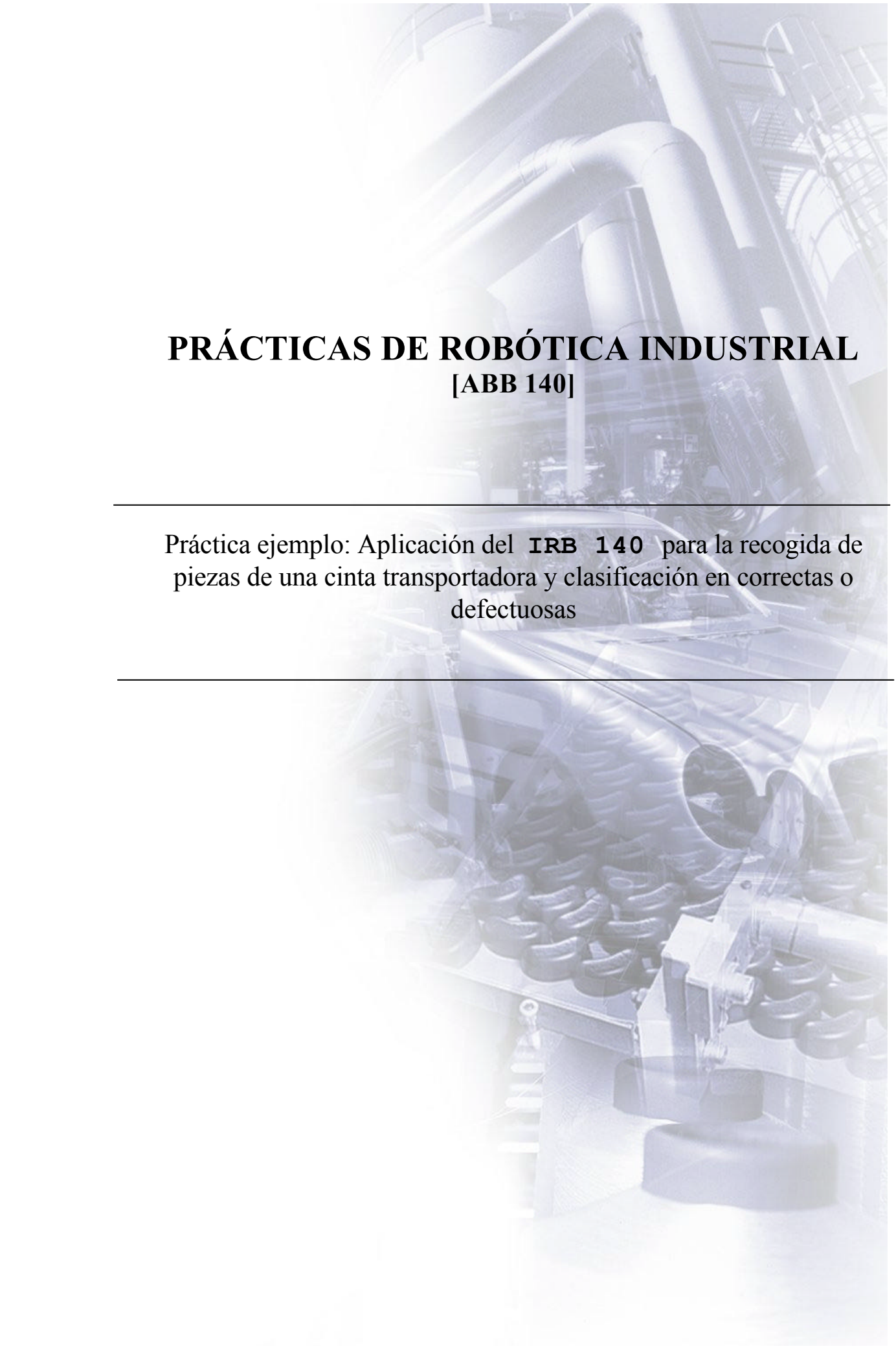


PRÁCTICAS DE ROBÓTICA INDUSTRIAL

[ABB 140]

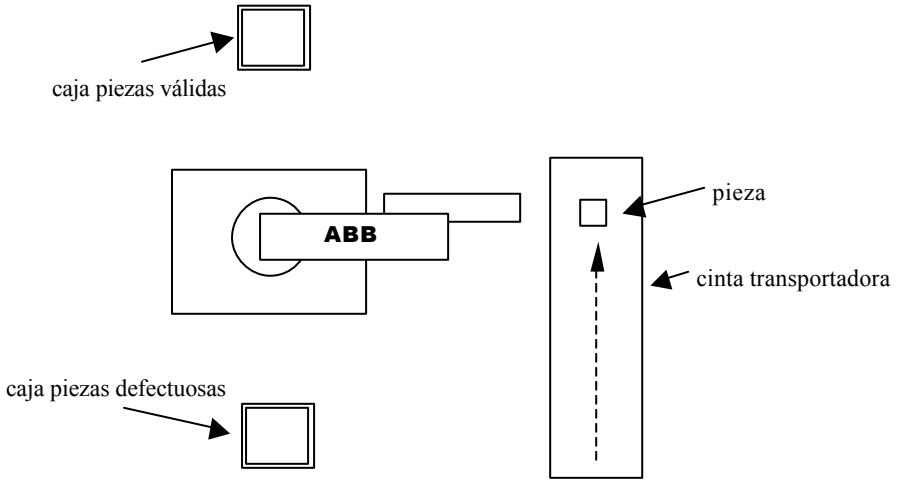
Práctica ejemplo: Aplicación del **`IRB 140`** para la recogida de piezas de una cinta transportadora y clasificación en correctas o defectuosas

---

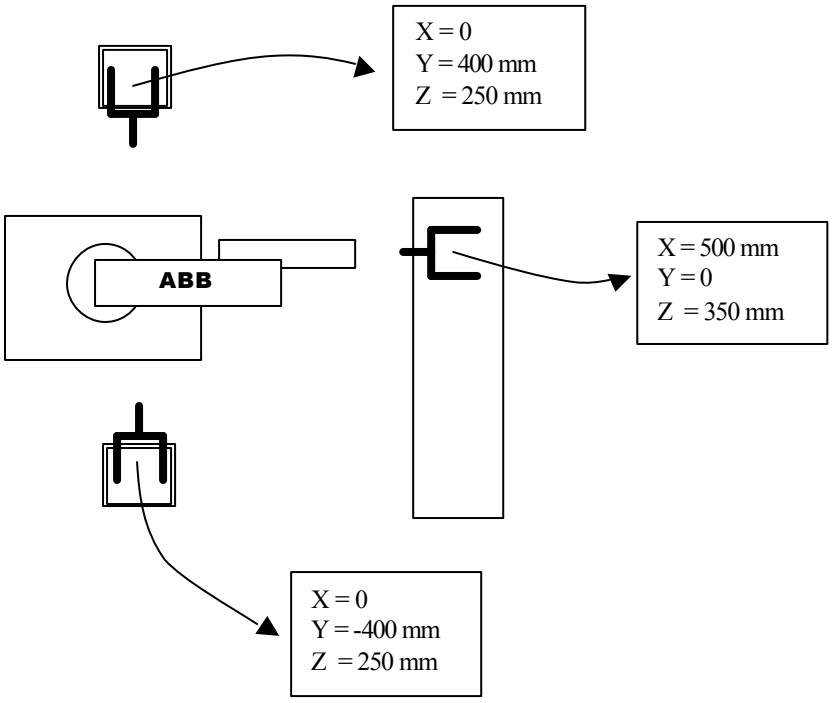
1. Planteamiento del problema

Se desea utilizar el robot IRB-140 en un entorno de fabricación en el que las piezas provenientes de una cinta transportadora pueden ser consideradas correctas o defectuosas. Las piezas correctas deberán ser depositadas en una caja y las piezas incorrectas en otra.

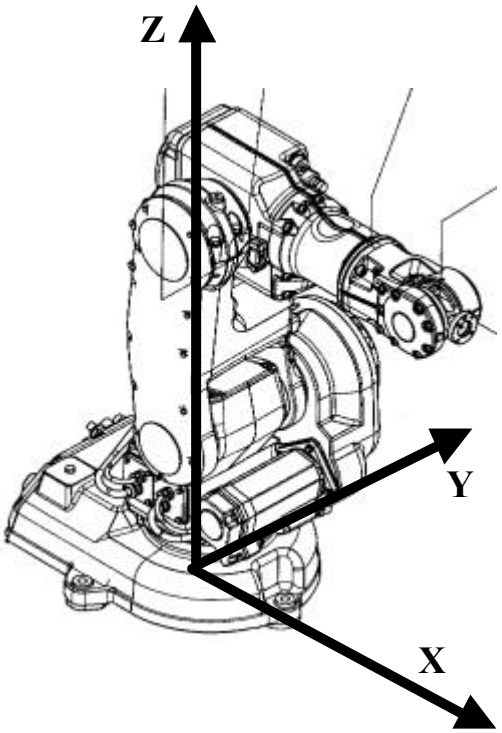
La disposición del entorno es la que se describe a continuación:



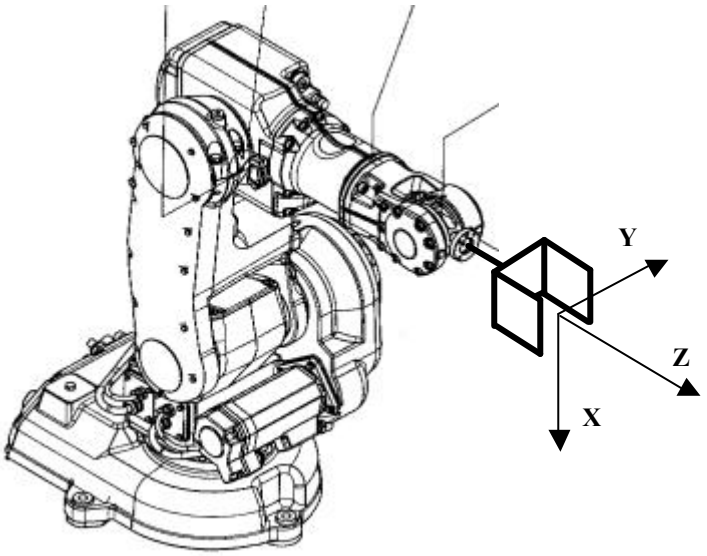
Las posiciones y orientaciones deseadas para la pinza del robot en cada uno de los puntos finales (recogida de pieza y cada una de las dos descargas) se representan en el gráfico siguiente:



Los valores x, y, z y las orientaciones deben ir referidos al sistema de coordenadas de la base del robot:



Y para la pinza consideraremos un sistema de coordenadas igual al que utiliza el IRB-140 para el extremo de su brazo, pero desplazado hasta el punto de agarre de las piezas:



2. Definición de la pinza utilizada

Se indican tanto los datos geométricos como los datos de carga:

Datos geométricos o distancia al extremo del robot del punto de agarre medidos en el sistema de coordenadas del extremo del robot:

- X = 0
- Y = 0
- Z = 120 mm

Datos de carga:

- Peso de la pinza: 1,5 kg
- Centro de gravedad expresado en el sistema de coordenadas del extremo del robot:
  - X = 0
  - Y = 0
  - Z = 60 mm
- Momentos de inercia:
  - Mx = 0.01 kgm^2
  - My = 0.01 kgm^2
  - Mz = 0.01 kgm^2

3. Configuraciones deseadas para cada posición

Dado que las posiciones y orientaciones pedidas se pueden alcanzar con distintas configuraciones del robot, se especifican cuáles deberán usarse:

- Posición de reposo, de agarre de la pieza en la cinta y de descarga de pieza válida:
  - config_1 := [0, 0, -1, 1]
- Posición de descarga de pieza defectuosa:
  - config_2 := [-1, -1, 0, 1]

4. Secuencia de operaciones deseada

- El robot debe ir inicialmente a una posición de reposo, alineada con la posición de recogida de las piezas en la cinta pero separada de ella 20cm medidos sobre el eje z.
- En esta posición esperará hasta que la señal de entrada pieza_presente se active; en ese momento se dirigirá a la pieza y cerrará la pinza activando la señal de salida cerrar_pinza. Se esperarán 2 segundos antes de hacer ningún movimiento, para asegurar que la pinza se ha cerrado.
- En función del valor de la señal de entrada pieza_correcta, llevará la pieza a una u otra de las cajas y la depositará en ella, abriendo la pinza una vez alcanzada la posición.
- Volverá a la posición de reposo a la espera de una nueva pieza.
- Todos los movimientos se efectuarán a una velocidad de 200mm/s y con una precisión de 5mm salvo el de aproximación a la pieza en la cinta que se efectuará a 50mm/s y con precisión máxima (fine).
- Se utilizarán siempre trayectorias lineales.

5. Primer paso resolución: cálculo de las posiciones y orientaciones pedidas

POSICIÓN DE REPOSO

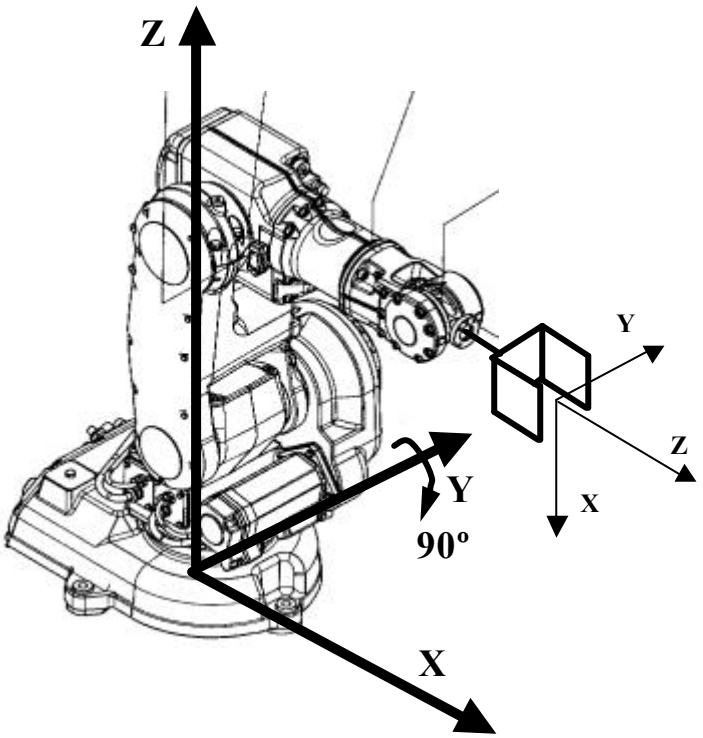
De acuerdo con lo dicho en el enunciado, tendrá como coordenadas respecto de la base:

- X = 300mm
- Y = 0
- Z = 350mm

Nota: los 20cm de desplazamiento sobre el eje z de la herramienta corresponden a 20cm de desplazamiento sobre el eje x de la base del robot. Siempre se considerarán las posiciones y orientaciones referidas a la base del robot.

Ahora resta expresar la orientación del sistema de coordenadas de la pinza con respecto al sistema de coordenadas de la base mediante un cuaternio. Lo más sencillo será buscar los giros que es necesario provocar en el sistema de coordenadas de la base para obtener el sistema de coordenadas de la herramienta en la orientación deseada.

Para esta primera posición, será necesaria una rotación de +90º sobre el eje Y de la base



Esta rotación, expresada mediante un cuaternio quedará:

Q1 = rot(j, 90) = [cos45, 0, sen45, 0] = [0.7071, 0.0, 0.7071, 0.0]

Por lo tanto, tendremos:

- pos_reposo := [300.0, 0.0, 350.0];
- ori_reposo := [0.7071, 0.0, 0.7071, 0.0];
- conf_reposo := [0, 0, -1, 1];

POSICIÓN DE AGARRE DE PIEZA EN LA CINTA

Se trata de una posición muy similar, sólo cambia el desplazamiento sobre el eje x; las orientaciones y la configuración no varían:

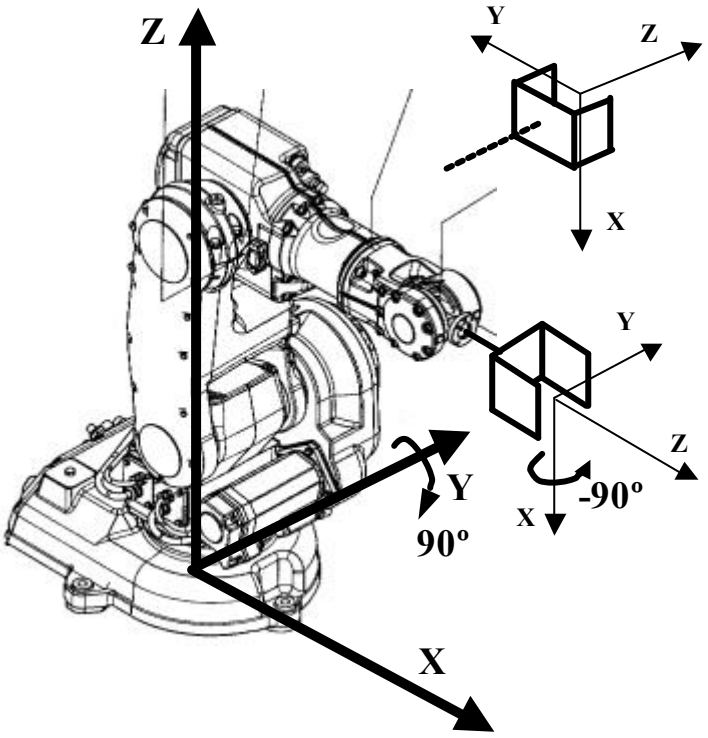
- pos_cinta := [500.0, 0.0, 350.0];
- ori_cinta := [0.7071, 0.0, 0.7071, 0.0];
- conf_cinta := [0, 0, -1, 1];

POSICIÓN DE DESCARGA DE PIEZA CORRECTA

La posición X, Y, Z es conocida:

- X = 0
- Y = 400mm
- Z = 250mm

La orientación resultará un poco más difícil de obtener: se desea llegar al sistema de coordenadas de descarga de la pieza a partir del sistema de coordenadas de la base:



Como se ve en la figura, la forma más sencilla de obtener el sistema de coordenadas deseado es mediante dos giros consecutivos:

- Un primer giro de 90º alrededor del eje y de la base.
- Un segundo giro de –90º alrededor del eje x del sistema resultante tras el primer giro.

El cuaternio que representa el primero de los giros ya se dedujo anteriormente:

Q1 = rot(j, 90) = [cos45, 0, sen45, 0] = [0.7071, 0.0, 0.7071, 0.0]

El segundo se puede deducir análogamente:

Q2 = rot(i, -90) = [cos(-45), sen(-45), 0, 0] = [0.7071, -0.7071, 0.0, 0.0]

La combinación de los dos giros se obtiene como el producto de los dos cuaternios. Dado que el segundo de los giros se efectúa sobre un eje perteneciente al sistema ya girado, el segundo cuaternio se deberá postmultiplicar:

Qtot = Q1° Q2

De acuerdo con la fórmula del producto de cuaternios, el resultado quedará:

Qtot = [0.5, -0.5, 0.5, 0.5]

Los datos resultantes son, por tanto:

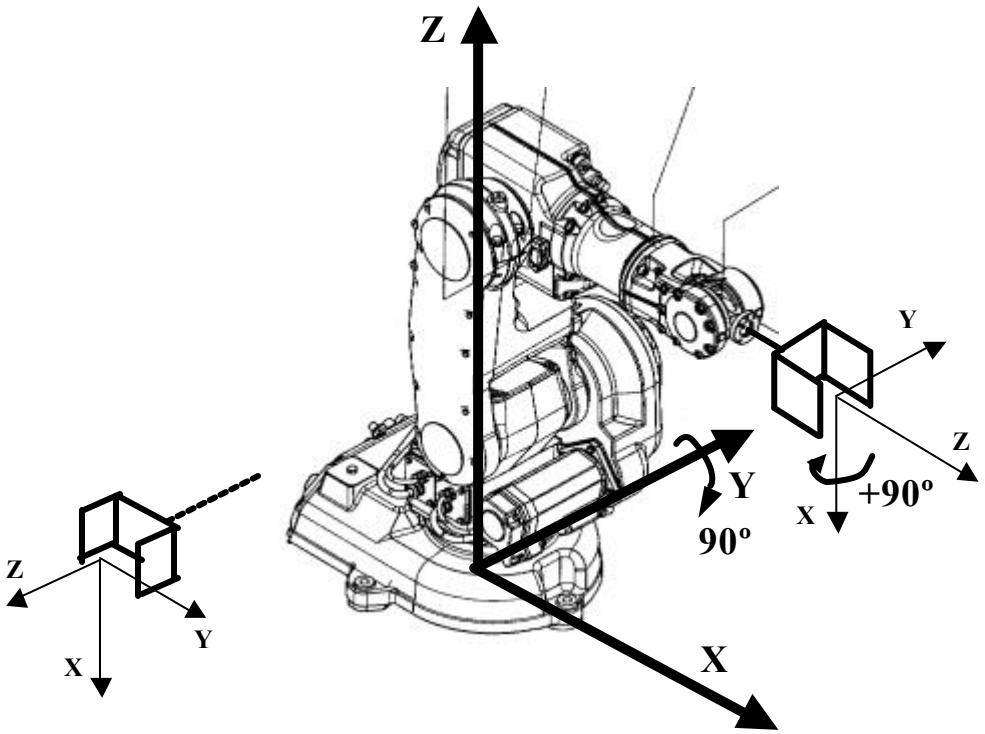
- pos_correcta := [0.0, 400.0, 250.0];
- ori_correcta := [0.5, -0.5, 0.5, 0.5];
- conf_correcta := [0, 0, -1, 1];

POSICIÓN DE DESCARGA DE PIEZA DEFECTUOSA

La posición es conocida:

- X = 0
- Y = -400mm
- Z = 250mm

Y la orientación se calcula de modo análogo al caso anterior pero con la particularidad de que en este caso el segundo giro es de +90º y no de -90º:



En este caso los cuaternios quedan:

Q1 = rot(j, 90) = [cos45, 0, sen45, 0] = [0.7071, 0.0, 0.7071, 0.0]
Q2 = rot(i, 90) = [cos45, sen45, 0, 0] = [0.7071, 0.7071, 0.0, 0.0]

Y el resultado final:

Qtot = Q1° Q2 = [0.5, 0.5, 0.5, -0.5]

Por lo tanto, nuestras variables serán:

- pos_defect := [0.0, -400.0, 250.0];
- ori_defect := [0.5, 0.5, 0.5, -0.5];
- conf_defect := [-1, -1, 0, 1];

Nótese que en este caso la configuración es distinta: esta posición no sería alcanzable con la configuración anterior.

6. Segundo paso resolución: definición de la herramienta

La definición de la herramienta es inmediata a partir de los datos ofrecidos. Los campos que se necesitan cumplimentar para definir una herramienta son los siguientes:

robhold: define si la herramienta está unida al robot (TRUE) o se encuentra fija en el espacio (FALSE). En nuestro caso:

pinza_robhold := TRUE;

tframe: posición (x, y, z) y orientación (q1, q2, q3, q4) del sistema de coordenadas de la herramienta con respecto al sistema de coordenadas del extremo del robot. El sistema de coordenadas de la herramienta lo situaremos en el punto de agarre de piezas, tal y como se expresa en el enunciado. Por lo tanto, la posición será:

- X = 0
- Y = 0
- Z =120mm

Y la orientación será la misma que la del extremo del robot, con lo cual se expresará mediante el cuaternio identidad:

- Q1 = 1
- Q2 = 0
- Q3 = 0
- Q4 = 0

pinza_tframe := [[0.0, 0.0, 120.0], [1.0, 0.0, 0.0, 0.0]];

tload: carga de la herramienta. Incluye los siguientes datos:

- peso en kg.
- centro de gravedad expresado en el sistema de coordenadas de la herramienta.
- orientación de los ejes de inercia de la herramienta: siempre deber ser coincidente con el sistema de coordenadas de la herramienta, por tanto será el cuaternio identidad.
- Momentos de inercia respecto al centro de masas expresados en kgm^2.

Dado que todos los datos son conocidos, quedará:

pinza_tload := [1.5, [0.0, 0.0, 60.0], [1.0, 0.0, 0.0, 0.0], 0.01, 0.01, 0.01];

7. Escritura del programa

```
%%%
  VERSION:1
  LANGUAGE:ENGLISH
%%%

MODULE practica

  ! ningun eje externo conectado
  CONST extjoint ejes_ext := [9E9, 9E9, 9E9, 9E9, 9E9, 9E9];

  ! posicion y orientacion de reposo para el robot
  CONST pos pos_reposo := [300.0, 0.0, 350.0];
  CONST orient ori_reposo := [0.7071, 0.0, 0.7071, 0.0];
  CONST confdata conf_reposo := [0, 0, -1, 1];
  CONST robtarget rob_reposo := [pos_reposo, ori_reposo, conf_reposo, ejes_ext];

  ! posicion y orientacion de agarre de la pieza sobre la cinta
  CONST pos pos_cinta := [500.0, 0.0, 350.0];
  CONST orient ori_cinta := [0.7071, 0.0, 0.7071, 0.0];
  CONST confdata conf_cinta := [0, 0, -1, 1];
  CONST robtarget rob_cinta := [pos_cinta, ori_cinta, conf_cinta, ejes_ext];

  ! posición y orientación de descarga de pieza correcta
  CONST pos pos_correcta := [0.0, 400.0, 250.0];
  CONST orient ori_correcta := [0.5, -0.5, 0.5, 0.5];
  CONST confdata conf_correcta := [0, 0, -1, 1];
  CONST robtarget rob_correcta := [pos_correcta, ori_correcta, conf_correcta, ejes_ext];

  ! posición y orientación de descarga de pieza defectuosa
  CONST pos pos_defect := [0.0, -400.0, 250.0];
  CONST orient ori_defect := [0.5, 0.5, 0.5, -0.5];
  CONST confdata conf_defect := [-1, -1, 0, 1];
  CONST robtarget rob_defect := [pos_defect, ori_defect, conf_defect, ejes_ext];

  ! herramienta utilizada (pinza)
  CONST bool pinza_robhold := TRUE;
  CONST pose pinza_tframe := [[0.0, 0.0, 120.0], [1, 0, 0, 0]];
  CONST loaddata pinza_tload := [1.5, [0.0, 0.0, 60.0], [1, 0, 0, 0], 0.01, 0.01, 0.01];
  CONST tooldata pinza := [pinza_robhold,pinza_tframe, pinza_tload];

  PROC main()

    ! desplazamiento a pos de reposo, rapido y con precision baja
    MoveL rob_reposo, v200, z5, pinza;

    ! espera hasta tener una pieza disponible
    WaitDI pieza_presente, 1;

    ! desplazamiento hacia la pieza, lento y con maxima precision
    MoveL rob_cinta, v50, z5, pinza;

    ! activa pinza y espera 2 segundos
    Set cerrar_pinza;
    WaitTime 2;

    if (pieza_correcta=TRUE) THEN

      ! desplazamiento a caja piezas correctas, rapido y poco preciso
      MoveL rob_correcta, v200, z5, pinza;

    ELSE

      ! desplazamiento a caja piezas defectuosas, rapido y poco preciso
      MoveL rob_correcta, v200, z5, pinza;

    ENDIF

    ! se abre la pinza y se esperan dos segundos
    Reset cerrar_pinza;
    WaitTime 2;

  ENDPROC

ENDMODULE
```