2022
INFORME

Proyectos

# “DIGIBOT”

## DIGIBOT DESARROLLO DE GEMELOS DIGITALES EN OPERACIONES ROBOTIZADAS

## INFORME DE RESULTADOS

**Número de proyecto: 22200005**
**Expediente: IMAMCA/2022/2**
**Duración: Del 01/01/2022 al 31/12/2022**
**Coordinado en AIDIMME por: SÁNCHEZ ASINS,JOSÉ LUIS**
**Línea de I+D: INDUSTRIA 4.0**

GENERALITAT VALENCIANA

IVACE
INSTITUTO VALENCIANO DE COMPETITIVIDAD EMPRESARIAL

# ÍNDICE

*1 Objetivos del proyecto. ..... 3*

*2 Resultados obtenidos ..... 3*

*3 Actividades realizadas, desarrollo del proyecto. ..... 4*

**3.1 Desarrollo de la aplicación de reconocimiento de objetos ..... 4**

3.1.1 Preparación de la captura de imágenes de entrenamiento ..... 4

3.1.2 Desarrollo de los algoritmos de identificación ..... 13

**3.2 Desarrollo de un sistema de agarre polivalente ..... 21**

3.2.1 Garra de sujeción por vacio ..... 21

3.2.2 Garra de sujeción tipo pinza. ..... 25

3.2.3 Resumen de pruebas de sujeción. ..... 28

**3.3 Implementación de la aplicación en un robot colaborativo ..... 29**

3.3.1 Prueba de comunicaciones MODBUS ..... 29

3.3.2 Funcionamiento del servidor. ..... 30

3.3.3 Pruebas realizadas con el servidor MODBUS ..... 31

3.3.4 Programación del robot colaborativo UR 16e ..... 36

# 1 Objetivos del proyecto.

El objetivo del proyecto, que se plantea a dos años (2022-2023) es el desarrollo de metodología para la programación de robots mediante IA a través del uso de gemelos digitales, para lo cual se pretende utilizar métodos de aprendizaje por refuerzo aplicados a un proceso previamente seleccionado.

Los objetivos específicos del proyecto son los siguientes:

- Evaluar técnicas de reconocimiento de objetos de cualquier geometría y en cualquier disposición espacial, en un plano, mediante entrenamiento de redes neuronales.
- Desarrollar una aplicación para la identificación, mediante visión artificial, de una cantidad limitada de objetos diferentes en cualquier disposición espacial.
- Implementar la aplicación anterior en el control de un robot para la recogida selectiva de objetos, en disposición aleatoria sobre un plano.
- Desarrollar una interfaz que permita realizar simulaciones de manejo del robot, utilizando RV.

# 2 Resultados obtenidos

Durante 2022 se han obtenido los siguientes resultados:

- Se ha definido la metodología para la programación de robots integrando técnicas de IA para la identificación de objetos
- Se han evaluado técnicas de reconocimiento de objetos de cualquier geometría y en cualquier disposición espacial, en un plano, mediante inteligencia artificial.
- Se han desarrollado redes neuronales para la identificación de objetos mediante imágenes, estableciendo su posición y el ángulo de giro de su eje de simetría.
- Se ha desarrollado un sistema de transferencia de los datos posición espacial-ángulo de giro a un robot, para que éste pueda recoger, de forma selectiva, un objeto concreto entre un conjunto de elementos diversos
- Se han evaluado diversos efectores para el agarre de piezas, para valorar su utilización como sistema universal de sujeción
- Se ha diseñado un sistema de sujeción específico, adecuado para un conjunto de piezas dadas de geometrías dispares

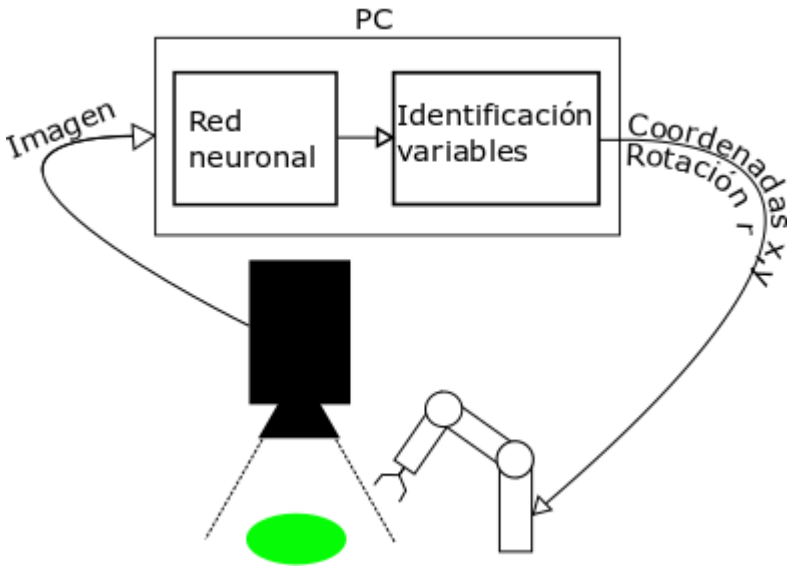
# 3 Actividades realizadas, desarrollo del proyecto.

## 3.1 Desarrollo de la aplicación de reconocimiento de objetos

Para desarrollar este sistema se utilizará un conjunto de algoritmos de inteligencia artificial que, previo entrenamiento, permitan reconocer los objetos seleccionados.

### 3.1.1 Preparación de la captura de imágenes de entrenamiento

Para obtener los mejores resultados, vamos a definir el proceso que se deberá ejecutar, y con ello las condiciones más favorables para llevar a cabo este reconocimiento.




La cámara obtendrá una imagen que será enviada a un ordenador que la procesará mediante el uso de redes neuronales convolucionales, que identificará si se encuentra alguna pieza en la imagen. En caso positivo, otro programa se ocupará de identificar en qué posición de la imagen se encuentran y cómo se traduce esto a las coordenadas reales y la forma en la que el robot debe recoger la pieza. Esta información será enviada al robot.

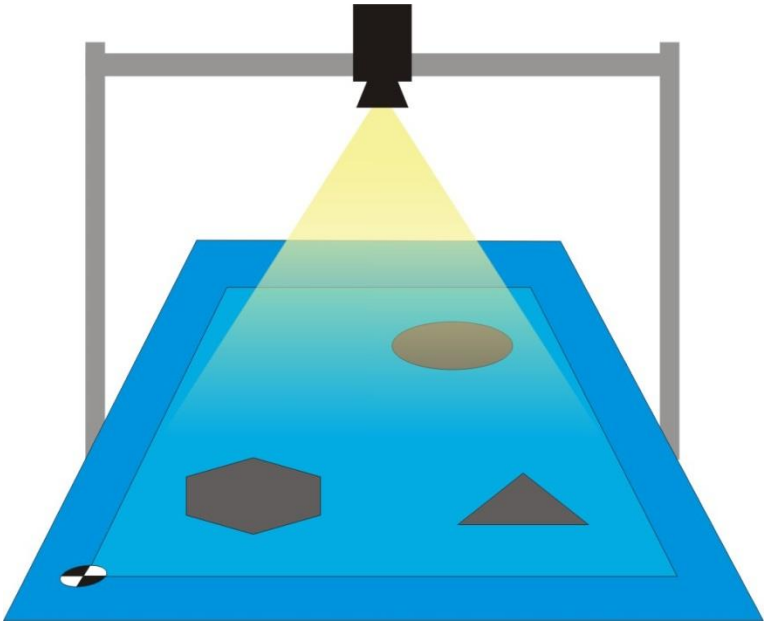
Con el objetivo de poder determinar las coordenadas referentes al robot, es necesario que la cámara y el robot se encuentren en posiciones fijas, así como que la cámara sea la misma. En caso de que estas variables cambien, se debería pasar por un proceso de calibración con el fin de obtener la asociación de coordenadas imagen - coordenadas robot.

Para las imágenes del entrenamiento de la red neuronal es necesario definir el tipo de piezas a identificar. Para seleccionarlas será necesario que cumplan con:

- Distintos materiales (metal, plástico, madera o compuestos)
- Distintas formas y tamaños (pero aptas para poder cogerlas con alguna hta.)

Para generar las imágenes es necesario disponer de la cámara o cámaras con su correspondiente óptica (las que se usarán

1. La cámara hay que montarla perpendicularmente al plano a fotografiar a una distancia fija.
2. Una vez instalada la cámara se realizará 1 fotografía para definir en el plano de trabajo el campo de visión de la cámara a la distancia de instalación.
3. Se marca dicho campo de visión para poder mover las piezas dentro del mismo.
4. Definido el campo de visión, se realizará una calibración para futuros desarrollos. Esta calibración permitirá calcular la relación mm/píxel de la imagen que servirá en las funciones de ubicación de piezas. Se utilizará un tablero tipo ajedrez con un tamaño de cuadro conocido para los posteriores cálculos. Además, se puede añadir un punto de referencia para establecer el punto 0 para las coordenadas.
5. Para las fotografías se colocarán las piezas en diferentes posiciones dentro del campo de visión. Además, para cada posición se modificarán las condiciones de luz (sobreexponiendo y subexponiendo).

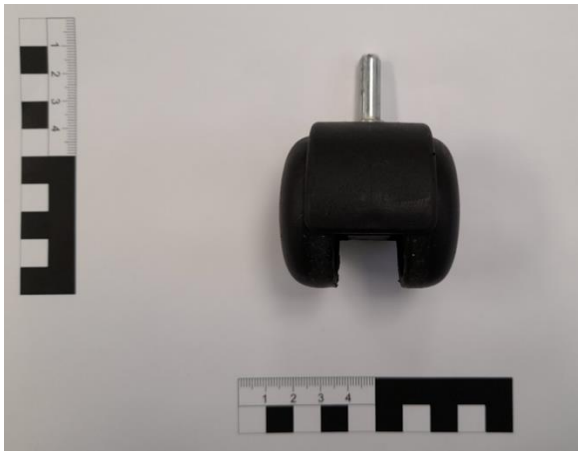


- ***Piezas para reconocer***

Las piezas seleccionadas son cinco objetos de distintas formas y tamaños. Todas las piezas son de plástico negro, esto nos permite que sea posible sacar correctamente todas las piezas con una sola configuración de parámetros de cámara. Ya que, si se tratara de materiales distintos, como metálicos, plásticos, maderas u otros, en la misma foto no saldrían con la calidad suficiente para poder ser reconocidos.







- ***Sistema de Visión Keyence Serie CV-X.***

Para la realización de las imágenes que se emplearán en el entrenamiento de las redes neuronales, se ha empleado el equipo de visión artificial con la cámara y lente que se empleará posteriormente para las aplicaciones desarrolladas para el reconocimiento de piezas mediante dichas RRNN.

Este sistema de visión está formado por un controlador (CV-X420), una cámara (CM-200) y una lente (CA-LH8). Este equipo representa una forma sencilla de resolver aplicaciones en las que se requiere un guiado de robot fácil y rápido.

Cámara y controlador Keyence. Fuente: https://www.keyence.com/

Se muestran a continuación las características técnicas del modelo CV-200M:

| | | |
|---|---|---|
| Modelo | | CV-200M |
| Tipo | | Cámara*1 |
| Elemento de imagen | | Elemento<br>receptor de imágenes CCD monocromático de 1/1.8 pulgadas, lectura de píxeles cuadrados/de todos los píxeles, 2,010,000 píxeles<br>Tamaño de<br>célula de la unidad: 4.4 x 4.4 µm 0.17 x 0.17 Mil |
| Conteo de píxeles válidos | | 1,920,000 píxeles 1600 (H) x 1200 (V) *2 |
| Frecuencia de transferencia de píxeles | | 40 MHz |
| Sistema de barrido | | Progresivo (58.5 ms: modo de 2,000,000 píxeles, 47.6 ms: modo de 1,000,000 píxeles)<br>Entrelazado (32.7 ms: modo de 2,000,000 píxeles, 27<br>ms: modo de 1,000,000 píxeles) |
| Sistema de transferencia | | Transferencia serial digital |
| Obturador electrónico | | 1/15, 1/30, 1/60, 1/120, 1/240, 1/500, 1/1000, 1/2000, 1/5000, 1/10000, 1/20000, 0.05 ms a 9000 ms pueden fijarse con valores numéricos. |
| Montura del lente | | Montura C |
| Resistencia ambiental | Temperatura ambiente | De 0 a +40 °C 32 a 104 °F |
| | Humedad relativa | 35 a 85 % HR (Sin condensación) |
| Peso | | Aprox. 110 g (sin lente) |

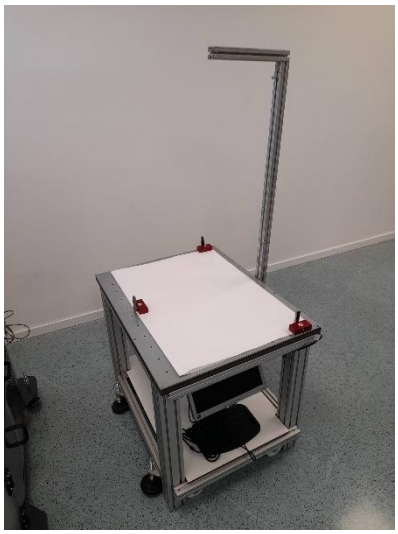
Se muestran a continuación las características técnicas del modelo CA-LH8:

| | |
|---|---|
| Modelo | CA-LH8*1 |
| Punto focal | 8 mm 0.31" |
| Rango de número F (apertura) | F1.4 a F16 |
| WD mínima | 0.1 m 0.3' |
| Montaje | Montaje C |
| Tamaño del filtro | 27.0 mm<br>1.06" P0.5 |
| Sensor de imagen compatible | 2/3" |
| Distorsión de TV | -0.6 % (-0.28 %)*2 |
| Capacidad de resolución | 100 ciclos/mm en el centro, 80 ciclos/mm en la periferia |
| Rango de temperatura/ humedad ambiente | 0 a +50 °C 32 a 122 °F, 35 a 80 % HR (Sin condensación) |
| Peso | Aprox. 90 g |

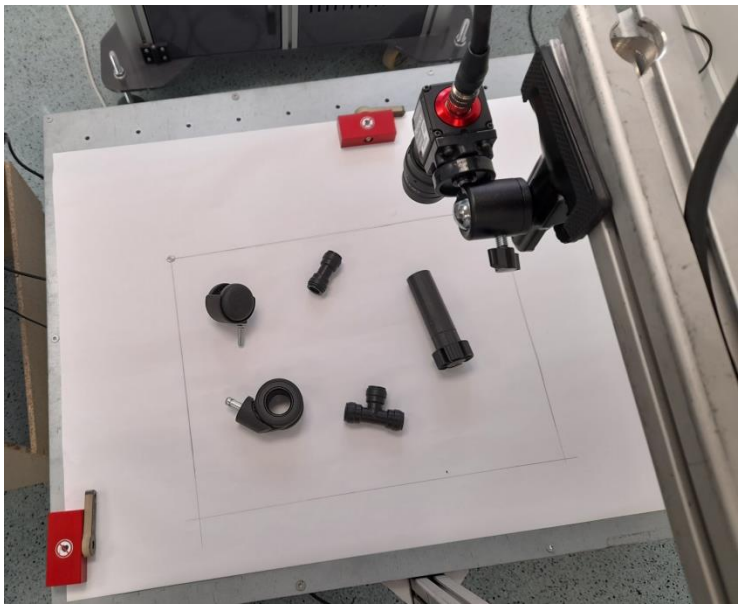
- ***Disposición de trabajo.***

Las imágenes del entrenamiento de las redes neuronales se han realizado estableciendo un área de trabajo similar a la que se trabajaría con el robot

Para ello se ha montado una mesa con un soporte, instalando la cámara a una altura a la que permita obtener una imagen con todas las piezas en el campo de visión.



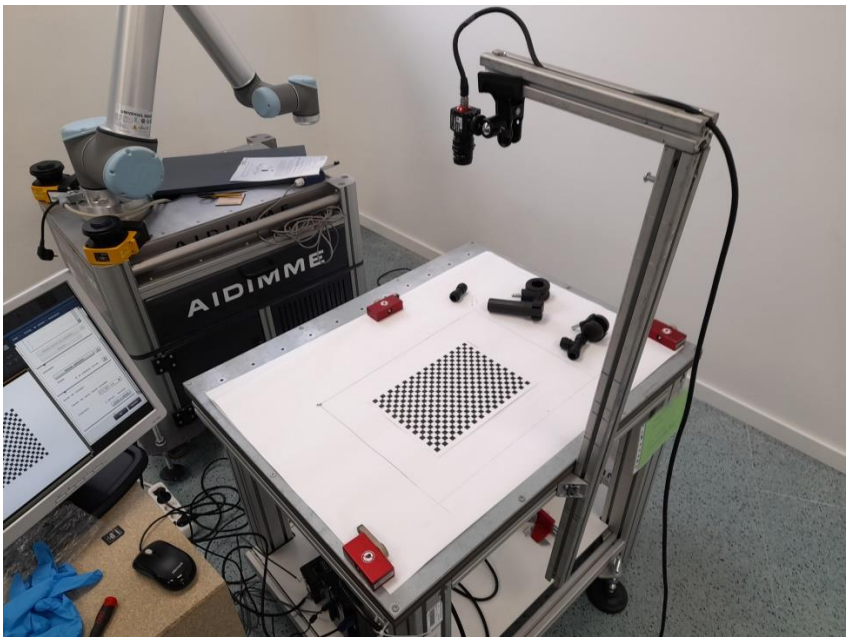
Además, se ha tenido en cuenta una altura suficiente para que entre el plano de trabajo y la cámara (que se supone instalada en el cabezal del robot), quepan las herramientas necesarias para que el robot pueda agarrar las piezas.

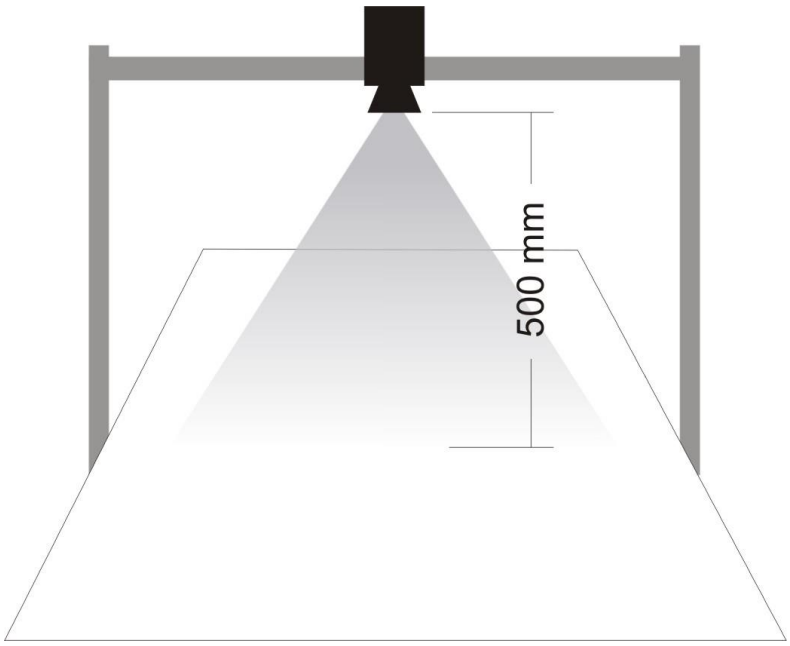



Una vez establecida la disposición de los elementos de visión y configurada la cámara para la toma de las imágenes, se marca en el plano de trabajo, el área de visión de la cámara, para tener visible la zona en el que se pueden mover las piezas.

Previamente al proceso de toma de imágenes, se realiza un calibrado de la cámara, para corregir la deformación que se genera por la lente.

Para esa calibración se ha empleado un patrón de cuadros con una dimensión de 10 mm x10 mm.




De la instalación y calibración realizada, se extraen los siguientes datos.

Altura de instalación de la cámara: **500 mm**

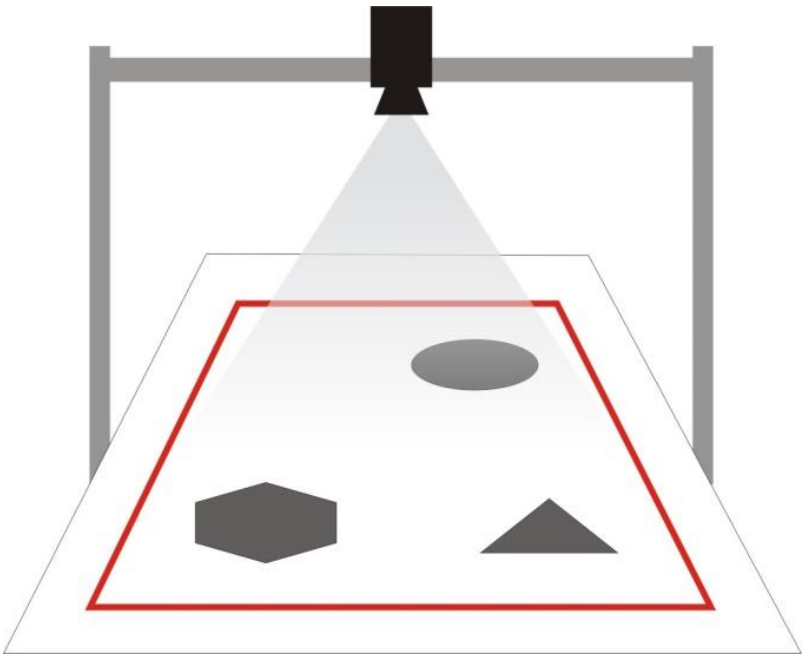
Relación mm/px. Resultante de la calibración: **0,284224 mm/px**

El tamaño teórico del campo de visión atendiendo a las características técnicas de la cámara y lente es:

| Tamaño objeto | | Dist. Trabajo | Precisión teórica máx | |
|---|---|---|---|---|
| **Horizontal (mm)** | Vertical (mm) | Z (mm) | Horizontal | Vertical |
| **450** | 333,4 | 501,39 | 0,2813 | 0,2779 |

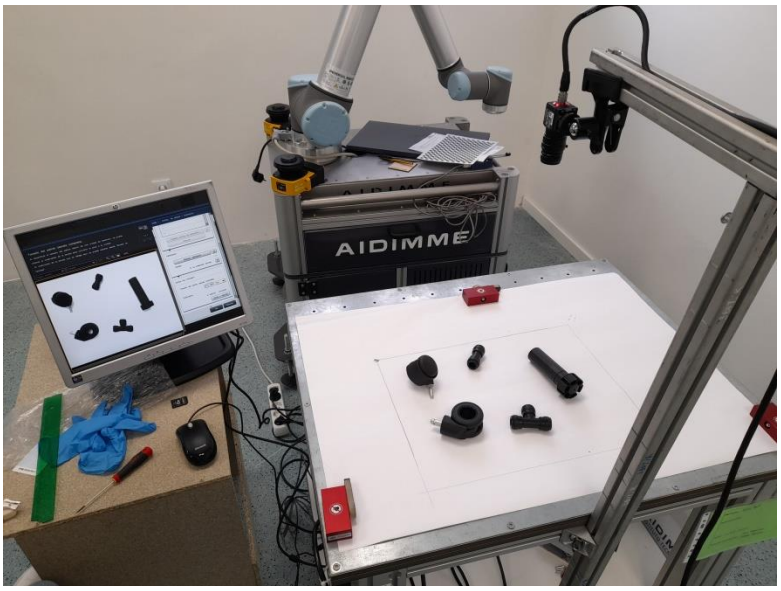
Si calculamos el tamaño del campo de visión con los datos obtenidos, teniendo en cuenta que la imagen generada por la cámara es de 1600x1200 px., el tamaño real resultante debe ser:

| | Tamano px | mm/px | mm |
|---|---|---|---|
| Horizontal | 1600 | **0,284224** | 455 |
| Vertical | 1200 | **0,284224** | 341 |



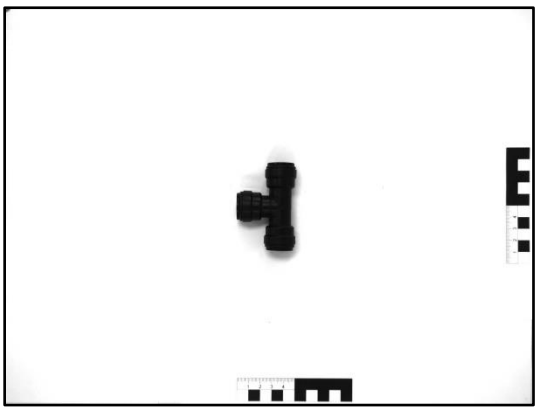
- ***Obtención de Imágenes.***

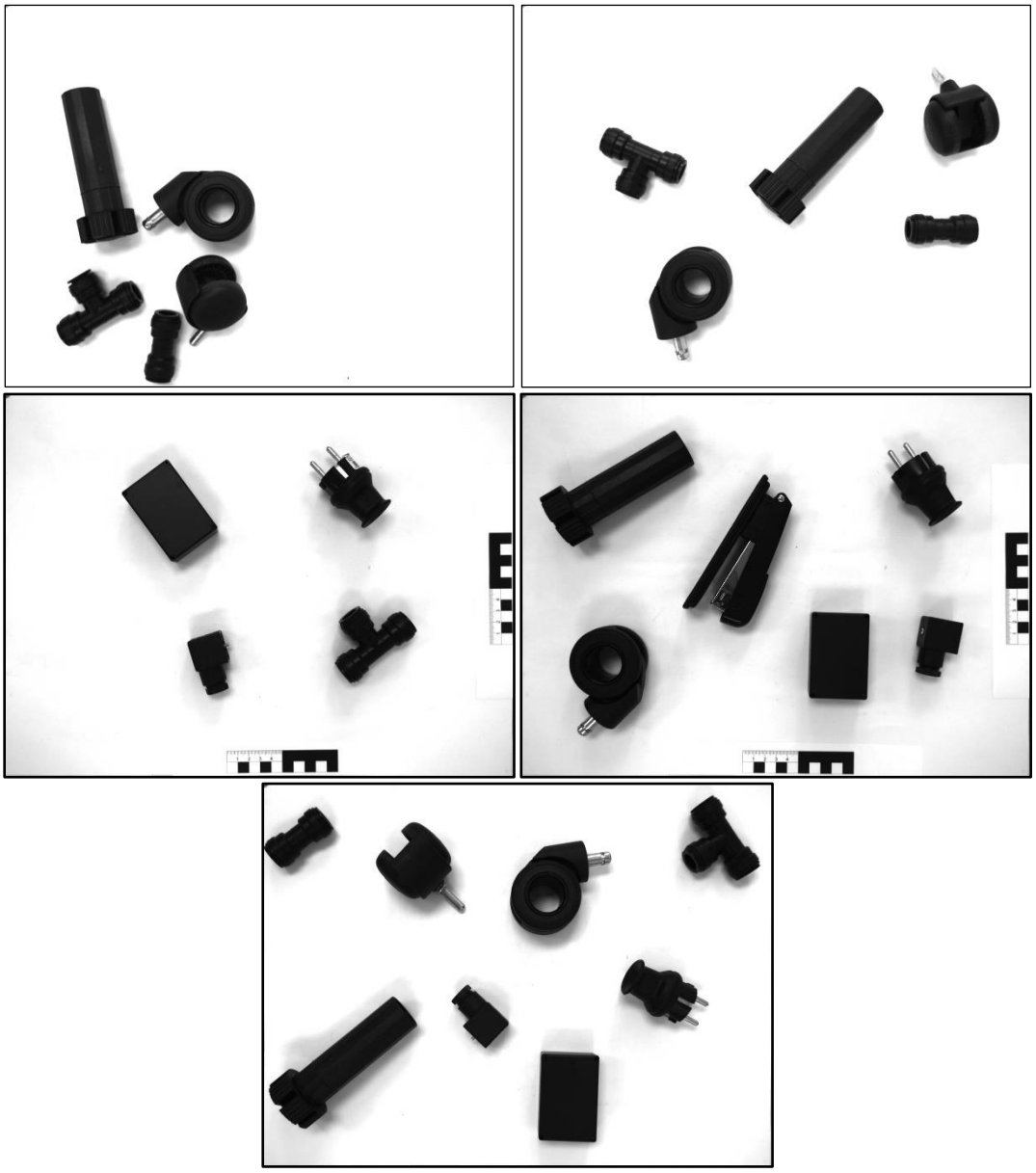
Tras la instalación y configuración del equipo, la calibración de la cámara y mediciones necesarias, se realiza la toma de imágenes para el entrenamiento de la inteligencia artificial.




Estas se han realizado diferenciando varios tipos de imágenes.

- Imágenes de cada pieza para enseñar al sistema cual es cada una de las piezas
- Imágenes combinando las piezas de 2 en 2
- Imágenes con las 5 piezas agrupadas en el área de visión
- Imágenes con las 5 piezas repartidas en el área de visión
- Imágenes con 1 piezas de las entrenadas y otros elementos no entrenados.
- Imágenes con varias piezas de las entrenadas y otros elementos no entrenados.
- Imágenes con todas las piezas entrenadas y no entrenados.





### 3.1.2 Desarrollo de los algoritmos de identificación

El proyecto Digibot ha supuesto una cantidad significante de desafíos a nivel de inteligencia artificial, en especial a lo que se refiere a la obtención, análisis e interpretación de resultados a través de imágenes. En particular, se puede dividir el trabajo realizado en el campo de la visión artificial en dos partes: Sistema dinámico de selección e interpretación de coordenadas y el Sistema de detección e identificación de objetos.

Seguidamente se desarrollan estos dos aspectos, así como la manera en la que se articula la conexión con el robot para que este pueda realizar su labor.

- Sistema dinámico de selección e interpretación de coordenadas

La necesidad de este sistema radica en la casuística de un sistema conformado por dos elementos principales: el sistema de visión y el robot.

Diagrama de la disposición del robot y el sistema de visión.
(Fuente: elaboración propia)

El sistema de visión es capaz de entender en qué punto se encuentra un objeto respecto a su posición, es decir, desde un sistema de referencia completamente arbitrario. Para que el robot sea consciente de a qué punto debe desplazarse desde su referencia, es necesario hacer un cambio de coordenadas desde las coordenadas obtenidas por el sistema de visión, hasta las utilizadas por el robot.

Este cambio es relativamente sencillo de hacer programáticamente, sin embargo, explicitar este cambio en la programación del sistema implica que cualquier pequeño movimiento de la cámara o el robot desde su punto de referencia original implicarían la reprogramación del sistema, inhabilitándolo completamente hasta que los ingenieros realizaran un nuevo ejercicio de ajuste y programación.

Por esta razón, es necesaria una solución que permita definir un sistema de coordenadas común para ambas partes de manera rápida, un sistema de calibración que además permita adaptar el sistema de visión a las condiciones lumínicas y ambientales de ese momento.

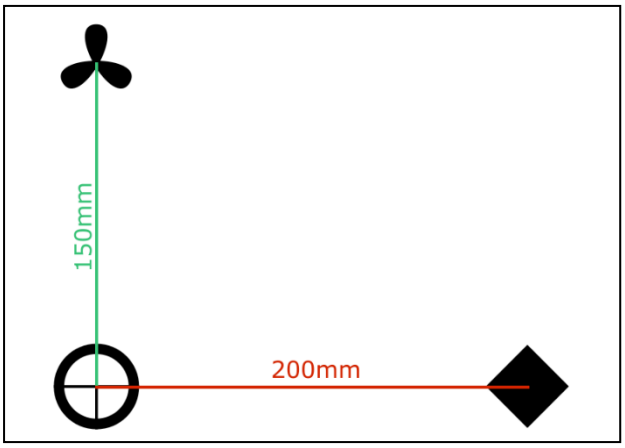
- Diseño del sistema

Se ha diseñado un sistema de posicionamiento que es capaz de establecer coordenadas que va a enviar el sistema de visión al robot de manera dinámica. Para ello, es necesario fijar un punto de inicio en el espacio de trabajo, además será necesario que el sistema pueda identificar en qué dirección crece el eje x y el eje y. Para este cometido, se van a utilizar 3 símbolos, el primero de ellos indicará el punto de origen, mientras que la unión mediante una línea recta y orientada de este con los otros dos definirá dónde se encuentran los ejes.

Los símbolos elegidos para este cometido son los siguientes:

Símbolos del sistema de referencia
(Fuente: elaboración propia)

En este caso, la cruz indica el punto 0, el vector rombo-cruz indica la dirección del eje de abscisas y el vector trébol-cruz hace lo propio con el eje de ordenadas. Sin embargo, falta definir la escala entre estos, para ello, se va a utilizar una medida estándar de forma que, al identificar la distancia entre 2 símbolos, se conozca automáticamente la escala. En particular, la distancia entre la cruz y el trébol será de 150mm y de 200mm entre la cruz y el trébol, tal como se muestra en la siguiente ilustración:




Sistema de referencias
(Fuente: elaboración propia)

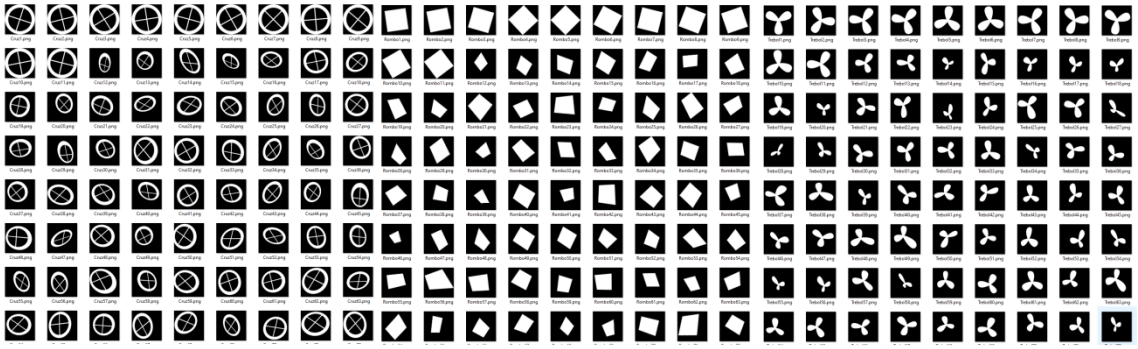
Estas medidas permiten maximizar el tamaño de los símbolos, haciendo que sean más fácilmente identificables a la vez que sean más fáciles de identificar desde mayores distancias, a su vez, se maximiza la distancia que existe entre los símbolos, pero permitiendo que estos quepan perfectamente dentro de una hoja de papel DIN A4.

- **Identificación mediante visión artificial**

A continuación, es necesario desarrollar un sistema de visión artificial que identifique las figuras y genere las coordenadas, para esto, se ha utilizado Python, la librería de visión por computadora OpenCV y la librería Keras y Tensorflow para generar el aprendizaje automático, por último, para desarrollar una interfaz gráfica de usuario (GUI), se ha utilizado la librería tkinter.

El primer paso ha sido el entrenamiento de un sistema de inteligencia artificial que sea capaz de, dada la imagen de una pieza, identificar de si se trata de una de las figuras, y de ser así, de cuál. Para esto se comienza generando una base sólida de ejemplos que pueda reconocer la cámara, normalmente sería necesario realizar muchísimas fotos de los símbolos, sin embargo, al tratarse de símbolos bidimensionales, monocromáticos y tener el diseño 2D en formato vectorial en el ordenador, es posible realizar variaciones de estas figuras programáticamente, permitiendo así tener

miles de ejemplos de entrenamiento en cuestión de segundos, tal como se puede ver en la siguiente ilustración:



Conjunto de entrenamiento creado programáticamente.
(Fuente: elaboración propia)

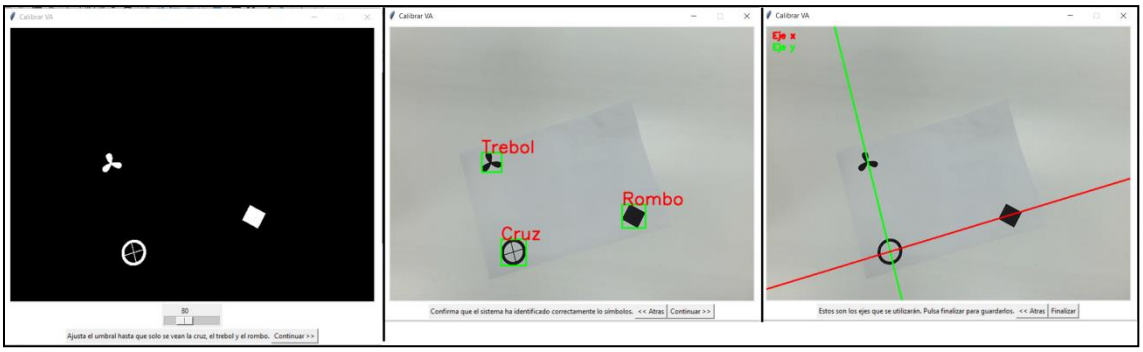
Es necesario ahora utilizar algoritmos de clasificación para obtener un modelo que nos permita clasificar la imagen observada por la cámara. Para esto, al tratarse de un conjunto de entrenamiento de imágenes, se han utilizado redes neuronales convolucionales, este tipo de algoritmo intercala dos tipos de capas distintas (Convolución y Pooling) para extraer cuales son los puntos identificativos de cada imagen y poder así diferenciar unos símbolos de otros.

Una vez con el modelo entrenado es posible realizar clasificaciones. Sin embargo, todavía es necesario un sistema para extraer las figuras de una imagen más grande. Se parte de la premisa de que el espacio de trabajo contará con un fondo blanco, por ello, se utiliza la herramienta de umbral para dividir el espacio, y después se identifican las componentes conexas dentro del espacio de trabajo, obteniendo así las posiciones de los candidatos a figura. Una vez obtenidas, se extrae la subimagen, y esta se envía al modelo entrenado, que identificará de qué figura se trata, fijando así las coordenadas de esta.

Por último en este apartado, para facilitar al usuario la calibración del sistema mediante la fijación de un sistema de referencia, se ha desarrollado una interfaz gráfica de usuario, esta consta de 3 pantallas.

- La primera pedirá al usuario que fije un nivel de umbral de trabajo, esto facilita el funcionamiento del sistema en distintas condiciones lumínicas.
- La segunda pantalla pide al usuario confirmación de que se han identificado las figuras correctamente, asegurando así que el sistema de referencias creado es el que se desea.
- Por último, la pantalla final muestra una previsualización de los ejes de coordenadas que van a ser utilizados.

Se pueden ver estas pantallas en la siguiente ilustración:




Pantallas del sistema de calibración de VA. (Fuente: elaboración propia)

El resultado de todo este proceso es un archivo json que guarda la posición de las tres figuras, así como el umbral definido por el usuario. De esta forma, cuando el sistema de identificación de piezas funcione, leerá el nivel de umbral que debe aplicar y sobre qué coordenadas debe enviar los datos.

- **Sistema de detección e identificación de objetos**

El siguiente paso es el desarrollo del sistema de detección e identificación de objetos. Para el proyecto se han identificado 5 tipos distintos de objeto que podrá reconocer el sistema de visión y manipular el robot. En este caso serán: 2 tipos distintos de ruedas de silla de oficina, un conector de tubería recto, un conector de tubería en T, y una pata de mueble de cocina:

Piezas a reconocer por el sistema de visión.
(Fuente: elaboración propia)

El sistema de visión debe ser capaz de obtener 4 datos: posición en X, posición en Y, tipo de pieza y rotación.

- **Identificación de posición X, Y**

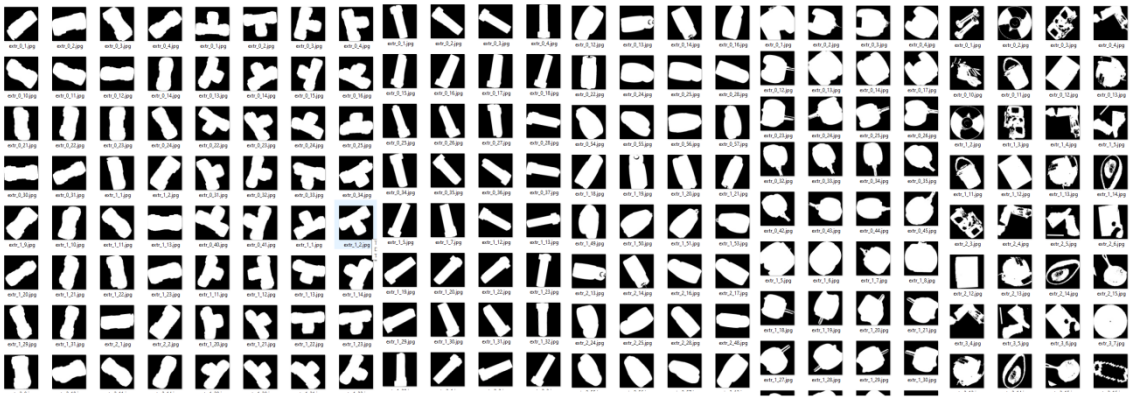
En este caso, se aplica un funcionamiento similar al explicado en el apartado anterior, se utiliza un sistema de umbral para separar la pieza del fondo, al realizar esto, se identifica la componente

conexa a la que pertenece, de ahí obtenemos el cuadrado en el que encaja la pieza, las coordenadas X, Y del centro de este cuadrado serán las coordenadas que debamos enviar al robot.

- Identificación de tipo de pieza

Una vez más, se aplica un sistema similar al utilizado en la identificación de figura del apartado anterior.

Se comienza generando una base de imágenes para realizar un entrenamiento, para ello en este caso, al tratarse de piezas físicas, sí es necesario obtener imágenes que permitan generar una base de entrenamiento suficientemente rica. Además de esto, se pueden utilizar métodos programáticos para realizar variaciones plausibles de estas imágenes, ampliando así en gran medida el conjunto de entrenamiento. Se puede ver una muestra de este conjunto en la siguiente ilustración:



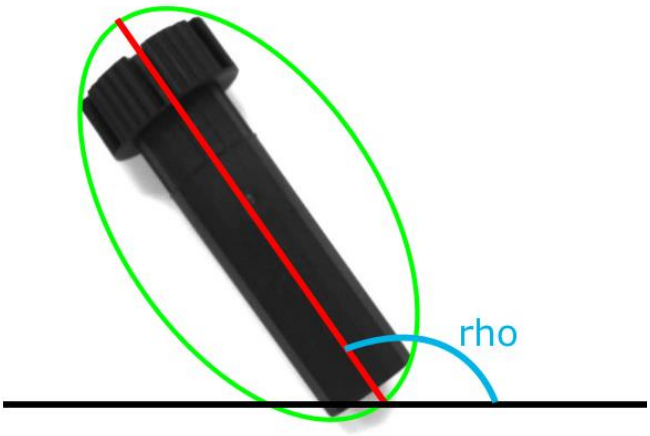
Vista parcial del conjunto de entrenamiento (Fuente: elaboración propia)

Se puede apreciar también en las últimas columnas de la ilustración anterior, que se incluye una sexta categoría de entrenamiento en la que se incluye una colección aleatoria de objetos, esto se realiza para generar una categoría "desconocido" de forma que el sistema tenga la posibilidad de descartar un objeto en caso de que no lo conozca en lugar de tener que, forzosamente, agruparlo en una de las 5 categorías. De esta forma, cuando se introduzca un objeto en el área de trabajo distinto, el sistema lo obviará.

Por último, se realiza igualmente el aprendizaje la extracción y la clasificación como se ha desarrollado en el apartado anterior.

- **Identificación de rotación**

Para obtener la rotación de un objeto, se circunscribe dentro de una elipse utilizando la librería OpenCV, a continuación, se identifica el eje mayor de la elipse, y se haya el ángulo de rotación de este eje respecto a la horizontal, tal como se muestra en la imagen:

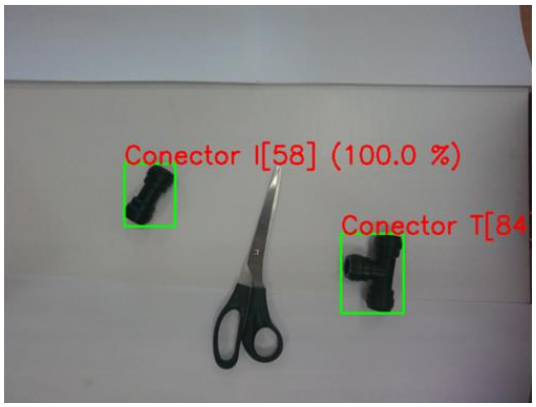



Obtención de la rotación de una pieza.
(Fuente: elaboración propia)

- **Resultados del sistema**

En las siguientes imágenes puede apreciarse cómo el sistema identifica las piezas del conjunto objetivo, marcándolas y fijando su posición y ángulo, e identifica las que no pertenecen al conjunto, sin marcarlas ni posicionarlas.






- **Comunicación con el robot y sincronía de datos**

El último paso para completar el sistema, es enviar los datos al robot. Para este cometido, el sistema de visión se ocupará de mantener actualizada la información de la posición de una de las piezas identificadas en un archivo JSON. Por su parte, un script independiente se ocupa de preguntar al robot continuamente su estado utilizando el protocolo MODBUS, el robot utiliza una variable para indicar si está funcionando o por el contrario, está listo para recibir alguna pieza con la que trabajar.

Cuando el script detecte que efectivamente el robot está esperando pieza, y que el sistema de visión ha identificado una pieza correctamente y ha incluido su información en el archivo JSON, utilizará nuevamente el protocolo MODBUS para enviarle al robot la información necesaria de la pieza. En el apartado 3.3.1 se describe las pruebas realizadas con este protocolo.

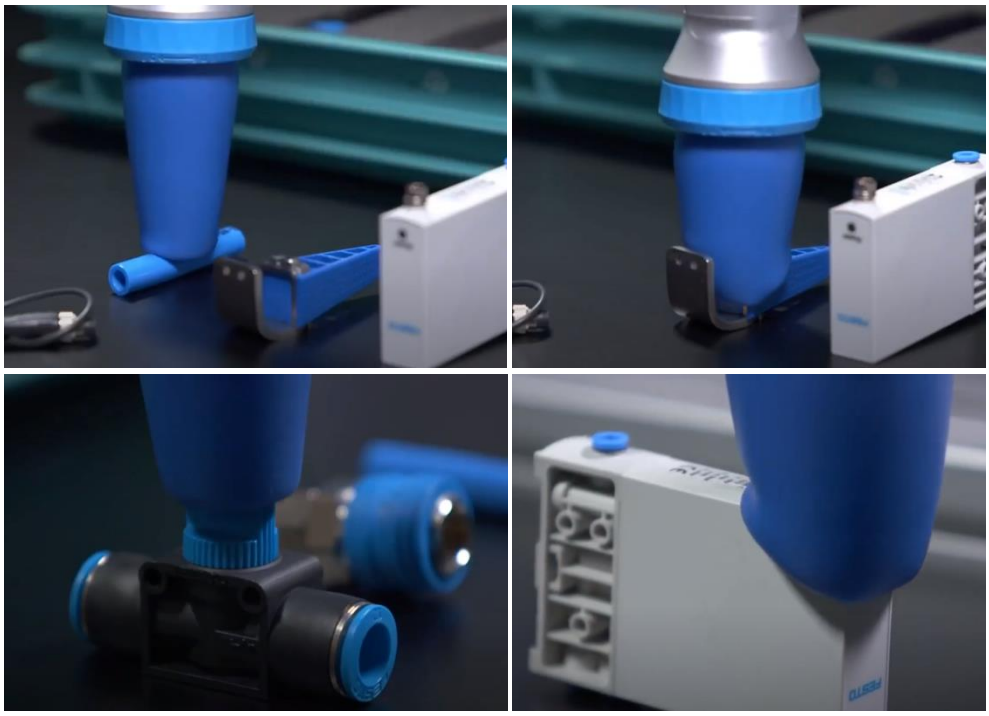
## 3.2 Desarrollo de un sistema de agarre polivalente

En un primer análisis se examina la morfología de las piezas a manipular. Es importante observar la morfología de las superficies (planas, convexas, esquinas,…), la rugosidad superficial, el peso y centro de gravedad, el tamaño y el tipo de material para evaluar su resistencia.
Todos estas características son importantes para garantizar un buen agarre y que la pinza pueda abarcar bien toda la pieza.


En el sistema de agarre polivalente se han estudiado dos tipos de solución, sujeción por vacio y agarre tipo pinza. La pinza por vacio tiene las ventajas de su alta adaptabilidad a cualquier forma y la posibilidad de agarrar objetos frágiles. Por el contrario, la fuerza de agarre es más reducida que en otro tipo de pinzas y el posicionamiento de las piezas cuando se liberan es más inexacto.

### 3.2.1 Garra de sujeción por vacio

Se trata de una garra adaptativa DHEF de la marca FESTO. Esta garra realiza la sujeción mediante vacio. En una primera fase de aproximación la parte flexible está hinchada para que al acercarse al objeto rodearlo. Una vez realizada la aproximación se produce el vacio en la parte flexible de tal manera que el objeto queda sujeto a la garra. La flexibilidad que presenta está garra es enorme y además permite el uso en robots colaborativos.

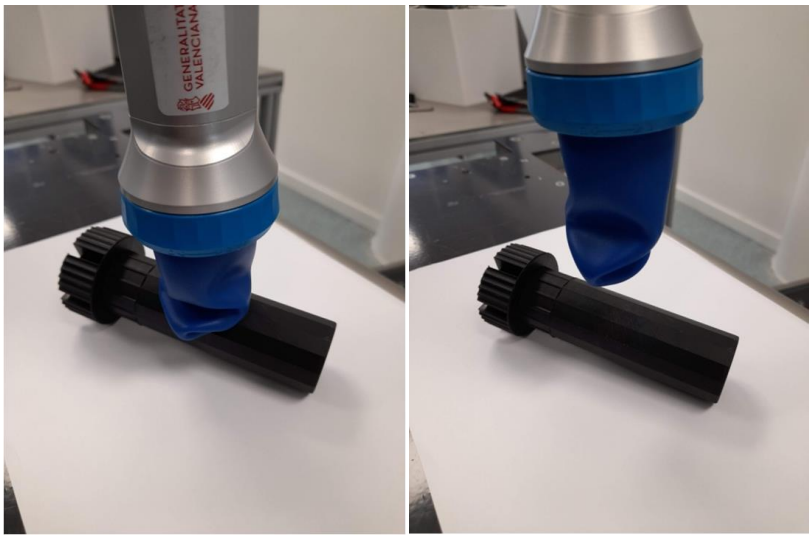


Ejemplos de agarre de piezas con las garra adaptativa DHEF.
(Fuente: Festo.com)

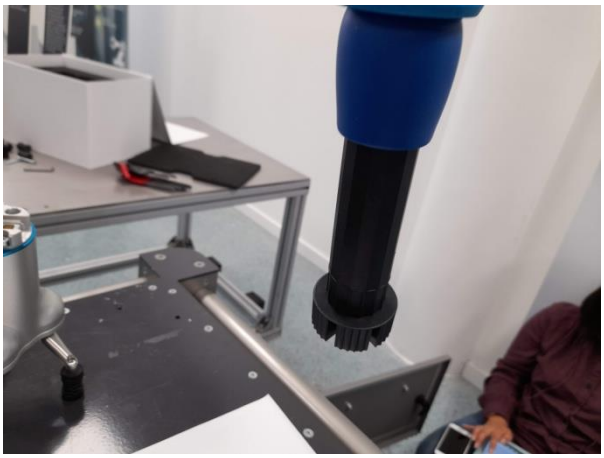
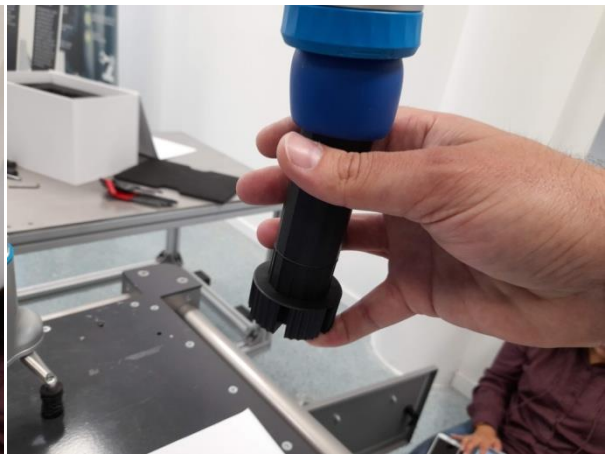
Se han realizado las pruebas de agarre con las distintas piezas. A continuación se detallan los resultados de las mismas:

- **Pieza 1.**

Se probaron dos posiciones, tumbada y en vertical. Se pudo comprobar que en la posición tumbada la garra no era capaz de sujetar la pieza con suficiente fuerza para garantizar el agarre.



En vertical si que había un buen agarre pero dado que en la otra posición no es valida y las piezas no tienen una posicion fija de agarre, el uso de esta pinza para la pieza 1 no es viable.

- **Pieza 2:**

Se probaron las dos posiciones posibles, tumbada y vertical. En posición vertical la pieza era menos estable pero se comprobó que en ambas se podía realizar el agarre.



- **Pieza 3:**

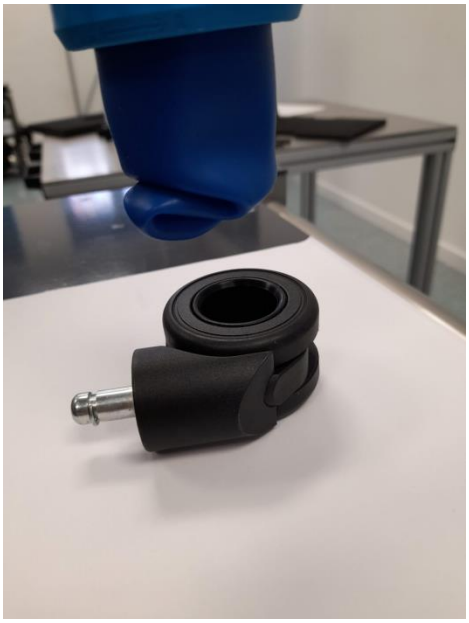
Se probó la posición tumbada dado que es la unica posicion estable sin que exista un util de posicionamiento. En las pruebas no pudo garantizarse el agarre.


- **Pieza 4:**

La pieza 4 es muy inestable cuando existe el contacto en la fase de aproximación y se desplaza cuando la garra intenta abrazarla. Al realizar el vacio la garra no esta en contacto suficiente y no se puede garantizar el agarre.

- **Pieza 5:**

Al igual que con la pieza 4, las pruebas no pudierón garantizar el agarre en gran parte debido a la inestabilidad de la pieza cuando se realiza el contacto con la garra pero en el punto de agarre que presentaba mas estabilididad se determinó que el tamaño de la zona de agarre era demasiado grande y la garra no podía abrazarlo suficientemente.

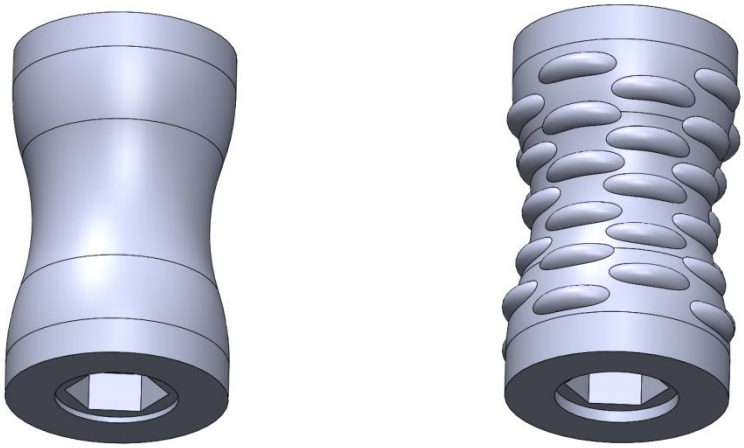


### 3.2.2 Garra de sujeción tipo pinza.

La garra de sujeción tipo pinza es una pinza de 3 dedos flexible de la marca OnRobot. El sistema de agarre de esta pinza se lleva a cabo mediante la aproximación a la pieza y el cierre de los tres dedos mediante su rotación para atrapar la pieza en el interior.

Aunque esta garra presenta una gran flexibilidad tiene una menor adaptabilidad a las superficies y la zona de agarre se limita al volumen delimitado por la altura de los dedos y el radio de cierre de los mismos. La ventaja frente a la garra de vacio es que tiene una fuerza de agarre mucho mayor si la pieza es resistente.

Revisando la morfología de las piezas se puede observar que todas ellas tienen superficies convexas y como los dedos que vienen con la garra son totalmente cilindricos se ha decidido rediseñar y fabricar unos dedos con una superficie concava que mejore el agarre. En otra versión se han incluido resaltes para favorecer la rugosidad.



Modelos 3D de los dedos desarrollados para mejorar la sujeción.
(Fuente: elaboración propia)

Pinza OnRobot y dedos desarrollados para la sujeción.
(Fuente: elaboración propia)

Al igual que con la pinza adaptativa de vacio se han realizado las pruebas de agarre con las distintas piezas. A continuación se detallan los resultados de las mismas:

- **Pieza 1:**

Se probó la posición tumbada puesto que es la unica posicion utilizada en el proyecto. Las pruebas pudierón validar el agarre tal como puede verse en las imágenes.

- **Piezas 2 y 3:**

Se realizarón las pruebas con la posición tumbada y como son piezas con el mismo diametro se pudo realizar el agarre. En las pruebas se vió que era bastante habitual que en el momento del agarre hubiera algún movimiento relativo entre las piezas y las garras debido a la rotacion de los dedos. Ademas la pieza 3 presento alguna dificultar en el posicionamiento de la garra previo al cierre porque habian mas posibilidades de choque con el conducto central. Para evitar estos problemas basta con tener en cuenta la posicíon inicial de la garra en el acercamiento y la apertura.

- **Piezas 4 y 5:**

Son piezas similares y se realizarón las pruebas con la posición tumbada en ambos casos. Al igual que en las piezas 2 y 3, se ha tenido algún problema en la aproximacion cuando un dedo coincidia con la posición del vastago metálico de la rueda.En el caso de la pieza 4 el agarre era algo mas inestable en la zona de contacto con la rueda pero se ha podido garantizar el agarre.

### 3.2.3 Resumen de pruebas de sujeción.

Tras la realización de las pruebas de sujeción se han analizado los resultados y se ha podido determinar que, a pesar de la versatilidad de la garra de vacío, la inestabilidad y morfologia de las piezas 4 y 5 (ruedas) hace inviable su utilización. Alternativamente se podrian utilizar si el uso permite incorporar un util de posicioando de las piezas que en el proyecto actual no tiene sentido.

La garra tipo pinza se ha demostrado como la más eficaz para las piezas empleadas en el proyecto y se ha mejorado su agarre implementando una modificación en los dedos para favorecer la sujeción de piezas convexas como las ruedas o tubos que se estan empleando. Aun así se ha determinado que en las piezas 3, 4 y 5 se debe prestar especial atencion al posicionado relativo de la pinza y la pieza para evitar que los dedos choquen contra las partes de las piezas en el momento de cierre de los dedos.

## 3.3 Implementación de la aplicación en un robot colaborativo

### 3.3.1 Prueba de comunicaciones MODBUS

Debido a que el robot UR únicamente dispone de un cliente para el protocolo de comunicaciones industriales Modbus y carece de un servidor Modbus propio hemos necesitado desarrollar un servidor de dicho protocolo propio que actúe de intermediario entre el sistema de reconocimiento de imágenes y la controladora del robot.

Las características del servidor MODBUS son las siguientes:

Tamaño

El servidor de Modbus proporciona un banco de memoria formado por registros (palabras de memoria de 16 bits de tamaño) denominados por el protocolo como "holding registers". El banco de memoria permite direccionar una cantidad máxima de 65536 registros diferentes.

Estructura de datos

Por otra parte el protocolo Modbus explicita que el orden de los bytes es de tipo Big-Endian, es decir el byte mas significativo precede al menos significativo. Sin embargo, el protocolo Modbus no indica cual es el orden a seguir si se necesita trabajar con datos cuyo tamaño sea múltiplo de 16 bits por lo que queda a discreción de los implementadores el coordinar la naturaleza e interpretación correcta de los datos.

Tipo de datos soportados

El servidor Modbus funciona correctamente con los siguientes tipos de datos

- Entero con signo de 16 bits
- Entero sin signo de 16 bits
- Entero con signo de 32 bits
- Entero sin signo de 32 bits
- Números reales en coma flotante de 32 bits

Operaciones soportadas

Estos registros permiten tanto las operaciones de lectura como de escritura. En nuestro caso, el robot realizará básicamente operaciones de lectura para poder obtener los datos generados por el sistema de reconocimiento de imágenes que a su vez realizará operaciones de escritura en el Modbus Server.

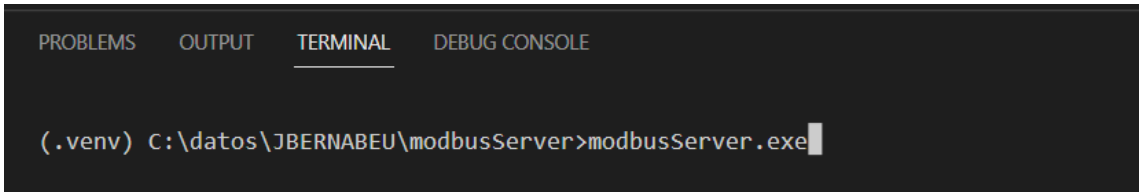
Configuración

El servidor Modbus funciona sobre el protocolo TCP/IP de internet pudiendo interactuar con él en el puerto TCP número 502. Tanto la dirección de internet como el puerto de comunicaciones es parametrizable pudiéndose ajustar la configuración efectuando los

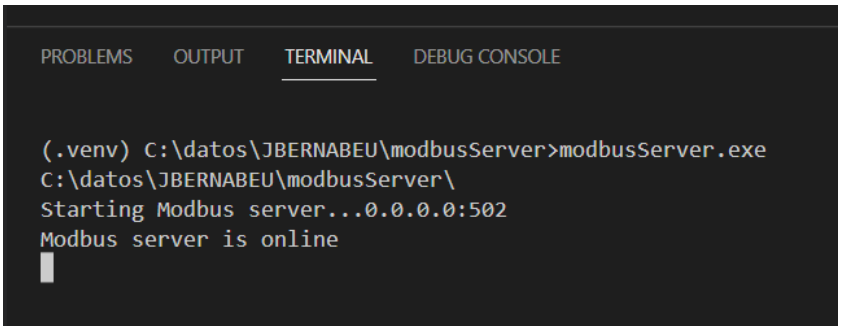
cambios pertinentes en el fichero de configuración que a tal efecto se sitúa en la carpeta "configuracion/configuracion.json"

### 3.3.2 Funcionamiento del servidor.

Para iniciar el servidor y conseguir que se quede a la escucha de las peticiones de lectura/escritura realizadas por los clientes Modbus se necesita ejecutar el siguiente comando realizado desde un terminal de comandos



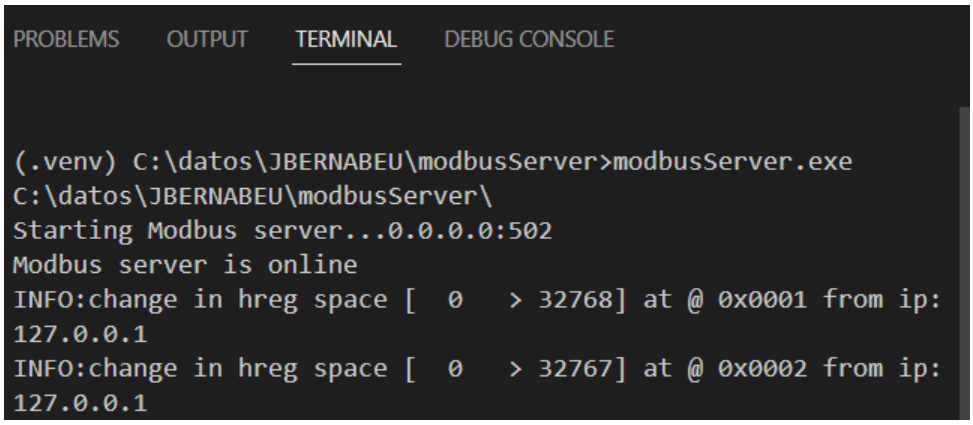

Cuando el servidor se inicializa correctamente se produce el siguiente resultado por pantalla




A partir de este momento ya se pueden realizar peticiones. Con el fin de permitir el seguimiento del intercambio de información el servidor indicará para cada petición, la dirección desde la que se realiza, la dirección de memoria afectada y los valores numéricos afectados.

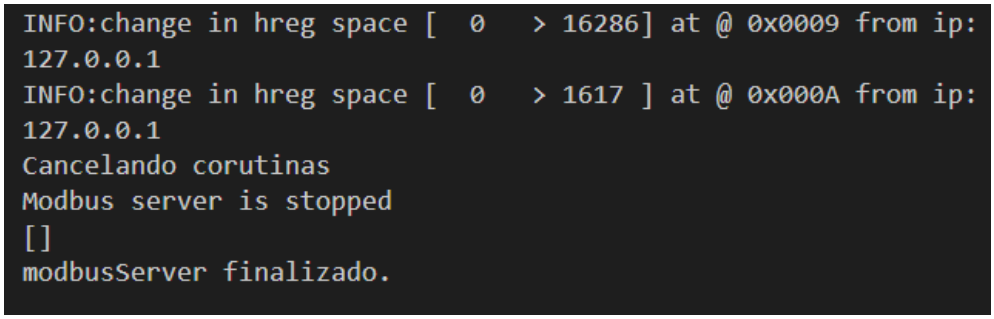
Inicialmente el banco de memoria de los registros se encuentra inicializado a 0.

En el ejemplo siguiente se observa como el registro con dirección 1, y el registro con dirección 2 cambia de valor, del 0 inicial al 32768 y 32767 siendo el cambio efectuado desde el cliente Modbus con dirección IP de internet 127.0.0.1.




Para finalizar la ejecución del servidor Modbus, hay que situarse en la terminal de comandos desde la que se inicio su ejecución y pulsar la combinación de teclas CTRL + C para finalizar su ejecución.

El sistema nos informará de la finalización de la ejecución de este con los mensajes indicados en la pantalla siguiente.




### 3.3.3 Pruebas realizadas con el servidor MODBUS

Pruebas de funcionamiento del MODBUS Server

Se comprueba para cada tipo de datos que soporta nuestro Modbus Server que tras una operación de escritura en un registro dado el valor leído se corresponde con el escrito y que durante las operaciones de lectura y escritura los datos siempre se interpretan correctamente.

```
(.venv) C:\datos\JBERNABEU\modbusServer>python modbusClient.p
y
Valores actuales en modbus server:  [8, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0]

Test signed short int 16
Tesk Ok -32768 -32768
Tesk Ok 32767 32767

Test unsigned short int 16
Tesk Ok 65535 65535

Test signed int 32
Tesk Ok -2147483648 -2147483648
Tesk Ok 2147483647 2147483647

Test unsigned int 32
Tesk Ok 4294967295 4294967295

Test numeros float
Test OK 1.2345678 1.2345677614212036
Valores actuales en modbus server:  [12, 32768, 32767, 65535,
 32768, 32767, 65535, 65535, 65535, 16286, 1617, 0]
```

Siendo el orden de las peticiones recibidas por el servidor el siguiente

```
(.venv) C:\datos\JBERNABEU\modbusServer>modbusServer.exe
C:\datos\JBERNABEU\modbusServer\
Starting Modbus server...0.0.0.0:502
Modbus server is online
INFO:change in hreg space [  0   > 32768] at @ 0x0001 from ip:
127.0.0.1
INFO:change in hreg space [  0   > 32767] at @ 0x0002 from ip:
127.0.0.1
INFO:change in hreg space [  0   > 65535] at @ 0x0003 from ip:
127.0.0.1
INFO:change in hreg space [  0   > 32768] at @ 0x0004 from ip:
127.0.0.1
INFO:change in hreg space [  0   > 32767] at @ 0x0005 from ip:
127.0.0.1
INFO:change in hreg space [  0   > 65535] at @ 0x0006 from ip:
127.0.0.1
INFO:change in hreg space [  0   > 65535] at @ 0x0007 from ip:
127.0.0.1
INFO:change in hreg space [  0   > 65535] at @ 0x0008 from ip:
127.0.0.1
INFO:change in hreg space [  0   > 16286] at @ 0x0009 from ip:
127.0.0.1
INFO:change in hreg space [  0   > 1617 ] at @ 0x000A from ip:
127.0.0.1
```

Como se puede observar todas las pruebas efectuadas tienen un resultado satisfactorio, pudiendo por lo tanto afirmar que el Modbus Server funciona correctamente para los tipos de datos especificados.

*Observaciones*. En el caso de tipos de datos de 32 bits las operaciones de lectura y escritura del test han tenido que seguir unas especificaciones comunes que no están definidas por el estándar Modbus.

Prueba de comunicación Modbus Server <-> Controladora Robot.

Configuración del robot para interactuar con el Modbus Server.

Los parámetros que configurar son:

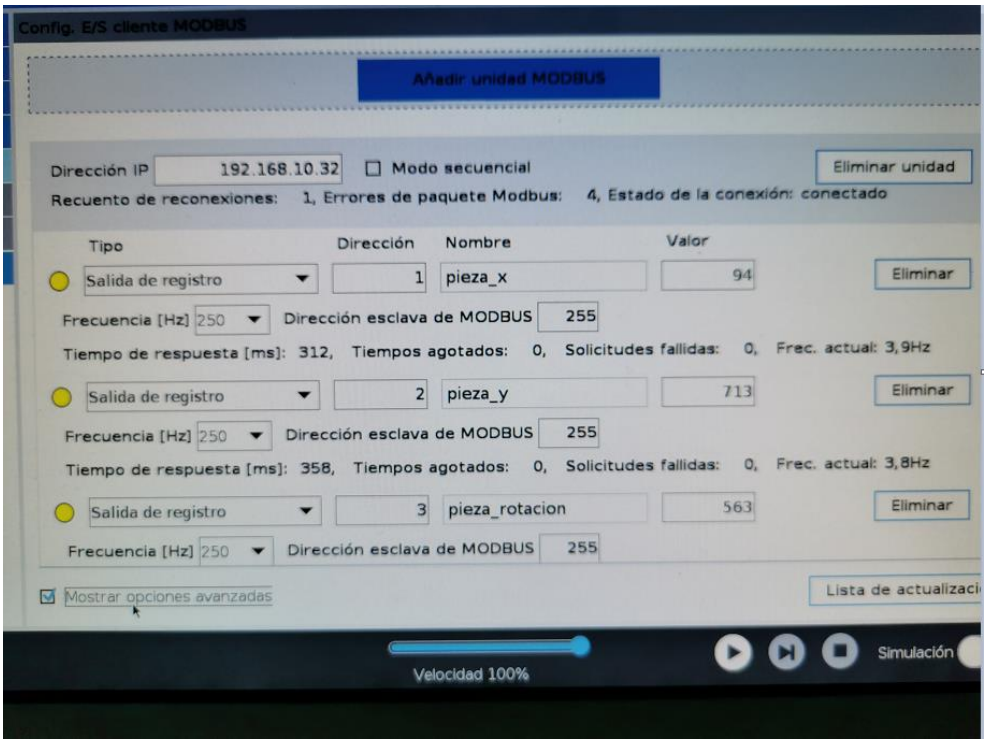
Dirección IP del equipo donde el Modbus server se encuentra a la escucha.

Tipo de datos. El protocolo Modbus tiene varios tipos de datos predefinidos, en este caso hay que especificar que los datos son de tipo registro para permitir el intercambio de valores numéricos de 16 bit.

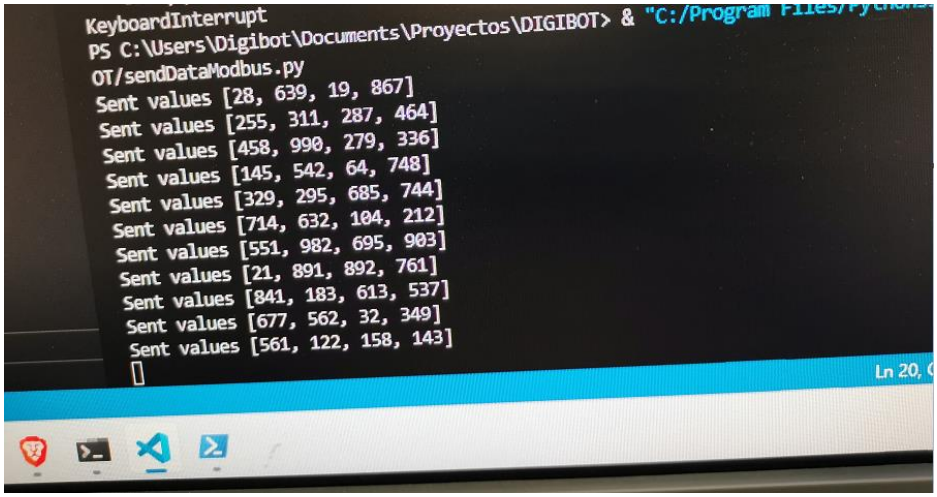
Dirección: dirección del registro en el banco de memoria del Modbus server.

Frecuencia: Frecuencia de actualización de los datos. Es decir, frecuencia con la que la controladora del robot accederá al Modbus server para refrescar el dato.

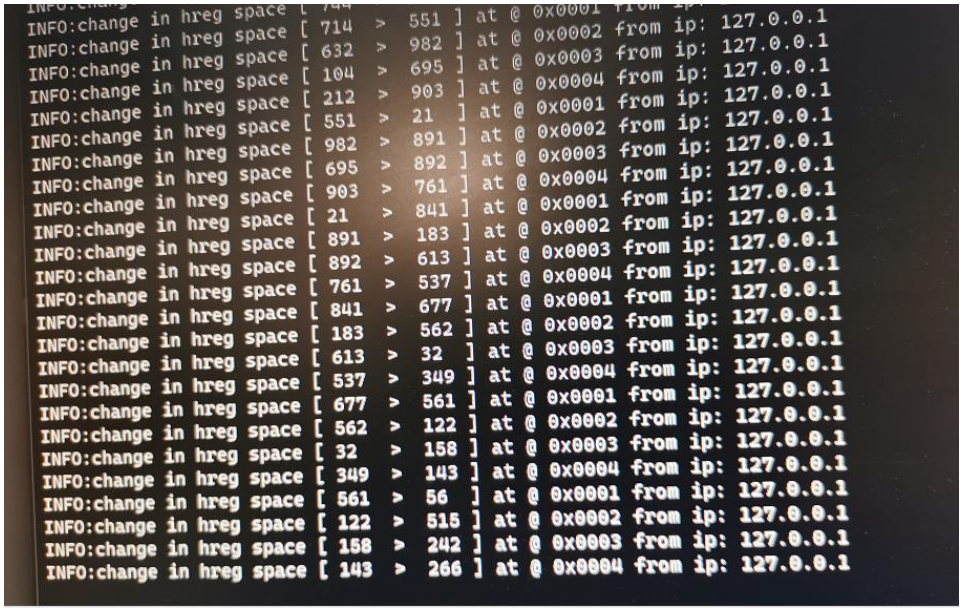
Dirección esclava de Modbus. En nuestro caso es una constante con valor 255.



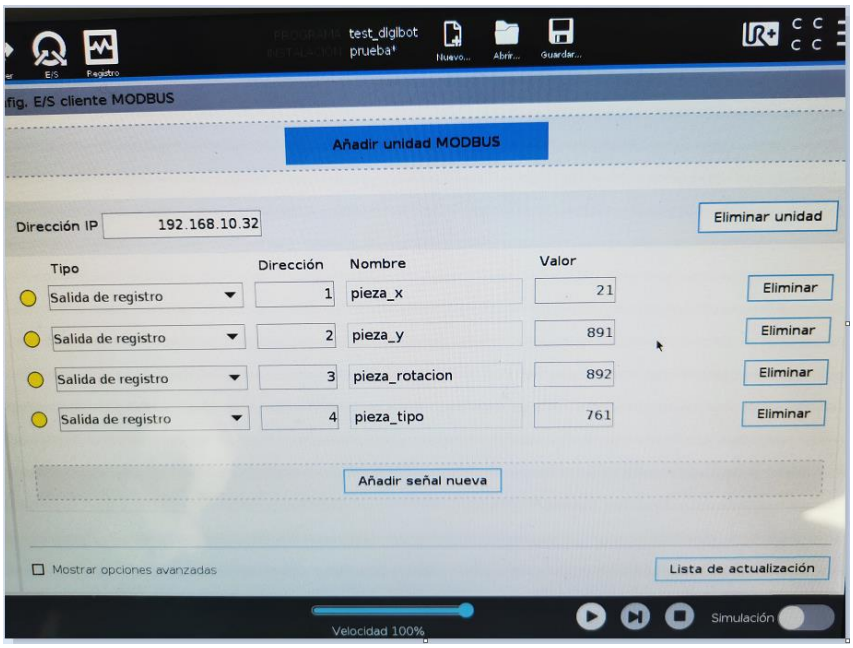
Tras configurar las comunicaciones de la controladora del robot con el Modbus Server, procedemos a enviar datos desde el sistema de reconocimiento de imagen al Modbus Server

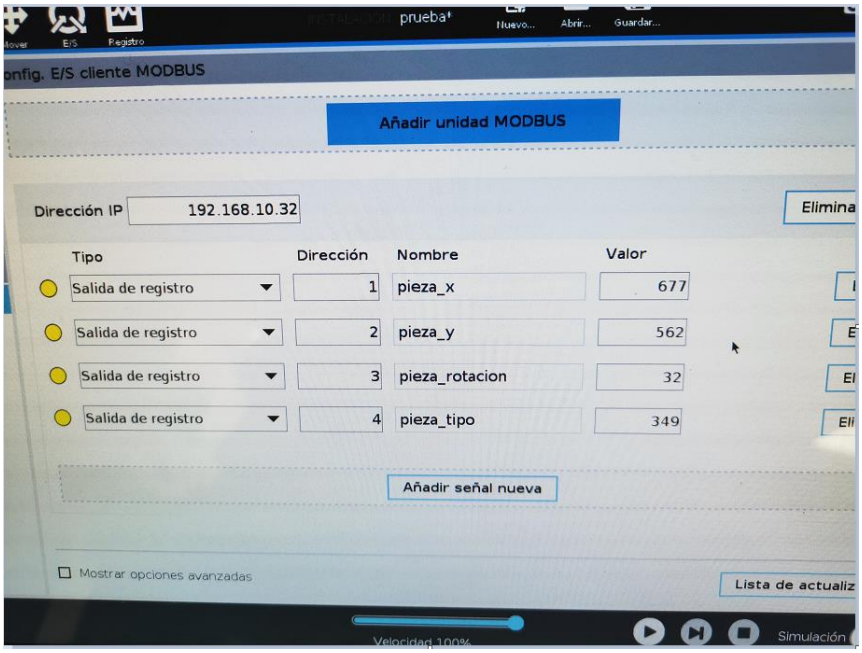



Comprobamos que los datos se escriben correctamente en el Modbus Server




Y por último, comprobamos que la controladora del robot es capaz de acceder a los datos y actualizar los mismos.




El resultado de esta prueba también es satisfactorio por lo que podemos afirmar que el robot puede interactuar correctamente con el servidor Modbus pudiendo enviar y recibir datos desde o hacia terceras partes que en el caso del proyecto que nos ocupa es el sistema de reconocimiento de imágenes.

### 3.3.4 Programación del robot colaborativo UR 16e

#### *3.3.4.1 Configuración de la Instalación*

Para la programación del robot se ha establecido en la parte de comunicaciones un protocolo Modbus, ya verificado según se indica en los puntos anteriores, mediante un servidor que envía al robot una serie de registros con los datos obtenidos por el sistema de cámara de visión artificial y la RRNN.

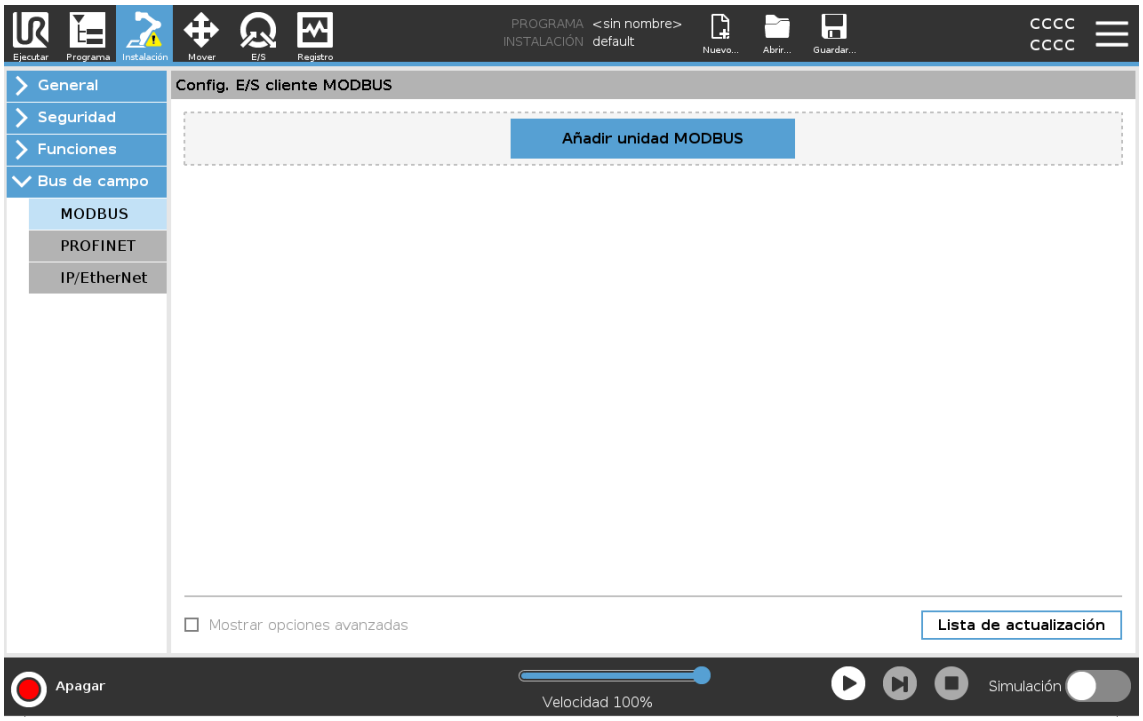
Estos datos son la coordenada X, la coordenada Y y la Rotación del punto de referencia del objeto reconocido por la RRN. Además del tipo de pieza que ha reconocido, este último con la correspondencia siguiente:

- 0 - Conector I
- 1 - Conector T
- 2 –
- 3 - Pata
- 4 - Rueda 1
- 5 - Rueda 2

El valor 2 no corresponde a ningún objeto ni se enviará dato alguno, porque este es el código reservado para piezas desconocidas.

Para el funcionamiento de las comunicaciones con el robot, hay que realizar previamente en el mismo la configuración de IP´s, así como los registros y su tipo.

Para ello en la pantalla de Instalación en el sistema Polyscope del robot UR, hay que crearlos y configurarlos.




Se configura la Dirección IP del servidor Modbus y seguidamente se añaden los registros necesarios.


En este caso se han creado las salidas de registro:

- **Tipopieza** para el dato de tipo de pieza

- **Ref_X** para el dato de coordenada X
- **Ref_y** para el dato de coordenada Y
- **Rotación** para el dato de Rotación

Para que los datos se envíen y se reciban a petición, se incluye un registro **Piezaenviada** y otro con **PedirPieza** con valores 0 y 1 que enviarán al servidor Modbus a fin de indicar si este puede mandar datos o no.

#### *3.3.4.2 Programa Robot.*

Para la programación del robot se ha empleado la programación "nativa" del sistema Polyscope de Universal Robots, generando un árbol de programación con los comandos necesarios. Además se ha incluido en dicha programación una parte de programación mediante script, empleando comandos de programación URScript.

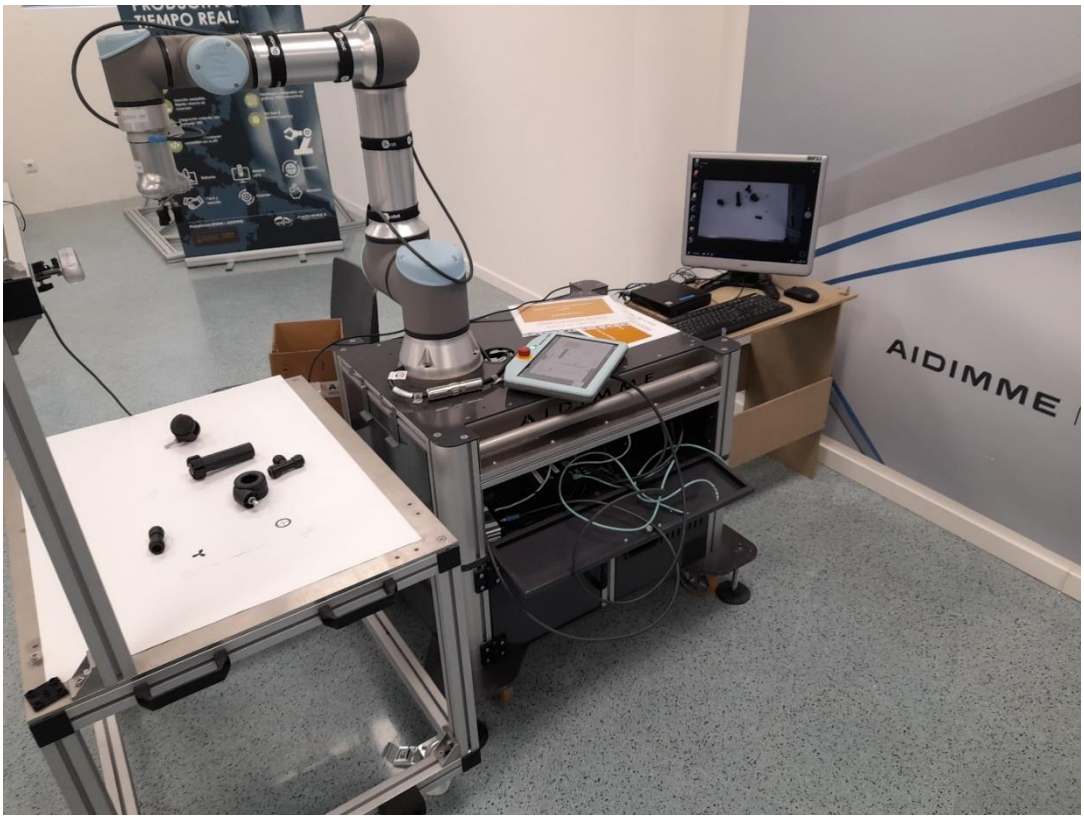
Su vista en la consola portátil del robot o pendant se muestra a continuación:


En los script se ha programado la construcción de las coordenadas del punto de referencia enviado en los distintos registros Modbus, obteniendo el punto al que debe moverse el brazo robot.

### *3.3.4.3 Pruebas realizadas*

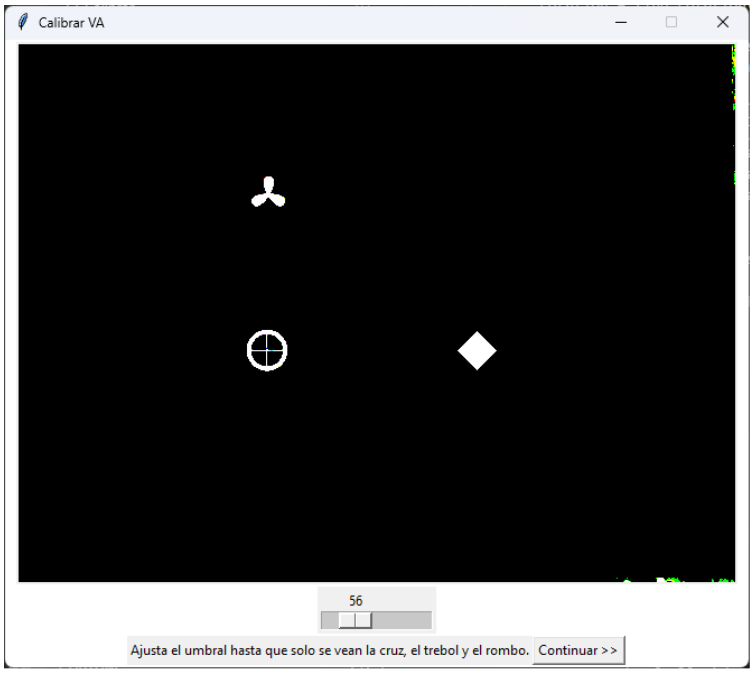
Con la programación realizada, la acción que ejecuta el robot tras recibir del sistema de visión la identificación de la pieza, su posición y ángulo, es posicionar la garra sobre la pieza seleccionada, recogiéndola y llevándola al punto deseado. Se han realizado pruebas de identificación y recogida de piezas con el UR 16e, resultando exitosas todas ellas.

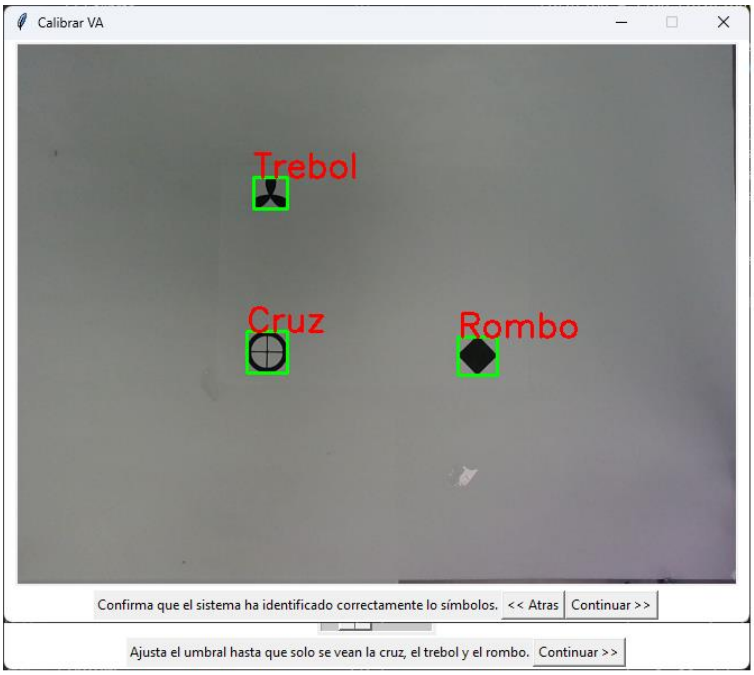
Seguidamente se muestran algunas imágenes de estas pruebas, realizadas con la instalación de la imagen siguiente.

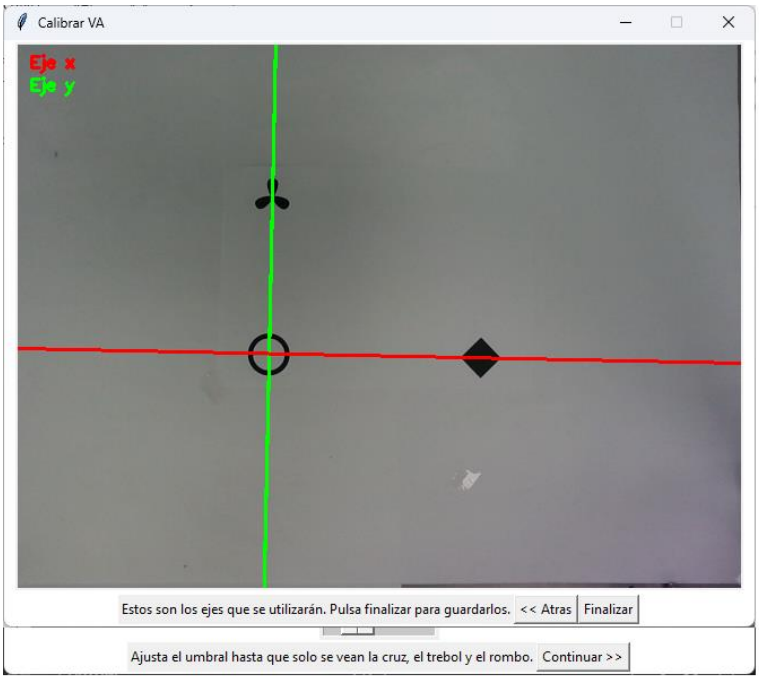



*Instalación del robot, cámara, controlador y piezas*

***Calibración del sistema para fijar los ejes de referencia en el campo de visión del robot***


Calibrar VA
56
Ajusta el umbral hasta que solo se vean la cruz, el trebol y el rombo.
Continuar >>


Calibrar VA
Trebol
Cruz
Rombo
Confirma que el sistema ha identificado correctamente lo símbolos.
<< Atras
Continuar >>
Ajusta el umbral hasta que solo se vean la cruz, el trebol y el rombo.
Continuar >>




*Identificación y recogida de piezas con el robot*