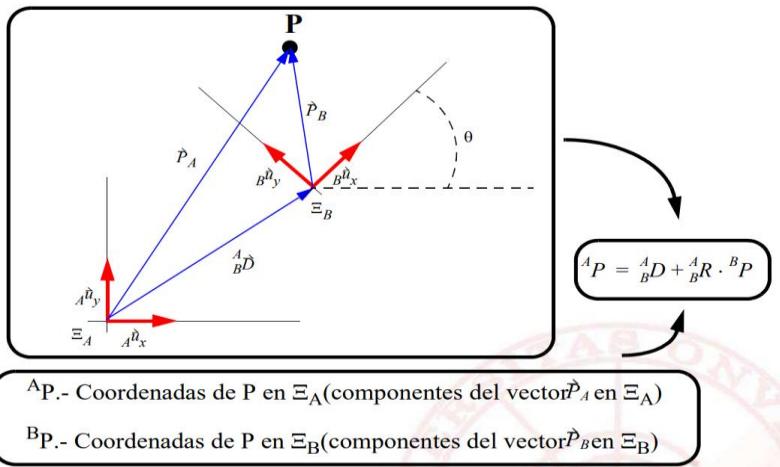
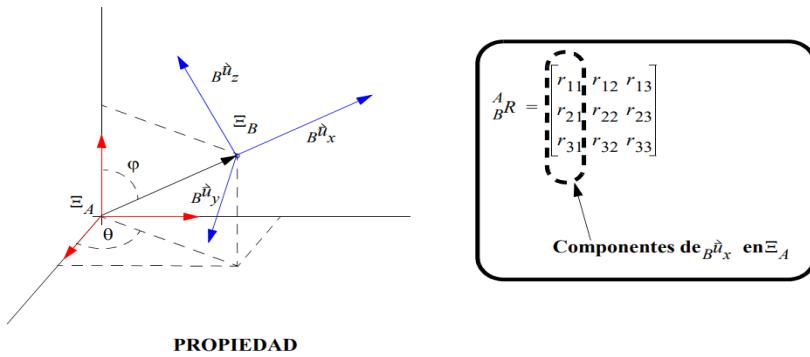


Tema 3:

Relación entre sistemas de referencia:



Sistema de referencia en 3D

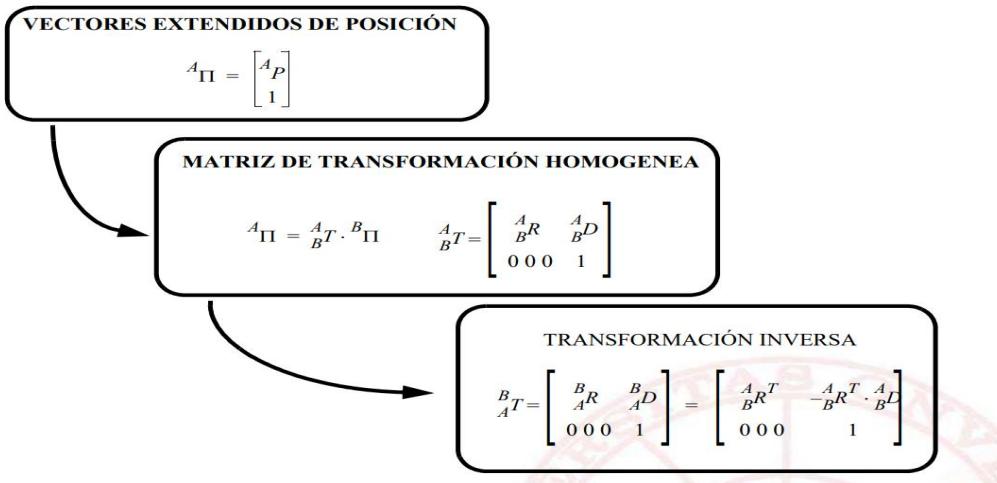


PROPIEDAD

$$A_B R = B_A R^{-1} = B_A R^T$$

Relaciones de transformación homogénea: vectores extendidos, matriz de transformación homogénea y transformación inversa.

Relaciones de Transformación Homogénea



Transformaciones binarias: Transformaciones realizadas en referencia a los ejes coordenados

$$Tras(Y, a) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & a \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$Rot(Z, \theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$Rot(Y, \theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Otras matrices de transformación: Roll = balanceo, pitch = inclinación y yaw = orientación.

Otras matrices de transformación

ANGULOS 'ROLL' 'PITCH' Y 'YAW'

Desde {A} hasta {B} Girando alrededor de los ejes de {A}

ROLL = BALANCEO
 γ

Giro en Eje X

PITCH = INCLINACION
 β

Giro en Eje Y

YAW = ORIENTACIÓN
 α

Giro en Eje Z

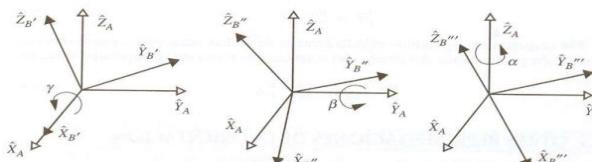


Figura 3.18: Ángulos RPY.

$${}^A_R R_{RPY} = \begin{bmatrix} (\cos\alpha\cos\beta) & (\cos\beta\sin\alpha - \cos\alpha\sin\beta) & (\cos\alpha\sin\beta + \sin\alpha\sin\beta) \\ (\sin\alpha\cos\beta) & (\sin\beta\sin\alpha + \cos\alpha\cos\beta) & (\sin\alpha\sin\beta - \cos\alpha\cos\beta) \\ -\sin\beta & c\beta s\gamma & c\beta c\gamma \end{bmatrix}$$

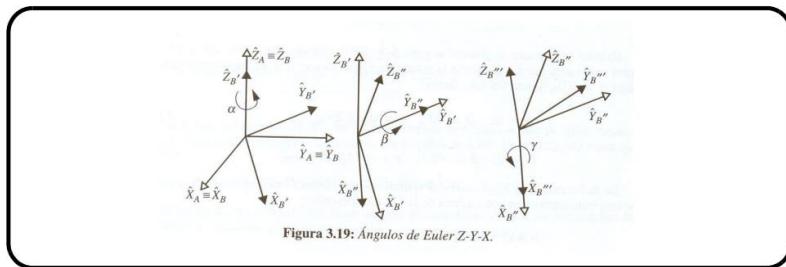
Otras matrices de transformación: Ángulos de Euler:

Otras matrices de transformación

ANGULOS DE EULER Z - Y - X

Desde {A} hasta {B} Girando alrededor de los ejes del sistema de referencia girado

Giro en Eje Z → Giro en Eje Y → Giro en Eje X



$${}^B_R_{zyx} = \begin{bmatrix} (\cos\beta) & (\sin\beta\sin\gamma) & (\sin\beta\cos\gamma) \\ (\sin\beta) & (\cos\beta\sin\gamma) & (\cos\beta\cos\gamma) \\ -\sin\gamma & \cos\gamma & \cos\gamma \end{bmatrix}$$

Otras matrices de transformación Interpretación de las matrices de transformación

ANGULOS DE EULER Z - Y - Z

Desde {A} hasta {B} Girando alrededor de los ejes del sistema de referencia girado

Giro en Eje Z → Giro en Eje Y → Giro en Eje Z

$${}^B_R_{zyz} = \begin{bmatrix} (\cos\beta\cos\gamma - \sin\beta\sin\gamma) & (-\cos\beta\sin\gamma - \sin\beta\cos\gamma) & (\cos\beta) \\ (\cos\beta\sin\gamma + \sin\beta\cos\gamma) & (-\sin\beta\sin\gamma + \cos\beta\cos\gamma) & (\sin\beta) \\ -\sin\gamma & \sin\beta\cos\gamma & \cos\beta \end{bmatrix}$$

Ejemplos de Robots Manipuladores:

3.1.2 Fundamentos de Robots Manipuladores

Ejemplos de Robot Manipuladores



VW vrs1 (VOLKSWAGEN)



KR3 (KUKA)



PUMA (Unimation)

Robots en el entorno industrial:

- Ambientes de trabajo peligrosos.
- Ciclos de trabajo repetitivos.
- Necesidad de precisión.
- Tareas de difícil realización por los humanos.
- Operaciones que se realizan 24h-día.
- Flexibles y reprogramables.
- Posible conexión con otros sistemas computadores.

Tareas en el entorno industrial:

- Paletización.
- Soldadura.
- Alimentación de máquina.
- Manipulación de productos.
- Pintado.
- Etiquetado.
- Pick and place.

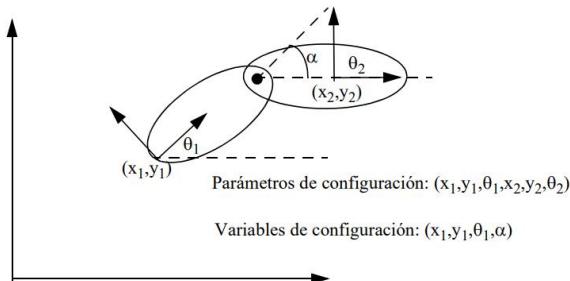


Diferentes Conceptos:

Configuración: (de un objeto arbitrario) la especificación de la posición de cada uno de los puntos del mismo. Usualmente, la configuración de un sólido rígido se realiza determinando la posición y orientación de cierto sistema de referencia solidario al mismo. Se denominan Parámetros de Configuración al conjunto de valores que describen las relaciones de traslación y orientación entre el sistema de referencia global y el sistema solidario al robot.

Se denominan **Variables Generalizadas o Variables de Configuración** al conjunto mínimo de magnitudes que permiten determinar la configuración del sistema. Ejemplo: 6 parámetros de configuración, 4 Variables Generalizadas.

Ejemplo: 6 parámetros de configuración, 4 Variables Generalizadas.

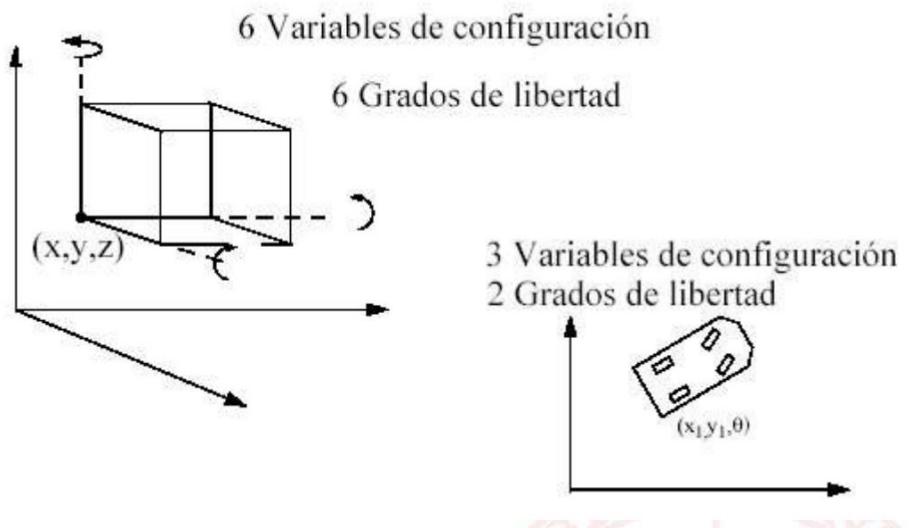


Espacio de Configuración: Es el espacio formado por todas las configuraciones posibles del robot.

Espacio de trabajo: Es el subconjunto del espacio de configuración que puede ser ocupado por el robot. Algunos autores denominan espacio del trabajo al **subconjunto del espacio cartesiano** que puede ser ocupado por el robot.

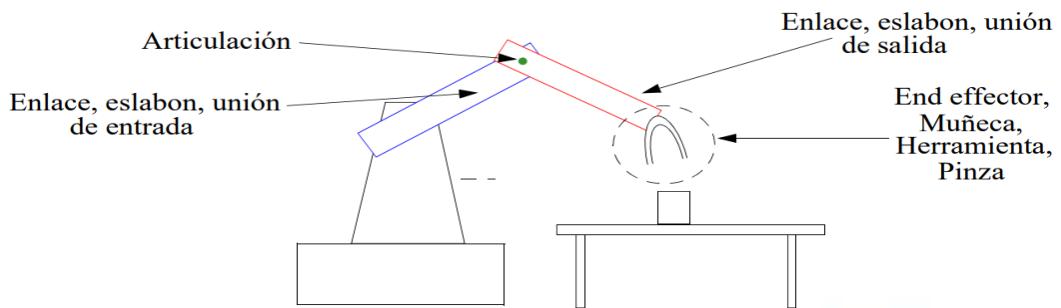
Variables de Configuración Vs Variables de estado: Las variables de estado representan el menor conjunto posible de magnitudes que permiten describir la situación actual y la evolución futura del sistema. Si se está realizando un estudio cinemático del robot, las variables de estado coinciden con las variables de configuración. Esto no es cierto si se considera el modelo dinámico.

Grados de Libertad: Número de variables de configuración que pueden cambiarse de forma independiente.



CARACTERÍSTICAS Y MORFOLOGÍA

Manipulador: mecanismo formado por elementos articulados entre sí, destinado al agarre, desplazamiento y manipulación de objetos.



Desde un punto de vista formal se trata de una cadena cinemática abierta formada por un conjunto de eslabones o elementos de la cadena interrelacionados mediante articulaciones. Las articulaciones permiten el movimiento relativo entre los sucesivos eslabones.

Principales características de un Manipulador Grados de Libertad:

- Parámetros que permiten establecer la orientación y posición del elemento final del manipulador.

b) Posibles movimientos básicos independientes.

Zonas de trabajo: Hay que distinguir distintas zonas: espacio alcanzable, área de acceso con orientación de la herramienta final etc..

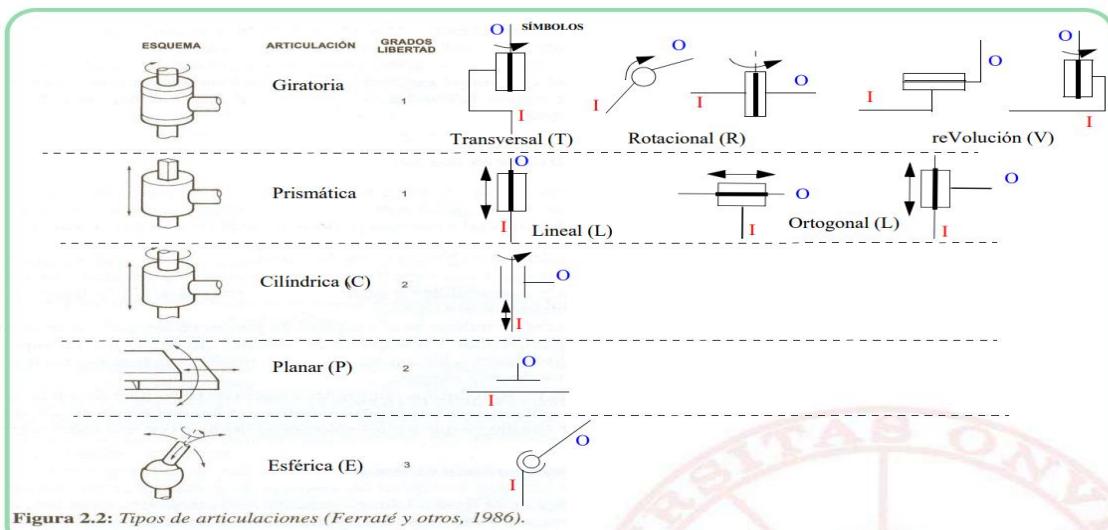
Capacidad de Carga: Peso en Kilogramos que puede transportar la garra del manipulador.

Precisión en la Repetitividad: Grado de exactitud en la repetición de los movimientos en una tarea programada. Ejemplo: Puma 550=0.1 mm

Velocidad lineal: velocidad de movimiento del elemento final. Ejemplo: Puma 550 alcanza 0,5 m/s

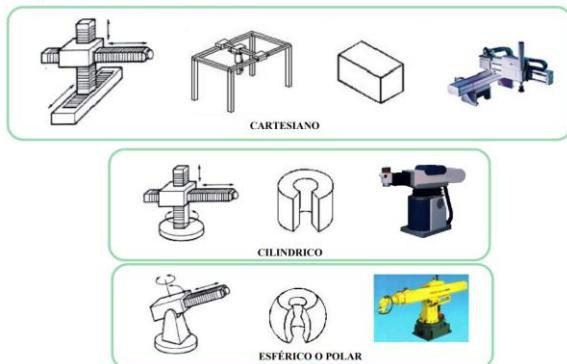
Tipos de articulaciones:

Tipo de articulaciones

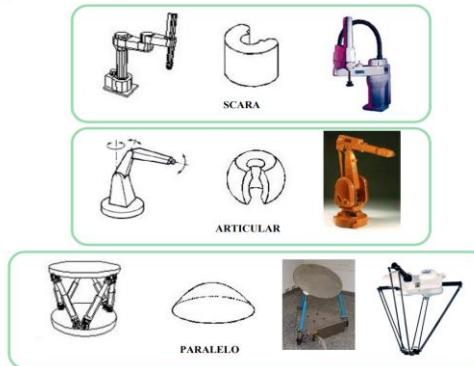


Configuraciones cinemáticas:

Configuraciones cinemáticas I



Configuraciones cinemáticas II



Notación de la configuración de un Manipulador:

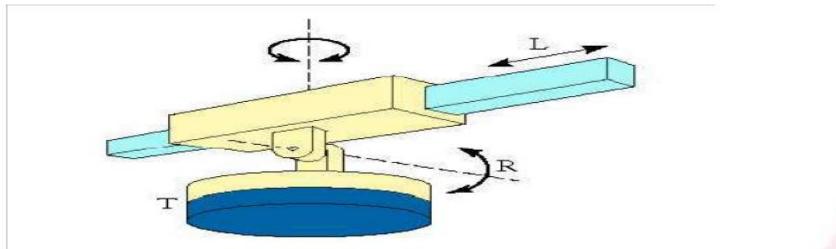
Es posible usar los símbolos (L, O, R, T, V) para designar los tipos de articulaciones usadas para construir un manipulador.

Para separar el brazo robótico de la muñeca se utiliza (:) TLR : TR

Ejemplo: Configuración Polar TRL Consiste de un brazo con desplazamiento (L) que puede rotar respecto ejes verticales (T) y horizontales (R).

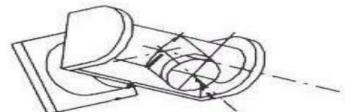
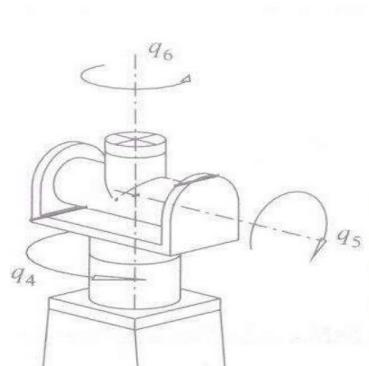
Ejemplo: Configuración Polar TRL

Consiste de un brazo con desplazamiento (L) que puede rotar respecto ejes verticales (T) y horizontales (R)



Configuración de la muñeca:

Configuración de la muñeca



- * Localizada al final del brazo
- * El brazo robótico determina la posición de la muñeca
- * El efecto instalado al final de la muñeca
- * La función es orientar el efecto final
- * Dos o tres grados de libertad

Ejemplo: TRT

Efecto final:

Herramienta especial que permite realizar una tarea específica Dos tipos: Pinzas: Coger y manipular objetos Herramientas: Realizar un proceso pintar, soldar, etc.

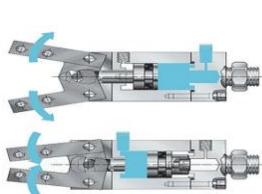
Efecto final

Herramienta especial que permite realizar una tarea específica

Dos tipos:

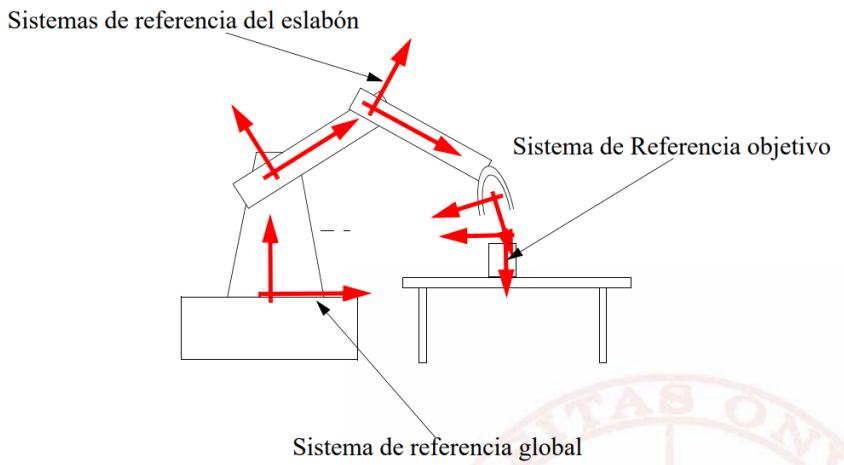
Pinzas: Coger y manipular objetos

Herramientas: Realizar un proceso pintar, soldar, etc.



Introducción: Problemas Geométricos:

3.2.1 Introducción: Problemas Geométricos



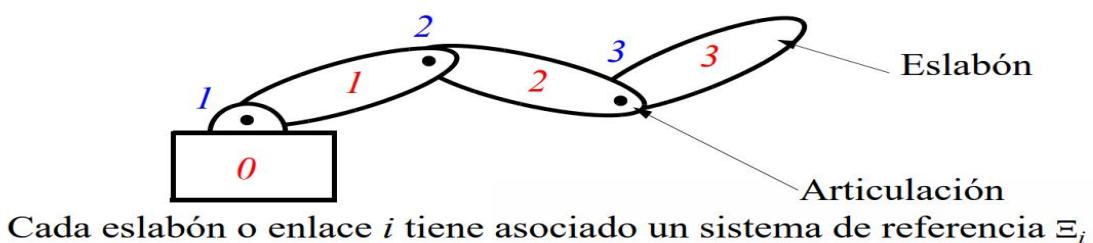
Modelo Cinemático Directo: Parámetros Denavit-Hattemberg.

- **Modelo Cinemático Directo:** Consiste en determinar la posición y orientación del extremo final del robot con respecto al sistema de la base del robot a partir de conocer los valores de las articulaciones y los parámetros geométricos.
- **Parámetros Denavit-Hattemberg:** Las articulaciones se numeran empezando desde 1. Los eslabones o enlaces se numeran desde 0 empezando por la base.

Parámetros Denavit-Hattemberg

Las articulaciones se numeran empezando desde 1

Los eslabones o enlaces se numeran desde 0 empezando por la base



Objetivo: Transformar las coordenadas dadas en Ξ_i en coordenadas en Ξ_{i-1}

Parámetros Denavit-Hattemberg:

La idea consiste en definir una serie de parámetros que permitan realizar siempre la misma serie de transformaciones para transformar las coordenadas dadas en E_i en coordenadas en E_{i-1} :

- Translación a lo largo del eje Z_i
- Rotación a lo largo del eje Z_i
- Traslación a lo largo del eje X_{i-1}
- Rotación a lo largo del eje X_{i-1}

Parámetros Denavit-Hattemberg: Parámetros

Cuatro parámetros, dos relacionados con el tamaño y forma del eslabón y otros dos para describir la posición relativa de los eslabones.

Parámetros

Cuatro parámetros, dos relacionados con el tamaño y forma del eslabón y otros dos para describir la posición relativa de los eslabones.

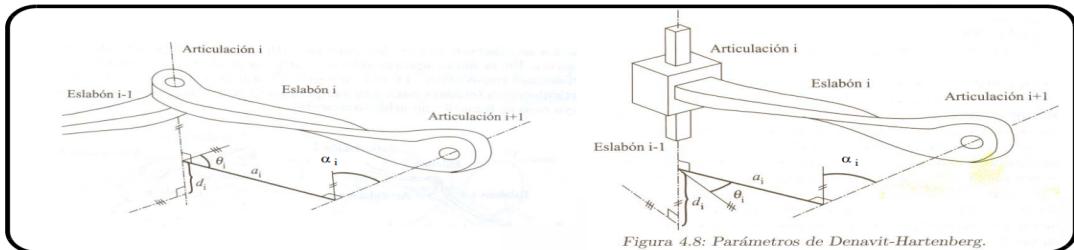


Figura 4.8: Parámetros de Denavit-Hartenberg.

θ_i .- **Variable articular:** ángulo formado entre las normales comunes a los ejes $i-1$ e $i+1$

d_i .- **Variable articular:** (distancia entre las intersecciones de las normales comunes a los ejes $i-1$ e $i+1$)+(desplazamiento de una articulación prismática)

a_i .- **Variable articular:** ángulo formado entre las normales comunes a los ejes $i-1$ e $i+1$

di .- **Variable articular:** (distancia entre las intersecciones de las normales comunes a los ejes $i-1$ e $i+1$)+(desplazamiento de una articulación prismática)

a_i .-distancia entre el eje de la articulación i y el eje de la articulación $i+1$ medida sobre la línea perpendicular común (si los ejes se cortan $a_i = 0$)

α_i .-ángulo entre los ejes de la articulación i e $i+1$ si estos se cortasen en los puntos de corte de la perpendicular común

Parámetros Denavit-Hattemberg: Elección de los sistemas de referencia

El eje Zi se hace coincidir con el eje de la articulación i . El origen de “ Ei ” se escoge en el punto de corte con la normal común a la articulación $i+1$. El eje Xi se hace coincidir con esa misma normal. el eje Yi se elige siguiendo la regla de la mano derecha,

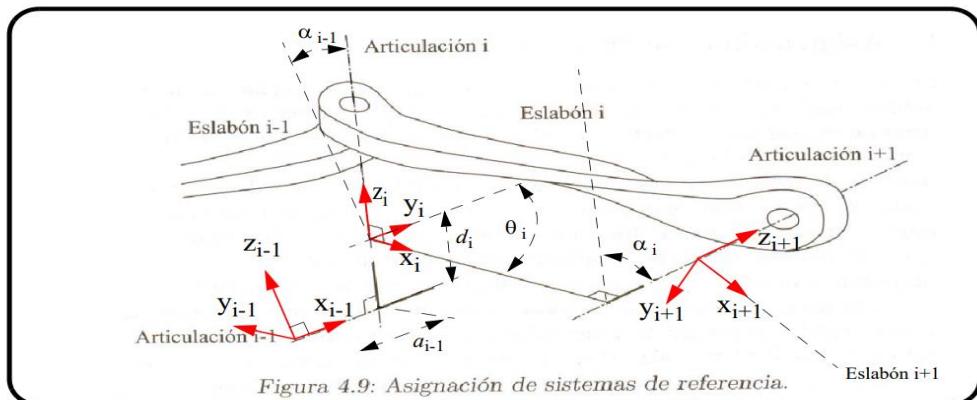
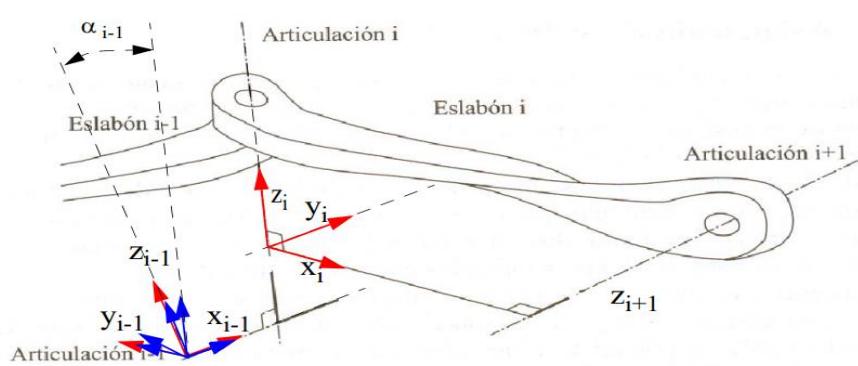
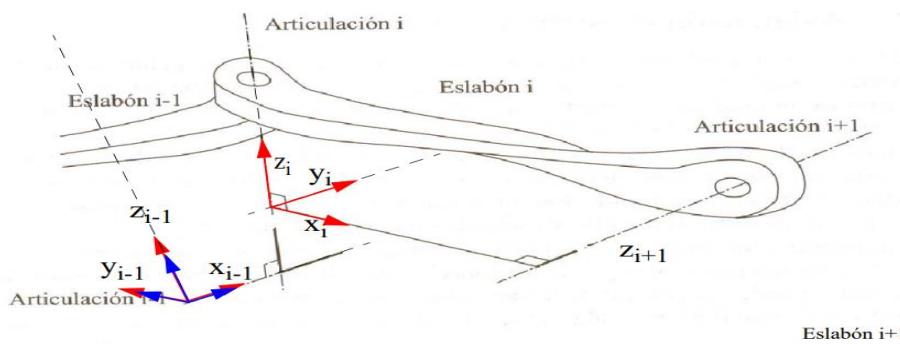
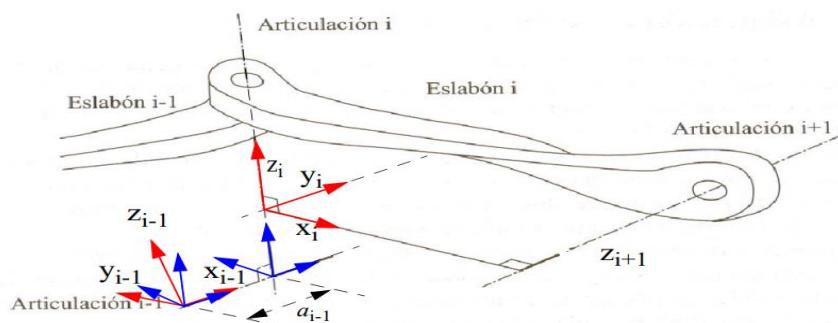


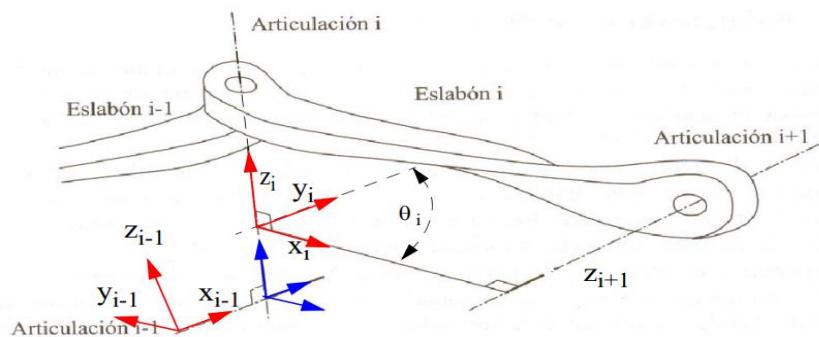
Figura 4.9: Asignación de sistemas de referencia.



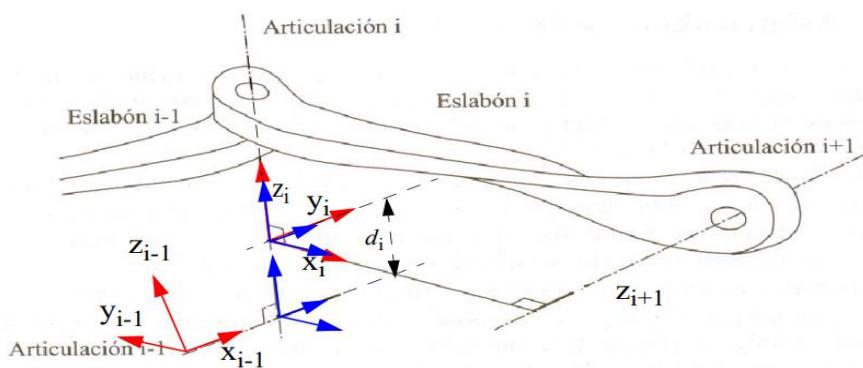
$$Rot(X_{i-1}, \alpha_{i-1})$$



$$Rot(X_{i-1}, \alpha_{i-1}) \cdot Tras(X_{i-1}, a_{i-1})$$



$${}^{i-1}_iT = Rot(X_{i-1}, \alpha_{i-1}) \cdot Tras(X_{i-1}, a_{i-1}) \cdot Rot(Z_p, \theta_i)$$



$${}^{i-1}_iT = Rot(X_{i-1}, \alpha_{i-1}) \cdot Tras(X_{i-1}, a_{i-1}) \cdot Rot(Z_p, \theta_i) \cdot Tras(Z_p, d_i)$$

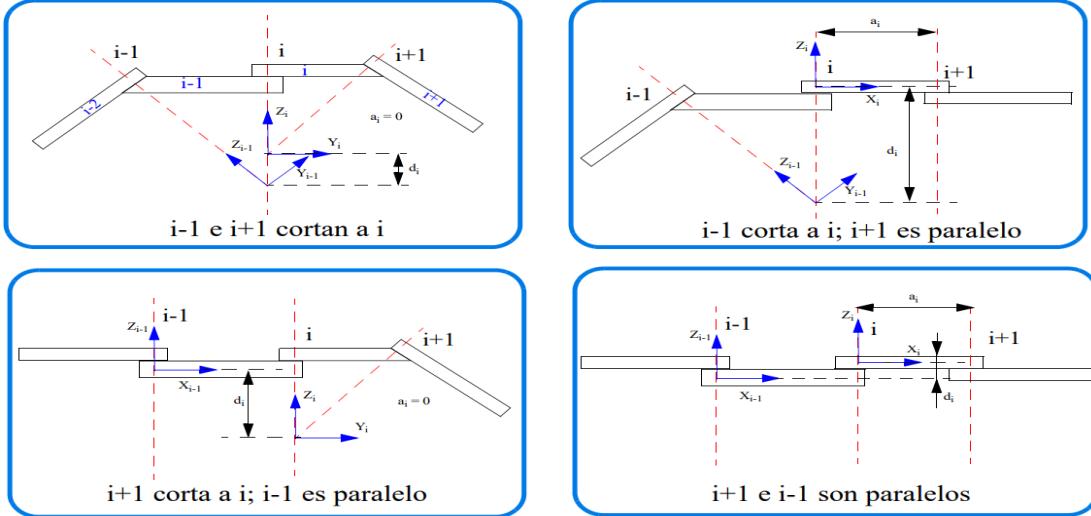
Casos especiales:

Si el eje i-1 o el i+1 cortan al eje i o son paralelos al mismo, hay que añadir las siguientes reglas:

- En cualquier caso Theta i mide el ángulo formado por los ejes Xi y el Xi-1.
- Si se produce un corte entre dos ejes, la distancia di se mide desde ese punto de corte.

- Si el eje i se corta con el eje i+1 el origen del sistema de referencia i se fija en ese punto de corte.
- Si el eje i se corta con el eje i+1, X_i se elige perpendicular al plano que definen dichos ejes
- Si los ejes son paralelos se escoge la perpendicular común que contiene el centro del eslabón i.
- Si los ejes coinciden, el origen del sistema de referencia se escoge en el extremo del enlace.

Casos Especiales



Procedimiento para determinar los parámetros Denavit-Hattemberg:

- 1º) Identificar los ejes
- 2º) Ubicar los orígenes de los sistemas de referencia
- 3º) Dibujar los ejes Z_i
- 4º) Dibujar los ejes X_i e Y_i
- 5) Escoger el sistema de referencia 0 (de manera que α_{01} , a_0 , θ_{01} y d_0 sean iguales a cero)
- 6º) El origen del último sistema de referencia se fija en el efecto final
- 7º) Identificar Parámetros: (para α_{01} y a_0 observamos las relaciones con el sistema de referencia i+1 y para θ_{01} y d_0 observamos las relaciones con el sistema de referencia i-1)

Matrices de Transformación:

Para representar i con respecto a i-1 se definen una serie de transformaciones en función de los parámetros considerados:

PROBLEMA CINEMÁTICO DIRECTO:

Dados los parámetros Denavit-Hartenberg de las n articulaciones de un manipulador, encontrar la posición y orientación del sistema de referencia objetivo en el espacio cartesiano

$${}^{i-1}_iT = \text{Rot}(X_{i-1}, \alpha_{i-1}) \cdot \text{Tras}(X_{i-1}, a_{i-1}) \cdot \text{Rot}(Z_i, \theta_i) \cdot \text{Tras}(Z_i, d_i)$$

PROBLEMA CINEMÁTICO DIRECTO

El problema cinemático directo consiste en:

Dados los parámetros Denavit-Hattemberg de las n articulaciones de un manipulador, encontrar la posición y orientación del sistema de referencia objetivo en el espacio cartesiano

$$P(q, d) = {}_1^0T \cdot {}_2^1T \cdots {}_{n-1}^nT = {}_n^0T$$

Problema cinemático inverso:

Cálculo de las variables articulares a partir de la posición y orientación deseadas para la herramienta final.

Orientación deseada
Posición deseada

$${}^0_nT = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} {}^0x_{extremo} \\ {}^0y_{extremo} \\ {}^0z_{extremo} \end{bmatrix} = {}_1^0T \cdot {}_2^1T \cdots {}_{n-1}^nT$$

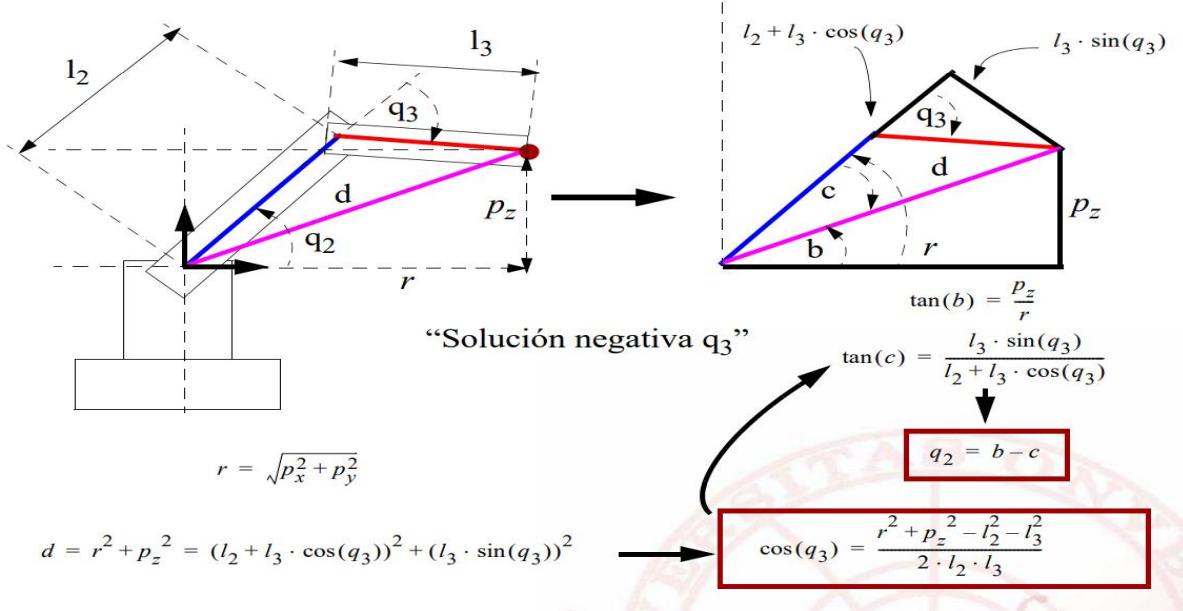
Ecuaciones no lineales:

- Soluciones numéricas
- Soluciones Algebraicas
- Soluciones Geométricas
- Solución Pieper

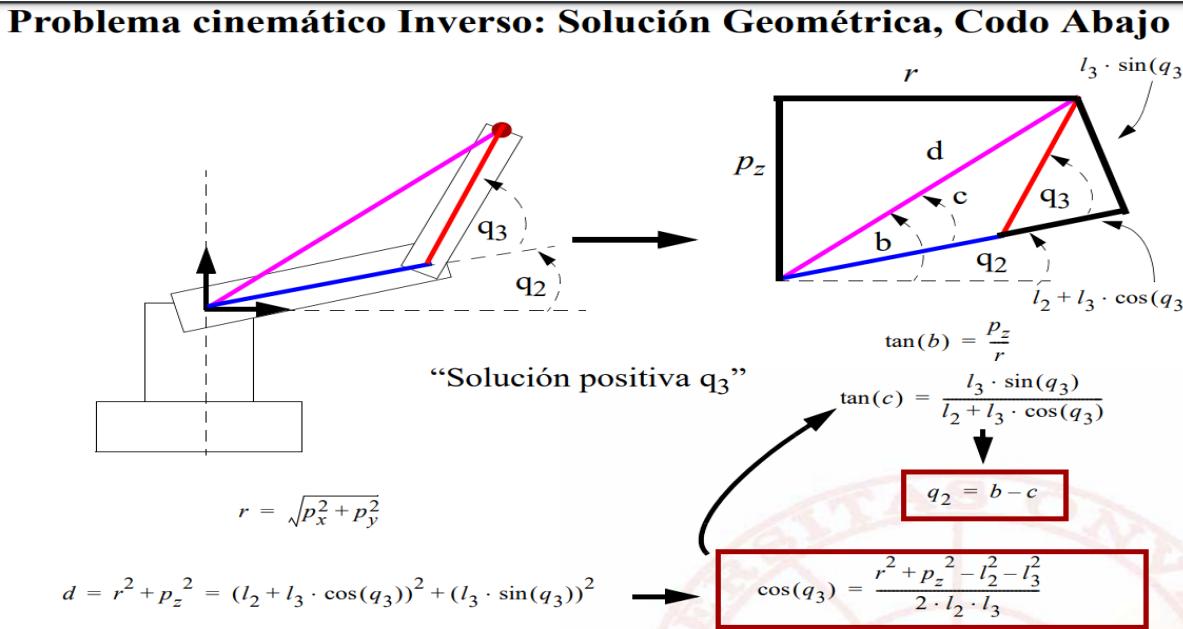
Ecuaciones no lineales:

- Soluciones numéricas
- Soluciones Algebraicas
- Soluciones Geométricas
- Solución Pieper

Problema cinemático Inverso: Solución Geométrica. Codo Arriba



Problema cinemático Inverso: Solución Geométrica. Codo Abajo

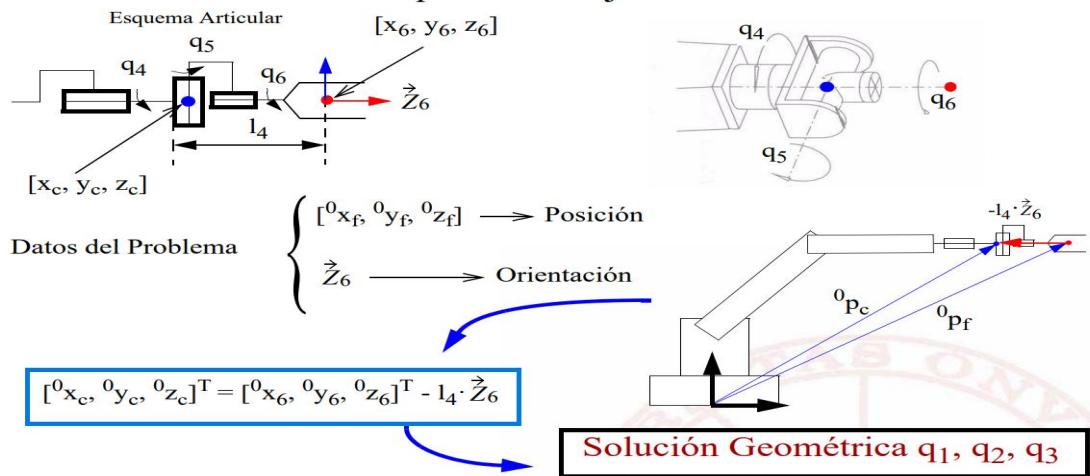


Problema cinemático Inverso: Solución Pieper-I

Caso de una muñeca en la que los tres ejes se cortan

Problema cinemático Inverso: Solución Pieper-I

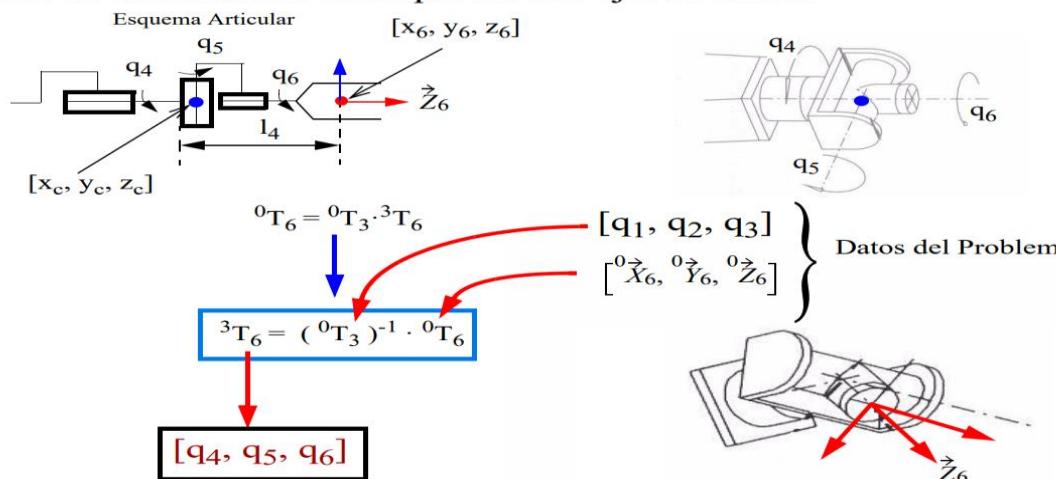
Caso de una muñeca en la que los tres ejes se cortan



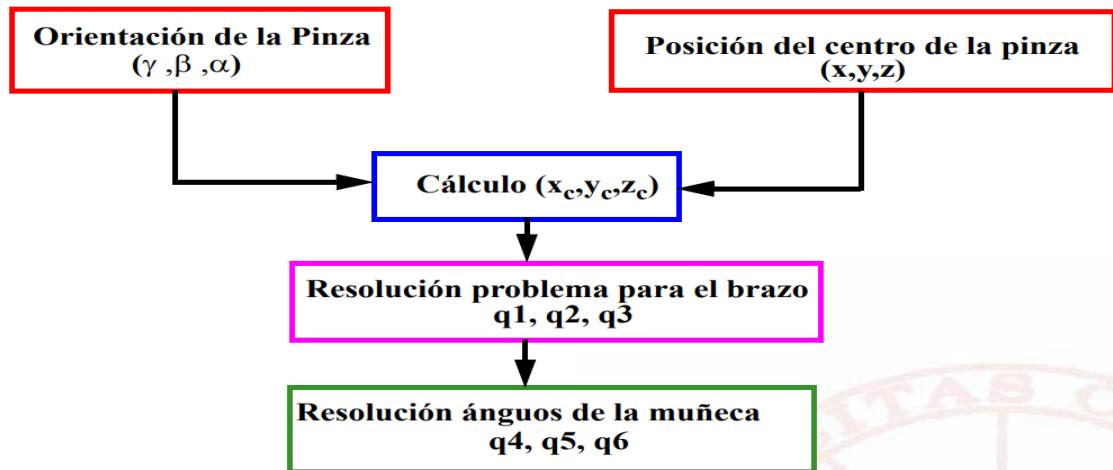
Problema cinemático Inverso: Solución Pieper-II

Problema cinemático Inverso: Solución Pieper-II

Caso de una muñeca en la que los tres ejes se cortan



Problema cinemático Inverso: Solución Pieper



Cálculo del Jacobiano:

Problema Directo

Conocer la velocidad del efecto final conocidas las velocidades articulares.



J → **Jacobiano**

¿Cómo se obtiene? -> Derivando

Problema cinemático Directo

Problema cinemático Directo

$$\left. \begin{array}{l} P_E = (x_E, y_E, z_E) \\ \theta_E = (\alpha, \phi, \psi) \end{array} \right\} \quad \begin{bmatrix} P_E \\ \theta_E \end{bmatrix} = G(q_1, q_2, \dots q_n)$$

Derivando

$$\begin{bmatrix} P'_E \\ \theta'_E \end{bmatrix} = J(q_1, q_2, \dots q_n) \cdot \begin{bmatrix} q'_1 \\ \dots \\ q'_n \end{bmatrix}$$

Problema Inverso

Conocer las velocidades articulares dada la velocidad del efecto final -> **Inversa**



$$\begin{bmatrix} q'_1 \\ \dots \\ q'_n \end{bmatrix} = J^{-1}(q_1, q_2, \dots q_n) \cdot \begin{bmatrix} P'_E \\ \theta'_E \end{bmatrix}$$

Singularidades:

Son aquellas configuraciones donde no es posible encontrar la inversa del manipulador. Coincidien con aquellas en las que el determinante del Jacobiano se hace igual a cero.

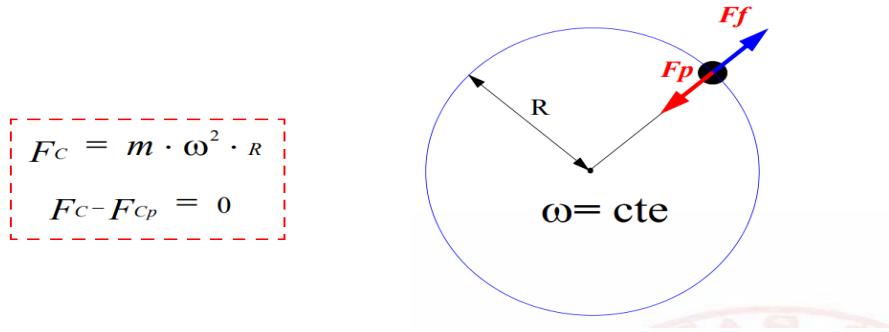
$$\text{Det}[J(q_1, q_2, \dots q_n)] = 0$$

Representan situaciones en las que hay direcciones en los que no es posible mover el efecto final (límites del espacio de trabajo o alineación de ejes articulaciones) Ejemplo:

PROBLEMAS DINÁMICOS:

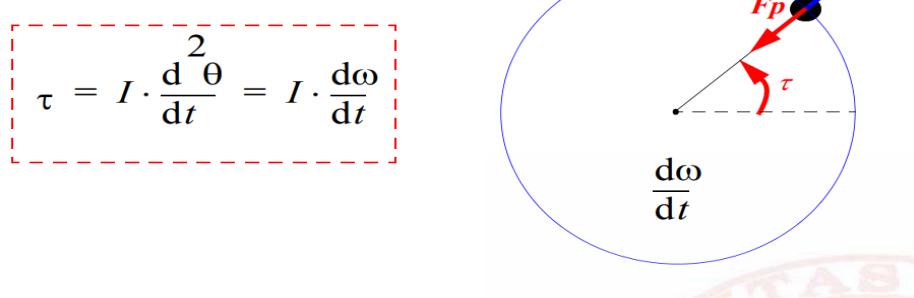
Principios Físicos: Fuerza Centrífuga vs Centrípeta

Principios Físicos: Fuerza Céntrifuga vs Centrípeta



Principios Físicos: Pares aplicados

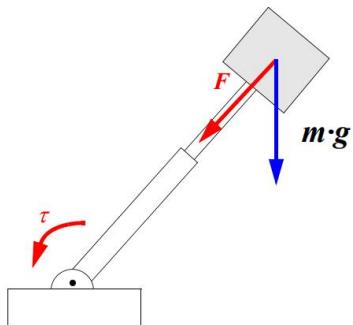
Principios Físicos: Pares aplicados



Modelo Dinámico de un Manipulador.

Modelo Dinámico: Ejemplo de Cálculo del Modelo

Modelo Dinámico: Ejemplo de Cálculo del Modelo



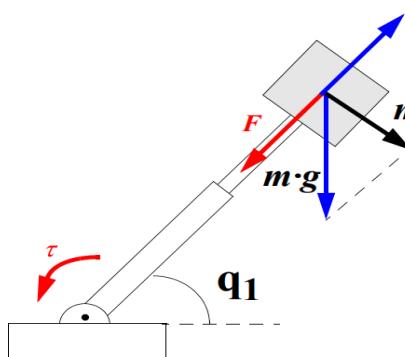
$$\sum \tau = I \cdot \frac{d^2 \theta}{dt^2}$$

$$\sum F = m \cdot \frac{d^2 x}{dt^2}$$

Vamos a considerar que la articulación telescópica permanece con una extensión constante

Vamos a considerar que la articulación telescópica permanece con una extensión constante.

Modelo Dinámico: Ejemplo de Cálculo del Modelo

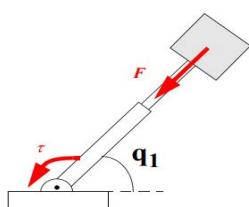


$$\tau - l \cdot m \cdot g \cdot \cos(q_1) = I \cdot \frac{d^2 q_1}{dt^2}$$

$$F + m \cdot g \cdot \sin(q_1) - l \cdot m \cdot q_1^2 = 0$$

OBJETIVO: Encontrar una expresión que proporcione los pares y fuerzas a aplicar a partir de las aceleraciones y velocidades articulares.

Modelo Dinámico: Ejemplo de Cálculo del Modelo



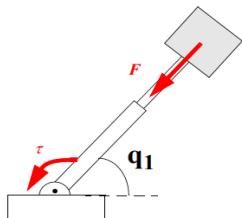
OBJETIVO: Encontrar una expresión que proporcione los pares y fuerzas a aplicar a partir de las aceleraciones y velocidades articulares

$$\begin{bmatrix} \tau \\ F \end{bmatrix} = \underbrace{\begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix}}_{\text{Matriz de Inercia}} \cdot \begin{bmatrix} \ddot{q}_1 \\ \ddot{d} \end{bmatrix} + \underbrace{\begin{bmatrix} 0 & 0 \\ -l \cdot m \cdot q_1^2 & 0 \end{bmatrix}}_{\text{Matriz Centrífuga y de Coriolis}} \cdot \begin{bmatrix} \dot{q}_1 \\ \dot{d}_1 \end{bmatrix} + \underbrace{\begin{bmatrix} -l \cdot m \cdot g \cdot \cos(q_1) \\ m \cdot g \cdot \sin(q_1) \end{bmatrix}}_{\text{Matriz de gravedad}}$$

Simulación de la Dinámica.

OBJETIVO: Encontrar una expresión que proporcione las aceleraciones dados los pares y fuerzas aplicadas y conocidas las velocidades y posiciones articulares.

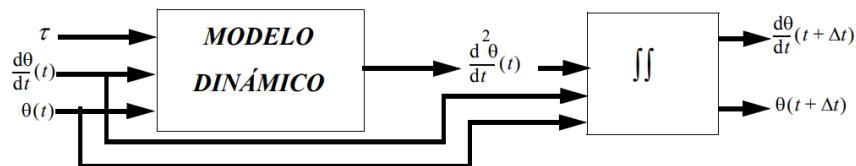
Simulación de la Dinámica



OBJETIVO: Encontrar una expresión que proporcione las aceleraciones dados los pares y fuerzas aplicadas y conocidas las velocidades y posiciones articulares

$$\begin{bmatrix} \ddot{q}_1 \\ \ddot{\theta} \end{bmatrix} = [I]^{-1} \cdot \begin{bmatrix} \tau \\ F \end{bmatrix} - \begin{bmatrix} 0 & 0 \\ -l \cdot m \cdot q_1^2 & 0 \end{bmatrix} \cdot \begin{bmatrix} \dot{q}_1 \\ \dot{\theta} \end{bmatrix} - \begin{bmatrix} -l \cdot m \cdot g \cdot \cos(q_1) \\ m \cdot g \cdot \sin(q_1) \end{bmatrix}$$

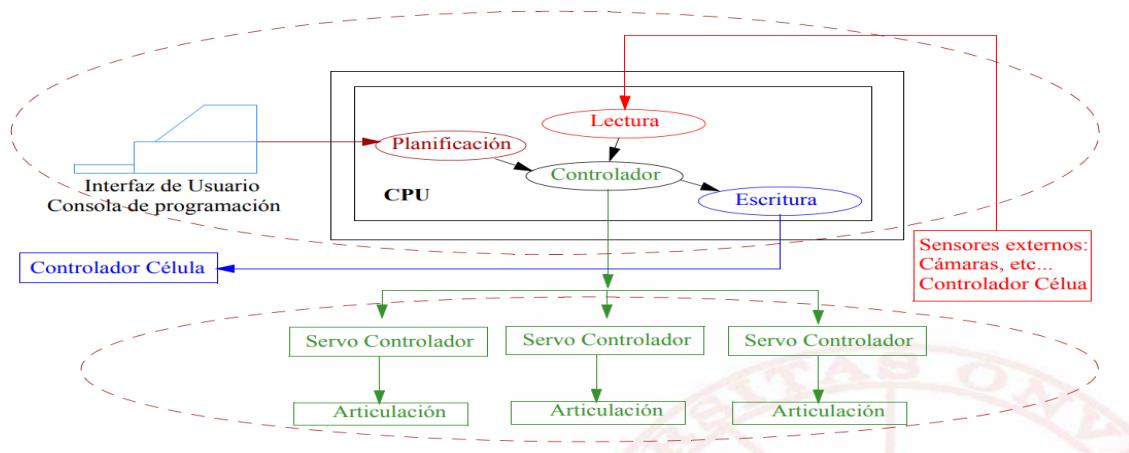
Simulación de la Dinámica



ESTRATEGIAS DE CONTROL

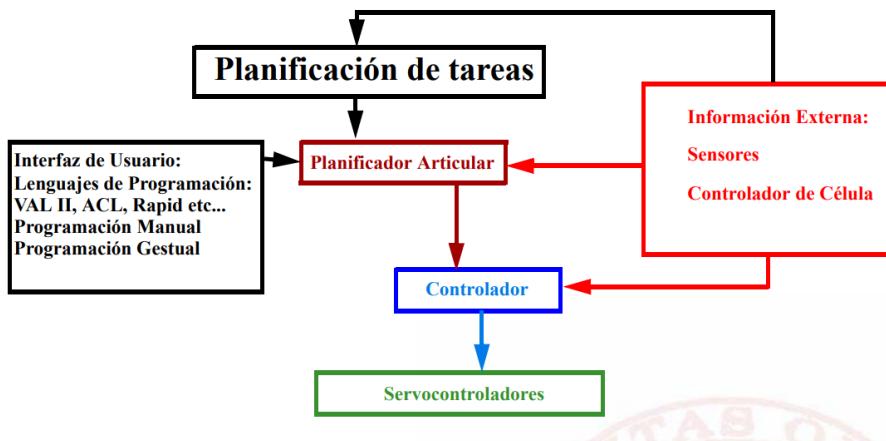
Arquitectura Manipulador Industrial

Arquitectura Manipulador Industrial



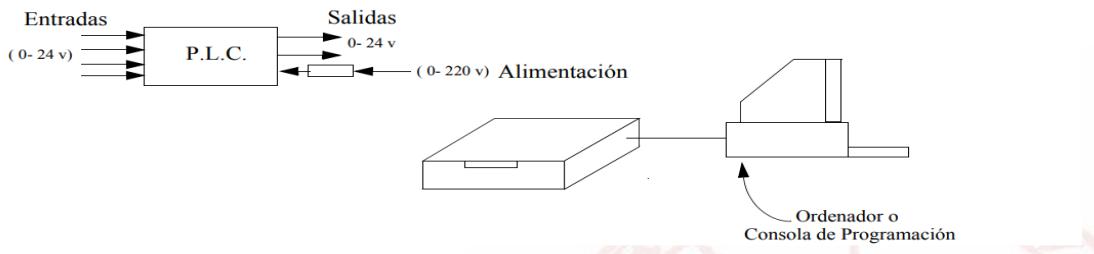
Jerarquía de control -> Planificación tareas, planificador articular....

Jerarquía de Control



Control de Células de Producción: PLC -> Máquina electrónica programable por personal no informático, destinada a cumplir en un ambiente industrial y en tiempo real funciones de automatismos lógicos, combinatorios y secuenciales.

PLC: máquina electrónica programable por personal no informático, destinada a cumplir en un ambiente industrial y en tiempo real funciones de automatismos lógicos, combinatorios y secuenciales.



Programación

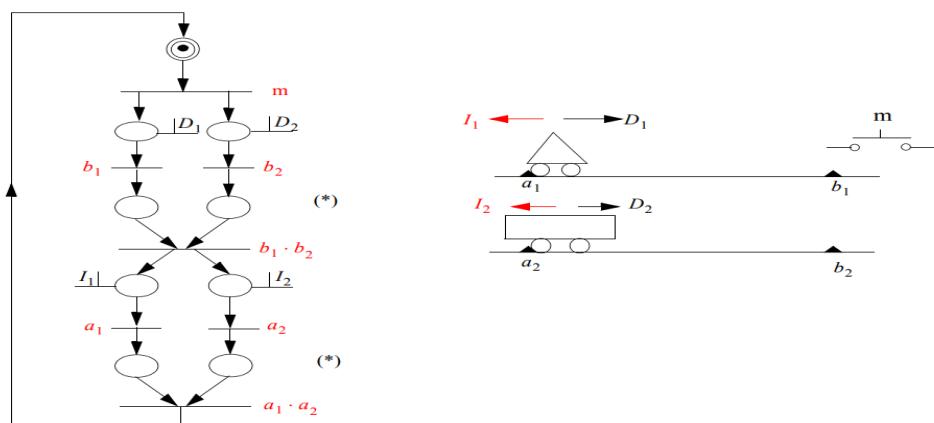
Programación concurrente: Redes de Petry, Grafcet

Programación

Programación Concurrente: Redes de Petry, Grafset

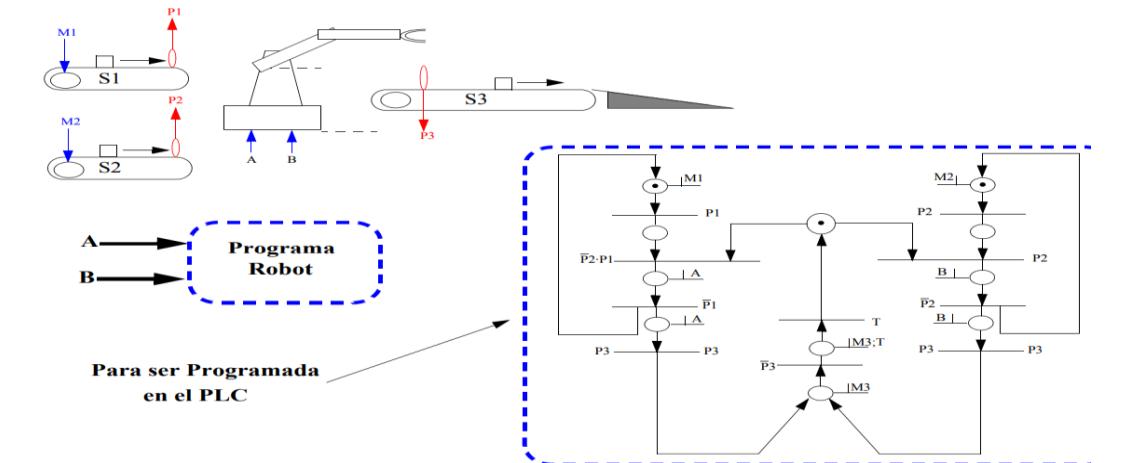
1

RdP



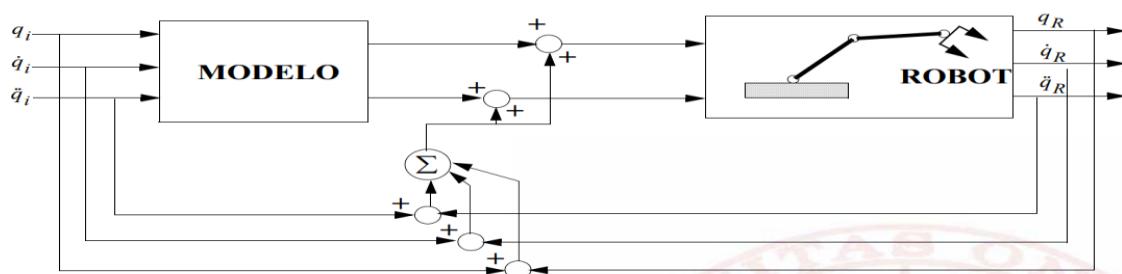
Ejemplo Control de Célula: Recurso Compartido:

Ejemplo Control de Célula: Recurso Compartido:



Estrategia de Control Articular:

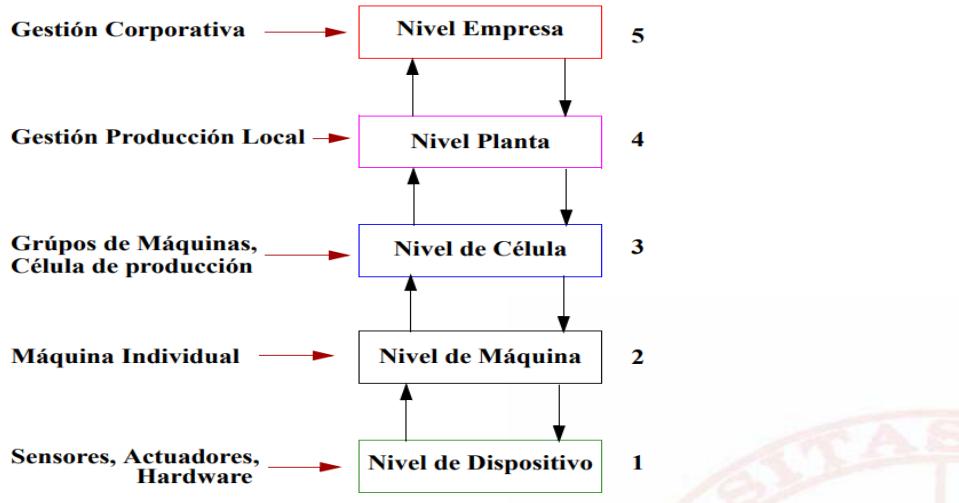
Estrategias de Control Articular



PROGRAMACIÓN DE MANIPULADORES:

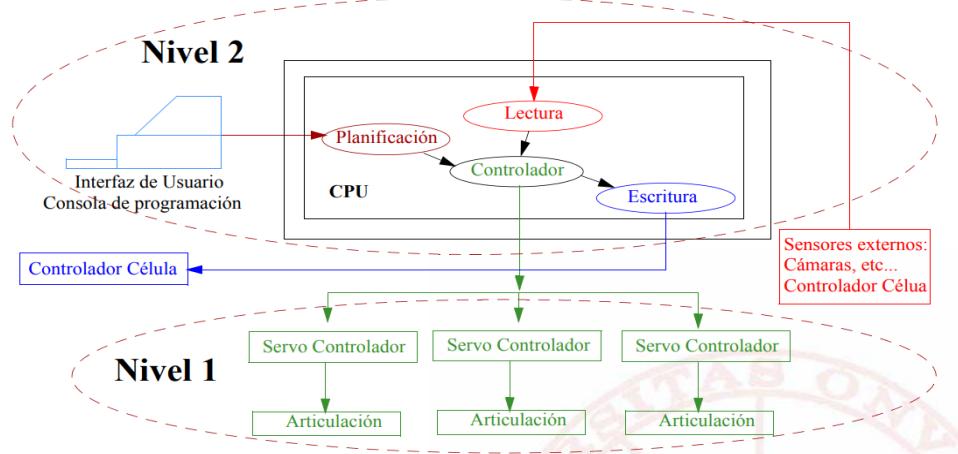
Niveles de Automatización y Células de Producción.

Niveles de Automatización y Células de Producción



Arquitectura Manipulador Industrial: Niveles 2 y 1

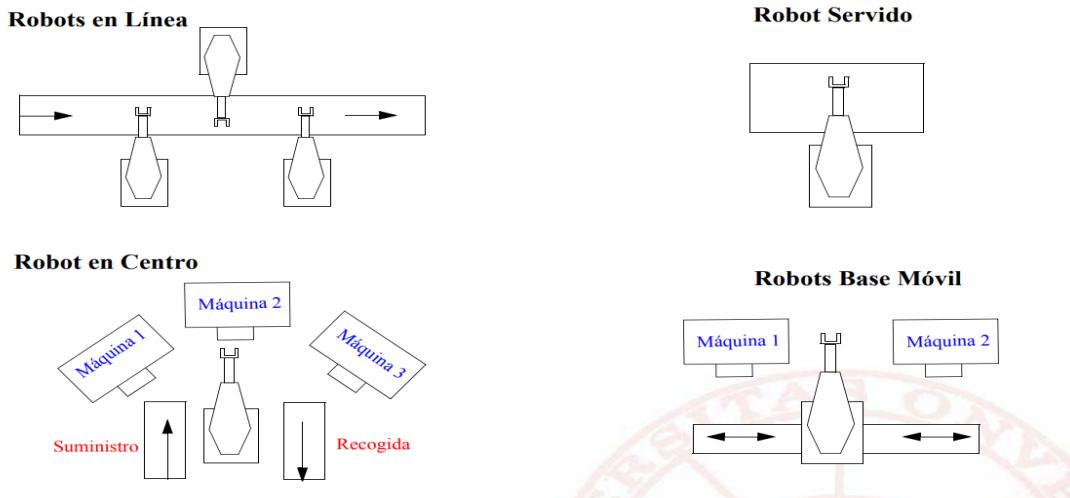
Arquitectura Manipulador Industrial: Niveles 2 y 1)



El manipulador dentro de la Célula de Producción.

El Manipulador dentro de la Célula de Producción: Nivel 3.

El Manipulador dentro de la Célula de Producción: Nivel 3



Programación de Robots Industriales.

Programación de Robots Industriales

Es posible hablar de diferentes niveles de programación.

- **Programación a bajo nivel:** El programa maneja directamente los controladores de las articulaciones accediendo a los valores de los sensores y actuadores de cada articulación.
- **Programación a nivel articular:** El control del movimiento está delegado en elementos incorporados a la arquitectura de control, reposando en el programa la responsabilidad de planificar las trayectorias en el espacio de trabajo o en el articular
- **Programación de nivel superior:** El sistema de control provee de funciones que permiten que el programa se dedique a especificar la tarea a realizar de una forma más o menos abstracta.

Programación articular y de bajo nivel:

Es habitual diferenciar entre el control de las articulaciones y la definición de la trayectoria articular.

Es necesario establecer cuáles son los objetivos de manipulación definidos por la tarea y traducir dichos objetivos al espacio articular, aplicando las técnicas de cálculo del problema cinemático inverso o del inverso del jacobiano.

Con esta información, el sistema de control establecerá las acciones necesarias para que los actuadores hagan que las articulaciones adopten las configuraciones necesarias.

Lenguajes de programación de programación de dispositivos electrónicos basados en microprocesadores, tales como ensamblador, C, Java etc.

Programación de alto nivel:

La programación de alto nivel en robot industriales consiste en indicar paso por paso las diferentes acciones (moverse a un punto, adoptar una configuración, abrir o cerrar la pinza, etc.) que éste deberá realizar durante su funcionamiento.

Actualmente no existe normalización en relación a los procedimientos de programación de robots, cada fabricante desarrolla su método particular, el cual es válido solamente para sus propios robots.

Existen varios criterios para clasificar los métodos de programación. Según el sistema utilizado para indicar la secuencia de acciones:

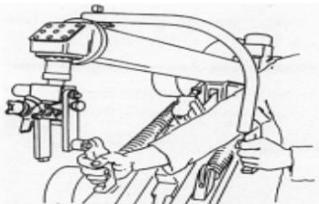
- Programación por guiado.
- Programación Textual

1) Programación por Guiado

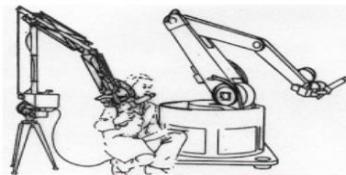
La programación por guiado o aprendizaje consiste en hacer realizar al robot, o a una maqueta del mismo, la tarea, registrando las configuraciones adoptadas para su posterior repetición en forma automática.

Programación por Guiado

La programación por guiado o aprendizaje consiste en hacer realizar al robot, o a una maqueta del mismo, la tarea, registrando las configuraciones adoptadas para su posterior repetición en forma automática.



Pasivo Directo



Pasivo Indirecto

Guiado activo: Esta posibilidad permite emplear el propio sistema de accionamiento del robot, controlado desde una botonera joystick para que sea éste el que mueva sus articulaciones.

Existen dos formas básicas de registro de los movimientos:

Registro en puntos de paso: El robot es guiado por los puntos por los cuales se desea que pase durante la fase de ejecución automática del programa. Durante el guiado, se registran dichos puntos y la unidad de control establecen las trayectorias que interpolan dichos puntos.

Registro continuo: Se registran con una frecuencia de muestreo fija los movimientos de guiado.

2) Programación Textual

Este método de programación permite indicar la tarea al robot a través de un lenguaje de programación específico. Un programa se entiende como una serie de órdenes que son editadas y posteriormente ejecutadas, por lo tanto, existe un texto para el programa.

La programación textual se puede clasificar en tres niveles:

- Nivel robot: las órdenes se refieren a los movimientos a realizar por el robot.
- Nivel objeto: las órdenes se refieren al estado en que deben ir quedando los objetos.
- Nivel tarea: las órdenes se refieren al objetivo a conseguir.

Actualmente, la mayoría de los lenguajes de programación son de nivel robot, entre los que destacan por orden cronológico los siguientes:

- AL (Universidad de Stanford - 1974)

- AML (IBM - 1979)
- LM (Universidad de Grenoble - 1981)
- VAL II (Unimation-1984)
- V+(ADEPT - 1989)
- RAPID (ABB - 1994)

A nivel destacan los siguientes ejemplos:

- LAMA (MIT - 1976)
- AUTOPASS (IBM - 1977)
- RAPT (Universidad de Edimburgo- 1978)

Programación textual, una primera aproximación:

Nivel tarea: se especifica qué es lo que debe hacer el robot en lugar de cómo debe hacerlo.

Ensamblar A con D

Nivel Objeto: Las instrucciones se dan en función de los objetos a manejar:

- Situar B sobre D haciendo coincidir LADO_B1 con LADO_D1;
- Situar A dentro D haciendo coincidir EJE_A con EJE_HUECO_ y BASE_A con BASE_D;

Nivel Robot: Se debe especificar cada uno de los movimientos que ha de realizar el robot, como velocidad, direcciones de aproximación y salida, apertura y cierre de la pinza, etc.

- Mover_a P1 via P2; Situarse en un punto sobre la pieza B
- Vel = 0.2 * VELMAX; Reducir la velocidad
- Pinza = ABRIR; Abrir la pinza
- Prec = ALTA ; Aumentar la precisión
- Mover_recta_a P3; Descender verticalmente en línea recta
- Pinza = CERRAR ; Cerrar la pinza para coger la pieza B
- Espera= 0.5; Esperar para garantizar cierre de pinza
- Mover_recta_a P1; Ascender verticalmente en longa recta
- Prec = MEDIA; Decrementar la precisión
- Vel = VELMAX; Aumentar la velocidad
- mover_a P4 via P2; Situarse sobre la pieza C
- Prec = ALTA; Aumentar la precisión
- Vel = 0.2 * VELMAX ; Reducir velocidad
- Mover_recta_a P5; Descender verticalmente en línea recta
- Pinza = ABRIR; Abrir pinza

Requisitos para los sistemas de programación de robots

Tradicionalmente los requerimientos generales que se han establecido para un sistema de programación de robots son los siguientes:

- Entorno de Programación
- Modelado del Entorno
- Tipo de Datos
- Manejo de Entradas/Salidas (digital y análoga)
- Control del Movimiento del Robot
- Control del flujo de ejecución del programa

Programación en VAL II y V+.

El lenguaje VAL (Victor's Assembly Language), fue el primer lenguaje para robots comercialmente disponible.

VAL fue introducido en 1979 por Unimation, Inc. para su serie de robots PUMA. Este lenguaje forma parte de la primera generación de lenguajes de programación de robots.

VAL II es un lenguaje y sistema de control basado en computador diseñado para los robots industriales Unimation. VAL II, reemplazó a en 1984.

Este lenguaje, forma parte de la segunda generación de lenguajes de programación de robots.

Programación en VAL II y V+.

Los lenguajes de la segunda generación salvan algunas limitaciones de los lenguajes de la primera generación y las añaden a estas nuevas capacidades.

A estos lenguajes de la segunda generación, se les ha llamado lenguajes de programación estructurada, porque poseen las construcciones de control estructuradas utilizadas en los lenguajes de programación de la computadora

El lenguaje **V+** es un lenguaje de programación textual de alto nivel, desarrollo en 1989 por **Adept Technology**. Es una evolución del VAL II, y a veces se denomina indistintamente V+ y VAL III.

Programación en VAL II y V+.

Un programa en estos lenguajes consiste en un conjunto de instrucciones secuenciadas en líneas de programa. El formato general de cada línea es:

número_linea <etiqueta> <instrucción> <;comentario>

La etiqueta es un número entero que permite identificar un lugar en el código para poder ser direccionado con sentencias GOTO.

La primera línea de un programa es la instrucción ".PROGRAM" seguida del nombre del programa y de los parámetros que deba recibir o devolver.

.PROGRAM nombre <(lista de parámetros)>

El final del programa se indica con una línea que contiene la instrucción "**.END**"

VARIABLES

Respecto a su ámbito, es posible definir variables **globales, locales o automáticas**.

Variables globales, son aquella cuyo valor que pueden ser accedida por cualquier de los programas que se encuentren en memoria.

Variables locales se definen mediante la **instrucción LOCAL**. Estas variables solo pueden ser utilizadas en el ámbito del programa o rutina donde han sido definidas. Mantienen su valor entre llamadas al programa.

Variables automáticas. Son parecidas a las variables locales, pero cada vez que se entra al programa, se crea una copia separada de las existentes hasta ese momento. El valor de cada copia se pierde cada vez que se sale del programa. Estas últimas, se crean con la **instrucción AUTO**.

Respecto de uso, existen dos tipos de variables: las **variables de punto, y las variables reales**.

Dentro de las variables de punto podemos distinguir dos tipos: **puntos de precisión y localizaciones en el espacio**.

Un **punto de precisión** es un punto en el espacio articular del robot. Se declara añadiendo # delante del nombre del punto. Se puede definir utilizando la **función PPOINT**:

#lugar = PPOINT (a , b, c, d, e, f)

También es posible establecer un punto de precisión con el **comando Here**:

HERE #lugar

Un **punto de localización** en el espacio consta de una tupla de seis parámetros (X, Y, Z, y, p, r), donde X, Y, Z representan las coordenadas cartesianas absolutas en milímetros; y donde y (yaw), p (pitch) y r (roll) representan los ángulos de Euler ZYX.

La **sentencia SET-TRANS** se utiliza para definir una variable, punto de localización.

Por ejemplo, una posición a la que llamaremos garra1 se define en un programa de la siguiente manera:

SET garra1 = TRANS (123.789, -488.511, 755, 0, 180, -61.87)

Se recomienda la consulta de la bibliografía Ollero, (2001), Capítulo 11, para conocer más detalles sobre la definición de transformaciones compuestas y especificación de localizaciones.

En cuanto a las **variables reales**, su definición se realiza mediante una **asignación a un nombre cualquiera de un valor entero o real**.

Escribir A=12 declara y asigna la variable A con el valor 12. En el caso de VAL II, el valor debe estar comprendido obligatoriamente entre 6.0E-39 y 2.854E+38 para valores positivos y hasta -6.0E-39 para valores negativos.

Operaciones matemáticas con variables reales:

- +,-,*,/
- ABS(expr)
- SIGN(expr) da el signo de la expresión
- FRACT(expr) Da la parte fraccionaria de la expresión
- INT(expr) Da la parte entera de la expresión
- Etc...

SENTENCIAS DE CONFIGURACIÓN Y MOVIMIENTO

- **MOVE <VARIABLE DE PUNTO>**: Mueve al robot a la configuración descrita por la variable por medio de una interpolación articulada.
- **MOVES <VARIABLE DE PUNTO>**: Mueve al robot a la configuración descrita por la variable. La mano es movida en línea recta. La "S" indica "straight line"

- **DELAY:** Causa que el movimiento del robot pare por un periodo de tiempo específico
- **OPENI o CLOSEI:** Abre o cierra inmediatamente la pinza, asume un estado abierto o cerrado respectivamente.
- **APPRO <VARIABLE DE POSICION>, <VALOR>:** Mueve el robot a la posición indicada por la variable de posición, pero aumentando o disminuyendo la coordenada Z con el valor especificado. El movimiento se realiza mediante interpolación lineal
- **DEPART <VALOR>:** Movimiento de decremento de la coordenada Z con el valor especificado.
- **SPEED <VALOR> [ALWAYS]:** Establece la velocidad a la que queremos que se mueva el robot. <VALOR> indica un porcentaje de la velocidad máxima. Si se especifica el término ALWAYS, entonces mantendrá la velocidad hasta el final del programa o hasta que encuentre otra sentencia SPEED.

SENTENCIAS DE GESTIÓN DE SEÑALES EXTERNAS

- **SIG <CANAL>:** Devuelve true si la señal del canal está a ON.
- **WAIT SIG <CANAL>:** Para la ejecución del programa hasta que la señal del canal de entrada especificado esté en ON
- **SIGNAL <CANAL>:** Activa o desactiva las señales externas de salida específicas.
- **RESET:** Apaga todas las señales de salida.

SENTENCIAS DE CONTROL DE FLUJO

- **GOTO <ETIQUETA>:** Realiza un parte incondicional del programa identificado por ETIQUETA.
- **IF GOTO:** Salto condicional a la marca especificada.
- **CALL:** Llama a una subrutina.
- **RETURN:** Final de una Subrutina.
- **CALL <NOMBRE_DEL_PROGRAMA>:** Permite desde un programa llamar a otro <NOMBRE_DE_PROGRAMA>, de modo que se desvía el flujo de ejecución al programa indicado, retornando cuando haya finalizado.
- **PROMPT "<MENSAJE>,<variable>,<variable>,...:** Esta instrucción permite, a partir del teclado, entrar los valores que se asignan a las variables.
- **CASE:** Proceso iniciado con una estructura del CASE definiendo el valor de interés.
- **TYPE {/carácter de control},{variable},{etc.}:** Esta instrucción permite hacer salir por pantalla un mensaje durante la ejecución de un programa y/o hacer salir en pantalla el contenido de una o varias variables declaradas.
- **WHILE:** Inicia un proceso con la estructura WHILE si la condición es TRUE o salta la estructura del WHILE si la condición es inicialmente FALSA

EJEMPLOS DE PROGRAMAS

Estrategias para afrontar el diseño de un programa

1º Diseñar el flujo del programa

2º Diseñar los movimientos

3º Cálculo de las configuraciones que se quieren alcanzar

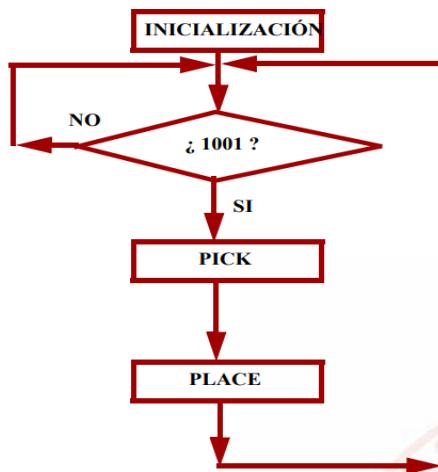
4º Escribir el código

5º Comprobar el funcionamiento

Ejemplo 1: Realizar un programa para realizar una tarea de pick and place de una pieza que será detectada mediante un sensor conectado al canal 1001. El manipulador debe coger la pieza situando la pinza en el punto (431,610,523) con unos ángulos de Euler ZXY (0,180,45) y debe depositar la pieza en el punto (227,-548,712) con unos ángulos de Euler ZXY (0,180,45).

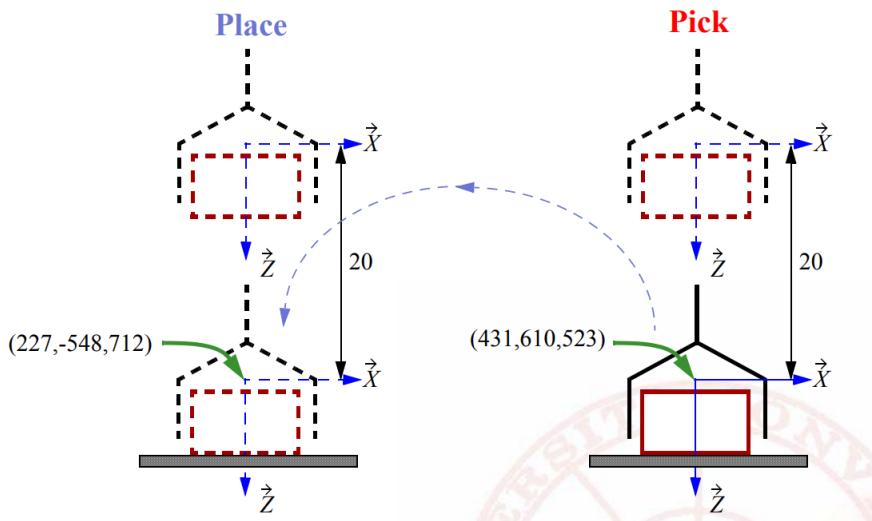
Planificando el flujo del programa:

PLANIFICANDO EL FLUJO DEL PROGRAMA



Planificando movimientos:

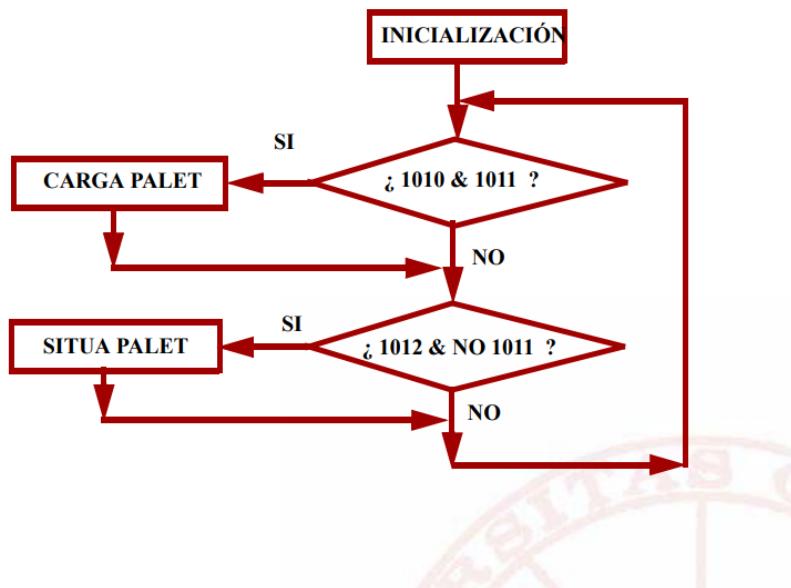
PLANIFICANDO MOVIMIENTOS



Ejemplo 2: Programar un manipulador para que cargue un palet con cajas que le son servidas. El palet estará cargado con un total de 3 cajas. Cada caja mide 100 mm de alto al igual que el palet sobre el que irán cargadas. Cuando haya una caja preparada para ser cargada el canal 1010 estará a nivel alto. Cuando haya un palet colocado para ser cargado el canal 1011 estará a nivel alto. Igualmente si no hay palet preparado para ser cargado, el manipulador deberá coger un palet de la una zona donde debe haber un palet de reserva. Si hay palet de reserva estará activado el canal 1012. Una vez el palet

ha sido cargado con las tres cajas, debe encenderse una luz que se activará con el canal de salida 10 y el manipulador quedar a la espera de que el palet sea retirado para volver a iniciar de nuevo el ciclo de carga. Las cajas se recogerán en la posición (431,610,100) y el palet estará situado en (227,-548,202). El palet vacío estará ubicado en (431,610,523,0,180,45). Las cajas y el palet se recogerán con la muñeca invertida.

Planificando flujo del programa:



Planificando movimientos:

