {r30,r31,0,0};

Quaterniones correspondientes a la configuración no deformada con el objeto de lograr la mayor simplificación de las ecuaciones resultantes..	(* 2.- Rotación para segundo y tercer eslabones *) r2 = {r20,r21,0,0}; (* 3.- Rotación para los tres eslabones *) r1 = {r10,0,0,r13}; (* Se definen normas de los Quaterniones *) normas3 = Flatten[{normas1, r13->(1-r10^2)^(1/2), r21->(1-r20^2)^(1/2), r31->(1-r30^2)^(1/2)}]; normas4 = Flatten[{normas2, (1-r10^2)^(1/2)->r13, (1-r20^2)^(1/2)->r21, (1-r30^2)^(1/2)->r31}];
Se aplica la primera rotación, que consiste en rotar el tercer eslabón con el Quaternión r3, esto afecta solamente a la base local g_j.	(* Se aplica primera rotación de la secuencia ("r3") *) (* Rotación del tercer eslabón (base "g") con Quaternión r3 *) g13 = Simplify[Expand[Rotacion[r3,g12]]//.normas3] //.normas4; g23 = Simplify[Expand[Rotacion[r3,g22]]//.normas3] //.normas4; g33 = Simplify[Expand[Rotacion[r3,g32]]//.normas3] //.normas4;
Se aplica la segunda rotación, que consiste en rotar el segundo eslabón con el Quaternión r2, que también hace girar de la misma manera al eslabón 3 y con ello se afecta a las bases locales f_j y g_j, que son las actualizadas.	(*Se aplica segunda rotación de la secuencia ("r2")*) (* Rotación del segundo eslabón (base "f") con Quaternión r2 *) f12 = Simplify[Expand[Rotacion[r2,f11]]//.normas3] //.normas4; f22 = Simplify[Expand[Rotacion[r2,f21]]//.normas3] //.normas4; f32 = Simplify[Expand[Rotacion[r2,f31]]//.normas3] //.normas4; (* Rotación del tercer eslabón (base "g") con Quaternión r2 *) g14 = Simplify[Expand[Rotacion[r2,g13]]//.normas3] //.normas4; g24 = Simplify[Expand[Rotacion[r2,g23]]//.normas3] //.normas4; g34 = Simplify[Expand[Rotacion[r2,g33]]//.normas3] //.normas4;
Se aplica la tercera rotación, que consiste en rotar el primer eslabón con el Quaternión r1, lo que hace girar de la misma forma a los otros dos eslabones y con ello se afecta a las	(* Se aplica tercera rotación de la secuencia ("r1") *) (* Rotación del primer eslabón (base "e") con Quaternión r1 *)

bases locales e_j, f_j y g_j, que son las actualizadas. Al aplicar las rotaciones, también se simplifican las expresiones obtenidas, aplicando las normas unitarias de los Quaterniones y funciones de simplificación propias de Mathematica[5].	e11 = Simplify[Expand[Rotacion[r1,e1]]//.normas3] //.normas4; e21 = Simplify[Expand[Rotacion[r1,e2]]//.normas3] //.normas4; e31 = Simplify[Expand[Rotacion[r1,e3]]//.normas3] //.normas4; (* Rotación del segundo eslabón (base "f") con Quaternión r1 *) f13 = Simplify[Expand[Rotacion[r1,f12]]//.normas3] //.normas4; f23 = Simplify[Expand[Rotacion[r1,f22]]//.normas3] //.normas4; f33 = Simplify[Expand[Rotacion[r1,f32]]//.normas3] //.normas4; (* Rotación del tercer eslabón (base "g") con Quaternión r1 *) g15 = Simplify[Expand[Rotacion[r1,g14]]//.normas3] //.normas4; g25 = Simplify[Expand[Rotacion[r1,g24]]//.normas3] //.normas4; g35 = Simplify[Expand[Rotacion[r1,g34]]//.normas3] //.normas4;
Una vez obtenidas las bases locales en su forma final, después de aplicadas todas las rotaciones propuestas, se obtienen las ecuaciones que representan los eslabones del manipulador y la posición del elemento terminal como: **$b_1 = L_1{*}e_3^{I} = L_1{*}e31$; (Eslabón 1)** **$b_2 = L_2{*}f_3^{III} = L_2{*}f33$; (Eslabón 2)** **$b_3 = L_3{*}g_3^{V} = L_3{*}g35$; (Eslabón 3)** **(Posición del elemento terminal)** **$r_p = b_1 + b_2 + b_3$** Estas expresiones son simplificadas sustituyendo por constantes a aquellos términos que se repiten dentro de las diferentes ecuaciones, como se puede observar.	(* Vectores que definen los eslabones del manipulador *) b1 = L1*e31; (* Eslabón 1 *) b2 = L2*f33; (* Eslabón 2 *) b3 = L3*g35; (* Eslabón 3 *) (* Constantes que se repiten y que servirán para simplificar las ecuaciones y optimizar operaciones *) constantes = {P0*P1->A1, P0^2->(1-A2)/2, Q0*Q1->A3, Q0^2->(1-A4)/2, r10*r13->A5, r10^2->(1-A6)/2, r20*r21->A7,r20^2->(1-A8)/2, r30*r31->A9,r30^2->(1-A10)/2, A2*A8->4*A1*A7+B1, A2*A7->B2-A1*A8, A10*A8->4*A7*A9+B3, A10*A7->B4-A8*A9, A2*A4->4*A1*A3+B5, A2*A3->B6-A1*A4, B4*B5->C1-B3*B6, B3*B5->C2+4*B4*B6}; (* Se simplifican las expresiones *) b2 = Expand[b2//.constantes]; b2 = Expand[Collect[b2,{L2,A5,A6}]//.constantes]; b3 = Expand[b3//.constantes]; b3 = Expand[Collect[Collect[b3,{L3,A5,A1,A2,A3,A4, A6}] //.constantes,{L3,A5,A6,B3,B4}]//.constantes]; b3 = Expand[b3//.constantes];

<table>
<tr><td></td><td>(*Vector que define posición del elemento terminal *)

rp = b1 + b2 + b3;</td></tr>
<tr><td>En esta parte se presenta el resultado obtenido en la sección anterior, es decir, las ecuaciones simplificadas del manipulador, así como, las constantes utilizadas en dichas ecuaciones. Esta sección es la que propiamente se utiliza para realizar la simulación de la cinemática directa del manipulador de tres grados de libertad, ya que es la que contiene las ecuaciones ya simplificadas. Puede observarse, que además de las ecuaciones de los eslabones y de la posición del elemento terminal, se tiene una ecuación para e11 = e_1^1; ésta sirve para mostrar el movimiento del primer eslabón, que al girar sobre su propio eje, requiere que se dibuje una de las bases perpendiculares al él para ver claramente su movimiento.
Como al realizar la simulación es necesario evaluar numéricamente las ecuaciones del manipulador, se genera un respaldo de dichas ecuaciones en esta misma sección, almacenándolas en nuevas variables.</td><td>Ecuaciones optimizadas para la posición deformada (Secuencia 3-2-1)

ClearAll[r10,r13,r20,r21,r30,r31,L1,L2,L3];

(* Ecuaciones Simplificadas *)

b1 = {0, 0, L1};

b2 = {-4*A5*B2*L2, -2*A6*B2*L2, B1*L2};

b3 = {-4*A5*C1*L3, -2*A6*C1*L3, C2*L3};

rp = {-4*A5*(B2*L2 + C1*L3),
-2*A6*(B2*L2 + C1*L3),
L1 + B1*L2 + C2*L3};

(* Componentes ortogonales de rp *)

rp1 = rp[[1]]; rp2 = rp[[2]]; rp3 = rp[[3]];

(* Componente 1 de la Base local del eslabón 1 (para dibujar la base) *)

e11 = {-A6,2*A5, 0};

(* Constantes de las ecuaciones *)

A1 = P0*P1; A2 = 1-2*P0^2;
A3 = Q0*Q1; A4 = 1-2*Q0^2;
A5 = r10*r13; A6 = 1-2*r10^2;
A7 = r20*r21; A8 = 1-2*r20^2;
A9 = r30*r31; A10 = 1-2*r30^2;

B1 = A2*A8 - 4*A1*A7;
B2 = A2*A7 + A1*A8;
B3 = A10*A8 - 4*A7*A9;
B4 = A10*A7 + A8*A9;
B5 = A2*A4 - 4*A1*A3;
B6 = A2*A3 + A1*A4;
C1 = B4*B5 + B3*B6;
C2 = B3*B5 - 4*B4*B6;

(* Longitud de los eslabones del manipulador *)

L1 = 6; L2 = 5; L3 = 4;

(* Respaldo de ecuaciones *)

a11=e11; z1=b1; z2=b2; z3=b3; rx=rp;</td></tr>
</table>

Finalmente, en esta parte del programa, se presenta la generación de las gráficas que nos permitirán observar la simulación de la cinemática directa del manipulador en cuestión.

Para lograr generar las gráficas adecuadas, para poder observar la simulación, primero generamos mediante la función ***SurfaceGrafics*** una gráfica que contenga el espacio donde se mostrará el manipulador con una superficie simulando el piso y con la indicación de los ejes coordenados, la cual se almacena inicialmente en las variables **g1** y **g2**.

Definimos los parámetros de rotación de cada eslabón, en función de los ángulos que utilizaremos para girar cada uno de los eslabones, que serán:

Eslabón 1 → **theta1**
Eslabón 2 → **theta2**
Eslabón 3 → **theta3**

Se generan dos ciclos ***For***, el primero (variable de control **i**) servirá para controlar cual eslabón deberá moverse, o sea, cuándo variar que ángulo, lo cual se logra utilizando la función ***Switch***, y el segundo ciclo (variable de control **j**) nos permitirá generar varias gráficas del movimiento de cada eslabón. La variable **i** toma valores de 1 a 3, ya que son tres eslabones y la variable **j** toma valores de 1 a n, donde n es el número de gráficas a generar por cada eslabón.

Una vez que los parámetros de rotación tienen un valor numérico, se evalúan numéricamente las ecuaciones del manipulador, haciendo uso del respaldo de ecuaciones realizado anteriormente

Con las ecuaciones, ya numéricas, se modifica la gráfica almacenada en **g2** agregándole un punto en la posición que corresponde a **rp**.

De la misma forma, con las ecuaciones numéricas, se determinan los diferentes puntos de la gráfica a generar, es decir, se genera un punto para cada inicio y final de cada eslabón, además de generar puntos

CINEMATICA DIRECTA

Graficación para visualizar la simulación.

```
ClearAll[r10,r13,r20,r21,r30,r31];

(* Se define el espacio tridimensional en que se
   graficará el manipulador, que se usará como
   gráfica base *)

 (* Suma de la longitud del eslabón 2 y 3 *)
   L23 = L2+L3;

g1=g2=SurfaceGraphics[{{0,0,0},{0,0,0},{0,0,0}},
   MeshRange->{{-(L23),L23},{-(L23),L23}},
   AxesLabel->{Eje X,Eje Y,Eje Z},
              DefaultFont->{"Negrita",12},
   PlotRange->{{-(L23+.1),(L23+.1)},
              {-(L23+.1),(L23+.1)},
              {(L1-(L23)-.1),(L1+L23+.1)}},
   BoxRatios->{1,1,1},Axes->True];

(* Se inicializan en cero los ángulos de rotación de
   los tres eslabones *)

     theta1 = 0; theta2 = 0 ;theta3 = 0;

(* Se definen los valores de los parámetros de las
   rotaciones en función de los ángulos de rotación*)

              r10=Cos[theta1/2*Degree];
              r13=Sin[theta1/2*Degree];
              r20=Cos[theta2/2*Degree];
              r21=Sin[theta2/2*Degree];
              r30=Cos[theta3/2*Degree];
              r31=Sin[theta3/2*Degree];

(* Se inicializa un ciclo para mover los tres
   eslabones, uno a la vez *)

For[i=1, i<=3, i++,

(* Se inicializa otro ciclo para generar varias gráficas
   para cada eslabón en diferentes posiciones*)

For[j=1, j<=5, j++,

(* Se selecciona, mediante la función Switch, cual
   eslabón se moverá según el valor de la variable i,
   lo cual se logra cambiando sus parámetros de
   rotación *)

 Switch[i,3,  (* Se mueve el primer eslabón *)
              theta1+=15;
```

para mostrar el vector **e11** de la base del primer eslabón.

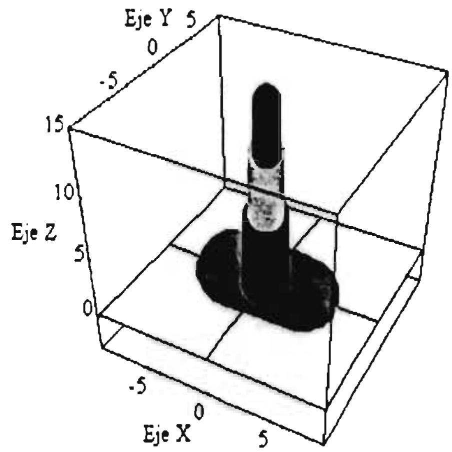
Con los puntos anteriores, se genera la gráfica correspondiente a la posición del manipulador, utilizando para ello la función ***Graphics3D***, que nos permite trazar líneas de diferente color y espesor entre los puntos determinados.

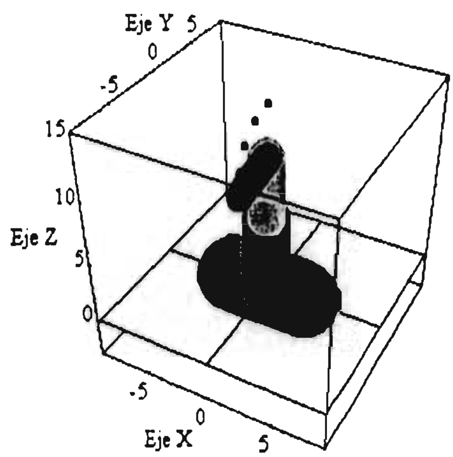
Finalmente se muestran en la pantalla, de manera combinada, las gráficas **g1, g2** y **g3** que se generan en cada pasada de los ciclos, utilizando para esto la función ***Show*** la cual envía a la pantalla objetos gráficos y permite que se combinen varias gráficas.

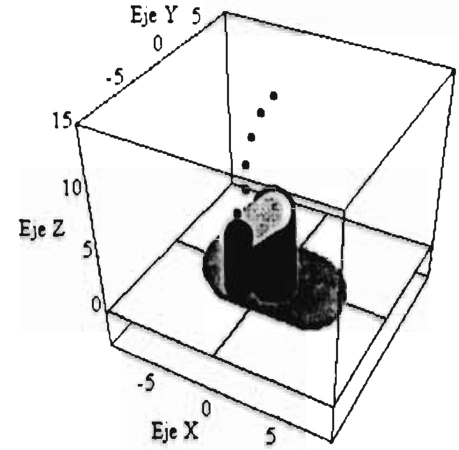
Se muestran, a manera de ejemplo, algunas gráficas de un proceso.

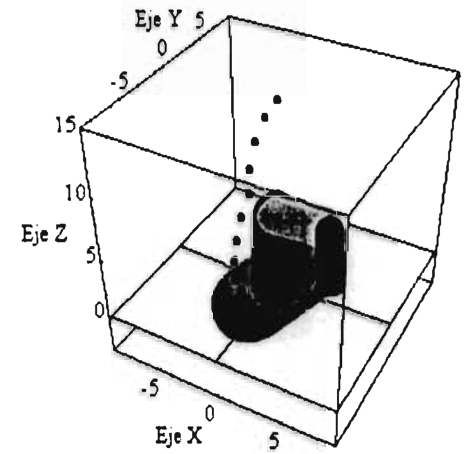
```
         r10=N[Cos[theta1/2 Degree]];
         r13=N[Sin[theta1/2 Degree]];,
      2, (* Se mueve el segundo eslabón *)
         theta2+=15;
         r20=N[Cos[theta2/2 Degree]];
         r21=N[Sin[theta2/2 Degree]];,
      1, (* Se mueve el tercer eslabón *)
         r30=N[Cos[theta3/2 Degree]];
         r31=N[Sin[theta3/2 Degree]];
         theta3+=25; ];

(* Se asignan valores a las ecuaciones del
   manipulador, según los valores de los
   parámetros, utilizando el respaldo de
   ecuaciones*)

base=N[a11]; b1=N[z1]; b2=N[z2]; b3=N[z3];
rp=N[rx];

(* Se agrega a la grafica base la posición del
   elemento terminal como un punto *)

g2={Graphics3D[{PointSize[0.05],
                  Point[rp],RGBColor[0,1,0]}],g2};

(* Longitud del vector que servirá de base del
   manipulador *)

a=3;

(* Se definen y calculan los puntos en el espacio
   tridimensional para graficar el manipulador *)

punto1 = {0,0,0};
punto2 = N[a*base];
punto3 = N[-a*base];
punto4 = N[b1];
punto5 = N[punto4 + b2];
punto6 = N[punto5 + b3];

(* Se genera la gráfica del manipulador en la
   posición correspondiente *)

g3=Graphics3D[{

(* Base rotatoria *)
        {AbsoluteThickness[40],RGBColor[0,.5,0],
         Line[{punto2,punto3}]},
(* Eslabón 1 *)
        {AbsoluteThickness[25],RGBColor[0,1,0],
         Line[{punto1,punto4}]},
(* Eslabón 2 *)
        {AbsoluteThickness[20],RGBColor[1,0,0],
         Line[{punto4,punto5}]},
```

```
(* Eslabón 3 *)
        {AbsoluteThickness[15],RGBColor[0,0,1],
         Line[{punto5,punto6}]}}];

(* Se envían a pantalla las gráficas g1,g2 y g3,
   combinadas *)

Show[g1,g3,g2,Axes->True];
                              ]
                                     ]
```







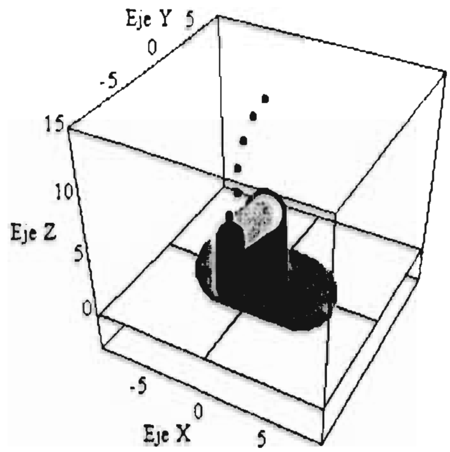


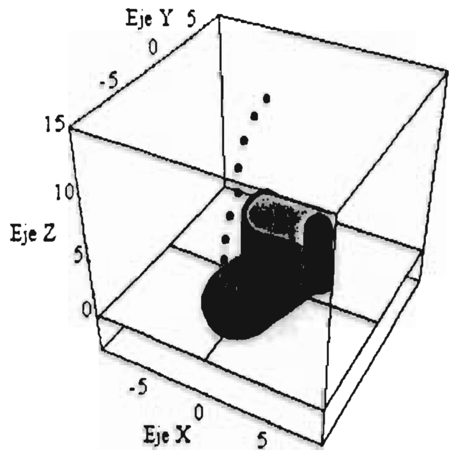
4.1.2 Cinemática Inversa.

La cinemática inversa del robot de tres grados de libertad, como se mencionó anteriormente, consiste en que, dada la posición del elemento terminal, se deben determinar los ángulos que deberán desplazarse cada una de las juntas del manipulador para alcanzar la posición deseada. En este caso, consiste en que dada la configuración no deformada (P_0, P_1, Q_0, Q_1,) y la posición del elemento terminal (r_p = {$r_p x$, $r_p y$, $r_p z$}), se deben determinar los parámetros $r1_0$, $r1_3$, $r2_0$, $r2_1$, $r3_0$, $r3_1$ que satisfagan la ecuación de la posición del elemento terminal, que es la siguiente:

$$r_p = \{- 4*A5*(B2*L_2 + C1*L_3),\ - 2*A6*(B2*L_2 + C1*L_3),\ L_1 + B1*L_2 + C2*L_3\} \qquad (4.9)$$

Para resolver la cinemática inversa del robot de tres grados de libertad, debemos plantear un sistema de 6 ecuaciones que nos permitan determinar los 6 parámetros de rotación buscados. De la ecuación de r_p obtenemos 3 ecuaciones y con las normas unitarias de los Quaterniones se completa el sistema, el cual se muestra a continuación.





4.1.2 Cinemática Inversa.

La cinemática inversa del robot de tres grados de libertad, como se mencionó anteriormente, consiste en que, dada la posición del elemento terminal, se deben determinar los ángulos que deberán desplazarse cada una de las juntas del manipulador para alcanzar la posición deseada. En este caso, consiste en que dada la configuración no deformada (P_0, P_1, Q_0, Q_1,) y la posición del elemento terminal (r_p = {$r_p x$, $r_p y$, $r_p z$}), se deben determinar los parámetros $r1_0$, $r1_3$, $r2_0$, $r2_1$, $r3_0$, $r3_1$ que satisfagan la ecuación de la posición del elemento terminal, que es la siguiente:

$$r_p = \{-4{*}A5{*}(B2{*}L_2 + C1{*}L_3),\ -2{*}A6{*}(B2{*}L_2 + C1{*}L_3),\ L_1 + B1{*}L_2 + C2{*}L_3\} \qquad (4.9)$$

Para resolver la cinemática inversa del robot de tres grados de libertad, debemos plantear un sistema de 6 ecuaciones que nos permitan determinar los 6 parámetros de rotación buscados. De la ecuación de r_p obtenemos 3 ecuaciones y con las normas unitarias de los Quaterniones se completa el sistema, el cual se muestra a continuación.

$$r_px = -4{*}A5{*}(B2{*}L_2 + C1{*}L_3)$$
$$r_py = -2{*}A6{*}(B2{*}L_2 + C1{*}L_3)$$
$$r_pz = L_1 + B1{*}L_2 + C2{*}L_3 \qquad (4.10)$$
$$r1_0^2 + r1_3^2 = 1$$
$$r2_0^2 + r2_1^2 = 1$$
$$r3_0^2 + r3_1^2 = 1$$

Siendo:

$$\begin{array}{ll} A1 = P_0{*}P_1; & A2 = 1\text{-}2{*}P_0^2; \\ A3 = Q_0{*}Q_1; & A4 = 1\text{-}2{*}Q_0^2; \\ A5 = r1_0{*}r1_3; & A6 = 1\text{-}2{*}r1_0^2; \\ A7 = r2_0{*}r2_1; & A8 = 1\text{-}2{*}r2_0^2; \\ A9 = r3_0{*}r3_1; & A10 = 1\text{-}2{*}r3_0^2; \\ B1 = A2{*}A8 - 4{*}A1{*}A7; & B2 = A2{*}A7 + A1{*}A8; \\ B3 = A10{*}A8 - 4{*}A7{*}A9; & B4 = A10{*}A7 + A8{*}A9; \\ B5 = A2{*}A4 - 4{*}A1{*}A3; & B6 = A2{*}A3 + A1{*}A4; \\ C1 = B4{*}B5 + B3{*}B6; & C2 = B3{*}B5 - 4{*}B4{*}B6; \end{array} \qquad (4.11)$$

Lo anterior nos permite ubicar el elemento terminal en el punto $\{r_px,\ r_py,\ r_pz\}$ dentro de su espacio de trabajo, ahora si deseamos que el elemento terminal siga una trayectoria determinada, una de las cosas que podemos hacer es discretizar dicha trayectoria en varios puntos a lo largo de ella y determinar los parámetros de rotación de los eslabones del manipulador para cada punto, resolviendo el sistema de ecuaciones (4.11), lo que nos permitirá ubicar el elemento terminal sobre dicha secuencia de puntos. Lo anterior no permite ubicar el elemento terminal en cualquier punto de la trayectoria, solo en los puntos en que dicha trayectoria se discretizó. A continuación se presenta el programa desarrollado en Mathematica[5] para la cinemática inversa del manipulador de tres grados de libertad, el cual es continuación del anterior donde se presenta la cinemática directa.

Este programa es parte integral del anterior, es decir, para que funcione deberá tenerse la parte anterior donde se presenta la modelación y la cinemática directa, ya que se requieren las ecuaciones del manipulador. En esta sección se presenta el cálculo para tres diferentes trayectorias y desde luego la graficación para observar la simulación.	**CINEMATICA INVERSA** **Solución de ecuaciones para diferentes trayectorias**
En esta parte se determinan los parámetros de rotación de los eslabones, para una sucesión de puntos que pertenecen a una línea recta, la cual se define por sus dos puntos extremos(S, S1). Para lo anterior, primeramente recuperamos las ecuaciones del manipulador, las cuales se respaldaron en otras variables, (deberá resolverse la cinemática inversa para determinar la configuración no deformada y ejecutar la celda donde están las ecuaciones optimizadas para la configuración no deformada).	**LINEA RECTA** (* Se inicializan variables *) ClearAll[r10, r13, r20, r21, r30, r31, "solp", "solq", "solr", "solucion"]; (*Se recuperan las ecuaciones del manipulador*) b1=z1; b2=z2; b3=z3; rp=rx; rp1=rp[[1]]; rp2=rp[[2]]; rp3=rp[[3]]; (* Variables de control *) t = 0; (* Para discretizar en varios puntos la trayectoria seleccionada *)

Para determinar la puntos sobre la trayectoria, utilizamos la forma paramétrica de una recta que pasa por dos puntos $p_1 = \{x_1, y_1, z_1\}$ y $p_2 = \{x_1, y_1, z_1\}$:

$$px = x_1 + (x_2 - x_1)t$$
$$py = y_1 + (y_2 - y_1)t$$
$$pz = z_1 + (z_2 - z_1)t$$

Con lo anterior se realiza un ciclo utilizando como variable de control del mismo, a la variable **t**, con lo que determinamos puntos sobre la trayectoria a seguir y resolvemos el sistema de ecuaciones (4.1).

$$r_p x = -4{*}A5{*}(B2{*}L_2 + C1{*}L_3),$$
$$r_p y = -2{*}A6{*}(B2{*}L_2 + C1{*}L_3),$$
$$r_p z = L_1 + B1{*}L_2 + C2{*}L_3\}$$
$$r1_0^2 + r1_3^2 = 1$$
$$r2_0^2 + r2_1^2 = 1$$
$$r3_0^2 + r3_1^2 = 1$$

Este sistema de ecuaciones se resuelve utilizando la función Findroot usando los valores iniciales mostrados. Una vez que se resuelve el sistema de ecuaciones anterior, se verifica si la precisión de la solución es la adecuada, si es así, se fijan como valores iniciales, para resolver la cinemática inversa del siguiente punto, a la solución obtenida y se continua el ciclo, si la solución está fuera de la precisión se envía un mensaje y se restauran los valores iniciales continuando el ciclo hasta terminar.

```
i = 1; (* Para guardar la solución de cada punto
        de la trayectoria *)
(* Valores iniciales para resolver el sistema de
   ecuaciones *)

ir10 = .7071; ir13 = .7071; ir20 = .3; ir21 = .9;
ir30 = .5; ir31 = .8;

(* Trayectoria en línea recta dada por los puntos
   extremos *)

        S0 = {5,5,7};  S1 = {5,-5,7};

(* Se inicia un ciclo para resolver el problema
   cinemático inverso en los puntos en que será
   discretizada la trayectoria *)

For[t=0, t<1, t+=.1,

(* Se determinan las coordenadas del punto
   correspondiente de la trayectoria*)
        px=S0[[1]]+(S1[[1]]-S0[[1]])*t;
        py=S0[[2]]+(S1[[2]]-S0[[2]])*t;
        pz=S0[[3]]+(S1[[3]]-S0[[3]])*t;

(* Solución del problema cinemático inverso para
   el punto seleccionado *)

solucion[i] =
FindRoot[{rp1 == px, rp2 == py, rp3 == pz,
          r10^2+r13^2 == 1, r20^2+r21^2 == 1,
          r30^2+r31^2 == 1},
         {r10,ir10},{r13,ir13},{r20,ir20},
         {r21,ir21},{r30,ir30},{r31,ir31},
         MaxIterations->50];

(* Se verifica la precisión de la solución *)

        e = 10^-5; (* Precisión requerida *)

If[Abs[px-(rp1/.solucion[i])] < e &&
  Abs[py-(rp2/.solucion[i]) ]< e &&
  Abs[pz-(rp3/.solucion[i])] < e,

(* Si la solución está en la precisión fijada, se
   definen como valores iniciales, para resolver la
   cinemática inversa del siguiente punto de la
   trayectoria a los valores de la solución obtenida,
   si no es así se restauran los valores iniciales
   utilizados *)

ir10=r10/.solucion[i]; ir13=r13/.solucion[i];
ir20=r20/.solucion[i]; ir21=r21/.solucion[i];
ir30=r30/.solucion[i]; ir31=r31/.solucion[i],
```

(* Se imprime un mensaje si la solución no está en la precisión definida *)

Print["No se encontró solución para el punto:",
{px,py,pz}];

(* Se restauran valores iniciales para resolver las ecuaciones si no se encontró solución para el punto en cuestión *)

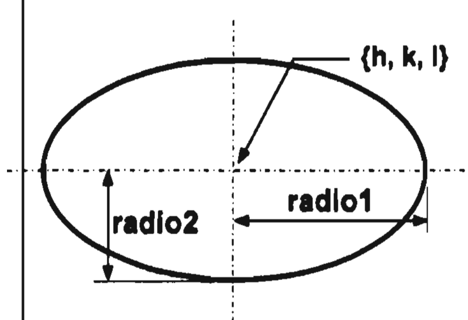
ir10=.7071; ir13=.7071; ir20=.3; ir21=.9;
ir30=.5; ir31=.8];

(* Se incremente la variable i y se cierra el ciclo For *)

i++]

En esta sección, al igual que en la anterior se resuelve la cinemática inversa para una secuencia de puntos sobre una trayectoria, que en este caso se trata de una elipse o circunferencia. El proceso que se sigue es el mismo que para la línea recta, solamente cambia la manera de determinar los puntos sobre la trayectoria, que en este caso se realiza utilizando la forma paramétrica de una elipse en el plano y-z, con centro en {h, k, l} con radios *radio1* y *radio2*.

px = h;
py = radio1 * Cos[t] +k ;
pz = radio2 * Sin[t] + l;



CIRCUNFERENCIAS O ELIPSES

(* Se inicializan variables *)
ClearAll[r10, r13, r20, r21, r30, r31, "solp", "solq", "solr", "solucion"];

(*Se recuperan las ecuaciones del manipulador*)
b1=z1; b2=z2; b3=z3; rp=rx; rp1=rp[[1]];
rp2=rp[[2]]; rp3=rp[[3]];

(* Variables de control *)

t = 0; (* Para discretizar en varios puntos la trayectoria seleccionada *)
i = 1; (* Para guardar la solución de cada punto de la trayectoria *)

(*Variables de control para definir la trayectoria *)

radio1 = 7; (* Radio uno de la elipse *)
radio2 = 5; (* Radio dos de la elipse *)
h=5;k=0;l=6; (* Centro de la elipse *)

(* Valores iniciales para resolver el sistema de ecuaciones *)

ir10 = .7071; ir13 = .7071; ir20 = .3; ir21 = .9;
ir30 = .5; ir31 = .8;

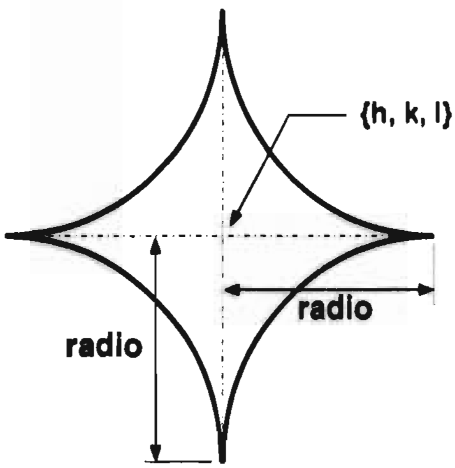
(* Se inicia un ciclo para resolver el problema cinemático inverso en los puntos en que será discretizará la trayectoria *)

For[t=0, t<=2*N[Pi], t+=N[Pi]/9,

	(* Se determinan las coordenadas del punto correspondiente de la trayectoria *) px = h; py = radio1*Cos[t] + k; pz = radio2*Sin[t] + l; (* Solución del problema cinemático inverso para el punto seleccionado *) solucion[i] = FindRoot[{rp1 == px, rp2 == py, rp3 == pz, r10^2+r13^2 == 1, r20^2+r21^2 == 1, r30^2+r31^2 == 1}, {r10,ir10},{r13,ir13},{r20,ir20}, {r21,ir21},{r30,ir30},{r31,ir31}, MaxIterations->50]; (* Se verifica la precisión de la solución *) e = 10^-5; (* precisión requerida *) If[Abs[px-(rp1/.solucion[i])] < e && Abs[py-(rp2/.solucion[i])] < e && Abs[pz-(rp3/.solucion[i])] < e, (* Si la solución está en la precisión fijada se definen como valores iniciales, para resolver la cinemática inversa del siguiente punto de la trayectoria, a los valores de la solución obtenida, si no es así se restauran los valores iniciales utilizados *) ir10=r10/.solucion[i]; ir13=r13/.solucion[i]; ir20=r20/.solucion[i]; ir21=r21/.solucion[i]; ir30=r30/.solucion[i]; ir31=r31/.solucion[i], (* Se imprime un mensaje si la solución no está en la precisión definida *) Print["No se encontró solución para el punto:", {px,py,pz}]; (* Se restauran valores iniciales para resolver las ecuaciones si no se encontró solución para el punto en cuestión *) ir10=.7071; ir13=.7071; ir20=.3; ir21=.9; ir30=.5; ir31=.8]; (* Se incremente la variable i y se cierra el ciclo For *) i++]

De la misma forma que se determinan los parámetros para las trayectorias anteriores, ahora se determinará una trayectoria en forma de ***astroride***, la cual se ubicará en el plano x-z, con centro en {h, k, l} cuya forma paramétrica es:

$px = radio*Cos^3(t) + h;$
$py = k;$
$pz = radio*Sin^3(t) + l;$

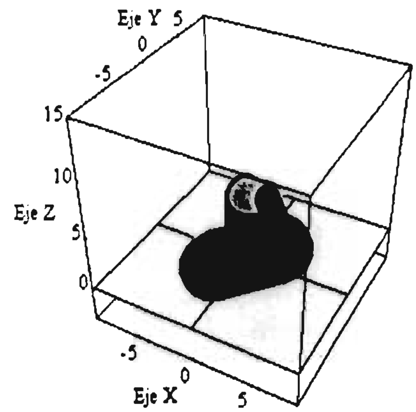
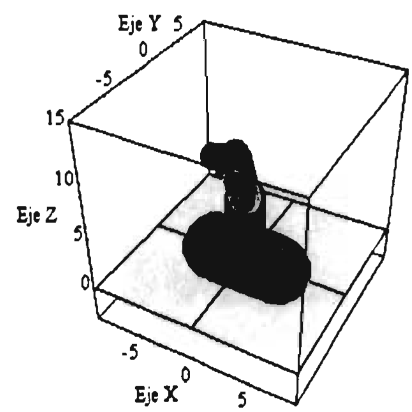


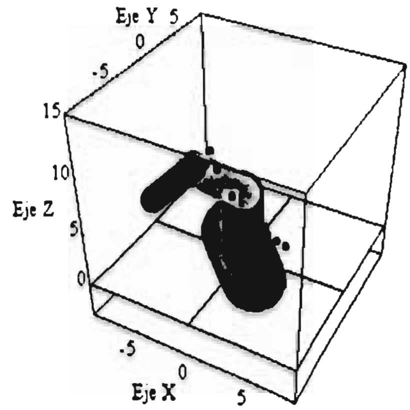
ASTROIDE

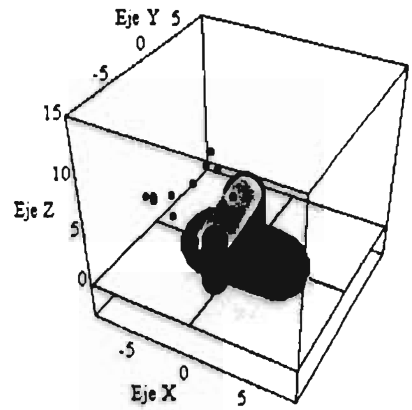
```
(* Se inicializan variables *)
ClearAll[r10, r13, r20, r21, r30, r31, "solp", "solq",
"solr", "solucion"];

(*Se recuperan las ecuaciones del manipulador*)
b1=z1; b2=z2; b3=z3; rp=rx; rp1=rp[[1]];
rp2=rp[[2]]; rp3=rp[[3]];

(* Variables de control *)
t = 0; (* Para discretizar en varios puntos la
        trayectoria seleccionada *)
i = 1; (* Para guardar la solución de cada punto
        de la trayectoria *)

(* Variable de control para definir la trayectoria *)

        radio = 6; (* Radio de la astroide *)
        h=0;k=-5;l=7; (* Centro de la elipse *)

(* Valores iniciales para resolver el sistema de
   ecuaciones *)

ir10 = .7071; ir13 = .7071; ir20 = .3; ir21 = .9;
ir30 = .5; ir31 = .8;

(* Se inicia un ciclo para resolver el problema
   cinemático inverso en los puntos en que será
   discretizada la trayectoria *)

For[t=0, t<=2*N[Pi], t+=N[Pi]/11,

(* Se determinan las coordenadas del punto
   correspondiente de la trayectoria *)
          px = radio*Cos[t]^3 + h;
          py = k;
          pz = radio*Sin[t]^3 + l;

(* Solución del problema cinemático inverso para
   el punto seleccionado *)

solucion[i] =
FindRoot[{rp1 == px, rp2 == py, rp3 == pz,
          r10^2+r13^2 == 1, r20^2+r21^2 == 1,
          r30^2+r31^2 == 1},
          {r10,ir10},{r13,ir13},{r20,ir20},
          {r21,ir21},{r30,ir30},{r31,ir31},
          MaxIterations->50];

(* Se verifica la precisión de la solución *)

e = 10^-5; (* precisión requerida *)
```

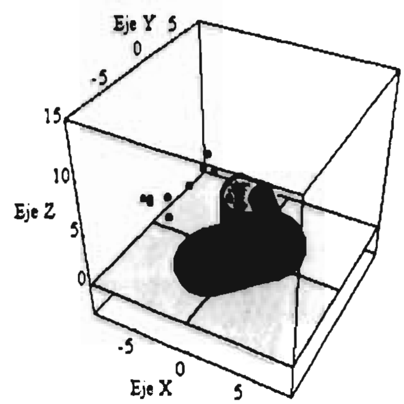
	`` If[Abs[px-(rp1/.solucion[i])] < e && Abs[py-(rp2/.solucion[i])] < e && Abs[pz-(rp3/.solucion[i])] < e, (* Si la solución está en la precisión fijada se definen como valores iniciales, para resolver la cinemática inversa del siguiente punto de la trayectoria, a los valores de la solución obtenida, si no es así se restauran los valores iniciales utilizados *) ir10=r10/.solucion[i]; ir13=r13/.solucion[i]; ir20=r20/.solucion[i]; ir21=r21/.solucion[i]; ir30=r30/.solucion[i]; ir31=r31/.solucion[i], (* Se imprime un mensaje si la solución no está en la precisión definida *) Print["No se encontró solución para el punto: ", {px,py,pz}]; (* Se restauran valores iniciales para resolver las ecuaciones si no se encontró solución para el punto en cuestión *) ir10=.7071; ir13=.7071; ir20=.3; ir21=.9; ir30=.5; ir31=.8]; (* Se incremente la variable i y se cierra el ciclo For *) i++] ```
Una vez determinados los parámetros de rotación para la trayectoria seleccionada, ahora se generarán las gráficas que permitirán ver la simulación de la cinemática inversa del manipulador de tres grados de libertad. Para poder realizarlo, recuperamos los valores de los parámetros almacenados en la variable solución[j] y los almacenamos en variables separadas para cada par de parámetros. Al igual que en la cinemática inversa, se genera una gráfica que servirá como base para graficar el manipulador, la cual se almacena en las dos variables g1 y g2, posteriormente se inicia un ciclo, que se repetirá tantas veces como valores de los parámetros se hayan calculado para la trayectoria seleccionada (variable "i"). En cada pasada del ciclo se asignan los valores correspondientes a los parámetros y con ellos se genera la gráfica del manipulador de la misma forma que se hizo en la cinemática directa. Se mostrarán, a manera	**Graficación para visualizar la simulación de la trayectoria seleccionada.** ``` ClearAll[r10,r13,r20,r21,r30,r31] (*Se recuperan los parámetros de rotación almacenados en la variable solucion[j], obtenidos de resolver el problema cinemático inverso y se almacenan en variables para cada par de parámetros *) For[j=1, j<i, j++, solr1[j]={r10,r13}/.solucion[j]; solr2[j]={r20,r21}/.solucion[j]; solr3[j]={r30,r31}/.solucion[j]] (* Se define el espacio tridimensional en que se graficará el manipulador, que la llamaremos gráfica base *) (* Suma de la longitud del eslabón 2 y 3 *) L23 = L2+L3; ```

de ejemplo, algunas gráficas de una trayectoria.	g1=g2=SurfaceGraphics[{{0,0,0},{0,0,0},{0,0,0}}, MeshRange->{{-(L23),L23},{-(L23),L23}}, AxesLabel->{Eje X,Eje Y,Eje Z}, DefaultFont->{"Negrita",12}, PlotRange->{{-(L23+.1),(L23+.1)}, {-(L23+.1),(L23+.1)}, {(L1-(L23)-.1),(L1+L23+.1)}}, BoxRatios->{1,1,1},Axes->True]; (* Se inicia un ciclo para generar las gráficas de cada punto de la trayectoria *) For[j=1, j<i, j++, (* Se asigna el valor de los parámetros para el punto de la trayectoria correspondiente almacenados en las variables solr?[j]*) r10=solr1[j][[1]]; r13=solr1[j][[2]]; r20=solr2[j][[1]]; r21=solr2[j][[2]]; r30=solr3[j][[1]]; r31=solr3[j][[2]]; (* Se asignan valores a las ecuaciones del manipulador, según los valores de los parámetros, utilizando el respaldo de ecuaciones *) base=N[a11]; b1=N[z1]; b2=N[z2]; b3=N[z3]; rp=N[rx]; (* Longitud del vector que servirá de base del manipulador *) a = 3; (* Se definen los puntos en el espacio tridimensional para graficar el manipulador *) punto1 = {0,0,0}; punto2 = N[a*base]; punto3 = N[-a*base]; punto4 = N[b1]; punto5 = N[punto4 + b2]; punto6 = N[punto5 + b3]; (* Se agrega a la gráfica base (g2) la posición del elemento terminal, como un punto *) g2={Graphics3D[{{PointSize[0.01], Point[rp]}}], g2}; (* Se genera la gráfica del manipulador en la posición correspondiente *)

	g3=Graphics3D[{ (* Base rotatoria *) {AbsoluteThickness[40],RGBColor[0,.5,0], Line[{punto2,punto3}]}, (* Eslabón 1 *) {AbsoluteThickness[25],RGBColor[0,1,0], Line[{punto1,punto4}]}, (* Eslabón 2 *) {AbsoluteThickness[20],RGBColor[1,0,0], Line[{punto4,punto5}]}, (* Eslabón 3 *) {AbsoluteThickness[15],RGBColor[0,0,1], Line[{punto5,punto6}]}}] ; (* Se envían a pantalla las gráficas g1,g2 y g3, combinadas *) Show[g1, g3, g2, Axes->True];]  


Eje Y
5
0
-5
15
10
Eje Z
5
0
-5
0
Eje X
5


Eje Y
5
0
-5
15
10
Eje Z
5
0
-5
0
Eje X
5


Eje Y
5
0
-5
15
10
Eje Z
5
0
-5
0
Eje X
5

4.2 Cinemática de un robot de cuatro grados de libertad.

En la siguiente figura se muestra la configuración del robot que se simulará en está sección, el cual fue previamente modelado en el capítulo anterior.

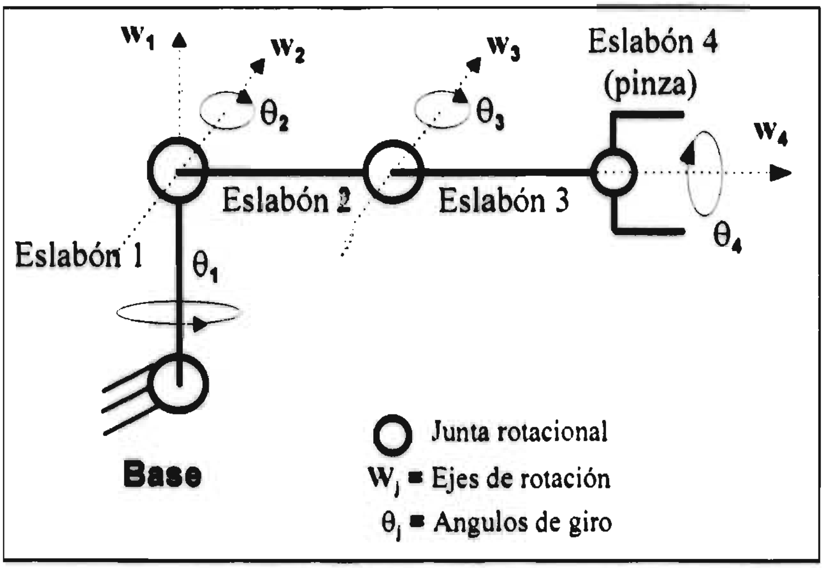


Figura 4.2 *Configuración del manipulador de cuatro grados de libertad*

4.2.1 Cinemática Directa.

El problema cinemático directo para este manipulador, consiste en que dados los desplazamientos angulares θ_1, θ_2, θ_3 y θ_4, se debe determinar la posición y orientación del elemento terminal, así como la configuración resultante de sus eslabones. Para realizar lo anterior hacemos uso de las ecuaciones (3.85), (3.86), (3.87) determinadas en el capitulo anterior, que al evaluarlas se resuelve la cinemática directa de este manipulador. Dichas ecuaciones se muestran a continuación.

$$
\begin{aligned}
b_1 &= \{0, 0, L_1\} \\
b_2 &= \{-4{*}A5{*}B2{*}L_2, -2{*}A6{*}B2{*}L_2, B1{*}L_2\} \qquad (4.12) \\
b_3 &= \{-4{*}A5{*}C1{*}L_3, -2{*}A6{*}C1{*}L_3, C2{*}L_3\} \\
b_4 &= \{-4{*}A5{*}C1{*}L_4, -2{*}A6{*}C1{*}L_4, C2{*}L_4\} \\
b_6 &= \{(A12{*}A6 - 4{*}A11{*}A5{*}C2){*}L_6, -2{*}(A12{*}A5 + A11{*}A6{*}C2){*}L_6, -4{*}A11{*}C1{*}L_6\};
\end{aligned}
$$

$$
\begin{aligned}
r_p = \{&-4{*}A5{*}(B2{*}L_2 + C1{*}(L_3+L_4)), \\
&-2{*}A6{*}(B2{*}L_2 + C1{*}(L_3+L_4)), L_1 + B1{*}L_2 + C2{*}(L_3+L_4)\} \qquad (4.13)
\end{aligned}
$$

Donde:

$$
\begin{aligned}
A1 &= P_0{*}P_1; & A2 &= 1-2{*}P_0^2; \\
A3 &= Q_0{*}Q_1; & A4 &= 1-2{*}Q_0^2; \\
A5 &= r1_0{*}r1_1; & A6 &= 1-2{*}r1_0^2; \\
A7 &= r2_0{*}r2_1; & A8 &= 1-2{*}r2_0^2; \qquad (4.14)
\end{aligned}
$$

$$A9 = r3_0 * r3_1; \quad A10 = 1\text{-}2*r3_0^2;$$
$$A11 = r4_0 * r4_3; \quad A12 = 1\text{-}2*r4_0^2;$$
$$B1 = A2*A8 - 4*A1*A7; \qquad B2 = A2*A7 + A1*A8;$$
$$B3 = A10*A8 - 4*A7*A9; \qquad B4 = A10*A7 + A8*A9;$$
$$B5 = A2*A4 - 4*A1*A3; \qquad B6 = A2*A3 + A1*A4;$$
$$C1 = B4*B5 + B3*B6; \qquad C2 = B3*B5 - 4*B4*B6;$$

Las variables de las ecuaciones anteriores tienen el siguiente significado:

$L_1, L_2, L_3, L_4, L_5 \rightarrow$ Longitud de los eslabones del manipulador.
$b_1, b_2, b_3, b_4, b_5 \rightarrow$ Eslabones del manipulador.
$r_p \rightarrow$ Posición del elemento terminal.
$P_0, P_1, Q_0, Q_1 \rightarrow$ Parámetros de rotación en la configuración no deformada.
$r1_0, r1_3, r2_0, r2_1, r3_0, r3_1, r4_0, r4_3 \rightarrow$ Parámetros de rotación en la configuración deformada.

Los parámetros de rotación tienen la siguiente relación con las rotaciones de los eslabones del manipulador.

$$P_0 = \text{Cos}\,(\varphi_1/2); \quad P_1 = \text{Sen}\,(\varphi_1/2);$$
$$Q_0 = \text{Cos}\,(\varphi_2/2); \quad Q_1 = \text{Sen}\,(\varphi_2/2); \qquad (4.15)$$

$$r1_0 = \text{Cos}\,(\theta_1/2); \quad r1_3 = \text{Sen}\,(\theta_1/2);$$
$$r2_0 = \text{Cos}\,(\theta_2/2); \quad r2_1 = \text{Sen}\,(\theta_2/2); \qquad (4.16)$$
$$r3_0 = \text{Cos}\,(\theta_3/2); \quad r3_1 = \text{Sen}\,(\theta_3/2);$$
$$r4_0 = \text{Cos}\,(\theta_4/2); \quad r4_3 = \text{Sen}\,(\theta_4/2);$$

Siendo:

φ_1 el ángulo de rotación del eslabón 2 en la configuración no deformada.
φ_2 el ángulo de rotación del eslabón 3 en la configuración no deformada.

θ_1 el ángulo de rotación del eslabón 1 en la configuración deformada.
θ_2 el ángulo de rotación del eslabón 2 en la configuración deformada.
θ_3 el ángulo de rotación del eslabón 3 en la configuración deformada.
θ_4 el ángulo de rotación de la pinza en la configuración deformada.

Antes de poder realizar el proceso de cinemática directa del manipulador, debemos determinar la configuración no deformada del robot mediante un proceso de cinemática inversa, que consiste en calcular los parámetros $P=\{P_0,P_1,0,0\}$ y $Q=\{Q_0,Q_1,0,0\}$ dada la posición del elemento terminal R_p, que en este caso es el extremo del eslabón 3 donde esta unida la pinza (Ecuaciones (3.71), (3.72)). Es decir, con las siguientes ecuaciones puede plantearse un sistema de ecuaciones, como se muestra.

$$R_p = \{0,\ 2*((A2*A3 + A1*A4)*L_3 - A1*L_2),\ L_1 - A2*L_2 + (A2*A4 - 4*A1*A3)*L_3\}; \qquad (4.17)$$

Donde:

$$A1 = P_0 * P_1; \quad A2 = 1 - 2*P_0^2; \qquad (4.18)$$
$$A3 = Q_0 * Q_1; \quad A4 = 1 - 2*Q_0^2;$$

Con la ecuación para R_p podemos plantear 2 ecuaciones si damos la posición para el elemento terminal como $R_p = \{0, R_py, R_pz\}$ y con las normas unitarias de los Quaterniones P y Q completamos un sistema de 4 ecuaciones con 4 incógnitas, que al resolverlo obtenemos los parámetros que determinan la configuración no deformada.

$$2*((A2*A3 + A1*A4)*L_3 - A1*L_2) = R_p y$$
$$L_1 - A2*L_2 + (A2*A4 - 4*A1*A3)*L_3 = R_p z \qquad (4.19)$$
$$P_0^2 + P_1^2 = 1$$
$$Q_0^2 + Q_1^2 = 1$$

Con lo anterior podemos ahora resolver el problema cinemático directo y realizar la correspondiente simulación computacional, que como se mencionó se realizará en Mathematica®. A continuación se presenta el programa desarrollado para este caso. Con el objeto de mostrar la forma de utilizar Mathematica® para obtener las ecuaciones del manipulador, tanto en su configuración no deformada como en la deformada.

En la primera parte del programa se define la transformación rotación de un vector tridimensional y de un Quaternión, y de algunas otras funciones necesarias que se muestran a continuación. Transformación de un Quaternión en un vector tridimensional. $q = \{q_0, q_1, q_2, q_3\} \in Q$ **Tv[q] = $\{q_1, q_2, q_3\}$.** Transformación vectorial inversa. $v = \{v_1, v_2, v_3\} \in \Re^3$ **TvInv[v] = $\{0, v_1, v_2, v_3\}$.** Transformación de un Quaternión en un Quaternión vectorial. **TvQ[q] = $\{0, q_1, q_2, q_3\}$.** Producto de dos Quaterniones. $p = \{p_0, p_1, p_2, p_3\}$, $q = \{q_0, q_1, q_2, q_3\} \in Q$ **ProductoQ[p,q] =** $\{p_0{*}q_0 - p_1{*}q_1 - p_2{*}q_2 - p_3{*}q_3,\ p_0{*}q_1 + p_1{*}q_0 + p_2{*}q_3 - p_3{*}q_2,\ p_0{*}q_2 - p_1{*}q_3 + p_2{*}q_0 + p_3{*}q_1,\ p_0{*}q_3 + p_1{*}q_2 - p_2{*}q_1 + p_3{*}q_0\}$ Conjugado de un Quaternión. **ConjugadoQ[p] = $\{p_0, -p_1, -p_2, -p_3\}$** Rotación de un vector $v \in \Re^3$ con un	**Definición de la Transformación Rotación.** (* Se apagan algunos mensajes del sistema *) Off[General::spell1] Off[General::spell] (* Se define la Transformación de un Quaternión a un vector tridimensional Tv[] *) Tv[q_]:={q[[2]],q[[3]],q[[4]]}; (* Se define la Transformación inversa de Tv[] *) TvInv[v_]:={0,v[[1]],v[[2]],v[[3]]}; (* Se define la Transformación de un Quaternión a Quaternión vectorial *) TvQ[q_]:={0,q[[2]],q[[3]],q[[4]]}; (* Se define la Multiplicación entre dos Quaterniones *) ProductoQ[p_,q_]:= {p[[1]]*q[[1]] - p[[2]]*q[[2]] - p[[3]]*q[[3]] - p[[4]]*q[[4]], p[[1]]*q[[2]] + p[[2]]*q[[1]] + p[[3]]*q[[4]] - p[[4]]*q[[3]], p[[1]]*q[[3]] - p[[2]]*q[[4]] + p[[3]]*q[[1]] + p[[4]]*q[[2]], p[[1]]*q[[4]] + p[[2]]*q[[3]] - p[[3]]*q[[2]] + p[[4]]*q[[1]]}; (* Se define el conjugado de un Quaternión *) ConjugadoQ[p_]:={p[[1]],-p[[2]],-p[[3]],-p[[4]]}; (* Se define la Transformación Rotación de un vector tridimensional *) Rotacion[p_,v_]:= Tv[ProductoQ[p, ProductoQ[TvInv[v], ConjugadoQ[p]]]];

Quaternión **p**, siendo $\lVert \mathbf{p} \rVert = 1$. Según la definición (2.13). $\rho(\mathbf{p},\mathbf{q}) = \frac{1}{\lVert \mathbf{p} \rVert^2} \bullet \left(\mathbf{p} * \mathbf{q} * \bar{\mathbf{p}}\right), \quad \forall \mathbf{q} \in \mathbf{Q}$ **Rotacion[p,v] = Tv[p * TvInv[v] * $\bar{\mathbf{p}}$]** Rotación de **q** $\in$ **Q** con un Quaternión **p**. siendo $\lVert \mathbf{p} \rVert = 1$. De acuerdo con la definición (2.13) y considerando que el primer elemento de **q** no cambia al rotar cuando $\lVert \mathbf{p} \rVert = 1$.. **RotacionQ[p,q] = {q_0,0,0,0} + TvQ[p*q* $\bar{\mathbf{p}}$]**	(* Se define la Transformación Rotación de un Quaternión *) RotacionQ[p_,q_]:={q[[1]],0,0,0}+ TvQ[ProductoQ[p,ProductoQ[q,ConjugadoQ[p]]]];
Se realizan definiciones iniciales necesarias, como son la base canónica de $\Re^3$ y las bases que servirán como bases locales en los eslabones. También se define la forma local de los Quaterniones que nos permitirán rotar, tanto en la posición no deformada como en la deformada.	**Definiciones iniciales** (* Base canónica *) e1 = {1,0,0}; e2 = {0,1,0}; e3 = {0,0,1}; (* Bases locales de los eslabones *) (* Eslabón 1 *) e1 = {1,0,0}; e2 = {0,1,0}; e3 = {0,0,1}; (* Eslabón 2 *) f1 = {1,0,0}; f2 = {0,1,0}; f3 = {0,0,1}; (* Eslabón 3 *) g1 = {1,0,0}; g2 = {0,1,0}; g3 = {0,0,1}; (* Eslabón 4 *) h1 = {1,0,0}; h2 = {0,1,0}; h3 = {0,0,1}; (*Quaterniones para rotar a posición no deformada*) P = {P0,P1,0,0}; (* Eslabón 2 *) Q = {Q0,Q1,0,0}; (* Eslabón 3 *) (* Quaterniones para rotar a posición deformada *) r1 = {r10, 0, 0, r13}; (* Eslabón 1 *) r2 = {r20, r21, 0, 0}; (* Eslabón 2 *) r3 = {r30, r31, 0, 0}; (* Eslabón 3 *) r4 = {r40, 0, 0, r43}; (* Eslabón 4 *)
En esta sección del programa se realiza la Modelación de la configuración no deformada del manipulador. Como la forma de las bases locales de los eslabones, en la configuración	**MODELADO DE LA CONFIGURACION NO DEFORMADA** **Modelo para la configuración no deformada**

<table>
<tr>
<td>
no deformada, se obtuvo en el capitulo anterior, entonces para facilitar la obtención del las ecuaciones del manipulador, simplemente obtenemos de manera directa la forma de las bases locales, que son:

$\mathbf{f}_j^{I} = \rho(\mathbf{P}, \mathbf{f}_j)$ (Eslabon 2)
$\mathbf{g}_j^{II} = \rho((\mathbf{P}^*\mathbf{Q}), \mathbf{g}_j)$ (Eslabon 3)
$\mathbf{h}_j^{II} = \rho((\mathbf{P}^*\mathbf{Q}), \mathbf{h}_j)$ (Pinza)

Para lo anterior es necesario definir las normas unitarias de los Quaterniones P y Q, las cuales se almacenan en las variables "normas1" y "normas2", para aplicarlas en las ecuaciones resultantes de las rotaciones.

Las variables que se utilizan para las bases locales de los eslabones son:

ej = Primer eslabón.
fj = Segundo eslabón.
gj = Tercer eslabón.
hj = Pinza. j = 1,2,3.
</td>
<td>
(* Se inicializan variables *)
ClearAll[P0,P1,Q0,Q1,L1,L2,L3,b1,b2,b3];

(* Se definen variables para las normas unitarias de los Quaterniones P y Q *)
normas1= {P1->(1-P0^2)^(1/2),Q1->(1-Q0^2)^(1/2)};
normas2= {(1-P0^2)^(1/2)->P1,(1-Q0^2)^(1/2)->Q1};

(* Se obtienen los productos de Quaterniones necesarios para rotar las bases *)

PQ = ProductoQ[P,Q];

(* Se obtiene la forma de las bases locales de los eslabones en la configuración no deformada *)

(* Eslabón 1 *)
e1 = {1,0,0}; e2 = {0,1,0}; e3 = {0,0,1};

(* Eslabón 2 *)
f11 = Expand[Expand[Rotacion[P,f1]]//.normas1] //.normas2;
f21 = Expand[Expand[Rotacion[P,f2]]//.normas1] //.normas2;
f31 = Expand[Expand[Rotacion[P,f3]]//.normas1] //.normas2;

(* Eslabón 3 *)
g12 = Expand[Expand[Rotacion[PQ,g1]]//.normas1] //.normas2;
g22 = Expand[Expand[Rotacion[PQ,g2]]//.normas1] //.normas2;
g32 = Expand[Expand[Rotacion[PQ,g3]]//.normas1] //.normas2;

(* Eslabón 4 *)
h12 = Expand[Expand[Rotacion[PQ,h1]]//.normas1] //.normas2;
h22 = Expand[Expand[Rotacion[PQ,h2]]//.normas1] //.normas2;
h32 = Expand[Expand[Rotacion[PQ,h3]]//.normas1] //.normas2;
</td>
</tr>
<tr>
<td>
Los eslabones del manipulador y la posición del elemento terminal son identificados, en la posición no deformada, por las variables siguientes:

b1 = Eslabón 1.
b2 = Eslabón 2.
b3 = Eslabón 3.
b4 = Eslabón 4.
b5 = Eslabón 4.
</td>
<td>
(* Vectores que definen los eslabones del manipulador *)

b1 = L1*e3; (* Eslabón 1 *)
b2 = L2*f31; (* Eslabón 2 *)
b3 = L3*g32; (* Eslabón 3 *)
b4 = L4*h32; (* Eslabón 4 (pinza)*)
b5 = L5*h12; (* Eslabón 4 (pinza)*)

(* Constantes que se repiten y que servirán para simplificar las ecuaciones y optimizar operaciones *)
</td>
</tr>
</table>

Rp = b1+b2+b3; Rp = Posición del elemento terminal. Las ecuaciones se simplifican utilizando las siguientes constantes.. A1 = P0*P1; A2 = 1 - 2*P0^2; A3 = Q0*Q1; A4 = 1 - 2*Q0^2;	constantes = {P0*P1->A1, P0^2->(1-A2)/2, Q0*Q1->A3, Q0^2->(1-A4)/2}; b2 = b2//.constantes; b3 = Expand[Expand[b3//.constantes]//.constantes]; b4 = Expand[Expand[b4//.constantes]//.constantes]; (*Vector que define posición del elemento terminal *) Rp = b1+b2+b3;
Es esta parte se muestran las ecuaciones obtenidas en la sección anterior, las cuales se utilizarán para determinar los parámetros P y Q, que definen la configuración no deformada en una posición dada del elemento terminal, que será de donde arranque la simulación de la cinemática directa o inversa del manipulador en cuestión. Es decir, antes de generar una simulación computacional de la cinemática directa o inversa del manipulador, debemos determinar la configuración no deformada realizando un proceso de cinemática inversa para calcular P y Q para una posición conocida del elemento terminal, la cual debe tener la forma siguiente: Rp = {0,Rpy, Rpz}.	**Ecuaciones optimizadas para la posición no deformada.** ClearAll[P0,P1,Q0,Q1] (* Ecuaciones optimizadas *) b1 = {0, 0, L1}; (* Eslabón 1 *) b2 = {0, -2*A1*L2, -A2*L2}; (* Eslabón 2 *) b3 = {0, 2*(A2*A3 + A1*A4)*L3, (* Eslabón 3 *) (A2*A4 - 4*A1*A3)*L3}; (* Pinza *) b4 = {0, 2*(A2*A3+A1*A4)*L4, (A2*A4-4*A1*A3)*L4}; b5 = {L5, 0, 0}; (* Posición el elemento terminal *) Rp = {0, 2*((A2*A3 + A1*A4)*L3 - A1*L2), L1 - A2*L2 + (A2*A4 - 4*A1*A3)*L3}; (* Constantes de las ecuaciones *) A1 = P0*P1; A2 = 1 - 2*P0^2; A3 = Q0*Q1; A4 = 1 - 2*Q0^2; (* Longitud de los eslabones del manipulador *) L1 = 6; L2 = 5; L3 = 4;
Se resuelve el sistema de ecuaciones mostrado en (4.19), para P0,P1,Q0,Q1, utilizando la función FindRoot, cuya sintaxis se puede consulta en el Apéndice, al igual que todas las funciones que se utilizan en este trabajo. Los resultados obtenidos se redondean si alguno de ellos está muy cercano a cero, finalmente se almacenan en las variables correspondientes (P0, P1, Q0, Q1) para poder ser utilizados posteriormente, además,	**Cinemática Inversa para determinar la Configuración no Deformada** (* Conociendo la posición del elemento terminal se determinan los Quaterniones "P" y "Q" correspondientes *) ClearAll[P0,P1,Q0,Q1] (* Se da la posición del elemento terminal *)

se imprimen dichos valores.

```
posicion = {0, 0, 15};

(* Se determinan "P" y "Q" utilizando los siguiente
   valores iniciales *)

iP0 = .7071; iP1 = .7071; iQ0 = .866; iQ1 = .5;

(* Se almacena la solución en la variable sol y se
   utiliza la función FindRoot para determinar la
   solución del sistema de ecuaciones *)

sol =
FindRoot[{Rp2 == posicion[[2]],Rp3 == posicion[[3]],
          P0^2+P1^2 == 1, Q0^2+Q1^2 == 1},
          {P0,iP0}, {P1,iP1}, {Q0,iQ0}, {Q1,iQ1},
          MaxIterations->50];

(* Se asignan los valores de la solución a las
   variables correspondientes  *)

P0=P0/.sol;  P1=P1/.sol;  Q0=Q0/.sol;  Q1=Q1/.sol;

(* Se redondean los valores de la solución *)

If[Abs[P0]==1,P1=0,If[Abs[P1]==1,P0=0]];
 If[Chop[Abs[Q0]-1]==0,Q0=1;Q1=0,
 If[Chop[Abs[Q1]-1]==0,Q0=0;Q1=1]];

(* Se imprimen los resultados *)

Print["P0 = ",P0 ,"   ", "P1 = ", P1, "   Q0 = ",
       Q0, "   ", "Q1 = ", Q1]
```

De la misma forma que se realizó la modelación de la configuración no deformada, ahora se presenta el cálculo de la forma final de las bases locales de los eslabones, las cuales se obtuvieron en el capítulo anterior y son:

$$e_j^{I} = \rho(r1,\ e_j)$$

$$f_j^{III} = \rho(r1*r2*P,\ f_j)$$

$$g_j^{V} = \rho(r1*r2*r3*P*Q,\ g_j)$$

$$h_j^{VI} = \rho(r1*r2*r3*r4*P*Q,\ h_j)$$

También en este caso, se definen las normas unitarias de los Quaterniones utilizados, almacenándolas en las variables "normas3" y "normas4", que incluyen las normas unitarias de los Quaterniones correspondientes a la configuración no

MODELADO DE LA CONFIGURACION DEFORMADA

Modelo para la configuración deformada

```
ClearAll[r10,r13,r20,r21,r30,r31,L1,L2,L3];

(* Posición no deformada conocida *)
       P = {P0,P1,0,0}; Q = {Q0,Q1,0,0};

(* Quaterniones para rotar a la configuración
   deformada *)
       r1 = {r10,0,0,r13};  (* Rotación Eslabón 1*)
       r2 = {r20,r21,0,0};  (* Rotación Eslabón 2*)
       r3 = {r30,r31,0,0};  (* Rotación Eslabón 3*)
       r4 = {r40,0,0,r43};  (* Rotación Eslabón 4*)

(* Normas unitarias de los Quaterniones *)
normas3 = Flatten[{normas1,
     r13->(1-r10^2)^(1/2), r21->(1-r20^2)^(1/2),
     r31->(1-r30^2)^(1/2), r43->(1-r40^2)^(1/2)}];
```

deformada.	normas4 = Flatten[{normas2, (1-r10^2)^(1/2)->r13, (1-r20^2)^(1/2)->r21, (1-r30^2)^(1/2)->r31, (1-r40^2)^(1/2)->r43}]; (* Se obtienen los productos de Quaterniones necesarios para rotar las bases *) r1r2P = ProductoQ[r1,ProductoQ[r2,P]]; r1r2r3PQ = ProductoQ[r1,ProductoQ[r2, ProductoQ[r3,ProductoQ[P,Q]]]]; r1r2r3PQr4 = ProductoQ[r1,ProductoQ[r2, ProductoQ[r3,ProductoQ[P, ProductoQ[Q,r4]]]]]; (* Se obtiene la forma final de las bases locales de los eslabones *) (* Base local del eslabón 1 *) e11 = Expand[Expand[Rotacion[r1,e1]]//.normas3] //.normas4; e21 = Expand[Expand[Rotacion[r1,e2]]//.normas3] //.normas4; e31 = Expand[Expand[Rotacion[r1,e3]]//.normas3] //.normas4; (* Base local del eslabón 2 *) f13 = Expand[Expand[Rotacion[r1r2P,f1]]//.normas3] //.normas4; f23 = Expand[Expand[Rotacion[r1r2P,f2]]//.normas3] //.normas4; f33 = Expand[Expand[Rotacion[r1r2P,f3]]//.normas3] //.normas4; (* Base local del eslabón 3 *) g15 = Expand[Expand[Rotacion[r1r2r3PQ,g1]] //.normas3]//.normas4; g25 = Expand[Expand[Rotacion[r1r2r3PQ,g2]] //.normas3]//.normas4; g35 = Expand[Expand[Rotacion[r1r2r3PQ,g3]] //.normas3]//.normas4; (* Base local del eslabón 4 *) h16 = Expand[Expand[Rotacion[r1r2r3PQr4,h1]] //.normas3]//.normas4; h26 = Expand[Expand[Rotacion[r1r2r3PQr4,h2]] //.normas3]//.normas4; h36 = Expand[Expand[Rotacion[r1r2r3PQr4,h3]] //.normas3]//.normas4;

<table>
<tr>
<td>Una vez obtenidas las bases locales en su forma final, se obtienen las ecuaciones que representan los eslabones del manipulador y la posición del elemento terminal como:

$b_1 = L_1 * e_3^{I} = L_1$*e31; (Eslabón 1)
$b_2 = L_2 * f_3^{III} = L_2$*f33; (Eslabón 2)
$b_3 = L_3 * g_3^{V} = L_3$*g35; (Eslabón 3)
$b_4 = L_4 * h_3^{VI} = L_4$*h36; (Pinza)
$b_5 = L_5 * h_1^{VI} = L_5$*h16; (Pinza)

(Posición del elemento terminal)

$r_p = b_1 + b_2 + b_3 + b_4$

Estas expresiones son simplificadas sustituyendo por constantes a aquellos términos que se repiten dentro de las diferentes ecuaciones, como se puede observar.</td>
<td>(* Vectores que definen los eslabones del manipulador *)
b1 = L1*e31; (* Eslabón 1 *)
b2 = L2*f33; (* Eslabón 2 *)
b3 = L3*g35; (* Eslabón 3 *)
b4 = L4*h36; (* Eslabón 4 (pinza)*)
b5 = L5*h16; (* Eslabón 4 (pinza)*)

(* Constantes que se repiten y que servirán para simplificar las ecuaciones y optimizar operaciones *)

constantes = {P0*P1->A1, P0^2->(1-A2)/2,
Q0*Q1->A3, Q0^2->(1-A4)/2,
r10*r13->A5, r10^2->(1-A6)/2,
r20*r21->A7, r20^2->(1-A8)/2,
r30*r31->A9, r30^2->(1-A10)/2,
r40*r43->A11,r40^2->(1-A12)/2,
A2*A8->4*A1*A7+B1, A2*A7->B2-A1*A8,
A10*A8->4*A7*A9+B3, A10*A7->B4-A8*A9,
A2*A4->4*A1*A3+B5, A2*A3->B6-A1*A4,
A10*A4->4*A3*A9-B7, A4*A9->B8-A10*A3,
B4*B5->C1 - B3*B6, B3*B5->C2+4*B4*B6};

b2 = Expand[b2//.constantes];
b2 = Expand[Collect[b2,{L2,A5,A6}]//.constantes];

b3 = Expand[b3//.constantes];
b3 = Collect[Collect[Collect[b3,{L3,A5,A1,A2,A3,A4,
A6}]//.constantes, {L3,A5,A6,B3,B4}]
//.constantes,{L3,A5,A6,B3,B4,B5,B6}];
b3 = Expand[b3//.constantes];

b4 = Expand[b4//.constantes];
b4 = Collect[Collect[Collect[b4,{L4,A5,A1,A2,A3,A4,
A6}]//.constantes, {L4,A5,A6,B3,B4}]
//.constantes,{L4,A5,A6,B3,B4,B5,B6}];
b4 = Expand[b4//.constantes];

b5 = Expand[b5//.constantes];
b5 = Collect[Collect[Collect[b5,{L4,A5,A1,A2,A3,A4,
A6}]//.constantes, {L4,A5,A6,B3,B4}]
//.constantes, {L4,A5,A6,B3, B4,B5,B6}];
b5 = Expand[b5//.constantes];

(*Vector que define posición del elemento terminal *)

rp = Factor[b1 + b2 + b3 + b4];</td>
</tr>
<tr>
<td>En esta parte se presenta el resultado obtenido en la sección anterior, es decir, las ecuaciones simplificadas del manipulador, así como, las constantes utilizadas en dichas ecuaciones. Esta sección es la que propiamente se utiliza para realizar la</td>
<td>Ecuaciones optimizadas para la posición deformada (Secuencia 3-2-1)

ClearAll[r10, r13, r20, r21, r30, r31, r40, r43, L1, L2, L3, L4, L5];</td>
</tr>
</table>

simulación de la cinemática directa del manipulador de tres grados de libertad, ya que es la que contiene las ecuaciones ya simplificadas. Puede observarse, que además de las ecuaciones de los eslabones y de la posición del elemento terminal, se tiene una ecuación para **e11** = e_1^1; ésta sirve para mostrar el movimiento del primer eslabón, que al girar sobre su propio eje, requiere que se dibuje una de las bases perpendiculares al él para ver claramente su movimiento.
Como al realizar la simulación es necesario evaluar numéricamente las ecuaciones del manipulador, se genera un respaldo de dichas ecuaciones en esta misma sección, almacenándolas en nuevas variables.

```
(* Ecuaciones Simplificadas *)

b1 = {0, 0, L1};
b2 = {-4*A5*B2*L2, -2*A6*B2*L2, B1*L2};
b3 = {-4*A5*C1*L3, -2*A6*C1*L3, C2*L3};

(* Pinza *)

b4 = {-4*A5*C1*L4, -2*A6*C1*L4, C2*L4};
b5 = {(A12*A6 - 4*A11*A5*C2)*L5,
     -2*(A12*A5 + A11*A6*C2)*L5, -4*A11*C1*L5};

(* Posición del elemento terminal *)

rp = {-4*A5*(B2*L2 + C1*(L3+L4)),
     - 2*A6*(B2*L2 + C1*(L3+L4)),
       L1 + B1*L2 + C2*(L3+L4)};

(* Componente 1 de la Base local del eslabón 1
   (para dibujar la base) *)

e11 = {-A6, 2*A5, 0};

(* Constantes de las ecuaciones *)

 A1 = P0*P1;    A2 = 1-2*P0^2;
 A3 = Q0*Q1;    A4 = 1-2*Q0^2;
 A5 = r10*r13;  A6 = 1-2*r10^2;
 A7 = r20*r21;  A8 = 1-2*r20^2;
 A9 = r30*r31;  A10 = 1-2*r30^2;
A11 = r40*r43;  A12 = 1-2*r40^2;
B1 = A2*A8 - 4*A1*A7;  B2 = A2*A7 + A1*A8;
B3 = A10*A8 - 4*A7*A9;  B4 = A10*A7 + A8*A9;
B5 = A2*A4 - 4*A1*A3;  B6 = A2*A3 + A1*A4;
C1 = B4*B5 + B3*B6;    C2 = B3*B5 - 4*B4*B6;

(* Longitud de los eslabones del manipulador *)

        L1 = 6; L2 = 5; L3 = 4; L4 = 2;  L5 = 1.5;

(* Respaldo de ecuaciones *)

z1=b1; z2=b2; z3=b3; z4=b4; z5=b5; rx=rp;
```

Finalmente, en esta parte del programa, se presenta la generación de las gráficas que nos permitirán observar la simulación de la cinemática directa del manipulador en cuestión.

Para lograr generar las gráficas adecuadas, para poder observar la simulación, primero generamos mediante la función ***SurfaceGrafics*** una gráfica que contenga el

CINEMATICA DIRECTA

Graficación para visualizar la simulación.

```
ClearAll[r10,r13,r20,r21,r30,r31];

(* Se define el espacio tridimensional en que se
   graficará el manipulador, que se usará como
   gráfica base *)
```

espacio donde se mostrará el manipulador con una superficie simulando el piso y con la indicación de los ejes coordenados, la cual se almacena inicialmente en las variables **g1** y **g2**.

Definimos los parámetros de rotación de cada eslabón, en función de los ángulos que utilizaremos para girar cada uno de los eslabones, que serán:

Eslabón 1 → **theta1**
Eslabón 2 → **theta2**
Eslabón 3 → **theta3**

Se generan dos ciclos ***For***, el primero (variable de control **i**) servirá para controlar cual eslabón deberá moverse, o sea, cuándo variar que ángulo, lo cual se logra utilizando la función ***Switch***, y el segundo ciclo (variable de control **j**) nos permitirá generar varias gráficas del movimiento de cada eslabón. La variable **i** toma valores de 1 a 3, ya que son tres eslabones y la variable **j** toma valores de 1 a n, donde n es el número de gráficas a generar por cada eslabón.

Una vez que los parámetros de rotación tienen un valor numérico, se evaluan numéricamente las ecuaciones del manipulador, haciendo uso del respaldo de ecuaciones realizado anteriormente

Con las ecuaciones, ya numéricas, se modifica la gráfica almacenada en **g2** agregándole un punto en la posición que corresponde a **rp**.

De la misma forma, con las ecuaciones numéricas, se determinan los diferentes puntos de la gráfica a generar, es decir, se genera un punto para cada inicio y final de cada eslabón, además de generar puntos para mostrar el vector **e11** de la base del primer eslabón.

Con los puntos anteriores, se genera la gráfica correspondiente a la posición del manipulador, utilizando para ello la función ***Graphics3D***, que nos permite trazar líneas de diferente color y espesor entre los puntos determinados.

```
(* Suma de la longitudes de los eslabones 2,3 y 4 *)
          L234 = L2 + L3 + L4;

g1=g2=SurfaceGraphics[{{0,0,0},{0,0,0},{0,0,0}},
   MeshRange->{{-L234,L234},{-L234,L234}},
   AxesLabel->{Eje X,Eje Y,Eje Z},
          DefaultFont->{"Negrita",12},
   PlotRange->{{-(L234+.1),(L234+.1)},
          {-(L234+.1),(L234+.1)},
          {(L1-L234-.1),(L1+L234+.1)}},
   BoxRatios->{1,1,1},Axes->True];

(* Se inicializan en cero los ángulos de rotación de
   los cuatro eslabones *)

   theta1 = 0; theta2 = 0; theta3 = 0; theta4 = 0;

(* Se definen los valores de los parámetros de las
   rotaciones *)

          r10=Cos[theta1/2*Degree];
          r13=Sin[theta1/2*Degree];
          r20=Cos[theta2/2*Degree];
          r21=Sin[theta2/2*Degree];
          r30=Cos[theta3/2*Degree];
          r31=Sin[theta3/2*Degree];
          r40=Cos[theta4/2*Degree];
          r43=Sin[theta4/2*Degree];

(* Se inicializa un ciclo para mover los cuatro
   eslabones, uno a la vez *)

For[i=1, i<=4, i++,

(* Se inicializa otro ciclo para generar varias gráficas
   para cada eslabón en diferentes posiciones*)

For[j=1, j<=5, j++,

(* Se selecciona cual eslabón se moverá según el
   valor de la variable i, lo cual se logra cambiando
   sus parámetros de rotación *)

Switch[i,3,  (* Se mueve primer eslabón *)
          r10=N[Cos[theta1/2 Degree];
          r13=N[Sin[theta1/2 Degree];theta1+=15,
       2,  (* Se mueve segundo eslabón *)
          theta2+=15;
          r20=N[Cos[theta2/2 Degree];
          r21=N[Sin[theta2/2 Degree]; ,
       1,  (* Se mueve tercer eslabón *)
          theta3+=15;
          r30=N[Cos[theta3/2 Degree];
          r31=N[Sin[theta3/2 Degree]; ,
```

<table>
<tr>
<td>Finalmente se muestran en la pantalla, de manera combinada, las gráficas g1, g2 y g3 que se generan en cada pasada de los ciclos, utilizando para esto la función Show la cual envía a la pantalla objetos gráficos y permite que se combinen varias gráficas.

Se muestran, a manera de ejemplo, algunas gráficas de un proceso.</td>
<td><pre>
    4, (* Se mueve cuarto eslabón *)
        theta4+=25;
        r40=N[Cos[theta4/2 Degree];
        r43=N[Sin[theta4/2 Degree]];

(* Se asignan valores a las ecuaciones del
 manipulador, según los valores de los parámetros,
 utilizando el respaldo de ecuaciones *)

base=N[e11]; b1=N[z1]; b2=N[z2]; b3=N[z3];
b4=N[z4]; b4=N[z4]; rp=N[rx];

(* Longitud del vector que servirá de base del
manipulador *)
                    a=3;

(* Se definen los puntos en el espacio tridimensional
   para graficar el manipulador *)

punto1 = {0,0,0};
punto2 = N[a*base];
punto3 = N[-a*base];
punto4 = N[b1];
punto5 = N[punto4 + b2];
punto6 = N[punto5 + b3];
punto7 = N[punto6 + b5];
punto8 = N[punto6 - b5];
punto9 = N[punto7 + b4];
punto10 = N[punto8 + b4];

(* Se agrega a la gráfica g2 la posición del elemento
   terminal como un punto *)

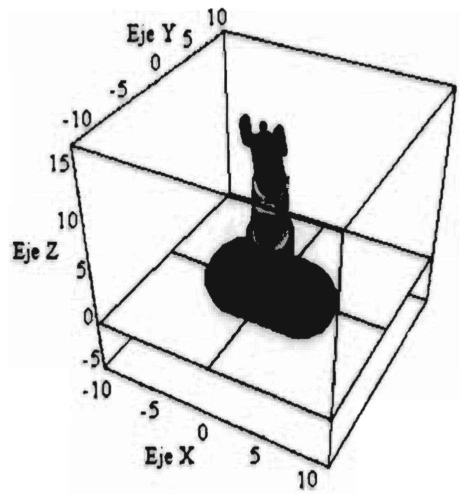
g2={Graphics3D[{PointSize[0.05],
     Point[rp], RGBColor[0,1,0]}],g2};

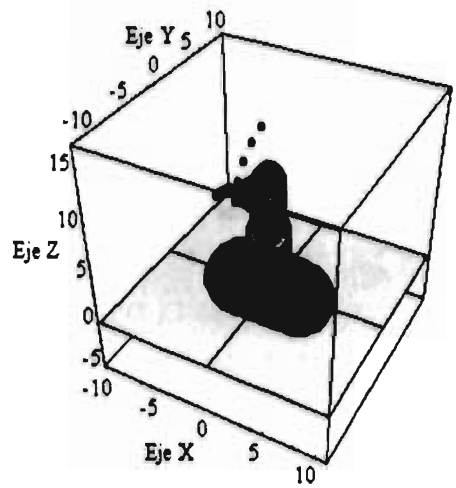
(* Se genera la gráfica del manipulador en la
   posición correspondiente *)

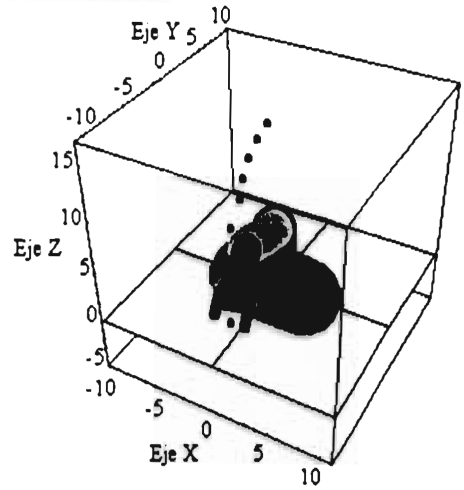
g3=Graphics3D[{

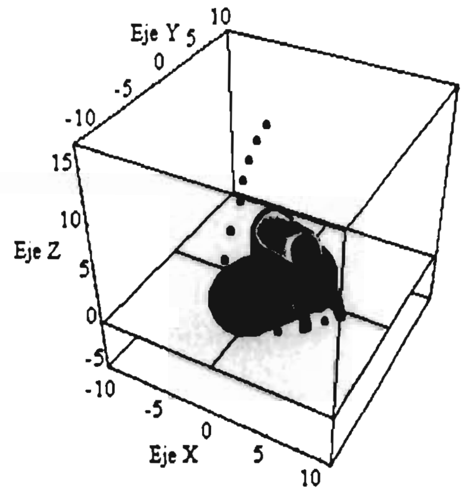
(* Base rotatoria *)
        {AbsoluteThickness[40],RGBColor[1,.3,0],
         Line[{punto2,punto3}]},
(* Eslabón 1 *)
        {AbsoluteThickness[25],RGBColor[0,1,0],
         Line[{punto1,punto4}]},
(* Eslabón 2 *)
        {AbsoluteThickness[20],RGBColor[1,.4,0],
         Line[{punto4,punto5}]},
(* Eslabón 3 *)
        {AbsoluteThickness[15],RGBColor[0,0.5,1],
         Line[{punto5,punto6}]},
</pre></td>
</tr>
</table>

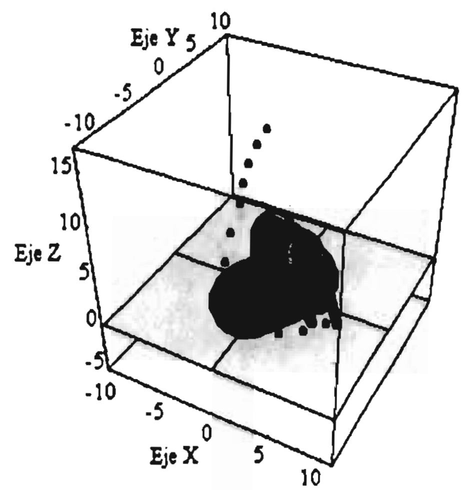
```
(* Pinza *)
        {AbsoluteThickness[5],RGBColor[0,0,0],
         Line[{punto6,punto7}]},
        {AbsoluteThickness[5],RGBColor[0,0,0],
         Line[{punto6,punto8}]},
        {AbsoluteThickness[5],RGBColor[0,0,0],
         Line[{punto7,punto9}]},
        {AbsoluteThickness[5],RGBColor[0,0,0],
         Line[{punto8,punto10}]}}];

(* Se envían a pantalla las gráficas g1,g2 y g3,
   combinadas *)

Show[g1,g3,g2,Axes->True];
```






Eje Y
Eje Z
Eje X
10
5
0
-5
-10
15
10
5
0
-5
-10
-5
0
5
10


Eje Y
Eje Z
Eje X
10
5
0
-5
-10
15
10
5
0
-5
-10
-5
0
5
10


Eje Y
Eje Z
Eje X
10
5
0
-5
-10
15
10
5
0
-5
-10
-5
0
5
10

4.2.2 Cinemática Inversa.

La cinemática inversa del robot de cuatro grados de libertad, como se mencionó anteriormente, consiste en que, dada la posición y orientación del elemento terminal, se deben determinar los ángulos que deberán desplazarse cada una de las juntas del manipulador para alcanzar la posición deseada. En este caso, consiste en que dada la configuración no deformada (P_0, P_1, Q_0, Q_1,) y la posición del elemento terminal (r_p = {r_px, r_py, r_pz}), se deben determinar los parámetros $r4_0$ y $r4_3$ que orienten el elemento terminal y los parámetros $r1_0$, $r1_3$, $r2_0$, $r2_1$, $r3_0$, $r3_1$ que satisfagan la ecuación de la posición del elemento terminal, que es la siguiente:

$$\begin{aligned} r_p = \{&- 4*A5*(B2*L_2 + C1*(L_3+L_4)),\\ &- 2*A6*(B2*L_2 + C1*(L_3+L_4)),\ L_1 + B1*L_2 + C2*(L_3+L_4)\} \end{aligned} \quad (4.20)$$

Para resolver la cinemática inversa del robot de cuatro grados de libertad, debemos plantear un sistema de 6 ecuaciones que nos permitan determinar los 6 parámetros de rotación que permitan ubicar el elemento terminal. De la ecuación de r_p obtenemos 3 ecuaciones y con las normas unitarias de los Quaterniones se completa el sistema, el cual se muestra a continuación.

$$\begin{aligned} r_px &= - 4*A5*(B2*L_2 + C1*(L_3+L_4))\\ r_py &= - 2*A6*(B2*L_2 + C1*(L_3+L_4))\\ r_pz &= L_1 + B1*L_2 + C2*(L_3+L_4)\\ r1_0^2 &+ r1_3^2 = 1\\ r2_0^2 &+ r2_1^2 = 1\\ r3_0^2 &+ r3_1^2 = 1 \end{aligned} \quad (4.21)$$

Siendo:

$$\begin{array}{ll} A1 = P_0*P_1; & A2 = 1-2*P_0^2;\\ A3 = Q_0*Q_1; & A4 = 1-2*Q_0^2;\\ A5 = r1_0*r1_3; & A6 = 1-2*r1_0^2;\\ A7 = r2_0*r2_1; & A8 = 1-2*r2_0^2;\\ A9 = r3_0*r3_1; & A10 = 1-2*r3_0^2;\\ B1 = A2*A8 - 4*A1*A7; & B2 = A2*A7 + A1*A8;\\ B3 = A10*A8 - 4*A7*A9; & B4 = A10*A7 + A8*A9;\\ B5 = A2*A4 - 4*A1*A3; & B6 = A2*A3 + A1*A4;\\ C1 = B4*B5 + B3*B6; & C2 = B3*B5 - 4*B4*B6; \end{array} \quad (4.22)$$

Lo anterior nos permite ubicar el elemento terminal en el punto {r_px, r_py, r_pz} dentro de su espacio de trabajo, ahora si deseamos que el elemento terminal siga una trayectoria determinada, una de las cosas que podemos hacer es discretizar dicha trayectoria en varios puntos a lo largo de ella y determinar los parámetros de rotación de los eslabones del manipulador para cada punto, resolviendo el sistema de ecuaciones (4.21), lo que nos permitirá ubicar el elemento terminal sobre dicha secuencia de puntos. Lo anterior no permite ubicar el elemento terminal en cualquier punto de la trayectoria, solo en los puntos en que dicha trayectoria se discretizó.

Para determinar la orientación del elemento terminal ($r4_0$ y $r4_3$), debemos considerar el hecho de que, cuando la pinza esta en la posición inicial (b_5 paralelo al eje x), su orientación con respecto al plano de la base del manipulador (plano x-y) no cambia con los movimientos del los eslabones. Sabiendo lo anterior, podemos orientar la pinza con respecto al plano x-y, en cualquier posición de los eslabones, con una rotación fija, pero si se desea orientarlo con respecto a otra referencia, deberá determinarse el ángulo formado entre el vector b_5 y una línea perpendicular a b_4. A continuación se presenta el programa desarrollado en Mathematica® para la cinemática inversa del manipulador de cuatro grados de libertad, el cual es continuación del anterior donde se presenta la cinemática directa.

<table>
<tr>
<td>Este programa es parte integral del anterior, es decir, para que funcione deberá tenerse la parte anterior donde se presenta modelación y la cinemática directa, ya que se requieren las ecuaciones del manipulador. En esta sección se presenta el cálculo para tres diferentes trayectorias y desde luego la graficación para observar la simulación.</td>
<td>CINEMATICA INVERSA

Solución de ecuaciones para diferentes trayectorias</td>
</tr>
<tr>
<td>En esta parte se determinan los parámetros de rotación de los eslabones, para una sucesión de puntos que pertenecen a una línea recta, la cual se define por sus dos puntos extremos (S0, S1). Para lo anterior, primeramente recuperamos las ecuaciones del manipulador, las cuales se respaldaron en otras variables, (deberá resolverse la cinemática inversa para determinar la configuración no deformada y ejecutar la celda donde están las ecuaciones optimizadas para la configuración deformada).

Para determinar la puntos sobre la trayectoria, utilizamos la forma paramétrica de una recta que pasa por dos puntos $p_1 = \{x_1, y_1, z_1\}$ y $p_2 = \{x_1, y_1, z_1\}$:

$$px = x_1 + (x_2 - x_1)t$$
$$py = y_1 + (y_2 - y_1)t$$
$$pz = z_1 + (z_2 - z_1)t$$

Con lo anterior se realiza un ciclo utilizando como variable de control del mismo, a la variable t, con lo que determinamos puntos sobre la trayectoria a seguir y resolvemos el sistema de ecuaciones (4.21).

$$r_px = - 4^*A5^*(B2^*L_2 + C1^*L_3),$$
$$r_py = - 2^*A6^*(B2^*L_2 + C1^*L_3),$$
$$r_pz = L_1 + B1^*L_2 + C2^*L_3\}$$
$$r1_0^2 + r1_3^2 = 1$$
$$r2_0^2 + r2_1^2 = 1$$
$$r3_0^2 + r3_1^2 = 1$$
Este sistema de ecuaciones se resuelve utilizando la función Findroot usando los</td>
<td>LINEA RECTA

(* Se inicializan variables *)
ClearAll[r10,r13,r20,r21,r30,r31,r40,r43,"solr1", "solr2","solr3","solr4", "solucion"];

(*Se recuperan las ecuaciones del manipulador*)
b1=z1; b2=z2; b3=z3; b4=z4; b5=z5; rp=rx;
rp1=rp[[1]]; rp2=rp[[2]]; rp3=rp[[3]];

(* Variables de control *)
t = 0; (* Para discretizar en varios puntos la trayectoria seleccionada *)
i = 1; (* Para guardar la solución de cada punto de la trayectoria *)

(* Valores iniciales para resolver el sistema de ecuaciones *)

ir10=.7071; ir13=.7071; ir20=.7071; ir21=.7071;
ir30=.7071; ir31=.7071;

(* Trayectoria en línea recta dada por los puntos extremos *)

S0={-8,-6,10};S1={8,-6,4};

(* Se inicia un ciclo para resolver el problema cinemático inverso en los puntos en que será discretizada la trayectoria *)

For[t=0, t<1, t+=.1,

(* Se determinan las coordenadas del punto correspondiente de la trayectoria*)
px=S0[[1]]+(S1[[1]]-S0[[1]])*t;
py=S0[[2]]+(S1[[2]]-S0[[2]])*t;
pz=S0[[3]]+(S1[[3]]-S0[[3]])*t;</td>
</tr>
</table>

valores iniciales mostrados. Una vez que se resuelve el sistema de ecuaciones anterior, se verifica si la precisión de la solución es la adecuada, si es así, se fijan como valores iniciales, para resolver la cinemática inversa del siguiente punto, a la solución obtenida y se continua el ciclo, si la solución está fuera de la precisión se envía un mensaje y se restauran los valores iniciales continuando el ciclo hasta terminar.

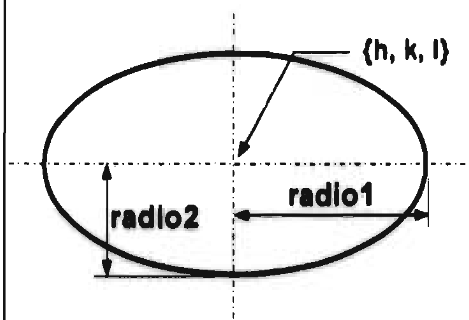
Una vez determinados los parámetros para ubicar el elemento terminal sobre la trayectoria, se determina la rotación necesaria para su orientación en cada punto de la trayectoria ($r4_0$, $r4_3$). En este caso deseamos que el eje transversal de la pinza (b_5) sea perpendicular a la trayectoria propuesta. Para determinar los parámetros $r4_0$ y $r4_3$, calculamos el ángulo formado entre una línea que sea perpendicular a la trayectoria y perpendicular a b_4 (producto cruz entre la trayectoria y b_4) y el eje transversal de la pinza b_5.

```
(* Solución del problema cinemático inverso para
   el punto seleccionado *)

solucion[i] =
FindRoot[{rp1 == px, rp2 == py, rp3 == pz,
          r10^2+r13^2 == 1, r20^2+r21^2 == 1,
          r30^2+r31^2 == 1},
         {r10,ir10},{r13,ir13},{r20,ir20},
         {r21,ir21},{r30,ir30},{r31,ir31},
         MaxIterations->50];

(* Se verifica la precisión de la solución *)

e = 10^-5; (* Precisión requerida *)

If[Abs[px-(rp1/.solucion[i])] < e &&
  Abs[py-(rp2/.solucion[i]) ]< e &&
  Abs[pz-(rp3/.solucion[i])] < e,

(* Si la solución está en la precisión fijada, se
   definen como valores iniciales, para resolver la
   cinemática inversa del siguiente punto de la
   trayectoria a los valores de la solución obtenida,
   si no es así se restauran los valores iniciales
   utilizados *)

ir10=r10/.solucion[i]; ir13=r13/.solucion[i];
ir20=r20/.solucion[i]; ir21=r21/.solucion[i];
ir30=r30/.solucion[i]; ir31=r31/.solucion[i],

(* Se imprime un mensaje si la solución no está en
   la precisión definida *)

Print["No se encontró solución para el punto:",
       {px,py,pz}];

(* Se restauran valores iniciales para resolver las
   ecuaciones si no se encontró solución para el
   punto en cuestión *)

ir10=.7071; ir13=.7071; ir20=.7071; ir21=.7071;
ir30=.7071; ir31=.7071];

(* Se incremente la variable i y se cierra el ciclo
   For *)

          i++ ]

(* Se determina la orientación de la pinza
   (perpendicular a la trayectoria) *)

(* Se almacenan en variables separadas la
   solución para cada par de parámetros *)
```

	For[j=1, j<i, j++, solr1[j]={r10,r13}/.solucion[j]; solr2[j]={r20,r21}/.solucion[j]; solr3[j]={r30,r31}/.solucion[j]]; (* Se determina un vector unitario en dirección de la trayectoria (tray) *) tray = (S0 - S1); tray = (tray)/(Sqrt[tray[[1]]^2 + tray[[2]]^2 + tray[[3]]^2]); (* Se inicia un ciclo para determinar la orientación de la pinza en cada punto de la trayectoria *) For[j=1, j<i, j++, (* Se asigna el valor de los parámetros para el punto de la trayectoria correspondiente almacenados en las variables solr?[j] *) r10=solr1[j][[1]];r13=solr1[j][[2]]; r20=solr2[j][[1]];r21=solr2[j][[2]]; r30=solr3[j][[1]];r31=solr3[j][[2]]; (* Se ubica la pinza en la posición original *) r40=1;r43=0; (* Se determina un vector perpendicular a la trayectoria y perpendicular al eje longitudinal de la pinza (b4) *) Lh = Cruz[b4,tray]; (* Se calcula el ángulo formado entre el vector anterior y el eje transversal de la pinza (b5), que es el ángulo que debe rotar la pinza *) cosenoang = (b5.Lh)/(Sqrt[Lh[[1]]^2 + Lh[[2]]^2 + Lh[[3]]^2]*L5); If[cosenoang!=0, ang = ArcCos[cosenoang], ang =90*Degree] ; (* Se almacena en forma de parámetros el ángulo que debe rotar la pinza *) solr4[j] = {Cos[ang/2],Sin[ang/2]};]
En esta sección, al igual que en la anterior se resuelve la cinemática inversa para una secuencia de puntos sobre una trayectoria, que en este caso se trata de una elipse o circunferencia. El proceso que se sigue es el mismo que para la línea recta, solamente	**CIRCUNFERENCIAS O ELIPSES** (* Se inicializan variables *) ClearAll[r10,r13,r20,r21,r30,r31,r40,r43,"solr1", "solr2","solr3","solr4", "solucion"];

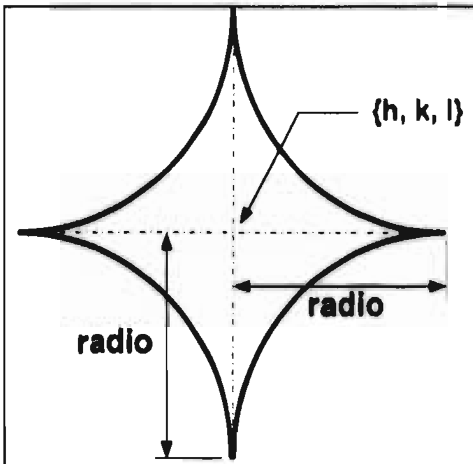
cambia la manera de determinar los puntos sobre la trayectoria, que en este caso se realiza utilizando la forma paramétrica de una elipse en el plano y-z, con centro en {h, k, l} con radios *radio1* y *radio2*.

px = h;
py = radio1 * Cos[t] +k ;
pz = radio2 * Sin[t] + l;




```
(*Se recuperan las ecuaciones del manipulador*)
b1=z1; b2=z2; b3=z3; b4=z4; b5=z5; rp=rx;
rp1=rp[[1]]; rp2=rp[[2]]; rp3=rp[[3]];

(* Variables de control *)
t = 0; (* Para discretizar en varios puntos la
         trayectoria seleccionada *)
i = 1; (* Para guardar la solución de cada punto
        de la trayectoria *)

(*Variables de control para definir la trayectoria *)

        radio1 = 7; (* Radio uno de la elipse *)
        radio2 = 5; (* Radio dos de la elipse *)
        h=5;k=0;l=6; (* Centro de la elipse *)

(* Valores iniciales para resolver el sistema de
   ecuaciones *)

ir10 = .7071; ir13 = .7071; ir20 = .3; ir21 = .9;
ir30 = .5; ir31 = .8;

(* Se inicia un ciclo para resolver el problema
   cinemático inverso en los puntos en que será
   discretizará la trayectoria *)

For[t=0, t<=2*N[Pi], t+=N[Pi]/9,

(* Se determinan las coordenadas del punto
   correspondiente de la trayectoria *)

       px = h;
       py = radio1*Cos[t] + k;
       pz = radio2*Sin[t] + l;

(* Solución del problema cinemático inverso para
   el punto seleccionado *)

solucion[i] =
FindRoot[{rp1 == px, rp2 == py, rp3 == pz,
          r10^2+r13^2 == 1, r20^2+r21^2 == 1,
          r30^2+r31^2 == 1},
          {r10,ir10},{r13,ir13},{r20,ir20},
          {r21,ir21},{r30,ir30},{r31,ir31},
          MaxIterations->50];

(* Se verifica la precisión de la solución *)

e = 10^-5; (* precisión requerida *)

If[Abs[px-(rp1/.solucion[i])] < e &&
  Abs[py-(rp2/.solucion[i])] < e &&
  Abs[pz-(rp3/.solucion[i])] < e,
```

	(* Si la solución está en la precisión fijada se definen como valores iniciales, para resolver la cinemática inversa del siguiente punto de la trayectoria, a los valores de la solución obtenida, si no es así se restauran los valores iniciales utilizados *) ir10=r10/.solucion[i]; ir13=r13/.solucion[i]; ir20=r20/.solucion[i]; ir21=r21/.solucion[i]; ir30=r30/.solucion[i]; ir31=r31/.solucion[i], (* Se imprime un mensaje si la solución no está en la precisión definida *) Print["No se encontró solución para el punto:", {px,py,pz}]; (* Se restauran valores iniciales para resolver las ecuaciones si no se encontró solución para el punto en cuestión *) ir10=.7071; ir13=.7071; ir20=.3; ir21=.9; ir30=.5; ir31=.8]; (* Se incremente la variable i y se cierra el ciclo For *) i++] (* Se determina la orientación de la pinza, se considera que el eje transversal de la pinza deberá se paralelo al plano x-y *) (* Como el eje transversal de la pinza no cambia su orientación al moverse el manipulador, si arranca de su posición inicial (b5 paralelo al plano x-y), simplemente la pinza debe rotar cero grados *) For[j=1, j<i, j++, solr4[j] = {Cos[0/2],Sin[0/2]}];
De la misma forma que se determinan los parámetros para las trayectorias anteriores, ahora se determinará una trayectoria en forma de ***astroride***, la cual se ubicará en el plano x-z, con centro en {h, k, l} cuya forma paramétrica es: **px = radio*Cos^3(t) + h;** **py = k;** **pz = radio*Sin^3(t) + l;**	**ASTROIDE** (* Se inicializan variables *) ClearAll[r10,r13,r20,r21,r30,r31,r40,r43,"solr1", "solr2","solr3","solr4", "solucion"]; (*Se recuperan las ecuaciones del manipulador*) b1=z1; b2=z2; b3=z3; b4=z4; b5=z5; rp=rx; rp1=rp[[1]]; rp2=rp[[2]]; rp3=rp[[3]]; (* Variables de control *) t = 0; (* Para discretizar en varios puntos la trayectoria seleccionada *)




```
i = 1; (* Para guardar la solución de cada punto
        de la trayectoria *)

(* Variable de control para definir la trayectoria *)

        radio = 6; (* Radio de la astroide *)
        h=0;k=-5;l=7; (* Centro de la elipse *)

(* Valores iniciales para resolver el sistema de
   ecuaciones *)

ir10 = .7071; ir13 = .7071; ir20 = .3; ir21 = .9;
ir30 = .5; ir31 = .8;

(* Se inicia un ciclo para resolver el problema
   cinemático inverso en los puntos en que será
   discretizada la trayectoria *)

For[t=0, t<=2*N[Pi], t+=N[Pi]/11,

(* Se determinan las coordenadas del punto
   correspondiente de la trayectoria *)
          px = radio*Cos[t]^3 + h;
          py = k;
          pz = radio*Sin[t]^3 + l;

(* Solución del problema cinemático inverso para
   el punto seleccionado *)

solucion[i] =
FindRoot[{rp1 == px, rp2 == py, rp3 == pz,
          r10^2+r13^2 == 1, r20^2+r21^2 == 1,
          r30^2+r31^2 == 1},
          {r10,ir10},{r13,ir13},{r20,ir20},
          {r21,ir21},{r30,ir30},{r31,ir31},
          MaxIterations->50];

(* Se verifica la precisión de la solución *)

e = 10^-5; (* precisión requerida *)

If[Abs[px-(rp1/.solucion[i])] < e &&
  Abs[py-(rp2/.solucion[i])] < e &&
  Abs[pz-(rp3/.solucion[i])] < e,

(* Si la solución está en la precisión fijada se
   definen como valores iniciales, para resolver la
   cinemática inversa del siguiente punto de la
   trayectoria, a los valores de la solución obtenida,
   si no es así se restauran los valores iniciales
   utilizados *)

ir10=r10/.solucion[i]; ir13=r13/.solucion[i];
ir20=r20/.solucion[i]; ir21=r21/.solucion[i];
```

<table>
<tr>
<td></td>
<td>

ir30=r30/.solucion[i]; ir31=r31/.solucion[i],

(* Se imprime un mensaje si la solución no está en la precisión definida *)

Print["No se encontró solución para el punto: ", {px,py,pz}];

(* Se restauran valores iniciales para resolver las ecuaciones si no se encontró solución para el punto en cuestión *)

ir10=.7071; ir13=.7071; ir20=.3; ir21=.9;
ir30=.5; ir31=.8];

(* Se incremente la variable i y se cierra el ciclo For *)

i++]

(* Se determina la orientación de la pinza, se considera que el eje transversal de la pinza deberá se paralelo al plano x-y *)

(* Como el eje transversal de la pinza no cambia su orientación al moverse el manipulador si arranca de su posición inicial (b5 paralelo al plano x-y), simplemente la pinza debe rotar cero grados *)

For[j=1, j<i, j++, solr4[j] = {Cos[0/2],Sin[0/2]}];

</td>
</tr>
<tr>
<td>

Una vez determinados los parámetros de rotación para la trayectoria seleccionada, ahora se generarán las gráficas que permitirán ver la simulación de la cinemática inversa del manipulador de cuatro grados de libertad. Para poder realizarlo, recuperamos los valores de los parámetros almacenados en la variable solución[j] y los almacenamos en variables separadas para cada par de parámetros. Al igual que en la cinemática inversa, se genera una gráfica que servirá como base para graficar el manipulador, la cual se almacena en las dos variables g1 y g2, posteriormente se inicia un ciclo, que se repetirá tantas veces como valores de los parámetros se hayan calculado para la trayectoria seleccionada (variable "i"). En cada pasada del ciclo se asignan los valores correspondientes a los parámetros y con ellos se genera la gráfica del manipulador de la misma forma que se hizo en la cinemática directa. Se mostrarán, a manera

</td>
<td>

Graficación para visualizar la simulación de la trayectoria seleccionada.

ClearAll[r10,r13,r20,r21,r30,r31,r40,r43]

(* Se recuperan los parámetros de rotación almacenados en la variable solucion[j], obtenidos de resolver el problema cinemático inverso y se almacenan en variables para cada par de parámetros *)

For[j=1, j<i, j++, solr1[j]={r10,r13}/.solucion[j];
solr2[j]={r20,r21}/.solucion[j];
solr3[j]={r30,r31}/.solucion[j]];

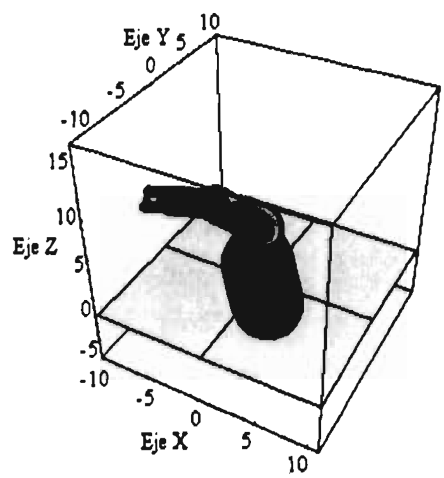
(* Se define el espacio tridimensional en que se graficará el manipulador, que se usará como gráfica base *)

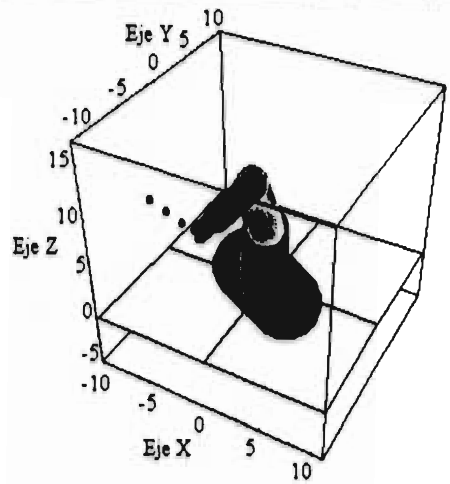
(*Suma de la longitudes de los eslabones 2,3 y 4 *)
L234 = L2 + L3 + L4;

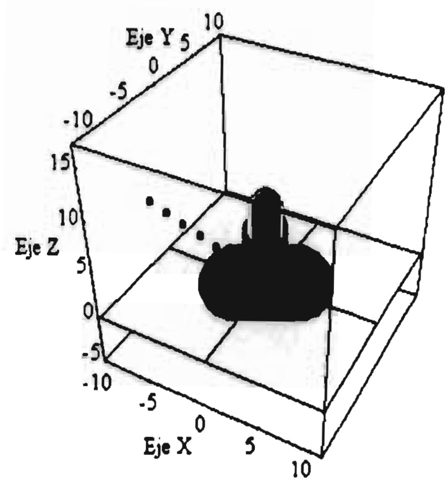
</td>
</tr>
</table>

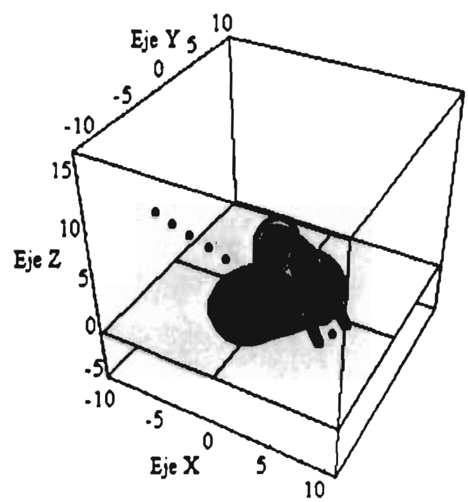
de ejemplo, algunas gráficas de una trayectoria.	g1=g2=SurfaceGraphics[{{0,0,0},{0,0,0},{0,0,0}}, MeshRange->{{-L234,L234},{-L234,L234}}, AxesLabel->{Eje X,Eje Y,Eje Z}, DefaultFont->{"Negrita",12}, PlotRange->{{-(L234+.1),(L234+.1)}, {-(L234+.1),(L234+.1)}, {(L1-L234-.1),(L1+L234+.1)}}, BoxRatios->{1,1,1},Axes->True]; (* Se inicia un ciclo para generar las gráficas de cada punto de la trayectoria *) For[j=1, j<i, j++, (* Se asigna el valor de los parámetros para el punto de la trayectoria correspondiente almacenados en las variables solr?[j]*) r10=solr1[j][[1]];r13=solr1[j][[2]]; r20=solr2[j][[1]];r21=solr2[j][[2]]; r30=solr3[j][[1]];r31=solr3[j][[2]]; r40=solr4[j][[1]];r43=solr4[j][[2]]; (* Se asignan valores a las ecuaciones del manipulador, según los valores de los parámetros, utilizando el respaldo de ecuaciones *) base = N[e11]; b1 = N[z1]; b2 = N[z2]; b3 = N[z3]; b4 = N[z4]; b5 = N[z5]; rp = N[rx]; (* Longitud del vector que servirá de base del manipulador *) a = 3; (* Se definen los puntos en el espacio tridimensional para graficar el manipulador *) punto1 = {0,0,0}; punto2 = N[a*base]; punto3 = N[-a*base]; punto4 = N[b1]; punto5 = N[punto4 + b2]; punto6 = N[punto5 + b3]; punto7 = N[punto6 + b5]; punto8 = N[punto6 - b5]; punto9 = N[punto7 + b4]; punto10 = N[punto8 + b4]; (* Se agrega a la gráfica base (g2) la posición del elemento terminal, como un punto *) g2 = {Graphics3D[{PointSize[0.01], Point[rp],RGBColor[0,1,0]}],g2};

```
(* Se genera la gráfica del manipulador en la
   posición correspondiente *)

g3=Graphics3D[

(* Base rotatoria *)
          {{AbsoluteThickness[40],RGBColor[1,.3,0],
           Line[{punto2,punto3}]},
(* Eslabón 1 *)
          {AbsoluteThickness[25],RGBColor[0,1,0],
           Line[{punto1,punto4}]},
(* Eslabón 2 *)
          {AbsoluteThickness[20],RGBColor[1,.4,0],
           Line[{punto4,punto5}]},
(* Eslabón 3 *)
          {AbsoluteThickness[15],RGBColor[0,0.5,1],
           Line[{punto5,punto6}]},
(* Pinza *)
          {AbsoluteThickness[5],RGBColor[0,0,0],
           Line[{punto6,punto7}]},
          {AbsoluteThickness[5],RGBColor[0,0,0],
           Line[{punto6,punto8}]},
          {AbsoluteThickness[5],RGBColor[0,0,0],
           Line[{punto7,punto9}]},
          {AbsoluteThickness[5],RGBColor[0,0,0],
           Line[{punto8,punto10}]}}];

(* Se envían a pantalla las gráficas g1,g2 y g3,
   combinadas *)

Show[g1,g3,g2,Axes->True];
                                        ]
```




Eje Y
Eje Z
Eje X
10
5
0
-5
-10
15


Eje Y
Eje Z
Eje X
10
5
0
-5
-10
15


Eje Y
Eje Z
Eje X
10
5
0
-5
-10
15

4.3 Cinemática de un robot tipo SCARA de cuatro grados de libertad.

En la siguiente figura se muestra la configuración del robot que se simulará en está sección, el cual fue previamente modelado en el capitulo anterior.

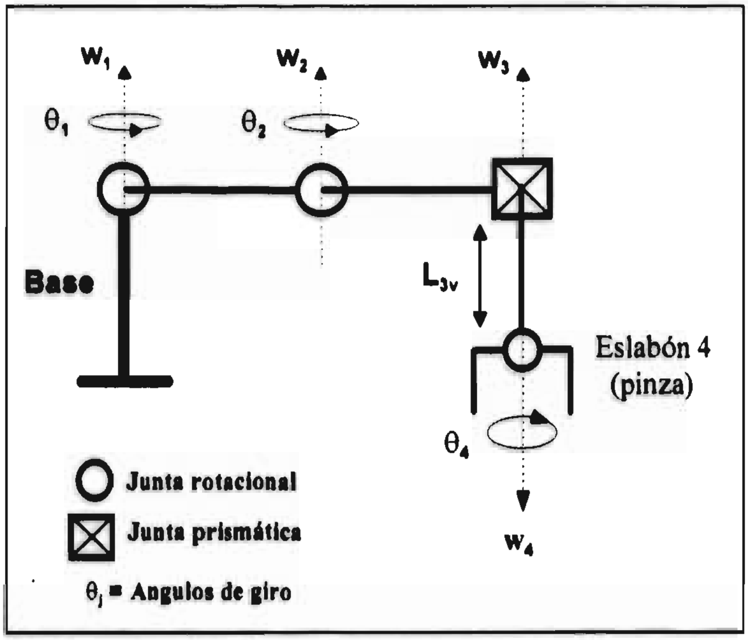


Figura 4.3 *Configuración del manipulador SCARA de cuatro grados de libertad*

4.3.1 Cinemática Directa.

El problema cinemático directo para este manipulador, consiste en que dados los desplazamientos angulares θ_1, θ_2, θ_4, y el desplazamiento lineal L_{3v}, se debe determinar la posición y orientación del elemento terminal, así como la configuración resultante de sus eslabones. Para realizar lo anterior hacemos uso de las ecuaciones (3.108), (3.109), (3.110) determinadas en el capitulo anterior, que al evaluarlas se resuelve la cinemática directa de este manipulador. Dichas ecuaciones se muestran a continuación.

$$\begin{aligned} b_1 &= \{B1{*}L_1, -2{*}B2{*}L_1, 0\} \\ b_2 &= \{C1{*}L_2, -2{*}C2{*}L_2, 0\} \\ b_3 &= \{0, 0, -L_3\} \\ b_{3v} &= \{0, 0, -(L_{3v}+l_{3v})\} \\ b_4 &= \{0, 0, -L_4\} \\ b_5 &= \{(4{*}A9{*}C2 - A10{*}C1){*}L_5, 2{*}(A9{*}C1 + A10{*}C2){*}L_5, 0\} \end{aligned} \qquad (4.23)$$

$$r_p = \{B1{*}L_1 + C1{*}L_2, -2{*}(B2{*}L_1 + C2{*}L_2), -(L_3 + L_{3v} + l_{3v} + L_4)\} \qquad (4.24)$$

Donde:

$$A1 = P_0{*}P_3; \qquad A2 = 1-2{*}P_0^2;$$
$$A3 = Q_0{*}Q_3; \qquad A4 = 1-2{*}Q_0^2;$$

$$\begin{array}{lll} A5 = r1_0 * r1_3; & A6 = 1-2*r1_0^2; & \\ A7 = r2_0 * r2_3; & A8 = 1-2*r2_0^2; & \\ A9 = r4_0 * r4_3; & A10 = 1-2*r4_0^2; & \\ B1 = A2*A8 - 4*A1*A7; & B2 = A2*A7 + A1*A8; & \\ B3 = A10*A8 - 4*A7*A9; & B4 = A10*A7 + A8*A9; & \\ B5 = A2*A4 - 4*A1*A3; & B6 = A2*A3 + A1*A4; & \\ C1 = B4*B5 + B3*B6; & C2 = B3*B5 - 4*B4*B6; & \end{array} \qquad (4.25)$$

Las variables de las ecuaciones anteriores tienen el siguiente significado:

$L_1, L_2, L_3, L_{3v}, L_4, L_5 \rightarrow$ Longitud de los eslabones del manipulador.
$b_1, b_2, b_3, b_{3v}, b_4, b_5 \rightarrow$ Eslabones del manipulador.
$r_p \rightarrow$ Posición del elemento terminal.
$P_0, P_3, Q_0, Q_3 \rightarrow$ Parámetros de rotación en la configuración no deformada.
$r1_0, r1_3, r2_0, r2_3, r4_0, r4_3 \rightarrow$ Parámetros de rotación en la configuración deformada.

Los parámetros de rotación tienen la siguiente relación con las rotaciones de los eslabones del manipulador.

$$\begin{array}{ll} P_0 = \mathrm{Cos}\,(\varphi_1/2); & P_3 = \mathrm{Sen}\,(\varphi_1/2); \\ Q_0 = \mathrm{Cos}\,(\varphi_2/2); & Q_3 = \mathrm{Sen}\,(\varphi_2/2); \end{array} \qquad (4.26)$$

$$\begin{array}{ll} r1_0 = \mathrm{Cos}\,(\theta_1/2); & r1_3 = \mathrm{Sen}\,(\theta_1/2); \\ r2_0 = \mathrm{Cos}\,(\theta_2/2); & r2_3 = \mathrm{Sen}\,(\theta_2/2); \\ r4_0 = \mathrm{Cos}\,(\theta_4/2); & r4_3 = \mathrm{Sen}\,(\theta_4/2); \end{array} \qquad (4.27)$$

Siendo:

φ_1 el ángulo de rotación del eslabón 2 en la configuración no deformada.
φ_2 el ángulo de rotación del eslabón 3 en la configuración no deformada.

θ_1 el ángulo de rotación del eslabón 1 en la configuración deformada.
θ_2 el ángulo de rotación del eslabón 2 en la configuración deformada.
θ_4 el ángulo de rotación de la pinza en la configuración deformada.

Antes de poder realizar el proceso de cinemática directa del manipulador, debemos determinar la configuración no deformada del robot mediante un proceso de cinemática inversa, que consiste en calcular los parámetros $P=\{P_0,P_1,0,0\}$ y $Q=\{Q_0,Q_1,0,0\}$ dada la posición del elemento terminal R_p, que en este caso es el extremo del eslabón 3 variable donde esta unida la pinza (Ecuaciones (3.96), (3.97)). Es decir, con las siguientes ecuaciones puede plantearse un sistema de ecuaciones, como se muestra.

$$R_p = \{- A2*L_1 + L_2*(A2*A4 - 4*A1*A3),\ 2*(A1*L_1 - L_2*(A2*A3 + A1*A4)),\ -(L_3 + l_{3v})\}; \qquad (4.28)$$

Donde:

$$\begin{array}{ll} A1 = P_0*P_1; & A2 = 1 - 2*P_0^2; \\ A3 = Q_0*Q_1; & A4 = 1 - 2*Q_0^2; \end{array} \qquad (4.29)$$

Con la ecuación para R_p podemos plantear 3 ecuaciones si damos la posición para el elemento terminal como $R_p = \{R_px, R_py, R_pz\}$ y con las normas unitarias de los Quaterniones P y Q completamos un sistema de 5 ecuaciones con 5 incógnitas, que al resolverlo obtenemos los parámetros que determinan la configuración no deformada.

$$
\begin{aligned}
-A2{*}L_1 + L_2{*}(A2{*}A4 - 4{*}A1{*}A3) &= R_p x \\
2{*}(A1{*}L_1 - L_2{*}(A2{*}A3 + A1{*}A4)) &= R_p y \\
-(L_3 + l_3) &= R_p z \qquad (4.30) \\
P_0^2 + P_1^2 &= 1 \\
Q_0^2 + Q_1^2 &= 1
\end{aligned}
$$

Con lo anterior podemos ahora resolver el problema cinemático directo y realizar la correspondiente simulación computacional, que como se mencionó se realizará en Mathematica®. A continuación se presenta el programa desarrollado para este caso. Con el objeto de mostrar la forma de utilizar Mathematica® para obtener las ecuaciones del manipulador, tanto en su configuración no deformada como en la deformada.

En la primera parte del programa se define la transformación rotación de un vector tridimensional y de un Quaternión, y de algunas otras funciones necesarias que se muestran a continuación.	**Definición de la Transformación Rotación.** (* Se apagan algunos mensajes del sistema *) Off[General::spell1] Off[General::spell]
Transformación de un Quaternión en un vector tridimensional. $q = \{q_0, q_1, q_2, q_3\} \in Q$ **Tv[q] = $\{q_1, q_2, q_3\}$.**	(* Se define la Transformación de un Quaternión a un vector tridimensional Tv[] *) Tv[q_]:={q[[2]],q[[3]],q[[4]]}; (* Se define la Transformación inversa de Tv[] *)
Transformación vectorial inversa. $v = \{v_1, v_2, v_3\} \in \Re^3$ **TvInv[v] = $\{0, v_1, v_2, v_3\}$.**	TvInv[v_]:={0,v[[1]],v[[2]],v[[3]]}; (* Se define la Transformación de un Quaternión a Quaternión vectorial *)
Transformación de un Quaternión en un Quaternión vectorial. **TvQ[q] = $\{0, q_1, q_2, q_3\}$.**	TvQ[q_]:={0,q[[2]],q[[3]],q[[4]]}; (* Se define la Multiplicación entre dos Quaterniones *)
Producto de dos Quaterniones. $p = \{p_0, p_1, p_2, p_3\}$, $q = \{q_0, q_1, q_2, q_3\} \in Q$ **ProductoQ[p,q] =** $\{p_0{*}q_0 - p_1{*}q_1 - p_2{*}q_2 - p_3{*}q_3,$ $p_0{*}q_1 + p_1{*}q_0 + p_2{*}q_3 - p_3{*}q_2,$ $p_0{*}q_2 - p_1{*}q_3 + p_2{*}q_0 + p_3{*}q_1,$ $p_0{*}q_3 + p_1{*}q_2 - p_2{*}q_1 + p_3{*}q_0\}$	ProductoQ[p_,q_]:= {p[[1]]*q[[1]] - p[[2]]*q[[2]] - p[[3]]*q[[3]] - p[[4]]*q[[4]], p[[1]]*q[[2]] + p[[2]]*q[[1]] + p[[3]]*q[[4]] - p[[4]]*q[[3]], p[[1]]*q[[3]] - p[[2]]*q[[4]] + p[[3]]*q[[1]] + p[[4]]*q[[2]], p[[1]]*q[[4]] + p[[2]]*q[[3]] - p[[3]]*q[[2]] + p[[4]]*q[[1]]}; (* Se define el conjugado de un Quaternión *)
Conjugado de un Quaternión.	ConjugadoQ[p_]:={p[[1]],-p[[2]],-p[[3]],-p[[4]]}; (* Se define la Transformación Rotación de un vector tridimensional *)

ConjugadoQ[p] = { p_0, - p_1, - p_2, - p_3 } Rotación de un vector v $\in \Re^3$ con un Quaternión p. **Rotacion[p,v] = Tv[p*TvInv[v]* $\bar{p}$]** Rotación de q $\in$ Q con un Quaternión p, considerando que el primer elemento no cambia al rotar cuando $\\|p\\| = 1$. **RotacionQ[p,q] = {q_0,0,0,0} + TvQ[p*q* $\bar{p}$]**	Rotacion[p_,v_]:= Tv[ProductoQ[p, ProductoQ[TvInv[v], ConjugadoQ[p]]]]; (* Se define la Transformación Rotación de un Quaternión *) RotacionQ[p_,q_]:={q[[1]],0,0,0}+ TvQ[ProductoQ[p,ProductoQ[q,ConjugadoQ[p]]]]; (* Se define el Producto Cruz entre dos vectores tridimensionales *) Cruz[a_,b_]:= {a[[2]] b[[3]] - a[[3]] b[[2]], - a[[1]] b[[3]] + a[[3]] b[[1]], a[[1]] b[[2]] - a[[2]] b[[1]]}
Se realizan definiciones iniciales necesarias, como son la base canónica de $\Re^3$ y las bases que servirán como bases locales en los eslabones. También se define la forma local de los Quaterniones que nos permitirán rotar, tanto en la posición no deformada como en la deformada.	**Definiciones iniciales** (* Base canónica *) e1 = {1,0,0}; e2 = {0,1,0}; e3 = {0,0,1}; (* Bases locales de los eslabones *) (* Eslabón 1 *) e1 = {1,0,0}; e2 = {0,1,0}; e3 = {0,0,1}; (* Eslabón 2 *) f1 = {1,0,0}; f2 = {0,1,0}; f3 = {0,0,1}; (* Eslabón 3 *) g1 = {1,0,0}; g2 = {0,1,0}; g3 = {0,0,1}; (* Eslabón 4 (pinza) *) h1 = {1,0,0}; h2 = {0,1,0}; h3 = {0,0,1}; (*Quaterniones para rotar a posición no deformada*) P = {P0,0,0,P3}; (* Eslabón 2 *) Q = {Q0,0,0,Q3}; (* Eslabón 3 *) (* Quaterniones para rotar a posición deformada *) r1 = {r10, 0, 0, r13}; (* Eslabón 1 *) r2 = {r20, 0, 0, r23}; (* Eslabón 2 *) r4 = {r40, 0, 0, r43}; (* Eslabón 4 *)
En esta sección del programa se realiza la Modelación de la configuración no deformada del manipulador. Como la forma de las bases locales de los eslabones, en la configuración no deformada, se obtuvo en el capitulo	**MODELADO DE LA CONFIGURACION NO DEFORMADA** **Modelo para la configuración no deformada.**

anterior, entonces para facilitar la obtención del las ecuaciones del manipulador, simplemente obtenemos de manera directa la forma de las bases locales, que son: $\mathbf{e}_j^{I} = \rho(\mathbf{P}, \mathbf{e}_j)$ (Eslabon 1) $\mathbf{f}_j^{II} = \rho((\mathbf{P}^*\mathbf{Q}), \mathbf{f}_j)$ (Eslabon 2) $\mathbf{g}_j^{II} = \rho((\mathbf{P}^*\mathbf{Q}), \mathbf{g}_j)$ (Eslabon 3) $\mathbf{h}_j^{II} = \rho((\mathbf{P}^*\mathbf{Q}), \mathbf{h}_j)$ (Pinza) Para lo anterior es necesario definir las normas unitarias de los Quaterniones P y Q, las cuales se almacenan en las variables "normas1" y "normas2", para aplicarlas en las ecuaciones resultantes de las rotaciones. Las variables que se utilizan para las bases locales de los eslabones son: ej = Primer eslabón. fj = Segundo eslabón. gj = Tercer eslabón. hj = Pinza. j = 1,2,3.	(* Se inicializan variables *) ClearAll[P0,P1,Q0,Q1,L1,L2,L3,b1,b2,b3]; (* Se definen variables para las normas unitarias de los Quaterniones P y Q *) normas1= {P1->(1-P0^2)^(1/2),Q1->(1-Q0^2)^(1/2)}; normas2= {(1-P0^2)^(1/2)->P1,(1-Q0^2)^(1/2)->Q1}; (* Se obtienen los productos de Quaterniones necesarios para rotar las bases *) PQ = ProductoQ[P,Q]; (* Se obtiene la forma de las bases locales de los eslabones en la configuración no deformada *) (* Eslabón 1 *) e11 = Expand[Expand[Rotacion[P,e1]]//.normas1] //.normas2; e21 = Expand[Expand[Rotacion[P,e2]]//.normas1] //.normas2; e31 = Expand[Expand[Rotacion[P,e3]]//.normas1] //.normas2; (* Eslabón 2 *) f12 = Expand[Expand[Rotacion[PQ,f1]]//.normas1] //.normas2; f22 = Expand[Expand[Rotacion[PQ,f2]]//.normas1] //.normas2; f32 = Expand[Expand[Rotacion[PQ,f3]]//.normas1] //.normas2; (* Eslabón 3 *) g12 = Expand[Expand[Rotacion[PQ,g1]]//.normas1] //.normas2; g22 = Expand[Expand[Rotacion[PQ,g2]]//.normas1] //.normas2; g32 = Expand[Expand[Rotacion[PQ,g3]]//.normas1] //.normas2; (* Eslabón 4 *) h12 = Expand[Expand[Rotacion[PQ,h1]]//.normas1] //.normas2; h22 = Expand[Expand[Rotacion[PQ,h2]]//.normas1] //.normas2; h32 = Expand[Expand[Rotacion[PQ,h3]]//.normas1] //.normas2;
Los eslabones del manipulador y la posición del elemento terminal son identificados, en la posición no deformada, por las variables siguientes:	(* Vectores que definen los eslabones del manipulador *) b1 = L1*e11; (* Eslabón 1 *) b2 = L2*f12; (* Eslabón 2 *)

b1 = Eslabón 1. b2 = Eslabón 2. b3 = Eslabón 3. b4 = Eslabón 3. b5 = Eslabón 3. Rp = Posición del elemento terminal. Las ecuaciones se simplifican utilizando las siguientes constantes.. A1 = P0*P1; A2 = 1 - 2*P0^2; A3 = Q0*Q1; A4 = 1 - 2*Q0^2;	b3 = L3*(-g32); (* Eslabón 3 fijo *) b3v = l3v*(-g32); (* Eslabón 3 variable *) b4 = L4*(-h32); (* Pinza *) b5 = L5*h12; (* Pinza *) (* Se simplifican las expresiones *) constantes = {P0*P3->A1, P0^2->(1-A2)/2, Q0*Q3->A3, Q0^2->(1-A4)/2}; b1 = Expand[Expand[b1//.constantes]//.constantes]; b2 = Expand[Expand[b2//.constantes]//.constantes]; b5 = Expand[Expand[b5//.constantes]//.constantes]; (* Vector que define posición del elemento terminal*) Rp = b1+b2+b3+b3v;
Es esta parte se muestran las ecuaciones obtenidas en la sección anterior, las cuales se utilizarán para determinar los parámetros P, Q y l3v, que definen la configuración no deformada en una posición dada del elemento terminal, que será de donde arranque la simulación de la cinemática directa o inversa del manipulador en cuestión. Es decir, antes de generar una simulación computacional de la cinemática directa o inversa del manipulador, debemos determinar la configuración no deformada realizando un proceso de cinemática inversa para calcular P y Q para una posición conocida del elemento terminal, la cual debe tener la forma siguiente: Rp = {Rpx,Rpy, Rpz}.	**Ecuaciones optimizadas para la posición no deformada.** ClearAll[P0,P3,Q0,Q3,l3v] (* Ecuaciones optimizadas *) b1 = {-A2*L1, 2*A1*L1, 0}; (* Eslabón 1 *) b2 = {(A2*A4 - 4*A1*A3)*L2, - 2*L2*(A2*A3 + A1*A4), 0}; (* Eslabón 2 *) b3 = {0, 0, -L3}; (* Eslabón 3 fijo *) b3v = {0, 0, -l3v}; (* Eslabón 3 variable *) (* pinza *) b4 = {0, 0, -L4}; b5 = {(A2*A4 - 4*A1*A3)*L5, - 2*L5*(A2*A3 + A1*A4), 0}; (* Posición del elemento terminal *) Rp = {L2*(A2*A4-4*A1*A3)-A2*L1, 2*(A1*L1-L2*(A2*A3+A1*A4)), -(L3+l3v)}; (* Constantes de las ecuaciones *) A1 = P0*P3; A2 = 1 - 2*P0^2; A3 = Q0*Q3; A4 = 1 - 2*Q0^2; (* Longitud de los eslabones del manipulador *) L1 = 6; L2 = 5; L3 = 1;

Se resuelve el sistema de ecuaciones mostrado en (4.19), para P0,P1,Q0,Q1, utilizando la función FindRoot, cuya sintaxis se puede consulta en el Apéndice, al igual que todas las funciones que se utilizan en este trabajo. Los resultados obtenidos para los Quaterniones se redondean si alguno de ellos está muy cercano a cero, finalmente se almacenan en las variables correspondientes (P0, P1, Q0, Q1) para poder ser utilizados posteriormente, además, se imprimen dichos valores.	**Cinemática Inversa para determinar la Configuración no Deformada** (* Conociendo la posición del elemento terminal se determinan los Quaterniones "P" y "Q" y la longitud l3v correspondientes *) ClearAll[P0,P3,Q0,Q3, l3v] (* Posición del elemento terminal *) posicion = {11,0,-1}; (* Componentes ortogonales de Rp *) Rp1 = Rp[[1]]; Rp2 = Rp[[2]]; Rp3 = Rp[[3]]; (* Se determinan "P", "Q" y "l3v" utilizando los siguiente valores iniciales *) iP0 = .7071; iP3 = .7071; iQ0 = .866; iQ3 = .5; il3v=0; (* Se almacena la solución en la variable sol *) sol = FindRoot[{Rp1 == posicion[[1]], Rp2 == posicion[[2]], Rp3 == posicion[[3]], P0^2+P3^2 == 1, Q0^2+Q3^2 == 1}, {P0,iP0},{P3,iP3},{Q0,iQ0}, {Q3,iQ3},{l3v,il3v},MaxIterations->50] (* Se asignan los valores de la solución a las variables correspondientes *) P0=P0/.sol; P3=P3/.sol; Q0=Q0/.sol; Q3=Q3/.sol; l3v=l3v/.sol; (* Se redondean los valores de los Quaterniones de la solución *) If[Chop[Abs[P0]-1]==0,P0=1;P3=0, f[Chop[Abs[P3]-1]==0,P0=0;P3=1]]; If[Chop[Abs[Q0]-1]==0,Q0=1;Q3=0, If[Chop[Abs[Q3]-1]==0,Q0=0;Q3=1]]; (* Se imprimen los resultados *) Print["P0 = ",P0," ","P3 = ",P3," Q0 = ",Q0," ", "Q3 = ",Q3, " ","l3v = ",l3v]
De la misma forma que se realizó la modelación de la configuración no deformada, ahora se presenta el cálculo de la forma final de las bases locales de los	**MODELADO DE LA CONFIGURACION DEFORMADA** **Modelo para la configuración deformada.**

<table>
<tr>
<td>
eslabones, las cuales se obtuvieron en el capitulo anterior y son:

$e_j^{II} = \rho(r1{*}P, e_j)$
$f_j^{IV} = \rho(r1{*}r2{*}P{*}Q, e_j)$
$g_j^{IV} = \rho(r1{*}r2{*}P{*}Q, e_j)$
$h_j^{VI} = \rho(r1{*}r2{*}r4{*}P{*}Q, e_j)$

También en este caso, se definen las normas unitarias de los Quaterniones utilizados, almacenándolas en las variables "normas3" y "normas4", que incluyen las normas unitarias de los Quaterniones correspondientes a la configuración no deformada.
</td>
<td>
<pre>
ClearAll[r10,r13,r20,r21,r40,r41,L1,L2,L3,L3v,L4,P0,
P3,Q0,Q3];

(* Posición no deformada conocida *)
        P = {P0,0,0,P3}; Q = {Q0,0,0,Q3};

(* Quaterniones para rotar a la configuración
   deformada *)
      r1 = {r10,0,0,r13}; (* Rotación Eslabón 1*)
      r2 = {r20,0,0,r23}; (* Rotación Eslabón 2*)
      r3 = {r30,0,0,r33}; (* Rotación Eslabón 3*)
      r4 = {r40,0,0,r43}; (* Rotación Eslabón 4*)

(* Normas unitarias de los Quaterniones *)
normas3 = Flatten[{normas1, r13->(1-r10^2)^(1/2),
      r23->(1-r20^2)^(1/2), r43->(1-r40^2)^(1/2)}];
normas4 = Flatten[{normas2, (1-r10^2)^(1/2)->r13,
      (1-r20^2)^(1/2)->r23, (1-r40^2)^(1/2)->r43}];

(* Se obtienen los productos de Quaterniones
   necesarios para rotar las bases *)

r1P = ProductoQ[r1,P];
r1r2PQ = ProductoQ[r1,ProductoQ[r2,
         ProductoQ[P,Q]]];
r1r2r4PQ = ProductoQ[r1,ProductoQ[r2,
         ProductoQ[r4,ProductoQ[P,Q]]]];

(* Se obtiene la forma final de las bases locales de
   los eslabones *)

(* Base local del eslabón 1 *)
e12 = Expand[Expand[Rotacion[r1P,e1]]//.normas3]
      //.normas4;
e22 = Expand[Expand[Rotacion[r1P,e2]]//.normas3]
      //.normas4;
e32 = Expand[Expand[Rotacion[r1P,e3]]//.normas3]
      //.normas4;

(* Base local del eslabón 2 *)
f14 = Expand[Expand[Rotacion[r1r2PQ,f1]]
      //.normas3]//.normas4;
f24 = Expand[Expand[Rotacion[r1r2PQ,f2]]
      //.normas3]//.normas4;
f34 = Expand[Expand[Rotacion[r1r2PQ,f3]]
      //.normas3]//.normas4;

(* Base local del eslabón 3 *)
g14 = Expand[Expand[Rotacion[r1r2PQ,g1]]
      //.normas3]//.normas4;
g24 = Expand[Expand[Rotacion[r1r2PQ,g2]]
      //.normas3]//.normas4;
g34 = Expand[Expand[Rotacion[r1r2PQ,g3]]
      //.normas3]//.normas4;
</pre>
</td>
</tr>
</table>

	(* Base local del eslabón 4 *) h15 = Expand[Expand[Rotacion[r1r2r4PQ,h1]] //.normas3]//.normas4; h25= Expand[Expand[Rotacion[r1r2r4PQ,h2]] //.normas3]//.normas4; h35 = Expand[Expand[Rotacion[r1r2r4PQ,h3]] //.normas3]//.normas4;
Una vez obtenidas las bases locales en su forma final, se obtienen las ecuaciones que representan los eslabones del manipulador y la posición del elemento terminal como: $\mathbf{b}_1 = L_1{}^* e_1^{II} = L_1{}^* e12;$ **(Eslabón 1)** $\mathbf{b}_2 = L_2{}^* f_1^{IV} = L_2{}^* f14;$ **(Eslabón 2)** $\mathbf{b}_3 = L_3{}^*(-g_3^{IV}) = -L_3{}^* g34;$ **(Eslabón 3 fijo)** $\mathbf{b}_{3v} = (L_{3v}+L_{3v})^*(-g_3^{IV}) =$ $\quad -(L_{3v}+L_{3v})^* g34;$ **(Eslabón 3 variable)** $\mathbf{b}_4 = L_4{}^*(-h_3^{V}) = -L_4{}^* h35;$ **(Pinza)** $\mathbf{b}_5 = L_5{}^* h_1^{V} = L_5{}^* h15;$ **(Pinza)** **(Posición del elemento terminal)** $r_p = \mathbf{b}_1 + \mathbf{b}_2 + \mathbf{b}_3 + \mathbf{b}_{3v} + \mathbf{b}_4$ Estas expresiones son simplificadas sustituyendo por constantes a aquellos términos que se repiten dentro de las diferentes ecuaciones, como se puede observar.	(* Vectores que definen los eslabones del manipulador *) b1 = L1*e12; (* Eslabón 1 *) b2 = L2*f14; (* Eslabón 2 *) b3 = L3*(-g34); (* Eslabón 3 fijo *) b3v = (L3v+l3v)*(-g34); (* Eslabón 3 variable *) b4 = L4*(-h35); (* Pinza *) b5 = L5*h15; (* Pinza *) (* Constantes para simplificar *) constantes = {P0*P3->A1, P0^2->(1-A2)/2, Q0*Q3->A3, Q0^2->(1-A4)/2, r10*r13->A5, r10^2->(1-A6)/2, r20*r23->A7, r20^2->(1-A8)/2, r40*r43->A9, r40^2->(1-A10)/2, A2*A6->4*A1*A5+B1, A2*A5->B2-A1*A6, A4*A8->4*A3*A7+B3, A4*A7->B4-A3*A8, B1*B3->C1+4*B2*B4, B1*B4->C2-B2*B3}; (* Se simplifican las expresiones *) b1 = Expand[Expand[b1//.constantes]//.constantes]; b2 = Expand[Expand[Expand[b2//.constantes] //.constantes]//.constantes]; b5 = Expand[Expand[Expand[b5//.constantes] //.constantes]//.constantes]; (* Vector que define posición del elemento terminal*) rp = Factor[b1+b2+b3+b3v+b4];
En esta parte se presenta el resultado obtenido en la sección anterior, es decir, las ecuaciones simplificadas del manipulador, así como, las constantes utilizadas en dichas ecuaciones. Esta sección es la que propiamente se utiliza para realizar la simulación de la cinemática directa del manipulador de tres grados de libertad, ya que es la que contiene las ecuaciones ya simplificadas. Como al realizar la simulación es necesario evaluar numéricamente las ecuaciones del manipulador, se genera un	**Ecuaciones optimizadas para la posición deformada.** ClearAll[r10,r13,r20,r23,r40,r43,L1,L2,L3v,L4]; (* Ecuaciones Simplificadas *) b1 = {B1*L1, -2*B2*L1, 0}; (* Eslabón 1 *) b2 = {C1*L2, -2*C2*L2, 0}; (* Eslabón 2 *) b3 = {0, 0, -L3}; (* Eslabón 3 fijo *) b3v = {0, 0, -(L3v+l3v)}; (* Eslabón 3 variable *)

<table>
<tr>
<td>respaldo de dichas ecuaciones en esta misma sección, almacenándolas en nuevas variables.</td>
<td><pre>(* Pinza *)

b4 = {0, 0, -L4};
b5 = {(4*A9*C2 - A10*C1)*L5,
        2*(A9*C1 + A10*C2)*L5, 0};

(* Posición del elemento terminal *)

rp = {B1*L1 + C1*L2, -2*(B2*L1 + C2*L2),
       -(L3+l3v+L3v+L4)};

(* Constantes de las ecuaciones *)

A1 = P0*P3;   A2 = 1-2*P0^2;
A3 = Q0*Q3;   A4 = 1-2*Q0^2;
A5 = r10*r13; A6 = 1-2*r10^2;
A7 = r20*r23; A8 = 1-2*r20^2;
A9 = r40*r43; A10 = 1-2*r40^2;

B1 = A2*A6 - 4*A1*A5;   B2 = A2*A5 + A1*A6;
B3 = A4*A8 - 4*A3*A7;   B4 = A4*A7 + A3*A8;
C1 = B1*B3 - 4*B2*B4;   C2 = B1*B4 + B2*B3;

(* Longitud de los eslabones del manipulador *)

base = 7; L1 = 6; L2 = 5; L3 = 1; L4 = 1; L5 =2;

(* Respaldo de ecuaciones *)

z1=b1; z2=b2; z3=b3; z3v=b3v; z4=b4; z5=b5;
rx=rp;</pre></td>
</tr>
<tr>
<td>Finalmente, en esta parte del programa, se presenta la generación de las gráficas que nos permitirán observar la simulación de la cinemática directa del manipulador en cuestión.

Para lograr generar las gráficas adecuadas, para poder observar la simulación, primero generamos mediante la función SurfaceGrafics una gráfica que contenga el espacio donde se mostrará el manipulador con una superficie simulando el piso y con la indicación de los ejes coordenados, la cual se almacena inicialmente en las variables g1 y g2.

Definimos los parámetros de rotación de cada eslabón, en función de los ángulos que utilizaremos para girar cada uno de los eslabones, que serán:</td>
<td>CINEMATICA DIRECTA

Graficación para visualizar la simulación.<pre>ClearAll[r10,r13,r20,r23,r40,r43,L3v];

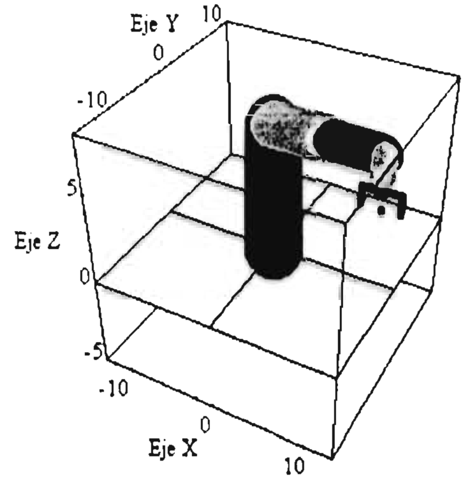
(* Se define el espacio tridimensional en que se
   graficará el manipulador, que se usará como
   gráfica base *)

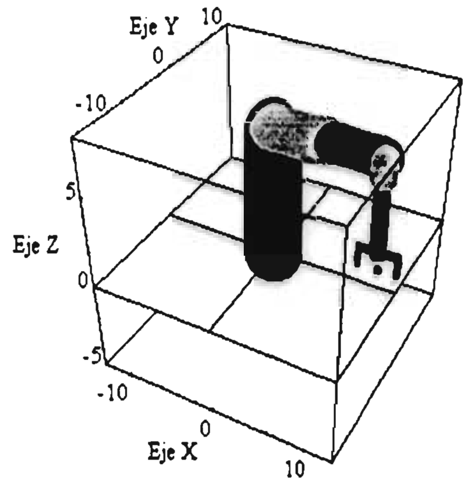
 (* Suma de la longitudes de eslabones *)
            L235 = L2+L3+L5;

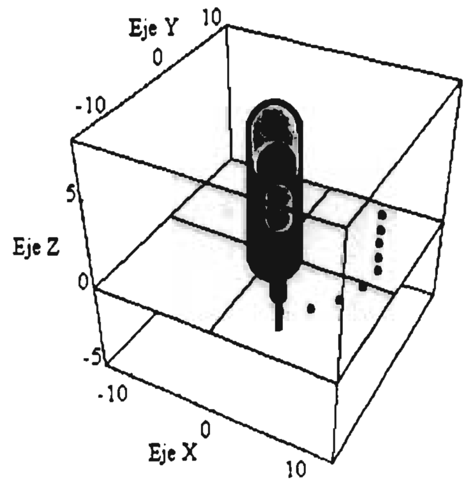
g1=g2=SurfaceGraphics[{{0,0,0},{0,0,0},{0,0,0}},
   MeshRange->{{-(L235+5),(L235+5)},
         {-(L235+5),(L235+5)}},
   AxesLabel->{Eje X,Eje Y,Eje Z},
         DefaultFont->{"Negrita",12},
   PlotRange->{{-(L235+5),(L235+5)},
         {-(L235+5),(L235+5)}, {-5,(base+1)}},
   BoxRatios->{1,1,1},Axes->True];</pre></td>
</tr>
</table>

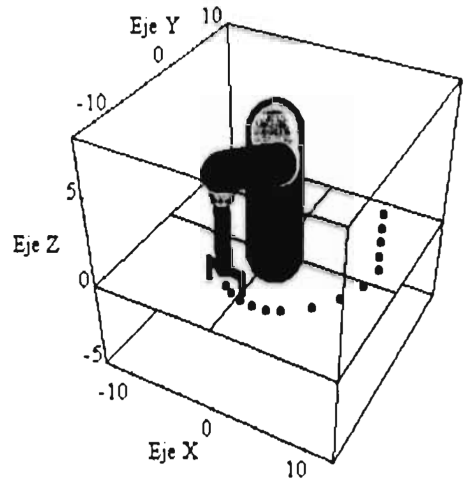
Eslabón 1 → **theta1** Eslabón 2 → **theta2** Eslabón 3 → **theta3**	(* Se inicializan en cero los ángulos de rotación *) theta1 = 0; theta2 = 0; theta3 = 0; L3v = 0;
Se generan dos ciclos ***For***, el primero (variable de control **i**) servirá para controlar cual eslabón deberá moverse, o sea, cuándo variar que ángulo, lo cual se logra utilizando la función ***Switch***, y el segundo ciclo (variable de control **j**) nos permitirá generar varias gráficas del movimiento de cada eslabón. La variable **i** toma valores de 1 a 3, ya que son tres eslabones y la variable **j** toma valores de 1 a n, donde n es el número de gráficas a generar por cada eslabón.	(* Se definen los valores de los parámetros de las rotaciones *) r10=Cos[theta1/2 Degree]; r13=Sin[theta1/2 Degree]; r20=Cos[theta2/2 Degree]; r23=Sin[theta2/2 Degree]; r40=Cos[theta3/2 Degree]; r43=Sin[theta3/2 Degree]; (* Se inicializa un ciclo para mover todos eslabones, uno a la vez *)
Una vez que los parámetros de rotación tienen un valor numérico, se evaluan numéricamente las ecuaciones del manipulador, haciendo uso del respaldo de ecuaciones realizado anteriormente	For[i=1, i<=4, i++, (* Se inicializa otro ciclo para generar varias gráficas para cada eslabón en diferentes posiciones*)
Con las ecuaciones, ya numéricas, se modifica la gráfica almacenada en **g2** agregándole un punto en la posición que corresponde a **rp**.	For[j=1, j<=5, j++, (* Se selecciona cual eslabón se moverá según el valor de la variable i, lo cual se logra cambiando sus parámetros de rotación *)
De la misma forma, con las ecuaciones numéricas, se determinan los diferentes puntos de la gráfica a generar, es decir, se genera un punto para cada inicio y final de cada eslabón, además de generar puntos para mostrar el vector **e11** de la base del primer eslabón.	Switch[i,1, (* Se mueve eslabón 1 *) r10=Cos[theta1/2 Degree]; r13=Sin[theta1/2 Degree]; theta1-=15 , 2, theta2-=15; (* Se mueve eslabón 2 *) r20=Cos[theta2/2 Degree]; r23=Sin[theta2/2 Degree];, 3, L3v+=.8, (* Se mueve eslabón 3 *)
Con los puntos anteriores, se genera la gráfica correspondiente a la posición del manipulador, utilizando para ello la función ***Graphics3D***, que nos permite trazar líneas de diferente color y espesor entre los puntos determinados.	4, theta3+=25; (* Se mueve eslabón 4 *) r40=Cos[theta3/2 Degree]; r43=Sin[theta3/2 Degree]]; (* Se asignan valores a las ecuaciones del manipulador, según los valores de los parámetros, utilizando el respaldo de ecuaciones *)
Finalmente se muestran en la pantalla, de manera combinada, las gráficas **g1, g2** y **g3** que se generan en cada pasada de los ciclos, utilizando para esto la función ***Show*** la cual envía a la pantalla objetos gráficos y permite que se combinen varias gráficas.	b1=N[z1]; b2=N[z2]; b3=N[z3]; b3v=N[z3v]; b4=N[z4]; b5=N[z5]; rp=N[rx]; (* Se definen los puntos en el espacio tridimensional para graficar el manipulador *)
Se muestran, a manera de ejemplo, algunas gráficas de un proceso.	punto1 = {0,0,0}; punto2 = {0,0,base}; punto3 = punto2 + b1; punto4 = punto3 + b2; punto5 = punto4 + b3;

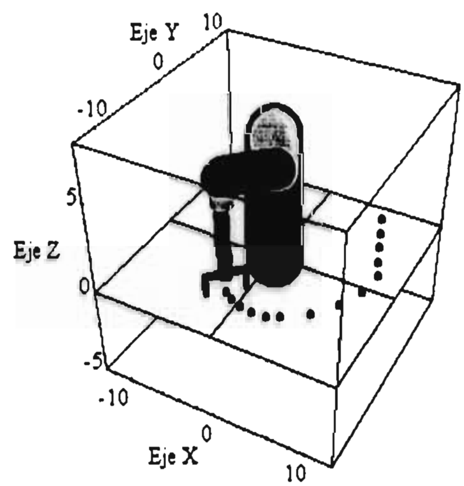
```
 punto6 = punto5 + b3v;
 punto7 = punto6 + b5;
 punto8 = punto6 - b5;
 punto9 = punto7 + b4;
punto10 = punto8 + b4;
punto11 = punto2 + rp;

(* Se agrega a la gráfica g2 la posición del elemento
   terminal como un punto *)

g2={Graphics3D[{PointSize[0.01],
      Point[punto11], RGBColor[0,1,0]}],g2};

(* Se genera la gráfica del manipulador en la
   posición correspondiente *)

g3=Graphics3D[{

(* Base *)
          {AbsoluteThickness[30],RGBColor[0,1,0],
           Line[{punto1,punto2}]},
(* Eslabón 1 *)
          {AbsoluteThickness[25],RGBColor[1,.4,0],
           Line[{punto2,punto3}]},
(* Eslabón 2 *)
          {AbsoluteThickness[20],RGBColor[0,.6,1],
           Line[{punto3,punto4}]},
(* Eslabón 3 fijo *)
          {AbsoluteThickness[15],RGBColor[1,.4,0],
           Line[{punto4,punto5}]},
(* Eslabón 3 variable*)
          {AbsoluteThickness[8],RGBColor[0,0,1],
           Line[{punto5,punto6}]},
(* Pinza *)
          {AbsoluteThickness[3],RGBColor[0,0,0],
           Line[{punto7,punto8}]},
          {AbsoluteThickness[3],RGBColor[0,0,0],
           Line[{punto7,punto9}]},
          {AbsoluteThickness[3],RGBColor[0,0,0],
           Line[{punto8,punto10}]} }];

(* Se envían a pantalla las gráficas g1,g2 y g3
   combinadas *)

Show[g1,g3,g2,Axes->True];
                                          ]
                                              ];
```


Eje Y
10
0
-10
5
Eje Z
0
-5
-10
0
Eje X
10


Eje Y
10
0
-10
5
Eje Z
0
-5
-10
0
Eje X
10


Eje Y
10
0
-10
5
Eje Z
0
-5
-10
0
Eje X
10





4.3.2 Cinemática inversa.

La cinemática inversa del robot SCARA, consiste en que, dada la posición y orientación del elemento terminal, se deben determinar los ángulos que deberán desplazarse cada una de las juntas del manipulador para alcanzar la posición deseada. En este caso, consiste en que dada la configuración no deformada (P_0, P_1, Q_0, Q_1, l_{3v}) y la posición del elemento terminal (r_p = {$r_p x$, $r_p y$, $r_p z$}), se deben determinar los parámetros $r4_0$ y $r4_3$ que orienten el elemento terminal y los parámetros $r1_0$, $r1_3$, $r2_0$, $r2_3$ que satisfagan la ecuación de la posición del elemento terminal, que es la siguiente:

$$r_p = \{B1*L_1 + C1*L_2, -2*(B2*L_1 + C2*L_2), -(L_3 + L_{3v} + l_{3v} + L_4)\} \qquad (4.31)$$

Para resolver la cinemática inversa del robot SCARA, debemos plantear un sistema de 5 ecuaciones que nos permitan determinar los 4 parámetros de rotación y la traslación del eslabón 3 (L_{3v}) que permitan ubicar el elemento terminal en la posición deseada. De la ecuación de r_p obtenemos 3 ecuaciones y con las normas unitarias de los Quaterniones se completa el sistema, el cual se muestra a continuación.

$$\begin{aligned}
r_p x &= B1*L_1 + C1*L_2 \\
r_p y &= -2*(B2*L_1 + C2*L_2) \\
r_p z &= -(L_3 + L_{3v} + l_{3v} + L_4) \\
r1_0^2 + r1_3^2 &= 1 \\
r2_0^2 + r2_3^2 &= 1 \\
r3_0^2 + r3_1^2 &= 1
\end{aligned} \tag{4.32}$$

Siendo:

$$\begin{array}{ll}
A1 = P_0*P_3; & A2 = 1-2*P_0^2; \\
A3 = Q_0*Q_3; & A4 = 1-2*Q_0^2; \\
A5 = r1_0*r1_3; & A6 = 1-2*r1_0^2; \\
A7 = r2_0*r2_3; & A8 = 1-2*r2_0^2; \\
A9 = r4_0*r4_3; & A10 = 1-2*r4_0^2; \\
B1 = A2*A8 - 4*A1*A7; & B2 = A2*A7 + A1*A8; \\
B3 = A10*A8 - 4*A7*A9; & B4 = A10*A7 + A8*A9; \\
B5 = A2*A4 - 4*A1*A3; & B6 = A2*A3 + A1*A4; \\
C1 = B4*B5 + B3*B6; & C2 = B3*B5 - 4*B4*B6;
\end{array} \tag{4.33}$$

Lo anterior nos permite ubicar el elemento terminal en el punto $\{r_p x, r_p y, r_p z\}$ dentro de su espacio de trabajo, ahora si deseamos que el elemento terminal siga una trayectoria determinada, una de las cosas que podemos hacer es discretizar dicha trayectoria en varios puntos a lo largo de ella y determinar los parámetros de rotación de los eslabones del manipulador para cada punto, resolviendo el sistema de ecuaciones (4.32), lo que nos permitirá ubicar el elemento terminal sobre dicha secuencia de puntos. Lo anterior no permite ubicar el elemento terminal en cualquier punto de la trayectoria, solo en los puntos en que dicha trayectoria se discretizó.

Para determinar la orientación del elemento terminal ($r4_0$ y $r4_3$), debemos considerar el hecho de que solamente cambia de orientación el eje transversal de la pinza (b_5), el cual siempre es paralelo al plano x-y. Basándonos en lo anterior, podemos orientar la pinza solamente con respecto al plano x-y, lo cual podemos hacer determinando el ángulo entre b_5 y la proyección sobre el plano x-y, de la línea de referencia con la que se desea orientar la pinza. A continuación se presenta el programa desarrollado en Mathematica® para la cinemática inversa del manipulador tipo SCARA de cuatro grados de libertad, el cual es continuación del anterior donde se presentó la cinemática directa.

Este programa es parte integral del anterior, es decir, para que funcione deberá tenerse la parte anterior donde se presenta modelación y la cinemática directa, ya que se requieren las ecuaciones del manipulador. En esta sección se presenta el cálculo para tres diferentes trayectorias y desde luego la graficación para observar la simulación.	**CINEMATICA INVERSA** **Solución de ecuaciones para diferentes trayectorias**

En esta parte se determinan los parámetros de rotación de los eslabones, para una sucesión de puntos que pertenecen a una línea recta, la cual se define por sus dos puntos extremos (S0, S1). Para lo anterior, primeramente recuperamos las ecuaciones del manipulador, las cuales se respaldaron en otras variables, (deberá resolverse la cinemática inversa para determinar la configuración no deformada y ejecutar la celda donde están las ecuaciones optimizadas para la configuración deformada).

Para determinar la puntos sobre la trayectoria, utilizamos la forma paramétrica de una recta que pasa por dos puntos $p_1 = \{x_1, y_1, z_1\}$ y $p_2 = \{x_1, y_1, z_1\}$:

$$px = x_1 + (x_2 - x_1)t$$
$$py = y_1 + (y_2 - y_1)t$$
$$pz = z_1 + (z_2 - z_1)t$$

Con lo anterior se realiza un ciclo utilizando como variable de control del mismo, a la variable **t**, con lo que determinamos puntos sobre la trayectoria a seguir y resolvemos el sistema de ecuaciones (4.32).

$$r_p x = B1 * L_1 + C1 * L_2$$
$$r_p y = -2*(B2*L_1 + C2*L_2)$$
$$r_p z = -(L_3 + L_{3v} + l_{3v} + L_4)$$
$$r1_0^2 + r1_3^2 = 1$$
$$r2_0^2 + r2_3^2 = 1$$
$$r3_0^2 + r3_1^2 = 1$$

Este sistema de ecuaciones se resuelve utilizando la función Findroot usando los valores iniciales mostrados. Una vez que se resuelve el sistema de ecuaciones anterior, se verifica si la precisión de la solución es la adecuada, si es así, se fijan como valores iniciales, para resolver la cinemática inversa del siguiente punto, a la solución obtenida y se continua el ciclo, si la solución está fuera de la precisión se envía un mensaje y se restauran los valores iniciales continuando el ciclo hasta terminar.

Una vez determinados los parámetros para ubicar el elemento terminal sobre la trayectoria, se determina la rotación necesaria para su orientación en cada punto de la

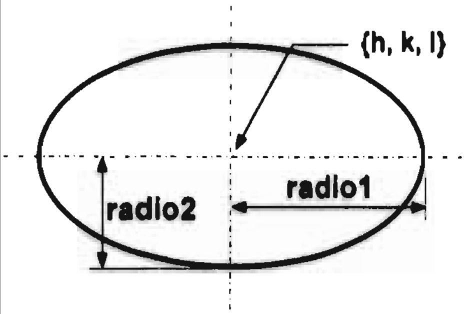
LINEA RECTA

```
(* Se inicializan variables *)
ClearAll[r10, r13, r20, r23, r40, r43, "solr1", "solr2",
"solr4", "solL3v", "solucion", radio];

(* Se recuperan las ecuaciones del manipulador *)
b1=z1; b2=z2; b3=z3; b3v=z3v; b4=z4; b5=z5;
rp=rx; rp1=rp[[1]]; rp2=rp[[2]]; rp3=rp[[3]];

(* Variables de control *)
t=0; (* Para discretizar en varios puntos la
        trayectoria seleccionada *)
i=1; (* Para guardar la solución de cada punto de
        la trayectoria *)

(* Valores iniciales para resolver el sistema de
   ecuaciones *)

        ir10 = .5; ir13 = .86; ir20 = .86; ir23 = .5;

(* Trayectoria en línea recta dada por los puntos
   extremos *)

        S0 = {9, 6, 1}; S1 = {6, -9, 4};

(* Se inicia un ciclo para resolver el problema
   cinemático inverso en los puntos en que será
   discretizada la trayectoria *)

For[t=0, t<=1, t+=.1,

(* Se determinan las coordenadas del punto
   correspondiente de la trayectoria *)
        px = S0[[1]]+(S1[[1]]-S0[[1]])*t;
        py = S0[[2]]+(S1[[2]]-S0[[2]])*t;
        pz = S0[[3]]+(S1[[3]]-S0[[3]])*t;

(* Se determina el desplazamiento del eslabón 3 *)

        solL3v[i] = base-(L3+L4)-pz;

(* Solución del problema cinemático inverso para
   el punto seleccionado *)

solucion[i] = FindRoot[{rp1==px, rp2==py,
        r10^2+r13^2==1, r20^2+r23^2==1},
        {r10,ir10},{r13,ir13},{r20,ir20},{r23,ir23},
        MaxIterations->50];

(* Se verifica la precisión de la solución *)

        e = 10^-5; (* precisión requerida *)
```

trayectoria ($r4_0$, $r4_3$). En este caso deseamos que el eje transversal de la pinza (b_5) sea perpendicular a la trayectoria propuesta. Para determinar los parámetros $r4_0$ y $r4_3$, calculamos el ángulo formado entre b_5 y la proyección de la trayectoria sobre el plano x-y, siendo el ángulo que deberá girar la pinza el complementario al ángulo calculado. Para determinar el sentido de giro de la pinza se realiza el producto cruz(producto cruz b_5 y la proyección de la trayectoria sobre el plano x-y.

```
If[N[Abs[px-(rp1/.solucion[i])]<e &&
     Abs[py-(rp2/.solucion[i])]<e],

(* Si la solución está en la precisión fijada, se
   definen como valores iniciales para resolver la
   cinemática inversa del siguiente punto de la
   trayectoria a los valores de la solución obtenida,
   si no es así se restauran los valores iniciales
   utilizados *)

  ir10=r10/.solucion[i];ir13=r13/.solucion[i];
  ir20=r20/.solucion[i];ir23=r23/.solucion[i];,

(* Se imprime un mensaje si la solución no está en
   la precisión definida *)

  Print["No se encontró solución para el punto: ",
        {px,py,pz}];

(* Se restauran valores iniciales para resolver las
   ecuaciones si no se encontró solución para el
   punto en cuestión *)

  ir10 =.5; ir13 =.86; ir20 =.5; ir23 =.86; ];  i++];

(* Cálculo de la orientación del elemento terminal
   (b5 perpendicular a la trayectoria) *)

(* Se almacenan en variables separadas la
   solución para cada par de parámetros *)

For[j=1, j<i, j++, solr1[j]={r10,r13}/.solucion[j];
                   solr2[j]={r20,r23}/.solucion[j]];

(* Se determina un vector unitario en dirección de
   la proyección de trayectoria (tray) sobre el plano
   x-y *)

        tray = (S0 - S1);
        tray = (tray)/(Sqrt[tray[[1]]^2 +
                 tray[[2]]^2 + tray[[3]]^2]);
        tray =   {tray[[1]],tray[[2]],0};

(* Se inicia un ciclo para determinar la orientación
   de la pinza en cada punto de la trayectoria *)

For[j=1, j<i, j++,

(* Se asigna el valor de los parámetros para el
   punto de la trayectoria correspondiente
   almacenados en las variables solr?[j] *)

          r10=solr1[j][[1]];r13=solr1[j][[2]];
          r20=solr2[j][[1]];r23=solr2[j][[2]];
```

	(* Se ubica la pinza en la posición original *) r40=1;r43=0; (* Se calcula el ángulo entre el vector b5 (pinza) y la proyección de la trayectoria sobre el plano x-y *) cosenoang = (tray.b5)/L5; If[N[cosenoang]!=0, ang=ArcCos[cosenoang]/Degree, ang = 90] ; (* El ángulo que debe girar la pinza es el complementario de "ang" y el sentido del giro lo obtenemos con un producto cruz entre la trayectoria y b5 *) signo = N[Cruz[tray,b5][[3]]]; signo = signo/Abs[signo]; ang = signo*(90-ang); (* Se almacena en una variable los parámetros correspondientes *) solr4[j]=N[{Cos[ang/2*Degree],Sin[ang/2*Degree]}];]
En esta sección, al igual que en la anterior se resuelve la cinemática inversa para una secuencia de puntos sobre una trayectoria, que en este caso se trata de una elipse o circunferencia. El proceso que se sigue es el mismo que para la línea recta, solamente cambia la manera de determinar los puntos sobre la trayectoria, que en este caso se realiza utilizando la forma paramétrica de una elipse en el plano x-y, con centro en {h, k, l} con radios *radio1* y *radio2*. **px = radio1*Cos[t] + h;** **py = radio2*Sin[t] + k;** **pz = l;**	**CIRCUNFERENCIAS O ELIPSES** (* Se inicializan variables *) ClearAll[r10, r13, r20, r23, r40, r43, "solr1", "solr2", "solr4", "solL3v", "solucion", radio]; (* Se recuperan las ecuaciones del manipulador *) b1=z1; b2=z2; b3=z3; b3v=z3v; b4=z4; b5=z5; rp=rx; rp1=rp[[1]]; rp2=rp[[2]]; rp3=rp[[3]]; (* Variables de control *) t=0; (* Para discretizar en varios puntos la trayectoria seleccionada *) i=1; (* Para guardar la solución de cada punto de la trayectoria *) (* Variables de control para definir la trayectoria *) radio1 = 8; (* Radio uno de la elipse *) radio2 = 3; (* Radio dos de la elipse *) h=0;k=-7;l=0; (* Centro de la elipse *) (* Valores iniciales para resolver el sistema de ecuaciones *) ir10 = .5; ir13 = .86; ir20 = .5; ir23 = .86; (* Se inicia un ciclo para resolver el problema cinemático inverso en los puntos en que será discretizada la trayectoria *)



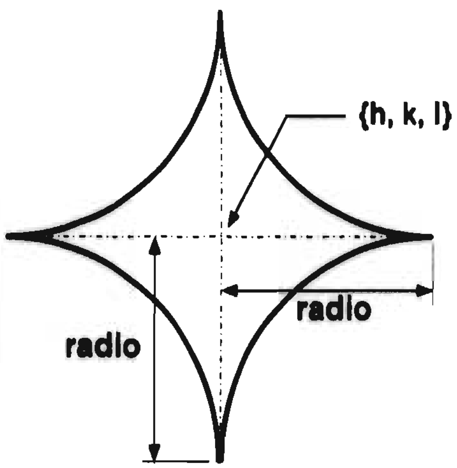
Para determinar la orientación de la pinza, que para este caso diremos que b_5 deberá ser paralelo al radio de la elipse, determinamos el ángulo que deberá girar la pinza, mediante el producto escalar entre el radio de la elipse y b_5. El sentido de giro se determina con el producto cruz entre b_5 y el radio.

```
For[t=0, t<=2*N[Pi], t+=N[Pi]/9,

(* Se determinan las coordenadas del punto
   correspondiente de la trayectoria *)

        px = radio1*Cos[t] + h;
        py = radio2*Sin[t] + k;
        pz = l;

(* Se almacena en una variable la proyección
   sobre el plano x-y del radio de la elipse que
   pasa por el punto de la trayectoria seleccionado
   *)

           radio[i]={px,py,0}-{h,k,0};

(* Se determina el desplazamiento del eslabón 3 *)

       solL3v[i] = base-(L3+L4)-pz;

(* Solución del problema cinemático inverso para
   el punto seleccionado *)

solucion[i] = FindRoot[{rp1==px, rp2==py,
        r10^2+r13^2==1, r20^2+r23^2==1},
        {r10,ir10},{r13,ir13},{r20,ir20},{r23,ir23},
        MaxIterations->50];

(* Se verifica la precisión de la solución *)

       e = 10^-5; (* precisión requerida *)

If[N[Abs[px-(rp1/.solucion[i])]<e &&
   Abs[py-(rp2/.solucion[i])]<e],

(* Si la solución está en la precisión fijada, se
   definen como valores iniciales para resolver la
   cinemática inversa del siguiente punto de la
   trayectoria a los valores de la solución obtenida,
   si no es así se restauran los valores iniciales
   utilizados *)

  ir10=r10/.solucion[i];ir13=r13/.solucion[i];
  ir20=r20/.solucion[i];ir23=r23/.solucion[i];,

(* Se imprime un mensaje si la solución no está en
   la precisión definida *)

  Print["No se encontró solución para el punto: ",
        {px,py,pz}];
```

	(* Se restauran valores iniciales para resolver las ecuaciones si no se encontró solución para el punto en cuestión *) ir10 =.5; ir13 =.86; ir20 =.5; ir23 =.86;]; i++]; (* Cálculo de la orientación del elemento terminal (b5 paralelo al radio de la elipse *) (* Se almacenan en variables separadas la solución para cada par de parámetros *) For[j=1, j<i, j++, solr1[j]={r10,r13}/.solucion[j]; solr2[j]={r20,r23}/.solucion[j]]; (* Se inicia un ciclo para determinar la orientación de la pinza en cada punto de la trayectoria *) For[j=1, j<i, j++, (* Se asigna el valor de los parámetros para el punto de la trayectoria correspondiente almacenados en las variables solr?[j] *) r10=solr1[j][[1]];r13=solr1[j][[2]]; r20=solr2[j][[1]];r23=solr2[j][[2]]; (* Se ubica la pinza en la posición original *) r40=1;r43=0; (* Se calcula el ángulo entre el vector b5 (pinza) y la línea recta (radio) del centro de la elipse al punto de la trayectoria seleccionado *) cosenoang = (radio[j].b5)/(Sqrt[radio[j][[1]]^2 + radio[j][[2]]^2]*L5); If[N[cosenoang]!=0, ang = ArcCos[cosenoang]/Degree, ang = 90] ; (* El ángulo que debe girar la pinza es el ángulo calculado, el sentido se determina mediante el producto cruz entre b5 y el radio *) signo = N[Cruz[b5,radio[j]][[3]]]; signo = signo/Abs[signo]; ang = signo*(ang); (* Se almacena en una variable los parámetros correspondientes *) solr4[j]=N[{Cos[ang/2*Degree],Sin[ang/2*Degree]}];]

De la misma forma que se determinan los parámetros para las trayectorias anteriores, ahora se determinará una trayectoria en forma de ***astroride***, la cual se ubicará en el plano x-y, con centro en {h, k, l} cuya forma paramétrica es:

$$px = radio{*}Cos^{3}(t) + h;$$
$$py = radio{*}Sin^{3}(t) + k;$$
$$pz = l;$$



Para determinar la orientación de la pinza, que para este caso deberá cumplir que b_5 y el eje x sean paralelos, determinamos el ángulo que deberá girar la pinza, mediante el producto escalar entre e_1 y b_5. El sentido de giro se determina con el producto cruz entre b_5 y e_1.

ASTROIDE

```
(* Se inicializan variables *)

ClearAll[r10, r13, r20, r23, r40, r43, "solr1", "solr2",
"solr4", "solL3v", "solucion", radio];

(* Se recuperan las ecuaciones del manipulador *)
b1=z1; b2=z2; b3=z3; b3v=z3v; b4=z4; b5=z5;
rp=rx; rp1=rp[[1]]; rp2=rp[[2]]; rp3=rp[[3]];

(* Variables de control *)
t=0; (* Para discretizar en varios puntos la
        trayectoria seleccionada *)
i=1; (* Para guardar la solución de cada punto de
        la trayectoria *)

(* Variables de control para definir la trayectoria *)
        radio = 10;    (* Radio de la astroide *)
        h=0;k=0;l=2;  (* Centro de la elipse *)

(* Valores iniciales para resolver el sistema de
   ecuaciones *)

        ir10 = .5; ir13 = .86; ir20 = .5; ir23 = .86;

(* Se inicia un ciclo para resolver el problema
   cinemático inverso en los puntos en que será
   discretizada la trayectoria *)

For[t=0, t<=2*N[Pi], t+=N[Pi]/11,

(* Se determinan las coordenadas del punto
   correspondiente de la trayectoria *)
        px = radio*Cos[t]^3 + h;
        py = radio*Sin[t]^3 + k;
        pz = l;

(* Se determina el desplazamiento del eslabón 3 *)

        solL3v[i] = base-(L3+L4)-pz;

(* Solución del problema cinemático inverso para
   el punto seleccionado *)

solucion[i] = FindRoot[{rp1==px, rp2==py,
              r10^2+r13^2==1, r20^2+r23^2==1},
             {r10,ir10},{r13,ir13},{r20,ir20},{r23,ir23},
              MaxIterations->50];

(* Se verifica la precisión de la solución *)

        e = 10^-5; (* precisión requerida *)
```

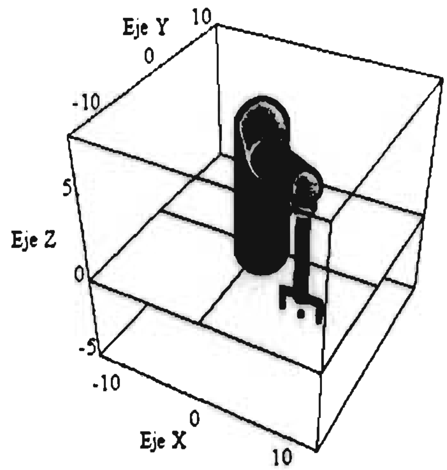
```
If[N[Abs[px-(rp1/.solucion[i])]<e &&
     Abs[py-(rp2/.solucion[i])]<e],

(* Si la solución está en la precisión fijada, se
   definen como valores iniciales para resolver la
   cinemática inversa del siguiente punto de la
   trayectoria a los valores de la solución obtenida,
   si no es así se restauran los valores iniciales
   utilizados *)

  ir10=r10/.solucion[i];ir13=r13/.solucion[i];
  ir20=r20/.solucion[i];ir23=r23/.solucion[i];,

(* Se imprime un mensaje si la solución no está en
   la precisión definida *)

  Print["No se encontró solución para el punto: ",
        {px,py,pz}];

(* Se restauran valores iniciales para resolver las
   ecuaciones si no se encontró solución para el
   punto en cuestión *)

  ir10 =.5; ir13 =.86; ir20 =.5; ir23 =.86; ];   i++];

(* Cálculo del la orientación del elemento terminal,
   deberá conservar su orientación original *)

(* Se almacenan en variables separadas la
   solución para cada par de parámetros *)

For[j=1, j<i, j++,solr1[j]={r10,r13}/.solucion[j];
                  solr2[j]={r20,r23}/.solucion[j]];

(* Se inicia un ciclo para determinar la orientación
   de la pinza en cada punto de la trayectoria *)

For[j=1, j<i, j++,

(* Se asigna el valor de los parámetros para el
   punto de la trayectoria correspondiente
   almacenados en las variables solr?[j] *)

       r10=solr1[j][[1]];r13=solr1[j][[2]];
       r20=solr2[j][[1]];r23=solr2[j][[2]];

(* Se ubica la pinza sin rotación *)

     r40 = 1;  r43=0;

(* Se determina el ángulo que debe rotar la pinza,
   que es el ángulo entre b5 y e1 *)

     ang = ArcCos[(b5.{1,0,0}/L5)];
```

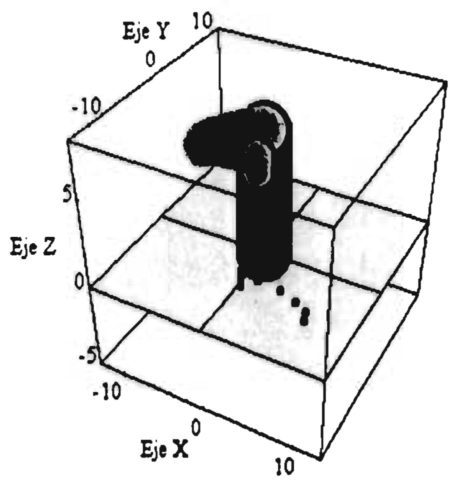
	(* Se determina el sentido de rotación de la pinza*) signo = Cruz[b5,{1,0,0}][[3]]; signo = signo/Abs[signo]; ang = signo*ang; (* Se almacena en una variable el ángulo calculado en forma de parámetros *) solr4[j] = {Cos[ang/2],Sin[ang/2]};]
Una vez determinados los parámetros de rotación para la trayectoria seleccionada, ahora se generarán las gráficas que permitirán ver la simulación de la cinemática inversa del manipulador tipo SCARA de cuatro grados de libertad.. Al igual que en la cinemática inversa, se genera una gráfica que servirá como base para graficar el manipulador, la cual se almacena en las dos variables g1 y g2, posteriormente se inicia un ciclo, que se repetirá tantas veces como valores de los parámetros se hayan calculado para la trayectoria seleccionada (variable "i"). En cada pasada del ciclo se asignan los valores correspondientes a los parámetros y con ellos se genera la gráfica del manipulador de la misma forma que se hizo en la cinemática directa. Se mostrarán, a manera de ejemplo, algunas gráficas de una trayectoria.	**Graficación para visualizar la simulación de la trayectoria seleccionada.** ClearAll[r10,r13,r20,r23,r40,r43]; (* Se recuperan los parámetros de rotación almacenados en la variable solucion[j], obtenidos de resolver el problema cinemático inverso For[j=1, j<i, j++, solr1[j]={r10,r13}/.solucion[j]; solr2[j]={r20,r23}/.solucion[j]];*) (* Se define el espacio tridimensional en que se graficará el manipulador, que se usará como gráfica base *) (* Suma de la longitudes de eslabones *) L235 = L2+L3+L5; g1=g2=SurfaceGraphics[{{0,0,0},{0,0,0},{0,0,0}}, MeshRange->{{-(L235+5),(L235+5)}, {-(L235+5),(L235+5)}}, AxesLabel->{Eje X,Eje Y,Eje Z}, DefaultFont->{"Negrita",12}, PlotRange->{{-(L235+5),(L235+5)}, {-(L235+5),(L235+5)}, {-5,(base+1)}}, BoxRatios->{1,1,1},Axes->True]; (* Se inicia un ciclo para generar las gráficas de cada punto de la trayectoria *) For[j=1, j<i, j++, (* Se asigna el valor de los parámetros para el punto de la trayectoria correspondiente *) r10=solr1[j][[1]]; r13=solr1[j][[2]]; r20=solr2[j][[1]]; r23=solr2[j][[2]]; r40=solr4[j][[1]]; r43=solr4[j][[2]]; L3v=solL3v[j];

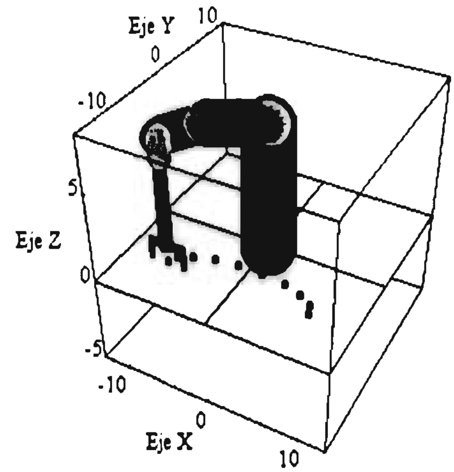
	(* Se asignan valores a las ecuaciones del manipulador, según los valores de los parámetros, utilizando el respaldo de ecuaciones *) b1=N[z1]; b2=N[z2]; b3=N[z3]; b3v=N[z3v]; b4=N[z4]; b5=N[z5]; rp=N[rx]; (* Se definen los puntos en el espacio tridimensional para graficar el manipulador *) punto1 = {0,0,0}; punto2 = {0,0,base}; punto3 = punto2 + b1; punto4 = punto3 + b2; punto5 = punto4 + b3; punto6 = punto5 + b3v; punto7 = punto6 + b5; punto8 = punto6 - b5; punto9 = punto7 + b4; punto10 = punto8 + b4; punto11 = punto2 + rp; (* Se agrega a la gráfica base la posición del elemento terminal como un punto *) g2={Graphics3D[{PointSize[0.01], Point[punto11], RGBColor[0,1,0]}], g2}; g3=Graphics3D[{ (* Base *) {AbsoluteThickness[30],RGBColor[0,1,0], Line[{punto1,punto2}]}, (* Eslabón 1 *) {AbsoluteThickness[25],RGBColor[1,.4,0], Line[{punto2,punto3}]}, (* Eslabón 2 *) {AbsoluteThickness[20],RGBColor[0,.6,1], Line[{punto3,punto4}]}, (* Eslabón 3 fijo *) {AbsoluteThickness[15],RGBColor[1,.4,0], Line[{punto4,punto5}]}, (* Eslabón 3 variable*) {AbsoluteThickness[8],RGBColor[0,0,1], Line[{punto5,punto6}]}, (* Pinza *) {AbsoluteThickness[3],RGBColor[0,0,0], Line[{punto7,punto8}]}, {AbsoluteThickness[3],RGBColor[0,0,0], Line[{punto7,punto9}]}, {AbsoluteThickness[3],RGBColor[0,0,0], Line[{punto8,punto10}]} }];

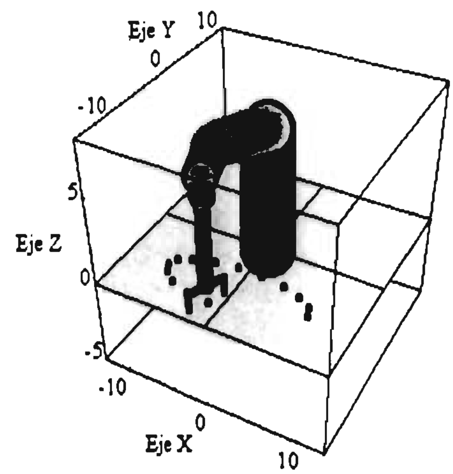
(* Se envían a pantalla las gráficas g1,g2 y g3, combinadas *)

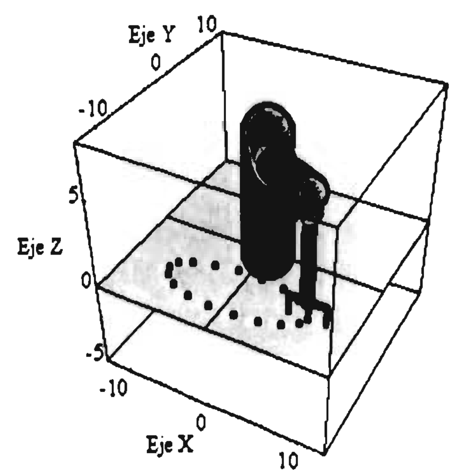
Show[g1,g3,g2,Axes->True];]






Eje Y
10
0
-10
5
Eje Z
0
-5
-10
0
Eje X
10


Eje Y
10
0
-10
5
Eje Z
0
-5
-10
0
Eje X
10


Eje Y
10
0
-10
5
Eje Z
0
-5
-10
0
Eje X
10

CAPITULO V

5. EVASION DE OBSTACULOS EN LA CINEMATICA INVERSA (Robot de tres grados de libertad).

Finalmente, en este capitulo, mostraremos una aplicación de la modelación con rotaciones y reflexiones del robot de tres grados de libertad (Figura 5.1) que se presentó en el capítulo 2. La aplicación consiste en lograr que el manipulador evada un obstáculo virtual en el proceso de cinemática inversa. Como sabemos que la diferencia entre los dos modelos obtenidos para el robot en cuestión es, principalmente, que al resolver la cinemática inversa para una misma posición del elemento terminal, se obtienen dos diferentes configuraciones de los eslabones del manipulador, utilizaremos esta propiedad con el fin de lograr la evasión del obstáculo mencionado. Es decir, si el obstáculo impide que el manipulador se posicione en un punto determinado, cuando se usa el modelo con rotaciones, se seleccionará el modelo (rotaciones y reflexiones) para posicionar el manipulador en el punto deseado, sin impactar el obstáculo. A continuación presentamos los sistemas de ecuaciones que se deberán resolver, para determinar la cinemática inversa del manipulador de tres grados de libertad.

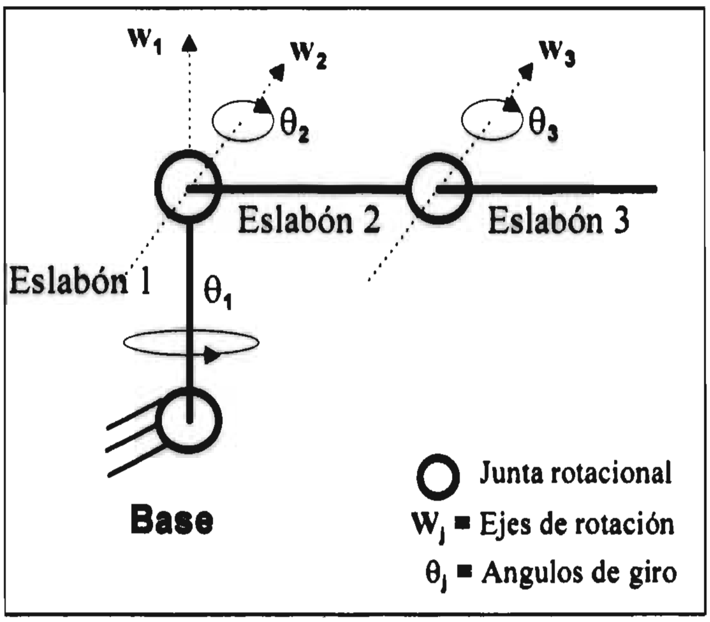


Figura 5.1. *Configuración del manipulador de tres grados de libertad*

Ecuaciones para la configuración no deformada (única). La cinemática inversa para la configuración no deformada consiste en calcular los parámetros $P=\{P_0,P_1,0,0\}$ y $Q=\{Q_0,Q_1,0,0\}$ dada la posición del elemento terminal $R_p = \{0, R_py, R_pz,\}$ (Ecuaciones (3.11), (3.12)), que satisfagan el siguiente sistema de ecuaciones.

$$2*((A2*A3 + A1*A4)*L_3 - A1*L_2) = R_p y$$
$$L_1 - A2*L_2 + (A2*A4 - 4*A1*A3)*L_3 = R_p z \qquad (5.1)$$
$$P_0^2 + P_1^2 = 1$$
$$Q_0^2 + Q_1^2 = 1$$

Siendo:

$$A1 = P_0*P_1; \quad A2 = 1 - 2*P_0^2; \qquad (5.2)$$
$$A3 = Q_0*Q_1; \quad A4 = 1 - 2*Q_0^2;$$

Ecuaciones para la configuración deformada (rotaciones). La cinemática inversa para la configuración deformada, con el modelo con puras rotaciones(Ecuaciones (3.41), (3.42), (3.43)), consiste en que, dada la posición del elemento terminal $r_p = \{r_p x, r_p y, r_p z\}$, calcular los parámetros $r1_0$, $r1_3$, $r2_0$, $r2_1$, $r3_0$, $r3_1$, que satisfagan el siguiente sistema de ecuaciones.

$$r_p x = -4*A5*(B2*L_2 + C1*L_3)$$
$$r_p y = -2*A6*(B2*L_2 + C1*L_3)$$
$$r_p z = L_1 + B1*L_2 + C2*L_3 \qquad (5.3)$$
$$r1_0^2 + r1_3^2 = 1$$
$$r2_0^2 + r2_1^2 = 1$$
$$r3_0^2 + r3_1^2 = 1$$

Siendo:

$$A1 = P_0*P_1; \quad A2 = 1-2*P_0^2;$$
$$A3 = Q_0*Q_1; \quad A4 = 1-2*Q_0^2;$$
$$A5 = r1_0*r1_3; \quad A6 = 1-2*r1_0^2;$$
$$A7 = r2_0*r2_1; \quad A8 = 1-2*r2_0^2;$$
$$A9 = r3_0*r3_1; \quad A10 = 1-2*r3_0^2; \qquad (5.4)$$
$$B1 = A2*A8 - 4*A1*A7; \quad B2 = A2*A7 + A1*A8;$$
$$B3 = A10*A8 - 4*A7*A9; \quad B4 = A10*A7 + A8*A9;$$
$$B5 = A2*A4 - 4*A1*A3; \quad B6 = A2*A3 + A1*A4;$$
$$C1 = B4*B5 + B3*B6; \quad C2 = B3*B5 - 4*B4*B6;$$

Ecuaciones para la configuración deformada (rotaciones y reflexiones). La cinemática inversa para la configuración deformada, utilizando el modelo con rotaciones y reflexiones (Ecuaciones (3.59), (3.60), (3.61)), consiste en que, dada la posición del elemento terminal $r_p = \{r_p x, r_p y, r_p z\}$, se deben determinar los parámetros $R1_0$, $R1_3$, $R2_0$, $R2_1$, $R3_0$, $R3_1$, que satisfagan el siguiente sistema de ecuaciones.

$$r_p x = 4*A5*(B2*L_2 + C1*L_3)$$
$$r_p y = 2*A6*(B2*L_2 + C1*L_3)$$
$$r_p z = L_1 - B1*L_2 + C2*L_3 \qquad (5.5)$$
$$R1_0^2 + R1_3^2 = 1$$
$$R2_0^2 + R2_1^2 = 1$$
$$R3_0^2 + R3_1^2 = 1$$

Siendo:

$$
\begin{array}{ll}
A1 = P_0{*}P_1; & A2 = 1\text{-}2{*}P_0^2; \\
A3 = Q_0{*}Q_1; & A4 = 1\text{-}2{*}Q_0^2; \\
A5 = R1_0{*}R1_3; & A6 = 1\text{-}2{*}R1_0^2; \\
A7 = R2_0{*}R2_1; & A8 = 1\text{-}2{*}R2_0^2; \\
A9 = R3_0{*}R3_1; & A10 = 1\text{-}2{*}R3_0^2; \\
B1 = A2{*}A8 + 4{*}A1{*}A7; & B2 = A1{*}A8 - A2{*}A7; \\
B3 = A10{*}A8 + 4{*}A7{*}A9; & B4 = A10{*}A7 - A8{*}A9; \\
B5 = A2{*}A3 + A1{*}A4; & B6 = A2{*}A4 - 4{*}A1{*}A3; \\
C1 = B4{*}B6 - B3{*}B5; & C2 = B3{*}B6 + 4{*}B4{*}B5;
\end{array}
\qquad (5.6)
$$

Lo anterior nos permite ubicar el elemento terminal en el punto $\{r_px,\ r_py,\ r_pz\}$ dentro de su espacio de trabajo con las dos posibles configuraciones del manipulador, ahora si deseamos que el elemento terminal siga una trayectoria determinada, una de las cosas que podemos hacer es discretizar dicha trayectoria en varios puntos a lo largo de ella y determinar los parámetros de rotación de los eslabones del manipulador para cada punto, resolviendo los sistema de ecuaciones anteriores, lo que nos permitirá ubicar el elemento terminal sobre dicha secuencia de puntos.. A continuación se presenta el programa desarrollado en Mathematica® para mostrar, mediante la simulación computacional, el efecto de la evasión de obstáculos que se quiere mostrar.

En la primera parte del programa se define la transformación rotación de un vector tridimensional y de un Quaternión, y de algunas otras funciones necesarias que se muestran a continuación.	**Definición de las Transformaciones Rotación y Reflexión.**
Transformación de un Quaternión en un vector tridimensional.	(* Se apagan algunos mensajes del sistema *)
$q = \{q_0,\ q_1,\ q_2,\ q_3\} \in Q$	Off[General::spell1] Off[General::spell]
Tv[q] = $\{q_1, q_2, q_3\}$.	(* Se define la Transformación de un Quaternión a un vector tridimensional Tv[] *)
Transformación vectorial inversa.	Tv[q_]:={q[[2]],q[[3]],q[[4]]};
$v = \{v_1,\ v_2,\ v_3\} \in \Re^3$	(* Se define la Transformación inversa de Tv[ ] *)
TvInv[v] = $\{0,v_1, v_2, v_3\}$.	TvInv[v_]:={0,v[[1]],v[[2]],v[[3]]};
Transformación de un Quaternión en un Quaternión vectorial.	(* Se define la Transformación de un Quaternión a Quaternión vectorial *)
TvQ[q] = $\{0,q_1, q_2, q_3\}$.	TvQ[q_]:={0,q[[2]],q[[3]],q[[4]]};
Producto de dos Quaterniones.	(* Se define la Multiplicación entre dos Quaterniones *)
$p = \{p_0,\ p_1,\ p_2,\ p_3\},\ q = \{q_0,\ q_1,\ q_2,\ q_3\} \in Q$	
ProductoQ[p,q] = $\{p_0{*}q_0 - p_1{*}q_1 - p_2{*}q_2 - p_3{*}q_3,$ $p_0{*}q_1 + p_1{*}q_0 + p_2{*}q_3 - p_3{*}q_2,$ $p_0{*}q_2 - p_1{*}q_3 + p_2{*}q_0 + p_3{*}q_1,$ $p_0{*}q_3 + p_1{*}q_2 - p_2{*}q_1 + p_3{*}q_0\}$	ProductoQ[p_,q_]:= {p[[1]]*q[[1]] - p[[2]]*q[[2]] - p[[3]]*q[[3]] - p[[4]]*q[[4]], p[[1]]*q[[2]] + p[[2]]*q[[1]] + p[[3]]*q[[4]] - p[[4]]*q[[3]], p[[1]]*q[[3]] - p[[2]]*q[[4]] + p[[3]]*q[[1]] + p[[4]]*q[[2]], p[[1]]*q[[4]] + p[[2]]*q[[3]] - p[[3]]*q[[2]] + p[[4]]*q[[1]]};

Conjugado de un Quaternión. **ConjugadoQ[p] = { p_0, - p_1, - p_2, - p_3 }** Rotación de un vector $v \in \Re^3$ con un Quaternión **p**, siendo $\lVert p \rVert = 1$. Según la definición (2.13). $$\rho(p,q) = \frac{1}{\lVert p \rVert^2} \bullet \left(p * q * \bar{p}\right), \quad \forall q \in Q$$ **Rotacion[p,v] = Tv[p * TvInv[v] * $\bar{p}$]** Rotación de $q \in Q$ con un Quaternión **p**, siendo $\lVert p \rVert = 1$. De acuerdo con la definición (2.13) y considerando que el primer elemento de **q** no cambia al rotar cuando $\lVert p \rVert = 1$.. **RotacionQ[p,q] = {q_0,0,0,0} + TvQ[p*q* $\bar{p}$]** Reflexión de un vector $v \in \Re^3$ con un Quaternión **p**, siendo $\lVert p \rVert = 1$. Según la definición (2.27). $$R(p,q) = \frac{1}{\lVert p \rVert^2} \bullet \left(\overline{p * q * \bar{p}}\right), \quad \forall q \in Q$$ **Reflexion[p,v] = Tv[$\overline{p * TvInv[v] * \bar{p}}$]** Reflexión de $q \in Q$ con un Quaternión **p**, siendo $\lVert p \rVert = 1$. De acuerdo con la definición (2.27) y considerando que el primer elemento de **q** no cambia al rotar cuando $\lVert p \rVert = 1$.. **ReflexionQ[p,q] =** **{q_0,0,0,0} + TvQ[$\overline{p * q * \bar{p}}$]**	(* Se define el conjugado de un Quaternión *) ConjugadoQ[p_]:={p[[1]],-p[[2]],-p[[3]],-p[[4]]}; (* Se define la Transformación Rotación de un vector tridimensional *) Rotacion[p_,v_]:=Tv[ProductoQ[p, ProductoQ[TvInv[v], ConjugadoQ[p]]]]; (* Se define la Transformación Rotación de un Quaternión *) RotacionQ[p_,q_]:={q[[1]],0,0,0} + TvQ[ProductoQ[p,ProductoQ[q,ConjugadoQ[p]]]]; (* Se define la Transformación Reflexión de un vector tridimensional *) Reflexion[p_,q_]:= Tv[ConjugadoQ[ProductoQ[p, ProductoQ[TvInv[q],ConjugadoQ[p]]]]]; (* Se define la Transformación Reflexión de un Quaternión *) ReflexionQ[p_,q_]:={q[[1]],0,0,0} + TvQ[ConjugadoQ[ProductoQ[p, ProductoQ[q,ConjugadoQ[p]]]]];
Se realizan definiciones iniciales necesarias, como son la base canónica de $\Re^3$ y las bases que servirán como bases locales en los eslabones. También se define la forma local de los Quaterniones que nos permitirán rotar, tanto en la posición no deformada como en la deformada.	**Definiciones iniciales** (* Base canónica de R3*) e1 = {1,0,0}; e2 = {0,1,0}; e3 = {0,0,1}; (* Bases locales de los eslabones *) (* Eslabón 1 *) e1 = {1,0,0}; e2 = {0,1,0}; e3 = {0,0,1};

	(* Eslabón 2 *) f1 = {1,0,0}; f2 = {0,1,0}; f3 = {0,0,1}; (* Eslabón 3 *) g1 = {1,0,0}; g2 = {0,1,0}; g3 = {0,0,1}; (*Quaterniones para rotar a posición no deformada*) P = {P0,P1,0,0}; (* Eslabón 2 *) Q = {Q0,Q1,0,0}; (* Eslabón 3 *) (* Quaterniones para rotar a posición deformada *) (* Para rotaciones *) r1 = {r10,0,0,r13}; (* Eslabón 1 *) r2 = {r20,r21,0,0}; (* Eslabón 2 *) r3 = {r30,r31,0,0}; (* Eslabón 3 *) (* Para rotaciones y reflexiones *) R1 = {R10,0,0,R13}; (* Eslabón 1 *) R2 = {R20,R21,0,0}; (* Eslabón 2 *) R3 = {R30,R31,0,0}; (* Eslabón 3 *)
En esta sección del programa se realiza la Modelación de la configuración no deformada del manipulador. Como la forma de las bases locales de los eslabones, en la configuración no deformada, se obtuvo en el capitulo anterior, entonces para facilitar la obtención del las ecuaciones del manipulador, simplemente obtenemos de manera directa la forma de las bases locales, que son: $\mathbf{f}_j^{I} = \rho(\mathbf{P}, \mathbf{f}_j)$ (Eslabón 2) $\mathbf{g}_j^{II} = \rho((\mathbf{P}^*\mathbf{Q}), \mathbf{g}_j)$ (Eslabón 3) Para lo anterior es necesario definir las normas unitarias de los Quaterniones P y Q, las cuales se almacenan en las variables "normas1" y "normas2", para aplicarlas en las ecuaciones resultantes de las rotaciones. Las variables que se utilizan para las bases locales de los eslabones son: ej = Primer eslabón. fj = Segundo eslabón. gj = Tercer eslabón.	**MODELADO DE LA CONFIGURACION NO DEFORMADA** **Modelo para la configuración no deformada** (* Se inicializan variables *) ClearAll[P0,P1,Q0,Q1,L1,L2,L3,b1,b2,b3]; (* Se definen variables para las normas unitarias de los Quaterniones P y Q *) normas1 ={P1->(1-P0^2)^(1/2),Q1->(1-0^2)^(1/2)}; normas2 ={(1-P0^2)^(1/2)->P1,(1-Q0^2)^(1/2)->Q1}; (* Se obtienen los productos de Quaterniones necesarios para rotar las bases *) PQ = ProductoQ[P,Q]; (* Se obtiene la forma de las bases locales de los eslabones en la configuración no deformada *) (* Eslabón 1 *) e1 = {1,0,0}; e2 = {0,1,0}; e3 = {0,0,1}; (* Eslabón 2 *) f11 = Expand[Expand[Rotacion[P,f1]]//.normas1] //.normas2; f21 = Expand[Expand[Rotacion[P,f2]]//.normas1] //.normas2; f31 = Expand[Expand[Rotacion[P,f3]]//.normas1]

	//.normas2; (* Eslabón 3 *) g12 = Expand[Expand[Rotacion[PQ,g1]]//.normas1] //.normas2; g22 = Expand[Expand[Rotacion[PQ,g2]]//.normas1] //.normas2; g32 = Expand[Expand[Rotacion[PQ,g3]]//.normas1] //.normas2;
Los eslabones del manipulador y la posición del elemento terminal son identificados, en la posición no deformada, por las variables siguientes: b1 = Eslabón 1. b2 = Eslabón 2. b3 = Eslabón 3. Rp = b1+b2+b3; Rp = Posición del elemento terminal. Las ecuaciones se simplifican utilizando las siguientes constantes.. A1 = P0*P1; A2 = 1 - 2*P0^2; A3 = Q0*Q1; A4 = 1 - 2*Q0^2;	(* Vectores que definen los eslabones del manipulador *) b1 = L1*e3; (* Eslabón 1 *) b2 = L2*f31; (* Eslabón 2 *) b3 = L3*g32; (* Eslabón 3 *) (* Constantes que se repiten y que servirán para simplificar las ecuaciones y optimizar operaciones *) constantes = {P0*P1->A1, P0^2->(1-A2)/2, Q0*Q1->A3, Q0^2->(1-A4)/2}; b2 = b2//.constantes; b3 = Expand[Expand[b3//.constantes]//.constantes]; (* Vector que define posición del elemento terminal*) Rp = b1+b2+b3;
Es esta parte se muestran las ecuaciones obtenidas en la sección anterior, las cuales se utilizarán para determinar los parámetros P y Q, que definen la configuración no deformada en una posición dada del elemento terminal, que será de donde arranque la simulación de la cinemática directa del manipulador en cuestión. Es decir, antes de generar una simulación computacional de la cinemática directa del manipulador de tres grados de libertad, debemos determinar la configuración no deformada realizando un proceso de cinemática inversa para calcular P y Q para una posición conocida del elemento terminal, la cual debe tener la forma siguiente: Rp = {0,Rpy, Rpz}.	**Ecuaciones optimizadas para la posición no deformada.** ClearAll[P0,P1,Q0,Q1] (* Ecuaciones optimizadas *) b1 = {0, 0, L1}; (* Eslabón 1 *) b2 = {0, -2*A1*L2, -A2*L2}; (* Eslabón 2 *) b3 = {0, 2*(A2*A3 + A1*A4)*L3, (* Eslabón 3 *) (A2*A4 - 4*A1*A3)*L3}; (* Posición el elemento terminal *) Rp = {0, 2*((A2*A3 + A1*A4)*L3 - A1*L2), L1 - A2*L2 + (A2*A4 - 4*A1*A3)*L3}; (* Constantes de las ecuaciones *) A1 = P0*P1; A2 = 1 - 2*P0^2; A3 = Q0*Q1; A4 = 1 - 2*Q0^2; (* Longitud de los eslabones del manipulador *) L1 = 6; L2 = 5; L3 = 4;

Se resuelve el sistema de ecuaciones mostrado en (5.1), para P0,P1,Q0,Q1, utilizando la función FindRoot, cuya sintaxis se puede consulta en el Apéndice, al igual que todas las funciones que se utilizan en este trabajo. Los resultados obtenidos se redondean si alguno de ellos está muy cercano a cero, finalmente se almacenan en las variables correspondientes (P0, P1, Q0, Q1) para poder ser utilizados posteriormente, además, se imprimen dichos valores.	**Cinemática Inversa para determinar la Configuración no Deformada** (* Conociendo la posición del elemento terminal se determinan los Quaterniones "P" y "Q" correspondientes *) ClearAll[P0,P1,Q0,Q1] (* Se da la posición del elemento terminal *) posicion = {0, 0, 15}; (* Se determinan "P" y "Q" utilizando los siguiente valores iniciales *) iP0 = .7071; iP1 = .7071; iQ0 = .866; iQ1 = .5; (* Se almacena la solución en la variable sol y se utiliza la función FindRoot para determinar la solución del sistema de ecuaciones *) sol = FindRoot[{Rp2 == posicion[[2]],Rp3 == posicion[[3]], P0^2+P1^2 == 1, Q0^2+Q1^2 == 1}, {P0,iP0}, {P1,iP1}, {Q0,iQ0}, {Q1,iQ1}, MaxIterations->50]; (* Se asignan los valores de la solución a las variables correspondientes *) P0=P0/.sol; P1=P1/.sol; Q0=Q0/.sol; Q1=Q1/.sol; (* Se redondean los valores de la solución *) If[Abs[P0]==1,P1=0,If[Abs[P1]==1,P0=0]]; If[Chop[Abs[Q0]-1]==0,Q0=1;Q1=0, If[Chop[Abs[Q1]-1]==0,Q0=0;Q1=1]]; (* Se imprimen los resultados *) Print["P0 = ",P0 ," ", "P1 = ", P1, " Q0 = ", Q0, " ", "Q1 = ", Q1]
De la misma forma que se realizó la modelación de la configuración no deformada, ahora se presenta el cálculo de la forma final de las bases locales de los eslabones, las cuales se obtuvieron en el capitulo anterior y son: $\mathbf{e}_j^{I} = \rho(\mathbf{r1}, \mathbf{e}_j)$ $\mathbf{f}_j^{III} = \rho(\mathbf{r1*r2*P}, \mathbf{f}_j)$	**MODELADO DE LA CONFIGURACION DEFORMADA** **Modelo para la configuración deformada con rotaciones.** ClearAll[r10,r13,r20,r21,r30,r31,L1,L2,L3]; (* Posición no deformada conocida *) P = {P0,P1,0,0}; Q = {Q0,Q1,0,0};

$g_j^V = \rho(r1 * r2 * r3 * P * Q, g_j)$ También en este caso, se definen las normas unitarias de los Quaterniones utilizados, almacenándolas en las variables "normas3" y "normas4", que incluyen las normas unitarias de los Quaterniones correspondientes a la configuración no deformada.	(* Quaterniones para rotar a la configuración deformada *) r1 = {r10,0,0,r13}; (* Rotación Eslabón 1*) r2 = {r20,r21,0,0}; (* Rotación Eslabón 2*) r3 = {r30,r31,0,0}; (* Rotación Eslabón 3*) (* Normas unitarias de los Quaterniones *) normas3 = Flatten[{normas1, r13->(1-r10^2)^(1/2), r21->(1-r20^2)^(1/2), r31->(1-r30^2)^(1/2), r43->(1-r40^2)^(1/2)}]; normas4 = Flatten[{normas2, (1-r10^2)^(1/2)->r13, (1-r20^2)^(1/2)->r21, (1-r30^2)^(1/2)->r31, (1-r40^2)^(1/2)->r43}]; (* Se obtienen los productos de Quaterniones necesarios para rotar las bases *) r1r2P = ProductoQ[r1,ProductoQ[r2,P]]; r1r2r3PQ = ProductoQ[r1,ProductoQ[r2, ProductoQ[r3,ProductoQ[P,Q]]]]; (* Se obtiene la forma final de las bases locales de los eslabones *) (* Base local del eslabón 1 *) e11 = Expand[Expand[Rotacion[r1,e1]]//.normas3] //.normas4; e21 = Expand[Expand[Rotacion[r1,e2]]//.normas3] //.normas4; e31 = Expand[Expand[Rotacion[r1,e3]]//.normas3] //.normas4; (* Base local del eslabón 2 *) f13 = Expand[Expand[Rotacion[r1r2P,f1]]//.normas3] //.normas4; f23 = Expand[Expand[Rotacion[r1r2P,f2]]//.normas3] //.normas4; f33 = Expand[Expand[Rotacion[r1r2P,f3]]//.normas3] //.normas4; (* Base local del eslabón 3 *) g15 = Expand[Expand[Rotacion[r1r2r3PQ,g1]] //.normas3]//.normas4; g25 = Expand[Expand[Rotacion[r1r2r3PQ,g2]] //.normas3]//.normas4; g35 = Expand[Expand[Rotacion[r1r2r3PQ,g3]] //.normas3]//.normas4;
Una vez obtenidas las bases locales en su forma final, se obtienen las ecuaciones que representan los eslabones del manipulador y la posición del elemento terminal como:	(* Vectores que definen los eslabones del manipulador en el modelo con rotaciones *) b1 = L1*e31; (* Eslabón 1 *) b2 = L2*f33; (* Eslabón 2 *) b3 = L3*g35; (* Eslabón 3 *)

$$b_1 = L_1{}^* e_3^{I} = L_1{}^*e31; \qquad (\textbf{Eslabón 1})$$

$$b_2 = L_2{}^* f_3^{III} = L_2{}^*f33; \qquad (\textbf{Eslabón 2})$$

$$b_3 = L_3{}^* g_3^{V} = L_3{}^*g35; \qquad (\textbf{Eslabón 3})$$

(Posición del elemento terminal)

$$r_p = b_1 + b_2 + b_3$$

Estas expresiones son simplificadas sustituyendo por constantes a aquellos términos que se repiten dentro de las diferentes ecuaciones, como se puede observar.

```
(* Constantes que se repiten y que servirán para
simplificar las ecuaciones y optimizar operaciones *)

constantes = {P0*P1->A1, P0^2->(1-A2)/2,
              Q0*Q1->A3, Q0^2->(1-A4)/2,
              r10*r13->A5, r10^2->(1-A6)/2,
              r20*r21->A7, r20^2->(1-A8)/2,
              r30*r31->A9, r30^2->(1-A10)/2,
         A2*A8->4*A1*A7+B1, A2*A7->B2-A1*A8,
        A10*A8->4*A7*A9+B3,  A10*A7->B4-A8*A9,
        A2*A4->4*A1*A3+B5, A2*A3->B6-A1*A4,
        B4*B5->C1 - B3*B6, B3*B5->C2+4*B4*B6};

b2 = Expand[b2//.constantes];
b2 = Expand[Collect[b2,{L2,A5,A6}]//.constantes];

b3 = Expand[b3//.constantes];
b3 = Collect[Collect[Collect[b3,{L3,A5,A1,A2,A3,A4,
      A6}]//.constantes, {L3,A5,A6,B3,B4}]
      //.constantes,{L3,A5,A6,B3,B4}];
b3 = Expand[b3//.constantes];

(*Vector que define posición del elemento terminal *)

        rp = Factor[b1 + b2 + b3];
```

De la misma forma que se realizó la modelación con puras rotaciones, ahora se determina la forma final de las bases locales de los eslabones, para la modelación con rotaciones y reflexiones, las cuales se obtuvieron en el capitulo anterior y son:

$$e_j^{I} = \rho(r1,\ e_j\ )$$

$$f_j^{III} = R(\, r1 * \overline{r2} * P\ ,\, f_j)$$

$$g_j^{V} = \rho(\, r1 * \overline{r2} * r3 * P * Q,\, g_j)$$

También en este caso, se definen las normas unitarias de los Quaterniones utilizados, almacenándolas en las variables "normas3" y "normas4", que incluyen las normas unitarias de los Quaterniones correspondientes a la configuración no deformada.

Modelo para la configuración deformada con rotaciones y reflexiones.

```
(* A las variables utilizadas exclusivamente para el
   modelo con rotaciones y las distinguiremos
   agregándoles al final una letra "R" *)

ClearAll[R10,R13,R20,R21,R30,R31,L1,L2,L3];

(* Posición no deformada conocida *)

P = {P0,P1,0,0}; Q = {Q0,Q1,0,0};

(* Quaterniones para rotar a la configuración
   deformada *)
     R1 = {R10,0,0,R13};  (* Rotación Eslabón 1*)
     R2 = {R20,R21,0,0};  (* Rotación Eslabón 2*)
     R3 = {R30,R31,0,0};  (* Rotación Eslabón 3*)

(* Normas unitarias de los Quaterniones *)

normas5 = Flatten[{normas1,
        R13->(1-R10^2)^(1/2),
        R21->(1-R20^2)^(1/2),
        R31->(1-R30^2)^(1/2)}];

normas6 = Flatten[{normas2,
        (1-R10^2)^(1/2)->R13,
```

```
        (1-R20^2)^(1/2)->R21,
        (1-R30^2)^(1/2)->R31}];

(* Se obtienen los productos de Quaterniones
   necesarios para rotar las bases *)
R1R2cP = ProductoQ[R1,ProductoQ[
            ConjugadoQ[R2],P]];
R1R2cR3PQ = ProductoQ[R1,ProductoQ[
   ConjugadoQ[R2],ProductoQ[R3,ProductoQ[P,Q]]]];

(* Se obtiene la forma final de las bases locales de
   los eslabones *)
(* Base local del eslabón 1 *)
e11R = Expand[Expand[Rotacion[R1,e1]]//.normas5]
         //.normas6;
e21R = Expand[Expand[Rotacion[R1,e2]]//.normas5]
         //.normas6;
e31R = Expand[Expand[Rotacion[R1,e3]]//.normas5]
         //.normas6;

(* Base local del eslabón 2 *)
f13R = Expand[Expand[Reflexion[R1R2cP,f1]]
         //.normas5]//.normas6;
f23R = Expand[Expand[Reflexion[R1R2cP,f2]]
         //.normas5]//.normas6;
f33R = Expand[Expand[Reflexion[R1R2cP,f3]]
         //.normas5]//.normas6;

(* Base local del eslabón 3 *)
g15R = Expand[Expand[Rotacion[R1R2cR3PQ,g1]]
          //.normas5]//.normas6;
g25R = Expand[Expand[Rotacion[R1R2cR3PQ,g2]]
          //.normas5]//.normas6;
g35R = Expand[Expand[Rotacion[R1R2cR3PQ,g3]]
          //.normas5]//.normas6;
```

Una vez obtenidas las bases locales en su forma final, se obtienen las ecuaciones que representan los eslabones del manipulador y la posición del elemento terminal como:

$$\mathbf{b_1R = L_1{*}\, e_3^{I} = L_1{*}e31R;} \quad \textbf{(Eslabón 1)}$$

$$\mathbf{b_2R = L_2{*}\, f_3^{III} = L_2{*}f33R;} \quad \textbf{(Eslabón 2)}$$

$$\mathbf{b_3R = L_3{*}\, g_3^{V} = L_3{*}g35R;} \quad \textbf{(Eslabón 3)}$$

(Posición del elemento terminal)

$$\mathbf{r_pR = b_1R + b_2R + b_3R}$$

Estas expresiones son simplificadas sustituyendo por constantes a aquellos términos que se repiten dentro de las

```
(* Vectores que definen los eslabones del
   manipulador en el modelo con rotaciones y
   reflexiones *)
        b1R = L1*e31R;   (* Eslabón 1 *)
        b2R = L2*f33R;   (* Eslabón 2 *)
        b3R = L3*g35R;   (* Eslabón 3 *)

(* Constantes que se repiten en las ecuaciones y
   que servirán para simplificar las ecuaciones y
   optimizar operaciones *)

constantes = {P0*P1->A1, P0^2->(1-A2)/2,
             Q0*Q1->A3, Q0^2->(1-A4)/2,
           R10*R13->A5R, R10^2->(1-A6R)/2,
           R20*R21->A7R, R20^2->(1-A8R)/2,
           R30*R31->A9R, R30^2->(1-A10R)/2,
             A2*A8R->-4*A1*A7R+B1R,
             A1*A8R->B2R+A2*A7R,
```

diferentes ecuaciones, como se puede observar.	A10R*A8R->B3R-4*A7R*A9R, A10R*A7R->B4R+A8R*A9R, A2*A3->B5R-A1*A4, A2*A4->B6R+4*A1*A3, B4R*B6R->C1R+B3R*B5R, B3R*B6R->C2R-4*B4R*B5R}; b2R = Expand[Expand[b2R//.constantes] //.constantes]; b3R = Collect[Expand[b3R//.constantes], {L3,A1,A2,A3,A4}]; b3R = Collect[Expand[b3R//.constantes], {L3,B3R,B4R,A7R}]; b3R = Expand[Expand[b3R//.constantes] //.constantes]; (* Vector que define posición del elemento terminal*) rpR = Factor[b1R + b2R + b3R];
En esta parte se presenta el resultado de la obtenido de las secciones anteriores, es decir, las ecuaciones simplificadas del manipulador, así como, las constantes utilizadas en dichas ecuaciones. Esta sección es la que propiamente se utiliza para realizar la simulación de la cinemática inversa del manipulador de tres grados de libertad, ya que es la que contiene las ecuaciones ya simplificadas. Puede observarse, que además de las ecuaciones de los eslabones y de la posición del elemento terminal, se tiene una ecuación para e11 = e_1^1 ; ésta sirve para mostrar el movimiento del primer eslabón, que al girar sobre su propio eje, requiere que se dibuje una de las bases perpendiculares al él para ver claramente su movimiento. Como al realizar la simulación es necesario evaluar numéricamente las ecuaciones del manipulador, se genera un respaldo de dichas ecuaciones en esta misma sección, almacenándolas en nuevas variables.	**Ecuaciones optimizadas para determinar configuración deformada** ClearAll[r10, r13, r20, r21, r30, r31, R10, R13, R20, R21, R30, R31]; (* Ecuaciones del manipulador *) (* Con Rotaciones *) (* Eslabón 1 *) b1 = {0, 0, L1}; (* Eslabón 2 *) b2 = {-4*A5*B2*L2, -2*A6*B2*L2, B1*L2}; (* Eslabón 3 *) b3 = {-4*A5*C1*L3, -2*A6*C1*L3, C2*L3}; (* Posición del elemento terminal *) rp = {-4*A5*(B2*L2 + C1*L3), -2*A6*(B2*L2 + C1*L3), L1 + B1*L2 + C2*L3}; (* Componente 1 de la Base local del eslabón 1 (para dibujar la base) *) e11 = {-A6, 2*A5, 0}; (* Con Reflexiones *) (* Eslabón 1 *) b1R = {0, 0, L1}; (* Eslabón 2 *) b2R = {4*A5R*B2R*L2, 2*A6R*B2R*L2, -B1R*L2}; (* Eslabón 3 *) b3R = {4*A5R*C1R*L3, 2*A6R*C1R*L3, C2R*L3};

	(* Posición del elemento terminal *) rpR = {4*A5R*(B2R*L2 + C1R*L3), 2*A6R*(B2R*L2 + C1R*L3), L1 - B1R*L2 + C2R*L3}; (* Componente 1 de la Base local del eslabón 1 (para dibujar la base) *) e11R = {-A6R, 2*A5R, 0}; (* Constantes de las ecuaciones *) A1 = P0*P1; A2 = 1-2*P0^2; A3 = Q0*Q1; A4 = 1-2*Q0^2; A5 = r10*r13; A6 = 1-2*r10^2; A7 = r20*r21; A8 = 1-2*r20^2; A9 = r30*r31; A10 = 1-2*r30^2; B1 = A2*A8 - 4*A1*A7; B2 = A2*A7 + A1*A8; B3 = A10*A8 - 4*A7*A9; B4 = A10*A7 + A8*A9; B5 = A2*A4 - 4*A1*A3; B6 = A2*A3 + A1*A4; C1 = B4*B5 + B3*B6; C2 = B3*B5 - 4*B4*B6; A5R = R10*R13; A6R = 1-2*R10^2; A7R = R20*R21; A8R = 1-2*R20^2; A9R = R30*R31; A10R = 1-2*R30^2; B1R = A2*A8R + 4*A1*A7R; B2R = A1*A8R - A2*A7R; B3R = A10R*A8R + 4*A7R*A9R; B4R = A10R*A7R - A8R*A9R; B5R = A2*A3 + A1*A4; B6R = A2*A4 - 4*A1*A3; C1R = B4R*B6R - B3R*B5R; C2R = B3R*B6R + 4*B4R*B5R; (* Longitud de los eslabones del manipulador *) L1 = 6; L2 = 5; L3 = 4; (* Respaldo de ecuaciones *) z1=b1; z2=b2; z3=b3; rx=rp; z1R=b1R; z2R=b2R; z3R=b3R; rxR=rpR;
En esta parte se presenta el cálculo de la cinemática inversa para diferentes trayectorias del elemento terminal, tanto para el modelo con rotaciones, como para el modelo con rotaciones y reflexiones.	**CINEMATICA INVERSA** **Solución de ecuaciones para diferentes trayectorias**
En esta parte se determinan los parámetros de rotación de los eslabones, para una sucesión de puntos que pertenecen a una línea recta, la cual se define por sus dos	**LINEAS RECTAS** (* Se inicializan variables *) ClearAll[r10,r13,r20,r21,r30,r31,"solucion"];

puntos extremos(S0, S1). Para lo anterior, primeramente recuperamos las ecuaciones del manipulador, las cuales se respaldaron en otras variables, (deberá resolverse la cinemática inversa para determinar la configuración no deformada y ejecutar la celda donde están las ecuaciones optimizadas para la configuración no deformada).

Para determinar la puntos sobre la trayectoria, utilizamos la forma paramétrica de una recta que pasa por dos puntos $p_1 = \{x_1, y_1, z_1\}$ y $p_2 = \{x_1, y_1, z_1\}$:

$$px = x_1 + (x_2 - x_1)t$$
$$py = y_1 + (y_2 - y_1)t$$
$$pz = z_1 + (z_2 - z_1)t$$

Con lo anterior se realiza un ciclo utilizando como variable de control del mismo, a la variable **t**, con lo que determinamos puntos sobre la trayectoria a seguir y resolvemos el sistema de ecuaciones (5.3), para el modelo con rotaciones.

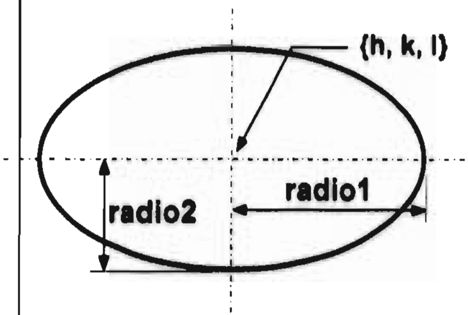
$$r_px = -4*A5*(B2*L_2 + C1*L_3),$$
$$r_py = -2*A6*(B2*L_2 + C1*L_3),$$
$$r_pz = L_1 + B1*L_2 + C2*L_3\}$$
$$r1_0^2 + r1_1^2 = 1$$
$$r2_0^2 + r2_1^2 = 1$$
$$r3_0^2 + r3_1^2 = 1$$

Este sistema de ecuaciones se resuelve utilizando la función Findroot usando los valores iniciales mostrados. Una vez que se resuelve el sistema de ecuaciones anterior, se verifica si la precisión de la solución es la adecuada, si es así, se fijan como valores iniciales, para resolver la cinemática inversa del siguiente punto, a la solución obtenida y se continua el ciclo, si la solución está fuera de la precisión se envia un mensaje y se restauran los valores iniciales continuando el ciclo hasta terminar. Después se realiza la solución del sistema de ecuaciones (5.5), con lo que se obtienen los parámetros de rotación

```
ClearAll[R10,R13,R20,R21,R30,R31];

(* Se recuperan las ecuaciones del manipulador *)
b1=z1; b2=z2; b3=z3; rp=rx; r p1=rp[[1]];
rp2=rp[[2]]; rp3=rp[[3]];
b1R=z1R; b2R=z2R; b3R=z3R; rpR=rxR;
rp1R=rpR[[1]];rp2R=rpR[[2]];rp3R=rpR[[3]];

(* Variables de control *)
t = 0; (* Para discretizar en varios puntos la
        trayectoria seleccionada *)
i = 1; (* Para guardar la solución de cada punto de la
        trayectoria *)

(* Valores iniciales para resolver el sistema de
   ecuaciones *)

ir10 = .7071; ir13 = .7071; ir20 = .3; ir21 = .9;
ir30 = 5; ir31 = .8;

(* Trayectoria en línea recta dada por los puntos
extremos *)

            S0 = {5,5,7};  S1 = {5,-5,7};

(* Se resuelve la cinemática inversa para el
   modelo con Rotaciones *)

(* Se inicia un ciclo para resolver el problema
   cinemático inverso en los puntos en que será
   discretizada la trayectoria *)

For[t=0, t<1, t+=.1,

(* Se determinan las coordenadas del punto
   correspondiente de la trayectoria*)
          px = S0[[1]]+(S1[[1]]-S0[[1]])*t;
          py = S0[[2]]+(S1[[2]]-S0[[2]])*t;
          pz = S0[[3]]+(S1[[3]]-S0[[3]])*t;

(* Solución del problema cinemático inverso para el
   punto seleccionado *)

solucion[i] = FindRoot[{rp1==px, rp2==py, rp3==pz,
           r10^2+r13^2 == 1, r20^2+r21^2 == 1,
           r30^2+r31^2 == 1}, {r10,ir10},{r13,ir13},
           {r20,ir20},{r21,ir21},{r30,ir30}, {r31,ir31},
           MaxIterations->50];

(* Se verifica la precisión de la solución *)
        e = 10^-5; (* Precisión requerida *)
If[Abs[px - (rp1/.solucion[i])]<e &&
  Abs[py - (rp2/.solucion[i])]<e &&
  Abs[pz - (rp3/.solucion[i])]<e,
```

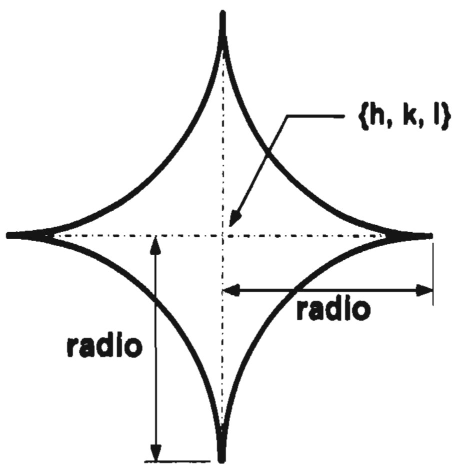
correspondientes a al modelo con rotaciones y reflexiones. $r_p x = 4*A5*(B2*L_2 + C1*L_3)$ $r_p y = 2*A6*(B2*L_2 + C1*L_3)$ $r_p z = L_1 - B1*L_2 + C2*L_3$ $R1_0^2 + R1_3^2 = 1$ $R2_0^2 + R2_1^2 = 1$ $R3_0^2 + R3_1^2 = 1$	(* Si la solución está en la precisión fijada, se definen como valores iniciales, para resolver la cinemática inversa del siguiente punto de la trayectoria a los valores de la solución obtenida, si no es así se restauran los valores iniciales utilizados *) ir10=r10/.solucion[i]; ir13=r13/.solucion[i]; ir20=r20/.solucion[i]; ir21=r21/.solucion[i]; ir30=r30/.solucion[i]; ir31=r31/.solucion[i], (* Se imprime un mensaje si la solución no está en la precisión definida *) Print["No se encontró solución para el punto: ", {px,py,pz}]; (* Se restauran valores iniciales para resolver las ecuaciones si no se encontró solución para el punto en cuestión *) ir10 = .7071; ir13 = .7071; ir20 = .3; ir21 = .9; ir30 = .5; ir31 = .8]; i++] **(* Se resuelve la cinemática inversa para el modelo con Reflexiones *)** (* Se inicia un ciclo para resolver el problema cinemático inverso en los puntos en que será discretizada la trayectoria *) i = 1; (* Se inicializa la variable de control i *) For[t=0, t<1, t+=.1, (* Se determinan las coordenadas del punto correspondiente de la trayectoria*) px = S0[[1]]+(S1[[1]]-S0[[1]])*t; py = S0[[2]]+(S1[[2]]-S0[[2]])*t; pz = S0[[3]]+(S1[[3]]-S0[[3]])*t; (* Solución del problema cinemático inverso para el punto seleccionado *) solucionR[i] = FindRoot[{rp1R==px, rp2R==py, rp3R==pz, R10^2+R13^2 == 1, R20^2+R21^2 == 1, R30^2+R31^2 == 1}, {R10,ir10},{R13,ir13},{R20,ir20}, {R21,ir21}, {R30,ir30},{R31,ir31}, MaxIterations->50]; (* Se verifica la precisión de la solución *)

	If[Abs[px - (rp1R/.solucion[i])]<e && Abs[py - (rp2R/.solucion[i])]<e && Abs[pz - (rp3R/.solucion[i])]<e, (* Si la solución está en la precisión fijada, se definen como valores iniciales, para resolver la cinemática inversa del siguiente punto de la trayectoria a los valores de la solución obtenida, si no es así se restauran los valores iniciales utilizados *) ir10=R10/.solucion[i]; ir13=R13/.solucion[i]; ir20=R20/.solucion[i]; ir21=R21/.solucion[i]; ir30=R30/.solucion[i]; ir31=R31/.solucion[i];, (* Se imprime un mensaje si la solución no está en la precisión definida *) Print["No se encontró solución para el punto: ", {px,py,pz}]; (* Se restauran valores iniciales para resolver las ecuaciones si no se encontró solución para el punto en cuestión *) ir10 = .7071; ir13 = .7071; ir20 = .3; ir21 = .9; ir30 = .5; ir31 =.8] i++]
En esta sección, al igual que en la anterior se resuelve la cinemática inversa para una secuencia de puntos sobre una trayectoria, para los dos modelos disponibles, que en este caso se trata de una elipse o circunferencia. El proceso que se sigue es el mismo que para la línea recta, solamente cambia la manera de determinar los puntos sobre la trayectoria, que en este caso se realiza utilizando la forma paramétrica de una elipse en el plano y-z, con centro en {h, k, l} con radios *radio1* y *radio2*. **px = h;** **py = radio1 * Cos[t] +k ;** **pz = radio2 * Sin[t] + l;**	**CIRCUNFERENCIAS O ELIPSES** (* Se inicializan variables *) ClearAll[r10,r13,r20,r21,r30,r31,"solucion"]; ClearAll[R10,R13,R20,R21,R30,R31]; (* Se recuperan las ecuaciones del manipulador *) b1=z1; b2=z2; b3=z3; rp=rx; r p1=rp[[1]]; rp2=rp[[2]]; rp3=rp[[3]]; b1R=z1R; b2R=z2R; b3R=z3R; rpR=rxR; rp1R=rpR[[1]];rp2R=rpR[[2]];rp3R=rpR[[3]]; (* Variables de control *) t = 0; (* Para discretizar en varios puntos la trayectoria seleccionada *) i = 1; (* Para guardar la solución de cada punto de la trayectoria *) (* Valores iniciales para resolver el sistema de ecuaciones *) ir10 = .7071; ir13 = .7071; ir20 = .3; ir21 = .9; ir30 = 5; ir31 = .8; (* Variables de control para definir la trayectoria *)




```
          radio1 = 7;    (* Radio uno de la elipse *)
          radio2 = 5;    (* Radio dos de la elipse *)
          h=5;k=0;l=6; (* Centro de la elipse *)

(* Se resuelve la cinemática inversa para el
   modelo con Rotaciones *)

(* Se inicia un ciclo para resolver el problema
   cinemático inverso en los puntos en que será
   discretizada la trayectoria *)

For[t=0, t<=2*N[Pi], t+=N[Pi]/9,

(* Se determinan las coordenadas del punto
   correspondiente de la trayectoria*)
            px = h;
            py = radio1*Cos[t] + k;
            pz = radio2*Sin[t] + l;

(* Solución del problema cinemático inverso para el
   punto seleccionado *)

solucion[i] = FindRoot[{rp1==px, rp2==py, rp3==pz,
               r10^2+r13^2 == 1, r20^2+r21^2 == 1,
               r30^2+r31^2 == 1}, {r10,ir10},{r13,ir13},
              {r20,ir20},{r21,ir21},{r30,ir30}, {r31,ir31},
               MaxIterations->50];

(* Se verifica la precisión de la solución *)
          e = 10^-5; (* Precisión requerida *)
If[Abs[px - (rp1/.solucion[i])]<e &&
  Abs[py - (rp2/.solucion[i])]<e &&
  Abs[pz - (rp3/.solucion[i])]<e,

(* Si la solución está en la precisión fijada, se
   definen como valores iniciales, para resolver la
   cinemática inversa del siguiente punto de la
   trayectoria a los valores de la solución obtenida, si
   no es así se restauran los valores iniciales
   utilizados *)

      ir10=r10/.solucion[i]; ir13=r13/.solucion[i];
      ir20=r20/.solucion[i]; ir21=r21/.solucion[i];
      ir30=r30/.solucion[i]; ir31=r31/.solucion[i],

(* Se imprime un mensaje si la solución no está en
   la precisión definida *)

Print["No se encontró solución para el punto: ",
      {px,py,pz}];

(* Se restauran valores iniciales para resolver las
   ecuaciones si no se encontró solución para el
   punto en cuestión *)
```

	ir10 = .7071; ir13 = .7071; ir20 = .3; ir21 = .9; ir30 = .5; ir31 = .8]; i++] **(* Se resuelve la cinemática inversa para el modelo con Reflexiones *)** (* Se inicia un ciclo para resolver el problema cinemático inverso en los puntos en que será discretizada la trayectoria *) i = 1; (* Se inicializa la variable de control i *) For[t=0, t<=2*N[Pi], t+=N[Pi]/9, (* Se determinan las coordenadas del punto correspondiente de la trayectoria*) px = h; py = radio1*Cos[t] + k; pz = radio2*Sin[t] + l; (* Solución del problema cinemático inverso para el punto seleccionado *) solucionR[i] = FindRoot[{rp1R==px, rp2R==py, rp3R==pz, R10^2+R13^2 == 1, R20^2+R21^2 == 1, R30^2+R31^2 == 1}, {R10,ir10},{R13,ir13},{R20,ir20}, {R21,ir21}, {R30,ir30},{R31,ir31}, MaxIterations->50]; (* Se verifica la precisión de la solución *) If[Abs[px - (rp1R/.solucion[i])]<e && Abs[py - (rp2R/.solucion[i])]<e && Abs[pz - (rp3R/.solucion[i])]<e, (* Si la solución está en la precisión fijada, se definen como valores iniciales, para resolver la cinemática inversa del siguiente punto de la trayectoria a los valores de la solución obtenida, si no es así se restauran los valores iniciales utilizados *) ir10=R10/.solucion[i]; ir13=R13/.solucion[i]; ir20=R20/.solucion[i]; ir21=R21/.solucion[i]; ir30=R30/.solucion[i]; ir31=R31/.solucion[i];, (* Se imprime un mensaje si la solución no está en la precisión definida *) Print["No se encontró solución para el punto: ", {px,py,pz}]; (* Se restauran valores iniciales para resolver las

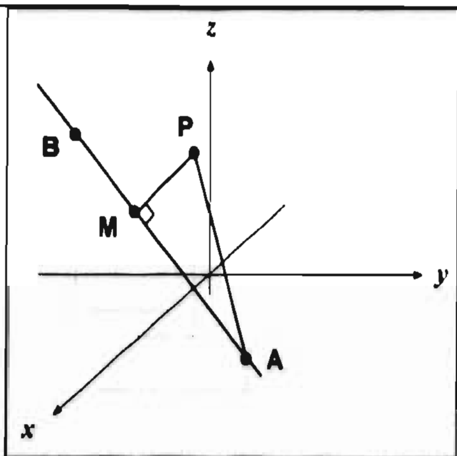
	ecuaciones si no se encontró solución para el punto en cuestión *) ir10 = .7071; ir13 = .7071; ir20 = .3; ir21 = .9; ir30 = .5; ir31 =.8] i++]
De la misma forma que se determinan los parámetros para las trayectorias anteriores, ahora se determinará una trayectoria en forma de ***astroide***, la cual se ubicará en el plano y-z, con centro en {h, k, l} cuya forma paramétrica es: **px = h;** **py = radio*Cos[t]^3 + k;** **pz = radio*Sin[t]^3 + l;**  	**ASTROIDE** (* Se inicializan variables *) ClearAll[r10,r13,r20,r21,r30,r31,"solucion"]; ClearAll[R10,R13,R20,R21,R30,R31]; (* Se recuperan las ecuaciones del manipulador *) b1=z1; b2=z2; b3=z3; rp=rx; r p1=rp[[1]]; rp2=rp[[2]]; rp3=rp[[3]]; b1R=z1R; b2R=z2R; b3R=z3R; rpR=rxR; rp1R=rpR[[1]];rp2R=rpR[[2]];rp3R=rpR[[3]]; (* Variables de control *) t = 0; (* Para discretizar en varios puntos la trayectoria seleccionada *) i = 1; (* Para guardar la solución de cada punto de la trayectoria *) (* Valores iniciales para resolver el sistema de ecuaciones *) ir10 = .7071; ir13 = .7071; ir20 = .3; ir21 = .9; ir30 = 5; ir31 = .8; (* Variables de control para definir la trayectoria *) radio = 6; (* Radio de la astroide *) h=5;k=0;l=7; (* Centro de la astroide *) **(* Se resuelve la cinemática inversa para el modelo con Rotaciones *)** (* Se inicia un ciclo para resolver el problema cinemático inverso en los puntos en que será discretizada la trayectoria *) For[t=0, t<=2*N[Pi], t+=N[Pi]/11, (* Se determinan las coordenadas del punto correspondiente de la trayectoria*) px = h; py = radio*Cos[t]^3 + k; pz = radio*Sin[t]^3 + l; (* Solución del problema cinemático inverso para el punto seleccionado *) solucion[i] = FindRoot[{rp1==px, rp2==py, rp3==pz, r10^2+r13^2 == 1, r20^2+r21^2 == 1,

```
               r30^2+r31^2 == 1}, {r10,ir10},{r13,ir13},
               {r20,ir20},{r21,ir21},{r30,ir30}, {r31,ir31},
               MaxIterations->50];
(* Se verifica la precisión de la solución *)

          e = 10^-5; (* Precisión requerida *)

If[Abs[px - (rp1/.solucion[i])]<e &&
   Abs[py - (rp2/.solucion[i])]<e &&
   Abs[pz - (rp3/.solucion[i])]<e,

(* Si la solución está en la precisión fijada, se
   definen como valores iniciales, para resolver la
   cinemática inversa del siguiente punto de la
   trayectoria a los valores de la solución obtenida, si
   no es asl se restauran los valores iniciales
   utilizados *)

        ir10=r10/.solucion[i]; ir13=r13/.solucion[i];
        ir20=r20/.solucion[i]; ir21=r21/.solucion[i];
        ir30=r30/.solucion[i]; ir31=r31/.solucion[i],

(* Se imprime un mensaje si la solución no está en
   la precisión definida *)

Print["No se encontró solución para el punto: ",
        {px,py,pz}];

(* Se restauran valores iniciales para resolver las
   ecuaciones si no se encontró solución para el
   punto en cuestión *)

ir10 = .7071; ir13 = .7071; ir20 = .3; ir21 = .9;
ir30 = .5; ir31 = .8];   i++ ]

(* Se resuelve la cinemática inversa para el
   modelo con Reflexiones *)

(* Se inicia un ciclo para resolver el problema
   cinemático inverso en los puntos en que será
   discretizada la trayectoria *)

i = 1; (* Se inicializa la variable de control i *)

For[t=0, t<=2*N[Pi], t+=N[Pi]/11,

(* Se determinan las coordenadas del punto
   correspondiente de la trayectoria*)
           px = h;
           py = radio*Cos[t]^3 + k;
           pz = radio*Sin[t]^3 + l;

(* Solución del problema cinemático inverso para el
   punto seleccionado *)
```

	solucionR[i] = FindRoot[{rp1R==px, rp2R==py, rp3R==pz, R10^2+R13^2 == 1, R20^2+R21^2 == 1, R30^2+R31^2 == 1}, {R10,ir10},{R13,ir13},{R20,ir20}, {R21,ir21}, {R30,ir30},{R31,ir31}, MaxIterations->50]; (* Se verifica la precisión de la solución *) If[Abs[px - (rp1R/.solucion[i])]<e && Abs[py - (rp2R/.solucion[i])]<e && Abs[pz - (rp3R/.solucion[i])]<e, (* Si la solución está en la precisión fijada, se definen como valores iniciales, para resolver la cinemática inversa del siguiente punto de la trayectoria a los valores de la solución obtenida, si no es así se restauran los valores iniciales utilizados *) ir10=R10/.solucion[i]; ir13=R13/.solucion[i]; ir20=R20/.solucion[i]; ir21=R21/.solucion[i]; ir30=R30/.solucion[i]; ir31=R31/.solucion[i];, (* Se imprime un mensaje si la solución no está en la precisión definida *) Print["No se encontró solución para el punto: ", {px,py,pz}]; (* Se restauran valores iniciales para resolver las ecuaciones si no se encontró solución para el punto en cuestión *) ir10 = .7071; ir13 = .7071; ir20 = .3; ir21 = .9; ir30 = .5; ir31 =.8] i++]
Una vez determinados los parámetros de rotación para la trayectoria seleccionada, ahora se generarán las gráficas que permitirán ver la simulación computacional de la cinemática inversa del manipulador de tres grados de libertad. Para poder realizarlo, recuperamos los valores de los parámetros almacenados en la variable solución[j] y los almacenamos en variables separadas para cada par de parámetros, se genera una gráfica que servirá como base para graficar el manipulador, la cual se almacena en las dos variables g1 y g2, posteriormente se inicia un ciclo, que se repetirá tantas veces como valores de los parámetros se hayan calculado para la trayectoria seleccionada	**Graficación para visualizar la simulación de la trayectoria seleccionada.** ClearAll[r10, r13, r20, r21, r30, r31, R10, R13, R20, R21, R30, R31] (* Se recuperan los parámetros de rotación almacenados en la variable solucion[j], obtenidos de resolver el problema cinemático inverso y se almacenan en variables para cada par de parámetros *) For[j=1, j<i, j++, solr1[j]={r10,r13}/.solucion[j]; solr2[j]={r20,r21}/.solucion[j]; solr3[j]={r30,r31}/.solucion[j]; solR1[j]={R10,R13}/.solucionR[j];

(variable "i"). En cada pasada del ciclo se asignan los valores correspondientes a los parámetros y con ellos se evalúan numéricamente los dos conjuntos de ecuaciones del manipulador, y con ellas se obtienen los puntos necesarios para graficar el manipulador en las dos configuraciones correspondientes a los dos modelos matemáticos. Para poder mostrar el efecto de la aplicación de los dos modelos mencionados en el proceso de evasión de obstáculos, se define el obstáculo, que será un cubo unitario ubicado de manera aleatoria dentro del área de trabajo del manipulador. Con el obstáculo definido, se debe determinar primeramente si para el modelo con puras rotaciones existe colisión entre uno de los eslabones del manipulador y el obstáculo, si no es el caso, se graficará el manipulador con dicho modelo, pero si existe colisión, se verificará si se produce colisión para el modelo con rotaciones y reflexiones, graficando el manipulador con este modelo y enviando, en su caso, un mensaje si aún no se pudo evadir el obstáculo con este cambio.
Para determinar si existe colisión entre el eslabón 2 o el 3 y el obstáculo, utilizamos el cálculo de la distancia más corta entre los puntos que representan las esquinas del cubo obstáculo y las líneas que representan los eslabones 2 y 3, consideramos que existe colisión cuando la distancia mencionada sea menor que la unidad. El cálculo de dicha distancia lo hacemos mediante el siguiente procedimiento: En la siguiente figura se muestra el punto **P** y una gráfica de una recta que pasa por **A** y **B**, el punto **M** es el pie de la perpendicular de **P** a la recta que pasa por **A** y **B**.

```
            solR2[j]={R20,R21}/.solucionR[j];
            solR3[j]={R30,R31}/.solucionR[j]];

g1=g2=SurfaceGraphics[{{0,0,0},{0,0,0},{0,0,0}},
   MeshRange->{{-(L2+L3),L2+L3},
               {-(L2+L3),L2+L3}},
   AxesLabel->{Eje X,Eje Y,Eje Z},
   DefaultFont->{"Negrita",12},
   PlotRange->{{-(L2+L3+.1),(L2+L3+.1)},
               {-(L2+L3+.1),(L2+L3+.1)},
               {(L1-(L2+L3)-.1),(L1+L2+L3+.1)}},
   BoxRatios->{1,1,1},Axes->True];

(* Se define aleatoriamente la posición del obstáculo
   *)

puntoi = {Random[Integer,{1,4}],
     Random[Integer,{-3,3}],Random[Integer,{7,10}]};

(* Se inicia un ciclo para generar las gráficas de
   cada punto de la trayectoria *)

For[j=1, j<i, j++,

(* Se asigna el valor de los parámetros para el punto
   de la trayectoria correspondiente almacenados en
   las variables solr?[j]*)

          r10=solr1[j][[1]];   r13=solr1[j][[2]];
          r20=solr2[j][[1]];   r21=solr2[j][[2]];
          r30=solr3[j][[1]];   r31=solr3[j][[2]];
          R10=solR1[j][[1]]; R13=solR1[j][[2]];
          R20=solR2[j][[1]]; R21=solR2[j][[2]];
          R30=solR3[j][[1]]; R31=solR3[j][[2]];

(* Se asignan valores a las ecuaciones del
   manipulador, según los valores de los
   parámetros, utilizando el respaldo de
   ecuaciones *)

base = N[e11]; b1 = N[z1]; b2 = N[z2]; b3 = N[z3];
rp=N[rx];
baseR = N[e11R]; b1R = N[z1R]; b2R = N[z2R];
b3R = N[z3R]; rpR = N[rxR];

(* Longitud del vector que servirá de base del
   manipulador *)

        a = 3;

(* Se definen los puntos en el espacio tridimensional
   para graficar el manipulador tanto para rotaciones
   como para reflexiones *)
```



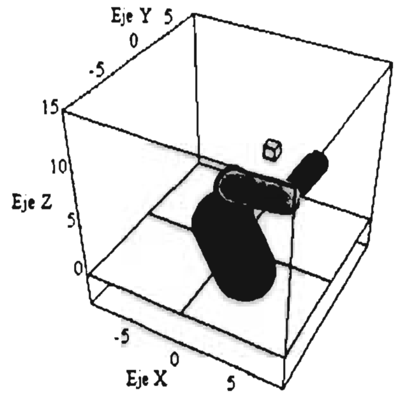
Por lo anterior, la distancia más corta del punto P a la recta mostrada, es la magnitud del vector $\overline{PM}$. Para determinar $|\overline{PM}|$, debemos encontrar $|\overline{AM}|$ y $|\overline{AP}|$ y aplicar el teorema de Pitágoras para determinar la distancia buscada. Para determinar $|\overline{AP}|$ lo hacemos simplemente determinado el vector $\overline{AP}$ por diferencia de coordenadas entre los puntos P y A. Para determinar $|\overline{AM}|$, encontramos la proyección escalar de $\overline{AP}$ sobre $\overline{AB}$ de la siguiente forma; [8].

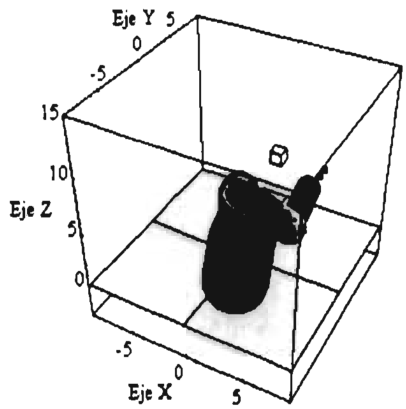
$$|\overline{AM}| = \frac{\overline{AP} \bullet \overline{AB}}{|\overline{AB}|}$$

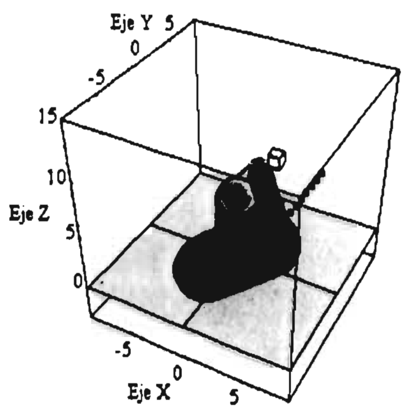
```
(* Modelo con Rotaciones*)
punto1 = {0,0,0};
punto2 = N[a*base];
punto3 = N[-a*base];
punto4 = N[b1];
punto5 = N[punto4 + b2];
punto6 = N[punto5 + b3];

(* Modelo con Reflexiones*)
punto1R = {0,0,0};
punto2R = N[a*baseR];
punto3R = N[-a*baseR];
punto4R = N[b1R];
punto5R = N[punto4R + b2R];
punto6R = N[punto5R + b3R];

(* Se generan los puntos que representan un cubo
   unitario, que será el obstáculo a evadir y que
   servirán para saber si hay colisión con el eslabón
   2 o 3 *)

For[k=1, k<9, k++,

   Switch[k, 1,  punto = puntoi,
             2,  punto = punto + { 0, 0, 1},
             3,  punto = punto + { 0, 1,-1},
             4,  punto = punto + { 0, 0, 1},
             5,  punto = punto + { 1,-1, 0},
             6,  punto = punto + { 0, 0,-1},
             7,  punto = punto + { 0, 1, 0},
             8,  punto = puntof = punto + { 0, 0, 1} ];

(* Se determina si hay colisión con el obstáculo *)

 (* Se inicializa variable para control de la colisión *)

   colision=0;

(* Eslabón 2, modelo con rotaciones *)

AB = b2;           (* Vector AB *)
AP = punto-b1;  (* Vector AP *)
 (* Magnitud del vector AP *)
nAP = Sqrt[AP[[1]]^2+AP[[2]]^2+AP[[3]]^2];
 (* Magnitud del vector AB *)
nAB = Sqrt[AB[[1]]^2+AB[[2]]^2+AB[[3]]^2];
 (* Magnitud del vector AM *)
AM = (AP.AB)/nAB;
 (* Distancia más corta del punto analizado al
    eslabón 2 *)
PM1 = Sqrt[nAP^2-AM^2];
(* Se verifica si el resultado es real, si es imaginario
   significa que la distancia es cero *)
If[Im[PM1]!=0, PM1=0];
```

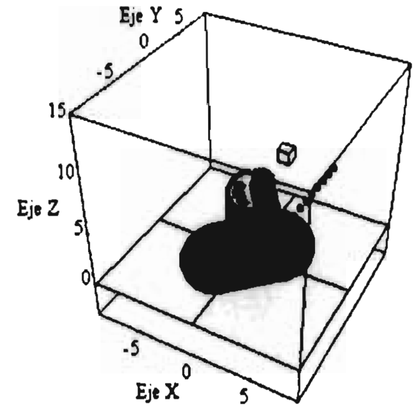
```
(* Eslabón 3, modelo con rotaciones *)

AB = b3;
AP = punto-(b1+b2);
nAP = Sqrt[AP[[1]]^2+AP[[2]]^2+AP[[3]]^2];
nAB = Sqrt[AB[[1]]^2+AB[[2]]^2+AB[[3]]^2];
AM = (AP.AB)/nAB;
PM2 = Sqrt[nAP^2-AM^2];
If[Im[PM2]!=0, PM2=0];

(* Si la distancia calculada para cualquiera de los
   dos eslabones fue menor a 1, la variable colision
   se iguala a 1 y se verifica ahora las distancias a
   los eslabones 2 y 3 para el modelo con
   reflexiones *)

If[PM1<=1 || PM2<=1, colision=1;

(* Eslabón 2, modelo con reflexiones *)

AB = b2R;
AP = punto-b1R;
nAP = Sqrt[AP[[1]]^2+AP[[2]]^2+AP[[3]]^2];
nAB = Sqrt[AB[[1]]^2+AB[[2]]^2+AB[[3]]^2];
AM = (AP.AB)/nAB;
PM3 = Sqrt[nAP^2-AM^2];
If[Im[PM3]!=0, PM3=0];

(* Eslabón 3, modelo con reflexiones *)

AB = b3R;
AP = punto-(b1R+b2R);
nAP = Sqrt[AP[[1]]^2+AP[[2]]^2+AP[[3]]^2];
nAB = Sqrt[AB[[1]]^2+AB[[2]]^2+AB[[3]]^2];
AM = (AP.AB)/nAB;
PM4 = Sqrt[nAP^2-AM^2];
If[Im[PM4]!=0, PM4=0];

(* Si cualquiera de las distancias calculadas es
   menor a 1, se asigna a la variable colision el valor
   de 2 *)

If[PM3<=1 || PM4<=1, colision=2]];   ];

(* Se agrega a la gráfica base (g2) el obstáculo y la
   posición del elemento terminal *)

g2 = {Graphics3D[{{Cuboid[puntoi,puntof]},
                  {PointSize[0.01],Point[rp]},
                  {PointSize[0.01],Point[rpR]}}],g2};

(* Se genera la gráfica para manipulador con
   Rotaciones, si no hay colisión con el obstáculo
```

	para este modelo (colisión = 0) *) If[colision==0, g3 = Graphics3D[{ (* Base rotatoria *) {AbsoluteThickness[40],RGBColor[0,.5,0], Line[{punto2,punto3}]}, (* Eslabón 1 *) {AbsoluteThickness[25],RGBColor[0,1,0], Line[{punto1,punto4}]}, (* Eslabón 2 *) {AbsoluteThickness[20],RGBColor[1,0,0], Line[{punto4,punto5}]}, (* Eslabón 3 *) {AbsoluteThickness[15],RGBColor[0,0,1], Line[{punto5,punto6}]}}];, (* Se genera la gráfica para manipulador con Reflexiones, si hay colisión con el obstáculo para el modelo con Rotaciones (colisión > 0), y en su caso, si tampoco se pudo evadir el obstáculo con el modelo con Reflexiones, se envía un mensaje indicándolo (colision > 1) *) g3 = Graphics3D[{ (* Mensaje a emitir si no se pudo evadir el obstáculo *) If[colision==2, {Text["No se pudo evadir el obstáculo", {0,0,15}]},{}], (* Base rotatoria *) {AbsoluteThickness[40],RGBColor[0,.5,0], Line[{punto2R,punto3R}]}, (* Eslabón 1 *) {AbsoluteThickness[25],RGBColor[0,1,0], Line[{punto1R,punto4R}]}, (* Eslabón 2 *) {AbsoluteThickness[20],RGBColor[1,0,0], Line[{punto4R,punto5R}]}, (* Eslabón 3 *) {AbsoluteThickness[15],RGBColor[0,0,1], Line[{punto5R,punto6R}]}}]]; (* Se envían a pantalla las gráficas g1, g2 y g3, combinadas *) Show[g1,g3,g2,Axes->True];]


Eje Y
5
0
-5
15
10
Eje Z
5
0
-5
0
Eje X
5


Eje Y
5
0
-5
15
10
Eje Z
5
0
-5
0
Eje X
5


Eje Y
5
0
-5
15
10
Eje Z
5
0
-5
0
Eje X
5


Eje Y
5
0
-5
15
10
Eje Z
5
0
-5
0
Eje X
5

CONCLUSIONES

En l proceso de realización del presente trabajo se obtuvieron las siguientes conclusiones:

⊗ La aplicación del Lenguaje de Cálculo Formal Mathematica para efectuar demostraciones formales simbólicas, permite disminuir drásticamente el tiempo utilizado y los errores que se producen al realizar dichas demostraciones manualmente.

⊗ Las secuencias de movimientos, en el proceso de modelación, no afecta la forma final de las ecuaciones resultantes, pero se presentan diferentes configuraciones intermedias en cada secuencia.

⊗ Las secuencias de movimientos pueden facilitar o dificultar la obtención de las ecuaciones cinemáticas finales, ya que en unas secuencias se requiere actualizar ejes de rotación en los Quaterniones correspondientes, lo que provoca que las ecuaciones resultantes sean más complejas y deban ser simplificadas utilizando propiedades de las operaciones con Quaterniones, mientras que en otras secuencias de movimientos no se requiere actualizar ejes de rotación, lo que permite que las ecuaciones obtenidas ya tengan su forma final simplificada.

⊗ La secuencia de movimientos más eficiente para modelar la cinemática de cadenas abiertas con Quaterniones, es iniciando con el eslabón terminal y finalizando con el eslabón base.

⊗ La forma general de la rotación de la base local de un eslabón cualquiera, en la configuración deformada de una cadena abierta con juntas rotacionales, es:

$$\mathbf{e}^{n}_{j} = \rho(\mathbf{r}_1 \cdot \mathbf{r}_2{}^{*} \ldots {}^{*}\mathbf{r}_n \, , \, \mathbf{e}^{x}_{j})$$

Donde:

$\mathbf{e}^{n}_{j}$ = Base local del eslabón n.

$\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_n$ = Quaterniones que rotan los eslabones 1, 2, ... n. Siendo el eslabón base el número uno, y el eslabón n, el eslabón analizado.

$\mathbf{e}^{x}_{j}$ = Base local del eslabón n en la configuración no deformada.

n = Número de eslabón analizado.

La numeración de los eslabones se realiza empezando en el eslabón base, es decir en el que se soporta todo el mecanismo, y se continua secuencialmente hasta finalizar en el eslabón terminal.

⊗ Para modelar una traslación (junta prismática) con el álgebra de Quaterniones, basta con ubicar una base local en la junta, de la misma forma que se hace para las juntas rotacionales, y utilizar la longitud del eslabón correspondiente como una variable la cual nos permitirá definir la traslación deseada.

⊗ La combinación de rotaciones y reflexiones nos permite obtener un modelo cinemático deferente al que se obtiene utilizando solamente rotaciones, la aplicación conjunta de estos dos modelos para un manipulador de tres grados de libertad, permite que el manipulador logre evadir, en determinados casos, un obstáculo que impide su movimiento cuando se realiza un proceso de cinemática inversa.

REFERENCIAS

[1] Reyes Avila Luis, 1990. Quaternions: Une Representation Parametrique Systematique Des Rotaions Finies.
Partie 1: Le Cadre Theorique.
Rapport de Recherche, INRIA No. 1303 - Rocquencourt, France.

[2] Reyes Avila Luis, 1990. Quaternions: Une Representation Parametrique Systematique Des Rotaions Finies.
Partie 2: Quelques Aplications.
Rapport de Recherche, INRIA No. 1454 - Rocquencourt, France.

[3] Stephen Wolfram, 1992. Mathematica a System for Doing Mathematics by Computer.
Addison-Weley Publishing Company, Inc..

[4] Cameron Smith and Nacy Blachman, 1995. The Mathematica Graphics Guidebook.
Addison-Wesley Publishing Company, Inc.

[5] Mark W. Spong and M. Vidyasagar, 1989. Robot Dynamics and Control.
John Wiley & Sons.

[6] Ilan Vardi, 1991. Computational Recreations in Mathematica.
Addison-Wesley Publishing Company, Inc.

[7] Stephen Wolfram, 1992. Mathematica Reference Guide.
Addison-Wesley Publishing Company, Inc..

[8] Louis Leithold, 1973. El cálculo con geometría analítica.
HARLA.

APENDICE A

LENGUAJE DE CALCULO FORMAL MATHEMATICA®.

En este apéndice se presentará el uso y sintaxis de los comandos de Mathematica utilizados en el desarrollo del presente trabajo. La primera versión de Mathematica se generó en 1988, se dice ser un sistema para hacer matemáticas en computadora. Combina manipulación simbólica, numérica, gráficos sobresalientes, y un idioma de programación sofisticado. A causa de su versatilidad, Mathematica se ha establecido como el sistema de álgebra en computadora elegido por muchos usuarios de la computadora. De los aproximadamente 100,000 usuarios de Mathematica, 28% son ingenieros, 21% son científicos en computación, 20% son científicos físicos, 12% son científicos matemáticos, y el 12% restante son hombres de negocios y científicos sociales. Dos terceras partes de los usuarios están en la industria y el gobierno, y solamente el 8% son estudiantes, este último porcentaje va en aumento. De cualquier modo, debido a su especial naturaleza y sofisticación, los usuarios principiantes necesitan conocer la sintaxis especial requerida para hacer que Mathematica ejecute la instrucciones en la manera que se desea. El propósito de este apéndice es servir como una guía a usuarios principiantes de Mathematica, para que logren comprender cabalmente los programas presentados en este trabajo. A continuación se presentan algunas reglas básicas en la sintaxis general en Mathematica.

Cinco Reglas Básicas de la sintaxis en Mathematica:

1.- Los argumentos de las funciones son dados entre paréntesis cuadrados **[.]**, paréntesis redondos **(.)** se usan para agrupar operaciones; vectores, matrices, y las listas son dadas entre llaves **{.}**; los paréntesis cuadrados dobles **[[.]]** se usan para indexación de listas y tablas, los paréntesis con asterisco **(*** *comentario* ***)** sirven para especificar comentarios.

2.- Los nombres de las funciones propias de Mathematica tienen su primera letra mayúscula; si un nombre de función consta de dos o más palabras, la primera letra de cada palabra será mayúscula sin dejar espacios intermedios..

3.- La multiplicación es representada por un espacio o por *.

4.- Las potencias son denotadas por ^.

5.- Si no se consigue respuesta u obtiene una contestación incorrecta, puede deberse a una entrada o ejecución incorrecta de un comando. En algunos casos la cantidad de memoria asignada a Mathematica puede causar una caída del sistema; como algunas personas, Mathematica no es perfecto y algunos errores pueden ocurrir.

Ahora se mostraran los comandos, funciones y operaciones utilizadas, presentando una breve descripción, su sintaxis, y en algunos casos, un ejemplo.

- **-> (Reglas de remplazo).-** Una regla de remplazo, es una relación entre dos expresiones que son equivalentes, y que en un momento dado, se desea utilizar para remplazar una por la otra. Las reglas de remplazo se definen como: **expr1 -> expr2**, que significa que la expr1 se remplazará por expr2. Muchas de las funciones propias de Mathematica utilizan opciones, las cuales deben ser expresadas como reglas de remplazo.

■

- ***//.* (Remplazo de expresiones).-** Cuando se requiere sustituir dentro de una expresión, cualquier variable, utilizamos al final de la expresión el símbolo ***//.*** seguido por la lista de reglas de remplazo. Por ejemplo, si se desea remplazar el término (a^2+b^2) en la expresión (a^2+b^2)x + (a^2+b^2)y, se realiza lo siguiente: (a^2+b^2)x + (a^2+b^2)y*//.*{(a^2+b^2)->c}, con lo que se obtiene como resultado cx + cy.

■

- **AddTo.-** ***x* += *dx*** suma ***dx*** **a** ***x*** y regresa el nuevo valor de ***x***.

■

- **ClearAll[***symb1,symb2, ...* **].-** Limpia todos los valores, definiciones, atributos, mensajes y asociaciones con los símbolos *symboli*. ClearAll[*"form1", "form2", ...*] limpia todos los símbolos cuyos nombres coinciden con cualquiera de las *formi*.

■

- **Collect[***expr, {x1,x2, ... }***].-** Junta los términos que contienen la variable *x1, x2, ...* elevada a la misma potencia de *xi*. Ejemplos:

  Collect[*x + n x + m, x*] ---> *m + (1 + n) x*;
  Collect[Expand[*(1+x+y)^3*], *x*] ---> *1 + x + 3 y + 3 y + y + x (3 + 3 y) + x (3 + 6 y + 3 y)*.

■

- **Definición de Funciones.-** Mathematica posee una gran cantidad de funciones propias, para realizar muchas operaciones, pero es común que el usuario requiera definir sus propias funciones. Por lo anterior mostraremos mediante un ejemplo la definición de una función, que para mayor compresión simplemente construiremos la función para determinar el producto escalar en $\Re^3$. Si tenemos dos vectores $\mathbf{a} = \{a_x, a_y, a_z\}$ y $\mathbf{b} = \{b_x, b_y, b_z\}$, sabemos que su producto escalar será: $a \bullet b = a_x{}^*b_x + a_y{}^*b_y + a_z{}^*b_z$, esto se programará en una función que llamaremos *ProductoEscalarR3*, como se muestra a continuación.

  ProductoEscalarR3[a_,b_]:= a[[1]]*b[[1]] + a[[2]]*b[[2]] + a[[3]]*b[[3]];

  En este caso la función tiene dos argumentos, que como se observa, deben encerrarse entre paréntesis cuadrados, y además, deben tener a continuación del nombre del parámetro un guión como se muestra. El signo **:=** se utiliza para asignar a las función, las operaciones que se realizaran con los parámetros, que en este caso, es un producto escalar entre dos vectores. Finalmente el punto y coma ";" sirve para separar dos sentencias, o también, para evitar que una operación genere una salida.

  Ejemplo de la ejecución en Mathematica de la función construida:

```
In[1] :=

     (* Se define la función *)

     ProductoEscalarR3[a_,b_]:= a[[1]]*b[[1]] + a[[2]]*b[[2]] + a[[3]]*b[[3]];
     (* Se asignan componentes arbitrarias a dos vectores cualesquiera *)

     x = {2,3,1};  y = {1,2,5};

     (* Se aplica el producto *)
```

```
ProductoEscalarR3[x,y]

Out[6] :=

13
```

■

En este trabajo se definieron las siguientes funciones:

◊ **ConjugadoQ[p].-** Obtiene el Conjugado del Quaternión p, siendo $\bar{\mathbf{p}} = \{\mathbf{a}, -\mathbf{b}, -\mathbf{c}, -\mathbf{d}\}$ el conjugado de $\mathbf{p} = \{\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}\}$.

◊ **Cruz[a,b].-** Realiza el producto cruz de los vectores $\mathbf{a},\mathbf{b} \in \Re^3$.

◊ **ProductoQ[p,q].-** Ejecuta la multiplicación de dos Quaterniones **p** y **q**, según la definición dada en la ecuación (2.2).

◊ **Reflexion[p,v].-** Obtiene la reflexión del vector $\mathbf{v} \in \Re^3$, con el Quaternión **p** como parámetro. Para realizarla, primeramente se transforma el vector **v** a forma de Quaternión aplicándole TvInv[v] y se aplica la definición de reflexión (2.27), finalmente se transforma el Quaternión resultante de la rotación a su forma vectorial con la aplicación de Tv[], con lo que obtenemos el vector v en su nueva posición. Se considera que la norma del Quaternión **p** es unitaria.

◊ **ReflexionQ[p,q].-** Realiza la reflexión del Quaternión **q** con el Quaternión **p** como parámetro, utilizando la definición de reflexión (2.27) y sabiendo que al rotar un Quaternión solamente cambia su parte vectorial, conservando constante su parte escalar. Se considera que la norma del Quaternión **p** es unitaria.

◊ **Rotacion[p,v].-** Realiza la rotación del vector $\mathbf{v} \in \Re^3$, con el Quaternión **p** como parámetro de rotación. Para realizarla, primeramente se transforma el vector **v** a forma de Quaternión aplicándole TvInv[v] y se aplica la definición de rotación (2.13), finalmente se transforma el Quaternión resultante de la rotación a su forma vectorial con la aplicación de Tv[], con lo que obtenemos el vector **v** en su nueva posición. Se considera que la norma del Quaternión **p** es unitaria.

◊ **RotacionQ[p,q].-** Rota el Quaternión **q** con el Quaternión **p** como parámetro de rotación, utilizando la definición de rotación (2.13) y sabiendo que al rotar un Quaternión solamente cambia su parte vectorial, conservando constante su parte escalar. Se considera que la norma del Quaternión **p** es unitaria.

◊ **Tv[q].-** Transforma el Quaternión **q** en un vector de $\Re^3$. Es decir, si se aplica Tv[] al Quaternión $p = \{p_0, p_1, p_2, p_3\}$, se obtiene la parte vectorial de p, o sea:

$$\mathbf{p_v} = \{\mathbf{p_1}, \mathbf{p_2}, \mathbf{p_3}\} = \mathbf{Tv[p]}$$

◊ **TvInv[q].-** Transforma un vector de $\Re^3$ en un Quaternión agregando un cero como primer elemento del Quaternión.

◊ **TvQ[q].-** Transformación del Quaternión **q** en un Quaternión vectorial. Es decir, reemplaza el primer elemento del Quaternión (parte escalar) por cero.

■

- **Expand[*expr*].-** Esta función ejecuta los productos y potencias enteras positivas, así como, las sumas en *expr*, hasta que todos los términos son monomiales, agrupando finalmente términos similares. Expand[] solo trabaja sobre potencias enteras positivas. Por ejemplo, si aplicamos está función a la expresión $(x + y) \wedge 2 + 7 (3 + x) (x + y)$, obtenemos como resultado $21x + 8x^2 + 21y + 9xy + y^2$.

■

- **Factor[*poly*].-** Factoriza un polinomio sobre los enteros. Factor trabaja solamente con enteros exactos o coeficientes racionales, no con números reales. Factor[*expr, Trig -> True*] trata las funciones trigonométricas como funciones racionales de exponenciales, y las factoriza. Ejemplo: Factor[*Sin[2x], Trig -> True*] ---> *2 Cos[x] Sin[x]*.

■

- **FindRoot[*{eqn1,eqn2, ... }, {x, x0}, {y, y0}, ...*].-** Realiza la búsqueda de una solución numérica para las ecuaciones simultáneas *eqni*. FindRoot regresa una lista reglas de remplazo para x, y, ... en la misma forma que se obtiene de Solve. Si solamente se especifica un valor de inicio de x, FindRoot busca una solución usando el método de Newton. Si se especifican dos valores iniciales, FindRoot usa una variante del método de la secante. Una de las opciones que se pueden especificar es, **MaxIterations** que sirve para indicar el número de iteraciones que deberán realizarse. Ejemplo:

In[1] :=

```
(* Se resuelve numéricamente el siguiente sistema de ecuaciones *)

ec1 = 6*x + 8*y^2 == 30;    (* Ecuación 1 *)
ec2 = 15*x^2 - 5*y == 40;   (* Ecuación 2 *)

(* Utilizando el método de Newton, se aplica la opción MaxIterations para indicar que
   deseamos que se realicen 50 iteraciones, si no se indica se realizan 15 iteraciones *)

FindRoot[{ec1,ec2},{x,1},{y,1},MaxIterations->50]

(* Utilizando el método de la secante *)

FindRoot[{ec1,ec2},{x,1,2},{y,1,2},MaxIterations->50]
```

Out[5] :=

```
{x -> 1.78447, y -> 1.55295}
```

Out[7] :=

```
{x -> 1.78447, y -> 1.55295}
```

■

- **Flatten[*list*].-** Elimina los anidamientos de la lista *list*. Ejemplos:

  Flatten[*{a, {b, c}, {d}}*] ---> *{a, b, c, d}*.

■

- **For[*start, test, incr, body*].-** (Ciclo For) Ejecuta *start*, después evalúa repetidamente *body* y *incr* hasta que la comparación *test* sea falsa.

■

- **Graphics3D[***primitives, options***].-** Representa una imagen gráfica en tres dimensiones. Graphics3D despliega su contenido usando Show. Las siguientes tipos de gráficos *primitives* se pueden usar.

Cuboide	Cuboid[*{xmin, ymin, zmin}, ...*]
Linea	Line[*{{x1,y1,z1}, ... }*]
punto	Point[*{x, y, z}*]
Polígono	Polygon[*{{x1,y1,z1}, ... }*]
Texto	Text[*expr, {x, y, z}*]

Algunos de los modificadores para los gráficos que se utilizan son:

AbsoluteThickness[d]	Espesor absoluto de la línea.
PointSize[r]	Tamaño del punto especificado.
RGBColor[r, g, b]	Color del gráfico

Las siguientes opciones se pueden especificar.

Opción	**Default**	**Descripción**
Axes	False	Si dibuja ejes.
AxesLabel	None	Etiquetas de los ejes
BoxRatios	Automatic	Relación entre los lados de la caja 3D. que rodea el gráfico
DefaultFont	$DefaultFont	Tipo de letra a utilizar por defecto
PlotRange	Automatic	Rango de los valores a graficar

Un ejemplo se muestra en la Función *Show*.

■

- **If[***condition, t, f* **].-** Devuelve *t* si la condición evaluada (*condition*) es verdadera, y *f* si es falsa. If[*condition, t, f, u*] regresa *u* si la condición evaluada no es ni verdadera ni falsa. **If** solamente evalúa el argumento determinado por el valor de la condición.

■

- **Im[z].-** Proporciona la parte imaginaria de el número complejo *z*. Im[*expr*] se deja sin evaluar si *expr* no es numérica.

■

- **Increment[** *x* **] or** ***x++*.-** Incrementa el valor de x en una unidad y regresa el valor de x antes del incremento.

■

- **Insert[***list, elem, n***].-** Inserta *elem* en la posición *n* en la lista *list*. Si *n* es negativo, la posición es contada desde el fin de la lista. Insert[*list, elem, {i, j,...}*] inserta *elem* en la posición *{i, j, ...}* en *list*. Ejemplos:

  Insert[*{a, b, c}, x, 2*] ---> *{a, x, b, c}*.
  Insert[*{a, b, c}, x, {{-1}, {1}}*] ---> *{x, a, b, c, x}*.
  Insert[*{{a, b}, {c, d}}, x, {2, 1}*] ---> *{{a, b}, {x, c, d}}*.

■

- **List[***e1,e2, ...* **] = {***e1,e2, ...* **}.-** Es una lista de elementos. Las listas son objetos muy generales que representan colecciones de expresiones. {a, b, c} representa un vector de $\Re^3$, {{a, b}, {c, d}} representa una matriz 2x2. Las listas pueden ser utilizadas también para representar tensores. Para hacer referencia a un elemento de una lista se utilizan los paréntesis cuadrados, ejemplos:

$a=\{x,y,z\}$, A[[*n*]] se refiere al elemento *n*, A[[1]]=x, A[[2]]=y, A[[3]]=z
$b=\{\{x_1, y_1, z_1\},\{x_2, y_2, z_2\},\{x_3, y_2, z_2\}\}$, b[[*n*,*m*]] se refiere al elemento *m* de la sublista *n*.

■

- **PowerExpand[*expr*].-** Expande todas las potencias de los productos y potencias de *expr*. Por ejemplo, PowerExpand[Sqrt[x y]] ---> Sqrt[x]*Sqrt[y]. PowerExpand convierte (a*b)^c a a^c*b^c. PowerExpand también convierte (a^b)^c a a^(b c).

■

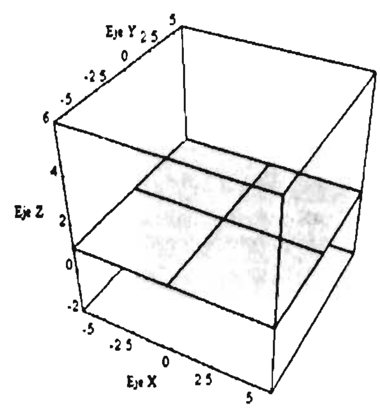
- **Random[ ].-** Genera un número real pseudoaleaotorio uniformemente distribuido en el rango de 0 a 1. Random[*type, range*] regresa un número pseudoaleatorio del tipo especificado en *type* , perteneciente al rango *range*. Los tipos posibles son: Integer, Real y Complex. El rango por defecto es de 0 a 1. Se puede especificar el rango como {*min, max*}; si se especifica un rango de *max* es equivalente a {*0, max*}.

■

- **Show[*graphics, options*].-** Despliega en pantalla gráficas en dos y tres dimensiones (*graphics*) usando las opciones especificadas en *options*. Show[*g1,g2,* ...] muestra varias gráficas combinadas. Show puede usarse con Graphics, Graphics3D, SurfaceGraphics, ContourGraphics, DensityGraphics and GraphicsArray. Las opciones especificadas en Show eliminan las incluidas en la expresión gráfica *graphics*. Ejemplo:

In[1]:=

```
(* Se genera una superficie gráfica *)

superficie = SurfaceGraphics[{{1,1,1},{1,1,1},{1,1,1}}, MeshRange->{{-6,6},{-6,6}},
        AxesLabel->{Eje X,Eje Y,Eje Z}, DefaultFont->{"Negrita",12},
        PlotRange->{{-6,6},{-6,6},{-2,6}}, BoxRatios->{1,1,1}];

(* Se despliega en pantalla la superficie gráfica generada por SurfaceGraphics *)

Show[superficie, Axes->True]
```

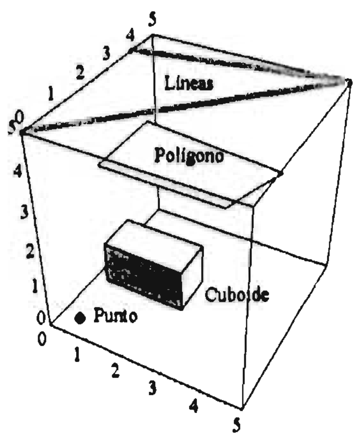


Out[2]=

-SurfaceGraphics-

In[3]:=

```
(* Se genera una gráfica tridimensional donde se muestra una línea, un punto, un cuboide
   y un polígono. *)

grafico3D = Graphics3D[{Text["Líneas", {1,5,4}], Text["Polígono", {1,5,2}],
                Text["Cuboide", {3,4,-1}],Text["Punto", {1.2,1,0}],
               {Polygon[{{1,1,4},{1,3,4},{4,4,3},{4,1,4}}]},
               {Cuboid[{1,1,1},{3,2,2}]},
               {AbsoluteThickness[5],RGBColor[1,.5,0],Line[{{0,0,5},{5,5,5},{0,4,5}}]},
                {PointSize[0.02],Point[{.5,.5,0}]}}, DefaultFont->{"Negrita",16}];

(* Se despliega en pantalla la gráfica generada por Graphics3D *)

Show[grafico3D, Axes->True, PlotRange->{{0,5},{0,5},{0,5}}]
```



Out[2]=
-Graphics3D-

■

- **Simplify[*expr*].-** Realiza una secuencia de transformaciones algebraicas sobre *expr*, y regresa la forma más simple encontrada.

■

- **Solve[*eqs, vars*].- Solve[*eqs, vars*]** intenta resolver una ecuación o conjunto de ecuaciones para las variables *vars*. **Solve[*eqs, vars, elims*]** intenta resolver las ecuaciones para *vars* eliminando las variables *elims*. Las ecuaciones son dadas de la forma pi == pd. Ecuaciones simultáneas pueden especificarse en forma de lista. Se puede especificar una sola variable o una lista de ellas. **Solve[*eqs*]** trata de resolver para todas las variables en *eqs*. Ejemplo: **Solve[*3 x+ 9== 0, x*]**. **Solve[]** da soluciones explícitas como reglas de la forma **x-> sol**. Cuando hay varias variables, se da como una lista de reglas la solución: **{ x-> sx, y-> sy }**. Cuando hay varias soluciones, **Solve[]** da una lista de ellas. **Solve[]** trata principalmente con ecuaciones lineales y polinomiales. Ejemplo:

In[1] :=
(* Se resuelve simbólicamente la ecuación general de segundo grado *)

Solve[a*x^2+b*x+c==0,x]

Out[2] :=
```
                   2                             2
         -b - Sqrt[b  - 4 a c]          -b + Sqrt[b  - 4 a c]
  {{x -> ---------------------}, {x -> ---------------------}}
                  2 a                           2 a
```

■

- **SurfaceGraphics[*array*].-** Es una representación gráfica de una superficie tridimensional, con alturas de cada punto sobre una rejilla especificado por los valores en *array*. SurfaceGraphics[*array, shades*] representa una superficie, cuyas partes son sombreadas de acuerdo a la lista *array*. SurfaceGraphics puede se desplegada usando Show. SurfaceGraphics utiliza las mismas opciones que Graphics3D, con las siguientes adiciones:

Opción	Default	Descripción
HiddenSurface	True	Si elimina superficies ocultas
Mesh	True	Si dibuja una malla sobre la superficie
MeshStyle	Automatic	Especifica el estilo de para la dibujar la malla
MeshRange	Automatic	Rango x,y donde se dibujará la superficie

Un ejemplo se muestra en la función ***Show***.

■

- **Switch[*expr, form1,value1,form2,value2, ...*].-** Evalúa *expr*, luego la compara con cada una de las *formi*, evaluando y regresando el *valuei* correspondiente a la primera igualdad encontrada. Solamente el *valuei* correspondiente al primer *formi* que es igual a *expr* será evaluado.

■

APENDICE B

PROGRAMAS INCLUIDOS EN DISCO.

En este apéndice se muestra el contenido del disco flexible incluido en esta tesis, dando una breve descripción y la forma de uso de cada uno de los programas que contiene.

1.- **cap_2.ma ⇒ Demostraciones Capítulo II (Antecedentes).** En este programa se presentan todas las demostraciones, del capítulo II, donde se utilizó Mathematica. El programa esta dividido en celdas que contienen cada una de las demostraciones realizadas. Para su utilización, basta simplemente con ejecutar la celda que contenga la demostración que deseamos verificar.

2.- **3gr_cin.ma ⇒ Cinemática de un robot manipulador de 3 grados de libertad utilizando rotaciones.** En este programa se presenta la modelación y simulación de la cinemática de un robot manipulador de tres grados de libertad. Este programa se divide en las siguientes celdas principales:

a).- **Definición de la Transformación Rotación.** En esta celda se define la transformación rotación y otras definiciones necesarias.
b).- **Definiciones Iniciales.** Aquí se define la base canónica de $\Re^3$ y las bases locales correspondientes a cada eslabón del manipulador, así como los Quaterniones necesarios para la modelación.
c).- **MODELADO DE LA CONFIGURACION NO DEFORMADA.** Celda título.
d).- **Modelo para la configuración no deformada (Secuencia 3-2).** Se realiza la obtención de las ecuaciones cinemáticas correspondientes a la configuración no deformada.
e).- **Ecuaciones optimizadas para la configuración no deformada.** Se presentan las ecuaciones obtenidas en la celda anterior, con las cuales se determinarán los parámetros de rotación correspondientes a la configuración no deformada.
f).- **Cinemática Inversa para determinar la configuración no deformada.** En esta celda se realiza el proceso de cinemática inversa para determinar la configuración no deformada correspondiente a una posición del elemento terminal del robot.
g).- **MODELADO DE LA CONFIGURACION DEFORMADA** Celda título.
h).- **Modelo para la configuración deformada (Secuencia 3-2-1).** Se obtienen las ecuaciones cenemáticas para la configuración deformada del robot de tres grados de libertad.
i).- **Ecuaciones optimizadas para la configuración deformada.** Se presentan las ecuaciones obtenidas en la celda anterior, las cuales serán utilizadas para poder realizar el proceso de simulación de la cinemática del robot.
j).- **CINEMATICA DIRECTA.** Celda título.
k).- **Graficación para visualizar la simulación.** Esta celda permite generar las gráficas necesarias para generar una simulación de la cinemática directa.
l).- **CINEMATICA INVERSA.** Celda título.
m).- **Solución de ecuaciones para diferentes trayectorias.** Celda título.
n).- **LINEAS RECTAS.** En esta celda se determinan los parámetros de rotación necesarios para que el elemento terminal del robot siga una trayectoria en línea recta, la cual es discretizada en una serie de puntos.
o).- **CIRCUNFERENCIAS O ELIPSES.** En esta celda se determinan los parámetros de rotación necesarios para que el elemento terminal del robot siga una trayectoria en forma de elipse o circunferencia, la cual es discretizada en una serie de puntos.

p).- **ASTROIDE.** En esta celda se determinan los parámetros de rotación necesarios para que el elemento terminal del robot siga una trayectoria en forma de astroide, la cual es discretizada en una serie de puntos.

q).- **Graficación para visualizar la simulación de la trayectoria seleccionada.** Finalmente, esta celda nos genera las gráficas necesaria para observar la simulación de la cinemática inversa del robot en cuestión. La simulación puede ser de cualquiera de las trayectorias programadas en las celdas anteriores. la que se obtenga dependerá de cual fue la celda de trayectoria que se ejecutó anteriormente.

Si deseamos generar las gráficas necesarias para observar la simulación de la cinemática del robot, debemos seguir una secuencia de ejecución de celdas, es decir debemos ejecutar en secuencia las siguientes celdas: **a, d, e, g, l,** para la cinemática directa. Para la cinemática inversa, si ya se ejecutaron las anteriores, se debe ejecutar una de las celdas correspondiente a una trayectoria (**n, o, p**) y la celda **q**. Una vez generadas las gráficas, basta con seleccionar la celda que las contiene y presionar el icono correspondiente para observar la simulación.

3.- **4gr_cin.ma ⇒ Cinemática de un robot manipulador de 4 grados de libertad utilizando rotaciones.** En este programa se presenta la modelación y simulación de la cinemática de un robot manipulador de cuatro grados de libertad, el cual se presentó en el capítulo III. Este programa se divide las mismas celdas que el anterior y se ejecuta de la misma forma.

4.- **Scara.ma ⇒ Cinemática de un robot manipulador de 4 grados de libertad tipo SCARA utilizando rotaciones.** En este programa se presenta la modelación y simulación de la cinemática de un robot manipulador tipo SCARA. Sus celdas y modo de ejecución son similares a los anteriores.

5.- **Evasion.ma ⇒ Ejemplo de aplicación de la modelación con una combinación de rotaciones y reflexiones de un robot manipulador de 3 grados de libertad en el proceso de evasión de obstáculos.** En este programa se presenta la modelación de la cinemática de un robot manipulador de tres grados de libertad utilizando una combinación de rotaciones y reflexiones, y se aplican para mostrar la simulación de la cinemática inversa con un proceso de evasión de obstáculos. Este programa se divide en la siguientes celdas:

a).- **Definición de las Transformaciones Rotación y Reflexión.** En esta celda se definen las transformaciones rotación y reflexión, así como algunas otras definiciones necesarias.

b).- **Definiciones Iniciales.** Aquí se define la base canónica de $\Re^3$ y las bases locales correspondientes a cada eslabón del manipulador, así como los Quaterniones necesarios para la modelación.

c).- **MODELADO DE LA CONFIGURACION NO DEFORMADA.** Celda título.

d).- **Modelo para la configuración no deformada.** Se realiza la obtención de las ecuaciones cinemáticas correspondientes a la configuración no deformada.

e).- **Ecuaciones optimizadas para la configuración no deformada.** Se presentan las ecuaciones obtenidas en la celda anterior, con las cuales se determinarán los parámetros de rotación correspondientes a la configuración no deformada.

f).- **Cinemática inversa para determinar la configuración no deformada.** En esta celda se realiza el proceso de cinemática inversa para determinar la configuración no deformada correspondiente a una posición del elemento terminal del robot.

g).- **MODELADO DE LA CONFIGURACION DEFORMADA** Celda título.

h).- **Modelo para la configuración deformada con rotaciones.** Se obtienen las ecuaciones cenemáticas para la configuración deformada del robot de tres grados de libertad, utilizando solamente rotaciones.

i).- **Modelo para la configuración deformada con rotaciones y reflexiones.** Se obtienen las ecuaciones cenemáticas para la configuración deformada del robot de tres grados de libertad, utilizando una combinación de rotaciones y reflexiones.

j).- **Ecuaciones optimizadas para la configuración deformada.** Se presentan las ecuaciones obtenidas en la celdas anterior, las cuales serán utilizadas para poder realizar el proceso de simulación de la cinemática del robot.

k).- **CINEMATICA INVERSA.** Celda título.

l).- **Solución de ecuaciones para diferentes trayectorias.** Celda título.

m).- **LINEAS RECTAS.** En esta celda se determinan los parámetros de rotación necesarios para que el elemento terminal del robot siga una trayectoria en línea recta, la cual es discretizada en una serie de puntos.

n).- **CIRCUNFERENCIAS O ELIPSES.** En esta celda se determinan los parámetros de rotación necesarios para que el elemento terminal del robot siga una trayectoria en forma de elipse o circunferencia, la cual es discretizada en una serie de puntos.

o).- **ASTROIDE.** En esta celda se determinan los parámetros de rotación necesarios para que el elemento terminal del robot siga una trayectoria en forma de astroide, la cual es discretizada en una serie de puntos.

p).- **Graficación para visualizar la simulación de la trayectoria seleccionada.** Finalmente, esta celda nos genera las gráficas necesaria para observar la simulación de la cinemática inversa del robot en cuestión, ubicando un obstáculo dentro del espacio de trabajo del robot de manera aleatoria y decidiendo que modelo cinemático utilizar para evadir dicho obstáculo. La simulación puede ser de cualquiera de las trayectorias programadas en las celdas anteriores, la que se obtenga dependerá de cual fue la celda de trayectoria que se ejecutó anteriormente.

Para lograr obtener las gráficas necesarias para observar la simulación deseada, se debe realizar la ejecución de las siguientes celdas: primeramente **a, e, f, i,** después la celda correspondiente a la trayectoria deseada (**m, n, o**), y finalmente la celda **p** con lo que se generarán las gráficas para la simulación correspondiente.