Trabajo Fin de Grado
Grado en Ingeniería de Tecnologías Industriales

Diseño, programación y simulación de estaciones robotizadas industriales con Robotstudio

Autor: Agustín Ramos Hurtado
Tutor: David Muñoz de la Peña Sequedo

Dep. Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2016

Trabajo Fin de Grado
Grado de Ingeniería en Tecnologías Industriales

Diseño, programación y simulación de estaciones robotizadas industriales con Robotstudio

Autor:
Agustín Ramos Hurtado

Tutor:
David Muñoz de la Peña Sequedo
Profesor titular

Dep. Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2016

Trabajo Fin de Grado: Diseño, programación y simulación de estaciones robotizadas industriales con Robotstudio

Autor: Agustín Ramos Hurtado

Tutor: David Muñoz de la Peña Sequedo

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2016

El Secretario del Tribunal

A mi familia

A mis amigos

A mis maestros y profesores

Agradecimientos

En primer lugar me gustaría agradecer a David Muñoz, tutor de este trabajo de fin de grado, por su guía y orientación, sin la cual hubiese sido imposible el correcto entendimiento de los programas utilizados y el adecuado proceso de realización de éste.

A la Entidad Colaboradora EUCOMSA, por su gran apoyo y ayuda al permitirme desarrollar este proyecto en su estancia, y a los operarios y empleados junto a los que he trabajado durante estos meses, por sus múltiples consejos y enseñanzas.

A mis maestros y profesores, por el arduo trabajo de transmitirme sus conocimientos durante años, por sus diferentes puntos de vista y formas de enseñar y porque sin su apoyo no habría llegado hasta donde estoy hoy.

A mis padres, por estar siempre conmigo, por enseñarme a crecer y a que si caigo debo levantarme, por mostrarme que hay que esforzarse y persistir en lo que uno ama en la vida.

Por todo esto y mucho más,

Mil gracias.

Agustín Ramos Hurtado

Sevilla, 2016

Resumen

La redacción de este trabajo se ha realizado con el objetivo de diseñar estaciones de control y programación de robots manipuladores de la casa ABB utilizando el programa Robotstudio. Por un lado, se realiza una guía inicial de utilización del programa y otra guía más completa sobre creación de estaciones. Esta segunda guía se expone para una estación de ejemplo creada por el propio autor.

Por otro lado, como objetivo final se desarrolla una estación real de soldadura de la línea de producción de una empresa, en la cual el autor ha estado durante los meses de Abril y Mayo de 2016 realizando medidas, diseñando y programando la estación robotizada.

Abstract

The wording of this work has been done with the aim of designing control stations and programming of robot manipulators belonging to the ABB home using the software RobotStudio. On the one hand, it has been made an initial guide to learn to use the program and a more comprehensive guide about the creation of stations. This second guide is exposed for an example station created by the author.

On the other hand, the ultimate goal it has developed has been a real welding station of the production line of a company, in which the author has been during the months of April and May 2016 taking measurements, designing and programming the robotized station.

Índice

ÍNDICE DE FIGURAS

ÍNDICE DE PLANOS

1 INTRODUCCIÓN

Una meta es más que un sueño: es un sueño sobre el que se trabaja.

- Leopoldo Fernández Pujals -

Este trabajo se redacta con objeto de desarrollar estaciones de control y programación de robots manipuladores de la casa ABB utilizando el programa Robotstudio. Se realizará una guía de utilización del mismo, así como el desarrollo de una estación real de montaje y soldadura de la línea de producción de una empresa.

1.1. Motivación

La motivación de este trabajo de fin de grado parte de la idea de demostrar con la mayor fidelidad posible una estación robotizada industrial de soldadura. Dicha estación a simular pertenece a una empresa situada en la localidad de Utrera (Sevilla, España) denominada EUCOMSA (Europea de Construcciones Metálicas, S.A.)

El autor de este trabajo ha realizado tareas de programación y control de robots manipuladores de soldadura de EUCOMSA durante los meses de Abril y Mayo de 2016, completando un total de 165 horas de trabajo en dicha estancia.

Además, la realización de este trabajo también llevará a mostrar una introducción sobre la utilización del software Robotstudio, así como la explicación paso a paso del diseño y programación de una estación, partiendo de una de ejemplo creada por el autor.

Por tanto, se puede interpretar que la motivación de este trabajo es doble: por un lado, la exposición de una guía de uso de Robotstudio y por otro, la creación y simulación de un proceso real robotizado.

Desde un punto de vista más general, una mayor precisión en la soldadura, evitar errores por mano de obra inexperta y una mejora en la velocidad de la producción de una empresa son las principales causas que llevan a motivar al ser humano a automatizar procesos industriales de montaje de piezas y soldadura mediante estaciones robotizadas como las expuestas en este trabajo.

1.2. El robot industrial. Aplicaciones

La robótica industrial (de manipulación) nace de exigencias prácticas en la producción. Se trata de un elemento importante de la automatización flexible, encaminada a la reducción de costes.

Un robot industrial es un manipulador reprogramable multifuncional que consta de una secuencia de elementos estructurales rígidos, denominados eslabones, conectados entre sí mediante articulaciones. Éstas pueden ser lineales o rotacionales. Las configuraciones más comunes para estos robots son:

- Configuración cartesiana: el robot se mueve linealmente en los tres ejes cartesianos (x, y, z) (3 grados de libertad).
- Configuración cilíndrica: el robot se mueve linealmente en dos ejes cartesianos y rotacionalmente respecto a un tercer eje (3 grados de libertad).
- Configuración polar: el robot se mueve linealmente en un eje y rotacionalmente respecto a otros dos ejes (3 grados de libertad).
- Configuración SCARA: el robot se mueve rotacionalmente respecto a dos ejes paralelos y linealmente respecto al mismo eje, normalmente el eje Z (3 grados de libertad).
- Configuración esférica: el robot se mueve rotacionalmente respecto a 3 o más ejes (tres o más grados de libertad).

Los robots utilizados en este trabajo son todos con configuración esférica, como el ilustrado en la figura 1-1.

Figura 1-1. Robot manipulador de
la marca ABB. Modelo IRB 2400.
(6 grados de libertad)

El robot industrial está diseñado bien para mover materiales y piezas mediante movimientos programados, o bien para la ejecución de otro tipo de tareas potencialmente muy diversas. Las tareas más comunes que puede desarrollar un robot industrial, además de las tareas propias de manipulación de piezas (carga/descarga, embalado, paletizado, etc.) son:

- Ensamblado
- Mecanizado (taladrado, pulido…)
- Pintura
- Soldadura (por punto o contínua)
- Sellado

En este trabajo se estudian procesos robotizados de carga y descarga de material, así como de paletizado del mismo. A su vez, también se programarán multiples tareas de soldadura de diversas piezas.

De esta forma, los principales factores que intervienen a la hora de robotizar una tarea pueden ser:

- Flexibilidad en los programas de fabricación: adaptabilidad del sistema productivo a cambios en el mercado.
- Incremento de la productividad: es necesario tener en cuenta que existe un intervalo de productividad para el cual lo más rentable es la utilización de robots industriales, pero fuera del cual los costes de la robotización son demasiado elevados y no es rentable su uso. En la figura 1-2 se puede observar dicho intervalo.

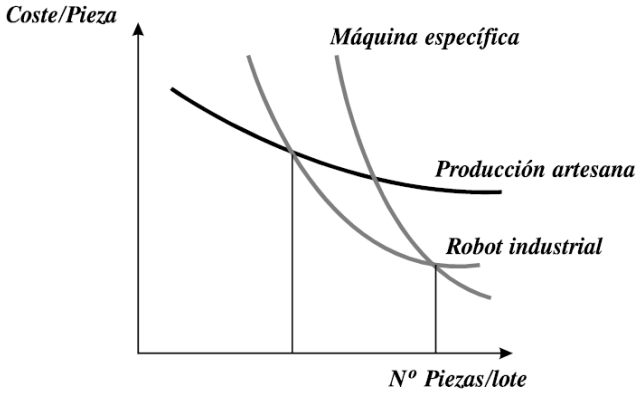


Figura 1-2. Número de piezas a procesar frente a su coste de producción.

- Ahorro de materias primas.
- Incremento de la calidad: efecto derivado de la automatización.
- Reducción del tiempo: factor importante para cumplir con los plazos frente al cliente y para ahorrar en material inmovilizado.
- Realización de tareas imposibles a mano (complejidad geométrica).
- Reducción del coste de mano de obra.
- Mejora de la seguridad del trabajador: incremento de la seguridad en tareas peligrosas e incremento en la comodidad del operario, el cual pasa a supervisar en lugar de manipular.

1.3. Estructura del Trabajo

El trabajo expuesto se puede dividir en dos bloques principales:

Un primer bloque consiste en una explicación guiada del uso del programa Robotstudio y de la implementación de una estación, aplicando en una estación de ejemplo los conocimientos adquiridos.

Así, de forma introductoria se ahondarán en conceptos como la creación de controladores y herramientas, la creación de un conjunto de puntos y de planos de trabajo, el estudio de las distintas posibles configuraciones que puede tener un robot en cada punto definido y la creación de trayectorias.

Posteriormente, se expondrán diseños de herramientas y otras geometrías de trabajo creadas en programas como Auto CAD o Catia y su comunicación con el software Robotstudio para poder trabajar con dichas piezas.

También se verán los procedimientos de creación y diseño de los SC (Smart Components, Componentes Inteligentes en Castellano) de gran utilidad a la hora de crear una estación automatizada y cuya potencia de uso es bastante amplia. De esta forma, se analizará el conexionado entre los SC y los controladores de los robots, proceso de conexión denominado lógica de estación.

Para finalizar el primer bloque se realizará una introducción a RAPID, el lenguaje de programación utilizado en Robotstudio creado por la casa ABB. Se verán conceptos de realización de trayectorias con este lenguaje, comandos de soldadura, órdenes de movimiento desde los controladores a los SC, sincronización entre distintos robots manipuladores conectados a un mismo controlador o con controladores independientes entre sí, etc.

Por último se simulará el funcionamiento de la estación de ejemplo cuyo diseño y programación se ha explicado paso a paso.

En el segundo gran bloque del trabajo se implementan los conceptos explicados anteriormente de Robotstudio a un proceso real de soldadura:

Partiendo de una estación ABB de EUCOMSA se realizarán las mediciones pertinentes para realizar un diseño geométrico de ésta lo más fiel posible a la realidad con el software Auto CAD.

Después, se importarán estas piezas a Robotstudio para posteriormente incorporar los robots manipuladores, dotarlas de inteligencia artificial diseñando las SC y programar los robots desde RAPID.

Una vez creada la estación real, se realizarán pruebas de simulación con piezas que actualmente está produciendo la empresa para sus proyectos de ingeniería y que necesiten el proceso de soldadura dentro de la línea de producción de dicho producto. Se programarán los robots, por tanto, para que realicen las operaciones de soldadura en dichas piezas.

Por último, se realizará la transferencia de los programas al robot real y así poder visualizar el correcto funcionamiento de la estación de soldadura. También se concluirá exponiendo posibles mejoras de diseño de los robots ABB de EUCOMSA para así poder mejorar la productividad de la fábrica.

2 IMPLEMENTACIÓN EN ROBOTSTUDIO

Cuando el alumno está preparado, aparece el maestro.

- Lucio Anneo Séneca -

El programa Robotstudio es un software de programación de robots industriales, diseñado y patentado por la empresa ABB, que permite un abanico de posibilidades en el mundo de la automatización industrial y la robótica manipuladora gracias a a la versatilidad de su entorno de simulación y del lenguaje RAPID.

Para los programas y estaciones creadas en este trabajo se ha utilizado la versión 5.61.02 de este software, penúltima versión existente a fecha de redacción, por detrás de la versión 6.01.01.

2.1. Introducción al programa Robotstudio

2.1.1 Creación de una nueva estación

Para comenzar a trabajar con Robotstudio, una vez se tenga el programa instalado y ejecutado en el ordenador, lo primero es hacer clic en 'Crear una nueva estación vacía' tal y como se aprecia en la figura adjunta.

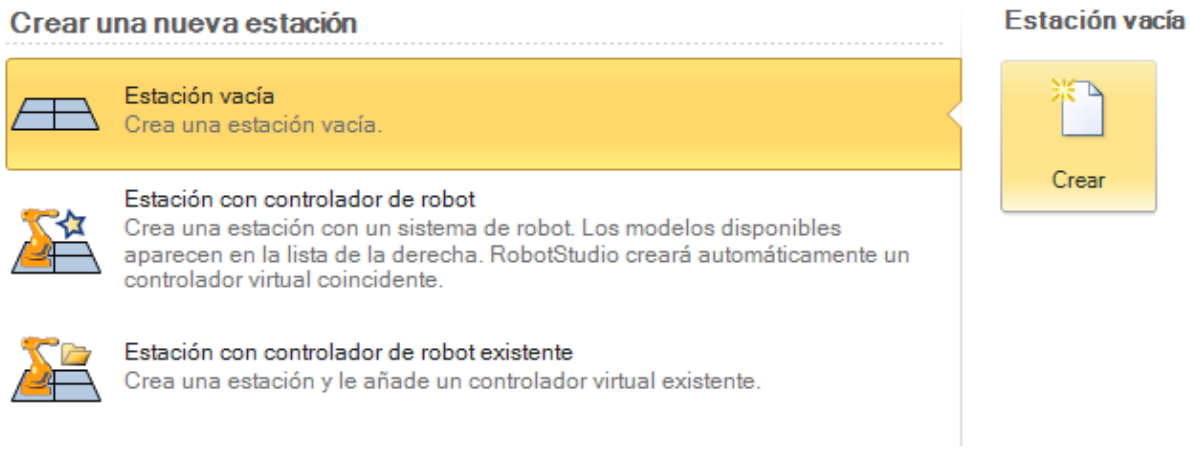


Figura 2-1. Inicio de Robotstudio. Pestaña para la creación de estaciones.

Una vez se tenga la estación vacía creada aparecen en el programa 7 pestañas que se utilizarán a lo largo del trabajo. Estas son: Pestaña Archivo, Pestaña Inicio, Pestaña Modelado, Pestaña Simulación, Pestaña Controlador, Pestaña RAPID y Pestaña Complementos.

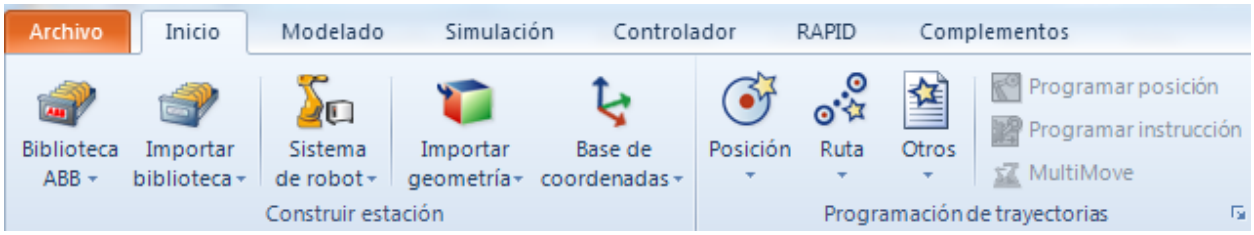


Figura 2-2. Pestañas del programa Robotstudio

A continuación, para poder trabajar con un robot es necesario importar en Robotstudio el mismo. Para ello, dentro de la Pestaña inicio hay que pinchar sobre Biblioteca ABB. De esta forma se desplegará un listado de módelos de robots de la marca ABB con distintas aplicaciones: robots manipuladores, de pintura, posicionadores y tracks.

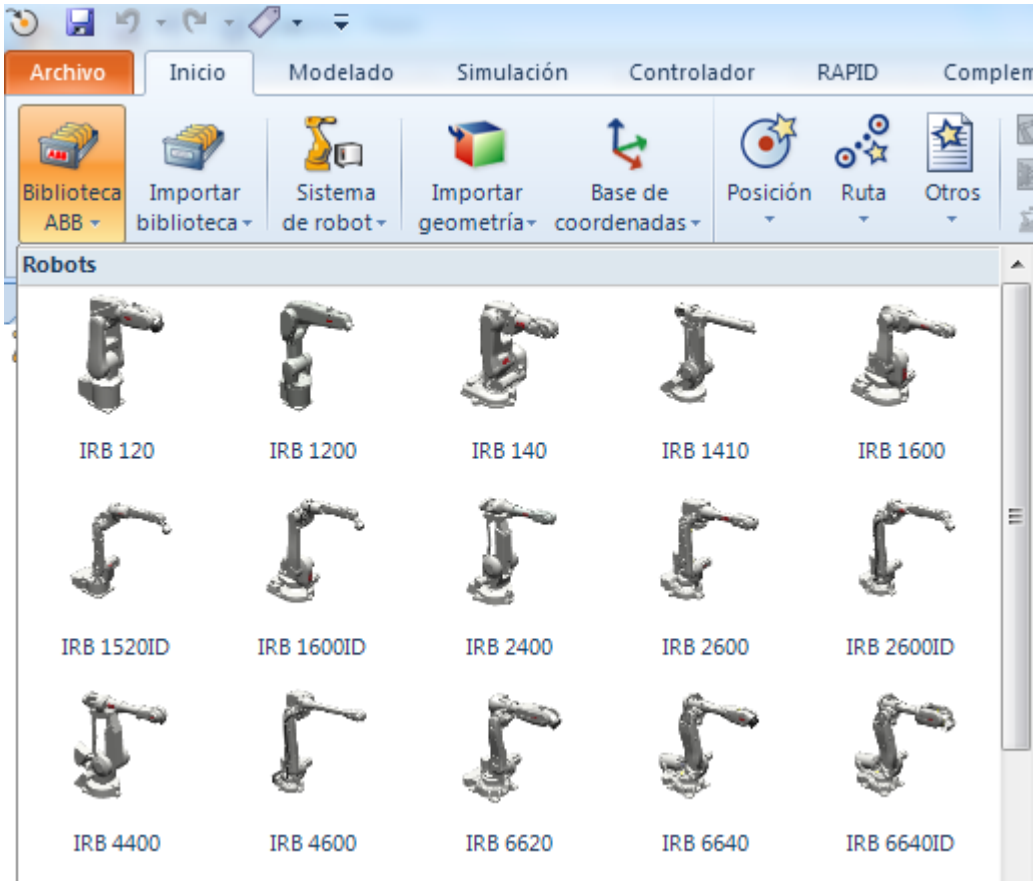


Figura 2-3. Robots de la Biblioteca ABB

En este trabajo tan solo se utilizan modelos de robots del primer tipo, por lo que para esta primera introducción se escogerá uno de estos. Más en concreto, para este subapartado introductorio se elegirá por ejemplo el IRB 140.

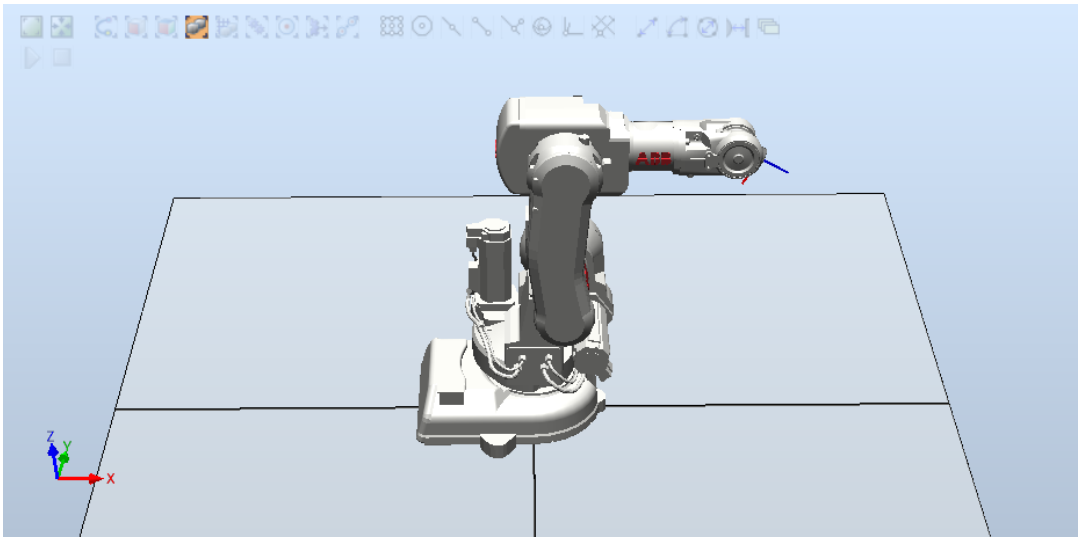


Figura 2-4. Robot ABB modelo IRB 140

2.1.2 Creación de la herramienta de un robot

Para poder realizar operaciones y crear posiciones hay que hacerlo respecto a una herramienta de trabajo del robot. Esta herramienta va incorporada a la muñeca del mismo y puede tratarse desde una pinza o un imán para manipular objetos hasta una pistola de soldadura para unir piezas metálicas entre sí.

Existe una gran variedad de herramientas, las cuales se pueden encontrar dentro de la Pestaña Inicio realizando clic en Importar Biblioteca -> Equipamiento.

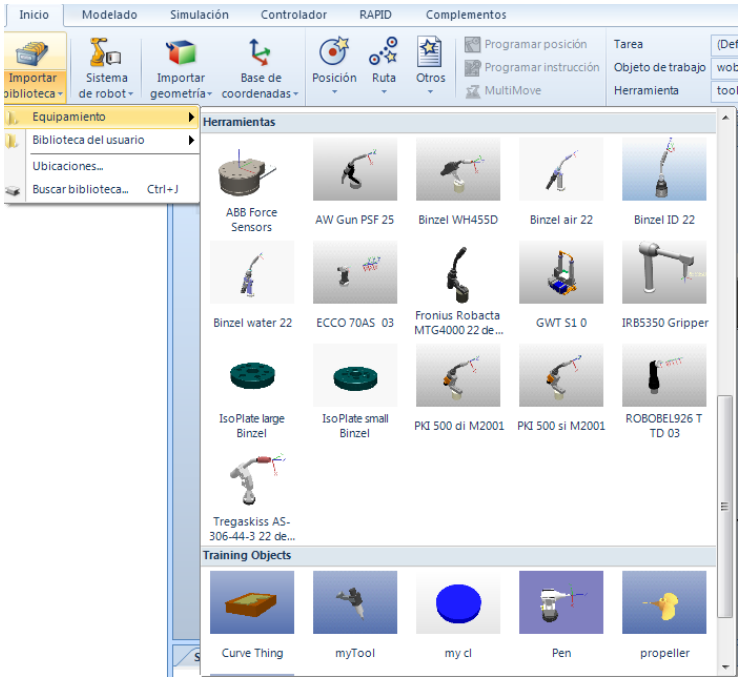


Figura 2-5. Herramientas equipadas en Robotstudio.

También existe la posibilidad de crear una herramienta propia partiendo de una geometría. Así, se puede crear un cilindro, un cono o una geometría importada desde otro programa para utilizarla como herramienta de trabajo.

Por ejemplo, se puede crear un cilindro dirigiéndose a la Pestaña Modelado y pinchando sobre 'Sólido'.

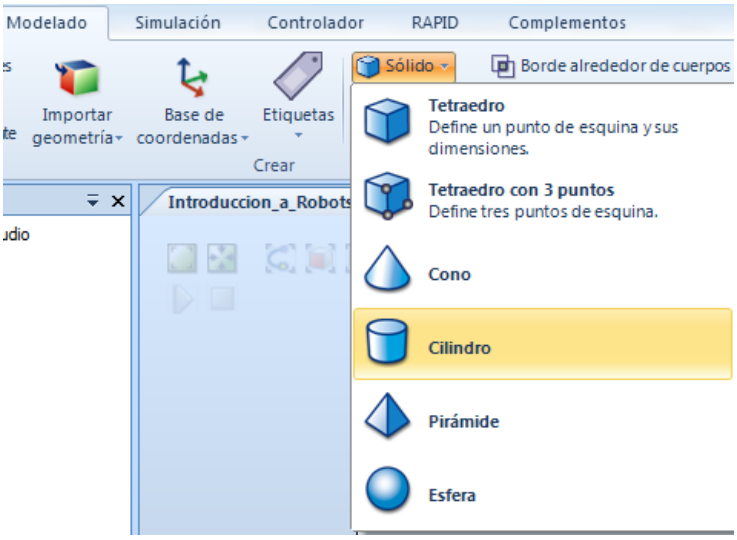


Figura 2-6. Pestaña para creación de sólidos

A continuación, es necesario configurar las características del sólido, en este caso el radio y la altura. Para ello, en primer lugar es necesario medir el diámetro de la muñeca del robot, ya que la herramienta cilíndrica deberá tener uno similar.

Si se quiere realizar una medida en Robotstudio hay que hacer clic dentro de la ventana de simulación sobre 'Punto a punto' y posteriormente sobre 'Ajustar a objetos' por ejemplo, como se aprecia en la figura 2-7. Esta operación de medida de distancias será de gran utilidad durante todo el trabajo para poder verificar la correcta magnitud de las piezas creadas, así como de sus posiciones relativas.

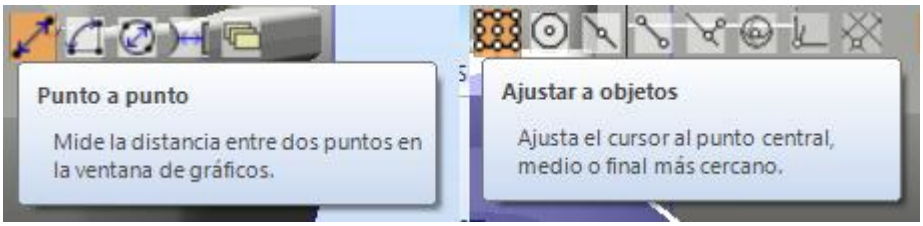


Figura 2-7. Botones para medir distancias y ajustar a objetos, respectivamente.

Una vez que la medida esté realizada se configuran los parámetros del cilindro y se crea la pieza.Hacer notar que si no se dice nada la pieza se crea por defecto en la posición (0, 0, 0).

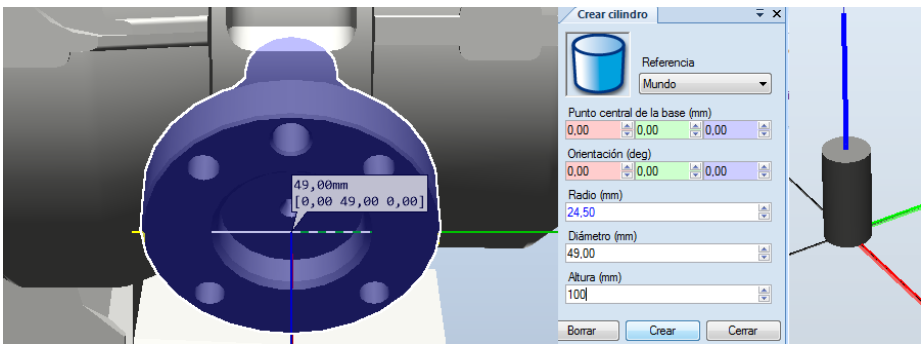


Figura 2-8. Medida de la muñeca y creación del cilindro.

En este punto, el siguiente paso consiste en transformar en herramienta la geometría creada, paso necesario para que el programa interprete la pieza como una herramienta del robot. Para esto, hay que ir a la Pestaña Modelado-> Crear herramienta.

En primer lugar hay que ponerle un nombre a la herramienta, en este caso 'Cilindro' y seleccionar como pieza una existente, en este caso la creada anteriormente.

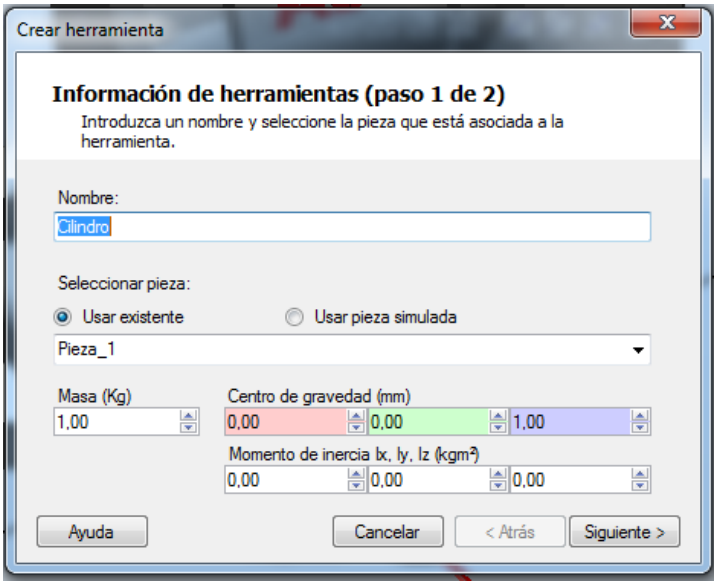


Figura 2-9. Crear herramienta. Paso 1 de 2.

En segundo lugar es necesario asociarle una posición y orientación relativa al extremo de la herramienta, para que las posiciones del robot que se crean posteriormente se realicen respecto a dicho extremo. En este caso, al ser creado el cilindro con una altura de 100 mm. en el eje Z habrá que asociarle esa posición en ese eje. Dicha posición es denominada TCP (Tool Central Point) y se le asigna realizando clic al botón de la flecha que aparece en la figura 2-10.

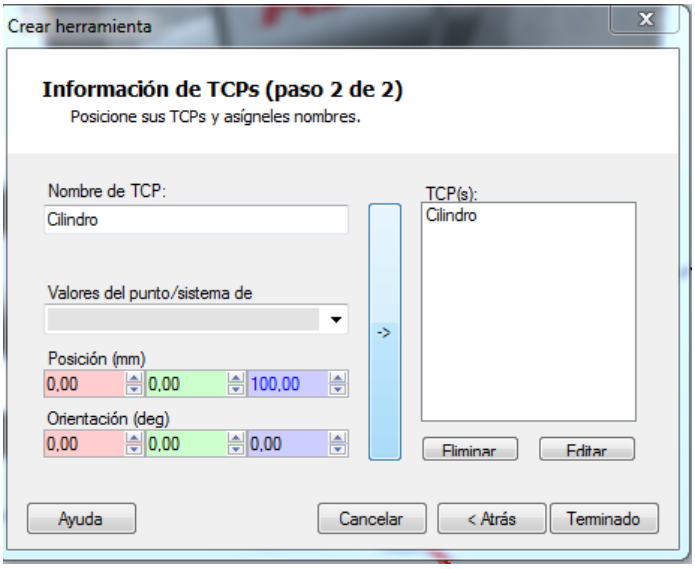


Figura 2-10. Crear herramienta. Paso 2 de 2.

Por último, una vez finalizado el proceso de creación de la herramienta, hay que adjuntarla al robot y actualizar su posición para que se sitúe en la muñeca de éste.

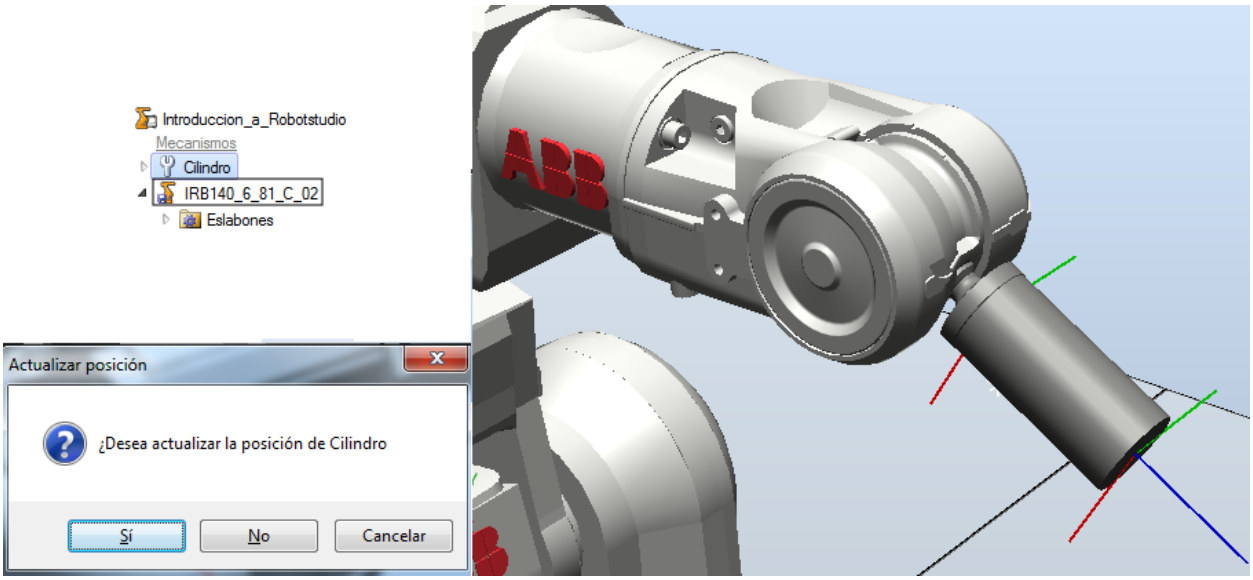


Figura 2-11. Actualización de la posición de la herramienta.

Este último paso de la figura 2-11 también es necesario realizarlo si se adjunta una herramienta importada de la biblioteca de Robotstudio, en vez de utilizar una pieza creada anteriormente.

2.1.3 Creación del controlador de un robot

Hasta ahora, lo que se tiene es un brazo con una herramienta adjunta. Sin embargo, con esto tan solo no se pueden realizar movimientos ni programar trayectorias. Para dotar de "inteligencia" al robot es necesario crear un controlador asociado al mismo.

Para ello, hay que ir a la Pestaña Inicio -> Sistema de robot -> Desde diseño…

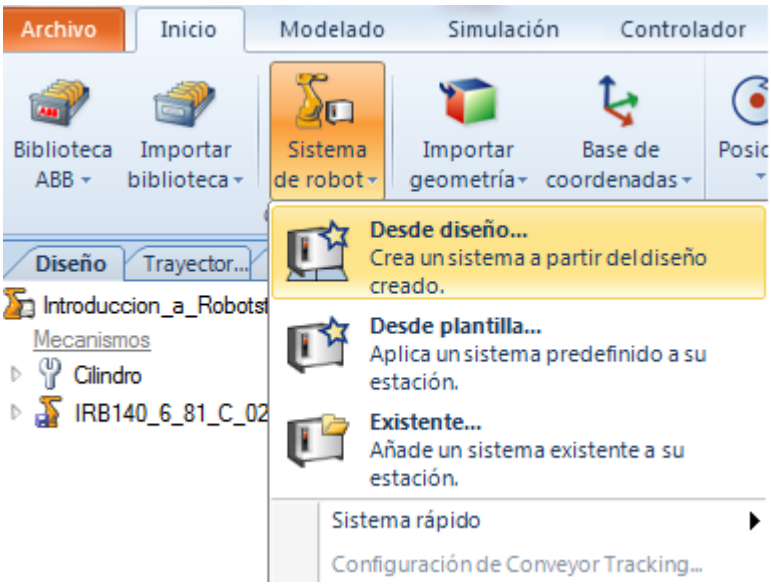


Figura 2-12. Creación de un controlador desde diseño.

A continuación, es necesario poner un nombre al controlador a crear, en este caso "Ejemplo_Intro". Posteriormente hay que pulsar Siguiente, se elige el IRB 140 como mecanismo a controlar y se le vuelve a dar a Siguiente de nuevo.

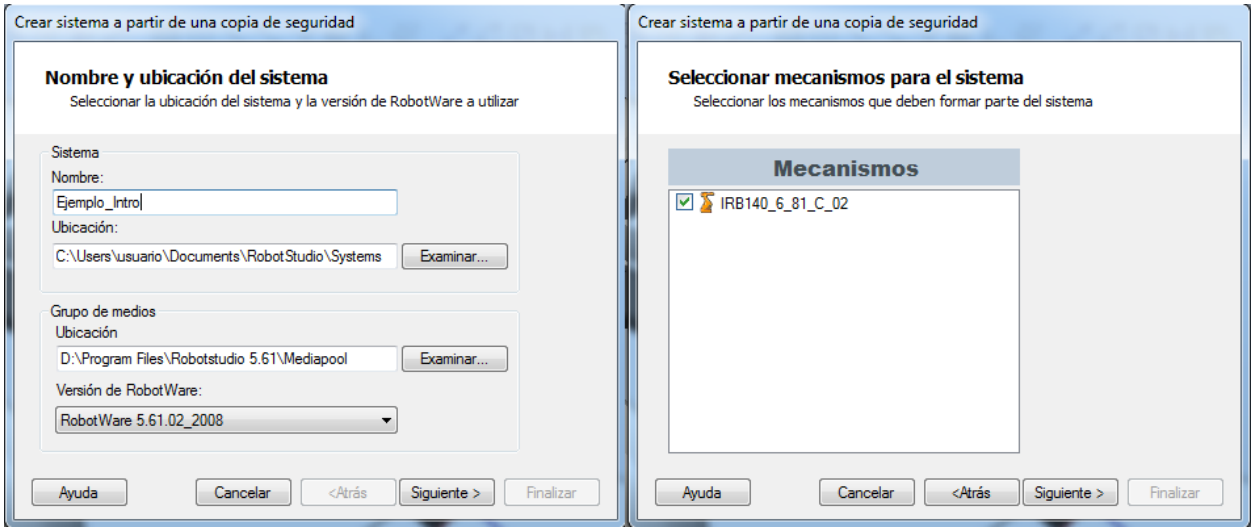


Figura 2-13. Pasos para la creación de un controlador.

Por último, se pide configurar las opciones del sistema a crear. En este paso, haciendo clic en Opciones… se despliega un listado de parámetros a configurar. Más adelante, en la estación de ejemplo y en la estación real se ahondarán más en estas opciones. De momento, se dejan las que vienen por defecto y se pulsa en Finalizar, creándose así el controlador del robot.

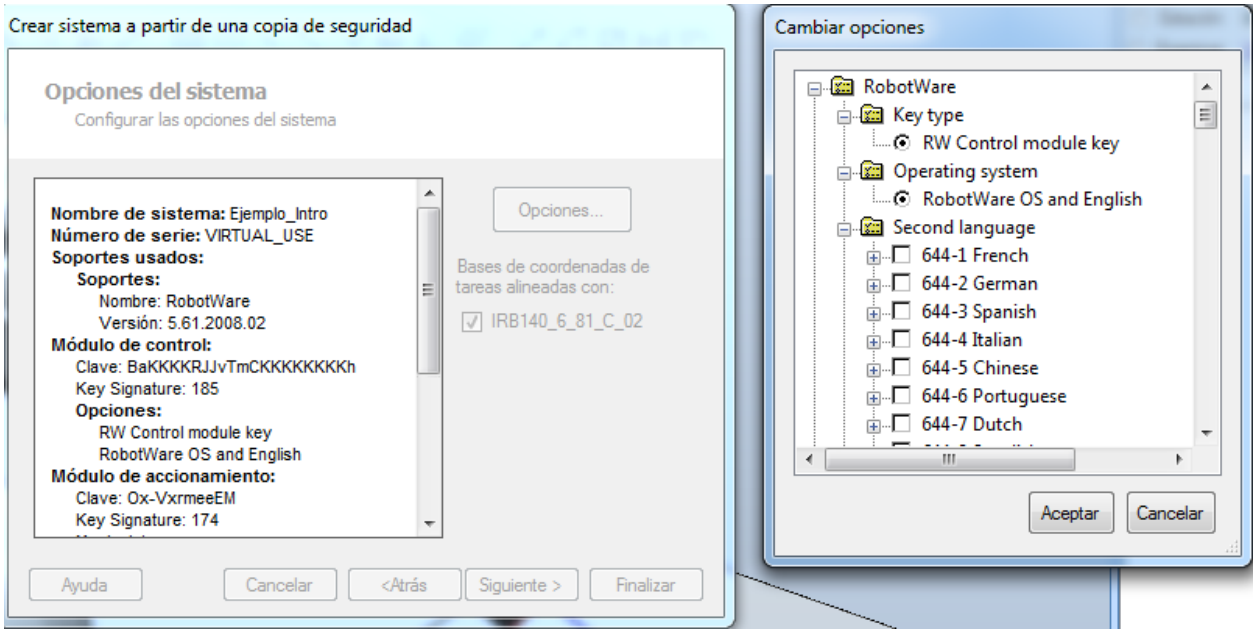


Figura 2-14. Opciones del sistema.

2.1.4 Creación de planos de trabajo y posiciones

En primer lugar, para diseñar un plano de trabajo se va a crear un sólido para construir en éste el mismo. Así, se crea un hexaedro de lado 100 mm. cuyo punto de esquina es el [500,-50,400].

Para crear el plano u objeto de trabajo hay que hacer clic dentro de la Pestaña Inicio en Otros -> Crear objeto de trabajo.

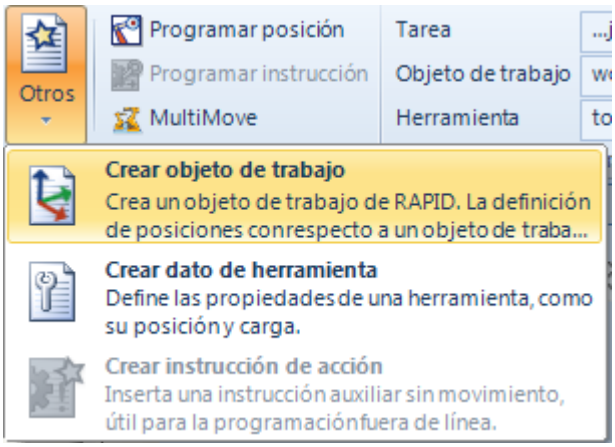


Figura 2-15. Pestaña de creación de un objeto de trabajo.

A continuación, es necesario dotar el objeto de trabajo de un nombre (‘Cubo’, en este caso) y posicionar su sistema de coordenadas en la esquina superior del cubo, tal y como aparece en la figura 2-16. Por último, pinchando en Aceptar y Crear ya se tiene el objeto de trabajo creado.

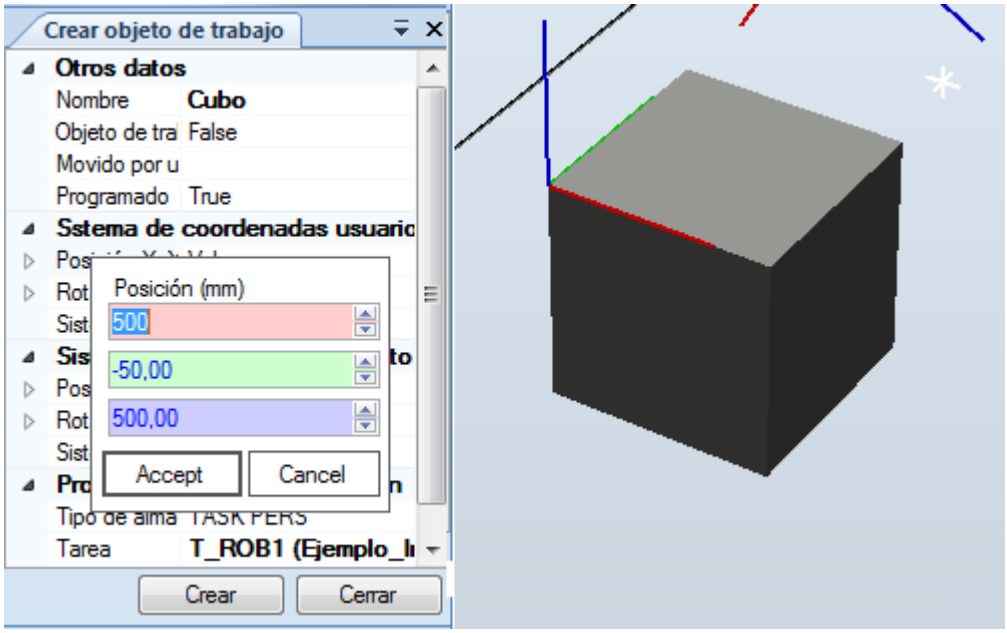


Figura 2-16. Configuración para crear un objeto de trabajo.

Una vez creado el objeto de trabajo ‘Cubo’, hay que comprobar primero que, dentro de la Pestaña Inicio, en Parámetros aparece “Objeto de trabajo: Cubo” y “Herramienta: Cilindro” para referir las posiciones al Objeto de trabajo y a la Herramienta creada. En caso de no aparecer alguno de éstos, antes de todo hay que modificarlo.

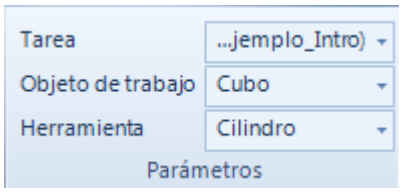


Figura 2-17. Parámetros de la Pestaña Inicio.

El siguiente paso es crear posiciones relativas al Cubo y al Cilindro y una posición inicial del robot relativa al work object 0 que trae predeterminado el programa. Para ello, hay que ir a Posición -> Crear Punto, dentro de la Pestaña Inicio.

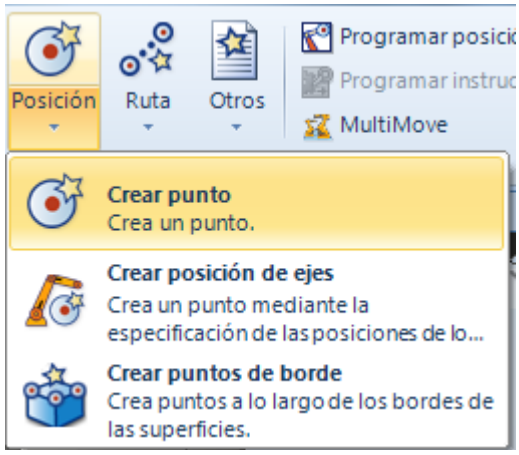


Figura 2-18. Crear punto.

A continuación, se crean por ejemplo los cuatro puntos correspondientes a las esquinas superiores del hexaedro. Para esto, una vez dentro de la pestaña 'Crear punto' se señala 'Ajustar a objetos' en la ventana de simulación y simplemente se hace clic sobre cada una de las esquinas.

Por último, tras comprobar que la tarea y el objeto de trabajo son correctos se pincha en Crear.

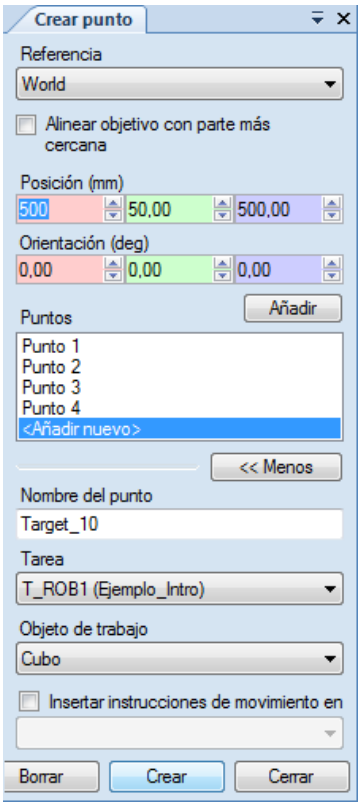


Figura 2-19. Configuración de la creación de puntos

Para la creación de una posición inicial, en primer lugar se va a situar de forma manual el robot en una posición acorde a ésta. Para ello, es necesario ir al apartado ‘Mano alzada’ dentro de la Pestaña Inicio y hacer clic en el tercer botón: ‘Movimiento de eje’.

Tras mover manualmente los ejes del robot a una posición inicial se crea el punto correspondiente utilizando ‘Ajustar a objetos’ como se explicó anteriormente, pero en este caso en el centro del efector final del cilindro. Notar que esta posición inicial se crea respecto al objeto de trabajo Work Object 0, y no respecto a ‘Cubo’.

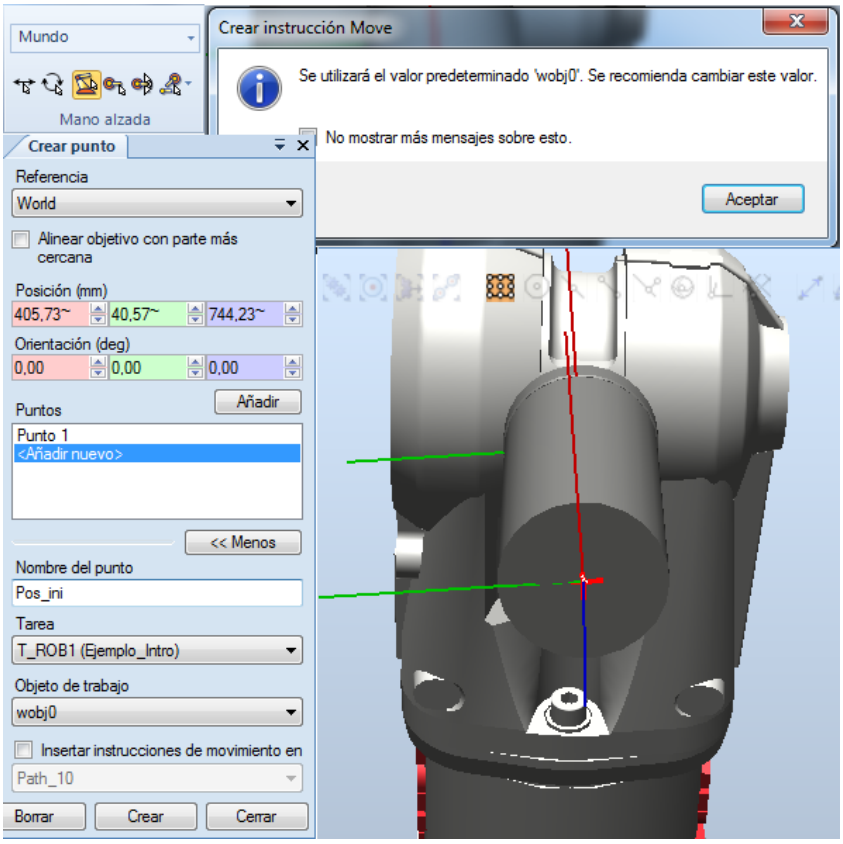


Figura 2-20. Creación de la posición inicial del robot.

2.1.5 Creación de trayectorias

Con el conjunto de puntos ya creados, antes de crear una trayectoria es necesario visualizar la posición de la herramienta Cilindro en cada uno de estos 5 puntos, ya que con bastante probabilidad, la orientación de la herramienta asignada por defecto a estos puntos sea tal que el robot no pueda alcanzarlos, o bien lo haga con una configuración que atraviese el propio cubo.

Por tanto, si se va al primer punto creado (denominado Target_10 por defecto) y se hace clic derecho sobre él, hay que buscar la opción 'Ver herramienta en la posición -> Cilindro' y pinchar sobre ésta.

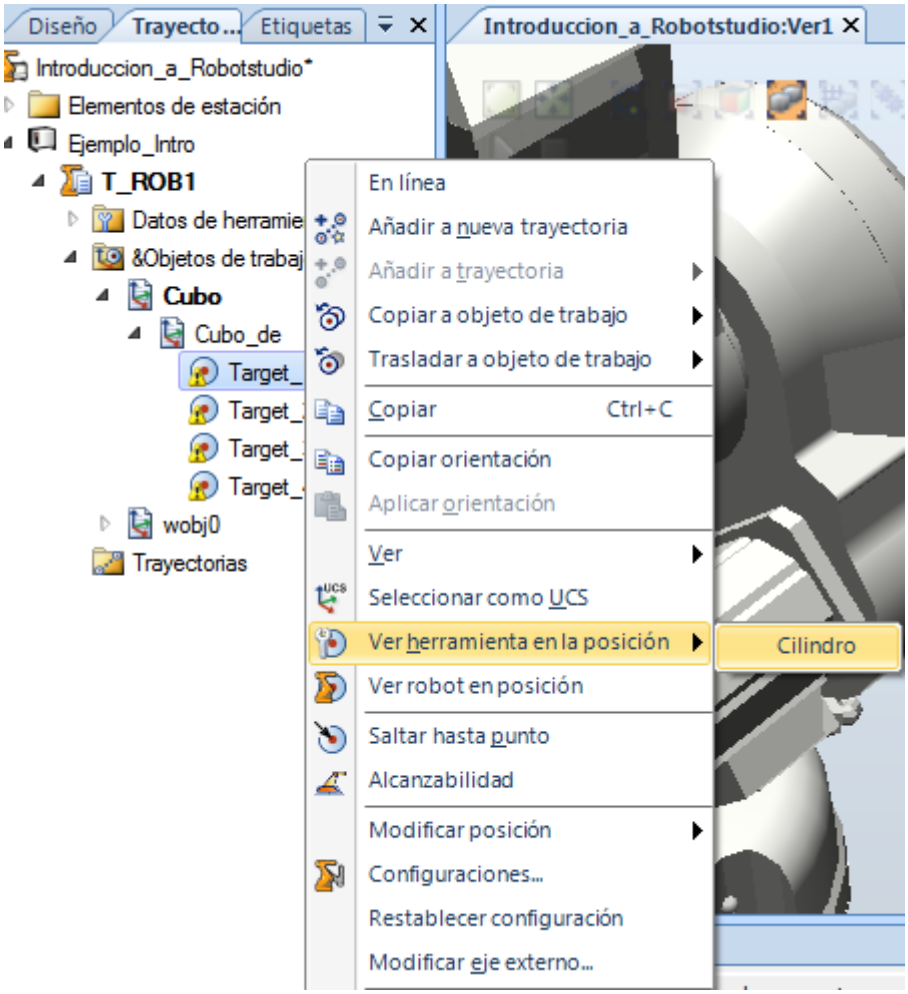


Figura 2-21. Ver herramienta en la posición.

Como se aprecia en la figura 2-22, con dicha orientación la herramienta no puede acceder bien al punto Target_10, ya que si se hace clic en 'Ver robot en posición' (en la Figura 2-21, justo debajo de 'Ver herramienta en la posición') saldrá el mensaje de que el punto Target_10 está fuera de alcance.

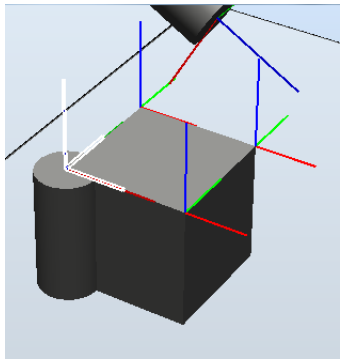


Figura 2-22. Herramienta en la posición con orientación errónea.

Para solucionar esto, es necesario volver a hacer clic derecho sobre el punto Target_10, pero esta vez pinchar sobre Modificar posición -> Girar. A continuación hay que configurar el giro de forma que el robot pueda alcanzar el punto correctamente. En el caso del ejemplo esto se consigue realizando un giro de 180 grados sobre el eje X o el eje Y.

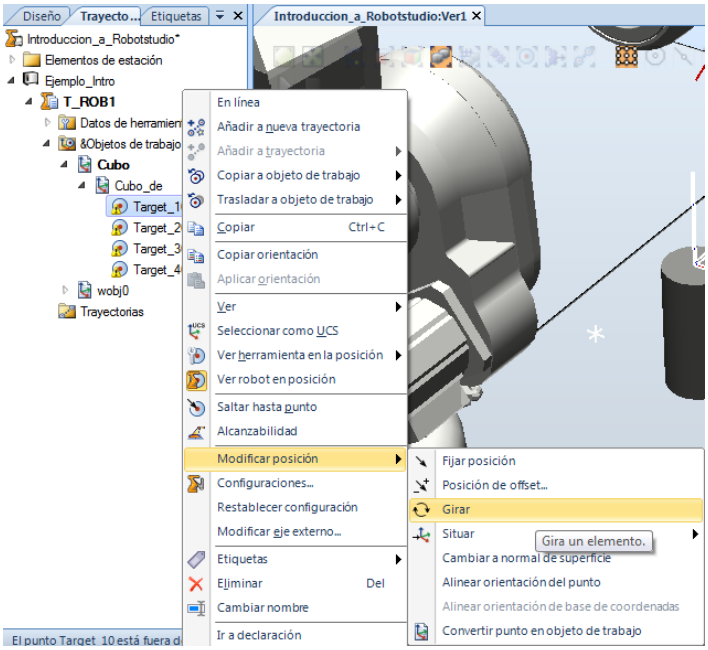


Figura 2-23. Girar posición.

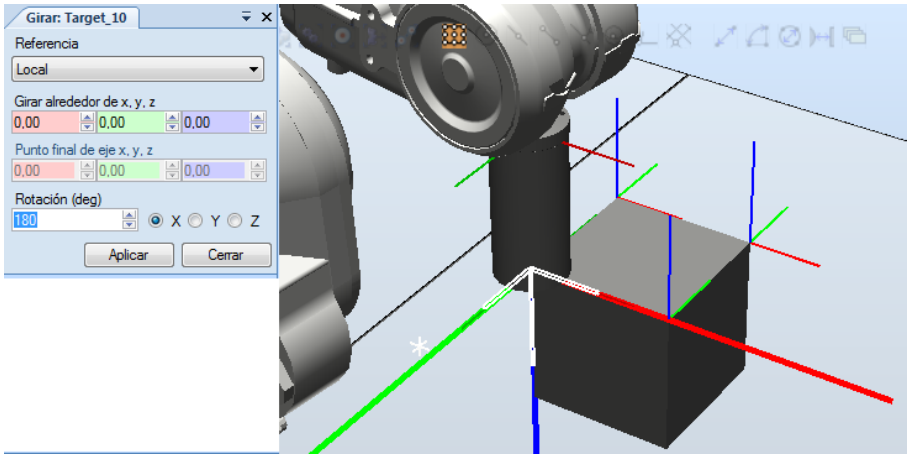


Figura 2-24. Posición Target_10 girada 180 grados respecto al eje X.

Para realizar la misma operación de giro en el resto de puntos creados basta con clicar con el botón derecho en Target_10 y pinchar sobre 'Copiar orientación'. Tras esto, hay que señalar los otros puntos y clicando en el botón derecho pinchar sobre 'Aplicar orientación'.

Respecto al punto Pos_ini_10, para el caso particular que se está ejemplificando se realizará un giro de 180 grados respecto del eje X y otro de 45 grados en el eje Y de la herramienta, para así evitar posibles zonas fuera de la alcanzabilidad del brazo.

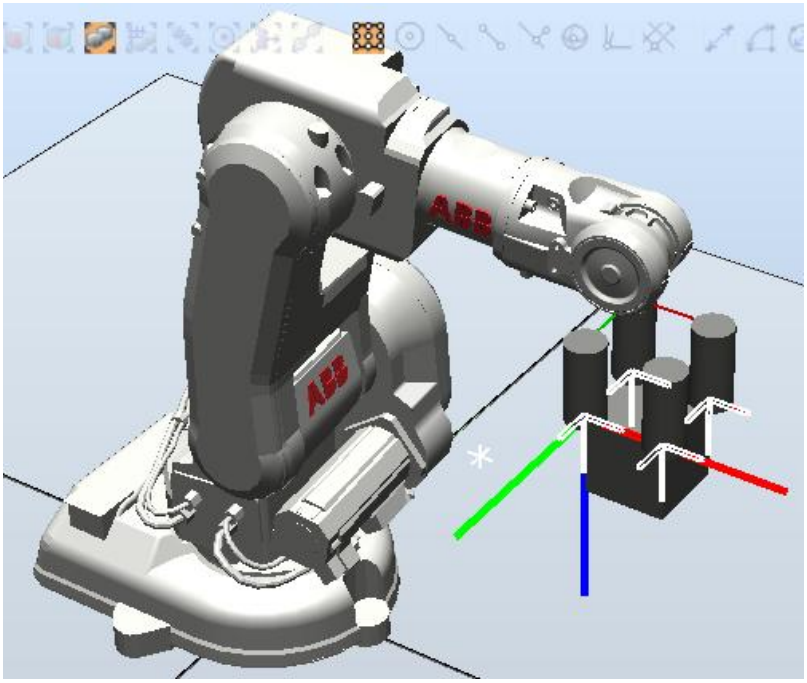


Figura 2-25. Puntos con la orientación correcta.

Con la orientación modificada, ahora hay que realizar una configuración de los ejes del robot. Esto es debido a que el brazo manipulador puede acceder a cada uno de estos puntos con diferentes configuraciones de sus ejes, por tanto, hasta que no se le asigne una saldrá un símbolo de advertencia junto al símbolo de cada posición.

Las configuraciones de los ejes del robot se designan con una serie de cuatro números enteros que especifican en qué cuadrante se encuentran los ejes. Por ejemplo, la configuración [-1, 0, 1, 0] significa:

El primer entero (-1) especifica la posición del eje 1: en algún punto del primer cuadrante negativo (rotación entre 0 y -90 grados).

El segundo entero (0) especifica la posición del eje 4: en algún punto del primer cuadrante positivo (rotación entre 0 y 90 grados).

El tercer entero (1) especifica la posición del eje 6: en algún punto del segundo cuadrante positivo (rotación entre 90 y 180 grados).

El cuarto entero (0) especifica la posición del eje X: un eje virtual utilizado para especificar el centro de la muñeca respecto a los demás ejes.

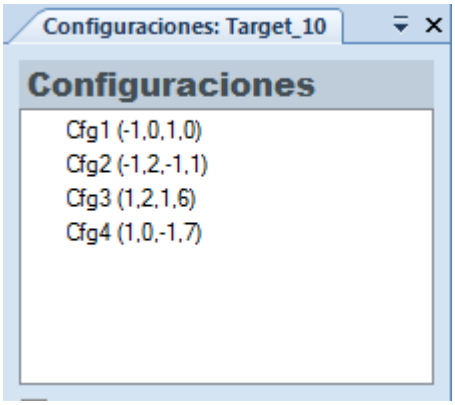


Figura 2-26. Configuraciones del robot para Target_10.

Eligiendo la configuración para cada posición buscando la mayor cantidad de ejes del robot en el primer cuadrante ya se puede crear una trayectoria.

Por tanto, a continuación hay que dirigirse hacia el apartado ‘Trayectorias’ que aparece en la ventana de la izquierda ‘Trayectorias y puntos’ dentro de la Pestaña Inicio, hacer clic encima con el botón derecho y pulsar ‘Crear trayectoria’.

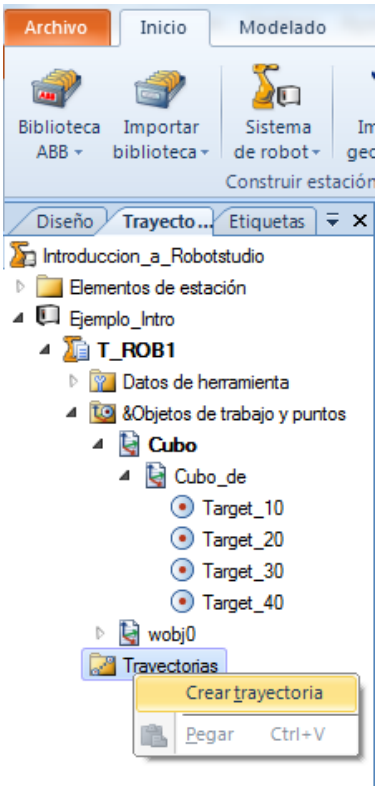


Figura 2-27. Creación de una trayectoria.

Con esto, se crea una trayectoria denominada por defecto Path_10. Por último, para asociar una serie de puntos a dicha trayectoria simplemente se arrastran los puntos creados hacia Path_10 en el orden en el que se quiera realizar dicha trayectoria. Por ejemplo, se le puede decir al robot que primero se mueva a su posición inicial, luego recorra las cuatro esquinas superiores del hexaedro para finalmente volver a su posición inicial.

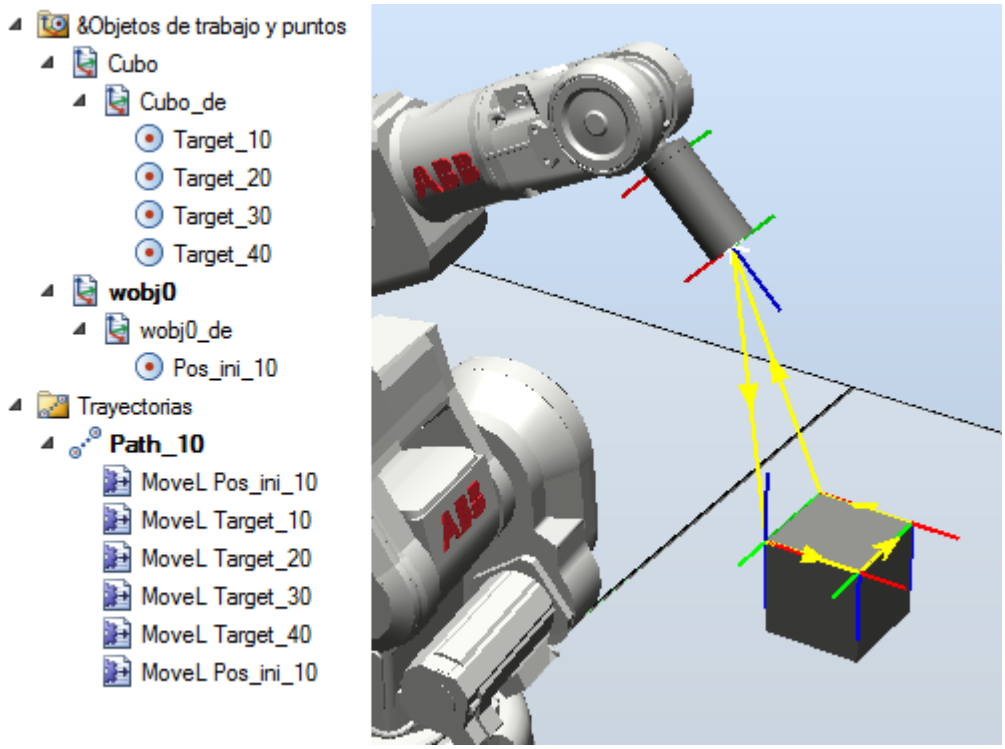


Figura 2-28. Trayectoria Path_10.

2.1.6 Introducción a RAPID

En primer lugar, si se quiere programar en RAPID, hay que sincronizar la estación creada con el controlador virtual. De esta forma, se consigue que la trayectoria Path_10 se sincronice también, paso obligatorio para que en la Pestaña RAPID aparezcan los módulos de programa 'CalibData' y 'Module1', necesarios para programar el robot.

Así, dirigiéndose a la Pestaña Inicio -> Sincronizar, haciendo clic encima y luego en Aceptar se consigue tener la estación sincronizada correctamente.

A continuación, para poder programar es necesario entrar en la Pestaña RAPID y en la ventana de la izquierda haciendo clic dentro de Controlador -> Estación actual (Ejemplo_Intro) -> RAPID -> T_ROB1 -> Módulos de programa (Module1) se abre la ventana de programación en RAPID. En este caso, debido a la sincronización realizada anteriormente, ya aparecen los comandos de definición de las posiciones creadas y la trayectoria Path_10 con sus movimientos correspondientes.

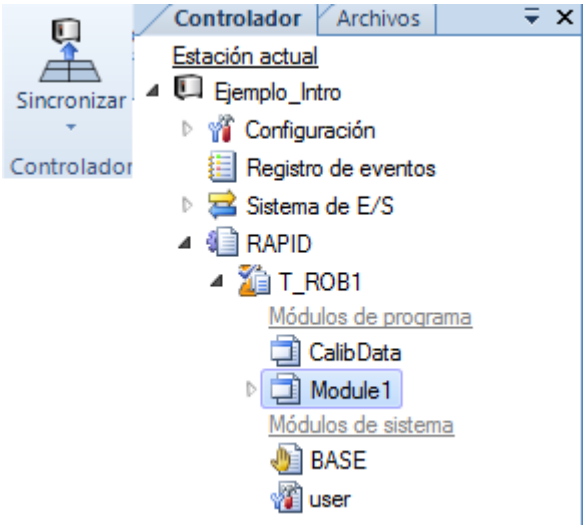


Figura 2-29. Sincronización y entrada al Module1.

Ahora se va a realizar una breve explicación sobre el uso de los comandos de RAPID más utilizados en este trabajo.

En primer lugar, el módulo del programa 'Module1' engloba todos los programas. Éste se inicia con la instrucción 'MODULE Module1' y finaliza con 'ENDMODULE'. Para cada programa, se inicializa con 'PROC Nombre_del_programa ()' para terminarlo con 'ENDPROC'. Dentro de Module1 existe un apartado, denominado 'Declaraciones de datos' donde están definidas las posiciones del robot. Esta definición se realiza mediante el comando CONST seguido de la palabra robtarget, el nombre de la posición, sus coordenadas relativas al objeto de trabajo y la configuración de ejes en dicho punto.

```
CONST robtarget Target_10:=[[0,0,0],[0,1,0,0],[-1,0,1,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
```

En segundo lugar, la instrucción principal de movimiento del robot es MoveL (o MoveJ) cuyos argumentos especifican la velocidad del movimiento, la precisión con la que se mueve el robot hasta el objetivo, la herramienta asociada a la posición y el plano de trabajo en la que está definida.

```
MoveL Target_10,v1000,z100,Cilindro\WObj:=Cubo;
```

Para el argumento de la velocidad (speeddata) los datos predefenidos van desde v5 hasta v7000, además de un valor vmax que depende del tipo de robot utilizado. Así, el valor v5 indica una velocidad de movimiento de 5 mm/s y v7000 hace referencia a que la velocidad es de 7000 mm/s.

Para el argumento de la precisión del movimiento (zonedata) los datos predefinidos van desde fine hasta z200. Estos valores hacen referencia al error en distancia (en mm) que hay desde el TCP de la herramienta del robot

al punto objetivo en el momento en el que se ejecuta la instrucción y el robot está pasando por dicha posición. Por ejemplo, z200 significa un error de 200 mm, z50 un error de 50 mm y fine que no existe error a la hora de aproximarse a la posición (a z0 le corresponde un error de 0.3 mm).

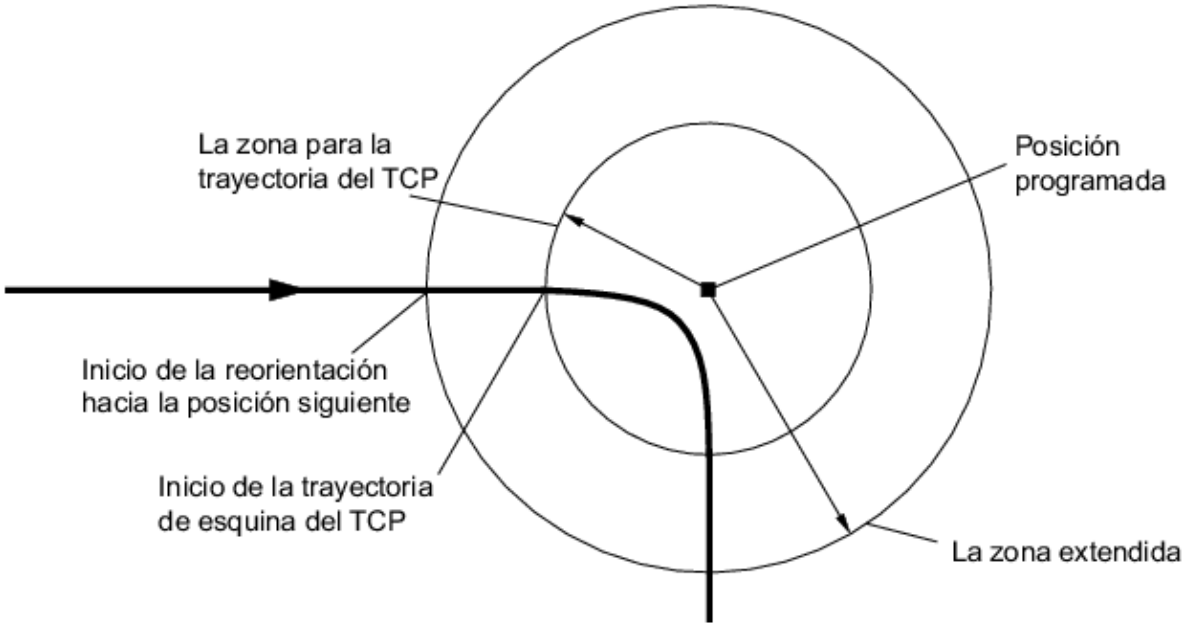


Figura 2-30. Error de precisión en la trayectoria de aproximación del TCP a una posición determinada.

Respecto a la instrucción 'Move' existen tres principales:

- MoveL: el robot realiza el movimiento de un punto a otro siguiendo una línea recta.
- MoveJ: el robot realiza el movimiento siguiendo una trayectoria no lineal.
- MoveC: el robot realiza una trayectoria circular. En la instrucción MoveC se distiguen dos posiciones, una primera que define un punto intermedio del circulo y otra final que define el punto destino.

Existe la posibilidad de definir posiciones relativas a otras posiciones ya existentes. Para ello, se tiene las instrucciones Offs y RelTool.

Con la instrucción Offs hay dos maneras de definir una nueva posición. Se puede, por un lado, crear una nueva variable de tipo 'robtarget' para luego definirla respecto a otra posición ya creada con anterioridad. O bien, por otro lado, se puede utilizar el comando Offs junto a la instrucción Move, sin necesidad de definir una nueva variable.

Así, en el caso del ejemplo que se muestra a continuación aparecen ambos métodos. En el primero se crea una posición llamada p1 que se encuentra desplazada 20 mm en los ejes X e Y y 50 mm en el Z respecto a Pos_ini_10. En el segundo, se ejecuta la acción de desplazar el robot siguiendo una trayectoria no lineal hacia un punto situado en la misma posición que la variable p1 antes creada. Es importante tener en cuenta que los desplazamientos relativos se hacen respecto a los ejes del objeto de trabajo en los que se encuentra Pos_ini_10, factor a tener en cuenta, ya que es la diferencia principal respecto a la instrucción RelTool.

```
VAR robtarget p1;
p1 := Offs(Pos_ini_10,20,20,50);


MoveJ Offs(Pos_ini_10,20,20,50),v1000,z100,Cilindro\WObj:=wobj0;
```

Con la instrucción RelTool es posible realizar movimientos respecto a otros puntos, pero en este caso utilizando los ejes cartesianos X, Y, Z relativos a la herramienta de trabajo utilizada, en vez de hacerlo respecto a un objeto de trabajo como se explicaba en la instrucción Offs. Además, debido a esto también se pueden realizar giros relativos a estos tres ejes, algo que con la instrucción Offs no era posible. Este detalle es de gran importancia, ya que permite modificar tanto la posición como la orientación de la herramienta en un punto, lo que será útil más adelante.

```
MoveJ RelTool(Pos_ini_10,50,10,30,\Rz:=45),v1000,z100,Cilindro\WObj:=wobj0;
```

En la línea del ejemplo, se realiza un desplazamiento no lineal de 50 mm respecto al eje X, 10 respecto al Y y 30 respecto al Z, referidos a los ejes de la herramienta Cilindro y relativos a Pos_ini_10, además de un giro de 45º respecto al eje Z de la misma herramienta.

En los programas a desarrollar en este trabajo, los controladores utilizados tienen asignados un conjunto de señales de entrada y salida digitales (Digital Inputs [DI], Digital Outputs [DO]). Para esperar que una señal de entrada digital se active o desactive se utiliza la instrucción WaitDI seguida de la entrada correspondiente y de un 1 o un 0, respectivamente.

```
WaitDI PiezaCinta,1;
```

Para activar y desactivar las salidas digitales se utilizan las instrucciones Set y Reset, respectivamente, seguidas de la señal de salida que se quiera modificar. Como se aprecia, para escribir un comentario se utiliza el comando '!' seguido de lo que se quiera comentar.

```
Set ActivaGiroMesa;!Activa el giro de la mesa
Reset ActivaGiroMesa;!Desactiva el giro de la mesa
```

Si se quiere que el programa espere un tiempo antes de ejecutar la siguiente instrucción se utiliza WaitTime seguido del número de segundos que se quiera esperar.

```
WaitTime 6; !Espera 6 segundos
```

Existen diversas formas de realizar bucles con RAPID, las principales instrucciones para esto son GOTO, FOR y el bucle IF para comparar variables. Estas instrucciones se utilizan como sigue.

```
Inicio: !Ejecución cíclica
GOTO Inicio; !Volvemos para procesar nueva pieza
```

```
FOR i FROM 1 TO 10 DO
ENDFOR
```

```
IF i=1 THEN

ENDIF
```

Con la instrucción GOTO se ejecuta de manera cíclica e indefinida el proceso que haya dentro de 'Inicio'. Con el comando FOR, para el ejemplo, se ejecuta cíclicamente en un bucle de 10 iteraciones las operaciones que se encuentren en el interior del bucle. En el caso de la instrucción IF, el programa entra en caso de que la variable i valga 1 e ignora dicho bucle en caso contrario.

Por último, para entrar en la rutina de un programa desde otro, simplemente se escribe el nombre del programa que se quiera llamar seguido de punto y coma.

2.2. Guía de diseño, programación y simulación de una estación (aplicación a una estación de ejemplo)

Tras saber utilizar los comandos básicos de Robotstudio, en este segundo subapartado se tiene como objetivo describir una guía de cómo crear una estación robotizada completa. Para ello, se ha diseñado y programado una estación de ejemplo como apoyo a la explicación, inventada en su totalidad por el autor.

Dicha estación automatizada está formada por los siguientes elementos:

- Robot 1: IRB 1600ID. Alcanzabilidad: 150 cm. Herramienta: Pistola de soldadura Tregaskiss 22 deg.
- Robot 2: IRB 2600. Alcanzabilidad: 165 cm. Herramienta: Pinza para manipulación de cilindros.
- Robot 3: IRB 2600. Alcanzabilidad: 185 cm. Herramienta: Imán para manipulación de cilindros.
- Piezas cilíndricas a fabricar.
- Cinta transportadora de entrada.
- Cinta transportadora de salida.
- Mesa de trabajo 1: mesa en la que se realizan las operaciones de soldadura.
- Plataforma giratoria: plataforma cilíndrica situada en la mesa de trabajo 1, en la que se colocan los cilindros para su operación de soldadura.
- Pinzas de agarre de cilindros.
- Mesa de trabajo 2: mesa en la que se colocan los cilindros ya montados.
- 2 cajas de colocación de cilindros, situadas en la mesa de trabajo 2.
- Bases de los tres robots.
- Vallas.

El proceso de producción automatizado que se efectúa en la estación es el siguiente:

Por la cinta transportadora de entrada van apareciendo piezas azules o verdes de forma aleatoria. Las piezas azules son dos cilindros, uno colocado encima del otro. Las piezas verdes son los dos mismos cilindros, añadiendo a su vez un tronco de cono encima de estos.

Al llegar el primer cilindro al final de la cinta de entrada, el robot 2 lo coge con la pinza y lo sitúa en la mesa de trabajo 1, encima de la plataforma giratoria. A continuación las pinzas de agarre se cierran girando para sujetar al cilindro, el robot 1 realiza la tarea de aproximación hacia éste para efectuar la soldadura y la plataforma ejecuta un giro de 360º. Si se trata de un cilindro azul se desarrollará una soldadura simple, mientras que si se tiene un cilindro verde se efectuará una soldadura doble.

Al finalizar el proceso de soldadura, las pinzas de agarre se abren hasta su posición inicial y el robot 2 vuelve a coger el cilindro ya soldado para situarlo en la cinta de salida.

Cuando el cilindro llega al final de la cinta transportadora de salida, el robot 3 lo coge y lo ubica en una caja de colocación de cilindros. En función de si el color del cilindro es azul o verde, el robot 3 lo coloca en la caja azul o verde, respectivamente.

El número máximo admisible de cilindros en cada caja es de cuatro, por tanto, una vez llenada una, ésta se resetea simulando la actuación de un operario sustituyéndola por una nueva. De esta forma, se consigue un proceso productivo automático en serie que se efectúa cíclicamente.

2.2.1. Diseño en Auto CAD de la geometría de la estación

En primer lugar, si se quiere diseñar una estación se han de crear los sólidos que la componen, ya que, como se ha visto, ésta no solo está formada por brazos robóticos, sino también por mesas, cintas transportadoras, vallas, etc.

Debido a que el programa Robotstudio dispone de una escasa cantidad de geometrías y de un conjunto de herramientas de creación y modificación de sólidos bastante limitado, se ha optado por la utilización del programa AutoCAD 2014 3D para el diseño de las piezas de la estación.

En este ejemplo, los brazos robóticos, la plataforma giratoria y las vallas son las únicas geometrías que se han importado directamente de la Biblioteca de Robotstudio. Las bases de los robots y la mesa de trabajo 2 son simplemente prismas rectangulares creados con Robotstudio, siendo el resto de geometrías diseño del autor con el programa AutoCAD antes mencionado, tal y como se muestra a continuación.

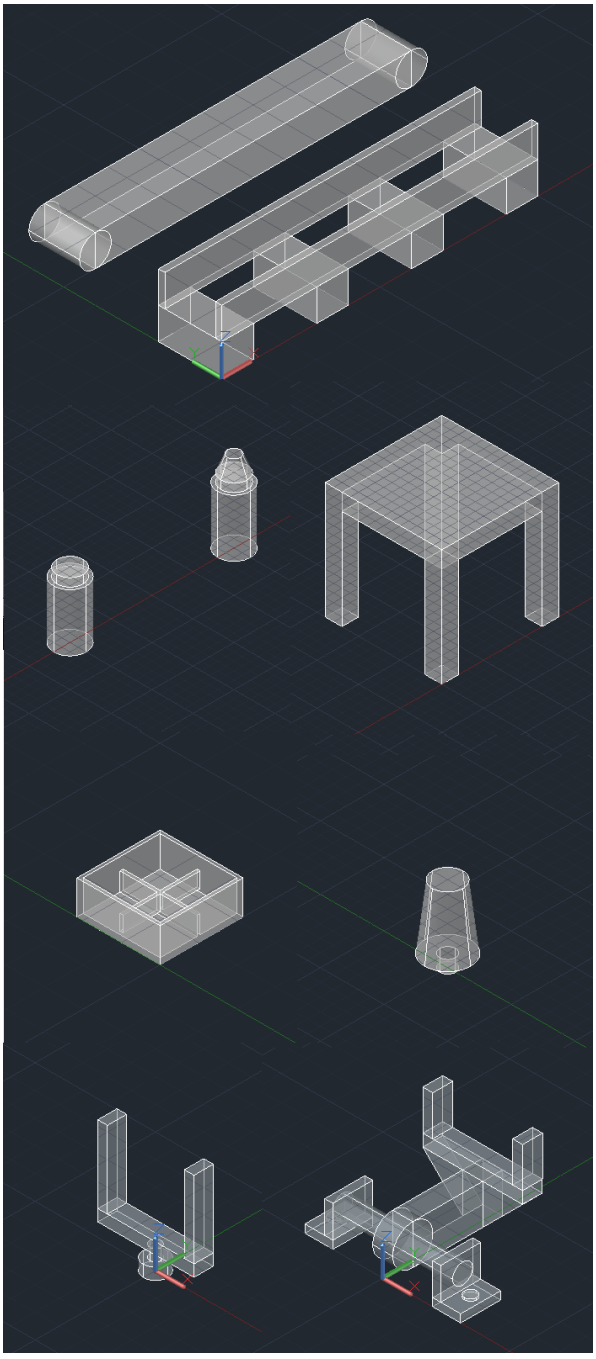


Figura 2-31**.** De arriba a debajo, de izquierda a derecha: cinta transportadora con su mesa de apoyo, cilindros azul y verde, mesa de trabajo 1, caja de colocación de cilindros, imán de manipulación, pinza de manipulación, pinza de agarre.

El conjunto de planos de las piezas se encuentran en el Anexo A de este trabajo.

Respecto a la importación de sólidos desde AutoCAD a Robotstudio el proceso es el siguiente.

Primero, en el programa AutoCAD hay que hacer clic sobre Exportar -> Otros formatos, tal y como se aprecia en la Figura 2-32. A continuación, se ha de guardar el archivo como tipo ACIS (.sat), ya que es el formato que admite Robotstudio.

Figura 2-32. Exportación de sólidos en Auto CAD.

El siguiente paso tras hacer clic en Guardar es designar los sólidos de Auto CAD que se quieren traspasar. Una vez señalados estos y pulsando el botón Enter ya se tienen guardados dichos sólidos en el archivo ACIS correspondiente.

Para importar dicho archivo en el programa Robotstudio, dentro de este programa es necesario ir a la Pestaña Inicio -> Importar geometría -> Buscar geometría… Una vez hecho esto, el siguiente y último paso es buscar el archivo .sat que se quiera importar y pulsar sobre Abrir.

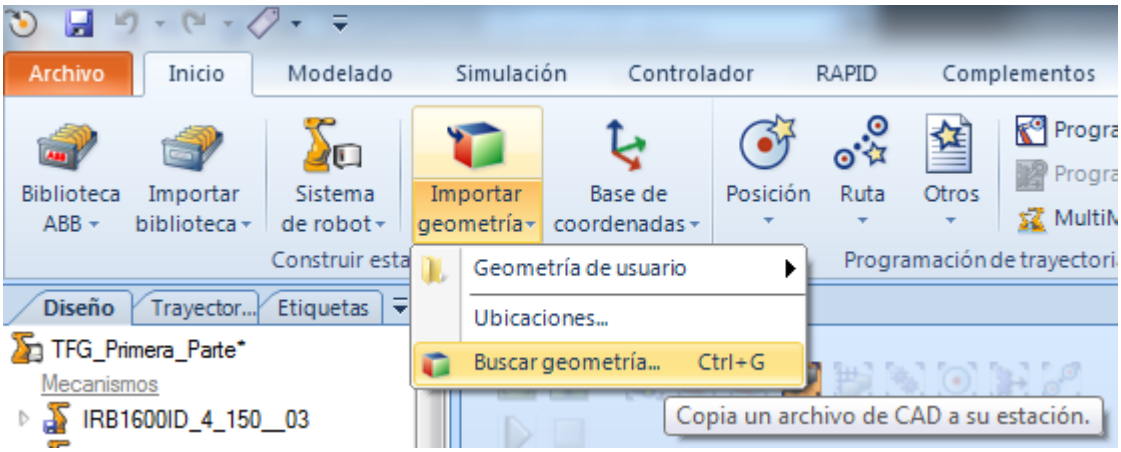


Figura 2-33. Importación de sólidos a Robotstudio.

Hacer notar que en todo momento se hace referencia a una exportación de sólidos, ya que los archivos tipo ACIS no admiten líneas, polilíneas, etc. Tan solo se permiten guardar sólidos y superficies.

Tras la importación de todos los sólidos y su posicionamiento relativo entre ellos, la estación queda como en la Figura 2-34.

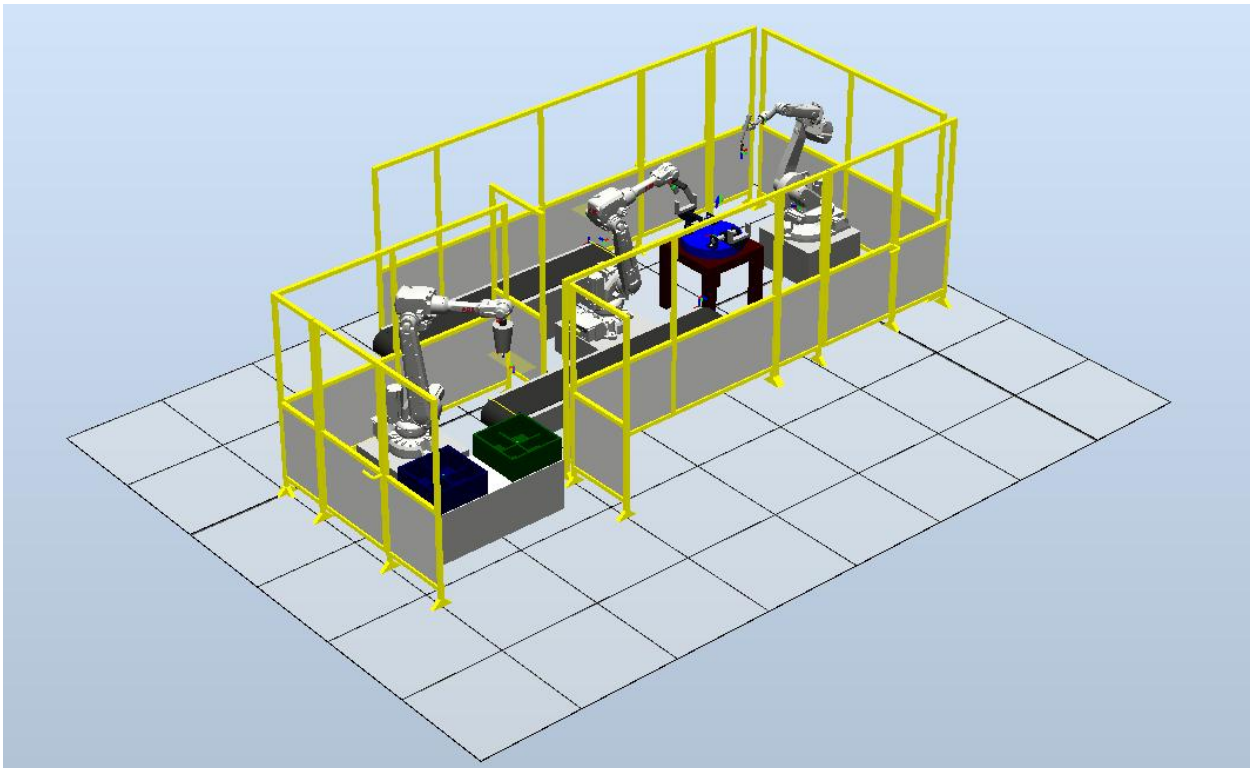


Figura 2-34. Estación completa.

2.2.2. Creación y diseño de los Smart Components

Los Smart Components o Componentes Inteligentes son elementos asociados a los sólidos, piezas o robots de la estación, los cuales tienen un comportamiento controlado por señales y propiedades del sistema.

Estos Smart Components (a partir de ahora SC) tienen un conjunto de componentes subordinados muy amplio, los cuales se dividen en 6 categorías:

1. Señales y propiedades: dentro de esta categoría se pueden encontrar puertas lógicas, contadores, temporizadores, expresiones matemáticas, convertidores y demás elementos para modificar una señal o una propiedad del sistema a programar.
2. Primitivos paramétricos: en este apartado existen los componentes necesarios para crear de forma automática sólidos y líneas, así como para generar copias de componentes gráficos ya existentes.
3. Sensores: está categoría reúne el conjunto de sensores a utilizar en los procesos de producción automático de este trabajo. Incluye sensores de colisión, de línea, de superficie, volumétricos, etc.
4. Acciones: hace referencia a componentes como conectar o desconectar dos objetos entre sí, eliminar un objeto o simplemente hacerlo visible/invisible.
5. Manipuladores: este apartado reúne los componentes necesarios para mover un objeto de forma lineal, hacerlo rotar un ángulo dado, moverse a lo largo de una curva o posicionarlo en un lugar determinado. También permite mover los ejes del mecanismo de un brazo robótico a una posición elegida.
6. Otros: en esta última categoría se encuentra un conjunto de SC de distinta variedad: representación de una cola de objetos, generación de un número aleatorio, detención de la simulación, reproducción de un sonido, etc.

Para la explicación de cómo se diseñan SC el autor se va a apoyar en la estación de ejemplo. Por tanto, una vez se tiene la geometría de la estación completamente diseñada es necesario "dotar de inteligencia" los obejtos que correspondan utilizando los SC para que se efectúen las operaciones convenientemente.

Con esto, se han creado un total de 7 SC que se pasarán a describir y explicar detalladamente a continuación.

2.2.2.1. SC Cinta Entrada

El primer SC a describir es el de la cinta transportadora de entrada de cilindros. Los componentes inteligentes asociados a la cinta de entrada son simplemente dos. Un sensor de línea denominado Fin_cinta y un sensor de superficie llamado Tubo_verde. El primero simula un sensor de distancia que detecta la llegada de un cilindro al final de la cinta, mientras que el segundo detecta si el cilindro es de tipo verde o azul.

El diseño de este SC es muy sencillo, de forma que solo tiene una entrada y dos salidas. La entrada Activa_Sensores activa ambos sensores antes mencionados. La salida Pieza_Cinta está unida al sensor de línea, activándose en caso de que un cilindro llegue al final de la cinta. La salida Cilindro_Verde se activa en caso de que el sensor de superficie detecte a un cilindro de color verde al final de la cinta. Esto se consigue gracias a que dicho sensor está situado a una altura tal que detecta el tronco de cono propio de los cilindros de color verde. Por tanto, en caso de llegar al final de la cinta un cilindro azul no detectaría nada y dicha señal de salida no se activaría.

El esquema de bloques del diseño de este SC se muestra en la figura 2-35.

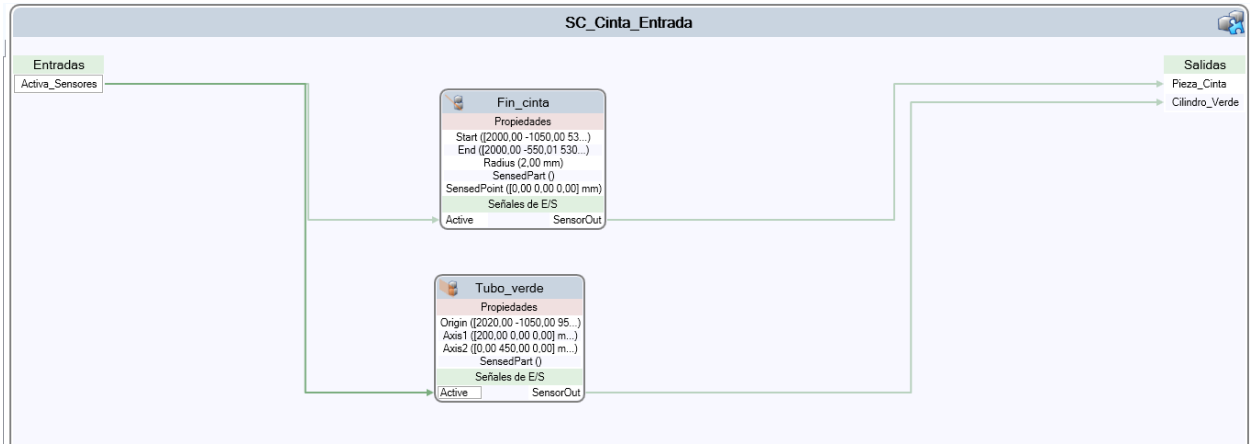


Figura 2-35. Diseño del SC de la cinta de entrada.

2.2.2.2. SC Cinta Salida

En este SC se tienen dos sensores similares a los expuestos en el SC de la cinta de entrada, denominados Fin_cinta_salida al sensor de línea y Tubo_verde_2 al sensor de superficie. Además, la entrada asociada a estos sensores se llama ahora Activa_Sensor_Final_CintaSalida, mientras que las salidas del SC esta vez se denominan Pieza_Final_Cinta_Salida y Tubo_Verde_2, respectivamente.

Sin embargo, el SC de la cinta de salida tiene una diferencia respecto al de entrada. Esto es debido a que el robot 2 coloca los cilindros en el inicio de la cinta de salida en una posición dada, pero en ocasiones el robot puede situar un cilindro en dicha posición con cierto error de medida.

Por tanto, para asegurar una precisión en la colocación de cada cilindro en la cinta de salida se añade al diseño anterior un sensor de superficie al inicio de la cinta y a una altura media, de forma que cuando el sensor detecte al cilindro éste se ubique en la posición exacta deseada gracias a un componente de posicionamiento (posicionador). Este posicionador se ejecutará en caso de que la pieza haya dejado de estar cogida por el robot y el sensor de superficie lo esté detectando correctamente.

Para esta mejora del diseño serán necesarias dos entradas y una salida más a este SC, quedando el esquema de bloques tal y como se aprecia en la Figura 2-36.

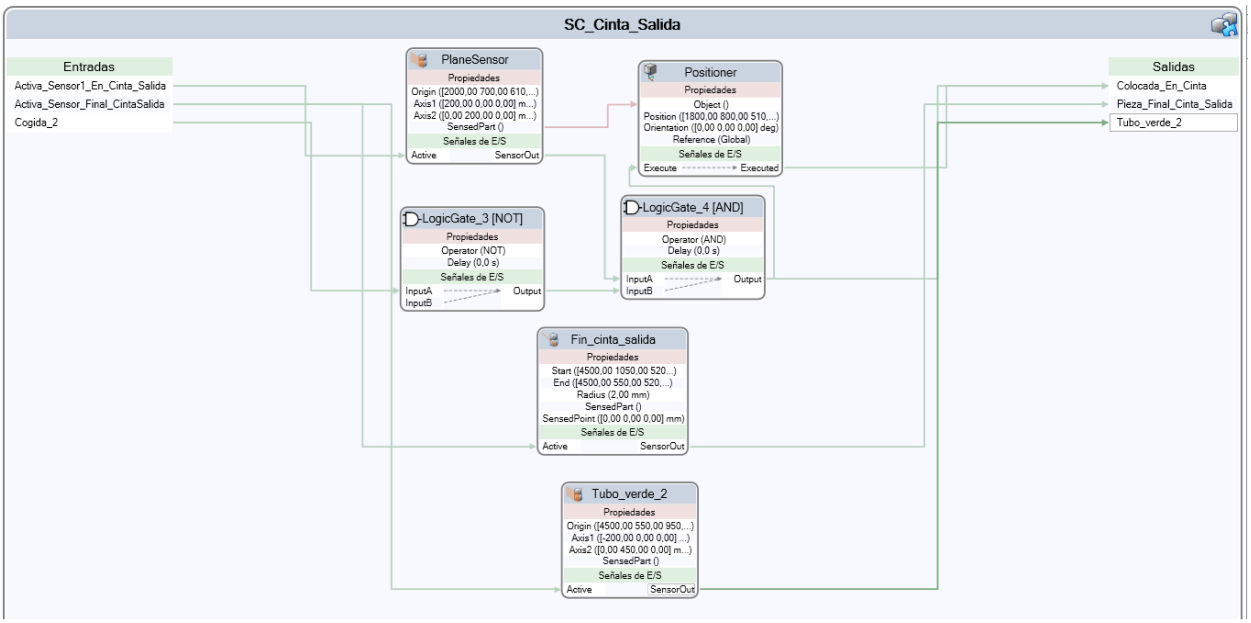


Figura 2-36. Diseño del SC de la cinta de salida.

2.2.2.3. SC Pinza

El SC Pinza se encarga de la labor a realizar por parte de la pinza herramienta del robot 2, la cual es la manipulación de los cilindros entre las dos cintas transportadoras y la mesa de trabajo 1. Con esto, se ha dispuesto de un sensor de superficie en la pinza, de forma que cuando detecte un cilindro y se active mediante la señal de entrada Activa_Sensor, dicho cilindro se pegue a la pinza mediante el componente Attacher. Así, se envía una señal para activar la salida Pieza_en_pinza.

Cuando se quiera despegar el cilindro de la pinza se desactivará la entrada Activa_sensor, ejecutando así el componente Detacher, el cual permite la separación del cilindro y la pinza y a su vez envía un reseteo a la salida Pieza_en_pinza. Entre la desactivación de la señal de entrada y la ejecución del Detacher se realiza un Delay o retardo de 0.5 segundos para asegurar que dicha ejecución se realiza satisfactoriamente.

El esquema del SC antes explicado se presenta a continuación en la Figura 2-37.

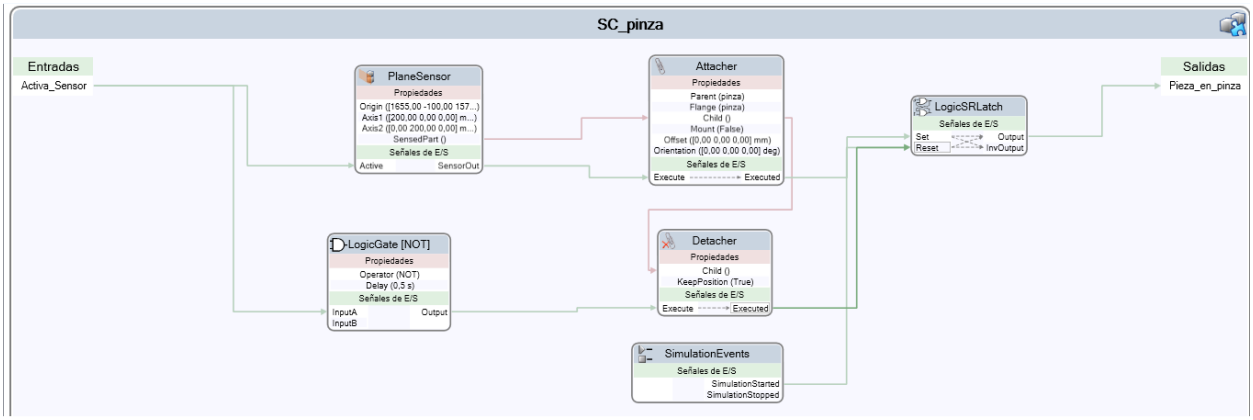


Figura 2-37. Diseño del SC de la pinza del robot 2.

2.2.2.4. SC Imán

Este SC corresponde al imán herramienta del robot 3. Dicha herramienta se encarga de coger los cilindros de la cinta transportadora de salida y clasificarlos en cajas según sean verdes o azules.

De esta forma, el diseño de SC Imán es similar al de SC Pinza explicado anteriormente, con la salvedad del sensor detector de cilindros. En este caso, se ha colocado un sensor de línea en el extremo de la herramienta cónica, en vez de un sensor de superficie como el de la pinza.

El esquema de bloques se presenta en la Figura 2-38.

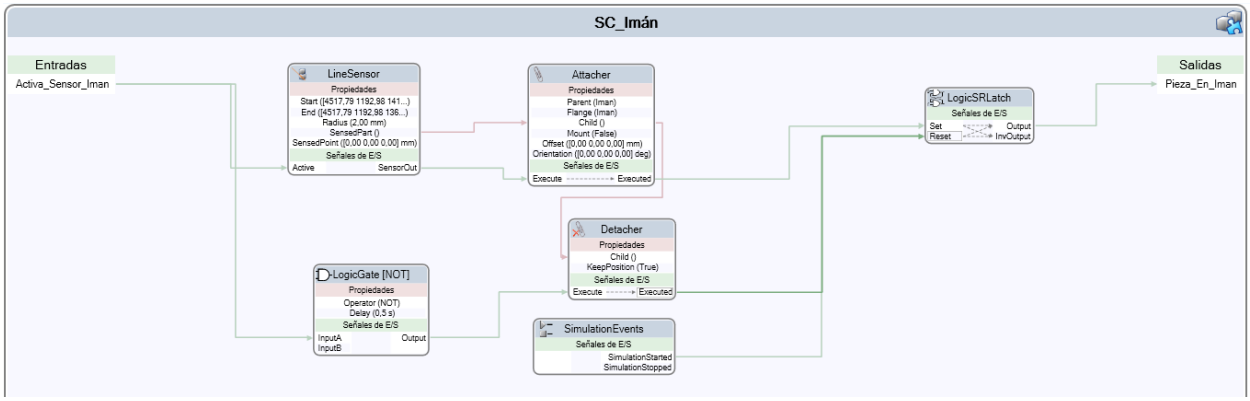


Figura 2-38. Diseño del SC del imán del robot 3.

2.2.2.5. SC Pinzas Agarre

El SC destinado al movimiento de las pinzas de agarre se encarga de que éstas giren 90º y -90º, para que sujeten a los cilindros cuando son colocados en la plataforma giratoria encima de la mesa de trabajo 1. Esto es necesario ya que las piezas deben estar bien sujetas cuando se produzca la soldadura de las mismas mientras la plataforma gira.

De esta manera, existen dos señales digitales de entrada, Gira_pinzas_cierre y Gira_pinzas_apertura, cuya activación permite ejecutar los movimientos de rotación para que se cierren o se abran las pinzas de agarre respectivamente.

Con esto, existen también dos salidas digitales, Pinzas_cerradas y Pinzas_abiertas, que se activan cuando se ha ejecutado el movimiento de cierre o apertura, respectivamente. El esquema de bloques del diseño es como sigue.

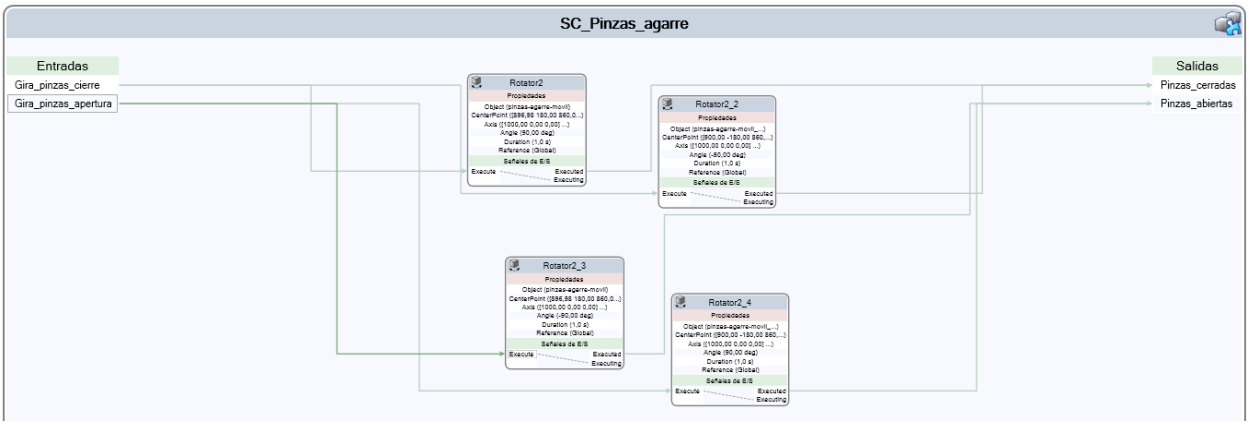


Figura 2-39. Diseño del SC de las pinzas de agarre.

2.2.2.6. SC cilindro giratorio

El SC de la plataforma cilíndrica giratoria controla el giro sobre el eje Z de dicha plataforma y de las pinzas de agarre. Por tanto, cuando se activa la señal de entrada digital Giro_mesa_trabajo se produce la rotación del cilindro y de las partes fijas y moviles de ambas pinzas de agarre. Dichas pinzas se han dividido en dos piezas independientes, ya que una está fija a la plataforma cilíndrica y otra se cierra y abre para sujetar al cilindro a soldar, tal y como se expuso en el subapartado anterior.

Así mismo, se ha colocado un sensor de superficie en el centro de la plataforma, para cuando detecte al cilindro y no esté cogido por la pinza lo coloque exactamente en dicho centro. Al igual que ocurría en el inicio de la cinta de salida, esto se hace para corregir los posibles errores de posicionamiento del cilindro a la hora de colocarlo por el robot 2 en la plataforma. El esquema del diseño del SC se muestra en la Figura 2-40.

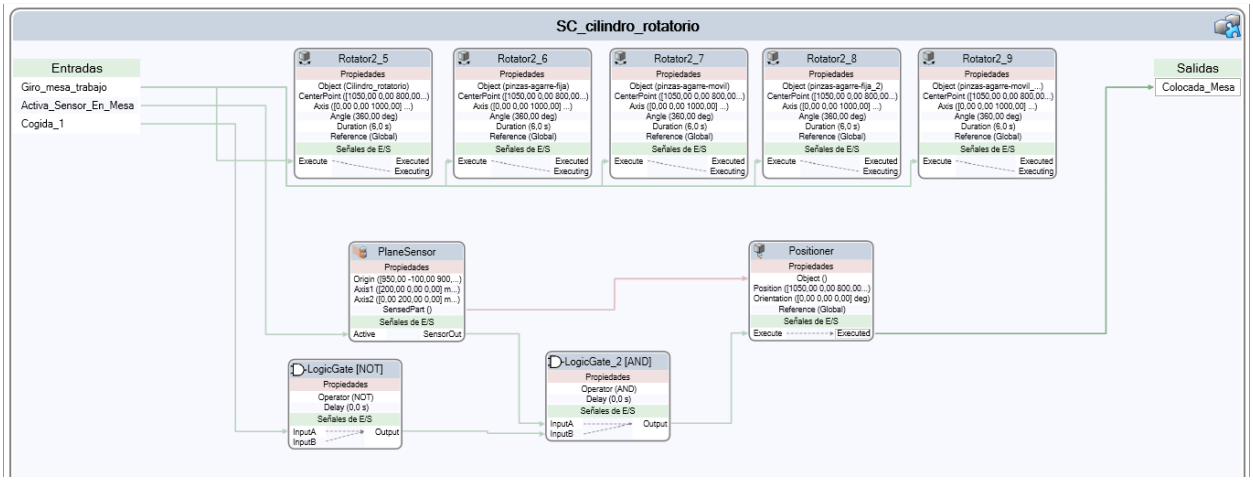


Figura 2-40. Diseño del SC de la plataforma cilíndrica rotatoria.

2.2.2.7. SC Piezas

El último SC a describir es el dedicado a las piezas a procesar, es decir, a los cilindros. Se puede decir que es el más importante y complejo de todos los SC vistos, ya que controla la creación y todos los movimientos de los dos tipos de cilindros con los que se trabaja en esta estación.

En primer lugar, para la creación de los cilindros se realiza lo siguiente: partiendo de dos cilindros originales (uno verde y otro azul), situados ambos en el origen de coordenadas (sin que sea posible su visibilidad) se efectúa una copia de uno de los dos. La elección de cuál de los dos cilindros copiar se realiza de forma aleatoria mediante el uso de un componente random.

Una vez realizada la copia y tras pasar un tiempo aleatorio entre 4 y 8 segundos desde el inicio de la simulación, dicha copia del cilindro se posiciona en el principio de la cinta transportadora de entrada. Al instante de transportar al cilindro a dicha posición se aplica un movimiento lineal sobre éste, de forma que simula su movimiento sobre la cinta transportadora de entrada hasta ser detectado por el sensor de línea del final de la cinta como se explicó en el subapartado del SC de la cinta de entrada. Una vez que aparece en la cinta el primer cilindro, este proceso de copia, posicionamiento y movimiento lineal por la cinta se efectúa cíclicamente apareciendo cada cilindro un tiempo aleatorio entre 4 y 8 segundos después, respecto a la aparación del cilindro anterior.

Para que aparezca en la cinta uno de los dos cilindros de forma aleatoria entre un intervalo de tiempo dado se utilizan los componentes random, comparador y temporizador, conectados entre sí como se observa en la Figura 2-41.

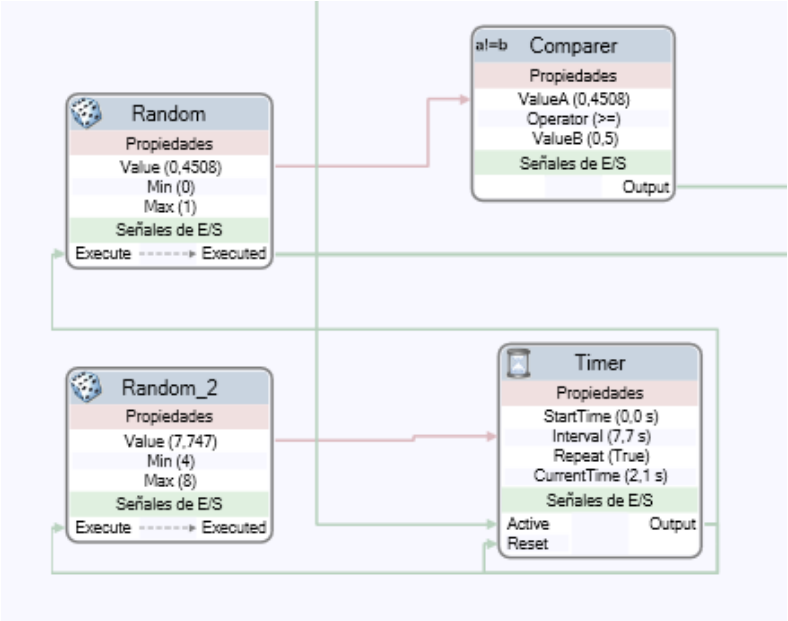


Figura 2-41. Randoms, temporizador y comparador.

Por otro lado, para poder seguir la trazabilidad de los cilindros durante el procesado es necesario encolarlos mediante el componente Queue. De hecho, lo que se ordena mover linealmente mediante el componente LinearMover es el conjunto de cilindros que se encuentren en Queue, por lo que se encolarán en una primera cola los cilindros que se vayan creando y posicionando en la cinta de entrada.

Sin embargo, debido a esto será necesario crear más de una cola, ya que en la cinta de entrada la cola de objetos se mueve en una dirección y sentido determinado y en la cinta de salida se mueven en la misma dirección pero sentido opuesto (otra cola con otro LinearMover distinto). Además, los cilindros deben estar siempre encolados (hasta cuando estén en la mesa de trabajo 1, donde no se mueven) para tener una trazabilidad exacta del material, es decir, para conocer el orden de llegada de los cilindros en función de su tipo (azul o verde) y que finalmente el robot 3 pueda clasificarlos en sus respectivas cajas correctamente.

Por eso, ha sido necesaria la creación de más de una cola de objetos para el diseño de este SC, con una correcta coordinación de encolamiento/desencolamiento en cada una de ellas y así asegurar que no se pierda la trazabilidad de los cilindros en todo el proceso.

Como se verá más adelante, las cajas donde finalmente se colocan los cilindros se tratarán como si fueran matrices 2x2. En la posición (2, 2) de cada caja, en la cual se ubica el cuarto cilindro de cada color, se ha colocado un sensor de superficie de forma que cuando detecte a una pieza cilíndrica significará que la caja está llena. Como se dijo, cuando una caja se complete con cuatro cilindros, dichas piezas desaparecen simulando la sustitución de la caja por una vacía. Por tanto, en el diseño de este SC se ha dispuesto dicho sensor de forma que cuando detecte un cilindro se espere un tiempo de cuatro segundos y a continuación se eliminen los objetos de la cola correspondiente, es decir, los cuatro cilindros colocados en la caja.

El esquema de bloques completo con la creación de cilindros a partir de los originales, los diferentes encolamientos de éstos distinguiendo en todo momento entre los dos tipos de cilindros a procesar, se muestra en la Figura 2-42.

Cabe mencionar que este SC tiene varias entradas digitales y no tiene ninguna salida. En concreto, tiene 6 señales digitales de entrada, de las cuales 4 son órdenes externas que permiten desencolar los objetos (cuando sea oportuno y en función de varios sensores de las otras SC) en las diferentes Queue que se han tenido que crear por lo expuesto anteriormente. A parte, existen otras dos entradas, Motor_Cinta_Entrada y Motor_Cinta_Salida, que permiten la activación de los movimientos lineales de los cilindros en las cintas de entrada y salida respectivamente.

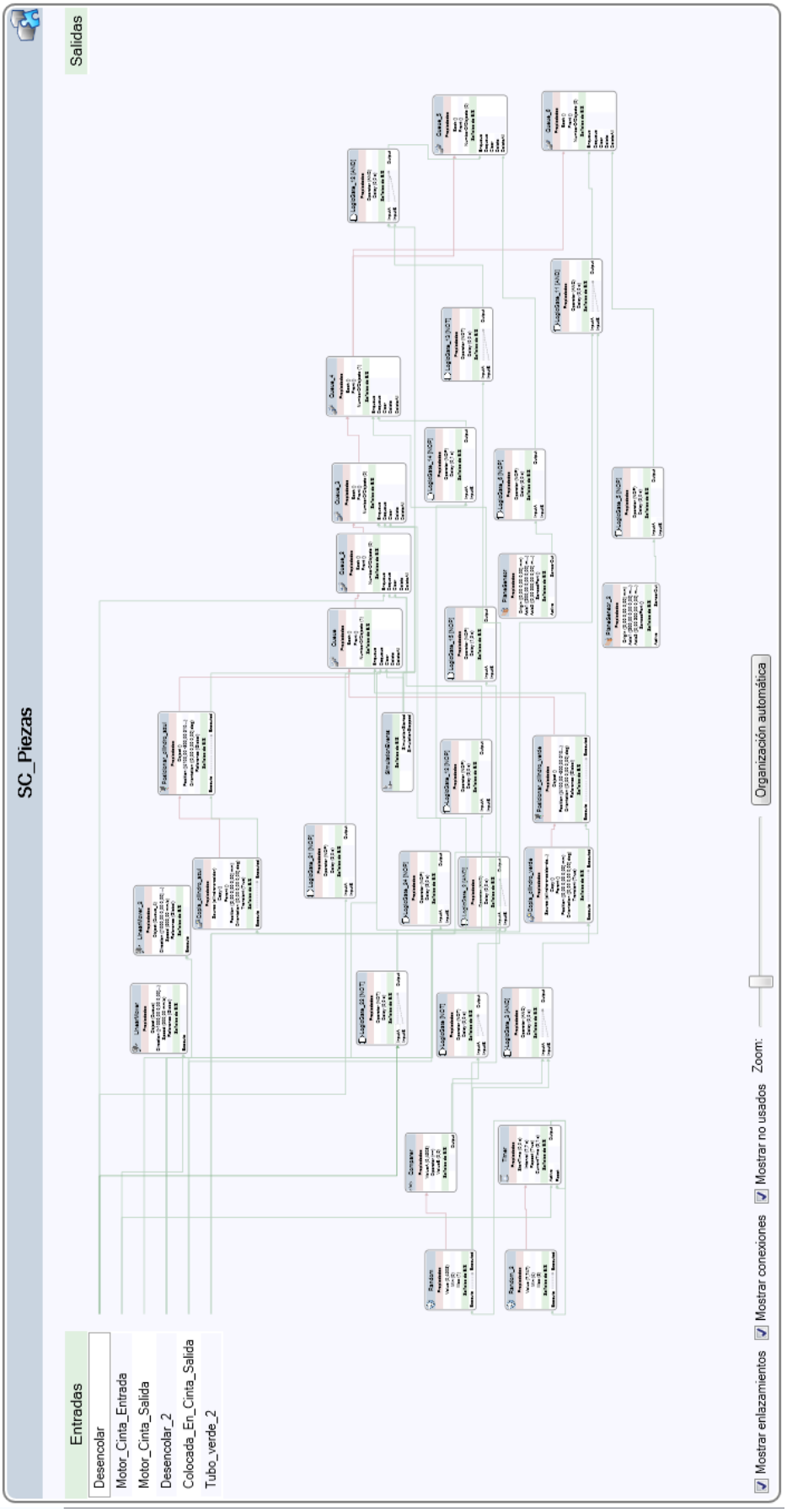


Figura 2-42. Diseño completo del SC de creación y procesado de las piezas cilíndricas.

La conexión de todos los SC expuestos en este apartado entre sí, para el correcto funcionamiento íntegro del sistema automatizado, se verá en el apartado 2.2.4. Lógica de estación.

2.2.3. Creación de módulos de E/S

Tras la creación de la geometría de la estación y dotar de inteligencia a las piezas necesarias por separado, el siguiente paso es "atribuirle inteligencia" a los tres brazos robóticos que componen la estación. Para ello, lo primero que hay que hacer es crear los controladores asociados a dichos robots, tal y como se explicó en el apartado 2.1.3. Creación del controlador de un robot.

Debido a que los robots 1 y 2 necesitan estar coordinados entre sí para la distribución y soldadura de cilindros en la mesa de trabajo 1, los controladores se han dispuesto de forma que uno solo controle a estos dos robots. El robot 3, al tener una tarea más independiente del resto de robots es dirigido por un segundo controlador. Por tanto, el control de los robots quedaría así:

- Controlador 1: robots 1 y 2.
- Controlador 2: robot 3.

Una vez creados los controladores, es necesario crear un módulo de entradas y salidas en cada uno de ellos. Para esto, dentro de la Pestaña Controlador y en la ventana izquierda con el mismo nombre, se hace clic en Configuración y dentro de éste en I/O. Al hacer esto se abrirá una ventana anexa a la derecha con un listado de elementos a configurar dentro de dicho controlador.

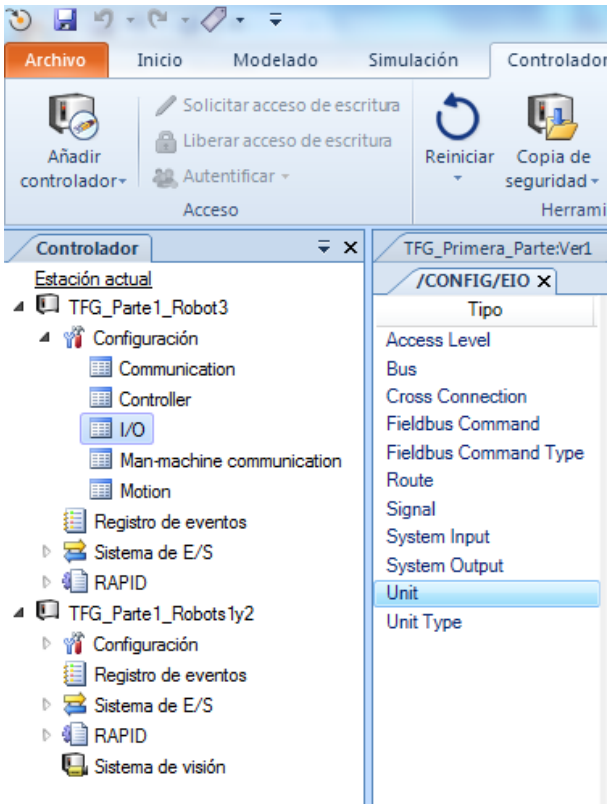


Figura 2-43. Configuración de un controlador.

El primer elemento a configurar es la unidad (Unit). Más bien, el objetivo es crear una nueva unidad. Para ello, se hace clic con el botón derecho del ratón sobre Unit y se pulsa sobre "Nuevo Unit…". A continuación saldrá una ventana de edición, en la que se pondrá como nombre 'ModuloES' y se conectará con el bus DeviceNet1. El tipo de unidad no afecta, por lo que se pone una cualquiera y como etiqueta de identificación se coloca el mismo nombre. El resto de parámetros se dejan por defecto.

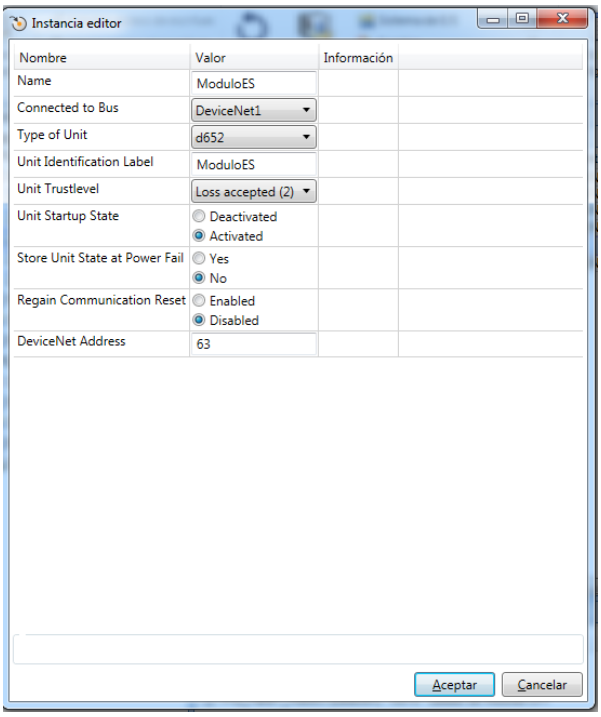


Figura 2-44. Editor del módulo de E/S.

Luego de crear el módulo, hay que establecer las señales de entrada y salida asociadas al mismo. De esta manera, se crearán un conjunto de señales de entrada digitales del controlador que corresponderán a las señales de salida de los SC que sean necesarias para el control de algún robot. Así mismo, también se establecerán un conjunto de señales de salida del controlador correspondientes a señales de entrada de los SC cuyo control provenga de alguno de los robots, en vez de un controlador externo ajeno al sistema automatizado.

Para crear entradas o salidas asociadas a un controlador hay que dirigirse a Signal, dentro de configuración e I/O como antes. A continuación hay que clicar con el botón derecho sobre éste y y pulsar sobre 'Nuevo Signal…'.

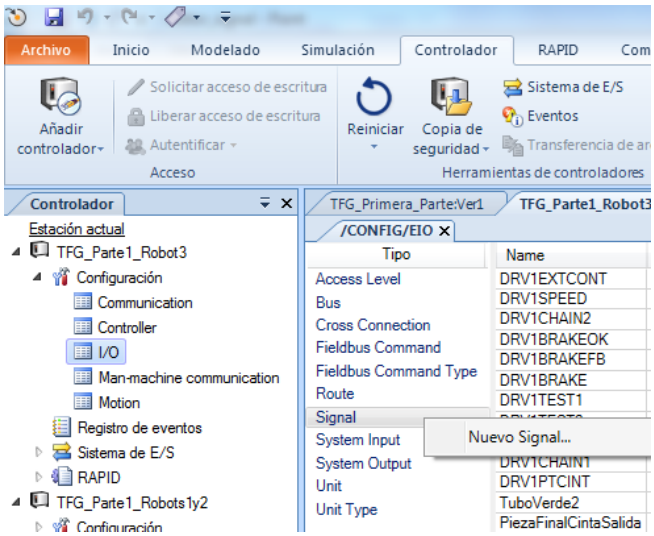


Figura 2-45. Configuración de una señal.

Para editar la señal hay que rellenar su nombre, tipo, unidad (ModuloES), etiqueta y mapearlo con un número que parte del cero en adelante, teniendo en cuenta que dicho mapeo es independiente entre las entradas y las salidas. Dicha edición se ejemplifica con la Figura 2-46 que se muestra a continuación.

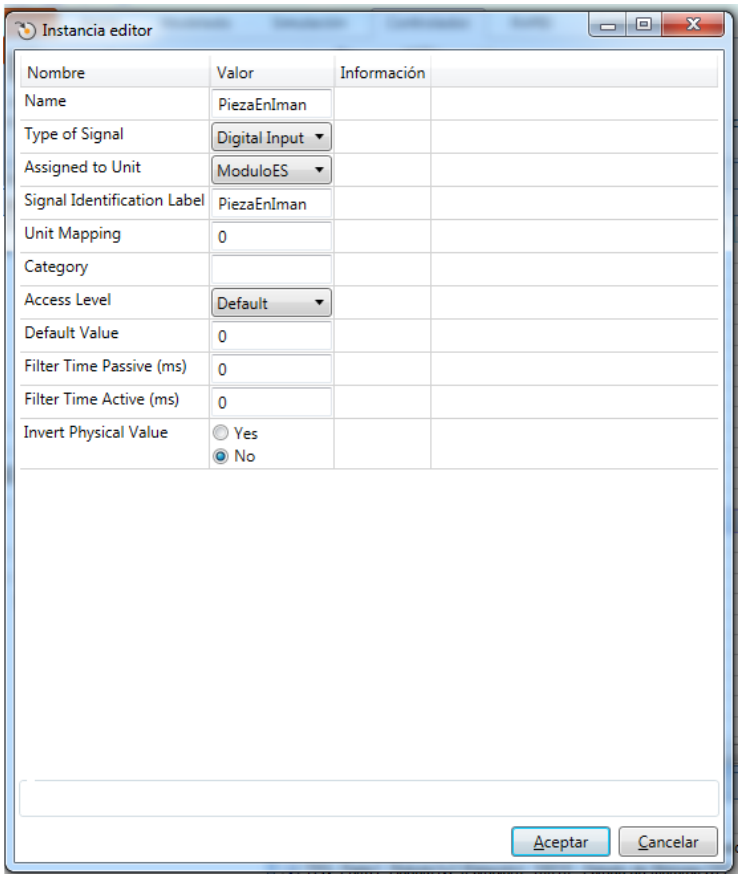


Figura 2-46. Editor de una señal de un controlador.

El listado completo de las entradas y salidas de los controladores 1 y 2 se encuentra en el Anexo B de este trabajo.

2.2.4. Lógica de estación

Tras crear los controladores de los robots y los SC es necesario interconectar de forma adecuada todos estos entre sí, paso indispensable para que el sistema funcione correctamente coordinando entre sí los movimientos automáticos de los robots y los sólidos inteligentes. Para ello, hay que acceder a la Pestaña Simulación y dentro de ésta a Lógica de estación.

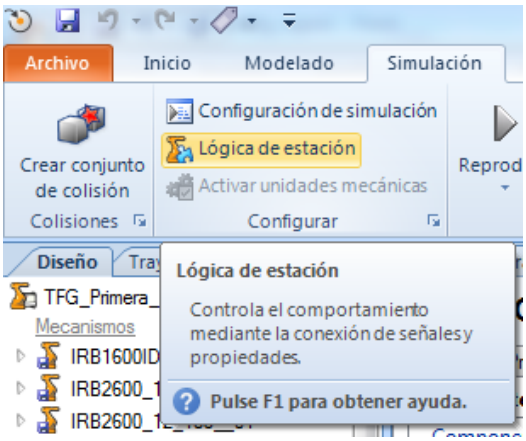


Figura 2-47. Acceso a la lógica de estación.

La lógica de la estación está formada por un conjunto de bloques correspondientes a los siete SC y los dos controladores anteriormente creados. Además, se pueden añadir otros componentes inteligentes subordinados como los vistos en el diseño de los SC, aunque en este sistema eso no será necesario. Para poder observar dicho esquema de bloques una vez abierta la lógica de estación, al igual que con los SC, hay que pulsar sobre la pestaña Diseño.

La conexión entre estos bloques es tal que las entradas a los controladores son salidas de los SC y viceversa, como se explicó en el subapartado anterior. En la Figura 2-48 se puede apreciar como queda el diseño.

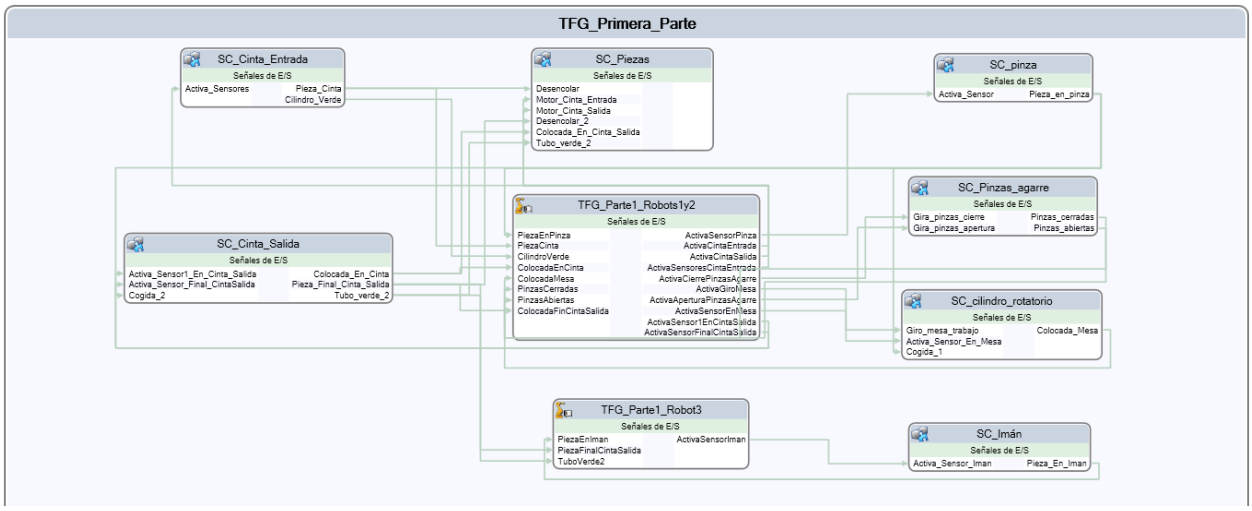


Figura 2-48. Lógica de estación del sistema completo.

Es interesante destacar que el controlador 1 (controlador de los robots 1 y 2) se encarga de activar las cintas transportadoras, el giro de la plataforma de la mesa de trabajo 1, los giros de las pinzas de agarre y todos los sensores del sistema excepto el sensor del imán, cuya activación la realiza el controlador 2.

A su vez y como consecuencia, al controlador 1 le llegan como entradas las señales de todos los sensores para que se programe su activación/desactivación, excepto las señales del sensor del imán del robot 3 y la de los dos sensores colocados al final de la cinta transportadora de salida, los cuales son gestionados por el controlador 2.

Como se puede apreciar, el controlador 1 tiene un mayor peso en los SC que el segundo controlador, al gestionar dos robots en vez de uno y centrarse en un conjunto de operaciones con un número de sensores y actuadores más elevado.

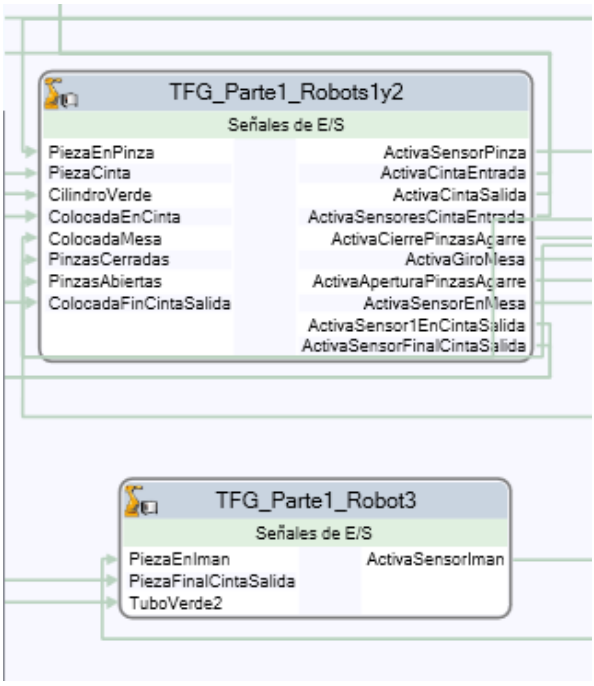


Figura 2-49. Lógica de estación de los dos controladores.

2.2.5. Programación en RAPID

A estas alturas, tras haber diseñado todos los SC, creado los controladores y conectarlos todos entre sí, lo único que queda para terminar es programar en RAPID los movimientos de los tres robots, así como la programación del control de las señales de los controladores asociadas a los SC que se encuentran en los módulos de E/S, para así manejar también con RAPID los componentes inteligentes oportunos.

Teniendo en cuenta que el código completo de los programas se encuentra en el Anexo C de este trabajo, se comenzará a describir la programación del robot 2, perteneciente al controlador 1, el cual se encarga de los movimientos de manipulación de cilindros entre la mesa de trabajo 1 y las cintas de entrada y salida.

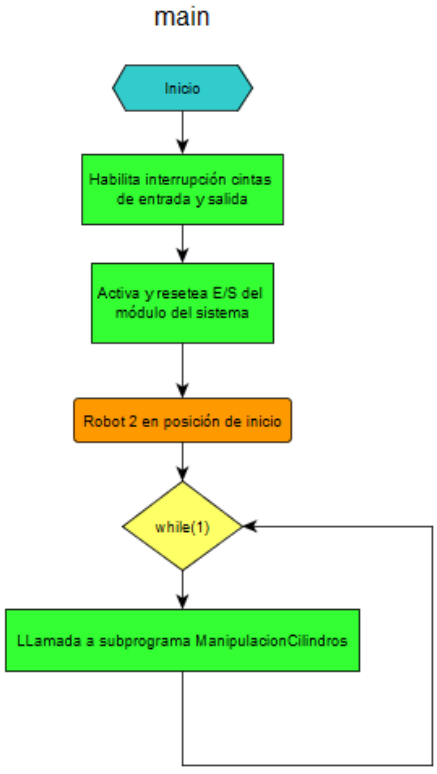
Nota: para un mejor entendimiento de la explicación del código, se muestran paralelamente los pseudocódigos de los subprogramas, realizados con el software yEd Graph Editor.

El programa del robot 1 está a su vez compuesto por dos subprogramas tipo TRAP y dos subprogramas tipo PROC. Un subprograma PROC es el principal, denominado main, y el otro, llamado ManipulacionCilindros, es subordinado.

Al ser el controlador 1 el encargado de monitorizar el movimiento de las cintas transportadoras de entrada y salida, en el programa de RAPID del robot 2 se crean dos suprogramas tipo TRAP, llamados ParaCintaEntrada y ParaCintaSalida que gestionan la parada de los motores de las cintas reseteando las señales de salida del módulo E/S de este primer controlador, ActivaCintaEntrada y ActivaCintaSalida.

Para ello, a su vez se ha diseñado una programación de las cintas utilizando variables de interrupción (intnum), Pieza_preparada_entrada y Pieza_preparada_salida, de forma que cuando un cilindro llegue al final de una de las dos cintas, se habilita su interrupción conectando la variable de interrupción con su subprograma TRAP que le corresponda. Diha conexión se efectúa dentro del subprograma PROC principal 'main', utilizando el comando'IEnable' y la instrucción CONNECT (intnum) WITH (trap). Habrá que habilitar por tanto uno para la cinta de entrada y otro para la de salida.

Con todo esto se consigue que cada vez que un cilindro sea detectado por un sensor de final de cinta su movimiento se pare, pero en cuando un robot coja el cilindro y el sensor deje de detectarlo, la cinta vuelva a funcionar hasta llegar al final la siguiente pieza, consiguiéndose un control cíclico desde RAPID por parte del robot 2 sobre el movimiento de las cintas muy eficiente.

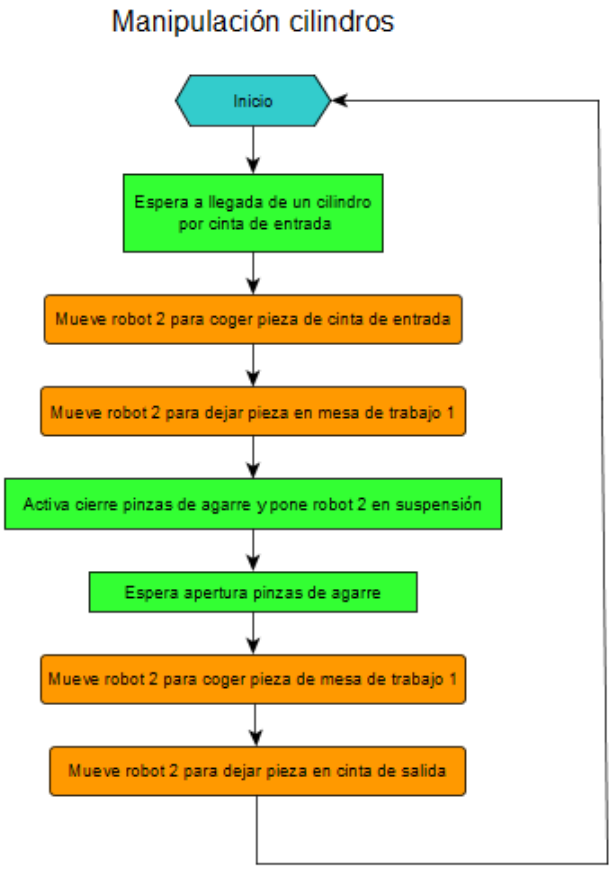


Tras la programación de la interrupción de las cintas es necesario programar el estado inicial activando todos los sensores de ambas cintas y poniendo en marcha el motor de la cinta de entrada utilizando la instrucción Set sobre las señales digitales de salida que corresponden del módulo E/S como se explicó en el subapartado 2.1.6. Introducción a RAPID. A su vez, también hay que colocar el robot 2 a su posición inicial Pos_ini.

A continuación, se efectúa una ejecución cíclica del subprograma PROC denominado ManipulacionCilindros mediante el uso de la instrucción GOTO, para así procesar las piezas cilíndricas dentro de un bucle infinito.

Con esto se cierra el subprograma PROC main y se inicia el código del subprograma ManipulacionCilindros.

En ManipulacionCilindros se efectúan todos los movimientos del robot 2, comenzando con la espera de éste a la llegada de un cilindro al final de la cinta de entrada, aproximación y cogida del cilindro y colocación en la plataforma giratoria de la mesa de trabajo 1.



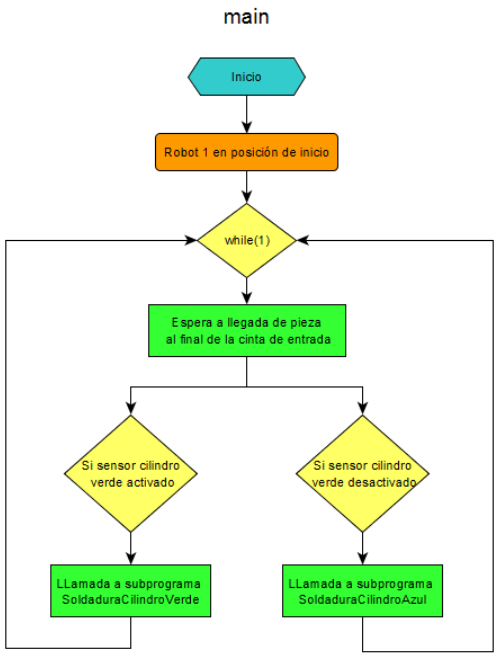
Una vez colocado el cilindro, se activa el cierre de las pinzas de agarre, el robot 2 vuelve a su posición inicial y se pone en suspensión temporal, a la espera de que se abran las pinzas de agarre, orden que le llega desde el programa del robot 1 que se explicará a continuación.

De esta forma, la comunicación entre los robots 1 y 2 (pertenecientes al mismo controlador) es mediante las señales PinzasCerradas, PinzasAbiertas, ActivaCierrePinzasAgarre y ActivaAperturaPinzasAgarre. Como las 4 pertenecen al módulo E/S del controlador 1, la comunicación entre ambos robots se consigue con señales internas del mismo.

Tras abrirse las pinzas de agarre, el robot 2 se aproxima a coger el cilindro ya soldado y lo coloca en la cinta de salida, todo mediante movimientos de aproximación. Por último, con el comando Set se activa el motor de la cinta transportadora de salida ActivaCintaSalida.

Cabe mencionar que para coger un cilindro con la pinza se activa el sensor de superficie de la pinza y se espera a que se ponga a uno PiezaEnPinza. Para soltar un cilindro se resetea (desactiva) el sensor de superficie y se espera a que se ponga a cero PiezaEnPinza.

Tras describir el programa del robot 2 se va a pasar a explicar el código del robot 1, perteneciente al mismo controlador y, como ya se ha visto, en coordinación con el robot 1.



El programa del robot 2 está compuesto por 3 subprogramas tipo PROC. Uno principal denominado main, y otros dos programas destinados a la soldadura. Uno llamado SoldaduraCilindroAzul en caso de procesar un cilindro azul u otro denominado SoldaduraCilindroVerde en caso de tratarse un cilindro verde.

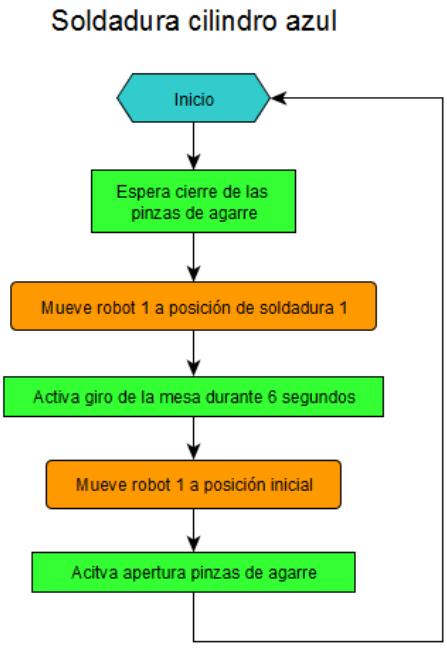
En el subprograma main se posiciona el robot 1 en su estado inicial y se realiza una ejecución cíclica de llamada de los programas de soldadura de los cilindros azul o verde con la instrucción GOTO. Para elegir si entrar en el subprograma del cilindro azul o verde, se espera a que el cilindro a soldar llegue al final de la cinta de entrada, y una

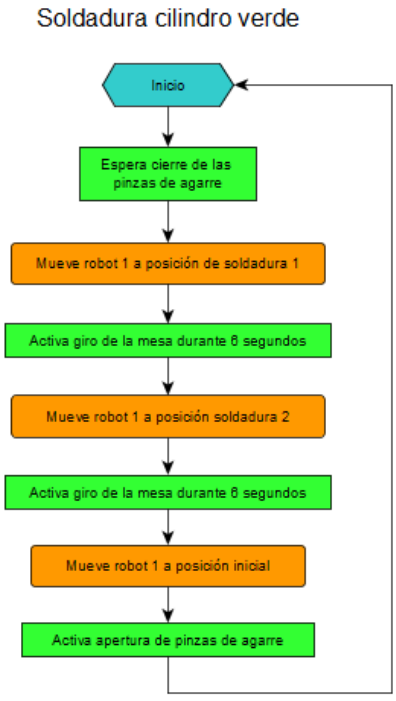
vez ahí se comprueba con un bucle IF si el sensor CilindroVerde está activado o no. En caso afirmativo entra en el subprograma de la soldadura del cilindro verde y en caso contrario en el del cilindro azul.

En el subprograma SoldaduraCilindroAzul el robot 1 está inicialmente en suspensión, a la espera a que desde el programa del robot 2 se active el cierre de las pinzas de agarre para realizar su movimiento de aproximación y colocarse entre el cilindro principal y el pequeño cilindro a soldar encima de éste. En ese momento se ordena activar el giro de la mesa durante un tiempo de seis segundos (ya que se diseñó en su SC correspondiente que girara a una velocidad de 60º/seg.). Así, la plataforma gira 360º, simulando la soldadura de ambas piezas cilíndricas sin necesidad de realizar movimientos peligrosos por parte del brazo robótico y la pistola de soldadura que tiene como herramienta.

Tras terminar, el robot 1 se retira a su posición inicial y se ordena la apertura de las pinzas de agarre, orden que permite que el robot 2 se aproxime a coger el cilindro como se expuso antes y dejando de nuevo en suspensión el robot 1 hasta la llegada de la siguiente pieza a soldar.

En el subprograma SoldaduraCilindroVerde se produce un proceso similar al del cilindro azul, con la salvedad de que aquí la soldadura es doble, y por tanto la plataforma hace dos giros de 360º. Primero hace uno para simular la soldadura entre el cilindro principal y el pequeño que tiene colocado encima, como en el caso azul. Segundo, hace otro giro de 360º para soldar el pequeño cilindro con el trozo de cono que tiene colocado encima, realizando los movimientos de aproximación pertinentes.



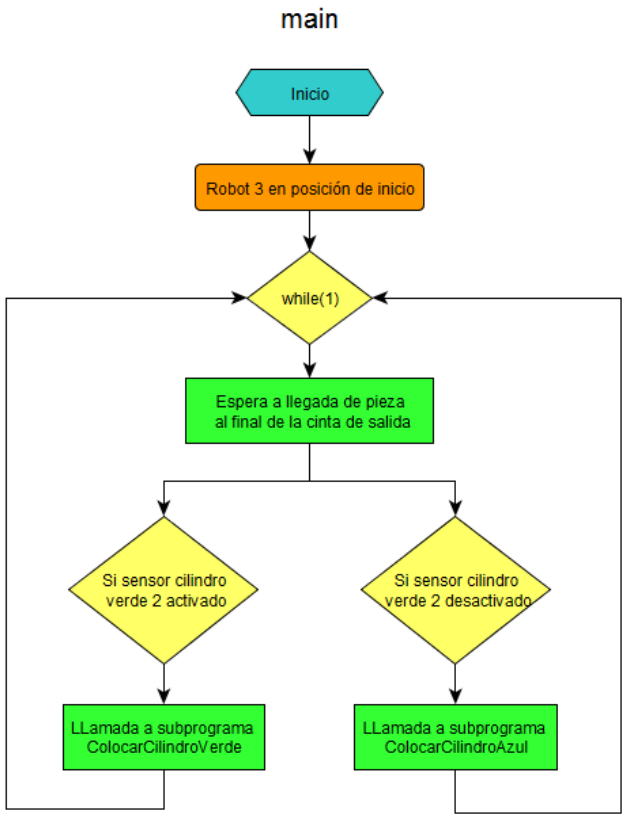


Por último se va a exponer el programa de RAPID del robot 3, perteneciente al controlador 2, cuya labor es la de clasificar a los cilindros ya soldados, que llegan de la cinta de salida, en dos cajas, una azul y otra verde.

En primer lugar, se crean dos variables numéricas ‘i’ y ‘j’, que servirán para contar el número de cilindros de cada color que hay ya colocados en cada caja. También se crea una constante numérica x que determina la distnacia entre centros de los cilindros una vez posicionados en las cajas, valor que será necesario utilizar más adelante.

En este módulo se tienen tres subprogramas PROC. El programa principal main y los subprogramas ColocarCilindroAzul y ColocarCilindroVerde.

Dentro del programa main se posiciona el robot 3 en su estado inicial Pos_ini_iman y se produce una llamada a los otros dos subprogramas de forma cíclica con la instrucción GOTO al igual que se realizaba en el programa del robot 1. De esta forma, si tras esperar a que un cilindro llegue al final de la cinta transportadora de salida, el sensor TuboVerde2 está a uno, se entrará en el subprograma ColocarCilindroVerde, y de estar a cero se entraría en ColocarCilindroAzul.

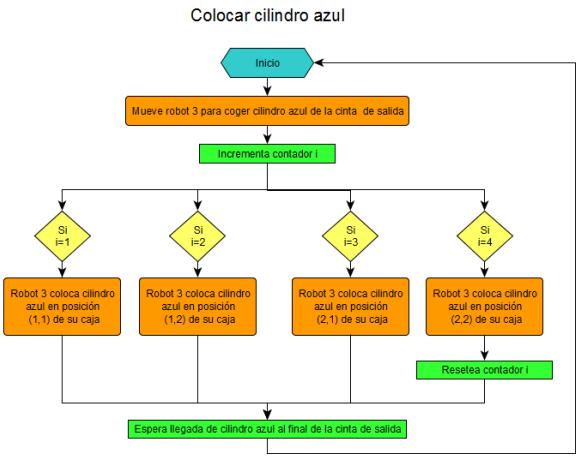


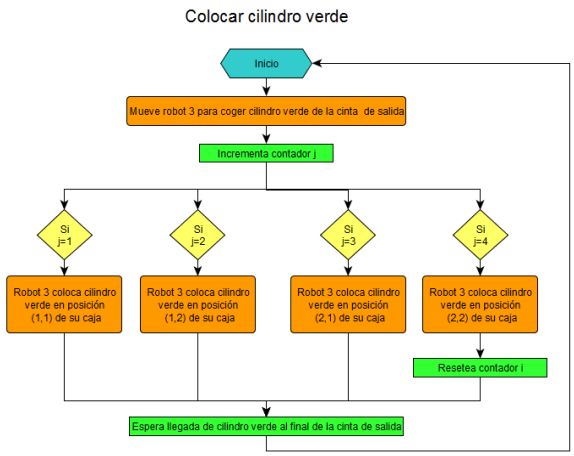
Dentro del subprograma ColocarCilindroAzul se realizan los movimientos de aproximación para la recogida del cilindro, se activa el sensor del imán una vez llegados a la pieza y se espera a que la pieza esté cogida. En este punto se incrementa en uno el número de cilindros a colocar en la caja (i), se realiza un movimiento de aproximación a la caja de color azul y se ubica el cilindro en la primer hueco de la caja, el cual corresponde con el más lejano para así evitar desde primera hora posibles choques entre cilindros cuando se coloquen el resto.

Las cajas, cuya capacidad es de cuatro cilindros, se han dispuesto de forma similar a una matriz 2x2 de manera que cada vez que se entre en el subprograma, se dirija a una posición u otra en función de cuatro bucles IF, los cuales permiten su entrada o no en relación al valor del contador de cilindros (i en en el caso de cilindros azules). Los movimientos relativos del robot son tales que la posición (1,1) de la caja azul corresponde a la posición (x, x), siendo x la constante antes definida como la distancia entre centros de cilindros colocados en la caja. La posición (2,2) de la caja es siempre la más cercana al robot (para evitar los choques antes mencionados) y corresponde con la posición relativa (0, 0).

Al finalizar de colocar la cuarta pieza se resetea el contador i, dándole un valor igual a cero.

El subprograma ColocarCilindroVerde es similar al anterior, con las diferencias de que ahora el contador es otro, denominado 'j' y que la posición (1, 1) de la caja es la (0, x) y la (2, 2) es la (x, 0). Esto se hace así debido a que es la mejor opción para evitar choques observando las posiciones relativas entre el robot y la caja verde.





2.3. Simulación de la estación de ejemplo

Con toda la programación de la estación terminada, queda simularla en el entorno de Robotstudio para observar el correcto funcionamiento del sistema automatizado creado. Para ello, dentro de la Pestaña RAPID hay que pulsar sobre Verificar programa, luego sobre Aplicar y decir que sí, y por último, si no se ha notificado de ningún error en la compilación, clicar sobre Inicio para comenzar la simulación. El proceso se ilustra en la Figura 2-50.

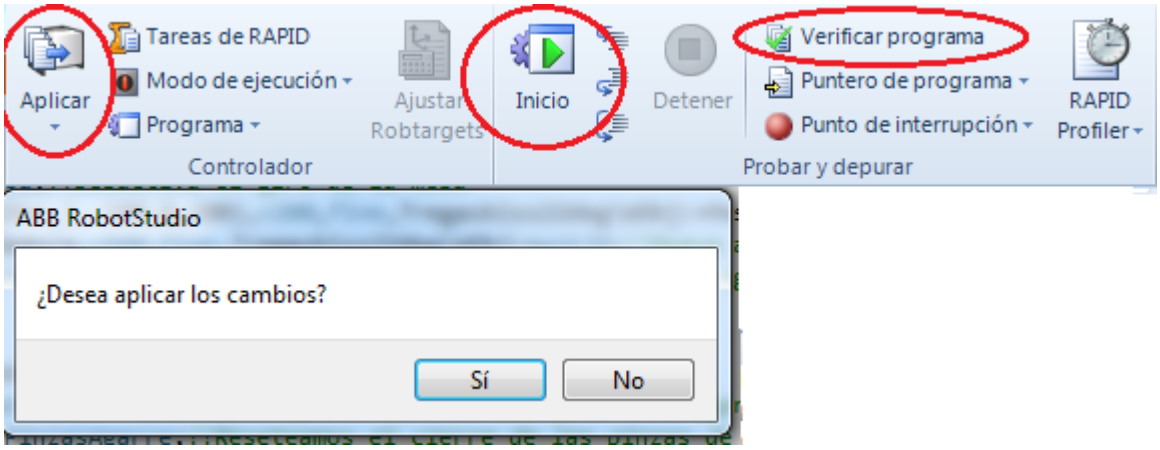


Figura 2-50. Compilación de un programa e inicio de la simulación.

En caso de quererse grabar una copia de un video de la simulación, dentro de la pestaña Simulación hay que dirigirse hacia Grabar Simulación. Para configurar las propiedades del video a grabar hay que clicar sobre el pequeño botón con el dibujo de una flecha justo debajo de 'Ver grabación'. Así, se pueden configurar parámetros como el instante en el que se quiere iniciar la grabación tras pulsar el inicio de la simulación, la ubicación donde se quiere guardar el video o el formato (wmv o mp4).

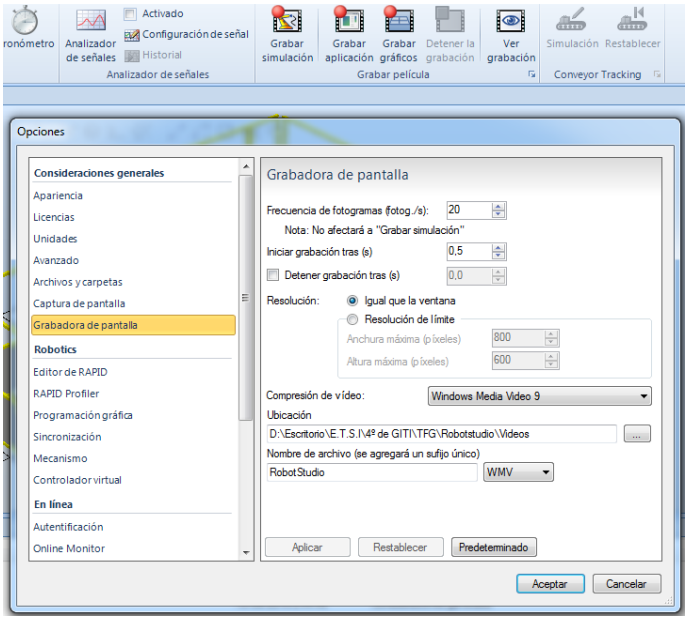


Figura 2-51. Grabación de la simulación y propiedades.

El video de la simulación de esta estación se encuentra adjunto en el CD anejo a este trabajo en su entrega. Para finalizar, se muestran varias capturas tomadas del vídeo con algunos detalles del proceso.

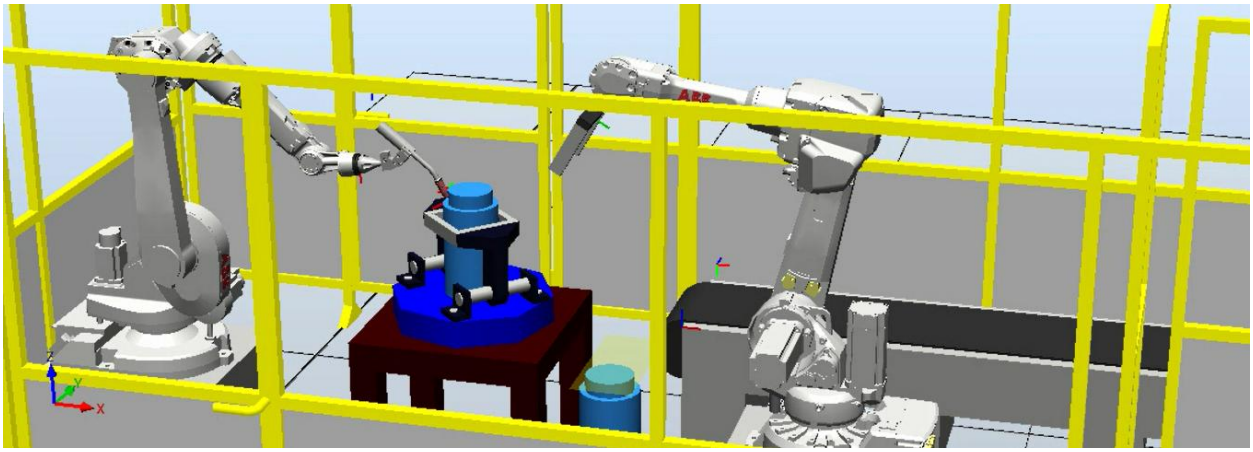


Figura 2-52. Soldadura de un cilindro azul.

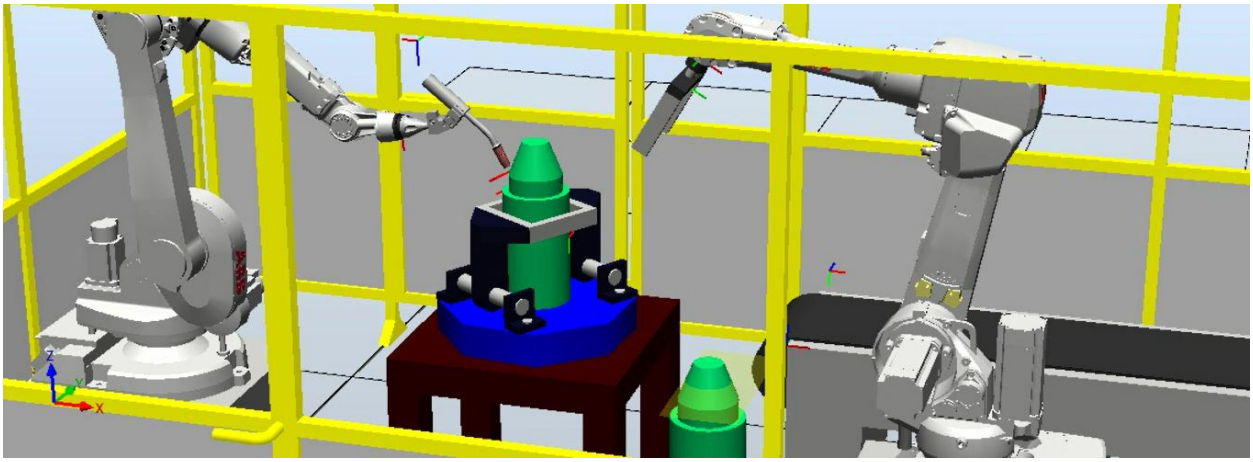


Figura 2-53. Soldadura de un cilindro verde.

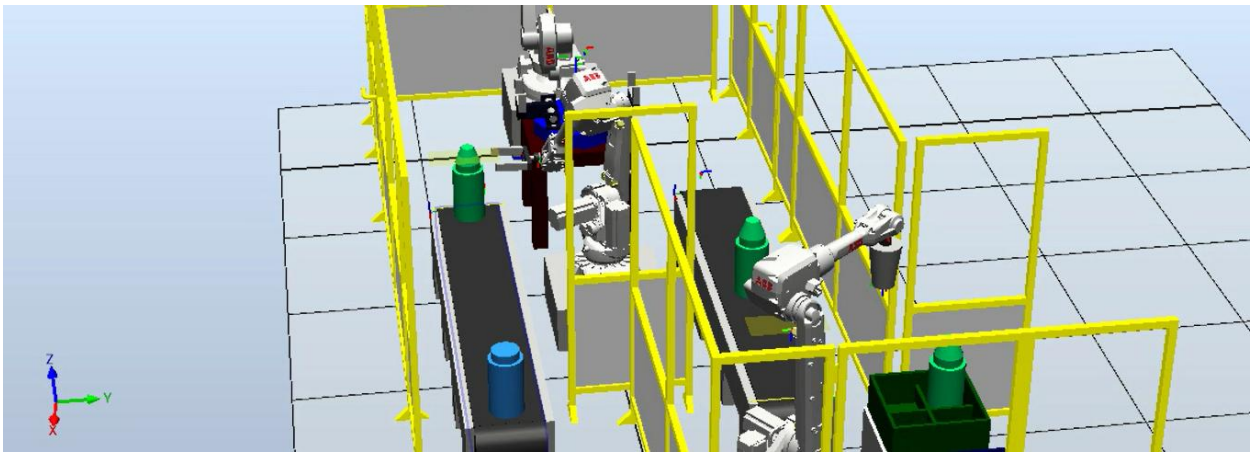


Figura 2-54. Llegada de los cilindros por la cinta transportadora de entrada.

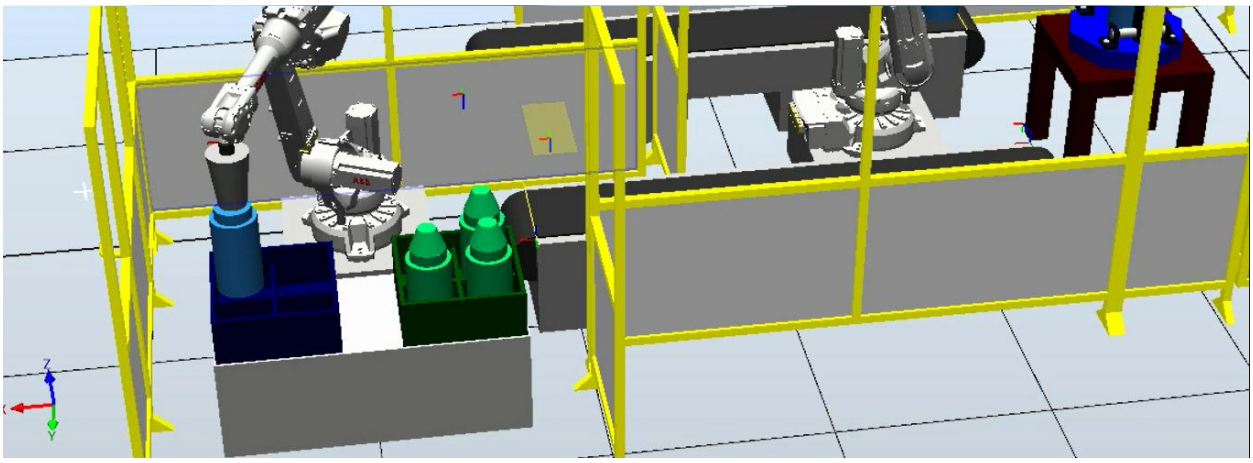


Figura 2-55. Colocación de un cilindro azul en una caja de su color.

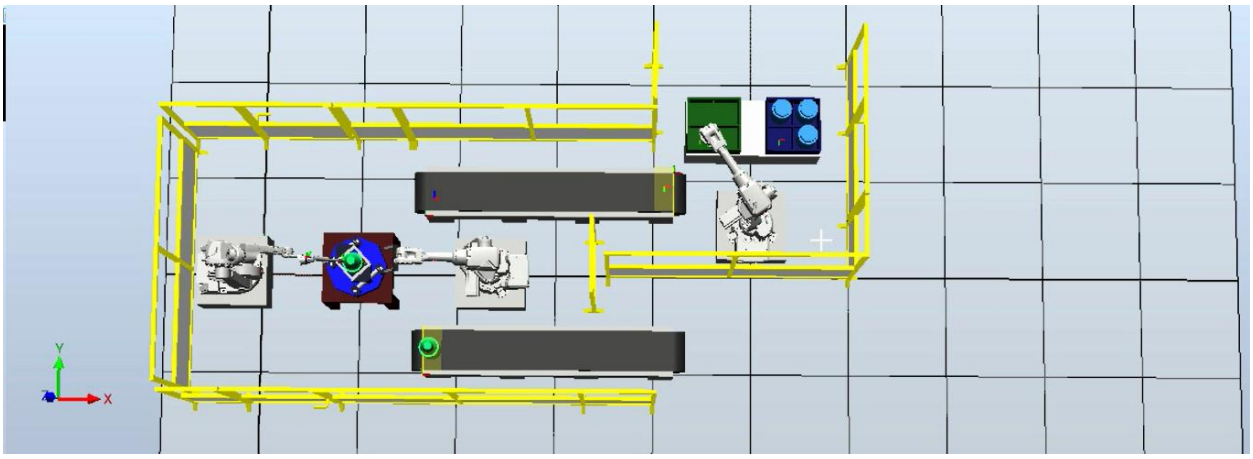


Figura 2-56. Planta de la estación completa en pleno funcionamiento.

3 IMPLEMENTACIÓN A UN PROCESO REAL DE SOLDADURA. ESTACIÓN ABB DE EUCOMSA

Today I will do what others won't,
So tomorrow I can accomplish what others can't.
- Jerry Rice -

En esta parte del trabajo el autor ha estado trabajando en el diseño y programación de una estación robotizada ABB en la Entidad Colaboradora denominada Europea de Construcciones Metálicas, S.A. (EUCOMSA) durante un periodo de tiempo que ha abarcado desde el 11 de Abril al 23 de Mayo de 2016, completando un total de 165 horas de trabajo. Más en concreto, la estación creada en Robotstudio por el autor tiene como objetivo la soldadura automatizada de piezas de acero (conjunto de chapas) utilizadas posteriormente para construir estructuras metálicas, tales como torres eléctricas o sopores de paneles solares.

De todas las estaciones ABB que dispone EUCOMSA, el autor ha diseñado y programado la correspondiente al denominado Robot IV, encargado de soldar varias chapas de acero de pequeño espesor (nudos) entre sí. Estas chapas a soldar forman parte de la estructura de un conjunto de placas solares cilindro-parabólicas destinadas al proyecto solar Ashalim de Abengoa, en Israel.

3.1. Mediciones de la estación real

La primera semana de trabajo el autor ha estado realizando tareas de medición sobre la estación del robot IV. Se han medido con detalle todos los elementos y piezas que conforman la estación para luego poder representarla en AutoCAD con la mayor fidelidad posible.

A grandes rasgos, la estación está compuesta por:

- Robot IRB 2400. Alcanzabilidad: 150 cm. Herramienta: Pistola de soldadura Dinse Dix Metz 594 (45º).
- Torch cleaner BRS-LC.
- Base robot.
- Base TCP (Torch cleaner).
- Canaletas.
- Mesa de trabajo.
- Gatos de soporte de las chapas.

- Nudos AA, BA, BB y BC.
- Soportes fijos de la mesa de trabajo.
- Soportes giratorios de la mesa de trabajo.
- Pantallas de protección.
- Vallas.

Es interesante recalcar que los conjuntos de chapas a soldar, también denominados nudos, son 4 tipos distintos. Éstos se distribuyen en la mesa de trabajo de manera que los cuatro se sueldan en un proceso en serie. Por otro lado, tras la finalización de la soldadura de los cuatro nudos, el robot siempre realiza un acercamiento al torch cleaner para una limpieza y refrigeración de la pistola de soldadura.

De todos los elementos de la estación, tan solo el TCP y el robot IRB son importación interna de la Biblioteca de Robotstudio. El resto de piezas han sido creación del autor mediante el uso de AutoCAD 3D 2014, a partir de las medidas pertinentes que se desarrollaron.

A continuación se muestran varias figuras con el diseño de los sólidos por separado en CAD. Los planos de todas las piezas se encuentran en el Anexo A.

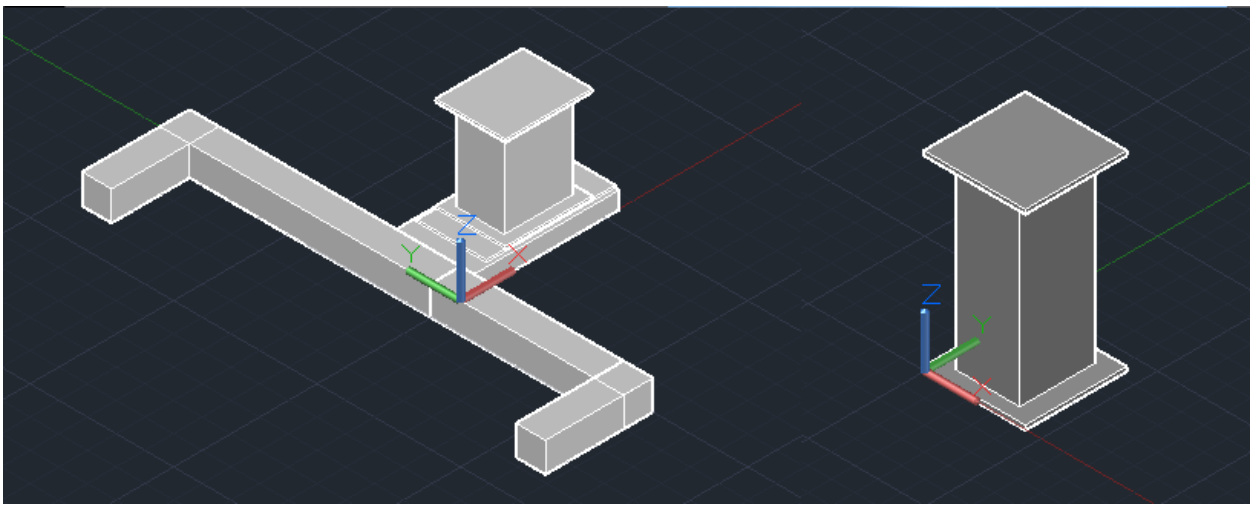


Figura 3-1. De izquierda a derecha. Base del robot y canaletas; base del TCP.

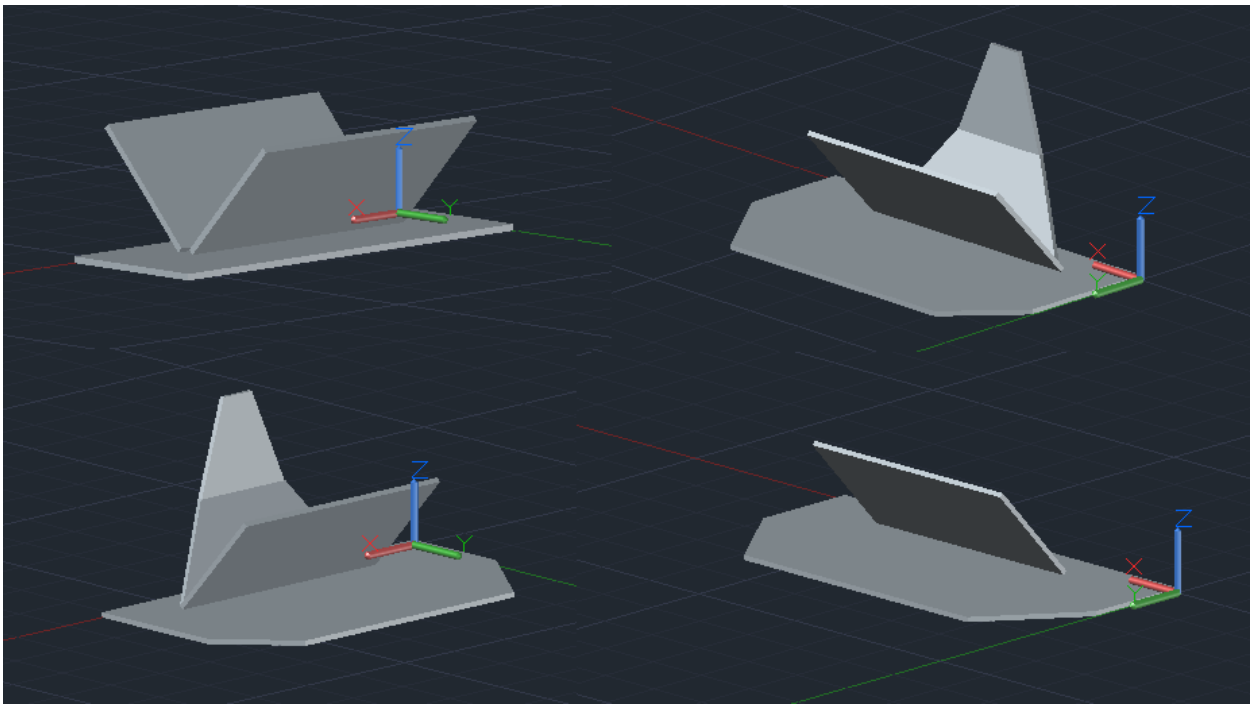


Figura 3-2. De arriba abajo, de izquierda a derecha. Nudo AA, nudo BA, nudo BB, nudo BC.

Debido a que el modelo de pistola de soldadura no se encuentra en la Biblioteca de Robotstudio, también ha sido tarea del autor diseñar dicha herramienta con AutoCAD.

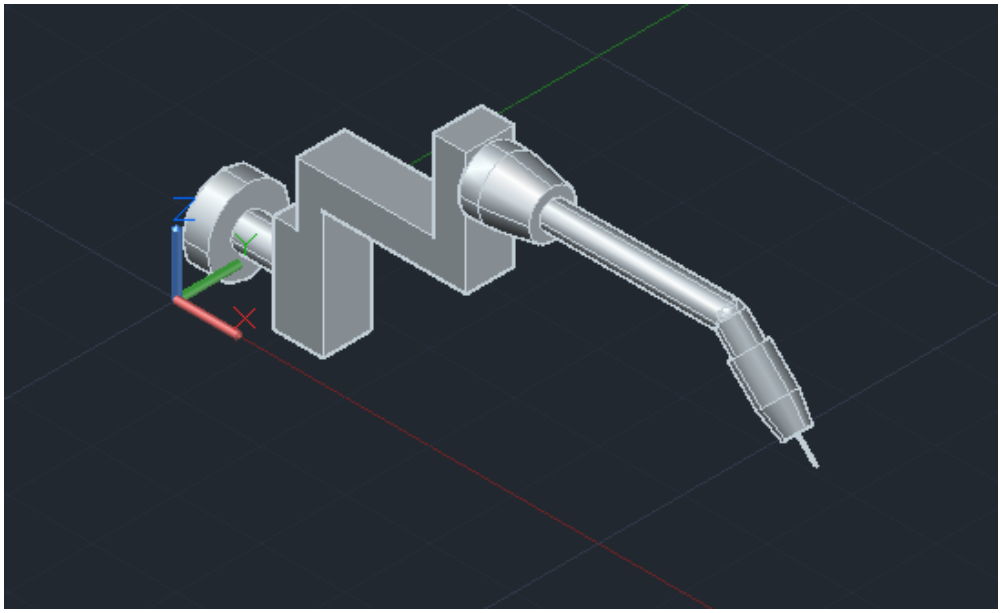


Figura 3-3. Pistola de soldadura Dinse Dix Metz 594 (45º).

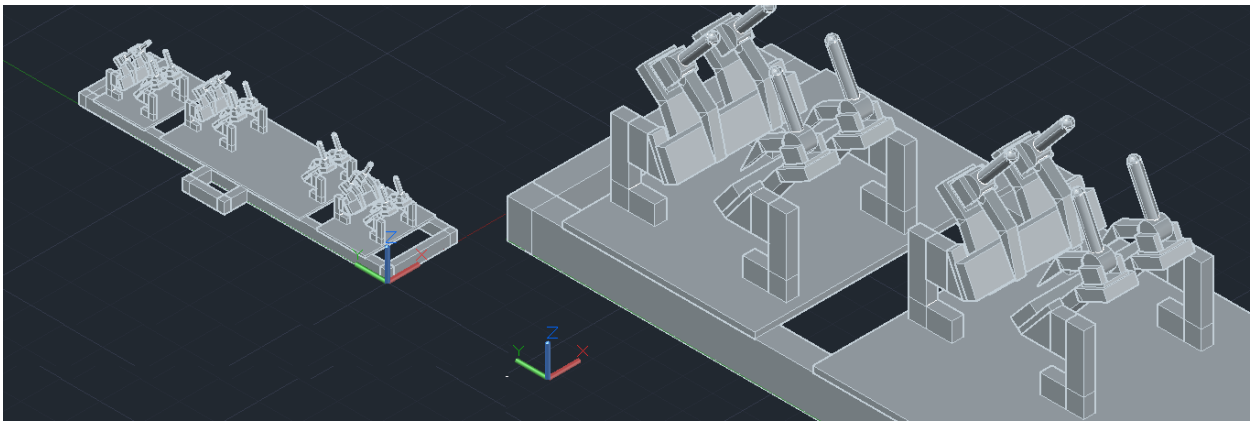


Figura 3-4. Mesa de trabajo y detalle de los gatos de agarre.

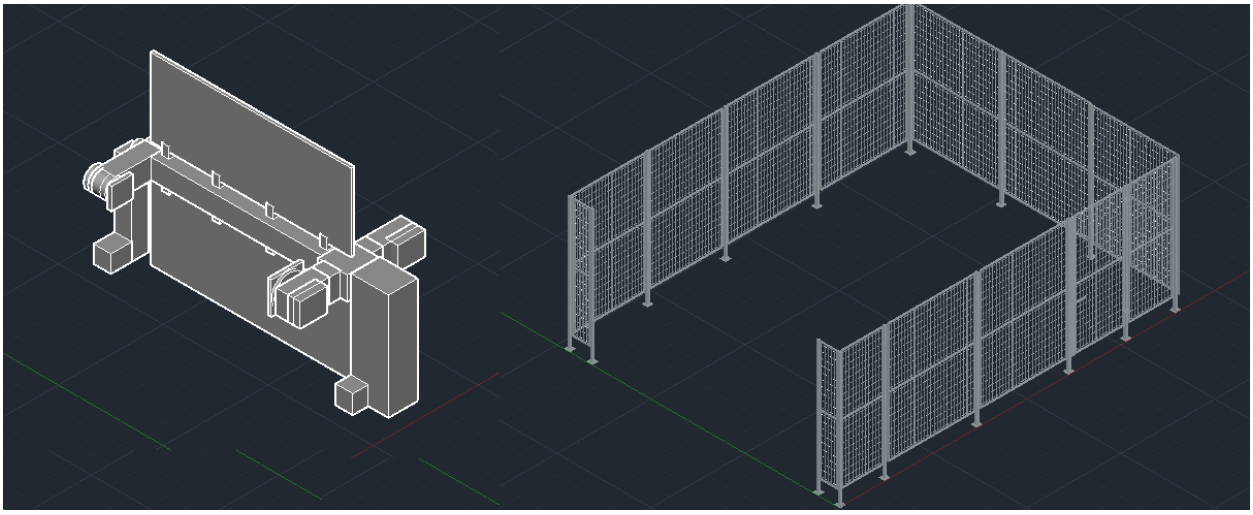


Figura 3-5. De izquierda a derecha. Soportes fijos y móviles, junto a las pantallas de protección; Vallas.

La estación completa en AutoCAD y en Robotstudio, tras dotar de color las piezas como en la estación real e importar el brazo robótico y el TCP, se observa en la figura 3-6.

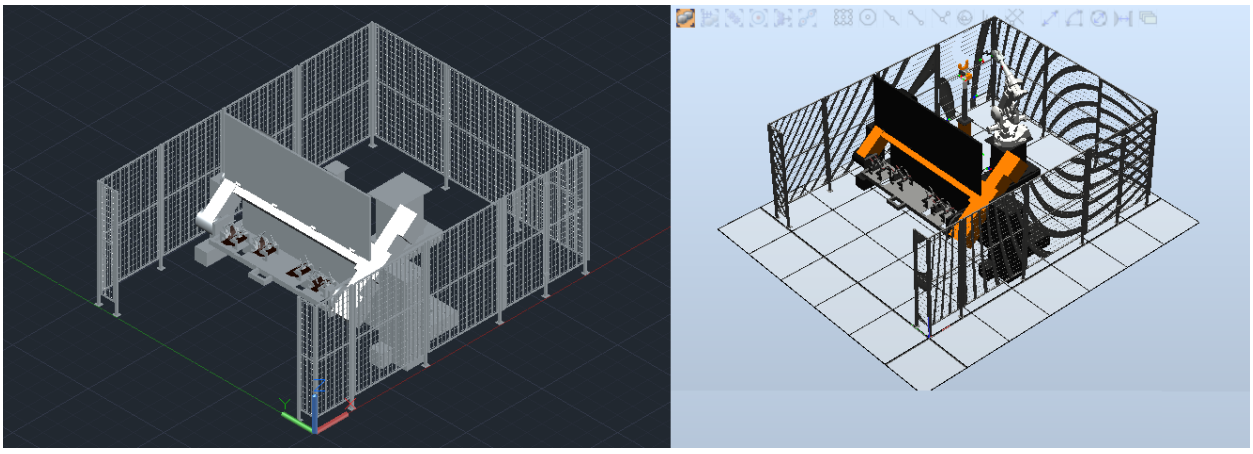


Figura 3-6. Estación completa en AutoCAD a la izquierda, en Robotstudio a la derecha.

Nota: debido a que existe una diferencia de escala entre AutoCAD y Robotstudio, para conseguir una similitud de medidas en ambos software es necesario dividir en AutoCAD entre 25.4 el tamaño de las piezas creadas. De esta forma, se consigue que en Robotstudio la estación tenga un tamaño como la real.

Por último se muestran unas fotos tomadas de la estación real, para poder observar la similitud entre la geometría de la estación diseñada y la existente en las instalaciones de EUCOMSA.

Figura 3-7. Estación de EUCOMSA, foto número 1.

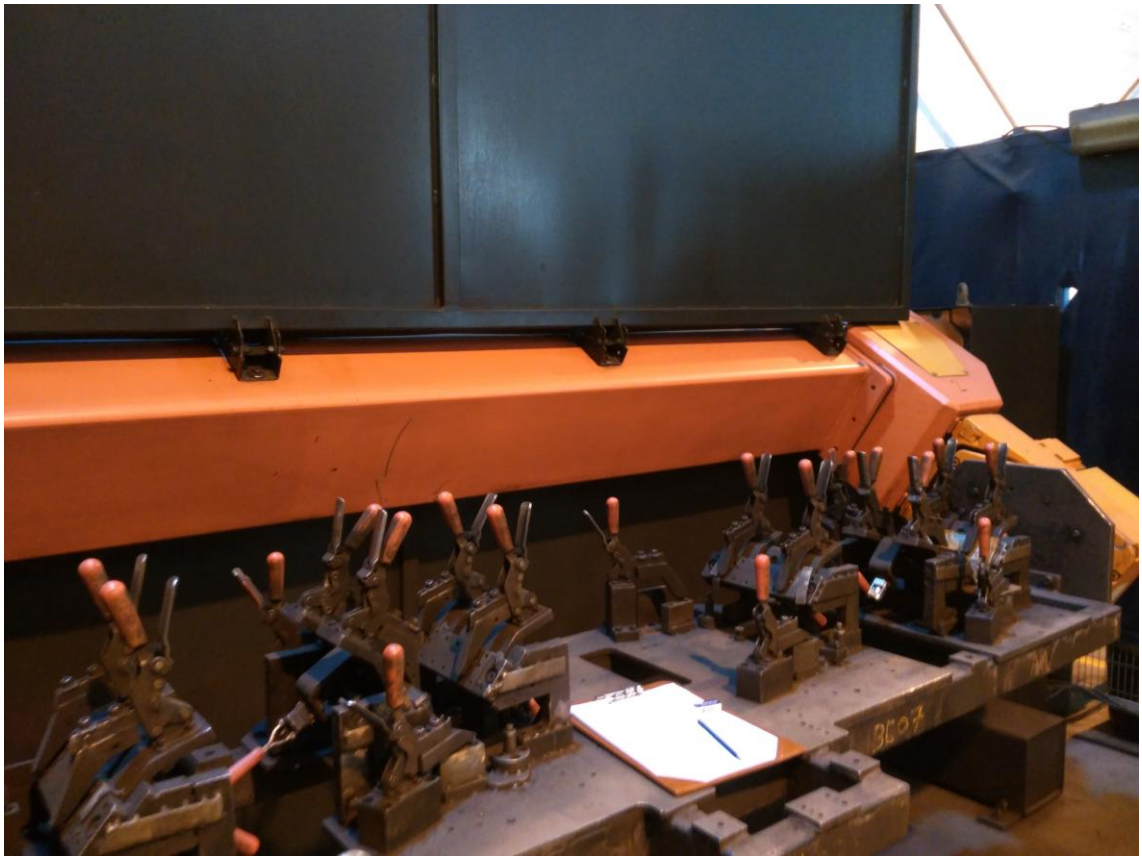


Figura 3-8. Estación de EUCOMSA, foto número 2.

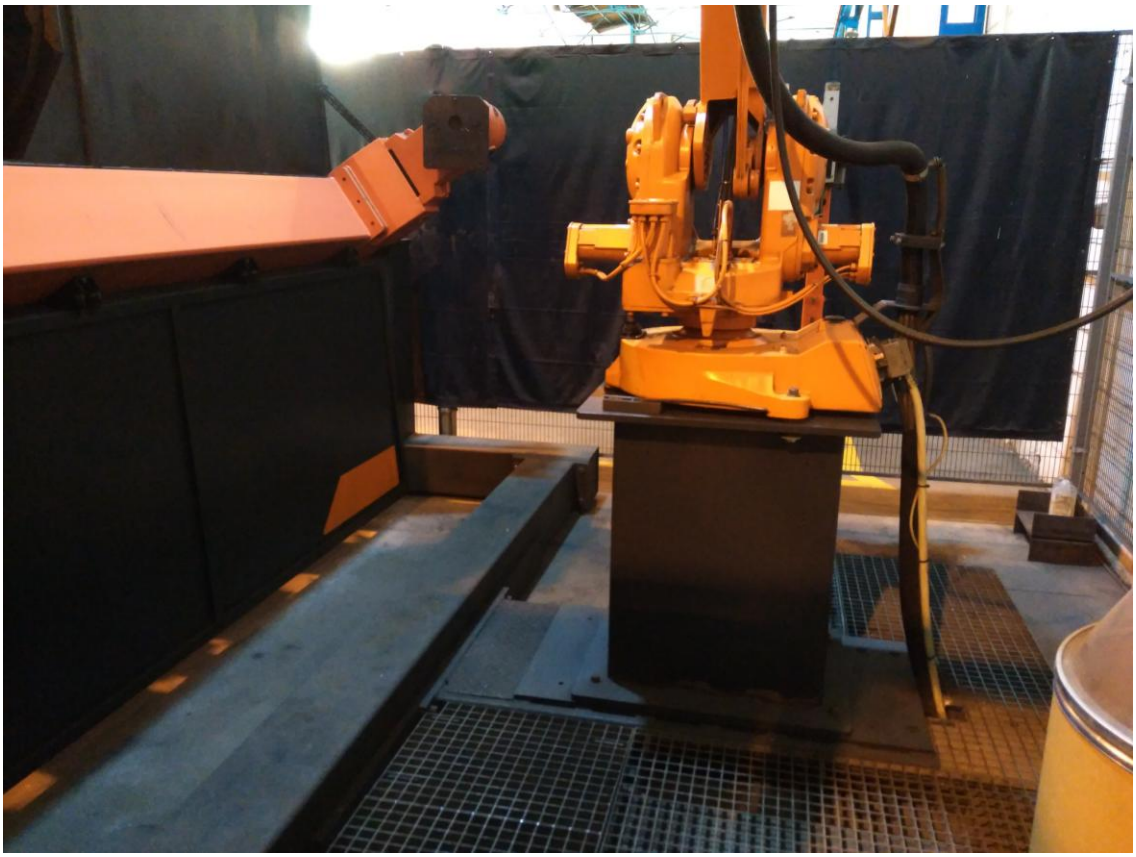


Figura 3-9. Estación de EUCOMSA, foto número 3.

3.2. Creación de la estación en Robotstudio. Simulación.

Tras diseñar la estación el siguiente paso es la creación de los SC. El automatismo de la estación diseñado en dichos SC sigue el siguiente proceso: los nudos se colocan en la mesa de trabajo uno a uno y conforme se ubican se van cerrando los gatos de agarre de los mismos. A continuación, la mesa realiza un giro de 180º hasta situarse en una zona en la que el brazo robótico alcance a los cuatro nudos. Posteriormente el robot pasa a soldar los nudos también uno a uno. Al terminar la soldadura la mesa gira 180º en dirección contraria, el robot realiza su limpieza y refrigeración en el Torch Cleaner y tras esto, los gatos se abren para que los nudos puedan ser reemplazados por otros cuatro aun sin soldar.

Para este proceso automático han sido necesarios seis Smart Components. El primero de ellos es el de creación de los nudos.

3.2.1. SC creación piezas

Con este SC se consigue mostrar los nudos en la mesa de trabajo en el orden en el que se ubican de izquierda a derecha. Este orden es: Nudo BA, nudo BB, nudo BC y nudo AA.

Al activarse la señal de entrada Start se esperan tres segundos y se muestra el primer nudo (BA). Para ello se utiliza el componente subordinado Show. Dos segundos después se muestra el nudo BB, dos después el nudo BC y un segundo más tarde el nudo AA. Esta relación de tiempos tiene un motivo que se explicará en el siguiente subapartado.

Cada vez que se muestra un nudo, se activa su señal digital de salida correspondiente ‘Nudo_XX_Mostrado’.

En la figura 3-10 se observa el diagrama de bloques de este SC.

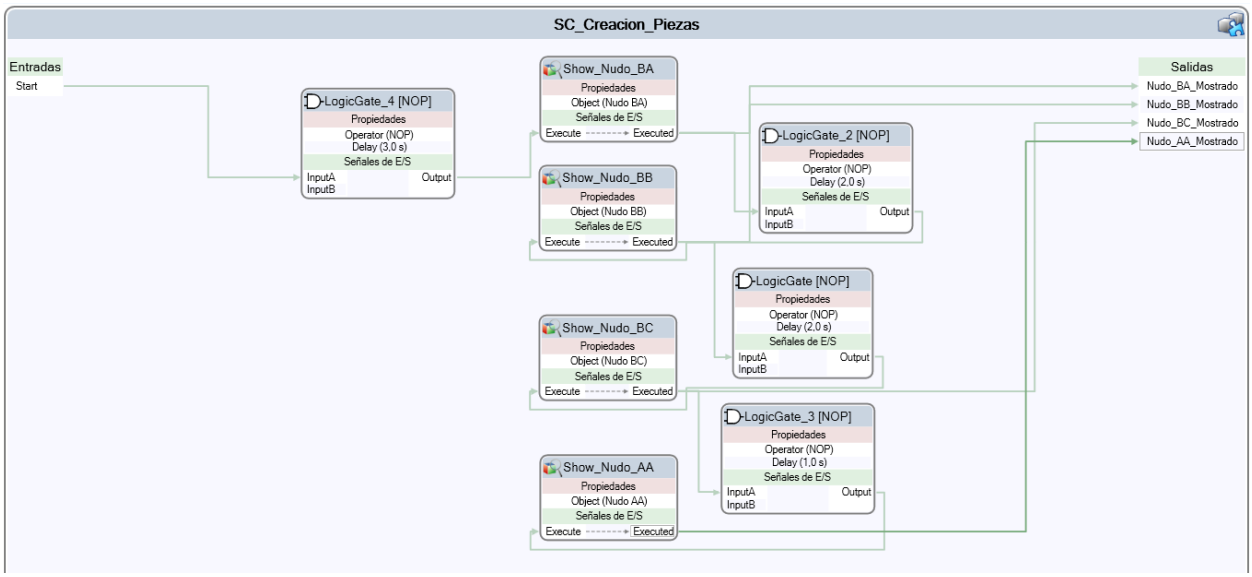


Figura 3-10. Esquema de bloques del SC de creación de las piezas.

3.2.2. SC Movimiento Gatos

En este SC se controla el movimiento giratorio de apertura y cierre de los gatos para sujetar los 4 nudos. Aunque en el proceso real es un operario el que coloca las piezas y cierra/abre los gatos, en esta estación simulada se ha efectuado el proceso de manera que sea todo automático desde su comienzo hasta su fin.

Con esto, se han creado 8 entradas al SC, una para dar la orden de cierre en cada pareja de gatos y otra para dar la orden de apertura también en cada pareja (Cierra_Gatos_X, Abre_Gatos_X). A su vez, se han diseñado 8 salidas para avisar cuando una pareja de gatos está cerrada o abierta (Gatos_X_Cerrados, Gatos_X_Abiertos).

Las órdenes de cierre de gatos se efectúan siguiendo el siguiente proceso: el giro de todos los gatos hasta su posición de cierre dura un segundo. De esta forma, cuando el primer nudo se muestra, se cierra instantáneamente los gatos situados más a su izquierda. Cuando estos gatos están completamente cerrados, se cierran los gatos situados más a su derecha. Por tanto, este proceso de cierre dura un tiempo de dos segundos. Debido a esto, el segundo nudo se muestra con un retardo de dos segundos respecto a la aparición del primero (como se explicaba en el subapartado 2.2.1.), ya que éstos se van mostrando conforme los gatos del anterior nudo están completamente cerrados. En el caso del tiempo de espera entre el tercer y cuarto nudo a mostrar es tan solo de un segundo, en vez de dos como en los anteriores, debido a que para sujetar al tercer nudo (nudo BC) solo hacen falta gatos a su derecha. Por tanto, el proceso de cierre de gatos para el nudo BC dura la mitad de tiempo que para el resto.

Para las órdenes de apertura de gatos el proceso es más simple, pues cuando llegue la señal a este SC de que el proceso de soldadura ha terminado, se abren primero todos los gatos ubicados a la izquierda de los nudos a la vez, para luego abrirse los situados a la derecha. Por tanto, el proceso de apertura completo de todos los gatos dura un tiempo de dos segundos.

Como se aprecia, en todo momento se abren siempre los gatos situados en el extremo izquierdo antes que en el derecho, esto se debe a que si se cerraran o abrieran todos a la vez se podría producir un choque entre gatos al entrar en el área de giro de los gatos de enfrente.

El esquema de bloques del SC se muestra en la figura 3-11.

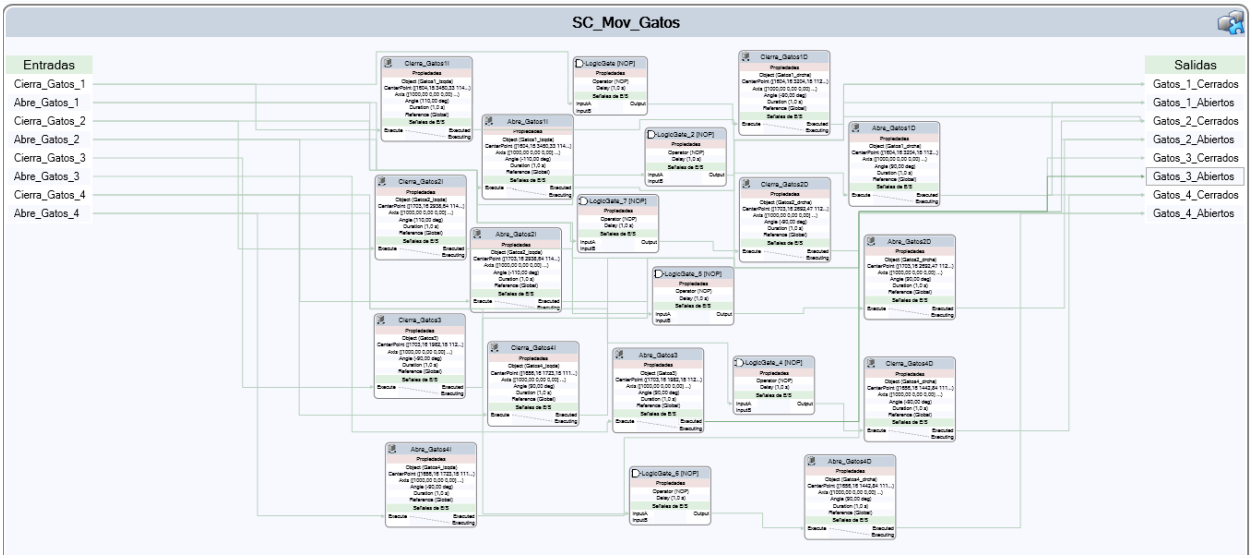


Figura 3-11. Esquema de bloques del SC del movimiento de los gatos.

3.2.3. SC Movimiento Carga->Soldadura

Una vez estén colocados los nudos y bien sujetos por los gatos, la mesa ha de girar hacia el lado en el que se encuentra el robot para que suelde las piezas. En el caso del robot real es el operario el que debe pulsar el botón de inicio en el panel de control para se produzca el giro. En el caso de la estación de Robotstudio, al no haber operario se ha programado este SC para que el giro se produzca de forma automática a los tres segundos después de cerrarse el último gato.

Para comprobar el tiempo de giro de la mesa, el autor cronometró un total de 10 medidas sobre la estación real. A partir de dichas medidas, se efectuó una media ponderada saliendo un tiempo de giro de aproximadamente 3 segundos.

El giro de los soportes giratorios es simple y se realiza mediante el componente de giro convencional en el que se indica los grados (180º) y el tiempo de giro (3 segundos).

En el caso de la mesa de trabajo, los gatos y los nudos, el giro es más complejo. Esto se debe a que estos sólidos deben girar pero manteniendo la oritentación constante durante el proceso, de forma que el plano de la mesa debe permanecer siempre paralelo al plano XY.

Para ello, el giro de estos elementos se ha realizado utilizando el subcomponente inteligente 'MoveAlongCurve', es decir, movimiento a lo largo de una curva. Por tanto, se ha creado una curva invisible en forma de círculo con el radio y centro apropiados, para que los sólidos antes mencionados puedan moverse a lo largo de dicha trayectoria. El movimiento a lo largo de una curva tiene unos parámetros de configuración distintos que el componente de giro. Así, para este bloque es necesario administrar la velocidad de giro, en vez del ángulo y el tiempo como ocurría con el subcomponente de giro convencional. Por ende, se ha calculado la velocidad de giro en mm/s, partiendo de que se debe girar 180º durante tres segundos.

$$\text{Radio del circulo} = r = 0.688 \text{ metros (valor medido)}$$

$$\text{Velocidad angular} = \omega = 60^{\circ}/\text{s} \ (180^{\circ} \text{ en 3 segundos})$$

$$\text{Velocidad lineal} = r\,\omega\,\frac{2\pi}{360} = 0.688 \times 60 \times \frac{2\pi}{360} = 0.7204719 \text{ m/s} = 720.4719 \text{ mm/s}$$

De esta forma, la mesa, las piezas y los gatos se moverán a lo largo de la trayectoria del círculo recorriendo 720.4719 milímetros cada segundo.

En este SC existen además dos señales de entrada y una de salida. La señal de entrada Marcha_Giro_Mesa da la orden de giro para que se produzca el movimiento hacia la posición de soldadura. Esta señal está comandada por el controlador del robot como se verá más adelante. Por otro lado, la señal de entrada Principio_Descarga se activa cuando el proceso de soldadura ha finalizado y resetea el temporizador de tres segundos para la siguiente soldadura. Por último, la señal de salida En_Posicion_Soldadura avisa cuando el giro de 180º ha terminado y la mesa está colocada en la posición de soldadura cercana al robot. Está señal será de gran utilidad, como se verá en los siguientes subapartados.

En la figura 3-12 se observa el esquema de bloques de este SC.

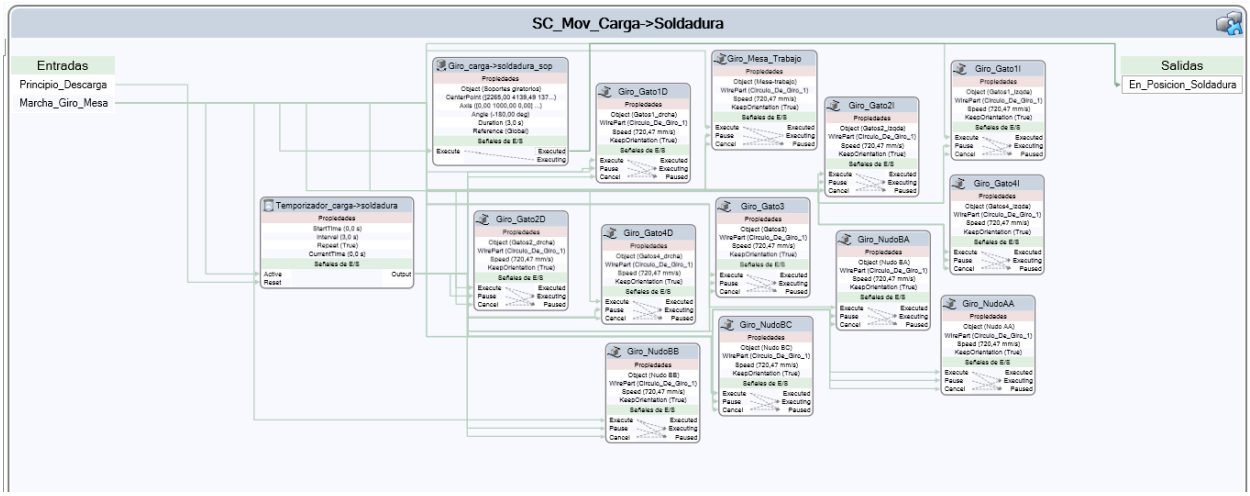


Figura 3-12. Esquema de bloques del SC del movimiento carga-soldadura de piezas.

3.2.4. SC Movimiento Soldadura->Descarga

El movimiento de Soldadura->Descarga es el proceso inverso del de Carga->Soldadura. Al igual que en el SC anterior, la duración del giro es de tres segundos. El giro se realiza en sentido opuesto, por lo que se ha tenido que diseñar otro círculo con mismo radio y centro que el anterior, pero girado 180º respecto al eje Z.

Existen dos entradas digitales: Principio_Soldadura y Fin_Soldadura, que marcan el inicio y el final del proceso de soldadura del robot. Fin_Soldadura activa el temporizador de tres segundos para que se produzca el giro cuando termine el robot de soldar. Principio_Soldadura resetea el temporizador para la siguiente soldadura.

Existe una señal de salida digital En_PosiciónDescarga, que se activa cuando ha finalizado el giro Soldadura->Descarga y la mesa vuelve a su posición inicial.

El resto de bloques son como los vistos en el subapartado 2.2.3., pero con un giro de sentido opuesto como ya se ha explicado. En la figura 3-13 se aprecia el diagrama de bloques de este SC.

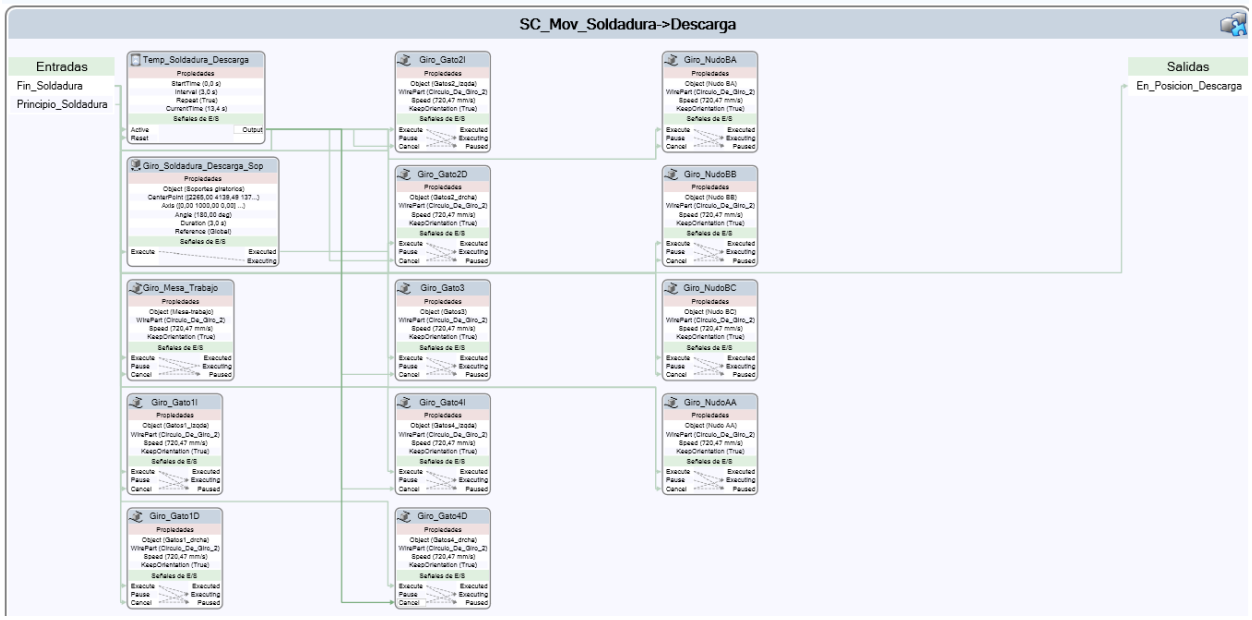


Figura 3-13. Esquema de bloques del SC del movimiento soldadura-descarga de piezas.

3.2.5. SC Corrección Posición Mesa Soldadura

Como se ha explicado, el giro de los soportes giratorios se realiza mediante la configuración de unos parámetros (ángulo y tiempo) y el giro de la mesa y las piezas mediante la velocidad. Debido a esta diferencia, existen ciertos centímetros de error en el giro, produciéndose uno de algo más de 180º para la mesa y las piezas. Dicho error se va acumulando si no se corrige cada vez que se produce un giro, por tanto es necesario reposicionar esos centímetros la mesa respecto a un objeto de referencia, que en este caso es el propio cuerpo de los soportes giratorios.

Se han creado por tanto dos SC, uno para reposicionar la mesa cuando gira hacia el lado de soldadura, y otro para cuando gira hacia el lado de descarga. En este caso, cuando gira hacia el lado de soldadura, cuando al SC le llega como señal de entrada En_Posicion_Soldadura actviada, desde el SC del movimiento carga->soldadura, se produce dicho reposicionamiento relativo a los soportes giratorios de la mesa, los gatos y las piezas, utilizando el componente Positioner.

En la figura 3-14 se observa el esquema de bloques de este SC.

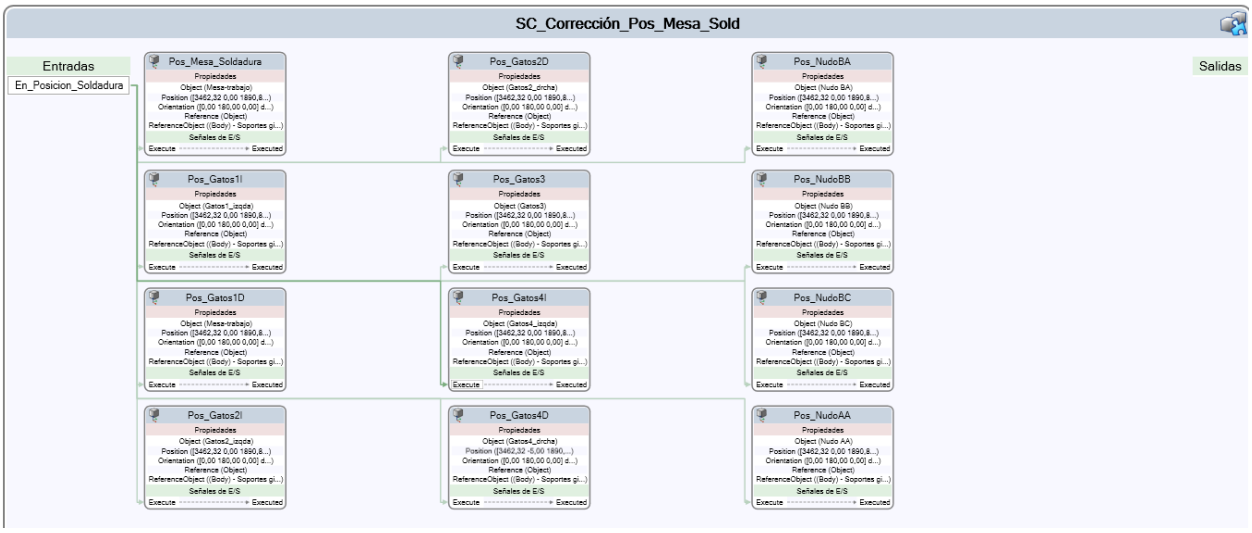


Figura 3-14. Esquema de bloques del SC de corrección de la mesa en posición de soldadura.

3.2.6. SC Corrección Posición Mesa Descarga

Al igual que ocurre al posicionar la mesa para la soldadura, pasa al posicionarla para la descarga. En este caso se corrige el error de posicionamiento se manera semejante a la corrección del subapartado 2.2.5, pero en este caso cuando se active la señal de entrada En_Posicion_Descarga, proveniente del SC del movimiento soldadura->descarga.

En la figura 3-15 se aprecia el esquema de bloques de este SC.

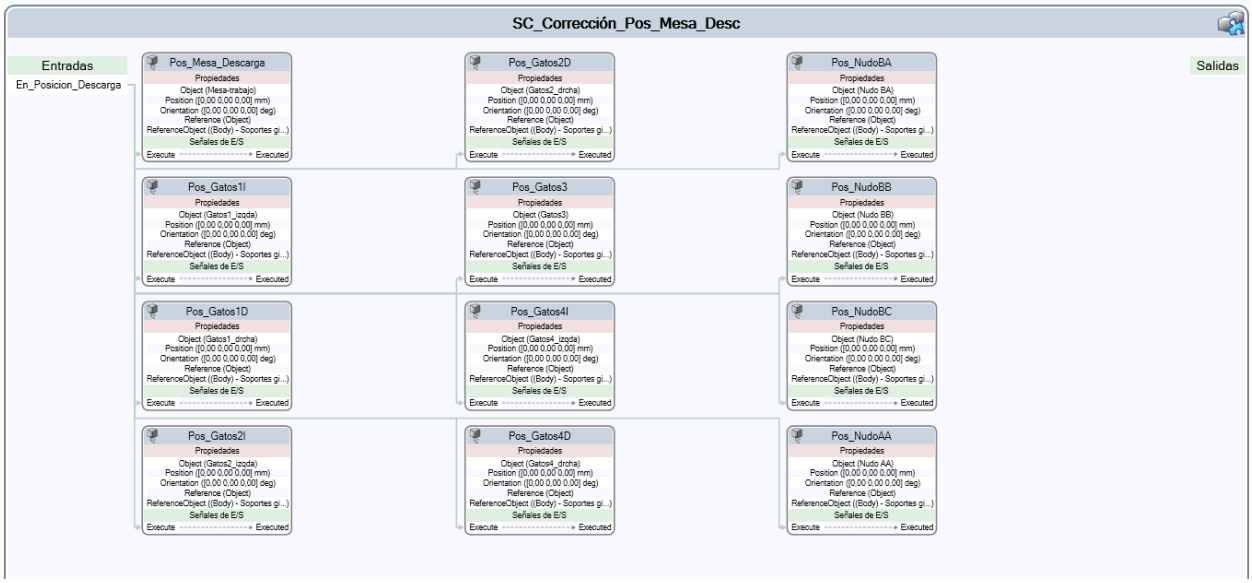


Figura 3-15. Esquema de bloques del SC de corrección de la mesa en la posición de descarga.

3.2.7. Controlador y lógica de estación

Tras haber explicado todos los SC que componen esta estación, se va exponer la creación del controlador y el diseño de la lógica de la estación.

Por tanto, se ha creado un controlador que gestiona los movimientos del robot y el inicio y el fin de los procesos de colocación de nudos y del giro de la mesa hacia una dirección u otra. Para ello, dentro del módulo de E/S de dicho controlador se han creado tres señales de entrada digitales y tres señales de salida digitales. Dicho conjunto de señales se expone en el Anexo B de este trabajo.

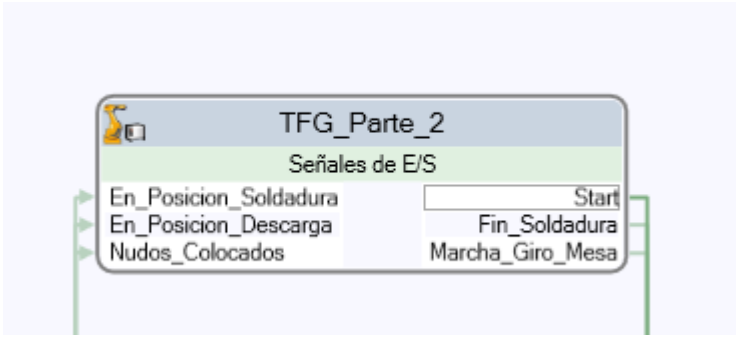


Figura 3-16. Controlador TFG_Parte_2.

Por otro lado, la lógica de estación que relaciona los seis SC entre sí y con el controlador se expone en la figura 3-17.

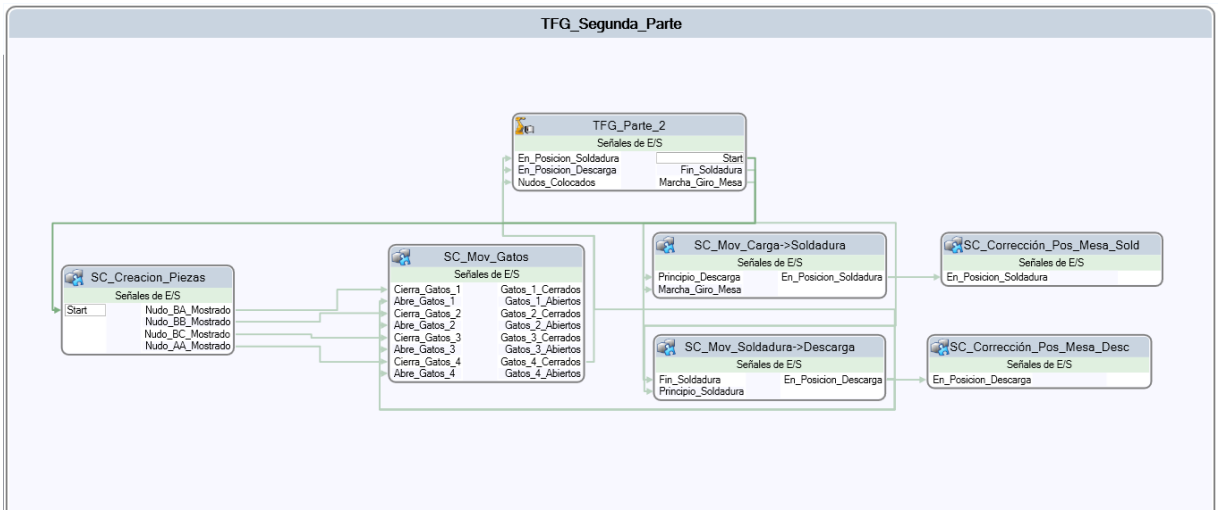


Figura 3-17. Lógica de estación del sistema completo.

Como se observa en la figura 3-17, el controlador TFG_Parte_2 da la orden 'Start' de creación de piezas. Conforme éstas se van creando, se va dando la orden de cierre de gatos al SC del movimiento de los gatos correspondientes a cada nudo. La orden para abrir los gatos proviene, como ya se dijo, del SC de movimiento soldadura->descarga, cuando la mesa vuelve a la posición de descarga. Dentro del conjunto de salidas del SC del movimiento de los gatos, en realidad solo una es de utilidad práctica. Esta es la señal de aviso de que los gatos de la última pieza (gatos número 4) están cerrados, indicando al controlador de que los nudos están correctamente colocados y agarrados y que se pude producir el giro de la mesa.

La orden de marcha para el giro de la mesa es también gestionada por el controlador y las órdenes de corrección de la mesa en cada posición provienen de sus SC de movimientos correspondientes.

Por otro lado, la salida En_Posicion_Soldadura también sirve para avisar al controlador de que la mesa está cercana al robot y que por tanto éste comience a realizar los movimientos de soldadura pertinentes. Cuando termine la soldadura el controlador avisa al SC de movimiento soldadura->descarga mediante la señal Fin_Soldadura, produciéndose el giro hacia la zona de carga/descarga de piezas.

3.2.8. Programación en RAPID

Lo último que queda por explicar para el diseño de esta estación es la programación de los movimientos del robot, así como el control desde RAPID de las señales del módulo E/S del controlador.

Por tanto, se ha creado un programa con un único subprograma principal PROC denominado main donde se efectúa todo el control.

En primer lugar, desde el main se resetea el comando Fin_Soldadura por si estaba activo del proceso de soldadura anterior y se activa la señal Start para que vayan mostrándose los nudos sobre la mesa. A continuación se mueve el robot a su posición inicial y se realiza la espera de que los nudos estén correctamente colocados y agarrados (Nudos_Colocados=1). Dicha señal proviene del cierre de los últimos gatos como se expuso en el subapartado anterior.

Una vez que la señal de Nudos_Colocados está a uno, se espera tres segundos y se pone en marcha el giro de la mesa. Luego, se espera a que En_Posicion_Soldadura esté a uno (giro de 180º completo) para resetear las salidas del controlador Start y Marcha_Giro_Mesa.

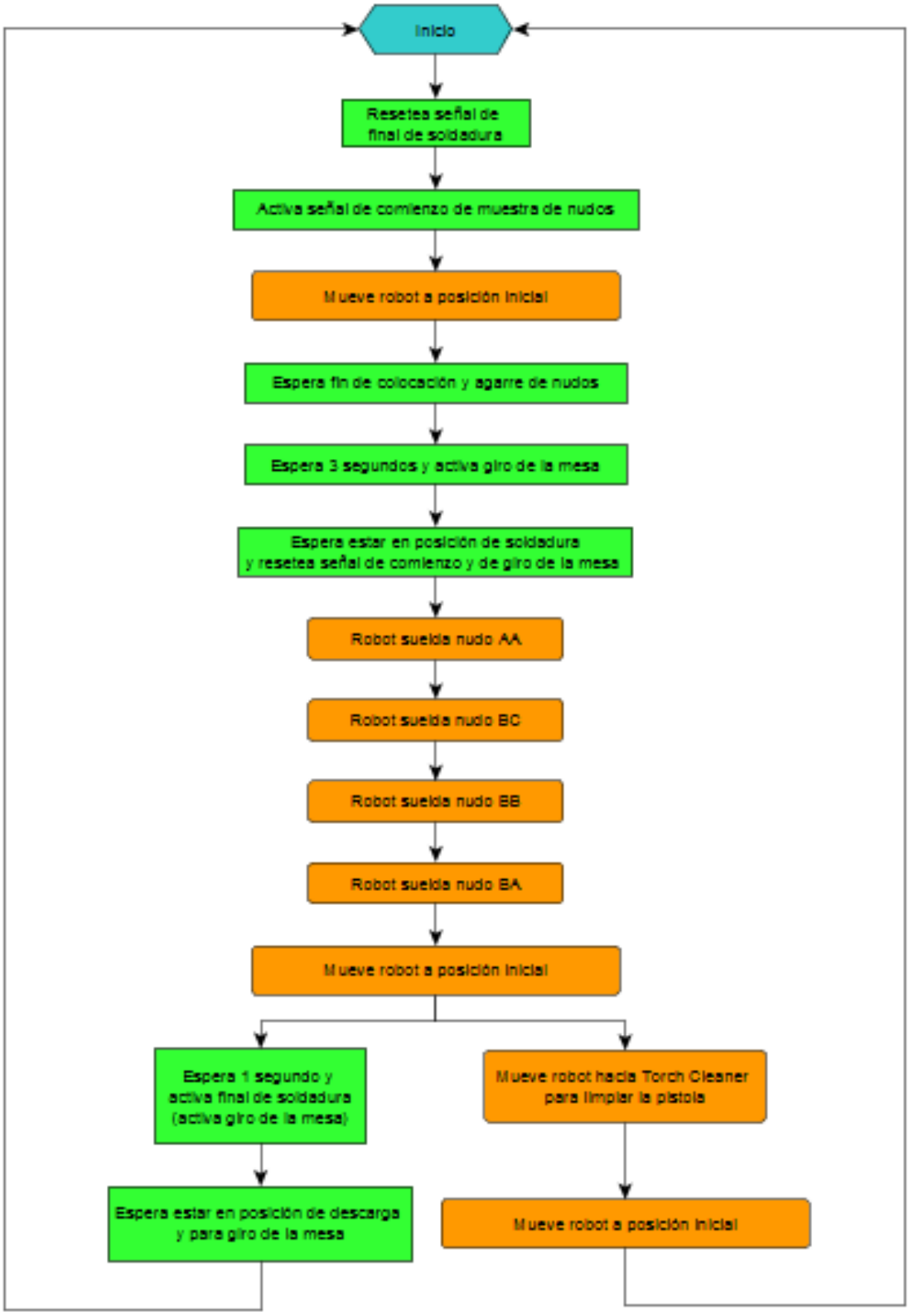
A continuación, con la mesa ya en la zona del robot, comienza el proceso de soldadura. El robot suelda las piezas en el orden inverso en el que se crearon, esto es: primero nudo AA, segundo nudo BC, tercero nudo BB y cuarto nudo BA. Este proceso de soldadura tiene varios repasos por nudo, de forma que se realizan varias pasadas para que la chapa horizontal quede bien soldada a las chapas inclinadas. El proceso es el siguiente:

- Nudo AA: tres pasadas. Primera pasada lineal entre una chapa inclinada y la chapa horizontal. Segunda pasada lineal entre la otra chapa inclinada y la horizontal. Tercera pasada en forma de zig-zag en el hueco entre las dos chapas inclinadas. Las tres pasadas se efectúan en el sentido negativo del eje X.
- Nudo BC: dos pasadas (al tener una sola chapa inclinada). Primera pasada lineal entre la chapa horizontal y la inclinada (ángulo de 45º entre la pistola y el plano XY). Segunda pasada lineal entre la chapa horizontal y la inclinada, pero cuatro centimétros por debajo (-4 en el eje Z) respecto a la primera pasada (ángulo de 45º entre la pistola y el plano XY).
- Nudo BB: tres pasadas. Primera pasada lineal entre una chapa inclinada y la chapa horizontal. Segunda pasada lineal entre la otra chapa inclinada y la horizontal. Tercera pasada en forma de zig-zag en el hueco entre las dos chapas inclinadas. Las dos primeras pasadas se efectúan en el sentido negativo del eje X, pero la tercera pasada se realiza en el sentido positivo del mismo eje.
- Nudo BA: tres pasadas, igual que el nudo BB. Primera pasada lineal entre una chapa inclinada y la chapa horizontal. Segunda pasada lineal entre la otra chapa inclinada y la horizontal. Tercera pasada en forma de zig-zag en el hueco entre las dos chapas inclinadas. Las dos primeras pasadas se efectúan en el sentido negativo del eje X, pero la tercera pasada se realiza en el sentido positivo del mismo eje.

Tras la soldadura del nudo BA, el robot vuelve a su posición inicial, se espera un segundo, se activa el comando Fin_Soldadura para que la mesa gire y se espera a que En_Posicion_Descarga se ponga a uno para finalizar dicho giro (mesa en posición de descarga).

A continuación se produce la limpieza de la pistola acercándose el robot al Torch Cleaner. Éste realiza tres aproximaciones, la primera para la refrigeración, la segunda para la regeneración automática del hilo de soldadura y la tercera para su limpieza. Se ha estimado un tiempo de 5 segundos en cada una de estas tres posiciones.

En la siguiente página se muestra el pseudocódigo con la explicación del programa de manera más resumida.


Inicio
Resetea señal de final de soldadura
Activa señal de comienzo de muestra de nudos
Mueve robot a posición inicial
Espera fin de colocación y agarre de nudos
Espera 3 segundos y activa giro de la mesa
Espera estar en posición de soldadura y resetea señal de comienzo y de giro de la mesa
Robot suelda nudo AA
Robot suelda nudo BC
Robot suelda nudo BB
Robot suelda nudo BA
Mueve robot a posición inicial
Espera 1 segundo y activa final de soldadura (activa giro de la mesa)
Mueve robot hacia Torch Cleaner para limpiar la pistola
Espera estar en posición de descarga y para giro de la mesa
Mueve robot a posición inicial

Como en este caso se está simulando una estación de soldadura real, se han implementado en Robotstudio las instrucciones de soldadura necesarias para que el robot suelde. En concreto, se han desarrollado las mismas instrucciones, configuradas con los mismos parámetros, que los comandos de soldadura del robot IV de la estación real de EUCOMSA.

Según la ayuda de Robotstudio, como documentación adicional aparece una aplicación sobre instrucciones Arc y Arc sensor que será de gran relevancia para este programa. Más en concreto, las tres instrucciones principales utilizadas para dicho programa han sido ArcLStart para comenzar un recorrido con soldadura, ArcL para continuar con otra trayectoria partiendo de la anterior y ArcLEnd para dejar de soltar hilo de soldadura y terminar así con dicha pasada.

En cuanto a la configuración de parámetros de soldadura, se ha desarrollado una exactamente igual a la del robot real. De esta forma, se han modelado tres parámetros denominados weld, seam y weave respectivamente. Para ello, primero hay que crear y configurar las variables tipo PERS necesarias (PERS seamdata, PERS welddata y PERS weavedata), para luego asociarlas a las instrucciones ArcL que les correspondan.

Para la variable seamdata se ha creado una denominada seam1 con todos sus parámetros a cero por defecto, pues esta variable no nos es de relevancia.

Para la variable welddata se han creado seis distintas: weld2, weld3, weld4, weld10, weld11 y weld26, cada una asociada a una pasada distinta sobre los nudos como se expuso antes. La configuración de welddata se efectúa sabiendo que los parámetros principales son cuatro, pero los dos últimos se dividen en nueve subparámetros respectivamente. A continuación se muestra la utilidad de cada uno de estos, tomando como ejemplo la configuración de weld10:

1. El primer parámetro hace referencia a la velocidad de soldadura. En el caso de weld10 vale 7.5 mm/s.
2. El segundo parámetro no influtye y se deja a cero por defecto.
3. El tercer parámetro hace referencia a los datos propios de la soldadura (arcdata) y se divide a su vez en nueve distintos:
    - El primero de ellos se refiere al número del programa de soldadura, en este caso el número 10.
    - El segundo es el retardo de fusión del hilo, 0.12 segundos para este ejemplo.
    - El tercero es la compensación de tensión, 22.75 V en este caso.
    - El cuarto es la velocidad de alimentación del hilo, es decir, la velocidad lineal con la que el hilo de soldadura sale por la pistola, en este caso 150 mm/s.
    - El quinto hace referencia a las propiedades dinámicas, 0.8 para este caso (80%).
    - El sexto es la intensidad de detección de contacto, para el ejemplo 8 amperios.
    - Los subparámetros 7, 8 y 9 no influyen y se dejan por defecto a cero.
4. El cuarto parámetro se deja con todos sus subparámetros a cero por defecto.

Para la variable weavedata se ha creado tan solo una: weave1. Esta variable hace referencia al movimiento de zig-zag de soldadura que se efectúa en algunos casos y sus parámetros configuran dicho movimiento. En este caso, las variables tipo weavedata tienen 15 parámetros que se presentan a continuación:

1) El primero hace referencia a la existencia o no de zig-zag, y en caso de existir, se refiere a los distintos tipos de zig-zag posibles. Para weave1 vale 1, que es un zig-zag tipo horizontal, como el observado en la figura 3-18.

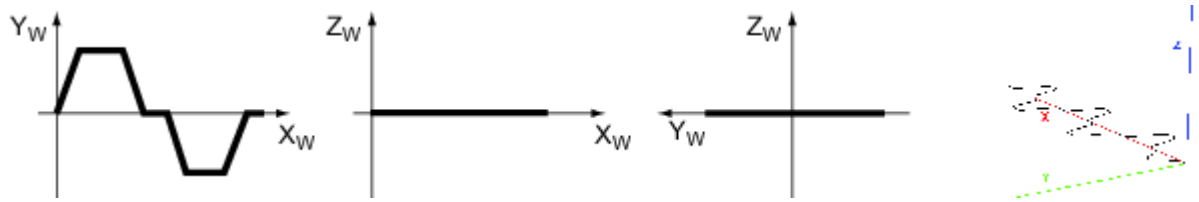


Figura 3-18. Onda de soldadura zig-zag tipo 1.

2) El segundo es el tipo de onda a usar. Se utiliza un valor igual a cero para decir que es una onda que puede ir en la dirección de cualquier eje, como en el caso de weave1.
3) El tercero mide la longitud de la onda. En este caso vale 7.98 mm. que es la distancia de separación entre las dos chapas inclinadas.
4) El cuarto es la anchura de la onda, que para weave1 vale 5 mm.
5) El quinto es la altura de la onda, que como es horizontal en este caso vale 0.
6) El sexto hace referencia al valor DL de la figura 3-19. Para weave1 vale 1.5 mm.

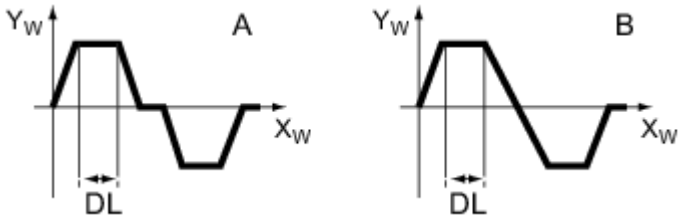


Figura 3-19. Distancia DL en la onda de soldadura de zig-zag.

7) El séptimo es el valor DC de la figura 3-20, para weave1 vale 0.5 mm.

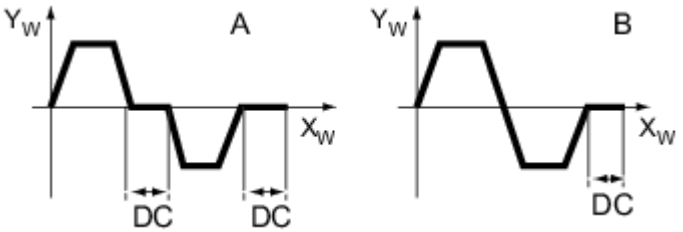


Figura 3-20. Distancia DC en la onda de soldadura de zig-zag.

8) El octavo es el valor DR de la figura 3-21, para weave1 vale 1.5 mm.

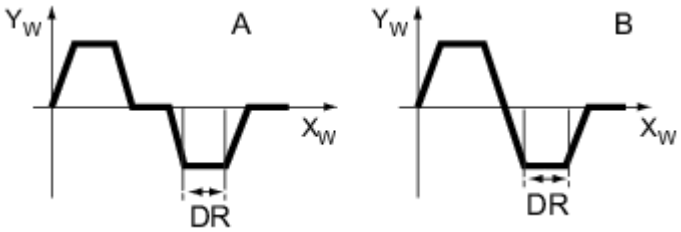


Figura 3-21. Distancia DR en la onda de soldadura de zig-zag.

9) El noveno hace referencia al ángulo respecto al eje Yw con el que la onda de soldadura incide inicialmente. Para weave1 vale 60º.

10) El resto de parámetros del weavedata se ponen por defecto a cero.

Gracias a la configuración completa de parámetros de manera semejante a la configuración real, se ha conseguido que el proceso de soldadura de la simulación de Robotstudio dure un tiempo similar

El código completo, con todas las instrucciones de movimiento, soldadura y las órdenes de activación y reset de las señales, se encuentra en el Anexo C de este trabajo.

3.2.9. Simulación

Tras programar la estación queda simularla para observar el comportamiento automático de la misma, tal y como se explicó para la estación de ejemplo anterior. La grabación de vídeo de dicha simulación se encuentra en el CD adjunto a este trabajo. A continuación se muestran varias capturas sobre dicho vídeo de detalles interesantes de la estación.

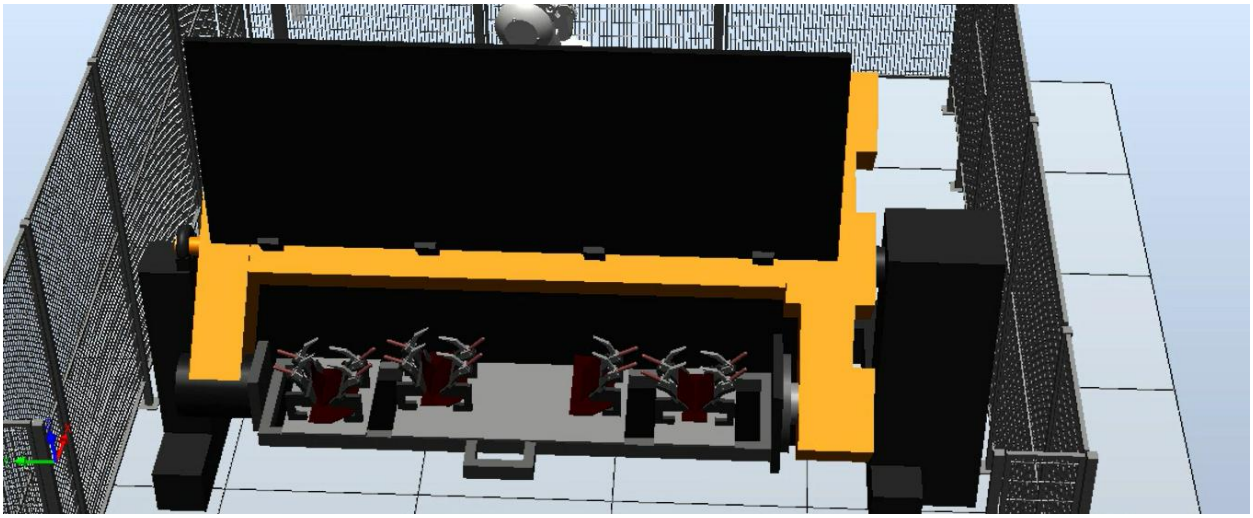


Figura 3-22. Mesa de trabajo en posición de descarga.

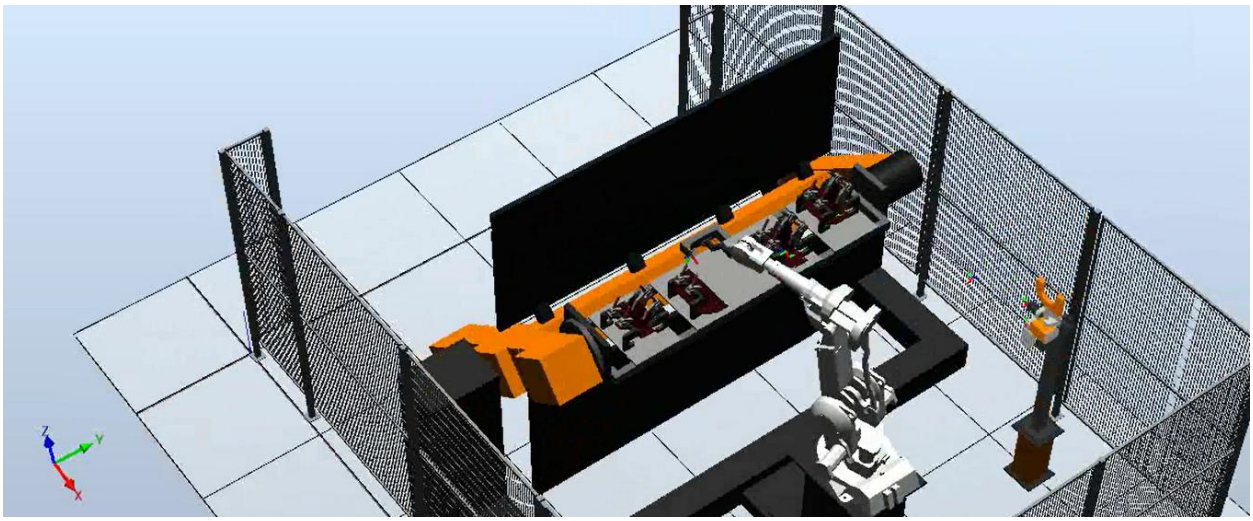


Figura 3-23. Mesa de trabajo en posición de soldadura.

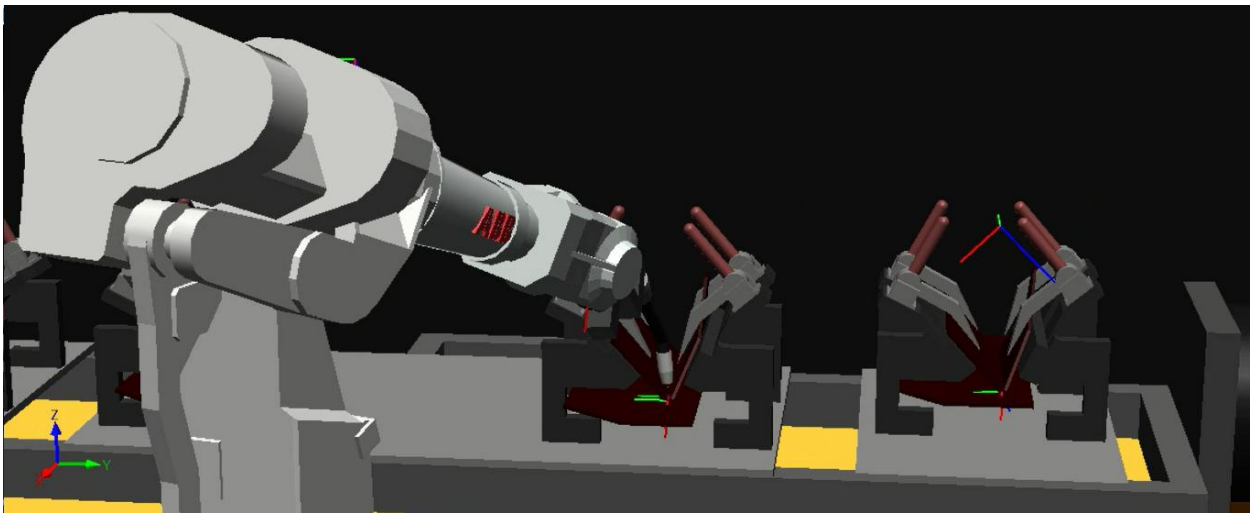


Figura 3-24. Soldadura del nudo BB.

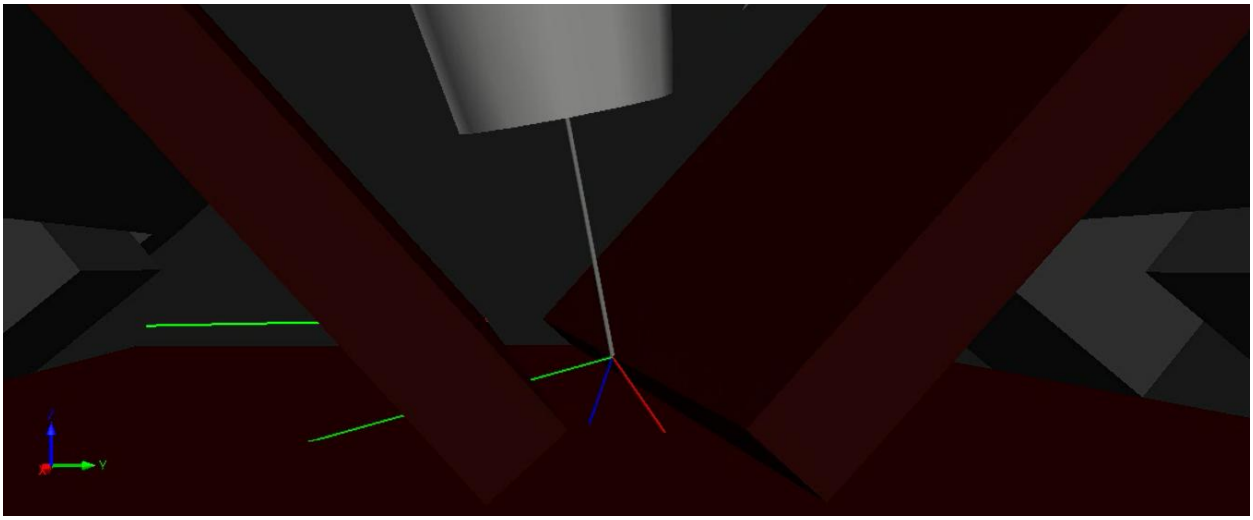


Figura 3-25. Aproximación a la segunda pasada de la soldadura del nudo AA.

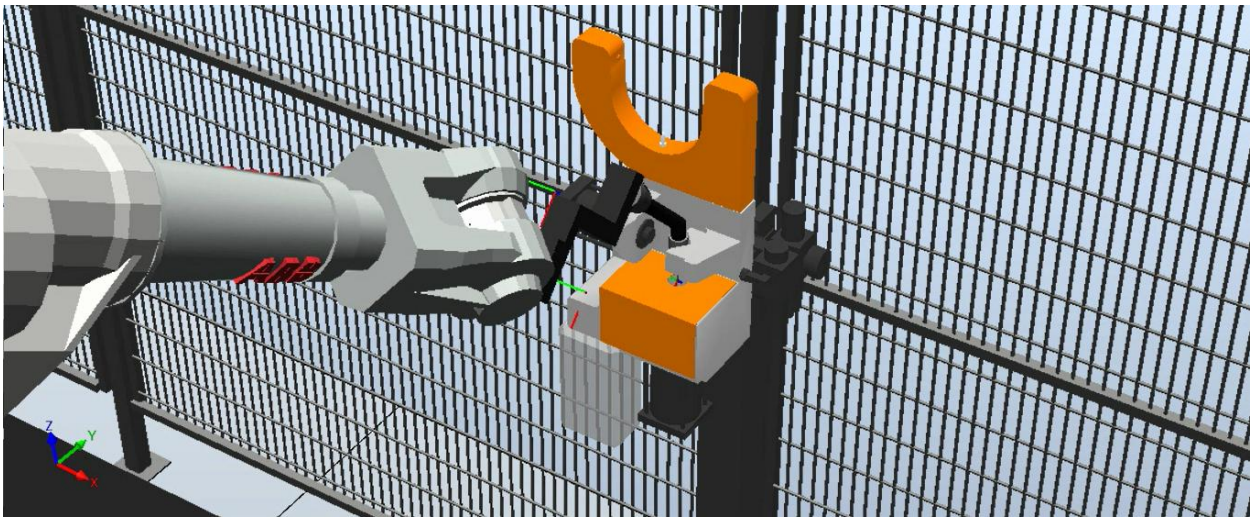


Figura 3-26. Limpieza en el Torch Cleaner del robot IV tras finalizar la soldadura.

3.3. Conclusiones y posibles mejoras

Para finalizar, se puede concluir que la estación creada en Robotstudio cumple con gran precisión la simulación de la estación real de EUCOMSA. Si bien se han intentado corregir todos los errores de posicionamiento para que exista una gran semejanza, no es posible garantizar con total exactitud una igualdad de magnitud y de distancia relativa entre los elementos que componen la estación simulada y la real, debido en parte por la gran cantidad de mediciones que se han tenido que llevar a cabo para la representación y la composición de la estación por diversas piezas de pequeño tamaño.

Como consecuencia de esto, de que el origen de coordenadas del robot real era ambiguo y difería del de la estación simulada y también que el robot IV de EUCOMSA estuvo continuamente en producción durante la mayor parte de la estancia del autor en la entidad colaboradora, finalmente no ha sido posible la transferencia del programa de RAPID al robot real.

Por otro lado, como ya se ha dicho, se ha programado la estación tal y como funciona en la realidad, sin atenerse a realizar mejoras sobre la misma. Pese a ello, se van a exponer algunas posibles mejoras que se pueden realizar sobre la estación, relativas principalmente a sensores, ya que como se ha visto en ningún momento se han utilizado para esta estación.

Una mejora es la relativa al choque de la pistola con los gatos en el proceso de soldadura. En el programa definitivo se ha realizado un análisis observando detalladamente la simulación, y a partir de ahí, utilizando la instrucción ArcL, se han efectuado trayectorias de soldadura intermedias en las que la pistola de soldadura se inclina un ángulo determinado y así evitar el choque con los gatos de agarre. Este proceso para esquivar los gatos durante la soldadura es el mismo que el que se realiza en las instalaciones de EUCOMSA. Sin embargo, una mejora pausible a aplicar en la estación simulada podría ser instalar un sensor de choque o un sensor volumétrico sobre la pistola de soldadura, de manera que avisara al controlador cada vez que se activa (cada vez que toca un gato) y suspender el programa para evitar riesgo de daño o rotura en el material, por ejemplo.

Otra mejora posible es incorporar un sensor de línea a la entrada de la estación por la zona de carga/descarga de material, de manera que en caso de detectar presencia mientras la mesa está girando, se suspenda el programa y la mesa se pare por seguridad.

Como se aprecia, estas mejoras a incorporar van destinadas a la prevención de riesgos laborales y mejora de la seguridad de los operarios. Por tanto, al tener como objetivo suspender temporalmente el proceso automático habitual de soldadura, se deben considerar externas a la producción normal de la estación robotizada.

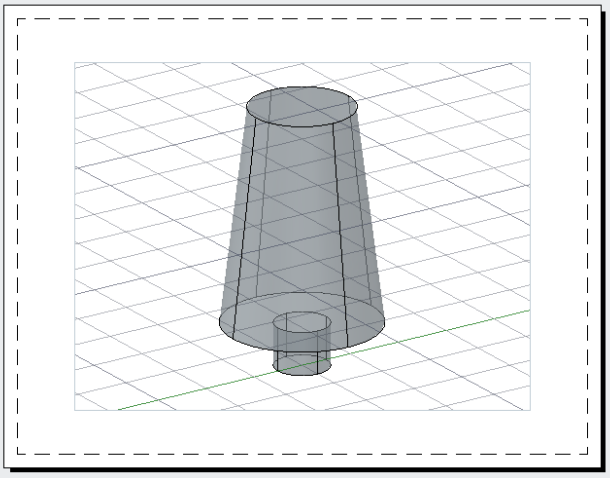
4 ANEXOS

Para empezar un gran proyecto, hace falta valentía.
Para terminar un gran proyecto, hace falta perseverancia.
- Anónimo -

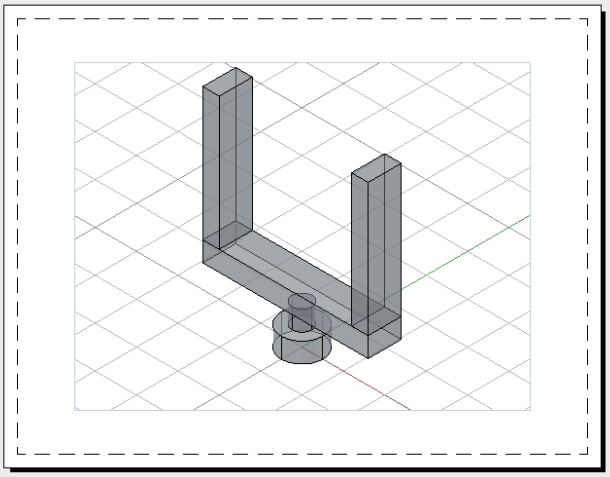
El último capítulo de este trabajo consiste en los anexos del mismo. Como se ha ido adelantando durante la redacción, el conjunto de anexos necesarios son tres. El anexo A para mostrar el conjunto de planos de todas las piezas diseñadas en CAD. El anexo B para las tablas en las que se encuentran los listados de señales de E/S de los controladores utilizados. Y el anexo C para mostrar los códigos de todos los programas de RAPID utilizados.

4.1. Anexo A: Planos

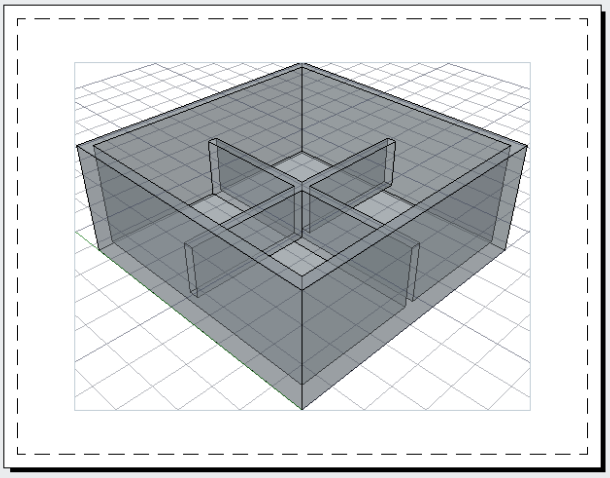
4.1.1. Planos de la estación de ejemplo



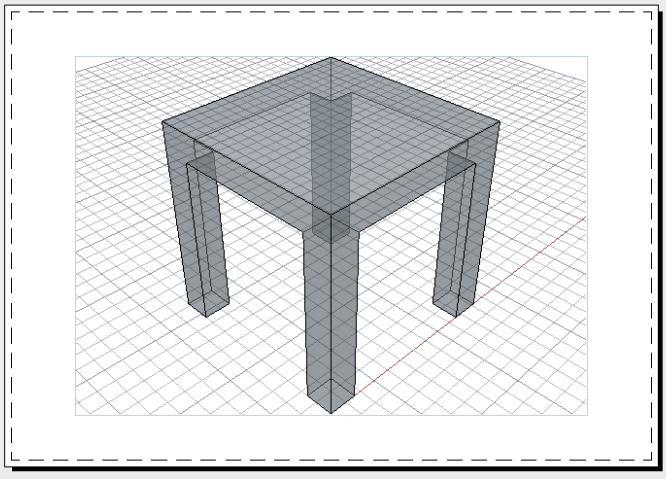
Plano 1. Perspectiva isométrica de la herramienta del robot 3.



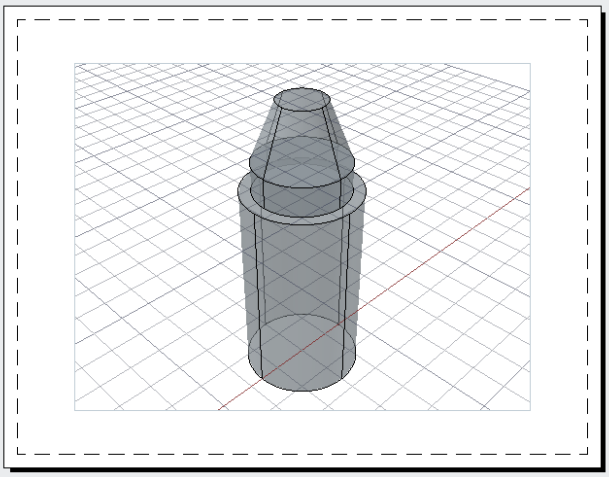
Plano 2. Perspectiva isométrica de la herramienta del robot 2.



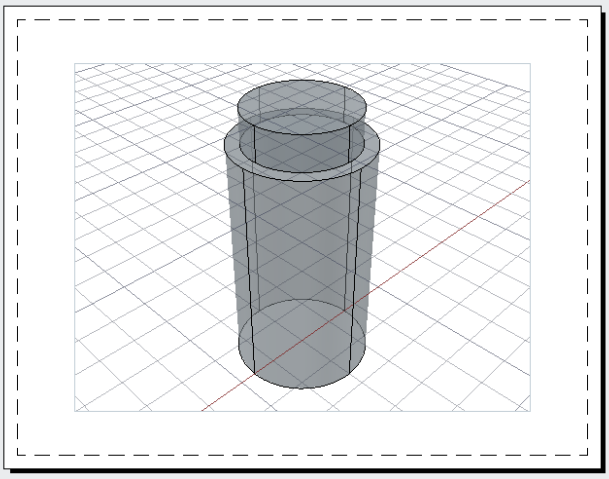
Plano 3. Perspectiva isométrica de la caja de colocación de cilindros.



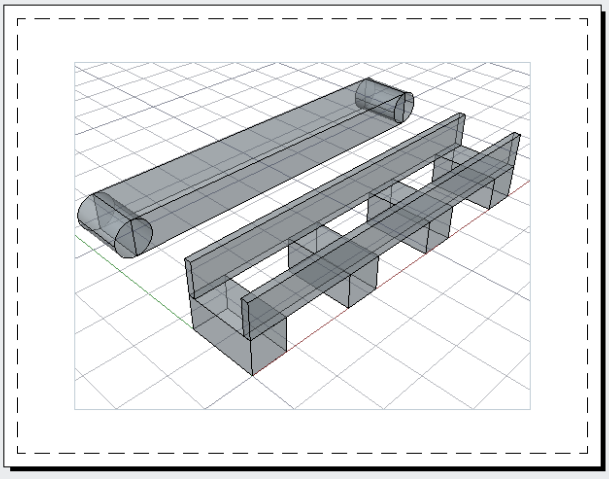
Plano 4. Perspectiva isométrica de la mesa de trabajo 1.



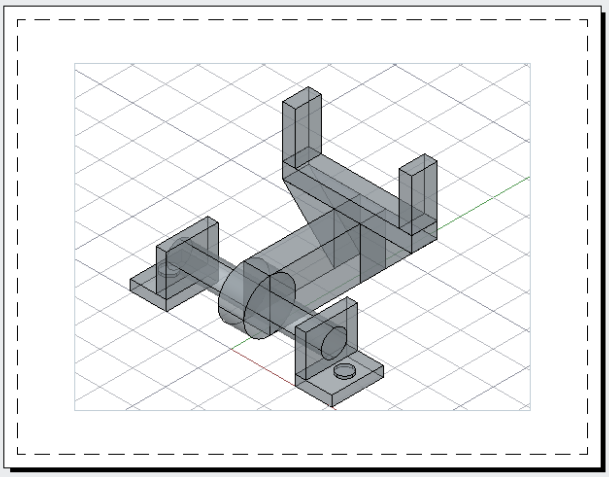
Plano 5. Perspectiva isométrica del cilindro verde.



Plano 6. Perspectiva isométrica del cilindro azul.

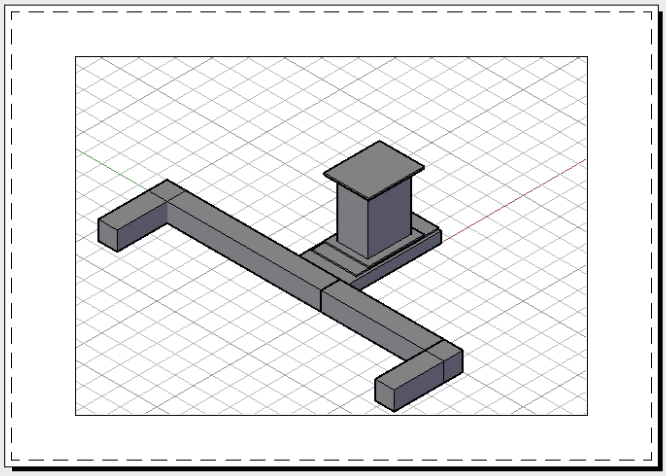


Plano 7. Perspectiva isométrica de la cinta transportadora (cuerpo y cinta).

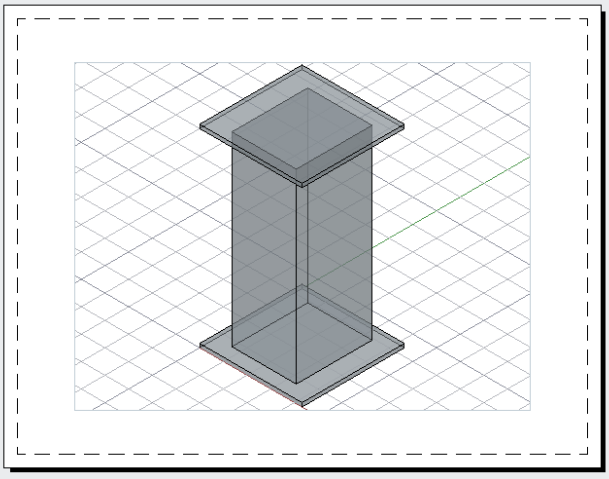


Plano 8. Perspectiva isométrica de una pinza de agarre.

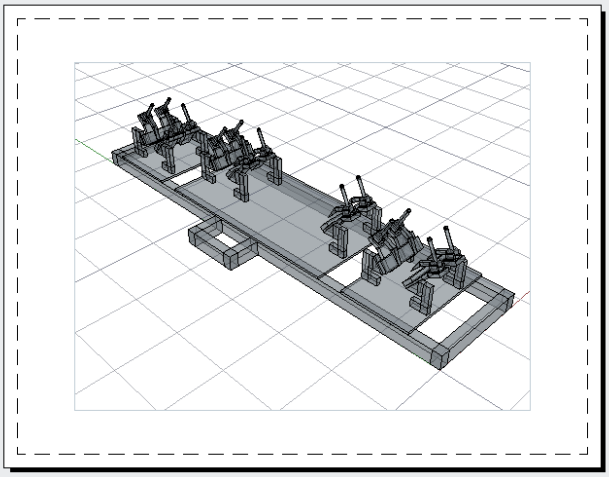
4.1.2. Planos de la estación de EUCOMSA



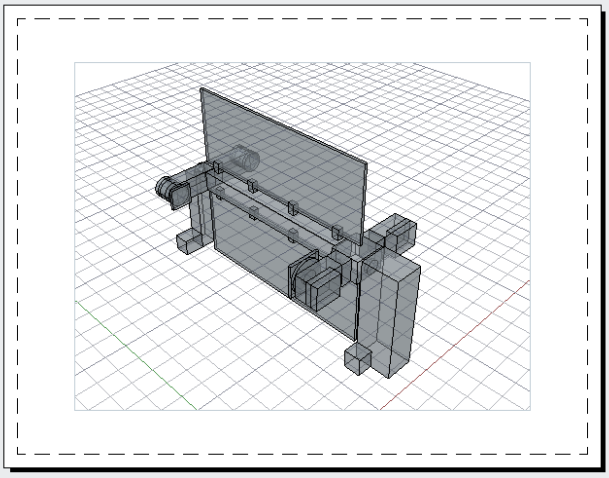
Plano 9. Perspectiva isométrica de la base del robot y las canaletas.



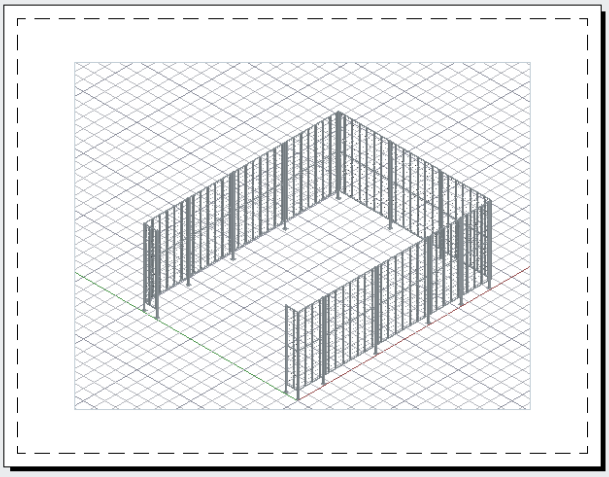
Plano 10. Perspectiva isométrica de la base del Torch Cleaner.



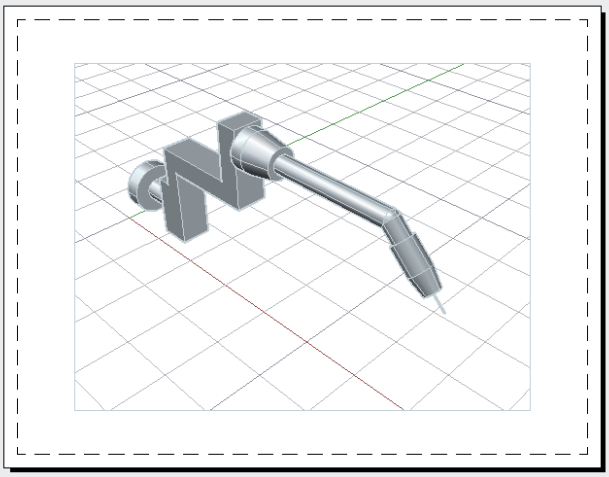
Plano 11. Perspectiva isométrica de la mesa de trabajo.



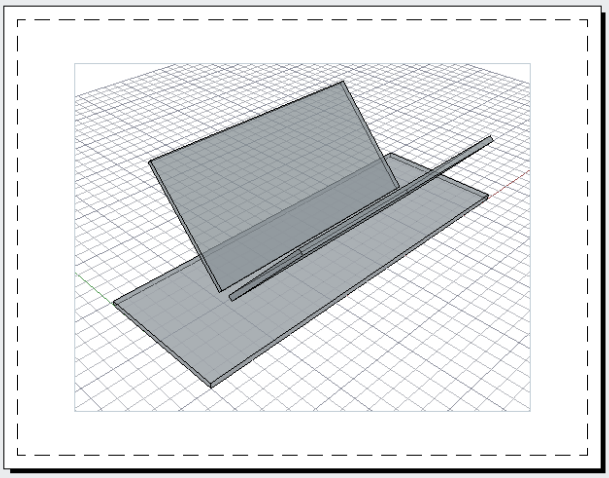
Plano 12. Perspectiva isométrica de los soportes (fijos y giratorios).



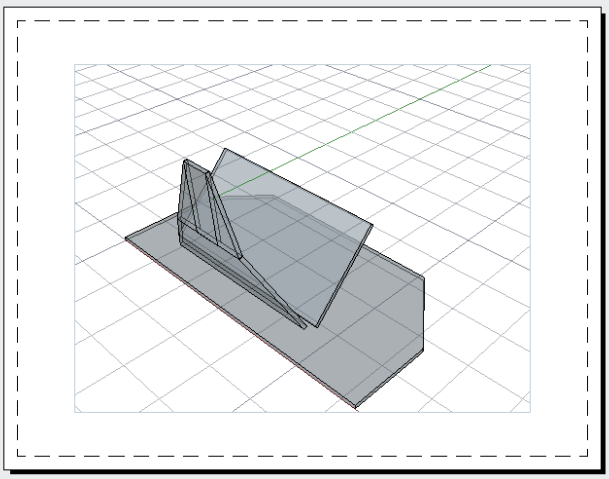
Plano 13. Perspectiva isométrica de las vallas del recinto de la estación.



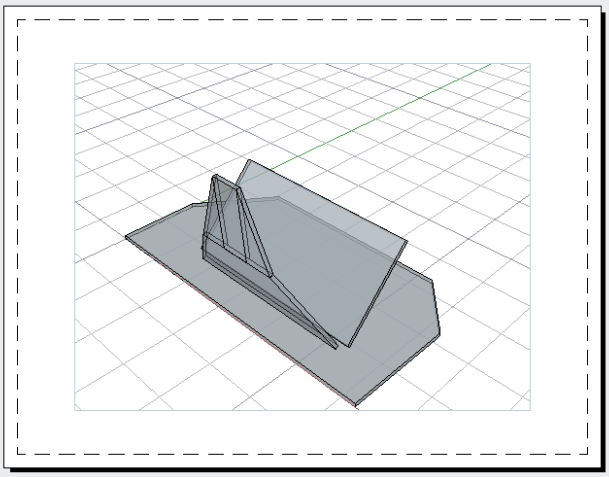
Plano 14. Perspectiva isométrica de la pistola de soldadura del robot.



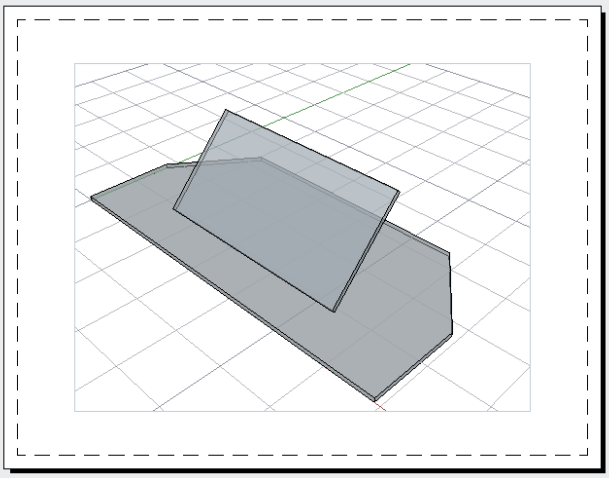
Plano 15. Perspectiva isométrica del nudo AA.



Plano 16. Perspectiva isométrica del nudo BA.



Plano 17. Perspectiva isométrica del nudo BB.



Plano 18. Perspectiva isométrica del nudo BC.

4.2. Anexo B: Señales

4.2.1. Estación de ejemplo

Módulo E/S. Controlador Robots 1 y 2.			
Entradas digitales	Mapeo	Salidas digitales	Mapeo
PiezaEnPinza	0	ActivaSensorPinza	0
PiezaCinta	1	ActivaCintaEntrada	1
CilindroVerde	2	ActivaCintaSalida	2
ColocadaEnCinta	3	ActivaSensoresCintaEntrada	3
ColocadaMesa	4	ActivaCierrePinzasAgarre	4
PinzasCerradas	5	ActivaGiroMesa	5
PinzasAbiertas	6	ActivaAperturaPinzasAgarre	6
ColocadaFinCintaSalida	7	ActivaSensorEnMesa	7
-	-	ActivaSensor1EnCintaSalida	8
-	-	ActivaSensorFinalCintaSalida	9

Módulo E/S. Controlador Robot 3.			
Entradas digitales	Mapeo	Salidas digitales	Mapeo
PiezaEnIman	0	ActivaSensorIman	0
TuboVerde2	1	-	-
PiezaFinalCintaSalida	2	-	-

4.2.2. Estación de EUCOMSA

Módulo E/S. Controlador Robot IV.			
Entradas digitales	Mapeo	Salidas digitales	Mapeo
En_Posicion_Soldadura	0	Start	0
En_Posicion_Descarga	1	Fin_Soldadura	1
Nudos_Colocados	2	Marcha_Giro_Mesa	2

4.3. Anexo C: Programas

4.3.1. Código de RAPID de la estación de ejemplo

ROBOT 1:

```
MODULE Module1
   CONST robtarget Pos_ini_soldadura:=[[890.046616726,0.889698843,1165.907316148],[0,0,1,0],[0,-
1,0,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
   CONST robtarget Pos_soldar_1:=[[-
80.155200976,0.003974025,339.842047736],[0.258819045,0,0.965925826,0],[-1,-
1,0,1],[9E9,9E9,9E9,9E9,9E9,9E9]];
   CONST robtarget Pos_soldar_2:=[[-
80.155240845,0.004275576,389.841280806],[0.258819045,0,0.965925826,0],[-1,-
1,0,1],[9E9,9E9,9E9,9E9,9E9,9E9]];

   PROC main()

   !!Robot en posición de inicio

   MoveJ Pos_ini_soldadura,v1000,z100,Tregaskiss22deg\WObj:=wobj0;

   Inicio: !Ejecución cíclica
   SingArea \Off; !!Restauramos la interpolación y la vigilancia de ejes.
   ConfL \On;
   WaitDI PiezaCinta,1; !!Esperamos a la pieza para parar la cinta de entrada
   IF CilindroVerde=0 AND PiezaCinta=1 THEN
      SoldaduraCilindroAzul; !!Entramos en la rutina de soldadura de la pieza azul
   ENDIF
    IF CilindroVerde=1 AND PiezaCinta=1 THEN
```

```
    SoldaduraCilindroVerde; !!Entramos en la rutina de soldadura de la pieza verde
  ENDIF
  GOTO Inicio; !Volvemos para procesar nueva pieza

  ENDPROC

  PROC SoldaduraCilindroAzul()
    WaitDI PinzasCerradas,1;!!Esperamos a que se cierren las pinzas de agarre (desde el módulo del
robot 2)
    Reset ActivaCierrePinzasAgarre;!!Reseteamos el cierre de las pinzas de agarre (lo ponemos a cero)
        MoveJ Offs(Pos_soldar_1,-
100,0,100),v500,fine,Tregaskiss22deg\WObj:=Mesa_trabajo_soldadura;!!Mueve a pos. aproximación
soldadura 1
    MoveL Pos_soldar_1,v200,fine,Tregaskiss22deg\WObj:=Mesa_trabajo_soldadura;!!Mueve a pos. de
soldadura 1
    Set ActivaGiroMesa;!!Activa el giro de la mesa
    WaitTime 6; !!Espera 6 segundos
    Reset ActivaGiroMesa;!!Desactiva el giro de la mesa
    MoveL Offs(Pos_soldar_1,-
100,0,100),v200,fine,Tregaskiss22deg\WObj:=Mesa_trabajo_soldadura;!!Mueve a pos. aproximación
soldadura 1
    MoveJ Pos_ini_soldadura,v500,fine,Tregaskiss22deg\WObj:=wobj0;!!Mueve a pos. inicial soldador
    Set ActivaAperturaPinzasAgarre;!!Activa la apertura de las pinzas de agarre
      ENDPROC

  PROC SoldaduraCilindroVerde()
    WaitDI PinzasCerradas,1;!!Esperamos a que se cierren las pinzas de agarre (desde el módulo del
robot 2)
    Reset ActivaCierrePinzasAgarre;!!Reseteamos el cierre de las pinzas de agarre (lo ponemos a cero)
        MoveJ Offs(Pos_soldar_1,-
100,0,100),v500,fine,Tregaskiss22deg\WObj:=Mesa_trabajo_soldadura;!!Mueve a pos. aproximación
soldadura 1
    MoveL Pos_soldar_1,v200,fine,Tregaskiss22deg\WObj:=Mesa_trabajo_soldadura;!!Mueve a pos. de
soldadura 1
    Set ActivaGiroMesa;!!Activa el giro de la mesa
    Reset ActivaGiroMesa;!!Desactiva el giro de la mesa
    WaitTime 6; !!Espera 6 segundos
    MoveL Offs(Pos_soldar_1,-
100,0,100),v200,fine,Tregaskiss22deg\WObj:=Mesa_trabajo_soldadura;!!Mueve a pos. aproximación
soldadura 1
    MoveJ Offs(Pos_soldar_2,-
100,0,100),v500,fine,Tregaskiss22deg\WObj:=Mesa_trabajo_soldadura;!!Mueve a pos. aproximación
soldadura 2
    MoveL Pos_soldar_2,v200,fine,Tregaskiss22deg\WObj:=Mesa_trabajo_soldadura;!!Mueve a pos. de
soldadura 2
    Set ActivaGiroMesa;!Activa el giro de la mesa
    Reset ActivaGiroMesa;!Desactiva el giro de la mesa
    WaitTime 6; !Espera 6 segundos
    MoveJ Offs(Pos_soldar_2,-
100,0,100),v200,fine,Tregaskiss22deg\WObj:=Mesa_trabajo_soldadura;!!Mueve a pos. aproximación
soldadura 2
    MoveJ Pos_ini_soldadura,v500,fine,Tregaskiss22deg\WObj:=wobj0;!!Mueve a pos. inicial soldador
```

```
    Set ActivaAperturaPinzasAgarre;!!Activa la apertura de las pinzas de agarre
        ENDPROC
ENDMODULE
```

ROBOT 2:

```
MODULE Module1
        CONST robtarget Pos_ini:=[[1700,0,1100],[0.183012702,-0.683012702,-
0.683012702,0.183012702],[0,0,-1,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
        CONST robtarget Origen_Cinta_Entrada:=[[100,350,210],[0,0,-0.707106781,0.707106781],[0,-
1,0,1],[9E9,9E9,9E9,9E9,9E9,9E9]];
        CONST robtarget Origen_mesa_circular:=[[100,0,200],[0.5,-0.5,-0.5,0.5],[0,-1,-
1,1],[9E9,9E9,9E9,9E9,9E9,9E9]];
        CONST robtarget Origen_Cinta_Salida:=[[100,150,210],[0.707106781,-0.707106781,0,0],[-1,0,-
1,1],[9E9,9E9,9E9,9E9,9E9,9E9]];

  VAR intnum Pieza_preparada_entrada; !Variable para la interrupción
  VAR intnum Pieza_preparada_salida; !Variable para la interrupción

  TRAP ParaCintaEntrada
    Reset ActivaCintaEntrada;
  ENDTRAP

  TRAP ParaCintaSalida
    Reset ActivaCintaSalida;
  ENDTRAP

  PROC main()

  !Habilitamos interrupción de las cintas de entrada y salida
  IEnable;
  CONNECT Pieza_preparada_entrada WITH ParaCintaEntrada;
  ISignalDI PiezaCinta,1, Pieza_preparada_entrada;

  IEnable;
  CONNECT Pieza_preparada_salida WITH ParaCintaSalida;
  ISignalDI ColocadaFinCintaSalida,1, Pieza_preparada_salida;

  !Reseteamos y activamos las E/S
  Set ActivaSensoresCintaEntrada;
  Set ActivaSensorFinalCintaSalida;
  Set ActivaCintaEntrada;
  Reset ActivaSensorPinza;

  !Robot en posición de inicio

  MoveJ Pos_ini,v500,z100,pinza\WObj:=wobj0;!Mueve a pos. inicial
```

```
Inicio: !Ejecución cíclica
SingArea \Off; !Restauramos la interpolación y la vigilancia de ejes.
ConfL \On;
ManipulacionCilindros;
GOTO Inicio; !Volvemos para procesar nueva pieza

ENDPROC

    PROC ManipulacionCilindros()
  WaitDI PiezaCinta,1; !!Esperamos a la pieza para parar la cinta de entrada

  MoveJ Offs(Origen_Cinta_Entrada,0,400,400),v500,fine,pinza\WObj:=Cinta_entrada;!!Mueve a pos. aproximación nº 1 cinta entrada
        MoveJ RelTool(Origen_Cinta_Entrada,0,0,-200),v500,fine,pinza\WObj:=Cinta_entrada;!!Mueve a pos. aproximación nº 1 cinta entrada
  MoveL RelTool(Origen_Cinta_Entrada,16,0,0),v200,fine,pinza\WObj:=Cinta_entrada;!!Mueve a pos. recogida pieza entrada (ajustado)
  Set ActivaSensorPinza;!!Activa sensor para coger pieza
  WaitDI PiezaEnPinza,1;!!Espera a que la pieza este cogida
  MoveL Offs(Origen_Cinta_Entrada,0,0,300),v200,fine,pinza\WObj:=Cinta_entrada;!!Mueve a pos. aproximación nº 2 cinta entrada
        Set ActivaCintaEntrada;
  MoveJ RelTool (Origen_mesa_circular,0,-100,-300),v500,fine,pinza\WObj:=Mesa_trabajo_circular;!!Mueve a pos. aproximación nº 1 mesa
  MoveJ RelTool (Origen_mesa_circular,0,-100,0),v200,fine,pinza\WObj:=Mesa_trabajo_circular;!!Mueve a pos. aproximación nº 2 mesa
  MoveL Origen_mesa_circular,v200,fine,pinza\WObj:=Mesa_trabajo_circular;!!Mueve a pos. de trabajo de la mesa
  Reset ActivaSensorPinza;!!Desactiva el sensor para coger la pieza
  WaitDI PiezaEnPinza,0;!!Espera a que la pieza esté sin coger
  Set ActivaSensorEnMesa;!!Posiciona la pieza en la mesa
  WaitDI ColocadaMesa,1;!!Espera a que la pieza esté exactamente en su posición en la mesa
  MoveL Offs(Origen_mesa_circular,200,0,0),v200,fine,pinza\WObj:=Mesa_trabajo_circular;!!Mueve a pos. aproximación nº 3 mesa
  MoveJ Pos_ini,v1000,z100,pinza\WObj:=wobj0;!!Mueve a pos. inicial
  Set ActivaCierrePinzasAgarre;!!Activa el cierre de las pinzas de agarre
  WaitDI PinzasAbiertas,1;!!Esperamos a que se abran las pinzas de agarre (desde el módulo del robot 1)
  Reset ActivaAperturaPinzasAgarre;!!Reseteamos la apertura de las pinzas de agarre (la ponemos a cero)
  MoveJ Offs(Origen_mesa_circular,200,0,0),v500,fine,pinza\WObj:=Mesa_trabajo_circular;!!Mueve a pos. aproximación nº 3 mesa
  MoveL Origen_mesa_circular,v200,fine,pinza\WObj:=Mesa_trabajo_circular;!!Mueve a pos. de trabajo de la mesa
  Set ActivaSensorPinza;!!Activa sensor para coger pieza
  WaitDI PiezaEnPinza,1;!!Espera a que la pieza este cogida
  MoveL Offs(Origen_mesa_circular,0,0,300),v200,fine,pinza\WObj:=Mesa_trabajo_circular;!!Mueve a pos. aproximación nº 1 mesa
  MoveL Offs(Origen_mesa_circular,200,350,300),v200,fine,pinza\WObj:=Mesa_trabajo_circular;!!Mueve a pos. aproximación nº 4 mesa
```

```
    MoveJ RelTool (Origen_Cinta_Salida,0,-300,0),v500,fine,pinza\WObj:=Cinta_salida;!!Mueve a pos.
aproximación nº 1 cinta salida
    MoveL Origen_Cinta_Salida,v200,fine,pinza\WObj:=Cinta_salida;!!Mueve a pos. depositar pieza en
cinta salida
    Reset ActivaSensorPinza;!!Desactiva el sensor para coger la pieza
    WaitDI PiezaEnPinza,0;!!Espera a que la pieza esté sin coger
    Set ActivaSensor1EnCintaSalida;!!Posiciona la pieza en la cinta de salida
    WaitDI ColocadaEnCinta,1;!!Espera a que la pieza esté exactamente en su posición de la cinta de
salida
    MoveL Offs(Origen_Cinta_Salida,0,-200,0),v200,fine,pinza\WObj:=Cinta_salida;!!Mueve a pos.
aproximación nº 2 cinta salida

    MoveJ Offs(Pos_ini,0,0,-300),v500,z100,pinza\WObj:=wobj0;!!Mueve a pos. inicial
    Set ActivaCintaSalida;!!Activamos la cinta de salida
        ENDPROC
ENDMODULE
```

ROBOT 3:

```
MODULE Module1
        CONST robtarget Recogida_cilindro:=[[2400,250,410],[0,1,0,0],[0,0,-
1,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
        CONST robtarget Coloca_tubo_azul:=[[1070,340,20],[0,1,0,0],[-1,0,-
2,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
        CONST robtarget Coloca_tubo_verde:=[[270,340,20],[0,1,0,0],[0,0,-
1,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
        CONST robtarget Pos_ini_iman:=[[692.975224681,482.210624927,1111.603897019],[0,-
0.707106781,0.707106781,0],[0,0,-1,0],[9E9,9E9,9E9,9E9,9E9,9E9]];

  VAR num i:=0; !!Número de cilindros azules con proceso de soldadura finalizado
  VAR num j:=0; !!Número de cilindros verdes con proceso de soldadura finalizado
  CONST num x:=260; !!Distancia entre centros de cilindros al colocar en la caja de salida

  PROC main()

  !!Robot en posición de inicio

  MoveJ Pos_ini_iman,v1000,z100,Iman\WObj:=wobj0;
  Reset ActivaSensorIman;

  Inicio: !!Ejecución cíclica
  SingArea \Off; !!Restauramos la interpolación y la vigilancia de ejes.
  ConfL \On;
  WaitDI PiezaFinalCintaSalida,1; !!Esperamos a la pieza para parar la cinta de salida
  IF TuboVerde2=0 AND PiezaFinalCintaSalida=1 THEN
    ColocarCilindroAzul; !!Entramos en la rutina de colocación de la pieza azul
  ENDIF
   IF TuboVerde2=1 AND PiezaFinalCintaSalida=1 THEN
```

```
    ColocarCilindroVerde; !!Entramos en la rutina de colocación de la pieza verde
ENDIF
GOTO Inicio; !!Volvemos para procesar nueva pieza

ENDPROC

PROC ColocarCilindroAzul()

        MoveJ Offs(Recogida_cilindro,0,0,200),v500,fine,Iman\WObj:=CintaSalida_2;
  MoveL Recogida_cilindro,v200,fine,Iman\WObj:=CintaSalida_2;
  Set ActivaSensorIman;!!Activa sensor para coger pieza
  WaitDI PiezaEnIman,1;!!Espera a que la pieza este cogida
  i:=i+1; !!Incrementamos el nº de cilindros azules ya soldados
  MoveJ Pos_ini_iman,v500,fine,Iman\WObj:=wobj0;
  MoveJ Offs(Coloca_tubo_azul,0,-200,700),v500,fine,Iman\WObj:=Mesa_blanca;

  IF i=1 THEN
        MoveJ Offs(Coloca_tubo_azul,x,x,900),v500,fine,Iman\WObj:=Mesa_blanca;
    MoveL Offs(Coloca_tubo_azul,x,x,400),v200,fine,Iman\WObj:=Mesa_blanca;
  ENDIF

  IF i=2 THEN
        MoveJ Offs(Coloca_tubo_azul,0,x,900),v500,fine,Iman\WObj:=Mesa_blanca;
    MoveL Offs(Coloca_tubo_azul,0,x,400),v200,fine,Iman\WObj:=Mesa_blanca;
  ENDIF

  IF i=3 THEN
        MoveJ Offs(Coloca_tubo_azul,x,0,700),v500,fine,Iman\WObj:=Mesa_blanca;
    MoveL Offs(Coloca_tubo_azul,x,0,400),v200,fine,Iman\WObj:=Mesa_blanca;
  ENDIF

  IF i=4 THEN
        MoveJ Offs(Coloca_tubo_azul,0,0,700),v500,fine,Iman\WObj:=Mesa_blanca;
    MoveL Offs(Coloca_tubo_azul,0,0,400),v200,fine,Iman\WObj:=Mesa_blanca;
    i:=0;
  ENDIF

  Reset ActivaSensorIman;!!Desactiva el sensor para coger la pieza
  WaitDI PiezaEnIman,0;!!Espera a que la pieza esté sin coger
  MoveL Offs(Coloca_tubo_azul,0,0,500),v200,fine,Iman\WObj:=Mesa_blanca;
        MoveJ Pos_ini_iman,v500,z100,Iman\WObj:=wobj0;
  WaitDI PiezaFinalCintaSalida,1; !!Esperamos a la pieza para parar la cinta de salida
     ENDPROC

PROC ColocarCilindroVerde()

        MoveJ Offs(Recogida_cilindro,0,0,300),v500,fine,Iman\WObj:=CintaSalida_2;
  MoveL Offs(Recogida_cilindro,0,0,100),v200,fine,Iman\WObj:=CintaSalida_2;
  Set ActivaSensorIman;!!Activa sensor para coger pieza
  WaitDI PiezaEnIman,1;!!Espera a que la pieza este cogida
  j:=j+1; !!Incrementamos el nº de cilindros verdes ya soldados
  MoveJ Pos_ini_iman,v500,fine,Iman\WObj:=wobj0;
```

```
    MoveJ Offs(Coloca_tubo_verde,150,-150,800),v500,fine,Iman\WObj:=Mesa_blanca;

    IF j=1 THEN
        MoveJ Offs(Coloca_tubo_verde,0,x,750),v500,fine,Iman\WObj:=Mesa_blanca;
      MoveL Offs(Coloca_tubo_verde,0,x,500),v200,fine,Iman\WObj:=Mesa_blanca;
    ENDIF

    IF j=2 THEN
        MoveJ Offs(Coloca_tubo_verde,x,x,800),v500,fine,Iman\WObj:=Mesa_blanca;
      MoveL Offs(Coloca_tubo_verde,x,x,500),v200,fine,Iman\WObj:=Mesa_blanca;
    ENDIF

    IF j=3 THEN
        MoveJ Offs(Coloca_tubo_verde,0,0,800),v500,fine,Iman\WObj:=Mesa_blanca;
      MoveL Offs(Coloca_tubo_verde,0,0,500),v200,fine,Iman\WObj:=Mesa_blanca;
    ENDIF

    IF j=4 THEN
        MoveJ Offs(Coloca_tubo_verde,x,0,800),v500,fine,Iman\WObj:=Mesa_blanca;
      MoveL Offs(Coloca_tubo_verde,x,0,500),v200,fine,Iman\WObj:=Mesa_blanca;
      j:=0;
    ENDIF

    Reset ActivaSensorIman;!!Desactiva el sensor para coger la pieza
    WaitDI PiezaEnIman,0;!!Espera a que la pieza esté sin coger
    MoveL Offs(Coloca_tubo_verde,0,0,600),v200,fine,Iman\WObj:=Mesa_blanca;
        MoveJ Pos_ini_iman,v500,z100,Iman\WObj:=wobj0;
    WaitDI PiezaFinalCintaSalida,1; !!Esperamos a la pieza para parar la cinta de salida
       ENDPROC


ENDMODULE
```

4.3.2. Código de RAPID de la estación de EUCOMSA

ROBOT IV:

```
MODULE Module1
    TASK PERS wobjdata
Mesasoldar:=[FALSE,TRUE,"",[[1627.66,1226,762.049],[0,0,0,1]],[[0,0,0],[1,0,0,0]]];
    CONST robtarget Pos_ini:=[[834.336287211,-
2.196369932,1420.994040955],[0.422618261,0,0.906307787,0],[-1,0,-1,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
    CONST robtarget
Pos_comienzo_AA:=[[151.244395021,359.028480792,88.782770625],[0.044642587,0.868162779,0.339
094114,-0.359604798],[0,-1,-1,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
    CONST robtarget
Pos_Comienzo_BA:=[[344.708621161,2132.47097009,84.782660869],[0.003996236,0.866016184,-
0.286788218,-0.409576022],[-1,0,-1,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
    CONST robtarget
Pos_Comienzo_BB:=[[446.128215482,1620.104186426,83.143978372],[0,0.923879533,0,-
0.382683432],[-1,0,-1,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
    CONST robtarget
Pos_Comienzo_BC:=[[204.879180481,909.791427686,84.143974372],[0.35355339,0.85355339,0.14644
661,-0.353553392],[-1,0,-1,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
    CONST robtarget Pos_ini_TCP_10:=[[-76.887488377,-
830.788519706,1420.994117819],[0.27059805,0.653281482,0.653281482,-0.27059805],[-2,0,-
1,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
    CONST robtarget Pos_TCP_1:=[[-70.464367665,-
1441.734355978,764.768185621],[0.27059805,0.653281482,0.653281482,-0.27059805],[-2,0,-
1,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
    CONST robtarget Pos_TCP_2:=[[-254.437742997,-
1444.278701474,695.218825428],[0.27059805,0.653281482,0.653281482,-0.27059805],[-2,0,-
1,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
    CONST robtarget Pos_TCP_3:=[[-146.411081959,-
1396.047216695,662.352936174],[0.27059805,0.653281482,0.653281482,-0.27059805],[-2,0,-
1,0],[9E9,9E9,9E9,9E9,9E9,9E9]];
    VAR robtarget Pos_comienzo_AA_2;
    VAR robtarget Pos_comienzo_BB_2;
    VAR robtarget Pos_comienzo_BA_2;
    VAR robtarget Pos_comienzo_BB_3;
    VAR robtarget Pos_comienzo_BA_3;
    PERS seamdata seam1 :=
[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,[0,0,0,0,0,0,0,0,0,0],0];
    PERS welddata weld2 := [7.5,0,[2,0.12,21.25,186.667,0.8,8,0,0,0],[0,0,0,0,0,0,0,0,0]];
    PERS welddata weld3 := [8.2,0,[3,0.12,24.5,205,0.8,8,0,0,0],[0,0,0,0,0,0,0,0,0]];
    PERS welddata weld4 := [7.5,0,[4,0.12,22.75,161.667,0.8,8,0,0,0],[0,0,0,0,0,0,0,0,0]];
    PERS welddata weld10 := [7.5,0,[10,0.12,22.75,150,0.8,8,0,0,0],[0,0,0,0,0,0,0,0,0]];
    PERS welddata weld11 := [6.7,0,[11,0.12,19.5,118.333,0.8,8,0,0,0],[0,0,0,0,0,0,0,0,0]];
    PERS welddata weld26 := [7.8,0,[26,0.12,24.75,245,0.8,8,0,0,0],[0,0,0,0,0,0,0,0,0]];
    PERS weavedata weave1 := [1,0,7.98,5,0,1.5,0.5,1.5,60,0,0,0,0,0,0];
    PROC main()
      Reset Fin_Soldadura;
```

```
            Set Start;
        MoveJ Pos_ini_TCP_10,v1000,z100,Dinse\WObj:=wobj0;
        WaitDI Nudos_Colocados,1;
        WaitTime 3;
        Set Marcha_Giro_Mesa;
            WaitDI En_Posicion_Soldadura,1;
        Reset Start;
        Reset Marcha_Giro_Mesa;

        !Soldadura Nudo AA

        MoveJ Pos_ini,v1000,z100,Dinse\WObj:=wobj0;
        Pos_comienzo_AA_2 := RelTool(Pos_comienzo_AA,0,0,0 \Rx := 15 \Rz:=10);
        MoveJ Offs(Pos_comienzo_AA,0,0,20),v1000,fine,Dinse\WObj:=Mesasoldar;
        ArcLStart Pos_comienzo_AA_2,v1000,seam1, weld2, fine, Dinse\WObj:=Mesasoldar;
        ArcL Offs(Pos_comienzo_AA_2,100,0,0),v1000,seam1,weld2,fine, Dinse\WObj:=Mesasoldar;
        ArcLEnd Offs(Pos_comienzo_AA,221.52,0,0),v1000,seam1, weld2, fine, Dinse\WObj:=Mesasoldar;
        MoveL Offs(Pos_comienzo_AA_2,221.52,0,20),v1000,fine,Dinse\WObj:=Mesasoldar;
        MoveL Offs(Pos_comienzo_AA_2,0,7.98,20),v1000,fine,Dinse\WObj:=Mesasoldar;
        ArcLStart Offs(Pos_comienzo_AA_2,0,7.98,0),v1000,seam1, weld2, fine,
Dinse\WObj:=Mesasoldar;
        ArcL Offs(Pos_comienzo_AA_2,100,7.98,0),v1000,seam1,weld2,fine, Dinse\WObj:=Mesasoldar;
        ArcLEnd Offs(Pos_comienzo_AA,221.52,7.98,0),v1000,seam1, weld2, fine,
Dinse\WObj:=Mesasoldar;
        MoveL Offs(Pos_comienzo_AA_2,221.52,7.98,20),v1000,fine,Dinse\WObj:=Mesasoldar;
        MoveL Offs(Pos_comienzo_AA_2,0,3.99,20),v1000,fine,Dinse\WObj:=Mesasoldar;
        ArcLStart Offs(Pos_comienzo_AA_2,0,3.99,0),v1000,seam1, weld3\weave:=weave1, fine,
Dinse\WObj:=Mesasoldar;
        ArcL Offs(Pos_comienzo_AA_2,100,3.99,0),v1000,seam1,weld3\weave:=weave1,fine,
Dinse\WObj:=Mesasoldar;
        ArcLEnd Offs(Pos_comienzo_AA,221.52,3.99,0),v1000,seam1, weld3\weave:=weave1, fine,
Dinse\WObj:=Mesasoldar;
        MoveL Offs(Pos_comienzo_AA,221.52,3.99,20),v1000,fine,Dinse\WObj:=Mesasoldar;
        MoveJ Pos_ini,v1000,z100,Dinse\WObj:=wobj0;

        !Soldadura Nudo BC

        MoveJ Offs(Pos_comienzo_BC,0,0,20),v1000,fine,Dinse\WObj:=Mesasoldar;
        ArcLStart Pos_comienzo_BC,v1000,seam1, weld11, fine, Dinse\WObj:=Mesasoldar;
        ArcLEnd Offs(Pos_comienzo_BC,243,0,0),v1000,seam1, weld11, fine, Dinse\WObj:=Mesasoldar;
        MoveL Offs(Pos_comienzo_BC,243,0,20),v1000,fine,Dinse\WObj:=Mesasoldar;
        MoveL Offs(Pos_comienzo_BC,0,0,20),v1000,fine,Dinse\WObj:=Mesasoldar;
        ArcLStart Offs(Pos_comienzo_BC,0,0,-4),v1000,seam1, weld4, fine, Dinse\WObj:=Mesasoldar;
        ArcLEnd Offs(Pos_comienzo_BC,243,0,-4),v1000,seam1, weld4, fine, Dinse\WObj:=Mesasoldar;
        MoveL Offs(Pos_comienzo_BC,243,10,-4),v1000,fine,Dinse\WObj:=Mesasoldar;
        MoveJ Pos_ini,v1000,z100,Dinse\WObj:=wobj0;

        !Soldadura Nudo BB

        Pos_comienzo_BB_2 := RelTool(Pos_comienzo_BB,0,0,0 \Rx := 10);
        Pos_comienzo_BB_3 := RelTool(Pos_comienzo_BB,0,0,0 \Rx := 20);
        MoveJ Offs(Pos_comienzo_BB,0,0,20),v1000,fine,Dinse\WObj:=Mesasoldar;
        ArcLStart Pos_comienzo_BB_2,v1000,seam1, weld10, fine, Dinse\WObj:=Mesasoldar;
```

```
        ArcLEnd Offs(Pos_comienzo_BB_2,-243,0,0),v1000,seam1, weld10, fine,
Dinse\WObj:=Mesasoldar;
        MoveL Offs(Pos_comienzo_BB_2,-243,0,20),v1000,fine,Dinse\WObj:=Mesasoldar;
        MoveL Offs(Pos_comienzo_BB_2,-14.63,4.29,18.64),v1000,fine,Dinse\WObj:=Mesasoldar;
        ArcLStart Offs(Pos_comienzo_BB_3,-14.63,4.29,-1.36),v1000,seam1, weld10, fine,
Dinse\WObj:=Mesasoldar;
        ArcL Offs(Pos_comienzo_BB_3,-54.63,4.29,-1.36),v1000,seam1,weld10,fine,
Dinse\WObj:=Mesasoldar;
        ArcLEnd Offs(Pos_comienzo_BB_2,-234.63,4.29,-1.36),v1000,seam1, weld10, fine,
Dinse\WObj:=Mesasoldar;
        MoveL Offs(Pos_comienzo_BB_2,-234.63,4.29,20),v1000,fine,Dinse\WObj:=Mesasoldar;
        MoveL Offs(Pos_comienzo_BB_2,-243,2.145,20),v1000,fine,Dinse\WObj:=Mesasoldar;
        ArcLStart Offs(Pos_comienzo_BB_2,-243,2.145,0),v1000,seam1, weld26\weave:=weave1, fine,
Dinse\WObj:=Mesasoldar;
        ArcL Offs(Pos_comienzo_BB_2,-183,2.145,0),v1000,seam1,weld26\weave:=weave1,fine,
Dinse\WObj:=Mesasoldar;
        ArcLEnd Offs(Pos_comienzo_BB_3,0,2.145,0),v1000,seam1, weld26\weave:=weave1, fine,
Dinse\WObj:=Mesasoldar;
        MoveL Offs(Pos_comienzo_BB,0,2.145,20),v1000,fine,Dinse\WObj:=Mesasoldar;
        MoveJ Pos_ini,v1000,z100,Dinse\WObj:=wobj0;

        !Soldadura Nudo BA

        Pos_comienzo_BA_2 := RelTool(Pos_comienzo_BA,0,0,0 \Rx:=-5 \Rz := -30);
        Pos_comienzo_BA_3 := RelTool(Pos_comienzo_BA,0,0,0 \Rx:=5 \Rz := -30);
        MoveJ Offs(Pos_comienzo_BA,0,0,20),v1000,fine,Dinse\WObj:=Mesasoldar;
        ArcLStart Pos_comienzo_BA_2,v1000,seam1, weld10, fine, Dinse\WObj:=Mesasoldar;
        ArcLEnd Offs(Pos_comienzo_BA_2,-243,0,0),v1000,seam1, weld10, fine,
Dinse\WObj:=Mesasoldar;
        MoveL Offs(Pos_comienzo_BA_2,-243,0,20),v1000,fine,Dinse\WObj:=Mesasoldar;
        MoveL Offs(Pos_comienzo_BA_2,-14.63,4.29,18.64),v1000,fine,Dinse\WObj:=Mesasoldar;
        ArcLStart Offs(Pos_comienzo_BA_3,-14.63,4.29,-1.36),v1000,seam1, weld10, fine,
Dinse\WObj:=Mesasoldar;
        !ArcL Offs(Pos_comienzo_BA_3,-54.63,4.29,-1.36),v1000,seam1,weld10,fine,
Dinse\WObj:=Mesasoldar;
        ArcLEnd Offs(Pos_comienzo_BA_2,-234.63,4.29,-1.36),v1000,seam1, weld10, fine,
Dinse\WObj:=Mesasoldar;
        MoveL Offs(Pos_comienzo_BA_2,-234.63,4.29,20),v1000,fine,Dinse\WObj:=Mesasoldar;
        MoveL Offs(Pos_comienzo_BA_2,-243,2.145,20),v1000,fine,Dinse\WObj:=Mesasoldar;
        ArcLStart Offs(Pos_comienzo_BA_2,-243,2.145,0),v1000,seam1, weld26\weave:=weave1, fine,
Dinse\WObj:=Mesasoldar;
        !ArcL Offs(Pos_comienzo_BA_2,-183,2.145,0),v1000,seam1,weld26\weave:=weave1,fine,
Dinse\WObj:=Mesasoldar;
        ArcLEnd Offs(Pos_comienzo_BA_3,0,2.145,0),v1000,seam1, weld26\weave:=weave1, fine,
Dinse\WObj:=Mesasoldar;
        MoveL Offs(Pos_comienzo_BA,0,2.145,20),v1000,fine,Dinse\WObj:=Mesasoldar;
        MoveJ Pos_ini,v1000,z100,Dinse\WObj:=wobj0;
        MoveJ Pos_ini_TCP_10,v1000,z100,Dinse\WObj:=wobj0;

        WaitTime 1;
        Set Fin_Soldadura;
        WaitDI En_Posicion_Descarga,1;
```

```
        Reset Fin_Soldadura;

        !Limpieza pistola

        MoveJ Offs(Pos_TCP_1,0,0,100),v1000,fine,Dinse\WObj:=wobj0;
        MoveL Pos_TCP_1,v100,fine,Dinse\WObj:=wobj0;
        WaitTime 5;
        MoveL Offs(Pos_TCP_1,0,0,100),v1000,fine,Dinse\WObj:=wobj0;
        MoveJ Offs(Pos_TCP_2,0,0,100),v100,fine,Dinse\WObj:=wobj0;
        MoveL Pos_TCP_2,v100,fine,Dinse\WObj:=wobj0;
        WaitTime 5;
        MoveL Offs(Pos_TCP_2,0,0,100),v1000,fine,Dinse\WObj:=wobj0;
        MoveL Offs(Pos_TCP_2,0,100,100),v100,fine,Dinse\WObj:=wobj0;
        MoveJ Offs(Pos_TCP_3,0,100,100),v100,fine,Dinse\WObj:=wobj0;
        MoveJ Offs(Pos_TCP_3,0,0,100),v100,fine,Dinse\WObj:=wobj0;
        MoveL Pos_TCP_3,v100,fine,Dinse\WObj:=wobj0;
        WaitTime 5;
        MoveJ Offs(Pos_TCP_3,0,100,0),v1000,fine,Dinse\WObj:=wobj0;
        MoveJ Offs(Pos_TCP_3,0,100,100),v1000,fine,Dinse\WObj:=wobj0;
        MoveJ Pos_ini_TCP_10,v1000,z100,Dinse\WObj:=wobj0;

         ENDPROC
ENDMODULE
```

REFERENCIAS

Material del curso de Robotstudio de la asignatura Control de robots manipuladores, del Máster en Ingeniería automática, robótica y telemática de la Universidad de Sevilla, impartido por el profesor David Muñoz.

Curso de Robotstudio de 7 temas de José Angel Alonso. Enlace:
https://www.youtube.com/playlist?list=PLgwW4GLuvZFGjfaZ90RTqxjYaw3DWJ9jI

Manual de Robotstudio, versión 5.14: Manual_español_v5.14.pdf

Manual de referencia de RAPID de la Ayuda de Robostudio.

Documentación adicional de la Ayuda de Robotstudio sobre instrucciones Arc y Arc Sensor.

Introducción a la robótica industrial. Ingeniería de sistemas y automática. Universidad de Zaragoza.

Temas 8 y 9 de la asignatura Automatización Industrial de 3º GITI, disa, Universidad de Sevilla.