UCM

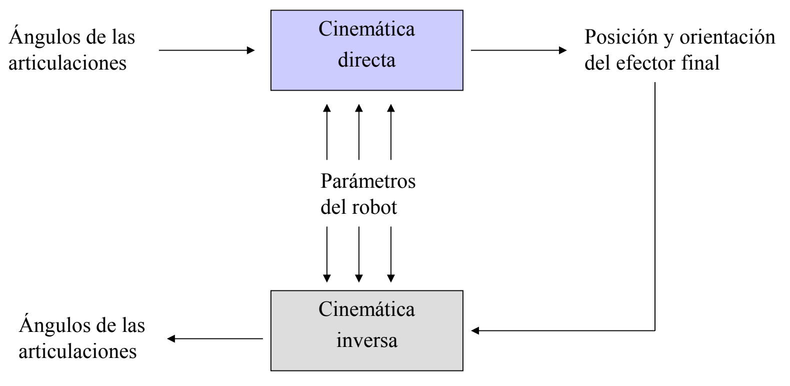
INTRODUCCIÓN (1)

- Definición:
  - Estudio analítico de la geometría del robot con respecto a un sistema de coordenadas de referencia fijo, como función del tiempo y sin considerar las fuerzas/momentos que intervienen en su movimiento.

- Aspectos de la cinemática:
  - *Problema cinemático directo*: dado el vector de ángulos de las articulaciones y los parámetros geométricos del robot, ¿cuál es la orientación y posición del efector final?
  - *Problema cinemático inverso*: dada la posición y orientación deseada para el efector final y los parámetros geométricos del robot, ¿puede alcanzar el manipulador esa posición?, ¿qué configuración permite alcanzarla?.

UCM

INTRODUCCIÓN (2)



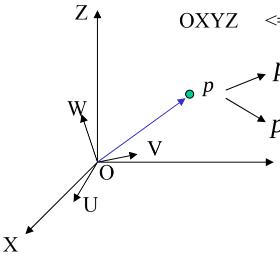
UCM

NOTACIÓN

•¿Qué es una matriz de transformación de coordenadas, R?

$$p_{xyz} = R \cdot p_{uvw}$$

$$p_{uvw} = Q \cdot p_{xyz}$$



OXYZ <=> OUVW

$$p_{xyz} = (p_x, p_x, p_x)^T$$

$$p_{uvw} = (p_u, p_v, p_w)^T$$

•¿Cómo se obtiene?

$$p_x = i_x \cdot p = i_x \cdot i_u \cdot p_u + i_x \cdot j_v \cdot p_v + i_x \cdot k_w \cdot p_w$$
$$p_y = j_y \cdot p = j_y \cdot i_u \cdot p_u + j_y \cdot j_v \cdot p_v + j_y \cdot k_w \cdot p_w$$
$$p_z = k_z \cdot p = k_z \cdot i_u \cdot p_u + k_z \cdot j_v \cdot p_v + k_z \cdot k_w \cdot p_w$$

$$\begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} = \underbrace{\begin{pmatrix} i_x \cdot i_u & i_x \cdot j_v & i_x \cdot k_w \\ j_y \cdot i_u & j_y \cdot j_v & j_y \cdot k_w \\ k_z \cdot i_u & k_z \cdot j_v & k_z \cdot k_w \end{pmatrix}}_{\text{R}} \cdot \begin{bmatrix} p_u \\ p_v \\ p_w \end{bmatrix}$$

• Propiedades: $Q = R^{-1} = R^T$

$$QR = R^T R = R^{-1} R = I_3$$

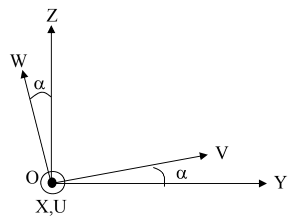
UCM

MATRICES DE ROTACIÓN
Matrices básicas

$$R = \begin{pmatrix} i_x \cdot i_u & i_x \cdot j_v & i_x \cdot k_w \\ j_y \cdot i_u & j_y \cdot j_v & j_y \cdot k_w \\ k_z \cdot i_u & k_z \cdot j_v & k_z \cdot k_w \end{pmatrix}$$

- EJEMPLO: *Matrices de rotación básicas*

$$i_x \equiv i_u \qquad R_{x,\alpha} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\operatorname{sen}\alpha \\ 0 & \operatorname{sen}\alpha & \cos\alpha \end{pmatrix}$$



$$R_{y,\phi} = \begin{pmatrix} \cos\phi & 0 & \operatorname{sen}\phi \\ 0 & 1 & 0 \\ -\operatorname{sen}\phi & 0 & \cos\phi \end{pmatrix} \qquad R_{z,\theta} = \begin{pmatrix} \cos\theta & -\operatorname{sen}\theta & 0 \\ \operatorname{sen}\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

UCM

MATRICES DE ROTACIÓN
Matriz de rotación compuesta

(El Sistema OUVW puede girar respecto a los ejes del Sistema OXYZ y respecto a su propio eje)

REGLAS:
1.- Inicialmente Sistemas coincidentes.
2.- Si OUVW está girado respecto a los ejes del sistema OXYZ ➡ *premultiplicar*
3.- Si OUVW está girado respecto su propio eje principal ➡ *postmultiplicar*

Ej: Encontrar R => Giro ϕ respecto al eje OY, rotación θ respecto al eje OW y rotación α respecto a OU

$$R = R_{y,\phi} \cdot I_3 \cdot R_{w,\theta} \cdot R_{u,\alpha} = \begin{pmatrix} c\phi & 0 & s\phi \\ 0 & 1 & 0 \\ -s\phi & 0 & c\phi \end{pmatrix} \cdot \begin{pmatrix} c\theta & s\theta & 0 \\ -s\theta & c\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & c\alpha & s\alpha \\ 0 & -s\alpha & c\alpha \end{pmatrix} =$$

$$= \begin{pmatrix} c\phi + c\theta & s\phi \cdot s\alpha - c\phi \cdot s\theta \cdot c\alpha & c\phi \cdot s\theta \cdot s\alpha + s\phi \cdot c\alpha \\ s\theta & c\theta \cdot c\alpha & -c\theta \cdot s\alpha \\ -s\phi + c\theta & s\phi \cdot s\theta \cdot c\alpha + c\phi \cdot s\alpha & c\phi \cdot c\alpha - s\phi \cdot s\theta \cdot s\alpha \end{pmatrix}$$

EJEMPLO

UCM

COORDENADAS HOMOGÉNEAS
Definiciones

- R3x3 no permite traslación y escalado.

$$\underbrace{p = (p_x, p_y, p_z)^T}_{\text{N componentes}} \Longrightarrow \underbrace{\hat{p} = (w \cdot p_x, w \cdot p_y, w \cdot p_z, w)^T}_{\text{N+1 componentes}}$$

Representación en coordenadas homogéneas

- No existe representación única:

$$\hat{p}_1 = (w_1 \cdot p_x, w_1 \cdot p_y, w_1 \cdot p_z, w_1)^T \equiv \hat{p}_2 = (w_2 \cdot p_x, w_2 \cdot p_y, w_2 \cdot p_z, w_2)^T$$

En robótica escala =1 (se utiliza en informática gráfica)

UCM

COORDENADAS HOMOGÉNEAS
Matriz de transformación

- R_{3x3} ===> T_{4x4} (Matriz de transformación en coordenadas homogéneas)

$$T = \left(\begin{array}{c|c} R_{3x3} & p_{3x1} \\ \hline f_{1x3} & 1x1 \end{array}\right) = \left(\begin{array}{c|c} \text{matriz de rotación} & \text{vector de posición} \\ \hline \text{transformación perspectiva} & \text{escalado} \end{array}\right)$$

- EJEMPLOS:

Rotación $T_{x,\alpha} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\operatorname{sen}\alpha & 0 \\ 0 & \operatorname{sen}\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$ Traslación $T_{tras} = \begin{pmatrix} 1 & 0 & 0 & dx \\ 0 & 1 & 0 & dy \\ 0 & 0 & 1 & dz \\ 0 & 0 & 0 & 1 \end{pmatrix}$

Escalado local y global

$$T_{escL} = \begin{pmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} ax \\ by \\ cz \\ 1 \end{pmatrix} \qquad T_{escG} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & s \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x \\ y \\ z \\ s \end{pmatrix}$$

UCM

COORDENADAS HOMOGÉNEAS
Matriz de transformación compuesta

- REGLAS:
  1.- Inicialmente ambos sistemas son coincidentes => $T_{4x4} = I_{4x4}$
  2.- Si el sistema OUVW rota/traslada respecto a los ejes principales OXYZ, entonces *premultiplicar* la matriz previa (resultante).
  3.- Si el sistema OUVW rota/traslada respecto a su propio eje principal, entonces *postmultiplicar* la matriz previa (resultante).

- EJEMPLO:

Encontrar T que represente una rotación α respecto al eje OX, una traslación de *a* unidades a lo largo del eje OX, una traslación de *d* unidades a lo largo del eje OZ, seguida de una rotación θ respecto del eje OZ.

$$T = T_{z,\theta} T_{z,d} T_{x,a} T_{x,\alpha} I_{4x4} = \begin{pmatrix} \cos\theta & -\operatorname{sen}\theta & 0 & 0 \\ \operatorname{sen}\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\operatorname{sen}\alpha & 0 \\ 0 & \operatorname{sen}\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

EJEMPLO

UCM

MATRIZ DE TRANSFORMACIÓN
Matriz de transformación general

- Puede considerarse como:
  1.- Una transformación de un sistema de coordenadas a otro.
  2.- Una descripción del origen y ejes del nuevo sistema en términos del sistema de referencia.
  3.- Una descripción del movimiento de un objeto desde una localización (sistema de referencia) a otra (nuevo sistema).
  4.- Un medio de calcular la localización de un punto de un objeto con respecto al sistema de referencia desde su localización respecto al nuevo sistema.

$$T = \begin{pmatrix} x_x & y_x & z_x & p_x \\ x_y & y_y & z_y & p_y \\ x_z & y_z & z_z & p_z \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \vec{x} & \vec{y} & \vec{z} & \vec{p} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$p = p_x i + p_y j + p_z k$ $\equiv$ localización del origen del nuevo sistema de referencia

$x = x_x i + x_y j + x_z k$ $\equiv$ dirección del eje x del nuevo sistema de referencia
$y = y_x i + y_y j + y_z k$ $\equiv$ dirección del eje y del nuevo sistema de referencia
$z = z_x i + z_y j + z_z k$ $\equiv$ dirección del eje z del nuevo sistema de referencia

UCM

MATRIZ DE TRANSFORMACIÓN
Matriz Homogénea inversa

- Se realiza de la forma:
  1.- La matriz de rotación R_{3x3} pasa a ser traspuesta .
  2.- La perspectiva y la escala no se modifican (a cero y a uno).
  3.- El punto de origen se modifica de la forma siguiente:

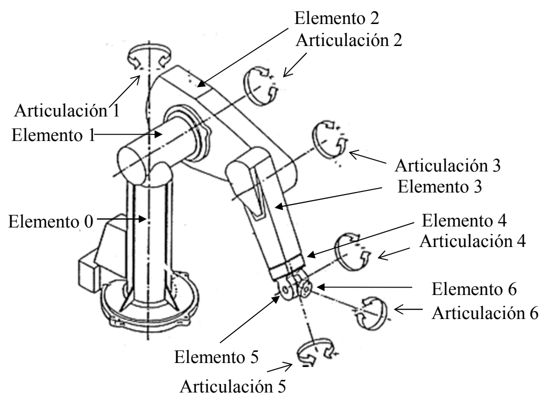
$$p_x \longrightarrow -\vec{p}\cdot\vec{x} \qquad p_y \longrightarrow -\vec{p}\cdot\vec{y} \qquad p_z \longrightarrow -\vec{p}\cdot\vec{z}$$

$$\boxed{T^{-1} = \begin{pmatrix} x_x & x_y & x_z & -\vec{p}\cdot\vec{x} \\ y_x & y_y & y_z & -\vec{p}\cdot\vec{y} \\ z_x & z_y & z_z & -\vec{p}\cdot\vec{z} \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} x_x & x_y & x_z & -p_x x_x - p_y x_y - p_z x_z \\ y_x & y_y & y_z & -p_x y_x - p_y y_y - p_z y_z \\ z_x & z_y & z_z & -p_x z_x - p_y z_y - p_z z_z \\ 0 & 0 & 0 & 1 \end{pmatrix}}$$

$$T = \begin{pmatrix} x_x & y_x & z_x & p_x \\ x_y & y_y & z_y & p_y \\ x_z & y_z & z_z & p_z \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \vec{x} & \vec{y} & \vec{z} & \vec{p} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

UCM

ELEMENTOS, ARTICULACIONES Y PARÁMETROS

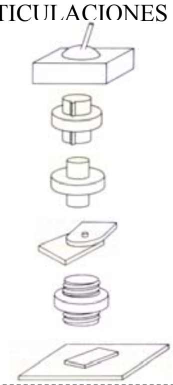


PUMA 560

N pares articulación-elemento
⇕
N grados de libertad
(Elemento 0 no considerado)

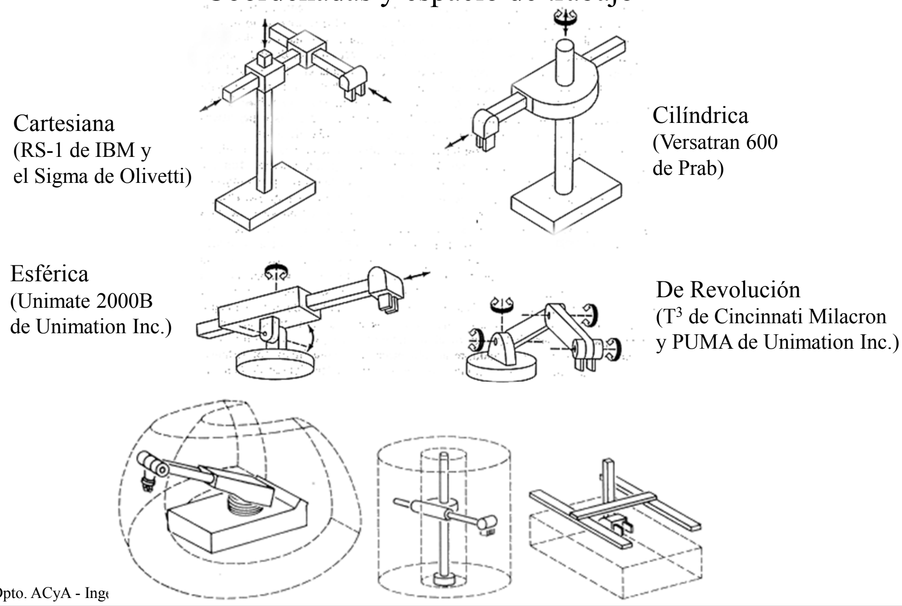
TIPOS DE ARTICULACIONES

- Esférica
- Prismática (deslizante)
- Cilíndrica
- Revolución (giratoria)
- De tornillo
- Planar



UCM

TIPOS DE ROBOTS
Coordenadas y espacio de trabajo



UCM

SISTEMA DE COORDENADAS D-H
Algoritmo de asignación (Denavit-Hartenberg)

Establecer el sistema de coordenadas de la base (x0,y0,z0):
eje z0 a lo largo del eje de movimiento de la articulación 1
y hacia fuera del hombro del robot

i=1...,n-1

Establecer los ejes de la articulación (xi,yi,zi)
Alinear zi con el eje de movimiento (giratorio o deslizante)
de la articulación i+1

Establecer el origen de coordenadas i
Intersección de los ejes zi y zi-1 (o intersección de las normales
comunes entre los ejes zi y zi-1 y el eje zi)

Establecer los ejes xi e yi
xi -> ±(zi-1 x zi)/|zi-1 x zi| (ó a lo largo de la normal
común a los ejes zi-1 y zi cuando son paralelos)
yi -> Para completar el sist. De coordenadas dextrógiro

Establecer el sistema de coordenadas de la mano (n)
zn -> a lo largo de la dirección del eje zn-1 y hacia fuera
xn -> normal a los ejes zn-1 y zn
yn -> de modo que el sist. Coordenadas sea dextrógiro

Encontrar los parámetros de la articulación y del elemento (i=1,...,n)
di, ai, θi, αi

UCM

PROBLEMA CINEMÁTICO DIRECTO
Parámetros del manipulador

$$\{\underbrace{a_i,\alpha_i}_{},\underbrace{d_i,\theta_i}_{}\}$$

Parámetros del elemento
(longitud y ángulo de torsión)

Parámetros de la articulación
(distancia y ángulo entre elementos i - i+1)

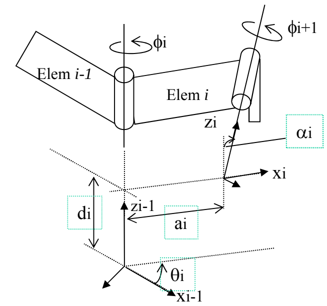
a_i : Distancia más corta entre los ejes zi-1 y zi
Corte del eje zi-1 con el eje xi hasta origen i.
Signo según eje xi

α_i : Ángulo entre el eje zi-1 y zi respecto del eje xi
Regla mano derecha: eje xi es dirección +

d_i : Distancia entre el origen de coord. del sistema i-1 hasta la intersección del eje zi-1 con el eje xi a lo largo del eje zi-1 Signo según eje zi-1

θ_i : Ángulo del eje xi-1al eje xi respecto del eje zi-1
Regla mano derecha: eje zi-1 es dirección +

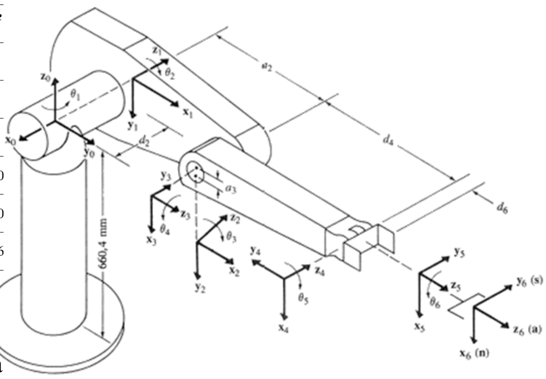
(Parámetros fijos)



UCM

PARÁMETROS DEL MANIPULADOR
Ejemplo. Robot PUMA

Artic i	θ_i	α_i	a_i	d_i	*Rango de la artic*
1	90	-90	0	0	-160 a +160
2	0	0	431,8 mm	149,09 mm	-225 a 45
3	90	90	-20,32 mm	0	-45 a 225
4	0	-90	0	433,07 mm	-110 a 170
4	0	90	0	0	-100 a 100
6	0	0	0	56,25	-266 a 266



Articulación de revolución o giratoria

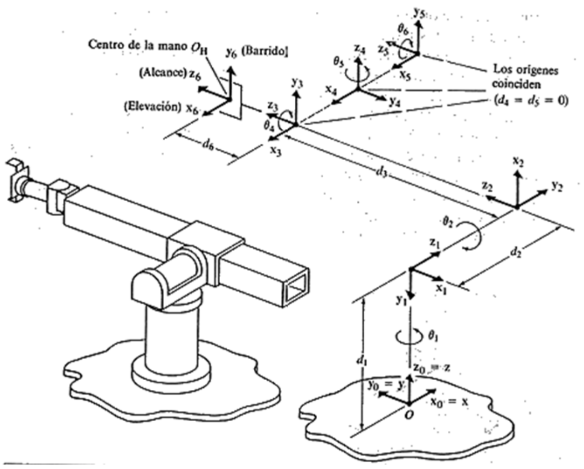
Variable de la articulación: θ_i

Parámetros fijos: α_i, a_i, d_i

UCM

PARÁMETROS DEL MANIPULADOR
Ejemplo. Robot STANFORD

Artic i	θ_i	α_i	a_i	d_i
1	-90	-90	0	d_1
2	-90	90	0	d_2
3	-90	0	0	d_3
4	0	-90	0	0
4	0	90	0	0
6	0	0	0	d_6



Articulación prismática

Variable de la articulación: d_i

Parámetros fijos: θ_i, α_i, a_i

UCM

SISTEMA DE COORDENADAS D-H
Representación Denavit-Hartenberg

Manipulador de *n* grados de libertad + Sist. Coord. De la base

$$(x_0, y_0, z_0), (x_1, y_1, z_1), \ldots\ldots, (x_n, y_n, z_n)$$

pi (coord. Sistema i) $\Rightarrow$ pi-1 (coord. Sistema i-1)

1.- Girar eje zi-1 un ángulo θi para alinear xi-1 con xi
2.- Moverse una distancia di por el eje zi-1 para hacer coincidir xi-1 y xi
3.- Trasladarse una distancia ai por eje xi para coincidir los dos orígenes
4.- Girar un ángulo αi respecto el eje xi para coincidir los dos sistemas de coordenadas

$$^{i-1}A_i = T_{z,d} \cdot T_{z,\theta} \cdot I \cdot T_{x,a} \cdot T_{x,\alpha}$$

$$\boxed{^0T_i = {}^0A_1 \, {}^1A_2 \ldots {}^{i-1}A_i = \prod_{j=1}^{i} {}^{j-1}A_j}$$

UCM

SISTEMA DE COORDENADAS D-H
Matriz de transformación entre articulaciones

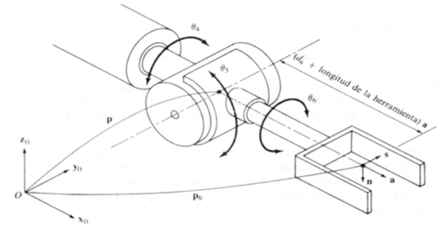
$$^{i-1}A_i = T_{z,d} \cdot T_{z,\theta} \cdot T_{x,a} \cdot T_{x,\alpha}$$

$$^{i-1}A_i = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} c\theta_i & -s\theta_i & 0 & 0 \\ s\theta_i & c\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & c\alpha_i & -s\alpha_i & 0 \\ 0 & s\alpha_i & c\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\boxed{^{i-1}A_i = \begin{pmatrix} c\theta_i & -c\alpha_i \cdot s\theta_i & s\alpha_i \cdot s\theta_i & a_i \cdot c\theta_i \\ s\theta_i & c\alpha_i \cdot c\theta_i & -s\alpha_i \cdot c\theta_i & a_i \cdot s\theta_i \\ 0 & s\alpha_i & c\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix}}$$

UCM

EFECTOR FINAL
Interpretación geométrica



n => vector normal de la mano
s => vector de deslizamiento de la mano
a => vector de aproximación de la mano
p => vector de posición de la mano

$$
{}^{0}T_6 = \begin{pmatrix} x_6 & y_6 & z_6 & p_6 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} {}^{0}R_6 & {}^{0}p_6 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} n & s & a & p \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} n_x & s_x & a_x & p_x \\ n_y & s_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{pmatrix}
$$

Representaciones de posición/orientación

Cartesiana	$(p_x, p_y, p_z)^T$	Cartesiana	$[n, s, a]$
Cilíndrica	$(rC\alpha, rS\alpha, d)^T$	Euler	$[\phi, \theta, \psi]$
Esférica	$(rC\alpha S\beta, rS\alpha S\beta, rC\beta)^T$	Roll, Picth, Yaw	$[R, P, Y]$

$$
T_{pos} = \begin{pmatrix} 1 & 0 & 0 & p_x \\ 0 & 1 & 0 & p_y \\ 0 & 0 & 1 & p_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad T_{orient} = \begin{pmatrix} [n,s,a] \text{ ó } R_{\phi,\theta,\psi} & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \\ 0 \;\; 0 \;\; 0 & 1 \end{pmatrix}
$$

$$
{}^{0}T_6 = T_{pos} \cdot T_{orient}
$$

UCM

EJEMPLO
Parámetros y coordenadas D-H

Pasos en la cinemática directa:

1.- Mover el manipulador a posición cero

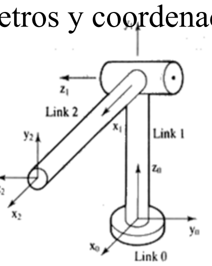
2.- Asignar ejes de coordenadas a cada articulación

3.- Obtener los parámetros de cada articulación

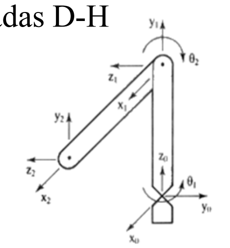
4.- Obtener relación entre articulaciones ${}^{i-1}A_i$

5.- Multiplicar todas las matrices para ${}^{i-1}A_i$ obtener ${}^{0}T_{ef}$

6.- Resolver los ángulos de las articulaciones para las coordenadas del efector final

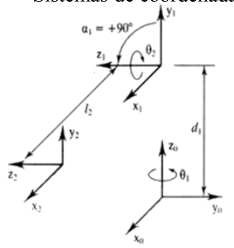


Manipulador en posición cero

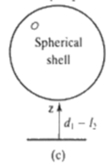


Esquema de las articulaciones

Sistemas de coordenadas



Espacio de trabajo



Parámetros de las articulaciones

Link	variable	θ	α	l	d
1	θ_1	θ_1	90°	0	d_1
2	θ_2	θ_2	0	l_2	0

UCM

Definición del problema

- Dada una posición y orientación para el efector final de un manipulador
  Obtener: *los ángulos correspondientes para cada articulación*

$$\left.\begin{array}{c} {}^0T_6 = \begin{pmatrix} n_x & s_x & a_x & p_x \\ n_y & s_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ \{a_i, \alpha_i, d_i, \theta_i\} \end{array}\right\} \xrightarrow{\text{Cinemática inversa}} \vec{q} = (q_1, q_2, q_3, q_4, \ldots, q_n)^T$$

- DIFICULTADES
  - –Existen multitud de soluciones para una misma posición/orientación del efector final.
  - –Existen múltiples métodos para encontrar la solución inversa (todos necesitan tomar decisiones para restringir la solución final)
    - Solución analítica o matricial (para determinados tipos de robots).
    - Métodos geométricos.
    - Métodos iterativos, cuaterniones duales, etc.

UCM

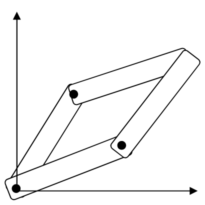
PROBLEMA CINEMÁTICO INVERSO
Redundancias

Redundancia: Un manipulador puede alcanzar una posición con *varias configuraciones* distintas de sus articulaciones

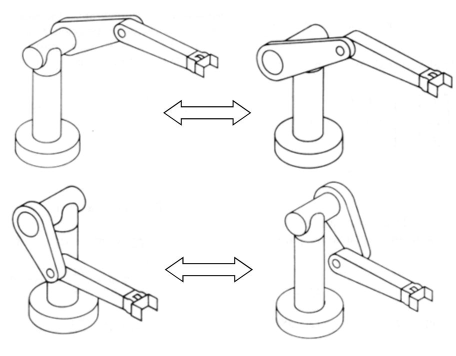
- Funciones matemáticas (raiz cuadrada, coseno, ...) en la solución cinemática inversa con indefinición en el signo, indica redundancias.
- N redundancias => 2^n

EJ:

2-grados de libertad



6-grados de libertad



UCM

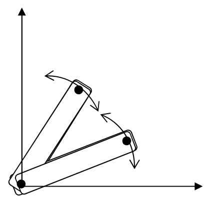
PROBLEMA CINEMÁTICO INVERSO
Degeneraciones

Degeneración: Un manipulador puede alcanzar una posición con *infinitas configuraciones* distintas de sus articulaciones

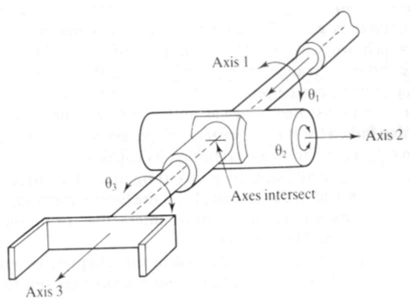
EJ:

El punto (0,0) es degenerado

$$\vartheta_2 = 180º$$
$$\vartheta_1 = atan2(p_y, p_x) - atan2(l_2 S_2, l_1 + l_2 C_2) =$$
$$= atan(0/0) - atan(0/0)$$



La suma de los ángulos θ_1 y θ_3 es cero, pero uno puede girar 180º respecto a otro para mantener la misma posición



UCM

RESOLUCIÓN
Método analítico. Algoritmo de resolución

Igualar la matriz transformación general con la matriz deseada para el efector final
$^0T_6 = (^0T_6)_D$

Buscar elementos en las matrices (ecuaciones válidas):
a) Con una única variable
b) Pares de elementos que dén lugar al dividir a una única variable (*mejor la función atan2*)
c) Elementos o pares de elementos que pueden ser simplificados usando identidades trigonométricas

Elegir un elemento seleccionado:
Despejar la variable de articulación en función de la posición del efector final

¿Más elementos? si / no

¿Hay imprecisiones? si / no

Apartarlos y buscar mejores soluciones
Los términos de ***p*** son más eficientes que los términos de ***x***, ***y***, y ***z***.

Buscar nuevas ecuaciones
premultiplicar por la inversa de A de la 1ª articulación
ó
postmultiplicar por la inversa de A de la última articulación

¿Faltan soluciones? si
Repetir hasta acabar A's
no

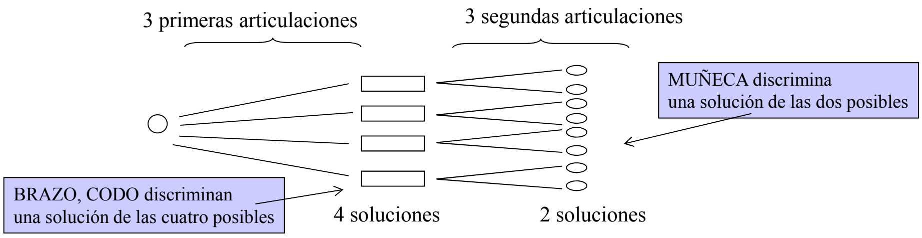
Si quedan articulaciones sin encontrar:
Tomar una de las descartadas
(tomar nota de las regiones con problemas)

Si no se encuentra solución a una articulación
El efector puede estar fuera del espacio de trabajo
(Las soluciones teóricas no pueden ser físicamente realizables debido a restricciones físicas).

UCM

RESOLUCIÓN
Método geométrico. Problema

- Debido a posibilidad de los dos signos (raiz cuadrada, cos, ...) en la solución existen diversas soluciones para el mismo problema.
  - El método geométrico da una orientación de qué signo tomar en cada caso.
- Para un robot PUMA:
  - Se toman tres indicadores de configuración: BRAZO, CODO y MUÑECA



Otros ejemplos: en hoja de problemas

UCM

PROGRAMACIÓN DE ROBOTS
Introducción. Tipos

- Reconocimiento de voz
  - Sería el más adecuado en muchos casos.
  - De momento no se ha avanzado lo suficiente:
    - Dificultad en reconocer las palabras adecuadas.
    - Tratar un vocabulario suficientemente extenso.
    - Entrenamiento para reconocer patrones de voz.
- Métodos guiados:
  - Consiste en enseñar y reproducir los movimientos que se desean.
  - Es el más utilizado en robot industriales (soldadura, pintura con spray, ...)
  - Método:
    - (1) control manual: mueve léntamente al robot a las posicicones deseadas,
    - (2) se reproduce lo enseñado,
    - (3) Si es correcto: el robot realiza los movimientos a velocidad apropiada.
- Programación alto nivel: { - orientado a nivel de robot; - a nivel de tarea }
  - Es la más utilizada hoy en día.
  - Aumenta la flexibilidad y versatilidad de los robot. Permite utilizar un robot para realizar distintas tareas.
  - Utilizado normalmente en robot con multiples sensores y robots móviles.

UCM

LENGUAJES DE NIVEL DE ROBOTS
Características

1.- Permitir establecer/especificar el lugar de trabajo

2.- Especificar la posición y describir acciones y movimientos.

3- Órdenes para usar los sensores: detectar anomalías o vigilar el progreso de una tarea.

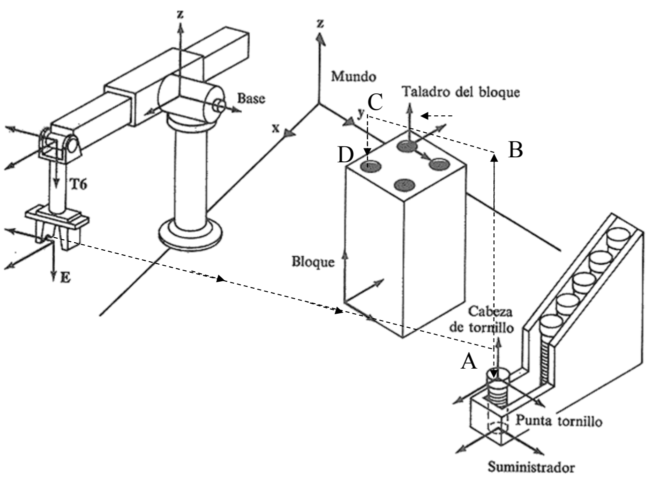
Lenguajes clásicos:

⊗ **AL** (Univ. Stanford)
- Lenguaje alto nivel con características de ALGOL y Pascal.
- Entorno para especificaciones a nivel de robot y nivel de tarea.
- Usa ALGOL como estructura de datos y de control.
- Entorno para modelado del mundo.

⊗ **AML** (IBM para el RS-1)
- Entorno para diferentes interfaces de usuario.
- Admite construcciones LISP y APL.
- Permite añadir datos y planificar una trayectoria en el espacio de segmentos (sujetos a restricciones de vel. y pos.)
- Produce movimientos absolutos y relativos.
- Vigila los sensores para producir interrupciones en el movimiento.

UCM

LENGUAJES DE NIVEL DE ROBOTS
Ejemplo



UCM

LENGUAJES DE NIVEL DE ROBOTS
Especificación de posición y lugar de trabajo

Posición y sistemas de referencia

AL:
base ← FRAME(nilrot, VECTOR(20,0,15)*pulgadas);
bloque ← FRAME(ROT(Z,90*grad), VECTOR(20,15,0)*pulgadas);
alimentador ← FRAME(nilrot, VECTOR(25,20,0)*pulgadas);
AML:
base = < <20,0,15>, EULERROT(<0,0,0>)>;
bloque = < <20,15,0>, EULERROT(<0,0,90>)>;
alimentador = < <25,20,0>, EULERROT(<0,0,0>)>;

Características de objetos

AL:
T6 ← *base* * TRANS (ROT(X,180*grad), VECTOR(15,0,0)*pulgadas);
E ← *T6* * TRANS (nilrot, VECTOR(0,0,5)*pulgadas);
punta_tornillo ← alimentador * TRANS(nilrot,nilvect);
cabeza_tornillo ← *punta_tornillo* * TRANS(nilrot, VECTOR(0,0,1)*pulgadas);
taladro_bloque ← *bloque* * TRANS(nilrot, VECTOR(0,2,3)*pulgadas);
AML:
T6 = DOT(*base*, < <15,0,0>, EULERROT(<180,0,0>)>);
E = DOT(*T6*, < <0,0,5>, EULERROT(<0,0,0>)>);
punta_tornillo = DOT(*alimentador*, < <0,0,0>, EULERROT(<0,0,0>)>);
cabeza_tornillo = DOT(*punta_tornillo*, < <0,0,1>, EULERROT(<0,0,0>)>);
taladro_bloque = DOT(*bloque*, < <0,2,3>, EULERROT(<0,0,0>)>);

UCM

LENGUAJES DE NIVEL DE ROBOTS
Especificación de movimiento

- Puede realizarse mediante un espacio cartesiano (AL) o mediante un espacio de segmentos (AML).
- Producen caminos ineficaces y programas muy largos
- Se debe controlar la velocidad, aceleración, deceleración y direcciones de salida y llegada.

Características de objetos

AL:
{Mover el brazo desde reposo a A y luego hacia cabeza_tornillo}
MOVE *barm* TO *A;*
MOVE *barm* TO *cabeza_tornillo;*
{otra forma de especificar movimiento}
MOVE *barm* TO *cabeza_tornillo* VIA *A*;
{desplazarse a lo largo del eje Z una pulgada (mov. Relativo)}
MOVE *barm* TO ⊗ - 1*Z*pulgadas;
AML:
-- Mover los segmentos 1 y 4 a 10 pulgadas y 20º (mov absoluto)
MOVE(<1, 4>, <10, 20>);
-- Mover segs. 1,3 y 6 1,2 pulgadas y 5º respectivamente (mov relativo)
DMOVE(<1,3,6>, <1,2,5>);

Sentencias de movimiento

AL:
{Mover el brazo desde cabeza_tornillo hasta A}
MOVE *barm* TO *A;*
WITH DEPARTURE = Z WRT *alimentador*;
WITH DURATION = 5*segundos;
{Abrir la mano 2,5 pulgadas}
OPEN *bhand* TO 2,5*pulgadas;
AML:
-- Mover los segmentos 1 y 4 a 10 pulgadas y 20º (vel= 1pulgada/s)
-- Aceleración y deceleración 1pulgada/s
MOVE(<1, 4>, <10, 20>, <1,1,1>);
-- Abrir la mano 2,5 pulgadas
MOVE(PINZA, 2,5);

UCM

LENGUAJES DE NIVEL DE ROBOTS
Uso de sensores y flujo de control

ScortBot

- No hay consenso y cada lenguaje tiene su propia sintaxis.
- Depende mucho de los sensores que permita utilizar.
- En general, son órdenes como:
  - Fuerza: FORCE(eje), TORQUE(eje), QPOSITION(<artic-i, artic-j>), ...
  - Control de flujo: <if-then-else>, <case->, <do-until>, <while-do>

AL:
{Test para detectar la presencia de agujeros con un sensor de fuerza}
MOVE *barm* TO ⊗ - 1*Z*pulgadas ON FORCE(Z) > 10*onzas*;*
DO ABORT("No hay agujero");
{Insertar tonillo}
MOVE *barm* TO *taladro_bloque*;
WITH FORCE(Z) = -10*onzas WITH FORCE(X) = 0*onzas WITH FORCE(Y) = 0*onzas WITH DURATION = 3*segundos;
AML:
-- Definir un monitor para los sensores de fuerza SLP y SLR; el monitor se encarga de detectar si el sensor excede el rango 0 a F
fmons=MONITOR(<SLP,SLR>, 1,0,F);
-- Mover el segmento 3 una pulgada y para si *fmons* es activado
DMOVE(<3>,<1>,*fmons);*

UCM

LENGUAJES DE NIVEL DE TAREA
Definición

- Programación en términos de los objetos a manejar y no de movimientos del robot.
- Planificación de la tareas:
  - Modelado del mundo.
  - Especificación de la tarea.
  - Síntesis del programa.
- Semejante a la generación automática de programas en inteligencia artificial.

Especificación de la tarea
↓
Descomposición de la tarea
↓ .... ↓
Conocimiento ⟷ Planificación de subtareas ⟷ Modelo
↓
Programa del robot

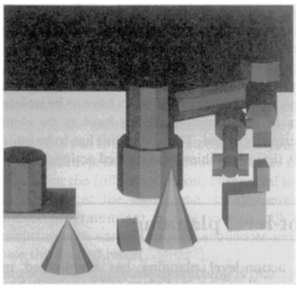
UCM

LENGUAJES DE NIVEL DE TAREA
Modelización del mundo

- Información espacial de los objetos: dimensión, volumen y forma.
- Formas de representar objetos (CSG => *Construcción de un sólido geométrico*):

1.- Composición de objetos y descomposición en células (GDP).
  * Operaciones de movimiento: rotación, traslación y escala.
  * Operaciones de combinación: unión, intersección, diferencia y complemento.

2.- Conjunto de fronteras y de puntos, superficies y cilindros generalizados.
  * Se utiliza un grafo dirigido que contiene: nodos objetos, caras, aristas y vértices.
  * Cuanto más se complica el objeto más ancho es el árbol, pero no más profundo.
    - Vértices: almacenados como coord. cartesianas.
    - Aristas: almacenadas como pares de vértices.
  * Dificultades en la especificación de las superficies curvadas (ej: cilindro con dodecahedros).

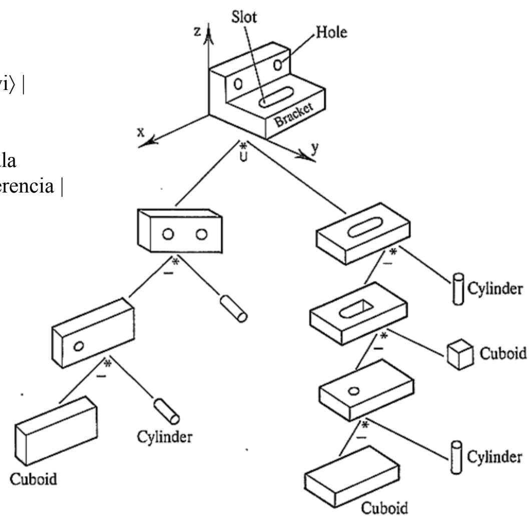


UCM

LENGUAJES DE NIVEL DE TAREA
Ejemplo GDP (1)

Especificación:

⟨ parte mecánica⟩ ::= ⟨objeto⟩
⟨objeto⟩ ::= ⟨primitiva⟩ |
  ⟨objeto⟩ ⟨op movi⟩ ⟨arg movi⟩ |
  ⟨objeto⟩ ⟨operador⟩ ⟨objeto⟩
⟨primitiva⟩ ::= cubo | cono | cilindro | ...
⟨op movi⟩ ::= rotación | traslación | escala
⟨arg movi⟩ ::= unión | intersección | diferencia |
  complemento



UCM

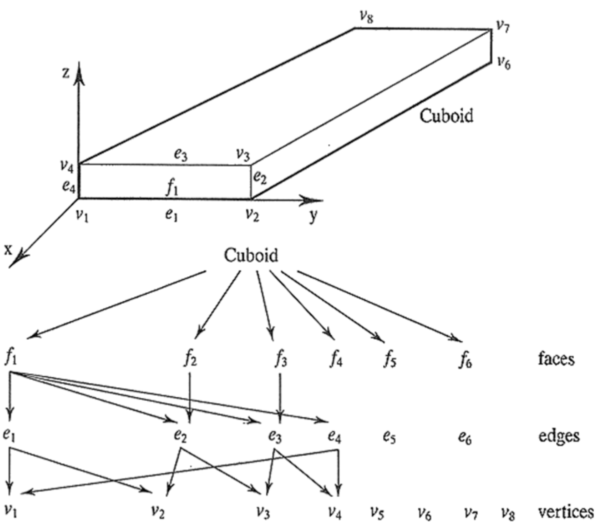
LENGUAJES DE NIVEL DE TAREA
Ejemplo GDP (2)

Ej: AUTOPASS
Sistema de modelización mediante sólidos: GDP (*Procesamiento de diseño geométrico*)

```
Tornillo: PROCEDURE(long_punta, radio_punta,ncaras_punta, longitud_cabeza, radio_cabeza,
ncaras_cabeza);
/* definir parámetros */
DECLARE
   long_punta, long_cabeza, radio_punta, radio_cabeza, ncaras_punta, ncaras_cabeza, FLOAT;
/*define la forma de la punta */
CALL SOLID(CYLIND, <<Punta>>, long_punta, radio_punta, ncaras_punta);
/*definir la forma de la cabeza*/
CALL SOLID(CYLIND, <<Cabeza>>, long_cabeza, radio_cabeza, ncaras_cabeza);
/*Establecer la unión ambos para obtener el tornillo */
CALL MERGE(<<Punta>>, <<Cabeza>>, union);
END Tornillo;
```


UCM

LENGUAJES DE NIVEL DE TAREA
Ejemplo. Representación frontera



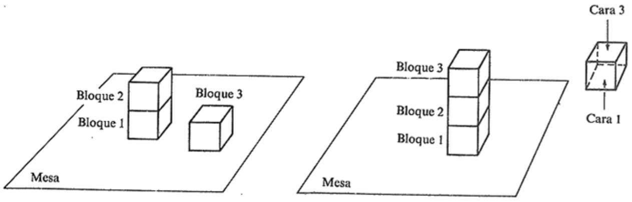
UCM

LENGUAJES DE NIVEL DE TAREA
Especificación de tareas

Ej: Descripción de estados del mundo

Estado A:
(Bloque1_cara1 AGAINST Mesa)
(Bloque1_cara3 AGAINST Bloque2_cara1)
(Bloque3_cara1 AGAINST Mesa)

Estado B:
(Bloque1_cara1 AGAINST Mesa)
(Bloque1_cara3 AGAINST Bloque2_cara1)
(Bloque2_cara3 AGAINST Bloque3_cara1)



Ej: Insertar y atornillar un tornillo

PLACE *tornillo* ON *bloque* SUCH THAT *punta_tornillo* IS ALIGNED WITH *taladro_bloque;*
DRIVE IN *tornillo* AT *cabeza_tornillo* SUCH THAT TORQUE IS EQ 12,0 IN_LBS USING *destornillador;*

UCM

LENGUAJES DE NIVEL DE TAREA
Síntesis del programa

- Fase más importante y difícil de la planificación de tareas.
- Pasos principales:
  - Planificación de la sujeción:
    - Configuración estable, según la geometría del objeto y que reduzca lo más posible la incertidumbre.
    - Agarre de objetos con seguridad.
  - Planificación del desplazamiento:
    - Salida cuidadosa desde la posición inicial.
    - Desplazamiento libre hasta la configuración deseada sin choques.
    - Aproximación cuidadosa hasta el destino.