GRADO EN INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

Curso académico 2016-2017

SISTEMA PARA EL EMPAQUETADO AUTOMÁTICO DE JUGUETES DE PIEZAS ENCAJABLES

TRABAJO FINAL DE GRADO

AUTOR:

D. Miguel Lloret Pompa

TUTOR:

D. Carlos Ricolfe Viala

Tabla de contenidos

MEMORIA

GRADO EN INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

Curso académico 2016-2017

DISEÑO DE UN SISTEMA ROBÓTICO PARA EL EMPAQUETADO AUTOMÁTICO DE JUGUETES DE PIEZAS ENCAJABLES

Resumen

El objetivo es diseñar e implementar un sistema de empaquetado automático de piezas con distintas formas y colores dentro de un molde. El sistema será capaz de detectar las piezas que llegan por una cinta transportadora y las colocará en la posición correspondiente del molde. En caso de que las piezas de un determinado tipo estén completadas dentro del molde, el sistema las depositará en un buffer para rellenar el siguiente molde.

Índice

1. Introducción

En este proyecto, se quiere que varias piezas con diferentes formas (cuadrados, círculos y pentágonos) y colores (azul, rojo, verde y negro) se depositen en su correspondiente sitio, dentro de un molde, de acuerdo a las dos características anteriores.

De acuerdo a estos datos, el problema se podría abarcar de diferentes formas.

La primera de ellas, es programar un brazo robótico industrial para que recoja las piezas de una cinta transportadora y las deposite en el lugar correcto. Este sistema, sería factible siempre y cuando las piezas llegaran al lugar de recogida en el mismo orden y, estuvieran orientadas con el mismo ángulo en el que se van a depositar. De la misma forma, el molde con los huecos para las piezas debería estar en el mismo lugar y con la misma orientación siempre.

La segunda, es utilizar un sistema de visión artificial el cual reconozca el color y forma de la pieza, le envíe la posición de esta al brazo robot y haga lo mismo con los lugares de depositado del molde. A este sistema no le afectaría ni el orden de llegada de las piezas ni la orientación de estas, puesto que sería capaz de reconocer las características de la pieza y su ángulo respecto al del sitio correspondiente.

Por todo lo dicho anteriormente, se ha decidido crear una aplicación para un sistema de visión industrial que identifique el color y la forma, tanto de las piezas, cuando lleguen al final de la cinta transportadora, como de los huecos del molde. Además, se usará un brazo robot para que recoja dichas piezas y las coloque en su correspondiente sitio.

1.1. Elección del robot y el equipo de visón

1.1.1. Elección del robot

Existen diferentes tipos de brazos robóticos:

- Cartesiano. Este es un robot cuyos ejes son lineales y perpendiculares entre sí.

Figura 1.Robot cartesiano.

- SCARA. Este es un robot de 4 grados de libertad. Está formado por dos articulaciones de rotación y otra prismática. Son caracterizados por tener gran velocidad y ser buenos para tareas repetitivas.

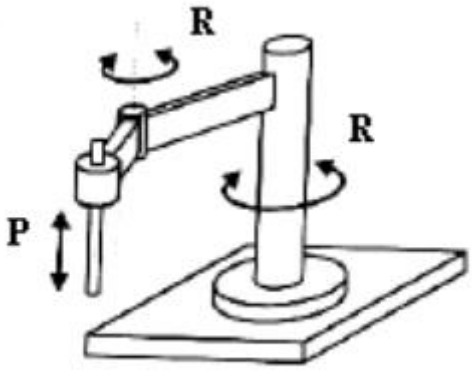


Figura 2.Esquema de robot SCARA.

- Articulado. Este es un robot de 6 grados de libertad debido a que está formado por tres articulaciones de rotación. A ellos suele ir unidos una muñeca formada por otras tres articulaciones del mismo tipo. Son los más comunes en las industrias dado a su gran versatilidad.

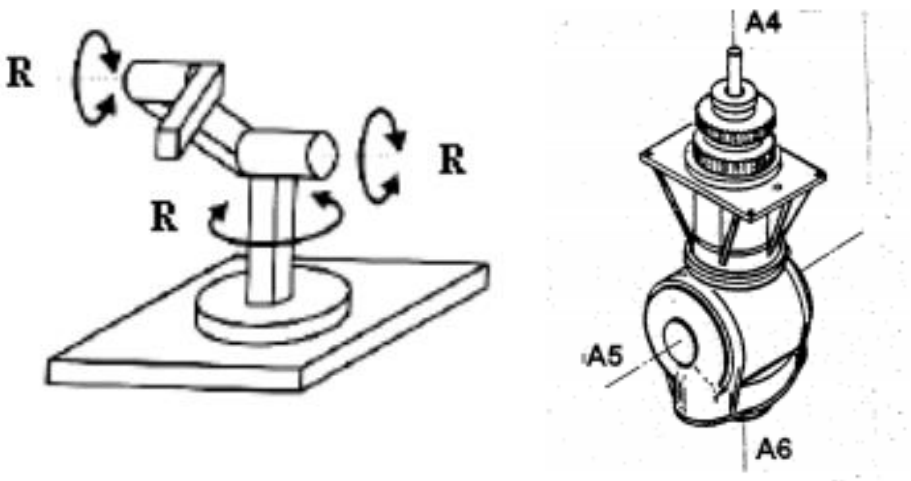


Figura 3. Izquierda: Esquema de robot articulado. Derecha: esquema de muñeca en línea

Para este proyecto se ha optado por un robot articulado, descartando el cartesiano por ser muy aparatoso además de poseer sólo 3 grados de libertad. El robot SCARA también era una buena opción para este trabajo pero también se ha descartado porque, debido a que las piezas a manipular pueden llegar en dos orientaciones diferentes, es más fácil trabajar con un robot articulado. Esto es, gracias a su muñeca en línea la cual permite hacer rotaciones sobre el mismo eje que esta perpendicular a la pieza (A6 en la figura 3).

Una vez decidido el tipo de robot que se va a utilizar, se tiene que decidir que marca usar. Existen varias empresas de fabricación y comercialización de gran renombre como KUKA, ABB, Fanuc, Motoman, etc.

De entre todos estos, Motoman y ABB son los que mejores productos ofertan de acuerdo a nuestras necesidades:

- Alcance entre los 700 y 900 mm.
- Carga útil menor de 6 kg.
- 6 ejes.

Por parte de Motoman, un ejemplo de posible robot industrial es el MH5LS II/MH5LF puesto que tiene 6 ejes controlados, una carga útil de 5kg y un alcance máximo de 895mm.

Figura 4.Robot industrial MH5LS II/MH5LF de la marca Motoman.

Por otra parte, ABB ofrece el IRB 400 un robot de 6 ejes controlables, con carga útil de 5kg y un alcance máximo de 810mm.

Figura 5.Robot industrial IRB 400 de la marca ABB.

Se ha decidido utilizar el IRB400 debido a que, tiene un alcance menor que el ofrecido por Motoman. La explicación de esto es que al no hacer falta un gran alcance, con el del IRB es suficiente y al ser menor, significa que el robot es más pequeño. Otro factor más importante es que estamos más familiarizados con el lenguaje de programación usado por la empresa ABB, el lenguaje RAPID.

1.1.2. Elección del sistema de visión

La cámara empleada para este proyecto es la CV-M77 de la marca JAI. Para la elección de esta, nos hemos apoyado en la página web de INFAIMON. Esta empresa es la líder en España en sistemas de visión artificial.

La web de esta compañía ofrece un buscador de productos en el cual se pueden aplicar filtros que hacen más fácil encontrar los productos que se adecuen a las necesidades del comprador.

Para este proyecto se quiere analizar dos zonas en una misma área. Por ese motivo, se necesita una cámara matricial, es decir, aquellas en las cuales su sensor cubre un área. Además, dado que una de las características a reconocer de las piezas es su color, es imprescindible que la cámara capture imágenes en el espectro de color.

Por otra parte, en anteriores trabajos ya se ha trabajado con la marca JAI y los productos que ofrece se han comprobado ser de gran calidad.

2. Breve historia de la robótica y de la visión artificial.

Los brazos robots y los equipos de visión artificial son muy comunes en gran cantidad de industrias, tanto en conjunto como por separado, debido a sus diversas utilidades. Pero, para llegar a tal grado de integración, se ha recorrido un largo camino a través de la historia.

2.1. Historia de la robótica.

Desde hace miles de años, el hombre se ha obsesionado en dar vida a seres artificiales que realicen sus tareas repetitivas, pesadas o difíciles de realizar por un ser humano.

Debido a esto, durante siglos los seres humanos han inventado máquinas que imiten las partes del cuerpo. Un ejemplo son los antiguos egipcios los cuales unieron brazos mecánicos, operados por sacerdotes, a las estatuas de sus dioses. Estos sacerdotes atribuían que el movimiento de los brazos era debido a sus dioses. Por otro lado, los griegos construyeron estatuas las cuales movían mediante sistemas hidráulicos.

Durante los siglos XVII y XVIII se construyeron muñecos mecánicos con algunas características de robots pero diseñados con el propósito de divertir.

En 1805, Henri Maillardert construyó una muñeca mecánica capaz de escribir y hacer dibujos. Esta funcionaba mediante una serie de levas las cuales servían de "programa" para el proceso de escritura y dibujo.

Fue en 1917 cuando la palabra robot se empleó por primera vez en una obra de teatro llamada "Los Robots Universales de Rossum" del dramaturgo checo Karel Capek. Esta proviene de la palabra checa '*robota'* la cual significa servidumbre o trabajador forzado.

Más tarde, en 1950, Isaac Asimov publica el libro *Yo Robot* donde escribió las tres leyes de la robótica:

- Un robot no puede dañar a un ser humano o permanecer pasivo si este puede sufrir daño.
- El robot debe obedecer las órdenes dadas por un ser humano salvo que estas entren en conflicto con la primera ley.
- El robot debe proteger su propia existencia salvo que entre en conflicto con las dos leyes anteriores.

A Asimov también se le atribuye el inventar el término robótica.

En la década de los 50, fueron varios los factores intervinieron para el desarrollo de los primeros robots. Los más importantes fueron el desarrollo tecnológico y la investigación en inteligencia artificial, dado que se desarrollaron maneras de imitar el procesamiento de información de los humanos con computadoras electrónicas y se inventaros múltiples mecanismos para probar diversas teorías.

En 1946 aparecieron las primeras patentes con los primitivos robots para el traslado de maquinaria de Devol. A su vez, en ese mismo año, apareció la primera computadora electrónica "ENIAC", la cual estaba construida a base de válvulas.

En 1954 el ingeniero George Devol patenta el primer robot programable. Dos años más tarde, junto con el negociante Josef Engelberger, crearían Unimation, la primera empresa dedicada a la robótica.

En 1960 comienza el desarrollo de Unimate Robot Systems y, un año más tarde, el primer robot de Unimate se instala en una planta de General Motors para la manipulación de material en una máquina de fundición de troquel.

En 1971 se desarrolló en la Standford University el *Standford Arm,* un pequeño brazo robótico de accionamiento eléctrico.

En 1978 Unimation desarrolla el PUMA (Programmable Universal Machine for Assembly). Un brazo robótico industrial desarrollado para General Motors.

En los años 80 aparecieron los primeros ordenadores empotrados y se hicieron numerosos avances en la robótica móvil. A partir de entonces, la tecnología ha ido avanzando exponencialmente y, cada vez más rápido hasta el punto de que los robots forman parte de la gran mayoría de las industrias del mundo.

2.2. Historia de la visión artificial

La visión artificial o visión por computador tiene como finalidad extraer información del mundo real a partir de imágenes.

Actualmente, la visión artificial se usa en diversos campos como la medicina, la seguridad o la industria. Pero, ¿cuál fue su origen?

Desde un aspecto práctico, el inicio de esta fue en 1961 cuando el científico Lawrence G. Roberts creó un programa que era capaz de "ver" una estructura de bloques, analizar su contenido y reproducirla desde otra perspectiva.

A partir de ese momento, y, con el desarrollo tecnológico, las cámaras cada vez eran mejores, los procesadores más potentes y surgieron programas más avanzados capaces de implementar la visión artificial en áreas diferentes.

3. Antecedentes

Ya sea en conjunto o por separado, los robots y los equipos de visión están presentes en la mayoría de las industrias modernas y, de diferentes áreas comerciales tales como la automovilística, la alimenticia o la farmacéutica.

En el caso de industrias como la electrónica se usa la visión por computador para verificar el posicionamiento de diferentes componentes electrónicos, la identificación de las resistencias según su valor o la alineación de los pines de ciertos componentes.

En industrias alimenticias es frecuente ver estos equipos en la inspección de embalajes, en control de calidad de diferentes alimentos o verificando de las etiquetas de los productos sean correctas o estén colocadas adecuadamente.

Un ejemplo en la industria del automóvil es el uso de brazos robots para pintar las carrocerías de los coches, el uso de cámaras de visión para detectar defectos en las piezas de los vehículos o el uso de ambos equipos para la inspección de los cordones de soldadura al unir las diferentes piezas.

Por otra parte, los brazos robot también se pueden encontrar desempeñando rutinas de soldadura punto a punto, por arco o por láser. Así como tareas de corte o mecanizado de piezas. Sin olvidar la manipulación o el embalaje de diferentes productos, desde pequeños componentes electrónicos hasta carrocerías de vehículos.

3.1. ¿Por qué automatizar un proceso?

Como se ha visto anteriormente, cada vez hay más industrias que optan por automatizar procesos que antes eran desempeñados por humanos. Esto es debido a que los robots, al contrario que los humanos, no enferman, no se cogen bajas por maternidad/paternidad ni por depresión. Tampoco rinden peor o mejor dependiendo de su estado emocional, no se cansan y sólo necesitan una persona que se encargue de su mantenimiento. Además su tiempo de vida es bastante largo, por lo que puede estar sin reemplazarse durante mucho tiempo.

Otra ventaja de los robots es que, bien programados, son capaces de hacer el trabajo con rapidez y con más precisión.

4. Materiales y programas empleados

El sistema para la adquisición y procesamiento de imágenes consta de 3 partes:

- Cámara, encargada de la adquisición de imágenes. Modelo JAI CV 77M

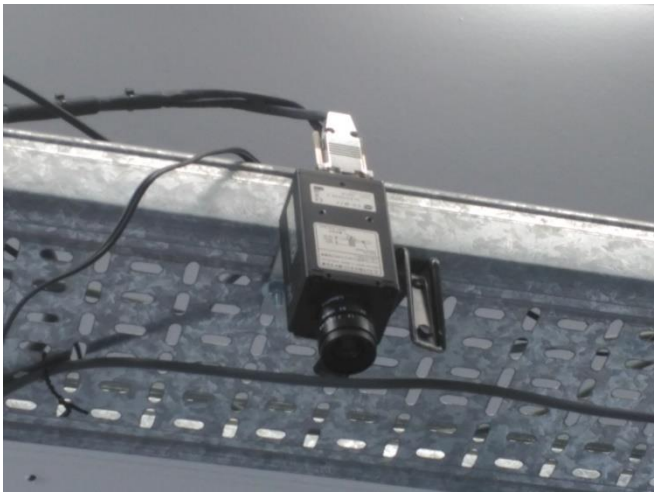


Figura 6. Cámara JAI CV 77M.

- Ordenador.

Figura 7. Ordenador.

- Sherlock 7. Programa informático usado para el procesamiento y análisis de imágenes.

Para la automatización del proceso se usaron:

- RobotStudio 6.03.01. Programa informático con el cual se programaron los movimientos del robot y se hicieron las simulaciones previas a las pruebas reales.

- Brazo robot ABB con ventosa. Encargado de recoger, mover y depositar las piezas.

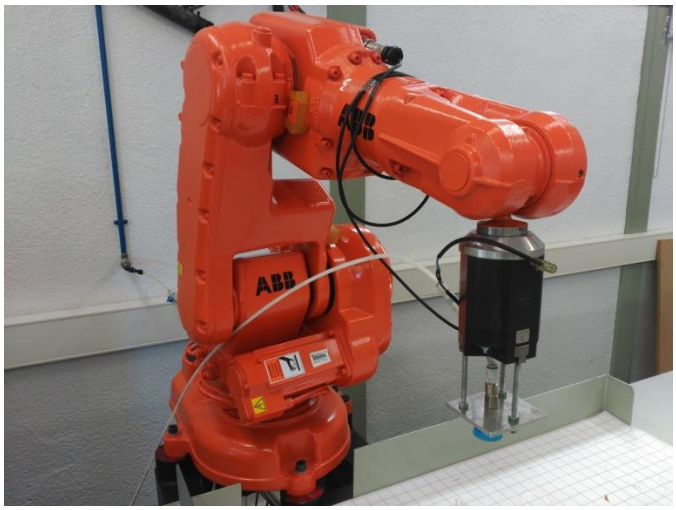


Figura 8. Robot industrial ABB.

Figura 9. Ventosa conectada al brazo robot.

- Cinta transportadora. Encargada de llevar las piezas hacia el robot.

Figura 10. Cinta transportadora.

- Piezas a manipular.

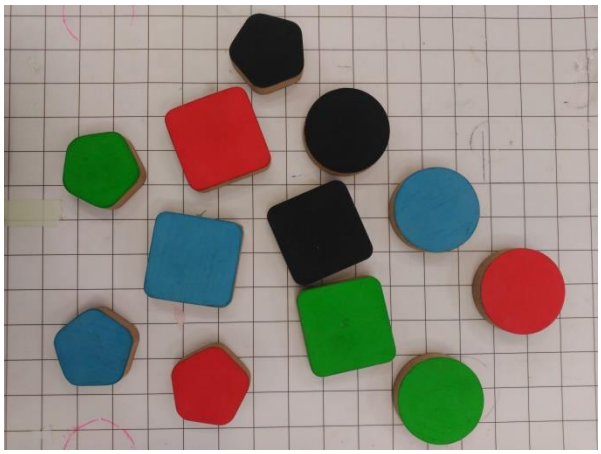


Figura 11. Piezas a manipular.

- Molde donde encajar las piezas.

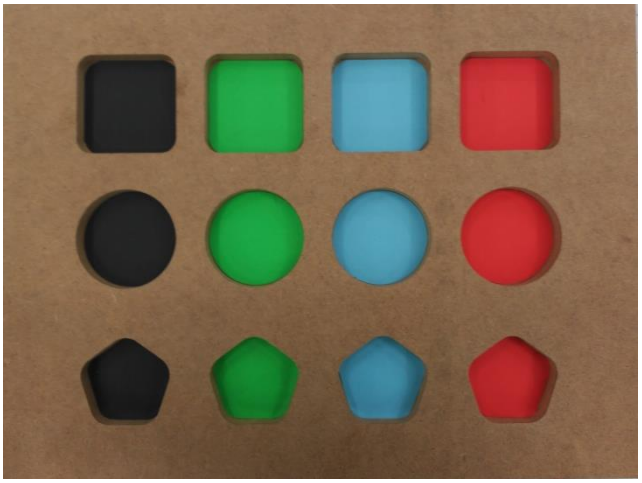


Figura 12. Molde.

- Flexpendant. Herramienta donde se puede mover y/o programar el robot. Además se pueden cargar y editar programas entre otras cosas.

Figura 13. Flexpendant.

5. Metodología empleada.

En primer lugar, para solucionar el problema planteado, se ha montado una representación de la estación de trabajo real. Es decir, se han colocado la cinta transportadora y el molde de forma que estén dentro del rango de movimiento del brazo robot.

5.1. Calibración de los equipos

Una vez montado el sistema, se procede al calibrado de los equipos. Esto es necesario debido a que la aplicación tiene que enviarle los datos al robot de forma que sean entendibles para este.

Para ello será necesario hacer la calibración tanto en el lugar de recogida de las piezas, como en el lugar de depositado de las mismas.

Se calibrará utilizando una matriz de transformación. Para obtenerla, se observarán varios puntos desde los dos sistemas de referencia, el del robot y el del sistema de visión. Con tal fin, se hará uso de una plantilla de calibración en la cinta y el molde donde irán las piezas se ha usado como plantilla para el lugar de recogida.

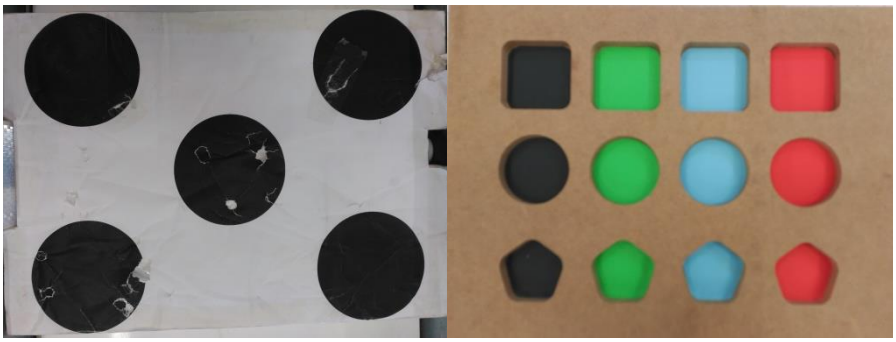


Figura 14. Izquierda: patrón de calibración en la cinta. Derecha: patrón de calibración en la mesa.

Para entender mejor el proceso a seguir, lo que se quiere es, a partir de los puntos de los patrones desde los dos sistemas de referencia, y mediante un algoritmo llamado homografía, una matriz que relacione dichos puntos.

Para eso, las plantillas necesitan ser colocadas en el lugar exacto en el cual las piezas serán recogidas y depositadas, tal y como se observa en la siguiente figura.

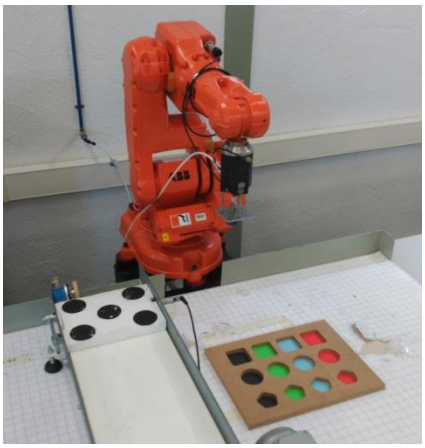


Figura 15. Sistema con los patrones en su correspondiente sitio.

Una vez hecho esto, se llevará al robot, manualmente, al centro de cada forma geométrica (Figura 11) y se obtiene la lectura de las coordenadas mediante la FlexPendant (Figura 12).

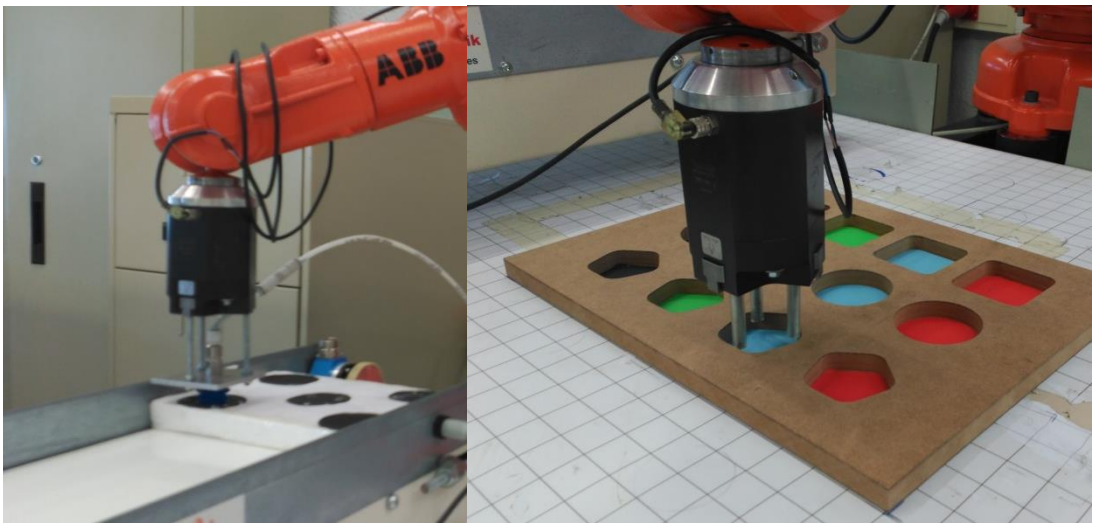


Figura 16. Izquierda: Robot calibrando en la cinta. Derecha: robot calibrando en el molde.

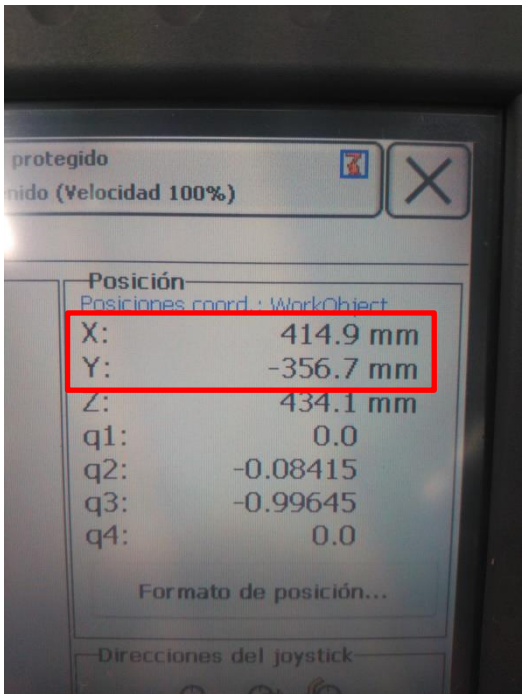


Figura 17. Posición del robot

Este proceso se repetirá para cada uno de los puntos de la cinta y del molde. Hay que tener en cuenta que conforme más puntos se cojan mejor, aunque con 5 en el caso de la cinta es suficiente. En el caso del molde, dado que hay 12 huecos sería un desperdicio no usarlos todos.

Una vez se tienen los puntos del robot, se procede a obtener dichos puntos desde el sistema de visión. Esto se hará mediante un análisis de la imagen y diferentes algoritmos, en los cuales no se entrará en detalle.

Cuando se obtengan estos puntos, se obtendrá la matriz de transformación con la ayuda de 2 programas de Matlab. En este caso, la matriz utilizada es la inversa de la matriz de transformación y, lo único que faltaría es obtener las coordenadas homogéneas y cartesianas a partir de las coordenadas del sistema de visión. En la siguiente imagen se pone un ejemplo de lo hecho en la aplicación para conseguirlo.

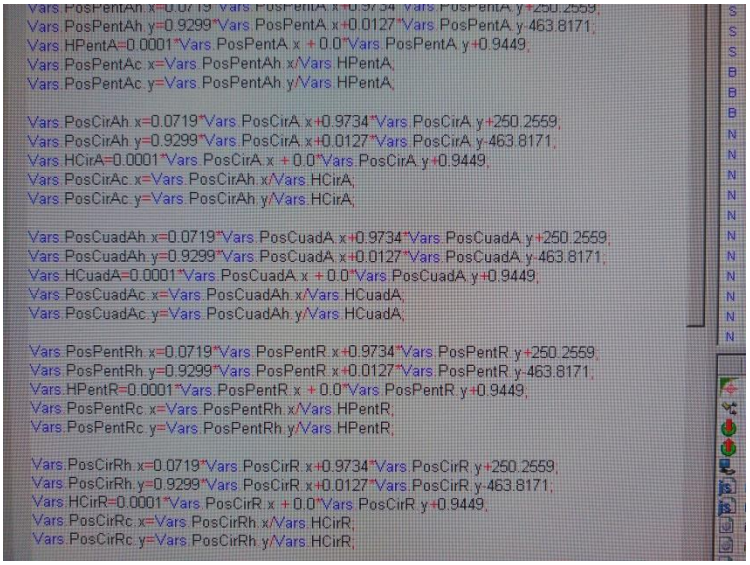


Figura 18.Paso a coordenadas cartesianas las coordenadas de las piezas

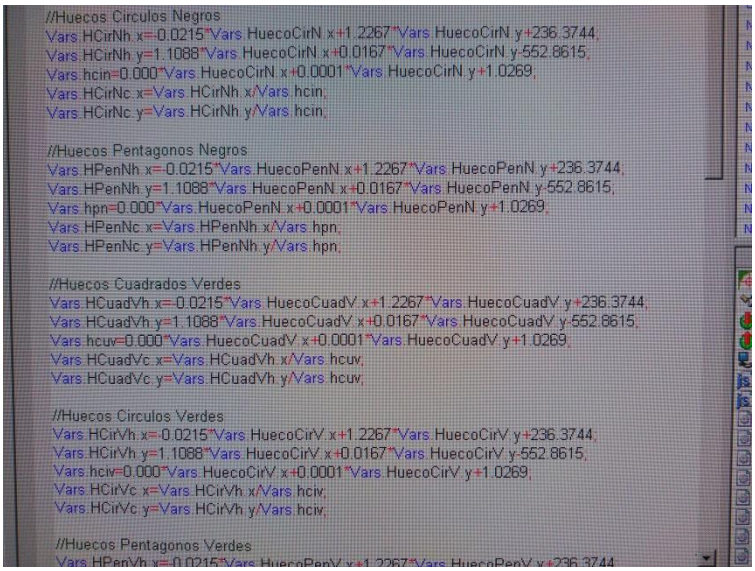


Figura 19. Paso a coordenadas cartesianas las coordenadas de los huecos.

Se ha decidido empezar por el diseño de la aplicación que será responsable de analizar las imágenes captadas por la cámara y, una vez terminada, se creará el código para el robot.

Para poder explicar cómo se ha diseñado dicha aplicación, se hará una pequeña introducción a la visión por computador y así hacer más fácil el entendimiento de los pasos seguidos.

5.2. Introducción a la visión artificial

Un sistema de visión está formado por dos tipos de componentes. El hardware, compuesto por el sistema de la adquisición (cámara), el ordenador y el sistema de visualización (pantalla). Por otra parte está el software, encargado de la adquisición, manipulación, pre procesamiento e interpretación de las imágenes.

Como se ha dicho anteriormente, los sistemas de visión son usados para crear aplicaciones capaces de analizar imágenes, ya sea en tiempo real o simples fotografías. Durante el desarrollo de estas aplicaciones se atraviesan diferentes etapas de diseño, las cuales se van a explicar a continuación.

En primer lugar, se necesita una **adquisición** de las imágenes que se quieren analizar. Seguidamente, si la imagen no es nítida, tiene demasiado brillo, es muy oscura, etc. existen técnicas de **preproceso** los cuales mejoran su calidad. Una vez se tiene una buena imagen, se pasa a la **segmentación**, esta etapa sirve para separar las partes de interés (los objetos) del resto (el fondo). A continuación, en la fase de **descripción**, se etiquetan los objetos con información para hacer más fácil su identificación y distinción. De entre toda la información usada, las características más discriminantes necesitan un proceso de aprendizaje. En la etapa de **reconocimiento**, se identifican o clasifican los objetos etiquetados previamente. Por último, se obtienen los resultados y se toman las decisiones necesarias.

En el siguiente apartado, se explicará cómo se ha diseñado la aplicación a través de las diferentes etapas presentadas anteriormente.

5.3. Diseño de la aplicación para el sistema de visión

En la etapa de adquisición, se ha usado una cámara JAI CV 77M anclada al techo de la sala a unos 2m de altura de la zona de trabajo. Esto es importante dado que si se colocara a una altura diferente la aplicación no funcionaría adecuadamente. Otro factor a tener en cuenta, es la iluminación de la estancia puesto que, al hacer un reconocimiento de color, una mayor o menor iluminación puede cambiar el tono del color y que el programa no lo reconozca. Esto se explicará con más detalle en el proceso de segmentación.

El objetivo de la aplicación es que esta pueda reconocer las formas cuadrado, círculo y pentágono, además de los colores azul, verde, rojo y negro. Se pensaron diferentes formas de abordar este objetivo.

La primera, fue hacer que la aplicación reconozca primero la geometría de las piezas y luego identificara el color de estas.

Para este caso, puesto que los algoritmos usados en el reconocimiento de formas no pueden trabajar con imágenes a color, se pasó la imagen a MONO8 (escala de grises). Esta sería la etapa de preprocesado.

A continuación, en la etapa de segmentación se tenía que separar las piezas del fondo. Para ello se usó una herramienta llamada *threshold* la cual se puede configurar para que, todos aquellos objetos que superen un nivel de gris elegido queden en negro y los que estén por debajo se vuelvan blancos.

Esta opción, presentaba el problema de que, cuando más claro era el color, menor nivel de gris tenía por lo que la configuración del *threshold* era muy baja y aparecían objetos que no eran de interés.

Otro problema era obtener el color a partir de la imagen en grises puesto que era en esta en la que se hacía el reconocimiento de forma.

Debido a esto, se descartó esta opción y se decidió reconocer los colores y luego identificar la geometría de las piezas.

En este caso no hace falta pre procesar la imagen. Por lo que se pasará a explicar las etapas de segmentación y descripción a la vez, dado que es más sencillo y entendible de explicar de esta forma.

Para el reconocimiento de color, se usa el algoritmo *Color Map*. Esta, es una herramienta de aprendizaje la cual, una vez se le ha "enseñado" los diferentes colores, les asigna a estos un valor en la escala de grises.

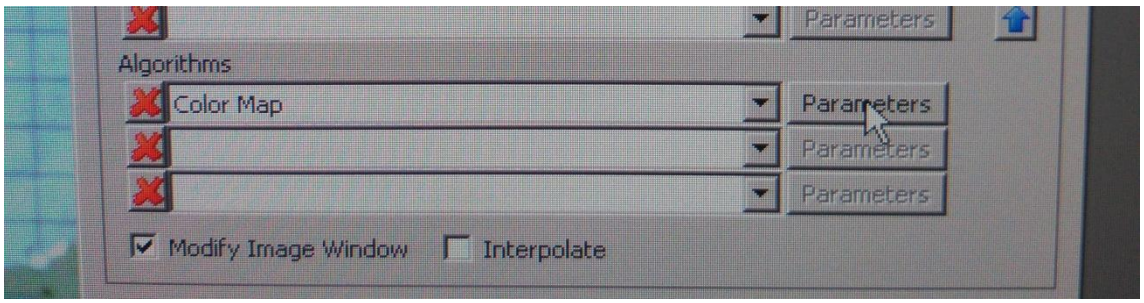


Figura 20. Algoritmo *Color Map.*

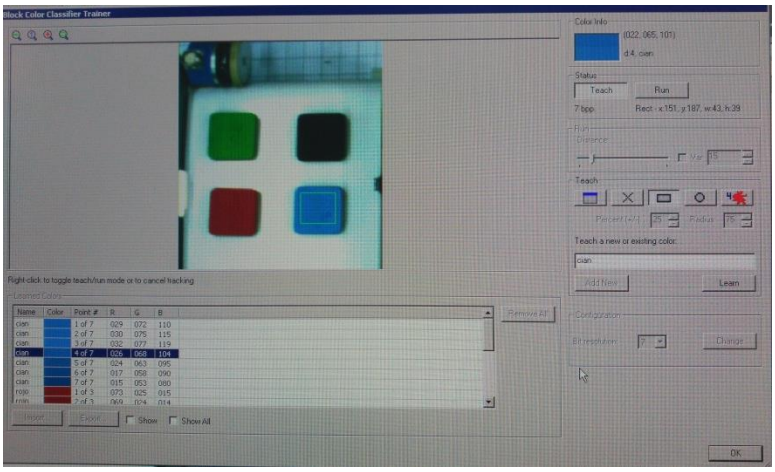


Figura 21. Entrenamiento de la herramienta.

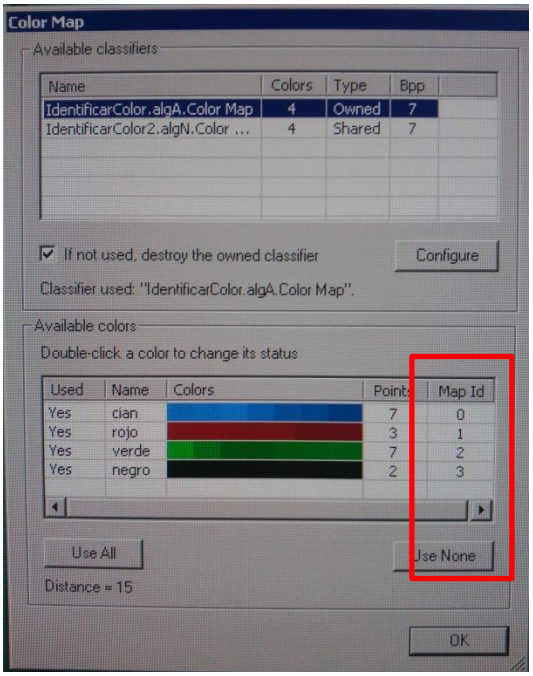


Figura 22. Valores de los colores en escala de grises

Es conveniente añadir que para la identificación de los colores se han usado varios puntos de cada color obtenidos con iluminaciones diferentes para hacer más robusta la aplicación.

Una vez el programa reconoce los colores, se abre una ventana para cada color a partir de la imagen obtenida por el *Color Map*.

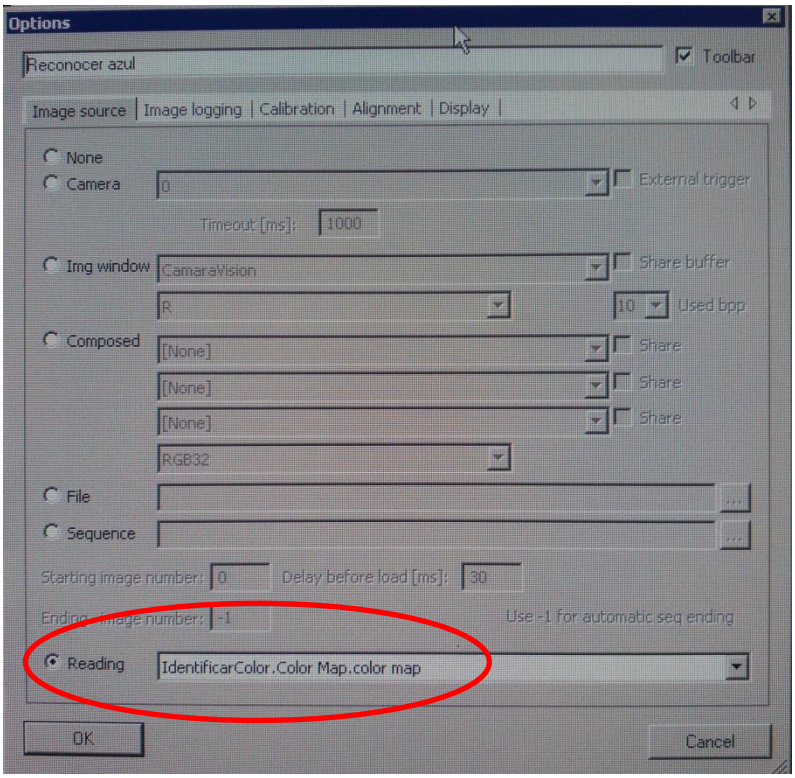


Figura 23. Elección de la imagen obtenida por el algoritmo *Color Map.*

Esta imagen está en escala de grises, por lo que se pueden usar los algoritmos correspondientes para la identificación de formas. Pero primero se debe segmentar la imagen, separando cada color del resto. Para ello se utiliza la herramienta *threshold band* la cual actúa como filtro pudiendo, el programador, elegir qué valor en la escala de grises quiere que aparezca en la imagen.

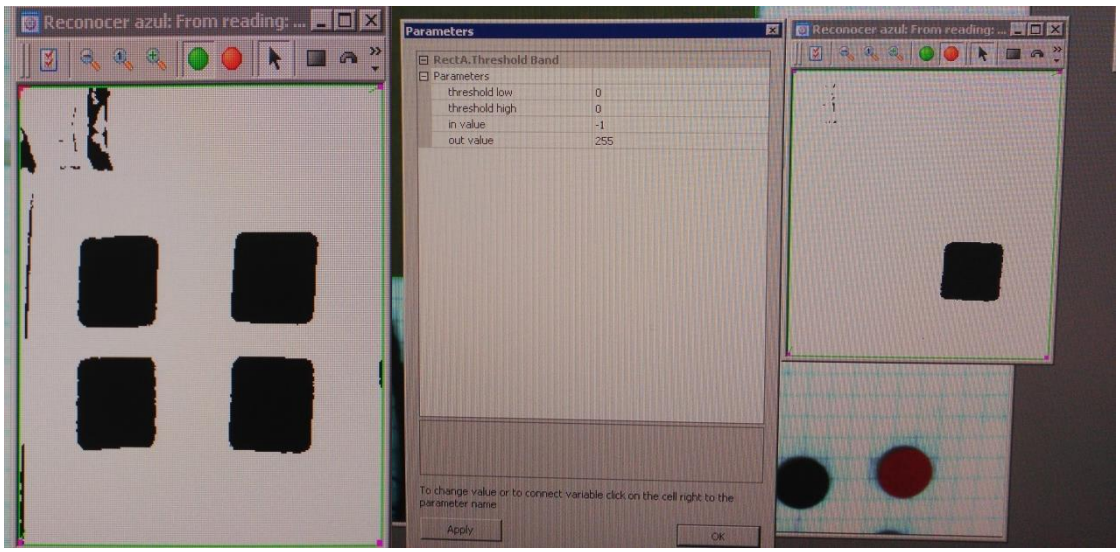


Figura 24. Izquierda: imagen sin *threshold band*. Derecha: *Threshold band* restringiendo para que sólo aparezca el color azul.

Una vez se han segmentado todas las imágenes para cada color, se procede a reconocer las formas de las piezas. Este también es un proceso de aprendizaje en el cual se le tiene que "enseñar" al programa las diferentes geometrías.

Para ello se usa el algoritmo *Search* – *Geometric* el cual aprende los bordes más prominentes dentro del área seleccionada.

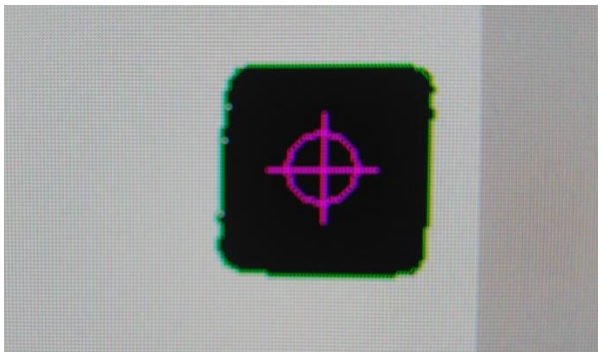


Figura 25. *Search* - *Geometric* reconociendo los bordes del cuadrado.

Además permite configurar varias opciones como el número mínimo de pixeles unidos que reconozca, el número de patrones a reconocer, etc.

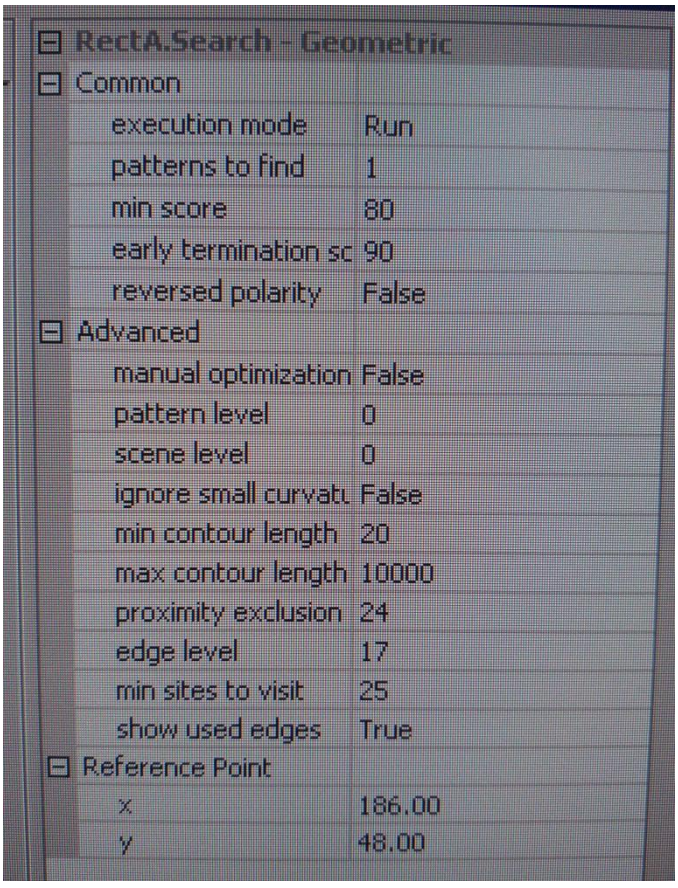


Figura 26. Opciones del *Search* - *Geometric.*

Este proceso de aprendizaje se hace para todas las formas geométricas de los diferentes colores. Con respecto a los huecos del molde donde se encajarían las piezas se seguiría el mismo proceso. Hecho esto, finalizarían las etapas de segmentación y descripción.

En la fase de reconocimiento, se crean las variables necesarias para obtener el máximo número de datos. Por ejemplo, una para que indique el número de cuadrados azules que se detectan, otra que referencie al ángulo de las piezas o del tablero y otra que guarde las coordenadas de la pieza que detecte, entre otras.

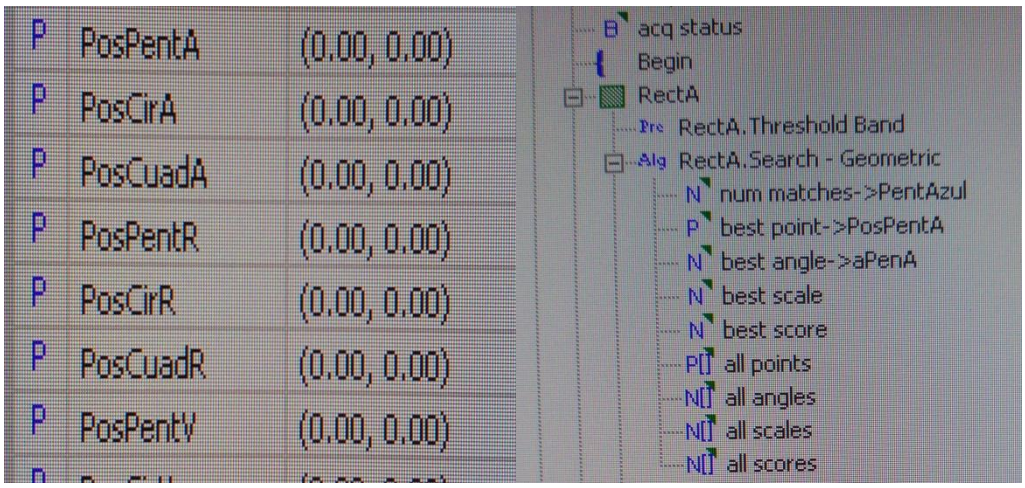


Figura 27. Izquierda: ejemplo de variables definidas. Derecha: ejemplo de asignación de variables.

Una vez terminadas estas etapas, se plantean varias cuestiones puesto que el sistema automático no consta sólo del subsistema de visión artificial. Estas son: ¿cómo sabe el robot donde coger la pieza?, ¿cómo sabe dónde depositarla?, ¿cómo sabe que pieza está cogiendo?; entre otras.

La respuesta a estas preguntas es que debe de haber una comunicación entre el robot, el cual espera recibir los datos necesarios, y la aplicación, responsable de enviar todos los datos al robot.

Los dispositivos utilizarán los protocolos TCP/IP mediante los cuales pueden formar conexiones para comunicarse. Para que esta conexión sea posible uno deberá actuar como maestro o servidor y otro como esclavo. En este caso el servidor será la aplicación creada con el Sherlock7 y el esclavo el robot. Además se deberá establecer un número de puerto por el que comunicarse.

Para que el programa de visión artificial sea el maestro, en el mismo Sherlock se clicará sobre Options → IO. A continuación se abrirá una ventana en la cual se seleccionará Tcp/Ip y se marcarán los datos de acuerdo a la siguiente imagen. En lo referente al robot, más adelante se explicará el proceso en cuestión.

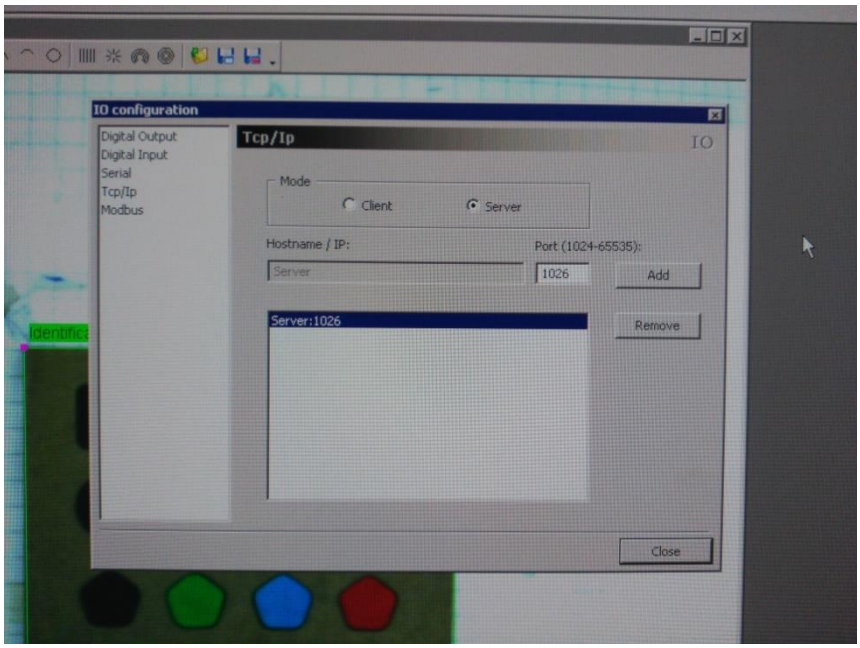


Figura 28. Creación en Sherlock7 servidor y puerto Tcp/Ip.

Una vez hecha la configuración, se procede a programar las funciones para recibir y entregar datos.

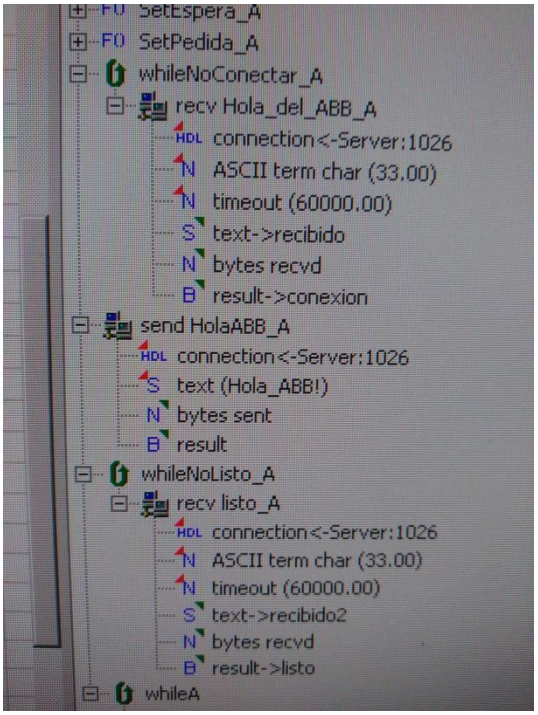


Figura 29. Código de la conexión con el robot.

En la imagen anterior, se presenta el código utilizado para que la aplicación se conecte con el robot. Dado que se ha determinado que el programa de visión sea el servidor, este espera hasta que el robot le pida conectarse. Una vez recibe la petición, este le contesta y espera la confirmación de que el robot está listo para empezar el proceso.

Una vez hay una comunicación establecida entre los dos dispositivos, la aplicación esperará hasta que el robot le pida los datos de la pieza y del lugar donde debe llevarla. A partir de ese momento, el programa reconocerá la pieza correspondiente, obtendrá la forma, color, ángulo y coordenadas, así como del lugar correspondiente. Para el envío de dichos datos, se ha condicionado que, dependiendo de la pieza que se detecte (círculo azul, cuadrado rojo, etc.) se cree una cadena de caracteres diferente.

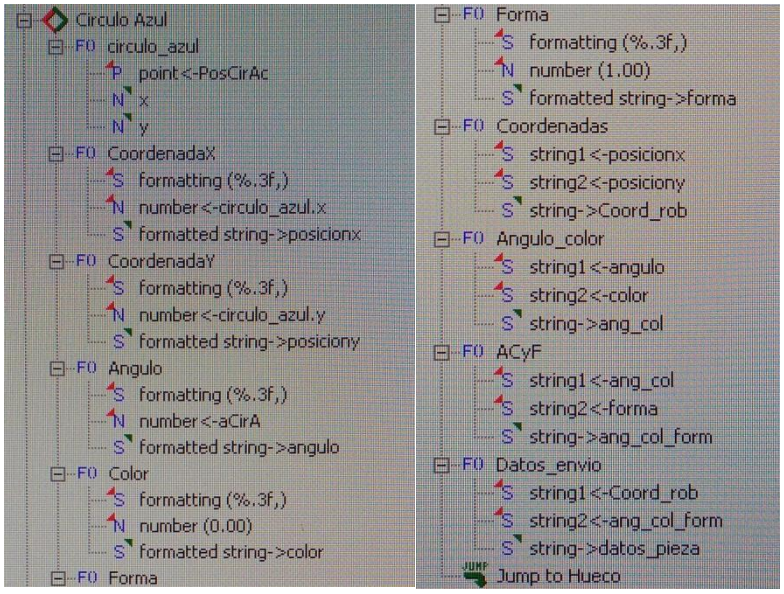


Figura 30. Ejemplo de la formación de una cadena de caracteres de las características de una pieza.

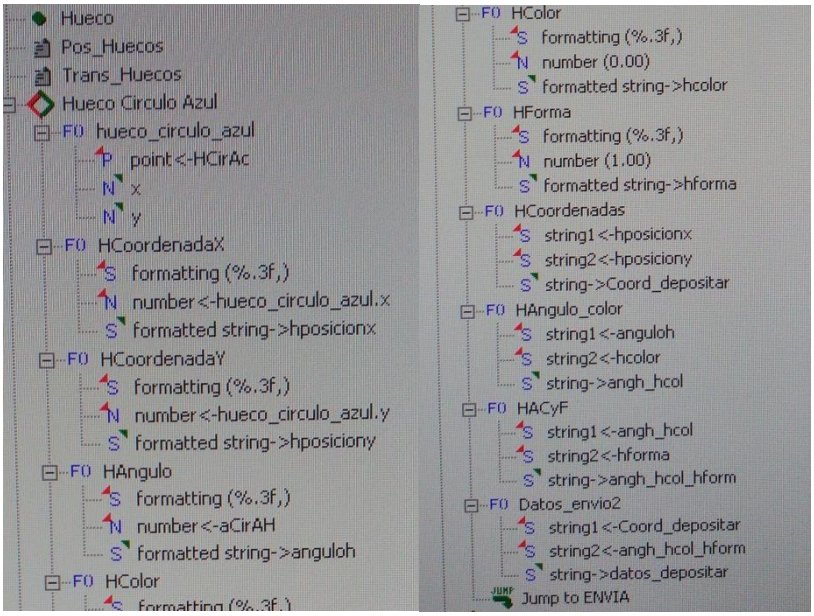


Figura 31. Ejemplo de la formación de una cadena de caracteres de las características del hueco del molde de la pieza correspondiente.

En las dos figuras anteriores se puede ver un ejemplo del proceso por el cual pasan los datos adquiridos en el análisis de la imagen para poder juntarse en una cadena de caracteres. Puesto que, como se verá más adelante, el programa del robot recibe una única cadena con toda la información, las dos cadenas anteriores se deben unir en una sola.

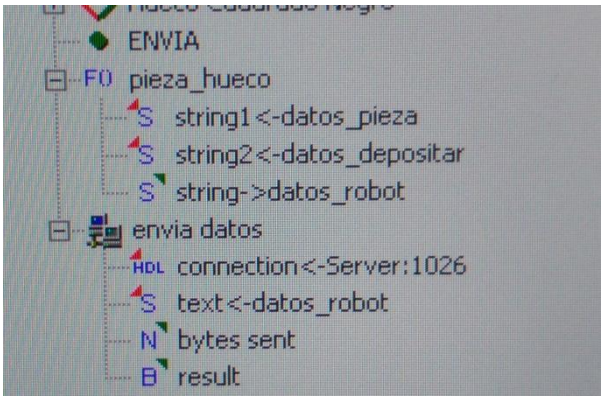


Figura 32. Unión de las dos cadenas de caracteres formadas en las figuras anteriores y envío de esta.

En la figura anterior, se expone cómo se han unido las cadenas con los datos de la pieza y del hueco al que corresponde y, su posterior envío al robot.

5.4. Breve introducción al lenguaje RAPID.

Como se ha visto en el apartado previo, hace falta una comunicación entre el sistema de visión con el robot para que este sea capaz de recoger las piezas y las deposite en el lugar adecuado. También se ha explicado la metodología empleada para crear la parte de la comunicación de la aplicación de visión.

A continuación, se va a comentar cómo se ha desarrollado el código de programa que hace que el robot se comunique debidamente. No sin antes introducir brevemente el lenguaje utilizado para dicha programación. El lenguaje RAPID.

RAPID es un lenguaje de programación de alto nivel utilizado para el control de robots industriales de la marca ABB.

El programa consta de tres partes, una rutina principal donde se inicia la ejecución, un conjunto de sub-rutinas donde se divide el programa en partes más pequeñas y los datos del programa los cuales definen las posiciones, valores, etc.

5.5. Diseño del programa RAPID para el robot ABB.

Dado que el brazo robot dependerá de los datos que reciba para efectuar los movimientos correspondientes, se explicará el programa en su totalidad. Es decir, no se comentará la conexión por una parte y los movimientos por otra.

Antes que nada, hay que saber que constará de una única rutina, la rutina principal y seguirá la siguiente estructura:

```
MODULE MainModule
Declaración de variables
    Proc main()
    Rutina
    ENDPROC
ENDMODULE
```

Antes de empezar a programar la rutina se deben declarar las variables que se van a utilizar así como las posiciones principales por las que pasará el robot. También, al tener que comunicarse con un dispositivo externo, se deberá crear el canal de comunicación. Este canal de comunicación se le conoce como *socket* y, como se ha dicho, permite la transmisión de datos utilizando el protocolo de red TCP/IP.

```
MODULE MainModule
    !DECLARACIÓN DE VARIABLES
    VAR socketdev canal1;
    VAR string respuesta:="";
    VAR string datos_pieza:="";
    VAR string datos_lugar:="";
    VAR num iteracion1;
    VAR num iteracion2;
    VAR bool valor1;
    VAR bool valor2;

    !DECLARACIÓN DE ROBTARGET

    CONST robtarget pIni:=[[0,450,647],[0,-0.70748,0.7674,0]
        ,[0,2,2,1],[9E9,9E9,9E9,9E9,9E9,9E9]];
```

Figura 33. Ejemplo de la declaración de algunas variables de diferente naturaleza y de puntos de trayectoria del robot.

A continuación, se expondrán imágenes de parte del código diseñado y se explicará su relevancia en el programa.

En esta imagen se ordena al robot que vaya a la posición de inicio y se borra cualquier cosa escrita en la flexpendant.

```
!PROGRAMA PRINCIPAL

PROC main()
    !Se mueve el robot a la posición de inicio

    MoveJ pIni,v300,fine,tool0;

    !SE BORRA LO QUE HAYA ESCRITO EN LA FLEXPENDANT

    TPErase;
```

Figura 34. Código1. Robot a inicio y borrar lo escrito en la flexpendant.

A continuación, se crea el canal de comunicación, se dice a qué dispositivo debe conectarse y se inicia una conversación con la aplicación para establecer la conexión.

```
!COMUNICACIÓN CON SHERLOCK

SocketCreate canal1;
SocketConnect canal1,"158.42.16.207",1026;
SocketSend canal1\Str:="Hola_Sherlock!";
SocketReceive canal1\Str:=respuesta;
SocketSend canal1\Str:="Cuando_quieras!";
WaitTime 1;
TPWrite respuesta;
```

Figura 35. Código2. Comunicación con la aplicación.

Seguidamente, se crea un bucle infinito, el cual acogerá la parte del programa en la que se reciben las coordenadas y se mueve el robot, se reinician algunas variables y se desactiva la succión de la ventosa. Además se ordena que se mueva la cinta de transporte y que pare cuando el sensor detecte un objeto.

```
WHILE TRUE DO
datos_pieza:="";
datos_lugar:="";
posx1:=0;
posx2:=0;
!Set GRIPPER_OPEN;
Set CONVEYOR_FWD; !SE ENCIENDE LA CINTA
WaitDI CONVEYOR_OBJ_SEN,1; !ESPERA HASTA QUE SE DETECTE OBJETO
Reset CONVEYOR_FWD; !PARA LA CINTA
```

Figura 36.Código3.Bucle infinito, activar ventosa y movimiento de cinta.

Cuando el sensor ha detectado objeto y la cinta está parada, se ordena una pequeña pausa. Esto se hace para que la aplicación de visión haga una buena captura de imagen. De lo contrario, si se hiciera la captura nada más el sensor detectara la pieza, esta sería errónea puesto que la cinta tarda unas milésimas de segundo en parar.

Una vez la cinta está parada, se borra la flexpendant y se envía un mensaje a la aplicación para que le envíe los datos, estos son recibidos y se escriben.

```
WaitTime 1;

!SE ENVÍA UN MENSAJE A SHERLOCK PARA QUE ANALICE LA IMAGEN
TPErase;

SocketSend canal1\Str:="Enviame_Datos!";
SocketReceive canal1\Str:=datos_pieza;
longitud_trama1:=StrLen(datos_pieza);
TPWrite "Longitud trama"\Num:= longitud_trama1;
TPWrite datos_pieza;
```

Figura 37.Código4. Recibimiento de datos.

En el siguiente trozo de código, se analiza la cadena de datos recibida. Lo que se hace es crear un bucle de 10 iteraciones (el programa recibe una cadena con 10 datos) y, se crea otro bucle para que en cada iteración se muestree la cadena de datos hasta que se encuentra con una “,”. Al encontrar la coma, se sabe la longitud de un dato y este se guarda en una variable.

```
!SE RECONOCE LA CADENA DE DATOS
IF longitud_trama1 > 0 THEN
    posicion_string_inicial1:=0;
    posicion_string_final1:=1;
    iteracion1:=1;

    WHILE iteracion1 <= 10 DO
        ChPos1:= posicion_string_inicial1 + posicion_string_final1;
        v1:= StrPart(datos_pieza,ChPos1,1);
        WHILE v1<>"," DO
            posicion_string_final1:= posicion_string_final1 + 1;
            ChPos1:= ChPos1 + 1;
            v1:= StrPart(datos_pieza,ChPos1,1);
        ENDWHILE
        Len1:= posicion_string_final1 - 1;
        Chpos2:= posicion_string_inicial1 + 1;
        dato1:=StrPart(datos_pieza,Chpos2,Len1);
```

Figura 38.Código5. Reconocimiento de la cadena de datos.

Una vez se ha guardado el dato en una variable, se procede a su identificación y se hace una conversión de tipo de variable, de *string* a *num.*

Dado que los datos de la cadena siguen siempre el mismo orden, se crean condiciones las cuales, dependiendo de la iteración en la que esté el bucle, se asignará el dato obtenido a su correspondiente variable (coordenadas x e y, ángulo, color o forma). También, dependiendo de la iteración indicará si el dato corresponde a la pieza o al hueco donde depositarla; del 1 al 5 corresponde a la pieza y las 5 posteriores al lugar.

```
IF iteracion1 = 1 THEN
    valor1:=StrToVal(dato1,posx1);
    TPWrite "Posición X = "\Num:=posx1;

ELSEIF iteracion1 = 2 THEN
    valor1:=StrToVal(dato1,posy1);
    TPWrite "Posición Y = "\Num:=posy1;

ELSEIF iteracion1 = 3 THEN
    valor1:=StrToVal(dato1,angulo1);
    !Conversión grados radianes
    angulo1grad:= angulo1 * (180/pi);
    TPWrite "Ángulo = "\Num:=angulo1grad;

ELSEIF iteracion1 = 4 THEN
    valor1:=StrToVal(dato1,color1);
    IF color1 = 0 THEN
        TPWrite "Color: Azul ";

    ELSEIF color1 = 1 THEN
        TPWrite "Color: Rojo ";

    ELSEIF color1 = 2 THEN
        TPWrite "Color: Verde ";

    ELSEIF color1 = 3 THEN
        TPWrite "Color: Negro ";
    ENDIF

ELSEIF iteracion1 = 5 THEN
    valor1:=StrToVal(dato1,forma1);
    IF forma1 = 0 THEN
        TPWrite "Forma: Cuadrado";

    ELSEIF forma1 = 1 THEN
        TPWrite "Forma: Círculo";

    ELSEIF forma1 = 2 THEN
        TPWrite "Forma: Pentágono";
    ENDIF

ELSEIF iteracion1 = 6 THEN
    valor1:=StrToVal(dato1,posx2);
    TPWrite "Posicion Hueco X ="\Num:= posx2;

ELSEIF iteracion1 = 7 THEN
    valor1:=StrToVal(dato1,posy2);
    TPWrite "Posicion Hueco y ="\Num:= posy2;

ELSEIF iteracion1 = 8 THEN
    valor1:=StrToVal(dato1,angulo2);
    !Conversión grados radianes
    angulo2grad:= angulo2 * (180/pi);
    TPWrite "Ángulo = "\Num:=angulo2grad;

ELSEIF iteracion1 = 9 THEN
    valor1:=StrToVal(dato1,color2);
    IF color2 = 0 THEN
        TPWrite "Color: Azul ";

    ELSEIF color2 = 1 THEN
        TPWrite "Color: Rojo ";

    ELSEIF color2 = 2 THEN
        TPWrite "Color: Verde ";

    ELSEIF color2 = 3 THEN
        TPWrite "Color: Negro ";
    ENDIF

ELSEIF iteracion1 = 10 THEN
    valor1:=StrToVal(dato1,forma2);
    IF forma2 = 0 THEN
        TPWrite "Forma: Cuadrado";

    ELSEIF forma2 = 1 THEN
        TPWrite "Forma: Círculo";

    ELSEIF forma2 = 2 THEN
        TPWrite "Forma: Pentágono";
    ENDIF

ENDIF
```

Figura 39.Código6. Instrucciones de condición.

Una vez se obtienen todos los datos, es decir, se cumplen todas las iteraciones y termina el bucle, es hora de analizar el ángulo real de las piezas. Esto quiere decir que, dado que la aplicación puede dar como valor a una misma posición varios ángulos diferentes, se debe saber el ángulo verdadero.

Por ejemplo, un cuadrado que está colocado a cero grados, la aplicación puede reconocer que está a 0, 90, 180 o 360 grados, todos correctos pero no se puede permitir que para una misma posición se tengan diferentes ángulos.

Por eso se ha creado un algoritmo el cual restará, de mayor a menor, los posibles valores para una posición y, si el resultado está dentro de un margen, el ángulo real será mínimo de los valores que se pretenden restar.

Además, puesto que las piezas pueden llegar en dos posiciones diferentes (excepto el círculo), se ha creado la condición de que si después de todas las operaciones hechas, el valor del ángulo no coincide con ninguno de los restados, significará que está en la otra posición. Por lo tanto, la orientación con la que el brazo debe recoger la pieza será diferente.

También se ha distinguido entre formas debido a que los ángulos exteriores del cuadrado y pentágono no coinciden.

En la siguiente imagen se observará el trozo de código encargado de realizar esta acción.

```
IF forma1 = 0 THEN  !CUADRADO
    anguloC0:= angulo1grad - 360;
    IF anguloC0 > -15 AND anguloC0 < 15 THEN
        angulo1grad:=0;
    ENDIF
    anguloC0:= angulo1grad - 270;
    IF anguloC0 > -15 AND anguloC0 < 15 THEN
        angulo1grad:=0;
    ENDIF
    anguloC0:= angulo1grad - 180;
    IF anguloC0 > -15 AND anguloC0 < 15 THEN
        angulo1grad:=0;
    ENDIF
    anguloC0:= angulo1grad - 90;
    IF anguloC0 > -15 AND anguloC0 < 15 THEN
        angulo1grad:=0;
    ENDIF
    IF angulo1grad > -15 AND angulo1grad < 15 THEN
        angulo1grad:=0;
    TPWrite "Angulo real"\Num:= angulo1grad;

    MoveJ Offs(pCinta1,posx1,posy1,0),v300,fine,tool0;
    MoveL Offs(pCinta1,posx1,posy1,-320),v100,fine,tool0;
    ELSE
    TPWrite "Angulo real"\Num:= angulo1grad;

    MoveJ Offs(pCinta45,posx1,posy1,0),v300,fine,tool0;
    MoveL Offs(pCinta45,posx1,posy1,-320),v100,fine,tool0;
    ENDIF

ENDIF

IF forma1 = 1 THEN  !CÍRCULO
   TPWrite "Angulo real"\Num:= angulo1grad;

   MoveJ Offs(pCinta1,posx1,posy1,0),v300,fine,tool0;
   MoveL Offs(pCinta1,posx1,posy1,-320),v100,fine,tool0;
ENDIF
```

```
IF forma1 = 2 THEN   !PENTÁGONO
    anguloP0:= angulo1grad - 360;
    IF anguloP0 > -15 AND anguloP0 < 15 THEN
        angulo1grad:=0;
    ENDIF
    anguloP0:= angulo1grad - 288;
    IF anguloP0 > -15 AND anguloP0 < 15 THEN
        angulo1grad:=0;
    ENDIF
    anguloP0:= angulo1grad - 216;
    IF anguloP0 > -15 AND anguloP0 < 15 THEN
        angulo1grad:=0;
    ENDIF
    anguloP0:= angulo1grad - 144;
    IF anguloP0 > -15 AND anguloP0 < 15 THEN
        angulo1grad:=0;
    ENDIF
        anguloP0:= angulo1grad - 72;
    IF anguloP0 > -15 AND anguloP0 < 15 THEN
        angulo1grad:=0;
    ENDIF
    IF angulo1grad > -15 AND angulo1grad < 15 THEN
        angulo1grad:=0;
    TPWrite "Angulo real"\Num:= angulo1grad;

    MoveJ Offs(pCinta1,posx1,posy1,0),v300,fine,tool0;
    MoveL Offs(pCinta1,posx1,posy1,-320),v100,fine,tool0;
    ELSE
    TPWrite "Angulo real"\Num:= angulo1grad;

    MoveJ Offs(pCinta36,posx1,posy1,0),v300,fine,tool0;
    MoveL Offs(pCinta36,posx1,posy1,-320),v100,fine,tool0;
    ENDIF

ENDIF
```

Figura 40.Código7. Izquierda: Algoritmo del cuadrado y círculo. Derecha: Algoritmo del pentágono.

Como se puede observar, dependiendo del ángulo, el robot irá a la cinta en diferentes orientaciones. Estás están internas cuando se ha declarado el punto pCinta1, pCinta45 y pCinta36.

A continuación, se activa la succión de la ventosa para que la coja y se lleve el robot a una posición más alta. Esto último se hace para evitar posibles colisiones con la cinta al desplazarse a la siguiente posición.

```
!EL ROBOT COGE LA PIEZA
    Reset GRIPPER_OPEN;

!ROBOT VUELVE A POSICIÓN COORDENADA ARRIBA

    MoveL Offs(pCinta1,posx1,posy1,0),v100,fine,tool0;
```

Figura 41.Código8. Recogida de la pieza.

Seguidamente, el robot lleva la pieza su correspondiente posición, la suelta y vuelve a una posición más alta.

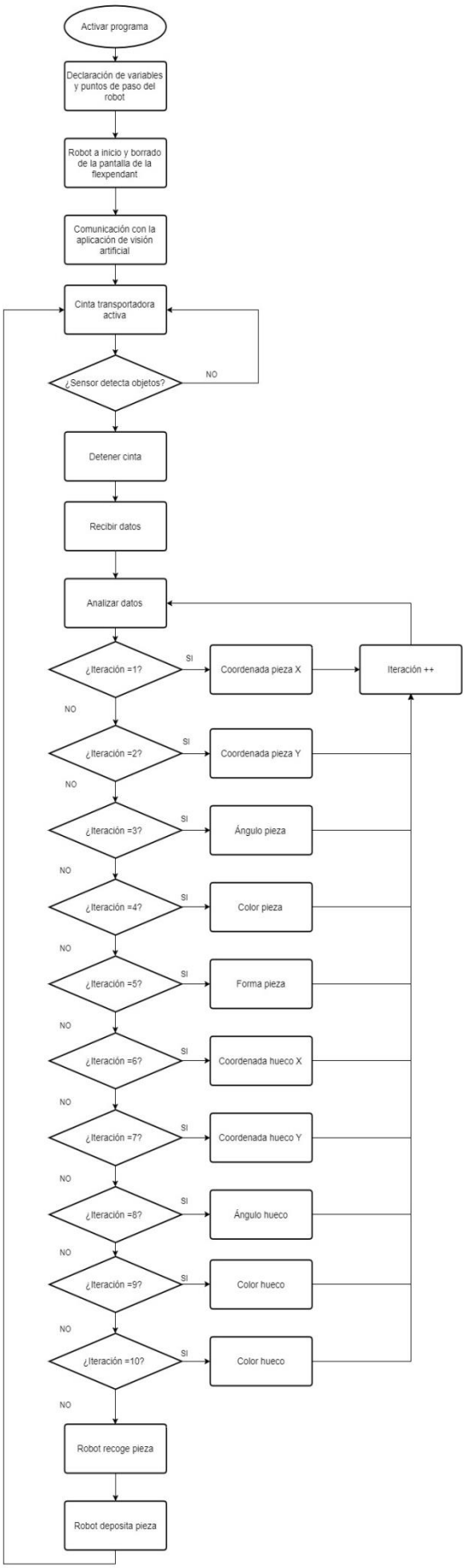
```
    MoveJ pTablero,v300,z20,tool0;
    MoveL Offs(pTablero,0,0,-200),v100,z5,tool0;
    MoveL Offs(pTablero,posx2,posy2,-330),v100,fine,tool0;
    MoveL Offs(pTablero,posx2,posy2,-350),v100,fine,tool0;

!EL ROBOT DEPOSITA LA PIEZA
    !Set GRIPPER_OPEN;

!ROBOT VUELVE A POSICIÓN COORDENADA ARRIBA

    MoveL Offs(pTablero,posx2,posy2,-200),v100,z10,tool0;
    MoveL pTablero,v100,fine,tool0;
```

Figura 42.Código9. El robot lleva la pieza al sitio, la suelta y vuelve a una posición más alta.

Finalmente, cuando se deposita la pieza, se vuelve a encender la cinta y se sigue el mismo proceso explicado hasta que se detenga manualmente.

5.6. Flujograma del programa de RAPID

A continuación se presenta un flujograma el cual representa gráficamente los pasos seguidos por el programa.



6. Resultados

Una vez se ha terminado con el diseño de la aplicación y la programación del robot es hora de comprobar si el conjunto funciona adecuadamente.

Cabe decir que, en la parte de visón se comprobó que el reconocimiento de color y de formas fuera correcto antes de empezar a diseñar la parte de comunicación con el robot. Esto fue necesario porque, una vez implementada la comunicación, sería más difícil corregir los fallos. Además de, no tener sentido empezar a enviar datos al robot sin funcionar la aplicación debidamente.

Antes de hacer las pruebas con el robot real, se utilizó el programa RobotStudio para simular el movimiento del autómata con el fin de prevenir riesgos, en el caso de que algo fallara. Esto es importante dado que puede evitar daños en equipos e incluso en el mismo robot.

Una vez que las simulaciones tienen el resultado esperado, se puede comprobar el funcionamiento del equipo real. Para ello, se carga el programa al robot, se pone en modo automático y se encienden los motores. Acto seguido, se ejecuta la aplicación de visión en modo continuo y se inicia el programa del robot.

Después de varias pruebas, se comprueba que el sistema reconoce las formas y los colores a la perfección dado que las deja en su correspondiente sitio. Sin embargo se observa un error sistemático en las coordenadas X (respecto el sistema de referencia del robot). Este error impide que las piezas encajen en los huecos del molde quedándose levemente fuera de ellos.

En lo referente a la orientación de las piezas, se puede decir el algoritmo creado para determinar los ángulos de estas, consigue realizar su función de forma correcta. No obstante, si las piezas están algo desviadas de la orientación correspondiente, puede haber un error a la hora de que encaje en su lugar. Esto es debido a que la pieza termine girada más, o menos de lo deseado.

7. Solución de los problemas de posicionamiento.

Como se ha comentado, hay un error sistemático en las coordenadas X de los huecos del molde. Dado que uno de los objetivos del proyecto es que el robot sea capaz de depositar las piezas en su correspondiente lugar dentro del molde, es necesario solucionar este problema.

Existen diversos métodos que se pueden utilizar para minimizar o incluso eliminar el error.

El primero y más sencillo, es añadir un offset a la coordenada X, es decir, dependiendo de si la pieza se queda más cerca o más lejos del robot, sumarle o restarle un valor a la coordenada X para que la pieza llegue al lugar deseado.

El segundo, sería recalibrar las posiciones del robot en el molde utilizando la ventosa. En la primera calibración se usó como referencia una pinza de 3 dedos, factor que puede haber afectado a una calibración no tan precisa como se desearía.

El tercer método, si con la segunda calibración no funcionara, sería asignar directamente, a los huecos, sus correspondientes coordenadas dependiendo de la forma y del color.

```
!EL ROBOT SE MUEVE A LA COORDENADA DEL LUGAR
    IF forma2 = 0 AND color2 = 3 THEN
        posx2 := 568 - 450 ;
        posy2 := -21;
    ENDIF
    IF forma2 = 1 AND color2 = 3 THEN
        posx2 := 652 - 450 ;
        posy2 := -21;
    ENDIF
    IF forma2 = 2 AND color2 = 3 THEN
        posx2 := 742 - 450 ;
        posy2 := -18;
    ENDIF
    IF forma2 = 0 AND color2 = 2 THEN
        posx2 := 568 - 450 ;
        posy2 := 59;
    ENDIF
    IF forma2 = 1 AND color2 = 2 THEN
        posx2 := 652 - 450 ;
        posy2 := 59;
    ENDIF
    IF forma2 = 2 AND color2 = 2 THEN
        posx2 := 742 - 450 ;
        posy2 := 59;
    ENDIF
IF forma2 = 0 AND color2 = 0 THEN
    posx2 := 568 - 450 ;
    posy2 := 139;
ENDIF
IF forma2 = 1 AND color2 = 0 THEN
    posx2 := 652 - 450 ;
    posy2 := 139;
ENDIF
IF forma2 = 2 AND color2 = 0 THEN
    posx2 := 742 - 450 ;
    posy2 := 136;
ENDIF
IF forma2 = 0 AND color2 = 1 THEN
    posx2 := 568 - 450 ;
    posy2 := 219;
ENDIF
IF forma2 = 1 AND color2 = 1 THEN
    posx2 := 653 - 450 ;
    posy2 := 219;
ENDIF
IF forma2 = 2 AND color2 = 1 THEN
    posx2 := 742 - 450 ;
    posy2 := 216;
ENDIF
```

Figura 43. Ejemplo de asignación de coordenadas a los huecos del molde según su forma y color.

8. Conclusiones

Los objetivos principales del trabajo se pueden dar por cumplidos. Reconocimiento de formas y colores, coger las piezas y depositarlas en su lugar correspondiente dependiendo de las dos características anteriores.

Con la realización de este trabajo, se han ampliado los conocimientos de visión artificial que se adquirieron al principio del curso académico.

Los equipos del laboratorio has sido más que suficientes para realizar el TFG. Aunque en condiciones reales no hubiera sido el elegido para una aplicación como la del proyecto debido a que, hoy en día, existen brazos robóticos menos aparatosos y más rápidos. Además, el sistema de visión artificial está algo anticuado puesto que la pila interna del sistema de visión integrado se ha agotado. Por otra parte, la cámara CV-M77 está, actualmente, descatalogada y ha sido difícil encontrar razones para explicar el por qué se ha usado esta cámara. Otra objeción es el hecho de tener que activar/desactivar manualmente la succión de la ventosa para coger o dejar las piezas.

A pesar de esto, se ha aprendido la importancia de la visión artificial para las diferentes aplicaciones en la industria, en los cuales hace falta analizar a diversas características de los productos que se manipulan.

La mayor dificultad en el desarrollo del trabajo ha sido a la hora de transformar las coordenadas obtenidas con el sistema de visión artificial en coordenadas entendibles para el robot. Esto ha sido porque se necesita calibrar el sistema para conseguir dicho objetivo y, si no se hace un calibrado perfecto de las áreas donde se va a trabajar aparecen errores en la transformación de las coordenadas.

También se ha aprendido a comunicar los dos sistemas entre sí. Algo bastante útil puesto que seguro será de utilidad en futuros proyectos.

Por último, decir que los documentos y la ayuda aportados por el tutor han sido de gran ayuda en la realización del trabajo puesto que se han resuelto varias dudas a lo largo del proyecto gracias a ello.

9. Bibliografía

[1]http://www.profesormolina.com.ar/tecnologia/robotica/historia.htm

[2]http://robotiica.blogspot.com.es/2007/10/historia-de-la-robtica.html

[3]https://roboticstoday.wikispaces.com/Historia+de+la+Rob%C3%B3tica

[4]https://poliformat.upv.es/access/content/group/GRA_12157_2016/CASTELLANO/7_Lenguaje%20RAPID.pdf

[5]https://poliformat.upv.es/access/content/group/GRA_12169_2016/Actividades/3.Ejercicios%20con%20Sherlock/Sherlock/Manual%20Sherlock_v7100.pdf

[6]http://www.jcee.upc.es/JCEE2010/pdf_ponencies/PDFs/25_11_10/INFAIMON-Vision%20artificial.pdf

[7]https://poliformat.upv.es/access/content/group/GRA_12157_2016/CASTELLANO/PRINCIPIOS%20BASICOS%20_Modo%20de%20compatibilidad_.pdf

[8]https://poliformat.upv.es/access/content/group/GRA_12169_2016/Unidades%20tem%C3%A1ticas/1-Introducci%C3%B3n%20a%20los%20sistemas%20de%20Visi%C3%B3n%20Artificial%20v0.pdf

Tabla de figuras

MANUAL DE USUARIO

GRADO EN INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

Curso académico 2016-2017

SISTEMA PARA EL EMPAQUETADO AUTOMÁTICO DE JUGUETES DE PIEZAS ENCAJABLES

Índice

1. Introducción

El siguiente documento explica al usuario los pasos a seguir para que el sistema funcione correctamente. También se detallan formas de solucionar posibles problemas que puedan ocurrir durante el funcionamiento de este.

2. Sistema

El sistema robotizado está controlado por la aplicación programada en el *Sherlock7* y por la flexpendant.

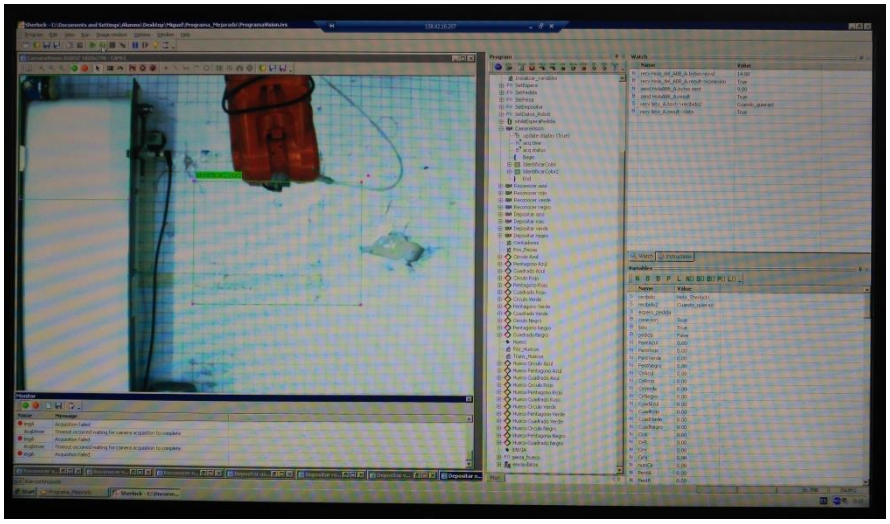


Imagen 1. Aplicación responsable de la visión.

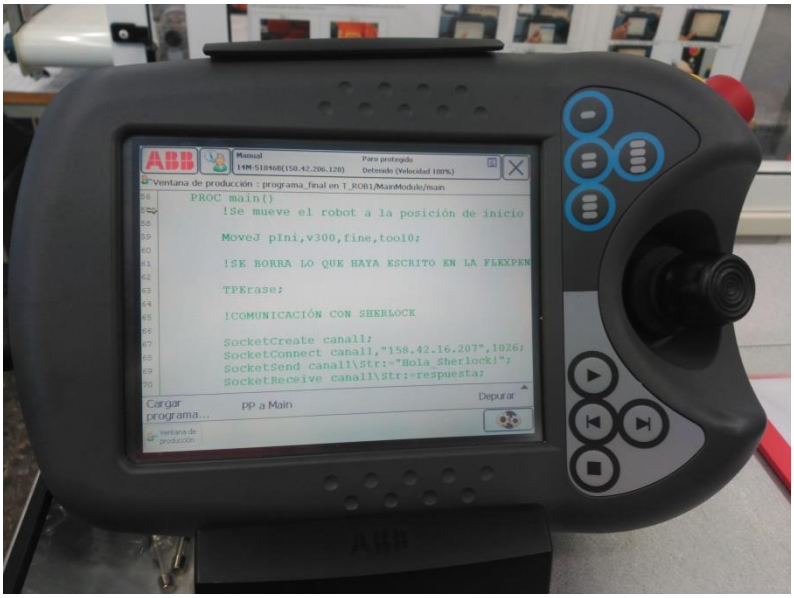


Imagen 2. Flexpendant.

3. Inicio de la producción

Dado que es un sistema automatizado, no se precisa de operarios con formación especializada para iniciar la producción. Será suficiente con seguir los pasos explicados a continuación:

- Primero hay que encender todos los equipos.
    - El ordenador conectado a la cámara no tiene pantalla, por lo que hay que conectarse vía escritorio remoto.

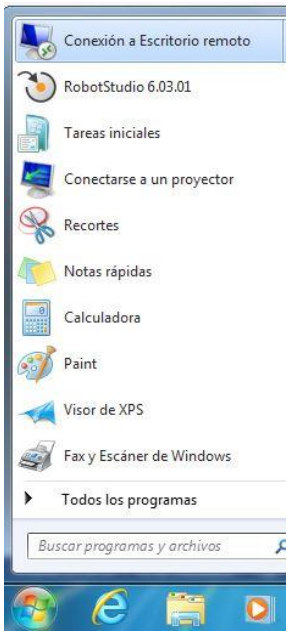


Imagen 3. Conexión a escritorio remoto 1.

Imagen 4. Conexión a escritorio remoto 2.

Una vez seguidos estos pasos sólo se tiene que insertar el usuario y contraseña.

- Para el robot y la flexpendant se tiene que cambiar la posición del interruptor de la unidad de control a ON.

Imagen 5. Interruptor de encendido/apagado de la unidad de control.

- Una vez encendidos los equipos, se abrirá la aplicación de visión artificial. También se tiene que cambiar el robot a modo automático y encender los motores. Esto se hará girando la llave situada en la unidad de control hacia la posición de más a la izquierda y apretando el botón blanco.

Imagen 6. Izquierda: robot en modo manual. Derecha: modo automático.

- En el caso de que el programa del robot no esté cargado, se deberá cargar mediante una memoria USB, que contenga el programa, conectándola a la unidad de control.

Imagen 7. Memoria USB conectada a la unidad de control del robot.

A continuación, se utiliza la flexpendant para descargar el programa en el robot. Esto se hace pulsando la pestaña superior izquierda.

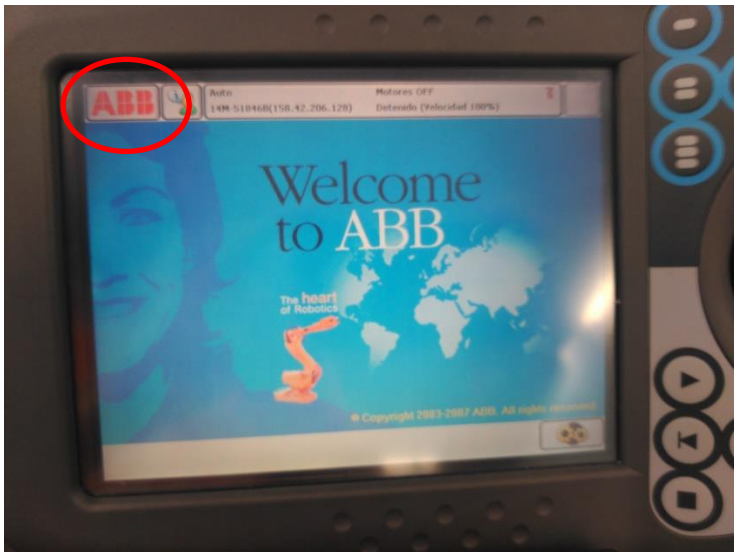


Imagen 8. Flexpendant ventana principal.

Seguidamente pulsamos en editor de programa:

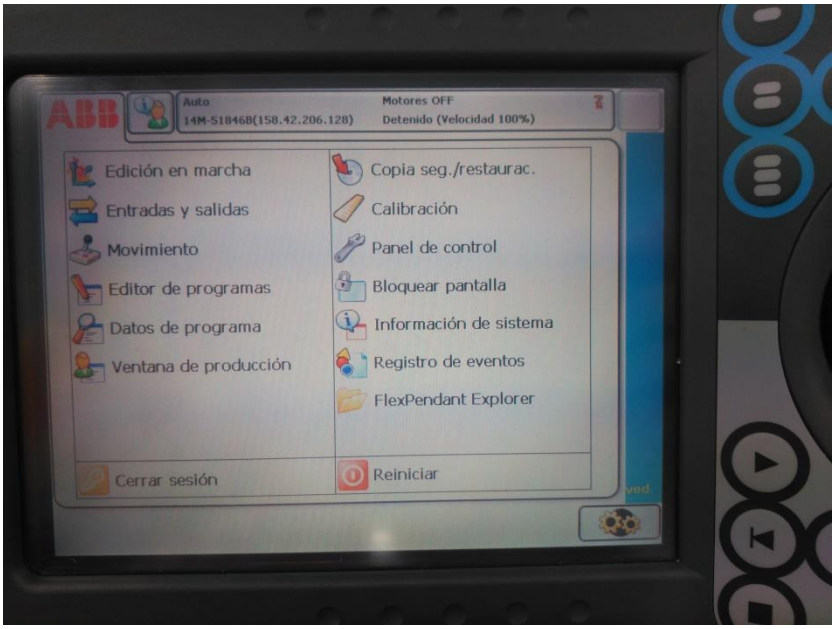


Imagen 9. Flexpendant ventana inicio.

Se abrirá una nueva ventana y se deberá pulsar la pestaña de “Tareas y programas”, arriba a la izquierda:

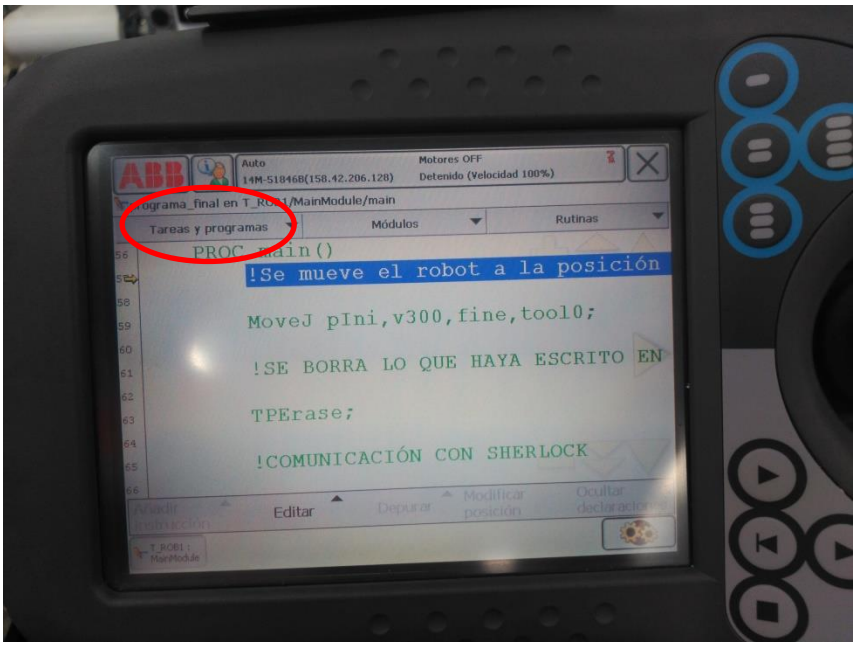


Imagen 10. Flexpendant editor de programas.

Entonces la pestaña se abrirá y pulsaremos en archivo, lo cual desplegará un menú y se elegirá la opción de “Cargar programa”.

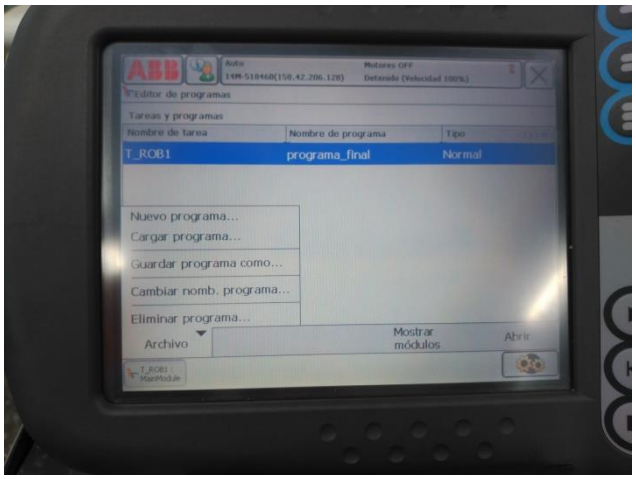


Imagen 11. Flexpendant tareas y programas.

A continuación, aparecerá una ventana nueva, buscaremos el archivo correspondiente, se seleccionará y se dará a OK.

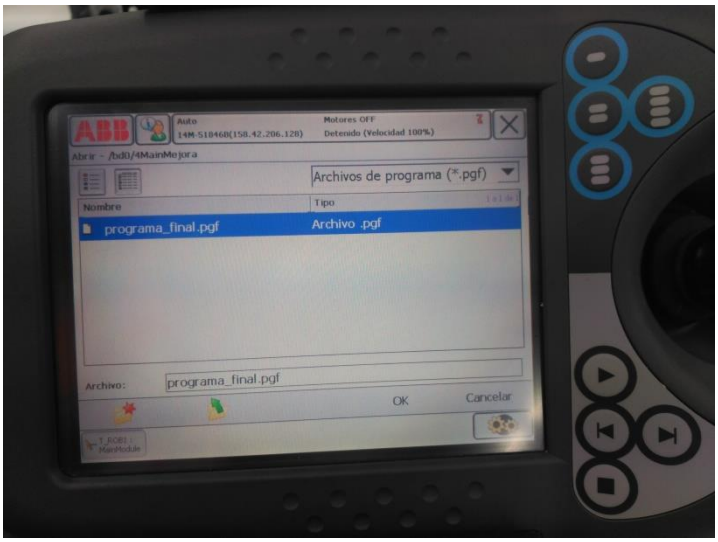


Imagen 12. Flexpendant cargar archivo.

- Una vez cargado el archivo y abierto el programa de visión, en este último se pulsará sobre el icono resaltado en la siguiente imagen.

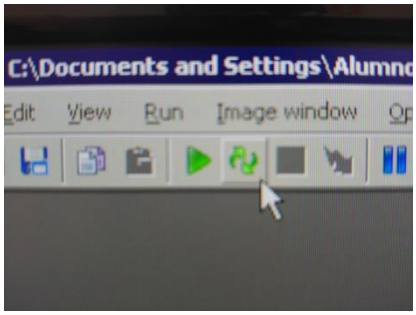


Imagen 13. Icono que hace que el programa se ejecute indefinidamente.

A continuación, se pulsara el botón”play” de la flexpendant.

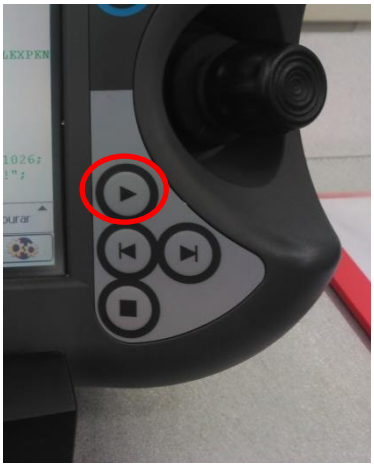


Imagen 14. Botón "play" en la flexpendant.

Es importante que el orden sea el indicado puesto que se ha configurado la aplicación como el servidor y el robot como cliente. También se debe pulsar en la flexpendant “PP to main” para que el programa empiece desde el principio. Está localizado en la ventana de producción en la parte inferior izquierda.

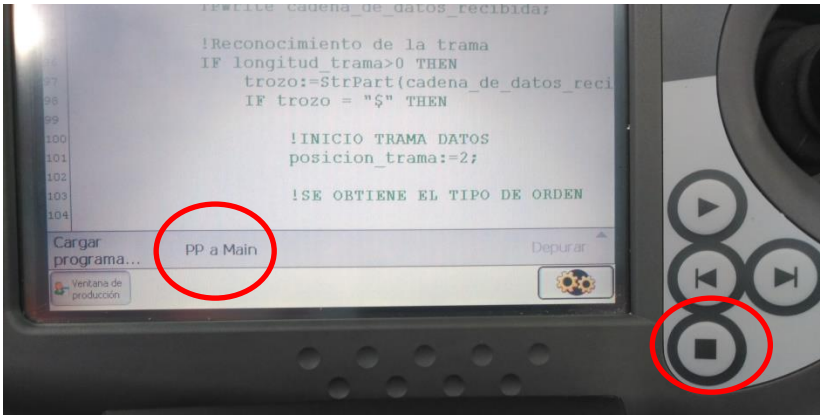


Imagen 15. Botón "stop" y comando "PP a Main" en la flexpendant.

En el caso de querer parar el sistema, se pulsará el botón “stop” de la flexpendant situado dos posiciones más abajo del “play”. A continuación, se pulsa el icono de la siguiente imagen en la aplicación de visión. Sirve para abortar la ejecución del programa.

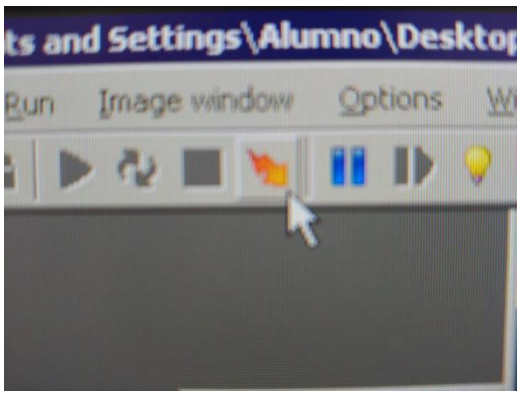


Imagen 16. Icono de abortar el funcionamiento.

Para volver a poner en marcha el sistema se siguen los mismos pasos explicados anteriormente. Es recomendable cerrar la aplicación y volverla a abrir dado que algunos datos se pueden haber quedado guardados. Esto, a veces, hace que el robot y el sistema de visión no se consigan comunicar. Por lo que no funciona el sistema entero.

4. Movimiento manual del robot y accionamiento de actuadores y sensores.

En este apartado se va a explicar cómo manejar el robot y los actuadores de forma manual.

Para este propósito, se usará la flexpendant. Pero, antes que nada, se necesita que el robot esté en modo manual (véase Imagen 6). Una vez comprobado esto, se puede coger el aparato para mayor comodidad a la hora de controlar los dispositivos.

Respecto a los actuadores y sensores, en la ventana de inicio (Imagen 9) existe una opción llamada “Entradas y salidas”. Si pulsamos sobre esta, se mostrará en la pantalla una lista con el nombre de estas y su valor. Si se quisiera manipular dicho valor, basta con pulsar sobre el nombre del elemento a controlar y aparecerá la opción de asignarle el valor 1 o 0.

El control del robot es más complejo. Aquí se necesita tener la flexpendant en la mano, puesto que es obligatorio mantener pulsado un botón, llamado el botón de hombre muerto, para poder mover el robot. Esto se hace para evitar accidentes dado que si se golpeara el joystick de movimiento por error, el robot no haría ninguna acción porque el botón no estaría pulsado.

Imagen 17. Botón de hombre muerto.

Para mover el autómata, pulsamos la opción “Movimiento” de la ventana de inicio (Imagen 9). A continuación, se abrirá la siguiente ventana:

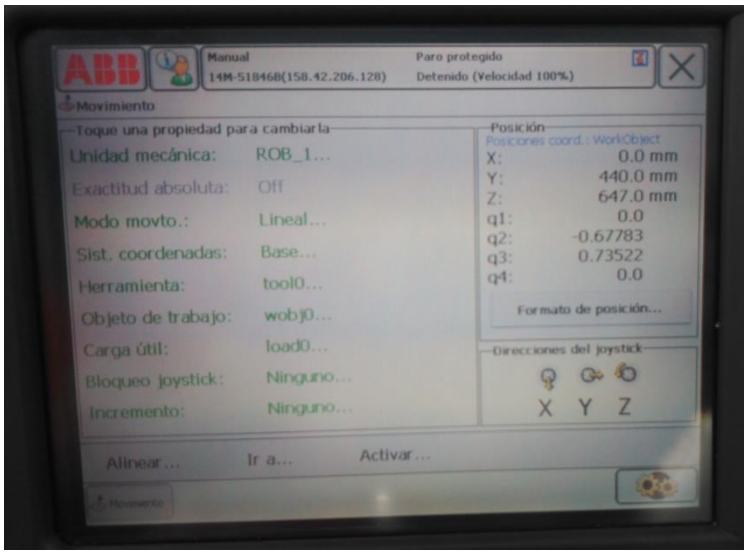


Imagen 18. Flexpendant movimiento lineal.

Se puede observar que en la tercera línea hay una propiedad llamada “Modo movto”. Esta indica la forma en la que se quiere que se mueva el robot. En la imagen está en modo lineal, lo que quiere decir que el robot se moverá de acuerdo a sus ejes de coordenadas. Más tarde se explicará cómo se puede cambiar.

En la parte derecha se observan dos apartados, la posición y direcciones del joystick. La posición indica la situación, en el espacio cartesiano, del extremo del robot. Además, también ofrece información sobre los cuaterniones. El otro apartado, indica la dirección en la que se tiene que mover el joystick para que el robot se mueva en el eje positivo de su sistema de referencia.

Como se ha mencionado anteriormente, el modo de movimiento se puede cambiar. Para hacerlo se debe pulsar encima de la propiedad con el mismo nombre y seleccionar el modo deseado.

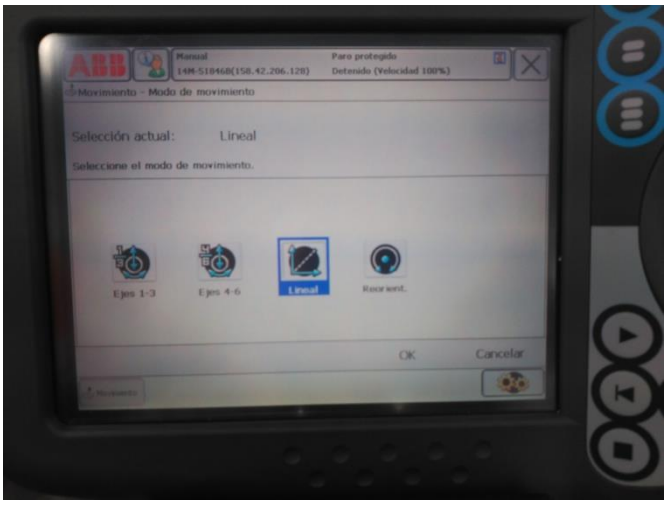


Imagen 19. Ventana de elección del modo de movimiento.

En los modos de “Ejes 1-3” y “Ejes 4-6”, tal y como indican sus nombres, se puede mover el robot según sus ejes. La pantalla de estos modos sólo se diferencia con el anterior en la parte de posición. En estos casos, se muestran los grados que están girados cada articulación.

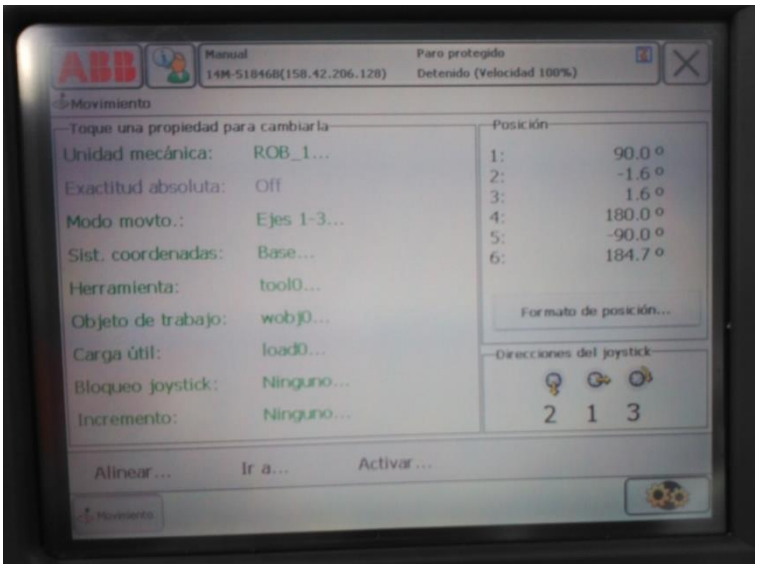


Imagen 20. Flexpendant ventana movimiento Ejes 1-3.

Tabla de imágenes

PLIEGO DE CONDICIONES

GRADO EN INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

Curso académico 2016-2017

SISTEMA PARA EL EMPAQUETADO AUTOMÁTICO DE JUGUETES DE PIEZAS ENCAJABLES

Índice

1. Definición y alcance del pliego

La presente especificación técnica se refiere al diseño e instalación de un sistema de embalaje automatizado de piezas planas según su geometría y color, compuesto por un robot industrial, una cinta transportadora y un equipo de visión artificial, para la recogida y deposición de piezas planas según su geometría y color.

El ámbito de aplicación de este documento se extiende a todos los sistemas eléctricos, electrónicos, mecánicos y neumáticos que forman parte de la instalación.

2. Condiciones de carácter general y normativa

Los materiales y equipos empleados deberán ser los señalados en el proyecto. Estos tienen que ser de buena calidad, además deben cumplir con la las prescripciones contenidas en los reglamentos y normativas vigentes.

En caso de no cumplimiento de estas características, supondrá el reemplazo de estos por otros que sí que las cumplan.

Los operarios deberán seguir en todo momento las directrices indicadas en este proyecto para la correcta instalación de los equipos.

Una vez esté todo instalado, se harán pruebas de todo el equipo hasta comprobar que el funcionamiento es correcto. En caso de que fuera necesario, se harán las modificaciones o ajustes necesarios.

La normativa con respecto al sistema de visión artificial en lo referente a la salud y las medidas de seguridad son las expuestas en el Real Decreto 1215/1997.

Respecto a los robots industriales, la normativa vigente es la siguiente:

- **UNE-EN ISO 10218-1:2012** Robots y dispositivos robóticos. Requisitos de seguridad para robots industriales. Parte 1: Robots. (ISO 10218-1:2011).
- **UNE-EN ISO 10218-2:2011** Robots y dispositivos robóticos. Requisitos de seguridad para robots industriales. Parte 2: Sistemas robot e integración. (ISO 10218-2:2011).
- **ISO 8373:2010** Robots manipuladores industriales - Vocabulario.
- **ISO 9283:2003** Robots manipuladores industriales – Prestaciones y métodos de ensayo relacionados.
- **ISO 9409-1:2004** Robots manipuladores industriales. Interfaces mecánicos. Parte 1: Platos (ISO 9409-1:2004).
- **ISO 9409-2:2003** Robots manipuladores industriales. Interfaces mecánicas. Parte 2: Ejes (ISO 9409-2:2002).
- **ISO 9946:1999** Robots manipuladores industriales. Presentación de las características (ISO 9946:1999). (RATIFICADA POR AENOR EN NOVIEMBRE 2000).
- **ISO 10218-1:2006** Robots para entornos industriales. Requisitos de seguridad. Parte 1: Robot (ISO 10218-1:2006) (Ratificada por AENOR en noviembre de 2007).
- **ISO 11593:1996** Robots manipuladores industriales. Sistemas de intercambio automático y efectivo. Vocabulario y presentación de características (ISO 11593:1996). (RATIFICADA POR AENOR EN NOVIEMBRE 2000).

- **ISO 14539:2000** Robots manipuladores industriales. Transporte de objetos con dispositivos de agarre tipo empuñadura. Vocabulario y presentación de características (ISO 14539:2000).
- **ISO/TR 13309:1995** Robots manipuladores industriales. Guía informativa de equipos de ensayo y métodos de metrología en operaciones para el funcionamiento de robots de acuerdo a ISO 9283.

3. Condiciones de carácter específico

El robot a utilizar deberá cumplir con las especificaciones presentadas en este proyecto:

- No debe superar los 150 kg de peso.
- El nivel de ruido no será mayor a 70 dB.
- El consumo de potencia en kW deberá ser menor que 0.5 kW a velocidad máxima.
- El robot debe poder utilizarse de forma manual o automática. En modo manual, este no se podrá usar desde equipos externos.
- Durante el modo manual es necesario que se disponga de un botón para habilitar el movimiento del robot mientras se mantenga pulsado.
- El robot debe cumplir las normativas expuestas en el apartado anterior.

4. Especificaciones de ejecución

El proceso deseado es que el robot recoja las piezas y, según su color y geometría las deposite en su lugar correspondiente. Debe cumplir las características siguientes:

- Las piezas pueden estar en diferentes posiciones a lo ancho de la cinta transportadora.
- Las piezas pueden estar orientadas de diferente forma a lo largo de la cinta.
- El sistema debe ser capaz de reconocer varias piezas a la vez y llevarlas a su posición correcta.

Para evitar accidentes, a los operarios no se les estará permitido acercarse a menos de tres metros de la zona de trabajo mientras el robot esté en funcionamiento.

Debido al tamaño de las piezas, el sistema utilizado para cogerlas debe ser tipo ventosa.

PRESUPUESTO

GRADO EN INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

Curso académico 2016-2017

SISTEMA PARA EL EMPAQUETADO AUTOMÁTICO DE JUGUETES DE PIEZAS ENCAJABLES

Índice

Precios descompuestos o unidades de obra					
1. Cuadro de precios elementales					
Ref.	**Ud.**	**Denominación**	**Precio (€)**		
Materiales					
m1	ud.	Brazo robótico industrial IRB 140 de la marca ABB	20000		
m2	ud.	Cámara CV-77 de la marca JAI	250		
m3	ud.	Sistema de visión integrado INFAIMON	700		
M.O.D					
h1	h.	Oficial 1ª programación de los equipos	40		
h2	h.	Oficial 1ª instalación de equipos eléctricos	35		
h3	h.	Oficial 1ª instalación de equipos mecánicos	35		
h4	h.	Oficial 2º mantenimiento	30		
2. Cuadro de precios descompuestos					
Ref.	**Ud.**	**Denominación**	**Precio (€)**	**Cantidad**	**Total (€)**
d1	ud.	Diseño e implementación de un sistema de empaquetado automático de piezas con distintas formas y colores dentro de un molde. El sistema está compuesto por el brazo robótico industrial IRB 140 de la marca ABB, por una cámara CV-77M de la marca JAI y por un sistema de visión integrado de la marca INFAIMON.			
m1	ud.	Brazo robótico industrial IRB 140 de la marca ABB	20000	1	20000
m2	ud.	Cámara CV-77 de la marca JAI	250	1	250
m3	ud.	Sistema de visión integrado INFAIMON	700	1	700
h1	h.	Oficial 1ª programación de los equipos	40	60	2400
h2	h.	Oficial 1ª instalación de equipos eléctricos	35	5	175
h3	h.	Oficial 1ª instalación de equipos mecánicos	35	6	210
h4	h.	Oficial 2º mantenimiento	30	4	120
	%	medios auxiliares sobre costes directos	10	23855	2385,5
				Total p.e.m	26240,5

3. Estado de mediciones					
Ref.	**Ud.**	**Partida**	**Cantidad**		
d1	ud.	Sistema de empaquetado automático	1		
4. Valoración					
Ref.	**Ud.**	**Denominación**	**Precio (€)**	**Cantidad**	**Total (€)**
d1	ud.	Diseño e implementación de un sistema de empaquetado automático de piezas con distintas formas y colores dentro de un molde. El sistema está compuesto por el brazo robótico industrial IRB 140 de la marca ABB, por una cámara CV-77M de la marca JAI y por un sistema de visión integrado de la marca INFAIMON.	26240,5	1	26240,5