2

Representación de la posición y la orientación

Introducción

La manipulación mediante robots implica el movimiento de sólidos y herramientas a través del espacio. Por tanto parece evidente la necesidad de representar adecuadamente las posiciones y orientaciones de los elementos que componen un robot.

En este Capítulo se presentan una serie de funciones de MATLAB, que facilitan el manejo de las matrices que representan la posición y la orientación. Asimismo, es posible visualizar los cuadros de referencia y comprobar gráficamente cómo les afectan las transformaciones de rotación y traslación.

El lector solamente necesita conocer el modo en que se manejan las matrices y los vectores en MATLAB para comprender y emplear las funciones que se describen en este Capítulo. Dichas funciones se emplean en el Capítulo 3 de Ollero [1].

Instrucciones MATLAB

En el siguiente cuadro se muestra un resumen de las funciones de MATLAB que facilitan las operaciones relacionadas con la representación de la posición y la orientación de cuerpos en el espacio.

Tabla 2.1: Instrucciones relacionados con la localización (posición y orientación).

	Propósito
eul2tr	Generar a partir de los ángulos de Euler Z-Y-Z la transformación homogénea correspondiente
frame	Representar gráficamente un cuadro de referencia con una posición y orientación dadas
ishomog	Comprobar si el argumento que se le pasa es una transformación homogénea
rotvec	Calcular la transformación de rotación en torno a un vector
rotx, roty, rotz	Calcular las transformaciones de rotación en torno a los ejes $\hat{X}$, $\hat{Y}$ y $\hat{Z}$ respectivamente
rpy2tr	Obtener la transformación homogénea correspondiente a los ángulos RPY dados
transl	Cálculos asociados a transformaciones de traslación
tr2eul	Obtener los ángulos de Euler Z-Y-Z a partir de una matriz de transformación homogénea
tr2rpy	Extraer de una transformación homogénea sus correspondientes ángulos RPY
trinv	Calcular la transformación inversa de una dada

eul2tr

Propósito

Generar a partir de los ángulos de Euler Z-Y-Z la transformación homogénea correspondiente.

Sintaxis

```
T=eul2tr([alpha beta gamma])
T=eul2tr(alpha,beta,gamma)
```

Descripción

Esta función devuelve la transformación homogénea que corresponde a los ángulos de Euler Z-Y-Z (`alpha`, `beta` y `gamma`), que se le pasan como parámetros.

Ver también

`rpy2tr`, `tr2eul`

Referencias

Corke, P.I. *A robotics toolbox for MATLAB*, IEEE Robotics and Automation Magazine, Vol.3, núm. 1, pp. 24-32, 1996.

Craig, J.J., *Introduction to robotics*, Addison Wesley, segunda ed., 1989.

Ollero, A., *Robótica: Manipuladores y robots móviles*, Marcombo-Boixareu editores, 2001.

frame

Propósito

Representar gráficamente un cuadro de referencia con una posición y orientación dadas.

Sintaxis

```
frame(TT,color,tam,opt)
```

Descripción

Esta función se encarga de representar cualquier cuadro de referencia mediante 3 ejes ortogonales. Para ello hay que pasarle como parámetro una transformación (`TT`) con la posición y orientación del cuadro de referencia a visualizar.

En el parámetro `TT` también es posible introducir una trayectoria de transformaciones y en este caso se visualizan todos los cuadros de referencia que componen dicha trayectoria.

En el parámetro `color` se especifica el color que se quiere para la representación del cuadro y mediante `tam` se indica el tamaño deseado para las flechas que representan el sistema de coordenadas.

Si `TT` es una trayectoria de transformaciones, es posible especificar que aparezcan simultáneamente todas las transformaciones en pantalla dando un valor cualquiera al parámetro `opt`. Si no se introduce dicho parámetro, los cuadros aparecen y desaparecen sucesivamente, y al final solamente queda el último en pantalla.

Ejemplos

Ver Ejemplo 3.3 en Ollero [1] y Ejemplo H.2.4.

A continuación se presentan una serie de ejemplos que ilustran las posibilidades de esta función.

Ejemplo H.2.1 (archivo `ejh21.m`)

Para representar un cuadro de referencia {A}, respecto al cual se van a referir los cuadros de los siguientes ejemplos, se puede escribir:

```
TA = [1 0 0 0;
      0 1 0 0;
      0 0 1 0;
      0 0 0 1] ;
frame (TA, 'c', 1);
grid on
```

El resultado se observa en la Figura 2.1.

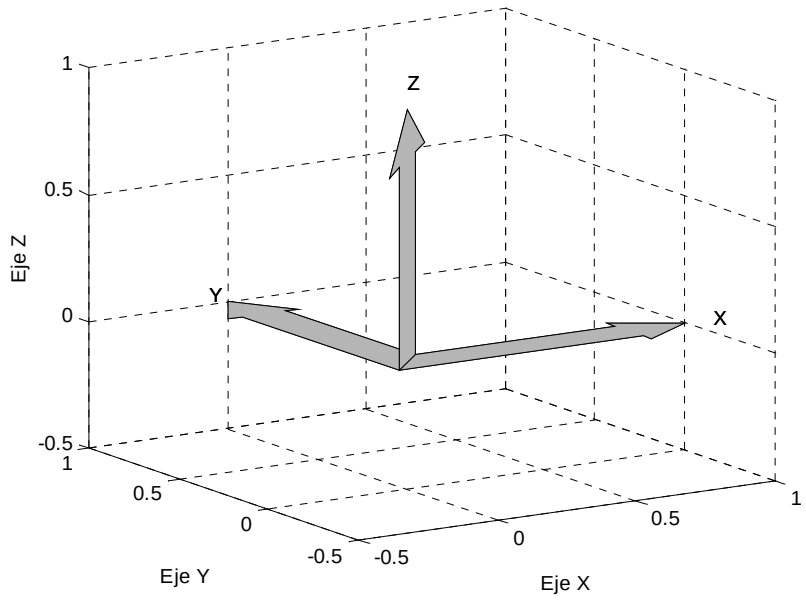


Figura 2.1: Cuadro de referencia {A}.

Ejemplo H.2.2 (archivo `ejh22.m`)

A continuación se van a representar 2 cuadros de referencia ({B} y {C}), generados a partir de {A} mediante rotación y traslación:

- {B} será {A} rotado 30º en torno al eje $\hat{X}$ y trasladado a las coordenadas (2, 4, 0).
- {C} será {A} rotado 60º en torno al eje $\hat{Z}$ y trasladado a las coordenadas (-1, -3 , 2).

Las líneas a ejecutar serían:

```
TA =[1 0 0 0;
     0 1 0 0 ;
     0 0 1 0;
     0 0 0 1];
TB = transl(2,4,0)*rotx(pi/6)*TA;
TC = transl(-1,-3,2)*rotz(pi/3)*TA;
frame(TA, 'c', 1);
frame(TB, 'b', 1);
frame(TC, 'w', 1);
```

```
axis([-2 3 -2 3 -2 3])%Se establecen los rangos de cada eje
                       para visualizar bien los marcos {A},
                       {B} y {C}.

grid on
view (-54,22)  % Se establece un punto de vista adecuado.
```

resultando el gráfico de la Figura 2.2.

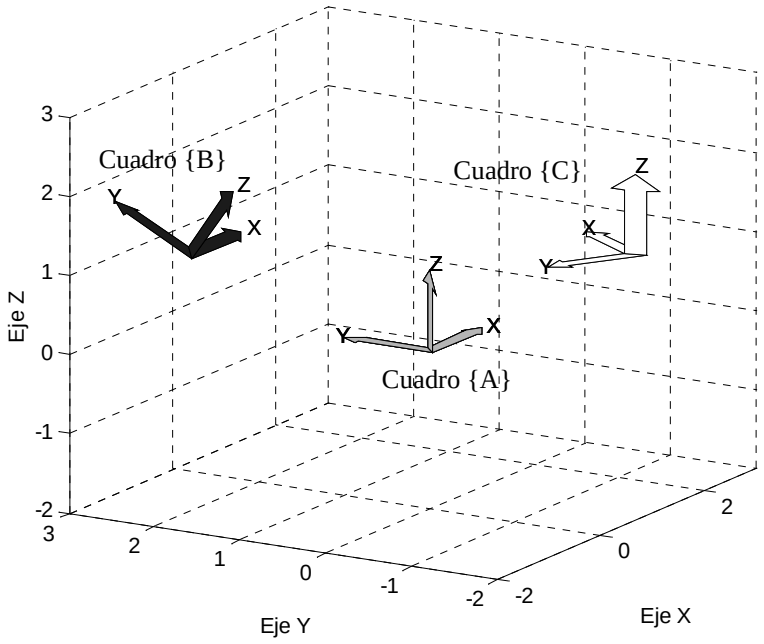


Figura 2.2: Cuadros de referencia {B} y {C}.

Ejemplo H.2.3 (archivo `ejh23.m`)

Para finalizar se va a ilustrar el modo en que se representaría una trayectoria de transformaciones. Se va a considerar como ejemplo, la trayectoria que resultaría de desplazar cada cuadro respecto al anterior una distancia de 1.1 unidades a lo largo de las direccionesde de los ejes $\hat{Y}$ y $\hat{Z}$. Dicha trayectoria se podría programar así:

```
w = 2;
T(:,:,1) = eye(4);
for np = 1:6
      T(:,:,w) = transl(0,1.1,1.1)*T(:,:,w-1);
      w = w+1;
end
frame(T,'c',1,0);
```

```
axis([-4 4 -2 6 0 8]) % Se establecen los rangos de cada eje
                        para visualizar bien los marcos de la
                        trayectoria.

grid on
view(-40,12)   % Se establece un punto de vista adecuado.
```

y el resultado es la Figura 2.3.

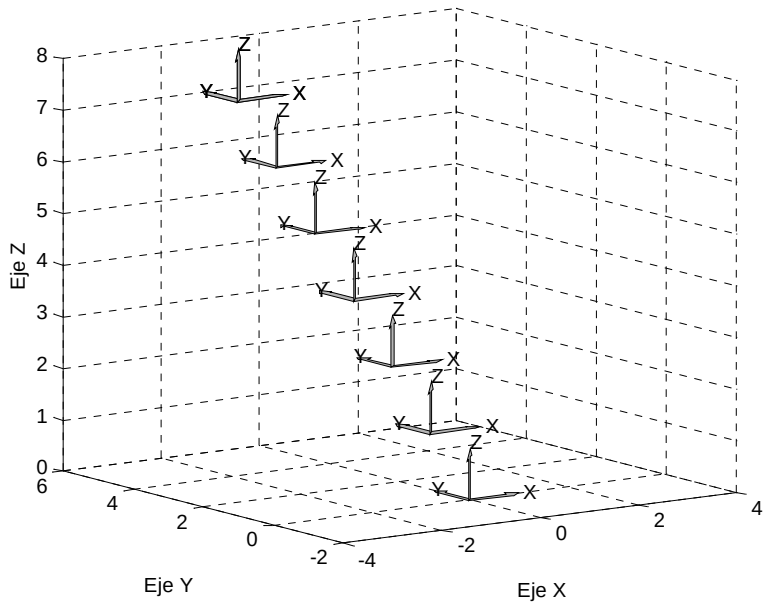


Figura 2.3: Trayectoria de transformaciones.

Referencias

Ollero, A., *Robótica: Manipuladores y robots móviles*, Marcombo-Boixareu editores, 2001.

ishomog

Propósito

Comprobar si el argumento que se le pasa es una transformación homogénea.

Sintaxis

```
ishomog (T)
```

Descripción

Devuelve "verdadero" si `T` es una matriz 4x4 que corresponde a una transformación homogénea.

Referencias

Corke, P.I., *A robotics toolbox for MATLAB*, IEEE Robotics and Automation Magazine, Vol.3, núm. 1, pp. 24-32, 1996.

rotvec

Propósito

Calcular la transformación de rotación en torno a un vector.

Sintaxis

```
T=rotvec(v,theta)
```

Descripción

Esta función devuelve la transformación homogénea T, que representa una rotación de theta radianes en torno al vector v, que se le pasa como parámetro.

Ejemplos

Ver Ejemplo H.2.5.

Ver también

`rotx`, `roty`, `rotz`

Referencias

Corke, P.I., *A robotics toolbox for MATLAB*, IEEE Robotics and Automation Magazine, Vol.3, núm. 1, pp. 24-32, 1996.

rotx, roty, rotz

Propósito

Calcular las transformaciones de rotación en torno a los ejes $\hat{X}$, $\hat{Y}$ y $\hat{Z}$ respectivamente.

Sintaxis

```
T=rotx(theta)
T=roty(theta)
T=rotz(theta)
```

Descripción

Devuelve la transformación homogénea `T` que representa una rotación de `theta` radianes en torno a los ejes $\hat{X}$, $\hat{Y}$ y $\hat{Z}$ respectivamente.

Ejemplos

Ver Ejemplo 3.3 en Ollero [1] y Ejemplos H.2.4, H.2.5 y H.2.6.

Ver también

`rotvec`

Referencias

Corke, P.I., *A robotics toolbox for MATLAB*, IEEE Robotics and Automation Magazine, Vol.3, núm. 1, pp. 24-32, 1996.

Ollero, A., *Robótica: Manipuladores y robots móviles*, Marcombo-Boixareu editores, 2001.

rpy2tr

Propósito

Obtener la transformación homogénea correspondiente a los ángulos RPY dados.

Sintaxis

```
T=rpy2tr([r p y])
T=rpy2tr(r,p,y)
```

Descripción

Esta función devuelve la transformación homogénea que corresponde a los ángulos RPY (roll/pitch/yaw). Dichos ángulos se le deben pasar expresados en radianes, y son los ángulos de rotación en torno a los ejes $\hat{X}$, $\hat{Y}$ y $\hat{Z}$ respectivamente.

Ver también

`tr2rpy`, `eul2tr`

Referencias

Corke, P.I., *A robotics toolbox for MATLAB*, IEEE Robotics and Automation Magazine, Vol.3, núm. 1, pp. 24-32, 1996.

Craig, J.J., *Introduction to robotics*. Addison Wesley, segunda ed., 1989.

Ollero, A., *Robótica: Manipuladores y robots móviles*, Marcombo-Boixareu editores, 2001.

tr2eul

Propósito

Obtener los ángulos de Euler Z-Y-Z a partir de una matriz de transformación homogénea.

Sintaxis

```
[alpha beta gamma]=tr2eul(T)
```

Descripción

La función `tr2eul` devuelve un vector de ángulos en radianes,que se corresponden con los ángulos de Euler Z-Y-Z asociados a la submatriz de rotación de la transformación homogénea `T` que se le pasa como parámetro.

Ejemplos

Ver Ejemplo H.2.4.

Ver también

`eul2tr`, `tr2rpy`

Referencias

Corke, P.I., *A robotics toolbox for MATLAB*, IEEE Robotics and Automation Magazine, Vol.3, núm. 1, pp. 24-32, 1996.

Craig, J.J., *Introduction to robotics*. Addison Wesley, segunda ed., 1989.

Ollero, A., *Robótica: Manipuladores y robots móviles*, Marcombo-Boixareu editores, 2001.

tr2rpy

Propósito

Extraer de una transformación homogénea sus correspondientes ángulos RPY.

Sintaxis

```
[r p y]=tr2rpy(T)
```

Descripción

Esta función devuelve un vector que contiene los ángulos RPY en radianes, que corresponden a la submatriz rotacional de la transformación homogénea T. Los ángulos calculados son respectivamente rotaciones en torno a los ejes $\hat{X}$, $\hat{Y}$ y $\hat{Z}$ respectivamente.

Ejemplos

Ver Ejemplo H.2.4

Ver también

`rpy2tr`, `tr2eul`

Referencias

Corke, P.I., *A robotics toolbox for MATLAB*, IEEE Robotics and Automation Magazine, Vol.3, núm. 1, pp. 24-32, 1996.

Craig, J.J., *Introduction to robotics*. Addison Wesley, segunda ed., 1989.

Ollero, A., *Robótica: Manipuladores y robots móviles*, Marcombo-Boixareu editores, 2001.

transl

Propósito

Cálculos asociados a transformaciones de traslación.

Sintaxis

```
T=transl(x,y,z)
T=transl(v)
v=transl(T)
xyz=transl(TC)
```

Descripción

Las dos primeras formas devuelven una matriz de transformación que representa una traslación dada por tres escalares (`x`, `y`, `z`), o bien por un vector `v`.

La tercera forma devuelve, en un vector columna (`v`) de 3 elementos, la parte traslacional de la transformación `T`.

La cuarta forma devuelve una matriz (`xyz`), cuyas filas contienen las coordenadas x, y y z de las traslaciones contenidas en cada una de las matrices de la trayectoria de transformaciones `TC`.

Ejemplos

Ver Ejemplo 3.3 en Ollero [1] y Ejemplos H.2.4 y H.2.6.

Ver también

`rotx`, `roty`, `rotz`, `rotvec`

Referencias

Corke, P.I., *A robotics toolbox for MATLAB*, IEEE Robotics and Automation Magazine, Vol.3, núm. 1, pp. 24-32, 1996.

Ollero, A., *Robótica: Manipuladores y robots móviles*, Marcombo-Boixareu editores, 2001.

trinv

Propósito

Calcular la transformación inversa de una dada.

Sintaxis

```
Tinv=trinv(T)
```

Descripción

Devuelve la matriz de transformación inversa (`Tinv` = ${}^{B}_{A}T$) de aquella que se le pasa como parámetro (`T` = ${}^{A}_{B}T$).

Algoritmo

El cálculo se basa en la fórmula:

$$ {}^{B}_{A}T = \begin{bmatrix} {}^{B}_{A}R & {}^{B}P_{ORGA} \\ 000 & 1 \end{bmatrix} = \begin{bmatrix} {}^{A}_{B}R^{T} & -{}^{A}_{B}R^{T}\ {}^{A}P_{ORGB} \\ 0\ 0\ 0 & 1 \end{bmatrix} \qquad (2.1) $$

Ejemplos

Ver Ejemplo 3.4 en Ollero [1].

Referencias

Craig, J.J., *Introduction to robotics*. Addison Wesley, segunda ed., 1989.

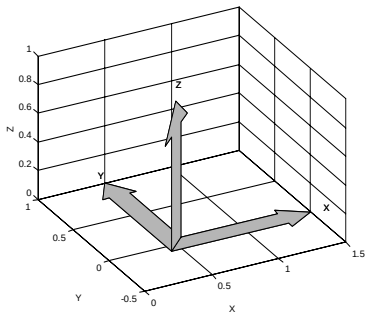
Ollero, A., *Robótica: Manipuladores y robots móviles*, Marcombo-Boixareu editores, 2001.

Ejemplos

A continuación se muestran una serie de ejemplos de la utilización de los comandos de este Capítulo.

Ejemplo H.2.4 (archivo `ejh24.m`)

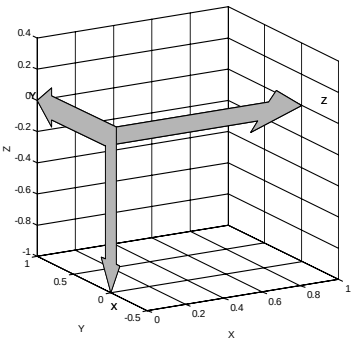
- Traslación pura de 0.5 m en la dirección del eje $\hat{X}$




```
T = transl(0.5,0,0)
frame(T,'c',1);
```

$$T = \begin{bmatrix} 1 & 0 & 0 & 0.5 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

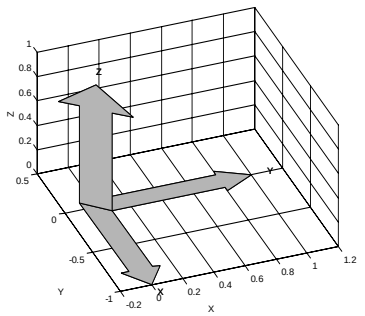
- Rotación de 90 grados en torno al eje $\hat{Y}$




```
T = roty (pi/2)
frame(T,'c',1);
```

$$T = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

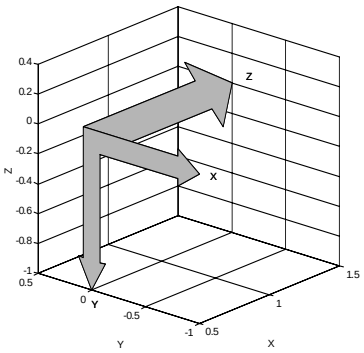
- Rotación de -90 grados en torno al eje $\hat{Z}$




```
T = rotz (-pi/2)
frame (T,'c',1);
```

$$T = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Concatenar las tres operaciones anteriores




```
T=transl(.5,0,0)*roty(pi/2)*rotz(-pi/2)
frame (T,'c',1);
```

$$T = \begin{bmatrix} 0 & 0 & 1 & 0'5 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Calcular los ángulos de Euler Z-Y-Z de la transformación resultante

```
tr2eul (T)
```

resultando `[0 1.5708 -1.5708]`.

- Calcular los ángulos RPY de la misma transformación

```
tr2rpy(T)
```

obteniéndose `[-1.5708 0.0000 -1.5708]`.

Ejemplo H.2.5

Un robot móvil provisto de sensores de proximetría detecta un obstáculo a una distancia d en la dirección de marcha. Se sabe que el ángulo de orientación del robot en el instante de la medida es θ. Se trata de determinar las coordenadas absolutas del obstáculo con respecto a un sistema de ejes de referencia solidario al vehículo pero con la misma orientación que el sistema absoluto.

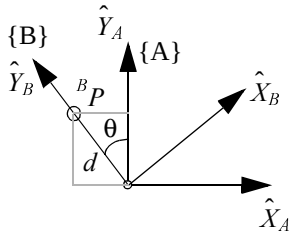


Figura 2.4: Detección de obstáculo desde un robot móvil.

El sistema {B} esta rotado un ángulo θ respecto al sistema {A}. Por tanto basta con utilizar la transformación homogénea de rotación en torno al eje $\hat{Z}$. Esta transformación viene dada por una matriz 4x4 y por tanto el vector ${}^{B}P$ debe ser de dimensión 4x1, es decir ${}^{B}P = \begin{bmatrix} 0 & d & 0 & 1 \end{bmatrix}^{T}$.

Por tanto para determinar las coordenadas absolutas simplemente hay que hacer el producto de la matriz de la transformación de rotación por el vector ${}^{B}P$. Para resolver esto con MATLAB bastaría con escribir lo siguiente:

```
syms d t real
PenB = [0 d 0 1]';
solucion = rotz (t)*PenB
```

En las dos primeras coordenadas del vector `solucion` están las coordenadas absolutas buscadas. En lugar de la función `rotz(t)` se podía haber usado también la función `rotvec(v,t)` que proporciona la transformación homogénea que corresponde a una rotación de un ángulo `t` en torno a un vector `v`. El modo de emplearla en esta caso habría sido:

```
rotvec([0 0 1]',t)
```

Ejemplo H.2.6

Considérese ahora un manipulador plano con una articulación de traslación y otra de rotación como el que se muestra en la Figura H.2.5.

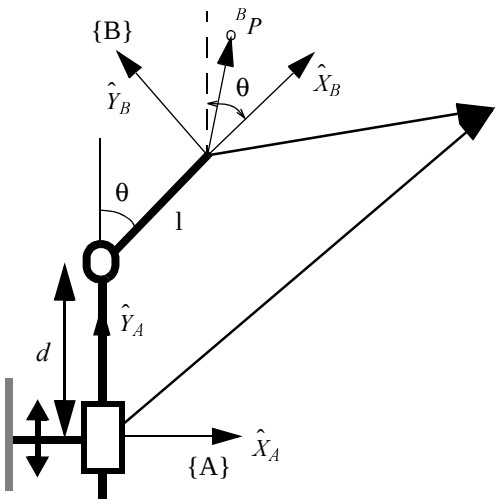


Figura 2.5: Cambio de sistemas de referencia en un manipulador plano.

Sean d y θ las variables de la primera y segunda articulación. Calcular las coordenadas en el sistema {A} en función de las coordenadas respecto al sistema {B}.

En este caso se tiene una rotación según un ángulo $\pi/2-\theta$, una traslación según el vector $\begin{bmatrix} l\sin\theta & l\cos\theta & 0 \end{bmatrix}^T$ y una última traslación según $\begin{bmatrix} 0 & d & 0 \end{bmatrix}^T$. Por tanto los comandos de MATLAB necesarios serían los siguientes:

```
syms t l d px py real
PenB = [px py 0 1]';
v1 = [l*sin(t) l*cos(t) 0];
v2 = [0 d 0];
solucion = transl(v2)*transl(v1)*rotz(pi/2-t)*PenB
```

Referencias

[1] Ollero, A., *Robótica: Manipuladores y robots móviles*, Marcombo-Boixareu editores, 2001.