UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Desarrollo e implementación de una aplicación robotizada empleando un robot colaborativo con sistema de visión

Trabajo Fin de Grado

Grado en Ingeniería Eléctrica

AUTOR/A: García de la Asunción, Alejandro

Tutor/a: Gracia Calandin, Luis Ignacio

CURSO ACADÉMICO: 2021/2022

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Trabajo de Fin de Grado

Grado en Ingeniería Eléctrica

Implementación de una aplicación robotizada empleando un robot colaborativo con un sistema de visión

Autor: Alejandro García de la Asunción

Director: Luis Ignacio Gracia Caladín

Departamento: Ingeniería de Sistemas y Automática

isa

Septiembre 2021

Resumen

El trabajo abordará el diseño e implementación de una aplicación robotizada para el robot colaborativo UR3 (de la marca Universal Robots) equipado con un sistema de visión. La tarea que realizará el robot consiste en reconocer distintos tipos de pieza y realizar el montaje de estas de forma adecuada. Dichas piezas serán diseñadas y fabricadas durante el trabajo y se realizarán pruebas experimentales que permitan validar el funcionamiento de todo el sistema.

En el propio trabajo, se ha tratado con aspectos como el diseño y la impresión 3D de las piezas, la programación del robot y la implementación del sistema de visión al robot. Además, a la vez que se explica en esta memoria el procedimiento que se ha realizado, se expondrá los diferentes problemas que han surgido durante este trabajo y cómo se han resuelto.

Palabras clave del TFG: robot colaborativo, visión artificial, Federación Internacional de Robótica, Organización Internacional de Estándares (ISO), posición de captura (Snapshot Position), Teaching, punto de paso, entrada/salida digital, variable.

Resum

El treball abordarà el disseny e implementació d'una aplicació robotitzada para el robot col·laboratiu UR3 (de la marca Universal Robots) equipat con un sistema de visió. La tasca que realitzarà el robot consistí en reconèixer diferents tipus de peça i realitzar el muntatge d'aquestes de forma adequada. Aquestes peces, seran dissenyades i fabricades durant el treball i es realitzaran probes experimentals que permeten validar el funcionament de tot el sistema.

En el propi treball, s'ha tractat con aspectes com el disseny i la impressió 3D de les peces, la programació del robot i la implementació del sistema de visió al robot. A més, alhora que s'explica en aquesta memòria el procediment que s'ha realitzat, s'exposaran els diferents problemes que han sorgit durant aquest treball i com s'han resolt.

Paraules clau del TFG: robot col·laboratiu, visió artificial, Federació Internacional de Robòtica, Organització Internacional d'Estàndards (ISO), posició de captura(Snapshot Position), Teaching, punt de pas, entrada/eixida digital, variable.

Abstract

In this project will address the design and implementation of an application robotized to the collaborative robot UR3 (of the brand Universal Robots) equipped with a vision system. The task that will perform robot consist of recognizing different types of parts and assembly them properly. These parts will be design and manufactured during project and experimental test will be carried out to validate the operation of entire system.

In the project itself, aspects such us design and 3D printing of parts, the programming of robot and implementation of vision system to robot have been dealt with. In addition, while procedure of that has been carried out is explained in this report, the different problems that have arisen during this project and how they have been resolved will be explained.

Keywords TFG: collaborative robot, artificial vision, International Federation of Robotics, International Organization of Standards (ISO), Snapshot Position, Teaching, waypoint, digital input/output, variable.

Documentos del Trabajo de Fin de Grado

Documento I: Memoria Descriptiva

Documento II: Planos

Documento III: Presupuesto

Documento IV: Pliego de Condiciones

Índice de la memoria Descriptiva

Índice de Planos

Índice de Costes

Índice Pliego de Condiciones

Documento I

Memoria Descriptiva

1. Introducción

1.1 Objetivo

El objetivo de este Trabajo de Fin de Grado es el diseño y la programación de un sistema robotizado capaz de realizar un montaje de una serie de diferentes tipos de piezas mediante el uso de un sistema de visión artificial.

De este modo, el robot junto al sistema de visión deberá reconocer e identificar los tipos de piezas que se presentan en una parte de su zona de trabajo (cuyas posiciones y orientaciones serán aleatorias), buscar en la otra zona su otra "pareja" (que tendrán una forma algo diferente para que puedan encajar y al juntar estas dos forman una forma geométrica) y posicionar correctamente la pieza. Este proceso se repetirá mínimo unas 3 veces ya que habrá 3 diferentes tipos de piezas.

1.2 Descripción de la aplicación diseñada

Esta aplicación consiste en reconocer piezas que estarán inicialmente en el campo de visión de la cámara y poder emparejar estas dos mitades por su forma correspondiente.

Para ello, se utilizarán 2 tipos de piezas distintas: un cubo y un cilindro donde cada pieza tendrá dos mitades, una que tendrá un pequeño hueco en el centro y la otra tendrá un saliente de la misma forma que el hueco de su otra mitad. El primer tipo de mitad, las que llevan el hueco en el centro, son las que el robot agarrará el robot y van a estar de forma que este hueco mire hacia abajo mientras que el otro tipo de mitad tendrá el saliente mirando hacia arriba.

Además, se añadirán otras pieza de forma rectangular y cuando el robot las identifique, se las considerará como defectuosas y las tendrá que depositar en una especie de "contenedor".

Este proceso terminará cuando no haya piezas en la parte donde estarán inicialmente las partes a ensamblar y mostrará un mensaje donde indicará al operario el número y porcentaje de piezas ensambladas y defectuosas.

1.3 Alternativas a la aplicación diseñada

Para poder decidir si este método es el más eficiente a la hora de hacer esta tarea, hay que estudiar las diferentes alternativas al robot colaborativo.

En primer lugar, podríamos contratar a un empleado para que hiciese la tarea de ensamblar estas piezas y considerar si son defectuosas o no, pero esto supone varios problemas. Primero supone un gasto mayor que el propio robot ya que este empleado tiene un sueldo al mes y el robot colaborativo solo tienes que comprarlo y hacerle un mantenimiento, lo que supone a la larga un ahorro considerable. Otro problema por considerar es que esta tarea es muy repetitiva y estos empleados tras unas horas de trabajo pueden cometer errores sin contar con el cansancio o los problemas que se pueden presentar a una persona al hacer una tarea repetitiva además que disminuye su productividad cuando pasa cierto tiempo.

Aunque esto se puede solucionar mediante el cambio de puestos en la fábrica cada cierto tiempo y la programación de descansos, se necesitaría un mayor número de trabajadores y una gran coordinación entre estos para evitar parones de la producción durante estos cambios.

En segundo lugar, se podría utilizar un sistema robotizado para esta aplicación supervisado por un operario, pero supondría para nuestro caso un gasto mayor de espacio ya que estos sistemas, aunque tengan el mismo tamaño que nuestro cobot, precisan una jaula para la seguridad de los trabajadores, además que se necesitaría un puesto fuera del área de trabajo para que el operario pueda observar el proceso con claridad y poder manejarlo desde su puesto. Este espacio extra que supone puede hacer que el robot no quepa en la nave o que no se pueda implementar otra línea de montaje por motivos de espacio. Además, supondría una instalación más costosa ya que se requiere un mayor conocimiento para poder programar el robot, se tendría que instalar la celda de seguridad, el robot y el puesto del operario fuera de esta área de trabajo.

Por último, se puede optar la alternativa 0. Consistiría en no ensamblar las piezas y que el propio cliente las ensamble, esta es viable cuando el producto final es sencillo de montar, no requiera una gran precisión para montarlo y se pueda hacer con herramientas sencillas. Pero requiere que en el manual de instrucciones se indique como poder montarlo además que, existe la posibilidad que algunos posibles clientes no compren el producto por el motivo que tienen que montarlo y existe el problema que se pueda envasar una pieza defectuosa.

1.4 Introducción a la robótica colaborativa

La principal diferencia de los robots colaborativos, o también conocidos como cobots, con el resto de los robots es que están preparados para trabajar juntamente con el operario permitiendo así trabajar conjuntamente y tener las ventajas de ambos; el robot aportaría fuerza, resistencia y precisión en las tareas mientras que el operario aporta destreza, flexibilidad y capacidad de resolver problemas (véase en la **figura 1.1** que en el recuadro rojo se muestra el tipo de colaboración que se puede llevar con algunos de estos cobots, donde el área verde es el área de trabajo del cobot y el área naranja es el área de trabajo del operario).

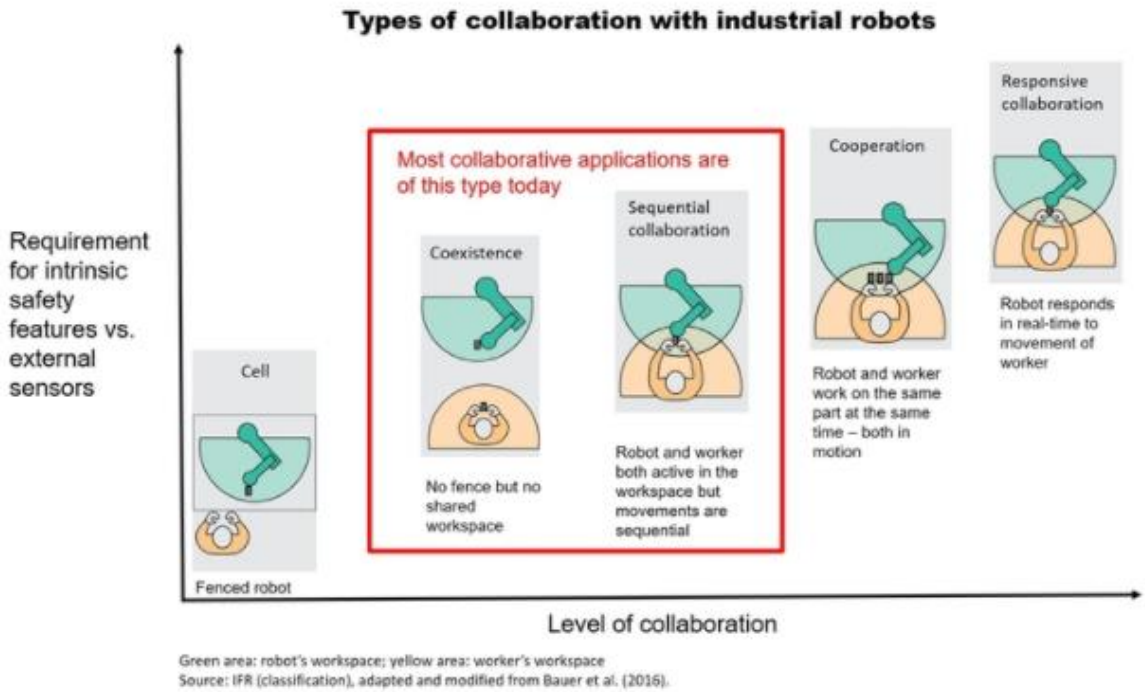


Figura 1.2: Tipos de colaboración con el robot colaborativo

La Federación Internacional de Robótica define 2 tipos de robots colaborativos:

- Un grupo que cumplen con la norma 10218-1 de la Organización Internacional de Estándares (ISO) que especifica requisitos y pautas para el diseño seguro inherente, para medidas de protección e información sobre el uso del robot.
- Otro grupo que no cumple con esta norma, pero no implica que no sean seguros ya que pueden estar siguiendo estándares de seguridad nacionales o internos.

Como es de esperar, estos cobots están diseñados de forma que si el humano entra en contacto directo con el robot ya sea deliberadamente o por accidente, esté garantizada su seguridad. Estas características técnicas que tienen estos robots incluyen sensores de fuerza y velocidad posicionados en base y articulaciones para que cuando se salga cierto margen se detenga la tarea del robot, una construcción, un diseño con contornos redondeados para evitar contornos puntiagudos que puedan dañar al operario (véase en la **figura 1.2**) y una construcción con materiales ligeros para disminuir la inercia del propio robot.

Figura 1.2: Robot colaborativo UR3

Estas características, permiten que algunos modelos de estos robots colaborativos no requieran el uso de jaulas de seguridad. Otras ventajas que tienen son que requieren un grado técnico no muy alto para programarlos y ponerlos en funcionamiento ya que el operador puede configurar el cobot en modo enseñanza y guía el brazo a partir de waypoints o puntos de paso para hacer la tarea. Una vez enseñada esta tarea, se almacena en la memoria y se programa cuando sea necesaria.

Por último, se debe puntuar que no todos los cobots pueden trabajar fuera de una jaula de seguridad ya que pueden ser demasiado pesados o de gran tamaño, que por más que tengan sensibilidad sus sensores y sean totalmente redondeados, pueden lesionar el operario debido a su propia inercia o por un objeto que esté transportando que lleve bordes afilados antes de que se pueda detener por completo.

2. Elección del robot y del sistema de visión.

2.1 Elección del Robot

Para la elección del robot utilizaremos un **robot manipulador** ya que nuestra aplicación consiste en manipular piezas y dentro de este tipo de robots se pueden diferenciar diferentes tipos de estos en el cual explicamos los principalmente usados:

- **Robot cartesiano:** es un robot con múltiples ejes perpendiculares entre sí (un total de 3 ejes) y que se desplazan linealmente cada uno de ellos. Generalmente estos robots suelen ser de gran tamaño y se utilizan para tareas sencillas. Su área de trabajo tiene un aspecto cúbico que dependerá del alcance de los ejes horizontales y verticales.

  En el caso de utilizar este tipo de robot, se utilizaría uno de tres ejes que tiene un total de 3 grados de libertad.

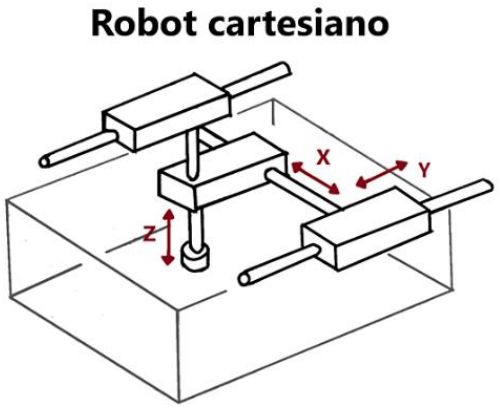


Figura 2.1: Estructura de un robot cartesiano de 3 ejes

- **Robot SCARA:** es un robot de 3 o 4 ejes compuesto generalmente por un par prismático y dos de rotación (en el caso de tener 4 ejes se le añadirá otro de rotación en el eje de la herramienta) y tienen 3 o 4 grados de libertad según el número de ejes que tengan. Se suelen utilizar en tareas repetitivas ya que tienen una mayor velocidad de desplazamiento que los cartesianos. Su área de trabajo es de forma cilíndrica teniendo un área prohibida en torno a la base del robot y en su parte trasera.

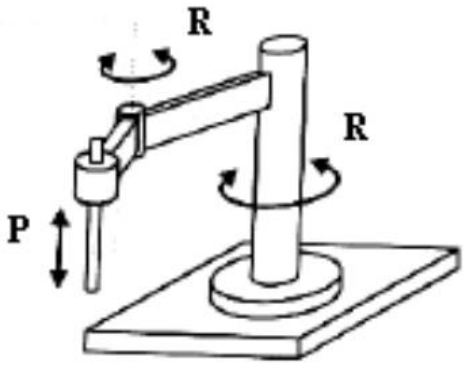


Figura 2.2: Estructura de un robot SCARA de 3 ejes

- **Robot articulado:** es un robot formado por al menos 3 juntas de rotación y su movimiento se basa en el giro de estas, donde una es rotacional (en la zona de la base) y las otras dos angulares. Estos tipos de robots poseen 6 grados de libertad, tienen un área de trabajo con forma esférica y pueden hacer movimientos tanto lineales como circulares.

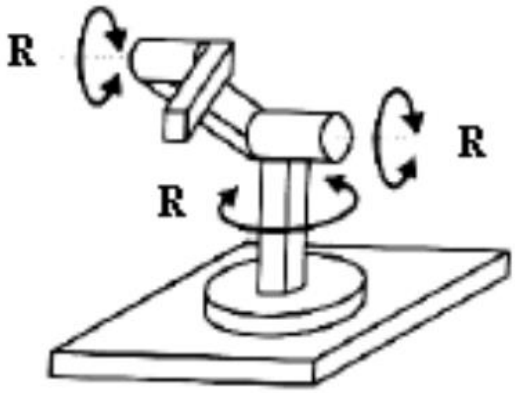


Figura 2.3: Estructura de un robot articulado de 3 articulaciones

En nuestro caso vamos a estos tipos de robots ya que con 6 grados de libertad va a tener más facilidades a la hora de agarrar las piezas ya que pueden ser posicionadas y orientadas de diferente manera y así nos aseguramos una manera más sencilla que la pinza pueda posicionarse adecuadamente.

El robot cartesiano se ha descartado por sus limitaciones a la hora de la orientación de la pieza ya que este no puede girar la herramienta y tendrían que estar las piezas orientadas siempre de una manera. Mientras que en el robot SCARA, aunque podría hacer la aplicación, se ha preferido mejor el robot articulado por tener más opciones de movimiento a este.

Tras haber elegido el tipo de robot, en el laboratorio del departamento disponen de dos robots de este tipo, uno de ABB y otro de Universal Robots.

Por parte del fabricante de ABB, disponemos del IRB-140 (**Figura 2.4**).

Figura 2.4: Robot IRB-140 de ABB

Y cuyas características son las siguientes:

- 6 ejes.
- Carga máxima: 6 kg
- Alcance máximo: 0,81 m (respecto al eje 5)
- Montaje: en cualquier ángulo.
- Área de trabajo:

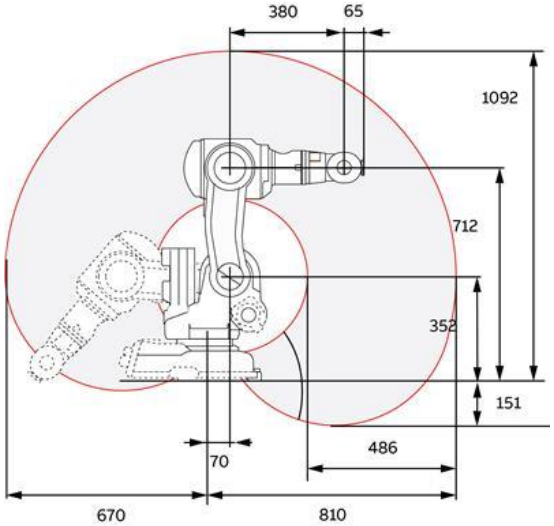


Figura 2.5: Área de trabajo del robot ABB IRB-140.

Por parte de Universal Robots, disponemos del robot colaborativo UR3 (**Figura 2.5**). con un alcance máximo de 500 mm y una carga útil de 3 kg.

Figura 2.6: Robot colaborativo UR3 de Universal Robots

Cuyas características son las siguientes:

- 6 ejes
- Carga útil: 3kg.
- Alcance máximo: 0,5 m.
- Área de trabajo: véase en el Documento II (Planos) de la memoria.

Comparando estos dos robots, finalmente se ha optado por trabajar con el robot colaborativo UR3 que, aunque tenga una menor capacidad de llevar peso y un menor alcance, es suficiente para el tipo de aplicación en que lo vamos a utilizar por lo que supone un ahorro de espacio.

Además, hay que destacar que el robot UR3 tiene un software de programación gratuito a diferencia que el de ABB es de pago y nos permite trabajar junto al robot al ser un robot colaborativo.

2.2 Elección del sistema de visión

En cuanto a la elección del sistema de visión, hemos elegido entre dos tipos de cámara:

- En primer lugar, se ha pensado en utilizar una cámara que se instala en la muñeca del brazo robot. Es un sistema de visión integrado “Wrist Camera” de Robotiq para el modelo UR3 (Figura 2.7).

  Esta cámara está diseñada para trabajar con los robots colaborativos de Universal Robots UR3, UR5 y UR10 donde cada robot tiene su propio modelo.

Figura 2.7: Wrist Camera

- En último lugar, se ha comparado la alternativa de la "Wrist Camera" con un sistema de visión fijo que va instalado en el área de trabajo del robot.

  Para adquirir las imágenes se utilizará la cámara fija JAI CV-M77 (Figura 2.8) y para controlar este sistema de visión, el IPD de Dalsa VA40, mediante el programa de visión artificial de Sherlock.

  Esto supondría que habría que crear un programa para que el robot y la cámara se puedan comunicar.

Figura 2.8: Cámara JAI CV-M77

Finalmente, para esta aplicación, se ha optado por utilizar la "Wrist Camera" ya que al estar diseñada para trabajar con el robot UR3 supone diversas ventajas las cuales vamos a explicar a continuación:

- Al ser un sistema de visión integrado y se instala directamente en la muñeca del propio robot, no hay que pensar en un lugar donde instalar la cámara para que tenga el campo de visión necesario sin que el robot obstaculice su visión.

  Además, la instalación es mucho más sencilla que la cámara fija puesto que ocupa más espacio y necesita un IPD para su funcionamiento.

- En cuanto a la programación, no es necesario establecer una conexión con el robot y la cámara puesto que una vez instalada, el software de la

cámara se controla desde la consola de programación del robot que, a diferencia que la cámara fija, se tiene que programar un programa para la conexión entre el robot y el sistema de visión.

- Por último, hace las tareas de pick and place más sencillas puesto que no hay que realizar una conversión de sistema de coordenadas de la cámara al robot, puesto que la cámara y el robot se emplazan en el mismo lugar.

3. Características del Robot UR3

El brazo robot UR3 es un robot de sobremesa flexible con una gran precisión y está diseñado para trabajar en entornos de trabajo reducidos ya que es el más pequeño de la gama del fabricante de Universal Robots.

Este robot, tiene un peso de 11 kg, tiene una carga útil de 3 kg y un alcance de 500 mm haciéndolo la mejor opción para hacer trabajos ligeros y repetitivos o que requieran una gran precisión: procesos como atornillado, control de calidad, manejo de herramientas entre otros.

Además, tiene una rápida instalación, son fáciles de programar y son más seguros que los robots tradicionales debido a su diseño y sus mecanismos de seguridad.

A continuación, se explicará los principales componentes del robot UR3:

3.1 Brazo del Robot

El brazo del robot está diseñado con 6 articulaciones rotatorias donde tienen un rango de movimiento de +/-360º en cada eje excepto la muñeca 3 que puede girar hacia los dos lados indefinidamente (Figura 3.1).

Figura 3.1: Articulaciones del UR3

Cada eje, está equipado con un motor AC de 3 fases y el movimiento giratorio de cada articulación son independientes a la anterior lo que le permite tener el mismo número de articulaciones que grados de libertad (GDL), es decir, 6. Esto le permite posicionarse la herramienta mediante 6 parámetros, 3 de posición (x, y, z) y 3 de orientación (α, β, γ) lo que permite manipular los objetos en el espacio de trabajo de cualquier manera.

3.2 Caja de Control

Es la unidad de control donde se encuentra la placa base, el tablero de control de seguridad y USB (Figura 3.2).

Figura 3.2: Interior de la Caja de control UR3

La **placa base** es el ordenador encargado de trabajar con los datos para realizar las tareas programadas, esta dispone de una conexión USB y de Ethernet.

El **tablero de control de seguridad** maneja todas la entradas y salidas del robot y la conexión entre los diferentes dispositivos al robot como, por ejemplo: sensores, dispositivos de seguridad, pulsadores e interfaz de la máquina.

Estas entradas y salidas se distribuyen de la siguiente forma:

- Las de color **amarillo**: Señales de seguridad
- Las de color **gris**: Entradas y salidas digitales de propósito general
- Las de color **verde**: Entradas y salidas analógicas.

Por último, la memoria USB contiene los softwares necesarios para operar con el robot.

Entre estos softwares hay que destacar el sistema operativo Linux, la interfaz de programación PolyScope y los programas guardados.

3.3 Consola de Programación.

La consola de programación es un dispositivo con una pantalla táctil que incluye un botón de alimentación y un pulsador de emergencia en la parte delantera. En la parte lateral derecha tiene un puerto USB y en la parte trasera un botón de movimiento libre.

Figura 3.3: Consola de Programación UR3

En la **pantalla táctil,** se nos mostrara la interfaz de programación PolyScope que permite encender el robot, programarlo y controlar las señales de dispositivos periféricos.

El **botón de alimentación** (botón gris) como su propio nombre indica, nos permitirá encender o apagar la consola de programación.

El **botón de parada de emergencia** (botón rojo) nos permite parar el robot mientras está ejecutando un programa en caso de una emergencia.

El **botón de movimiento libre** de la parte trasera nos permite mover al robot de forma manual al ser presionado.

Y el **puerto USB** de la parte lateral nos permite cargar o guardar los programas realizados durante este proyecto.

4. Características del equipo de visión

Como hemos mencionado anteriormente, este equipo de visión está diseñado para trabajar con los robots colaborativos de Universal Robots y se instala en la muñeca del brazo robot.

4.1 Hardware del sistema de visión.

Este modelo tiene un peso de 160 g y soporta una carga máxima de 10 kg, además, posee 4 orificios de atornillado para instalarlo en la muñeca de nuestro brazo robótico en apenas unos minutos.

Se alimenta a 24 V de DC mediante un cable flexible de 10 m conectado en la parte lateral de la cámara y a la caja de control del robot.

Para captar las imágenes correspondientes cuenta con un sensor CMOS con estas características:

- máxima resolución: 5 Mpx a 0,3 fps
- Máxima tasa de fotogramas: 30 fps a 0,3 Mpx
- Enfoque: de 70 mm a infinito, además, tiene autoenfoque
- Mínimo campo de visión: 10 x 7,5 cm
- Máximo campo de visión: 36 x 27 cm
- Angulo de visión horizontal: 50º (figura 4.1)
- Angulo de visión vertical: 39º (figura 4.1)

Como podemos observar en las características, la cámara puede sacrificar resolución para ofrecer una mayor tasa de fotogramas o a la inversa, lo ideal es que estos dos parámetros estén equilibrados, donde pueda ofrecer una resolución para poder reconocer los objetos correctamente y una tasa de fotogramas aceptable. Además, cuenta con 2 luces compuesta por 3 LEDS a los lados de la cámara tal y como se muestra en la imagen (figura 4.1).

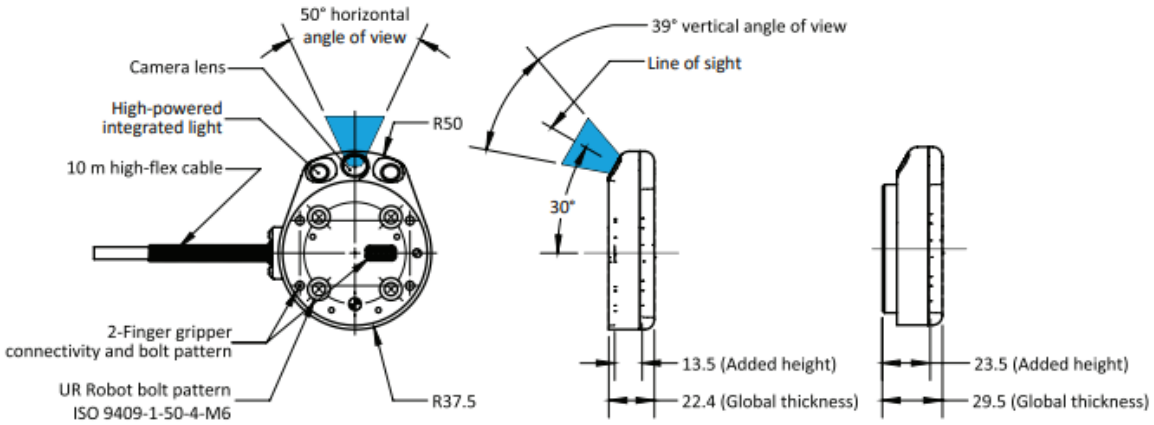


Figura 4.1: plano general de la "Wrist Camera"

Por último, cabe destacar que los puertos de conexión con la pinza de la cámara (ver figura 4.1 en parte central de la cámara) no se van a utilizar puesto que la pinza del laboratorio no es del modelo de Universal Robots, en su lugar, se instalará en la caja de control.

4.2 Software del equipo de visión.

El software de este sistema está diseñado para ser usado con los robots de Universal Robots. Por esto, una vez instalada se programa y se trabaja desde la propia consola de programación que dispone el brazo robot.

Está diseñada para trabajar en planos 2D. Por tanto, una vez definido el espacio de trabajo, será capaz de reconocer tanto su posición (x, y) tanto su orientación (eje z) del objeto enseñado.

También cuenta con una interfaz sencilla, para que la gente que no tenga muchos conocimientos sobre los sistemas de visión pueda programarla para tareas sencillas de "pick and place".

5. Instalación del robot

5.1 Montaje del robot

La base del brazo robótico debe de estar instalada en una superficie bastante resistente para soportar 10 veces su par de torsión total, 5 veces el peso del brazo robótico y no debe haber vibraciones; esta base, se ensamblará a la superficie mediante pernos M6 en sus 4 orificios que dispone.

Respecto a la caja de control, debe de estar en un sitio con un espacio mínimo de 50 mm por cada lado para que el flujo de aire sea lo suficiente para que la refrigeración de la caja de control.

5.2 Espacio del robot

También hay que tener en cuenta en la instalación el espacio de trabajo del brazo robot ya que una mala posición puede llegar a que el robot no pueda hacer la tarea o que la haga ineficientemente.

Este espacio de trabajo tiene una forma esférica de 500 mm desde la base del robot y tiene una zona cilíndrica por encima y debajo de la base del robot que se debe de evitar. Ya que las juntas del robot para hacer un movimiento en esa zona se mueven rápidamente, aunque la herramienta lo hiciera despacio, esto puede provocar una serie de riesgos además que el robot trabaje ineficientemente.

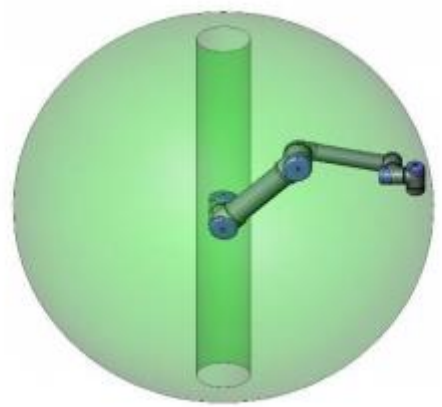


Figura 5.1: Espacio de trabajo del brazo robot junto al volumen a evitar.

Otra cosa por destacar es que en determinadas zonas de la zona de trabajo existe un peligro de enganche de las juntas debido a las propiedades físicas del brazo, para evitar estos, la pieza debe de estar al menos a 450 mm de la base del robot si se realiza un movimiento radial y de al menos de 200 mm de la base si se realiza un movimiento en la dirección tangencial (Figura 5.2).

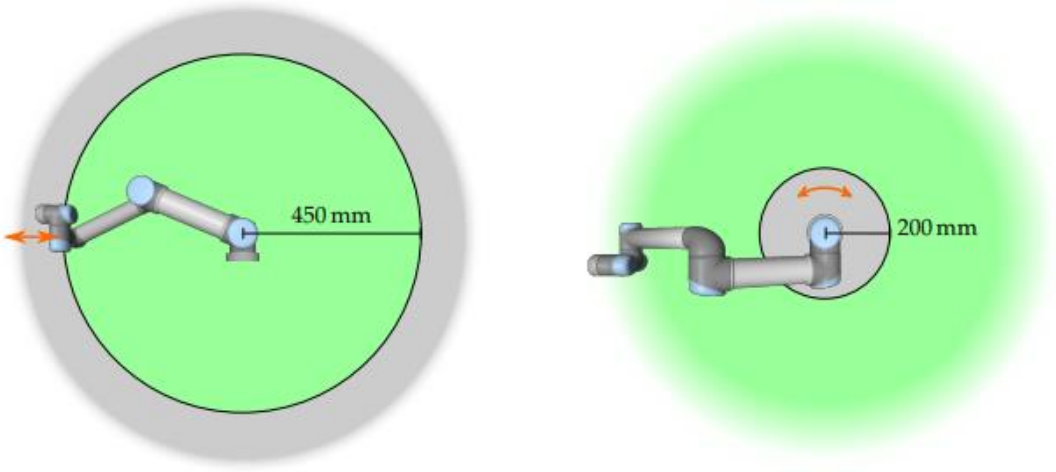


Figura 5.2: Recomendaciones para evitar enganches; en la derecha para movimientos radiales y en la izquierda para movimientos en la dirección tangencial.

5.3 Instalación eléctrica

Una vez instalado el brazo robot, se conecta el cable del brazo robótico al conector situado en la parte inferior de la caja de control. La caja de control recibe suministro eléctrico mediante un enchufe IEC que se conecta a una toma de corriente.

Para la instalación del robot, la instalación eléctrica del edificio debe tener los siguientes requisitos:

- **Conexión a tierra:** Es una serie de piquetas metálicas (generalmente de cobre) que están clavadas a la tierra. Su función principal es disipar corrientes indeseadas o servir de referencia a algunas protecciones o aparatos de medida.
- **Fusible principal:** es una protección térmica para proteger el robot, en la que normalmente se utilizan interruptores automáticos.
- **Dispositivo para corriente residual:** es un interruptor diferencial para evitar contactos indirectos a las personas. Estos interruptores requieren una conexión a tierra y normalmente tiene una tolerancia entre 30 y 300 miliamperios, aunque es más recomendable utilizar el de 30 miliamperios debido que 300 miliamperios pueden provocar daños o incluso casos de muerte a las personas.

Por último, en la siguiente tabla se muestran las especificaciones de la red eléctrica que debe de tener antes de conectar el robot:

Parámetro	Mín.	Típ.	Máx.	Unidad
Tensión de entrada	100	-	240	VCA
Fusible externo de red eléctrica (100-200 V)	8	-	16	A
Fusible externo de red eléctrica (200-240V)	8	-	16	A
Frecuencia de entrada	47	-	63	Hz
Potencia en espera	-	-	0,5	W
Potencia nominal de funcionamiento	90	150	325	W

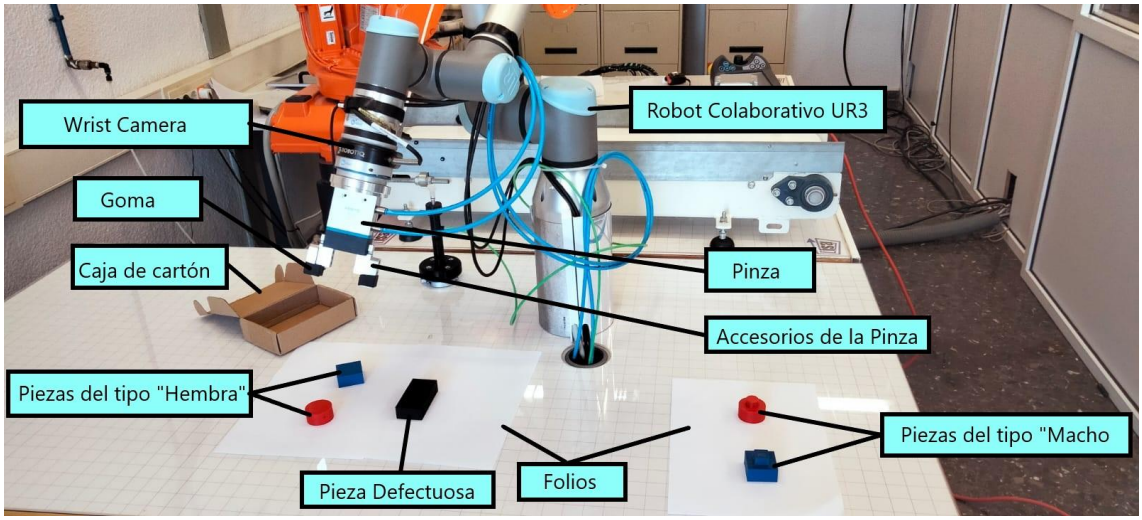
Figura 5.3: Tabla de especificaciones de la red eléctrica.

6. Materiales y herramientas utilizadas

A continuación, se mostrarán los diferentes materiales y herramientas que se han utilizado durante este proyecto:

- Brazo robot UR3 de Universal Robots. Se encarga de manipular las piezas en su zona de trabajo y de ejecutar el programa.
- Sistema de visión "Wrist Camera" de Robotiq. Se encarga de localizar e identificar los diferentes tipos de piezas que se presentan en la zona de trabajo del UR3.
- Pinza mecánica. Pinza acoplada al extremo del robot UR3 accionado mediante una electroválvula que se activa mediante una salida digital de la caja de control para agarrar o soltar las piezas respectivamente. Esta pinza tiene un máximo de apertura de 3 cm.
- Accesorios de la Pinza mecánica. Dos piezas de metal que se acoplan la pinza y se pueden acoplar en el interior o exterior de las piezas de la pinza. Según su colocación, le permitirá al robot operar con piezas más grandes o pequeñas.
- Goma. Trozo de goma negro que se le ha acoplado a los accesorios de la pinza para mejorar el agarre de la pinza.
- Consola de programación. Consola que permite programar, configurar y controlar tanto el robot como sus salidas y entradas. Además, nos permite importar o exportar programas que hayamos hecho en la consola o en un ordenador aparte y calibrar el equipo de visión.
- Piezas para ensamblar. Son piezas de plástico diseñadas en 3D. Hay de dos tipos: una con forma cilíndrica y otra con forma cúbica; cada una se pueden separar en dos piezas diferentes una que la denominaremos como "macho" ya que tiene un saliente y otra como "hembra" ya que tiene un agujero en el centro.
- Piezas defectuosas. Son piezas diseñadas en 3D que se consideran defectuosas ya que no tienen la misma forma que las anteriores. Cuando el robot identifique una de ellas las depositara en un contenedor aparte.

- Caja de cartón: una caja de cartón que servirá como contenedor para las piezas defectuosas.
- Cartulinas o folios: Puestos sobre la mesa para evitar reflejos de la mesa y facilitar la identificación de las piezas al equipo de visión



7. Diseño e Impresión de las piezas

7.1 Diseño de las Piezas en AutoCAD.

Para el diseño de las piezas se ha utilizado principalmente las herramientas 3D que nos ofrece el programa AutoCAD y para trabajar con ellas primero tenemos que activar en la esquina inferior derecha la “Referencia a objetos 3D” que se sitúa en un desplegable al activar este botón “  ”, después nos debemos asegurar que este activado las herramientas del “Modelado 3D” que se encuentra en este botón encontrado en la misma barra de herramientas que el anterior “  ” ya que por defecto el programa tiene activadas las herramientas de “dibujo y anotación” que son principalmente para hacer planos en 2D.

En este diseño de piezas, utilizaremos la escala de una unidad de medida en AutoCAD es igual a un milímetro para evitar complicaciones luego en la conversión a objeto 3D.

Después de activar esta barra de herramientas me dispondré a explicar brevemente los comandos utilizados para el diseño de estas piezas:

- Comando línea “  ” o polilínea “  ”. Comando que nos permite dibujar una línea o una serie de líneas sucesivas(en el caso de la polilínea) mediante la definición de dos puntos donde puedes introducir la distancia de estas líneas.
- Comando Extrusión. Comando que nos permite a partir de una forma en 2D formar una figura 3D, donde definiremos la forma a convertir en 3D y que distancia tiene en la coordenada Z.
- Comando Presionartirar “  ”. Comando parecido a Extrusión, pero este lo hace a partir de un área delimitada usado principalmente para figuras más complejas.
- Comando mover “  ”. Comando que nos permite mover elementos a partir de un punto base que definamos.
- Comando circulo. Comando que nos permite dibujar un círculo de diferentes maneras, pero en mi caso se ha dibujado mediante la definición de un centro y el radio de este.
- Comando Solido, Unión “  ”. Comando que nos permite unir dos sólidos sumando sus volúmenes, este comando requiere primero que los dos solidos estén en su posición correcta antes de utilizar este comando.
- Comando Solido, Diferencia “  ”. Comando parecido al Solido, Unión, pero este nos permite restar los volúmenes de dos sólidos. Para ello, primero definimos el sólido al que le queremos quitar un volumen y luego el volumen que le queremos restar.
- Podemos cambiar la vista de los sólidos en la pestaña visualizar y en “estilos visuales” podemos cambiar la manera de cómo se ven los sólidos, en mi caso las más utilizadas por mí son la “realista” para ver el resultado final y la de “rayos x” para que tenga algo de transparencia las piezas y poder inspeccionar correctamente los huecos.

7.2 Impresión en 3D de las piezas.

Para poder imprimir las piezas en la impresora necesitamos convertir el plano del AutoCAD a un archivo STL que puede leer la impresora. Para ello utilizaremos la aplicación de AutoCAD de AutoCAD Print Studio donde nos permite hacer esta conversión.

Para ello nos iremos a imprimir situado en el menú de inicio de AutoCAD y seleccionamos “impresión en 3D” donde nos pedirá que seleccionemos un sólido a imprimir.

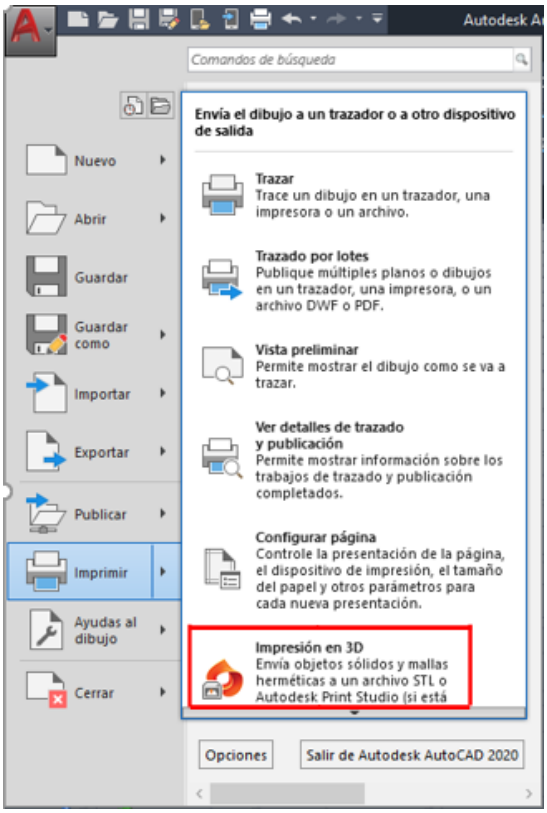


Figura 7.1: Menú de Imprimir del AutoCAD.

Una vez seleccionado el sólido a imprimir, nos saldrá una ventana donde deberemos introducir la escala de nuestra pieza en milímetros en los tres ejes (x, y, z) y le daremos a aceptar y nos convertirá nuestra pieza a un archivo STL donde la impresora lo puede detectar.

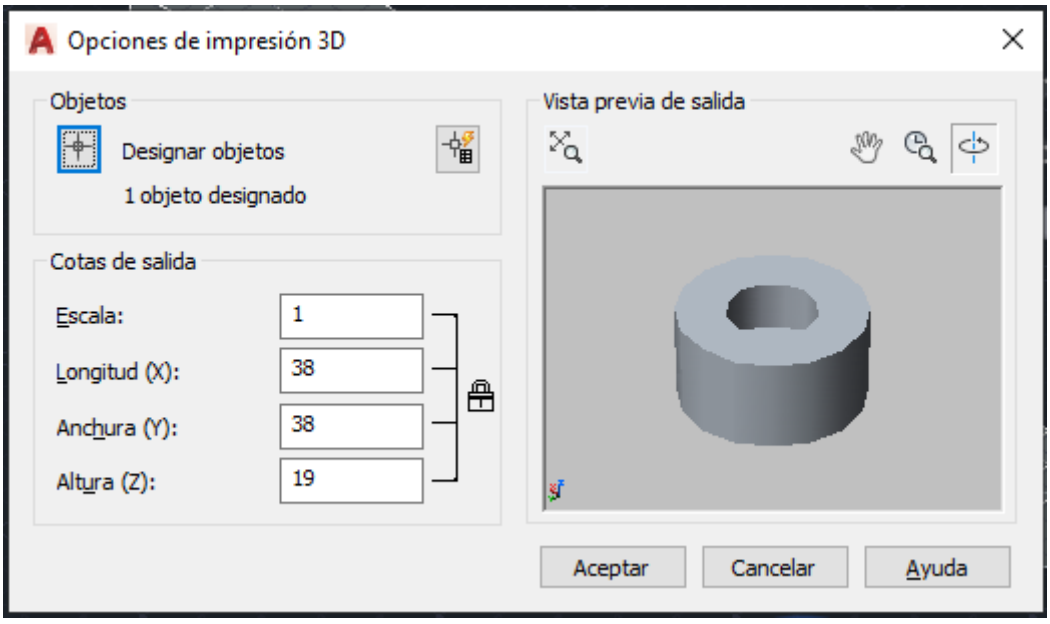


Figura 7.2: Pantalla de designación de escalas en la creación del modelo STL.

Una vez creado el archivo STL ya lo tenemos listo para que la impresora 3D lo imprima; en mi caso, contraté una empresa de impresión 3D llamada “Impresión 3D Valencia”.

Una vez recogidas las piezas, los tres tipos de piezas tienen esta forma:

Piezas Cuadradas:

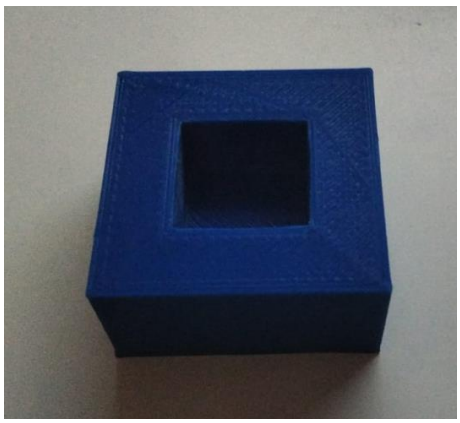


Figura 7.3: Pieza cuadrada "Hembra".

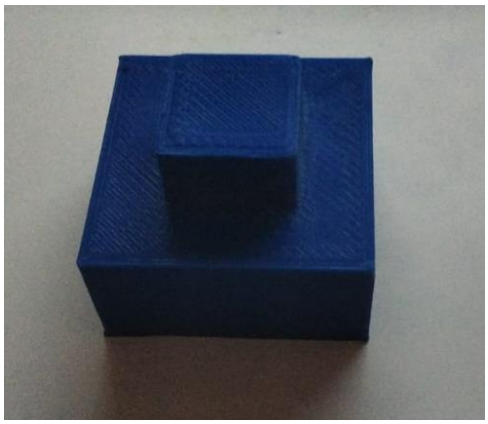


Figura 7.4: Pieza cuadrada "Macho".

Piezas Cilíndricas:

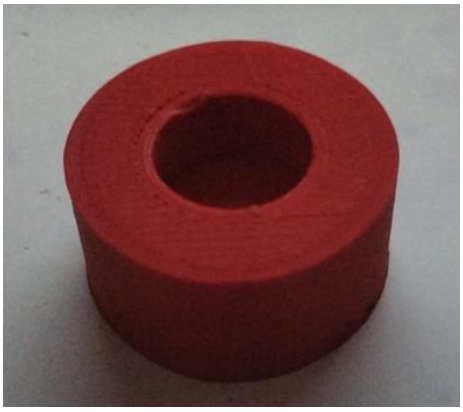


Figura 7.5: Pieza cilíndrica "Hembra".

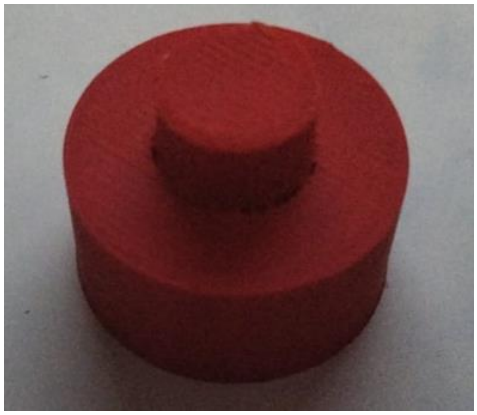


Figura 7.6: Pieza cilíndrica "Macho".

Piezas Defectuosas:

Figura 7.7: Pieza Defectuosa.

8. Configuración del Equipo de visión

Una vez instalada en la muñeca del brazo robot la “Wrist Camera”, primeramente, hay que verificar que esté bien conectada. Esto lo podemos comprobar en la pantalla de instalación en esta pantalla nos debe mostrar que hay una pestaña llamada cámara, además, en la pantalla del programa del robot se nos mostrará comandos adicionales relacionados con la cámara.

Una vez comprobado que la cámara esté bien instalada, debemos definir una “Snapshot Position” o posición de captura para que le podamos enseñar al robot los diferentes tipos de piezas. Una “Snapshot Position” es una posición definida del robot desde la cual se va a utilizar la cámara para distinguir los objetivos y también es un punto de referencia.

Siempre que creamos una “Snapshot Position” nueva tendremos que calibrar la cámara para esa posición, ya que esta se deberá de comprobar que es capaz de distinguir bien los diferentes objetivos en esa posición.

En mi caso he creado dos “Snapshot Position” una llamada “Snapcam_1” para detectar los diferentes tipos de piezas (recogida de las piezas) y otra llamada “Snapcam_2” para el ensamblaje de estas.

8.1 Creación de las “Snapshot Position”.

Para crear la posición de captura es requerido que definamos una “función” de tipo punto.

Para ello, nos iremos al menú de “Instalación”, nos iremos a la pestaña de “Función” y seleccionamos que es un “punto”. Acto seguido este punto lo ajustaremos para que tenga la posición deseada y marcaremos una casilla que pone “Variable”; esto nos permite utilizar esta función como una variable, por último, añadiremos otro punto y verificaremos que están los dos puntos en la pestaña de “Variable”, en mi caso me aparecerá dos puntos, uno llamado “Snapcam_1” y otro llamado “Snapcam_2”.

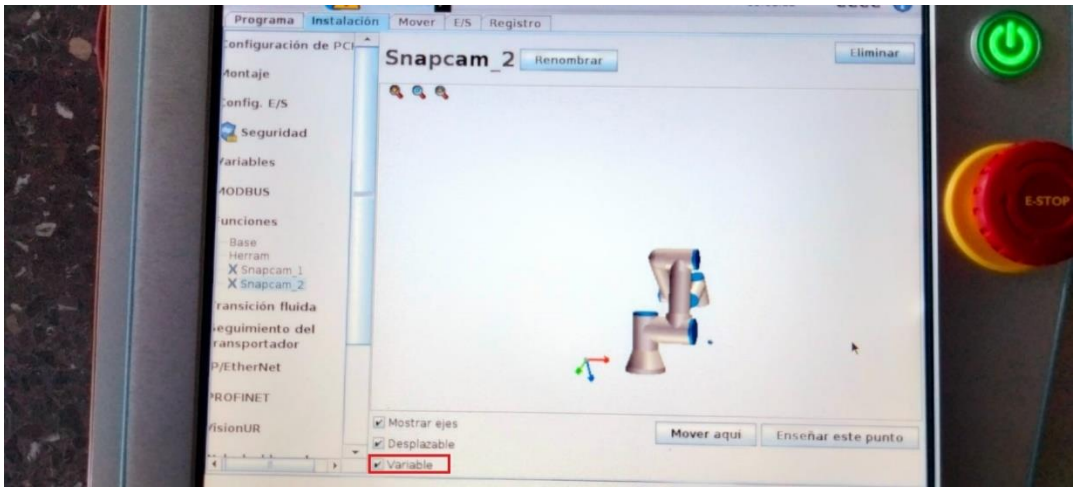
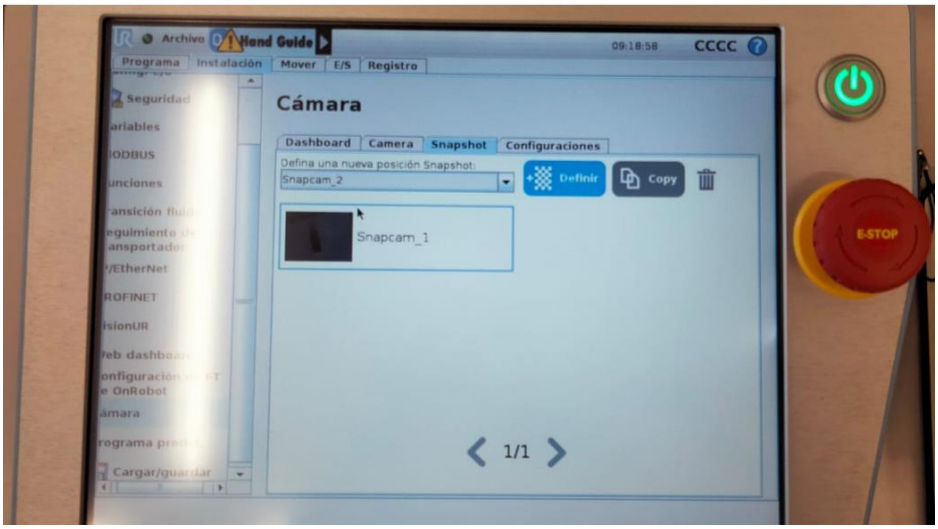


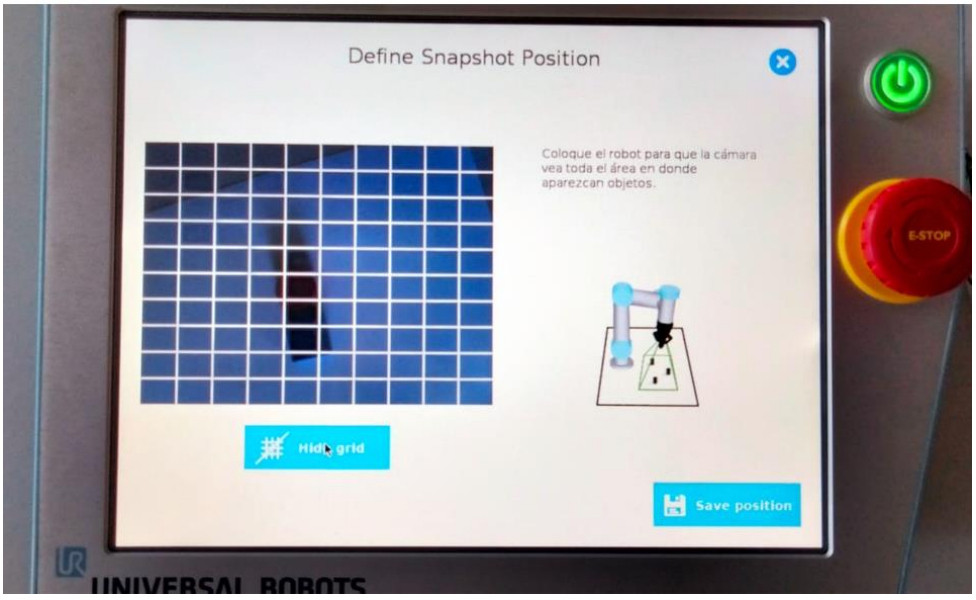
Figura 8.1: Pantalla de función en el punto Snapcam_2. En el recuadro rojo se muestra la casilla de variable la cual debemos de marcar.

Una vez creadas estos dos puntos, en la pestaña de “Cámara” en el mismo menú, seleccionaremos la pestaña de “Snapshot Position” y elegiremos un punto para cada una de estas.



<u>Figura 8.2:</u> Pestaña de Cámara en la ventana de Snapshot para definir posiciones de captura.

Una vez seleccionada, le daremos al botón de “Definir” y nos aparecerá una pantalla donde se muestra lo que está observando la cámara. En esta pantalla seleccionamos la opción de mostrar la rejilla o “Grid” y deberemos comprobar que las piezas para identificar sean de un tamaño mínimo al de un cuadrado de dicha rejilla ya que esto depende de que la cámara detecte bien las piezas y en caso contrario deberemos ajustar la posición en esta pantalla. Una vez comprobado, le daremos a “guardar posición”.



<u>Figura 8.3:</u> Pantalla de confirmación de la creación de la posición de captura con la opción de “grid”.

El último paso por realizar para crear nuestra “Snapshot Position” es calibrar la cámara.

Una vez realizado el paso anterior, nos aparecerá una pantalla que tiene un botón para calibrar, pero antes deberemos colocar la plantilla de calibración de forma que se vea todos los recuadros (Figura 8.4).

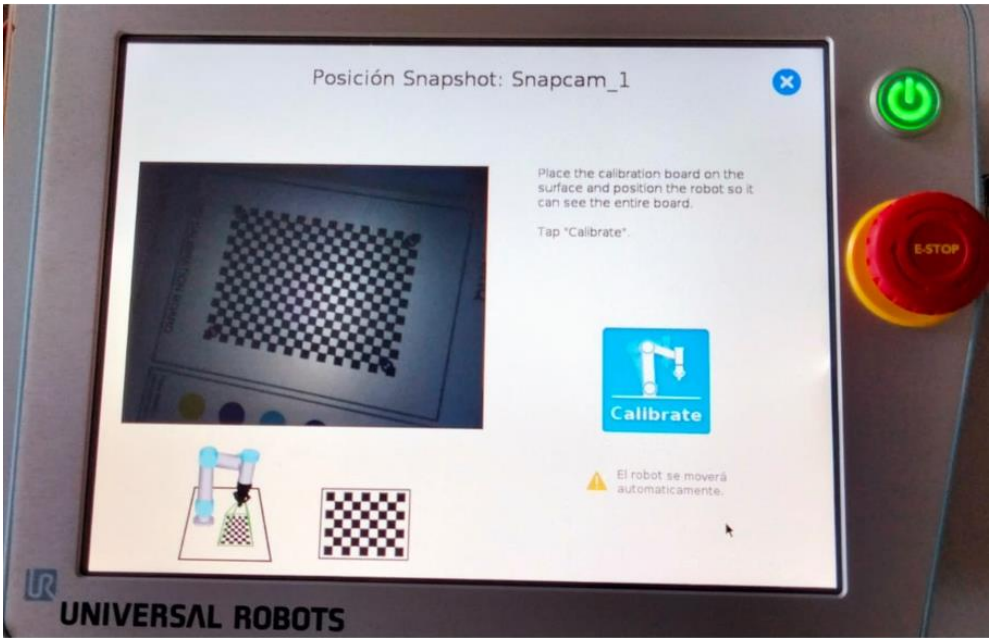


Figura 8.4: Pantalla de inicio de la calibración con su plantilla de calibración correctamente colocada.

Una vez puesta esta plantilla le daremos al botón y el robot comenzará a tomar 36 fotos desde diferentes ángulos donde el robot tardará aproximadamente 10 minutos en terminar la calibración. Cuando el robot haya terminado, deberemos comprobar en las primeras fotos que la cámara reconozca la rejilla (Figura 8.5) y en las últimas fotos que reconozca los cuadrados negros con precisión (Figura 8.6). En estas últimas fotos las líneas nos aparecerán mediante una escala de colores y deberemos comprobar que no haya ninguna línea roja lo que esto supondría que la precisión en esa zona es de ±4mm y es recomendable recalibrar el equipo.

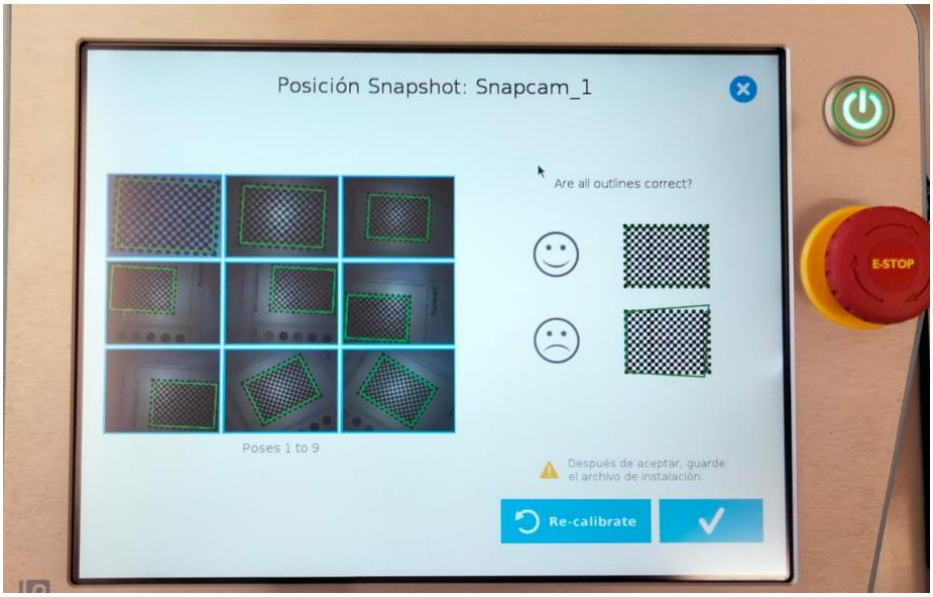


Figura 8.5: Comprobación de la identificación de la rejilla en la calibración.

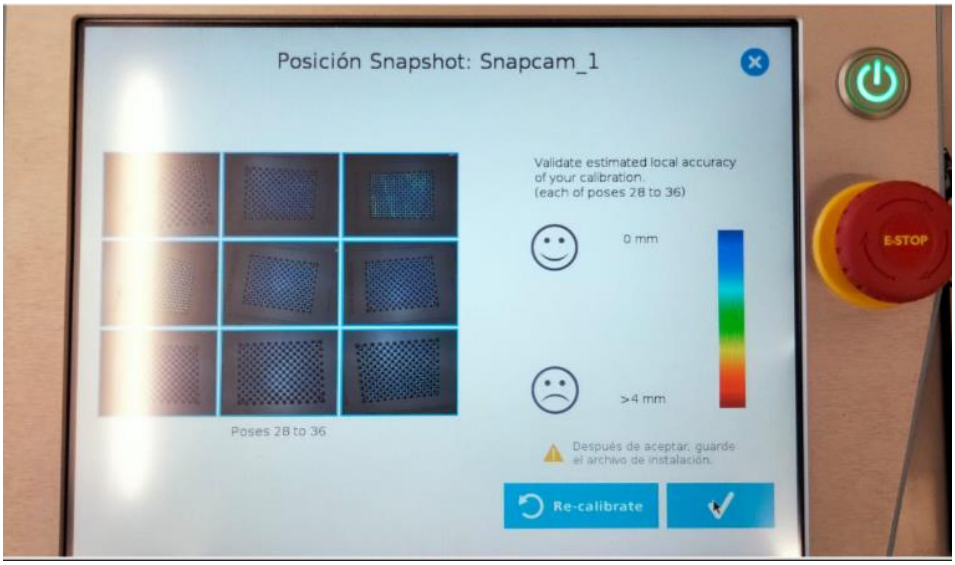


Figura 9.6: Comprobación de la identificación de recuadros en la calibración.

8.2 "Teaching" de las piezas.

Una vez calibrado y establecido la "Snapshot Position", toca enseñarle al robot los diferentes tipos de piezas y que debe hacer con ellas. Para esto, utilizaremos principalmente el comando "Camera Locate" situado en el menú de programación del robot en la pestaña de Estructura y en el apartado de "URCAps", que son los comandos referidos a la cámara. Una vez lo seleccionamos, nos aparece en el árbol de comandos el propio comando de "Camera Locate" y un subapartado de este que dice "para los objetos encontrados" donde podemos hacer una serie de instrucciones referidas al objeto en cuestión(Figura 9.7).

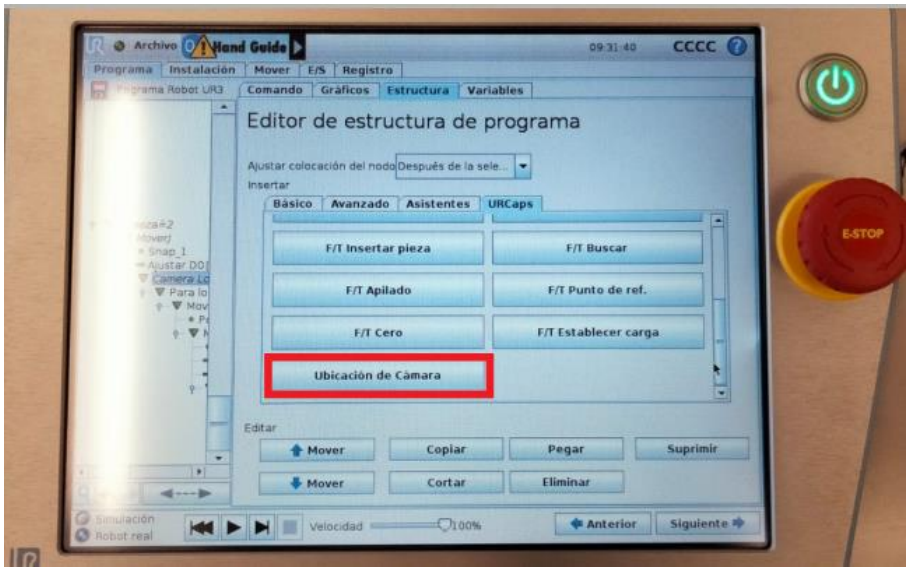


Figura 9.7: Pestaña de la URCaps; el recuadro rojo resalta el comando "camera locate".

Si pinchamos en el árbol el comando de “Camera Locate”, en la pestaña de Comando, nos saldrá un botón de “Teaching” donde le podremos enseñar un tipo de pieza al robot(Figura 9.8).

Figura 9.8: Ventana del “camera locate” con el botón de teach.

Depende del tipo de pieza que le queramos enseñar al robot, podemos utilizar 2 métodos diferentes: el método automático y el método paramétrico(Figura 9.9).

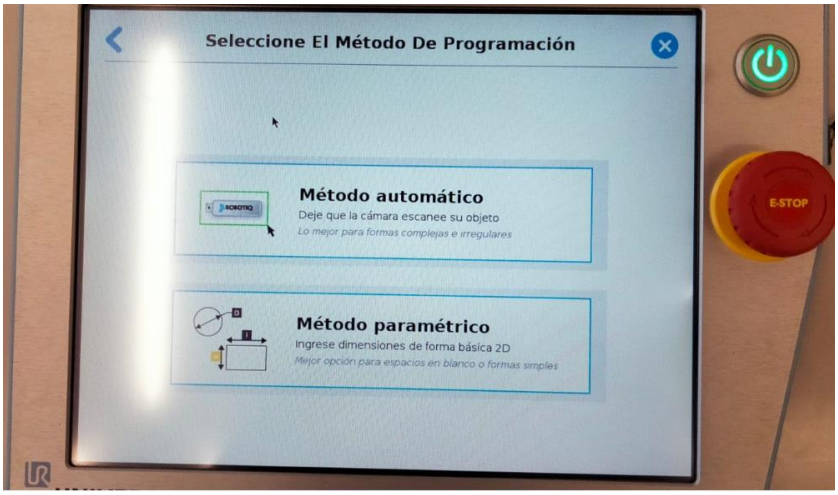


Figura 9.9: Pantalla de la selección del modo de la construcción del modelo 3D.

El **método paramétrico** construye el modelo 3D que necesita a partir de parámetros en 2D de 4 tipos de figuras simples (cuadrados, rectángulos, círculos y anillos) donde en el caso de los cuadrados y rectángulos nos pedirá los lados que lo componen y la altura de la pieza, mientras que, en el caso de los anillos y círculos nos pedirá los diámetros (Interior y exterior en el caso de que sea un anillo) y la altura de la pieza. Es el más aconsejable de usar cuando la pieza lo permite, ya que es el método que es más rápido y robusto puesto que, el robot buscará la silueta de la pieza e ignora espacios en blanco ocasionados por reflejos.

El **método automático** construye el modelo 3D que necesita a partir de una serie de fotos desde diferentes ángulos y un escaneo del objeto. Se suele utilizar para todas las piezas que no se pueden detectar mediante el método paramétrico ya que su silueta es irregular o no tiene la forma de estas 4 figuras. Además, es aconsejable que, para este método, las superficies de las piezas y del área de trabajo no sean reflectantes puesto puede hacer que falle en la detección del objeto y esto se repercuta a repetir el proceso de escaneo o una mala construcción del modelo debido a espacios en blanco por los reflejos lo cual puede generar fallos en el programa.

Para este proyecto, solo se ha utilizado el método paramétrico, ya que nuestras piezas tienen la silueta de estas figuras simples explicadas con anterioridad.

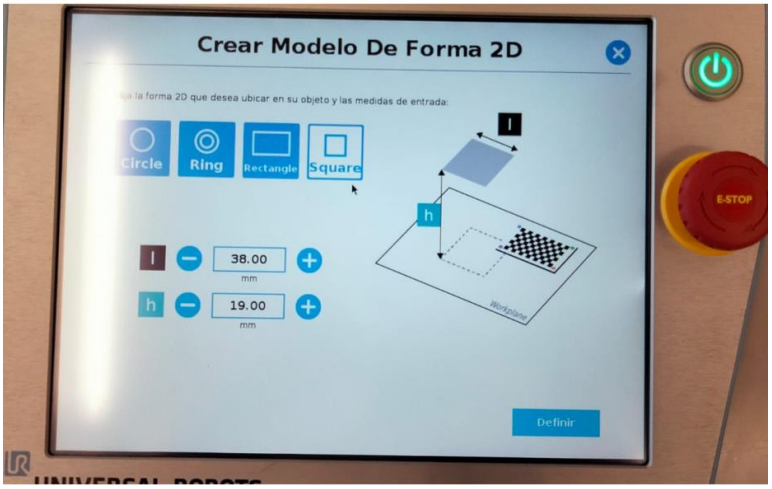


Figura 9.10: Pantalla de configuración del método paramétrico.

Una vez seleccionado el método paramétrico, se tiene que elegir el tipo de pieza simple a tratar (en mi caso se utilizó el circulo, el cuadrado y el rectángulo) donde cada uno nos pedirá que se indique la altura de la pieza y los lados o el diámetro que tienen. (Véase en la Figura 9.10).

Una vez se haya indicado el modelo 2D de la pieza a reconocer, se pasará al último paso, en este paso se puede ajustar el umbral de detección o “Detection threshold”. Este parámetro determina el umbral de coincidencia que debe de tener la pieza con el modelo 3D que ha creado el robot para que la cámara lo reconozca como este tipo de pieza, es recomendable que este umbral sea lo más alto posible ya que si es demasiado bajo, puede confundir una pieza parecida como buena y un umbral demasiado alto puede hacer que no detecte la pieza que se quiere manipular donde se ha ajustado un valor entre 70-90 %.

Una vez establecido el umbral de detección, toca hacer un test para ver si la pieza es detectada correctamente, además, se establecerá la posición y orientación base de esta.

Figura 9.11: Pantalla del ajuste del umbral de detección, del test de detección y de la toma de posición de referencia.

Una vez realizado el "Teach" correctamente es recomendable no mover la pieza antes de establecer los puntos de paso para agarrar la pieza o hacer el ensamblaje de esta, debido a que esta posición se va a tomar como referencia para ajustar estos puntos de paso.

9. Programación del Robot

En concreto, la programación del robot UR3 no requiere conocimientos de algún lenguaje de programación. Más bien es una serie de instrucciones de forma secuencial que queremos que haga, esto lo podemos ver principalmente en la interfaz de este, de hecho, lo único que podemos ver de código son las instrucciones que le hemos puesto en el árbol de programación situado a la izquierda de la pantalla en el menú de programación.

9.1 Conceptos Básicos

Menú "Archivo": Es un menú desplegable que está en todas las pantallas donde tiene un botón característico (Figura 9.1), en este menú se puede guardar o cargar programas y salir al menú de inicio.

Figura 9.1: botón desplegable del menú de Archivo.

Barra de menús: Barra situada en la parte superior de la pantalla que nos permite cambiar entre los diferentes menús; entre estos están: menú de programación, menú de instalación, menú de las entradas y salidas, menú de movimiento y el menú de registro.

Menú de Programación: menú que nos permite modificar o crear un programa para el robot, ejecutar dicho programa y hace un seguimiento de este a tiempo real mientras el robot lo está ejecutando.

Figura 9.2: Menú de Programación.

Menú de Instalación: menú que nos permite modificar los aspectos relacionados con instalación del robot como el seguimiento de la PCH, la carga útil, la configuración de la cámara, creación y modificación de variables globales para el programa, renombrar entradas o salidas, etc.

Figura 9.3: Menú de Instalación.

Menú de entradas y salidas (E/S): menú que nos permite controlar las entradas y salidas del robot, tanto digitales como analógicas. Este menú nos permite tanto modificar su estado como ver en qué valor están(Figura 9.1).

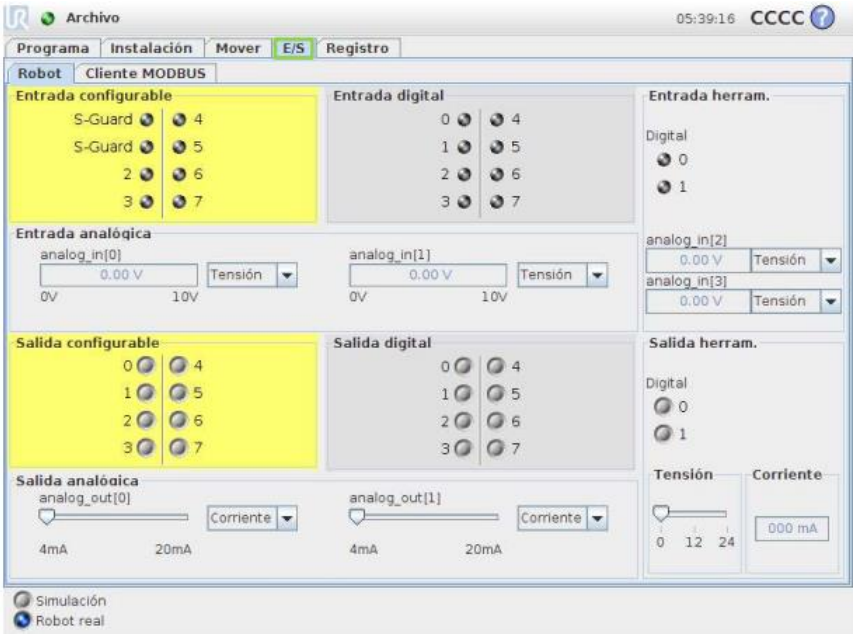


Figura 9.4: Menú de entradas y salidas (E/S).

Menú de movimiento (Mover): Menú que nos permite mover libremente el robot mediante una serie de flechas que mueven en una orientación o dirección refiriéndose a la herramienta o mediante el movimiento (en grados) de las diferentes articulaciones o juntas del robot por separado.

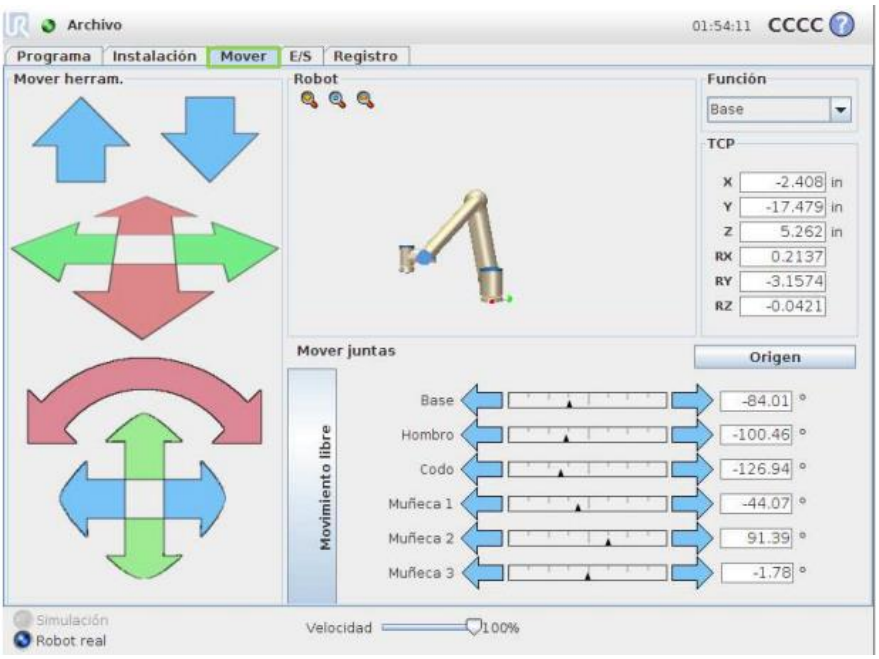


Figura 9.5: Menú de movimiento (Mover).

Menú de Registro: Menú que nos permite ver el estado del robot como la tensión que está recibiendo, la temperatura, el seguimiento de las cargas en cada articulación, la aparición de avisos y errores que pueden aparecer, etc.

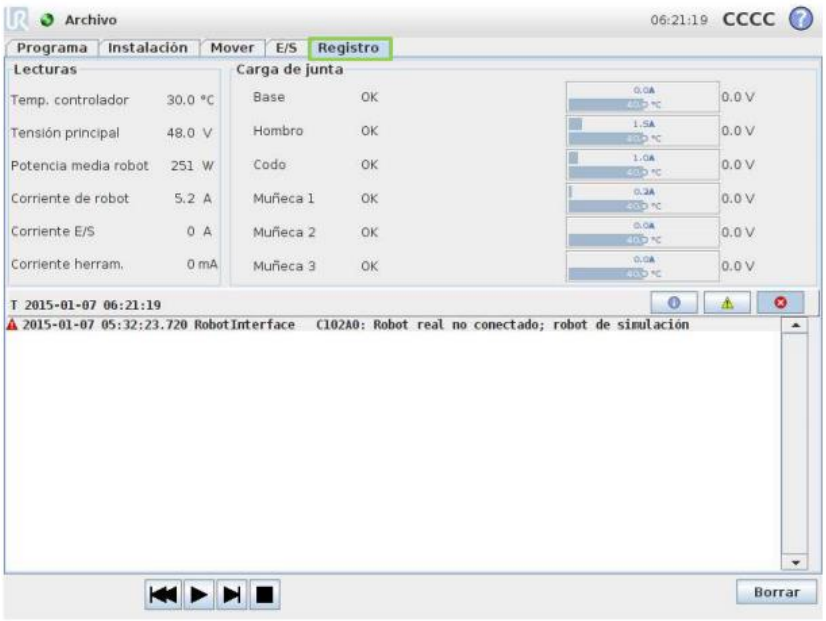


Figura 9.6: Menú de registro.

9.2 Menú Archivo

El menú de archivo es el que vamos a utilizar para guardar, cargar un programa o salir al menú de inicio.

Hay distintos modos para guardar, pero el más recomendado es el "Guardar como" ya que se puede elegir el directorio donde se guarda dicho archivo y así se evita algunos errores al guardar. Al guardar se crea distintos tipos de archivos:

- Un archivo tipo .urp: es un archivo manejado por PolyScope que contiene el código del programa a ejecutar.
- Un archivo tipo .instalación: es un archivo manejado por PolyScope que contiene las configuraciones de la instalación como las variables, la configuración de la PCH, etc.
- Un archivo tipo .variables: es un archivo donde se pueden consultar las variables que lleva el programa.
- Un archivo tipo .txt: es un archivo escrito en el bloc de notas que tiene el árbol del programa escrito.
- Un archivo. Script: es un archivo de visualización donde se puede el script del programa en lenguaje Python.

En cuanto al resto, solo hay que destacar que es recomendable al cargar un programa que todos estos archivos estén todos en un mismo directorio para evitar problemas o errores al cargar el programa.

9.3 Menú de Programación

A continuación, se explicará los diferentes elementos y pestañas más importantes del menú de programación.

Árbol de programación: Es un espacio situado a la izquierda de la pantalla donde nos muestra la serie de comandos o instrucciones que se ha puesto al robot, también nos permite seleccionar un comando en específico y editarlo o eliminarlo si le pinchamos a la ventana de "Comando" o "Estructura" respectivamente.

Además, si le damos al botón de "Run" situado en la parte inferior de la pantalla, nos permite ver que comando está ejecutando el robot para un mayor seguimiento del programa (Figura 9.6).

Ventana de Comando: Una ventana que nos permite editar los diferentes comandos que hayamos introducido como por ejemplo editar puntos de paso, editar el comando ajustar o el comando mover, etc. Para ello, se requiere primero seleccionar el comando a editar (Figura 9.6).

Figura 9.7: Árbol de programa (recuadro verde en la parte izquierda) y ventana de comando (parte derecha)

Ventana de Estructura: Ventana que nos permite añadir o eliminar comandos, cambiarlos de posición (hacia arriba o hacia abajo) o copiar y pegar comandos. Cuenta con varias pestañas en las que se clasifican los diferentes comandos (Figura 9.7).

Básico: Contiene los comandos más sencillos, por ejemplo, el comando mover, ajustar, esperar, añadir un punto de paso, etc.

Avanzado: Contiene los comandos referidos a las variables como por ejemplo el comando If o el comando asignación.

Asistente: Comando referido a herramientas, por ejemplo, el comando atornillado o el comando de paletizado.

URCAps: Contiene los comandos relacionados con el equipo de visión y esta ventana solo está presente si la cámara está instalada en el brazo robot.

Figura 9.8: Ventana de estructura del menú de programación.

9.4 Menú de Instalación.

A continuación, se va a explicar los diferentes elementos y pestañas más importantes de este menú.

Configuración de PCH: Nos permite cambiar el centro de gravedad, la carga y la inercia que tiene la herramienta del robot en el caso de que se haya acoplado algún elemento más como una cámara, una pinza, una herramienta para atornillar, etc. (Figura 9.2).

Montaje: Indica al robot donde está instalado la base del brazo robot ya que puede estar instalado en el suelo, en una pared o en el techo. También se puede ajustar para superficies ligeramente inclinadas y una diferente orientación de la base. Cuenta con una pequeña simulación para indicar como está posicionada actualmente la base del robot.(Figura 9.8)

Figura 9.9: Ventana de montaje del menú de instalación.

Variables: Pantalla en la que nos permite crear variables globales además de editarlas o eliminarlas. No permite crear o editar variables de tipo punto, línea o un plano ya que esto lo hace otro tipo de ventana la cual se hablará más adelante, pero permite ver si se han creado dichas variables.

Funciones: Ventana que nos permite crear, editar y eliminar variables de tipo punto, una línea o un plano. El proceso de creación es similar a la creación de un punto de paso donde podemos previsualizar la posición del robot, renombrar el punto, ajustarlo, etc.

9.5 Comandos utilizados en el programa

A continuación, se mostrarán los comandos utilizados en el proyecto y una explicación de cómo funcionan.

Comando "Mover": Este comando es una instrucción de que el robot se mueva a una posición indicada por medio de puntos de paso, en él se puede especificar la velocidad (en mm/s) y la aceleración (en mm/s^2) del robot. En este se puede incluir más de un punto de estos y se posicionará en estos de orden de arriba hacia abajo.

En este comando, los puntos de paso determinan la posición que debe el robot llegar mientras el tipo de “move” determina la trayectoria donde se pueden diferenciar 4 tipos:

MoveJ (por defecto): Es un movimiento no lineal donde el robot no tiene establecido una trayectoria en específico y toma la que es más rápido para llegar a este punto. Se suele llamar movimiento libre y se suele usar cuando no tiene importancia la trayectoria que tome el robot, puesto que no hay peligro de colisión.

Este movimiento siempre tiene una trayectoria circular puesto que cada articulación va lo más rápido posible a su posición especificada.

Figura 9.10: Robot UR3 haciendo un MoveJ

MoveL: Es un movimiento lineal donde el robot toma una dirección en línea recta, esto supone que el robot haga movimientos más complejos puesto que tiene que sincronizar las articulaciones de tal manera que la herramienta describa esta trayectoria. Este movimiento se suele utilizar cuando la trayectoria es importante o hay riesgo de colisión por ejemplo para aproximar o alejar la herramienta de la pieza entre otros (Figura 9.10).

Además, tiene una función que hace que el MoveL sea en función a un punto elegido. Si elegimos este punto que sea la Snapshot Position, los puntos de paso programados en este MoveL lo harán respecto de la posición guardada del Teaching de la pieza detectada. Esto permite que la pieza no tenga que estar en una orientación y posición fija, ya que estos puntos de paso, aunque parecen una posición fija, el origen de coordenadas no lo es, porque es el centro de la pieza detectada. Esta operación requiere que los puntos que vayamos a utilizar tomemos como referencia la posición que hemos puesto en el Teaching de la pieza ya que estos van a estar en función de esta posición. Si se ha desplazado la pieza puede haber un desajuste en estos puntos y se tendrían que ajustar a ojo o realizando un nuevo teach de la pieza.

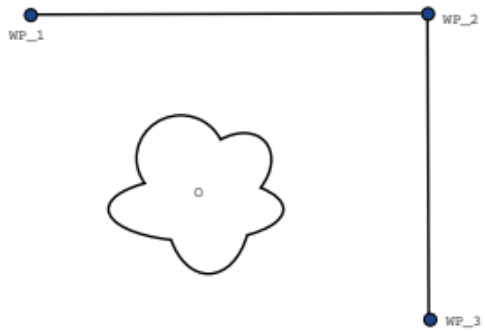


Figura 9.11: Gráfico sobre la trayectoria de un MoveL

MoveP: Es un movimiento parecido al MoveL, pero a diferencia de este es que mantiene en todo momento la velocidad constante. De esta manera, si hay un cambio brusco de dirección hará una transición en curva.

Este comando se suele utilizar cuando el robot está realizando una tarea que requiera una distribución regular de un material en específico como una soldadura o una distribución de pegamento sobre una zona. (Figura 9.11)

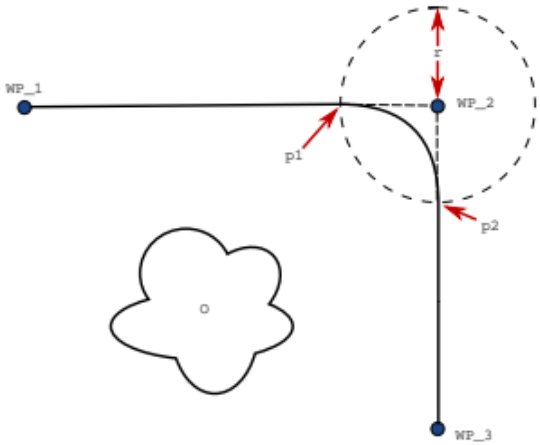


Figura 9.12: Gráfico de un movimiento MoveP

CircleMove: Es un MoveP donde se le añaden 2 puntos de paso para describir una trayectoria circular; el primero especifica un punto de ruta del arco a realizar y el segundo indica el final de este arco. (Figura 9.)

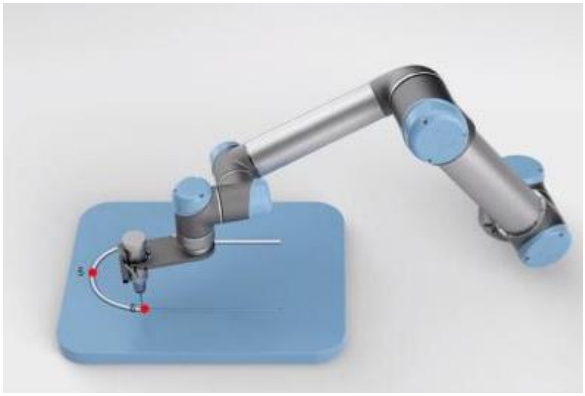


Figura 9.13: UR3 describiendo una maniobra CircleMove

Comando "Ajustar": comando básico que se encarga de cambiar el valor de una salida digital o analógica. En el caso de la salida digital se puede cambiar entre "High" (encendido) o "Low" (apagado) mientras las analógicas se pueden cambiar los valores en miliamperios.

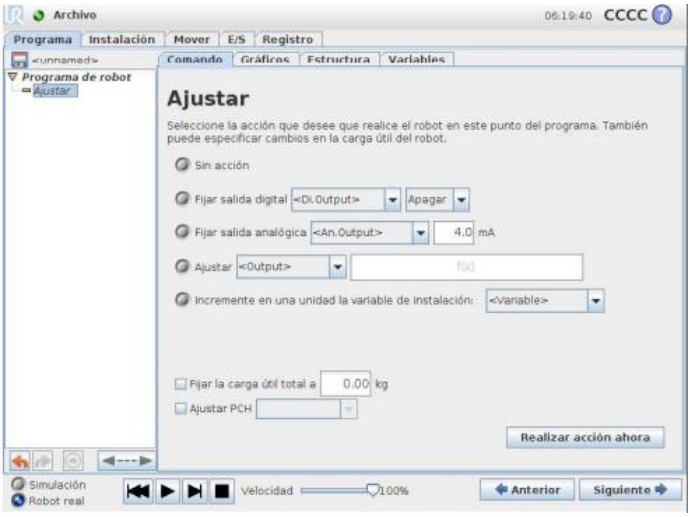


Figura 9.14: Pantalla del comando Ajustar.

Comando "Esperar": Como su propio nombre indica, hace que el robot espere un tiempo o hasta que se active o desactive una entrada digital, una entrada analógica llegue a cierto valor o por medio de una expresión que se haya introducido.

Figura 9.15: Pantalla de edición del comando "Esperar".

Comando "Asignar": Asigna un valor a una variable local o global ya sea mediante una expresión matemática o un valor fijo. En la pantalla se puede buscar una variable que se haya declarado antes mediante el desplegable y escribir la expresión en el recuadro al lado de este.

Figura 9.16: Pantalla del comando asignación o asignar.

Comando "If…else": Es un comando que comprueba una variable o cierto valor si se cumple con esta condición (es el valor correcto que se ha establecido), se hacen las instrucciones o comandos del interior de este. En el caso contrario, esta serie de comandos no se ejecutan y se pasan de largo. También al comando "If" se le puede añadir un "else" que es como otro If que solo se ejecuta en caso de que la comprobación sea falsa. Es decir, si se cumple con la condición y ya se ha realizado todos los comandos de dentro del If no se ejecutará ningún comando dentro del else y viceversa.

Comando "Camera Locate": Es un comando relacionado con la cámara donde se puede hacer un teach de un objeto y cuando se ejecuta el programa desde la posición de captura seleccionada lo va a buscar y puede ejecutar una serie de instrucciones referidas a este objeto.

Esta serie de instrucciones dentro del Camera Locate solo se pueden ejecutar si detecta el objeto enseñado, en este caso, se comporta muy parecido a un If.

Si ejecutamos programa podemos observar perfectamente cuando está intentando detectar la cámara puesto que se encienden las luces LED equipadas en esta y tarda unos segundos en reaccionar (Figura 9.16).

Se puede combinar con un MoveL para que los puntos de paso sean en función de la posición y orientación del objeto y así hacer un pick and place o cualquier otra función sin importar la posición inicial de este.

Comando "Aviso": Es un comando que permite poner un mensaje en la consola de programación hasta que un operario cierre el mensaje. Puede mostrar valores de variables en concreto o un texto escrito.

9.6 Zonas de Movimiento

Para la utilización del comando "Move" en las distintas partes se deben diferenciar dos zonas de trabajo diferentes:

Zona de Trabajo: Esta zona es la cual el robot va a interactuar con las piezas, por esto, hay peligro de colisión o de que un movimiento que no esté controlado pueda causar algún error en el programa. Debido a esto, en esta zona se va a utilizar el MoveL a 100 mm/s puesto que se necesita un movimiento controlado a baja velocidad y también porque es indispensable para los comandos de camera Locate por la función que tiene relacionado con este explicada anteriormente.

Zona Libre: En esta zona, el robot no tiene peligro de colisión con las piezas con las que interactúa y generalmente es una zona de transición de una zona de trabajo a otra.

Esta zona libre está delimitada por la posición de ataque o retirada de estas zonas de trabajo, es una posición en la que el robot está en posición para acercarse al objeto e interactuar con él y es el punto donde se hace una transición de un MoveJ a un MoveL y viceversa.

En esta zona se ha utilizado un MoveJ a la velocidad de 250 mm/s, ya que no hay peligro de colisión y suele ser una zona de transición, se utiliza esta velocidad para llegar lo antes posible al punto destinado evitando velocidades más altas que puedan llegar a ser peligrosas para el robot o el operario que esté cerca.

Estas dos zonas se distribuyen en el espacio de la manera que muestra el siguiente gráfico (Figura 9.17), donde podemos observar el punto que delimita estas dos zonas:

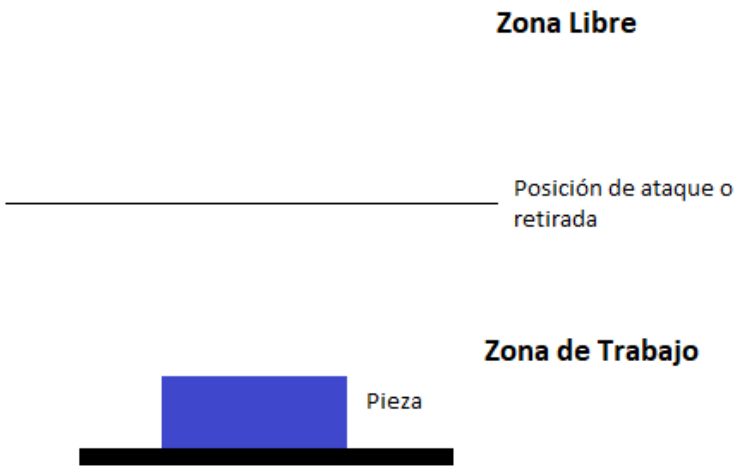


Figura 9.17: Gráfico sobre la distribución de las zonas de movimiento en el espacio.

Otra cosa que destacar, es la precisión que se ha utilizado en cada zona, en este caso, como es un robot programado mediante instrucciones y aprendizaje no es posible determinar la precisión correctamente y en todo caso es definida por el error de repetición que es ±0,1 mm o por algún error relacionado en la calibración o el teach en la cámara.

9.7 Conexión con dispositivos exteriores utilizados.

En cuanto a este apartado, se puede diferenciar dos tipos de dispositivos que se han conectado al robot, uno es el equipo de visión y el otro es la pinza.

Como se ha dicho anteriormente **la pinza** es un dispositivo que se acciona mediante aire comprimido y se controla mediante una electroválvula. Esta electroválvula se ha conectado a la salida digital numero 4 o también llamada DO[4], que está conectada de tal forma que el valor “high” o “encender” está configurado para que la pinza se abra y el valor “low” o “apagar” es para que se cierre.

En cuanto al equipo de visión, es un dispositivo que se instala en la muñeca del robot y una vez montada no hace falta configurar la conexión.

Otra cosa para destacar de este dispositivo es que, una vez instalado, nos aparece una ventana en “Estructura” extra que contiene comandos relacionado con este equipo y unas ventanas de instalación extras donde se nos permite comprobar la conexión de la cámara y configurar una Snapshot Position entre otras cosas.

9.8 Variables utilizadas

A continuación, se van a explicar la función de cada una de estas variables.

Variable "Snapcam_1": es una variable de tipo punto creada en la pestaña de función como se ha explicado anteriormente. Este punto es una "Snapshot Position" o posición de captura que sirve para identificar las piezas que llegan para ensamblar, en esta posición, van a llegar las piezas hembra o las defectuosas mediante un operario que las ponga allí y el robot las va a identificar y localizar desde este punto.

Variable "Snapcam_2": es otra variable de tipo punto que representa a la otra Snapshot Position, en esta, el robot lleva una pieza válida y busca la otra mitad de esta pieza para ensamblarla, en el caso de no encontrarla, el robot se dirige a la primera posición de captura y deja la pieza para dar tiempo a que pongan una nueva.

Variable "pieza": es una variable que hace marca una prioridad de ensamblar unas piezas sobre otras, primero prioriza las cuadradas sobre las demás y luego prioriza las circulares frente a las defectuosas. Para ello toma diferentes valores (0, 1 o 2) para diferenciar los distintos tipos de piezas y se sirve de los comandos If para implementar esta prioridad.

Variable "N_Piezas": variable que se utiliza para guardar el conteo de piezas totales que ha manipulado el cobot.

Variable "N_cubos": variable que guarda el número de piezas cuadradas que se han ensamblado.

Variable "N_cilind": variable que guarda el número de piezas cilíndricas que se han ensamblado

Variable "N_defectuosas": variable que guarda el número de piezas defectuosas que se han tirado al contenedor.

Variable Por_cubos: variable que devuelve el porcentaje de piezas cuadradas respecto a las totales en tanto por 100.

Variable Por_cilind: variable que devuelve el porcentaje de piezas cilíndricas respecto a las totales en tanto por 100.

Variable Por_defect: variable que devuelve el porcentaje de piezas defectuosas respecto a las totales en tanto por 100.

10. Estructura del Programa.

En este apartado se va a explicar con detalle en que consiste el programa, así como sus distintas posiciones y otros aspectos.

Esta explicación la vamos a dividir en diferentes apartados para una mejor explicación de estos.

10.1 Módulo Inicial.

Este módulo incluye las 3 primeras instrucciones para que el robot siempre empiece en las mismas condiciones.

En este, se mueve al "Snap_1" que coincide con la posición de captura "Snapcam_1" mediante un MoveJ. También se ajusta la pinza para que esté abierta y se pone la variable pieza en 0 para que, en caso de no haber una pieza del tipo cuadrada, se pase a la localización de las circulares.

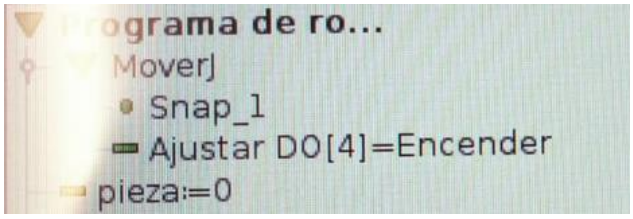


Figura 10.1: Árbol del programa correspondiente al módulo inicial.

10.2 Módulo de las piezas cuadradas.

Este módulo, comienza por el primer "camera Locate", en este se tiene el teach de las piezas cuadradas del tipo "hembra". Una vez localizado esta pieza, se hace un MoveJ al punto "Posición_cubo_1" esta posición es la de ataque que en este caso se pone principalmente la pinza perpendicular a la mesa y en una posición central de esta zona. Después, se pasa a un MoveL referido a la Snapcam_1 que se dirige al punto "Close_pinz_cu" a 100 mm/s para llegar a la pieza, seguido a esto cierra la pinza y espera 1 segundo.

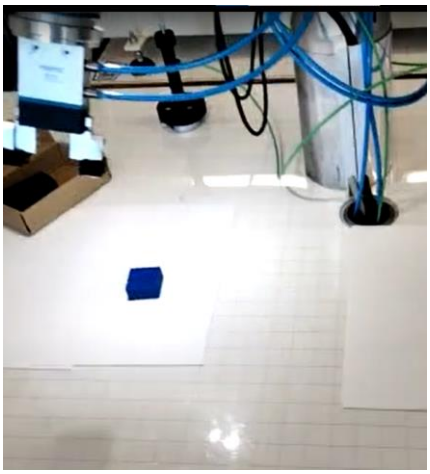


Figura 10.2: Robot UR3 en la posición Snapcam_1 buscando la pieza cuadrada.

Figura 10.3: Robot UR3 cogiendo la pieza cuadrada.

Una vez que la pinza haya agarrado a la pieza, se dirige mediante un MoveL a un punto de retirada y después mediante un MoveJ a la posición de captura “Snapcam_2”que, en este caso, es el mismo punto que el “Snap_2”.

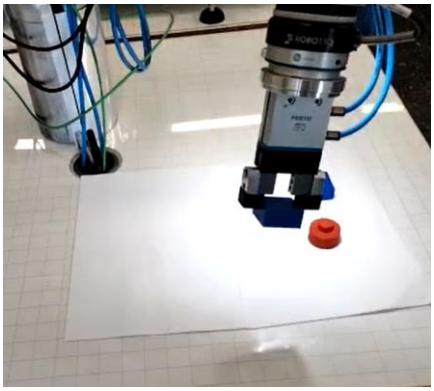


Figura 10.4: Camera locate en la Snapcam_2 de las piezas cuadradas.

Seguido a esto, se realiza un segundo “camera Locate” para identificar la pieza “macho” que le corresponde a la pieza agarrada donde una vez localizada, se dirige mediante un MoveJ a la posición de ataque “Atac_cubo” y después a la posición de ensamblaje mediante un MoveL referido a la posición de captura “Snapcam_2”(punto “Ensambl_cubo”); donde se abre la pinza, se espera 1 segundo y después se dirige a la posición de retirada que coincide con la posición de ataque “Atac_cubo”.

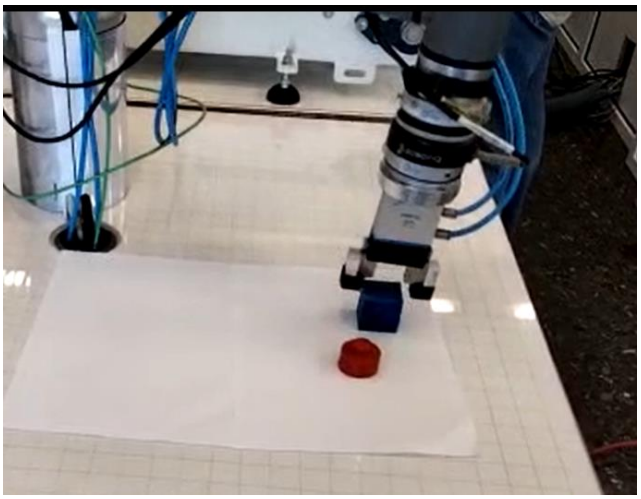


Figura 10.5: Robot UR3 ensamblando la pieza cuadrada.

Si se realiza este ensamblaje, se pone la variable pieza en 1 para en el caso de haber más de estas piezas se priorice el ensamblaje de estas antes que los otros tipos de piezas en este caso, se volvería al inicio del programa. Antes de volver a la siguiente operación se suma en uno en las variables N_piezas y N_cubos.

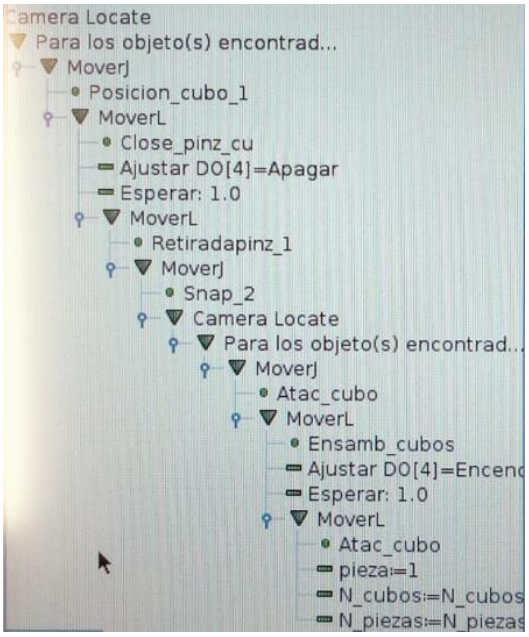


Figura 10.6: Árbol de programa correspondiente al módulo de las piezas cuadradas.

10.3 Módulo de las Piezas cilíndricas.

En el caso de que no se haya ensamblado la pieza cuadrada la variable pieza sigue en el estado inicial (pieza=0), en tal caso, se acepta la comprobación del If y se ejecutan las instrucciones referidas a esta pieza.

Antes de hacer el "Camera Locate", el programa comprueba que el robot está en la posición inicial (en la posición de la Snapcam_1 y con la pinza abierta) ya que el robot puede haber cogido la pieza cuadrada pero no haber encontrado su otra mitad para ensamblar, de este modo se asegura que el robot este en esta posición y suelte la pieza que lleve en la pinza.

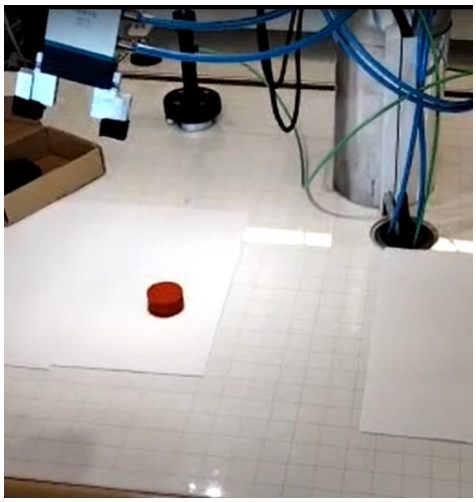


Figura 10.7: Robot UR3 en el "camera locate" de las piezas cilíndricas.

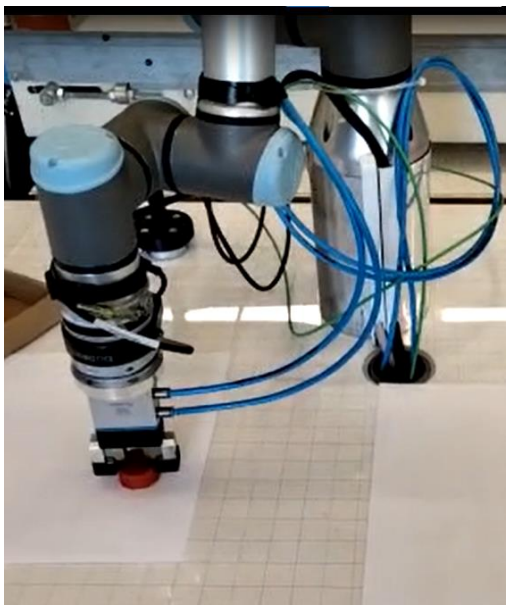


Figura 10.8: Robot UR3 cogiendo la pieza cilíndrica

Además, cambia el valor de la variable (pieza=2) para que en el caso de que no haya ninguna pieza circular pase directamente a las piezas defectuosas.

Tras estas operaciones este módulo no se diferencia mucho del anterior. Primero hace un "Camera Locate" para identificar la pieza circular, después se dirige a la posición de ataque mediante un MoveJ (es el "Ataq_cilind") y seguido a esto, se dirige a la posición de la pieza mediante un MoveL referido a la Snapcam_1 ("Coger_cilind"), cierra la pinza, espera 1 segundo y se retira al Ataq_cilind.

Seguido a esto, el robot se dirige al punto Snap_2 que coincide a la posición de captura "Snapcam_2"hace otro "Camera Locate" para identificar la otra mitad de la pieza. Localizada la otra mitad, el robot se dirige a la posición de ataque "pos_cilind" mediante un MoveJ y luego a la posición de ensamblaje llamando dicho punto "Ensambl_cilind_1" mediante el MoveL referido a esta posición de captura.

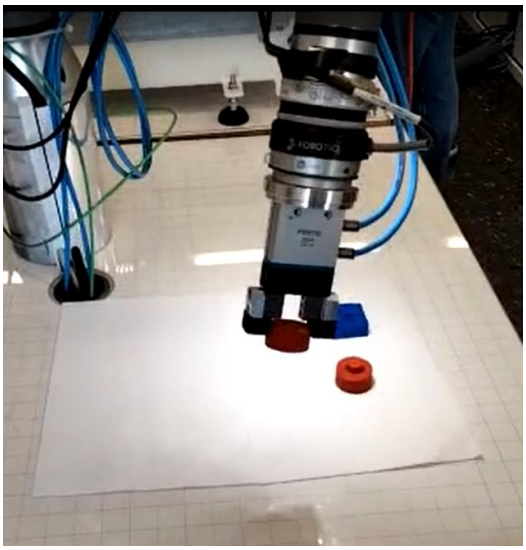


Figura 10.9: Robot UR3 en la posición Snapcam_2 buscando la pieza cilíndrica "Macho".

Una vez que esté en esta posición el robot se realiza el ensamblaje (se abre la pinza y espera 1 segundo para asegurarse de que la pieza se haya depositado), se retira al punto "pos_cilind" mediante un MoveL y se cambia la variable pieza=0 para priorizar las piezas cilíndricas y cuadradas en el caso de que hayan llegado más de estas. Por último, se suma en uno en las variables N_cilind y N_piezas.

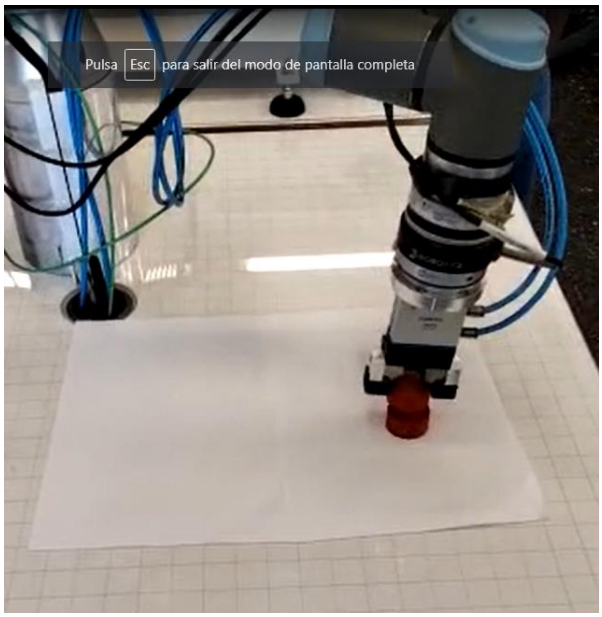


Figura 10.10: Robot UR3 ensamblando la pieza cilíndrica.

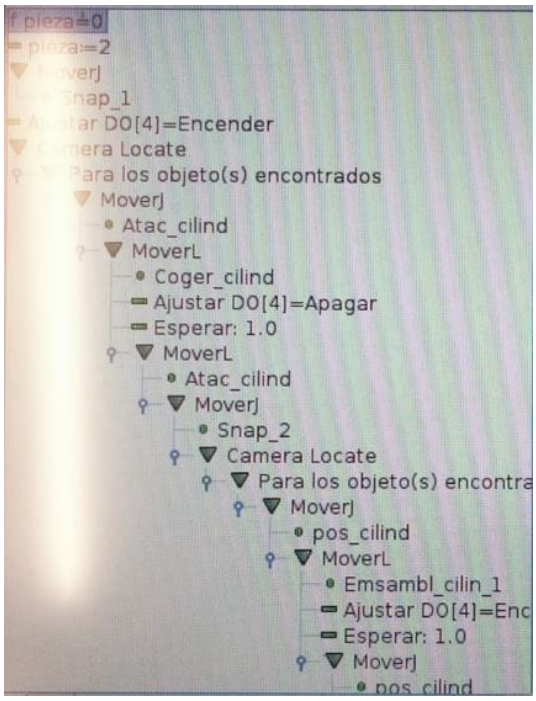


Figura 10.11: Árbol de programa correspondiente al módulo de las piezas cilíndricas.

10.4 Módulo de las piezas defectuosas.

En el caso de que no se ha ensamblado ninguno de estos dos tipos de piezas, la variable pieza seguiría teniendo el valor de dos y la condición del If que precede a este módulo es verdadera.

En este caso el robot primero hace que el robot esté en las condiciones iniciales para evitar errores de que pueda tener una pieza en la pinza o no esté en la posición correcta. También se cambia la variable “pieza” = 4 para en el caso de que no haya ninguna pieza se puede ejecutar el módulo final.

Seguido a esto, se realiza un “camera Locate” para identificar y localizar la pieza defectuosa, seguido a esto se inicia la secuencia de coger la pieza ya explicada en los casos anteriores.

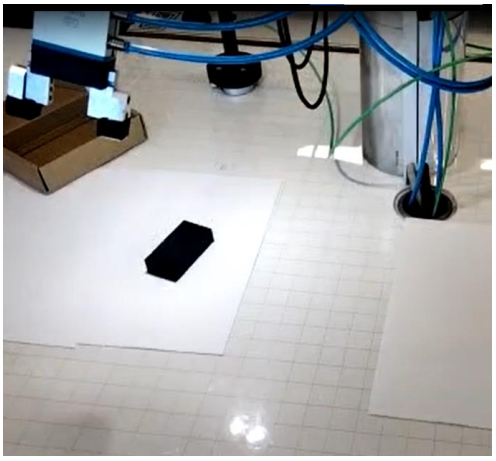


Figura 10.12: Camera locate de las piezas defectuosas.

Se dirige a la posición de ataque (“Pos_Defect”) después, a la localización de la pieza (“Coger_defect_1”), cierra la pinza y espera un segundo. Cabe destacar en este tipo de pieza que a diferencia de los otros dos tipos de piezas solo se puede agarrar de un solo lado, ya que la pieza a coger es un rectángulo y en este caso la pinza la va a coger desde el lado más corto.

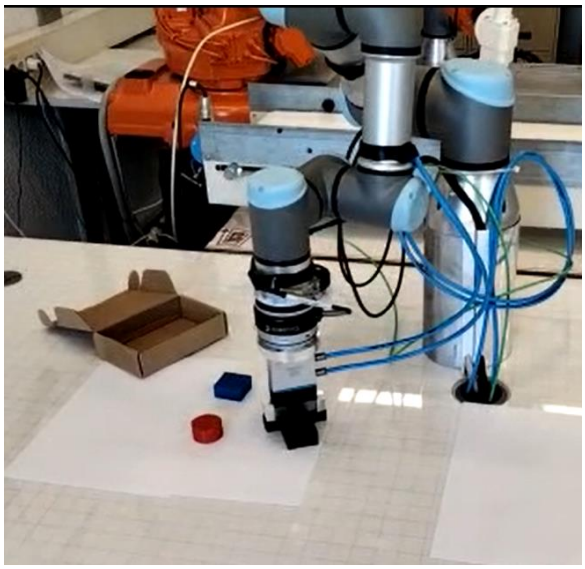


Figura 10.13: Robot UR3 cogiendo la pieza defectuosa.

Una vez hecha la secuencia de coger la pieza, la pinza se dirigirá al punto (Contenedor) mediante un MoveJ; este punto es fijo ya que simula la salida a un contenedor de piezas defectuosas o una cinta para reciclar el material, donde la pinza una vez en esta posición abre la pinza para soltar la pieza. Tras soltar la pieza en el contenedor se cambia la variable pieza en 2 para comprobar si hay más piezas antes de ejecutar el módulo final. También se suma en uno las variables N_defectuosas y N_piezas para realizar el conteo correspondiente.

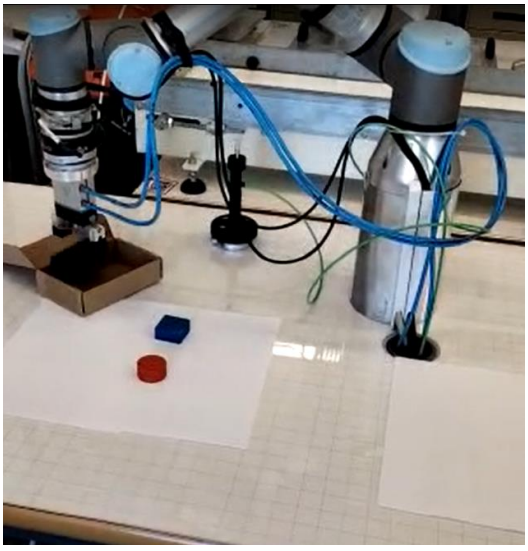


Figura 10.14: Robot UR3 tirando al contenedor la pieza defectuosa.

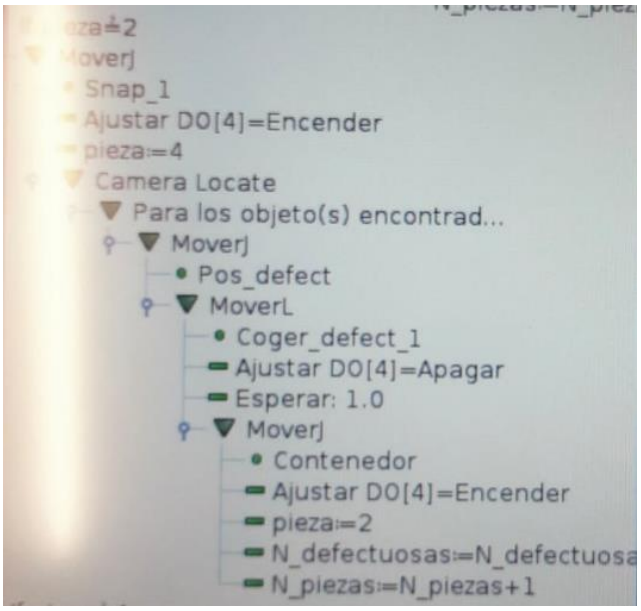


Figura 10.15: Árbol de programa correspondiente al módulo de piezas defectuosas.

10.5 Módulo final

En el caso de que no haya ninguna pieza en la posición de captura 1, se hace un aviso al operario de que no hay más piezas, se calculan los diferentes porcentajes de cada pieza y se muestra mediante avisos el numero de cada pieza que se ha ensamblado o tirado al contenedor además del porcentaje que corresponde a cada una de ellas en tanto por ciento.

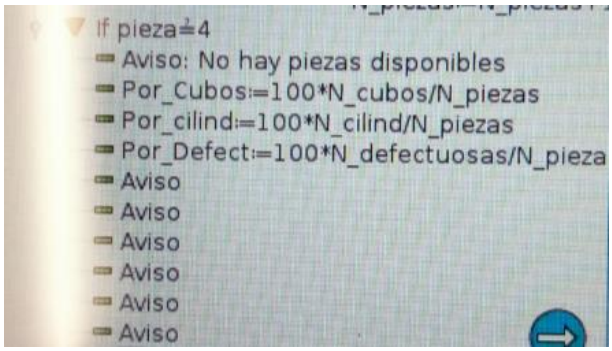


Figura 10.16: Árbol del programa correspondiente al módulo final.

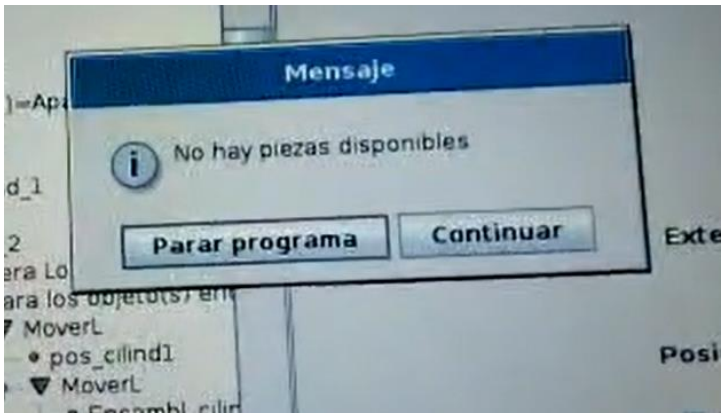
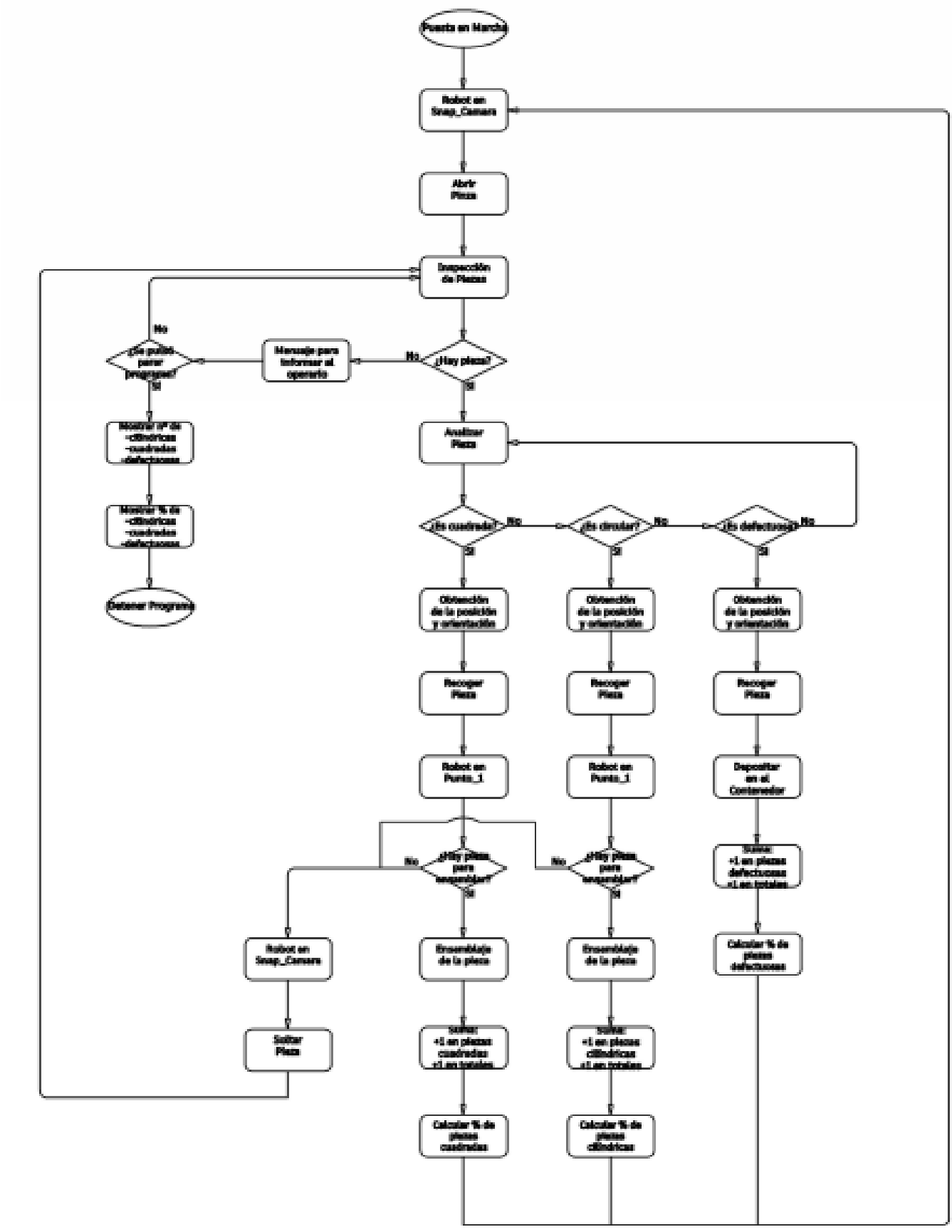


Figura 10.17: Uno de los mensajes finales que aparecen en la consola de programación.

11. Flujograma.

En este punto se muestra un flujograma para que se pueda un seguimiento mejor del proceso que hace el robot en este proyecto; con el fin de poder verlo mejor, va a estar en un archivo PDF a parte (“Flujograma”) donde se puede visualizar con mejor calidad de las letras.



12. Problemas surgidos en el proyecto

En este apartado se van a explicar los problemas que han surgido durante el proyecto y cómo se han podido solucionar total o parcialmente.

- El laboratorio tenía poca disponibilidad debido a que había más alumnos que utilizaban el mismo robot que yo esto y que solo hay un ejemplar de este, había algunas veces que solo tenía una o dos horas para hacer el programa. Esto se ha solucionado con una mejor organización con estos y haciendo una especie de lista de tareas.

- La pinza por sí sola no podía coger las piezas ya que solo tenía una apertura de 30 mm y mis piezas tenían un tamaño de 38 mm esto fue un inconveniente al primer día ya que después me di cuenta de que había un accesorio para esta pinza que ampliaba esa apertura en 50 mm .

- Después de instalar los accesorios al agarrar la pieza muchas veces resbalaban o no se cogían correctamente, esto lo consulté con el técnico del laboratorio y propuso en implementarle una especie de goma con una pistola de calor para mejorar el agarre de la pinza. Al final resultó una mejora bastante notable en este aspecto y se solucionó completamente este problema.

- Otro problema por destacar es que el programa PolyScope necesitaba el sistema operativo Linux y para poder instalarlo en mi ordenador necesité de una VirtualBox para emular el siguiente sistema operativo; también en mi ordenador no podía utilizar los comandos relacionados con la cámara y también era otra versión lo que supone que solo lo pude utilizar para familiarizarme con el programa.

- También hubo problemas relacionado con los puntos de paso ya que están en función de la posición inicial del teach si hay errores hay que ajustarlos a ojo o hacer de nuevo el teach y volver a hacer el punto con este problema opte de ajustarlos estos puntos a ojo mediante ensayo y error ya que hacer de nuevo el teach y ajustar de nuevo el punto de paso es muy susceptible de cometer errores sobre todo en el ensamblaje y repetirlo de nuevo.

- A veces la cámara reconocía las piezas cuadradas donde no las había o confundían las piezas defectuosas con las cuadradas ya que la mesa donde estaba instalado tenía un patrón de cuadricula y reflejaba la luz y esto dificultaba la visión de la cámara. Esto se solucionó colocando unas cartulinas blancas para evitar el reflejo y subiendo el parámetro de coincidencia (Detection Threshold) para que no confundiese las piezas.

13. Conclusión.

Se puede concluir que el objetivo principal de el ensamblaje de piezas mediante un equipo de visión se ha cumplido.

Durante el proyecto se ha implementado algunas funciones para acercarlo todo lo posible a un caso real como diferentes piezas para que en el caso de una línea común a 2 piezas sepa clasificar y diferenciar 2 tipos de piezas además de un caso de piezas defectuosas. Otra cosa para destacar es la priorización de unas piezas sobre otras ya que esto permite compensar factor limitante de la producción ya sea porque se necesitan más de ese modelo de pieza o que tarde más en producirse y esto permite algo de personalización porque en el momento de que el otro tipo de

pieza (circular) pase a ser el factor limitante, se puede cambiar para que sea esta la prioritaria. Esto lo podemos controlar mediante el conteo de piezas y el porcentaje de cada pieza y así poder controlar bien la producción.

Por otra parte, la programación del robot UR3 en sí no ha supuesto una gran dificultad debido a los conocimientos adquiridos en la asignatura Sistemas Robotizados y en los manuales y guías de usuario del propio robot UR3 que nos proporciona Universal Robots. También debido al sistema de visión, que solo había que instalarlo en el robot para poder utilizarlo y la única configuración que era necesaria hacer era la posición de captura, la calibración de esta o el teach de los objetos.

Otra cosa para destacar es una mejor profundización en el programa de dibujo de AutoCAD que, aunque en el primer año me dieron clases básicas de este programa, se había enseñado muy poco sobre el modelado 3D, cosa que me ha obligado a buscar información y profundizar algo más en este programa hasta que me ha permitido hacer figuras algo complejas mediante este programa.

En esencia, este proyecto ha sido un trabajo multidisciplinar que me ha otorgado una iniciación al tema de la visión artificial, la impresión 3D y los robots colaborativos cuyos conocimientos me servirán para tener una base de estos y poder profundizar más. Este proyecto me ha permitido desenvolverme en un entorno parecido al real, solventar los problemas que han ido apareciendo e implementar mejoras a este para que fuese más flexible y tenga una mejor capacidad de adaptación con los mínimos cambios.

14. Bibliografía.

Manuales de Instrucciones:

- "Manual de usuario del robot UR3" de Universal Robots.
- "Manual de PolyScope" de Universal Robots.
- "The URScript Programming Languaje" de Universal Robots.
- "Manual de Instrucciones del sistema de visión Wrist Camera para Universal Robots" de Robotiq.

Apuntes de la asignatura Sistemas Robotizados impartida en el departamento de Ingeniería de Sistemas y Automática de la Universidad Politécnica.

Páginas WEB:

https://www.elfinanciero.com.mx/tech/que-son-los-robots-colaborativos-y-por-que-son-ideales-para-las-pymes/

Federación Mundial de la Robótica: https://ifr.org/industrial-robots

http://platea.pntic.mec.es/vgonzale/cyr_0204/cyr_01/robotica/sistema/morfologia.htm#articulada

https://larraioz.com/iai/productos/robots-scara → Características robots scara

https://larraioz.com/iai/productos/robots-cartesianos → Características robots cartesianos

https://new.abb.com/products/robotics/es/robots-industriales/irb-140/datos → Robot IRB 140 de ABB

https://www.ni.com/third_party/jai/pdf/cv_m77db.pdf →Características cámara JAI CV-M77.

15. Anexos

15.1 Ficha Técnica del Robot UR3.

UNIVERSAL ROBOTS

UR3 Especificaciones técnicas

No. Artículo 110103

Brazo robótico de 6 ejes con un radio de acción de 500 mm

Peso:	11 kg
Carga útil:	3 kg
Alcance:	500 mm
Rango de las articulaciones:	+/- 360° Rotación infinita de la última articulación
Velocidad:	Articulaciones de muñeca: 360 grados/segundo Otras articulaciones: 180 grados/segundo Herramienta: Típico 1 m/s
Repetibilidad:	+/- 0,1 mm
Huella:	Ø 128 mm
Grados de libertad:	6 articulaciones giratorias
Tamaño de la caja de control (AnchoxAltoxLargo):	475 mm x 423 mm x 268 mm
Puertos de E/S	Caja de control / Conexión de herramienta Entrada digital: 16 / 2 Salida digital: 16 / 2 Entrada analógica: 2 / 2 Salida analógica: 2 / -
Fuente de alimentación E/S:	24 V 2A en caja de control y 12 V/24 V 600 mA en herramienta
Comunicación:	TCP/IP 100 Mbit: IEEE 802.3u, 100BASE-TX Ethernet socket y Modbus/TCP
Programación:	Interfaz gráfica de usuario PolyScope con pantalla táctil de 12 pulgadas con soporte
Ruido:	Comparativamente silencioso
Clasificación IP:	IP64
Consumo de energía:	100 vatios aprox. utilizando un programa típico
Operación de colaboración:	15 funciones avanzadas de seguridad ajustables
Materiales:	Aluminio, polipropileno
Temperatura:	El robot puede trabajar en un rango de temperaturas de 0-50 °C*
Fuente de energía:	100-240 VAC, 50-60 Hz
Cableado:	Cable entre el robot y la caja de control (6 m) Cable entre la pantalla táctil y la caja de control (4,5 m)

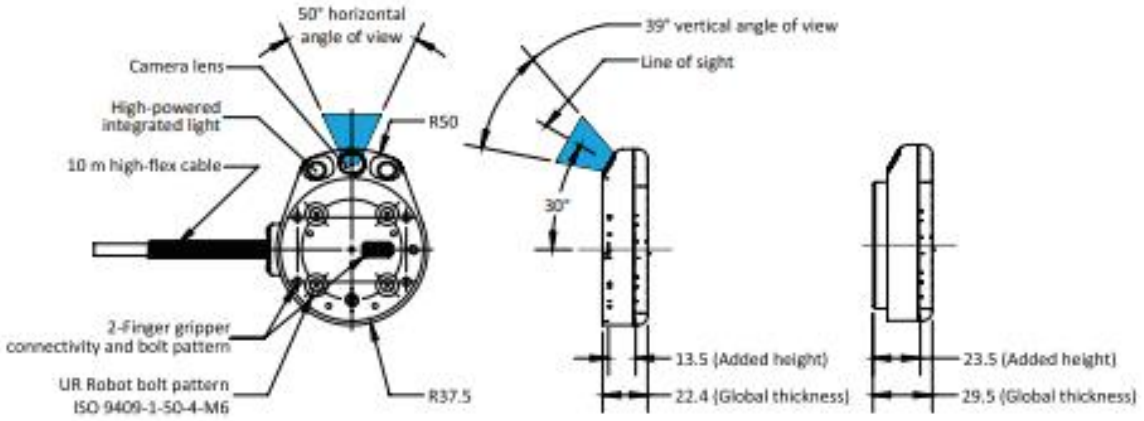
*) A alta velocidad continua de las articulaciones, la temperatura ambiente se reduce.

Universal Robots A/S
Energivej 25
DK-5260 Odense S
Dinamarca
+45 89 93 89 89

www.universal-robots.com
sales@universal-robots.com

15.2 Ficha Técnica de la “Wrist Camera”.

ROBOTIQ CAMERA TECHNICAL DATA



CAMERA SPECIFICATIONS

Maximal resolution	5 Mpx @ 2fps	2560 X 1920
Maximal framerate	30 fps @ 0.3 Mpx	640 X 480
Active array size	2592 X 1944	
Focus range	70 mm to infinity	
Autofocus	Yes	
Integrated lighting	6 LED diffuse white light	

MECHANICAL SPECIFICATIONS

Added height	13.5 mm (Without tool plate) 23.5 mm (With tool plate)	Distance between the robot flange and the tool base
Weight	160 g	Without the cable
Maximum load	10 kg	
Operating temperature	0°C to 50°C	
Environmental protection	Water-tight	
Operating conditions	Environment free from powerful electromagnetic interference and corrosive or explosive liquids or gases Non-condensing humidity level Lense must be free from dust, soot and water	

ELECTRICAL SPECIFICATIONS

Nominal supply voltage	24 V DC ±20%	
Quiescent power consumption	1 W	
Maximum power consumption	22 W	When light is on
Communication interface	USB 2.0	Software package available for Universal Robots

VISION SPECIFICATIONS*

	UR3	UR5	UR10
Minimum field of view (cm)	10 x 7.5	10 x 7.5	10 x 7.5
Maximum field of view (cm)	36 x 27	64 x 48	100 x 75
Minimum part size (% of field of view)	10%	10%	10%
Maximum part size (% of field of view)	60%	60%	60%
Maximum part height: smallest dimension	1:1	1:1	1:1

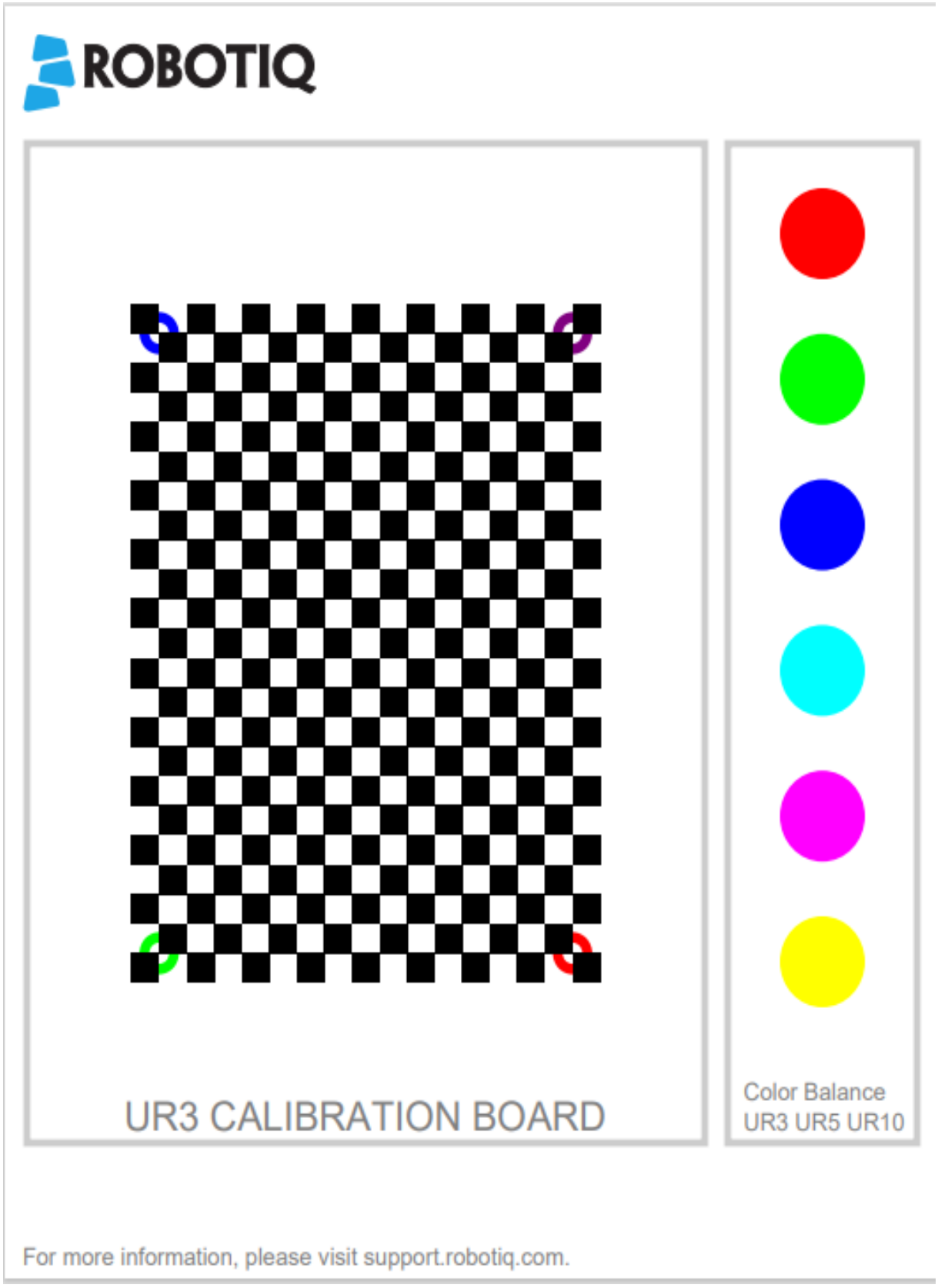
* Vision spec will change according to the teached snapshot position, see instruction manual for details

Compatibility: Universal Robots UR3, UR5 & UR10 with controller CB3.1 or higher only

robotiq.com
info@robotiq.com
T: 1.418.380.2788

Updated on September 9, 2016
Specifications subject to change without notice

15.3 Plantilla de Calibración de la Cámara



15.4 Estructura del Programa en Polyscope.

Programa

Programa de robot

MoverJ

Snap_1

Ajustar DO[4]=Encender

pieza=0

Camera Locate

Para los objeto(s) encontrados

MoverL

Posición_cubo_1

MoverL

Close_pinz_cu

Ajustar DO[4]=Apagar

Esperar: 1.0

MoverJ

Retiradapinz_1

MoverJ

Snap_2

Camera Locate

Para los objeto(s) encontrados

MoverL

Atac_cubo

MoverL

Ensamb_cubos

Ajustar DO[4]=Encender

Esperar: 1.0

MoverJ

Atac_cubo_1

Pieza=1

N_cubos=Ncubos+1

N_piezas=N_piezas+1

If pieza=0

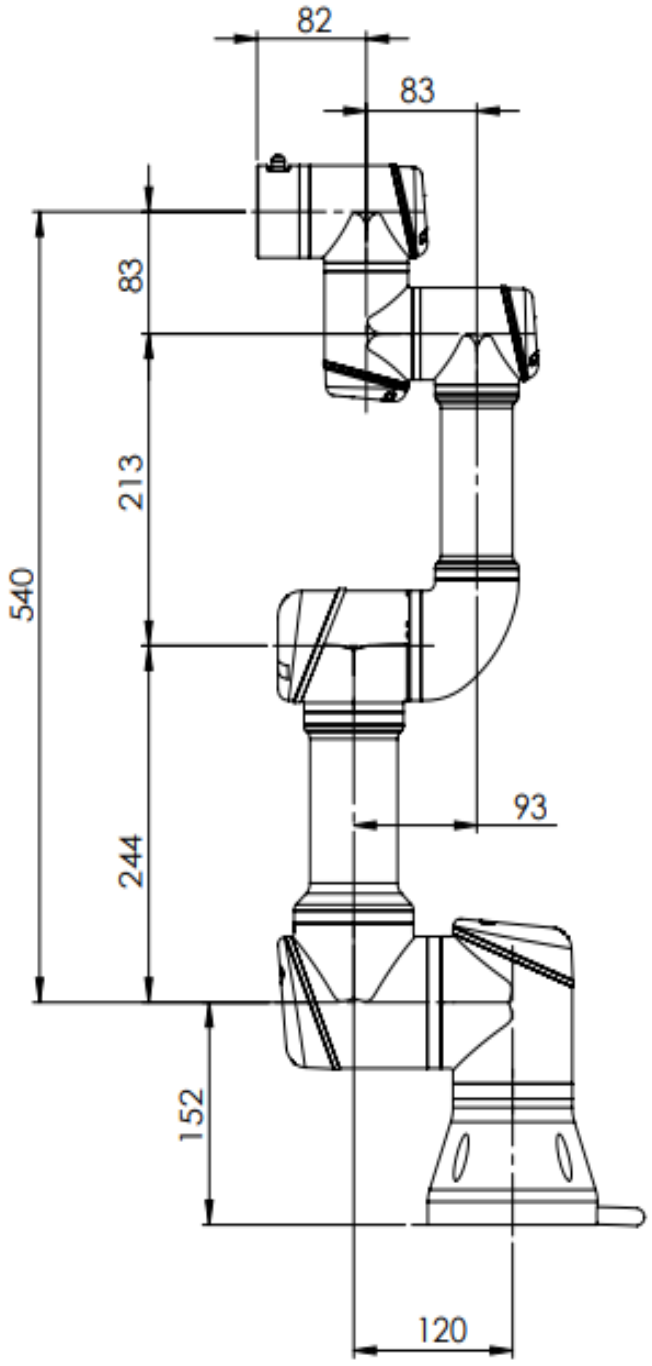
```
    pieza=2
    MoverJ
      Snap_1
    Ajustar DO[4]=Encender
    Camera Locate
      Para los objeto(s) encontrados
        MoverL
          Atac_cilind
          MoverL
            Coger_cilind
            Ajustar DO[4]=Apagar
            Esperar: 1.0
            MoverJ
              Atac_cilind_1
              MoverJ
                Snap_2
                Camera Locate
                  Para los objeto(s) encontrados
                    MoverL
                      Pos_cilind
                      MoverL
                        Ensambl_cilind
                        Ajustar DO[4]=Encender
                        Esperar: 1.0
                        MoverJ
                          pos_cilind_1
                          pieza=0
                          N_cilind=N_cilind+1
                          N_piezas=N_piezas+1
If pieza=2
    MoverJ
      Snap_1
      Ajustar DO[4]=Encender
```

```
        pieza=4
        Camera Locate
          Para los objeto(s) encontrados
              MoverJ
                Pos_defect
                MoverL
                  Coger_defect_1
                  Ajustar DO[4]=Apagar
                  Esperar: 1.0
                  MoverJ
                      Pos_defect
                      Contenedor
                      Ajustar DO[4]=Encender
                      pieza=2
                      N_defectuosas=N_defectuosas+1
                      N_piezas=N_piezas+1
If pieza=4
    Aviso: No hay piezas disponibles
    Por_Cubos=100*N_cubos/N_piezas
    Por_cilind=100*N_cilind/N_piezas
    Por_Defect=100*N_defectuosas/N_piezas
    Aviso:   N de piezas cuadradas
    Aviso
    Aviso:   N de piezas cilíndricas
    Aviso
    Aviso:   N de piezas defectuosas
    Aviso
    Aviso: % piezas cuadradas
    Aviso
    Aviso: % piezas cilíndricas
    Aviso
    Aviso: % piezas defectuosas
    Aviso
```

Documento II

Planos

Plano 1: Plano del Brazo del Robot UR3.



All dimension is in mm
For public use

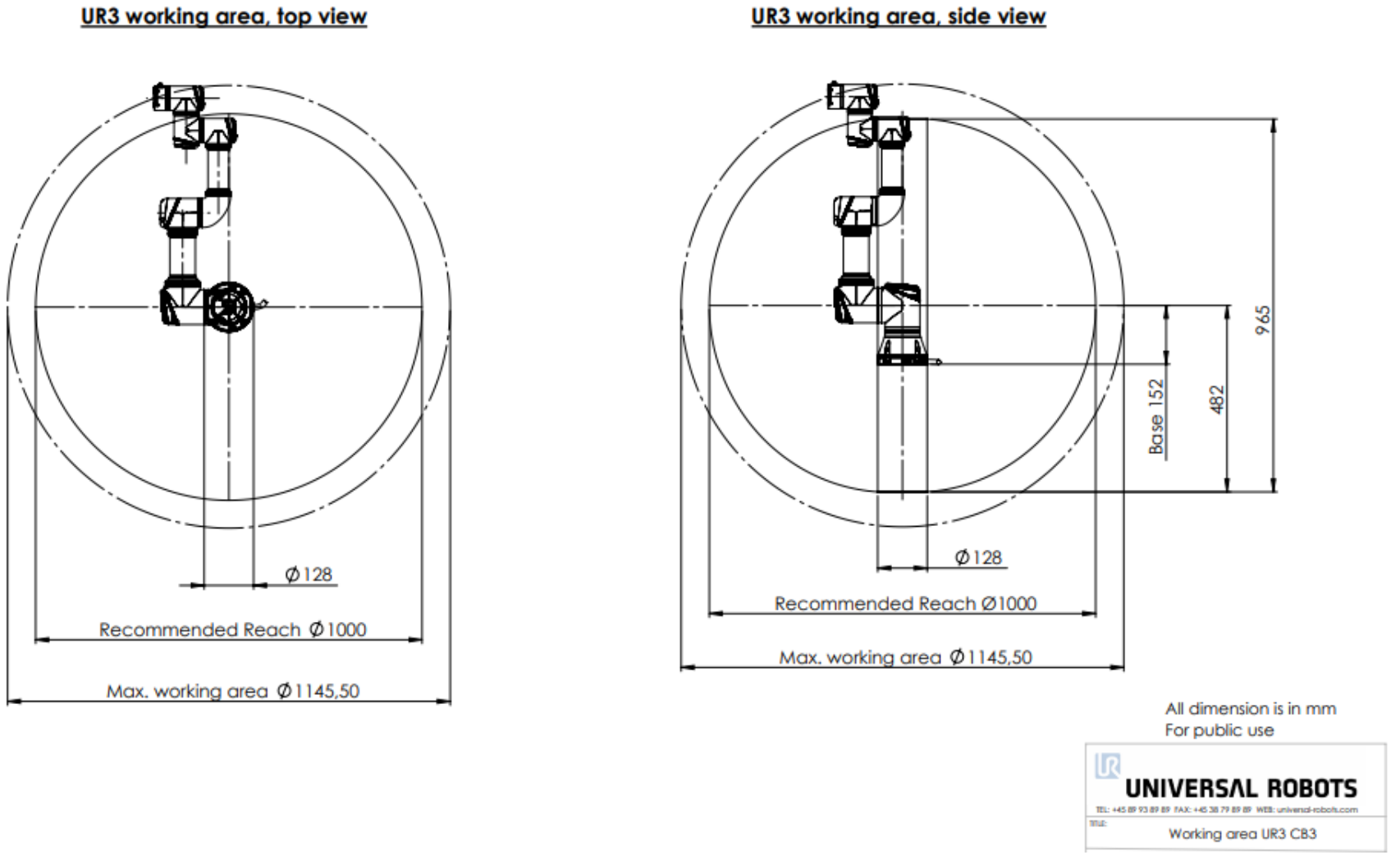
UNIVERSAL ROBOTS

TEL: +45 89 93 89 89 FAX: +45 38 79 89 89 WEB: universal-robots.com

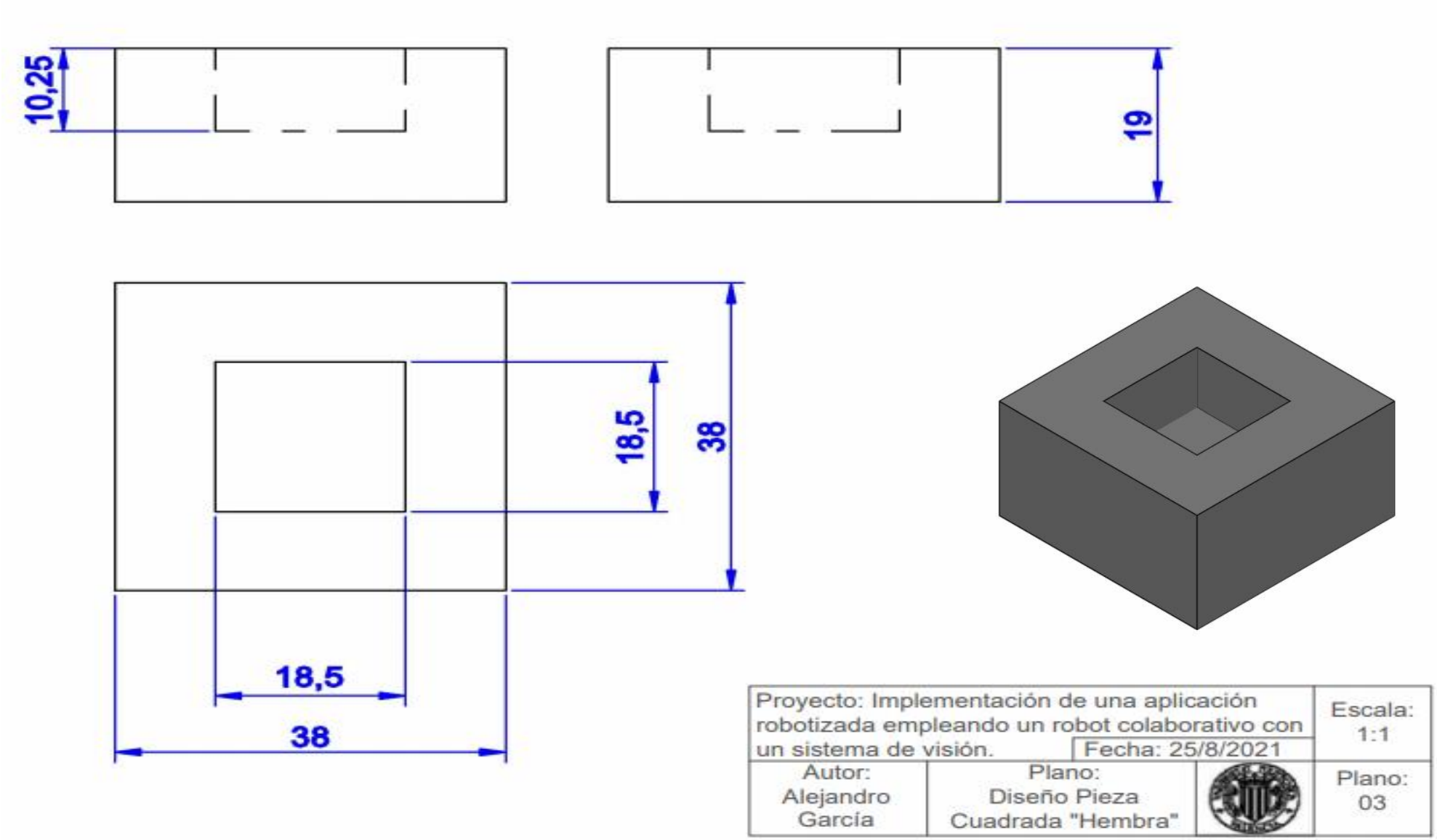
TITLE:

Working area UR3 CB3

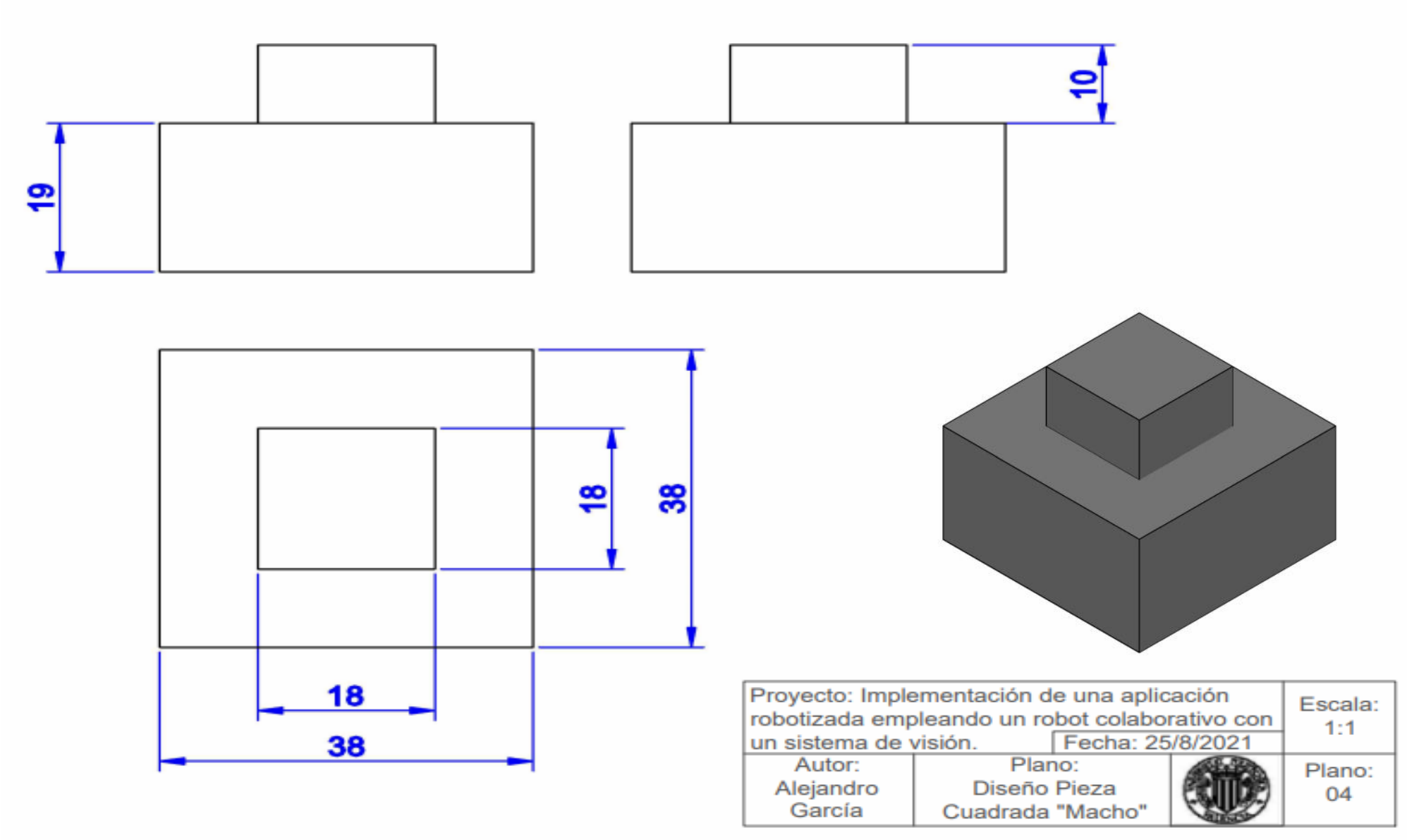
Plano 2: Área de Trabajo del UR3.



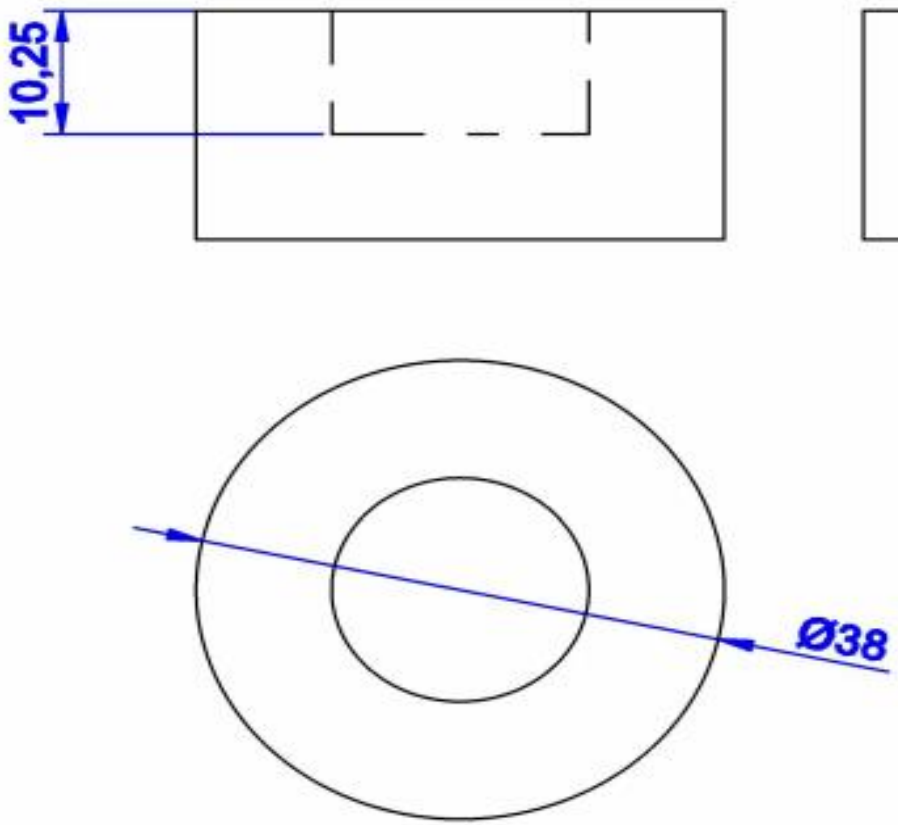
Plano 3: Diseño de la Pieza Cuadrada “Hembra”.

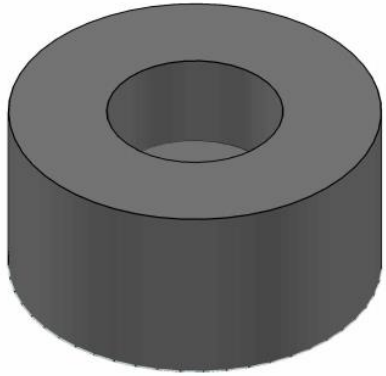


Plano 4: Diseño de la Pieza Cuadrada “Macho”.



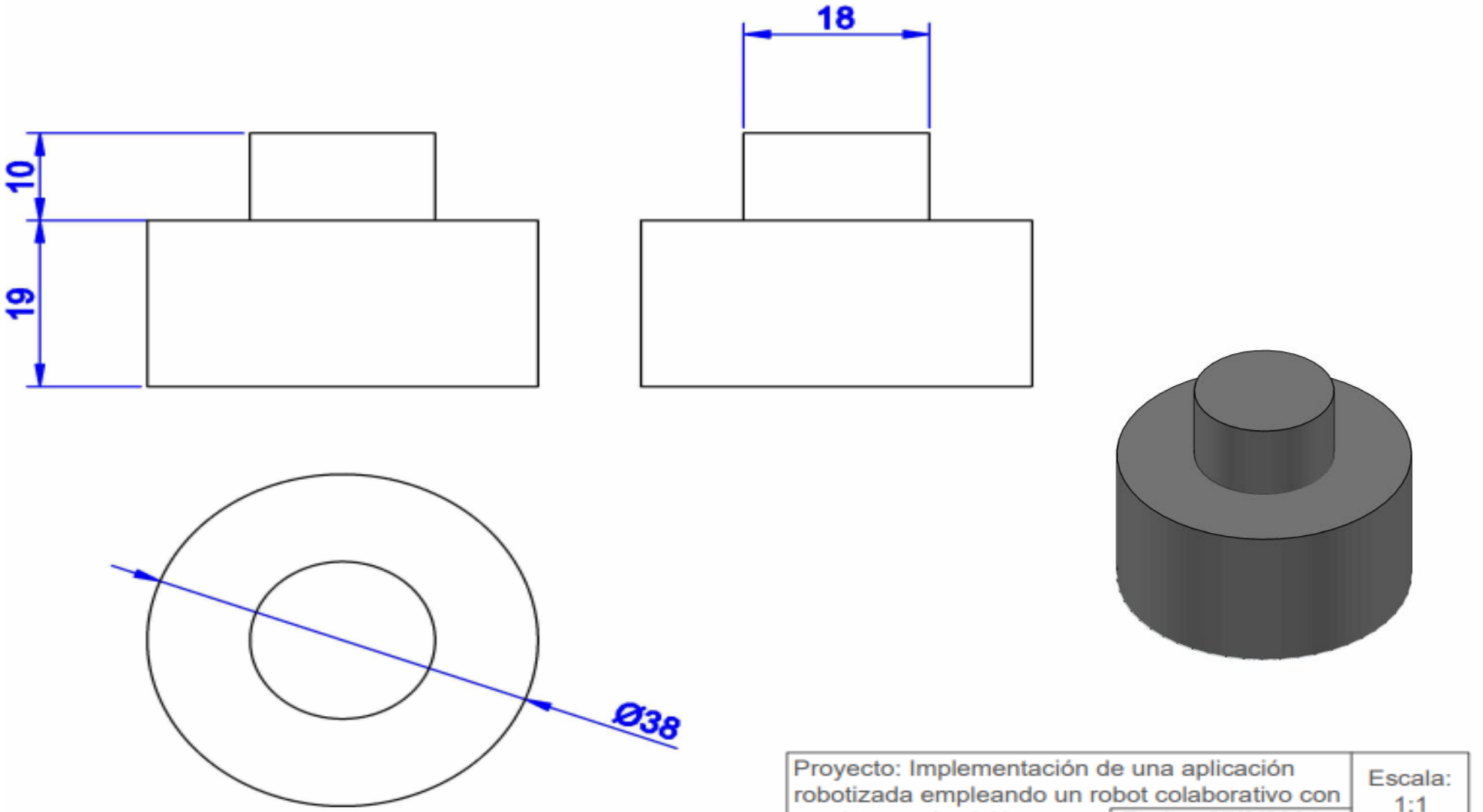
Plano 5: Diseño de la Pieza Cilíndrica “Hembra”.





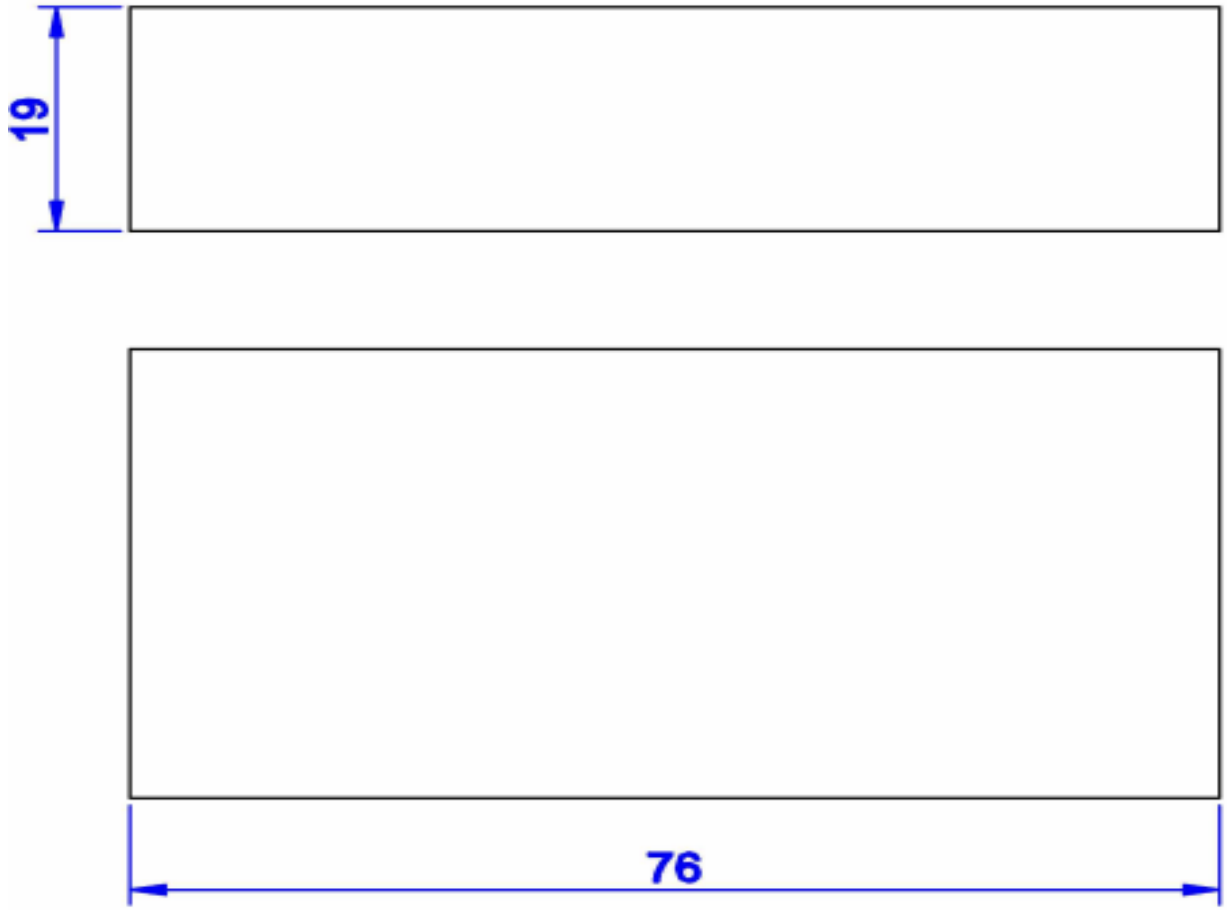
Proyecto: Implementación de una aplicación robotizada empleando un robot colaborativo con un sistema de visión. Fecha: 25/8/2021			Escala: 1:1
Autor: Alejandro García	Plano: Diseño Pieza Circular "Hembra"		Plano: 05

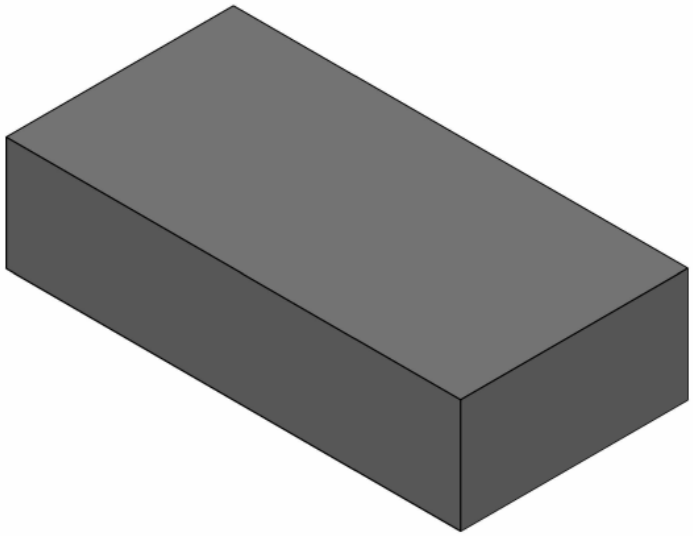
Plano 6: Diseño de la Pieza Cilíndrica “Macho”.



Proyecto: Implementación de una aplicación robotizada empleando un robot colaborativo con un sistema de visión. Fecha: 25/8/2021			Escala: 1:1
Autor: Alejandro García	Plano: Diseño Pieza Circular "Macho"		Plano: 06

Plano 7: Diseño de la Pieza Defectuosa.





Proyecto: Implementación de una aplicación robotizada empleando un robot colaborativo con un sistema de visión. Fecha: 25/8/2021			Escala: 1:1
Autor: Alejandro García	Plano: Diseño Pieza Defectuosa		Plano: 07

Documento III

Presupuesto

En el siguiente apartado se mostrará el desglose del coste del proyecto sin las amortizaciones de este.

1. Costes del Material empleado

Aquí se van a mostrar los costes que vienen dados por los materiales empleados del proyecto esto incluye tanto software como hardware.

Costes Material					
Referencia	*Material*	*Fabricante*	*Cantidad(Uds.)*	*Precio (€)*	*Total (€)*
M1	Robot Colaborativo UR3	Universal Robots	1	23.500,00	23.500,00
M2	Wrist Camera	Robotiq	1	5.500,00	5.500,00
M3	Ordenador	HP	1	600,00	600
M4	Piezas de Prueba	Impresión 3D Valencia	5	6,00	30
M5	Microsoft Windows 10	Microsoft	1	100	100
M6	Office Profesional 2020	Microsoft	1	100	100
M7	Simulador UR3	Universal Robots	1	0	0
				TOTAL	29.830,00 €

2. Costes de Mano de Obra

En este apartado se muestra el coste desglosado a cada etapa del proyecto en función del tiempo de cada tarea.

2.1 Costes Referidos a la etapa de Análisis y Planificación del proyecto

Etapa de Análisis y Planificación					
Referencia	*Tarea*	*Descripción*	*Horas (h)*	*Precio (€/h)*	*Total (€)*
A1	Planificación	Estudio del entorno y las condiciones de trabajo y de las características que debe tener la aplicación a diseñar.	20	20,00	400,00
A2	Investigación y documentación	Estudio sobre las diferentes opciones y documentación de la forma de programación, características, ventajas y desventajas, etc.	35	20,00	700,00
				TOTAL	1.100,00 €

2.2 Costes Referidos a la etapa de Diseño.

Etapa de Diseño					
Referencia	*Tarea*	*Descripción*	*Horas (h)*	*Precio (€/h)*	*Total (€)*
D1	Diseño de software	Diseño del código del programa del sistema robotizado a utilizar	25	20	500,00
D2	Diseño equipo de visión	Diseño del código referente al equipo de visión ya sea para la comunicación o para el software	10	20	200,00
D3	Test	Serie de test en un entorno simulado parecido al real que se hacen al programa para garantizar su correcto funcionamiento y corregir errores.	15	20	300,00
				TOTAL	1.000,00 €

2.3 Costes Referidos a la etapa de Implementación.

Etapa de Implementación					
Referencia	*Tarea*	*Descripción*	*Horas (h)*	*Precio (€/h)*	*Total (€)*
A1	Instalación	Instalación del equipo y de los softwares en el entorno de trabajo	4	20	80,00
A2	Pruebas de funcionamiento	Pruebas que se hacen en el entorno de trabajo para corregir errores imprevistos en el entorno simulado, además se hace un seguimiento de calidad de la aplicación	4	20	80,00
				TOTAL	160,00 €

3. Costes totales del proyecto

Costes Totales del proyecto	
Concepto	*Coste (€)*
Material	29.830,00
Mano de Obra	
Análisis y Planificación	1.100,00
Diseño	1.000,00
Implementación	160,00
Total sin IVA	32.090,00€
IVA	21%
TOTAL	38.828,9 €

Documento IV

Pliego de Condiciones

1. Definición y Alcance del Pliego

El objetivo de este documento es establecer las condiciones a cumplir y considerar durante la realización de este Trabajo Fin de Grado: implementación de una aplicación robotizada mediante un robot colaborativo y un sistema de visión.

El ámbito de aplicación de este documento se extiende a todos los equipos utilizados durante este proyecto ya sean mecánicos, eléctricos o electrónicos.

2. Condiciones generales y normativa

Tras la firma del contrato de compra por ambas partes se va a considerar que están de acuerdo con lo estipulado en este.

Los equipos y materiales utilizados en este proyecto serán los nombrados en este proyecto y deberán estar en perfectas condiciones para su uso y cumplir con la normativa que esté en vigor.

Tras comprar el equipo, el cliente tiene una garantía de 1 año tras haber recibido estos equipos. Esta garantía incluye el reemplazo del equipo o elemento en caso de ser defectuoso y estos se hará sin coste alguno.

La garantía no incluye el reemplazo o reparación del equipo en caso de tener una avería o desperfecto por un uso indebido o por un personal no autorizado.

Para la instalación el contratista tiene la obligación de proporcionar toda la información necesaria para una correcta instalación. Dicha instalación deberá ser realizada por personal capacitado.

Tras realizar la instalación se deben de hacer todas las pruebas necesarias con la supervisión de un agente del contratista hasta asegurarse de que el conjunto trabaja de una forma adecuada, correcta y en ausencia de problemas.

Todos los elementos de este proyecto cumplen con la normativa expuesta en el Real Decreto 1215/1997 sobre las disposiciones mínimas de seguridad y salud de los equipos de trabajo usados por trabajadores.

El robot colaborativo UR3 cumple con unas normas o parámetros de seguridad que se reflejan en su manual de usuario en el apartado de “Normas aplicadas”. Cuyas voy a destacar las más relevantes:

- **ISO 12100:2010:** Seguridad de la maquinaria, principios generales del diseño, evaluación y reducción de riesgos.
- **ISO 10218-1:2011:** Robots y dispositivos robóticos, requerimientos de seguridad para robots industriales. Parte 1: Robots.
- **ISO/TS 15066:2016**: Robots y dispositivos robóticos, requerimientos de seguridad para robots industriales. Operación Colaborativa.

- **EN 61000-6-4/A1:2011:** Compatibilidad electromagnética. Parte 6-2: Estándares generales-Inmunidad para entornos industriales. Parte 6-4: Estándares generales-emisión estándar para entornos industriales.
- **EN 61326-3-1:2008:** Equipamiento eléctrico para medida, control y usos de laboratorio. Requerimientos de compatibilidad electromagnética. Parte 3-1: Requerimientos de inmunidad para sistemas relacionados con la seguridad y para equipamientos para mejorar funciones relacionadas con la seguridad. Aplicaciones para industria general.
- **EN 60947-5-5/A11:2013:** Aparamenta de baja tensión y equipos de control. Parte 5-5: Dispositivos de circuitos de control y elementos de conmutación. Dispositivo de parada de emergencia con función de enclavamiento mecánico.
- **EN 60529/A2:2013:** Grados de protección frente a condiciones medioambientales (polvo, agua, etc). (Grados IP).
- **ISO 14118:**2000: Seguridad de la maquinaria, prevención de un arranque inesperado.
- **EN 60320-1:2015:** Acopladores de aparatos para propósitos familiares y similares. Parte 1: Requerimientos generales.
- **ISO 9409-1:2004:** Manipulación de robots industriales. Interfaces mecánicas. Parte 1: Placas.
- **EN 61140/A1:2006:** Protección ante descargas eléctricas. Aspectos comunes para la instalación y el equipamiento.
- **EN 60068-2-64:2008:** Pruebas ambientales. Parte 2-1: Tests -Test A: frío. Parte 2-2: Tests - Test B: Calor seco. Parte 2-27: Tests -Test Ea: Descargas eléctricas. Parte 2-64-Tests -Test Fh: Vibraciones.

3. Condiciones de carácter Económico.

El precio del proyecto en cuestión es fijo y no puede ser modificado mediante alguna negociación respecto al precio.

El pago de este se realizará en 2 partes:

1- Se abonará el 80 % del total tras firmar el contrato

2- Se abonará el 20 % restante en un periodo de 10 días naturales tras haber recibido todos los equipos.

El pago se hará mediante un cheque o una transferencia a la entidad bancaria indicada.

En el caso de no cumplir con los plazos de pago, esto supondrá una penalización o cargo de un 10 % del precio total al cliente.

En el caso de que no se entreguen los equipos en el plazo estipulado, el cliente tiene derecho a una compensación.

Una vez se acabe el periodo de garantía, cualquier consulta que haga el cliente le supondrá un coste en función de las horas empleadas.

4. Especificaciones de Ejecución.

Una vez instalados los equipos, la puesta en marcha se llevará a cabo tal y como se muestra en el manual de usuario adjuntado. Donde la programación de este sistema robotizado debe ejecutar lo descrito en este proyecto.

Para un correcto funcionamiento de este se debe tener en cuenta varias consideraciones:

- Hay que asegurarse de que los dos tipos de pieza (“Hembra” y “Macho”) deben de estar en el rango de visión de la cámara. Para ello se aconseja delimitar las dos zonas mediante una plantilla o algo parecido.
- Hay que asegurarse de que las piezas se depositen de la forma correcta (las de tipo “hembra” con el hueco hacia abajo y las del tipo “macho” el saliente hacia arriba)
- Estas piezas pueden llegar de una en una o varias a la vez además de que pueden estar en cualquier posición y orientación.
- Una vez ensamblada una pieza hay que asegurarse de que se retire la pieza ensamblada del área de trabajo.
- Señalizar el área de trabajo del robot, aunque sea colaborativo hay que evitar las colisiones al mínimo ya que estas pueden dañar el robot o causar heridas leves al personal.
- Este tipo de aplicación está pensado para piezas más pequeñas que el robot y que no puedan ser peligrosas en caso de alguna colisión. En caso de ser una pieza que pueda ser puntiaguda, sobresalga del brazo robótico o sea mínimamente peligrosa mientras esté en funcionamiento debe de estar o en una jaula de seguridad o en un espacio delimitado donde no haya ninguna persona u obstáculo dentro de este.
- Durante la cámara este localizando las piezas no es recomendable mover estas piezas ya que puede ocasionar un error de localización de estas.

El contratista no se hace responsable de los problemas que se pueden causar debido incumplir o no seguir estas instrucciones.