UNIVERSIDAD MIGUEL HERNÁNDEZ

MÁSTER UNIVERSITARIO EN INGENIERÍA INDUSTRIAL

TRABAJO FÍN DE MÁSTER

INTEGRACIÓN Y COMUNICACIONES EN UNA CÉLULA DE PALETIZADO CON VISIÓN ARTIFICIAL Y ROBOT COLABORATIVO.

TRABAJO FIN DE MÁSTER

Febrero - 2026

AUTOR. Guillermo Antonio, Gómez Orts

DIRECTORA. Mónica Ballesta Galdeano

Contenido

1. Introducción

La industria moderna se encuentra en constante desarrollo y evolución, actualmente dirigida hacia la automatización y robótica, es lo que conocemos como Industria 4.0.

En este contexto los robots colaborativos y las herramientas de visión artificial se han convertido en grandes aliados de los procesos productivos y están cada día más presentes en la industria en forma de aplicaciones muy diversas.

A continuación, expondré un estudio reciente el World Robotics 2024 publicado por la IFR o Federación Internacional de Robótica sobre el crecimiento de las instalaciones robóticas experimentado anualmente. El estudio ha sido realizado gracias a los datos proporcionados por casi todos los proveedores de robots industriales del mercado.

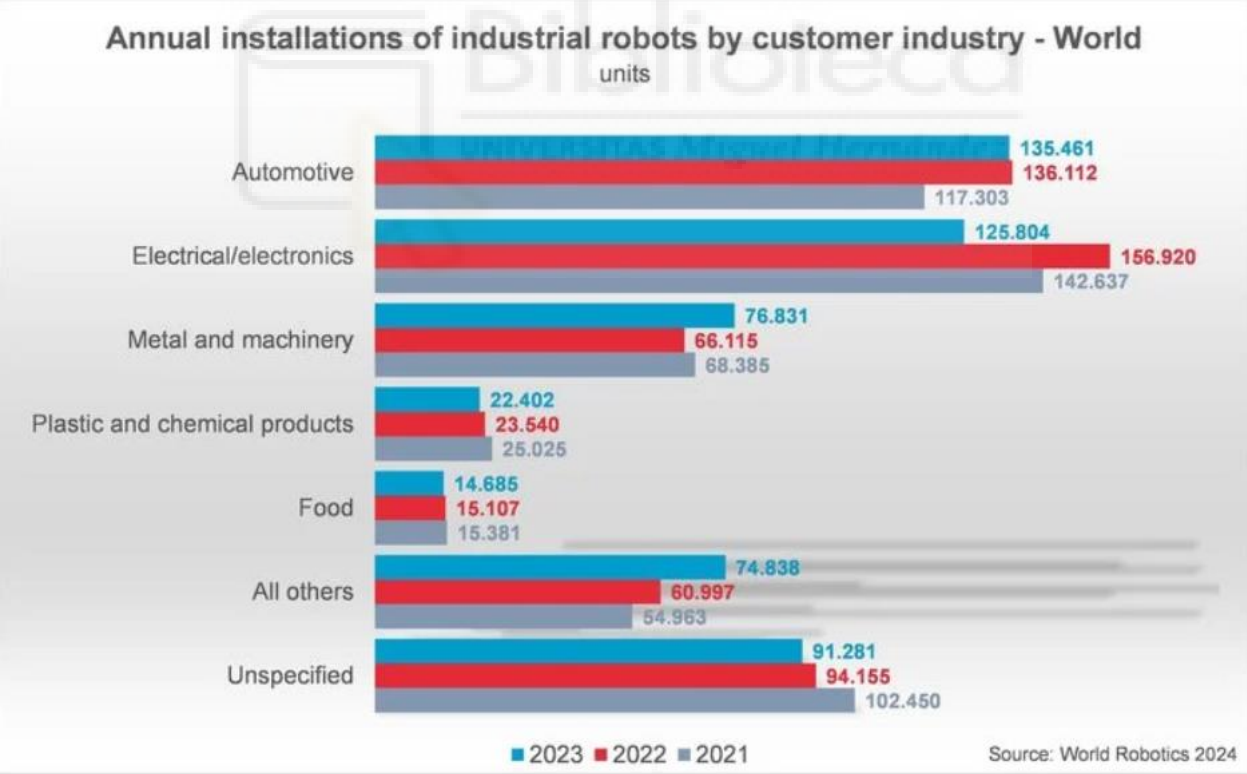


Figura 1. Instalaciones anuales de robots industriales por sector a nivel mundial (2024). Fuente: International Federation of Robotics (IFR).

Podemos observar en la imagen anterior como las ventas (e instalaciones de robots) ha sido creciente en la mayoría de las industrias, incluyendo la automotriz, del metal y maquinaria, etc.

En algunas otras, se produce una ligera caída de ventas, pero se mantienen en valores altos, lo que refleja la gran apuesta global de los diferentes mercados por la robotización industrial [1].

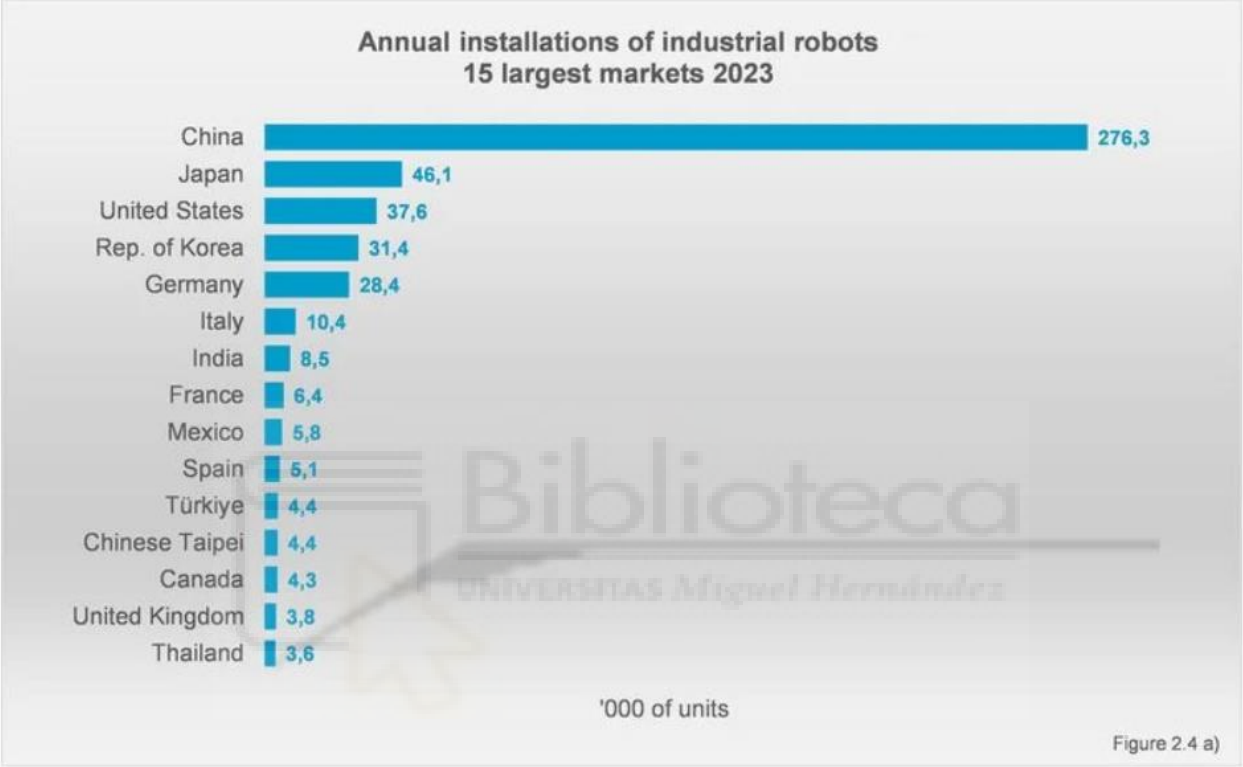


Figura 2. Instalaciones anuales de robots industriales por países (2024). Fuente: International Federation of Robotics (IFR).

En la imagen anterior extraída del mismo estudio podemos observar la posición de los diferentes países del mundo en lo que a instalación de robots industriales se refiere.

Llama la atención que además de los gigantes asiáticos (China, Japón, República de Corea) y de Estados Unidos, varios países europeos (entre ellos España) se encuentran entre los 15 principales consumidores.

Según la Asociación Española de Robótica y Automatización (AER Automation) en 2023 se alcanzaron las 5.000 instalaciones lo que representa un aumento del 48% respecto del año anterior. Este hito se ha producido por segunda vez desde el pico de instalaciones alcanzado en 2018 [2].

Figura 3. Crecimiento de las instalaciones en robótica a niveles de servicio e industrial (2024). Fuente: Asociación Española de Robótica y Automatización (AER).

Podemos observar que el porcentaje de instalaciones correspondiente a robótica industrial es del 48% mientras que el de robótica de servicio es del 17%.

Estos datos son facilitados al IFR para sus estudios anuales, por ejemplo, para el informe "World Robotics 2024", hay que destacar que estos datos son muy veraces y fiables pues se han tomado

según las declaraciones de los fabricantes y miembros de la asociación que suponen un 80% del mercado.

Se expone a continuación un gráfico de sectores que recoge los principales mercados en los que se han distribuido dichas ventas. Un 53% ha ido a parar al sector de la automoción, mientras que los siguientes sectores han sido el del metal/maquinaria y el de la alimentación con un 23% y 11% respectivamente.

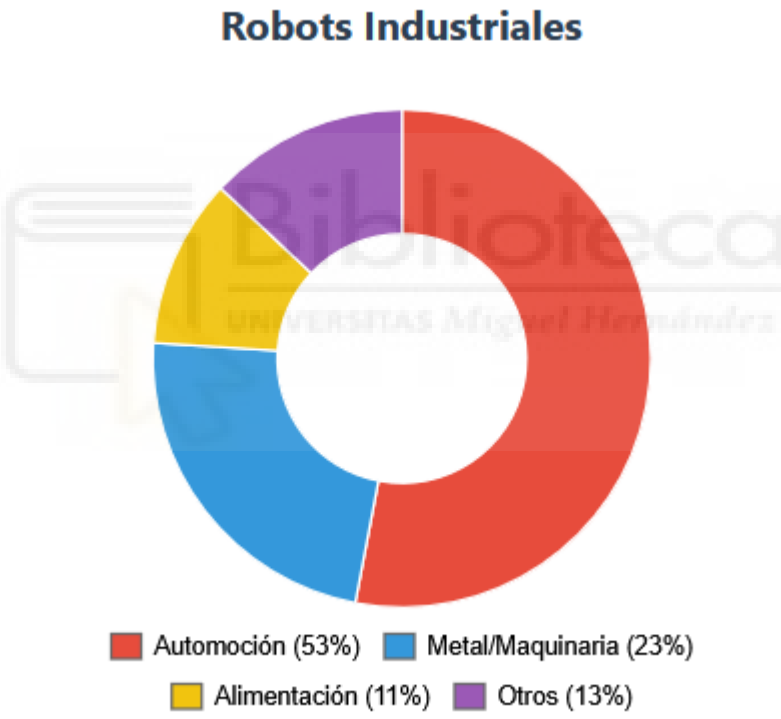


Figura 4. Instalaciones de robótica industrial por sector en España (2024). Fuente: Elaboración propia a partir de datos de la AER.

En cuanto a la robótica de servicio, encontramos que el consumidor principal ha sido el del transporte y logística con un 61% de las ventas, el sector siguiente fue el de Consumer Robots

(robots destinados a uso doméstico o personal) con un 16%, seguido del sector hostelero y del de limpieza.

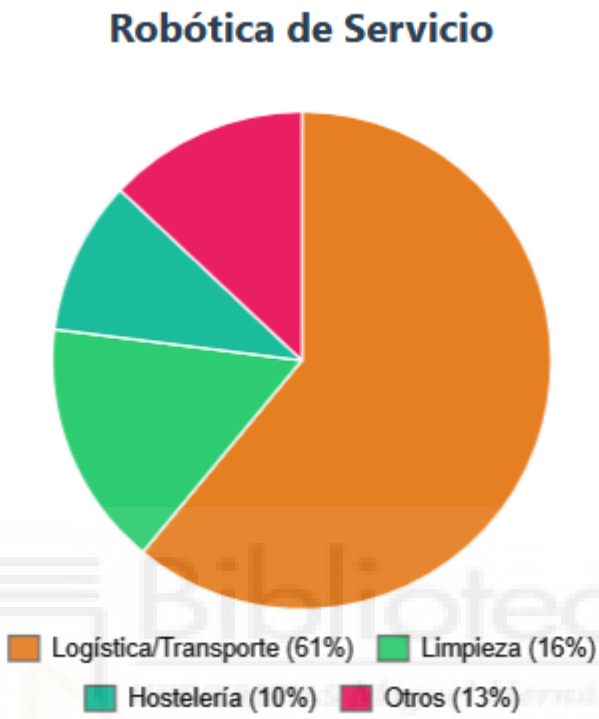


Figura 5. Instalaciones de robótica industrial por sector en España (2024). Fuente: Elaboración propia a partir de datos de la AER

1.1. Motivación y objeto del TFM

El objetivo general del proyecto es diseñar, desarrollar una aplicación real de paletizado por medio del robot UR3e y medios de control y visionado artificial como lo son el PLC S7-1214C o la cámara sensopart V20C.

Para cumplir esta premisa, se definen los siguientes objetivos específicos:

- **Integrar el hardware:** realizar la conexión física y eléctrica de todos los componentes implicados en el proyecto.
- **Configurar la comunicación:** establecer una comunicación real y eficiente entre el robot y los demás componentes por medio de protocolos industriales como Profinet y TCP/IP.
- **Programar el sistema de visión:** calibrar la cámara y desarrollar las tareas necesarias para una detección precisa de las piezas.
- **Programar el robot:** desarrollar la lógica de control del PLC y del robot para que este sea capaz de ejecutar el ciclo de pick and place en nuestra aplicación de paletizado.
- **Validar la solución:** ejecutar una aplicación de paletizado real en un laboratorio o simular dicha aplicación si se carece de los componentes necesarios.

Para cumplir con dichos objetivos, voy a realizar la integración de un robot UR3e con un PLC Siemens 1200 a través de Profinet y con una cámara Sensopart V20C a través de TCP/IP lo que constituirá una propuesta para dar respuesta a una demanda creciente de aplicaciones de robótica colaborativa.

Utilizaremos el robot UR3e, el PLC S7-1200, o la cámara Sensopart V20C porque son algunos de los materiales a los que he tenido acceso a través de mi experiencia como docente y alumno del

IES Antonio José Cavanilles, no obstante, vamos a estudiar sus hojas de características para comprobar la idoneidad de su uso.

El proyecto consiste en el reconocimiento de la posición de unos bloques de material plástico y el proceso de pick&place y paletizado de los mismos.

2. Antecedentes y Estado del Arte

Este proyecto surge como necesidad de representación de una aplicación real de paletizado y es que en la industria son numerosas las aplicaciones de la visión artificial en conjunto con los robots colaborativos.

Históricamente los robots industriales han operado en celdas aisladas de seguridad. Desde los primeros robots como el Unimate desarrollados para operar en fábricas o cadenas de montaje como la de General Motors hasta hace pocos años (alrededor de los años 90) la intervención del operario ha estado separada de la operación del robot. Actualmente y con la llegada de la robótica colaborativa (los cobots) esto es historia o está reservado a robots más complejos destinados a tareas peligrosas dentro de fábricas.

Los actuales cobots han dado la vuelta a esa situación, actualmente disponen de suficientes sensores y mecanismos de seguridad como para que su operación cercana al humano sea posible. Universal Robots es una marca líder en este campo.

Las empresas líderes en robótica como FANUC, ABB o KUKA han presenciado cómo los creadores de Universal Robots, fundada en Dinamarca en 2005 han conseguido irrumpir en esta industria con modelos muy accesibles a las pequeñas y medianas empresas. Su éxito se debió al lanzamiento del UR5, el primer cobot viable en el mundo. Debemos destacar de UR su enfoque por las interfaces de programación intuitivas gracias a su software Polyscope y la calidad y seguridad presente en todos sus productos.

En cuanto a la visión artificial se refiere, la empresa Sensopart de origen alemán fue fundada en 1994, inicialmente dedicada a la sensórica, comenzaron con sensores fotoeléctricos y de proximidad, pero pronto se enfocaron en las herramientas de visión artificial. Un ejemplo de ello son las cámaras de la serie VISOR, entre ellas la V20C.

Debemos destacar no solo la calidad de estas cámaras, sino también su interconectividad con otros equipos y robots. Por ejemplo, mediante las URCaps de las que hablaremos más adelante en este proyecto.

2.1. Robótica Industrial

En el informe World Robotics 2021, se determina que la clasificación en robot industrial o robot de servicio se realiza en función de su aplicación prevista. Los robots industriales son robots para su uso en aplicaciones de automatización industrial, mientras que un robot de servicio realiza tareas útiles para las personas o los equipos, excluyendo las aplicaciones de automatización industrial [3].

Para hacernos una idea mejor de que son estos robots y sus principales aplicaciones listaré los principales tipos de robots industriales

Podemos encontrarnos 7 tipos principales de robots industriales.

- Robot articulado
- Robot cartesiano
- Robot cilíndrico
- Robot polar (esférico)
- Robot Scara
- Robot Delta
- Robot colaborativo (Cobot)

2.1.2. Robots colaborativos

Los robots colaborativos son la evolución de los robots tradicionales, estos como hemos mencionado anteriormente destacan por su seguridad y capacidad para operar en el mismo espacio que los operarios humanos. Esto es gracias a la evolución de los sensores de fuerza integrados en estos robots, se caracterizan por su facilidad de programación frente a sus antepasados. A menudo, esta programación se realiza por interfaces gráficas o por guiado manual como puede ser el caso de este UR3e.

Además del fabricante UR, existen otros grandes fabricantes como FANUC, un gigante de la robótica tradicional que ofrece sus series CR y CRX al mundo de la robótica colaborativa o KUKA, otro pionero con su LBR iiwa, un robot de 7 ejes muy sensible y preciso. También ABB o Yaskawa, son referentes con modelos como el ABB YuMi o el MotoMan HC10.

Sin embargo, nos hemos decantado por la marca UR por su accesibilidad económica y características de seguridad e integración.

Figura 6. Fotografía del robot "Yumi" de ABB. Fuente: Leitbetriebe Austria, 2020.

En la imagen anterior podemos observar a un robot ABB Yumi uno de los cobots más representativos de la capacidad de estos de compartir escenario de trabajo sin limitaciones físicas (vallas) cuyo diseño y tamaño recuerdan al de un operador humano [4].

Este modelo es además un ejemplo de versatilidad pues ha demostrado ser un operario altamente eficiente en las más diversas industrias desde montando enchufes eléctricos a manipulando muestras de tejido, huesos o fluidos en las industrias sanitaria y farmacéutica.

2.2. Visión artificial

Figura 7: Imagen de un robot colaborativo y la cámara Sensopart V20C. Fuente: SensoPart Industriesensorik GmbH, 2021

En diferentes procesos industriales se utilizan diferentes tipos de visión artificial, son muy habituales en tareas de inspección, ya que son más rápidas y consistentes que los inspectores humanos, por no hablar de que no necesitan de períodos de descanso.

2.2.1. Aplicaciones de Visión Artificial

Algunas funcionalidades típicas se recogen en la siguiente imagen.

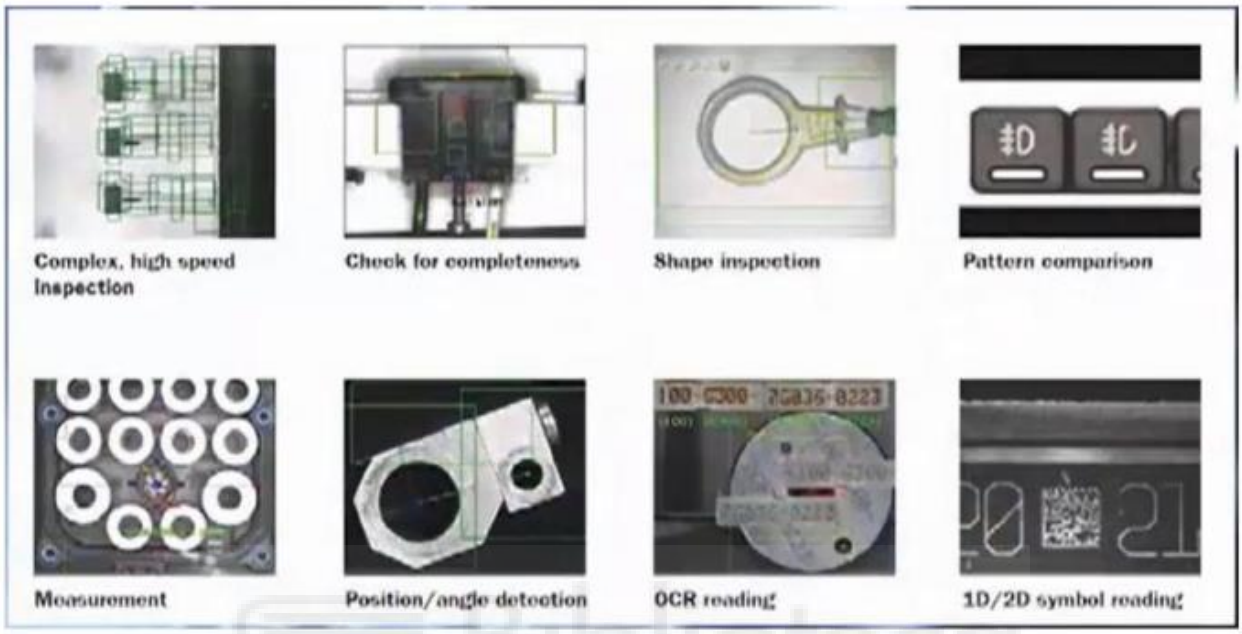


Figura 8. Diagrama de tareas comunes de inspección por visión artificial. Fuente: Industrial Vision Systems (IVS)

Podemos listar las siguientes:

- Inspección visual avanzada a altas velocidades.
- Comprobaciones de ensamblado correcto.
- Inspecciones de forma.
- Comparación de patrones.
- Medida.
- Detección de posición o modificación de ángulo.
- Lectura OCR.
- Decodificación de símbolos en 1D/2D.

Siendo las más típicas aquellas relacionadas con la medición, contaje, ubicación y decodificación.

Supongamos que las piezas que vamos a detectar, coger y desplazar no pueden ser situadas de cualquier manera, sino que requieren de un posicionamiento exacto y repetido que tan solo un complicado sistema electroneumático o una persona pudiera realizar pero disponemos de un robot UR3e que ocupa menos espacio que el anterior sistema electroneumático, trabaja sin descanso y es capaz de detectar detalles en la pieza que un observador humano no detectaría, pongamos por ejemplo pequeños defectos de tolerancias, medidas o códigos grabados muy difícilmente legibles.

2.3. PLCs

Un PLC es un tipo específico de ordenador industrial diseñado para soportar las adversidades típicas de los entornos industriales como son temperaturas extremas, polvo, vibraciones, interferencias electromagnéticas, etc.

Resumidamente un PLC o Programable Logic Controller es un autómata programable que puede realizar un programa cíclicamente en fracciones de segundo. Su función es leer es estado de sensores y otros dispositivos de entrada, ejecutar una lógica de control y gobernar actuadores por medio de sus salidas.

Es decir que entre las funciones principales de un PLC se encuentran:

- La lectura de entradas, habitualmente sensores, interruptores, pulsadores, etc.
- El procesamiento de código para ejecutar la lógica de un programa.
- La ejecución de las salidas, habitualmente actuadores (motores, válvulas,etc).

El funcionamiento real del PLC, sin embargo, es más complejo y detallado, y para conocerlo se hace necesario definir el concepto de ciclo de Scan.

El ciclo de scan de un PLC, consiste básicamente en el ciclo que el PLC repite desde que se produce su ARRANQUE y se pas a modo RUN o modo operativo, este consiste principalmente en los siguientes puntos.

Durante el ARRANQUE:

- Borrar las áreas de memoria.
- Iniciar las salidas con el último valor cargado.
- Ejecutar los OBs de arranque (en el caso de los Siemens)
- Copiar el estado de las entradas físicas a la memoría.

- Habilitar los procesos de alarma y las memorias.

Durante el modo RUN:

- Actualizar las salidas físicas escribiendo en ellas el contenido de la memoria.
- Leer el último estado de las entradas físicas y copiarlo a la memoria.
- Ejecutar el programa del usuario.
- Realizar un autodiagnóstico.
- Procesar las alarmas y comunicaciones configuradas.

Otro concepto importante que debemos comentar es el tiempo de vigilancia de ciclo, también conocido como "Watchdog".

Este es un mecanismo interno del PLC que cambia el estado operativo del mismo (de RUN a STOP) cuando el tiempo de ciclo supera unos valores predeterminados o configurados por el usuario.

De este modo, en aplicaciones muy sensibles el usuario puede configurar un tiempo de Watchdog de pocos cientos de milisegundos (por ejemplo, 50 milisegundos, si la CPU lo permite) mientras que en aplicaciones no tan sensibles es un parámetro que por defecto se establece en 2 o 3 décimas de segundo, es decir, 200 o 300 ms.

En el mercado existen variedad de fabricantes relevantes como Siemens, Omron, Schneider o Rockwell Automation (Allen-Bradley).

Por un lado, Omron cuenta con los modelos de la serie NX/NJ y con la plataforma de integración Sysmac, Schneider con modelos como el M221 o el M241 y su plataforma EcoStruxure o Rockwell con sus familias ControlLogix y CompactLogix, muy valoradas por su escalabilidad.

Por otro lado, Siemens es un fabricante de PLCs muy consolidado en el panorama de la automatización a nivel mundial, su alta consolidación y extensión de uso permiten facil accesibilidad a cantidad de recursos, materiales (documentación técnica) y soporte comunitario.

Dentro del fabricante Siemens podemos encontrarnos modelos tan básicos como los LOGO! hasta PLCs de mayor envergadura y capacidad como los S7-1200, S7-1500 y otros modelos más específicos.

3. Descripción del sistema

3.1. PLC S7-1200 1214C AC/DC/Rly

El S7 1214C AC/DC/Rly es un PLC muy versátil que cuenta con 14 entradas digitales integradas a 24V DC, 10 salidas digitales a relés de 2A, 2 entradas analógicas de 0 a 10 V en DC, su alimentación es posible en 230 V en AC, y tiene una memoria de usuario de 75 KB.

El tipo de comunicación es a través de Profinet y TCP/IP lo que nos permite conectar este dispositivo al robot UR3e y a la cámara V20C respectivamente.

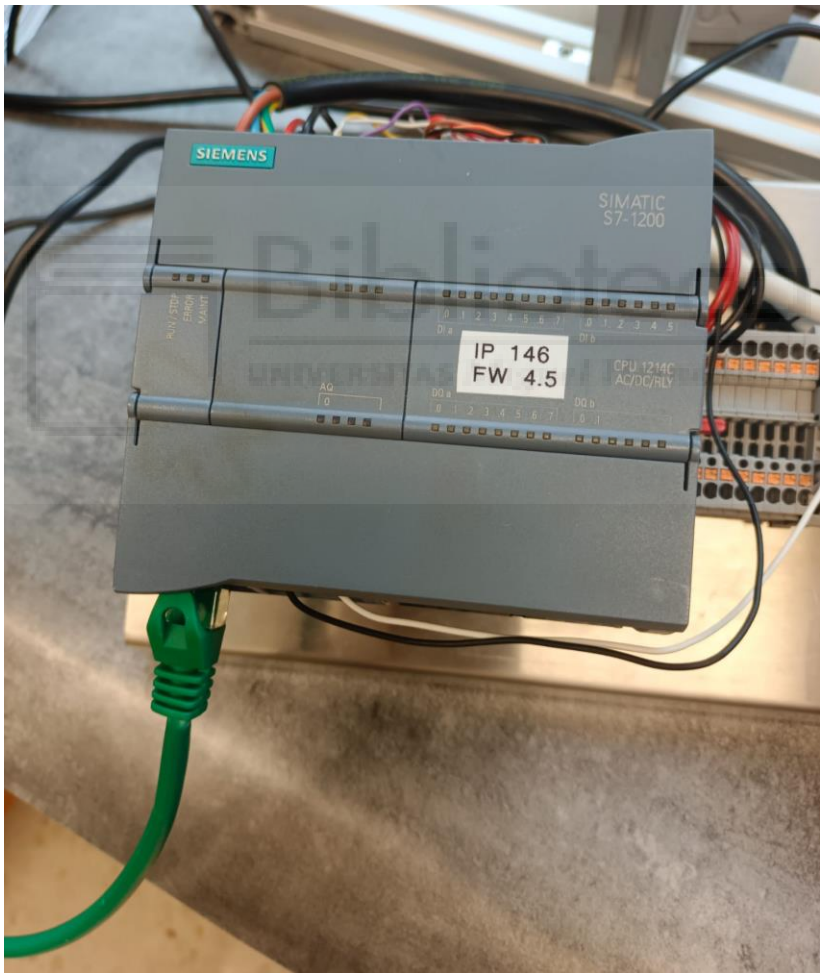


Figura 9. Imagen de nuestro PLC S7-1214C. Fuente: Elaboración propia

El software de programación TIA PORTAL es uno de los más ricos y estables para aplicaciones

de complejidad y control en tiempo real como la que vamos a desarrollar en este documento. Además, ofrece un entorno de programación claro y sencillo que es ideal para este proyecto. Omron en su gama NX también sería una buena opción para el proyecto, mientras que Schneider con sus PLCs M221 o M241 me parece algo limitado para su implementación.

Otro factor es la disponibilidad física de los equipos, contamos con más PLCs de la gama S7 1200 de Siemens que de la gama NX o CP de Omron, por lo que nos decantamos por el primero.

3.2. Cámara Sensopart V20C

Figura 10. Imagen de la cámara Sensopart V20C. Fuente: SensoPart Industriesensorik GmbH, 2021

La cámara Sensopart se trata de un modelo más reciente que otros modelos a los que he tenido acceso como otras cámaras de cognex y destaca por la facilidad de su conexión como dispositivo URCap a nuestro robot UR3e ya que admite una amplia gama de protocolos de comunicación: TCP/IP, Profinet, EtherNet/IP, sFTP, SMB, FTP, Puerto de servicio.

Figura 11: Imagen de nuestra cámara SensoPart V20C. Fuente: Elaboración propia.

Entre sus principales características se encuentran:

Tiene un tamaño reducido de 85x45x45 mm, sus materiales son de calidad (metal y vidrio), sus conectores son del tipo M12x1 y presenta un índice de protección IP65. Puede trabajar en rangos de temperatura desde los 0 a 50 ºC.

Utiliza un chip CMOS 1/ 2.9" (1440X1080 px), equipa láser rojo de clase 1, luz LED ultravioleta de 365 nm y un campo de visión estrecho ajustable desde los 100 a los 10.000 mm.

En cuanto a sus funciones de detección, destaca por su capacidad para gestionar hasta 255 tareas o trabajos de inspección independientes, esto nos permite realizar múltiples detecciones sobre la misma pieza, imaginemos que queremos detectar su forma, color y leer un código de barras al mismo tiempo.

Esto nos lleva al tipo de detectores que implementa comenzando por los detectores de tipo OCR que permiten la lectura de códigos de barras y de datos, los detectores de contorno, BLOB e incluso la posibilidad de medición de aristas.

También presenta características muy interesantes en cuanto a la detección de colores y múltiples herramientas para evaluar parámetros como el brillo, el contraste o la escala de grises.
Implementa funciones de seguimiento de la posición, esto quiere decir que incluso si las piezas no llegan siempre en la misma posición la cámara utiliza diferentes herramientas como el análisis de contornos, detección de bordes, etc. para reconocer la pieza.
Por último cabe destacar las opciones de apunte automático y puntería, esto es que dispone de un accionamiento de enfoque motorizado lo que facilita enormemente la tarea de configurar el campo de visión sobre el que vamos a trabajar.

Tenemos que recordar las diferentes posibilidades de conexión a controladores externos como es el caso de nuestro robot, encontramos las siguientes opciones: TCP/IP, Profinet y Ethernet/IP.
La transferencia de nuestras imágenes o vídeo en tiempo real se realizarán por medio de los siguientes protocolos: SFTP (Secure File Transfer Protocol), FTP (File Transfer Protocol) o SMB (Server Message Block).

Es gracias a estos protocolos y a las posibilidades de comunicación TCP/IP a velocidades de 1000 Mbit lo que permiten al robot ajustar sus cinemáticas de manera rápida y eficaz por ejemplo con la recepción de los datos de posición de una pieza.

3.3. Robot UR3e

Si lo comparamos con sus hermanos mayores nos daremos cuenta de que sus dimensiones, capacidad de carga y alcances son más que suficientes para cumplir con el propósito de este proyecto.

Figura 12. Robots colaborativos de UR. Fuente: Universal Robots A/S, 2018

Este robot está formado por los siguientes elementos: base, junta de la base, junta del hombro, junta de codo, junta de muñeca 1, junta de muñeca 2, junta de muñeca 3 y brida de la herramienta. Como así lo demuestra la siguiente imagen. Esto quiere decir que el robot UR3e es un robot de 6 grados de libertad o Degrees of Freedom (DOF).

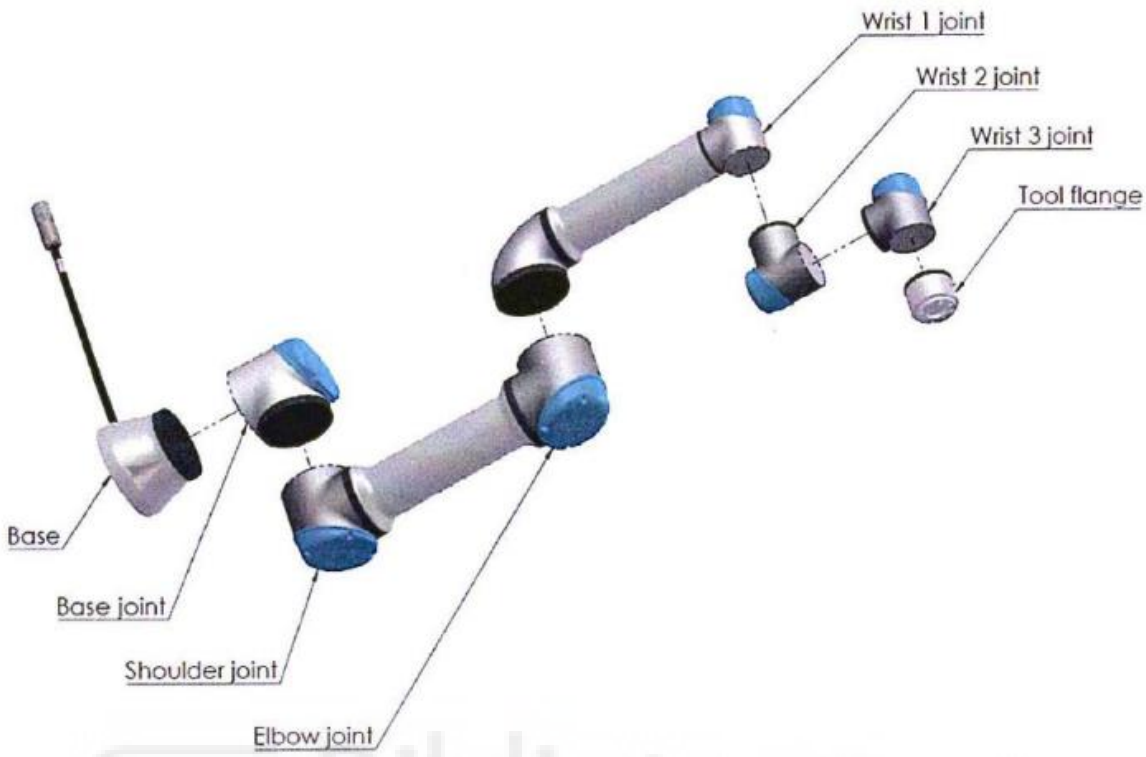


Figura 13. Diferentes partes y articulaciones del robot UR3e. Fuente: Universal Robots A/S, Manual de e-Series, 2018.

Los principales puertos de comunicaciones con los que cuenta el robot son los siguientes, como podemos apreciar la conexión principal es por puerto Ethernet (base y conectores RJ45), este se encuentra en la base del robot, junto a los puertos USB 2.0 y 3.0.

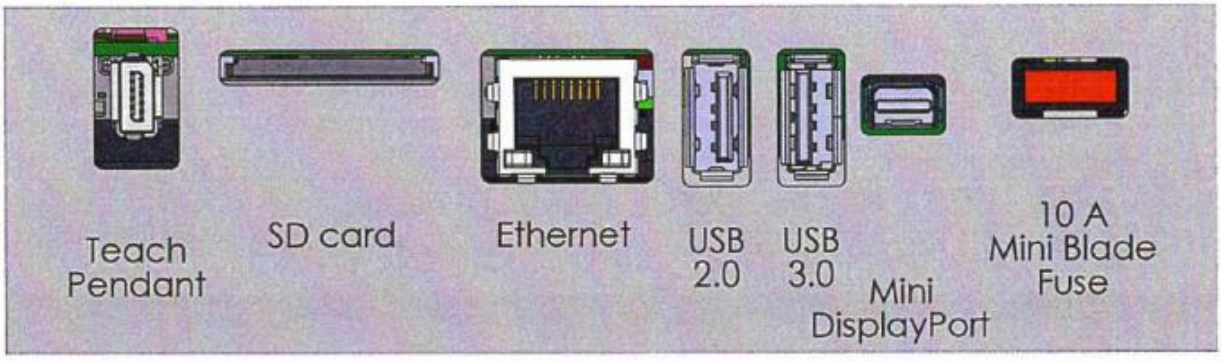


Figura 14. Puertos de comunicaciones y ranura SD del robot UR3e. Fuente: Universal Robots A/S, Manual de e-Series, 2018.

El RJ45 es el conector estándar para la comunicación a través de redes industriales (como Ethernet/IP) y otros protocolos como Profinet RT-Communicator. Usaremos este puerto para la comunicación con el controlador PLC de SIEMENS, el S7-1200 y con la cámara de visión artificial, SENSOPART.

Ahora veamos cómo conectamos las E/S

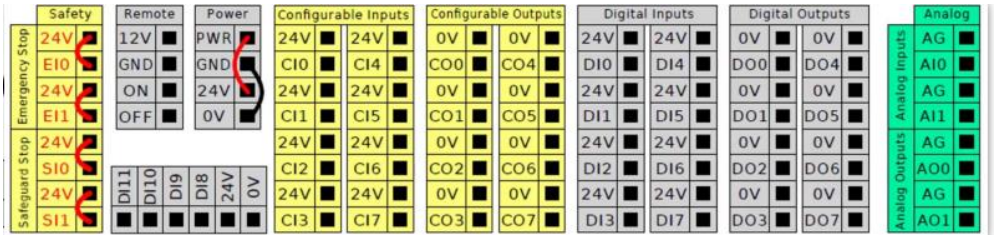


Figura 15. Borneros de entradas y salidas en UR3e. Fuente: Universal Robots A/S, Manual de e-Series, 2018.

Vamos de izquierda a derecha, EI0 y EI1 deben estar siempre alimentadas a 24V y mediante un contacto NC que a través de un pulsador pueda ser rápidamente abierto, pues constituyen la parada de emergencia, SI0 y SI1 también son las paradas de protección. Lo cual quiere decir que en el momento que se corte la alimentación a estas se parará el robot.

A continuación, vemos los borneros de *remote* y *power,* el bornero de potencia o de alimentación ya está cableado a la fuente de alimentación interna (bornes de 24V y 0V), estos a su vez, están conectados a una fuente de alimentación externa.

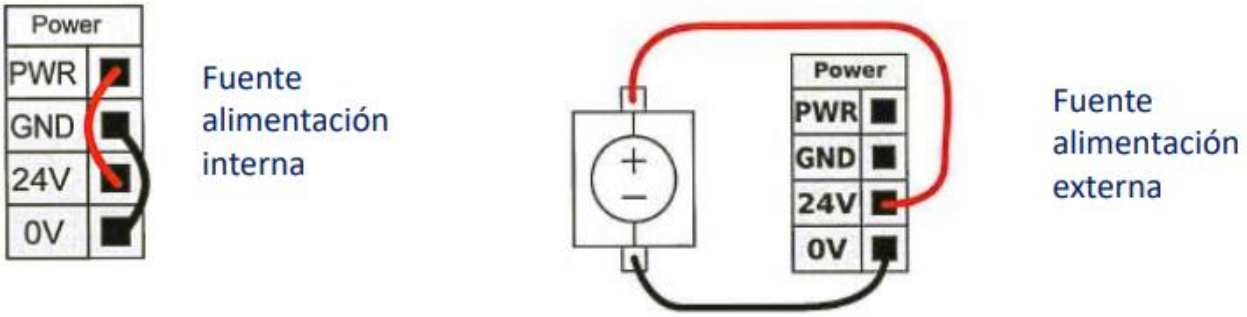


Figura 16. Detalle del bornero de "Power" del UR3e. Fuente: Universal Robots A/S, Manual de e-Series, 2018.

El motivo de no conectar directamente el robot a una fuente de alimentación externa es la sensibilidad del robot a las fluctuaciones de tensión e intensidad, la fuente de alimentación interna nos asegura una alimentación de entre 23 y 25 V y de entre 0 y 2 A, mientras que la entrada externa de 24V admite fluctuaciones de entre 20 a 29 V y de entre 0 a 6 A.

¿Cómo es la conexión de los botones de parada de emergencia?

Tal y como se describe en la siguiente imágen y cómo se ha descrito anteriormente siempre conectados a un pulsador del tipo NC, cuya pulsación interrumpa el paso de tensión al instante.

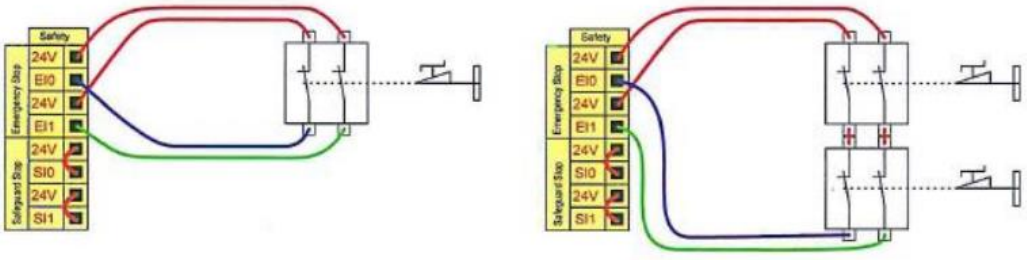


Figura 17. Detalle de los borneros de "Emergency Stop" y "Safeguard Stop" del UR3e. Fuente: Universal Robots A/S, Manual de e-Series, 2018.

La imagen de la izquierda muestra la conexión típica de una seta de emergencia. Se utiliza un sistema de doble canal para asegurar la redundancia del sistema de seguridad, esto permite al controlador vigilar si se ha producido el corte accidental de alguno de los hilos y diferenciarlo de una parada intencional.

A la derecha se muestra un sistema compuesto por más de una botonera de emergencia, repartidas por diferentes puntos de la instalación.

Otra idea interesante como es la parada de seguridad con reanudación automática, por ejemplo, un interruptor de puerta con el que se detiene el robot cuando se abre una puerta de seguridad. Del mismo modo, usamos un pulsador NC que al abrirse la puerta sea accionado y corte el suministro de tensión.

El resto de los bornes son de *Configurable Inputs* o Entradas configurables (CI0 a CI7), *Configurable Outputs* o Salidas configurables (CO0 a CO7), *Digital Inputs* (Entradas digitales; DI0 a DI7), *Digital Outputs* (Salidas digitales; DO0 a DO7) y *Analog Inputs* (Entradas analógicas; AI0 y AI1) y *Analog Outputs* (Salidas analógicas, AO1, AO1).

Las entradas digitales adicionales (DI8 a DI11) son entradas digitales de codificador para el seguimiento del transportador.

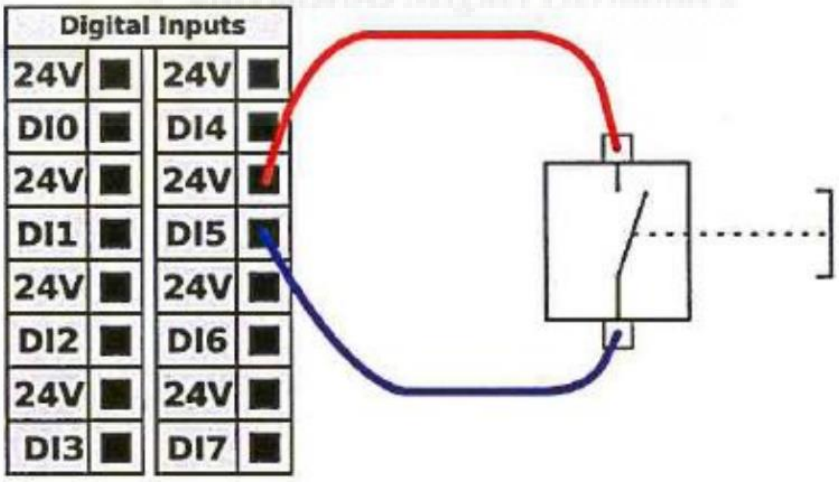


Figura 18. Detalle del bornero de "Digital Inputs" del UR3e. Fuente: Universal Robots A/S, Manual de e-Series, 2018.

Este ejemplo muestra cómo conectar un botón sencillo del tipo "Normally open" (NO) a entradas digitales.

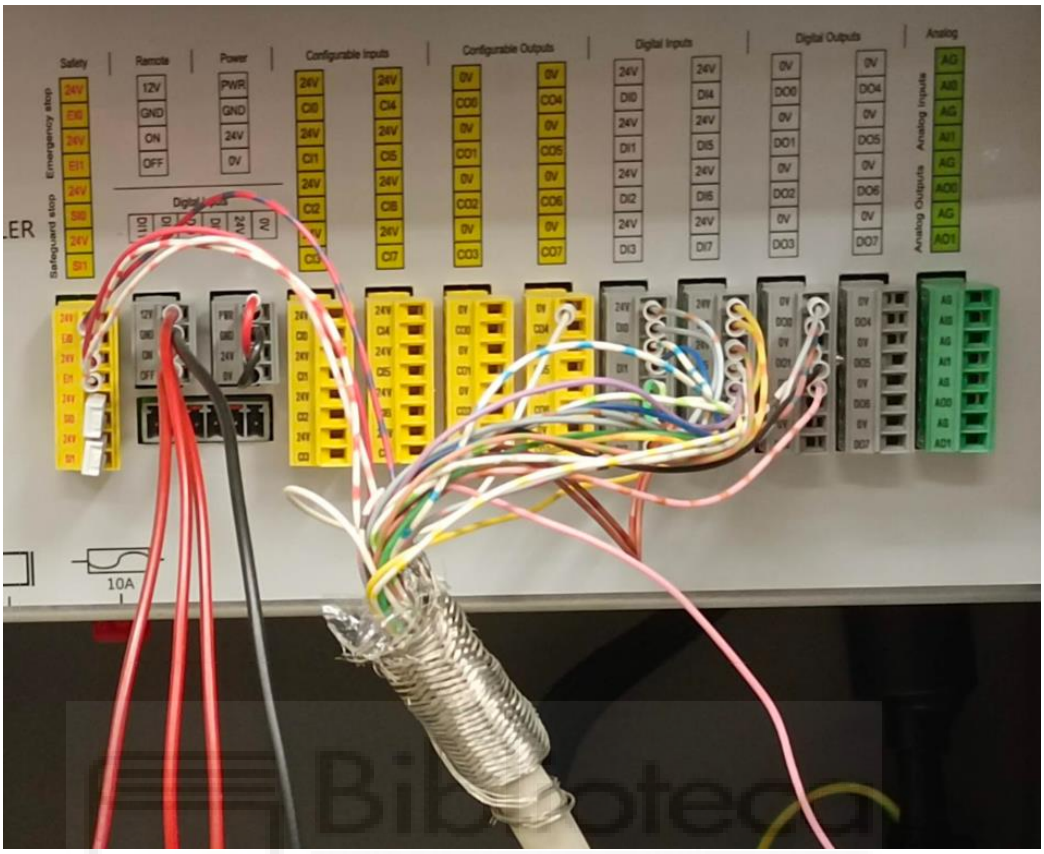


Figura 19. Detalle del bornero real de nuestro UR3e configurado para el proyecto. Fuente: Universal Robots A/S, Manual de e-Series, 2018.

Y aquí se muestra cómo sería un ejemplo de conexión real de nuestro Robot UR3e con una botonera externa (entradas y salidas digitales) y el PLC (Cables de encendido y apagado remoto).

3.3.1. Teach Pendant

El teach pendant es un elemento de interacción con el robot que se podría asemejar mucho con un HMI tradicional con la peculiaridad de que está diseñado para ser portable y nos permite un fácil acceso al manejo y supervisión del robot.

Figura 20. Imagen de la teach pendant del UR3e. Fuente: Universal Robots A/S, Manual de e-Series, 2018.

En cuanto a sus especificaciones físicas merece la pena comentar que cuenta con un botón de parado de emergencia (tipo seta) y también con una resistencia frente a polvo, salpicaduras de agua y suciedad IP54. Estas características son fundamentales en entornos industriales e incluso educativos.

También cuenta con un botón de movimiento libre ("freedrive") que permite el movimiento del robot guiado por la fuerza física del usuario, esto es muy útil cuando queremos mover el robot a un punto específico sin hacer uso de sus controles o de marcar una posición en pantalla.

Además su cable de varios metros de longitud permite el movimiento con el teach pendant alrededor de las instalaciones y del robot con facilidad.

El software que implementa esta consola portátil es el ya mencionado Polyscope.

Cuando accedemos a él, encontramos que se encuentra seccionado principalmente en un área de encabezado, pantalla central y una zona de pie de página.

En el encabezado o área superior del software encontramos las pestañas de “Ejecución de programas”, “Inicialización del robot”, “Instalación”, “Registro de errores”, “E/S” entre otros.

En la pantalla central se encuentran los principales elementos para la gestión en tiempo real del funcionamiento del robot y demás campos interactivos y es en el área inferior o pie de página dónde cobran importancia los controles de inicio o pausa, junto al selector de velocidad que se nos presenta en forma de deslizador.

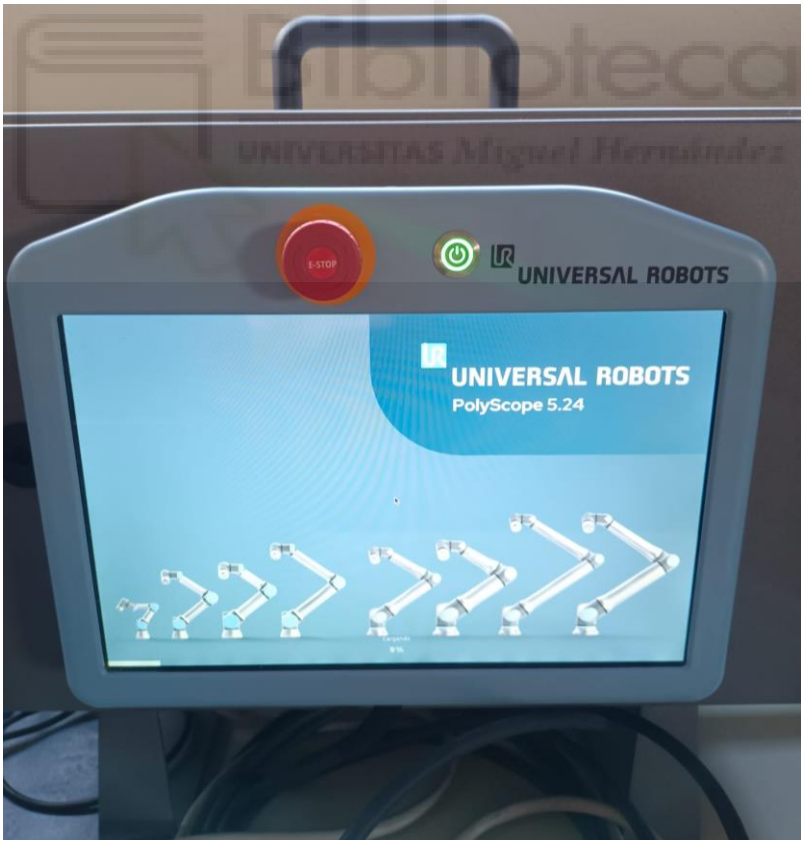


Figura 21 Detalle de la teach pendant de nuestro UR3e. Fuente: Elaboración propia, 2026.

Merecen una mención especial las posibilidades de configuración y ajustes de seguridad dentro de la pestaña “Instalación”, entre los que encontramos “límites del robot”, “límites de junta”, “planos”, “posición de la herramienta”, etc.

Un uso práctico de los planos de seguridad es la definición de planos para que el robot tenga presentes que límites de espacio no debe superar para no dañar por ejemplo otros equipos cercanos o al usuario.

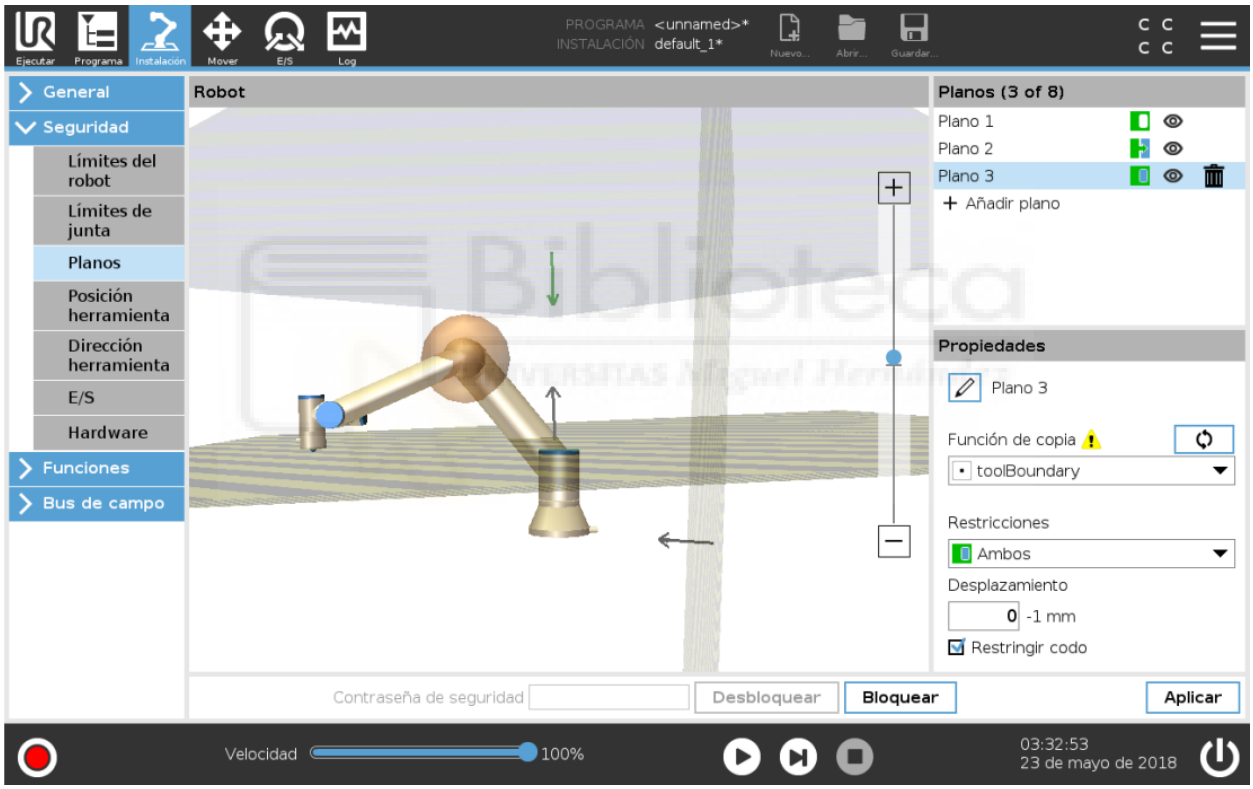


Figura 22. Visualización de planos de seguridad en Polyscope. Fuente: Universal Robots A/S, Manual de e-Series, 2018.

En nuestro caso, podríamos definir estos planos para salvaguardar elementos delicados cómo la cámara V20C de ser golpeada por nuestro robot accidentalmente.

Otro aspecto de la seguridad que podemos configurar es la protección por contraseña de determinadas configuraciones y opciones.

Estas pantallas y configuraciones se mostrarán más adelante en los diferentes pasos de programación del robot objeto de este trabajo.

4. Fundamento teórico

4.1 Cinemática del robot UR3e

La cinemática en robótica se define como el estudio de las diferentes posiciones, velocidades y aceleraciones que intervienen en el movimiento de los diferentes eslabones de un robot. Para su estudio se utilizan sistemas de referencia y transformaciones homogéneas y se utiliza la convención de Denavit-Hartenberg.

En el caso de nuestro robot UR3e habitualmente queremos determinar la posición y orientación de la herramienta o TCP en función de la posición y orientación de sus articulaciones o todo lo contrario, a partir de un punto de "destino" que el sistema realice todos los cálculos necesarios para posicionar y rotar sus articulaciones de manera que este punto sea alcanzable por el robot.

Ahora veremos que el robot UR3e implementa soluciones de cinemática inversa precisas y deterministas gracias a la disposición de sus ejes. Esto es fundamental para que el robot tenga una respuesta rápida en aplicaciones cómo la que es objeto de este trabajo pues el robot debe ser rápido y preciso en el cálculo de la posición y orientación de la herramienta para así situarse sobre la pieza a paletizar una vez señalada su posición por la cámara.

4.1.1. Parámetros Denavit-Hartenberg

Son los siguientes:

d_i (profundidad) representa la distancia desde el eje Z anterior hasta la normal común.

a_i (longitud) representa la longitud de la normal común.

α_i (torsión) representa el ángulo entre los ejes Z.

θ_i (ángulo) representa el giro del motor.

La imagen siguiente los representa en color rojo de una manera simplificada.

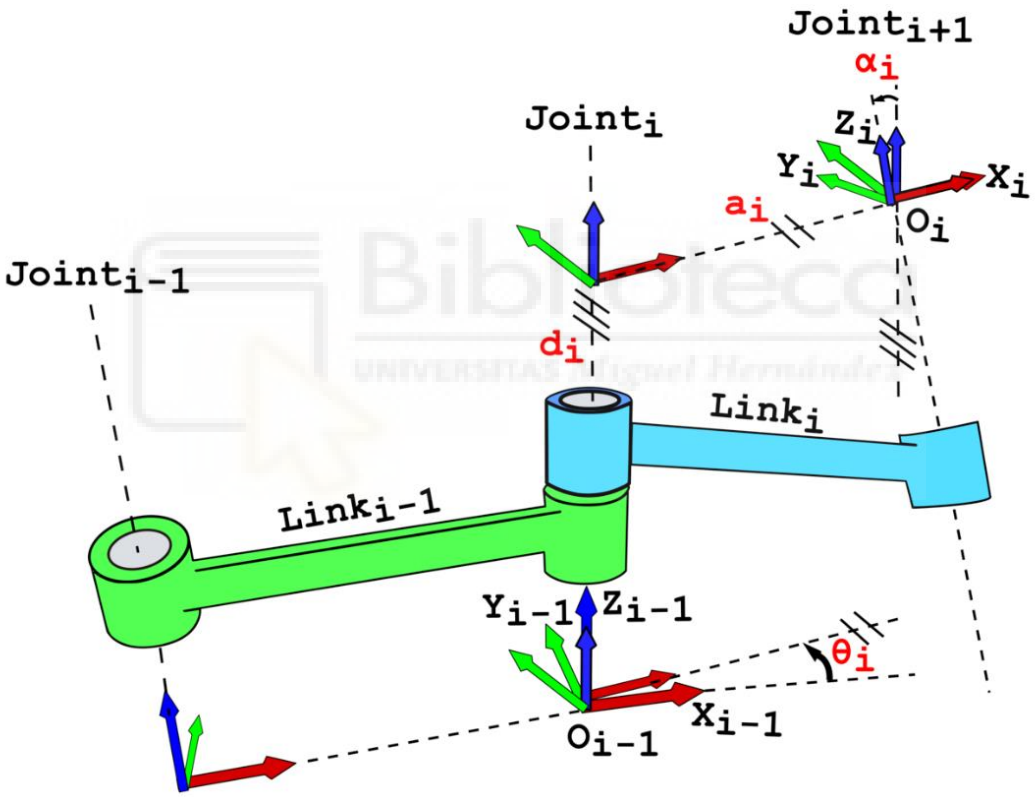


Figura 23. Esquema representativo de los parámetros de DH. Fuente: Wikimedia Foundation, 2024

Lo interesante de este mecanismo de representación es la facilidad de conversión del marco de referencia Oi –1 (Xi –1,Yi –1, Zi –1) a Oi (Xi, Yi, Zi) [5].

Según la documentación oficial de UR, la tabla de parámetros DH es la siguiente. Debemos tener en cuenta que para este robot los valores de inercia son despreciables en este robot por su reducido tamaño.

La columna de inercia se ha obviado para mayor simplificación de los datos [6].

UR3e						
Kinematics	a [m]	d [m]	alpha [rad]	**Dynamics**	Mass [kg]	Center of Mass [m]
Joint 1	0	0.15185	π/2	Link 1	1.98	[0, -0.02, 0]
Joint 2	-0.24355	0	0	Link 2	3.4445	[0.13, 0, 0.1157]
Joint 3	-0.2132	0	0	Link 3	1.437	[0.05, 0, 0.0238]
Joint 4	0	0.13105	π/2	Link 4	0.871	[0, 0, 0.01]
Joint 5	0	0.08535	-π/2	Link 5	0.805	[0, 0, 0.01]
Joint 6	0	0.0921	0	Link 6	0.261	[0, 0, -0.02]

Tabla 1. Valores de los parámetros de DH para nuestro robot UR3e. Fuente: Elaboración propia a partir de datos de UR Support.

4.2 Sistemas de referencia

Para que el robot UR3e se mueva de manera precisa, la controladora del robot es la encargada de realizar los cálculos partiendo de una base cartesiana (X,Y,Z) y una orientación específica de ejes. A la combinación de estos se le conoce como “frames” o marcos de referencia.

Encontramos los siguientes marcos de referencia principales:

- Marco de la base (Base frame)

La ubicación del marco de la base es el centro geométrico de la base del robot y define el punto (0,0,0) o de partida estático e incambiable de nuestro robot.

La orientación de los ejes es simple el eje Z apunta hacia arriba de manera perpendicular a la base del robot y los ejes X e Y se encuentran sobre el plano de la mesa donde se sitúa la base del robot.

- PCH o Punto Central de la Herramienta

La ubicación de este marco de referencia o punto es inicialmente el centro de la brida de salida de la herramienta. Pero tras la instalación de la herramienta, en este caso una pinza, se debe reconfigurar este punto al nuevo límite de nuestra herramienta.

Este es un paso crítico, pues de lo contrario el TCP quedará mal calibrado lo que podría acarrear colisiones y accidentes con la herramienta.

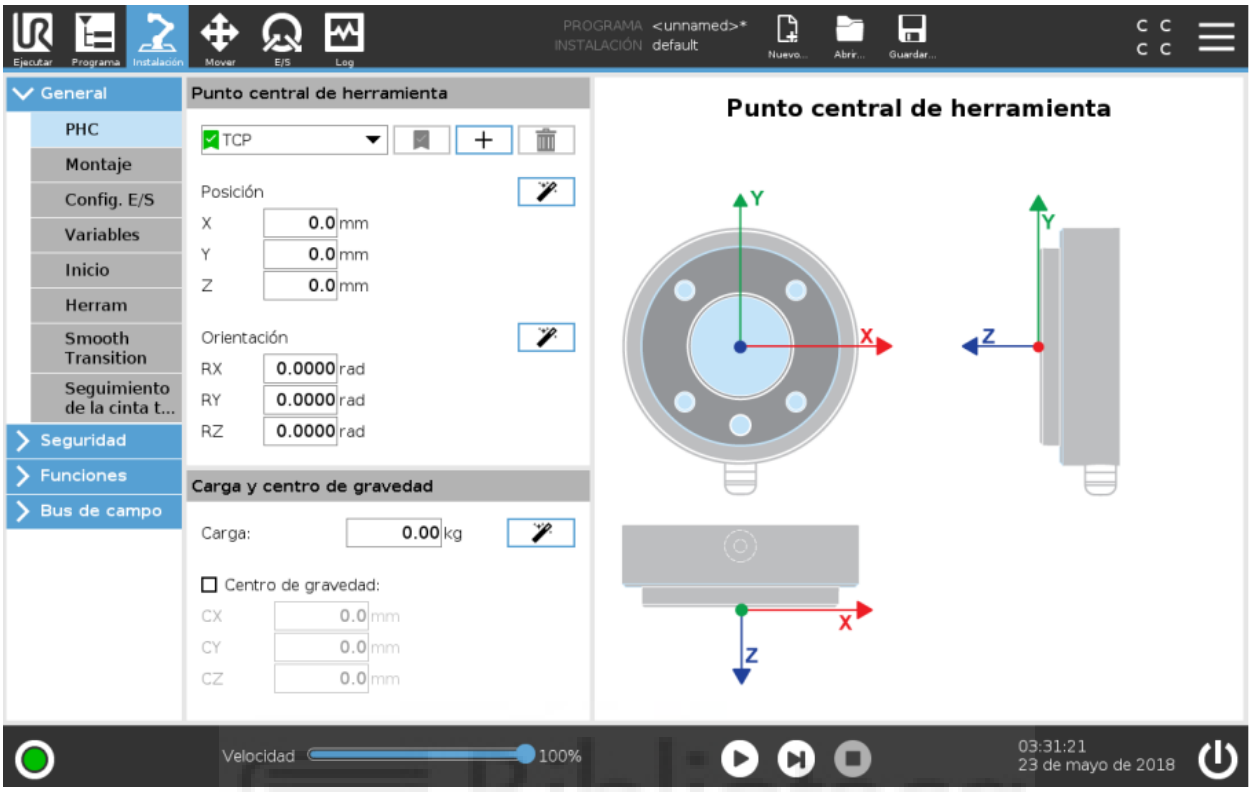


Figura 24. Definición de parámetros del PCH en polyscope. Fuente: Universal Robots A/S, Manual de e-Series, 2018.

Como podemos apreciar las coordenadas (x,y,z) definen la posición del PCH de la herramienta mientras que (Rx,Ry,Rz) definen la orientación de la herramienta con respecto al centro de la brida de salida.

En esta misma pestaña podemos definir la carga a sujetar por nuestro robot y su centro de gravedad, o podemos hacer uso de la función de estimación de carga.

4.3. Transformaciones geométricas

Haciendo uso de la matriz de Denavit-Hartenberg, cada enlace se puede describir mediante una transformación de coordenadas del sistema de coordenadas concurrente al sistema de coordenadas anterior

$$ {}^{n-1}T_n = \left[\begin{array}{cccc} \cos\theta_n & -\sin\theta_n\cos\alpha_n & \sin\theta_n\sin\alpha_n & r_n\cos\theta_n \\ \sin\theta_n & \cos\theta_n\cos\alpha_n & -\cos\theta_n\sin\alpha_n & r_n\sin\theta_n \\ 0 & \sin\alpha_n & \cos\alpha_n & d_n \\ \hline 0 & 0 & 0 & 1 \end{array}\right] = \left[\begin{array}{ccc|c} & R & & T \\ \hline 0 & 0 & 0 & 1 \end{array}\right] $$

Figura 25. Figura 26.Matriz de Denavit-Hartenberg. Fuente: Wikimedia Foundation, 2024

Donde R es la submatriz 3×3 que describe la rotación y T es la submatriz 3×1 que describe la traslación

Vamos a demostrar mediante un archivo de Excel el funcionamiento de esta matriz para el cálculo de la posición y orientación de las articulaciones de nuestro robot de manera directa.

		Denavit-Hartenberg-Parameter			
		Joint angle in rad	DH-Parameter from UR Supportwebsite		
UR5e					
Kinematics	**Θ [Grad]**	**Θ [rad]**	**a [m]**	**d [m]**	**α [rad]**
Joint 1 (Base)	270,00	4,71238898	0,0000	0,1625	1,57079633
Joint 2 (Shoulder)	-90,00	-1,57079633	-0,4250	0,0000	0
Joint 3 (Elbow)	90,00	1,57079633	-0,3922	0,0000	0
Joint 4 (Wirst 1)	90,00	1,57079633	0,0000	0,1333	1,57079633
Joint 5 (Wrist 2)	-90,00	-1,57079633	0,0000	0,0997	-1,5707963
Joint 6 (Wrist 3)	90,00	1,57079633	0,0000	0,0996	0

Figura 27. Parámetros de Denavit-Hartenberg (DH) para la configuración cinemática del robot UR3e. Fuente: Elaboración propia a partir de datos de UR Support.

Para la primera articulación.

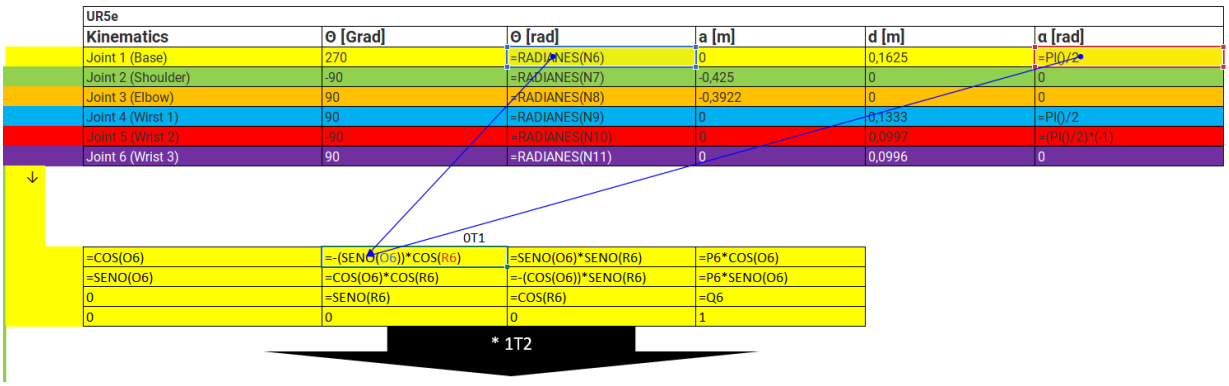


Figura 28: Parámetros de Denavit-Hartenberg (DH) para la primera articulación del robot UR3e. Fuente: Elaboración propia a partir de datos de UR Support.

Mostramos una captura del Excel de aplicación y cálculo de la cinemática directa de nuestro robot, en este caso, mostrando las fórmulas para comprobar la correcta aplicación de la matriz de DH.

Podemos observar como en la matriz efectivamente solo intervienen el giro de la articulación (theta) y la desviación respecto a Z (alpha), así como a(m) y d(m).

Y será por medio de la variación del giro de cada articulación que el robot alcanzará las posiciones deseadas.

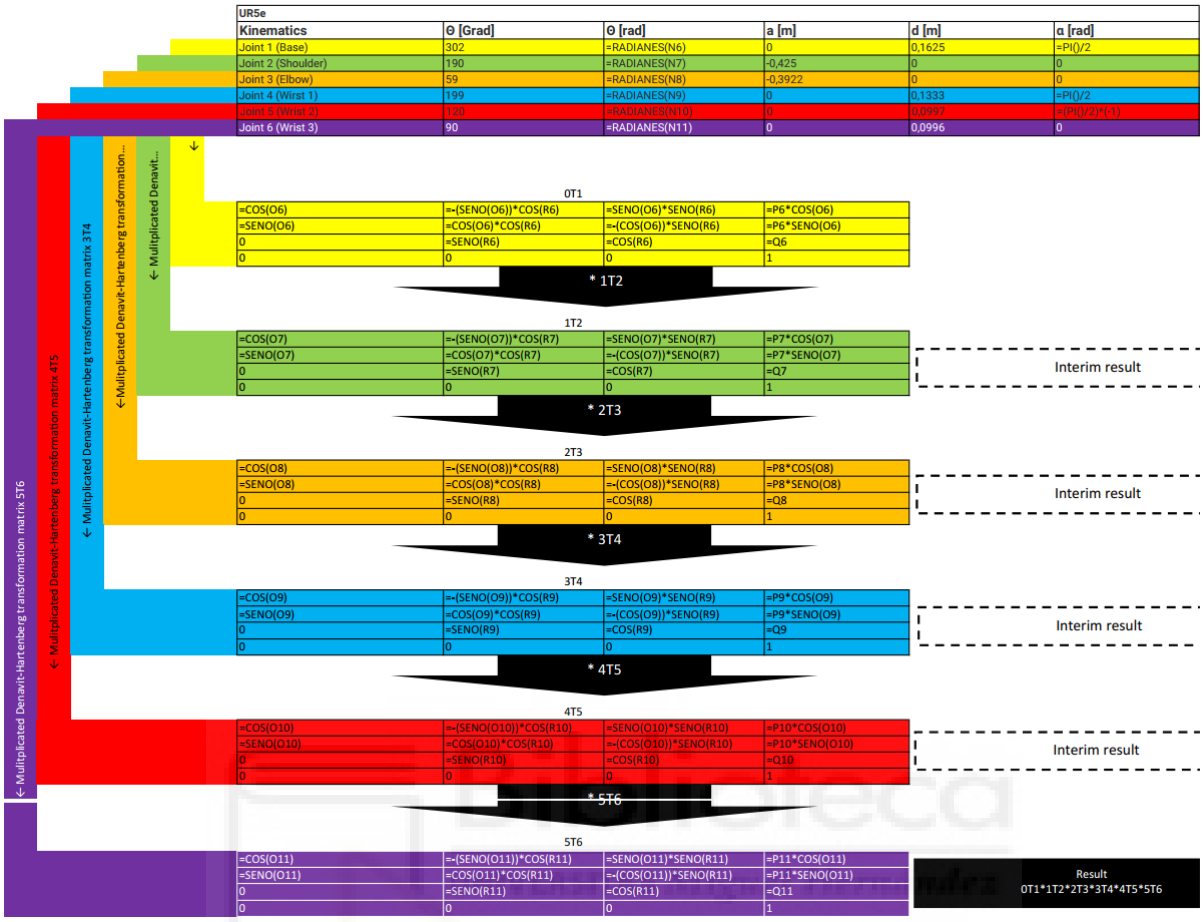
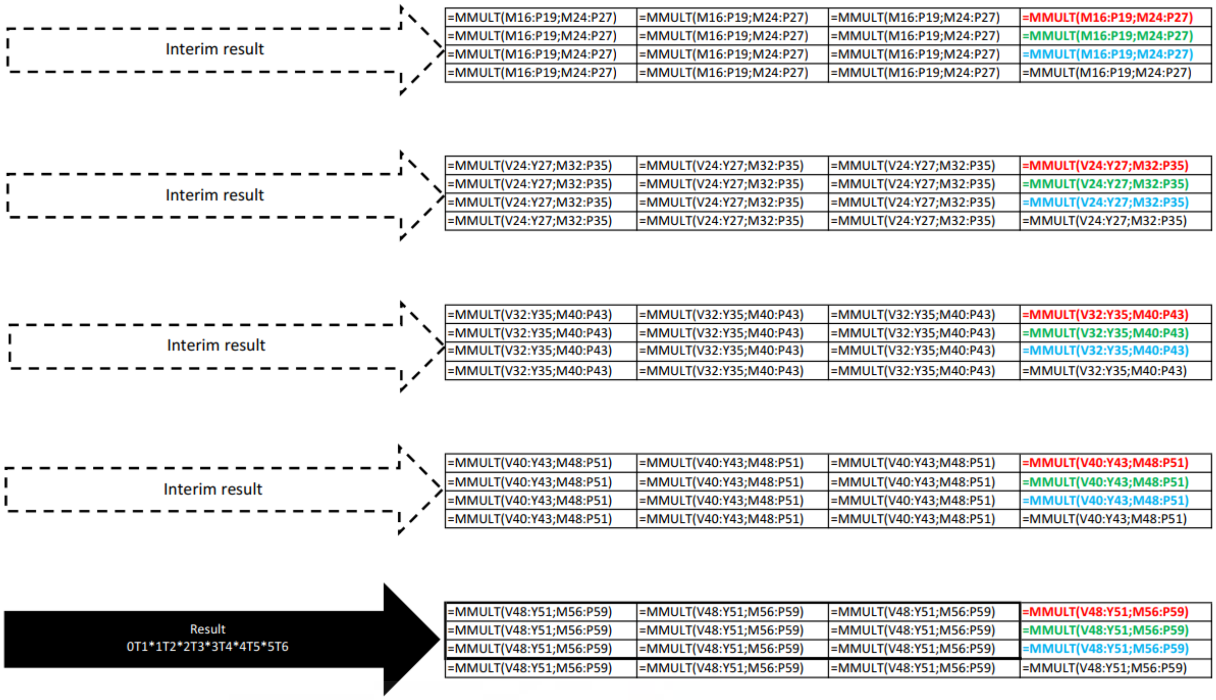


Figura 29. Parámetros de Denavit-Hartenberg (DH) para las diferentes articulaciones del robot UR3e. Fuente: Elaboración propia a partir de datos de UR Support.

Si repetimos este proceso 5 veces más obtenemos las sucesivas matrices de transformación de cada una de las articulaciones. Las 5 matrices de la derecha se corresponden con la multiplicación de la matriz de DH de cada articulación con la siguiente.



=MMULT(M16:P19;M24:P27)	=MMULT(M16:P19;M24:P27)	=MMULT(M16:P19;M24:P27)	=MMULT(M16:P19;M24:P27)
=MMULT(M16:P19;M24:P27)	=MMULT(M16:P19;M24:P27)	=MMULT(M16:P19;M24:P27)	=MMULT(M16:P19;M24:P27)
=MMULT(M16:P19;M24:P27)	=MMULT(M16:P19;M24:P27)	=MMULT(M16:P19;M24:P27)	=MMULT(M16:P19;M24:P27)
=MMULT(M16:P19;M24:P27)	=MMULT(M16:P19;M24:P27)	=MMULT(M16:P19;M24:P27)	=MMULT(M16:P19;M24:P27)

=MMULT(V24:Y27;M32:P35)	=MMULT(V24:Y27;M32:P35)	=MMULT(V24:Y27;M32:P35)	=MMULT(V24:Y27;M32:P35)
=MMULT(V24:Y27;M32:P35)	=MMULT(V24:Y27;M32:P35)	=MMULT(V24:Y27;M32:P35)	=MMULT(V24:Y27;M32:P35)
=MMULT(V24:Y27;M32:P35)	=MMULT(V24:Y27;M32:P35)	=MMULT(V24:Y27;M32:P35)	=MMULT(V24:Y27;M32:P35)
=MMULT(V24:Y27;M32:P35)	=MMULT(V24:Y27;M32:P35)	=MMULT(V24:Y27;M32:P35)	=MMULT(V24:Y27;M32:P35)

=MMULT(V32:Y35;M40:P43)	=MMULT(V32:Y35;M40:P43)	=MMULT(V32:Y35;M40:P43)	=MMULT(V32:Y35;M40:P43)
=MMULT(V32:Y35;M40:P43)	=MMULT(V32:Y35;M40:P43)	=MMULT(V32:Y35;M40:P43)	=MMULT(V32:Y35;M40:P43)
=MMULT(V32:Y35;M40:P43)	=MMULT(V32:Y35;M40:P43)	=MMULT(V32:Y35;M40:P43)	=MMULT(V32:Y35;M40:P43)
=MMULT(V32:Y35;M40:P43)	=MMULT(V32:Y35;M40:P43)	=MMULT(V32:Y35;M40:P43)	=MMULT(V32:Y35;M40:P43)

=MMULT(V40:Y43;M48:P51)	=MMULT(V40:Y43;M48:P51)	=MMULT(V40:Y43;M48:P51)	=MMULT(V40:Y43;M48:P51)
=MMULT(V40:Y43;M48:P51)	=MMULT(V40:Y43;M48:P51)	=MMULT(V40:Y43;M48:P51)	=MMULT(V40:Y43;M48:P51)
=MMULT(V40:Y43;M48:P51)	=MMULT(V40:Y43;M48:P51)	=MMULT(V40:Y43;M48:P51)	=MMULT(V40:Y43;M48:P51)
=MMULT(V40:Y43;M48:P51)	=MMULT(V40:Y43;M48:P51)	=MMULT(V40:Y43;M48:P51)	=MMULT(V40:Y43;M48:P51)

=MMULT(V48:Y51;M56:P59)	=MMULT(V48:Y51;M56:P59)	=MMULT(V48:Y51;M56:P59)	=MMULT(V48:Y51;M56:P59)
=MMULT(V48:Y51;M56:P59)	=MMULT(V48:Y51;M56:P59)	=MMULT(V48:Y51;M56:P59)	=MMULT(V48:Y51;M56:P59)
=MMULT(V48:Y51;M56:P59)	=MMULT(V48:Y51;M56:P59)	=MMULT(V48:Y51;M56:P59)	=MMULT(V48:Y51;M56:P59)
=MMULT(V48:Y51;M56:P59)	=MMULT(V48:Y51;M56:P59)	=MMULT(V48:Y51;M56:P59)	=MMULT(V48:Y51;M56:P59)

Figura 30. Detalle de las fórmulas aplicadas para el cálculo de los parámetros de Denavit-Hartenberg (DH) para las diferentes articulaciones del robot UR3e. Fuente: Elaboración propia a partir de datos de UR Support.

Por ejemplo, para obtener el primer parámetro de la submatriz de traslación de la última junta del robot podemos observar cómo depende de cada una de las submatrices obtenidas para cada articulación anterior.

Observamos que cada articulación está definida por una posición en X, Y, Z resultado de estas matrices de rotación.

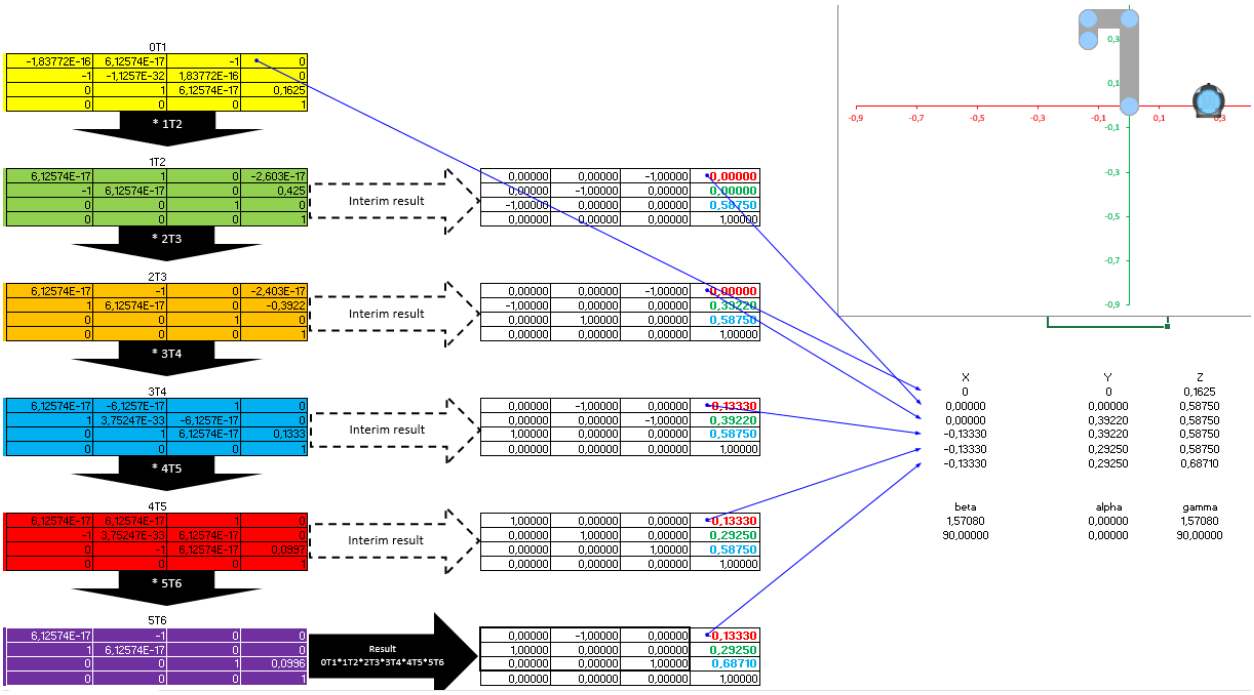


Figura 31. Detalle de cálculo por cinemática directa de las posiciones (X,Y,Z) del robot UR3e. Fuente: Elaboración propia a partir de datos de UR Support.

Por último, se muestran los ángulos de rotación beta, alpha y gamma calculados en radianes (arriba) y en grados (abajo).

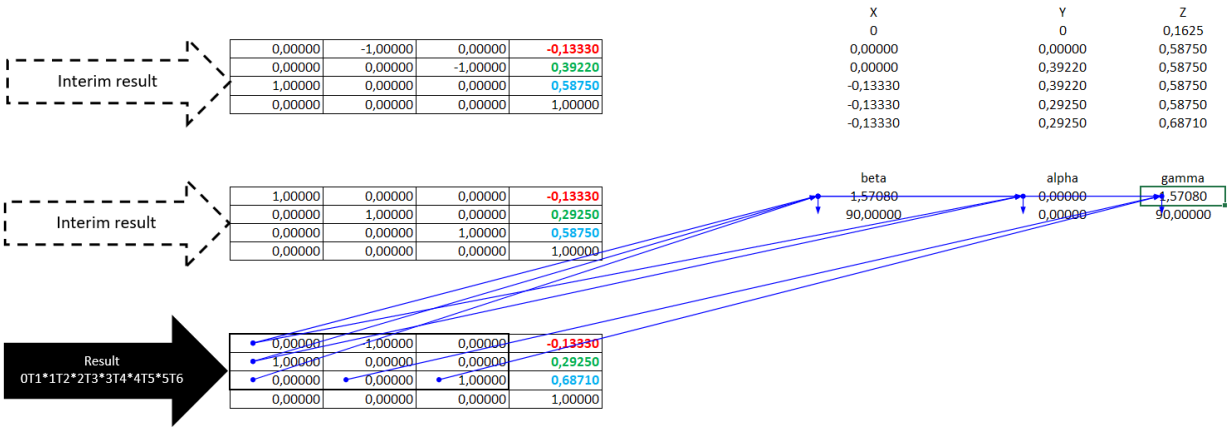


Figura 32. Detalle de cálculo por cinemática directa de los ángulos de rotación (Rx, Ry, Rz) del TCP del UR3e. Fuente: Elaboración propia a partir de datos de UR Support.

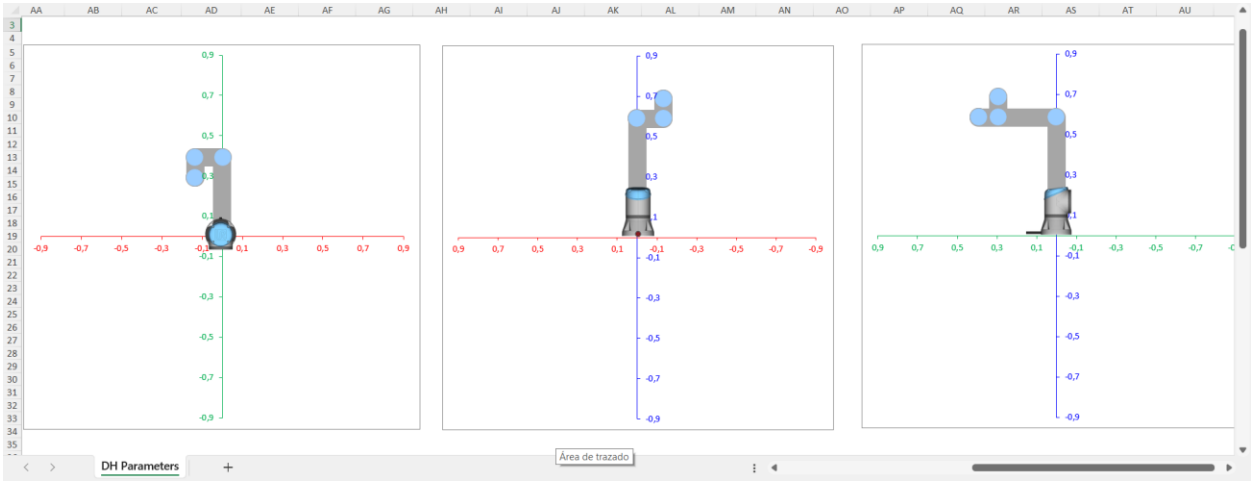


Figura 33. Gráficos de las posiciones adoptadas por el robot UR3e desde diferentes puntos de vista tras los cálculos realizados. Fuente: Elaboración propia a partir de datos de UR Support.

Estas son las diferentes poses del robot atendiendo a los ejes X-Y (vista de planta), X-Z (vista de alzado) e Y-Z (vista de perfil) de acuerdo con las rotaciones indicadas inicialmente en nuestra matriz de parámetros.

5. Desarrollo e implementación

La idea general del proyecto consiste en desarrollar los programas, comunicaciones y en definitiva la infraestructura necesaria para que el robot UR3e pueda mediante una cámara detectar la posición exacta de unos bloques de PLA y realizar el paletizado de estos.

La dificultad del proyecto no estriba tanto en la complejidad del programa del robot, sino en la interconexión de este con el PLC y la cámara de visión artificial.

5.1. Configuraciones iniciales

Por medio del software RealVNC Viewer accedemos a la pantalla Teach Pendant, es decir, al HMI (Human-Machine Interface) del robot para hacer más sencillo el proceso de configuración y programación.

De lo contrario, también podemos realizar la configuración de manera táctil a través de la teach pendant.

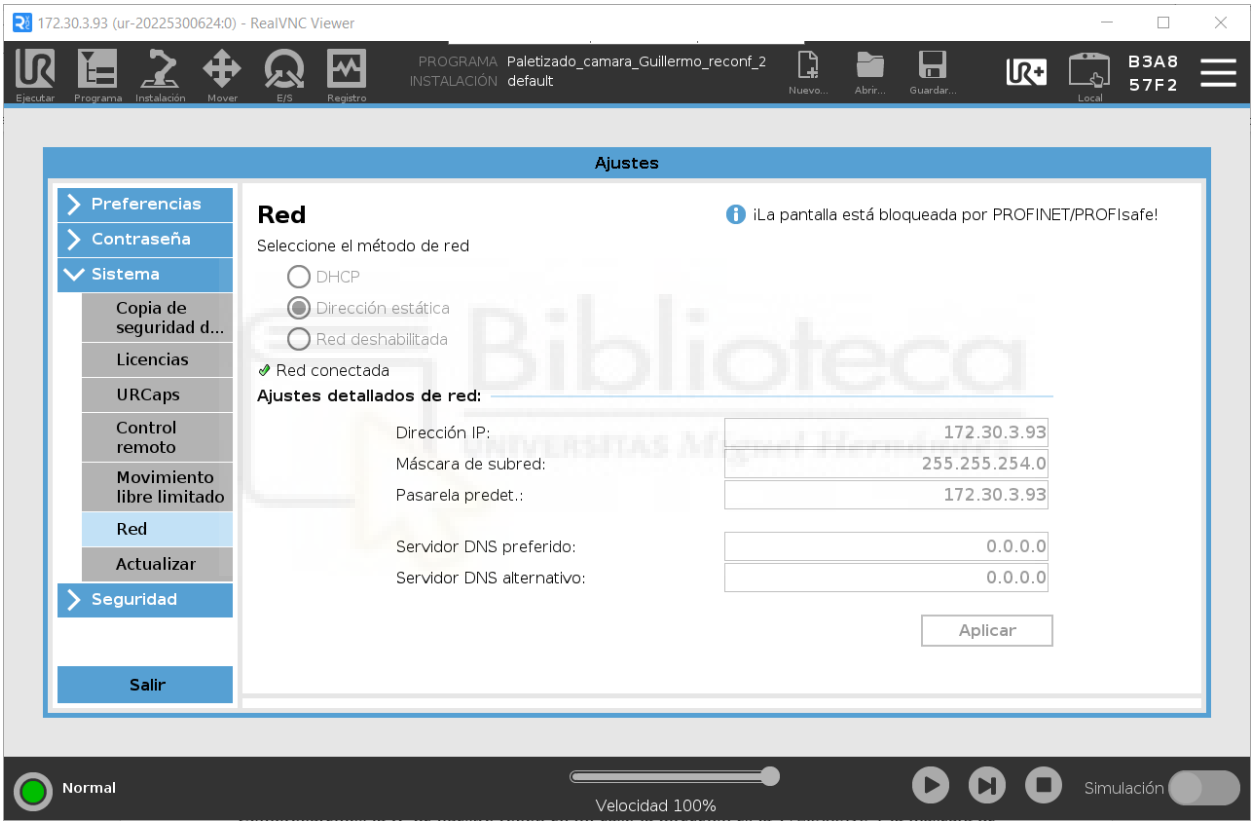


Figura 34. Comprobación de la dirección IP establecida en nuestro robot UR3e. Fuente: Elaboración propia.

Comprobaremos la IP de nuestro robot, en mi caso la dirección es la 172.30.3.93 y la máscara de subred 255.255.254.0.

Existe la posibilidad de integrar un arranque remoto en el robot por medio de activación de la entrada I0.3 en nuestro PLC S7-1200 que a su vez activa la salida Q0.0 y desde nuestra salida

del PLC Q0.0 vamos al borne del encendido remoto del robot. Mediante un pulsador en I1.4 lo que haríamos es activar la salida Q0.1 del robot que apagaría el mismo.

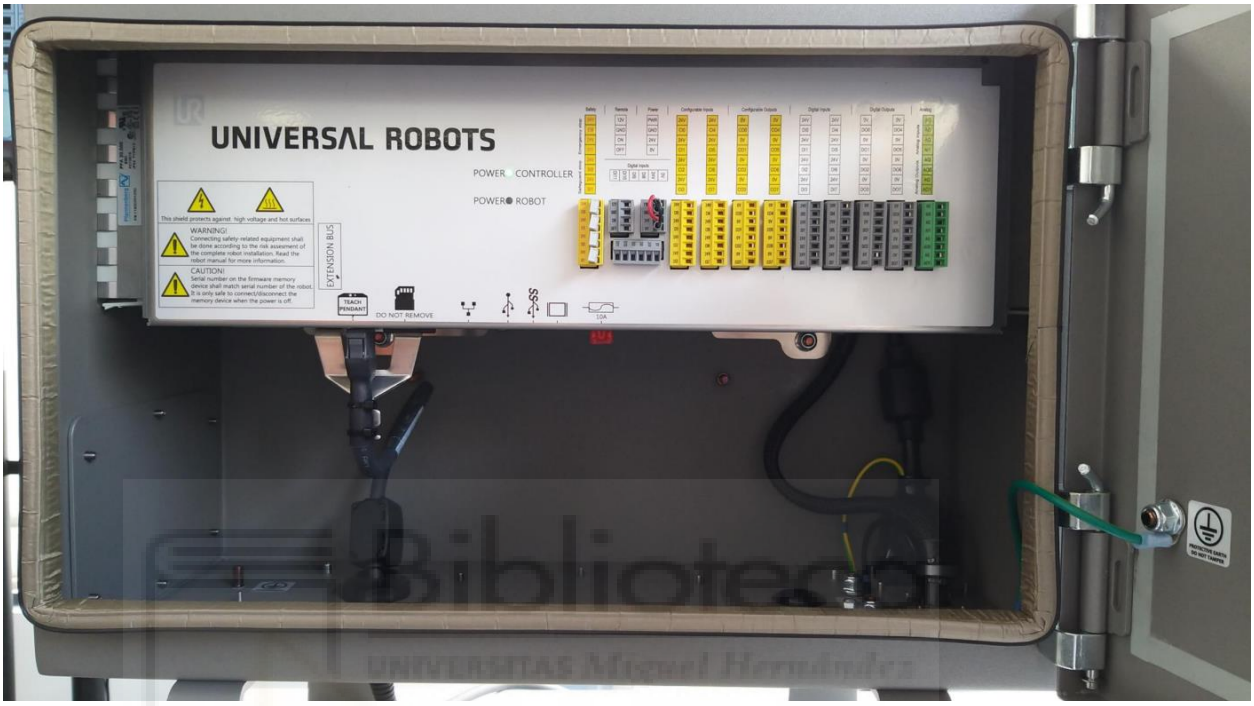


Figura 35. Fotografía de nuestro armario de conexiones del robot UR3e antes de realizar ninguna conexión relativa al proyecto actual. Fuente: Elaboración propia.

.

El común de las salidas lo conectamos al bornero de 12V del robot, es decir, al GND y la salida Q0.0 del PLC a ON, la salida Q0.1 iría cableada a OFF. Este bornero independientemente de si el robot está apagado nos está dando esa tensión de 12V que nos permite encenderlo.

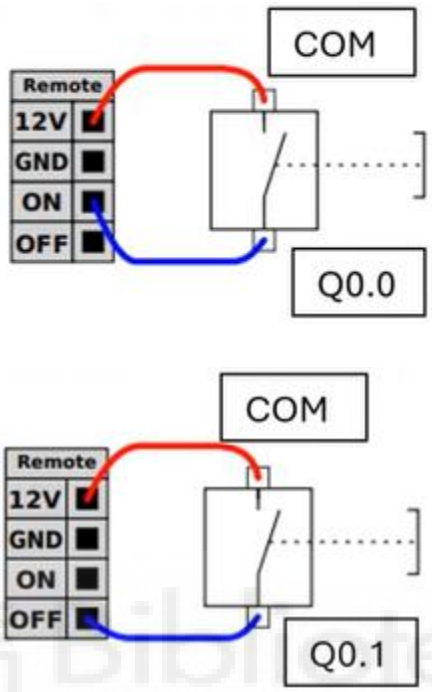


Figura 36. Esquema de encendido y desactivación remotos. Fuente: Universal Robots A/S, Manual de e-Series, 2018.

Estas variables deberán ser definidas también en el apartado de *Config. E/S* de *Instalación* dentro del software Polyscope, las variables son las siguientes:

- GPbi[0] Autoinicio: encendido del robot, liberación de frenos y carga del programa.
- GPbi[1] Start: inicio del programa configurado
- GPbi[2] Paro: detención del programa en ejecución

Figura 37. Configuración de registros de booleano en polyscope para carga e inicio del programa remotos. Fuente: Elaboración propia.

Desde la pestaña *Arranque* terminamos de configurar la opción de Autoinicio que programaremos posteriormente en nuestro PLC de modo que con la activación de I1.1 se active la entrada de registro GPbi[0] y se cargue nuestro programa a ejecutar.

Debemos recordar que desde el teachpendant en la esquina superior derecha debemos seleccionar la opción de *Control Remoto.*

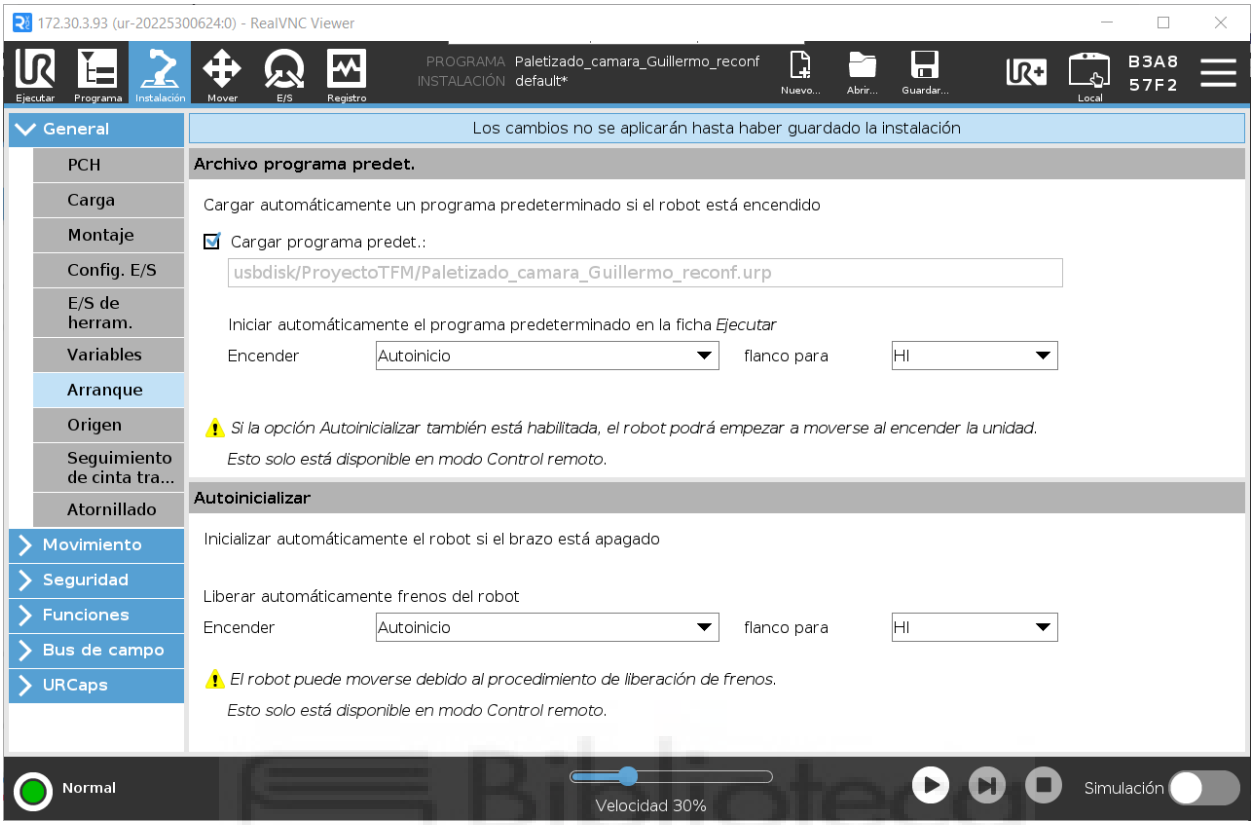


Figura 38. Configuración del arranque y autoinicialización del robot en polyscope. Fuente: Elaboración propia.

Podemos preguntarnos por qué el robot mantiene su posición incluso antes de encenderlo. La respuesta es sencilla, el robot posee unos frenos mecánicos en cada una de sus juntas que evitan el libre desplazamiento de estas cuando el robot no está alimentado. Al encender el robot un electroimán liberará los frenos, de hecho al hacerlo se escucha un sonido característico en cada uno de los seis ejes.

También se produce una leve sacudida como consecuencia de esta liberación, por eso es importante alejar al robot de lugares comprometidos, como por ejemplo, cuando la herramienta está situada muy cerca del suelo o algún otro plano de trabajo.

Definimos ahora la herramienta que nuestro robot va a utilizar para esta aplicación, nos hemos decidido por una pinza 2fg7 como la que se muestra a continuación.

Figura 39. Detalle de la pinza 2FG7 de On Robot. Fuente: Ficha técnica herramienta 2FG7, OnRobot A/S, 2021.

Para configurarla el primer paso es en *Instalación* desde la pestaña de *E/S de la herram.* seleccionar OnRobot.

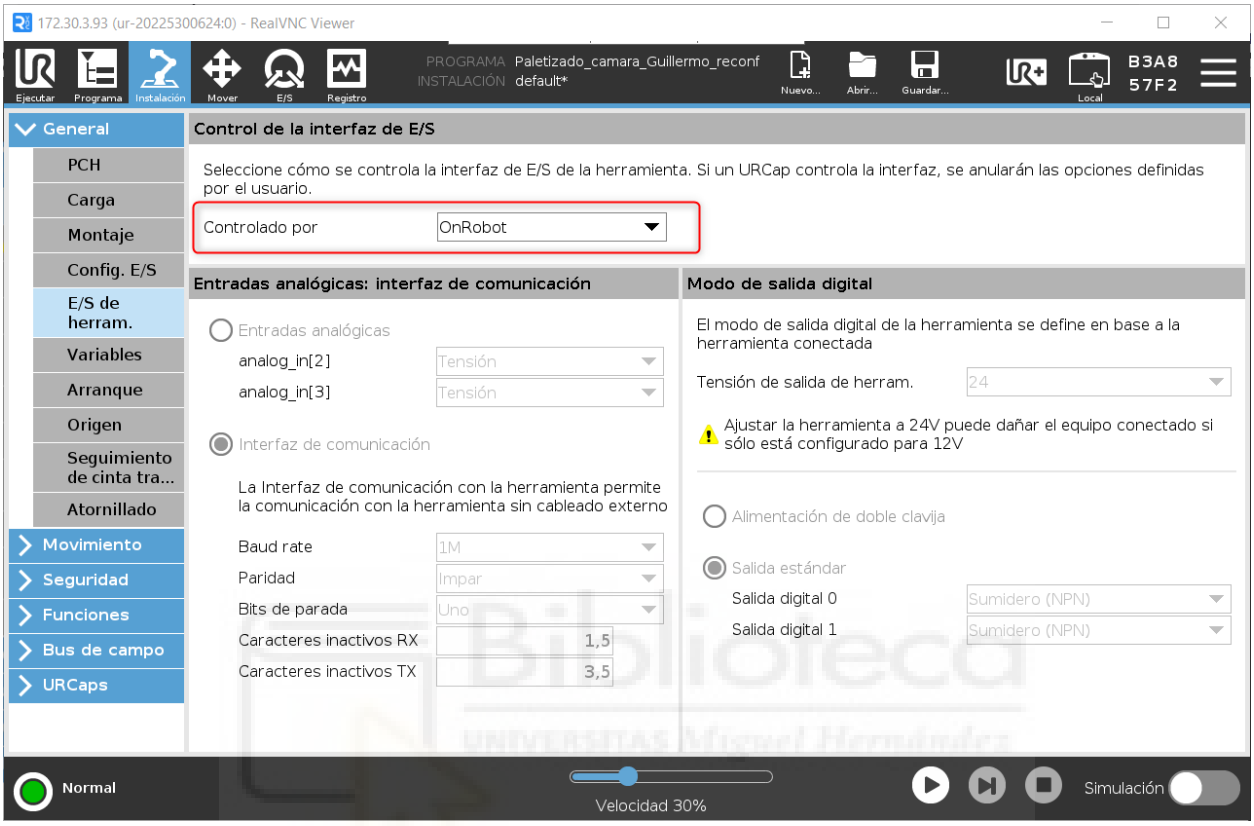


Figura 40. Configuración del control de la interfaz de E/S de la herramienta. Fuente: Elaboración propia.

Es necesario también definir la *posición de la herramienta*, debiendo especificar el radio de giro de la misma (35 cm aproximadamente) y la posición en Z de la misma, si la misma tiene una longitud de aproximadamente 100 mm podemos señalar una distancia de 135 mm por ejemplo.

Siempre algo superior siempre pues la herramienta está pensada para cargar con objetos pequeños pero que pueden ser de diferentes tamaños.

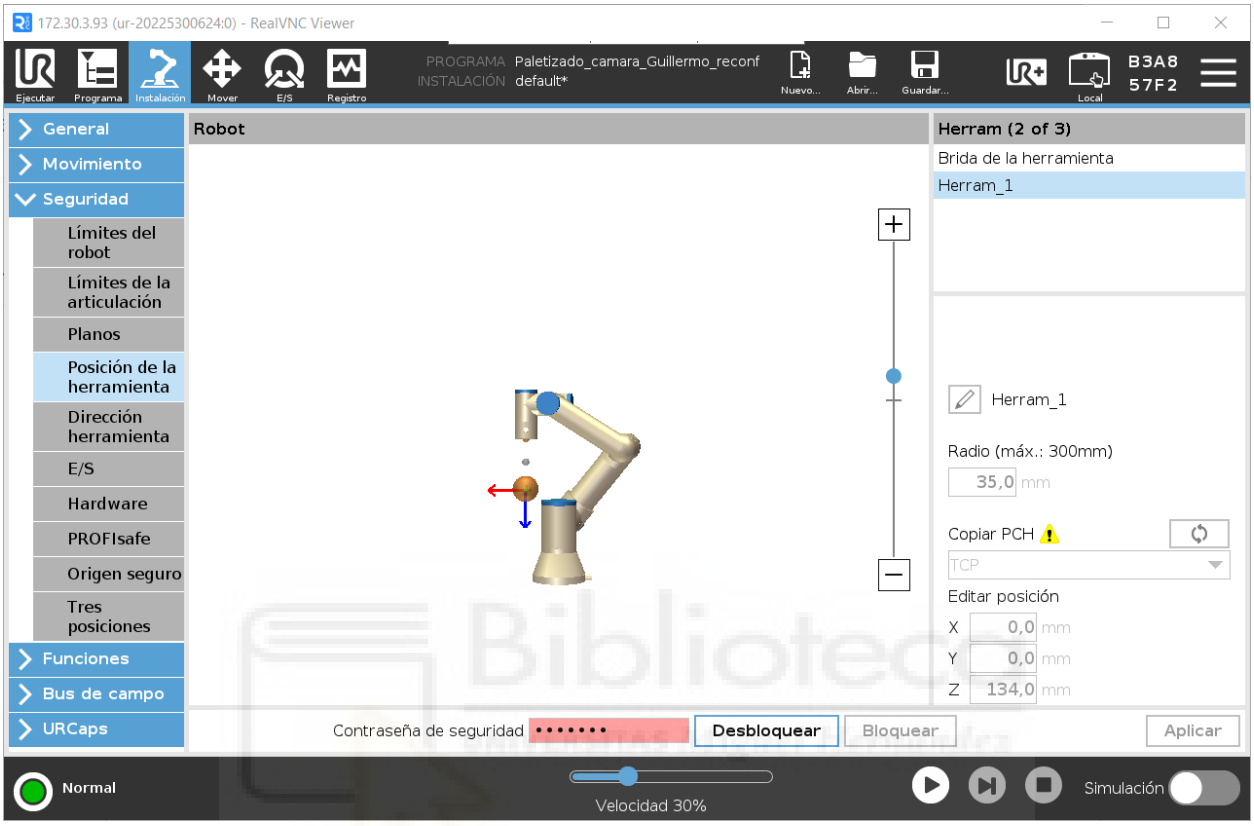


Figura 41. Configuración de la posición de la herramienta. Fuente: Elaboración propia.

Una vez hecho esto iniciamos el robot, para ello definimos (si no lo hemos hecho ya) una carga, es decir, un Payload y su peso.

En mi caso, la carga que soportará el brazo robot en su extremo, incluyendo la herramienta y el peso de los objetos a desplazar es de 1,12 kg.

En el apartado *carga* debemos señalar el peso de la pinza con la carga y además el centro de gravedad de la herramienta con la carga, que en este caso, dado que las piezas a cargar no superan los 20 ~ 30 gramos de peso y dado que no son muy voluminosas el centro de gravedad

será aproximadamente el de la herramienta más unos 8 mm, es decir, unos 58 mm.

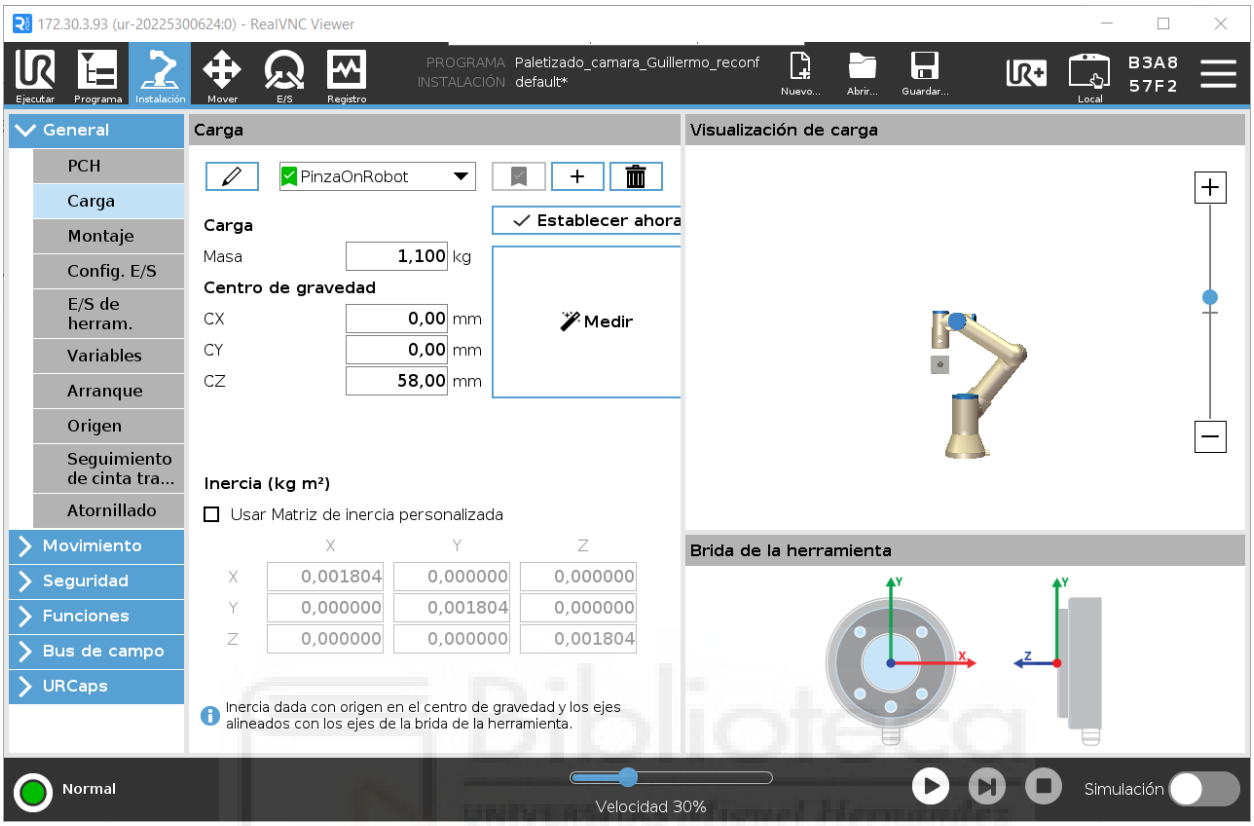


Figura 42. Configuración de la carga a soportar por el TCP (peso de la herramienta y carga adicional) y centro de gravedad de la herramienta y carga. Fuente: Elaboración propia.

5.1.1. Configuración y calibración de la cámara Sensopart

Para la calibración usamos una plantilla de calibración proporcionada por sensopart y accedemos al software Polyscope por medio de RealVNC y comenzamos instalando la URCap necesaria desde la propia página web del fabricante de la cámara, en este caso Sensopart.com. Desde la pestaña DOWNLOADS accedemos a la parte de descargas y descargamos desde Software >>Vision Sensors >> Carpeta VISOR Robotic Interfaces >> Archivo VISOR Robotic URCap V2.0.0.

Una vez ya descargado el archivo dentro de la carpeta descargada encontramos el manual de la URCap entre otros archivos.

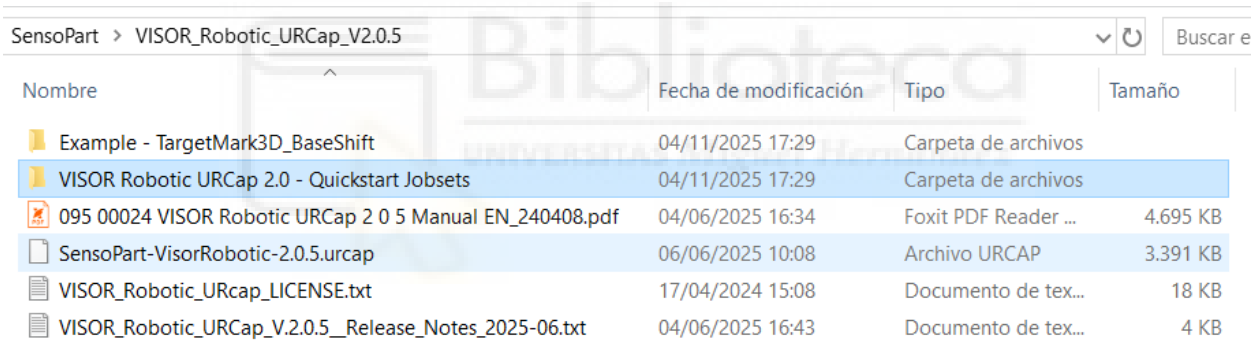


Figura 43. Archivo con formato. urcap utilizado. Fuente: Elaboración propia.

Tenemos el archivo con extensión. urcap que deberemos de copiar a un USB y tenemos dentro una serie de Quickstart Jobsets o tareas que vamos a instalar en la cámara.

Una vez copiados estos archivos será necesario también imprimir las calibration plates que si hemos instalado previamente el software de VISOR ya deberíamos encontrarlas en Sensopart>>Visor Vision Sensor >> Documentación >> Calibrationplates.

En mi caso he impreso la de 15x13 200mm a tamaño real, es necesario hacerlo así pues la cámara la utilizará para calibrar y ajustar las distancias de medida.

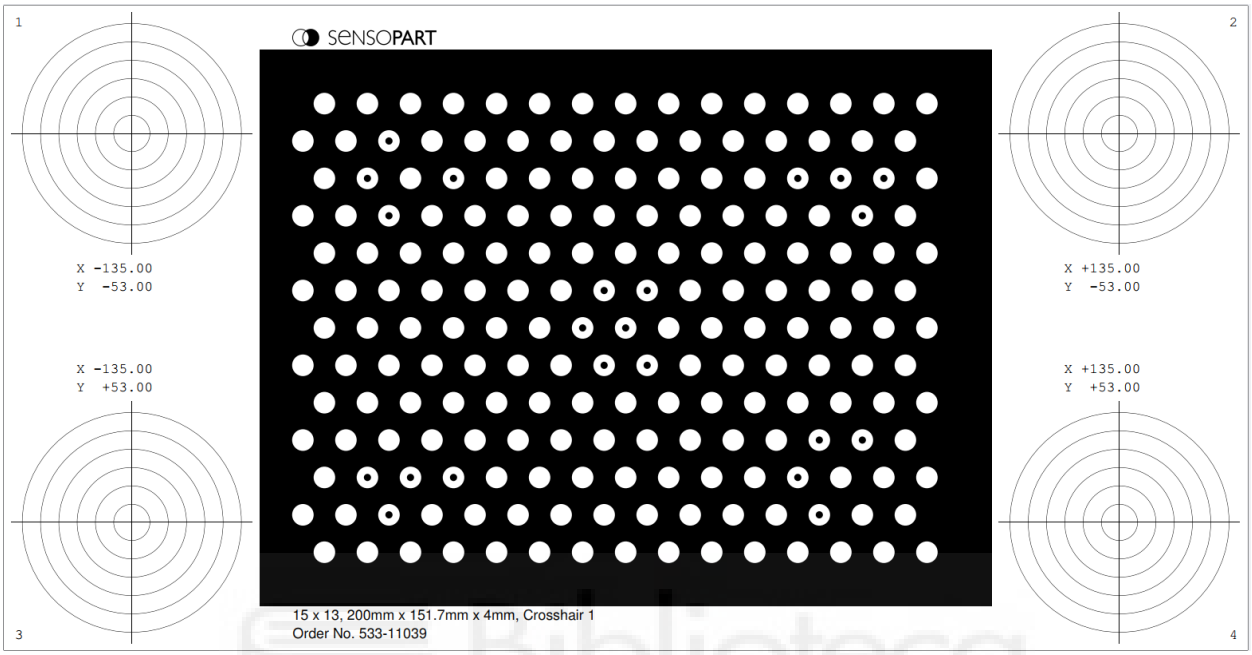


Figura 44. Plantilla de calibración. Fuente: SensoPart Industriesensorik GmbH

El siguiente paso será la instalación de la UR Cap (el archivo con extensión .urcap) que permitirá la integración de la cámara con el robot UR3e. Previamente en la pestaña de >Seguridad >>Permisos habremos seleccionado >>URCaps.

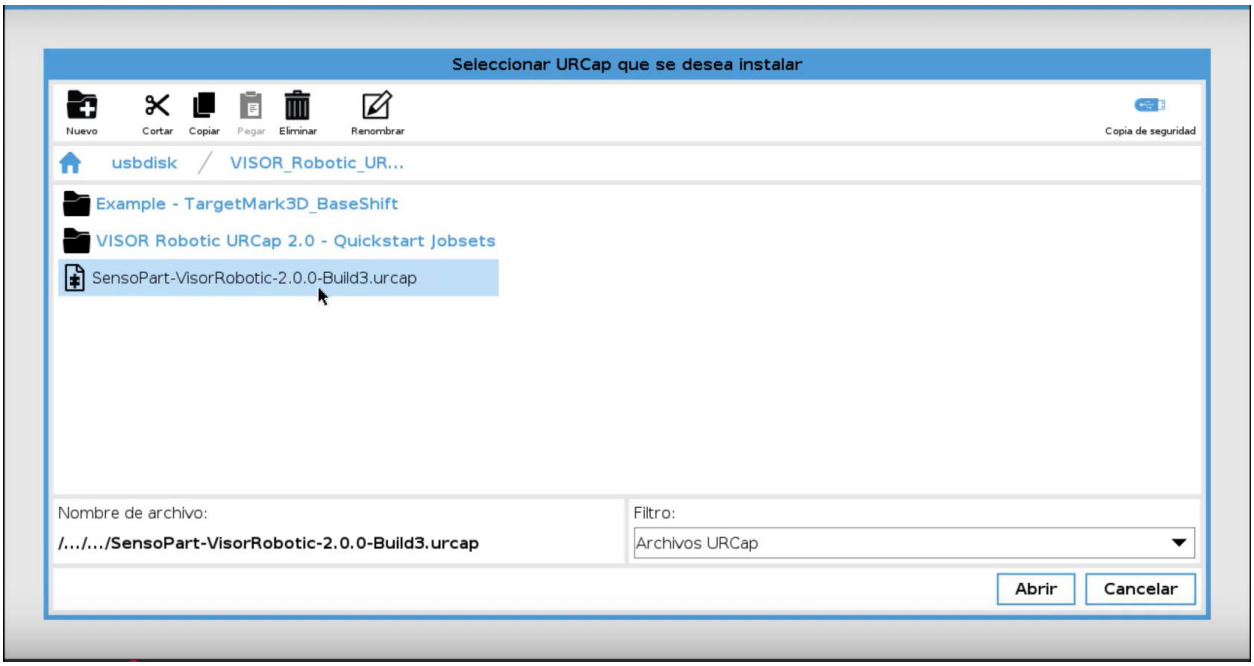


Figura 45. Carga por medio de dispositivo USB de la Urcap descargada en nuestro robot. Fuente: Elaboración propia.

Desde >>Instalación >>UR Caps podemos observar las UR Caps ya instaladas.

Si no está ya instalada debemos ir al fabricante y descargar la URCap de nuestra cámara, copiarla en una memoria USB y copiarla en el teachpendant (HMI) del robot.

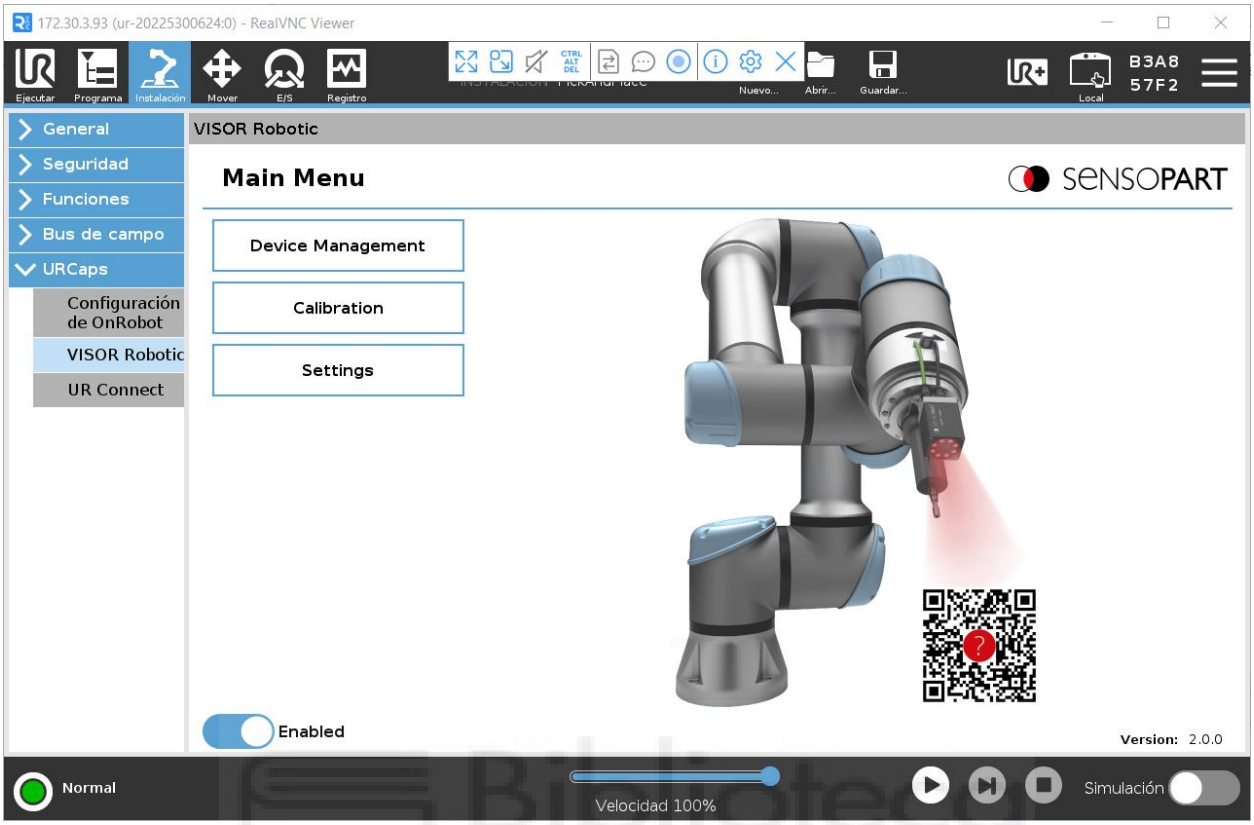


Figura 46. Acceso a la Urcap VISOR Robotic. Fuente: Elaboración propia.

Accedemos a la UR Cap de VISOR Robotic y desde Device Management localizamos la cámara, en nuestro caso con dirección IP 172.30.3.17. Comprobamos que estamos conectados a ella y si por algún motivo no aparece automáticamente ingresamos esta dirección a mano ya que la detección de equipos conectados a la red puede no ser siempre tan fina como desearíamos.

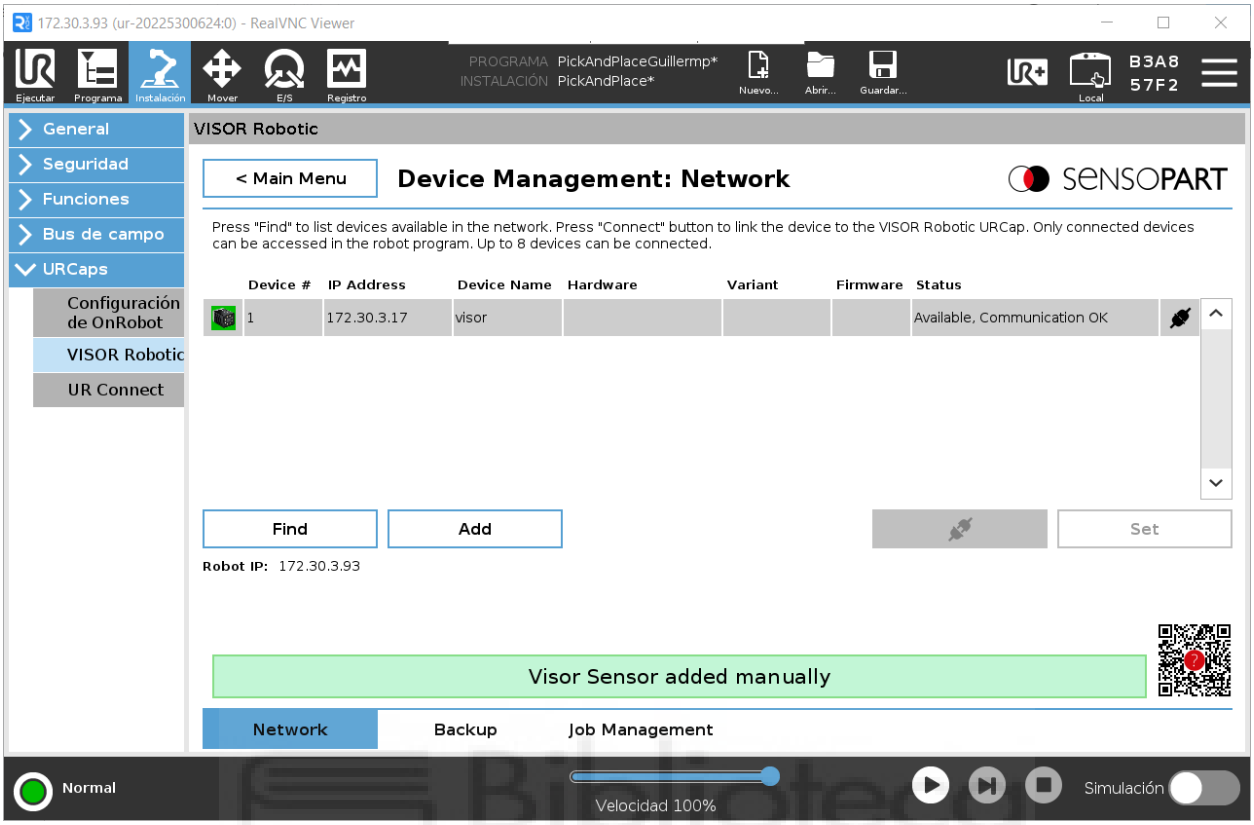


Figura 47. Configuración de la dirección IP de nuestra cámara SensoPart. Fuente: Elaboración propia.

Es importante comprobar que la opción de comunicación por Ethernet está habilitada en la cámara para ello accedemos al programa de SensoFind de VISOR Vision Sensor (pues muchas veces viene solo por defecto la comunicación por PROFINET) y desde ajustes >>interfaces seleccionamos Ethernet.

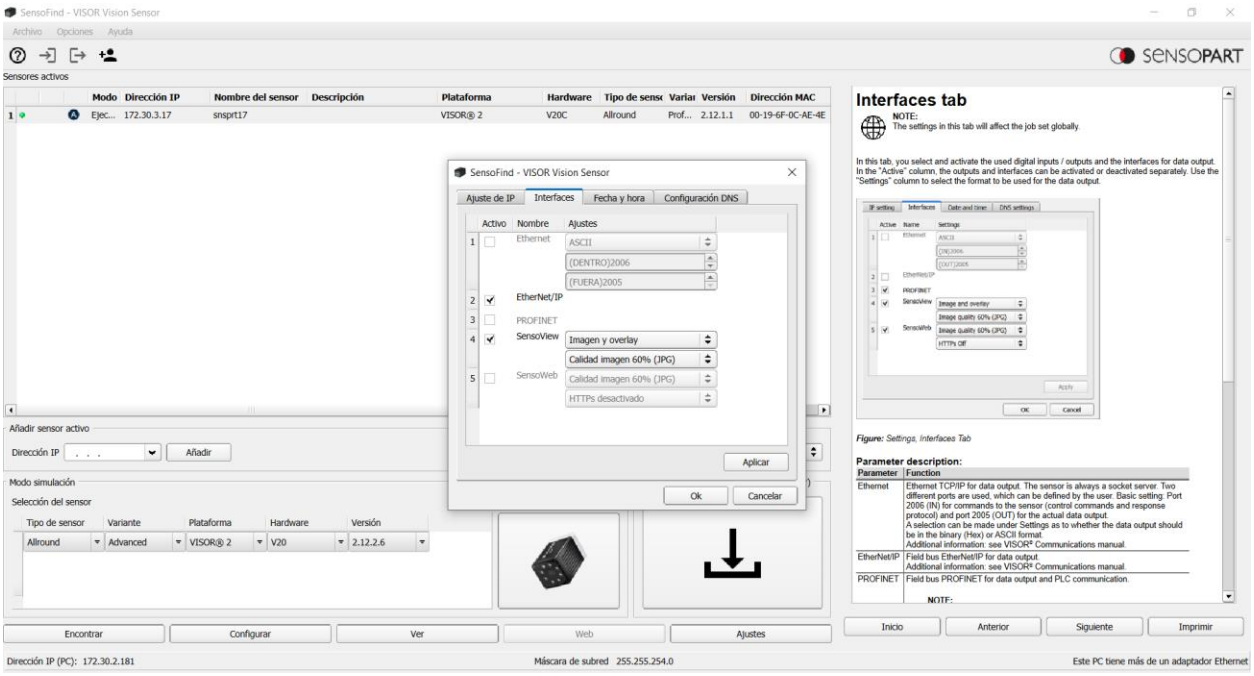


Figura 48. Vista de la interfaz de SensoFind. Fuente: Elaboración propia.

A continuación se muestra una vista de detalle del protocolo a seleccionar para la correcta comunicación de nuestro sistema cámara-robot.

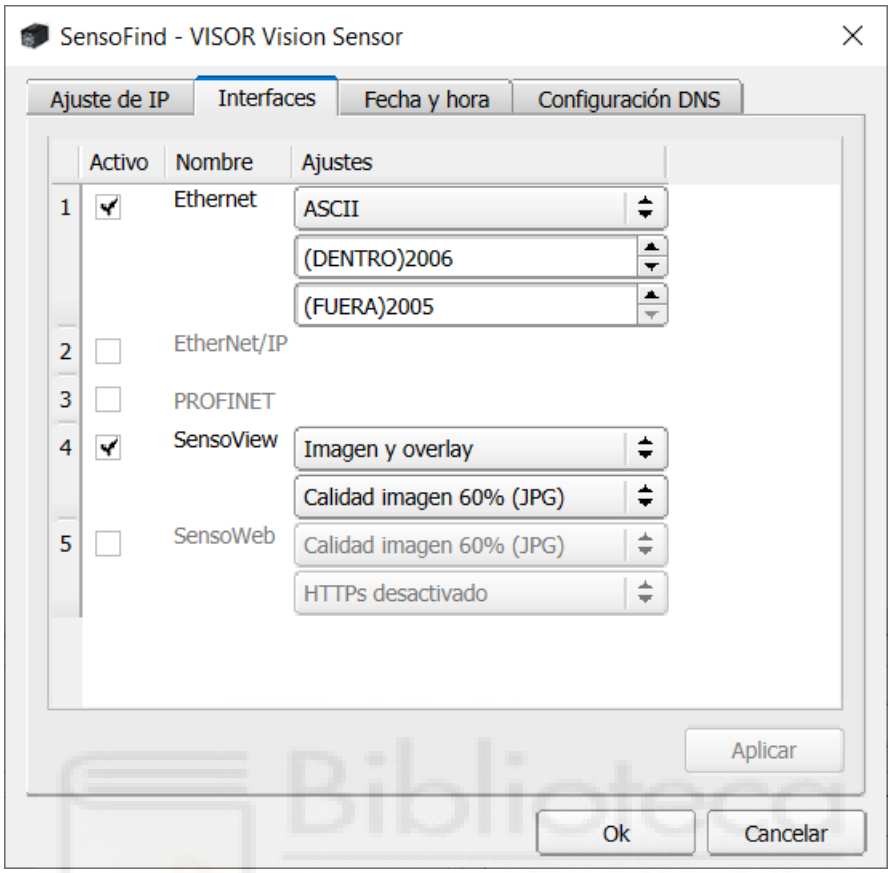


Figura 49. Selección de la opción Ethernet. Fuente: Elaboración propia.

Una vez hecho esto ya deberíamos poder detectar la cámara desde nuestro software de polyscope, es decir, desde nuestro robot y así empezar a configurar la interconexión de ambos.

Otro detalle interesante es la posibilidad de visionar desde el software de control del robot (polyscope) la imagen tomada de la cámara. Para ello si hacemos click en el margen superior derecho podremos acceder a las URCaps instaladas, seleccionamos la de la cámara y si la conexión es correcta deberíamos poder visualizar la imagen desde nuestra teachpendant.

Esto puede ser muy útil para operarios que se encuentren trabajando con el robot para no tener que recurrir a un PC externo si tan solo están realizando modificaciones en programa o para operarios que estén accediendo de forma remota al software de programación del robot.

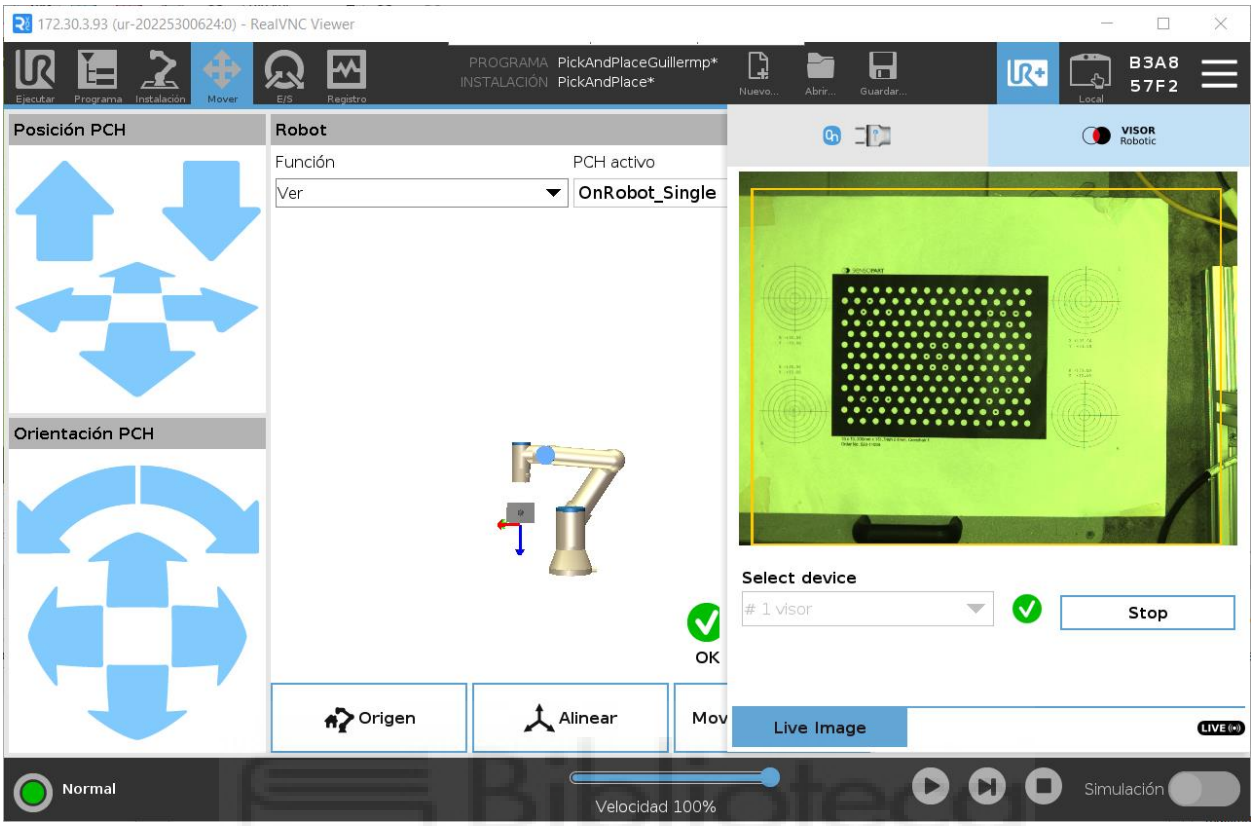


Figura 50. Prueba de la conexión. Fuente: Elaboración propia.

En este proyecto, emplearemos la tarea de detección de múltiples piezas. A continuación, ajustamos una serie de parámetros importantes como la velocidad del obturador (esto determinará la cantidad de luz que llega al sensor), la distancia de trabajo y el autoenfoque. El rectángulo morado es el que determina este último parámetro y debe contener el área de análisis de la cámara, una vez realizado el autoenfoque no debe tocarse tras calibrar o se desconfigura todo.

En la imagen siguiente se muestra la interfaz de SensoConfig y cómo en cada tarea que generemos se deben y pueden ajustar los parámetros de velocidad del obturador, ganancia,etc .

Ajustamos con la opción “Auto” tanto la distancia de trabajo como la velocidad del obturador.

Recomiendo no tocar la distancia de trabajo pues este parámetro está relacionado con el enfoque de la cámara y deberíamos realizar la calibración de esta de nuevo.

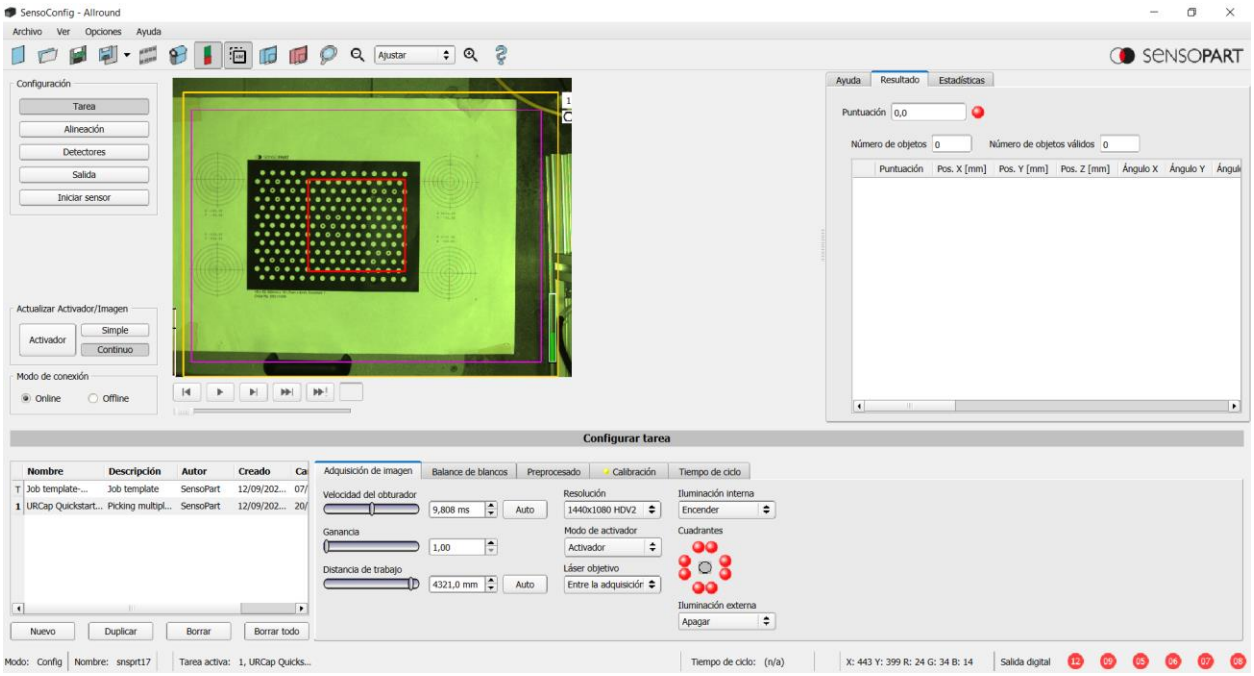


Figura 51. Configuraciones de la cámara desde SensoConfig.. Fuente: Elaboración propia.

Desde Polyscope es importante no olvidar el Z-Shift measurement plane pues si nosotros calibramos a una altura, es decir, situamos nuestra calibration plate a una distancia de la cámara pero luego nuestro plano de trabajo se encuentra a otra distancia debemos indicarlo. Y es que así va a ser en la mayoría de las aplicaciones (especialmente si requieren operaciones de pick and place de objetos con una altura determinada).

En mi caso se van a detectar y coger unas figuras con un saliente a modo de “asa” para que la pinza (TCP) del robot pueda sujetarlos. Estos cubos son impresos en 3D con el fín único de

demostrar la utilidad del robot UR3e y vemos que estos tienen una altura aproximada de 33 mm.

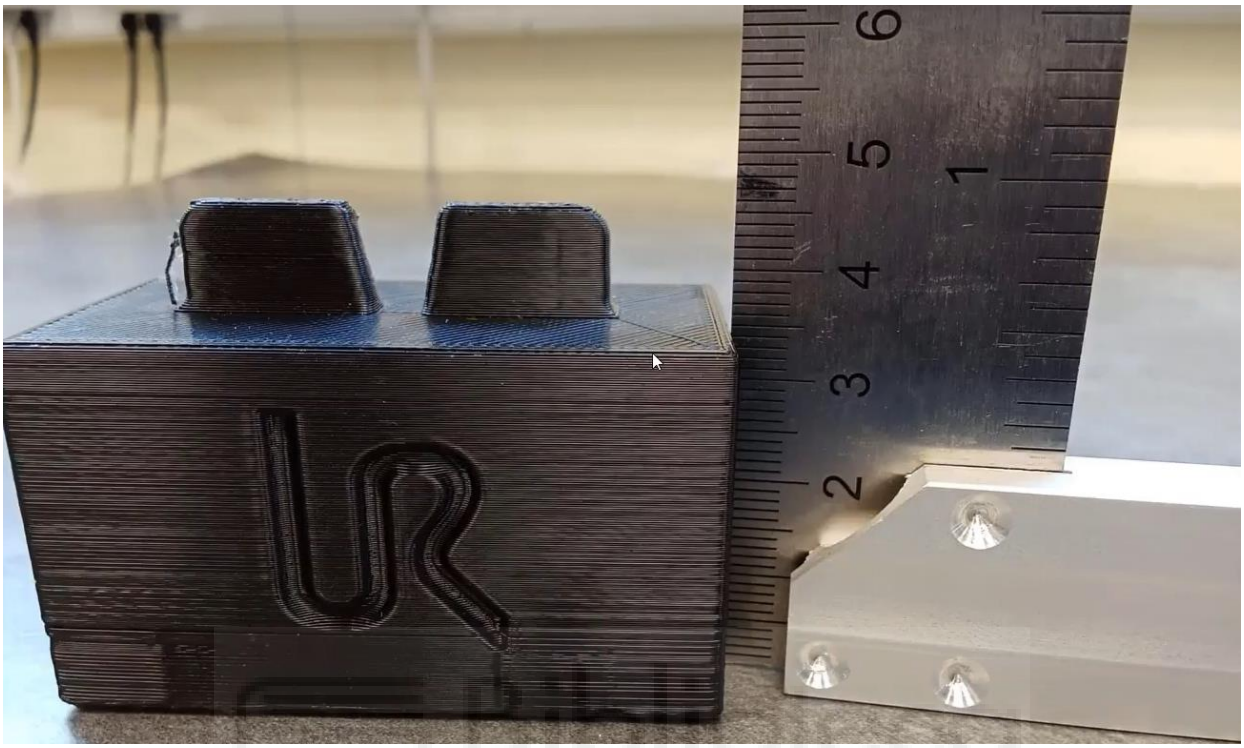


Figura 52. Medición de altura de la pieza con escuadra. Fuente: Elaboración propia.

Debemos distinguir entre plano de detección y plano de trabajo. Pues hemos configurado nuestro plano de detección a 0 mientras que nuestro plano de trabajo (para el TCP) va a ser a 33 mm de este.

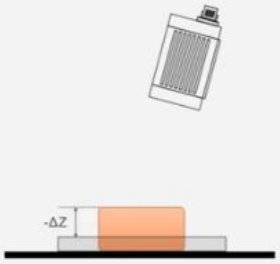
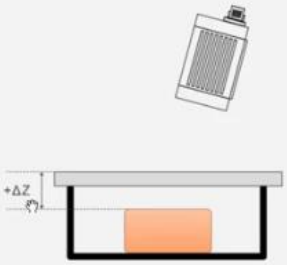
Parameter	Function
Z-shift of Measurement plane	The "Z-shift of Measurement plane" parameter can be used to move the measuring plane along the Z axis (perpendicular to the plane) in order to obtain more accurate results, if necessary. For Z=0, the calibration and the measurement plane are identical. For Z ≠ 0, the calibration plane is shifted relative to the measurement plane. The planes are always parallel. The sign of the shift results from the Z direction of the right-handed calibration coordinate system (thumb = X, index finger = Y, middle finger = Z).  ***Fig. 53:*** *"Z-shift of Measurement plane" negative*  ***Fig. 54:*** *"Z-shift of Measurement plane" positive*

Figura 53. Captura del manual de usuario de la cámara. Fuente: Hoja de datos cámara V20C, Gottenheim, SensoPart, 2021

El fabricante nos indica en el manual que si el objeto "sobresale" del plano de medida entonces debemos especificar un valor de variación de Z negativo, mientras que si se encontrase por debajo deberíamos especificar un valor de Z positivo.

Una vez completados todos los campos de esta etapa de calibrado del VISOR Robotic le damos a siguiente y a SET para establecer la calibración.

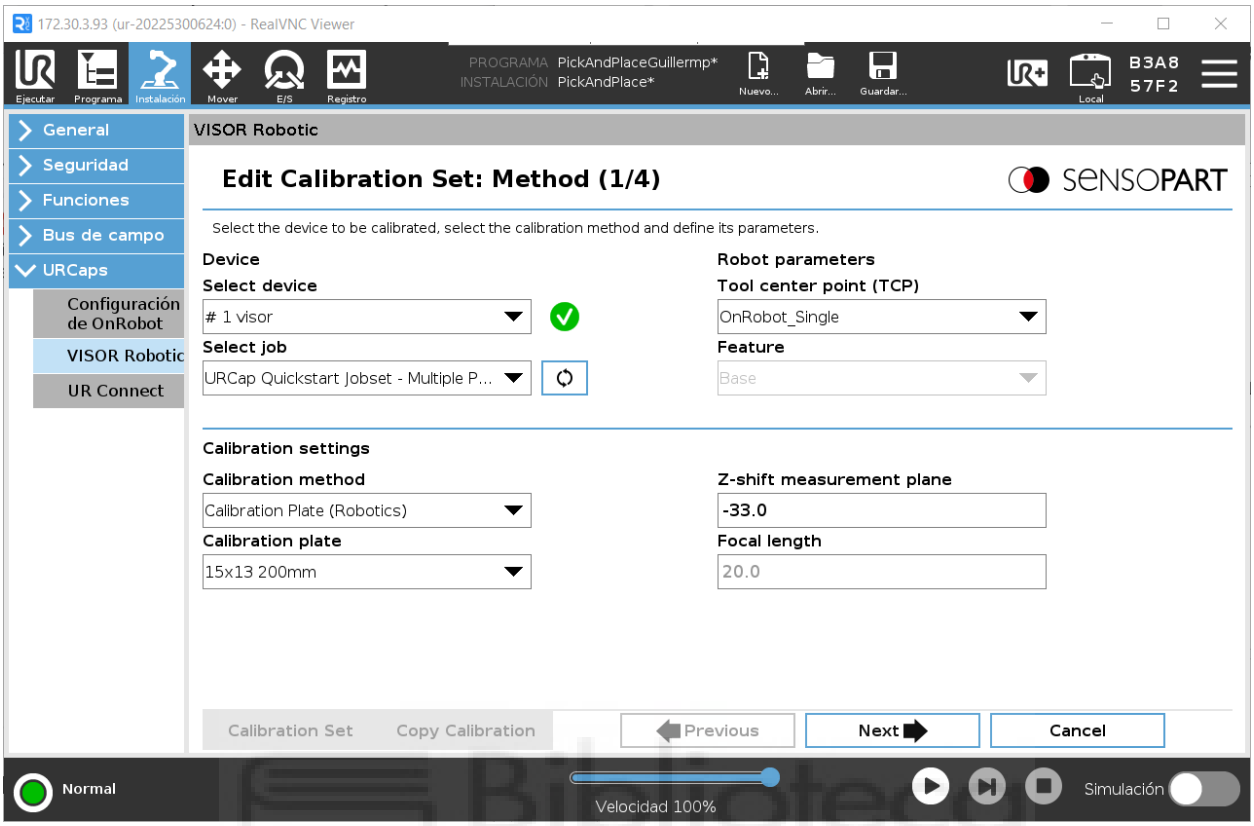


Figura 54. Calibración de la Urcap de la cámara y selección de la tarea de detección de múltiples piezas. Fuente: Elaboración propia.

De nuevo en Polyscope, en el Main Menu de la UR Cap de Visor Robotic, accedemos a la segunda pestaña, la pestaña de Calibration, en esta deberemos generar una nueva calibración para nuestra cámara. Si hubiese alguna calibración antigua lo más seguro es que no nos sirva, la calibration plate usada puede haber sido otra o la herramienta (TCP) puede también haber sido otra. Así que generamos una nueva calibración y desde el software de SensoConfig cargamos una nueva Tarea.

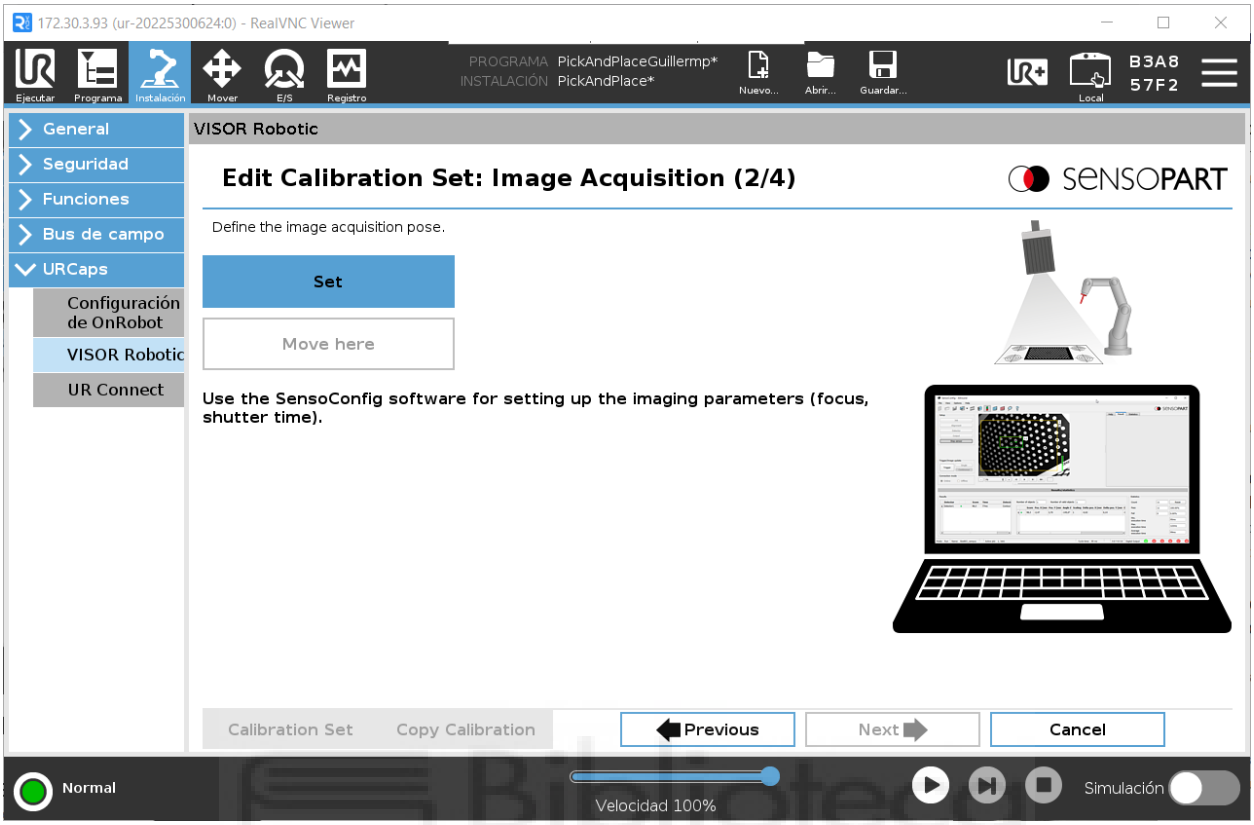


Figura 55. Configuración de la pose del robot cuando se toma la fotografía. Fuente: Elaboración propia.

Seleccionamos la posición en la que queremos que el robot se encuentre para la foto de calibración. Habitualmente llevamos al robot a una posición más "recogida" para evitar que interfiera con nuestra fotografía a la placa de calibración.

A continuación, llevaremos el TCP del robot a los 4 fiducial points. Estos puntos son las cuatro circunferencias o "dianas" que quedan situadas en las esquinas de nuestra calibration plate y sobre el centro de las mismas deberemos situar la punta de la herramienta del robot.

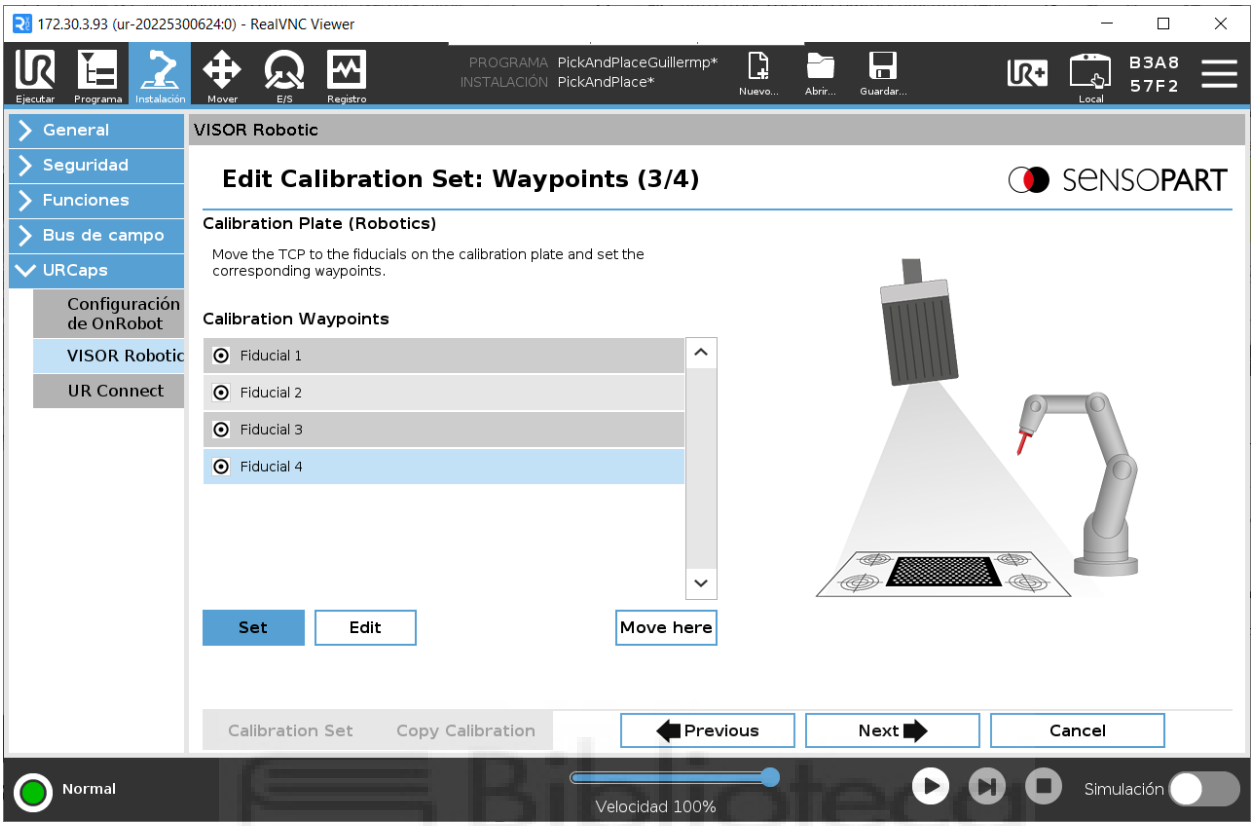


Figura 56. Calibración del TCP respecto de los puntos fiduciales. Fuente: Elaboración propia.

Este es un punto crítico pues se requiere de mucha precisión para llevar el extremo del brazo robot a cada uno de los puntos y yo recomiendo hacerlo evitando los movimientos libres, es decir, recomiendo hacerlos desde el control de los ejes del robot, aunque sea algo más largo el proceso pues no ayudará mucho después en términos de precisión.

Esto es debido a la falta de precisión humana que se infiere cuando movemos libremente el robot, en lugar de guiarlo por sus controles. Por último, comenzaría la calibración por parte de la cámara que dispara una foto y nos indica la desviación de los puntos de calibración y de los puntos fiduciales. Si está es mayor a la máxima permitida no dará por apta la calibración.

El último paso es la obtención de los resultados de calibración.

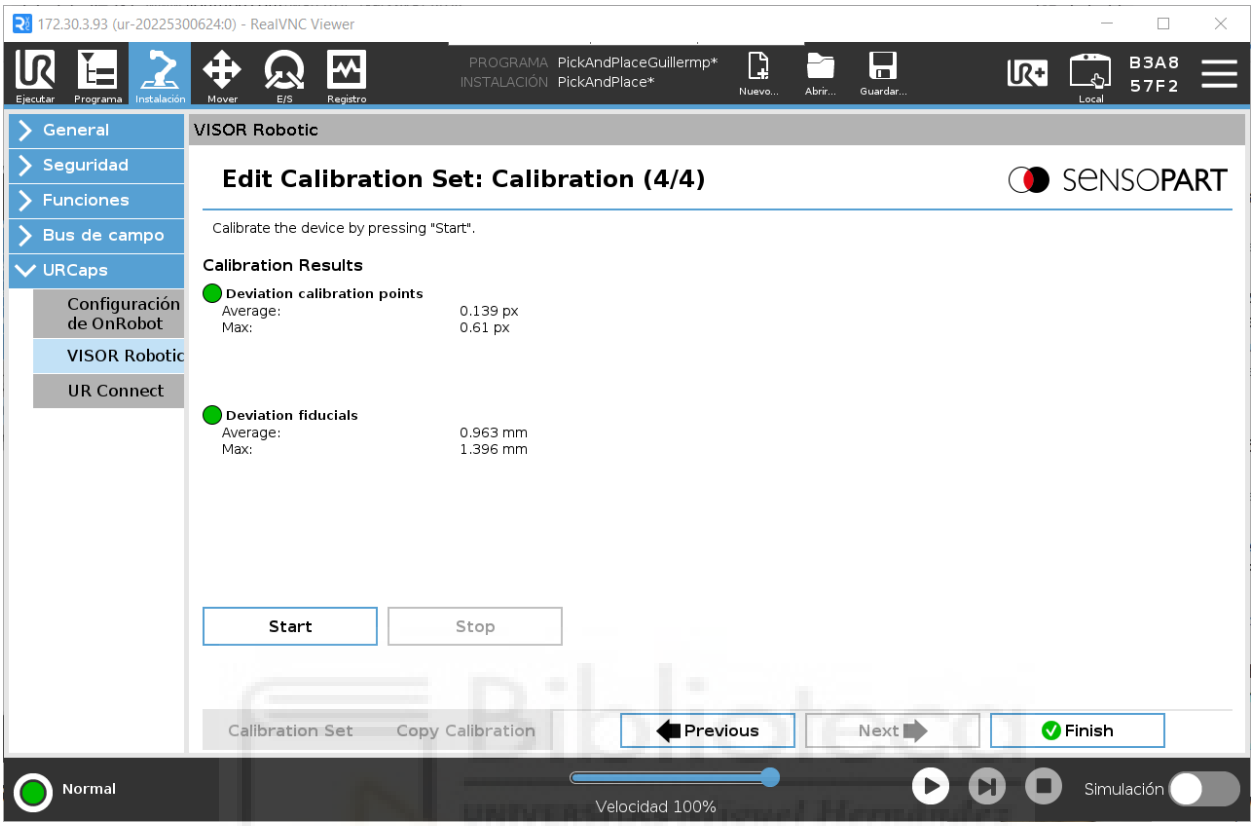


Figura 57. Resultados de calibración. Fuente: Elaboración propia.

Como vemos en la anterior imagen la desviación media es de 0.963 mm, es inferior al milímetro así que la daremos por válida, teniendo en cuenta que si fuera mayor y las exigencias de precisión de nuestra aplicación lo requiriese deberíamos realizar la calibración de nuevo.

Finalizada la calibración, vamos a ponerla a prueba, arrastramos a nuestro programa en polyscope la VISOR Robotic App que resulta en un programa de pick&place predeterminado que debemos acabar de configurar para ver que nuestra cámara está calibrada y trabaja en armonía con nuestro robot.

5.2. Programación del robot

En este apartado se llevará a cabo la programación del robot UR3e para que sea capaz de interpretar las imágenes de la cámara y de posicionarse adecuadamente sobre las piezas, recogerlas y realizar un movimiento de place o paletizado.

En primer lugar, arrastramos a nuestro árbol de programa el URCap de VISOR Robotic Application que consiste en una tarea sencilla de Easy Pick que básicamente consiste en configurar el contorno de la pieza que queremos detectar, de modo que la cámara informará al robot de la posición en X y en Y de la pieza, así como de la orientación de la herramienta para coger la pieza.

En este caso, queremos coger varias piezas así que seleccionamos el Jobset de múltiples piezas dentro de la variable Setup del programa, y procedemos a establecer una serie de características como el gripping pose o posicionamiento de agarre. Para ello moveremos el TCP del robot a dicha posición de agarre.

A continuación, seleccionamos el detector usado para localizar las partes (podemos consultar el nombre del detector en el software de SensoConfig). Para la configuración de un detector, se debe haber configurado previamente el ROY o región de interés (Recuadro amarillo que define la zona a analizar), el rango angular es otro parámetro de importancia, interesa que sea alto (en este caso, de -180º a 180º) de modo que sea más sencillo detectar el contorno de la pieza sin importar su orientación, el umbral que también influye en la capacidad de detección según el color de la pieza (a piezas más claras o menos contrastadas con el fondo, será necesario un umbral menor), etc.

Para mejorar la capacidad de detección de la cámara puede ser aconsejable acceder a *Optimización de contorno* y desactivar el modelo *de contraste mínimo* en automático, para ajustarlo manualmente.

El último paso "Teach gripping offset" llevará el robot a la posición de reposo, tomará la foto (importante que la pieza no se haya movido) se disparará una foto y la localización de la pieza será llevada al visor.

Ya solo queda completar el programa con los movimientos básicos de cualquier operación de pick and place, es decir, configurar liberación y agarre de herramienta, movimientos en L y en J, esperas de seguridad, etc.

Veámoslo en detalle.

En el árbol de proyecto seleccionamos la rama *# 1 Setup* y seleccionamos la tarea que estamos configurando desde SensoConfig, digo estamos, pues aquí viene un proceso que requiere de varias modificaciones habitualmente para ajustar la tarea y aumentar el porcentaje de éxito de esta.

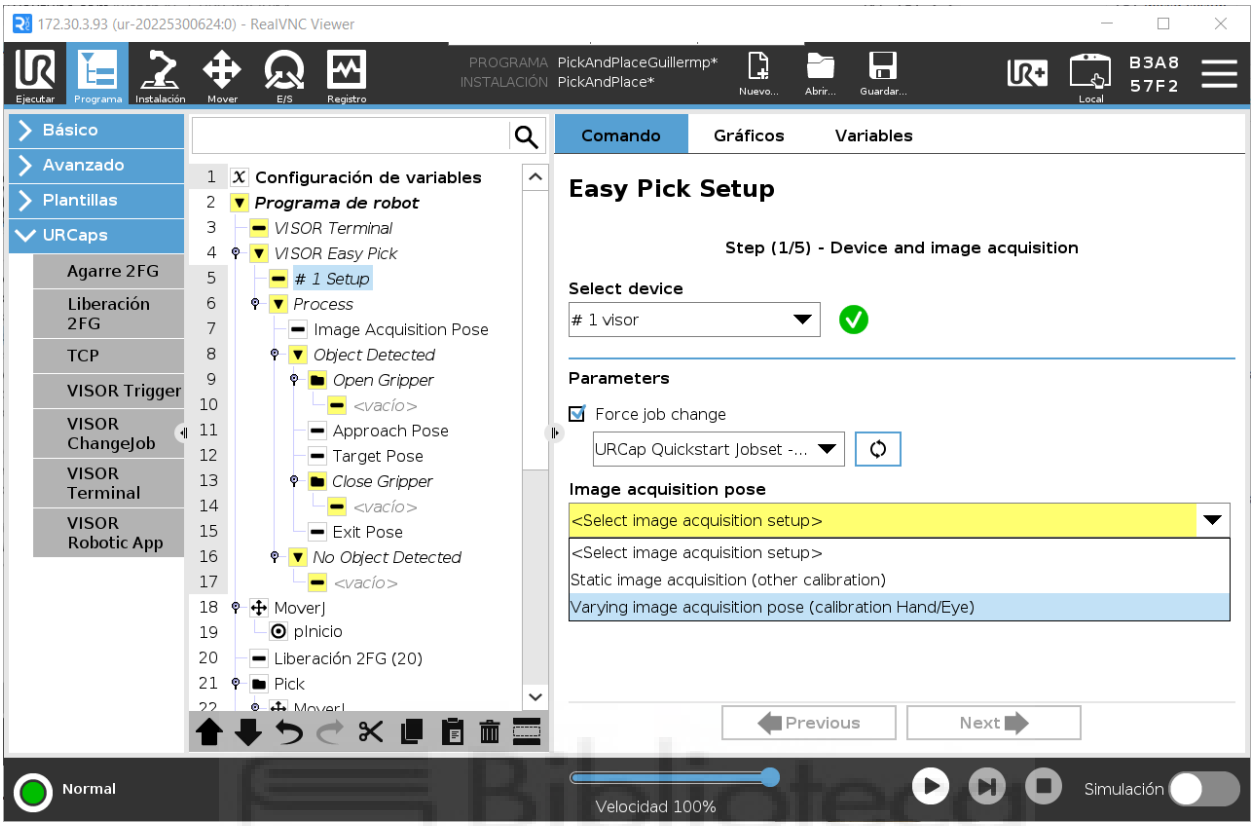


Figura 58. Configuración del VISOR Easy Pick. Fuente: Elaboración propia.

En el apartado de *Image acquisition pose* seleccionaremos la opción de *Static image acquisition (other calibration)* pues al estar la camara fija (montada sobre un soporte y no sobre el robot) la imagen que tomaremos será siempre estática, es decir, no variará el punto de vista de la cámara.

La imagen siguiente muestra el programa finalizado.

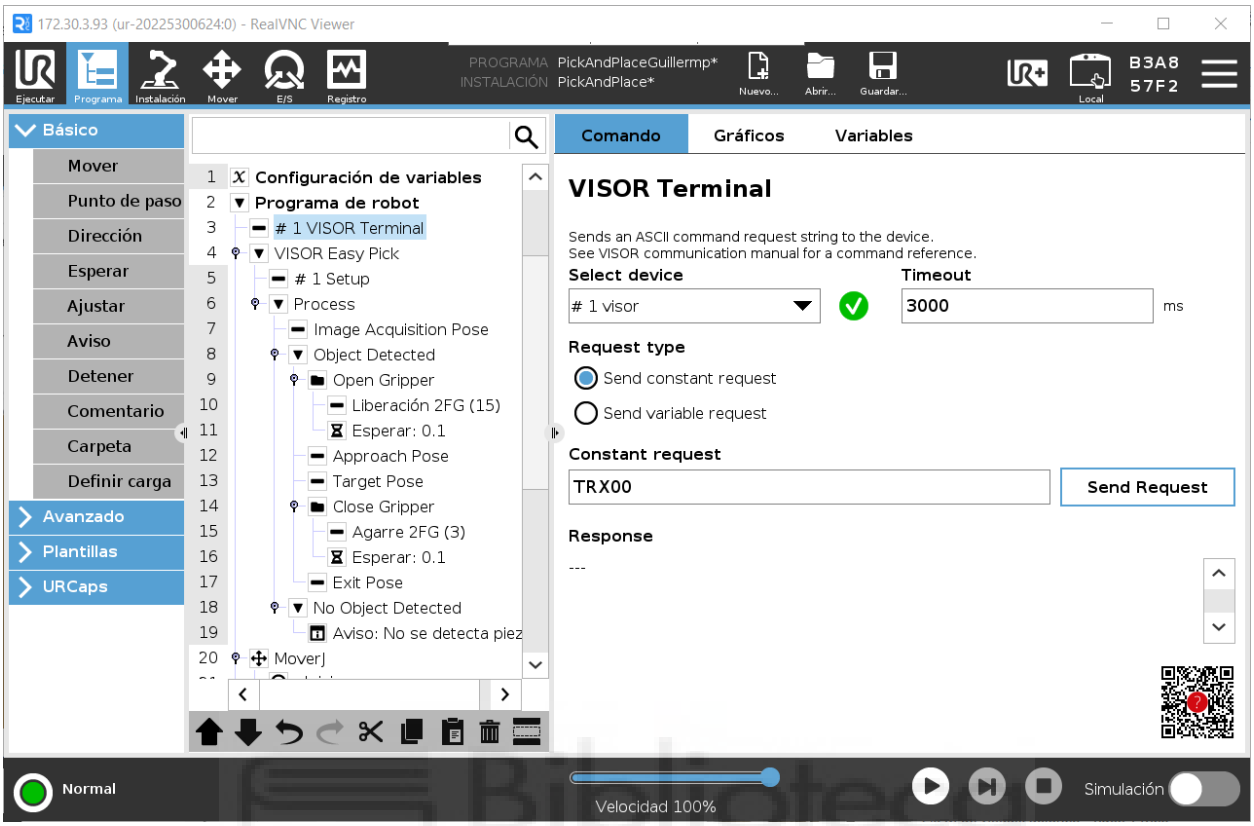


Figura 59. Configuración de la parte de programa relativa al VISOR Easy Pick finalizada. Fuente: Elaboración propia.

En caso de no detectar pieza u objeto desde SensoConfig podemos observar si estamos detectando correctamente el contorno de la pieza que buscamos.

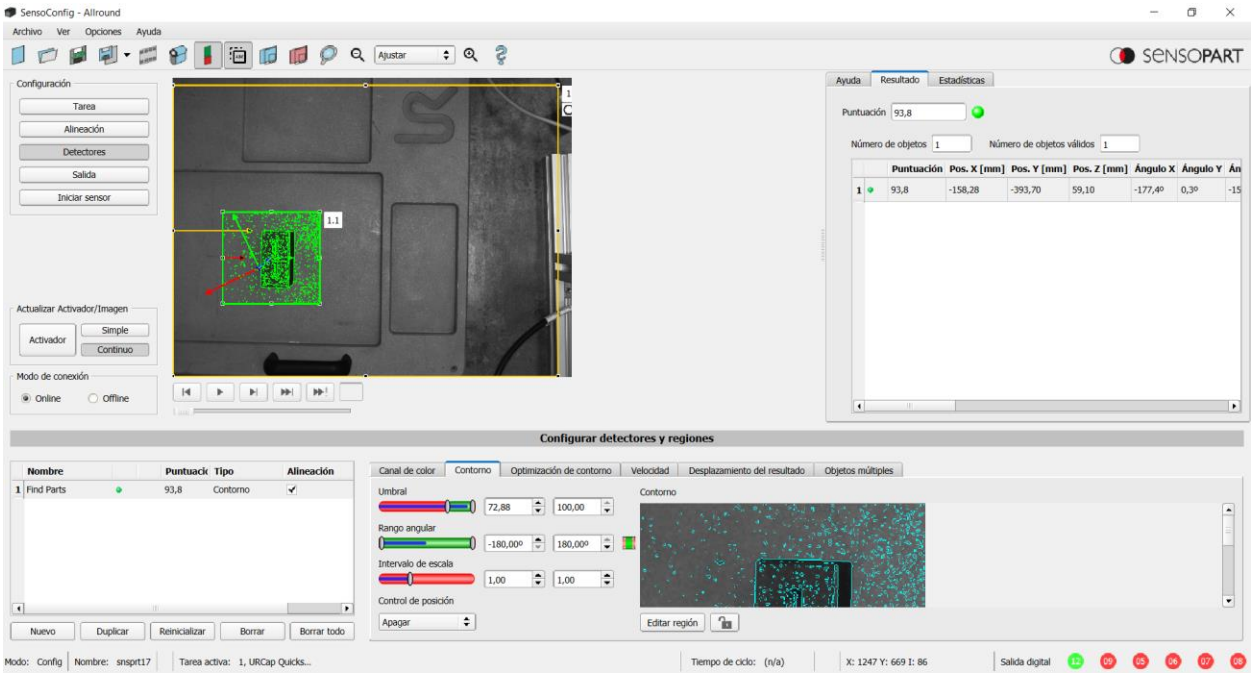


Figura 60. Configuración de los ajustes de umbral, rango angular o intervalo de escala relativos a la tarea Find Parts en SensoConfig. Fuente: Elaboración propia.

Podemos ajustar parámetros de relevancia como el umbral de detección o el rango angular. Este último nos permitirá detectar el contorno de la pieza a pesar de que esta se encuentre girada en cualquier dirección, siempre y cuando se encuentre dentro de nuestra área de trabajo.

Algunas de las piezas a detectar pueden ser de material blanco o transparente, para ello debemos ampliar el umbral con el que vamos a trabajar, pues de lo contrario no serán detectadas.

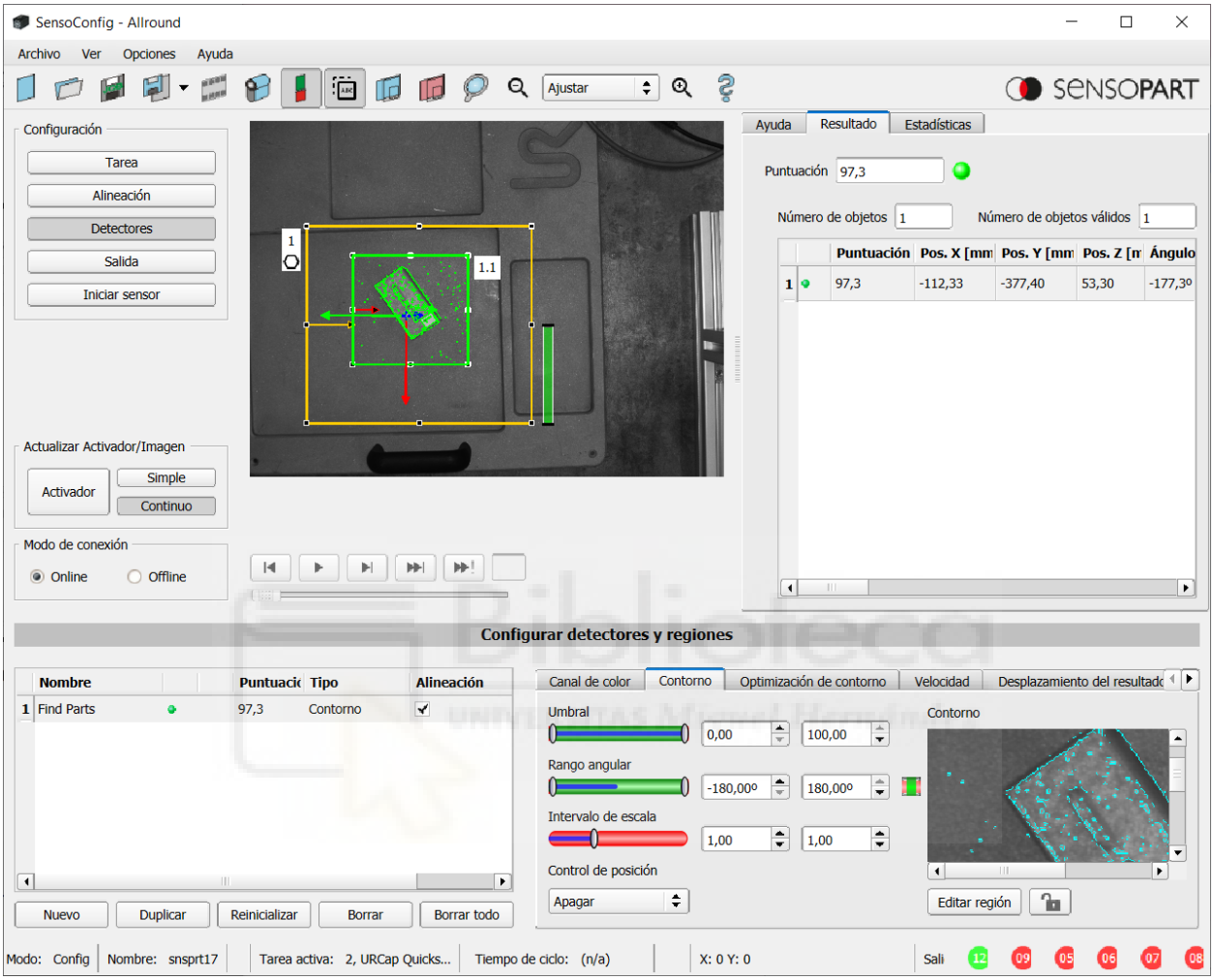


Figura 61. Comprobación de la puntuación obtenida en la detección de un objeto. Fuente: Elaboración propia.

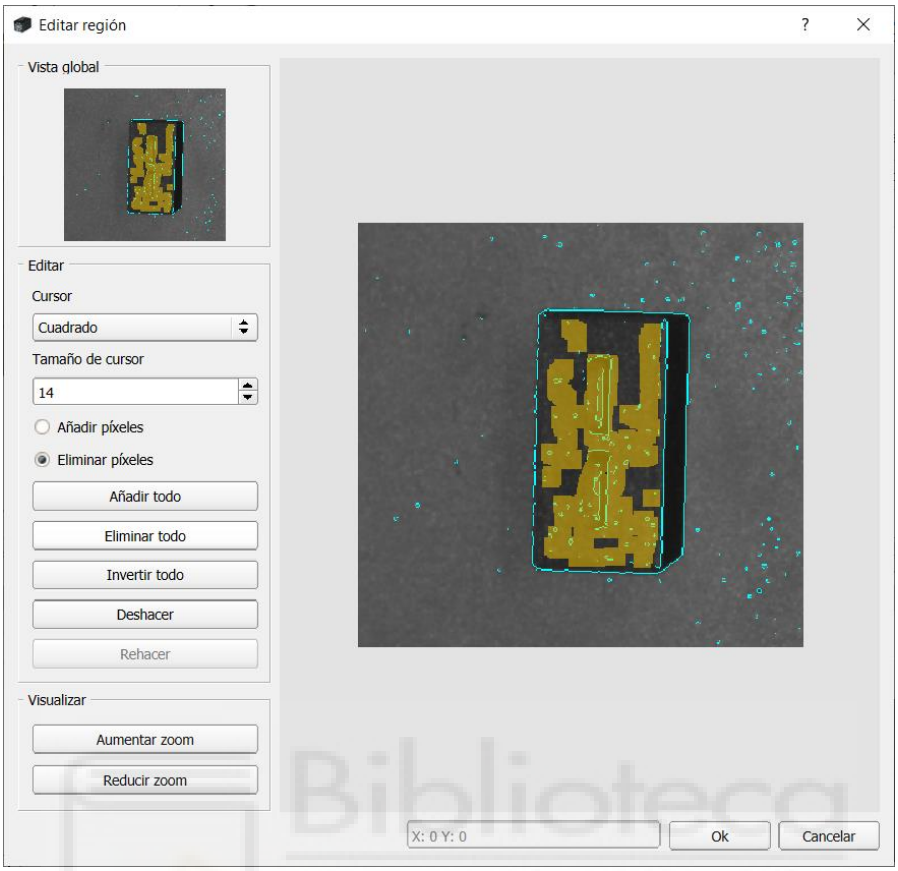


Figura 62. Detalle de la región o contorno de la pieza a detectar. Fuente: Elaboración propia.

También podemos desde editar región eliminar aquellos píxeles que no formen parte de nuestro contorno, facilitando así la detección del mismo.

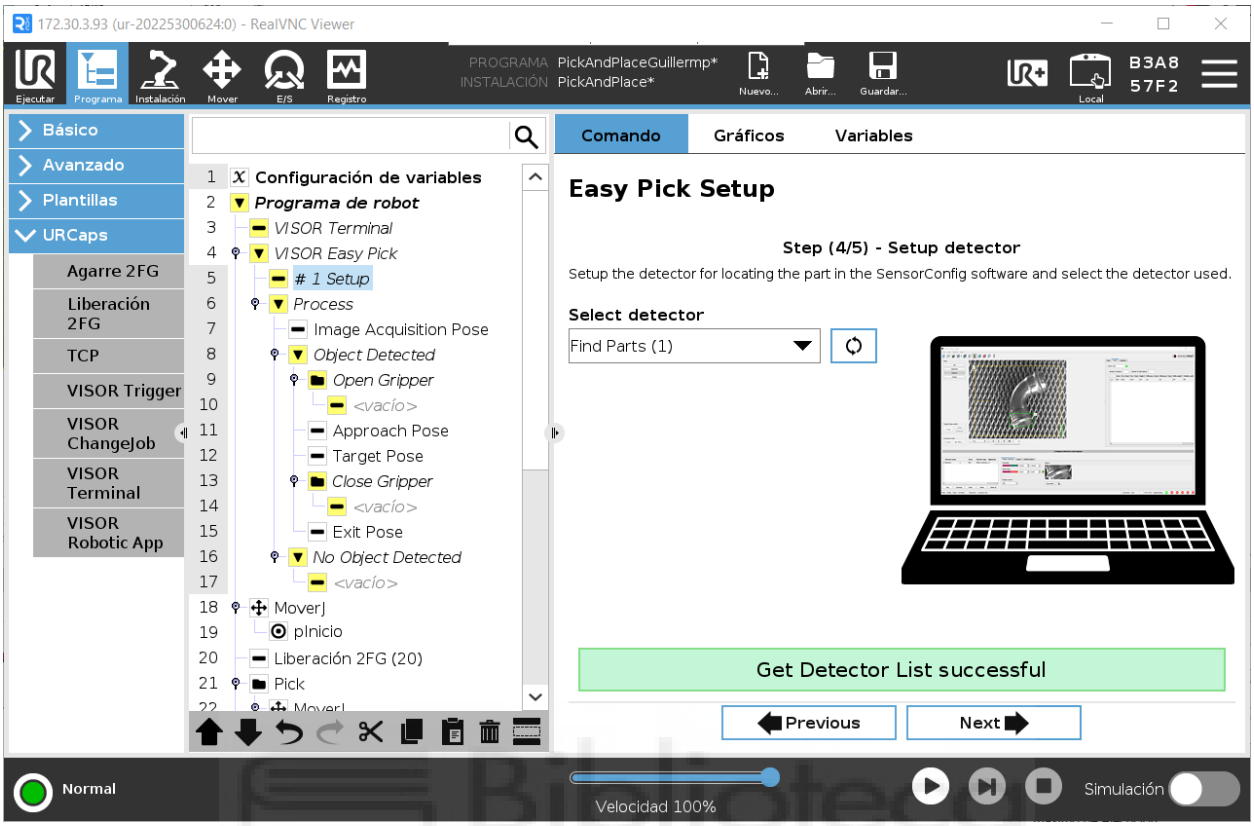


Figura 63. Selección del detector configurado en SensoConfig en nuestro programa del robot UR3e. Fuente: Elaboración propia.

Comprobamos que estamos haciendo uso del detector correcto y terminamos el setup de nuestra tarea VISOR easy pick.

Si leemos el programa, observamos que hay dos caminos posibles cuando el programa se ejecuta, o bien detecta un objeto (o varios), o no lo detecta.

Si detecta objeto, primero debemos configurar una liberación de la pinza (apertura) de 15mm, una espera de 0.1 segundos por seguridad, entonces vienen *Approach pose* y *Target pose*, es decir, las etapas de acercamiento y posicionado de la herramienta del robot sobre la pieza y en el apartado de *Close Gripper* un agarre de 3 mm (el ancho de la lengüeta de nuestra pieza) para sujetar nuestra pieza, por último *Exit pose* indica al robot que debe retirarse hasta la posición de salida.

Por último, si no se detecta pieza hemos decidido configurar un aviso “No se detecta pieza”.

Finalizada nuestra programación base, probamos el funcionamiento de la tarea básica de VISOR Easy pick para ello, un paso crucial es el de “Teach offset”, este paso servirá para colocar el robot sobre la pieza que deseamos coger y servirá de referencia para que una vez el visor detecte el resto de las piezas, la pinza se sitúe exactamente sobre cada una de ellas.

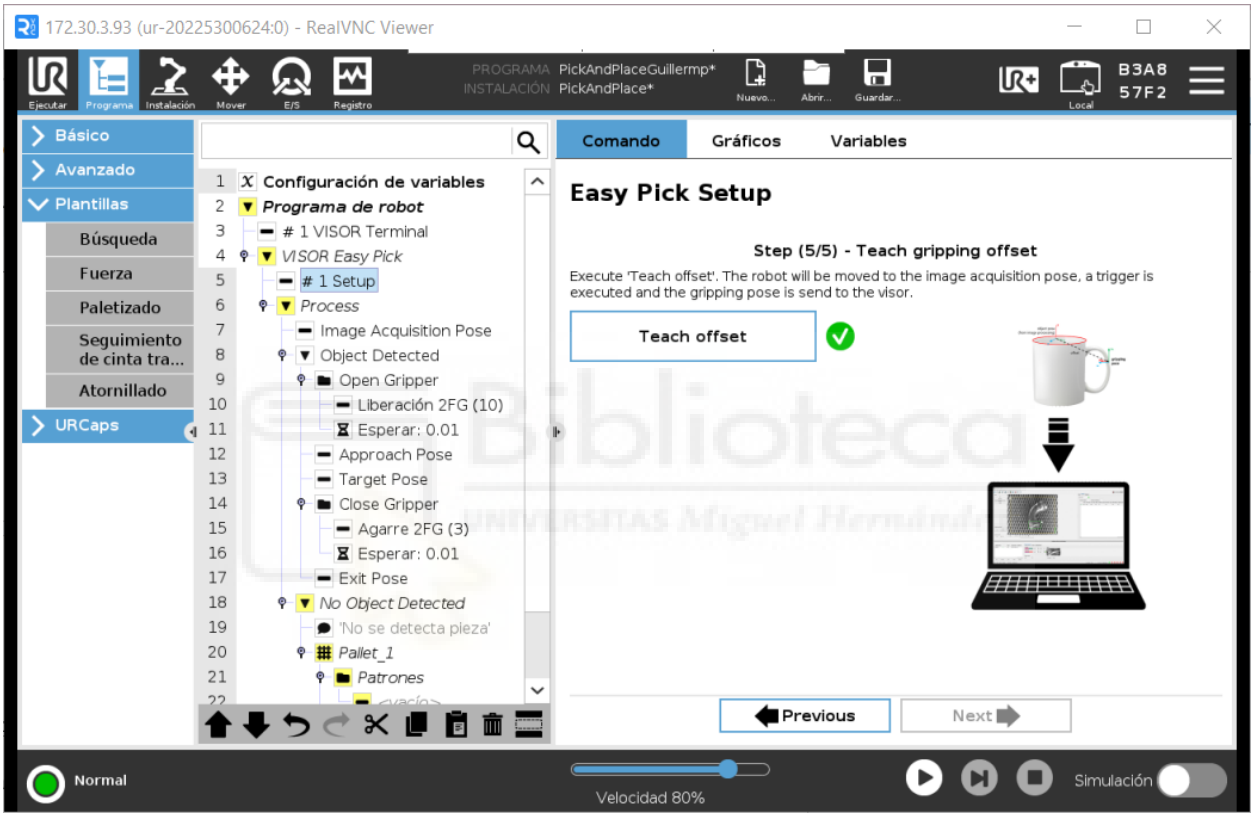


Figura 64. Setup de la tarea. Fuente: Elaboración propia.

Es altamente recomendable realizar este paso con solo una pieza o elemento a detectar.

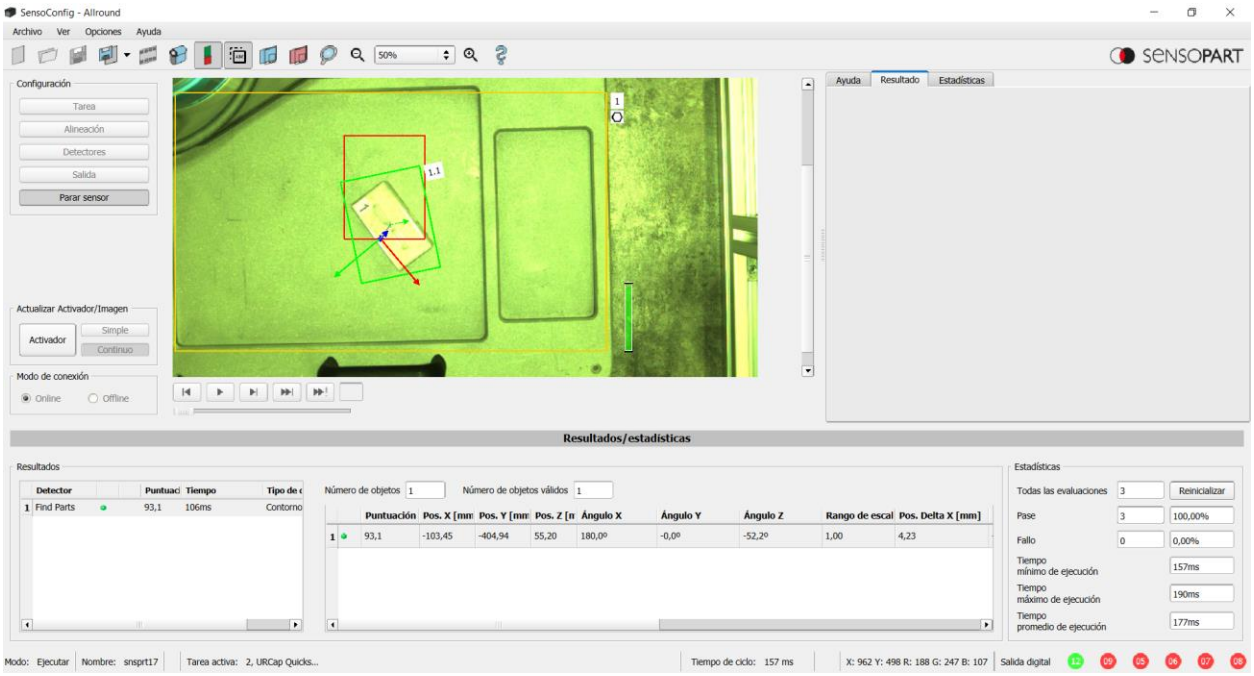


Figura 65. Comprobación de éxito de detección de la tarea. Fuente: Elaboración propia.

Una vez detectado este objeto, activamos el sensor y procedemos a colocar el resto de las piezas que en un entorno real estarían dispersas en nuestra área de trabajo.

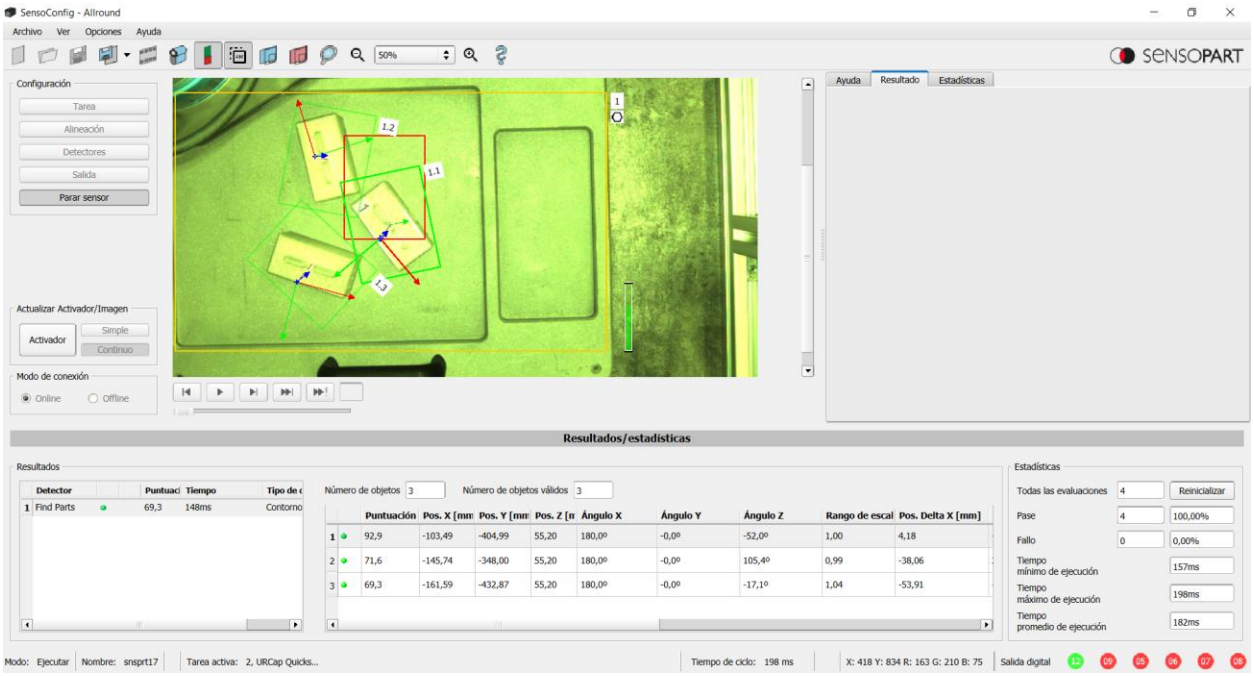


Figura 66.Puntuaciones obtenidas para resto de piezas. Fuente: Elaboración propia.

Podemos observar que nuestra tarea "Find Parts" detecta las tres piezas, es ahora cuando inicializamos el programa y comprobamos cómo el robot las coge una a una para después proceder a su paletizado.

5.2.1. Descripción de la parte del programa correspondiente al paletizado

Dentro del software polyscope en el apartado de programa podemos encontrar diversas plantillas como las de “Fuerza”, “Paletizado”, “Atornillado”, etc.

Estudiamos la de paletizado ya que nos interesa la posibilidad de integrar una solución ya preprogramada por el fabricante para así hacer está aplicación más sencilla y evitar errores de programación haciendo uso de bucles, etc.

Añadimos la plantilla inmediatamente después de nuestra “VISOR robotic app”, es decir, después de nuestra tarea de detección de piezas y recogida de estas por el robot. Ahora el robot ejecutará la secuencia detección de piezas y paletizado de las mismas, pero previamente debemos configurar la plantilla.

Hemos determinado realizar un palet de máximo dos alturas, en primer lugar, vamos a realizar un paletizado lineal, pero también podemos investigar las ventajas de hacer un paletizado rectangular o uno irregular.

Debemos señalar al robot la altura de las piezas pues este parámetro determinará la altura de capas para poder realizar un palet de más de una altura. También es muy recomendable añadir

una acción antes del paletizado, señalamos esta casilla como se muestra en la imagen siguiente.

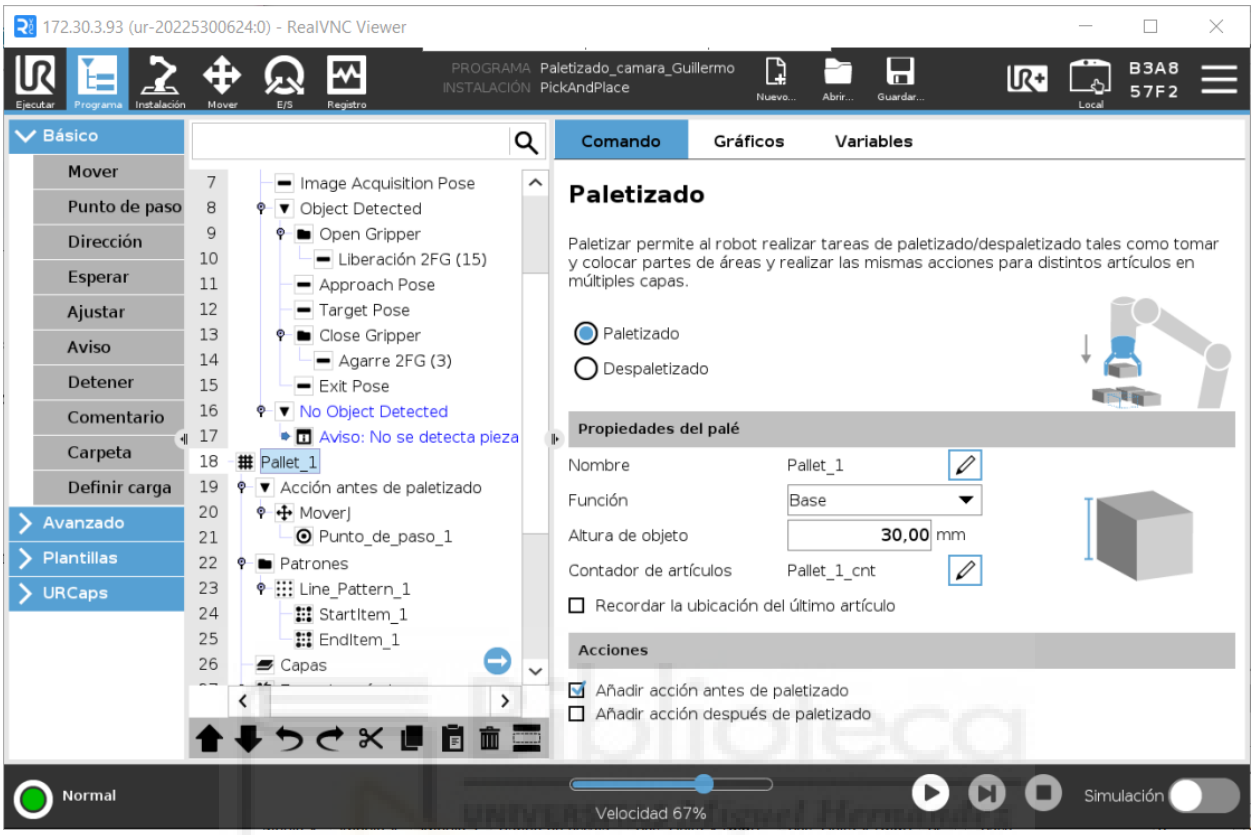


Figura 67.Configuración de la parte de programa correspondiente al paletizado. Fuente: Elaboración propia.

Hemos señalado “Añadir acción antes de paletizado” pues de lo contrario el asistente que nos ayudará a configurar los desplazamientos del robot para depositar las piezas tomará como punto de referencia el primer punto de depósito de las piezas en el palet.

Es por ello por lo que añadimos un punto de paso previo situado por ejemplo un poco más arriba en la vertical del palet para mayor comodidad, es nuestro “Punto_de_paso_1”.

Debemos continuar eligiendo el patrón a seguir por nuestra plantilla de paletizado, por simplicidad se ha escogido un patrón lineal pero los otros patrones no presentan mayor complejidad para su realización.

Podemos observar en la siguiente imagen la opción “Separador”, esto sería en el caso de existir algún tipo de elemento que dividiera las alturas del palet.

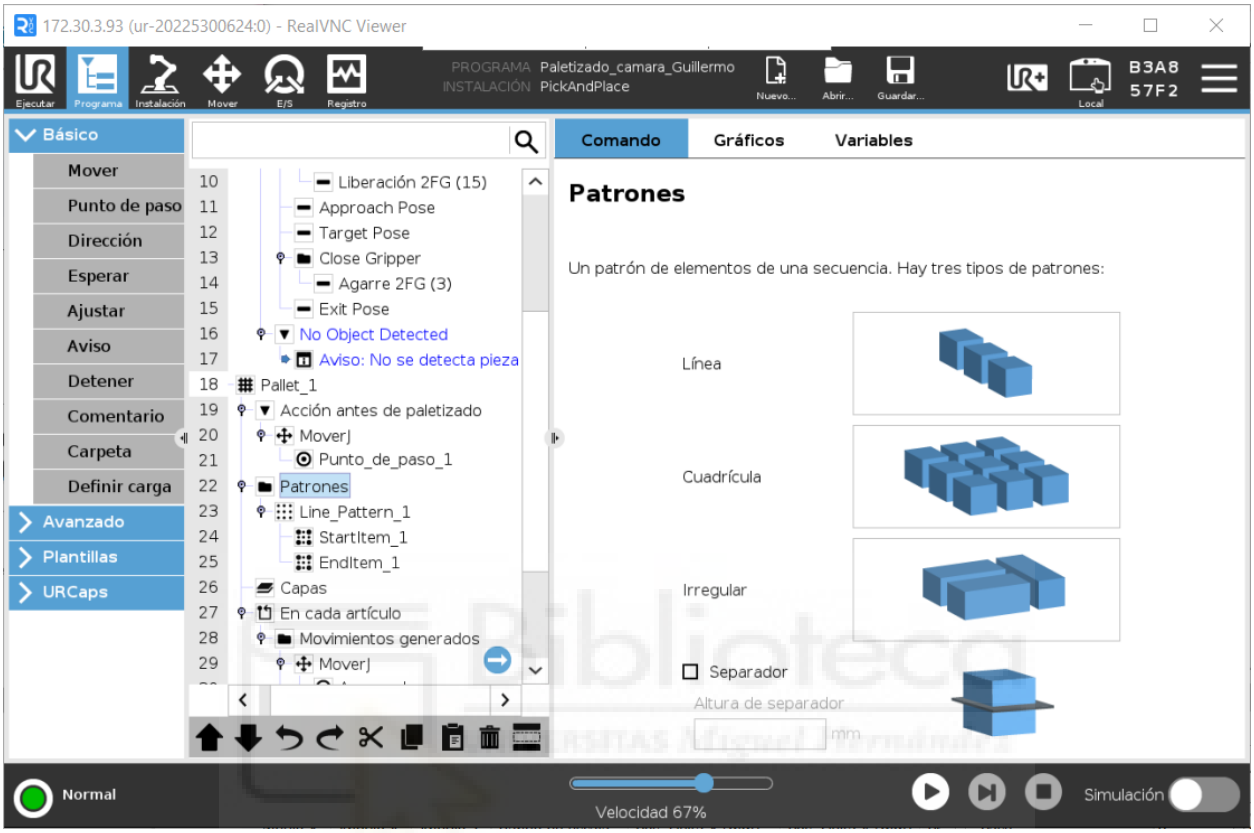


Figura 68. Selección del patrón a utilizar. Fuente: Elaboración propia.

A continuación, debemos especificar cuántos elementos vamos a paletizar en linea, dado que mi palet impreso en 3D tiene la capacidad justa para albergar 4 piezas de frente, decidimos dar algo de margen al robot y seleccionar un paletizado de 4 piezas en línea.

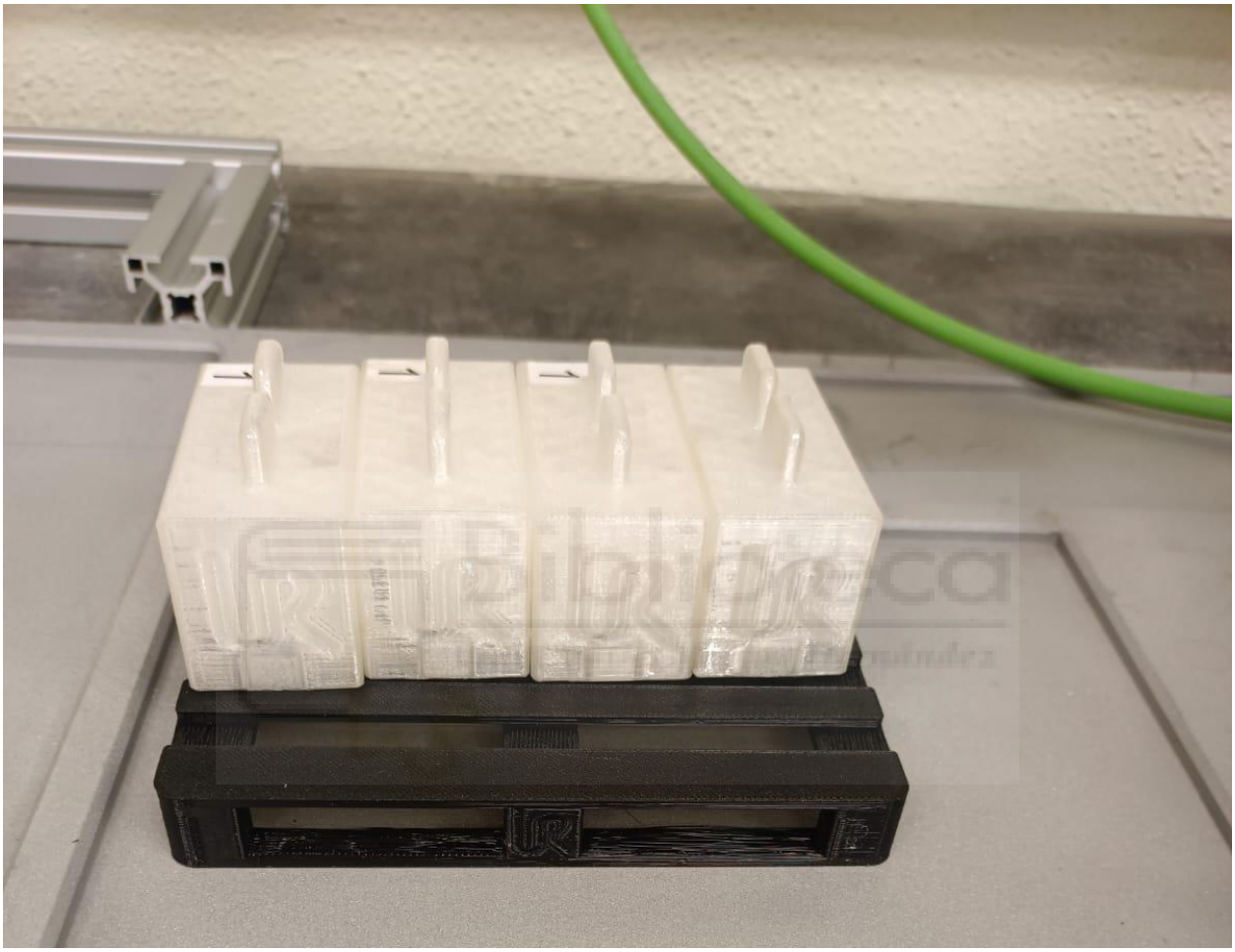


Figura 69. Resultado del paletizado en línea. Fuente: Elaboración propia.

Tenemos en cuenta que en una aplicación real es fundamental aprovechar al máximo el espacio disponible dentro del palet y por ello seleccionamos esta configuración.

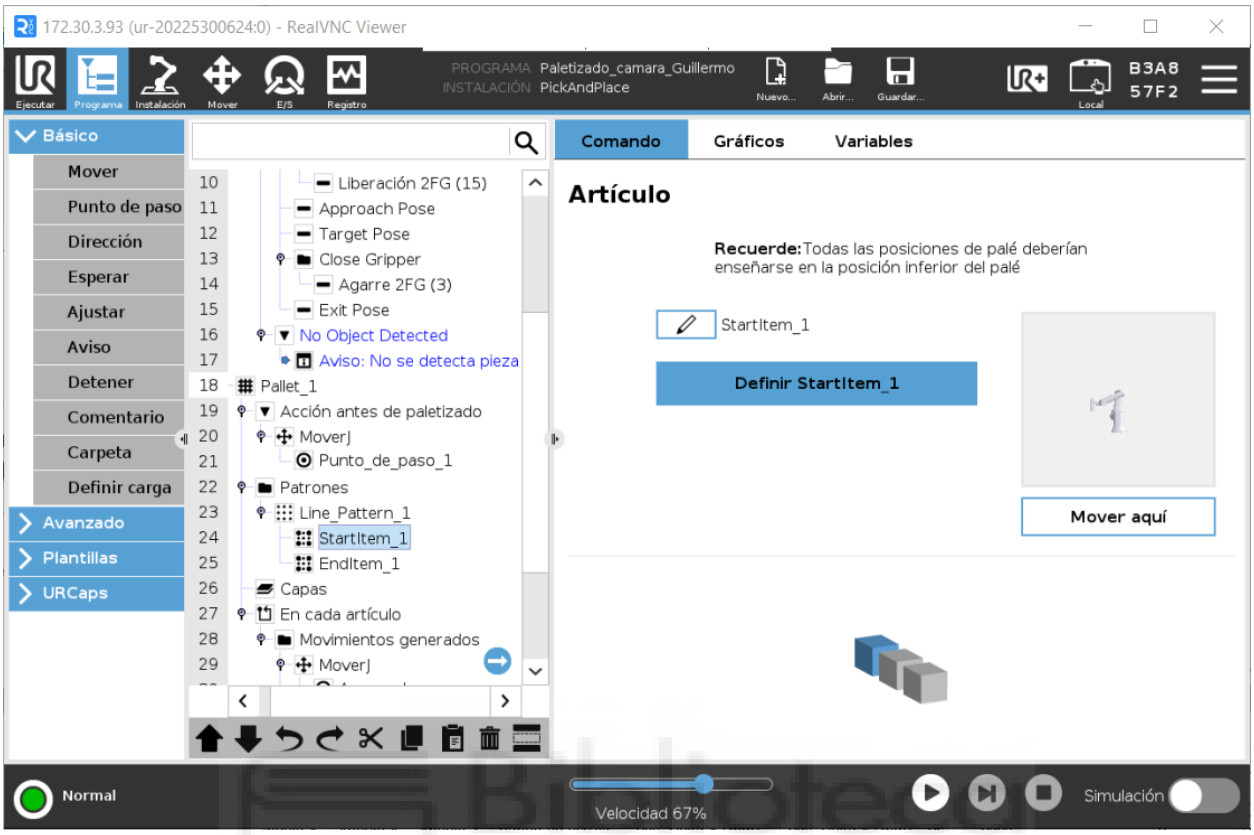


Figura 70. Definición de los puntos de "place" del primer y último objeto. Fuente: Elaboración propia.

Ahora debemos definir las posiciones inicial y final, será nuestro robot el que se encargará de determinar las posiciones intermedias de manera exacta y así es como obtendremos un paletizado similar a este.

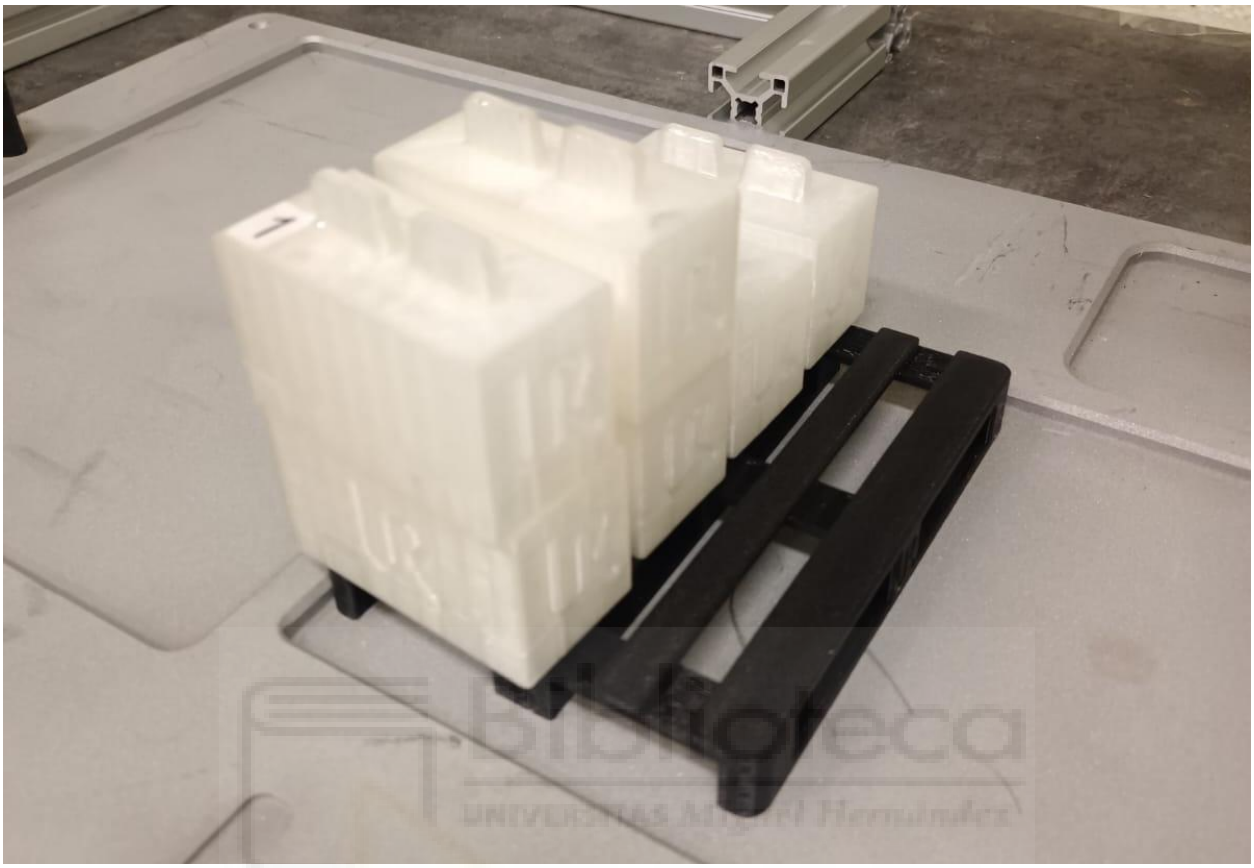


Figura 71. Paletizado de dos alturas. Fuente: Elaboración propia.

Debemos ahora señalar nuestro número de capas y si el paletizado será el mismo para cada una de ellas. En este caso sí, pues de lo contrario no podemos apilar nuestras piezas. Pero en otros muchos

casos reales es fundamental poder establecer diferentes capas con diferentes patrones de paletizado para garantizar una distribución de pesos efectiva en un palet.

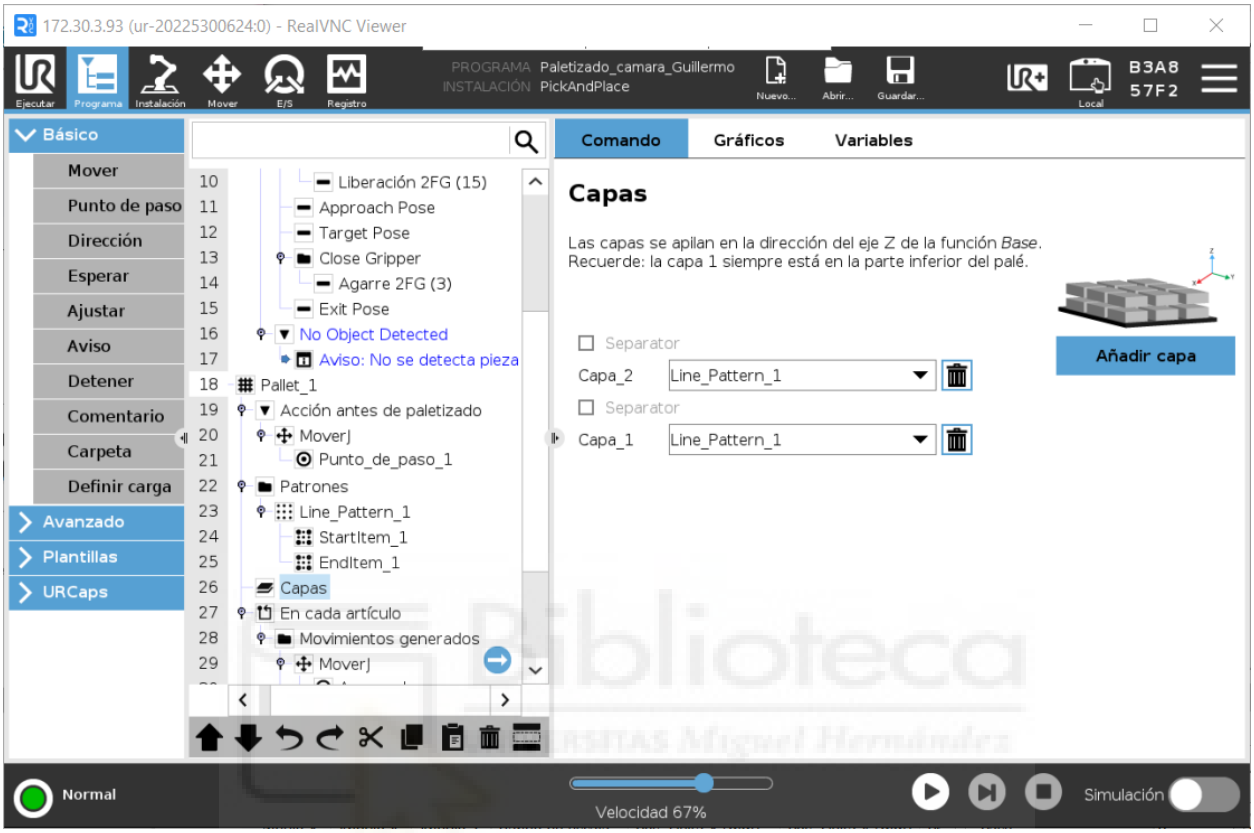


Figura 72. Definición del número de capas del paletizado. Fuente: Elaboración propia.

Ahora solo queda establecer los movimientos necesarios para el paletizado, dentro de "En cada artículo" nos aparecerá un asistente que nos ayuda a establecer los tres movimientos principales del paletizado.

El punto de acercamiento o "Approach", que definiremos con un MoverJ, el punto de "ToolActionPoint" que definiremos con un Mover y en el que insertamos una acción de liberación de la pinza para depositar nuestra pieza sobre el palet y por último, el punto de salida que definimos con un MoverL.

Como recomendación debemos ajustar estos puntos pensando siempre en las trayectorias de entrada y salida del brazo robot y la pinza ya que si establecemos puntos que requieran de grandes desplazamientos en X o en Y posiblemente se producirán colisiones con la carga.

5.3. Integración del sistema completo

5.3.1. Configuración y programación del PLC

Disponemos de un PLC S7-1200 físico, por lo que al acceder a Tia Portal el primer paso será agregar este dispositivo, pero en lugar de consultar el tipo de CPU exacto (1214C AC/DC/Rly), lo que hacemos es seleccionar CPU 1200 “sin especificar” y utilizamos la herramienta de detección de equipos en línea de Tia Portal.

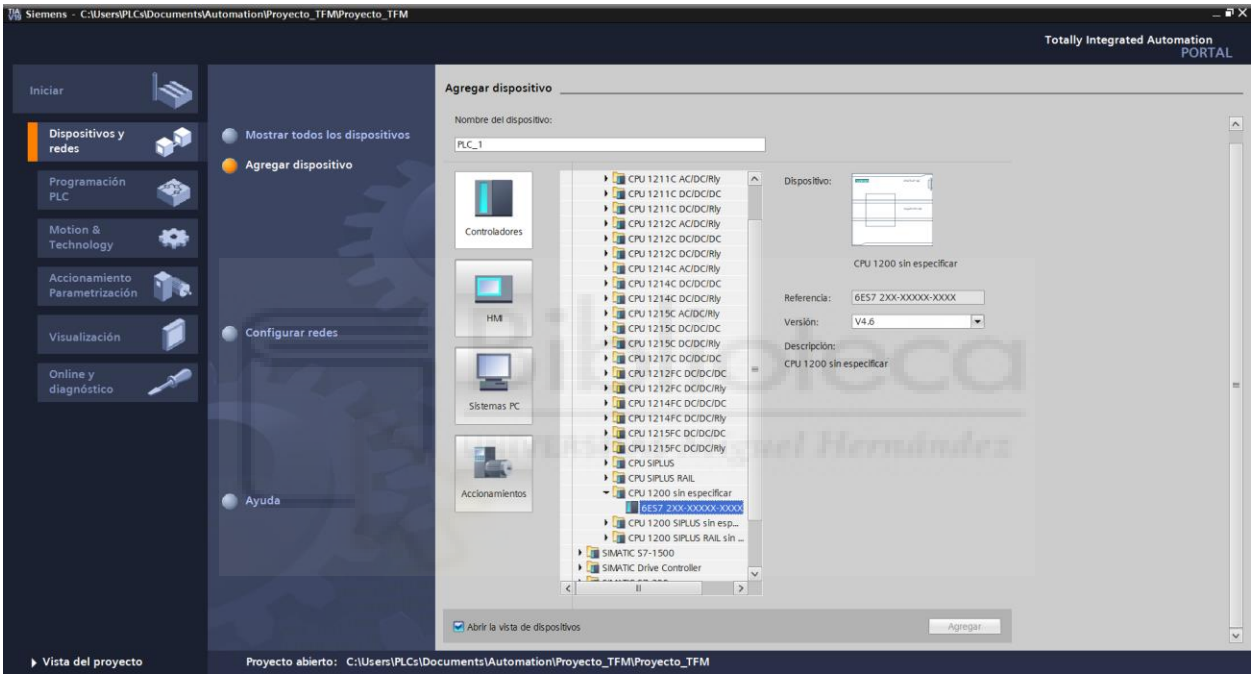


Figura 73. Selección del dispositivo CPU 1200 y posterior detección del mismo en Tia Portal. Fuente: Elaboración propia.

Una vez detectado, accedemos a Vista del proyecto y nos aparecerán los principales ajustes de seguridad que el fabricante Siemens nos recomienda. La primera es la protección de los datos de configuración del PLC por medio de contraseña.

En este proyecto, al tratarse de un proyecto personal y que no tiene fines empresariales o sensibles lo dejamos por defecto.

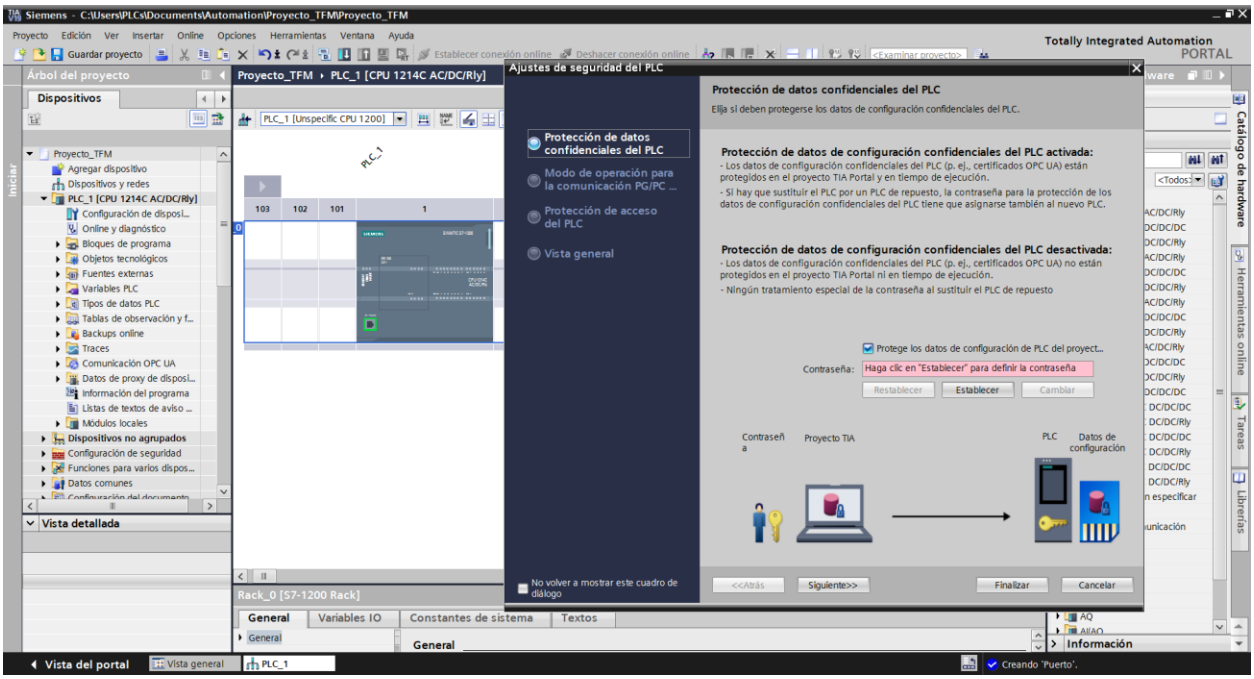


Figura 74. Configuración de las opciones de seguridad del proyecto. Fuente: Elaboración propia.

Por el mismo motivo, deshabilitamos el check de “Permitir sólo la comunicación PG/PC y HMI segura”, de este modo estamos permitiendo las comunicaciones entre HMI y PLC no cifradas por TLS y certificados. Esto es un ajuste que aplica principalmente a HMIs antiguos, y en nuestro caso, no aplica.

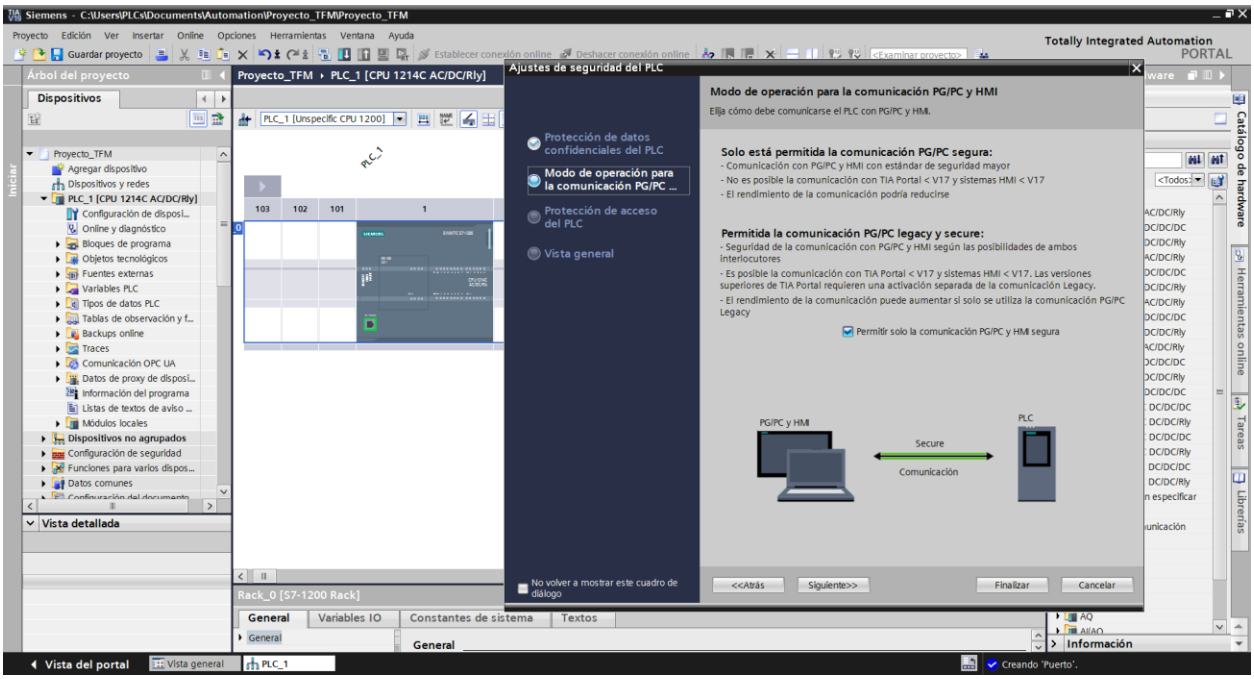


Figura 75. Modos de comunicación entre HMI y PLC. Fuente: Elaboración propia.

Finalmente, el último ajuste de configuración es la protección de acceso al PLC, esta es una característica de seguridad imprescindible para aplicaciones industriales dentro del entorno

empresarial, pero una vez más, no nos aplica debido a la naturaleza de este proyecto.

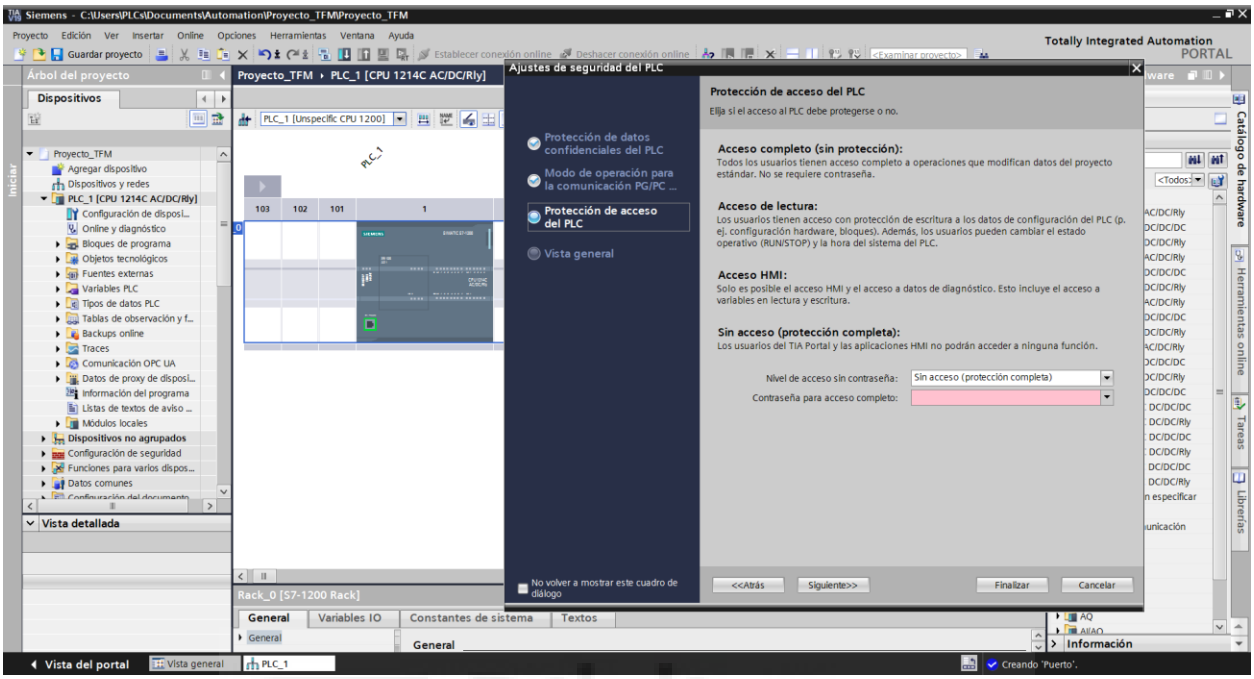


Figura 76. Configuración de la contraseña del proyecto. Fuente: Elaboración propia.

Pasamos al entorno de Polyscope, desde el menú de Instalación podemos acceder por el árbol de la izquierda a la pestaña “Bus de campo”, aquí debemos seleccionar la opción PROFINET y habilitarla.

Se recuerda que PROFINET es el protocolo de comunicación por defecto de los PLCs Siemens, este está basado en Ethernet. Este es un protocolo abierto diseñado para la comunicación de campo con diferentes dispositivos de I/O y HMI.

En nuestro caso, el dispositivo al que nos conectaremos es nuestro robot UR3e.

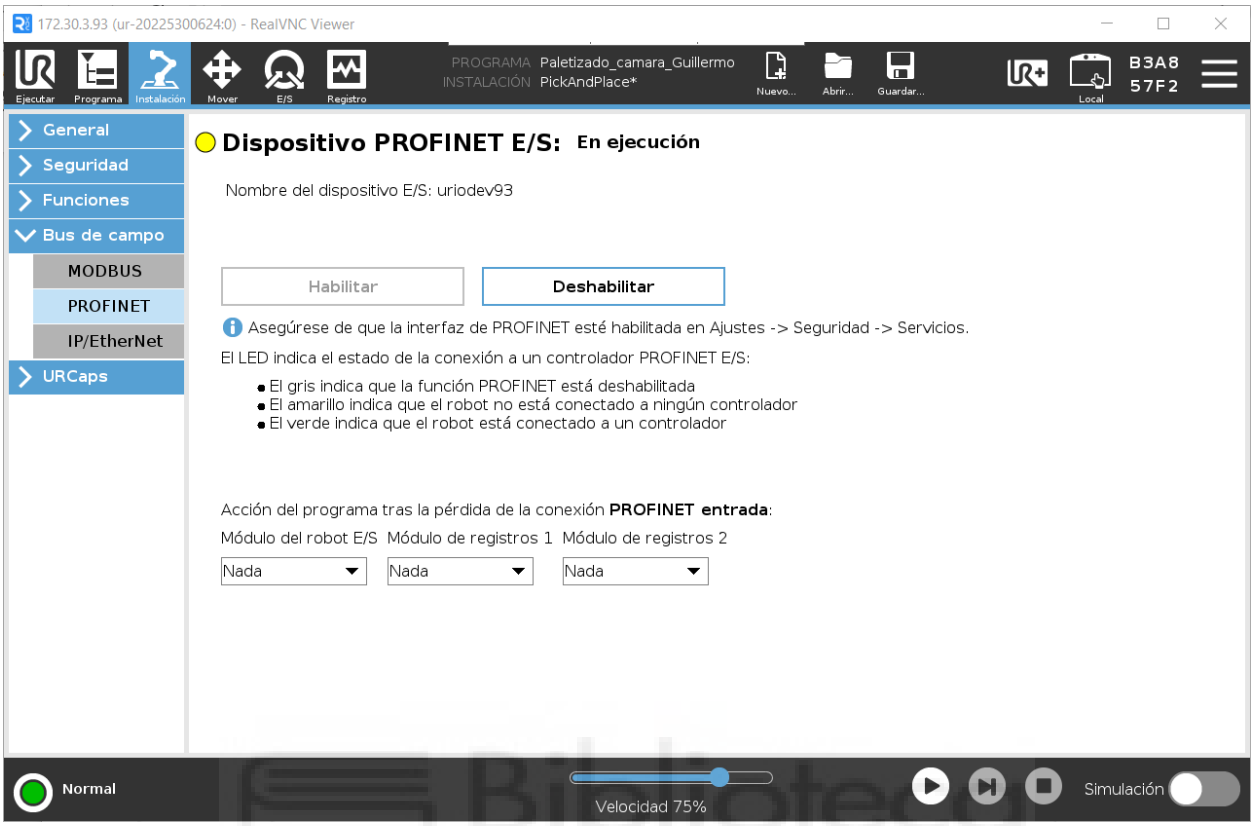


Figura 77. Habilitación de la comunicación por protocolo PROFINET entre PLC y robot. Fuente: Elaboración propia.

Cómo indica la imagen el verde indica que el robot está conectado a un controlador. Si se encuentra en amarillo quiere decir que el bus de campo está habilitado, pero no tiene un maestro asignado. Procedemos a comprobar que el robot se encuentra en red, para ello seleccionamos en tipo de interfaz la interfaz PN/IE y nuestra tarjeta de red, en mi caso la Realtek PCIe GbE Family Controller, entre muchos otros dispositivos localizamos el así llamado en el proyecto uriodev93 (pues su dirección IP es la 172.30.3.93).

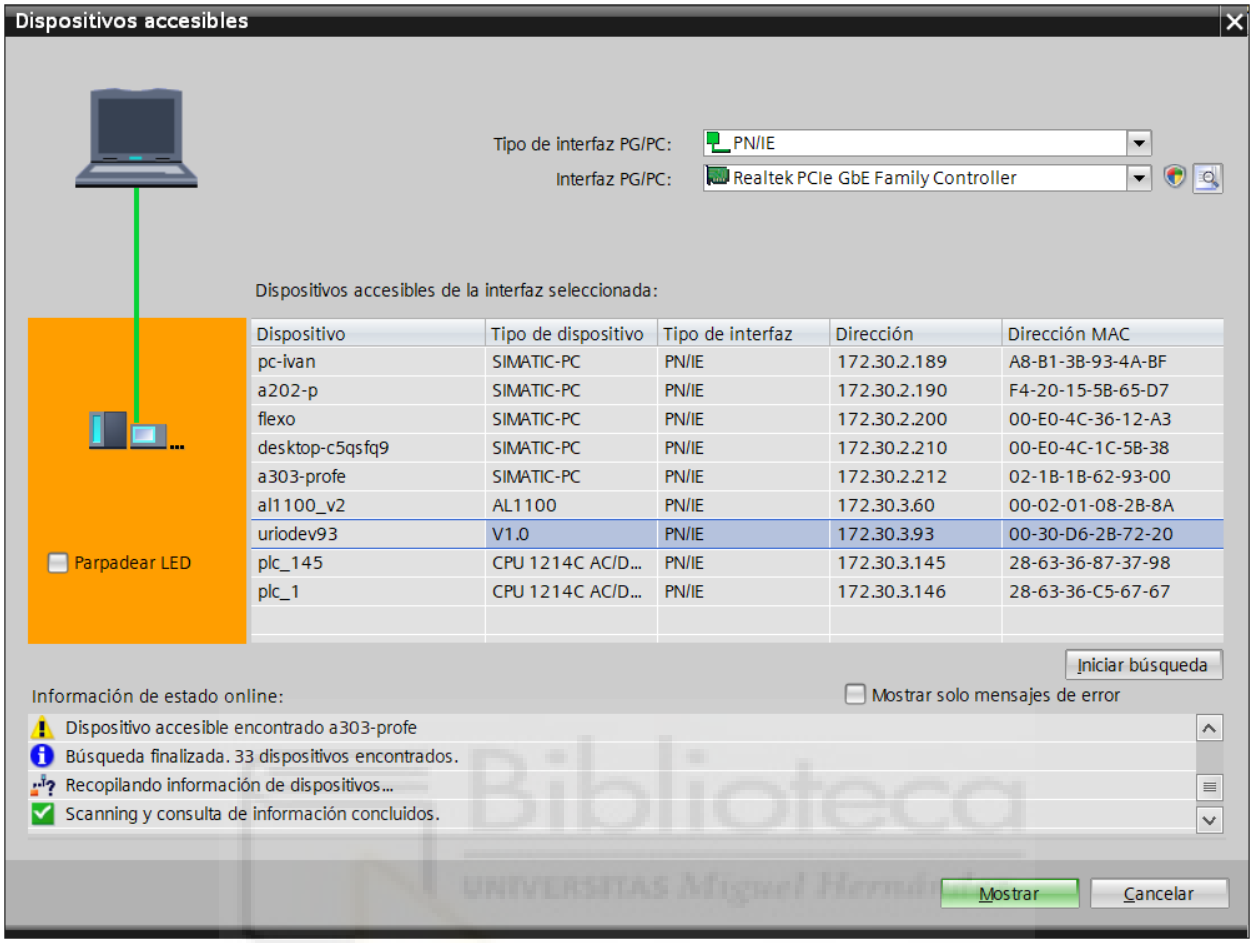


Figura 78. Detección de nuestro robot UR3e con IP:172.30.3.93 en Tia Portal. Fuente: Elaboración propia.

Desde la barra principal de Tia Portal, en concreto desde opciones, seleccionamos la opción de "Administrar archivos de descripción de dispositivos" e instalamos los archivos GSD necesarios para poder añadir nuestro robot al proyecto.

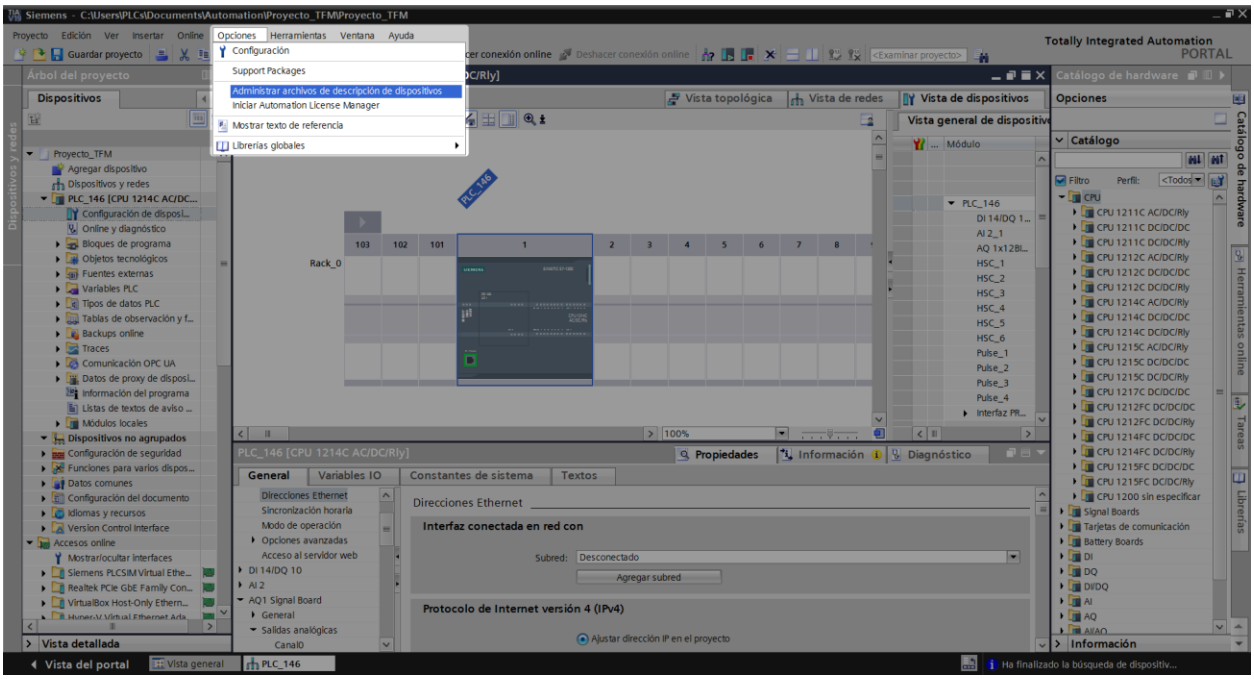


Figura 79. Detalle de configuración de los módulos de nuestro controlador robot. Fuente: Elaboración propia.

Es desde la página web del fabricante desde donde descargamos nuestro archivo GSD [7].

Este es un paso crítico incluso tras haber instalado la GSD del robot por primera vez, ya que cuando transcurra el tiempo es muy posible que debamos actualizar los controladores para no incurrir en estados de error en el PLC.

Esto puede ocurrir y ocurrió durante el desarrollo de este trabajo, por ejemplo cuando un fabricante actualiza los GSDs de un dispositivo.

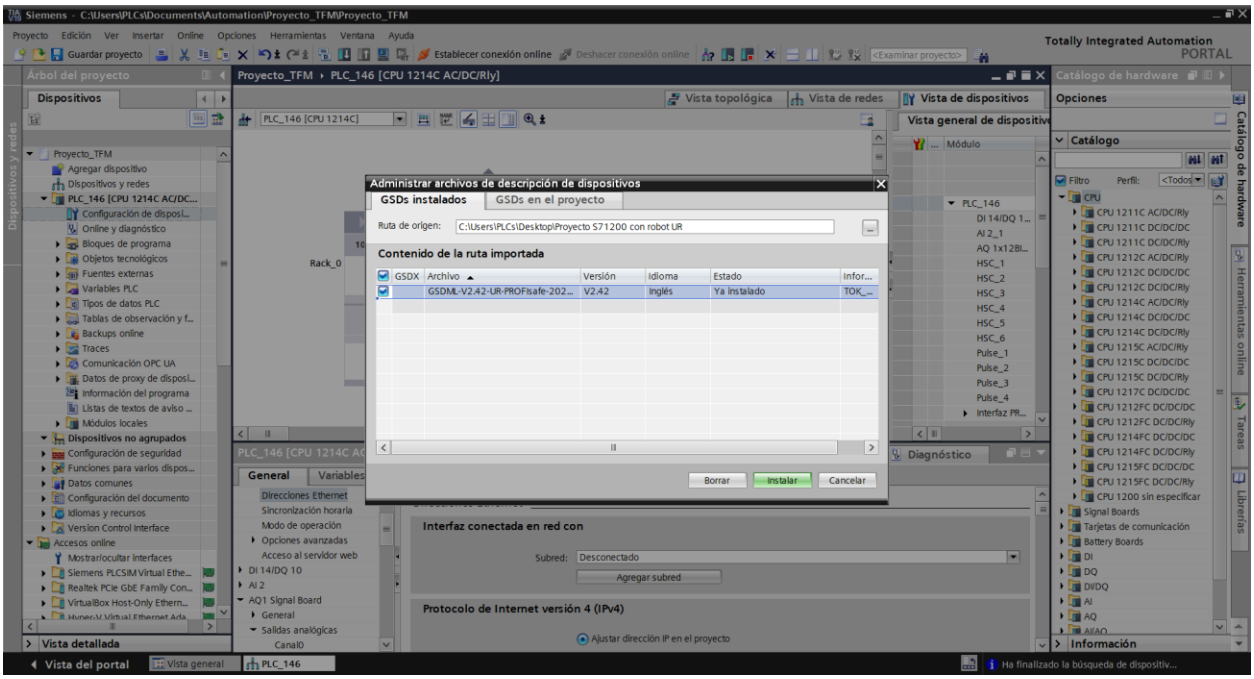


Figura 80.Instalación del arcivo GSD para el control del robot UR3e por PROFINET. Fuente: Elaboración propia.

Tras hacer esto, se actualizará nuestro catálogo de hardware y desde el árbol del catálogo de la derecha encontraremos nuestro robot en Otros dispositivos de campo, dentro de Otros dispositivos Ethernet, dentro de PROFINET IO, dentro de I/O, Universal Robots y finalmente en Collaborative robots.

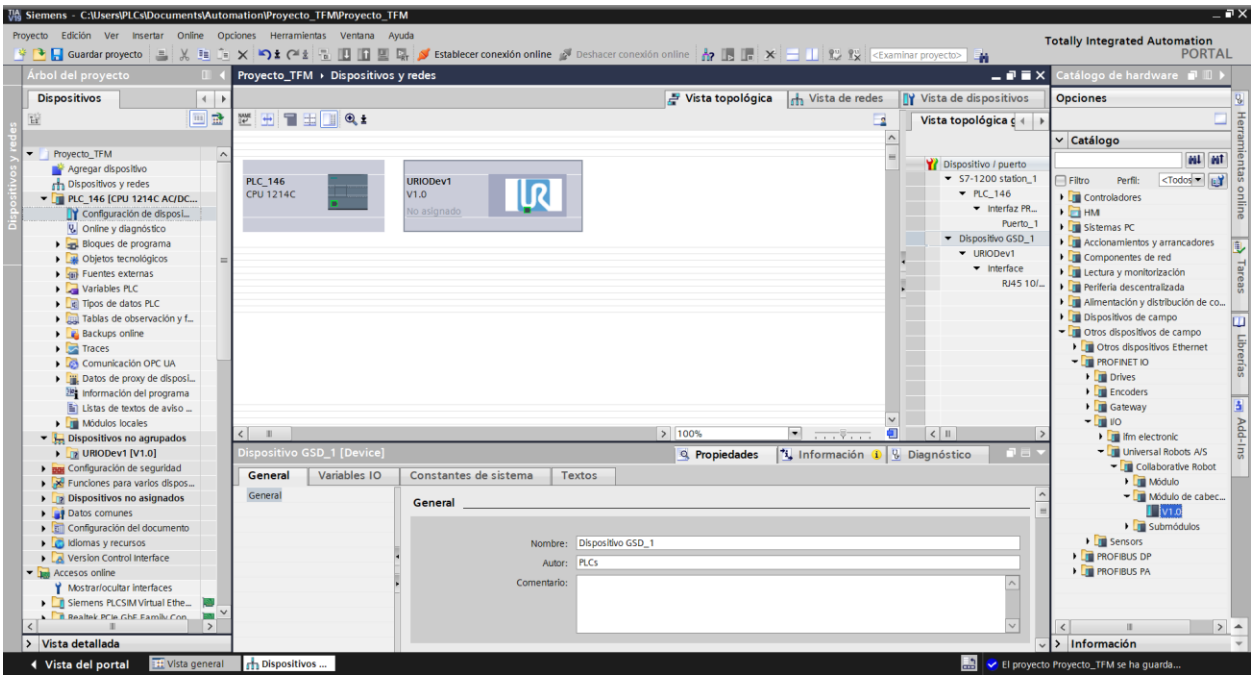


Figura 81. Vista topológica con el hardware de nuestro módulo robot insertado. Fuente: Elaboración propia.

Una vez podamos visualizar ambos en nuestra pestaña de Vista topológica, hacemos click en el puerto de comunicación Ethernet del robot y lo arrastramos a nuestro PLC_146 (recordamos que recibe este nombre por su dirección IP: 172.30.3.146).

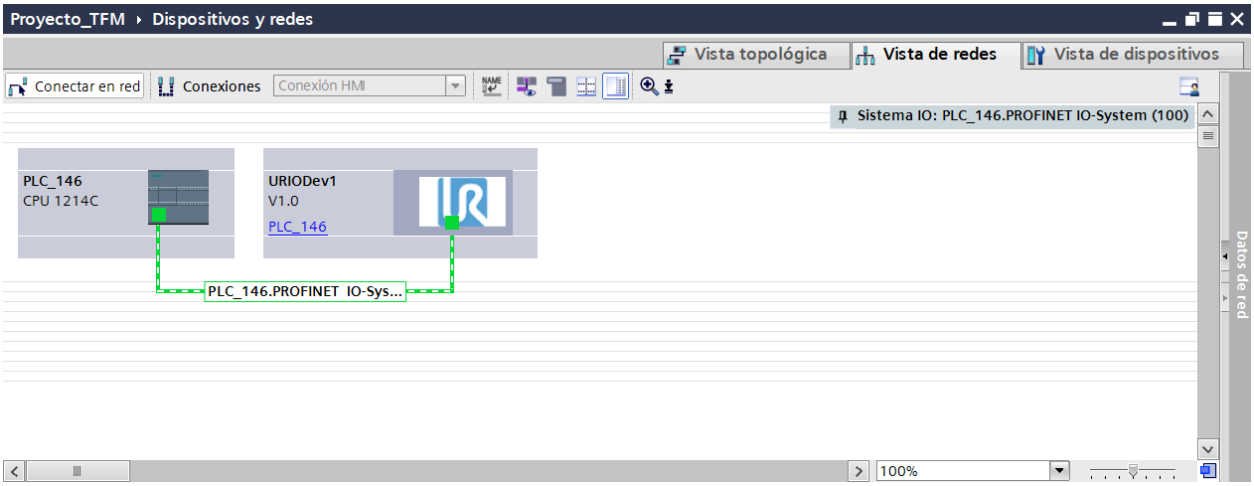


Figura 82. Vista de la conexión Cliente-Servidor configurada entre PLC y el robot. Fuente: Elaboración propia.

Aprovechamos para comprobar que la interfaz PROFINET se encuentra habilitada en la pestaña de *Ajustes,* aunque de no estarlo ya habríamos observado fallos de comunicación en el software TIA PORTAL.

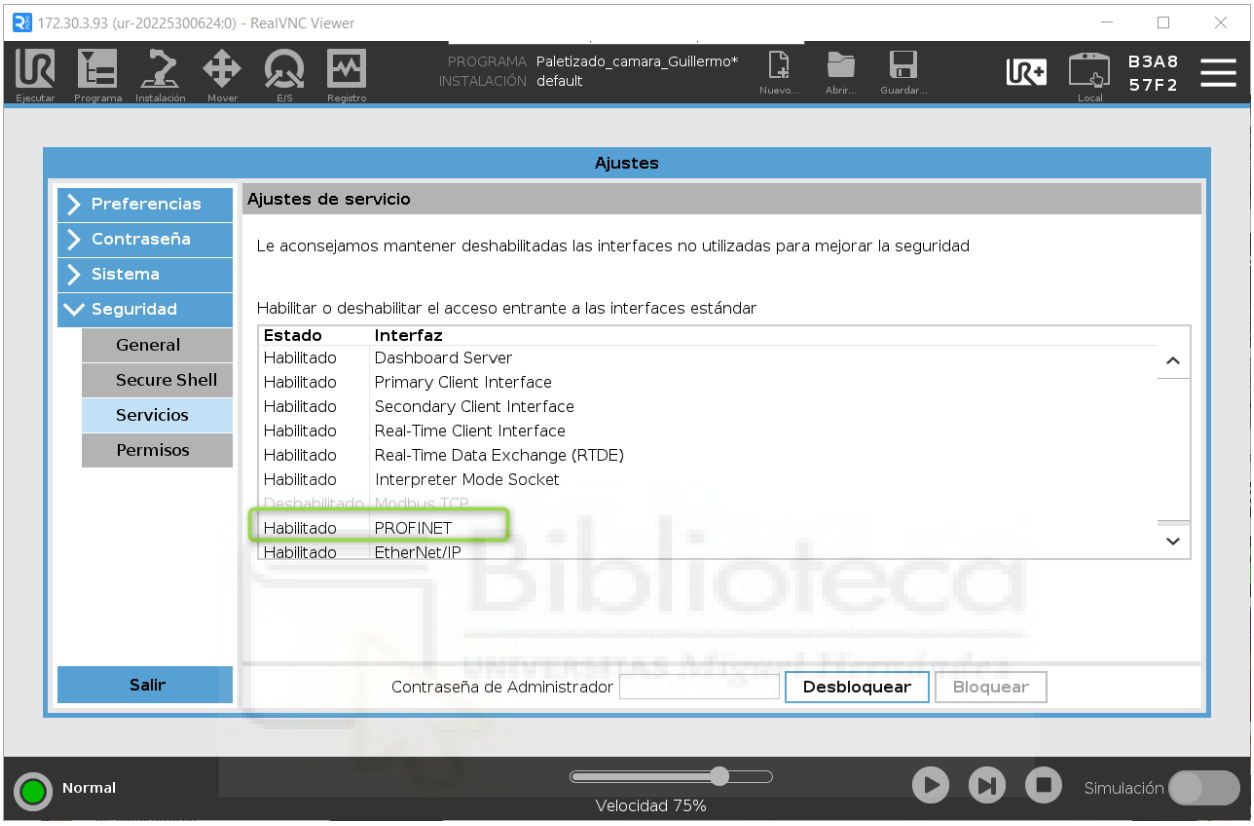


Figura 83. Comprobación del estado de la conexión. Fuente: Elaboración propia.

Una vez conectados, (o antes), si así lo vemos conveniente debemos definir la dirección IP de nuestro robot UR, recordamos que es la 172.30.3.93 y cómo máscara de subred la 255.255.254.0.

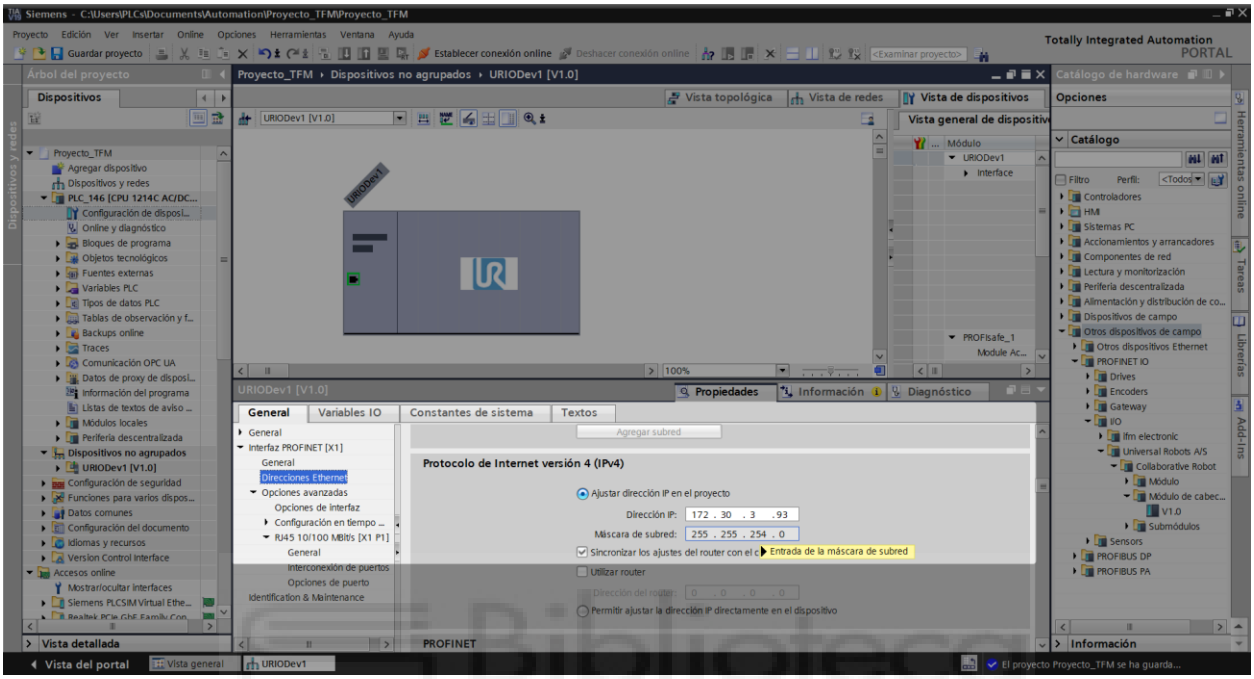


Figura 84. Asignación de la dirección IP a nuestro robot en el proyecto de Tia Portal. Fuente: Elaboración propia.

Por último, observamos que el dispositivo uriodev93 (nombrado así por su IP: 172.30.3.93) está conectado desde la pestaña de *Instalación* y desde la subpestaña *Bus de Campo, PROFINET.*

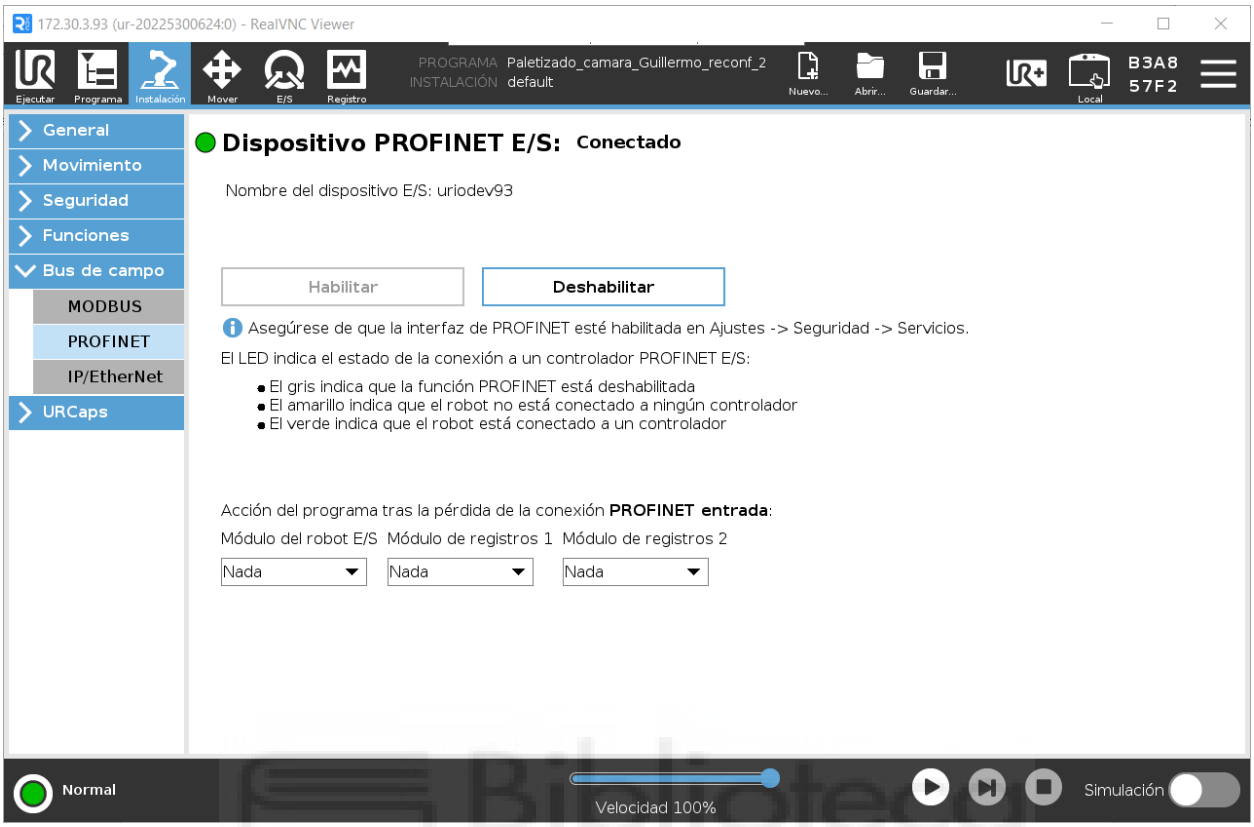


Figura 85. Comprobación de la conexión desde el software polyscope. Fuente: Elaboración propia.

Accedemos a la vista general de dispositivos y allí, vamos a arrastrar los diferentes módulos que necesita para funcionar del catálogo de hardware los añadimos todos uno a uno y comprobamos que las direcciones que se asignen en las entradas no se "solapen" con las ya existentes del PLC. Una práctica recomendable para ello es consultar primero qué direcciones ya se encuentran ocupadas y comenzar a escribir desde una entrada que permita un rango amplío de continuidad (por motivos organizativos).

En nuestro caso, las entradas comienzan desde la I100 a la I579, podemos observar que el nombre de todas ellas comienza con T2O, lo que significa "Tool to operator", es decir, Herramienta a operador, o lo que es lo mismo, entradas del robot al PLC. Por lo que la activación de determinadas

salidas en nuestro robot, conlleva la activación de unas entradas en nuestro PLC.

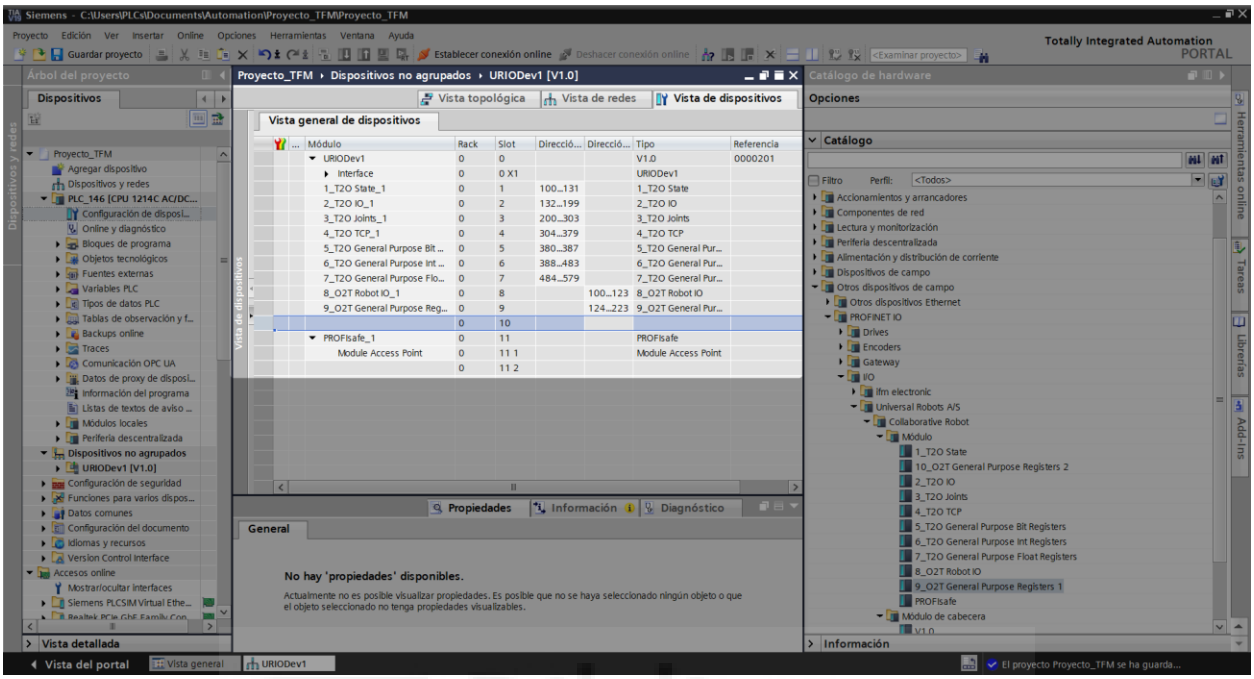


Figura 86. Asignación de direcciones a las entradas y salidas de los diferentes módulos del controlador de nuestro robot. Fuente: Elaboración propia.

Mientras que las salidas ocupan las direcciones Q100 a Q223, y su nombre comienza por “O2T”, lo que significa “Operator to Tool”, es decir, del operador a la herramienta. En nuestro caso, el operador es el PLC y la herramienta el robot. Por lo que la activación de las salidas Q100 a Q223 en nuestro PLC corresponderá con la activación de unas entradas en nuestro robot.

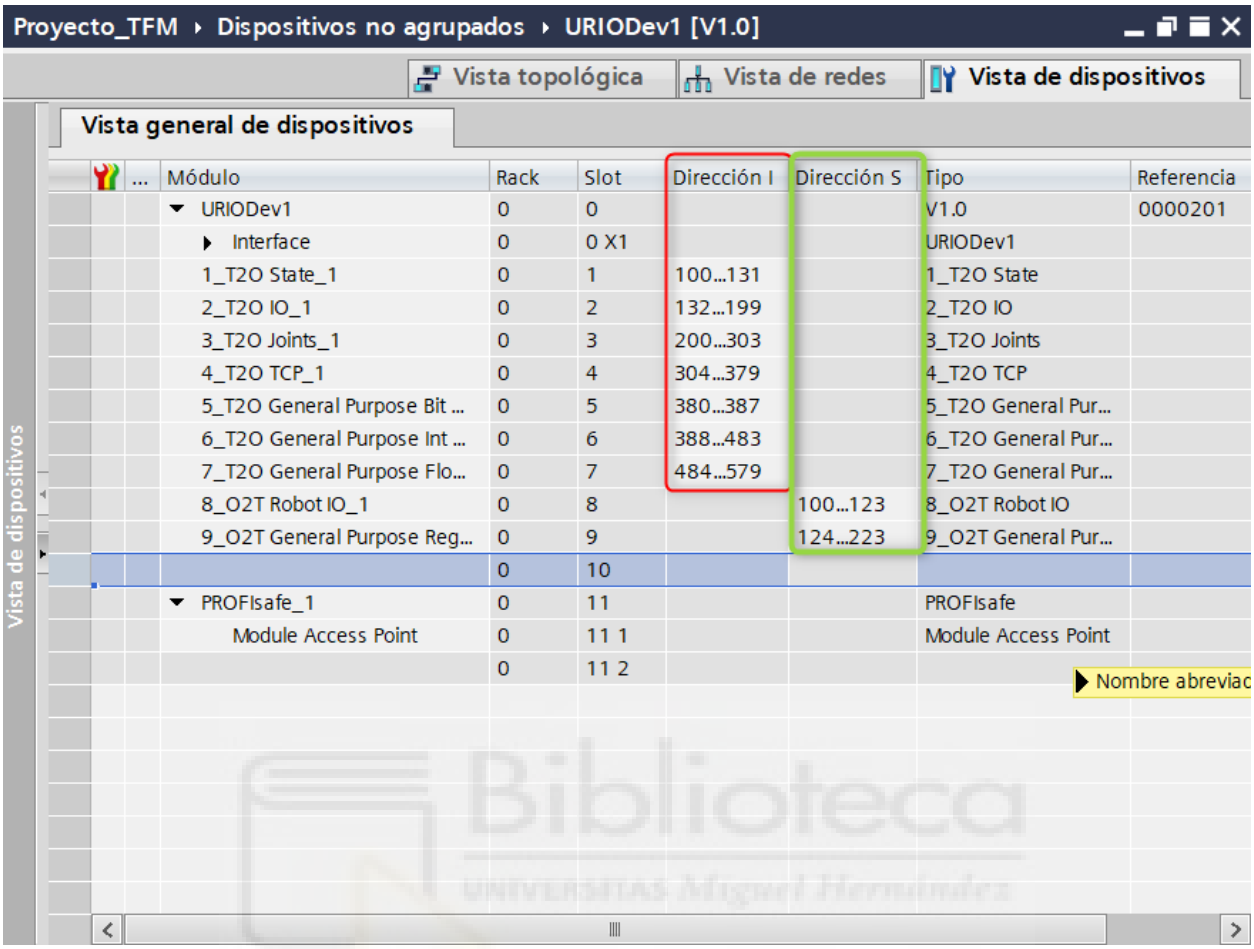


Figura 87. Direcciones escogidas. Fuente: Elaboración propia.

A continuación, otro paso no menos importante para facilitar la estructura de las variables, es añadir a nuestro proyecto el archivo de UR_datastruct.udt.

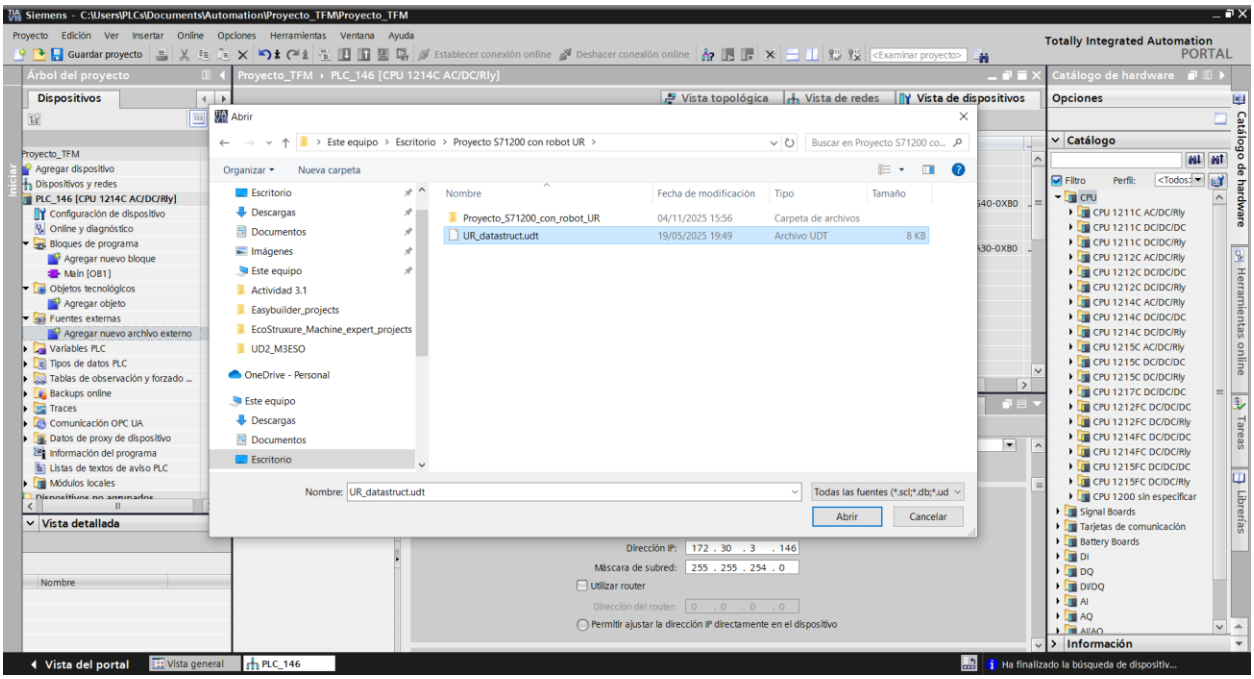


Figura 88. Importación de la fuente externa UR_datastruct.udt. Fuente: Elaboración propia.

Hacemos click derecho en la fuente externa y seleccionamos “Generar bloques a partir de la fuente”.

Este archivo es un archivo del tipo “User defined type”, desarrollado por UR para facilitar la comunicación del robot UR3e con nuestro PLC de Siemens. Es decir, está desarrollado para hacer nuestro trabajo en TIA PORTAL más sencillo, por un lado, tenemos los datos T2O (Tool to Operator) y por otro los datos O2T (Operator to Tool).

Teniendo claro que nuestro PLC es el “Operator” (operador) y nuestro robot es el “Tool” (herramienta).

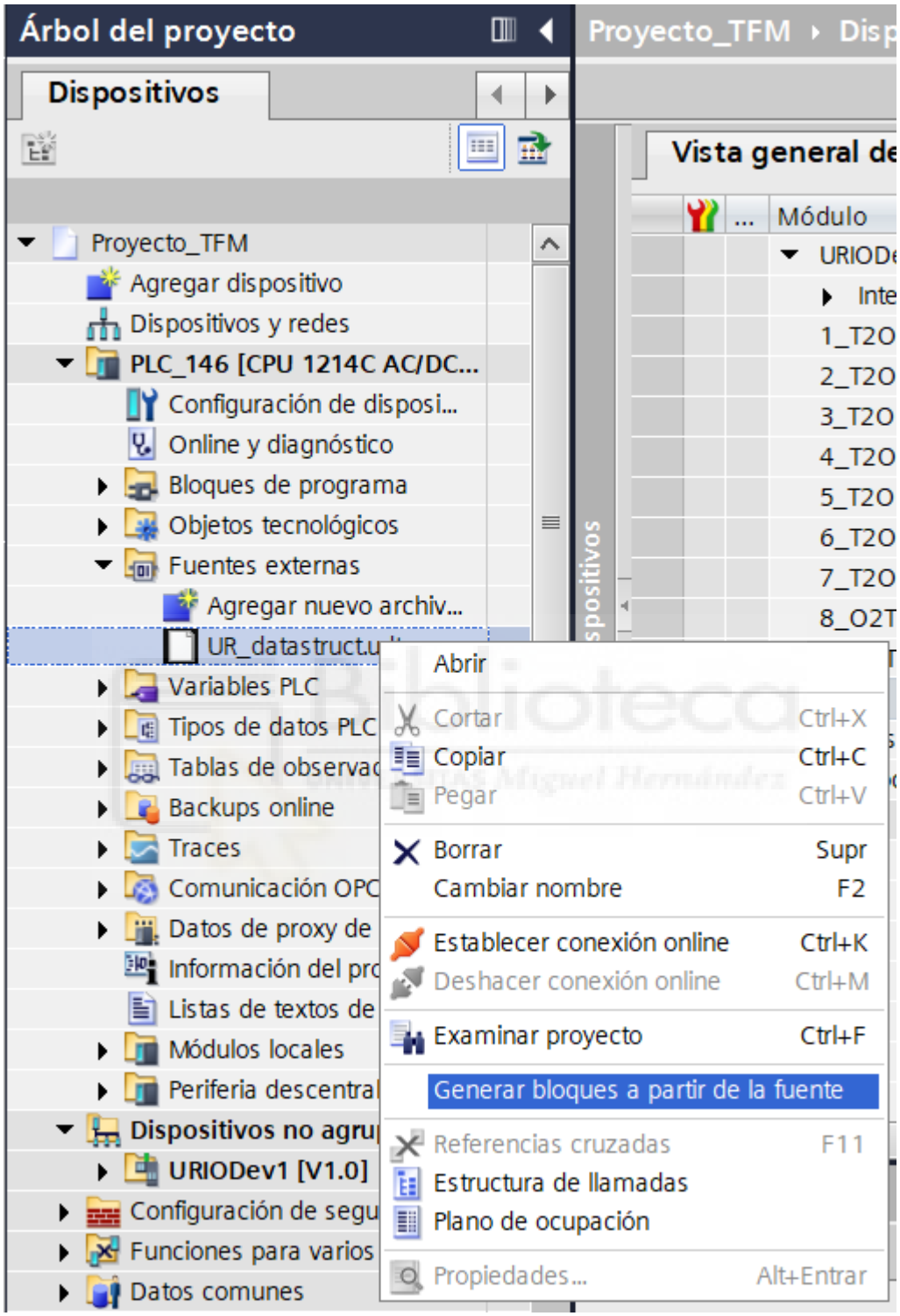


Figura 89. Detalle de configuración de los bloques de la estructura de datos importada. Fuente: Elaboración propia.

Accedemos a la pestaña variables de nuestro proyecto y generamos las dos variables fundamentales de las que colgarán todos los módulos anteriormente importados.

Estas dos variables son URI y URO y sus tipos de datos correspondientes son “UR_T2O” y “UR_O2T”

Automáticamente se nos generarán los desplegables con todos los tipos de variables contenidos en ellas, bien diferenciados.

Podemos observar que en URI con tipo de datos UR_T2O tenemos las variables de estado “State” y de entradas y salidas “IO”, “Joints”, “TCP” y las correspondientes a registros de datos de tipo Bits, Ints o Floats.

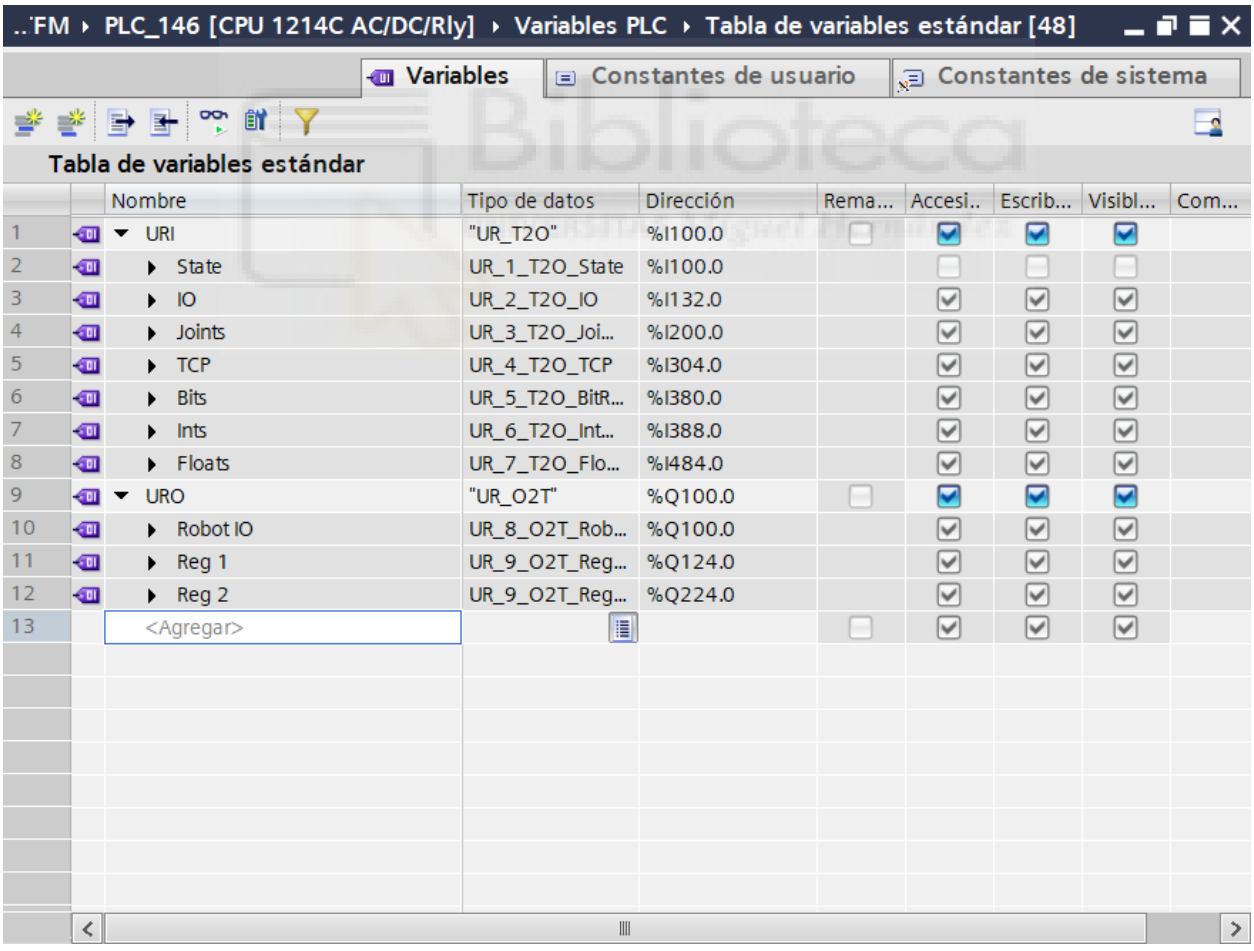


Figura 90. Comprobación de la organización de las variables generadas. Fuente: Elaboración propia.

Dentro de URO (UR_O2T), encontramos además de las variables Robot e IOs, las variables de tipo “Register” o registro, que en el software polyscope se corresponden con las entradas de tipo registro de booleano y se nombran con la nomenclatura “GPbi[0..127]”

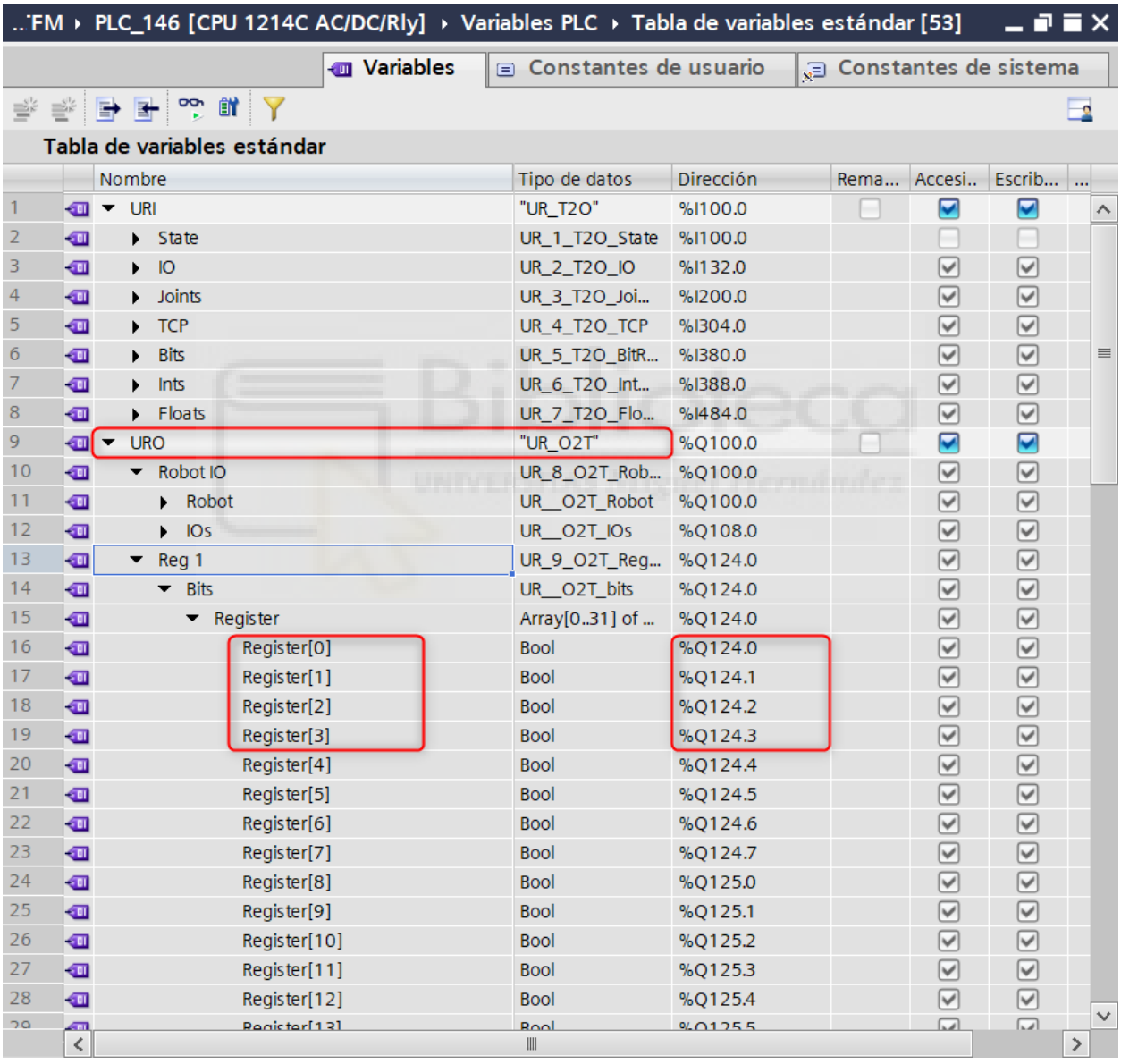


Figura 91. Variables de tipo 02T, llamadas registros de bits. Fuente: Elaboración propia.

Estas son especialmente importantes porque constituyen las salidas de registros de uso general. Y serán las salidas que utilizaremos para poder hacer uso de las funciones de inicio o paro remoto de nuestro robot.

5.3.2. Explicación del programa del PLC

El programa se realiza en ladder o diagrama de contactos, en el primer segmento se muestra que la activación de la entrada %1.0 activará la salida %Q124.0, que corresponde con la entrada de polyscope GPbi[0] definida como **Autoinicio**. Esta primera salida del PLC (entrada en nuestro robot), es la responsable de la carga automática del programa seleccionado en nuestro robot.

El segmento 2 muestra cómo la activación de la entrada %1.1 activa la salida %Q124.1 que se corresponde a su vez con la entrada GPbi[1] definida como **start** en nuestro robot. Esta es la responsable de la puesta en marcha de nuestro programa.

El segmento 3 muestra cómo la activación de la entrada %1.2 activa la salida %Q124.2 que se corresponde a su vez con la entrada GPbi[2] definida como **stop** en nuestro robot y como su nombre indica es la responsable del paro de nuestro programa.

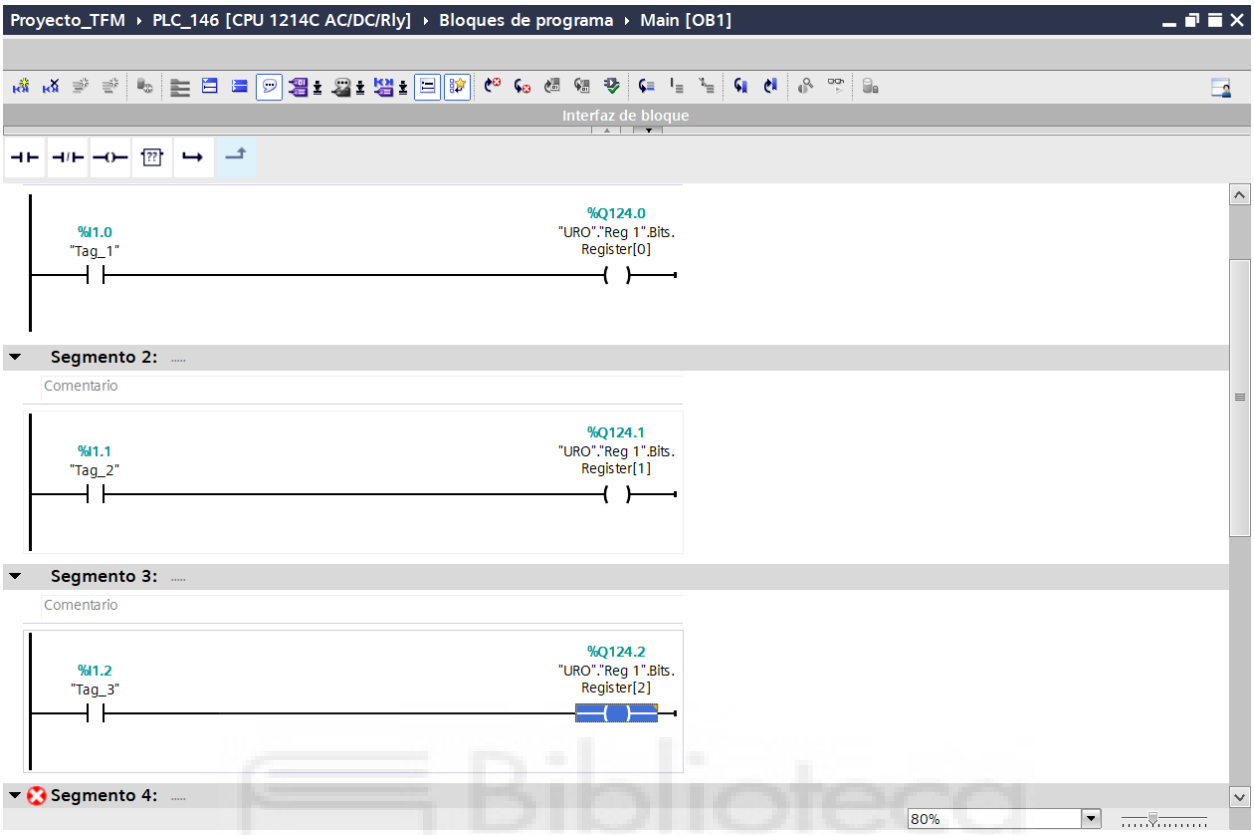


Figura 92. Primeros segmentos del programa. Fuente: Elaboración propia.

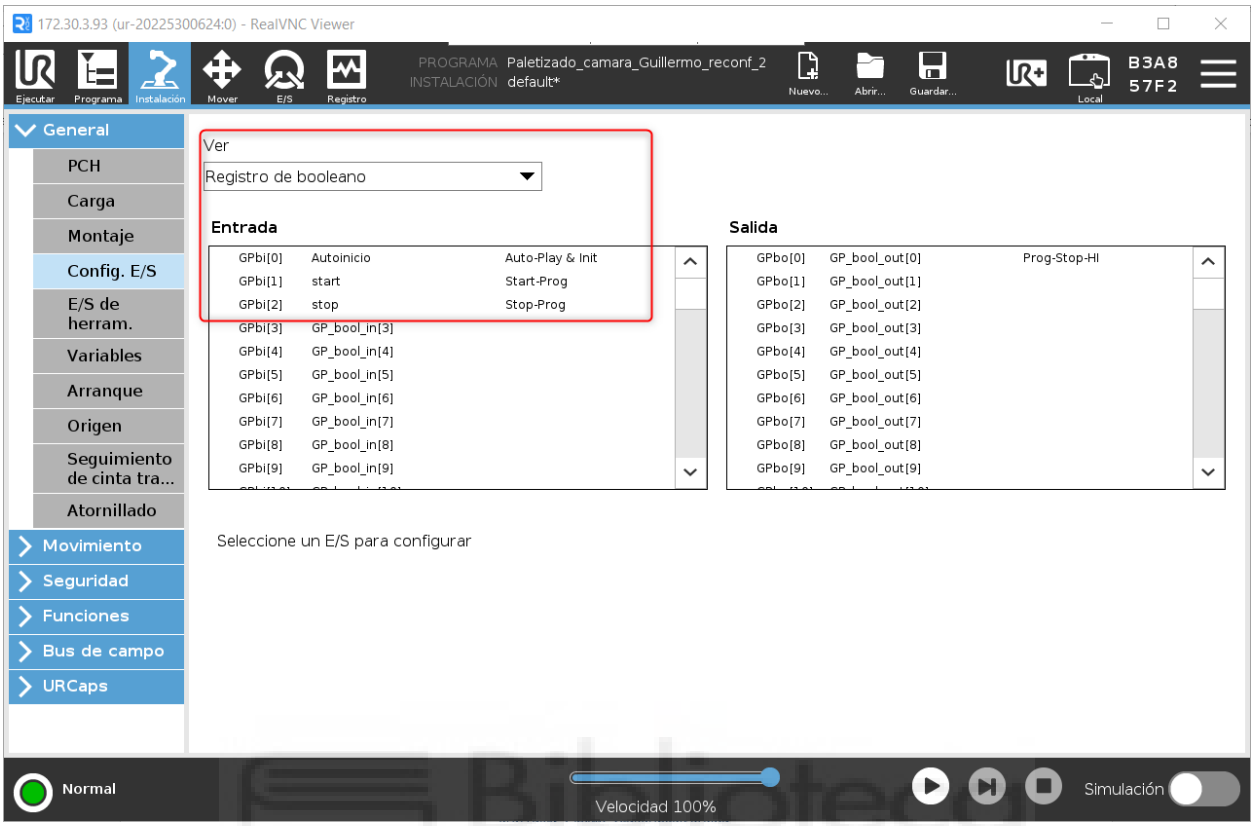


Figura 93. Definición de los primeros registros de tipo booleano (bits) como registros de autoinicialización y start-stop del programa. Fuente: Elaboración propia.

Para que realmente la activación de la primera entrada GPbi[0] "Autoinicio" cargue el programa en nuestro software polyscope y pase al estado ejecución, debemos previamente desde la pestaña *Instalación*, acceder a la subpestaña *Arranque* y marcar la casilla de "Cargar un programa determinado si el robot está encendido" y seleccionamos la condición de arranque que la variable Autoinicio se encuentre en estado HI (High), es decir de activación.

Otro detalle de vital importante es que la INSTALACIÓN de nuestro programa corresponda a *default*, es decir, sea la instalación por defecto y no a otra que hayamos guardado previamente con otro nombre, ya que, si encendemos nuestro robot, éste siempre cargará la instalación por defecto, es decir, la instalación ***default***.

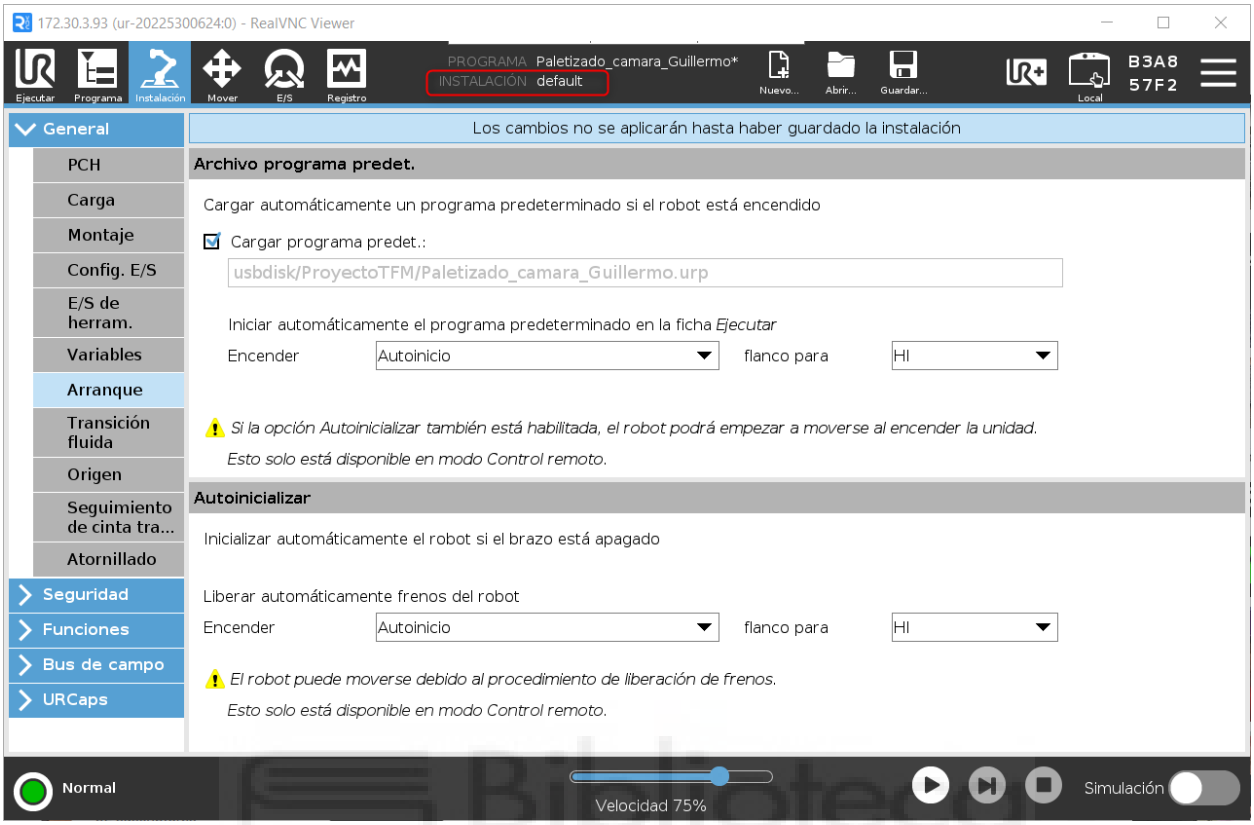


Figura 94. Detalle de la carga de programa determinado. Fuente: Elaboración propia.

El último ajuste relativo a la carga automática del programa consiste en habilitar desde ajustes la opción de control remoto, pues de lo contrario, no nos aparecerá está opción en el software polyscope.

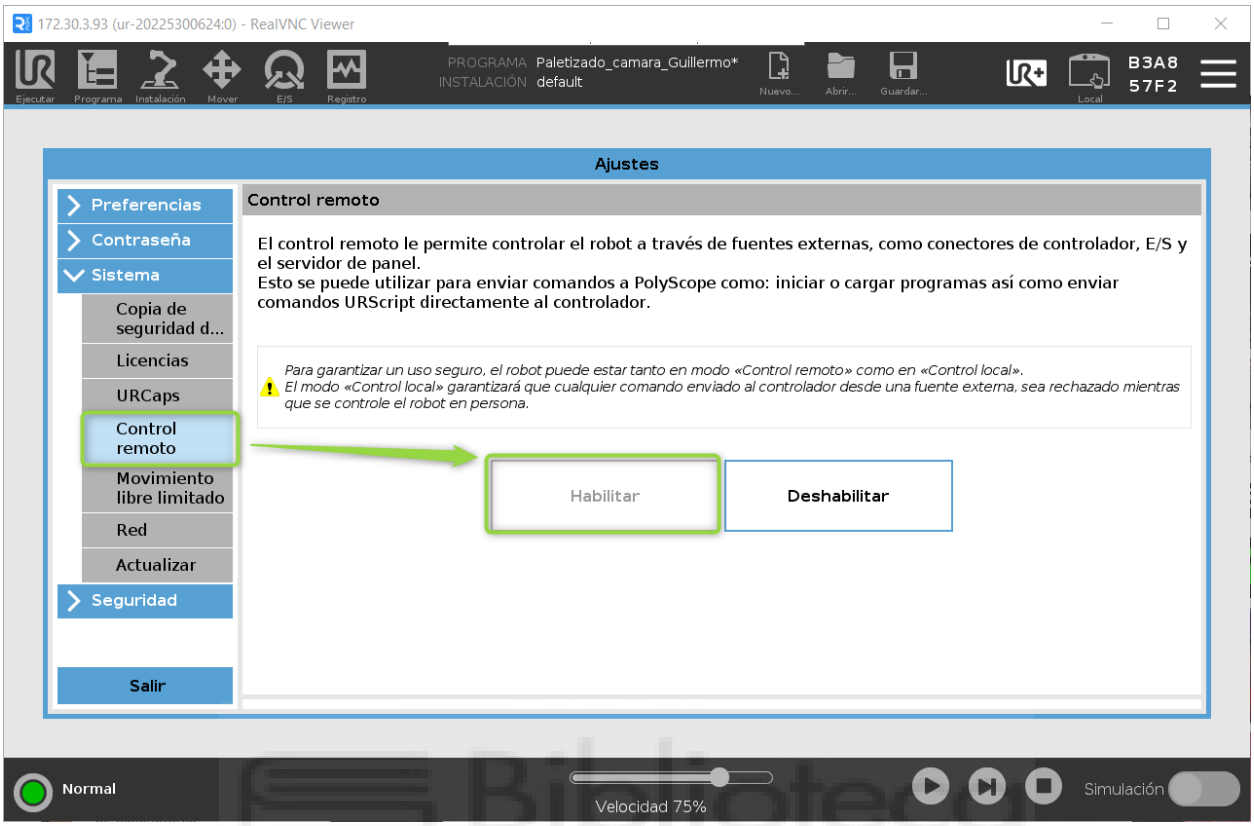


Figura 95. Habilitación del control remoto. Fuente: Elaboración propia.

Para ello debemos ir a la pestaña de *Ajustes* y dentro de ella seleccionar *Control Remoto*, una vez aquí verificamos que la opción *Habilitar* se encuentra marcada.

Es necesario recordar que este control es excluyente por lo que una vez nos encontremos en él solamente podremos "dirigir" las acciones del robot desde nuestro equipo externo al robot, y de igual modo, si nos encontramos en *Control local* no podremos mandar ninguna acción desde nuestro PLC.

A continuación explicamos los segmentos 6 y 7 del programa, estos consisten en un bloque MOVE que mueve de un área de Memoria del tipo Dword MD128 el número de piezas a paletizar al registro QD128 que corresponde con el primera área de registro de nuestro robot y que determinará cuántas piezas nuestro robot podrá paletizar como máximo en una ejecución de nuestro programa.

Es decir, si en esta variable MD128 escribimos 5 piezas nuestro robot paletizará un máximo de 5 piezas.

El segmento 7, consta de un comparador, una marca de ciclo de frecuencia 1 Hz y una salida Q0.7, básicamente establece una comparación entre el número de piezas recogidas y por tanto, paletizadas (variable ID388) con las piezas que hemos mandado a nuestro robot paletizar, cuando el robot haya paletizado las piezas que hemos indicado se activará la salida Q0.7 de manera intermitente con una frecuencia de 1 Hz.

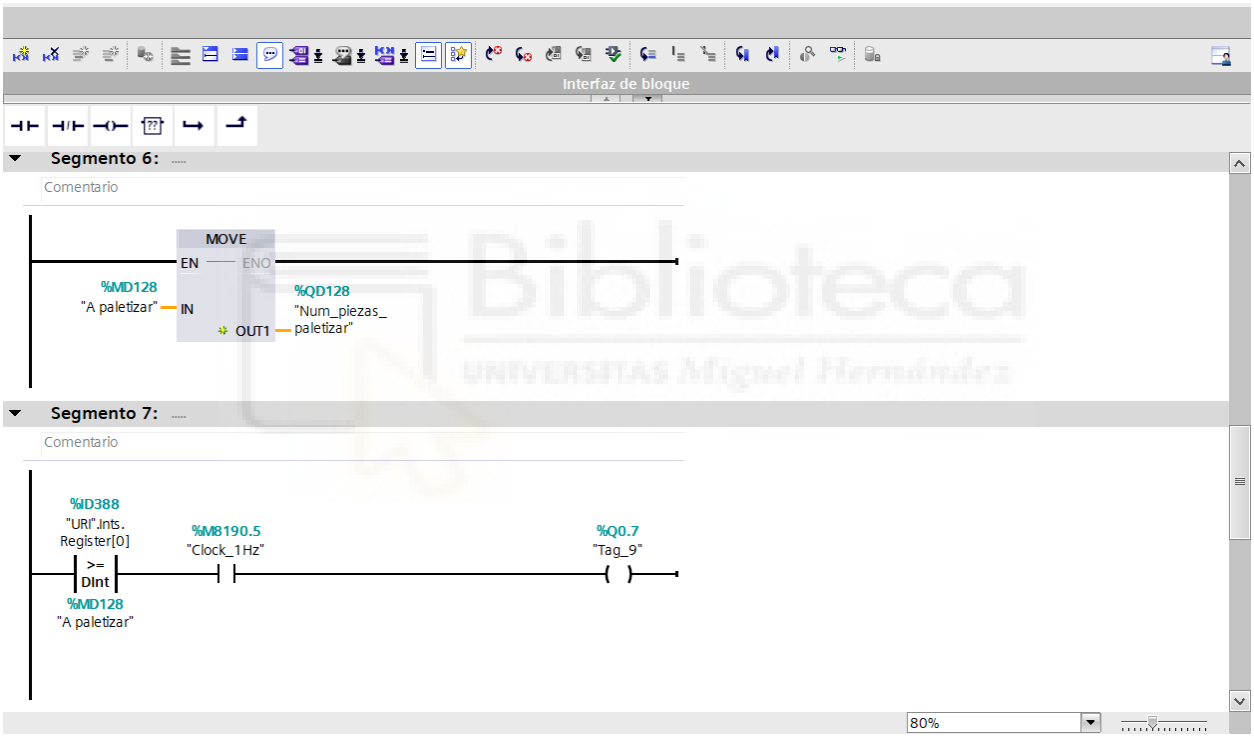


Figura 96. Segmentos intermedios del programa. Fuente: Elaboración propia.

Recordamos que para saber a qué registro debemos dirigirnos debemos consultar la tabla de variables de nuestro programa, y como se trata de un registro de salida para el PLC y de entrada para el robot, escogemos el primer registro libre de tipo Int en el área URO (O2T), en este caso el QD128

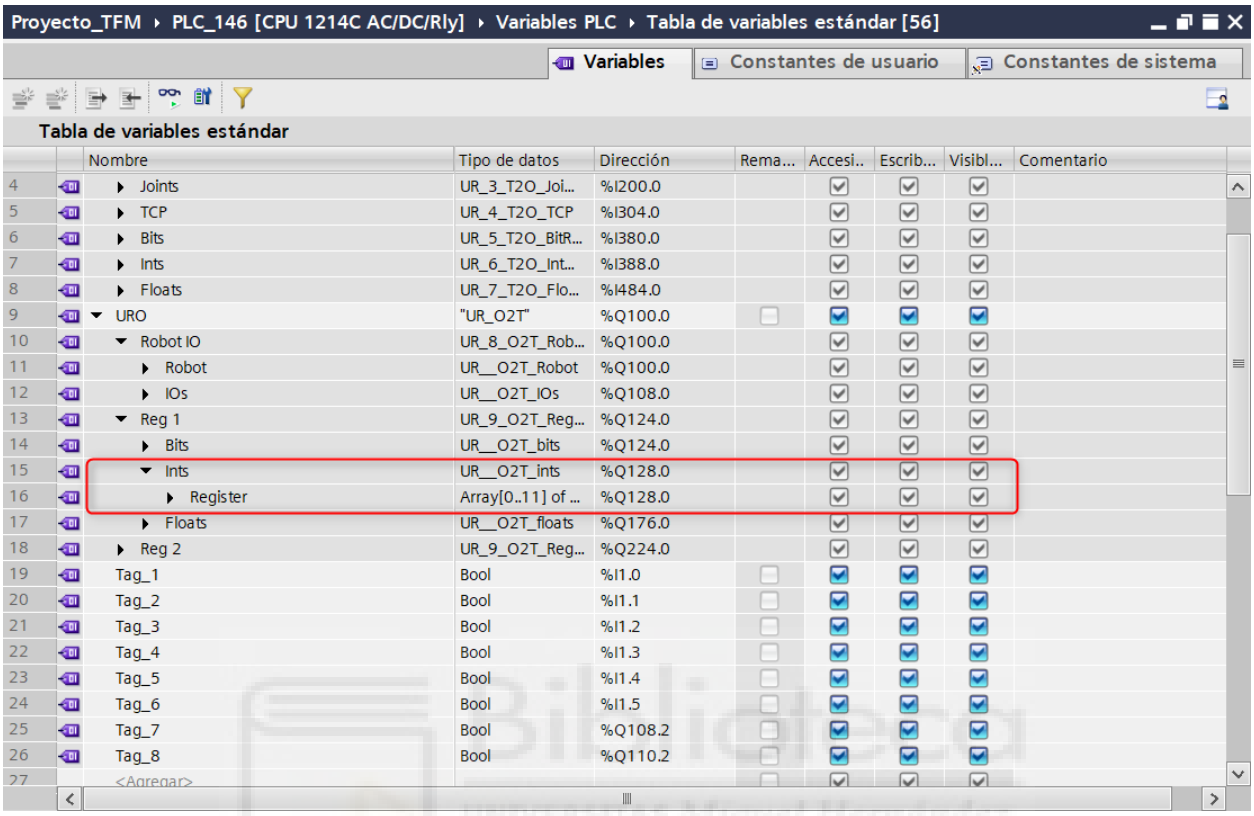


Figura 97. Búsqueda de los registros de tipo 02T de tipo entero. Fuente: Elaboración propia.

Del mismo modo, para establecer la comparación y que el robot nos diga cuántas piezas ha paletizado vamos a las variables correspondientes a URI (T2O) y seleccionamos la primera variable o registro de tipo Int disponible, en este caso la ID388 correspondiente a Register[0].

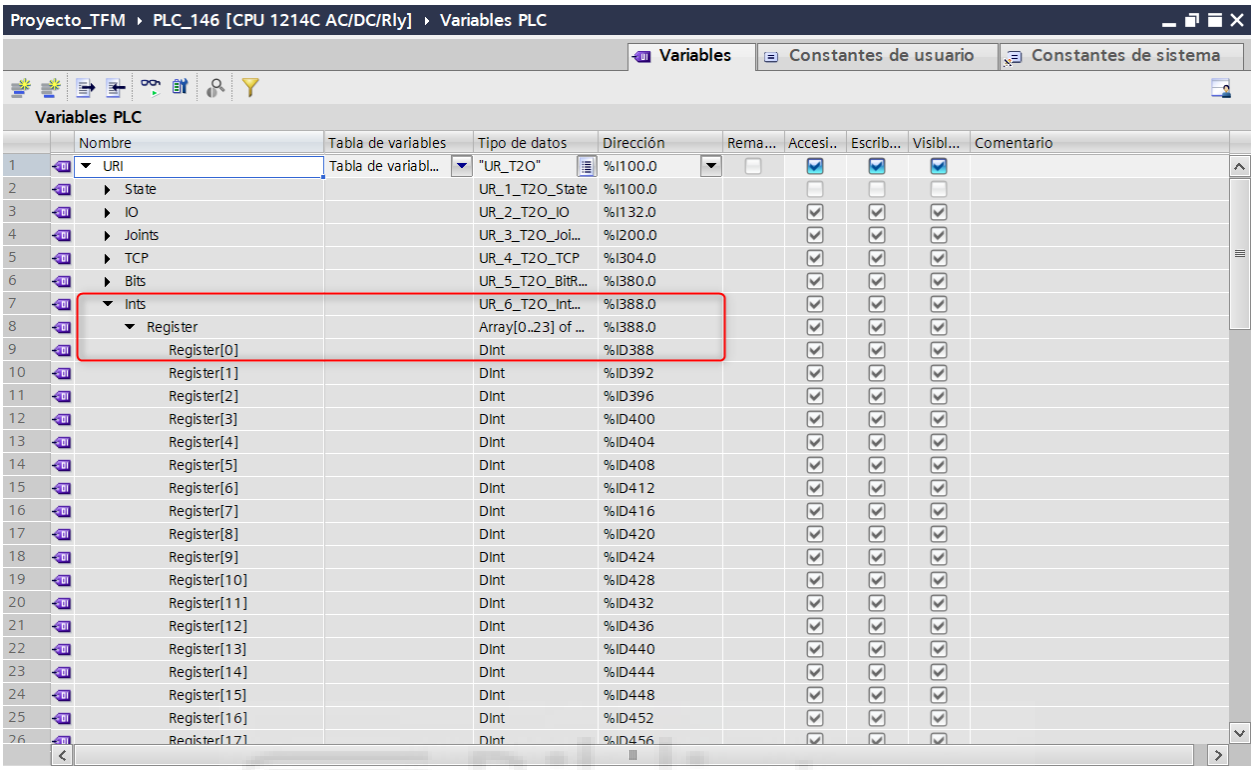


Figura 98. Busqueda de los registros T2O de tipo entero. Fuente: Elaboración propia.

Así mismo, se recuerda que las marcas de ciclo no vienen activadas por defecto en los PLCs y debemos activarlas desde *propiedades* en la configuración del PLC y asignarles una dirección específica, en nuestro caso la 8000.

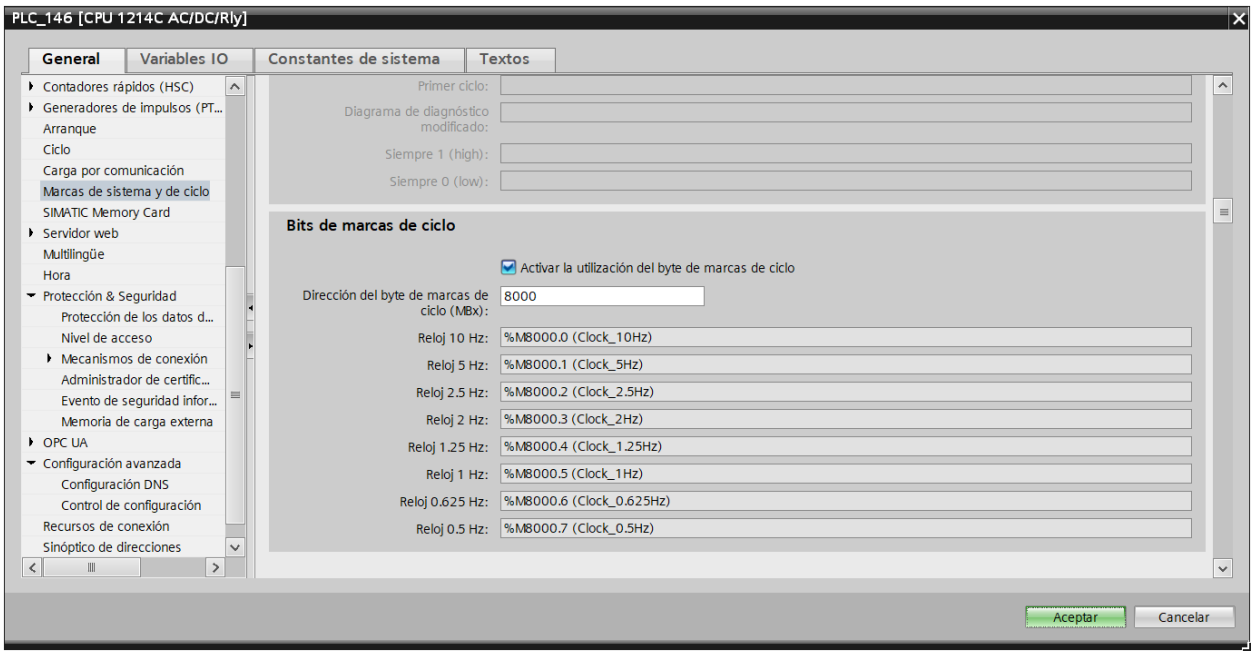


Figura 99. Activación del byte de marcas de ciclo. Fuente: Elaboración propia.

Por último, explicaremos los segmentos 8 y 9 de nuestro programa y de dónde se extraen las variables escogidas para que estos tengan efecto.

Estos son los segmentos correspondientes a la modificación remota del slider de velocidad del robot, una de las características de seguridad más intuitivas y prácticas de nuestro robot.

El segmento 8 activa el bit de máscara Q100 correspondiente a la Speed Slider Fraction Mask, esto es crucial porque actúa como medida de validación y sin este bit activo no podríamos modificar la velocidad del robot, lo que garantiza que no se produzca un cambio accidental.

El segmento 9 consiste en un bloque Move que desplaza el valor real que ajustemos de 0 a 1 desde la dirección de memoria MD204 al registro de salida del PLC (entrada en el robot) QD204.

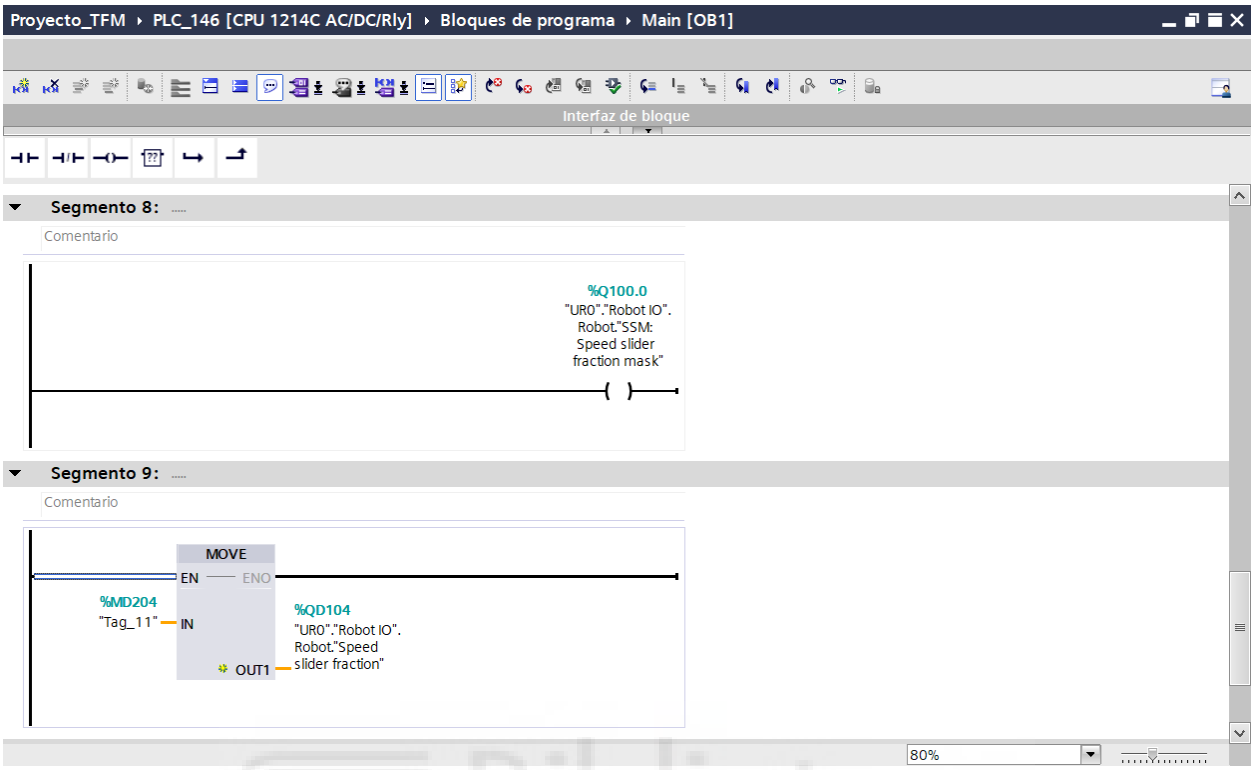


Figura 100. Últimos segmentos del programa. Fuente: Elaboración propia.

Estos registros se encuentran dentro de las variables URO (O2T), dentro de las llamadas variables "Robot", encontramos en primer lugar el bit de la *SSM: Speed slider fraction mask* en la dirección Q100.0 y por último la variable de tipo real *Speed Slider Fraction* en la dirección QD104.

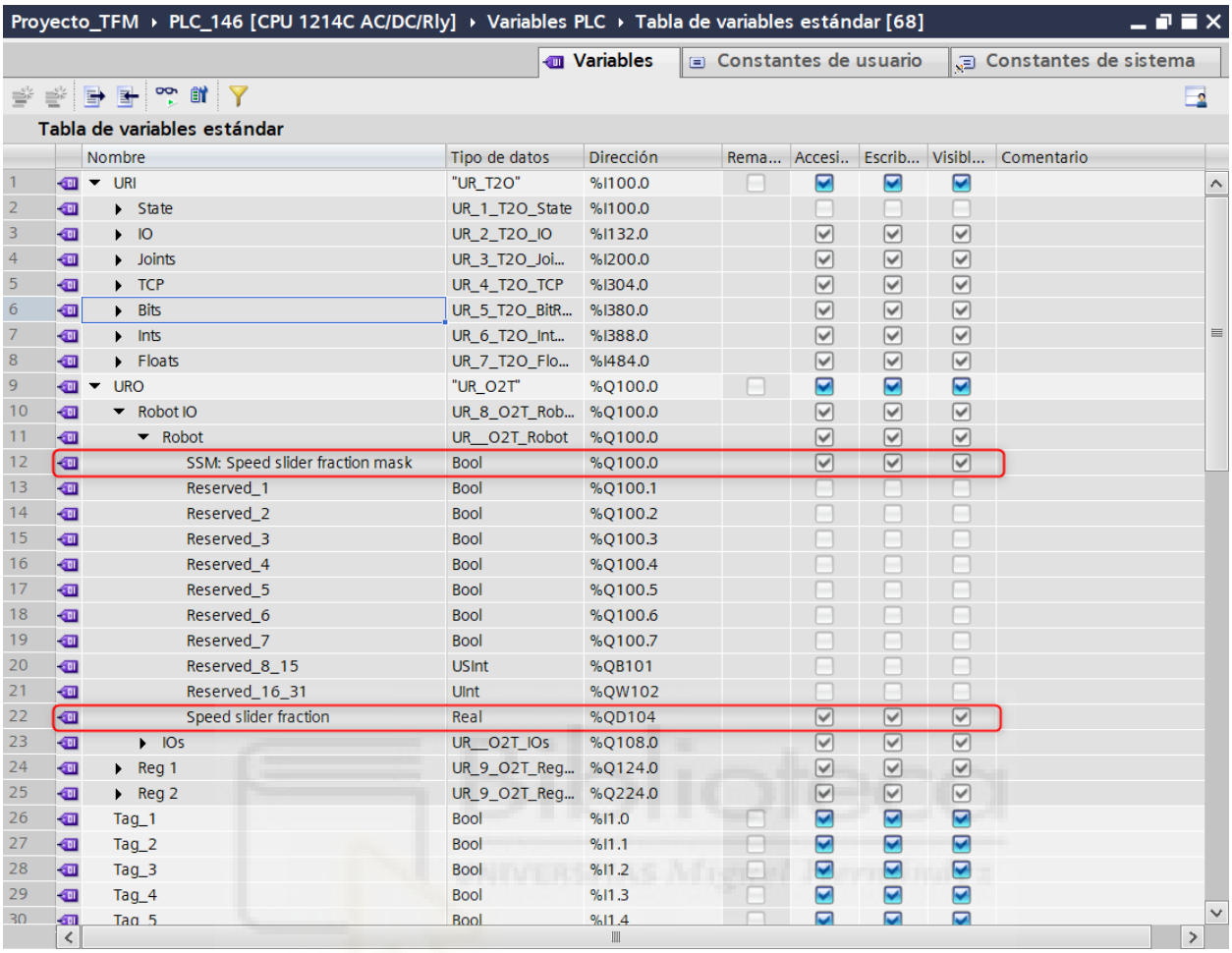


Figura 101. Búsqueda de las direcciones asociadas a la SSM y SSF. Fuente: Elaboración propia.

6. Resultados y validación

A continuación, se muestran algunas capturas que evidencian la comunicación efectiva entre nuestro sistema cámara-robot con nuestro PLC S7-1200, podemos observar a la izquierda del árbol de proyecto como todos los equipos involucrados en nuestra aplicación PLC (llamado PLC_146 en nuestro proyecto de Tia Portal) y *Periferia descentralizada* (nuestro robot, llamado urdevice93 en el proyecto) están en estado online (circulo verde) y conectados (check verde).

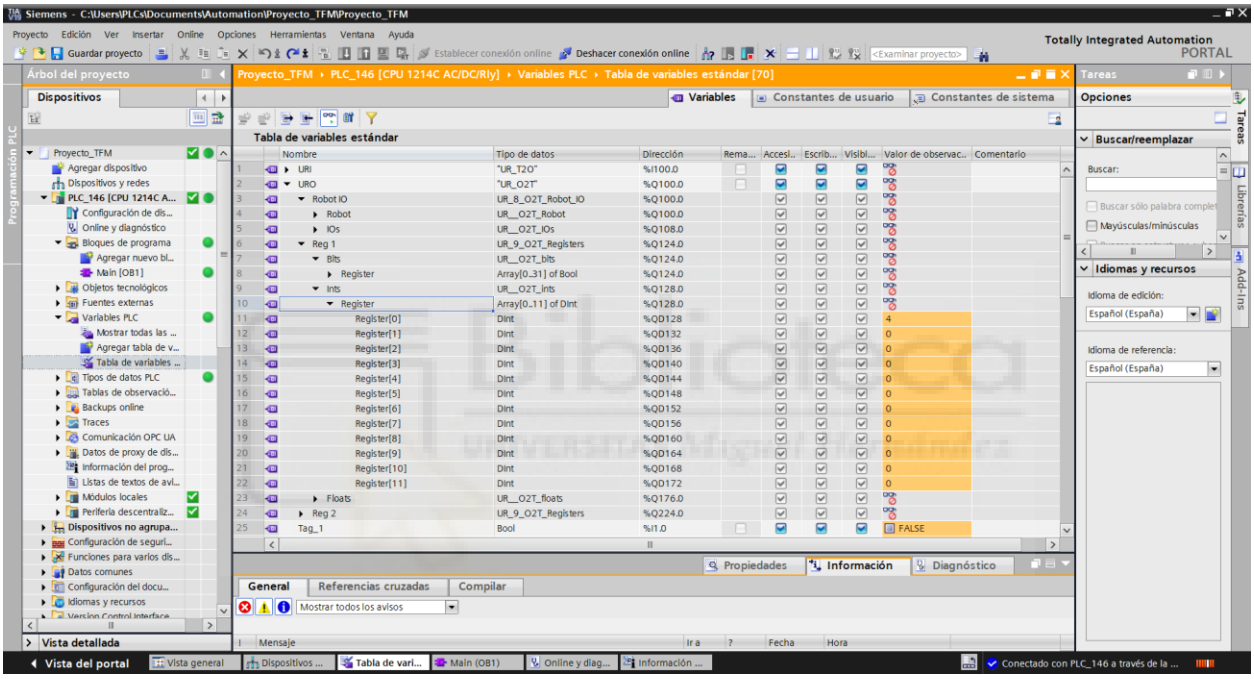


Figura 102. Comprobación del éxito de la comunicación y coherencia de los resultados. Fuente: Elaboración propia.

En la imagen siguiente si nos fijamos en el interior de la tabla de observación, en concreto en las líneas correspondientes a los registros de tipo bit (tipo de datos: UR_O2T_bits) observamos que los registros Q124.0, Q124.1 y Q124.2 están a TRUE, lo que quiere decir, que se ha enviado la orden de Autoinicio (carga del programa), start (inicio del programa) y stop (paro del programa) desde nuestro PLC.

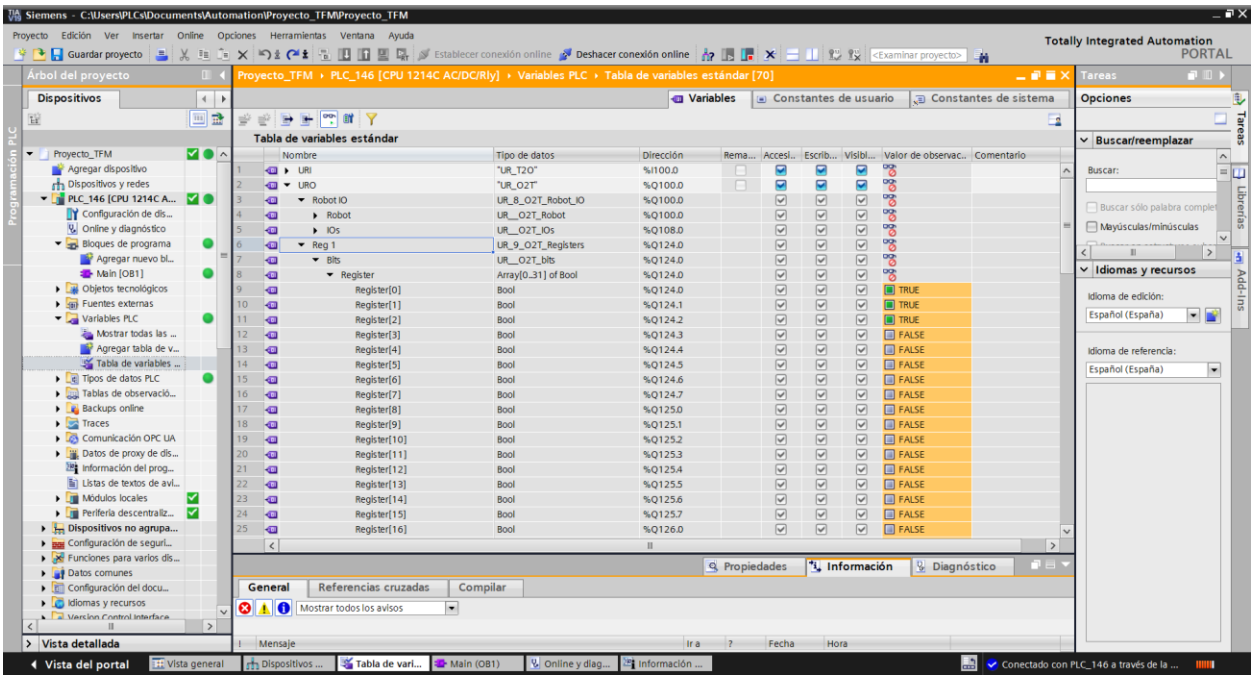


Figura 103. Detalle del funcionamiento de la aplicación. Fuente: Elaboración propia.

Tan solo queda verificar el correcto funcionamiento de la aplicación cámara-robot, para que nuestro sistema quede totalmente funcional.

En el punto 6.4 del presente proyecto se adjuntan enlaces a diferentes vídeos que muestran la ejecución del programa como prueba de su correcto funcionamiento.

De momento mostramos el aspecto que tendría nuestro banco de trabajo con las piezas a paletizar completamente desorganizadas y dentro de nuestra área de detección.

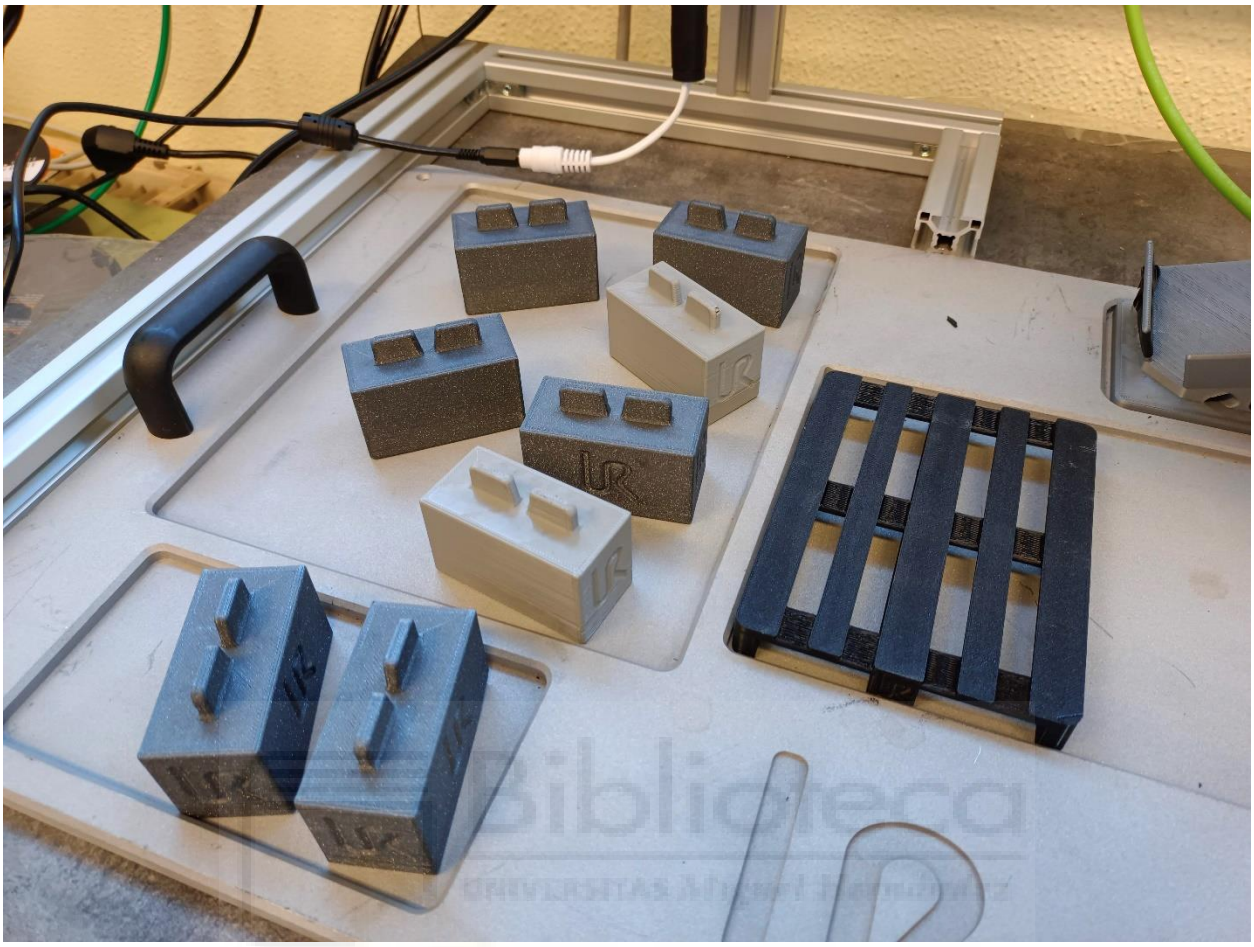


Figura 103. Área de trabajo y piezas a paletizar. Fuente: Elaboración propia.

Tras una ejecución correcta del ciclo de paletizado.

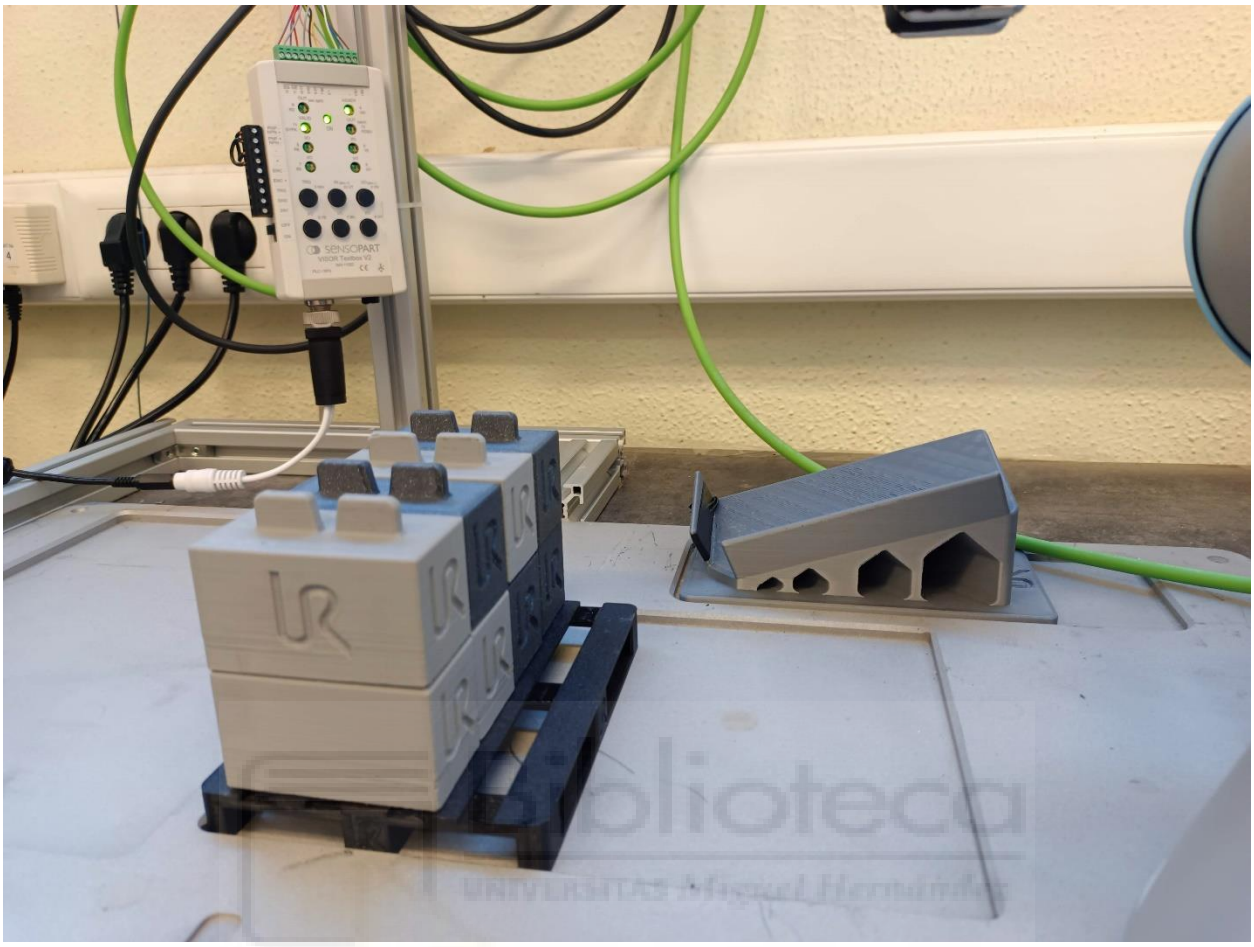


Figura 104. Área de trabajo y piezas ya paletizadas. Fuente: Elaboración propia.

Cabe destacar de la imagen anterior el elemento de alineación utilizado, es decir, la rampa que se muestra en la imagen sin la cual la alineación y posicionamiento tan preciso de las piezas sería una tarea mucho más costosa y que requeriría de ajustes en las calibraciones constantes.

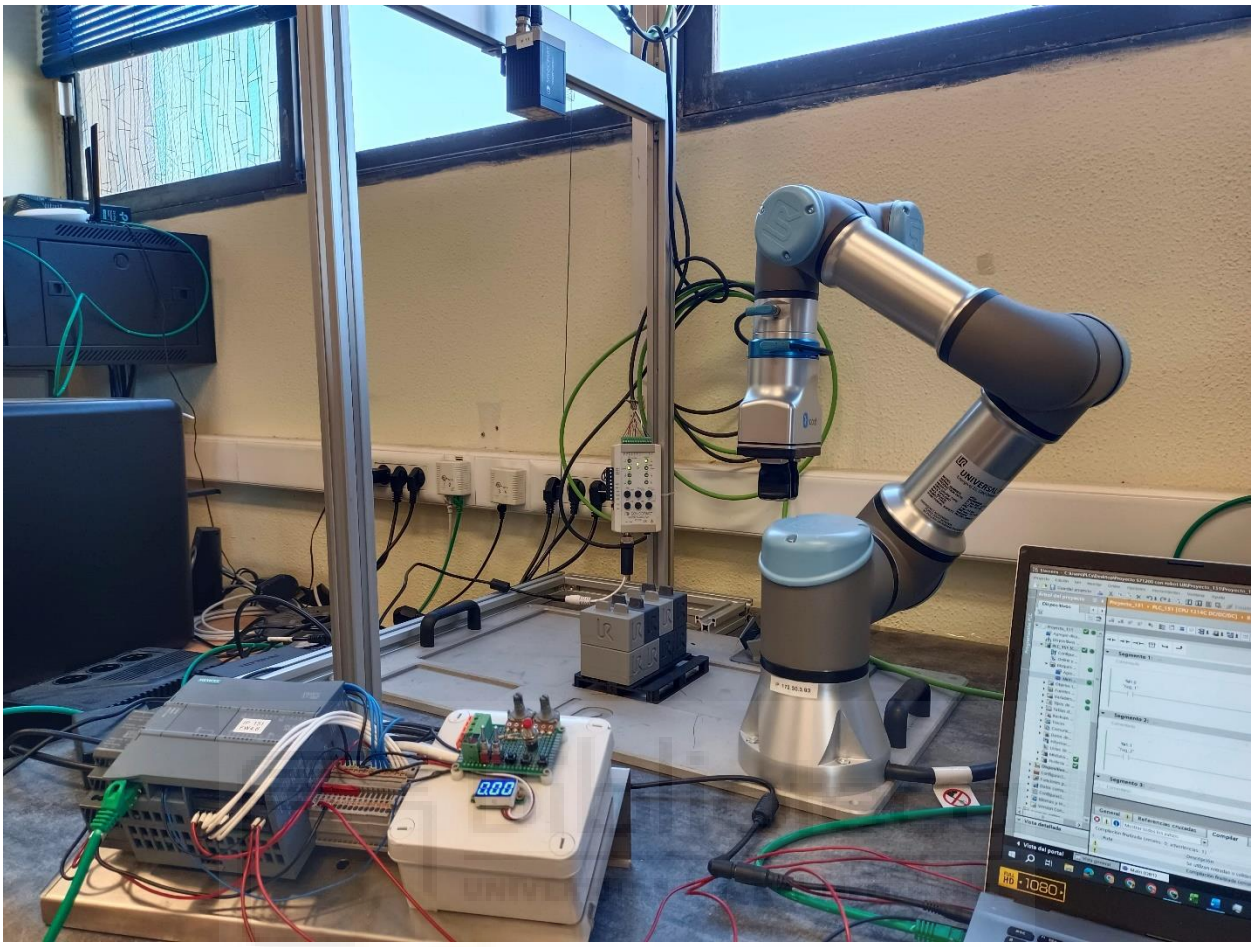


Figura 105. Área de trabajo, paletizado completo y robot en posición de espera. Fuente: Elaboración propia.

Como podemos observar el robot, se encuentra en la posición de espera a que se retiren las piezas del palet realizado y a que se depositen nuevas piezas en nuestra área de detección de la cámara.

6.1. Descripción de las pruebas realizadas

En este apartado vamos a recapitular y demostrar los objetivos conseguidos en este proyecto.

En primer lugar, realizaremos pruebas de comunicación PLC-Robot.

Desde el software Polyscope comprobaremos la conexión PROFINET al PLC, desde el software Tia Portal nos aseguramos de que el dispositivo esclavo uriodev93 sea localizable.

En segundo lugar, comprobaremos la conexión de la cámara con el robot.

Para ello utilizaremos la herramienta Sensofind de Vision Sensor.

En tercer lugar, realizamos las pruebas de calibración de la cámara, es decir, estudiamos los parámetros y configuraciones más efectivos para la detección de las piezas a paletizar y efectuaremos el calibrado de la aplicación "Visor Easy-Pick" llevando el TCP del robot a los 4 fiducial points de nuestra calibration plate tantas veces como sea necesario para obtener una precisión de calibración correcta.

En cuarto lugar, ejecutaremos el ciclo de trabajo normal de nuestra aplicación, es decir, ejecutaremos la tarea de Visor Easy-Pick y el paletizado programados en el robot tantas veces como sea necesario para comprobar su efectividad.

En último lugar realizaremos las pruebas de control remoto, demostrando así la comunicación práctica del PLC al robot, cargando nuestro programa en el robot mediante la activación de I1.0, iniciando dicho programa con I1.1 o deteniéndolo con I1.2. También realizaremos pruebas de control remoto de la velocidad del robot, de activación de salidas digitales de nuestro robot desde el PLC o de lectura del número de piezas paletizadas.

6.2. Resultados experimentales

Para poder discutir la validez de los resultados obtenidos debemos no solo presentar evidencias del funcionamiento de la aplicación sino también datos.

Estos datos son los correspondientes a las medidas de calibración del robot en la aplicación de Visor Easy-pick, o las puntuaciones obtenidas por los diferentes objetos detectados por la cámara Sensopart, así como la capacidad real de la aplicación para paletizar dichas piezas en un palet regular de 4x2.

Comencemos por los resultados de calibración del robot, en el trabajo se muestran capturas de que la desviación media obtenida en la calibración de los fiducial points es de 0.963 mm y la desviación de los puntos de calibración es de 0.139 px, estos son valores válidos según Sensopart para la correcta implementación de la cámara en aplicaciones de robótica.

Continuamos con las puntuaciones obtenidas por las piezas a paletizar cuando son detectadas por la cámara, es decir, que puntuación es asignada a cada pieza según el software Sensoconfig cuando tomamos una instantánea.

Las capturas realizadas y mostradas a lo largo de este trabajo muestran que para una pieza bien contrastada los valores habitualmente superan los 90 puntos de confianza.

Esto es crucial para la correcta aplicación del paletizado pues si la cámara no detecta con el suficiente grado de confianza una pieza puede ocurrir que el robot la obvie o que intente cogerla, pero por error de calibración no pueda asirla produciendo un hueco en el palet.

Por último, se ha demostrado la capacidad del robot para generar palets regulares de 4x2, es decir, de 4 piezas de base y dos alturas, en total apila hasta 8 piezas con facilidad y podríamos realizar

tantas modificaciones en la programación de este paletizado como fuera necesario para comprobar la precisión cinemática de nuestro robot.

6.3. Análisis y discusión de resultados

En este apartado realizaremos el análisis y discusión de los resultados obtenidos tras llevar a la práctica este proyecto. Debemos saber diferenciar entre una aplicación de uso colaborativo de una aplicación de uso industrial real, pues en el primer caso las pruebas realizadas han sido menores en número (en un ámbito industrial estas se realizan miles de veces) pero igualmente han sido estudiadas y realizadas con detalle.

En primer lugar, la precisión de los resultados de calibración (con una desviación media inferior a 1 mm) permitieron dar por válida una calibración que se realiza normalmente de manera manual y, en el mejor de los casos, haciendo uso del control por ejes del software polyscope. Esto unido a que se usó una herramienta de baja precisión como la pinza 2FG7 hacen que la calibración de la aplicación "Vision Easy-Pick" sea una tarea delicada que en ambientes industriales se revisaría cada cierto tiempo para asegurarnos de que las vibraciones o desajustes mecánicos no empeorarán la precisión.

También es importante mencionar que el parámetro Z-shift ajustado a -33 mm, es de vital importancia, dado que no son lo mismo plano de detección que plano de trabajo para nuestro robot, y sin este parámetro debidamente ajustado el robot se enfrentaría a colisiones en lugar de a operaciones de 'pick' con éxito.

En segundo lugar, debemos garantizar la repetibilidad y efectividad de los resultados obtenidos por la visión artificial.

Las puntuaciones obtenidas por los detectores de la cámara Sensopart V20C los valores críticos de detección incluso en condiciones de baja iluminación o de bajo contraste, pues como se ha

mostrado anteriormente en el proyecto las piezas de color blanco (impresas en 3D con filamento transparente) son detectadas sobre un fondo metálico con puntuaciones superiores a 60.
De este modo, las piezas más oscuras (grises o negras) resultan más fácilmente detectables aún con puntuaciones superiores a 80 en la mayoría de los casos.
También hay que mencionar que el rango de giro permitido en cada pieza desde -180º a 180º es constituye un poderosísimo ajuste de detección y permite a la cámara indicar al robot dónde y con qué orientación se encuentra la pieza "caiga como caiga" en nuestra área de trabajo.

En último lugar, analicemos los puntos más destacados de la seguridad de nuestro proyecto.
Desde el punto de vista de la seguridad y mantenimiento de los equipos la integración de nuestro PLC Siemens S7-1200 nos permite la adquisición de datos en tiempo real (por medio de su estructura .udt) esto nos permite tener un registro detallado del estado de las articulaciones del robot, de sus temperaturas, de las diferentes posiciones de las mismas durante todo el proceso, intensidades consumidas, etc.
Esto es un punto crucial en las aplicaciones de uso industrial y una potente herramienta para el mantenimiento predictivo del robot.

Por otro lado, también nos permite tener un registro de la eficacia del proceso, es decir, de las veces que el robot cumple con éxito con sus tareas (detección de piezas y paletizado posterior) y de las ocasiones en las que pudiera fallar, porque como todo sistema real, en algún momento fallará. Para esto deberíamos de auditar la pestaña de *Log* de polyscope trayendo los datos más significativos a una tabla de variables de nuestro PLC.

Para esto se propone en trabajos futuros la implementación de alguna herramienta de extracción de datos del PLC a una base de datos por medio de protocolos seguros como OPC/UA, un ejemplo de software a implementar para la extracción de estos datos sería KEPserverEX o Node-red.

6.4. Enlaces vídeos realizados como prueba de funcionamiento

Por si el lector está interesado en la visualización de los resultados obtenidos tras la realización de este TFM, se adjuntan los enlaces a los vídeos de elaboración propia siguientes.

1. Vídeo del paletizado de 8 piezas sin alineador
2. Vídeo del paletizado de 8 piezas con alineador
3. Vídeo de la evaluación de la cámara V20C

El primer enlace muestra como la aplicación puede realizar con éxito el palet de 4 piezas de base y dos alturas, pero debido a la suma de los errores de calibración que se van cometiendo, ya que ninguna calibración es perfecta, el palet queda mucho más irregular de lo previsto.

La solución encontrada se muestra en el segundo vídeo, dónde por medio de un elemento "alineador", es decir, por medio de depositar las piezas previamente al paletizado en una rampa alineadora, podemos mejorar la regularidad de los movimientos de "pick" del robot.

El tercer vídeo se adjunta como punto de vista de la ejecución del programa desde el software Sensoconfig de la cámara Sensopart y nos permite observar las evaluaciones que realiza la misma en la detección de las piezas que el robot habrá de paletizar.

7. Conclusiones y trabajos futuros

En conclusión, con el estudio y desarrollo del presente proyecto se han conseguido los siguientes puntos:

1. La integración y comunicación de tres equipos de diferentes fabricantes como lo son la cámara Sensopart V20C, el PLC S7-1200 de Siemens y el robot UR3e de Universal Robots, haciendo uso además de diferentes protocolos industriales (profinet y Ethernet/IP).
2. La demostración de que un robot colaborativo no es excesivamente complejo de programar e integrar para la realización de procesos que aporten valor a nuestra empresa, esto es debido a facilidades como las estructuras de datos o las UR-Caps que los fabricantes diseñan para asegurar compatibilidades entre diferentes equipos.
3. La seguridad y fiabilidad de los robots colaborativos en la escena actual, pudiéndose configurar planos de seguridad por software y un intercambio de datos actual que permita a PYMES no solo hacer uso de las capacidades del robot para diferentes tareas industriales, sino adelantarse en su mantenimiento alargando la vida útil de dichas tareas y equipos involucrados.

En cuanto a los trabajos futuros, existen multitud de posibilidades, variaciones y mejoras que podrían realizarse al hardware y software implicado en el proceso. Algunas de estas posibilidades son:

- La variación de los paletizados, por ejemplo, buscando aquel que minimice el espacio desaprovechado en cada palet haciendo uso de paletizados irregulares e investigando más

a fondo las plantillas ofrecidas por UR. Esto tendría mucho sentido en el caso de una aplicación de empaquetado real.

- Si en la aplicación interviniesen cintas transportadoras podríamos también integrar sistemas de seguimiento de cinta transportadora por medio de las entradas digitales DI8 a DI11 de nuestro robot.
- Desarrollo de una interfaz HMI avanzada en nuestro software Tia Portal para aportar mayor claridad y facilidad de supervisión de las variables principales del proceso o de parámetros internos del robot (variables T2O) cómo las diferentes variables de cada articulación o del TCP.
- Uso de software especializado para la extracción de datos de nuestro HMI (contenidos en tablas en nuestro PLC), tales como KepserverEX o Node-Red (ya mencionados) y almacenamiento de estas en bases de datos que permitan una monitorización integral y detallada de nuestro sistema.

En definitiva, las mejoras aportarían no solo funcionalidad a nuestra aplicación sino seguridad, fiabilidad y eficiencia.

8. Referencias

[1] Federación Internacional de Robótica (IFR), “World Robotics 2024 - Industrial Robots”, Fráncfort, IFR Statistical Department, 2024

[2] AER Automation, “Las ventas de robots industriales en España se disparan un 48% y la robótica de servicio crece un 17%”, Madrid, Asociación Española de Robótica y Automatización, 2024.

[3] Universal Robots, “Tipos de robots industriales y sus aplicaciones”, Odense, Blog de Universal Robots, 2021.

[4] Leitbetriebe Austria, “Congratulations YuMi: ABB robot has been setting new standards in collaboration for five years”, Viena, ABB Robotics / Leitbetriebe Austria, 2020.

[5] Wikipedia, “Denavit–Hartenberg parameters”, San Francisco, Wikimedia Foundation, 2024

[6] Universal Robots A/S, “DH-Parameters for calculations of kinematics and dynamics”, Odense, Centro de soporte técnico de Universal Robots, 2018

[7] Universal Robots A/S, “PROFINET how-to guide e-Series”, Odense, Centro de soporte técnico de Universal Robots, 2018.

Manuales técnicos

[8] Universal Robots A/S, “Universal Robots e-Series Manual de usuario UR3e (Versión 5.0.0)”, Odense, Universal Robots A/S, 2018

[9] SensoPart, “VISOR® V20 Allround Color (V20C-ALL-A3-U-N-M2-L)”, Gottenheim, SensoPart Industriesensorik GmbH, 2021

[10] Siemens Aktiengesellschaft, “SIMATIC Autómata programable S7-1200 G2 Manual de sistema”, Núremberg, Siemens AG Digital Industries, 2025.

[11] OnRobot A/S, “Ficha técnica 2FG7 (Versión 1.9.1)”, Odense, OnRobot A/S, 2021.

9. Anejos

9.1. Programa completo del PLC en Tia Portal V19

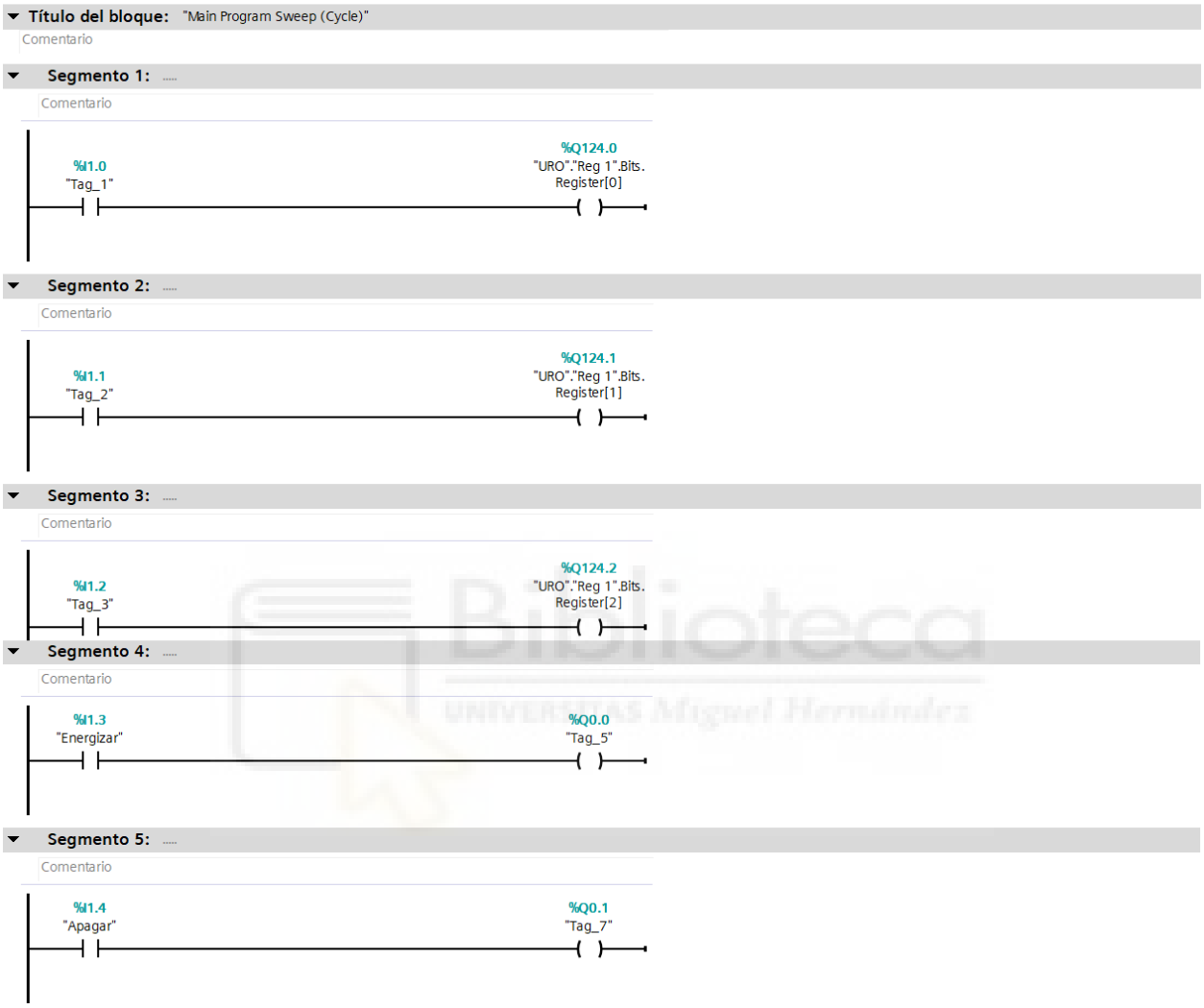


Figura 106. Primera parte del programa en Tia Portal. Fuente: Elaboración propia.

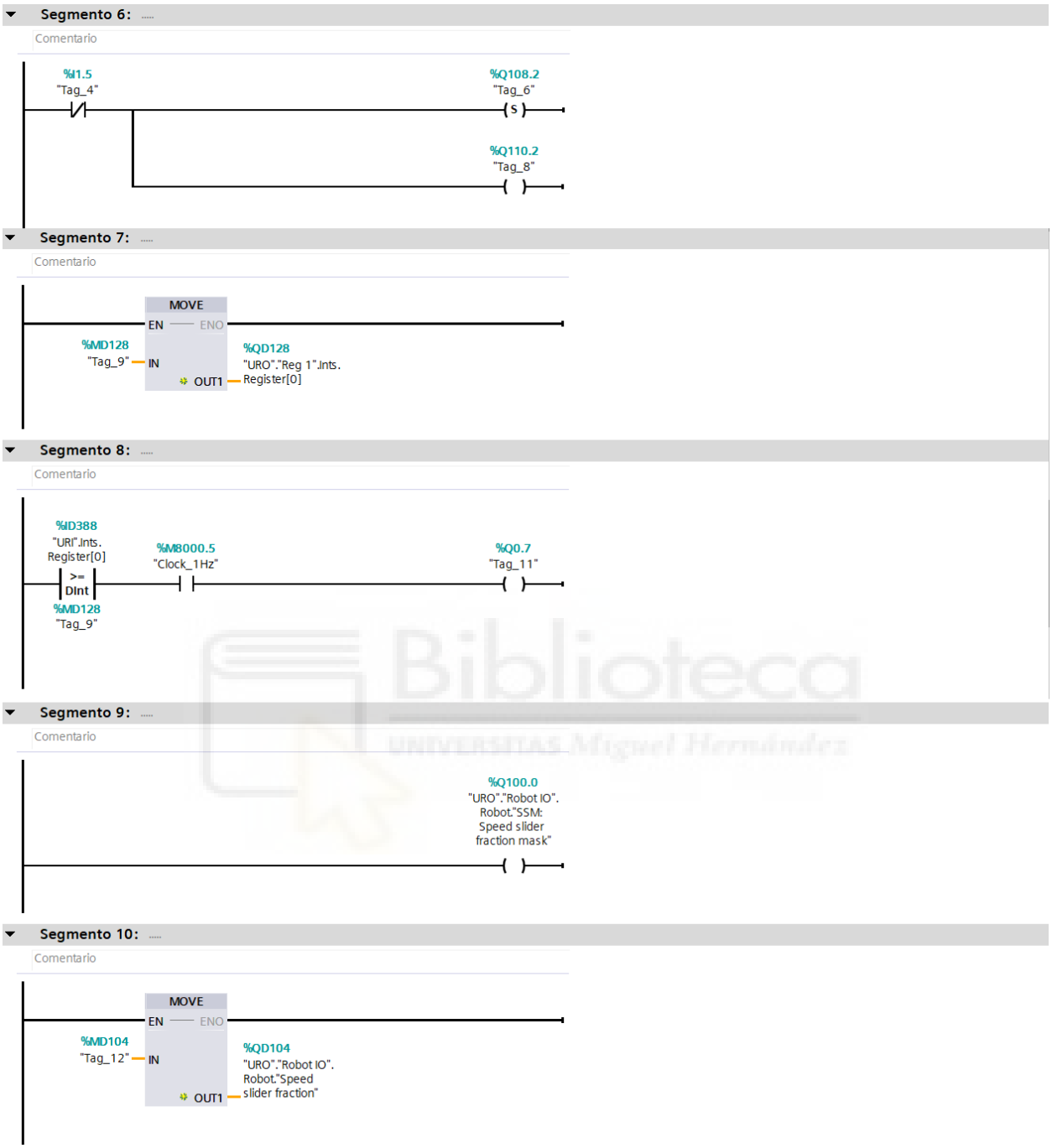


Figura 107. Segunda parte del programa en Tia Portal. Fuente: Elaboración propia.

9.2. Programa completo del robot UR3e en Polyscope

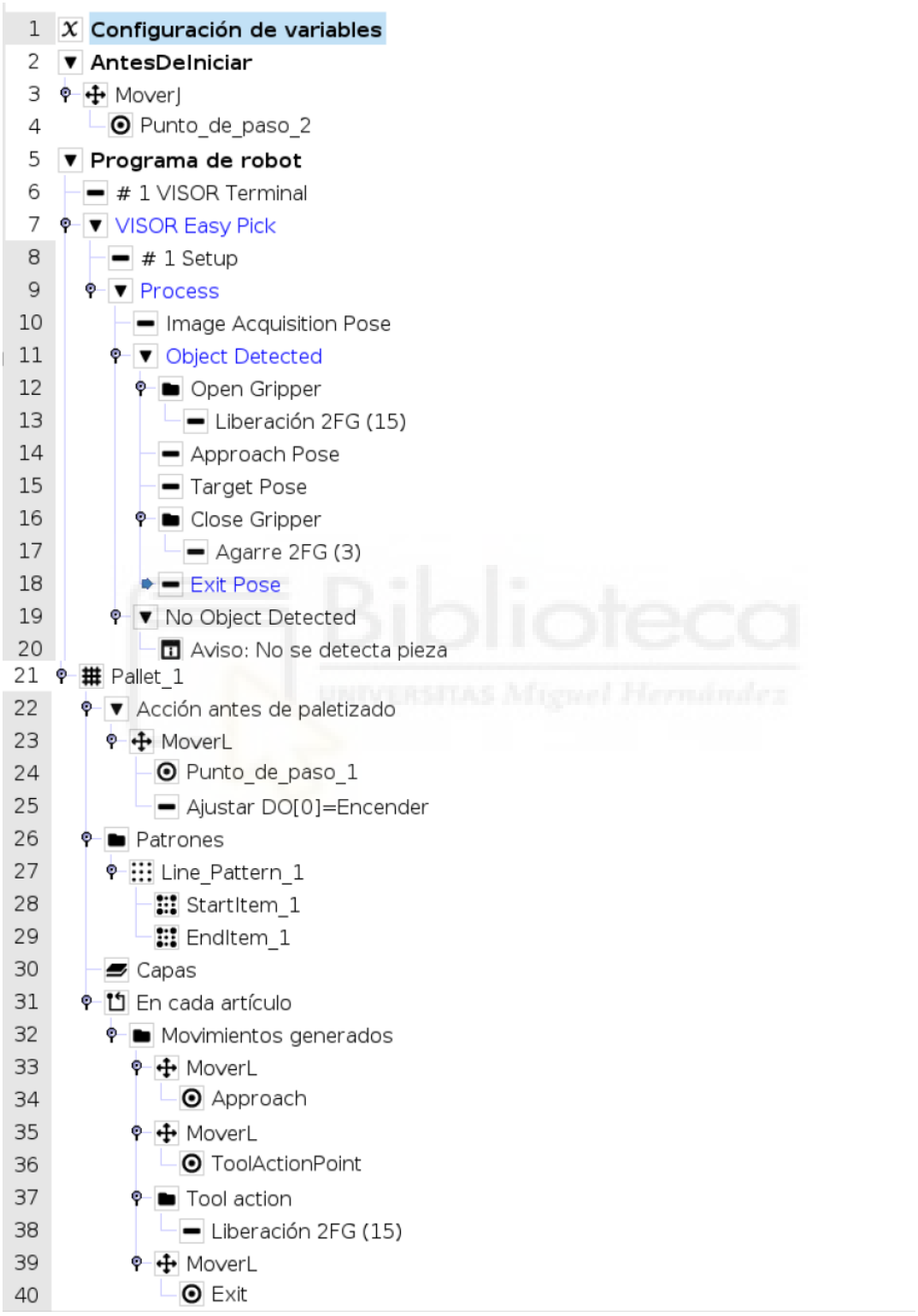


Figura 104. Programa completo en Polyscope antes de introducir el alineador. Fuente: Elaboración propia.

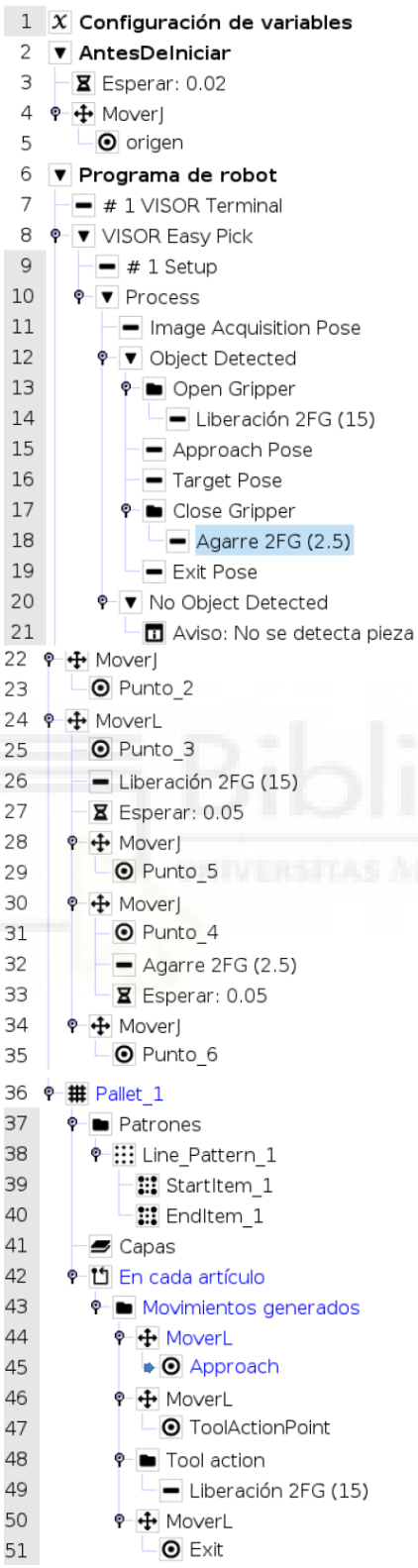


Figura 109. Primera parte del programa en Polyscope después de introducir el alineador. Fuente: Elaboración propia.

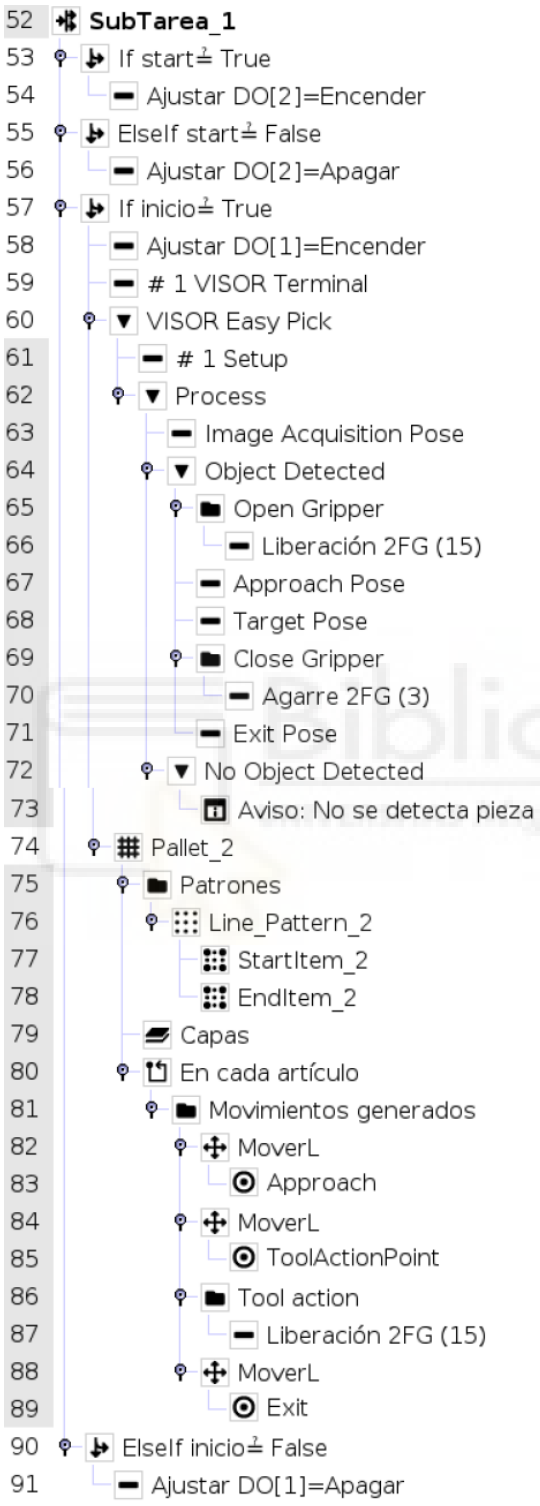


Figura 110. Segunda parte del programa en Polyscope después de introducir el alineador. Fuente: Elaboración propia.