Código Fuente 2.6 Integración numérica trapezoidal

```
%MATLAB Aplicado a Robótica y Mecatrónica
%Capítulo 2 Métodos numéricos
%Editorial Alfaomega
%Fernando Reyes Cortés
%Archivo cap2_trap.m Continúa 2.5
```

Integración numérica trapezoidal

```
tfinal=pi;
x=tini:tinc:tfinal;
f=sin(x);
I_trapz= trapz(x,f);
I_a= 1-cos(pi);
disp('Resultado')
%despliega los resultados comparativos
%del método trapezoidal con el analítico.
disp([I_trapz I_a])
```

En este ejemplo se ha calculado el área bajo la curva sobre el intervalo $[0, \pi]$. El cálculo se realiza una sola vez. Sin embargo, hay aplicaciones en control de robots manipuladores donde se requiere realizar el cálculo de la integral de manera sistemática conforme el tiempo evoluciona. A continuación se muestra un procedimiento recursivo donde se obtiene la integral por el método trapezoidal para pequeños intervalos del tamaño de $\frac{\pi}{1000}$ sobre un intervalo amplio $[0, 2\pi]$.

♣ ♣ Ejemplo 2.7

Calcular la integral de la función sen(x) por el método trapezoidal para cada valor de intervalo $\frac{\pi}{1000}$ contenido en el intervalo $[0, 2\pi]$

$$I_{trapz} = \int_0^{2\pi} \text{sen}(x)\, dx.$$

Solución

La idea es obtener un barrido de la integral de $\operatorname{sen}(x)$ para cada intervalo $\frac{\pi}{1000}$ que pertenece al intervalo $[0, 2\pi]$. Es decir, $x \in [0, \frac{\pi}{1000}, 2\frac{\pi}{1000}, 3\frac{\pi}{1000}, \cdots, 2\pi]$. En otras palabras, el intervalo va evolucionando por incrementos de $\frac{\pi}{1000}$ hasta llegar a 2π, con esto se requieren 2000 iteraciones para alcanzar el valor de 2π.

El programa que se presenta en el cuadro de código 2.7 contiene la forma de implementar el método trapezoidal para aproximar la integral de una función $f(x) = \operatorname{sen}(x)$ en el intervalo $[0, 2\pi]$. De la línea 4 a la 9 se define el intervalo de tiempo de integración. El incremento del paso de integración es cada $\frac{\pi}{1000}$.

En la línea 10 se obtiene la dimensión del vector de tiempo, con esa información se realizará el algoritmo recursivo, es decir el número de veces (2000) que se realizará el ciclo `for` para abarcar el intervalo $[0, 2\pi]$ el cual se presenta de las líneas 11 a la 21.

En la línea 12 se exploran los registros de la base de tiempo `t(k)`; cuando k=1 entonces se realiza el primer paso de integración del intervalo $[0, \frac{pi}{1000}]$ cuyo código abarca las líneas 13 a la 16. Este código es necesario por razones técnicas del lenguaje de MATLAB debido a que el primer elemento del vector de tiempo `t(1)` es cero, ts=0, entonces no se podría llevar a cabo el primer intervalo de integración `x=0:tinc:0`, marcando un error de programación. En su lugar se realiza `x=0:tinc:tinc`.

Para el segundo incremento del tiempo de integración y posteriores incrementos se ubica en las líneas 17 a la 21 sobre intervalo $[\frac{\pi}{1000}, \cdots, 2\pi]$. En la línea 23 se asigna la función `sen(x)` para el correspondiente k-ésimo intervalo de integración $[(k-1)\frac{pi}{1000}, k\frac{pi}{1000}]$. La línea 24 tiene la función `I_trapz(k)=trapz(x,f)`. Este procedimiento se realiza un número de veces (2000) hasta igualar la dimensión del vector de tiempo `t`.

En la línea 28 se grafica la integral de la función `sen(x)` por el método trapezoidal y se compara con el método analítico:

$$
\begin{aligned}
I_{trapz} &= \int_0^{2\pi} \operatorname{sen}(x)\, dx \\
&= -\cos(x)\Big|_0^{2\pi} \\
&= 1 - \cos(x), \quad x \in [0, 2\pi]
\end{aligned}
$$

Código Fuente 2.7 Método de integración trapezoidal

%MATLAB Aplicado a Robótica y Mecatrónica
%Capítulo 2 Métodos numéricos
%Editorial Alfaomega
%Fernando Reyes Cortés
%Archivo cap2_trapezoidal.m
% calcula la integral de la función sen(x)
% el cálculo se realiza por medio de
%integración iterativa (2000 iteraciones)
%con incrementos de $\frac{\pi}{1000}$
% para el intervalo $[0, 2\pi]$.

Método de integración trapezoidal

```
clc;
clear all;
close all;
% intervalo de integración
tini=0; % tiempo inicial de integración
tinc=pi/1000; % pasos de incremento del tiempo de integración
tfinal=2*pi; % tiempo final de integración
% base de tiempo de integración
t=tini:tinc:tfinal;
[m,n]=size(t); % dimensión del vector de tiempo
```

C´odigo Fuente 2.8 M´etodo de integraci´on trapezoidal

```
%Contin´ua programa 2.7
%MATLAB Aplicado a Rob´otica y Mecatr´onica
%Cap´ıtulo 2 M´etodos num´ericos
%Editorial Alfaomega
%Fernando Reyes Cort´es
%Archivo cap2_trapezoidal.m
```

M´etodo de integraci´on trapezoidal

```
for k=1:n
    % registro de la base de tiempo
    ts=t(k);
    if k==1
        % para el primer paso de integraci´on
        % se integra desde 0 a π/1000.
        x=0:tinc:tinc;
    else
        % para el segundo paso de integraci´on y posteriores
        % se integra desde x = 0 hasta el tiempo actual ts.
        x=0:tinc:ts;
    end
    f=sin(x); % funci´on a integrar
    I_trapz(k)= trapz(x,f); % integraci´on por m´etodo trapezoidal
end
% gr´afica comparativa entre los m´etodos
% anal´ıtico y trapezoidal
plot(x,I_trapz,x,1-cos(x))
```

La figura 2.7 muestra el resultado comparativo entre el m´etodo trapezoidal y el anal´ıtico para la integral de la funci´on sen(x) que genera el programa 2.7. Obs´ervese que el m´etodo trapezoidal es una aproximaci´on muy buena comparada con el exacto o m´etodo anal´ıtico, de tal forma que el resultado trapezoidal se superpone a la gr´afica del m´etodo anal´ıtico (cos(x)-1) en el intervalo $[0, 2\pi]$.

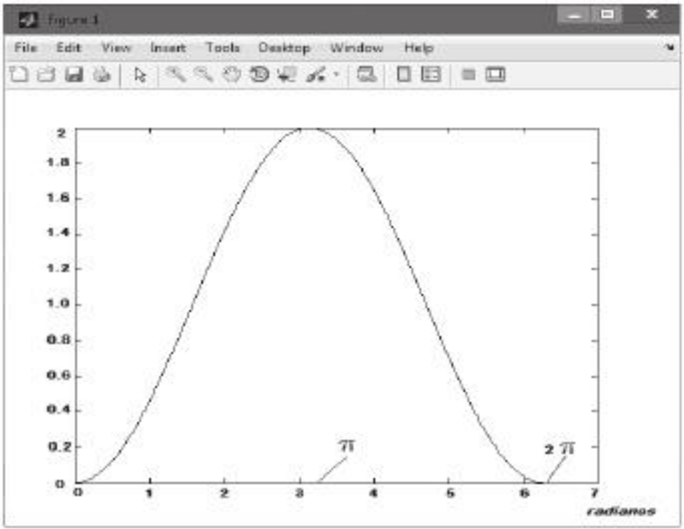


Figura 2.7 Comparación entre los métodos trapezoidal y analítico.

Regla de Simpson

Cuando el área bajo la curva de la función $f(x)$ es representada sobre areas de secciones cuadráticas y si el intervalo $[a, b]$ es dividido en $2n$ secciones iguales, entonces el área de $f(x)$ puede ser aproximada por la regla de Simpson cuya expresión está dada de la siguiente manera:

$$
\begin{aligned}
I_{simp} = & \frac{b-a}{6n}[f(x_0) + 4f(x_1) + 2f(x_2) + 4f(x_3) + \ldots + 2f(x_{2n-2}) \\
& +4f(x_{2n-1}) + f(x_{2n})]
\end{aligned} \tag{2.12}
$$

donde x_i representan los valores finales de cada sección, para $i = 1, 2, \cdots, n-1$; $x_0 = a$, y $x_{2n} = b$.

En MATLAB, la regla de Simpson se implementa en forma adaptiva por medio de la función quad; la tabla 2.1 muestra su sintaxis.

quad(f,a,b) aproxima la integral de una función $f(x)$ dentro de los límites finitos $[a, b]$, usando un algoritmo recursivo adaptable de cuadratura de Simpson, donde $a, b \in$ IR. El error de integración numérica que usa es 1e-06. La función $f(x)$ puede ser escalar o vectorial. En el caso vectorial quad retorna un vector de salida donde

Tabla 2.1 Funciones de cuadratura

quad('nombre_funcion',a,b)	Retorna el área de la función en el intervalo comprendido entre a y b usando la regla de Simpson.
q = quad(fun,a,b,tol)	Emplea un error absoluto de tolerancia tol en lugar del valor por default 1e-06 sobre el intervalo $[a, b]$. Valores grandes de tol producen integración numérica muy rápida, pero con menor exactitud.

cada componente tiene la integral de la correspondiente componente de la función vectorial $\mathbf{f}(\mathbf{x})$. La función quad puede ser menos eficiente (pobre exactitud) con funciones no suaves.

♣ Ejemplo 2.8

Calcular la integral por el método de Simpson de la función raíz cuadrática $f(x) = \sqrt{x}$ para intervalos no negativos $[a, b]$.

Solución

La función raíz cuadrada $f(x) = \sqrt{x}$ puede ser integrada analíticamente sobre el intervalo $[a, b]$, $a, b \in \mathbb{R}_+$ de la siguiente forma:

$$I = \int_a^b \sqrt{x}\, dx = \frac{2}{3}\left(b^{\frac{3}{2}} - a^{\frac{3}{2}}\right) \qquad (2.13)$$

Este procedimiento sirve de referencia para comparar el valor numérico que calcula el método de Simpson.

El cuadro 2.9 contiene el programa para calcular la integral de la función $f(x) = \sqrt{x}$ por el método de Simpson sobre un intervalo no negativo $[a, b]$ especificado por el usuario, y el resultado es comparado con el método analítico (2.13) el cual se encuentra implementado en la línea 17.

Por precaución, sobre los datos del intervalo definido por el usuario se inserta la instrucción `if` para verificar que los extremos del intervalo sean positivos y además que cumplan $a < b$.

Se hace la aclaración que dentro del lenguaje de programación de MATLAB la función `sqrt` acepta valores negativos y retorna números complejos. Sin embargo, en modelado dinámico y control de robots manipuladores usando moldeo de energía (estabilidad de Lyapunov) no se requiere la teoría de números complejos.

Código Fuente 2.9 Integración Simpson

```
%MATLAB Aplicado a Robótica y Mecatrónica
%Capítulo 2 Métodos numéricos
%Editorial Alfaomega
%Fernando Reyes Cortés
%Archivo cap2_simpson1.m
%Función quad para calcular la integral de la
%función raíz cuadrada por el método de Simpson.
%El resultado se compara con la integral analítica.
%El intervalo de integración es definido por el usuario.
```

Integración Simpson

```
1 clc;
2 clear all;
3 close all;
4 disp('Método de Simpson')
5 % los valores del intervalo [a, b] son proporcionados
6 % por el usuario desde el teclado
7 a=input('Introduzca el valor inicial del intervalo de integración: ');
8 b=input('Introduzca el valor final del intervalo de integración: ');
```

Código Fuente 2.10 Integración Simpson

```
% Continuación del programa 2.9
%MATLAB Aplicado a Robótica y Mecatrónica
%Capítulo 2 Métodos numéricos
%Editorial Alfaomega
%Fernando Reyes Cortés
%Archivo cap2_simpson1.m
```

Integración Simpson

```
if a>=0 && b>= 0
    if a>b
        disp('Error: el intervalo positivo [a,b] debe cumplir a<b.')
        return
    end
    % regla de Simpson
    % la función raíz cuadrada (sqrt) es indicada como
    % cadena de caracteres en la función quad
    I_simp=quad('sqrt',a,b);
    % método analítico
    I_a=2/3*(b^(3/2)-a^(3/2));
    fprintf('Valor analítico=%f \n Simpson: %f \n',I_a,I_simp)
    else % Para el caso de valores negativos
    disp('Error en los valores del intervalo de integración.')
    disp('El intervalo [a,b] con a<b debe contener únicamente valores positivos.')
end
```

La salida del programa cap2_simpson1.m produce resultados idénticos entre el método de Simpson con el analítico:

```
Valor analítico=33.333333
        Simpson=33.333333
```

corresponde al intervalo a=0 y b=10.

♣ Ejemplo 2.9

Calcular la integral de la función $f(x) = x^5$ por el método de Simpson.

Solución

La integral de la función $f(x) = x^5$ está dada por:

$$I = \int_a^b x^5 dx = \frac{1}{6}(b^6 - a^6) \tag{2.14}$$

El cuadro 2.11 contiene el código fuente para aproximar la integral de la función cuadrática $f(x) = x^2$ por el método de Simpson sobre un intervalo de [0, 10]. En la línea 6, la función cuadrática es utilizada a través de un manejador de funciones: f=@(x)x.^5; e incorporada la función quad(f,a,b) en la línea 7.

Código Fuente 2.11 Método de Simpson

```
%MATLAB Aplicado a Robótica y Mecatrónica
%Capítulo 2 Métodos numéricos
%Editorial Alfaomega
%Fernando Reyes Cortés
%Archivo cap2_simpson.m
```

Método de Simpson

```
1  clc;
2  clear all;
3  close all;
4  a=0; b=10;
5  x=a:b;
6  f= @(x)x.^5; %función cuadrática
7  I_simp=quad(f,a,b);
8  I_a= (1/6)*(b^6-a^6); % método analítico
9  % comparación entre el método analítico y el de Simpson
10 fprintf('Valor analítico=%f \n Simpson:%f \n',I_a,I_simp)
```

La salida del programa cap2_simpson.m es la siguiente:

```
Valor analítico=166666.666667
        Simpson=166666.666667
```

corresponde al intervalo [0, 10].

Funciones de cuadratura

MATLAB tiene varias funciones que calculan la integral de funciones por métodos numéricos conocidas como funciones de **cuadratura**. Adicional a la regla de Simpson quad, existen más opciones que se presentan a continuación:

quad8

La función quad8('nombre_funcion',a,b) retorna el área de la función en el intervalo [a, b] usando la regla de Newton-Cotes 8 panel.

quadl

q = quadl(f,a,b) aproxima la integral de la función f en un intervalo finito [a, b] dentro de un error 1e-06 usando el algoritmo recursivo adaptable de cuadratura de Lobatto. La sintaxis de quadl requiere que la función f sea una función manejador.

quadl acepta funciones vectoriales y retorna un vector con la integral de cada componente. quadl puede ser más eficiente (alta exactitud) que la función quad con funciones suaves.

quadgk

La función quadgk tiene buena eficiencia para funciones oscilatorias, y soporta intervalos de integración infinitos, así como manejar moderadamente valores singulares. Una ventaja de esta función es que soporta integración de contorno a lo largo de trayectorias continuas por trozos. Si el intervalo de integración es infinito, es decir $[a, \infty]$, entonces para que la integral de $f(x)$ exista, $f(x)$ debe caer como

$x \to \infty$. Esta es una condición para poderse utilizar quadgk, particularmente para funciones oscilatorias sobre intervalos infinitos, $f(x)$ debe caer muy rápida.

Una ventaja que tiene la función quadgk es que integrará funciones que tienen puntos singulares no muy fuertes. Si la función tiene puntos singulares dentro del intervalo (a, b), entonces lo recomendable es escribir la integral como la suma de integrales sobre subintervalos con puntos singulares como puntos finales, computar cada integral con quadgk y sumar los resultados.

quadv

La función quadv vectoriza a quad para una función vectorial.

Método de Euler

Un método ampliamente utilizado en robótica y sistemas mecatrónicos es la integración numérica discreta, la cual se puede establecer directamente del método de Euler.

Por ejemplo al derivar con respecto al tiempo la integral $I(t)$ de la función $f(t)$ se obtiene lo siguiente:

$$I(t) = \int_{t_1}^{t_2} f(t)dt \quad \Rightarrow \quad \dot{I}(t) = f(t) \quad \forall\, t \in [t_1, t_2]. \tag{2.15}$$

Partiendo de la definición matemática de la derivada se tiene:

$$\dot{I}(t) = \lim_{t \to 0} \frac{I(t + \quad t) - I(t)}{t} \tag{2.16}$$

(2.17)

donde $t = t_2 - t_1$, representa un infinitésimo intervalo de tiempo.

Si aproximamos a la derivada $\dot{I}(t)$ en su forma discreta $\dot{I}(t) \quad \dot{I}(t_k)$ entonces el tiempo discreto está dado por $t_k = kh$, $k = 1, 2, \cdots, n$ y h es el periodo de muestreo. De esta forma se cumple $t_{k-1} = (k-1)h$ y $\quad t_k = t_k - t_{k-1} = kh - (k-1)h = h$.

Por lo tanto la derivada discreta $\dot{I}(t_k)$ se puede realizar por diferenciación numérica de la posición $I(t_k)$ de la siguiente forma:

$$\dot{I}(t_k) = \frac{I(t_k) - I(t_{k-1})}{h} = f(t_k). \qquad (2.18)$$

La expresión 2.18 se conoce como método de Euler y sirve para estimar la velocidad por diferenciación numérica de la posición.

Despejando $I(t_k)$ de la expresión (2.18), la integral discreta adquiere la siguiente forma:

$$I(t_k) = I(t_{k-1}) + hf(t_k) \qquad (2.19)$$

La expresión (2.19) es muy simple, se convierte en una sumatoria y es adecuada para poderse implementar como algoritmo recursivo.

♣ Ejemplo 2.10

Calcular la integral de la función $f(t) = \operatorname{sen}(t)$ por el método de Euler en forma recursiva para el periodo de integración $t \in [0, 10]$ segundos. El incremento del tiempo es por pasos de un milisegundo.

Solución

En el cuadro 2.12 se presenta el código fuente que implementa el método de Euler para integrar la función $f(t) = \operatorname{sen}(t)$. El intervalo de integración es de 0 a 10 segundos, con incremento de un milisegundo. Este mismo incremento se toma como el valor del periodo de muestreo $h = 0.001$. Para propósitos de comparación el resultado generado por el método de Euler se compara con el método trapezoidal.

C´odigo Fuente 2.12 M´etodo de Euler

%MATLAB Aplicado a Rob´otica y Mecatr´onica
%Cap´ıtulo 2 M´etodos num´ericos
%Editorial Alfaomega
%Fernando Reyes Cort´es
%Archivo cap2_euler.m

M´etodo de Euler

```
clc;
clear all;
close all;
h=0.001; % periodo de muestreo
t=0:h:10; % intervalo de integraci´on
f=sin(t); % funci´on a integrar
I_e=0; % condici´on inicial de la integral por
% m´etodo de Euler
% m´etodo comparativo trapezoidal
I_trap=trapz(t,f);
% m´etodo comparativo regla de Simpson
I_simp=quad('sin',0,10);
[m, n]=size(t); % Orden del vector de tiempo
for k=1:n
    I_e=I_e+h*f(k);
end
fprintf('M´etodo Euler= %f \n M´etodo Trapezoidal= %f \n Regla de Simpson = %f
\n',I_e,I_trap, I_simp)
```

La salida del programa cap2_euler.m produce el siguiente resultado.

M´etodo Euler= 1.838799
M´etodo Trapezoidal= 1.839071
Regla de Simpson= 1.839072

Los resultados corresponden al intervalo [0, 10], con incrementos de 0.001.

Obsérvese que existe un mayor error de integración del método de Euler con respecto a los métodos trapezoidal y el de Simpson. Sin embargo, en robótica se utiliza el método de Euler para estimar la velocidad de movimiento del robot por diferenciación numérica de la posición.

2.5 Sistemas dinámicos de primer orden

Los sistemas dinámicos están formados por ecuaciones diferenciales, ya que representan la estructura matemática fundamental para modelar la dinámica de los robots manipuladores y de cualquier sistema mecatrónico. A través de estas ecuaciones es posible estudiar en detalle todos los fenómenos físicos que se encuentran presentes en la estructura mecánica de los robots.

Particularmente la ecuación diferencial ordinaria de primer orden:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) \tag{2.20}$$

representa la base matemática del modelado dinámico y control de robots manipuladores. Donde $\mathbf{x} \in \mathbb{R}^n$ es la solución de la ecuación diferencial (2.20); también representa la variable de estado, la cual proporciona información sobre la dinámica del sistema, por esta razón $\mathbf{x}$ es que es una función continua del tiempo $\mathbf{x} = \mathbf{x}(t)$. En otras palabras dentro de las propiedades intrínsecas de las variables de estado se encuentra que es una función implícita del tiempo. La variable $\dot{\mathbf{x}}$ representa los cambios de estado dinámicos $\mathbf{x}(t)$ en el tiempo.

La ecuación 2.20 es un sistema dinámico autónomo para modelar dinámica lineal y no lineal de sistemas mecatrónicos y robóticos. Este tipo de ecuación dinámica se emplea en la simulación, diseño de algoritmos de control, análisis y estudio de fenómenos físicos y construcción de sistemas mecatrónicos.

Prácticamente cualquier sistema dinámico autónomo (que no depende de manera explícita del tiempo) cuyo modelo matemático esté caracterizado por ecuaciones diferenciales de cualquier orden puede ser convertido mediante un adecuado cambio de variables de estado a la forma de la ecuación 2.20, entonces se puede simular mediante un adecuado procedimiento de integración numérica.

Método de Runge-Kutta

Uno de los métodos más importante para integración numérica de ecuaciones diferenciales de primer orden (2.20) son los métodos de Runge-Kutta. Dichos métodos se basan en aproximar la solución a través de series de Taylor. El método más simple, el denominado método de primer orden, usa una serie de expansión de Taylor de primer orden, el método de segundo orden usa la serie de expansión de Taylor de segundo orden, y así sucesivamente (el método de Euler es equivalente al de primer orden de Runge-Kutta). MATLAB tiene funciones del método de Runge-Kutta para los órdenes segundo, tercero, cuarto y quinto.

La serie de Taylor para evaluarla en el t_k-ésimo tiempo a la función $x(t_k)$ está dada por la siguiente expresión:

$$x(t_k) = x(t_{k-1}) + (t_k - t_{k-1})\dot{x}(t_{k-1}) + \frac{(t_k - t_{k-1})^2}{2!}\ddot{x}(t_{k-1}) + \dots + \frac{(t_k - t_{k-1})^n}{n!} \overbrace{\dot{\ddot{x}}\dots}^{n \text{ derivadas}}(t_{k-1}) + \cdots \tag{2.21}$$

El término $t_k - t_{k-1}$ representa un pequeño intervalo el cual será representado por $h = t_k - t_{k-1} = kh - (k-1)h$, entonces la serie de Taylor queda de la siguiente forma:

$$x(t_k) = x(t_{k-1}) + h\dot{x}(t_{k-1}) + \frac{h^2}{2!}\ddot{x}(t_{k-1}) + \dots + \frac{h^n}{n!} \overbrace{\dot{\ddot{x}}\dots}^{n \text{ derivadas}}(t_{k-1}) + \dots \tag{2.22}$$

Método de Runge-Kutta de primer orden

La integración numérica de la ecuación 2.20 por el método de Runge-Kutta de primer orden está dado de la siguiente manera:

$$x(t_k) = x(t_{k-1}) + h\dot{x}(t_{k-1}) = x(t_{k-1}) + hf(t_{k-1}) \tag{2.23}$$

La interpretación geométrica de la ecuación (2.23) significa que el valor de estimación de $x(t_k)$ es igual a la línea tangente que la une con el valor $x(t_{k-1})$ como se muestra en la figura 2.8. Este es un proceso iterativo para todos los puntos $x(t_k)$, con $k = 1, 2, \cdots, n$. El primer método de Runge-Kutta es muy simple, ya que aproxima

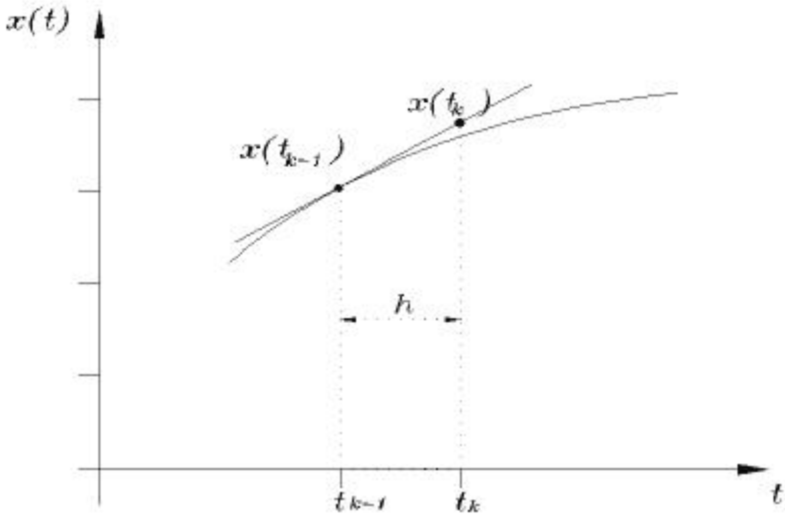


Figura 2.8 Cálculo de $x(t_k)$ usando el primer el método de Runge-Kutta primer orden.

la función con una serie de Taylor corta que une los puntos t_k y t_{k-1} con líneas tangentes, por lo cual la exactitud es pobre si el paso $h = t_k - t_{k-1}$ es grande o si la pendiente de la función cambia rápidamente.

Método de Runge-Kutta de orden mayor

Los métodos de Runge-Kutta de orden mayor (dos, tercero, cuarto, y quinto) se usan para aproximar funciones desconocidas; la aproximación se realiza a través de varias líneas tangentes, por lo tanto la exactitud es mejorada. Por ejemplo, en el método de integración de cuarto orden usa la serie de Taylor con las primeras cuatro derivadas, es decir la estimación de la función $x(t_k)$ es a través de 4 líneas tangentes.

Funciones ode

MATLAB tiene las funciones para integrar la soluci´on num´erica de ecuaciones diferenciales ordinarias de primer orden llamadas ode.

ode45

La sintaxis de la funci´on ode45 es la siguiente:

```
[t,x]=ode45('nombre_funcion',ts,cond_iniciales,opciones)
```

La funci´on ode45 utiliza los m´etodos de Runge-Kutta de cuarto y quinto orden. nombre_funcion representa una funci´on M-File donde est´a implementado el sistema din´amico en la estructura matem´atica $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$.

La funci´on ode45 retorna la soluci´on del sistema $\dot{\mathbf{x}} = \mathbf{f}(x)$, es decir $\mathbf{x}$, as´ı como el vector de tiempo t. El tiempo de simulaci´on o el intervalo de integraci´on se encuentra especificado por $ts = [t_{inicial}, t_{final}]$, por ejemplo se puede especificar como un intervalo [0, 10]. Tambi´en es v´alido expresarlo como: $ts = 0 : 0.001 : 10$, el cual incluye incrementos de tiempo de un milisegundo. Las condiciones iniciales se encuentran determinadas por cond_iniciales, su forma depende del orden del sistema. Por ejemplo, para un sistema escalar cond_inciales=0; para el caso vectorial $\mathbf{x}(0) \in IR^3$, tenemos cond_inciales=[0;0;0].

El cuarto par´ametro de opciones es muy importante debido a que contiene las propiedades de integraci´on num´erica, y de eso depende la simulaci´on del sistema. Para tal efecto se emplea la funci´on odeset de la siguiente forma:

```
opciones=odeset('RelTol',1e-3,'AbsTol',1e-6,'InitialStep',1.0e-3,'MaxStep',1.0e-3)
```

Error de integraci´on

En cada i-´esimo paso de integraci´on se estima un error local de la i-´esima componente de la soluci´on; este error debe ser menor o igual que el error aceptable, el cual es una funci´on de la tolerancia relativa

RelTol y de la tolerancia absoluta AbsTol. La expresi´on para $e(i)$ satisface la siguiente condici´on: $|e(i)| = \max(\text{RelTol}|x(i), \text{AbsTol}(i)|)$.

RelTol	RelTol representa el error relativo de tolerancia que se aplica a todas las componentes x_i del vector soluci´on $\mathbf{x} \in IR^n$.Este par´ametro tambi´en controla el n´umero correcto de d´ıgitos en todas las componentes x_i. El valor t´ıpico RelTol= 1e-3, el cual corresponde al 0.1 % de exactitud.
AbsTol	Error absoluto de tolerancia que se aplica a todas las componentes individuales x_i del vector soluci´on $\mathbf{x}$. Tambi´en determina la exactitud cuando la soluci´on se aproxima a cero. Si AbsTol es un vector, entonces su dimensi´on debe ser la misma del vector $\mathbf{x}$. El valor de cada componente de AbsTol se aplica a la correspondiente componente x_i. Si AbsTol es un escalar, ese valor se aplica a todas las componentes x_i del vector soluci´on $\mathbf{x}$. Generalmente, el valor t´ıpico que se le asigna a AbsTol es: AbsTol= 1e-6.
NormControl	NormControl significa la norma del error relativo. Esta opci´on solicita a la funci´on ode45 que el error de integraci´on en cada paso cumpla con la siguiente condici´on: $e \leq \max(\text{RelTol } \mathbf{x}, \text{AbsTol})$. Se habilita/deshabilita como: on (off). Para funciones suaves, la funci´on ode45 entrega un error equivalente a la exactitud solicitada. La exactitud es menor para problemas donde el intervalo de integraci´on es grande y para problemas moderadamente inestables.

Paso de integraci´on

InitialStep	Especifica el valor inicial del paso de integraci´on. Representa una cota superior en la magnitud del primer paso. En caso de no especificarlo, entonces el tamaño del paso se obtiene como la pendiente de la soluci´on en el tiempo inicial. Si la pendiente de todas las componentes de la soluci´on es cero, entonces el procedimiento puede generar un tamaño de paso muy grande. Por lo tanto, es recomendable iniciar con un valor adecuado esta opci´on.
MaxStep	Representa una cota superior del tamaño del paso, es un valor escalar positivo que se obtiene de la siguiente forma: $0.1\lvert t_{inicial} - t_{final}\rvert$. La ecuaci´on diferencial tiene soluciones o coeficientes peri´odicos, es recomendable inicializar MaxStep=$\frac{1}{4}$ del periodo.

La funci´on ode45 es la que presenta mayor exactitud en el m´etodo de integraci´on para el sistema $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$. Sin embargo, debido a esto tambi´en puede tomar m´as tiempo en dicho proceso.

A continuaci´on se describe otras opciones de funciones para resolver ecuaciones diferenciales ordinarias de primer orden, cuya sintaxis es exactamente la misma que la funci´on ode45. Sin embargo, tienen otras caracter´ısticas operativas.

ode23

La funci´on ode23 utiliza el segundo y tercer m´etodo de integraci´on. Tiene menor exactitud comparada con la funci´on ode45. Sin embargo, puede realizar m´as r´apido el proceso de integraci´on num´erica. La sintaxis est´a dada por:

```
[t,x]=ode23('nombre funcion',ts,cond iniciales,opciones)
```

ode113

```
[t,x]=ode113('nombre_funcion',ts,cond_iniciales,opciones)
```

Esta funci´on se recomienda para resolver sistemas din´amicos complicados; mantiene un error de integraci´on riguroso. La funci´on ode113 es un m´etodo de integraci´on de orden variable propuesto por Adams-Bashforth-Moulton.

ode15s

```
[t,x]=ode15s('nombre_funcion',ts,cond_iniciales,opciones)
```

Para el caso en que el proceso de integraci´on de la funci´on ode45 sea muy lento debido a la rigidez del sistema din´amico, entonces es recomendable usar la funci´on ode15s. Esta funci´on es de orden variable, multi pasos basada en diferenciaci´on num´erica (backward o m´etodo Gear).

ode23s

```
[t,x]=ode23s('nombre_funcion',ts,cond_iniciales,opciones)
```

Es muy útil para sistemas din´amicos masa-resorte con alta rigidez en que el resorte la matriz de masas es constante. La funci´on ode23s se basa en el m´etodo modificado de segundo orden de Rosenbrock. Debido a que el proceso de integraci´on es por pequeños pasos, puede ser m´as eficiente que la funci´on ode15s para tolerancias pequeñas en el error de integraci´on. Adem´as es mucho m´as efectivo en sistemas din´amicos con rigidez donde ode15s no lo es.

ode23t

```
[t,x]=ode23t('nombre_funcion',ts,cond_iniciales,opciones)
```

La función ode23t emplea el método trapezoidal usando una interpolación libre. Es recomendable para resolver sistemas dinámicos moderados en la rigidez del resorte, sin utilizar amortiguamiento. La exactitud en la integración es baja.

ode23tb

```
[t,x]=ode23tb('nombre_funcion',ts,cond_iniciales,opciones)
```

La función ode23tb es una combinación de métodos de Runge-Kutta y diferenciación numérica (backward) de segundo orden. Para sistemas dinámicos con alta rigidez del resorte y errores de integración muy pequeños, la exactitud de integración es baja.

Simulación de sistemas dinámicos $\dot{\mathbf{x}} = \mathbf{f}(x)$

A continuación se presenta una serie de ejemplos ilustrativos cuya finalidad es mostrar el empleo de las funciones ode en simulación de sistemas mecatrónicos y robots manipuladores.

♣ Ejemplo 2.11

Considere un sistema dinámico lineal escalar dado de la siguiente forma:

$$
\begin{aligned}
\dot{x}(t) &= -ax(t) + bu(t) \\
y(t) &= cx(t)
\end{aligned}
$$

donde x es la solución del sistema dinámico y representa la variable de estado. La entrada del sistema es $u(t)$ y la respuesta es y. Los parámetros están dados por $a, b, c \in I\!R_+$.

Realizar la simulación del sistema usando integración numérica a través de la función ode45. Para una entrada $u = 1$, y parámetros $a = 1.33$, $b = 2.6$ y $c = 1$.

Soluci´on

El sistema din´amico lineal escalar tiene una funci´on de transferencia (control cl´asico) dada por:

$$\begin{aligned} \frac{x}{u} &= \frac{b}{s+a} \\ y &= cx = \frac{cb}{s+a}u \end{aligned}$$

donde $s = jw$, j representa la parte imaginaria o compleja, y $w \in \mathbb{R}_+$ es la frecuencia en rad/seg. El par´ametro $a \in \mathbb{R}_+$ es el ancho de banda del sistema y $b \in \mathbb{R}_+$ significa la ganancia del sistema. Por otro lado, $c \in \mathbb{R}_+$ es una ganancia de un amplificador operacional que acopla de manera adecuada la impedancia entre la variable de estado x y la respuesta del sistema y.

Para obtener la ecuaci´on diferencial (modelo din´amico) a partir de la funci´on de transferencia, entonces s no representa frecuencia, en este caso se interpreta como un operador $s = \frac{d}{dt}$:

$$\begin{aligned} \frac{x}{u} = \frac{b}{s+a} &\Rightarrow sx + ax = bu \\ &\Rightarrow \dot{x} = -ax + bu \\ y &= cx \end{aligned}$$

donde se ha empleado que $sx = \frac{d}{dt}x = \dot{x}$.

La figura 2.9 muestra el diagrama a bloques del sistema din´amico lineal.

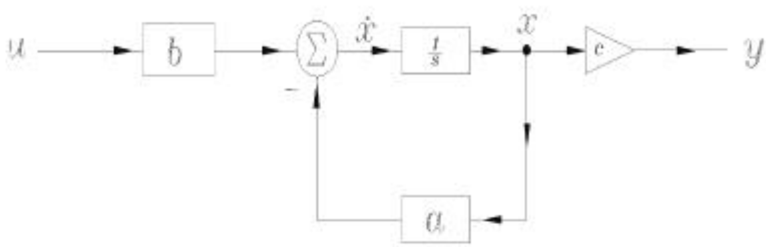


Figura 2.9 Diagrama a bloques de un sistema din´amico lineal.

El programa que implementa al sistema din´amico lineal o planta de estudio se encuentra contenido en el cuadro 2.13 archivo cap2_ejemplo26 (M-File). Para resolver el sistema din´amico, se emplea la funci´on de integraci´on num´erica ode45. Por facilidad, se emplea una entrada constante de magnitud unitaria. Los valores de los par´ametros de la planta son $a = 1.33$ y $b = 2.6$. La ganancia del amplificador es unitaria $c = 1$.

Para ejecutar correctamente la funci´on cap2_ejemplo26 se requiere del programa simuejemplo26 que se encuentra listado en el cuadro 2.13, donde se definen las propiedades de integraci´on usando la funci´on odeset, las cuales son incorporadas en la funci´on ode45. El tiempo de simulaci´on es de 0 a 10 segundos con incrementos de una mil´esima de segundo.

La salida del programa 2.13 se presenta en la figura 2.10, observe que la respuesta a una entrada constante de magnitud unitaria no produce sobre impulso para sistemas lineales de primer orden. En estado estacionario, la salida $x(t)$ llega a un valor de 1.9549 (frecuencia cero o DC: $s = 0$) debido a que la funci´on de transferencia adquiere la forma $\frac{b}{a} = 1.9549$.

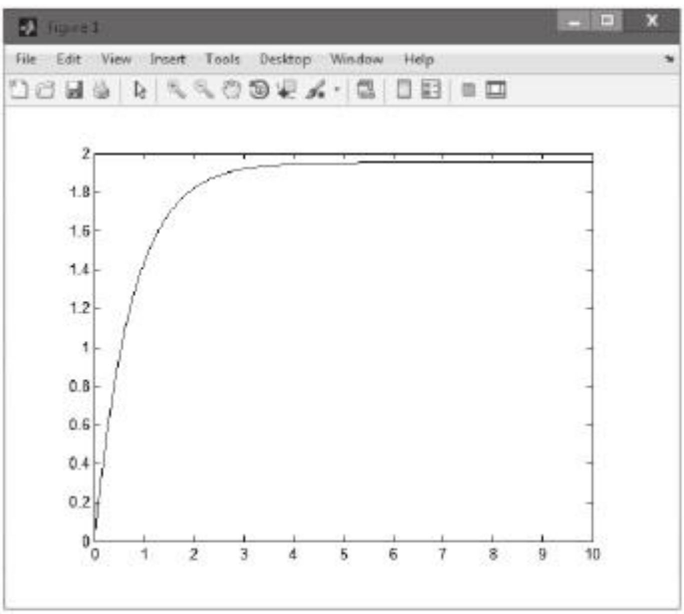


Figura 2.10 Respuesta a una entrada constante unitaria del sistema lineal escalar.

Código Fuente 2.13 Sistema lineal escalar

```
%MATLAB Aplicado a Robótica y Mecatrónica
%Capítulo 2 Métodos numéricos
%Editorial Alfaomega
%Fernando Reyes Cortés
%Archivo cap2_ejemplo26
% Simulación de un sistema lineal escalar
% de la forma ẋ = −ax + bu
%Entradas de la función: t tiempo
% x vector de estados
% ẋ (xp) es la velocidad del estado x
% u es la entrada al sistema
% a, b son los parámetros
% Salida derivada de la variable de estados: xp
% Para su correcta ejecución requiere del archivo
%cap2_simuejemplo26
```

Sistema lineal escalar

```
function xp =cap2_ejemplo26(t,x)
    %parámetros del sistema: a, b.
    %el parámetro a representa el ancho de banda
    %del sistema dinámico lineal.
    a=1.33;
    %la proporción b/a es la ganancia
    %de la función de transferencia en frecuencia cero.
    b=2.6;
    %señal de entrada unitaria
    u=1;
    %sistema dinámico lineal escalar
    xp=-a*x+b*u;
    %fin de función
end
```

Código Fuente 2.14 Simulación del ejemplo 2.11

```
%MATLAB Aplicado a Robótica y Mecatrónica
%Capítulo 2 Métodos numéricos
%Editorial Alfaomega
%Fernando Reyes Cortés
%Archivo cap2_simuejemplo26
% Realiza la simulación de un sistema dinámico lineal escalar
% Requiere del archivo cap2_ejemplo26.m
```

Simulación del ejemplo 2.11

```
clc;
clear all;
close all;
%tiempo de simulación
ti=0; %tiempo inicial
tf = 10; % tiempo de simulación (segundos)
h=0.001; % incremento de un milisegundo
ts=ti:h:tf; % vector de tiempo
cond_iniciales=0;
%opciones del proceso de integración numérica
opciones=odeset('RelTol',h, 'AbsTol',1e-06,'InitialStep',h,'MaxStep',h);
% función ode45 para integración numérica
% solución del sistema x(t) por integración numérica Runge-Kutta de cuarto y
quinto orden:
disp('Simulación de un sistema lineal escalar')
[t,x]=ode45('cap2_ejemplo26',ts,cond_iniciales,opciones);
% ganancia del amplificador que acopla impedancia de la variable de estados x(t)
con la salida.
c=1;
%salida del sistema
y=c*x;
% gráfica del sistema lineal escalar
plot(t,y)
```

♣ ♣ Ejemplo 2.12

Considere un sistema dinámico lineal de segundo orden, dado por la siguiente ecuación:

$$
\begin{aligned}
\mathbf{x}(t) &= \begin{bmatrix} 0 & 1 \\ -\omega_n^2 & -2\rho\omega_n \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 0 \\ \omega_n \end{bmatrix} u(t) \\
y(t) &= [1 \quad 0]\,\mathbf{x}(t)
\end{aligned}
$$

donde $\mathbf{x} \in \mathbb{R}^n$ es el vector solución del sistema dinámico y representa la variable de estado. La entrada del sistema es $u(t)$ y la respuesta es y. Los parámetros están dados por: $\omega_n \in \mathbb{R}_+$ es la frecuencia natural del sistema, $\rho \in \mathbb{R}_+$ es el factor de amortiguamiento (damping) y $s = jw$, $\mathbf{j}$ es la componente imaginaria o compleja, w es la frecuencia en rad/seg, t es la evolución del tiempo.

Solución

El sistema dinámico lineal de segundo orden puede ser deducido de la siguiente función de transferencia

$$
\frac{y}{u} = \frac{\omega_n}{s^2 + 2\rho\omega_n s + \omega_n^2}.
$$

Manipulando la función de transferencia y considerando que $s = \frac{d}{dt}$, $s^2 = \frac{d^2}{dt^2}$, $s^2 y(t) = \ddot{y}(t)$, $sy(t) = \dot{y}(t)$, se obtiene lo siguiente:

$$
\begin{aligned}
\frac{y}{u} = \frac{\omega_n}{s^2 + 2\rho\omega_n s + w_n^2} \quad &\Rightarrow \quad s^2 y(t) + 2\rho\omega_n s y(t) + \omega_n^2 y(t) = \omega_n u(t) \\
&\Rightarrow \quad \ddot{y}(t) + 2\rho\omega_n \dot{y}(t) + \omega_n^2 y(t) = \omega_n u(t).
\end{aligned}
$$

Ahora se realiza un cambio adecuado de variable. Sea $x_1(t) = y(t)$, $x_2(t) = \dot{x}_1(t) = \dot{y}(t)$, entonces

$$
\dot{x}_2(t) + 2\rho\omega_n x_2(t) + \omega_n^2 x_1(t) \quad = \quad \omega_n u(t).
$$

Por lo tanto, el sistema dinámico de la forma $\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + Bu(t)$ está dado de la siguiente forma:

$$\underbrace{\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}_{\dot{\mathbf{x}}} = \underbrace{\begin{bmatrix} 0 & 1 \\ -\omega_n^2 & -2\rho\omega_n \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}_{\mathbf{x}} + \underbrace{\begin{bmatrix} 0 \\ \omega_n \end{bmatrix}}_{B} u$$

$$y = \mathbf{c}^T \mathbf{x} = [1 \quad 0] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}.$$

El programa que implementa al sistema dinámico lineal de segundo orden se encuentra descrito en el cuadro 2.15 que corresponde al archivo cap2 ejemplo27 (M-File). Para resolver el sistema dinámico por integración numérica, se emplea la función ode45. Por facilidad, se emplea una entrada constante unitaria para estudiar la respuesta del sistema para varios valores de amortiguamiento ρ y frecuencia de resonancia ω_n.

Para ejecutar correctamente la función cap2 ejemplo27 se requiere del programa simuejemplo27 que se encuentra listado en el cuadro 2.15, donde se definen las propiedades de integración usando la función odeset, las cuales son incorporadas en la función ode45. El tiempo de simulación es de 0 a 10 segundos con incrementos de una milésima de segundo.

La salida del programa 2.15 se presenta en la figura 2.11, observe que la respuesta a una entrada constante de magnitud unitaria puede generar varios sobre impulsos, esto depende del valor que tome ρ. Por ejemplo, para valores de $0 < \rho < 1$ se le conoce como sub-amortiguado, se caracteriza por presentar una etapa transitoria muy abrupta, rápida y compuesta por varios sobre impulsos y oscilaciones. Cuando $\rho = 1$ se le conoce como amortiguamiento crítico, disminuye considerablemente las oscilaciones. Sobre amortiguado es para valores de $\rho > 1$ el cual no presenta oscilaciones, es muy lento para llegar a la etapa transitoria.

La figura 2.11 fue obtenida variando los valores de $\rho = 0.2, 0.4, 0.6, 0.8, 1, 2$, para generar la respuesta sub-amortiguada, amortiguamiento crítico y sobre amortiguada. Se puede observar las oscilaciones muy pronunciadas que presenta en la etapa sub-amortiguada y a medida que ρ va aumentando su valor, entonces las oscilaciones gradualmente van disminuyendo hasta desaparecer completamente para $\rho > 1$. Cuando $\rho = 0$, el sistema oscila a la frecuencia natural de resonancia ω_n.

C´odigo Fuente 2.15 Sistema lineal de segundo orden

```
%MATLAB Aplicado a Rob´otica y Mecatr´onica
%Cap´ıtulo 2 M´etodos num´ericos
%Editorial Alfaomega
%Fernando Reyes Cort´es
%Archivo cap2_ejemplo27
%Sistema din´amico lineal de segundo orden
%Para su correcta ejecuci´on requiere del
%programa cap2_simuejemplo27
%Entradas: t tiempo
% x vector de estados
% Salida xp=Ax+Bu
% la matriz A est´a formada por ρ factor
% de amortiguamiento y ωn la frecuencia natural
% de resonancia mec´anica.
```

Sistema lineal de segundo orden

```
function xp =cap2_ejemplo27(t,x)
    % frecuencia natural de resonancia
    wn=1;
    % factor de amortiguamiento
    rho=1;
    A=[ 0, 1;
          -wn^2, -2*rho*wn];
    %matriz B
    B=[0;
         wn];
    %señal de entrada
    u=1;
    %sistema din´amico lineal
    xp=A*x+B*u;
end
```

C´odigo Fuente 2.16 Simulaci´on del ejemplo 2.12

%MATLAB Aplicado a Rob´otica y Mecatr´onica
%Cap´ıtulo 2 M´etodos num´ericos
%Editorial Alfaomega
%Fernando Reyes Cort´es
%Archivo cap2_simuejemplo27
% Simulaci´on: sistema din´amico lineal de primer orden
% requiere del archivo cap2_ejemplo27.m

Simulaci´on del ejemplo 2.12

```
clc; clear all; close all;
% par´ametros de simulaci´on:
ti=0; tf = 10; h=0.001;
% intervalo de simulaci´on
ts=ti:h:tf;
cond_iniciales=[0;0];
opciones=odeset('RelTol',h, 'AbsTol',1e-06,'InitialStep',h,'MaxStep',h);
disp('Simulaci´on de un sistema lineal de segundo orden')
[t,x]=ode45('cap2_ejemplo27',ts,cond_iniciales,opciones);
y=x(:,1); %salida del sistema
plot(t,y)
```

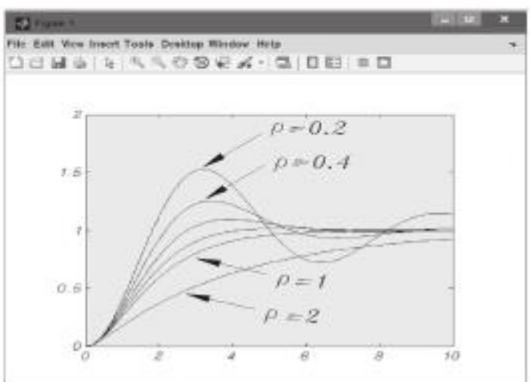


Figura 2.11 Respuesta a una entrada escal´on.

2.6 Resumen

Métodos de integración y diferenciación numérica representan una herramienta poderosa de cómputo en la implementación práctica para análisis y estudios de esquemas de control en simulación y en evaluación práctica de sistemas mecatrónicos y robots manipuladores.

Existen varios métodos prácticos para resolver una integral, entre los más importantes se encuentran trapezoidal, Simpson, Euler recursivo, Runge-Kutta. Sin embargo, dependiendo de la naturaleza del problema, la selección de un método influye directamente en el desempeño, calidad y fidelidad del proceso de simulación o experimentación.

Por ejemplo, el método recursivo de Euler es muy rápido y puede ser incluido en el código del experimento para realizarse en tiempo real. Sin embargo, la exactitud con que aproxima la integral es pobre en contraste con otros métodos como el trapezoidal o el de Simpson, que pudieran ser mejores opciones para propósitos de simulación.

Para el caso de esquemas de control de robots manipuladores donde se involucre alguna integral como el caso del algoritmo proporcional integral derivativo, el método recursivo de Euler puede ser una buena opción a pesar de que la exactitud en la aproximación de dicha integral no sea bueno. Esta característica no es importante debido a que el desempeño del algoritmo de control depende de la sintonía de las ganancias y por lo tanto la pobre exactitud queda absorbida por la ganancia integral.

Para el estudio y análisis de la dinámica en sistemas mecatrónicos o robots manipuladores es importante convertir mediante un adecuado cambio de variables de estado (variables fase) el modelo dinámico a la forma de una ecuación diferencial de primer orden $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$.

La simulación de estos sistemas dinámicos se realiza por medio de la función ode45 (método de Runge-Kutta para orden cuarto y quinto), con pasos de integración de un milisegundos. Las propiedades del método de integración, así como las condiciones iniciales del vector $\mathbf{x}(0)$ se configuran con la función odeset.

Parte I referencias selectas

Existe una extensa literatura sobre los aspectos b´asicos de MATLAB, as´ı como m´etodos num´ericos. A continuaci´on se presenta literatura selecta correspondiente a la Parte I donde el lector puede profundizar sobre los t´opicos presentados.

Cap´ıtulo 1 Conceptos b´asicos

http://www.mathworks.com

Delores M. Etter. "Engineering problem solving with MATLAB". Second edition. Prentice-Hall. 1997.

Shoichiro Nakamura. "An´alisis num´erico y visualizaci´on gr´afica con MATLAB ". Pearson Eduaci´on. 1997.

Ashish Tewari. "Modern control design with MATLAB and Simulink". John Wiley & Sons, LTD. 2002.

The MathWorks. "Matehmatics: The language of technical computing". The MathWorks. 2005.

The MathWorks "Getting started with MATLAB 7 ". The MathWorks. 2007.

David B´aez L´opez & Ofelia Cervantes Villagomez. " MatLab con aplicaciones a la ingenier´ıa, f´ısica y finanzas". 2da edici´on Alfaomega, diciembre 2011.

Capítulo 2 Métodos numéricos

- Tom M. Apostol. 'Calculus'. Reverté. 2da. ed., vols. I y II, 2006.
- Norman B. Hasser, Joseph P. La Salle, Joseph A. Sullivan. 'Análisis matemático'. Trillas, 1970.
- Watson Fulks. 'Cálculo avanzado'. Limusa. 1983.
- Adrian Biran & Moshe Breiner. 'MatLab for engineers'. Addison Wesley, 1997.
- William H Press, Saul A. Teukoisky, William T. Vetterling, & Brian P. Flannery. 'Numerical recipes in C++'. Cambridge university Press. 2002.

Parte I problemas propuestos

En esta sección se presentan una serie de ejercicios con la finalidad que el lector mejore sus conocimientos sobre métodos numéricos.

Capítulo 1 Conceptos básicos

1.1 Considere el programa 1.13, realizar los cambios necesarios en el código fuente para encontrar los mínimos y máximos locales de la función.

1.2 Dado la siguiente secuencia de números:

$$\text{datos} = \{1, 4, -3, 8, 12, 7, 2, 5, -12, 5, 7, 1, 0, 2, 22, 15, 4, 6, 2, -1, -3, 11\}$$

escribir un programa donde se use instrucciones **if**, **for**, while para realizar lo siguiente:

(a) Ordenar los números de menor a mayor.

(b) Ordenar los números de mayor a menor.

(c) Encontrar el mínimo y máximo.

1.3 Calcular la norma euclidiana de los siguientes vectores:

$$\mathbf{x} = \begin{bmatrix} 1 \\ -0.888 \\ 4.35 \\ 3 \\ 4 \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} 4 \\ 5.67 \\ 6 \\ -0.90 \end{bmatrix} \quad \mathbf{z} = \begin{bmatrix} 0.001 \\ 4 \\ 7 \\ -12.333 \\ \frac{3}{4} \end{bmatrix}.$$

Para todos los vectores $\mathbf{x} \in IR^5$, $\mathbf{y} \in IR^4$, $\mathbf{z} \in IR^5$, obtener la norma euclidiana a través de los siguientes métodos.

(a) $\|\mathbf{x}\| = \sqrt{\mathbf{x}^T\mathbf{x}}$.

(b) $\|\mathbf{x}\| = \sqrt{\sum_{i=1}^{n} x_i^2}$.

(c) $\|\mathbf{x}\| = \sqrt{x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2}$.

(d) $\|\mathbf{x}\| =$ norma($\mathbf{x}$,2).

1.4 En referencia del ejercicio inmediato anterior considere los vectores $\mathbf{x}, \mathbf{y}$; implementar la multiplicación y división entre las componentes de los vectores a través operaciones con arreglos.

1.5 Obtener la gráfica de las siguientes funciones:

(a) $f(x) = x^2, \; -10 < x < 10$.

(b) $f(x) = \tan(x), \; -15 < x < 15$.

(c) $f(x) = \frac{x}{1+x^2}, \; -100 < x < 100$.

(d) $f(x) = [1 - e^{-0.01*x}]\,\mathrm{senh}(x), \; -3 < x < 3$.

Capítulo 2 Métodos numéricos

2.1 Considere los siguientes sistemas de ecuaciones lineales:

$$\begin{bmatrix} 2 \\ 4 \\ 9 \end{bmatrix} = \begin{bmatrix} 2x_1 + 3.33x_2 + 4.56x_3 \\ 8x_1 + 12.56x_2 + 8.12x_3 \\ -3x_1 + 8.45x_2 - 7.12x_3 \end{bmatrix}$$

$$\begin{bmatrix} 6 \\ -3 \\ 10.44 \\ 6.12 \\ 0 \end{bmatrix} = \begin{bmatrix} 1.11x_1 + 8.13x_2 + 2.14x_3 + 6.73x_4 + 12.55x_5 \\ -4x_1 + -8.34x_2 + 4.67x_3 + 4.10x_4 + 2.48x_5 \\ 1.2x_1 + -5.6x_2 + 2.12x_3 + -3.24x_4 + 12.55x_5 \\ 2.35x_1 + 4.5x_2 + 5.89x_3 + 0.78x_4 + 0.34x_5 \\ 0.67x_1 + 1.21x_2 + -0.98x_3 + 0.34x_4 + 16.12x_5 \end{bmatrix}$$

Resuelva el sistema de ecuaciones usando los siguientes métodos

(a) $\mathbf{x} = A^{-1}\mathbf{y}$.

(b) $\mathbf{x}$=inv(A)$\mathbf{y}$.

(c) $\mathbf{x}$=A\\$\mathbf{y}$.

(d) $\mathbf{x}$=linesolve(A,y).

(e) Regla de Cramer.

2.2 ¿Qué argumentos puede utilizar para obtener la derivada de la función $|t|$?

2.3 ¿Cómo puede obtener la derivada con respecto al tiempo de una función discontinua como la función signo(t)?

2.4 Aproximar la derivada con respecto al tiempo por diferenciación numérica de las siguientes funciones:

$$\begin{aligned} f_1(t) &= t^7 \\ f_2(t) &= |t| \\ f_3(t) &= \text{signo}(t) \\ f_4(t) &= \tanh(t) \\ f_5(t) &= \text{atan}(t) \end{aligned}$$

Comparar los resultados con el correspondiente método analítico.

2.5 Calcular la integral de las siguientes funciones:

$$I_1 = \int x^7 dx \qquad I_2 = \int |x| dx$$

$$I_3 = \int \text{signo}(x)\, dx \qquad I_4 = \int \tanh(x)\, dx$$

$$I_5 = \int \text{atan}(x)\, dx$$

considere el intervalo de integración $\mu \in [-10, 10]$.

Realice el cálculo de la integral para cada función utilizando los siguientes métodos:

(a) Trapezoidal.
(b) Simpson.
(c) Euler.

comparar y analizar los resultados.

2.6 Utilizar el programa del ejemplo 2.7 para calcular la integral de la siguientes funciones

(a) e^{-x}
(b) $\cos(x)$
(c) x^6

realizando un barrido del intervalo de integración por pasos de 0.001 para $x \in [-10, 10]$. Comparar los resultados obtenidos con su respectivo método analítico.

2.7 Realizar la simulación de los siguientes sistemas dinámicos ($t \in [0, 10]$):

(a)

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} \text{sen}(x_2) \\ x_{31} \end{bmatrix}$$

(b)

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} \text{atan}(x_1 + x_2) \\ \cos(x_2 x_1) \end{bmatrix}$$

Parte II

Cinemática

La **Parte II** está dedicada al estudio de la cinemática directa de robots manipuladores y sistemas mecatrónicos, y se componen de tres capítulos. El capítulo 3 incluye **Preliminares matemáticos** para apoyar la representación de matrices homogéneas. El capítulo 4 **Cinemática directa** cubre los temas de cinemática directa e inversa, cinemática diferencial y jacobiano de robots manipuladores, así como la metodología Denavit-Hartenberg. El capítulo 5 **Cinemática directa cartesiana** está destinado para analizar el modelo que relaciona las coordenadas cartesianas con las coordenadas articulares para las principales configuraciones de los robots industriales (sin tomar en cuenta la orientación de la herramienta de trabajo).

- **Capítulo 3 Preliminares matemáticos** incluye los temas de producto punto entre vectores, matrices ortogonales y sus propiedades matemáticas, reglas de rotación, matrices de transformación homogénea para traslación y rotación, así como la presentación de varios ejemplos ilustrativos y el desarrollo de librerías para **MATLAB**.
- **Capítulo 4 Cinemática directa** presenta los conceptos de cinemática directa, cinemática inversa, cinemática diferencial y el jacobiano del robot. También explica la metodología de Denavit-Hartenberg la cual usa un grupo de parámetros geométricos para determinar la matriz de transformación homogénea que caracteriza al modelo cinemático de robots manipuladores industriales.
- **Capítulo 5 Cinemática directa cartesiana** realiza un análisis detallado de la cinemática cartesiana de las principales configuraciones de robots industriales. Un conjunto de librerías se desarrollan para llevar a cabo simulación con robots manipuladores.

La parte II concluye con:

- **Referencias selectas**
- **Problemas propuestos**

Capítulo

3 Preliminares matemáticos

$$H_i^{i-1} = \begin{bmatrix} R_i^{i-1} & \mathbf{d}_i^{i-1} \\ \mathbf{0}^T & 1 \end{bmatrix}$$

Objetivos

Presentar los preliminares matemáticos para la cinemática directa de robots manipuladores y sistemas mecatrónicos, asimismo desarrollar librerías en lenguaje **MATLAB** (toolbox) para las matrices de transformación homogénea de rotación y traslación.

Objetivos particulares:

- Producto interno de vectores.
- Matrices de rotación y traslación.
- Transformaciones homogéneas.
- Librerías de matrices homogéneas para **MATLAB**.

3.1 Introducción

El posicionamiento del extremo final del robot en el espacio tridimensional (pose) requiere de 6 coordenadas: 3 coordenadas para la posición cartesiana y 3 coordenadas para la orientación de la herramienta de trabajo. A la relación que existe entre las coordenadas articulares del robot con las coordenadas cartesianas y la orientación de la herramienta de trabajo colocada en el extremo final del robot se le denomina cinemática directa.

Como parte de la representación matemática de la cinemática directa de robots manipuladores se encuentra el uso de transformaciones homogéneas para representar orientación y traslación de la herramienta de trabajo, con respecto al sistema de referencia fijo ubicado generalmente en la base del robot.

Como preámbulo al tema de matrices de transformación homogénea están los conceptos de producto punto o escalar entre vectores y matrices ortogonales.

El producto punto permite utilizar proyecciones de ortogonalidad de los ejes principales de un sistema de referencia $\Sigma_1(x_1, y_1, z_1)$ relativo a otro sistema de referencia $\Sigma_0(x_0, y_0, z_0)$. Otro de los temas de prerrequisito es el de las matrices ortogonales para modelar la orientación y traslación de la herramienta de trabajo respecto al sistema fijo del robot $\Sigma_0(x_0, y_0, z_0)$. Resaltan las propiedades matemáticas de las matrices ortogonales que facilitan el análisis y descripción de movimientos de rotación y traslación. En función de las propiedades matemáticas de las matrices ortogonales se generan diferentes reglas de rotación.

Las matrices homogéneas incluyen estos conceptos para ofrecer una representación compacta de la matriz de rotación, que determina la orientación relativa de un sistema de referencia $\Sigma_1(x_1, y_1, z_1)$ con respecto a un sistema de referencia fijo $\Sigma_0(x_0, y_0, z_0)$ y del vector de coordenadas o de traslación $\mathbf{x}_0 = [x_0, y_0, z_0]^T$.

Con la estructura matemática de la transformación homogénea se desarrolla un conjunto de librerías para propósitos de simulación de cinemática directa de robots manipuladores y sistemas mecatrónicos.

3.2 Producto interno

Para modelar la orientación de la herramienta de trabajo del robot se encuentra el producto escalar o producto interno, también conocido como producto punto (dot product), el cual permite utilizar los conceptos de la geometría euclidiana tradicionales como longitudes, ángulos, proyecciones geométricas, ortogonalidad en dos y tres dimensiones de los sistemas de referencia asociados al robot y de la herramienta de trabajo.

El término espacio euclidiano se denomina así en honor del matemático y filósofo griego Euclides, y se utiliza para distinguirlo de otro tipo de espacios como los espacios curvos de la geometría no euclidiana y la teoría de la relatividad de Einstein. El espacio euclidiano es un tipo de espacio geométrico donde se satisfacen los axiomas de la geometría de Euclides. Como casos particulares del espacio euclidiano se encuentran la recta real, el plano y el espacio tridimensional, que corresponden a las dimensiones 1, 2 y 3, respectivamente.

El concepto abstracto de espacio euclidiano geométrico se generaliza para el caso general de espacio euclidiano n-dimensional, el cual también se considera como un espacio vectorial n-dimensional real.

Al producto interno $\mathbf{x} \cdot \mathbf{y}$ es una operación definida sobre dos vectores $\mathbf{x}, \mathbf{y} \in \mathrm{IR}^n$ de un espacio euclidiano cuyo resultado es un número o escalar (los espacios vectoriales que incluyen al producto interno reciben el nombre de espacios prehilbertianos).

Considere los siguientes vectores $\mathbf{x}, \mathbf{y} \in \mathrm{IR}^n$, el producto interno vectorial se define como:

$$\mathbf{x} \cdot \mathbf{y} = \mathbf{x}^T\mathbf{y} = \sum_{i=1}^{n} x_i y_i = x_1y_1 + x_2y_2 + \cdots + x_ny_n. \tag{3.1}$$

El producto punto se le llama producto escalar debido a que también se puede realizar como: $\mathbf{x} \cdot \mathbf{y} = \mathbf{x}^T \mathbf{y}$.

En un espacio euclidiano real, el producto interno (3.1) también acepta una definición geométrica dada por:

$$\mathbf{x}\cdot\mathbf{y} = \|\mathbf{x}\|\,\|\mathbf{y}\|\cos(\theta) \tag{3.2}$$

donde θ es el ángulo que forman los vectores $\mathbf{x}$ y $\mathbf{y}$ (ver figura 3.1), y las normas euclidianas se encuentran definidas por $\|\mathbf{x}\| = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$, $\|\mathbf{y}\| = \sqrt{y_1^2 + y_2^2 + \cdots + y_n^2}$, respectivamente.

Geométricamente $\|\mathbf{y}\|\cos(\theta)$ representa la proyección del vector $\mathbf{y}$ sobre la dirección del vector $\mathbf{x}$, como se presenta en la figura 3.1. El concepto de proyección geométrica se usa particularmente en las matrices rotacionales para modelar la orientación del sistema de referencia de la herramienta de trabajo del robot.

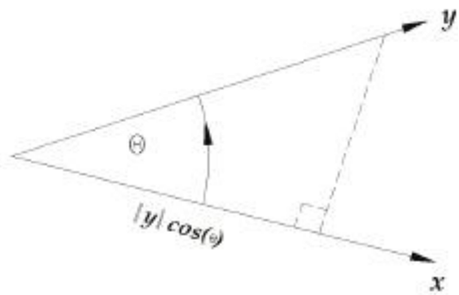


Figura 3.1 Producto interno vectorial $\mathbf{x}\cdot\mathbf{y}$.

Considere $\mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathbb{R}^n$, y $\alpha \in \mathbb{R}$; el producto interno tiene las siguientes propiedades:

- Conmutativa: $\mathbf{x}\cdot\mathbf{y} = \mathbf{y}\cdot\mathbf{x}$.
- Distributiva: $\mathbf{z}\cdot(\mathbf{x}+\mathbf{y}) = \mathbf{z}\cdot\mathbf{x} + \mathbf{z}\cdot\mathbf{y}$.
- Asociativa: $\alpha\mathbf{x}\cdot\mathbf{y} = \mathbf{x}\cdot\alpha\mathbf{y} = \mathbf{x}\cdot\mathbf{y}\alpha$.
- La expresión geométrica del producto escalar permite calcular el coseno del ángulo θ existente entre los vectores $\mathbf{x}$ y $\mathbf{y}$ de la siguiente manera:

$$\cos(\theta) = \frac{\mathbf{x}\cdot\mathbf{y}}{\|\mathbf{x}\|\,\|\mathbf{y}\|} = \frac{x_1y_1 + x_2y_2 + \cdots + x_ny_n}{\sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}\,\sqrt{y_1^2 + y_2^2 + \cdots + y_n^2}}.$$

- Vectores ortogonales: $\mathbf{x}\cdot\mathbf{y} = 0 \Rightarrow \mathbf{x}\perp\mathbf{y}$, $\theta = \frac{\pi}{2}$ rad (90 grados).

- Vectores paralelos o con la misma direcci´on si el ´angulo que forman es 0 rad (0 grados) o π rad (180 grados): $\mathbf{x}\cdot\mathbf{y} = \mathbf{x}\ \ \mathbf{y}$.
- $\mathbf{x}\cdot\mathbf{x} = 0 \Leftarrow\Rightarrow \mathbf{x} = \mathbf{0} \in \mathrm{IR}^n$.
- Si $\mathbf{x} = 0 \Rightarrow \mathbf{x}\cdot\mathbf{x} > 0$.
- $\sqrt{\mathbf{x}^T\mathbf{x}} = \sqrt{\sum_{i=1}^{n} x_i^2} = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$.

MATLAB contiene la funci´on dot(x,y) para realizar la operaci´on producto interno $\mathbf{x}\cdot\mathbf{y}$ de los vectores $\mathbf{x}, \mathbf{y} \in \mathrm{IR}^n$, con la siguiente estructura de sintaxis:

```
x · y  =  dot(x,y)
x · y  =  dot(x,y,n)
```

$\mathbf{x}\cdot\mathbf{y}$=dot(x,y,n) retorna el producto escalar de los vectores $\mathbf{x}, \mathbf{y}$ con dimensi´on n.

Nota: el producto punto $\mathbf{x}\cdot\mathbf{y}$ equivale a realizar la operaci´on en **MATLAB**: $\mathbf{x}\cdot\mathbf{y}$=x'*y.

La norma euclidiana $\mathbf{x}$ de un vector $\mathbf{x} \in \mathrm{IR}^n$, tambi´en conocida como norma 2, se puede calcular utilizando la funci´on norm con la siguiente sintaxis:

```
x  =norm(x,2)
```

♣ Ejemplo 3.1

Sean $\mathbf{x}, \mathbf{y} \in \mathrm{IR}^2$ con las siguientes componentes respectivamente:

$$\mathbf{x} = \begin{matrix} 2 \\ 4 \end{matrix} \qquad \mathbf{y} = \begin{matrix} 8 \\ -3 \end{matrix}$$

obtener el producto punto $\mathbf{x}\cdot\mathbf{y}$ y el ańgulo θ que forman entre los vectores $\mathbf{x}, \mathbf{y}$.

Solución

Puesto que los vectores $\mathbf{x}, \mathbf{y} \in \mathbb{R}^2$ están formados por $\mathbf{x} = [2 \quad 4]^T$ y $\mathbf{y} = [8 \quad -3]^T$, y tomando en cuenta la definición del producto punto se obtiene lo siguiente:

$$\mathbf{x} \cdot \mathbf{y} = x_1 y_1 + x_2 y_2 = 2\,(8) - 4\,(3) = 16 - 12 = 4.$$

El ángulo que forman los vectores $\mathbf{x}$ y $\mathbf{y}$ se obtiene como:

$$\cos(\theta) = \frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{x}\|\,\|\mathbf{y}\|} = \frac{x_1 y_1 + x_2 y_2}{\sqrt{x_1^2 + x_2^2}\sqrt{y_1^2 + y_2^2}}$$

$$= \frac{4}{\sqrt{2^2 + 4^2}\sqrt{8^2 + (-3)^2}} = \frac{4}{(4.4721)(8.544)} = 0.1046$$

lo que significa que $\theta = 1.465$ rad (83.99 grados).

Observe que evidentemente se cumple:

$$\mathbf{x} \cdot \mathbf{y} = \|\mathbf{x}\|\,\|\mathbf{y}\| \cos(\theta) = \sqrt{20}\sqrt{73}\cos(1.465) = (4.472135)(8.544003)(0.1046) = 4.$$

El cuadro 3.1 presenta el código fuente para realizar el desarrollo del ejemplo 3.1.

Código Fuente 3.1 Producto interno o escalar

%MATLAB Aplicado a Robótica y Mecatrónica.
%Editorial Alfaomega, Fernando Reyes Cortés.
%Capítulo 3 Cinemática %Programa cap3_prodint.m

Producto interno o escalar

```
clc; clear all; close all;
disp('Producto interno o escalar de vectores')
x = [2; 4 ]; %vector columna de dos renglones x ∈ IR^2
y = [8; -3]; %vector columna de dos renglones y ∈ IR^2
xdoty=dot(x,y) %producto interno x · y usando la función dot
xdoty_m=x'*y %forma matemática del producto interno x · y
theta=acos(x'*y/(norm(x,2)*norm(y,2))) %ángulo que forman los vectores
    x, y ∈ IR^2
%forma geométrica del producto interno: x · y = ‖x‖ ‖y‖ cos(θ)
xdoty_geometrica=norm(x,2)*norm(y,2)*cos(theta)
```

3.3 Matrices de rotación

La figura 3.2 muestra dos sistemas de referencia cartesianos, asociados a un cuerpo rígido, representados por $\Sigma_0(x_0, y_0, z_0)$ y $\Sigma_1(x_1, y_1, z_1)$, ambos sistemas comparten el mismo origen. El sistema de referencia $\Sigma_1(x_1, y_1, z_1)$ mantiene una orientación relativa al sistema de referencia fijo $\Sigma_0(x_0, y_0, z_0)$.

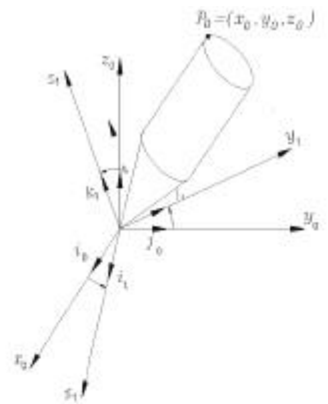


Figura 3.2 Sistemas de referencia fijo Σ_0 y rotado Σ_1.

Considérese un punto $\mathbf{p}$ sobre el cuerpo rígido, con respecto al sistema $\Sigma_0(x_0y_0z_0)$ tiene coordenadas $\mathbf{p}_0 = [x_0, y_0, z_0]^T$, el mismo punto $\mathbf{p}$ se representa como $\mathbf{p}_1 = [x_1, y_1, z_1]^T$ con respecto al sistema de referencia $\Sigma_1(x_1, y_1, z_1)$. El problema que se plantea es encontrar la relación que hay entre las coordenadas de un punto $\mathbf{p}_1$ en el sistema de referencia $\Sigma_1(x_1, y_1, z_1)$ con el vector $\mathbf{p}_0$ definido en el sistema de referencia $\Sigma_0(x_0, y_0, z_0)$. Considere vectores bases para cada sistema de referencia Σ_0 y Σ_1. Sean $\{\mathbf{i}_0, \mathbf{j}_0, \mathbf{k}_0\}$ vectores unitarios a lo largo de los ejes x_0, y_0, z_0, respectivamente. Es decir, $\mathbf{i}_0 = [1, 0, 0]^T$, $\mathbf{j}_0 = [0, 1, 0]^T$, $\mathbf{k}_0 = [0, 0, 1]^T$. Similarmente se definen los vectores unitarios $\{\mathbf{i}_1, \mathbf{j}_1, \mathbf{k}_1\}$ para el sistema $\Sigma_1(x_1, y_1, z_1)$.

Un vector que va desde el origen común para ambos sistemas hasta el punto $\mathbf{p}$, puede ser expresado en función de cualquiera de las dos bases de vectores unitarios de la siguiente forma:

$$\mathbf{p}_0 = p_{0x}\mathbf{i}_0 + p_{0y}\mathbf{j}_0 + p_{0z}\mathbf{k}_0 \quad \text{con respecto al sistema } \Sigma_0 \qquad (3.3)$$

$$\mathbf{p}_1 = p_{1x}\mathbf{i}_1 + p_{1y}\mathbf{j}_1 + p_{1z}\mathbf{k}_1 \quad \text{con respecto al sistema } \Sigma_1 \tag{3.4}$$

Los vectores $\mathbf{p}_0$, $\mathbf{p}_1$ representan al mismo punto $\mathbf{p}$. Tomando en cuenta las ecuaciones (3.3) y (3.4) la relación que hay entre sus componentes adquiere la siguiente forma:

$$\begin{aligned} p_{0x} &= \mathbf{p}_0 \cdot \mathbf{i}_0 = \mathbf{p}_1 \cdot \mathbf{i}_0 \\ &= p_{1x}\mathbf{i}_1 \cdot \mathbf{i}_0 + p_{1y}\mathbf{j}_1 \cdot \mathbf{i}_0 + p_{1z}\mathbf{k}_1 \cdot \mathbf{i}_0 \end{aligned} \tag{3.5}$$

$$\begin{aligned} p_{0y} &= \mathbf{p}_0 \cdot \mathbf{j}_0 = \mathbf{p}_1 \cdot \mathbf{j}_0 \\ &= p_{1x}\mathbf{i}_1 \cdot \mathbf{j}_0 + p_{1y}\mathbf{j}_1 \cdot \mathbf{j}_0 + p_{1z}\mathbf{k}_1 \cdot \mathbf{j}_0 \end{aligned} \tag{3.6}$$

$$\begin{aligned} p_{0z} &= \mathbf{p}_0 \cdot \mathbf{k}_0 = \mathbf{p}_1 \cdot \mathbf{k}_0 \\ &= p_{1x}\mathbf{i}_1 \cdot \mathbf{k}_0 + p_{1y}\mathbf{j}_1 \cdot \mathbf{k}_0 + p_{1z}\mathbf{k}_1 \cdot \mathbf{k}_0. \end{aligned} \tag{3.7}$$

Estas ecuaciones pueden ser escritas de manera compacta como:

$$\mathbf{p}_0 = R_0^1\mathbf{p}_1 \tag{3.8}$$

donde R_0^1 representa la siguiente matriz

$$R_0^1 = \begin{bmatrix} \mathbf{i}_1 \cdot \mathbf{i}_0 & \mathbf{j}_1 \cdot \mathbf{i}_0 & \mathbf{k}_1 \cdot \mathbf{i}_0 \\ \mathbf{i}_1 \cdot \mathbf{j}_0 & \mathbf{j}_1 \cdot \mathbf{j}_0 & \mathbf{k}_1 \cdot \mathbf{j}_0 \\ \mathbf{i}_1 \cdot \mathbf{k}_0 & \mathbf{j}_1 \cdot \mathbf{k}_0 & \mathbf{k}_1 \cdot \mathbf{k}_0 \end{bmatrix}. \tag{3.9}$$

La matriz $R_0^1 \in IR^{3\times 3}$ es la matriz de transformación de las coordenadas del punto $\mathbf{p}$ del sistema de referencia $\Sigma_1(x_1, y_1, z_1)$ hacia las coordenadas del sistema $\Sigma_0(x_0, y_0, z_0)$. En otras palabras, dado un punto $\mathbf{p}_1$ en el sistema $\Sigma_1(x_1, y_1, z_1)$, entonces $R_0^1\mathbf{p}_1$ representa el mismo vector expresado con respecto al sistema de referencia $\Sigma_0(x_0, y_0, z_0)$.

Obsérvese que las columnas de R_0^1 son los cosenos directores de los ejes coordenados x_1, y_1, z_1 respecto de los ejes coordenados x_0, y_0, z_0. Por ejemplo, la primera columna $[\mathbf{i}_1 \cdot \mathbf{i}_0, \mathbf{i}_1 \cdot \mathbf{j}_0, \mathbf{i}_1 \cdot \mathbf{k}_0]^T$ especifica la dirección del eje x_1 relativa al sistema de referencia Σ_0.

Similarmente se puede obtener el punto $\mathbf{p}_1$ en función del punto $\mathbf{p}_0$:

$$\begin{aligned} p_{1x} &= \mathbf{p}_1 \cdot \mathbf{i}_1 = \mathbf{p}_0 \cdot \mathbf{i}_1 \\ &= p_{0x}\mathbf{i}_0 \cdot \mathbf{i}_1 + p_{0y}\mathbf{j}_0 \cdot \mathbf{i}_1 + p_{0z}\mathbf{k}_0 \cdot \mathbf{i}_1 \end{aligned}$$

$$
\begin{aligned}
p_{1y} &= \mathbf{p}_1 \cdot \mathbf{j}_1 = \mathbf{p}_0 \cdot \mathbf{j}_1 \\
&= p_{0x}\mathbf{i}_1 \cdot \mathbf{j}_1 + p_{0y}\mathbf{j}_1 \cdot \mathbf{j}_1 + p_{0z}\mathbf{k}_1 \cdot \mathbf{j}_1 \\
p_{1z} &= \mathbf{p}_1 \cdot \mathbf{k}_1 = \mathbf{p}_0 \cdot \mathbf{k}_1 \\
&= p_{0x}\mathbf{i}_1 \cdot \mathbf{k}_1 + p_{0y}\mathbf{j}_1 \cdot \mathbf{k}_1 + p_{0z}\mathbf{k}_1 \cdot \mathbf{k}_1
\end{aligned}
$$

$$
\mathbf{p}_1 = R_0^1 \mathbf{p}_0 \tag{3.10}
$$

$$
R_1^0 = \begin{bmatrix} \mathbf{i}_0 \cdot \mathbf{i}_1 & \mathbf{j}_0 \cdot \mathbf{i}_1 & \mathbf{k}_0 \cdot \mathbf{i}_1 \\ \mathbf{i}_0 \cdot \mathbf{j}_1 & \mathbf{j}_0 \cdot \mathbf{j}_1 & \mathbf{k}_0 \cdot \mathbf{j}_1 \\ \mathbf{i}_0 \cdot \mathbf{k}_1 & \mathbf{j}_0 \cdot \mathbf{k}_1 & \mathbf{k}_0 \cdot \mathbf{k}_1 \end{bmatrix}. \tag{3.11}
$$

La matriz $R_0^1 \in IR^{3\times 3}$ (3.11) representa la matriz inversa de la transformación R_1^0 (3.9). Debido a que el producto interno de vectores unitarios cumple la propiedad conmutativa, entonces $\mathbf{i}_1 \cdot \mathbf{j}_1 = \mathbf{j}_1 \cdot \mathbf{i}_1$, $\mathbf{i}_0 \cdot \mathbf{j}_0 = \mathbf{j}_0 \cdot \mathbf{i}_0$, etc., por lo que resulta:

$$
R_0^1 = \left[R_1^0\right]^{-1} = \left[R_1^0\right]^T \tag{3.12}
$$

La matriz R_1^0 cuya inversa es su transpuesta se denomina **matriz ortogonal**. La norma de los vectores columna de R_1^0 son de magnitud unitaria y mutuamente ortogonales, el determinante de R_1^0 es ± 1. Si el sistema de referencia se selecciona de acuerdo con la regla de la mano derecha, entonces el determinante de R_1^0 es 1. La matriz R_1^0 se denomina matriz de rotación y pertenece a la clase de matrices ortogonales que se denotan como $SO(3)$.

Notación

La matriz R_1^0 representa la orientación del sistema $\Sigma_1(x_1, y_1, z_1)$ respecto al sistema $\Sigma_0(x_0, y_0, z_0)$. R_2^1 representa la orientación del sistema $\Sigma_2(x_2, y_2, z_2)$ respecto al sistema $\Sigma_1(x_1, y_1, z_1)$, y así sucesivamente. Mientras que la transformación inversa R_0^1 significa la orientación del sistema de referencia $\Sigma_0(x_0, y_0, z_0)$ relativa al sistema de referencia $\Sigma_1(x_1, y_1, z_1)$. De manera análoga, R_1^2 es la transformación inversa de coordenadas del sistema de referencia $\Sigma_1(x_1, y_1, z_1)$ hacia el sistema $\Sigma_2(x_2, y_2, z_2)$.

Hay varios métodos para expresar la orientación del sistema de referencia $\Sigma_1(x_1, y_1, z_1)$ con respecto al sistema de referencia fijo $\Sigma_0(x_0, y_0, z_0)$, entre los más usuales se encuentra la rotación con respecto a uno de los ejes principales x_0, y_0, y_0.

A continuación se tratan los casos de rotación alrededor de un eje principal.

Matriz de rotación alrededor del eje z_0

Considere que el sistema de referencia $\Sigma_1(z_1, y_1, z_1)$ el cual se encuentra rotado un ángulo θ alrededor del eje z_0 del sistema $\Sigma_0(z_0, y_0, z_0)$. Obtener la matriz resultante de transformación R_{10}.

Como se muestra en la figura 3.3, los ejes z_0 y z_1 son paralelos. El signo del ángulo θ está dado por la regla de la mano derecha. Por convención, un ángulo positivo es aquel cuyo sentido de rotación es contrario al movimiento de las manecillas del reloj.

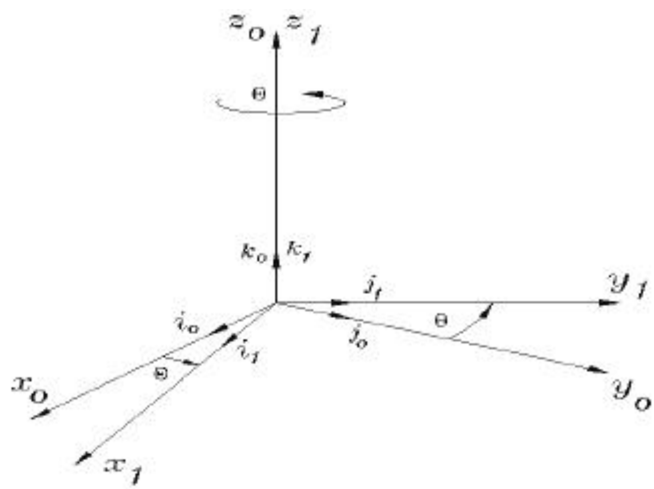


Figura 3.3 Rotación de un ángulo θ alrededor del eje z.

De la figura 3.3 se obtienen las siguientes ecuaciones:

$$\mathbf{i}_1 \cdot \mathbf{i}_0 = \cos(\theta) \qquad \mathbf{j}_1 \cdot \mathbf{i}_0 = -\operatorname{sen}(\theta)$$
$$\mathbf{j}_1 \cdot \mathbf{j}_0 = \cos(\theta) \qquad \mathbf{i}_1 \cdot \mathbf{j}_0 = \operatorname{sen}(\theta)$$
$$\mathbf{k}_0 \cdot \mathbf{k}_1 = 1$$

todos los demás productos punto son cero, puesto que el ángulo que existe entre los vectores unitarios $\mathbf{i}_0, \mathbf{i}_1, \mathbf{j}_0,$ y $\mathbf{j}_1$ con $\mathbf{k}_0$ y $\mathbf{k}_1$ es de 90 grados, excepto entre ellos mismos, puesto que $\mathbf{k}_0$ y $\mathbf{k}_1$ forman un angulo de 0 grados: $\mathbf{k}_1 \cdot \mathbf{i}_0 = \cos(\frac{\pi}{2}) = 0$, $\mathbf{k}_1 \cdot \mathbf{j}_0 = \cos(\frac{\pi}{2})$, $\mathbf{i}_1 \cdot \mathbf{k}_0 = \cos(\frac{\pi}{2}) = 0$, y $\mathbf{j}_1 \cdot \mathbf{k}_0 = \cos(\frac{\pi}{2}) = 0$.

Notación

La matriz (3.13) se conoce como matriz de rotación alrededor del eje z y es representada por $R_z(\theta)$.

$$R_z(\theta) = \begin{bmatrix} \cos(\theta) & -\operatorname{sen}(\theta) & 0 \\ \operatorname{sen}(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.13}$$

La matriz de rotación $R_z(\theta)$ se interpreta como una matriz que especifica la orientación del sistema de referencia $\Sigma_1(x_1, y_1, z_1)$ relativo al sistema de referencia $\Sigma_0(x_0, y_0, z_0)$.

$$\mathbf{p}_0 = R_z(\theta)\mathbf{p}_1 = \begin{bmatrix} \cos(\theta) & -\operatorname{sen}(\theta) & 0 \\ \operatorname{sen}(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{p}_1. \tag{3.14}$$

Por convención considérese que el ángulo θ es positivo en el sentido contrario al movimiento de las manecillas del reloj.

La relación inversa que determina la orientación del sistema $\Sigma_0(x_0, y_0, z_0)$ con respecto al sistema $\Sigma_1(x_1, y_1, z_1)$ está dada por:

$$\mathbf{p}_1 = R_z^T(\theta)\mathbf{p}_0 = \begin{bmatrix} \cos(\theta) & \operatorname{sen}(\theta) & 0 \\ -\operatorname{sen}(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{p}_0. \tag{3.15}$$

Por lo tanto, un punto $\mathbf{p}_0$ en el sistema $\Sigma_0(x_0, y_0, z_0)$ es transformado hacia un punto $\mathbf{p}_1$ en el sistema $\Sigma_1(x_1, y_1, z_1)$ incluyendo su orientación relativa.

Otra forma de obtener la matriz $R_z(\theta)$ que relaciona la orientación θ del sistema de referencia $\Sigma_1(z_1, y_1, z_1)$ con respecto al eje z_0 del sistema de referencia $\Sigma_0(z_0, y_0, z_0)$ es por medio de una proyección geométrica, es decir, analizando la proyección de los ejes x_1, y_1 sobre los ejes x_0, y_0 como se ve en la figura 3.4.

Los ejes z_1 y z_0 son paralelos entre sí, el plano $x_1 - y_1$ se encuentra rotado un ángulo θ con respecto al plano $x_0 - y_0$, entonces la proyección del punto $\mathbf{p}_1 = [p_{1x}, p_{1y}, p_{1z}]^T$

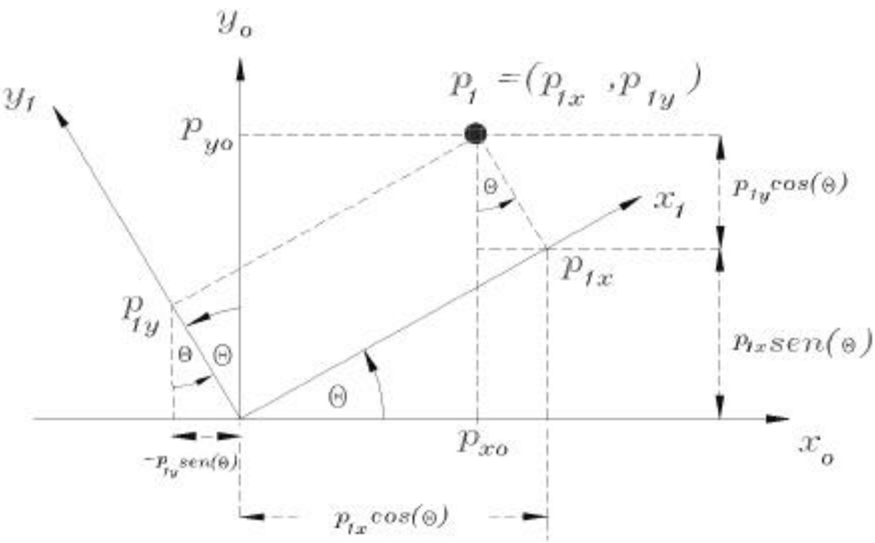


Figura 3.4 Rotaci´on θ grados del plano $x_1 - y_1$ con respecto al plano $x_0 - y_0$.

sobre los ejes del sistema Σ_0 son:

$$\begin{aligned} p_{0x} &= p_{1x}\cos(\theta) - p_{1y}\,\mathrm{sen}(\theta) \\ p_{0y} &= p_{1x}\,\mathrm{sen}(\theta) + p_{1y}\cos(\theta) \\ p_{0z} &= p_{1z} \\ \mathbf{p}_0 &= R_z(\theta)\mathbf{p}_1 = \begin{bmatrix} \cos(\theta) & -\mathrm{sen}(\theta) & 0 \\ \mathrm{sen}(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}\mathbf{p}_1. \end{aligned}$$

Propiedades de la matriz de rotaci´on $R_z(\theta)$

La matriz de rotaci´on $R_z(\theta)$ tiene varias propiedades importantes que a continuaci´on se presentan:

- $R_z(0) = \mathbf{I}$, $\mathbf{I} \in IR^{3\times 3}$ es la matriz identidad.
- $R_z(\theta)R_z(\beta) = R_z(\beta)R_z(\theta) = R_z(\theta+\beta) = R_z(\beta+\theta)$.
- $R_z(\theta)^{-1} = R_z(-\theta)$.

- $R_z(\theta)^T = R_z(\theta)^{-1}$.
- $R_z(\theta)R_z(\theta)^T = R_z(\theta)^T R_z(\theta) = I$.
- $\det[R_z(\theta)] = 1$ si el sistema de referencia cartesiano $\Sigma(z, y, z)$ es seleccionado por la regla de la mano derecha, en otro caso $\det[R_z(\theta)] = -1$.

Función matriz de rotación $R_z(\theta)$

La función matriz de rotación $R_z(\theta)$

$$R_z(\theta) = \begin{bmatrix} \cos(\theta) & -\mathrm{sen}(\theta) & 0 \\ \mathrm{sen}(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

tiene la siguiente sintaxis:

```
R=Rz(θ)
```

donde $\theta \in$ IR es el ángulo de rotación alrededor del eje z, y representa el argumento de entrada. Retorna la matriz de rotación R.

El cuadro 3.2 contiene el programa en código fuente para evaluar en forma simbólica las propiedades matemáticas de la matriz de rotación $R_z(\theta)$. Para simplificar el álgebra simbólica es recomendable utilizar la función `simplify`.

> Para expresiones matemáticas con variables simbólicas, la función `simplify` resulta importante para obtener un resultado matemático compacto, es decir factorizado y reducido.

> Cuando un programa se encuentra combinando cálculos numéricos con variables simbólicas, se recomienda usar la función vpa con una precisión de tres dígitos a través de `digits(3)`.

Código Fuente 3.2 Función $R_z(\theta)$

%MATLAB Aplicado a Robótica y Mecatrónica.
%Editorial Alfaomega, Fernando Reyes Cortés.
%Capítulo 3 Cinemática %Archivo Rz.m

Función $R_z(\theta)$

```
function R=Rz(theta)
    dato=whos('theta');
    if strcmp(dato.class, 'sym') %para variables simbólicas
        R=simplify([cos(theta), -sin(theta), 0;
                    sin(theta), cos(theta), 0;
                    0, 0, 1]);
        else %cálculos numéricos
        digits(3);
        R=simplify([ double(cos(theta)), double(-sin(theta)), 0;
                     double(sin(theta)), double(cos(theta)), 0;
                     0, 0, 1]);
    end
end
```

♣ ♣ Ejemplo 3.2

Escribir un programa en **MATLAB** para comprobar las propiedades de la matriz de rotación $R_z(\theta)$

Solución

En el cuadro 3.5 se presenta el programa en código fuente para **MATLAB** de las propiedades de la matriz de rotación $R_z(\theta)$.

Para que el resultado simbólico sea compacto o simplificado se emplea la función simplify.

Para visualizar los resultados de las propiedades matemáticas de la matriz $R_z(\theta)$ es necesario no insertar en cada línea el operador **;**).

Cuando se trabaja únicamente con variables simbólicas es importante utilizar la función `simplify` para obtener resultados algebraicos compactos.

Cuando hay una combinación entre variables simbólicas y cálculos numéricos es recomendable usar la función double como en el cuadro del código 3.2 de la matriz $R_z(\theta)$. Otras opciones que se pueden usar son la función vpa y `digits(3)` (ver el código 3.8).

Código Fuente 3.5 Propiedades de la matriz de rotación $R_z(\theta)$

%MATLAB Aplicado a Robótica y Mecatrónica.
%Editorial Alfaomega, Fernando Reyes Cortés.
%Capítulo 3 Cinemática %Archivo cap3_propiedadesRz.m

Propiedades de la matriz de rotación $R_z(\theta)$

```
clc;
clear all;
close all;
syms a b real
%resultados simbólicos
simplify(Rz(a)*Rz(b))   %Rz(a)Rz(b)
simplify(Rz(b)*Rz(a))   %Rz(b)Rz(a)
simplify(inv(Rz(a)))   %Rz(a)^-1
simplify(Rz(-a))   %Rz(a)^-1 = Rz(-a)
simplify(Rz(a)')   %Rz(a)^T = Rz(a)^-1
simplify(Rz(a)'*Rz(a))   %Rz(a)^T Rz(a) = I
simplify(Rz(a)*Rz(a)')   %Rz(a)Rz(a)^T = I
simplify(det(Rz(a)))   %det[Rz(a)]=1
%cálculo numérico
theta=90*pi/180; θ = π/2
Rz(theta) %Rz(θ)
```

♣ Ejemplo 3.3

Escribir un algoritmo en lenguaje **MATLAB** que realice la rotaci´on de 90 grados alrededor del eje z_0 de una imagen de prueba (flecha).

Soluci´on

El programa 3.4 contiene el c´odigo para realizar la rotaci´on de la imagen de prueba. El vector $\mathbf{p}_0 = [p_{x0}, p_{y0}, p_{z0}]^T$ tiene las coordenadas de la imagen de prueba definidas en el sistema de referencia $\Sigma_0(x_0, y_0, z_0)$. El angulo de rotaci´on θ es de 90 grados con respecto al eje z_0. El sistema $\Sigma_1(x_1, y_1, z_1)$ tiene la imagen rotada con respecto al sistema $\Sigma_0(x_0, y_0, z_0)$. La matriz de rotaci´on $R_{z_0}(\frac{\pi}{2})$ relaciona la orientaci´on que tiene la imagen rotada definida en $\Sigma_1(x_1, y_1, z_1)$ en relaci´on al sistema fijo $\Sigma_0(x_0, y_0, z_0)$.

En la figura 3.5 se muestra el resultado del programa en ambos sistemas de referencia $\Sigma_0(x_0, y_0, z_0)$ y el sistema de referencia rotado $\Sigma_1(x_1, y_1, z_1)$.

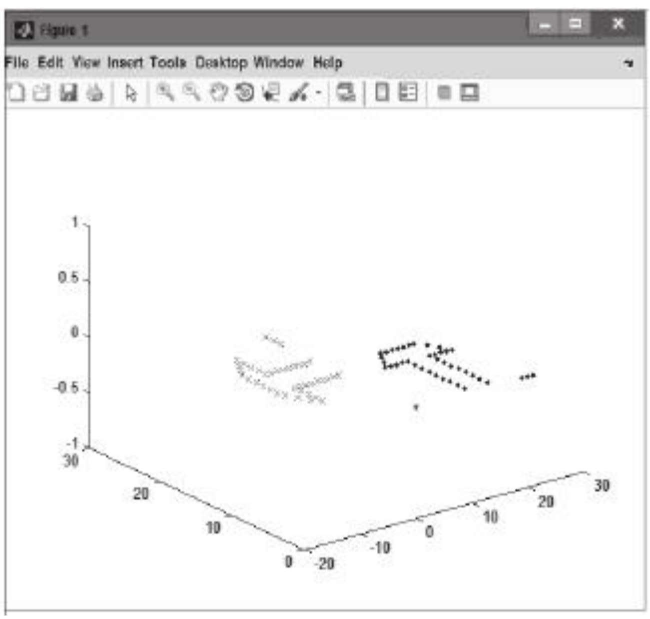


Figura 3.5 Rotaci´on de 90 grados alrededor del eje z_0.

Código Fuente 3.4 Rotación alrededor del eje z_0

%MATLAB Aplicado a Robótica y Mecatrónica.
%Editorial Alfaomega, Fernando Reyes Cortés.
%Capítulo 3 Cinemática %Archivo cap3_rotacionRz.m

Rotación alrededor del eje z_0

```
clc;
clear all;
close all;
%coordenadas del vector p0 = [px0; py0; px0]
pxo=[0,0,0,0,0,6,7,7, 8, 8, 9,9, 10,10, 10, 10, 10, 10, 10, 10, 10, 10, 11, 12, 13, 14,14,
14, 14, 14, 14, 14, 14,14, 14, 15, 15, 16,16, 17,17,18,19,20,21];
pyo=[0,0,0,0,0,9,9,10,9, 11, 9,12, 1, 2, 3, 4, 5, 6,7, 8, 9, 12, 12, 12, 12,1, 2, 3, 4, 5,
6, 7, 8, 9, 12, 9, 12, 9, 11, 9, 10, 9,0,0,0 ];
pzo=[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0];
theta=90*3.1416/180.0;
%matriz de rotación Rz(θ)
R_ztheta=[ cos(theta), -sin(theta), 0;
           sin(theta), cos(theta), 0;
           0, 0,1];
%conversión de coordenadas del sistema Σ0(x0, y0, z0) al sistema Σ1(x1, y1, z1)
% p1 = Rz(θ) p0
Sigma1=R_ztheta*[pxo; pyo; pzo];
% p1 = [px1; py1; pz1]
px1=Sigma1(1,:);
py1=Sigma1(2,:);
pz1=Sigma1(3,:);
%puntos graficados en los sistemas p0 ∈ Σ0(x0, y0, z0) y p1 ∈ Σ1(x1, y1, z1)
plot3(pxo,pyo,pzo,'.',px1,py1,pz1,'x')
```

♣ Ejemplo 3.4

Considere un punto $\mathbf{p}_1 = [p_{1x}, p_{1y}, p_{1z}]^T = [0.8, 0.5, 1]^T$ en el sistema de referencia $\Sigma_1(z_1, y_1, z_1)$, el cual mantiene una orientación relativa de 90 grados alrededor del eje z_0 del sistema fijo $\Sigma_0(z_0, y_0, z_0)$. Obtener la proyección del punto $\mathbf{p}_1$ en el sistema fijo Σ_0.

Solución

El punto $\mathbf{p}_1 = [p_{1x}, p_{1y}, p_{1z}]^T = [0.8, 0.5, 1]^T$ se encuentra en el sistema de referencia $\Sigma_1(z_1, y_1, z_1)$, la conversión de coordenadas al sistema de referencia fijo se realiza a través de: $\mathbf{p}_0 = R(z, \theta)\mathbf{p}_1$, obteniendo lo siguiente:

$$
\begin{aligned}
\mathbf{p}_0 &= \begin{bmatrix} \cos(\frac{\pi}{2}) & -\,\mathrm{sen}(\frac{\pi}{2}) & 0 \\ \mathrm{sen}(\frac{\pi}{2}) & \cos(\frac{\pi}{2}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.8 \\ 0.5 \\ 1 \end{bmatrix} \\
&= \begin{bmatrix} -0.5 \\ 0.8 \\ 1 \end{bmatrix}.
\end{aligned}
$$

Nótese que el punto $\mathbf{p}_0$ en el sistema de referencia fijo $\Sigma_0(z_0, y_0, z_0)$ se puede convertir al punto $\mathbf{p}_1$ en el sistema de referencia rotado $\Sigma_1(z_1, y_1, z_1)$, a través de la siguiente relación:

$$
\begin{aligned}
\mathbf{p}_1 &= \begin{bmatrix} \cos(\frac{\pi}{2}) & \mathrm{sen}(\frac{\pi}{2}) & 0 \\ -\,\mathrm{sen}(\frac{\pi}{2}) & \cos(\frac{\pi}{2}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -0.5 \\ 0.8 \\ 1 \end{bmatrix} \\
&= \begin{bmatrix} 0.8 \\ 0.5 \\ 1 \end{bmatrix}.
\end{aligned}
$$

La figura 3.6 muestra la descripción geométrica del punto $\mathbf{p}_1$ y su proyección sobre el sistema de referencia $\Sigma_0(z_0, y_0, z_0)$. Observe que los ejes z_0 y z_1 se mantienen paralelos entre sí.

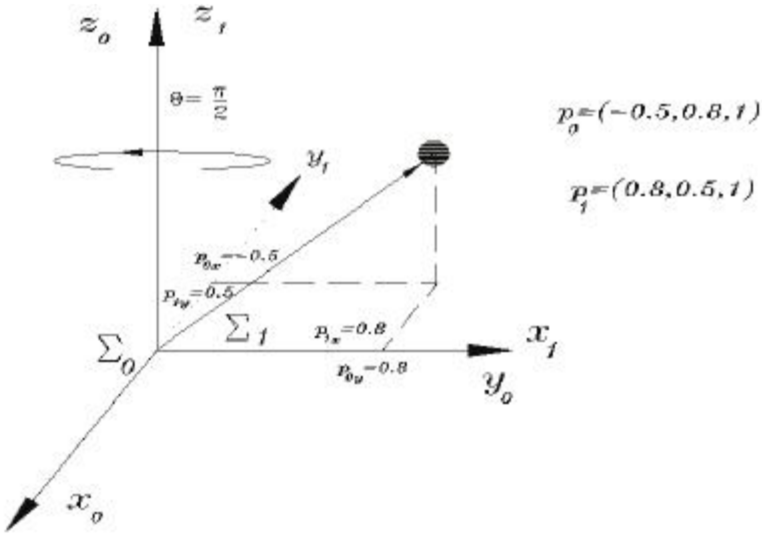


Figura 3.6 Rotaci´on relativa de 90 grados del sistema Σ_1 alrededor del eje z_0.

♣ Ejemplo 3.5

Considere un paralelep´ıpedo rectangular definido en el sistema de referencia $\Sigma_0(z_0, y_0, z_0)$; rotar el poliedro 180 grados alrededor del eje z_0. Describir la proyecci´on del sistema de referencia rotado $\Sigma_1(z_1, y_1, z_1)$ en el sistema de referencia fijo.

Soluci´on

Sea $\mathbf{p}_0 = [p_{0x}, p_{0y}, p_{0z}]^T$ un punto sobre el paralelep´ıpedo rectangular, al rotar este punto $\mathbf{p}_0$ 180 grados con respecto al eje z_0 se mueve r´ıgidamente junto con todo el s´olido quedando en el punto $\mathbf{p}_1 = [p_{1x}, p_{1y}, p_{1z}]^T$.

Todos los puntos que pertenecen al poliedro est´an sujetos a la siguiente transformaci´on: $\mathbf{p}_1 = R_z(\pi)\mathbf{p}_0$, es decir los puntos $\mathbf{p}_0$ definidos en el sistema $\Sigma_0(x_0, y_0, z_0)$ son transformados a coordenadas $\mathbf{p}_1$ en el sistema $\Sigma_1(x_1, y_1, z_1)$ por medio de la matriz de rotaci´on $R_z(\pi)$. La interpretaci´on geom´etrica de la rotaci´on del poliedro 90 grados alrededor del eje z_0 se presenta en la figura 3.7.

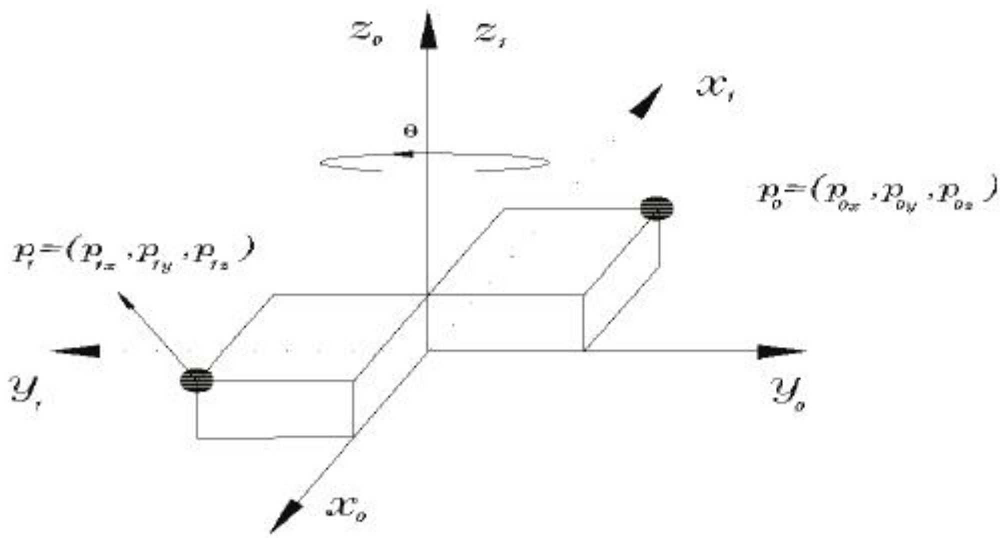


Figura 3.7 Rotación de 180 grados alrededor del eje z_0 de un paralelepípedo rectangular.

Matriz de rotación alrededor del eje x_0

Considere el sistema de referencia $\Sigma_1(z_1, y_1, z_1)$ el cual está rotado un ángulo θ alrededor del eje x_0 del sistema $\Sigma_0(z_0, y_0, z_0)$; obtener la matriz resultante de rotación.

En la figura 3.8 se muestra la rotación del sistema $\Sigma_1(z_1, y_1, z_1)$ con respecto al eje x_0 del sistema fijo $\Sigma_0(z_0, y_0, z_0)$; el ángulo de rotación θ gira alrededor del eje x_0 en sentido positivo (contrario a las manecillas del reloj).

En este caso el ángulo de rotación que existe entre los ejes x_0 y x_1 es cero, puesto que son ejes paralelos. Para la primera columna $[\mathbf{i}_1 \cdot \mathbf{i}_0 \quad \mathbf{i}_1 \cdot \mathbf{j}_0 \quad \mathbf{i}_1 \cdot \mathbf{k}_0]^T$ de la matriz de rotación $R_{10}(\theta)$ ecuación (3.9) tiene las siguientes componentes: $\mathbf{i}_1 \cdot \mathbf{i}_0 = \cos(0) = 1$, $\mathbf{i}_1 \cdot \mathbf{j}_0 = \cos(\frac{\pi}{2}) = 0$; $\mathbf{i}_1 \cdot \mathbf{k}_0 = \cos(\frac{\pi}{2}) = 0$. La segunda columna $[\mathbf{j}_1 \cdot \mathbf{i}_0 \quad \mathbf{j}_1 \cdot \mathbf{j}_0 \quad \mathbf{j}_1 \cdot \mathbf{k}_0]^T$ adquiere la siguiente forma: $\mathbf{j}_1 \cdot \mathbf{i}_0 = \cos(\frac{\pi}{2}) = 0$, $\mathbf{j}_1 \cdot \mathbf{j}_0 = \cos(\theta)$ y $\mathbf{j}_1 \cdot \mathbf{k}_0 = \cos(\theta - \frac{\pi}{2}) = \operatorname{sen}(\theta)$.

Finalmente, la tercera columna $[\mathbf{k}_1 \cdot \mathbf{i}_0 \quad \mathbf{k}_1 \cdot \mathbf{j}_0 \quad \mathbf{k}_1 \cdot \mathbf{k}_0]^T$ se encuentra definida

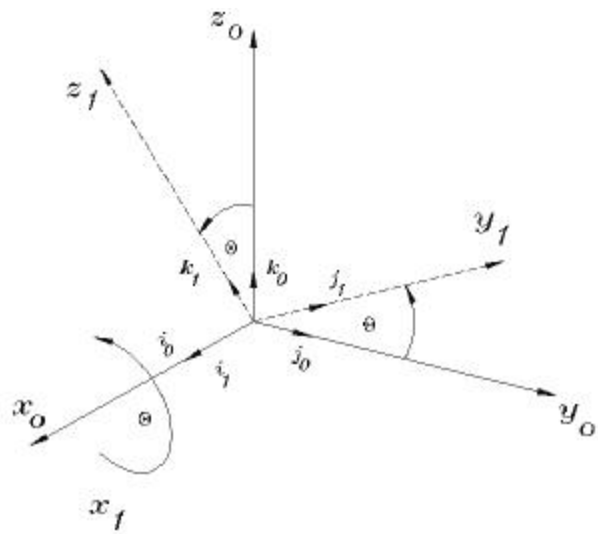


Figura 3.8 Rotación de un ángulo θ alrededor del eje x_0.

como: $\mathbf{k}_1 \cdot \mathbf{i}_0 = \cos(\frac{\pi}{2}) = 0$, $\mathbf{k}_1 \cdot \mathbf{j}_0 = \cos(\theta + \frac{\pi}{2}) = -\operatorname{sen}(\theta)$ y $\mathbf{k}_1 \cdot \mathbf{k}_0 = \cos(\theta)$.

Por lo tanto, la matriz correspondiente de rotación está dada por:

$$R_{10}(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\operatorname{sen}(\theta) \\ 0 & \operatorname{sen}(\theta) & \cos(\theta) \end{bmatrix}. \tag{3.16}$$

Observe que los ejes x_0 y x_1 coinciden entre sí, y el plano $z_1 - y_1$ se desplaza un ángulo θ de derecha a izquierda con respecto al plano $x_0 - y_0$.

Notación

La matriz (3.16) cuyo ángulo de rotación θ se realiza alrededor del eje x se denota por $R_x(\theta)$ y su estructura matemática es:

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\operatorname{sen}(\theta) \\ 0 & \operatorname{sen}(\theta) & \cos(\theta) \end{bmatrix}. \tag{3.17}$$

Las propiedades de la matriz de rotación $R_x(\theta)$ son las mismas para $R_z(\theta)$.

Matriz de rotaci´on alrededor del eje y_0

Considere los sistemas de referencia $\Sigma_1(z_1, y_1, z_1)$ y $\Sigma_0(z_0, y_0, z_0)$, el sistema $\Sigma_1(z_1, y_1, z_1)$ se encuentra rotado un ańgulo θ alrededor del eje y_0 del sistema fijo $\Sigma_0(z_0, y_0, z_0)$. Obtener la matriz resultante de rotaci´on.

De la figura 3.9 se puede obtener la matriz que relaciona la orientaci´on relativa del sistema de referencia $\Sigma_1(z_1, y_1, z_1)$ con respecto al sistema de referencia fijo $\Sigma_0(z_0, y_0, z_0)$. El ańgulo de rotaci´on θ es alrededor del eje y_0.

La matriz $R_{10}(\theta)$ adquiere la siguiente expresi´on:

$$R_{10}(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \mathrm{sen}(\theta) \\ 0 & 1 & 0 \\ -\mathrm{sen}(\theta) & 0 & \cos(\theta) \end{bmatrix} \tag{3.18}$$

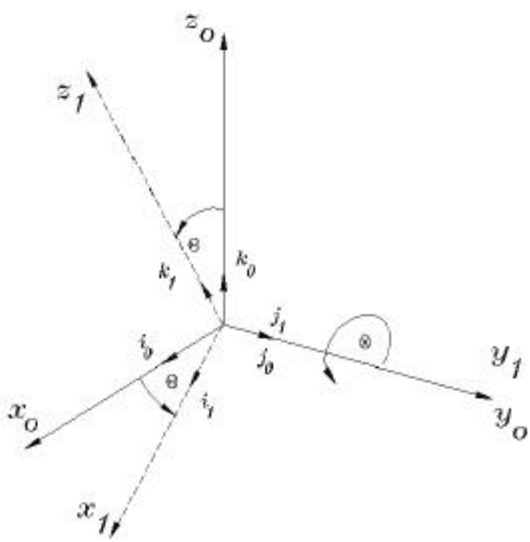


Figura 3.9 Rotaci´on de un ańgulo θ del sistema Σ_1 alrededor del eje y_0 del sistema Σ_0.

Observe que la rotaci´on del ´angulo θ alrededor del eje y_0 mueve el plano $z_1 - x_1$.

Notación

La matriz de rotación (3.18) describe la orientación relativa del sistema $\Sigma_1(z_1, y_1, z_1)$ con respecto al sistema de referencia fijo $\Sigma_0(z_0, y_0, z_0)$ usando una rotación alrededor del eje y_0, la cual se denota por:

$$R_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \operatorname{sen}(\theta) \\ 0 & 1 & 0 \\ -\operatorname{sen}(\theta) & 0 & \cos(\theta) \end{bmatrix} \tag{3.19}$$

Las propiedades de $R_y(\theta)$ son las mismas que $R_z(\theta)$.

3.4 Reglas de rotación

Generalmente la descripción de la orientación del sistema de referencia de la herramienta de trabajo del robot colocada en el extremo final del robot con respecto al sistema de referencia fijo en la base del robot involucra varias rotaciones sucesivas (composición de rotaciones). Existen varios métodos que permiten definir el orden o reglas de las rotaciones consecutivas.

Composición de rotaciones para sistemas de referencia sucesivos

Considere tres sistemas de referencia cartesianos definidos por $\Sigma_0(x_0, y_0, z_0)$ (sistema fijo), $\Sigma_1(x_1, y_1, z_1)$ tiene una rotación relativa al sistema fijo y sea el sistema de referencia $\Sigma_2(x_2, y_2, z_2)$ cuya orientación relativa es con respecto al sistema de referencia $\Sigma_1(x_1, y_1, z_1)$. Todos los sistemas de referencia cartesianos comparten el mismo origen como se presenta en la figura 3.10.

El problema que se plantea es encontrar la expresión matemática que relaciona la orientación del sistema de referencia $\Sigma_2(x_2, y_2, z_2)$ relativo al sistema de referencia fijo $\Sigma_0(x_0, y_0, z_0)$.

Puesto que los tres sistemas de referencia tienen el mismo origen, entonces un punto $\mathbf{p}$ puede ser representado en función de las coordenadas $\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2$ de la siguiente

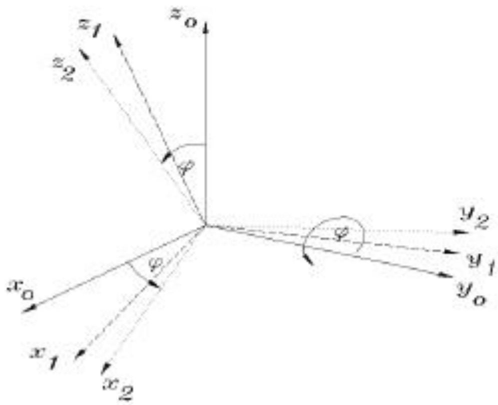


Figura 3.10 Composici´on de rotaciones.

forma:

$$\mathbf{p}_0 = R_{10}\mathbf{p}_1 \tag{3.20}$$
$$\mathbf{p}_1 = R_{21}\mathbf{p}_2 \tag{3.21}$$

donde la matriz R_{10} representa la orientaci´on del sistema de referencia $\Sigma_1(x_1, y_1, z_1)$ con respecto al sistema de referencia $\Sigma_0(x_0, y_0, z_0)$, y la matriz R_{21} describe la orientaci´on del sistema de referencia $\Sigma_2(x_2, y_2, z_2)$ respecto al sistema $\Sigma_1(x_1, y_1, z_1)$.

Por lo tanto, la relaci´on que hay entre un punto $\mathbf{p}_0 \in \Sigma_0(x_0, y_0, z_0)$ con las coordenadas $\mathbf{p}_2 \in \Sigma_2(x_2, y_2, z_2)$ est´a determinada por la siguiente expresi´on:

$$\mathbf{p}_0 = R_{10}R_{21}\mathbf{p}_2 \tag{3.22}$$

donde $R_{10}R_{21}$ representa la orientaci´on del sistema de referencia $\Sigma_2(x_2, y_2, z_2)$ respecto al sistema fijo $\Sigma_0(x_0, y_0, z_0)$, y significa una regla de composici´on para establecer el orden para transformar un punto $\mathbf{p}$ desde su representaci´on en el sistema $\Sigma_2(x_2, y_2, z_2)$ a su correspondiente representaci´on en el sistema $\Sigma_0(x_0, y_0, z_0)$.

La regla de transformaci´on $R_{10}R_{21}$ establece que el primer paso es transformar el punto $\mathbf{p}_2$ en coordenadas del sistema $\Sigma_1(x_1, y_1, z_1)$ por medio de R_{21}, posteriormente al punto $\mathbf{p}_0$ por medio de R_{10}. La regla de composici´on de rotaciones sucesivas se establece como:

$$R_{20} = R_{10}R_{21} \tag{3.23}$$

La regla de composición de rotaciones (3.23) puede ser interpretada de la siguiente manera: supóngase que inicialmente los tres sistemas Σ_1, Σ_2 y Σ_3 coinciden. Primero se rota el sistema de referencia $\Sigma_1(x_1, y_1, z_1)$ relativo a $\Sigma_0(x_0, y_0, z_0)$ de acuerdo a la transformación R_{10}. Ahora con los sistemas de referencia coincidentes Σ_1 y Σ_2, se rota el sistema $\Sigma_2(x_2, y_2, z_2)$ relativo a $\Sigma_1(x_1, y_1, z_1)$ de acuerdo a la transformación R_{21}. El sistema de referencia $\Sigma_2(x_2, y_2, z_2)$ tiene una orientación con respecto al sistema de referencia $\Sigma_0(x_0, y_0, z_0)$ dado por la matriz $R_{10}R_{21}$ como se muestra en la figura 3.11.

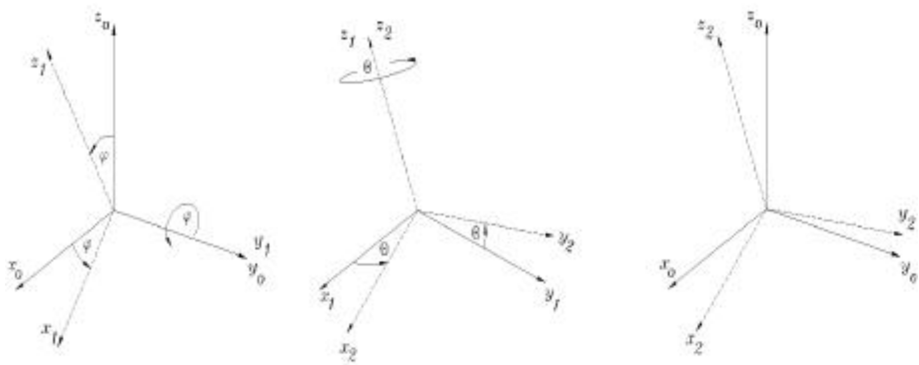


Figura 3.11 Regla de composición de rotaciones sucesivas.

La regla de composición de rotación sucesivas (3.23) se basa en generar nuevos sistemas de referencia consecutivos. Por ejemplo, se forma el sistema de referencia $\Sigma_1(x_1, y_1, z_1)$ como consecuencia de rotar un ángulo determinado alrededor de uno de los ejes principales del sistema $\Sigma_0(x_0, y_0, z_0)$, seguido por otra rotación ahora alrededor del eje y_1 del sistema actual $\Sigma_1(x_1, y_1, z_1)$ para obtener el sistema de referencia $\Sigma_2(x_2, y_2, z_2)$. Realizando giros en diferente orden, se obtienen otros tipos de representaciones.

Es importante subrayar que la multiplicación de matrices de rotación no es conmutativa, en consecuencia el orden de las rotaciones no es conmutativo.

♣ Ejemplo 3.6

Considere el procedimiento de rotación que se muestra en la figura 3.12; obtener la matriz resultante que define la rotación del sistema de referencia Σ_2 con respecto al sistema de referencia Σ_0.

Solución

En la figura 3.12 se pueden observar los tres sistemas de referencia Σ_0, Σ_1 y Σ_2. Todos los sistemas de referencia comparten el mismo origen.

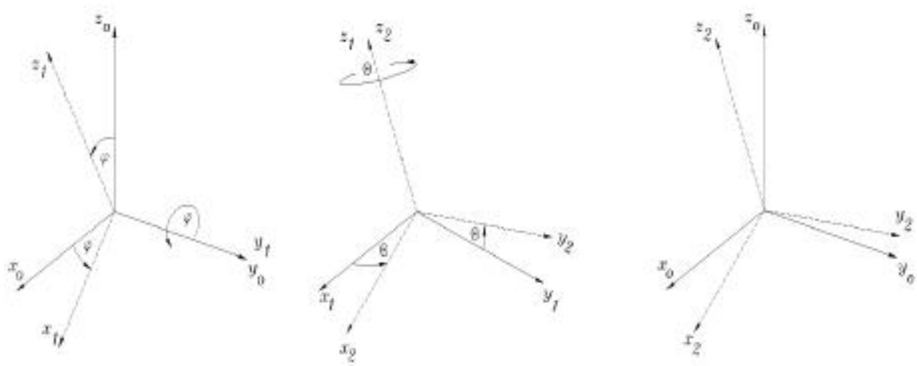


Figura 3.12 Rotación con respecto al sistema actual.

En el esquema izquierdo de la figura 3.12 se encuentran definidos los sistemas de referencia $\Sigma_0(x_0, y_0, z_0)$ y $\Sigma_1(x_1, y_1, z_1)$. Los ejes y_0 y y_1 son paralelos entre sí; existe una rotación por un ángulo φ alrededor del eje y_0 que se representa por la matriz $R_{y_0,\varphi}$. Posteriormente, continúa una rotación por un ángulo θ alrededor del eje z_1 para generar el sistema de referencia $\Sigma_2(x_2, y_2, z_2)$ (parte central de la figura 3.12), los ejes z_1 y z_2 son paralelos entre sí. La orientación relativa del sistema Σ_2 con respecto al sistema de referencia Σ_1 está determinada por la matriz $R_{z_1}(\theta)$.

El esquema derecho de la figura 3.12 muestra la orientación del sistema de referencia Σ_2 con respecto al sistema de referencia Σ_0, cuya matriz de rotación se encuentra dado por:

$$R_{20} = R_{10}R_{21} = R_{y_0}(\varphi)\, R_{z_1}(\theta)$$

$$= \begin{bmatrix} \cos(\varphi) & 0 & \operatorname{sen}(\varphi) \\ 0 & 1 & 0 \\ -\operatorname{sen}(\varphi) & 0 & \cos(\varphi) \end{bmatrix} \begin{bmatrix} \cos(\theta) & -\operatorname{sen}(\theta) & 0 \\ \operatorname{sen}(\theta) & \cos(\cos(\theta)) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos(\psi)\cos(\theta) & -\cos(\psi)\operatorname{sen}(\theta) & \operatorname{sen}(\psi) \\ \operatorname{sen}(\theta) & \cos(\theta) & 0 \\ -\operatorname{sen}(\varphi)\cos(\theta) & \operatorname{sen}(\varphi)\operatorname{sen}(\theta) & \cos(\varphi) \end{bmatrix}$$

donde $R_{10} = R_{y_0}(\varphi)$ y $R_{21} = R_{z_1}(\theta)$.

Para mostrar que el orden de las rotaciones es importante, considere el caso donde las rotaciones se realizan en orden inverso: $R_{20} = R_{21}R_{10} = R_{z_1}(\theta)R_{y_0}(\varphi)$. Es decir, primero la rotación alrededor del eje z_1, seguida por una rotación alrededor del eje y_0. Entonces, la matriz resultante está dada por:

$$\begin{aligned} R_{20} &= R_{z_1}(\theta)\, R_{y_0}(\varphi) \\ &= \begin{bmatrix} \cos(\theta) & -\operatorname{sen}(\theta) & 0 \\ \operatorname{sen}(\theta) & \cos(\cos(\theta)) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\varphi) & 0 & \operatorname{sen}(\varphi) \\ 0 & 1 & 0 \\ -\operatorname{sen}(\varphi) & 0 & \cos(\varphi) \end{bmatrix} \\ &= \begin{bmatrix} \cos(\theta)\cos(\psi) & -\operatorname{sen}(\theta) & \cos(\theta)\operatorname{sen}(\psi) \\ \operatorname{sen}(\theta)\cos(\psi) & \cos(\theta) & \operatorname{sen}(\theta)\operatorname{sen}(\psi) \\ -\operatorname{sen}(\psi) & 0 & \cos(\psi) \end{bmatrix} \end{aligned}$$

> Observe que es importante el orden de rotaciones: $R_{20} = R_{10}R_{21}$. Es decir, en general $R_{20} = R_{21}R_{10}$.

♣ ♣ Ejemplo 3.7

Considere una figura de prueba (flecha) que se encuentra definida en el sistema de referencia $\Sigma_0(x_0, y_0, z_0)$. Realizar rotaciones sucesivas por un angulo $\theta = \frac{\pi}{2}$ alrededor de los eje z_0, del sistema resultante $\Sigma_1(x_1, y_1, z_1)$ realizar una rotación $\theta = \frac{\pi}{2}$ alrededor del eje x_1, y finalmente del sistema generado $\Sigma_2(x_2, y_2, z_2)$ rotar alrededor del eje y_2 por $\theta = \frac{\pi}{2}$

Soluci´on

La figura de prueba es una flecha cuyas coordenadas se encuentran definidas en el sistema de referencia $\Sigma_0(x_0, y_0, z_0)$, cada coordenada de la figura de prueba se encuentra especificada por un punto $\mathbf{p}_0 = [p_{x0}, p_{y0}, p_{z0}]^T$. Un observador en el sistema de referencia $\Sigma_0(x_0, y_0, z_0)$ ver´a a la figura de prueba sin ning´un efecto de rotaci´on. Sin embargo, despu´es de realizar la rotaci´on $R_{z_0}(\frac{\pi}{2})$ un observador colocado en el sistema de referencia $\Sigma_0(x_0, y_0, z_0)$ ver´a a dicha figura rotada 90 grados en el nuevo sistema generado $\Sigma_1(x_1, y_1, z_1)$. Las coordenadas de la flecha en el sistema $\Sigma_1(x_1, y_1, z_1)$ son $\mathbf{p}_1 = R_{z_0}(\frac{\pi}{2})\mathbf{p}_0$. Posteriormente se realiza una nueva rotaci´on de 90 grados alrededor del eje x_1, dada por: $R_{x_1}(\frac{\pi}{2})$, con esto se genera el sistema $\Sigma_2(x_2, y_2, z_2)$. Las coordenadas de la flecha en este sistema $\Sigma_2(x_2, y_2, z_2)$ est´an especificadas por $\mathbf{p}_2 = R_{x_1}(\frac{\pi}{2})\mathbf{p}_1$. La ultima rotaci´on consiste en girar a la imagen de prueba (flecha) 90 grados alrededor del eje y_2. Esta rotaci´on genera el sistema de referencia $\Sigma_3(x_3, y_3, z_3)$ con coordenadas de la imagen $\mathbf{p}_3 = R_{y_2}(\frac{\pi}{2})\mathbf{p}_2$.

La relaci´on del sistema de referencia $\Sigma_3(x_3, y_3, z_3)$ con el sistema de referencia $\Sigma_0(x_0, y_0, z_0)$ es: $\mathbf{p}_3 = R_{y_2}(\frac{\pi}{2})R_{x_1}(\frac{\pi}{2})R_{z_0}(\frac{\pi}{2})\mathbf{p}_0$ o $\mathbf{p}_0 = [R_{y_2}(\frac{\pi}{2})R_{x_1}(\frac{\pi}{2})R_{z_0}(\frac{\pi}{2})]^T\mathbf{p}_3$.

La figura 3.13 muestra la sucesi´on de rotaciones.

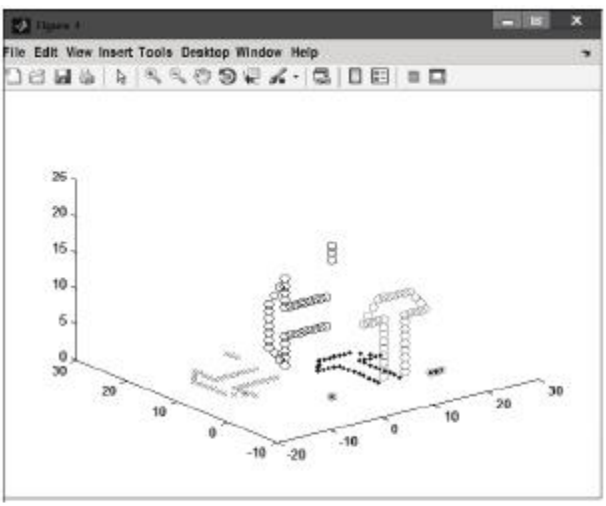


Figura 3.13 Rotaciones sucesivas de la figura de prueba alrededor de los ejes z_0, x_1, y_2.

C´odigo Fuente 3.5 Rotaciones alrededor de los ejes z_0, x_1, y_2

%MATLAB Aplicado a Rob´otica y Mecatr´onica.
%Editorial Alfaomega, Fernando Reyes Cort´es.
%Cap´ıtulo 3 Cinem´atica %Archivo cap3_rotacion.m

Rotaciones alrededor de los ejes z_0, x_1, y_2

```
clc; clear all; close all;
pzo=[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0];
pyo=[0,0,0,0,0,9,9,10,9, 11, 9,12, 1, 2, 3, 4, 5, 6,7, 8, 9, 12, 12, 12, 12,1, 2, 3, 4, 5,
6, 7, 8, 9, 12, 9, 12, 9, 11, 9, 10, 9,0,0,0 ];
pxo=[0,0,0,0,0,6,7,7, 8, 8, 9,9, 10,10, 10, 10, 10, 10, 10, 10, 10, 10, 11, 12, 13, 14,14,
14, 14, 14, 14, 14, 14,14, 14, 15, 15, 16,16, 17,17,18,19,20,21];
theta=90*3.1416/180.0;
R_ztheta=[ cos(theta), -sin(theta), 0; sin(theta), cos(theta), 0; 0, 0,1];
R_xtheta=[ 1, 0, 0; 0, cos(theta), -sin(theta); 0, sin(theta), cos(theta)];
R_ytheta=[ cos(theta), 0, sin(theta); 0, 1, 0; -sin(theta), 0, cos(theta)];
% [x1; y1; z1] = Rz(θ)^T [x0; y0; z0]
Sigma1=R_ztheta*[pxo; pyo; pzo];
px1=Sigma1(1,:);py1=Sigma1(2,:); pz1=Sigma1(3,:);
% [x2; y2; z2] = Rx(θ) [x1; y1; z1]
Sigma2=R_xtheta*[px1; py1; pz1];
px2=Sigma2(1,:); py2=Sigma2(2,:); pz2=Sigma2(3,:);
% [x3; y3; z3] = Ry(θ) [x2; y2; z2]
Sigma3=R_ytheta*[px2; py2; pz2];
px3=Sigma3(1,:); py3=Sigma3(2,:); pz3=Sigma3(3,:);
plot3(pxo,pyo,pzo,'.',px1,py1,pz1,'x',px2,py2,pz2,'o',px3,py3,pz3,'o')
```

3.5 Transformaciones de traslación

Considere el sistema de referencia cartesiano fijo $\Sigma_0(x_0, y_0, z_0)$ y el sistema de referencia $\Sigma_1(x_1, y_1, z_1)$, donde sus respectivos orígenes son no coincidentes. El origen del sistema de referencia Σ_1 se encuentra desplazado una distancia $\mathbf{d}_{10}$ con respecto al origen del sistema Σ_0, como se muestra en la figura 3.14.

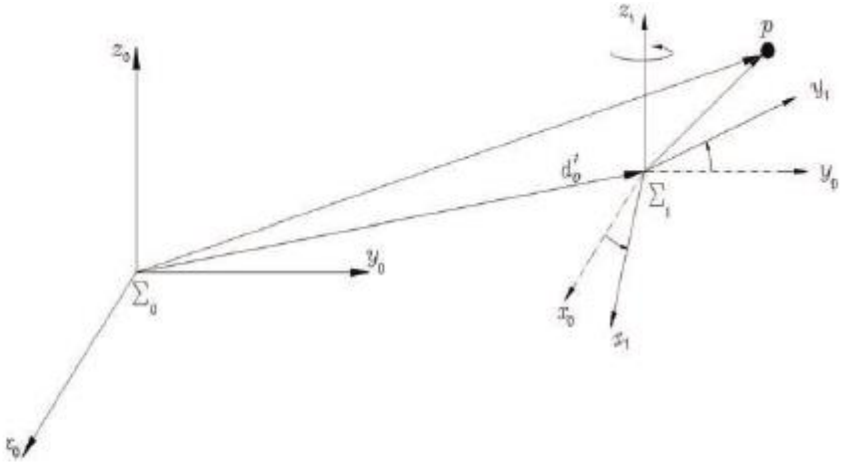


Figura 3.14 Transformaciones de traslación y rotación del sistema Σ_1 con respecto al sistema Σ_0.

El vector $\mathbf{d}_{10}$ está expresado en coordenadas del sistema Σ_0: $\mathbf{d}_{10} = [d_{10x}, d_{10y}, d_{10z}]^T$, entonces cualquier punto $\mathbf{p}$ tiene representación $\mathbf{p}_0$ y $\mathbf{p}_1$. La relación general entre los sistemas de referencia $\Sigma_0(x_0, y_0, z_0)$ y $\Sigma_1(x_1, y_1, z_1)$ incluyendo la matriz de rotación R_{10} y el vector de traslación $\mathbf{d}_{10}$ es:

$$\begin{aligned} \mathbf{p}_0 &= \mathbf{d}_{10} + R_{10}\mathbf{p}_1 \\ &= \begin{bmatrix} d_{10x} \\ d_{10y} \\ d_{10z} \end{bmatrix} + R_{10} \begin{bmatrix} p_{1x} \\ p_{1y} \\ p_{1z} \end{bmatrix}. \end{aligned} \tag{3.24}$$

Para el caso de tres sistemas de referencia cartesianos $\Sigma_0(x_0y_0z_0)$, $\Sigma_1(x_1y_1z_1)$ y

$\Sigma_2(x_2y_2z_2)$ (ver figura 3.15) se obtienen las siguientes expresiones:

$$\mathbf{p}_0 = \mathbf{d}_{10} + R_{10}\mathbf{p}_1 \tag{3.25}$$
$$\mathbf{p}_1 = \mathbf{d}_{21} + R_{21}\mathbf{p}_2 \tag{3.26}$$

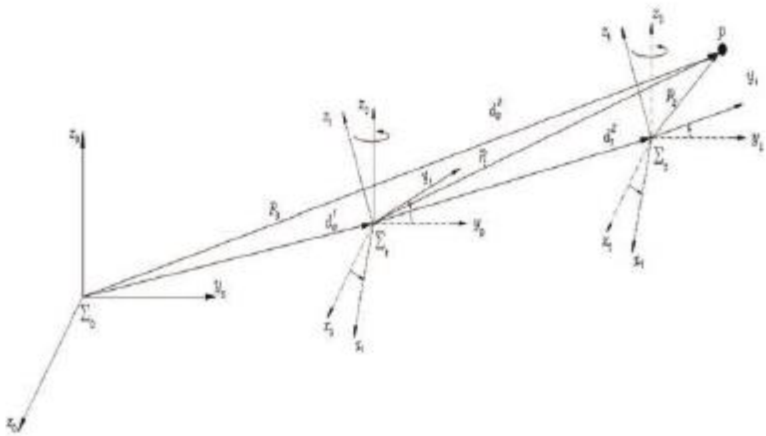


Figura 3.15 Traslaci´on y rotaci´on de Σ_1 y Σ_2 con respecto al sistema Σ_0.

Sustituyendo la ecuaci´on (3.26) en (3.25) se tiene:

$$\mathbf{p}_0 = \mathbf{d}_{10} + R_{10}\mathbf{d}_{21} + R_{10}R_{21}\mathbf{p}_2. \tag{3.27}$$

Por lo tanto, la regla de transformaci´on de traslaci´on y orientaci´on de un punto $\mathbf{p}_2$ en el sistema $\Sigma_2(x_2y_2z_2)$ hacia un punto $\mathbf{p}_0$ en el sistema $\Sigma_0(x_0y_0z_0)$ adquiere la siguiente estructura:

$$\mathbf{p}_0 = \mathbf{d}_{20} + R_{20}\mathbf{p}_2 \tag{3.28}$$

donde

$$R_{20} = R_{10}R_{21}$$
$$\mathbf{d}_{20} = \mathbf{d}_{10} + R_{10}\mathbf{d}_{21}$$

3.6 Transformaciones homogéneas

La notación más común para representar la transformación de traslación y rotación en forma compacta se conoce como **transformación homogénea**. Por ejemplo, para representar el caso de traslación y rotación del sistema $\Sigma_1(x_1y_1z_1)$ con respecto al sistema $\Sigma_0(x_0y_0z_0)$

$$\mathbf{p}_0 = \mathbf{d}_{10} + R_{10}\mathbf{p}_1$$

la transformación homogénea se realiza con la siguiente notación:

$$H_0^1 = \begin{bmatrix} R_{10} & \mathbf{d}_{10} \\ \cdots & \cdots \\ \mathbf{0}^T & 1 \end{bmatrix} \quad \text{Matriz de traslación} \tag{3.29}$$

donde $R_{10} \in SO(3)$ y $\mathbf{d}_{10} \in \mathbb{R}^3$. Para propósitos de acoplamiento en dimensiones, el vector $\mathbf{0}^T$ y el número 1 aparecen en el último renglón.

La representación inversa que relaciona el punto $\mathbf{p}_1$ en función del punto $\mathbf{p}_0$ adquiere la siguiente forma:

$$\mathbf{p}_1 = -R_{10}^T\mathbf{d}_{10} + R_{10}^T\mathbf{p}_0$$

entonces, la transformación homogénea inversa está determinada por:

$$H_0^{1\,-1} = \begin{bmatrix} R_{10}^T & -R_{10}^T\mathbf{d}_{10} \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{3.30}$$

Las matrices de rotación permiten modelar la orientación de la herramienta de trabajo colocada en el extremo final del robot, y junto con las transformaciones homogéneas dentro de una sola matriz incluye la orientación y posición de la herramienta de trabajo, formando la estructura del modelo cinemático directo.

3.7 Librerías para matrices homogéneas

En esta sección se presentan las librerías de transformación homogénea de rotación y traslación con respecto a los ejes principales x, y, z.

Matrices de transformación homogénea de rotación

Las matrices de transformación homogénea de rotación con respecto a los ejes x, y, z, respectivamente, tienen la siguiente estructura:

$$HR_x(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & -\operatorname{sen}(\theta) & 0 \\ 0 & \operatorname{sen}(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad HR_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \operatorname{sen}(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ -\operatorname{sen}(\theta) & 0 & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$HR_z(\theta) = \begin{bmatrix} \cos(\theta) & -\operatorname{sen}(\theta) & 0 & 0 \\ \operatorname{sen}(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Matrices de transformación homogénea de traslación

Las matrices de transformación homogénea de traslación con respecto a los ejes x, y, z respectivamente tienen la siguiente estructura:

$$HT_x(d) = \begin{bmatrix} 1 & 0 & 0 & d \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad HT_y(d) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & d \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad HT_z(d) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Matriz de transformación homogénea Denavit-Hartenberg

La matriz de transformación homogénea Denavit Hartenberg tiene la forma siguiente:

$$H = \begin{bmatrix} R & \begin{bmatrix} x \\ y \\ z \end{bmatrix} \\ \mathbf{0}^T & 1 \end{bmatrix}$$

Matriz de transformación homogénea $HR_x(\theta)$

La sintaxis de la función de transformación homogénea de rotación $HR_x(\theta)$ alrededor del eje x está dada por:

```
RHx=HRx(θ)
```

donde θ es el ángulo de rotación alrededor del eje x y es la variable de la función HRx(θ). Retorna la matriz de transformación homogénea RHx.

Código Fuente 3.6 Función $HR_x(\theta)$

%MATLAB Aplicado a Robótica y Mecatrónica.
%Editorial Alfaomega, Fernando Reyes Cortés.
%Capítulo 3 Cinemática %función HRx.m

Función $HR_x(\theta)$

```
function RHx=HRx(theta)
    dato=whos('theta');
    if strcmp(dato.class, 'sym') %variables simbólicas
        RHx=[1, 0, 0, 0;
                0, cos(theta), -sin(theta), 0;
                0, sin(theta), cos(theta), 0;
                0, 0, 0, 1];
        else digits(3); %cálculos numéricos
        RHx=round([1, 0, 0, 0;
                0, vpa(cos(theta),3), vpa(-sin(theta),3), 0;
                0, vpa(sin(theta),3), vpa(cos(theta),3), 0;
                0, 0, 0, 1]);
    end
end
```

Matriz de transformación homogénea $HR_y(\theta)$

La sintaxis de la función de transformación homogénea de rotación $HR_y(q)$ alrededor del eje y está dada por:

RHy=HRy(θ)

donde θ es el ángulo de rotación alrededor del eje y y es la variable de la función HRy(θ). Retorna la matriz de transformación homogénea RHy.

Código Fuente 3.7 Función $HR_y(\theta)$

%MATLAB Aplicado a Robótica y Mecatrónica.
%Editorial Alfaomega, Fernando Reyes Cortés.
%Capítulo 3 Cinemática %función HRy.m

Función $HR_y(\theta)$

```
function RHy=HRy(theta)
    dato=whos('theta'); if strcmp(dato.class, 'sym') %variables
    simbólicas
        RHy=[cos(theta), 0, sin(theta), 0;
                0, 1, 0, 0;
                -sin(theta), 0, cos(theta), 0;
                0, 0, 0, 1];
        else digits(3); %cálculos numéricos
        RHy=round([ vpa(cos(theta),3), 0, vpa(sin(theta),3), 0;
                0, 1, 0, 0;
                vpa(-sin(theta),3), 0, vpa(cos(theta),3), 0;
                0, 0, 0, 1]);
    end
end
```

Matriz de transformación homogénea $HR_z(\theta)$

La sintaxis de la función de transformación homogénea de rotación $HR_z(\theta)$ alrededor del eje z está dada por:

RHz=HRz(θ)

donde θ es el ángulo de rotación alrededor del eje z y es la variable de la función HRz(θ). Retorna la matriz de transformación homogénea RHz.

Código Fuente 3.8 Función $HR_z(\theta)$

```
%MATLAB Aplicado a Robótica y Mecatrónica.
%Editorial Alfaomega, Fernando Reyes Cortés.
%Capítulo 3 Cinemática %función HRz.m
```

Función $HR_z(\theta)$

```
function RHz=HRz(theta)
    dato=whos('theta');
    if strcmp(dato.class, 'sym') %variables simbólicas
        RHz=[cos(theta), -sin(theta), 0, 0;
             sin(theta), cos(theta), 0, 0;
             0, 0, 1, 0;
             0, 0, 0, 1];
        else digits(3); %cálculos numéricos
        RHz=round([ vpa(cos(theta),3), vpa(-sin(theta),3), 0, 0;
                    vpa(sin(theta),3), vpa(cos(theta),3), 0, 0;
                    0, 0, 1, 0;
                    0, 0, 0, 1]);
    end
end
```

Matriz de transformaci´on homog´enea $HT_x(d)$

La sintaxis de la funci´on de transformaci´on homog´enea de traslaci´on $HT_x(d)$ a lo largo del eje x est´a dada por:

```
THx=HTx(d)
```

donde d es el desplazamiento de traslaci´on sobre el eje x. Retorna la matriz de transformaci´on homog´enea THx.

C´odigo Fuente 3.9 Funci´on $HT_x(d)$

%MATLAB Aplicado a Rob´otica y Mecatr´onica.
%Editorial Alfaomega, Fernando Reyes Cort´es.
%Cap´ıtulo 3 Cinem´atica %funci´on HTx.m

Funci´on $HT_x(d)$

```
function Tx=HTx(d)
    Tx=[ 1 0 0 d; 0 1 0 0; 0 0 1 0; 0 0 0 1];
end
```

Matriz de transformaci´on homog´enea $HT_y(d)$

La sintaxis de la funci´on de transformaci´on homog´enea de traslaci´on $HT_y(d)$ a lo largo del eje y est´a dada por:

```
THy=HTy(d)
```

donde d es el desplazamiento lineal de traslaci´on sobre el eje y. Retorna la matriz de transformaci´on homog´enea THy.

Código Fuente 3.10 **Función $HT_y(d)$**

```
%MATLAB Aplicado a Robótica y Mecatrónica.
%Editorial Alfaomega, Fernando Reyes Cortés.
%Capítulo 3 Cinemática %función HTy.m
```

Función $HT_y(d)$

```
function Tz=HTy(d)
    Tz=[ 1 0 0 0; 0 1 0 d; 0 0 1 0; 0 0 0 1];
end
```

Matriz de transformación homogénea $HT_z(d)$

La sintaxis de la función de transformación homogénea de traslación $HT_z(d)$ a lo largo del eje z está dada por:

THz=HTz(d)

donde d es el desplazamiento lineal de traslación sobre el eje z. Retorna la matriz de transformación homogénea THz.

Código Fuente 3.11 **Función $HT_z(d)$**

```
%MATLAB Aplicado a Robótica y Mecatrónica.
%Editorial Alfaomega, Fernando Reyes Cortés.
%Capítulo 3 Cinemática %función HTz.m
```

Función $HT_z(d)$

```
function Tz=HTz(d)
    Tz=[ 1 0 0 0; 0 1 0 0; 0 0 1 d; 0 0 0 1];
end
```

3.7.7. Matriz de transformación DH

La función H_DH(H) extrae de la matriz de transformación homogénea H la matriz de rotación y el vector de coordenadas cartesianas.

La sintaxis de la función de transformación homogénea DH está dada por:

```
[R vect_d vect_cero c]=H_DH(H)
```

donde H es la variable de entrada y representa la matriz de transformación homogénea. Retorna la matriz de rotación R , el vector de coordenadas cartesianas vect_d, el vector vect_cero=$[0, 0, 0]^T$ y la constante unitaria c=1.

Código Fuente 3.12 Función H_DH

%MATLAB Aplicado a Robótica y Mecatrónica.
%Editorial Alfaomega, Fernando Reyes Cortés.
%Capítulo 3 Cinemática %función H_DH.m

Función H_DH

```
function [R vect_d vect_cero c]=H_DH(H)
    for i=1:3
        for j=1:3
            R(i,j)=H(i,j);
        end
    end
    %estructura de la matriz de transformación homogénea
    vect_d=[H(1,4); H(2,4); H(3,4)];
    vect_cero=[0;0;0]';
    c=1;
end
```

3.8 Resumen

Cinemática de robots manipuladores y sistemas mecatrónicos encuentra su fundamento en una clase particular de matrices denominadas ortogonales. Estos preliminares matemáticos constituyen la base para desarrollar propiedades, reglas y operaciones de traslación y rotación entre dos sistemas de referencia $\Sigma_0(x_0, y_0, z_0)$ y $\Sigma_1(x_1, y_1, z_1)$ bajo una estructura matemática denominada matriz de transformación homogénea.

La matriz ortogonal R_1^0 representa la orientación del sistema de referencia $\Sigma_0(x_0, y_0, z_0)$ con respecto al sistema $\Sigma_1(x_1, y_1, z_1)$. De particular interés son las matrices de rotación $R_x(\theta)$, $R_y(\theta)$, $R_z(\theta)$ ya que representan la rotación alrededor de los ejes principales x, y, z, respectivamente. El orden como se realice la rotación determina la regla o procedimiento de orientar un sistema de referencia con respecto a otro.

En este capítulo se ha desarrollado un conjunto de librerías para realizar operaciones de traslación y rotación. La librería R=Rz(θ) realiza la rotación del sistema de referencia $\Sigma_1(x_1, y_1, z_1)$ con respecto al sistema de referencia $\Sigma_0(x_0, y_0, z_0)$. Esta función puede trabajar con variables simbólicas y también para realizar aplicaciones numéricas. De manera análoga se puede implementar las funciones R=Rx(θ) y R=Ry(θ).

> Cuando hay combinación de cálculos numéricos con variables simbólicas, como es el caso de las matrices de transformación homogénea (rotación y traslación) la funciones vpa, round y double pueden ayudar a presentar el resultado numérico de manera conveniente.
> Por otro lado, para simplificar las expresiones simbólicas se recomienda usar la función `simplify`, por lo que el lector puede modificar el código fuente de las librerías desarrolladas para presentar resultados simbólicos o numéricos a su entera conveniencia.

La tabla 3.1 contiene el resumen de las librerías para realizar las matrices de transformación homogénea de traslación y rotación.

Tabla 3.1 Matrices de transformación homogénea

Función	Sintaxis
Matriz de transformación homogénea de rotación alrededor del eje x	RHx=HRx(θ) $\theta \in$ IR.
Matriz de transformación homogénea de rotación alrededor del eje y	RHy=HRy(θ) $\theta \in$ IR.
Matriz de transformación homogénea de rotación alrededor del eje z	RHz=HRz(θ) $\theta \in$ IR.
Matriz de transformación homogénea de traslación sobre el eje x	THx=HTx(d) $d \in$ IR.
Matriz de transformación homogénea de traslación sobre el eje y	THy=HTy(d) $d \in$ IR.
Matriz de transformación homogénea de traslación sobre el eje z	THz=HTz(d) $d \in$ IR.
Matriz de transformación homogénea Denavit-Hartenberg	[R vect_d vect_cero c]=H_DH(H) $R \in SO(3)$ es la matriz de rotación vect_d es el vector con coordenadas cartsianas: $[x \quad y \quad z]^T$ vect_cero=$[0, 0, 0]^T$ c=1.

Cap´ıtulo

4

Cinem´atica directa

l_i	α_i	$d_i \therefore \beta_i$	θ_i

$$H_i^{i-1} = H_{R_{zi}}(\theta_i) H_{T_{zi}}(d_i \therefore \beta_i) H_{T_{xi}}(l_i) H_{R_{xi}}(\alpha_i)$$

Objetivos

Presentar el modelo de cinemática directa de las principales configuraciones de robots industriales considerando los parámetros geométricos y desarrollar librerías en lenguaje MATLAB (toolbox) que permitan realizar aplicaciones en el área de cinemática directa de robots manipuladores.

Objetivos particulares:

- Matrices de rotación.
- Transformaciones homogéneas.
- Método de Denavit-Hartenberg.
- Cinemática directa cartesiana.
- Librerías de análisis y diseño de cinemática de robots industriales.

4.1 Introducción

Cinemática es la parte de la física que estudia el movimiento de sistemas mecánicos, sin tomar en cuenta las fuerzas que originan dicho movimiento, por lo tanto no involucra ecuaciones diferenciales como en el caso de la dinámica. Al estudio de la cinemática de sistemas mecatrónicos y robots manipuladores se le denomina **cinemática directa**, se refiere al estudio analítico del movimiento del robot con respecto a un sistema de referencia cartesiano fijo $\Sigma_0(x_0, y_0, x_0)$ relacionando la dependencia que existe entre las coordenadas articulares o generalizadas $\mathbf{q} \in IR^n$, y los parámetros geométricos (longitudes del i-ésimo eslabón l_i), con coordenadas cartesianas $[x, y, z]^T \in IR^3$ y la orientación $[\theta, \varphi, \psi]^T \in IR^3$ del extremo final del robot a través de una función vectorial $\mathbf{f}_R$ continua y diferenciable en la variable de estado $\mathbf{q}$, generalmente no lineal.

Cinemática directa es una función vectorial $\mathbf{f}_R(l_i, \mathbf{q})$ que relaciona las coordenadas articulares $\mathbf{q} \in IR^n$ y propiedades geométricas del sistema mecánico l_i con las coordenadas cartesianas $[x, y, z]^T \in IR^3$ del robot y la orientación $[\theta, \varphi, \psi]^T \in IR^3$ de la herramienta colocada en el extremo final. Es decir $\mathbf{f}_R : IR^n \rightarrow IR^m$ tal que:

$$\begin{bmatrix} x \\ y \\ z \\ \theta \\ \varphi \\ \psi \end{bmatrix} = \mathbf{f}_R(l_i, \mathbf{q}) \tag{4.1}$$

donde n indica el número de grados de libertad y la dimensión del vector de coordenadas articulares $\mathbf{q}$, m es la dimensión conjunta de las coordenadas cartesianas y la orientación de la herramienta de trabajo.

De manera general, el posicionamiento del extremo final del robot en el espacio tridimensional (pose) requiere de 6 coordenadas ($m = 6$): 3 coordenadas para la posición cartesiana y 3 coordenadas para la orientación de la herramienta de trabajo. Dependiendo de la aplicación del robot se pueden requerir menos coordenadas de posición y orientación. Por ejemplo, un robot para pintura de armaduras automotrices requiere las 6 coordenadas, en contraste con un robot que corta figuras de plástico sobre un plano requiere 2 coordenadas cartesianas de posición y ninguna de orientación. Cuando $n > m$ se denomina robots redundantes. El empleo de la cinemática directa resulta de utilidad en la planificación de trayectorias y en el control cartesiano. El papel fundamental de la cinemática directa, es computar la posición y orientación del extremo final del robot manipulador como una función de las variables articulares.

Un robot manipulador se considera como una serie de eslabones interconectados a través de articulaciones (servomotores) rotacionales o prismáticas en forma de cadena cinemática abierta, es decir el extremo final donde se coloca la herramienta no se encuentra conectada mecánicamente a la primera articulación (base) del robot. Desde el punto de vista topológico, la cadena cinemática se considera abierta cuando los dos extremos de la cadena no se tocan. De otra manera la cadena cinemática formaría un lazo si sus dos extremos están mecánicamente unidos.

La estructura mecánica del robot manipulador se caracteriza por tener un número de grados de libertad, los cuales determinan en forma única su configuración. Típicamente, cada grado de libertad está asociado a una articulación (variable articular $\mathbf{q}$).

4.2 Cinemática inversa

Dada la posición del extremo final del robot en coordenadas cartesianas $[x, y, z]^T$ y la orientación $[\psi, \theta, \varphi]^T$, con respecto a un sistema de referencia fijo $\Sigma_0(x_0, y_0, z_0)$, así como los parámetros geométricos l_i, entonces surge la pregunta natural: ¿pueden obtenerse las coordenadas articulares del robot $\mathbf{q}$ para que el extremo final del robot se posicione en las coordenadas cartesianas solicitadas, con la orientación requerida?

El problema planteado se conoce como **cinemática inversa** y representa un areá de la robótica de mayor complejidad que la cinemática directa. Para un robot manipulador siempre es posible encontrar el modelo de cinemática directa, mientras que en la cinemática inversa pueden haber varias soluciones e inclusive no existir solución analítica; si este es el caso, entonces como posibles formas de solución pueden proponerse redes neuronales, métodos numéricos, iterativos, geométricos, etcétera.

> La **cinemática inversa** es un problema no lineal que relaciona las coordenadas articulares en función de las coordenadas cartesianas y la orientación de la herramienta del extremo final del robot manipulador
>
> $$\mathbf{q} = \mathbf{f}_R^{-1}(x, y, z, l_i, \theta, \varphi, \psi) \tag{4.2}$$
>
> donde $\mathbf{f}_R^{-1}(x, y, z, l_i, \theta, \varphi, \psi)$ es función inversa de la ecuación (4.1).

4.3 Cinemática diferencial

La cinemática diferencial directa es la derivada con respecto al tiempo de la cinemática directa

$$\frac{d}{dt}[x \;\; y \;\; z \;\; \theta \;\; \varphi \;\; \psi]^T = \begin{bmatrix} \mathbf{v} \\ \mathbf{w} \end{bmatrix} = \frac{d}{dt}\mathbf{f}_R(\mathbf{q}) \tag{4.3}$$
$$= \frac{\partial \mathbf{f}_R(\mathbf{q})}{\partial \mathbf{q}}\dot{\mathbf{q}} = J(\mathbf{q})\dot{\mathbf{q}}.$$

Como se ve, ésta relaciona la velocidad articular $\dot{\mathbf{q}} \in IR^n$ con la velocidad lineal $\mathbf{v} = \frac{d}{dt}[x, y, z]^T = [\dot{x}, \dot{y}, \dot{z}]^T \in IR^3$ y la velocidad angular $\mathbf{w} = \frac{d}{dt}[\theta, \varphi, \psi]^T = [\dot{\theta}, \dot{\varphi}, \dot{\psi}]^T \in IR^3$, además el mapeo es descrito en términos de una matriz $J(\mathbf{q}) = \frac{\partial \mathbf{f}_R(\mathbf{q})}{\partial \mathbf{q}} \in IR^{6\times n}$ denominada **jacobiano del robot o jacobiano analítico**:

$$J(\mathbf{q}) = \begin{bmatrix} J_{\mathbf{v}}(\mathbf{q}) \\ J_{\mathbf{w}}(\mathbf{q}) \end{bmatrix}. \tag{4.4}$$

$J_{\mathbf{v}}(\mathbf{q}) \in IR^{3\times n}$ relaciona la velocidad articular $\dot{\mathbf{q}} \in IR^n$ con la velocidad lineal $\mathbf{v} \in IR^3$, mientras que $J_{\mathbf{w}}(\mathbf{q}) \in IR^{3\times n}$ relaciona la velocidad angular $\mathbf{w} \in IR^3$ con la velocidad articular $\dot{\mathbf{q}} \in IR^n$, es decir:

$$\mathbf{v} = J(\mathbf{q})\dot{\mathbf{q}} = J_{\mathbf{w}}(\mathbf{q})\dot{\mathbf{q}} \tag{4.5}$$

El jacobiano del robot representa una importante herramienta en robótica que sirve para caracterizar a un robot manipulador, encontrar configuraciones singulares, analizar redundancia, determinar la cinemática diferencial inversa, así como describir la relación entre la fuerza aplicada y los pares o torques resultantes del extremo final. Es indispensable para el análisis y diseño de algoritmos de control cartesiano.

Hay varias formas de seleccionar la orientación de la herramienta del robot manipulador: si de manera particular dicha orientación es representada por los ángulos de Euler (un sistema de referencia asociado al extremo final del robot o a la herramienta de trabajo), entonces la velocidad angular $\mathbf{w} = [\theta, \dot{\varphi}, \psi]^T \in IR^3$ relaciona la matriz jacobiano analítico, como se encuentra descrita en la ecuación (4.3). Otra posible forma de modelar la orientación de la herramienta del robot es expresarla directamente en un sistema de referencia específico, por ejemplo al origen localizado en la base del robot, entonces a la matriz $J(\mathbf{q})$ se le denomina **jacobiano geométrico** que depende de la configuración del robot manipulador. El jacobiano analítico difiere del jacobiano geométrico: básicamente la diferencia se encuentra en cómo modelar la orientación de la herramienta de trabajo del robot.

La cinemática diferencial inversa representa la relación entre la velocidad articular $\dot{\mathbf{q}}$ con la velocidad lineal de movimiento $\mathbf{v}$ y la velocidad angular $\mathbf{w}$, expresada en términos de la matriz inversa del jacobiano del robot:

$$\dot{\mathbf{q}} = J^{-1}(\mathbf{q}) \begin{bmatrix} \mathbf{v} \\ \mathbf{w} \end{bmatrix} \tag{4.6}$$

donde $J^{-1}(\mathbf{q}) \in IR^{6\times n}$ es la matriz inversa del jacobiano del robot, la cual existe si es una matriz cuadrada y su determinante es diferente a cero.

Si el determinante del jacobiano del robot $J(\mathbf{q})$ es cero, entonces se dice que no es de rango completo y se presentan problemas de **singularidades**.

Singularidad significa que no es posible indicarle un movimiento arbitrario al extremo final del robot, es decir para una velocidad lineal $\mathbf{v}$ y velocidad angular $\mathbf{w}$ finitas puede corresponder una velocidad articular $\dot{\mathbf{q}}$ infinita.

Puede existir un conjunto infinito de soluciones para la cinemática directa.

La cinemática inversa diferencial tiene un número infinito de soluciones.

En control cartesiano la fuerza aplicada al robot puede provocar un par infinito a las articulaciones del robot.

Dependiendo del tipo de robot, las singularidades pueden generar un número infinito de puntos de equilibrio en la ecuación en lazo cerrado, formada por la dinámica del robot y la estructura cartesiana de control.

4.4 Clasificación de robots industriales

Un robot industrial está compuesto por una serie consecutiva de eslabones y articulaciones para formar una cadena en cinemática abierta, la cual es la estructura mecánica básica de un robot industrial. La cadena en cinemática abierta está formada de la siguiente manera: la primera articulación sirve para formar la base; a continuación siguen conexiones sucesivas entre articulaciones y eslabones, en el extremo final del último eslabón no hay articulación, generalmente se destina a colocar la herramienta de trabajo para llevar a cabo una aplicación específica. El extremo final del robot no se encuentra conectado f´ısicamente a la base como se muestra en la figura 4.1:

Figura 4.1 Cadena en cinem´atica abierta.

Las articulaciones se construyen por medio de un servomotor y representan la interconexi´on entre dos eslabones consecutivos. Una articulaci´on puede realizar s´olo un tipo de movimiento, ya sea lineal, tambi´en conocida como prism´atica, y rotacional. La figura 4.2 presenta el tipo de articulaciones:

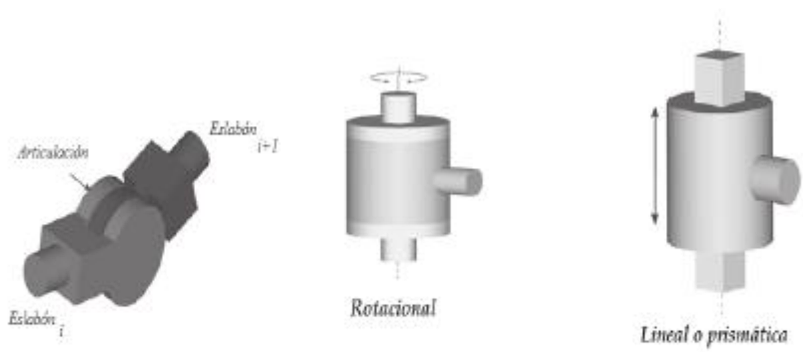


Figura 4.2 Tipo de articulaciones: rotacional y lineal o prism´atica.

La figura 4.3 muestra la analog´ıa entre el brazo humano y un brazo robot o robot industrial. La articulaci´on de la base corresponde a la cintura. La articulaci´on del hombro (shoulder) debe ser la de mayor capacidad con respecto a las otras articulaciones, ya que es la que mueve y soporta el peso de la articulaci´on del codo (elbow) y de la herramienta de trabajo, as´ı como la carga de objetos que realice durante una determinada aplicaci´on.

Dependiendo del tipo de articulaciones (lineales o rotacionales) que se encuentran incluidas en la estructura mec´anica en cinem´atica abierta de la base, hombro y codo del robot (sin incluir las articulaciones de la orientaci´on de la herramienta de

Figura 4.3 Base, hombro y codo de un robot industrial.

trabajo), se desprende la clasificaci´on general de robots manipuladores industriales, tambi´en conocidos como brazos robots: antropom´orfico, esf´erico, cil´ındrico, SCARA y cartesiano.

Tabla 4.1 Clasificaci´on de robots industriales

Robot	Caracter´ısticas
Antropom´orfico (RRR)	3 articulaciones rotacionales
SCARA (RRP)	2 articulaciones rotacionales y 1 prism´atica
Esf´erico (RRP)	2 articulaciones rotacionales y 1 prism´atica
Cil´ındrico (RPP)	1 articulaci´on rotacional y 2 prism´aticas
Cartesiano (PPP)	3 articulaciones prism´aticas

La nomenclatura empleada en robots industriales para representar el tipo de movimiento que realizan sus articulaciones est´a dada de la siguiente manera: R significa articulaci´on tipo rotacional, mientras que la letra P representa una articulaci´on prism´atica. El orden en que se presentan corresponde a las articulaciones de la base, hombro y codo, respectivamente. Por ejemplo, en la tabla 4.1 la notaci´on robot cil´ındrico (RPP) significa que la base es una articulaci´on rotacional, mientras que el hombro y codo corresponden a articulaciones prism´aticas.

En la figura 4.4 se muestra la clasificaci´on de las 5 configuraciones de robots industriales.

Figura 4.4 Clasificación de los robots industriales.

4.5 Convención Denavit-Hartenberg

El método de Denavit-Hartenberg es una herramienta ampliamente conocida en el área de ingeniería, ya que ofrece un procedimiento sencillo para obtener el modelo cinemático directo cuya estructura queda en términos de la transformaciones homogéneas.

Jaques Denavit y Richard S. Hartenberg en 1955 presentaron un procedimiento para obtener una representación mínima de la orientación y traslación de robots manipuladores. Consiste en determinar una tabla de parámetros relacionados con los eslabones del robot. La convención Denavit-Hartenberg toma como referencia el diagrama de un robot manipulador en cadena cinemática abierta como se muestra en la figura 4.5.

Las variables articulares en la representación Denavit-Hartenberg se denotan con θ_i para el tipo rotacional, prismática o lineal por d_i; este parámetro d_i también hace el papel de representar el ancho del servomotor de la articulación rotacional más el espesor de la placa métalica del eslabón, en este caso se denota por el símbolo β_i; la longitud del eslabón se representa con l_i y el ángulo de separación entre los ejes z_i y z_{i-1} se denota con α_i.

El ángulo θ_i es el ángulo entre los ejes x_{i-1} y x_i medido alrededor del eje z_{i-1}; d_i es

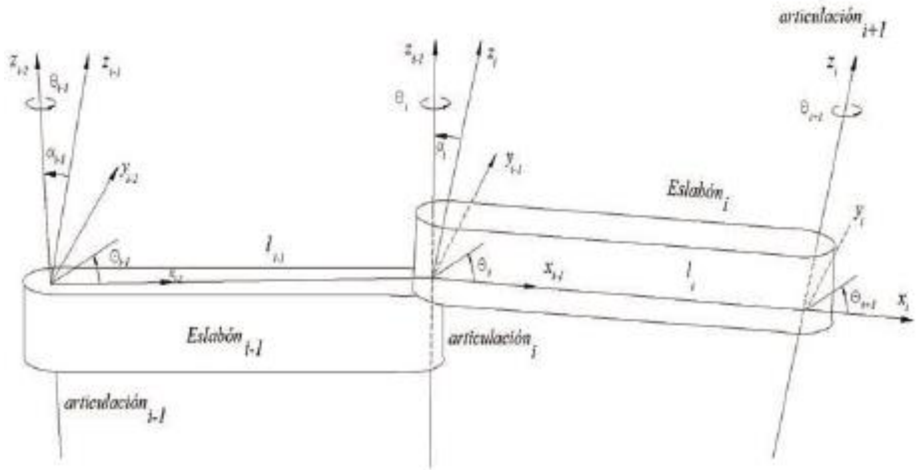


Figura 4.5 Convenci´on Denavit-Hartenberg para un robot manipulador.

la distancia del origen del sistema de referencia $i-1$ a la intersecci´on del eje x_i con el eje z_{i-1}. Su medici´on se realiza a lo largo del eje z_{i-1}, como se indica en la figura 4.6. Adicionalmente a las variables articulares θ_i y d_i, hay 2 par´ametros constantes que describen caracter´ısticas espec´ıficas del eslab´on i-´esimo. Esos par´ametros son: el par´ametro l_i se define como la distancia a lo largo del eje x_i desde el origen del sistema de referencia coordenado $i-1$ hasta la intersecci´on del eje z_{i-1} con el eje x_i. El otro par´ametro es el ´angulo entre los ejes z_i y z_{i-1} se denota por α_i, su medici´on es respecto a un plano normal a x_i. Una medici´on de a´ngulo positivo para α_i se toma en direcci´on del eje z_{i-1} hacia z_i. Por ejemplo, para un robot con 6 articulaciones rotacionales se requieren de 24 elementos para describir completamente su modelo cinem´atico $(l_i, \alpha_i, \beta_i, \theta_i)$.

Selecci´on de sistemas de referencia

En la metodolog´ıa Denavit-Hartenberg, primero se describir´a la convenci´on para asignar los sistemas de referencia cartesianos asociados a los eslabones del robot. En este punto es necesario aclarar que en la literatura de rob´otica, la convenci´on Denavit-Hartenberg no es única, depende de la selecci´on de los sistemas de referencia cartesianos en las articulaciones y eslabones, as´ı como en sus eslabones adyacentes.

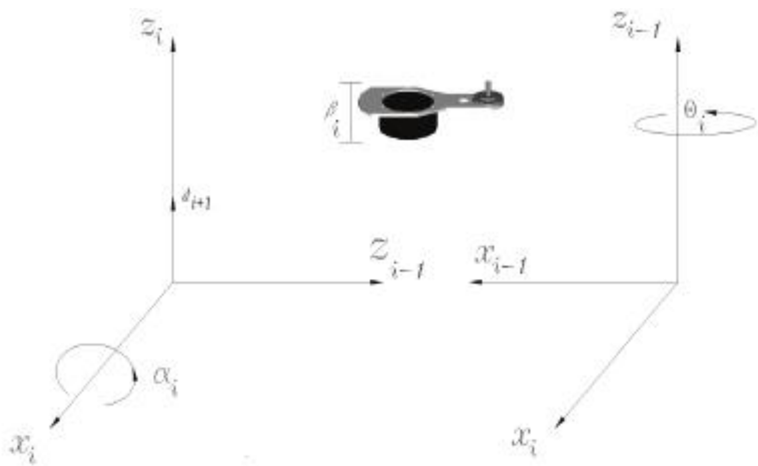


Figura 4.6 Convención para medir θ_i y α_i.

La cinemática directa del robot proporciona las coordenadas cartesianas del extremo final del robot relativo a un sistema de referencia cartesiano fijo $\Sigma_0(x_0, y_0, z_0)$; en la figura 4.5 se muestra la asignación de sistemas de referencia para las articulaciones $i-1$-ésima, i-ésima e $i+1$ de un robot manipulador.

En general se tiene el siguiente procedimiento:

- El eje z_i se asigna rígidamente a la articulación $i+1$. Es decir, z_0 es el eje de la articulación 1, z_1 es el eje de la articulación 2, y así sucesivamente.
- Localizar el origen $\mathbf{o}_i$ del sistema de referencia $\Sigma_i(x_i, y_i, z_i)$ en la intersección del eje z_i con la normal común a los ejes z_{i-1} y z_i.
- Seleccionar el eje x_{i-1} sobre la normal que une los ejes z_{i-1} y z_i en dirección de la articulación $i-1$ hacia la articulación i.
- Definir el ángulo de torsión α_i, este es el ángulo entre los ejes z_i y z_{i-1} y se mide con valor positivo en el sentido de las manecillas del reloj sobre el eje x_i.
- Seleccionar el eje y_i por la regla de la mano derecha.

La convención Denavit-Hartenberg proporciona una representación no única para los siguientes casos:

- Para el sistema de referencia $\Sigma_0(x_0, y_0, z_0)$ sólo la dirección del eje z_0 es especificada, entonces su origen $\mathbf{o}_0$ y el eje x_0 pueden ser seleccionados de manera arbitraria.

- Para el sistema de referencia $\Sigma_n(x_n, y_n, z_n)$ no existe la articulación $n+1$, entonces el eje z_n no está completamente definido, mientras que el eje x_n es normal al eje z_{n-1}. Típicamente la n-ésima articulación es rotatoria, por lo tanto z_n se alinea en la dirección de z_{n-1}.

- Cuando dos ejes consecutivos z_i y z_{i-1} son paralelos entre sí, la normal común entre ellos no es única.

- Cuando dos ejes consecutivos z_i y z_{i-1} se interceptan, la dirección del eje x_i es arbitraria.

- Cuando la articulación i-ésima es lineal o prismática, entonces la dirección de z_i es arbitraria.

De acuerdo con esta convención previamente descrita, a continuación se resumen los parámetros del i-ésimo eslabón:

1. l_i es la longitud del i-ésimo eslabón, es la distancia del eje z_{i-1} hacia el eje z_i medida sobre el eje x_{i-1}.

2. α_i es el ángulo de torsión, el cual representa el ángulo entre los ejes z_{i-1} a z_i medido en el sentido de las manecillas del reloj sobre el eje x_i.

3. d_i se emplea en articulaciones lineales o prismáticas y representa el desplazamiento lineal. Cuando la articulación es rotacional, entonces representa el offset o espesor del servomotor (la distancia de x_{i-1} a x_i medido sobre el eje z_{i-1}), se denota por β_i.

4. θ_i es el desplazamiento rotacional de x_{i-1} a x_i medido alrededor del eje z_{i-1}. El signo positivo de θ_i es el sentido contrario a las manecillas del reloj.

Obsérvese que l_i y β_i siempre serán positivos puesto que corresponden a longitudes, mientras que α_i, d_i, θ_i representan cantidades con signo.

Algoritmo Denavit-Hartenberg

A continuaci´on se describe el procedimiento para encontrar la cinem´atica directa a trav´es de la convenci´on Denavit-Hartenberg.

1. Localizar la direcci´on de los ejes $z_0, z_1, \cdots, z_{n-1}$.

2. Establecer el sistema de referencia cartesiano fijo $\Sigma_0(x_0, y_0, z_0)$ cuyo origen es colocado sobre el sistema de referencia en la base del robot. Los ejes x_0, y_0 son determinados de acuerdo con la regla de la mano derecha.

Una vez que el sistema de referencia $\Sigma_0(x_0, y_0, z_0)$ ha sido establecido, se inicia un proceso iterativo en el cual se define el sistema de referencia $\Sigma_i(x_i, y_i, z_i)$ usando el sistema de referencia $\Sigma_{i-1}(x_{i-1}, y_{i-1}, z_{i-1})$, iniciando con el sistema de referencia $\Sigma_1(x_1, y_1, z_1)$. En la figura 4.5 se muestra el procedimiento.

Llevar a cabo los pasos 3 al 5 para la articulaciones $i = 1, \cdots, n-1$.

3. Localizar el origen $\mathbf{o}_i$ en la intersecci´on de la normal com´un que une al eje z_i con el eje z_{i-1}.

Si el eje z_i intercepta al eje z_{i-1} colocar $\mathbf{o}_i$ en la intercepci´on.

Para el caso en que los ejes z_i y z_{i-1} son paralelos:

- Si la articulaci´on i-´esima es rotacional, colocar el origen $\mathbf{o}_i$ sobre la articulaci´on i-´esima, tal que $d_i = 0$.
- Si la articulaci´on i-´esima es prism´atica, colocar el origen $\mathbf{o}_i$ en un punto l´ımite f´ısico de la articulaci´on i-´esima, por ejemplo en un punto extremo.

4. Seleccionar el eje x_i a lo largo de la normal com´un que une a los ejes z_{i-1} y z_i, en direcci´on de la articulaci´on $i-1$ hacia la articulaci´on i.

5. Determinar y_i por la regla de la mano derecha.

6. Establecer el sistema de referencia del extremo final $\Sigma_n(x_n, y_n, z_n)$.

- Si la articulaci´on n-´esima es rotatoria, entonces alinear el eje z_n con el eje z_{n-1}

- Si la articulación n-ésima es prismática, entonces seleccionar el eje z_n de forma arbitraria. El eje x_n debe cumplir el paso 4.

7. Establecer la tabla 4.2 de parámetros de eslabones.

8. Obtener las matrices de transformaciones homogéneas (4.7):

$$H_{i-1} \text{ para } i = 1, 2, \cdots, n-1.$$

Tabla 4.2 Parámetros Denavit-Hartenberg

Características de eslabones	
l_i	Longitud del eslabón i-ésimo.
d_i	Articulaciones lineales o prismáticas. También representa el espesor del servomotor (β_i).
α_i	Ángulo entre los ejes z_{i-1} y z_i medido con respecto al eje x_i.
θ_i	articulaciones rotacionales; representa el ángulo entre los ejes x_{i-1} y x_i medido alrededor del eje z_{i-1}.

En la representación Denavit-Hartenberg cada transformación homogénea H_{i-1} se representa por el producto de cuatro transformaciones básicas:

$$
\begin{aligned}
H_{i-1}^{i} &= HR_{z_{i-1}}(\theta_i)\, HT_{z_{i-1}}(d_i(\beta_i))\, HT_{x_{i-1}}(l_i)\, HR_{x_{i-1}}(\alpha_i) \qquad (4.7)\\
&= \begin{bmatrix} \cos(\theta_i) & -\operatorname{sen}(\theta_i) & 0 & 0 \\ \operatorname{sen}(\theta_i) & \cos(\theta_i) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i(\beta_i) \\ 0 & 0 & 0 & 1 \end{bmatrix}
\begin{bmatrix} 1 & 0 & 0 & l_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\alpha_i) & -\operatorname{sen}(\alpha_i) & 0 \\ 0 & \operatorname{sen}(\alpha_i) & \cos(\alpha_i) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\\
&= \begin{bmatrix} \cos(\theta_i) & -\operatorname{sen}(\theta_i)\cos(\alpha_i) & \operatorname{sen}(\theta_i)\operatorname{sen}(\alpha_i) & l_i\cos(\theta_i) \\ \operatorname{sen}(\theta_i) & \cos(\theta_i)\cos(\alpha_i) & -\cos(\theta_i)\operatorname{sen}(\alpha_i) & l_i\operatorname{sen}(\theta_i) \\ 0 & \operatorname{sen}(\alpha_i) & \cos(\alpha_i) & d_i(\beta_i) \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned}
$$

La transformación homogénea total se obtiene como $H^n = H_0 H_1 \cdots H_{n-2} H_{n-1}$.

La cinemática directa es la forma general de transformaciones homogéneas que concatena los sistemas de referencia cartesianos asociados a los eslabones del robot, todos relativos al sistema de referencia fijo Σ_0.

4.6 Resumen

Cinemática directa relaciona las coordenadas articulares y propiedades geométricas del sistema mecánico con las coordenadas cartesianas del robot y la orientación de la herramienta colocada en el extremo final. En este capítulo se ha presentado los conceptos de cinemática inversa, cinemática diferencial y la importancia que presenta el jacobiano del robot en control cartesiano y en el tema de singularidades. Para las finalidades de la presente obra, cuando se relaciona las coordenadas articulares con las coordenadas cartesianas sin tomar en cuenta la orientación de la herramienta de trabajo, se denomina **cinemática directa cartesiana**, la cual será la base de análisis de las principales configuraciones de robots industriales.

El procedimiento Denavit-Hartenberg permite obtener el modelo de cinemática directa de robots manipuladores con eslabones en serie a través de la siguiente tabla de parámetros 3.1:

Tabla 4.3 Parámetros DH

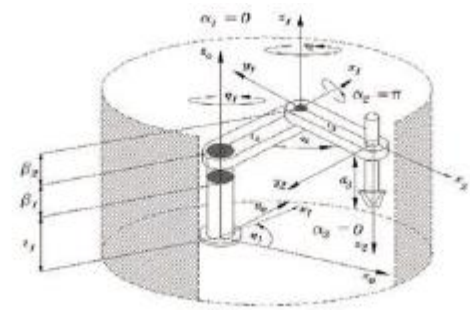
Eslabón$_i$	l_i	α_i	d_i	θ_i

La formulación Denavit-Hartenberg queda expresada en términos de matrices homogéneas con el estricto orden de transformaciones de traslación y rotación:

$$
\begin{aligned}
H_{i-1}^{i} &= H_{R_{z_i}}(\theta_{i+1}) H_{T_{z_i}}(d_{i+1}) H_{T_{x_i}}(l_{i+1}) H_{R_{x_i}}(\alpha_{i+1}) \\
&= \begin{bmatrix} \cos(\theta_i) & -\,\mathrm{sen}(\theta_i)\cos(\alpha_i) & \mathrm{sen}(\theta_i)\,\mathrm{sen}(\alpha_i) & l_i\cos(\theta_i) \\ \mathrm{sen}(\theta_i) & \cos(\theta_i)\cos(\alpha_i) & -\cos(\theta_i)\,\mathrm{sen}(\alpha_i) & l_i\,\mathrm{sen}(\theta_i) \\ 0 & \mathrm{sen}(\alpha_i) & \cos(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned}
$$

Cap´ıtulo 5

Cinem´atica directa cartesiana



Objetivos

Presentar el an´alisis de cinem´atica directa cartesiana, cinem´atica inversa y matriz jacobiano de las principales configuraciones de robots industriales considerando los par´ametros geom´etricos y desarrollar librer´ıas en lenguaje MATLAB (toolbox) que permitan realizar aplicaciones en el ´area de cinem´atica directa de robots manipuladores.

Objetivos particulares:

- Modelo de cinem´atica directa.
- Modelo de cinem´atica inversa.
- Jacobiano.
- Librer´ıas de an´alisis y diseño de cinem´atica de robots industriales.

5.1 Introducción

Esta sección analiza la cinemática directa cartesiana de las configuraciones más importantes en robótica industrial tales como el brazo robot (antropomórfico), configuración SCARA, esférico, cilíndrico y cartesiano. El análisis cinemático que se presenta es tomando en cuenta la relación que existe entre las coordenadas articulares con las coordenadas cartesianas del extremo final del robot (cinemática directa cartesiana), no se incluye la orientación de la herramienta de trabajo ubicada en el extremo final del robot.

Para el análisis cinemático cartesiano se utiliza la metodología Denavit-Hartenberg, y por lo tanto dicho modelo queda en función de transformaciones homogéneas.

El jacobiano analítico del robot se deduce como la derivada parcial de la cinemática directa cartesiana con respecto al vector de posición articular; el jacobiano analítico del robot proporciona información sobre el problema de singularidades que puede presentar el movimiento del robot en su espacio de trabajo. Esta información es muy importante en aplicaciones donde interviene la inversa de la matriz jacobiana, ya que una singularidad significa que el determinante es cero para un conjunto específico de valores de las posiciones articulares.

Debido a la importancia que tiene la cinemática inversa en la programación de tareas en espacio cartesiano y la conversión al espacio articular, se presenta la solución geométrica en detalle para cada una de las configuraciones de robots industriales.

Un conjunto de librerías en código fuente de MATLAB se desarrollan y documentan de la matriz de transformación homogénea, cinemática directa cartesiana, jacobiano y su determinante, cinemática inversa, en variables simbólicas y aplicaciones numéricas de todas las configuraciones analizadas. El código fuente de los ejemplos, aplicaciones y librerías de cada robot manipulador se encuentra disponible en el sitio WEB del libro:

http://virtual.alfaomega.com.mx

5.2 Brazo robot antropom´orfico

Dentro de la clasificaci´on de robots industriales, el **robot antropom´orfico** o brazo robot es la configuraci´on que m´as se utiliza debido a la destreza que presenta el movimiento del extremo final como una consecuencia de sus tres articulaciones rotacionales, lo cual lo hace ideal para un amplio espectro de aplicaciones tales como: industriales, quir´ofanos robotizados, fisioterapia, asistencia a personas con capacidades diferenciadas, maniobras submarinas y espaciales, etc. La configuraci´on antropom´orfica se divide en tres casos de estudio: p´endulo robot, robot planar de 2 grados de libertad y el brazo robot (movimiento tridimensional).

P´endulo robot

Como un caso particular del robot de 3 grados de libertad antropom´orfico, se encuentra el p´endulo robot de 1 grado de libertad formado por un servomotor que tiene acoplado mec´anicamente una barra met´alica de longitud l_1 y est´a sometido al fen´omeno de gravedad. El movimiento del p´endulo se encuentra en el plano vertical $x_0 - y_0$ como se indica en la figura 5.1.

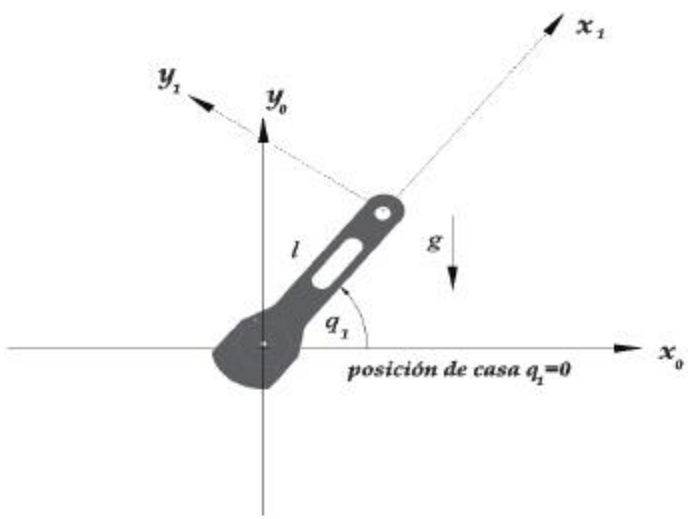


Figura 5.1 P´endulo robot.

Cinemática directa cartesiana del péndulo robot

El origen del sistema de referencia cartesiano $\Sigma_0(x_0, y_0, z_0)$ se coloca sobre la articulación del péndulo (en el respaldo del estator del servomotor), el eje z_0 se alinea con el eje de giro del servomotor el cual es perpendicular al plano $x_0 - y_0$. El servomotor del péndulo tiene un espesor de longitud β_1 el cual ya incluye el ancho de la barra metálica. El origen del sistema de referencia $\Sigma_1(x_1, y_1, z_1)$ se coloca en el extremo final de la barra y se mueve de manera conjunta al péndulo, el origen Σ_1 tiene las siguientes coordenadas con respecto al origen de $\Sigma_0(x_0, y_0, z_0)$: $[l_1 \cos(q_1), l_1 \operatorname{sen}(q_1), \beta_1]$. Debido al espesor β_1 el origen de Σ_1 mantiene una distancia β_1 sobre el eje z_0. Los ejes z_1 y z_0 son paralelos entre sí, por lo tanto el ángulo α_1 que existe entre ellos es $\alpha_1 = 0$.

El plano $x_1 - y_1$ tiene una rotación de q_1 grados con respecto al plano $x_0 - y_0$ del sistema fijo $\Sigma_0(x_0, y_0, z_0)$, y la matriz $R_{z_0}(q_1)$ determina dicha rotación:

$$R_1^0 = R_{z_0}(q_1) = \begin{bmatrix} \cos(q_1) & -\operatorname{sen}(q_1) & 0 \\ \operatorname{sen}(q_1) & \cos(q_1) & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Los parámetros Denavit-Hartenberg correspondientes al péndulo-robot se encuentran especificados en la tabla 5.1.

Tabla 5.1 DH del péndulo

Eslabón	l	α	d	θ
1	l_1	0	β_1	q_1

La matriz de transformación homogénea H_0^1 se obtiene de la ecuación (4.7), que para el caso del péndulo-robot adquiere la siguiente forma:

$$H_0^1 = H_R(z_0, q_1) H_T(z_0, \beta_1) H_T(x_0, l_1) H_R(x_0, 0) \qquad (5.1)$$

$$= \begin{bmatrix} \cos(q_1) & -\operatorname{sen}(q_1) & 0 & l_1 \cos(q_1) \\ \operatorname{sen}(q_1) & \cos(q_1) & 0 & l_1 \operatorname{sen}(q_1) \\ 0 & 0 & 1 & \beta_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \qquad (5.2)$$

Las coordenadas cartesianas del extremo final del péndulo robot (cinemática directa cartesiana), es decir el origen del sistema $\Sigma_1(x_1, y_1, z_1)$ con respecto al sistema fijo $\Sigma_0(x_0, y_0, z_0)$ se encuentra determinado por:

$$\begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} = \begin{bmatrix} l_1 \cos(q_1) \\ l_1 \operatorname{sen}(q_1) \\ \beta_1 \end{bmatrix}. \tag{5.3}$$

Las coordenadas cartesianas del extremo final del péndulo dependen de la ubicación de la posición de casa (home position), la cual es el punto de reposo u origen del péndulo que corresponde a $q_1 = 0$. Como se puede apreciar en la figura 5.1, la posición de casa está colocada sobre el eje x_{0_+}. La matriz de rotación $R_{z_0}(q_1)$ que determina la rotación entre los sistemas de referencia fijo $\Sigma_0(x_0, y_0, z_0)$ y $\Sigma_1(x_1, y_1, z_1)$ fue obtenida con respecto al primer cuadrante, es decir en el plano $x_{0_+} - y_{0_+}$.

Otra posibilidad para ubicar la posición de casa del péndulo es colocar el origen $q_1 = 0$ sobre el lado negativo del eje y_0 como se muestra en la figura 5.2.

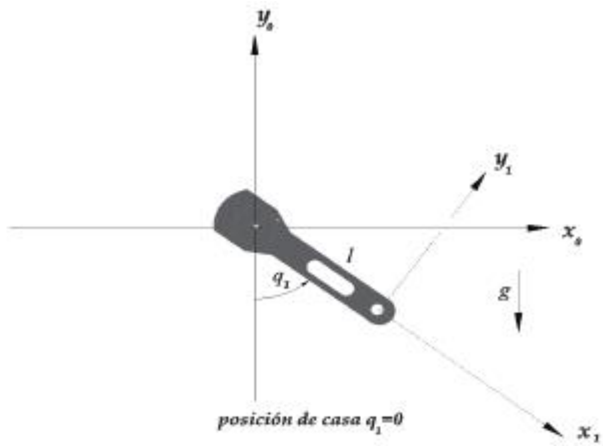


Figura 5.2 Cambio de posición de casa del péndulo sobre el eje y_{0_-}.

En este caso, para poder obtener la matriz de rotación resultante del péndulo en la nueva ubicación de la posición de casa, la matriz $R_{z_0}(q_1)$ debe ser precedida por una rotaci´on de un angulo igual a $-\frac{\pi}{2}$ alrededor del eje z_0, es decir:

$$
\begin{aligned}
R_0^1 &= R_{z_0}\left(-\frac{\pi}{2}\right)\begin{bmatrix} \cos(q_1) & -\,\mathrm{sen}(q_1) & 0 \\ \mathrm{sen}(q_1) & \cos(q_1) & 0 \\ 0 & 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} \cos(q_1) & -\,\mathrm{sen}(q_1) & 0 \\ \mathrm{sen}(q_1) & \cos(q_1) & 0 \\ 0 & 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} \mathrm{sen}(q_1) & \cos(q_1) & 0 \\ -\cos(q_1) & \mathrm{sen}(q_1) & 0 \\ 0 & 0 & 1 \end{bmatrix}
\end{aligned}
$$

La matriz de transformaci´on homog´enea para el p´endulo-robot (IV cuadrante) est´a determinada como:

$$
H_0^1 = \begin{bmatrix} R_{z_0}\left(-\frac{\pi}{2}\right)\begin{bmatrix} \cos(q_1) & -\,\mathrm{sen}(q_1) & 0 \\ \mathrm{sen}(q_1) & \cos(q_1) & 0 \\ 0 & 0 & 1 \end{bmatrix} & R_{z_0}\left(-\frac{\pi}{2}\right)\begin{bmatrix} l\,\mathrm{sen}(q) \end{bmatrix} \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{5.4}
$$

$$
= \begin{bmatrix} \mathrm{sen}(q_1) & \cos(q_1) & 0 & l_1\,\mathrm{sen}(q_1) \\ -\cos(q_1) & \mathrm{sen}(q_1) & 0 & -l_1\cos(q_1) \\ 0 & 0 & 1 & \beta_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{5.5}
$$

Las coordenadas del extremo final del p´endulo respecto a la posici´on de casa colocada sobre el eje y_{0-} adquiere la siguiente forma:

$$
\begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} = \begin{bmatrix} l_1\,\mathrm{sen}(q_1) \\ -l_1\cos(q_1) \\ \beta_1 \end{bmatrix} \tag{5.6}
$$

Jacobiano del péndulo

Debido a que el péndulo robot es un sistema escalar, no tiene matriz jacobiana. Sin embargo, se puede realizar la siguiente interpretación partiendo de la cinemática diferencial:

$$\begin{aligned}
\mathbf{v} &= \frac{d}{dt}\begin{bmatrix} l_1\cos(q_1) \\ l_1\,\mathrm{sen}(q_1)\end{bmatrix} = \frac{\partial \mathbf{f}_R(q_1)}{\partial q_1}\dot{q}_1 = \begin{bmatrix} -l_1\,\mathrm{sen}(q_1) \\ l_1\cos(q_1)\end{bmatrix}\dot{q}_1 \\
&= \underbrace{\begin{bmatrix} -l_1\,\mathrm{sen}(q_1) & 0 \\ 0 & l_1\cos(q_1)\end{bmatrix}}_{J(q_1)} \underbrace{\begin{bmatrix}\dot{q}_1 \\ \dot{q}_1\end{bmatrix}}_{\dot{\mathbf{q}}} = J(q_1)\dot{\mathbf{q}}
\end{aligned} \tag{5.7}$$

el jacobiano $J(q_1)$ se puede pensar como está expresado en la ecuación (5.7). Nótese que existe una singularidad cuando la variable articular $q_1 = 0, \pm n\pi$, ya que su determinante está dado como: $\det[J(q_1)] = -l_1^2\,\mathrm{sen}(q_1)\cos(q_1)$. También existen singularidades en $q_1 = \pm n\frac{\pi}{2}$, esto último puede verificarse en la cinemática inversa (5.8), si $q_1 = \pm n\frac{\pi}{2} \Rightarrow x_0 = 0$, entonces se indefine la función arco-tangente (por ejemplo división entre cero).

Cinemática inversa del péndulo

La cinemática inversa se obtiene despejando la variable q_1 de la ecuación (5.3), obteniendo:

$$q_1 = \operatorname{atan}\left(\frac{y_0}{x_0}\right). \tag{5.8}$$

Función transformación homogénea del péndulo H_0^1

La función transformación homogénea del péndulo H_0^1 se encuentra expresada como:

$$H_0^1 = \mathrm{H\_pendulo}()$$

El código MATLAB de la función transformación homogénea del péndulo se describe en el cuadro 5.1. En la línea 5 despliega la tabla 5.1 de parámetros DH. La línea 7 contiene el cálculo de la matriz homogénea $H_0^1 = H_{R_{z_0}}(q_1)H_{T_{z_0}}(\beta_1)H_{T_{x_0}}(l_1)H_{R_{x_0}}(0)$

y en la l´ınea 8 se emplea la funci´on H DH(H10) para obtener la matriz de rotaci´on y las coordenadas cartesianas.

C´odigo Fuente 5.1 H_pendulo.m

```
%MATLAB Aplicado a Rob´otica y Mecatr´onica.
%Editorial Alfaomega, Fernando Reyes Cort´es.
%Cap´ıtulo 5 Cinem´atica directa cartesiana.
```

H_pendulo.m

```
1  function H=H_pendulo()
2      syms l1 q1 beta1 real
3      disp('Par´ametros Denavit-Hartenberg del p´endulo')
4      disp('[ l alpha d q]') dh=[l1, 0, beta1, q1];;
5      disp(dh)
6      %H_0^1 = H_{Rz0}(q1)H_{Tz0}(β1)H_{Tx0}(l1)H_{Rx0}(0)
7      H10=HRz(q1)*HTz(beta1)*HTx(l1)*HRx(0) ;
8      [R10, cinemat_pendulo, cero, c]=H_DH(H10);
9      H=[R10, cinemat_pendulo;;
10          cero, c];
11 end
```

Funci´on cinem´atica directa cartesiana del p´endulo

La funci´on de cinem´atica directa cartesiana del p´endulo tiene la siguiente sintaxis:

$$[x_0, y_0, z_0]=\text{cinematica\_pendulo}(\beta_1, l_1, q_1)$$

donde β_1, l_1, q_1 representan el ancho del servomotor y espesor de la barra met´alica, longitud de la barra met´alica y posici´on articular del p´endulo, respectivamente. Esta funci´on retorna las coordenadas cartesianas del extremo final del p´endulo ubicadas en el sistema $\Sigma_0(x_0, y_0, z_0)$.

El c´odigo de la funci´on de cinem´atica cartesiana del p´endulo se encuentra en el cuadro 5.2; se contempla el caso simb´olico y num´erico.

C´odigo Fuente 5.2 cinematica_pendulo.m

%MATLAB Aplicado a Rob´otica y Mecatr´onica.
%Editorial Alfaomega, Fernando Reyes Cort´es.
%Cap´ıtulo 5 Cinem´atica directa cartesiana.

cinematica_pendulo.m

```
function [x0, y0, z0]=cinematica_pendulo(beta1,l1,q1)
    dato1=whos('beta1'); dato2=whos('l1'); dato3=whos('q1');
    v1=strcmp(dato1.class, 'sym'); v2=strcmp(dato2.class, 'sym');
    v3=strcmp(dato3.class, 'sym'); digits(3);
    if (v1 & v2 & v3) %caso simb´olico
        x0=l1*cos(q1);
        y0=l1*sin(q1);
        z0=beta1;
    end
    else %caso num´erico
    x0=double(simplify(vpa(l1*cos(q1),3)));
    y0= double(simplify(vpa(l1*sin(q1),3)));
    z0=double(simplify(vpa(beta1,3)));
end
```

Funci´on cinem´atica inversa del p´endulo

La funci´on de cinem´atica inversa del p´endulo se encuentra expresada como:

$$q_1\text{=cinv\_pendulo}(x_0, y_0)$$

donde x_0, y_0 representan las coordenadas del extremo final del p´endulo con respecto al sistema $\Sigma_0(x_0, y_0, z_0)$. Esta funci´on retorna la posici´on articular q_1 del p´endulo; observe que el modelo de cinem´atica inversa no depende del grosor del servomotor (β_1).

El código de la función de cinemática inversa se encuentra en el cuadro 5.3.

Código Fuente 5.3 cinv_pendulo.m

%MATLAB Aplicado a Robótica y Mecatrónica.
%Editorial Alfaomega, Fernando Reyes Cortés.
%Capítulo 5 Cinemática directa cartesiana.

cinv_pendulo.m

```
function q1=cinv_pendulo(x0,y0)
    %cinemática inversa del péndulo
    q1=atan(y0/x0);
end
```

♣ Ejemplo 5.1

Desarrollar un programa para MATLAB que presente de manera simbólica los parámetros DH, cinemática cartesiana y el jacobiano del péndulo. Además, contemple el análisis donde la posición de casa se traslada del primer al cuarto cuadrante.

Solución

En el cuadro 5.4 se presenta el programa que permite desplegar en forma símbolica los parámetros DH, cinemática cartesiana y el jacobiano del péndulo.

La línea 7 obtiene la transformación homogénea del péndulo H_0^1 empleando la función H_0^1=H_pendulo(). La línea 9 deduce la matriz de rotación R_1^0 y la cinemática cartesiana $\mathbf{f}_R(q_1)$ a través de la función H_DH(H_0^1).

En la línea 11 se realiza una rotación $H_{R_{z_0}}(-\frac{\pi}{2})$ (-90 grados alrededor del eje z_0) para trasladar la posición de casa al cuarto cuadrante. De la línea 13 a la 20 se presenta un ejemplo numérico, para trasladar las coordenadas del extremo final en la rotación del primer al cuarto cuadrante.

El programa 5.4 contiene la descripción completa del modelo cinemático directo del péndulo robot, el cual emplea las funciones de transformación homogénea H_0^1 incluye la matriz de rotación R_1^0 que relaciona la rotación del extremo final del eslabón con respecto a la base del péndulo, así como la cinemática directa cartesiana en variables simbólicas como aplicación numérica.

Código Fuente 5.4 cap5_pendulo.m

```
%MATLAB Aplicado a Robótica y Mecatrónica.
%Editorial Alfaomega, Fernando Reyes Cortés.
%Capítulo 5 Cinemática directa cartesiana.
```

cap5_pendulo.m

```
clc;
clear all;
close all;
format short
syms beta1 l1 q1 real
%H_0^1 = H_Rz0(q1) H_Tz0(β1) H_Tx0(l1) H_Rx0(0)
H10=H_pendulo() %H_0^1
%obtiene la matriz de rotación R_1^0 y cinemática directa f_r(q)
[R10, frq_pendulo, cero, c]=H_DH(H10) %R_1^0, f_R(β1, l1, q1)
jac_pendulo=jacobian(frq_pendulo, q1)
H10a=HRz(-pi/2)*H10 %H_0a^1 = H_Rz0(-π/2) H_0^1
%ejemplo numérico
q1=55*pi/180; %posición angular del péndulo
beta1=0.1; %ancho del servomotor más espesor de la barra
l1=0.45; beta1=0.1; %longitud del péndulo y espesor del servomotor
%cinemática cartesiana cuadrante I: f_RI(β1, l1, q1)
[x0,y0,z0]=cinematica_pendulo(beta1,l1,q1)
%cinemática directa en el cuadrante IV
%f_RIV(β1, l1, q1) = R_z(-π/2) f_RI(β1, l1, q1)
Rz(-pi/2)*cinematica_pendulo(beta1,l1,q1)
```

Robot planar vertical de dos grados de libertad

La figura 5.3 muestra un robot de dos grados de libertad con articulaciones rotacionales que se mueve en el plano vertical $x_0 - y_0$. El sistema de referencia fijo $\Sigma_0(x_0, y_0, z_0)$ se coloca en el respaldo del servomotor del hombro, de tal forma que el eje z_0 coincida con el eje de rotaci´on (perpendicular al plano de la hoja). El ancho de cada servomotor y espesor de la barra met´alica est´an determinados por β_1 y β_2. Los ejes x_0, y_0 se seleccionan con la regla de la mano derecha.

Cinem´atica directa cartesiana del brazo robot de 2 gdl

El sistema de referencia $\Sigma_1(x_1, y_1, z_1)$ se coloca en el extremo final del primer eslab´on, el eje z_1 se coloca paralelo al eje z_0 ($\alpha_1 = 0$). El origen del sistema $\Sigma_1(x_1, y_1, z_1)$ se ubica en la intersecci´on del eje x_0 con el eje z_1 (cuando $q_1 = 0$) y est´a a una distancia β_1 sobre el eje z_0. El origen del sistema de referencia $\Sigma_2(x_2, y_2, z_2)$ se coloca en el extremo final del robot. El ancho del servomotor del codo y el espesor de la segunda barra miden β_2. Cuando $q_2 = 0$ el origen de Σ_2 se encuentra a una distancia l_2 sobre el eje x_1 y a una distancia β_2 sobre el eje z_1. Los ejes x_i, y_i, con $i = 1, 2$ se seleccionan con la regla de la mano derecha. Los ejes z_1 y z_2 son paralelos entre s´ı ($\alpha_2 = 0$).

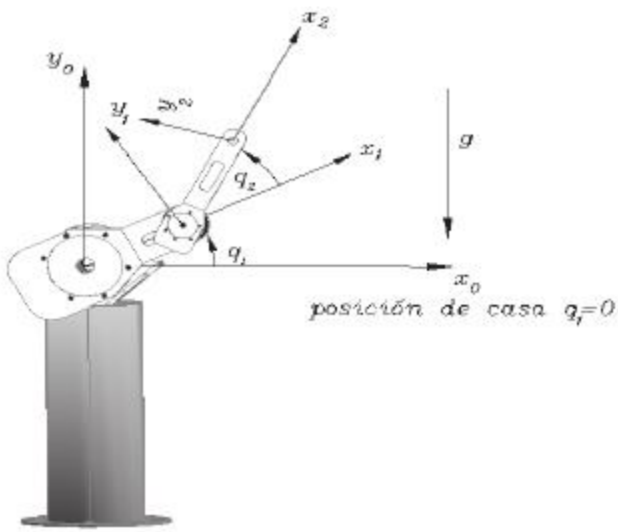


Figura 5.3 Robot planar vertical de dos gdl.

En la tabla 5.2 se muestran los par´ametros de los eslabones para la convenci´on

Denavit-Hartenberg de un robot planar de dos grados de libertad.

Tabla 5.2 DH del robot de 2 gdl

Eslabón	l_i	α_i	d_i	θ_i
1	l_1	0	β_1	q_1
2	l_2	0	β_2	q_2

Generalmente la posición de casa para un robot de dos grados de libertad se selecciona sobre el eje x_{0+}, midiendo el ángulo q_1 en sentido contrario a las manecillas del reloj, es decir del eje x_{0+} hacia el eje y_{0+}, con esta consideración se obtienen las siguientes matrices de transformación homogénea:

$$H_0^1 = H_{R_{z_0}}(q_1)\, H_{T_{z_0}}(\beta_1)\, H_{T_{x_0}}(l_1)\, H_{R_{x_0}}(0) \tag{5.9}$$

$$= \begin{bmatrix} \cos(q_1) & -\operatorname{sen}(q_1) & 0 & l_1\cos(q_1) \\ \operatorname{sen}(q_1) & \cos(q_1) & 0 & l_1\operatorname{sen}(q_1) \\ 0 & 0 & 1 & \beta_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.10}$$

$$H_1^2 = H_{R_{z_1}}(q_2)\, H_{T_{z_1}}(\beta_2)\, H_{T_{x_1}}(l_2)\, H_{R_{x_1}}(0) \tag{5.11}$$

$$= \begin{bmatrix} \cos(q_2) & -\operatorname{sen}(q_2) & 0 & l_2\cos(q_2) \\ \operatorname{sen}(q_2) & \cos(q_2) & 0 & l_2\operatorname{sen}(q_2) \\ 0 & 0 & 1 & \beta_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.12}$$

$$H_0^2 = H_0^1 H_1^2$$

$$= \begin{bmatrix} \cos(q_1+q_2) & -\operatorname{sen}(q_1+q_2) & 0 & l_1\cos(q_1)+l_2\cos(q_1+q_2) \\ \operatorname{sen}(q_1+q_2) & \cos(q_1+q_2) & 0 & l_1\operatorname{sen}(q_1)+l_2\operatorname{sen}(q_1+q_2) \\ 0 & 0 & 1 & \beta_1+\beta_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{5.13}$$

La cinemática directa de las coordenadas cartesianas del extremo final del robot (sin tomar en cuenta la orientación de la herramienta de trabajo) está dada por:

$$\begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} = \mathbf{f}_R(\mathbf{q}) = \begin{bmatrix} l_1\cos(q_1)+l_2\cos(q_1+q_2) \\ l_1\operatorname{sen}(q_1)+l_2\operatorname{sen}(q_1+q_2) \\ \beta_1+\beta_2 \end{bmatrix}. \tag{5.14}$$

Obs´ervese que la coordenada z_0 incluye el espesor de los dos servomotores con sus respectivos espesores de las barras met´alicas, de tal forma que el origen del sistema $\Sigma_2(x_2, y_2, z_2)$ se encuentra a una distancia $\beta_1 + \beta_2$ sobre el eje z_0 del sistema fijo $\Sigma_0(x_0, y_0, z_0)$.

La posici´on de casa de un robot manipulador planar de dos grados de libertad tambi´en puede ser ubicada sobre el eje y_{0-}. Esto equivale a hacer una rotaci´on alrededor del eje z_0 por -90 grados para la articulaci´on del hombro, es decir: $H_{R_{z_0}}(-\frac{\pi}{2})H_0$ y tambi´en para articulaci´on del codo $H_{R_{z_1}}(-\frac{\pi}{2})H_1$. El angulo q_1 se mide del eje y_{0-} hacia el eje x_{0+} (valor positivo en sentido contrario a las manecillas del reloj) como se muestra en la figura 5.4.

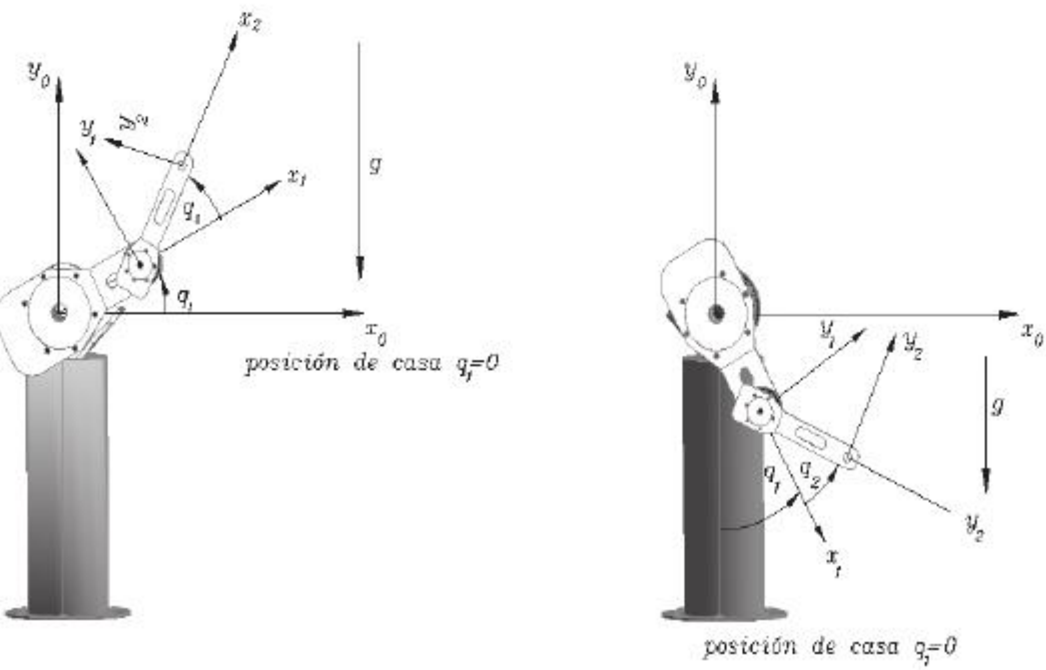


Figura 5.4 Rotaci´on de la posici´on de casa por -90 grados alrededor del eje z_0.

$$
\begin{aligned}
H_{0_{y_{0-}}} &= H_{R_{z_0}}\left(-\frac{\pi}{2}\right) H_0 \\
&= \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(q_1) & -\operatorname{sen}(q_1) & 0 & l_1\cos(q_1) \\ \operatorname{sen}(q_1) & \cos(q_1) & 0 & l_1\operatorname{sen}(q_1) \\ 0 & 0 & 1 & \beta_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned}
$$

$$
= \begin{bmatrix} \operatorname{sen}(q_1) & \cos(q_1) & 0 & l_1 \operatorname{sen}(q_1) \\ -\cos(q_1) & \operatorname{sen}(q_1) & 0 & -l_1 \cos(q_1) \\ 0 & 0 & 1 & \beta_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
\begin{aligned}
H_{1_{y_1-}}^{2} &= H_{R_{z_1}}\left(-\tfrac{\pi}{2}\right) H_1^2 \\
&= \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(q_2) & -\operatorname{sen}(q_2) & 0 & l_2 \cos(q_2) \\ \operatorname{sen}(q_2) & \cos(q_2) & 0 & l_2 \operatorname{sen}(q_2) \\ 0 & 0 & 1 & \beta_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
&= \begin{bmatrix} \operatorname{sen}(q_2) & \cos(q_2) & 0 & l_2 \operatorname{sen}(q_2) \\ -\cos(q_2) & \operatorname{sen}(q_2) & 0 & -l_2 \cos(q_2) \\ 0 & 0 & 1 & \beta_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
H_{0_{y_0-}}^{2} &= H_{0_{y_0-}}^{1} H_{1_{y_1-}}^{2} \\
&= \begin{bmatrix} \operatorname{sen}(q_1+q_2) & \cos(q_1+q_2) & 0 & l_1 \operatorname{sen}(q_1) + l_2 \operatorname{sen}(q_1+q_2) \\ -\cos(q_1+q_2) & \operatorname{sen}(q_1+q_2) & 0 & -l_1 \cos(q_1) - l_2 \cos(q_1+q_2) \\ 0 & 0 & 1 & \beta_1+\beta_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned}
$$

El mismo resultado se puede obtener premultiplicando la matriz homogénea de rotación $H_{R_{z_0}}\left(-\frac{\pi}{2}\right)$ a la matriz de transformación homogénea H_0^2 (5.13). Es decir, $H_{0_{y_0-}}^{2} = H_{R_{z_0}}\left(-\frac{\pi}{2}\right) H_0^2$.

Por lo que la cinemática directa de coordenadas cartesianas para un robot planar de dos grados de libertad cuya posición inicial es sobre el eje y_0 negativo:

$$
\begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} = \begin{bmatrix} l_1 \operatorname{sen}(q_1) + l_2 \operatorname{sen}(q_1+q_2) \\ -l_1 \cos(q_1) - l_2 \cos(q_1+q_2) \\ \beta_1 + \beta_2 \end{bmatrix} \tag{5.15}
$$

Debido a que la rotación se realiza alrededor de los ejes z_1 y z_2, ambos paralelos al eje z_0, no existe ninguna modificación de coordenadas de los parámetros geométricos β_1 y β_2. Por lo que el origen del sistema $\Sigma_2(x_2, y_2, z_2)$ siempre tendrá una distancia $\beta_1 + \beta_2$ sobre el eje z_0.

Jacobiano del brazo robot de 2 gdl

El jacobiano anal´ıtico del robot de 2 gdl se obtiene como:

$$J(\mathbf{q}) = \frac{\partial \mathbf{f}_R(\beta_1, \beta_2, l_1, l_2, \mathbf{q})}{\partial \mathbf{q}}$$

tomando en cuenta que los par´ametros geom´etricos $\beta_1, \beta_2, l_1, l_2$ son constantes, y considerando la posici´on de casa sobre el eje x_0 (primer cuadrante):

$$J(\mathbf{q}) = \begin{matrix} -l_1 \operatorname{sen}(q_1) - l_2 \operatorname{sen}(q_1 + q_2) & -l_2 \operatorname{sen}(q_1 + q_2) \end{matrix} \in IR^{2\times 2} \qquad (5.16)$$

El determinante del jacobiano es $\det[J(\mathbf{q})] = l_1 l_2 \operatorname{sen}(q_2)$, el cual es cero para $q_2 = 0, \pm n\pi$ y q_1 cualquier valor. Por lo tanto, cuando la articulaci´on del codo tiene alguno de esos valores para q_2 el robot entra en una singularidad. Esto es importante en aplicaciones donde interviene $\dot{\mathbf{q}} = J(\mathbf{q})^{-1}\mathbf{v}$.

Cinem´atica inversa del brazo robot de 2 gdl

La cinem´atica inversa (sin tomar en cuenta la orientaci´on de la herramienta de trabajo) de un robot manipulador de dos grados de libertad se obtiene por procedimiento geom´etrico (ver figura 5.5).

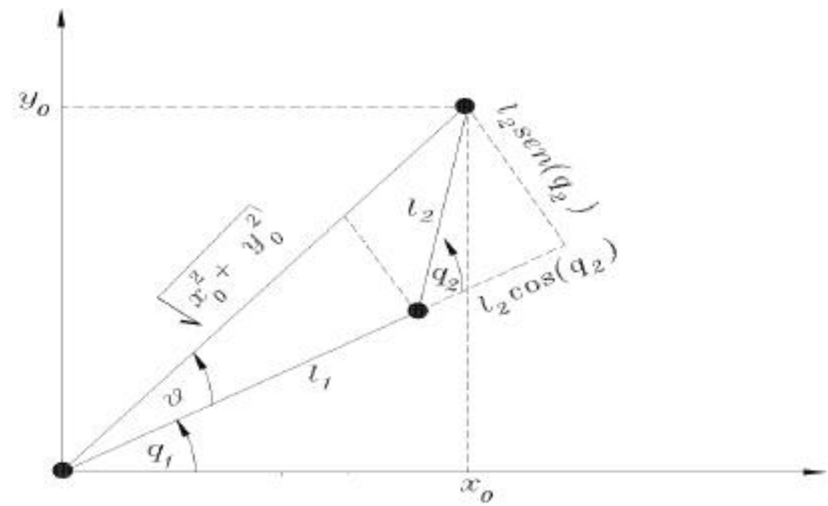


Figura 5.5 Cinem´atica inversa de un robot planar de dos grados de libertad.

El ángulo ϑ que se encuentra dentro del triángulo formado por los lados adyacente $l_1 + l_2\cos(q_2)$, cateto opuesto $l_2\,\mathrm{sen}(q_2)$ y la hipotenusa $\sqrt{x_0^2 + y_0^2}$, del teorema de Pitágoras se obtiene la solución para la variable articular q_2:

$$
\begin{aligned}
x_0^2 + y_0^2 &= [l_1 + l_2\cos(q_2)]^2 + l_2^2\,\mathrm{sen}^2(q_2)\\
&= l_1^2 + l_2^2\underbrace{[\cos^2(q_2) + \mathrm{sen}^2(q_2)]}_{=1} + 2l_1l_2\cos(q_2)\\
&= l_1^2 + l_2^2 + 2l_1l_2\cos(q_2)\\
q_2 &= \mathrm{acos}\left(\frac{x_0^2 + y_0^2 - l_1^2 - l_2^2}{2l_1l_2}\right)
\end{aligned}
$$

Observe que el ángulo ϑ satisface:

$$
\vartheta = \mathrm{atan}\left[\frac{l_2\,\mathrm{sen}(q_2)}{l_1 + l_2\cos(q_2)}\right]
$$

Ahora, tomando los ángulos $\vartheta + q_1$ dentro del triángulo formado por los catetos adyacente x_0, opuesto y_0 y la hipotenusa $\sqrt{x_0^2 + y_0^2}$ se cumple la siguiente expresión:

$$
\vartheta + q_1 = \mathrm{atan}\left[\frac{y_0}{x_0}\right]
$$

Entonces, la variable articular q_1 adquiere la siguiente forma:

$$
\begin{aligned}
q_1 &= \mathrm{atan}\left[\frac{y_0}{x_0}\right] - \vartheta\\
&= \mathrm{atan}\left[\frac{y_0}{x_0}\right] - \mathrm{atan}\left[\frac{l_2\,\mathrm{sen}(q_2)}{l_1 + l_2\cos(q_2)}\right]
\end{aligned}
$$

Por lo tanto, la cinemática inversa de un robot de 2 gdl no depende de β_1 y β_2 debido a que no hay proyecciones de estos parámetros sobre el plano $x_0 - y_0$:

$$
q_2 = \mathrm{acos}\left(\frac{x_0^2 + y_0^2 - l_1^2 - l_2^2}{2l_1l_2}\right) \tag{5.17}
$$

$$
q_1 = \mathrm{atan}\left[\frac{y_0}{x_0}\right] - \mathrm{atan}\left[\frac{l_2\,\mathrm{sen}(q_2)}{l_1 + l_2\cos(q_2)}\right] \tag{5.18}
$$

Función transformación homogénea del robot 2 gdl H_0^2

La función transformación homogénea H_0^2 del robot planar vertical de 2 gdl se encuentra expresada como:

$$H_0^2 = \texttt{H\_r2gdl()}$$

esta función retorna H_0^2 la transformación homogénea del robot planar vertical de 2 gdl. El código MATLAB de la función transformación homogénea H_0^2 del robot de 2 gdl se encuentra en el cuadro 5.5.

Código Fuente 5.5 H_r2gdl.m

%MATLAB Aplicado a Robótica y Mecatrónica.
%Editorial Alfaomega, Fernando Reyes Cortés.
%Capítulo 5 Cinemática directa cartesiana.

H_r2gdl.m

```
function H=H_r2gdl()
    syms q1 q2 beta1 beta2 l1 l2 alpha1 alpha2 real
    disp('Parámetros Denavit-Hartenberg del robot planar vertical de 2 gdl')
    disp('[ l alpha d q]')
    dh=[l1, 0, beta1, q1; l2, 0, beta2, q2];
    disp(dh)
    %H_0^1 = H_Rz0(q1) H_Tz0(β1) H_Tx0(l1) H_Rx0(0)
    H10=HRz(q1)*HTz(beta1)*HTx(l1)*HRx(0)
    %H_1^2 = H_Rz0(q2) H_Tz0(β2) H_Tx0(l2) H_Rx0(0)
    H21=HRz(q2)*HTz(beta2)*HTx(l2)*HRx(0)
    H20=simplify(H10*H21);% H_0^2 = H_0^1 H_1^2
    [R20, cinemat_r2gdl, cero, c]=H_DH(H20)
    H=[R20, cinemat_r2gdl; % R_20(q1, q2), f_R(q1, q2)
       cero, c];
end
```

Función cinemática directa del robot de 2 gdl

La función de cinemática directa de un robot de 2 gdl se encuentra dada por:

$$[x_0, y_0, z_0] = \text{cinematica\_r2gdl}(\beta_1, l_1, q_1, \beta_2, l_2, q_2)$$

donde los argumentos de entrada son los parámetros geométricos $\beta_1, l_1, \beta_2, l_2$ y posiciones articulares q_1, q_2 del hombro y codo, respectivamente. Esta función retorna las coordenadas cartesianas (x_0, y_0, z_0) en el sistema $\Sigma_0(x_0, y_0, z_0)$.

Código Fuente 5.6 cinematica_r2gdl.m

```
%MATLAB Aplicado a Robótica y Mecatrónica.
%Editorial Alfaomega, Fernando Reyes Cortés.
%Capítulo 5 Cinemática directa cartesiana.
```

cinematica_r2gdl.m

```
function [x0, y0, z0]=cinematica_r2gdl(beta1,l1,q1,beta2,l2,q2)
    dato1=whos('beta1'); dato2=whos('l1'); dato3=whos('q1');
    dato4=whos('beta2'); dato5=whos('l2'); dato6=whos('q2');
    v1=strcmp(dato1.class, 'sym'); v2=strcmp(dato2.class, 'sym');
    v3=strcmp(dato3.class, 'sym'); v4=strcmp(dato4.class, 'sym');
    v5=strcmp(dato5.class, 'sym'); v6=strcmp(dato6.class, 'sym');
    digits(3);
    if (v1 & v2 & v3 & v4 &v5 & v6) %caso simbólico
        x0=simplify(vpa(l1*cos(q1)+l2*cos(q1+q2),3));
        y0=simplify(vpa(l1*sin(q1)+l2*sin(q1+q2),3));
        z0=vpa(beta1+beta2,3);
        else %caso numérico
        x0=l1*cos(q1)+l2*cos(q1+q2);
        y0=l1*sin(q1)+l2*sin(q1+q2);
        z0=beta1+beta2;
    end
end
```

Función cinemática inversa del robot de 2 gdl

La función de cinemática inversa de un robot de 2 gdl tiene la siguiente sintaxis:

$$[q_1, q_2] = \text{cinv\_r2gdl}(l_1, l_2, x_0, y_0)$$

donde los argumentos de entrada son los parámetros geométricos l_1, l_2 y las coordenadas cartesianas del extremo final x_0, y_0 en el sistema $\Sigma_0(x_0, y_0, z_0)$. Esta función retorna las coordenadas articulares del hombro q_1 y codo q_2.

Código Fuente 5.7 cinv_r2gdl.m

%MATLAB Aplicado a Robótica y Mecatrónica.
%Editorial Alfaomega, Fernando Reyes Cortés.
%Capítulo 5 Cinemática directa cartesiana.

cinv_r2gdl.m

```
function [q1 q2]=cinv_r2gdl(l1,l2,x0,y0)
    q2=acos((x0.*x0+y0.*y0-l1*l1-l2*l2)/(2*l1*l2));
    q1=atan(y0./x0)-atan((l2*sin(q2))./(l1+l2*cos(q2)));
end
```

♣ ♣ Ejemplo 5.2

Diseñar un programa para MATLAB que permita desplegar en forma simbólica los parámetros DH, cinemática cartesiana y el jacobiano del robot vertical planar de 2 gdl. Además, programar una aplicación donde el extremo final del robot trace un círculo de radio $r = 0.2$m, con centro en $[x_c, y_c]^T = [0.3, -0.3]^T$ m, con periodo de movimiento de 6.28 segundos.

Solución

El programa 5.8 contiene el código fuente para MATLAB que soluciona el problema planteado. En la línea 4 se obtiene la matriz de transformación homogénea H_0^2 del robot de 2 gdl y en la línea 7 se obtiene la matriz de rotación R_0^2 y la cinemática directa cartesiana $\mathbf{f}_R(l_1, q_1, l_2, q_2)$. Las líneas 12 y 13 despliegan en forma simbólica el jacobiano y su determinante, respectivamente. La ecuación del círculo se encuentra implementada en las líneas 21 y 22:

$$x = x_c + r\cos(t)$$
$$y = y_c + r\,\mathrm{sen}(t)$$

donde x_c, y_c representan el centro del círculo, r es el radio.

La línea 24 convierte las coordenadas cartesianas del círculo en coordenadas articulares (q_1, q_2) por medio de la función cinv_r2gdl(l_1, l_2, x, y), y en la línea 26 se transforman las coordenadas articulares (q_1, q_2) en las coordenadas cartesianas del extremo final del robot en el sistema $\Sigma_0(x_0, y_0, z_0)$. Se genera una base de tiempo de 100 segundos con incrementos de 1 mseg. El periodo de trazo del círculo es de 6.28 segundos ($\mathrm{sen}(wt)$, con $w = 2\pi f = 1$, entonces $t = 2\pi$ segundos).

La figura 5.6 muestra el círculo generado por el extremo final del robot (ver línea 27) durante una simulación de 100 segundos, lo que significa que el robot traza 15 círculos que se superponen en la misma figura.

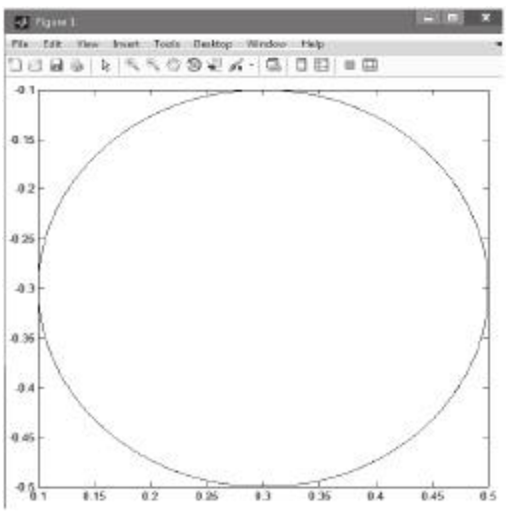


Figura 5.6 Trayectoria circular que traza el robot de 2 gdl.

Código Fuente 5.8 cap5_robot2gdl.m

```
%MATLAB Aplicado a Robótica y Mecatrónica.
%Editorial Alfaomega, Fernando Reyes Cortés.
%Capítulo 5 Cinemática directa cartesiana.
```

cap5_robot2gdl.m

```
clc; clear all ;close all;
format short
syms q1 q2 beta1 beta2 l1 l2 alpha1 alpha2 real
H20=H_r2gdl()
disp('Transformación homogénea del robot 2 gdl');
disp(H20);
[R20, cinemat_r2gdl,cero, c]=H_DH(H20) ;
disp('Matriz de rotación'); disp(R20);
disp('cinemática directa');
disp(cinemat_r2gdl);
[x0, y0, z0]=cinematica_r2gdl(beta1,l1,q1,beta2,l2,q2)
jac_r2gdl=jacobian([x0; y0], [q1;q2])
det_r2gdl=simplify(det(jac_r2gdl)) % det[J]= l1l2 sen(q2)
%ejemplo numérico
t=0:0.001:100;%parámetros del círculo: [xc, yc]^T = [0.3, −0.3]^T y radio r = 0.2
xc=0.3; yc=-0.3; r=0.20;
l1=0.45; l2=0.45;
beta1=0.1; beta2=0.1;
q1=[]; q2=[];
% ecuación del círculo
x=xc+r*sin(t) ;
y=yc+r*cos(t) ;
% cinemática inversa
[q1,q2]=cinv_r2gdl(l1,l2,x,y) ;
%coordenas cartesianas del extremo final del robot de 2 gdl
[x0, y0, z0]=cinematica_r2gdl(beta1,l1,q1,beta2,l2,q2) ;
plot(x0,y0)
```

Robot antropomórfico (RRR)

La gran mayoría de los robots industriales tienen la configuración antropomórfica (brazos robots) ya que presentan mayor destreza en su espacio de trabajo debido a que sus eslabones están unidos por articulaciones rotacionales. El espacio de trabajo de la configuración antropomórfica corresponde a una esfera, cuyo radio corresponde a la suma de longitudes de los eslabones. Por similitud con la forma anatómica con el brazo humano, la segunda articulación se conoce como **hombro** (shoulder) y la tercera articulación se llama **codo** (elbow). La figura 5.7 muestra ejemplos de robots industriales antropomórficos de la compañía ABB y FANUC que se utilizan en aplicaciones de traslado de objetos.

Figura 5.7 Robots industriales en configuración antropomórfica: ABB y FANUC.

Cinemática directa cartesiana del brazo robot de 3 gdl

Considérese un robot manipulador antropomórfico de tres grados de libertad como el que se presenta en la figura 5.8, donde el sistema de referencia fijo $\Sigma_0(x_0, y_0, z_0)$ se encuentra en la base del robot, el eje z_0 coincide con el eje de rotación de la articulación de la base. Los sistemas de referencia $\Sigma_1(x_1, y_1, z_1)$, $\Sigma_2(x_2, y_2, z_2)$ y $\Sigma_3(x_3, y_3, z_3)$ están seleccionados de tal forma que sus ejes z_1, z_2 y z_3 coincidan con sus respectivos ejes de rotación de cada articulación. El eje z_1 del sistema

$\Sigma_1(x_1, y_1, z_1)$ se encuentra ortogonal al eje z_0 ($\alpha_1 = \frac{\pi}{2}$); el eje z_1 es paralelo al eje z_2 ($\alpha_2 = 0$) y el eje z_2 se ha considerado paralelo al eje z_h de la herramienta de trabajo ($\alpha_3 = 0$). Note que el plano $x_2 - y_2$ se encuentra rotado un an´gulo q_3 con respecto al plano $x_1 - y_1$.

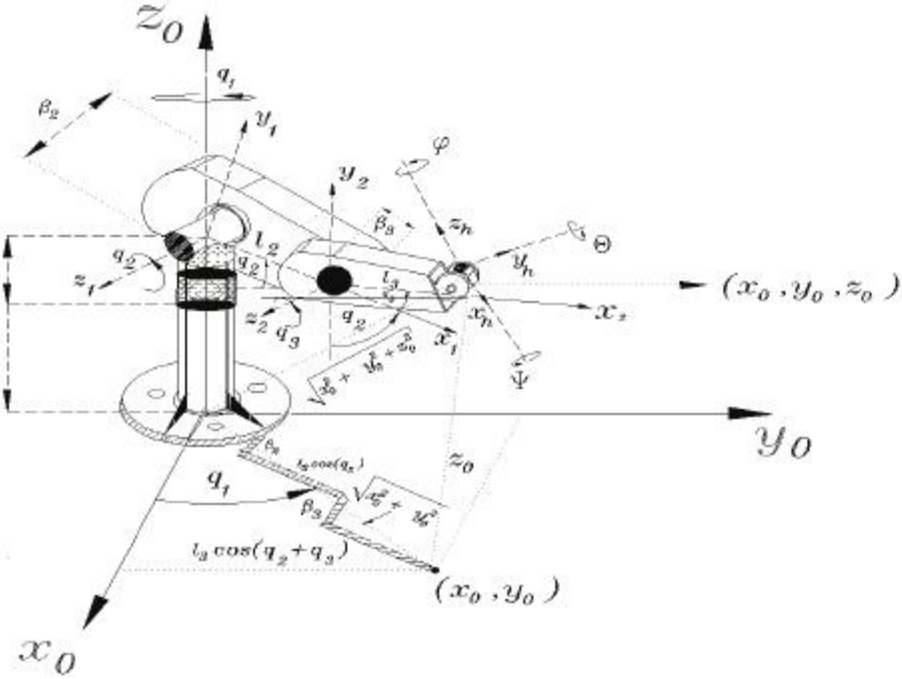


Figura 5.8 Robot antropom´orfico.

La tabla 5.3 contiene los par´ametros Denavit-Hartenberg para el robot antropom´orfico de tres grados de libertad.

Tabla 5.3 DH del robot de 3 gdl

Eslab´on	l_i	α_i	d_i	θ_i
1	0	$\frac{\pi}{2}$	$l_1 + \beta_1$	q_1
2	l_2	0	β_2	q_2
3	l_3	0	β_3	q_3

Los par´ametros geom´etricos de cada servomotor son: β_1, β_2, β_3 representan el ancho y espesor de las placas met´alicas para la base, hombro y codo, respectivamente; l_1