UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Localización y manipulación de objeto en espacio 3D con un robot industrial

TFG

Grado en Ingeniería Electrónica Industrial y Automática

Universidad Politécnica de Valencia

ETSID | Escuela Técnica Superior de Ingeniería del Diseño

Fecha:12 de julio de 2016

Autor:

Oscar Lucas Villalón

Profesor Tutor:

Carlos Rocolfe Viala

ÍNDICE

1. RESUMEN 4

2. INTRODUCCIÓN 4

1.1. OBJETIVOS DEL PROYECTO 5
2.1. DISEÑO EXPERIMENTAL 5
2.2. SISTEMA DE VISIÓN ARTIFICIAL 6
2.3. ABB IRB 140 BRAZO ROBOT 7
2.3.1. Estación de computarización 7
2.3.2. Brazo robot 7
2.4. PLANTEAMIENTO TEÓRICO 8

3. ACONDICIONAMIENTO DE LA ESCENA Y ZONA DE TRABAJO 8

3.1. ILUMINACIÓN 9
3.2. ROBOT 10
3.2.1. Herramienta 10

4. SISTEMA DE VISIÓN ARTIFICIAL 12

4.1. CÁMARA 12
4.2. CPU VISIÓN 12
4.3. SET UP DE VISIÓN ARTIFICIAL 12

5. PRE-CONFIGURACIÓN DE LA CÁMARA 13

5.1. CAMERA CONFIGURATOR 13

6. ELECCIÓN DEL OBJETO 14

7. CALIBRACIÓN DE LA CÁMARA 15

7.1. CALIBRACIÓN ANEXOS 2D 15
7.1.1. Comprobaciones matriz de transformación homogénea Anexos 2D 16
7.2. CALIBRACIÓN EN 3D 18
7.2.1. Eje Z 18
7.2.2. Eje X – Y 21

8. ALGORITMO DE CONTROL SHERLOCK 25

8.1. INICIALIZACIÓN 25
8.2. COMUNICACIÓN ROBOTSTUDIO 25
8.3. PROGRAMA LOCALIZACIÓN 26
8.3.1. Bloque While 27
8.3.1.1. Obtener imagen 28
8.3.1.2. Tratar la imagen 29
8.3.1.3. Localizar y posicionar objeto 30
8.3.2. Bloque printf 32
8.3.4. Bloques script 32

9. PROGRAMACIÓN ROBOTSTUDIO 34

9.1. PREPARACIÓN ROBOTSTUDIO 34
9.2. DECLARAR VARIABLES 35
9.3. DECLARACIÓN DE LOS ROBTARGET 35
9.4. COMUNICACIÓN CON SHERLOCK 35
9.4.1. Reconocimiento del string 35
9.5. PROGRAMA RAPID 36

9.5.1. Ir al punto reposo del robot *36*
9.5.2. Ir al punto establecido por Sherlock *36*
9.5.3. Mostar pelota *37*

10. PRUEBAS **38**

10.1. MÉTODO DE OBTENCIÓN DE PARÁMETROS 38
10.1.1. Parámetros para la calibración en 2D *38*
10.2. PRUEBAS ROBOTSTUDIO 39

11. CONCLUSIONES **40**

12. REFERENCIAS **41**

1. ANEXOS **42**

2. PROGRAMA RAPID **43**

2.1. TIEMPO UTILIZADO PARA ELABORAR EL TFG 47

3. ESTUDIOS SEGURIDAD **48**

4. PARÁMETROS INTERNOS DE LA CÁMARA **49**

4.1. RESULTADO CALIBRACIÓN 49

5. MATRIZ DENAVIT HARTENBERG IRB 140 **50**

6. RESUMEN ERRORES **51**

6.1. CPU VISIÓN ARTIFICIAL 51
6.2. PROBLEMA CALIBRACIÓN ANEXOS 2D 51
6.2.1. Modificacion parametros de la camara *52*

1. RESUMEN

En este trabajo final de grado, se muestra una solución para localizar un objeto dentro de un espacio y cogerlo con un brazo robot. Para resolver este problema se ha utilizado el robot IRB 140 de ABB y un sistema de visión artificial.

En este TFG contiene los siguientes puntos de interés;

La calibración del sistema de visión artificial utilizada para resolver el problema y un breve estudio de alternativas.

Programa utilizado para localizar el objeto de estudio y la programación en RAPID[1]. Comunicación entre los dos equipos de trabajo.

Problemática de localización de objetos en el laboratorio, la utilización de brazos robot para la interacción con humanos y equipos del laboratorio.

2. INTRODUCCIÓN

Hoy en día la tendencia en el sector industrial está tendiendo a aumentar el número de robots o automatismos en los procesos de producción. La mayoría de los países desarrollados tienen plantas de producción en cadena con robots de línea. Determinados estudios comparan el número de robots que tienen los países para diferenciarlos entre ellos.

Se ha implantado la utilización de brazos robots fijos para que trabajen substituyendo en algunos casos a las personas que realizan tareas repetitivas, o que entrañan peligro. Estos brazos robot, cada uno con unas características, trabajan en jaulas o zonas acotadas para evitar accidentes.

Los robots trabajan siguiendo diferentes instrucciones/comandos dependiendo del lenguaje de programación. Realizan la tarea para la que han sido programados con las variables del entorno y sin interacción directa con el ser humano.

Recientemente se están incorporando nuevos periféricos de entrada para mejorar esa interacción. Uno de esos periféricos son las cámaras de visión artificial.

En la actualidad las empresas fabricantes de robots trabajan por minimizar los efectos dañinos en caso de colisión y reducir así los efectos de los accidentes. Un accidente con uno de estos robots no es viable. Por ello, sistemas que permiten evitar los accidentes y mejorar la interacción, se han convertido en el objetivo primordial de muchas empresas.

La prevención en empresas, cada vez más es importante. Técnicos especialistas analizan los posibles focos de accidente para intentar eliminarlos o reducir sus efectos.

En este TFG se ha realizado un método de localización que consigue mejorar la interacción entre el hombre y la máquina. Mejorando de esta manera los procesos

[1] RAPID – Lenguaje de programación de alto nivel para controlar robots industriales de ABB

donde las personas y las maquinas tienen que interactuar. También serviría en el caso de que el objeto entrara en la línea de producción de manera aleatoria.

1.1. OBJETIVOS DEL PROYECTO

En este trabajo final de grado se ha propuesto diseñar un sistema de interacción con un brazo robot utilizando una cámara industrial. La idea es utilizar la visión artificial para localizar la posición de objetos de interés y otros objetos/zonas que evitar.

El sistema consta de una etapa de visión artificial (realizada con Sherlock[2]) y una etapa computacional que interacciona entre el mundo real, el brazo robot.

Con todo ello se pretende que el sistema visualice e identifique un objeto mediante unos algoritmos de visión artificial. Detectado el objeto, el programa asigna la tarea correspondiente a dicho objeto y además determina su posición en la escena. El programa calcula la trayectoria para alcanzar el objeto evitando colisiones con el entorno. El algoritmo modifica en tiempo real la posición del brazo robot para realizar la tarea.

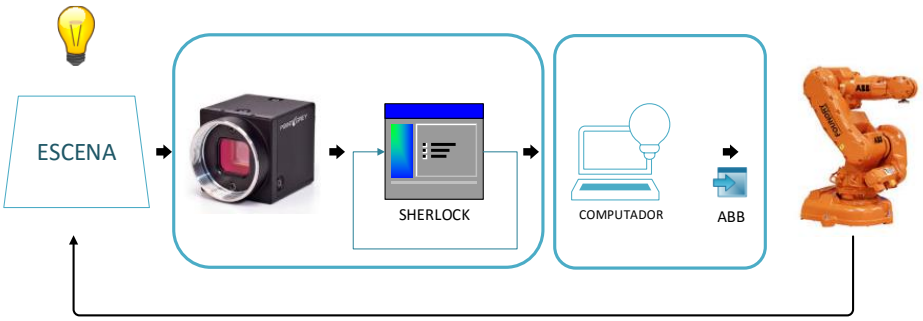


Figura 1 - Diagrama de flujo información

Para realizar este TFG se ha optado por realimentar y cerrar el lazo de control utilizando una cámara de visión artificial. Esto permitiría al sistema prever la colisión y de esta manera los posibles accidentes. Además de esto el sistema es capaz de localizar el objeto dentro de la zona de trabajo en las múltiples posiciones y transmitirla al robot.

2.1. DISEÑO EXPERIMENTAL

Este proyecto se propuso al tutor después de que él nos mostrara las posibilidades de los robots y la visión artificial. El ya disponía de un setup diseñado para proyectos similares. En el inicio del mismo se proporcionó un archivo llamado “Anexo” en el se describe como hacer un proyecto en el cual se trabaja en un plano en 2D. También se describe como realizar la comunicación con sockets. La información utilizada del documento se menciona en cada apartado.

El setup se introduce en la Ilustración 1. Durante este documento se describirá el funcionamiento de cada uno de estos equipos y se informara como se ha utilizado.

[2] Sherlock – Sistema de visión artificial diseñado para la industria.

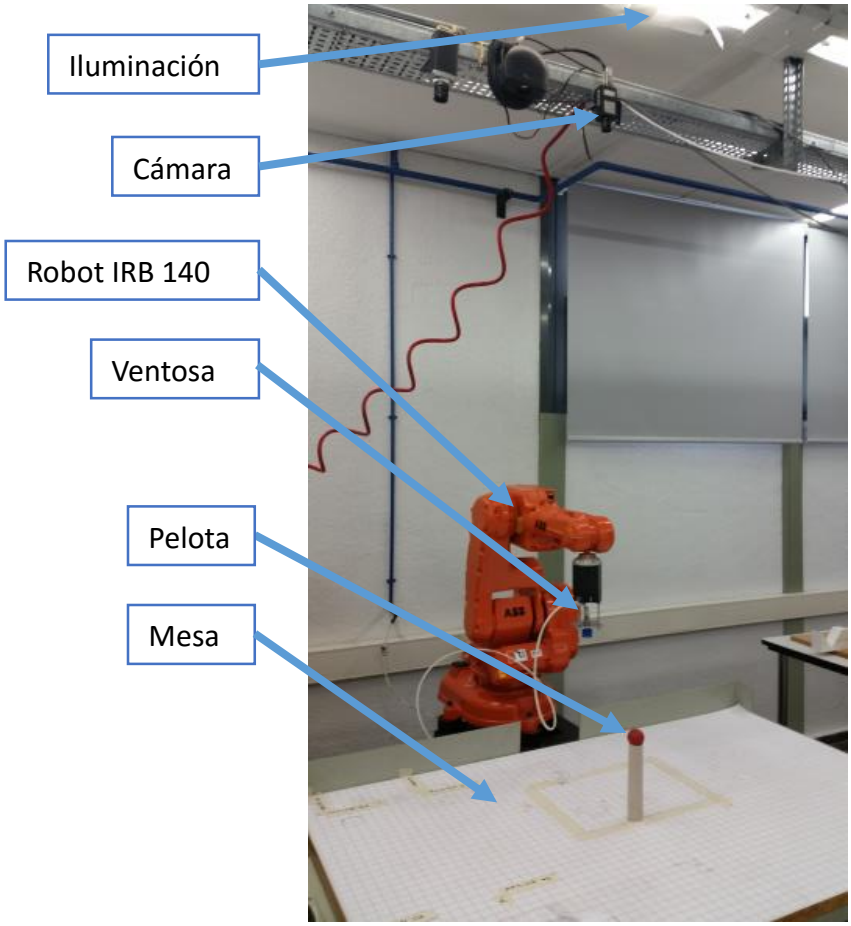


Ilustración 1 - Setup

2.2. SISTEMA DE VISIÓN ARTIFICIAL

El sistema de visión artificial consta de la CPU y la cámara. Estos equipos están ubicados a más de 2 metros sobre el suelo encima de la mesa. La CPU dispone de un cableado que no supera 1,5m de longitud lineales por ello se encuentra también en altura. Se detalla más información en el apartado 4.

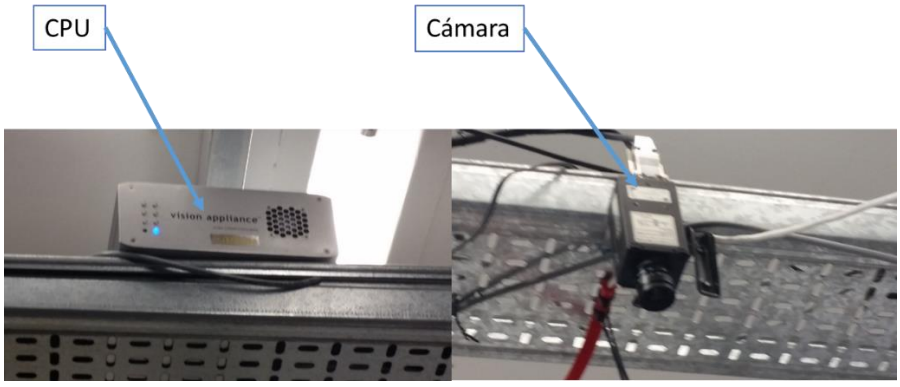


Ilustración 2 - Cámara y CPU

2.3. ABB IRB 140 BRAZO ROBOT

El Robot está compuesto por estructura de suportación con brazo robot modelo IRB 140 y estación de computarización tipo IRC5 M2004 Versión 3A3B.

2.3.1. ESTACIÓN DE COMPUTARIZACIÓN

Esta estación contiene el Flexpendant, que permite al operador de la maquina manipular el robot. También contiene la seta de emergencia, pulsador de arranque motores y selector manual automático.

Ilustración 3 - Estación de computarización.

Este quipo establece la comunicación entre en el usuario o programa y los diferentes actuadores del robot. Haciendo que sea más sencilla su utilización.

2.3.2. BRAZO ROBOT

El brazo robot, contiene los actuadores de la estación. También dispone de todos los elementos mecánicos necesarios para que el robot ejecute los movimientos con precisión.

Para que los encoders muestren una lectura fiable de la posición del robot, este dispone de unas marcas de calibración.

Ilustración 4 - Marcas de calibración

En la Ilustración 4 se muestran las marcas de calibración del robot. En concreto la marca de la articulación nº4. Estas marcas del robot se han utilizado para saber cuándo el robot estaba retorciéndose más de 180º y evitar que esto ocurra.

2.4. PLANTEAMIENTO TEÓRICO

Para afrontar este reto y conseguir cumplir los objetivos del proyecto. Se han utilizado los conocimientos adquiridos en las asignaturas a lo largo del Grado.

El planteamiento inicial del proyecto está formado por seis temas de estudio.

- Acondicionamiento de la escena y zona de trabajo.
- Sistema de visión artificial.
- Pre-configuración de la cámara.
- Elección del objeto.
- Calibración de la cámara.
- Algoritmo control Sherlock.
- Programar en RobotStudio.
- Pruebas.

Cada uno de estos estudios ha requerido una preparación posterior en el laboratorio para comprobar que efectivamente cumplía los requisitos iniciales. La supervisión del tutor durante estas comprobaciones ha ayudado a orientar el trabajo y acotar el problema desde diferentes perspectivas. A partir de las cuales se han tomado decisiones para ejecutar el proyecto.

3. ACONDICIONAMIENTO DE LA ESCENA Y ZONA DE TRABAJO

Acondicionar la zona de trabajo tiene como objetivo conseguir adquirir una imagen con la que poder trabajar posteriormente. De la adquisición depende casi el 90% del éxito de este trabajo. También pretende acotar el espacio utilizado por el robot.

El lazo de control se ha cerrado utilizando la cámara 'Figura 1 - Diagrama de flujo información'. Por lo tanto la escena y sus variables son determinantes para el sistema funcione correctamente. Cualquier variación en los parámetros de la escena generaría un error de cálculo en la posición del objeto. Estos errores han ocurrido y se han estudiado en el apartado 7.2.

3.1. ILUMINACIÓN

Para acondicionar la zona de trabajo se ha intervenido sobre la fuente de luz directamente.

En el laboratorio se dispone de focos de luz fluorescentes. El principal Problema de estos focos de luz es el parpadeo y al ser focos sin pantalla, la posible generación de sombras en la escena.

Para resolver estos problemas se han instalado unas tiras de papel directamente en los focos en forma de difusores. De esta manera se consigue eliminar la direccionalidad del foco, atenuar los brillos en la imagen y filtrar los parpadeos.

Además la variable de luminosidad que podíamos tener durante el día o la noche se ha eliminado cerrando la claraboya del laboratorio. Muchas de estas mejoras ya estaban implantadas. Ilustración 5.

Ilustración 5 - iluminación

Con los medios aportados y disponibles en el laboratorio esta es la mejor solución que se ha podido adoptar. La técnica de luz difusa uniforme hubiera sido la más adecuada para iluminar la zona de trabajo aumentando la calidad de la imagen considerablemente.

3.2. ROBOT

El robot utilizado para realizar este TFG es el ABB IRB 140, proporcionado por la Universidad Politécnica de Valencia. Es un robot de seis ejes de libertad motorizados capaz de mover 5Kg a una distancia máxima de 810mm. El robot esta inspirado en la anatomía de un brazo humano, posee 6 articulaciones de revolución, que le dan una movilidad muy similar a este.

Este robot está anclado en un puesto de trabajo enfrente a la mesa. Los dos juntos forman la escena.

La matriz de transformación entre el robot y la cámara influye en la posición de recepción del objeto. El anclaje del robot y de la cámara es muy importante. Pues la calibración establece una relación entre estos dos orígenes de coordenadas.

Los parámetros del robot y otros cálculos realizados durante el TFG pero no relevantes para el mismo están en el 5 de los anexos.

3.2.1. HERRAMIENTA

Los robots en si mismos no disponen de herramienta, todos ellos vienen montados hasta el punto de muñeca. Es en este punto del robot es donde se instala la herramienta.

Los robots disponen de infinidad de herramientas asignadas para hacer diferentes trabajos. Lo normal en estos casos es estudiar la aplicación del robot, y posteriormente comprar o diseñar una herramienta para que la ejecute correctamente.

En este proyecto se ha proporcionado una herramienta llamada ventosa. Esta herramienta de geometría simple. Se ha montado al punto de muñeca cada día que se accedía al laboratorio. A continuación se muestra la ventosa de la herramienta.

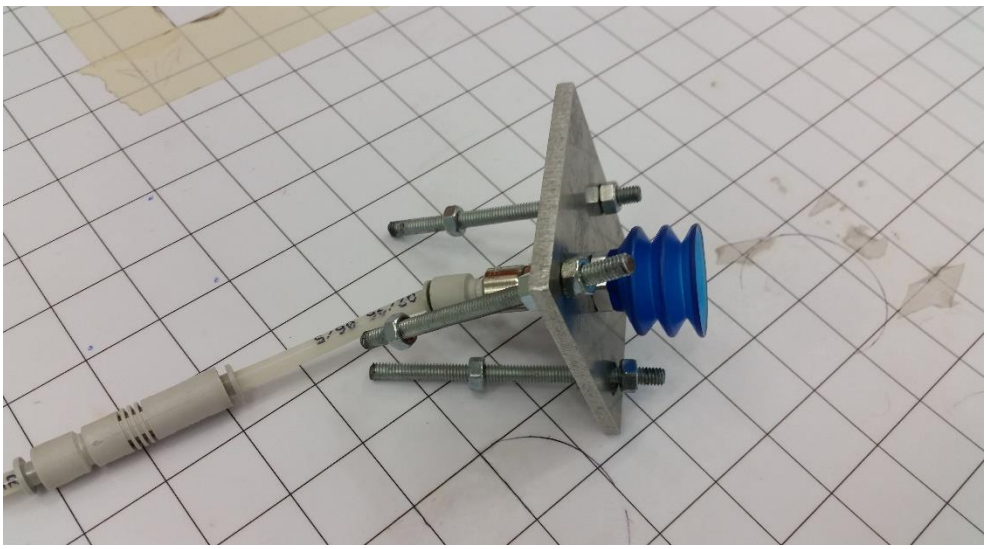


Ilustración 6 - Herramienta ventosa.

En la Ilustración 6 se muestra la ventosa de color azul. Esta ventosa permite mediante el efecto Venturi succionar el aire de la parte delantera. Con esta succión y la

estanqueidad que proporciona la ventosa con la superficie del material a succionar, se produce un agarre que permite mover objetos.

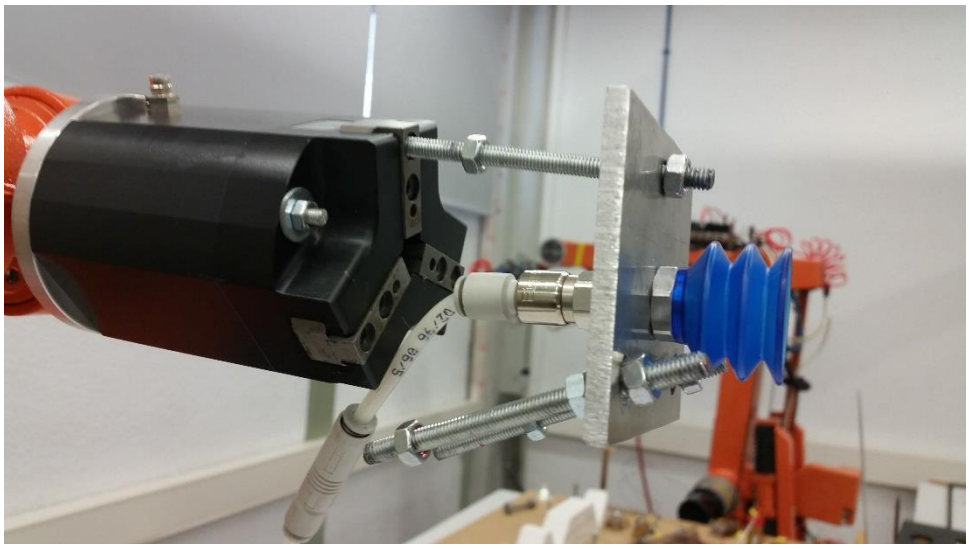


Ilustración 7 - Montaje herramienta

Pero esta herramienta nos proporcionó dos problemáticas. Una es la que observa en la Ilustración 7.

El montaje es mecánico y requería de muchos ajustes rudimentarios para un correcto calibrado, era complicada la replicación exacta del montaje.

Ilustración 8 – Pelota

La otra problemática la proporciono la pelota que se observa en la Ilustración 8. Esta pelota de color rojo, es fácilmente apreciable por el algoritmo de Sherlock, esto permite localizarla sin problemas, además tiene unas dimensiones y peso adecuado para la herramienta “ventosa”. Lo que no resulta tan adecuado es rugosidad, porque esta reducía la capacidad de succión de la ventosa. Por ello la pelota fue modificada aplicando un tratamiento de lijado. Que se puede apreciar mínimamente en la ilustración.

4. SISTEMA DE VISIÓN ARTIFICIAL

Los trabajos relacionados con la cámara son muy importantes, pues estos son la base del proyecto. Una modificación en alguno de los parámetros iniciales de la cámara genera la necesidad de volver a realizar todos cálculos necesarios para la calibración. En primer lugar se inicia el estudio conociendo la cámara y la CPU que se va a utilizar.

4.1. CÁMARA

La cámara utilizada para realizar el TFG es la cámara Modelo CV-M77. La cámara dispone de una óptica regulable. Con una distancia focal de 8mm. Esta se ha ajustado manualmente hasta que la imagen se veía nítida sobre la mesa.

Esta cámara proporciona una imagen de video de 1028(h) x 770(v) que es la obtenida en la pantalla del computador. Sobre esta imagen será sobre la que se realizan todos los filtrados y cálculos. Consiguiendo así transmitir la posición del objeto al robot.

Puesto que la cámara está anclada en una posición correcta no se modificó su posición.

4.2. CPU VISIÓN

La CPU del sistema de visión artificial, es un equipo destinado a obtener la imagen de la cámara y procesarla con el sistema operativo que posee. Este sistema operativo es Windows 95.

La CPU dispone de un puerto E/S, un teclado como periférico de entrada y un cable de red para acceder al equipo.

4.3. SET UP DE VISIÓN ARTIFICIAL

Para realizar la adquisición de las imágenes es necesario disponer de un entorno controlado, una cámara digital, tarjeta de adquisición de video y un programa para computarizar la información. Este sistema está formado por los dos apartados mencionados en los puntos anteriores.

En la Ilustración 9 vemos el setup dispuesto. Para utilizar el equipo se requería que se cumplieran las normas de seguridad del laboratorio y las de procedimiento. Anexo 3.

Ilustración 9 – Setup

Para acceder remotamente al equipo de visión artificial se hacía a través del escritorio compartido y la red Wifi de la UPV.

5. PRE-CONFIGURACIÓN DE LA CÁMARA

Para conseguir obtener una imagen nítida y poder distinguir los objetos es vital pre-configurar los parámetros internos y externos de la cámara. Modificando estos parámetros se obtiene un histograma diferente para cada imagen. Estos parámetros se han configurado desde el escritorio del equipo de Sherlock.

5.1. CAMERA CONFIGURATOR

El equipo tiene instalado un programa para modificar estos parámetros internos de la cámara. 'Camera configurator'. Este es un paso vital de la calibración. Cualquier modificación de estos parámetros implicaría volver a repetir el procedimiento.

Modificando parámetros de la cámara se obtiene una imagen con diferentes características.

De los parámetros internos de la cámara se modificó el contraste y el brillo. Manteniendo el tamaño de imagen. El resultado de la calibración se puede observar la siguiente ilustración.

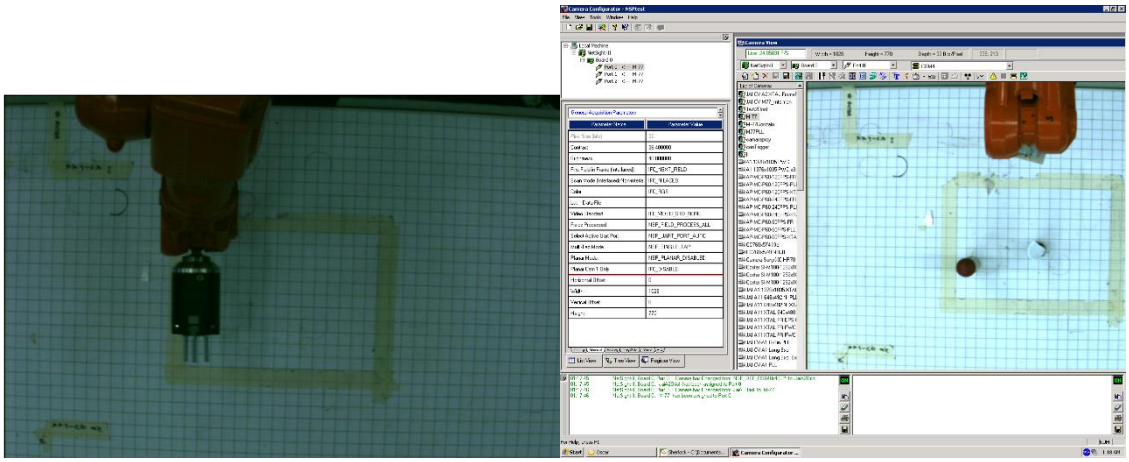


Ilustración 10 - Resultado pre-configuración

La imagen izquierda de la Ilustración 10, es la imagen que obtuve inicialmente sin modificar ningún parámetro y la de la derecha la imagen obtenida después de la modificación de los parámetros. De esta manera se obtuvo un mayor contraste.

Con estos ajustes, se puede observar mejor los objetos de diferente color al fondo de la imagen (mesa de color blanco con una cuadricula).

El resultado de las modificaciones se puede observar en estos histogramas.

A continuación se muestran los histogramas. En el histograma de la parte superior los colores rojo, azul y verde tienen una frecuencia de aparición en la imagen repartida en diferentes tonos, por lo tanto la imagen no es nítida.

En el histograma de la parte inferior se ven todos los colores agrupados y de mayor frecuencia en los claros. Por lo que, al colocar un objeto de color oscuro se vería reflejado aumentando la frecuencia de tonos del color del objeto.

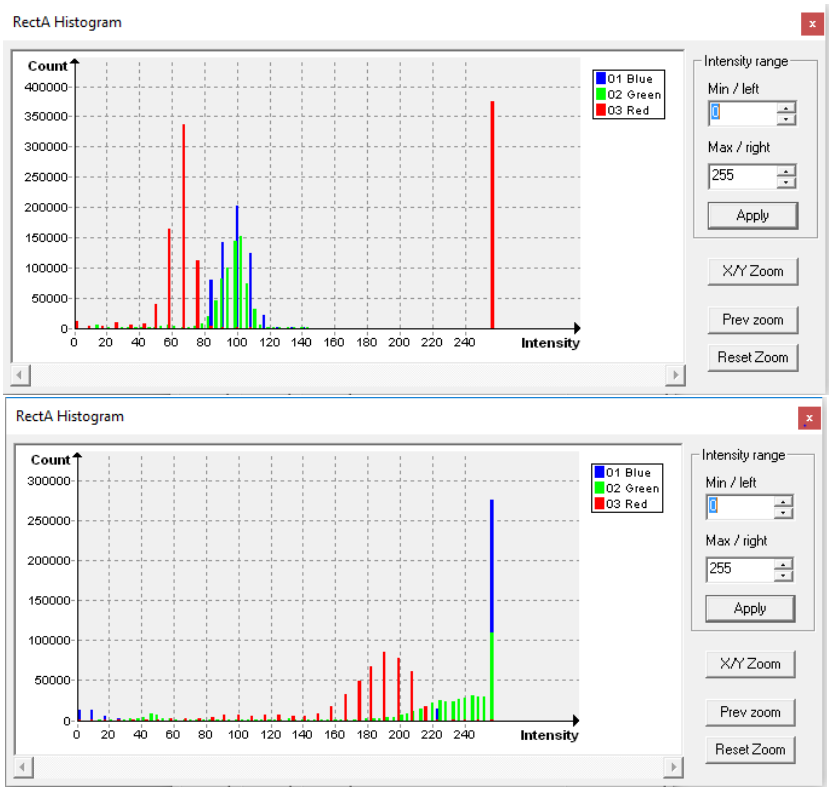


Ilustración 11 - Histogramas pre-configuración

6. ELECCIÓN DEL OBJETO

A continuación, se procedió a la elección de la pelota para el estudio. Para ello, se probó entre dos pelotas disponibles, una pelota de tenis y una pelota roja, proporcionadas por el profesor. Se colocaron en la escena y se obtuvo siguiente el histograma.

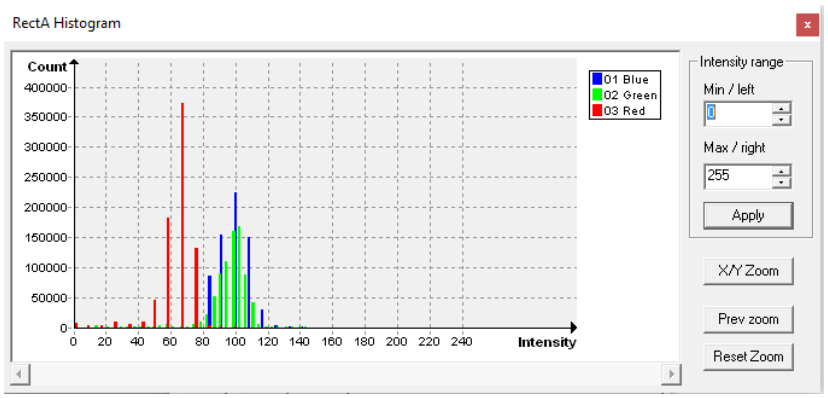


Ilustración 12 – Histograma pelotas

En este histograma se puede ver como el rojo se puede aislar con un filtro y se podría separar de los otros colores. Puesto que las cámaras observan los colores RGB y solo se disponía de dos pelotas una amarilla y otra roja. Se decidió utilizar la roja ya que

la amarrilla presentaba dos inconvenientes, el primero es que era demasiado clara y costaba separarla de los colores claros de la mesa, el segundo es que la pelota amarilla requería una herramienta diferente para cogerla, de la cual no disponía.

7. CALIBRACIÓN DE LA CÁMARA

Para obtener la posición del objeto en el plano, es necesario conocer la relación entre el origen de coordenadas de la cámara y el origen de coordenadas del Robot. Durante el TFG se han realizado diferentes tipos de calibración. Cada uno de ellos se ha estudiado por separado.

Para realizar todas estas calibraciones se ha utilizado el algoritmo de calibración de Sherlock que proporcionaba las variables a tiempo real de;

- Área
- Posición X
- Posición Y

7.1. CALIBRACIÓN ANEXOS 2D

Un manual de usuario fue proporcionado al inicio del proyecto por el profesor tutor llamado “Anexos”.

El proceso de calibración del manual consiste en mediante un patrón de calibración y el programa Sherlock. Localizar la posición X Y de cada uno de los círculos del patrón en Sherlock y en el FlexPendant.

Mediante un algoritmo de calibración programado en Sherlock (10.1.1) que reconoce los puntos del patrón Figura 2 y se asigna un valor leído de la cámara a un valor real del robot.

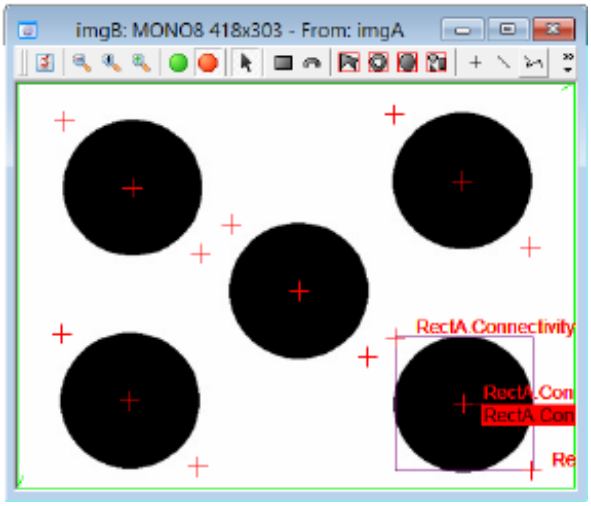


Figura 2 - Patrón calibración.

Junto con el manual se adjuntaba un fichero para poder realizar una matriz de transformación homogénea ”homografía”. Los puntos reales y de la cámara anotados en forma de matriz, en este algoritmo, proporcionan como resultado una matriz de transformación.

El resultado de esta calibración es una matriz de transformación que permite obtener la posición de un punto en el plano 2D próximo al patrón.

$$\begin{bmatrix} x_h \\ y_h \\ H \end{bmatrix} = \begin{bmatrix} -0.469 & 0.753 & 427.054 \\ 1.273 & 0.195 & -573.307 \\ -0.0009 & -0.0006 & 1.264 \end{bmatrix} * \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Figura 3 - Ecuación matricial

Con esta matriz de transformación y realizando unos cálculos se podía estimar la posición de un objeto en el plano.

Cuando se iniciaron las pruebas con los datos obtenidos, se comprobó que la calibración solo era precisa en la zona donde se colocaba el patrón y que también era vulnerable a modificaciones del eje Z. Por lo tanto no podíamos poner la pelota en otro punto de la mesa que fuera el dispuesto previamente con el patrón.

A continuación se procedió a tomar como patrón toda la mesa. De esta manera se creía que el patrón sería más fiable y podríamos poner la pelota en todas partes de la mesa. Pero los resultados de las pruebas realizadas mostraban errores y problemas para cambiar de plano positivo a plano negativo del eje de coordenadas del robot. Lo mismo pasó al incrementar el número de puntos en el plano de la mesa.

A partir de estos resultados se realizaron unas comprobaciones.

7.1.1. Comprobaciones matriz de transformación homogénea Anexos 2D

Para conocer mejor el problema se realizó una calibración de cuatro puntos en cruz sobre la mesa. El objetivo del estudio es conocer cuál es el problema de calibrar sobre el plano.

En la siguiente tabla están los puntos obtenidos con Sherlock a la izquierda, y a la derecha, los puntos obtenidos con el robot en la realidad. Estos puntos se han obtenido de manera experimental detallado en el apartado 10.1.1.

Puntos	Sherlock		Robot	
0	254	374	672,4	- 246,3
1	510	188	479,8	13,4
2	720	376	671,2	226,6
3	498	496	778,2	1,2

Tabla 1- Relación entre robot y Sherlock 2D

A continuación se muestra los mismos puntos, en un gráfico. En este grafico se pude apreciar que pese a que la distancia entre el centro y los puntos de los laterales es la misma. El resultado de la matriz de transformación da como resultado que no es igual la geometría de la figura. Pues esta debería mantenerse después del procesado matemático.

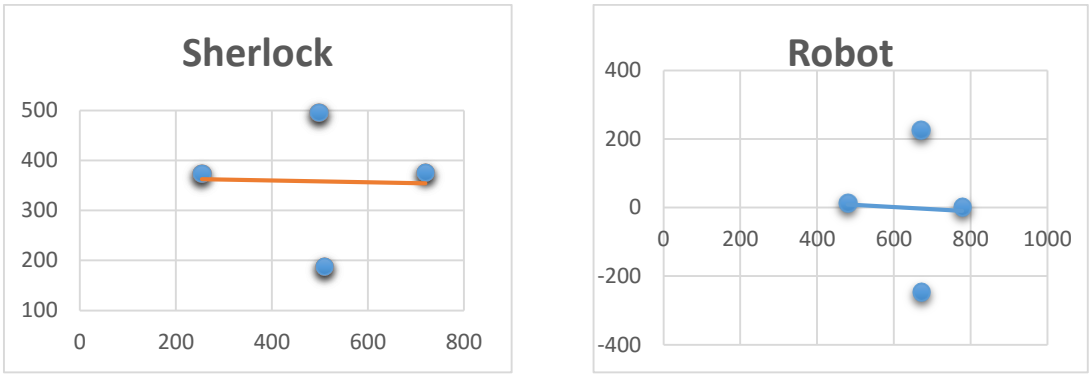


Figura 4 - Resultado calibración 2D

Este resultado explica como la calibración de todo el plano de visión de la camara no es precisa. Muestra errores que aumentan al aumentar el tamaño de la superficie. Por estos motivos se desestima este metodo de calibración. Mas informacion en el apartado anexo 6.2.

7.2. CALIBRACIÓN EN 3D

Para realizar el calibrado de la cámara en 3D se propuso una solución. Obtener la posición del objeto analizando como varían los parámetros obtenidos con la cámara y mediante un cálculo geométrico. A partir del área obtenida en Sherlock se realizar el cálculo de la altura. También el cálculo de la desviación en X e Y.

Este estudio se realizó después los estudios en 2D. Durante la calibración en 2D se obtuvieron algunos conocimientos que se han utilizado en la calibración en 3D. Estos conocimientos se muestran a continuación.

7.2.1. EJE Z

Analizando el entorno y observando las variables que obteníamos de Sherlock se utilizó el área de la pelota obtenida del algoritmo de Sherlock del apartado 8 para conocer la altura de la pelota.

El algoritmo de Sherlock nos proporciona la el área de la pelota en pixeles, este área varía en función de la altura de la pelota.

- Análisis del área en el eje de la cámara.

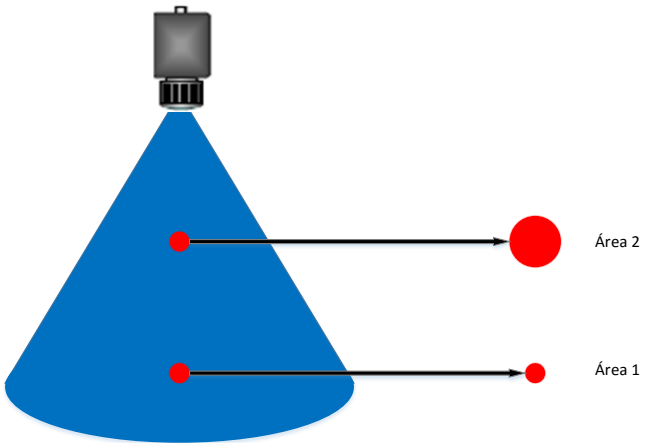


Figura 5 - Variación del área en función de la altura

En la Figura 5 se puede observar el resultado de modificar la altura de la pelota. En la cámara se obtiene una imagen como la de la derecha. La misma pelota con áreas diferentes en la imagen.

Área 2 > Área 1

El resultado de variar la altura es que el área de la pelota aumenta cuando esta aumenta. Esto establece una relación lineal entre el área y la altura. Siendo el área mayor cuando la pelota está más cerca de la cámara.

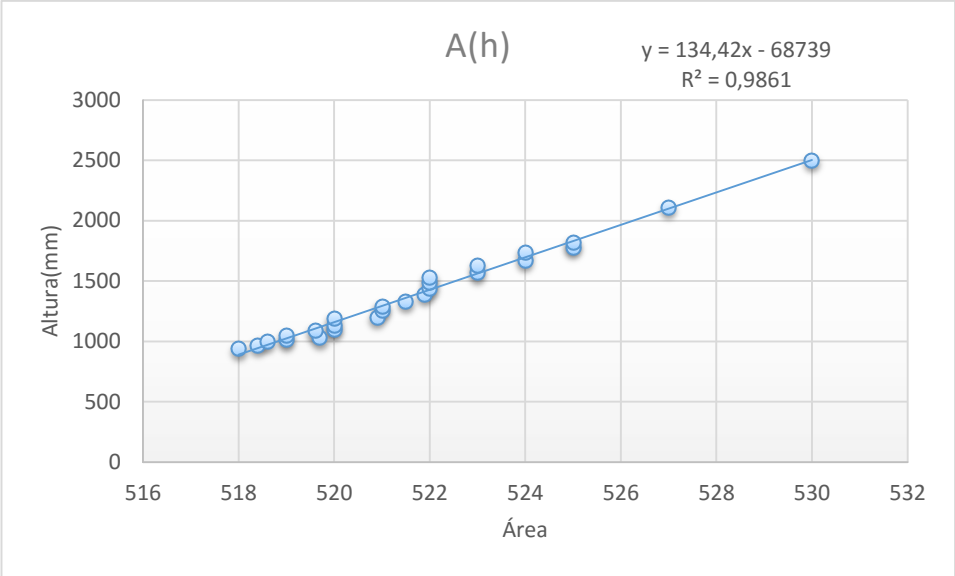


Figura 6 - Relación Altura y área.

En la Figura 6 se puede observar el resultado de la obtención de 26 parámetros obtenidos experimentalmente en el laboratorio. Con esta calibración se cambia de pixeles ”Área” a la variable Z del robot. Siendo está, relativamente precisa ya que el coeficiente de determinación R^2 está próximo a 1.

$$h_{(A)} = 134{,}42 * h_{(S)} - 68739$$

Realizando este estudio, se observó que el área de la pelota también variaba a la misma altura sobre la mesa, pero con diferente posición X-Y. Finalmente una vez realizado el estudio no se incluyó en el algoritmo pues el error en el centro de la imagen era muy pequeño.

- Estudió del área en diferentes puntos de X-Y con la misma altura.

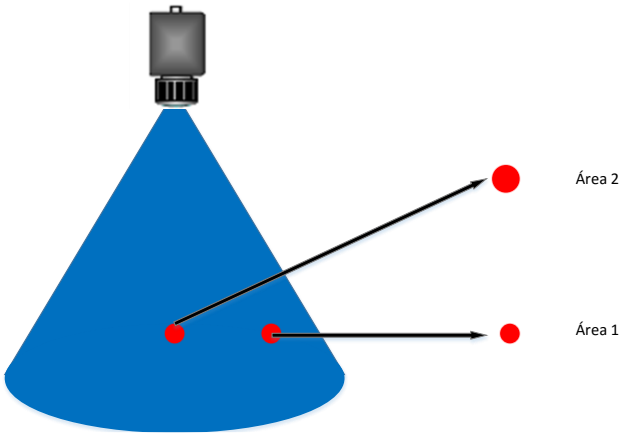


Figura 7 - Variación área X-Y

En la Figura 7 se observa la problemática. Pese a que la pelota se encuentra a la misma altura, el área de la misma es diferente.

Area1 < Área2

A raíz de esta problemática se analizó el problema. Se realizó un estudio sobre el eje X sin variar la altura.

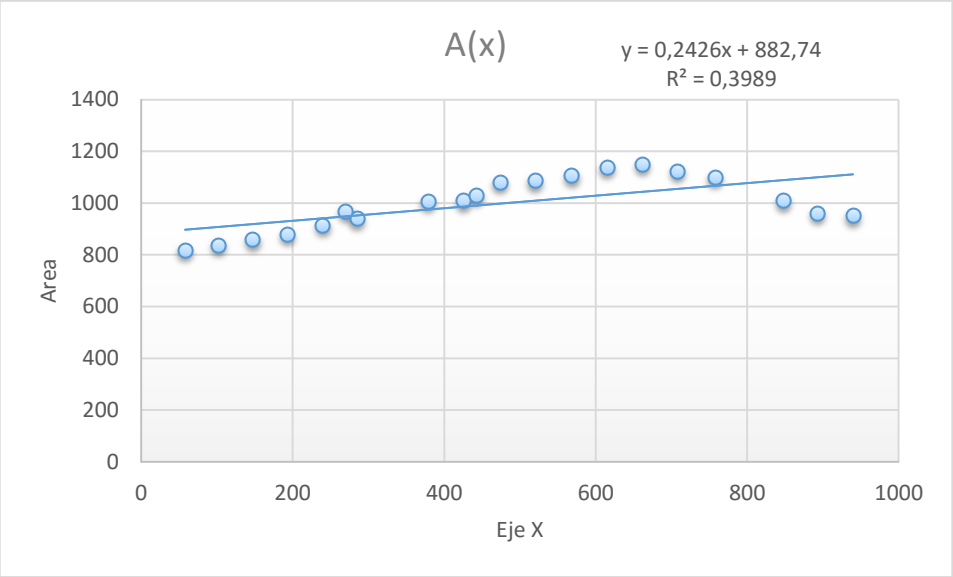


Figura 8 - Relación entre la altura y la posición X

En Figura 8 se observa como variaba el área en función de X. Con esta grafica podemos observar el comportamiento del área de la pelota en la recta X.

En este estudio se determinó que debido a que la lente de la cámara era esférica, la imagen se veía deformada modificando así la percepción de la pelota. Esto cambió el enfoque. Con la cámara lo que realmente obteníamos, era la distancia entre la cámara y la pelota. Esta relación se establecía como un péndulo en el que la lente de la cámara es el vértice.

Se realizó el mismo estudio sobre el Eje Y. Se dispuso una altura y se varió la posición Y.

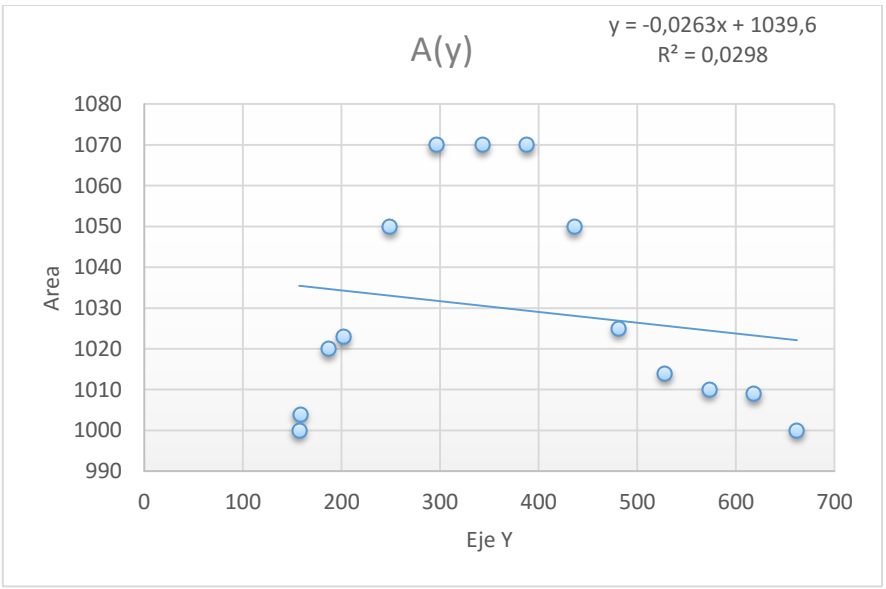


Figura 9 - Relación entre la altura y la posición Y

En este caso el coeficiente de determinación establecía que el modelo no era determinante ya que los puntos estaban muy dispersos.

En este estudio se llegó a la conclusión que la variación del área respecto al eje X e Y era muy pequeña en el eje de la cámara. Por lo tanto se despreció el efecto sobre el resultado final, siendo en el peor de los casos 50 mm en las esquinas de la mesa. En el centro de la mesa el error era muy pequeño.

7.2.2. Eje X – Y

La posición Y e Y de la cámara, está ligada a el centroide de la pelota. De este se sacan estos dos parámetros (Vars.X y Vars.Y). Se han utilizado haciendo una simple conversión y corrigiendo el offset.

El eje X de la cámara era el eje Y del robot y viceversa, este cambio de referencia se ha hecho durante la calibración.

En la siguiente tabla se adjunta los datos obtenidos para realizar la conversión. Obtenidos experimentalmente.

X Robot	Y Robot	X Sherlock	Y Sherlock
455,2	146,2	638,06	170,8
513,4	62,7	558,1	224,85
573,1	-19,8	477,9	278,91
683,1	261,9	746,21	385,75
602	321,4	802,78	310,14
683,1	380,6	856,49	387,05

Tabla 2 - Datos calibración eje X e Y

En las siguientes ilustraciones se muestran las gráficas que relacionan ambos ejes de coordenadas. Dando como resultado la transformación de un punto en Sherlock a el sistema de coordenadas del Robot.

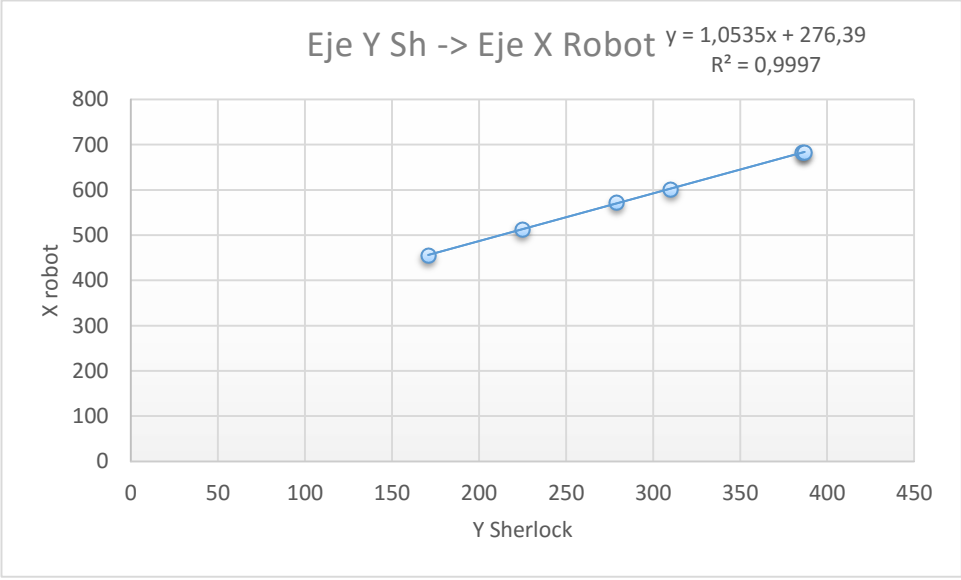


Ilustración 13 - X Mundo

A continuación se muestra el mismo grafico pero para el eje Y del robot.

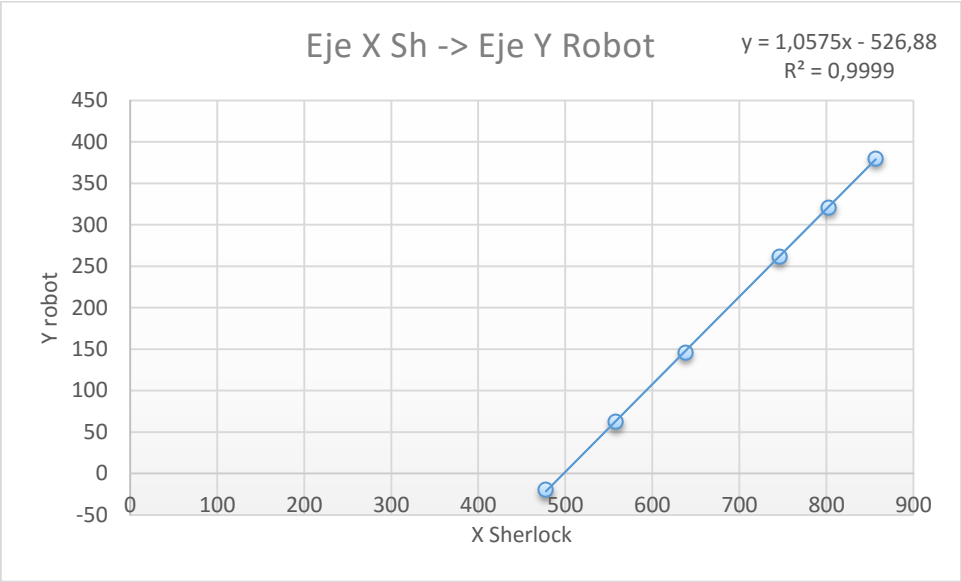


Ilustración 14 - Y Mundo

Durante los experimentos anteriores se obtuvo como conclusión que el centro de la pelota proporcionado por el algoritmo de Sherlock, formaba una recta con infinitas intersecciones posibles.

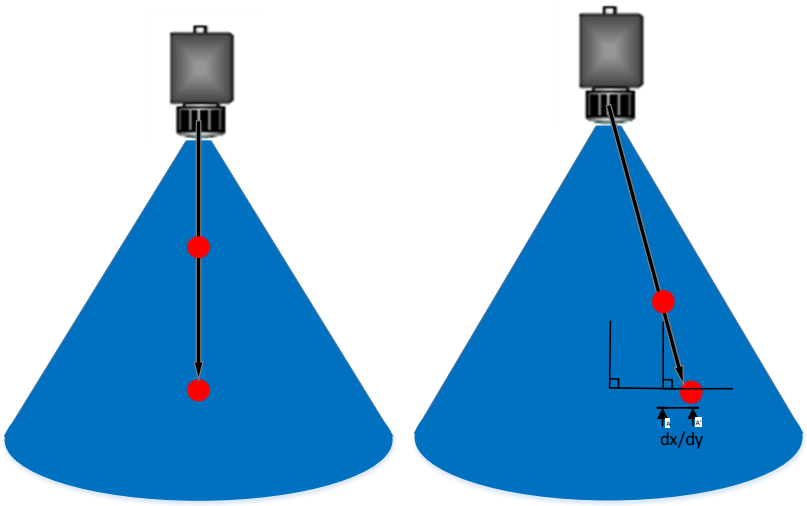


Figura 10 - Ajuste ejes X e Y

En la Figura 10 podemos ver como la pelota se encuentra en la misma posición X-Y para diferentes alturas. Esto genero un problema, en el eje de la cámara (imagen de la izquierda) no implica mayores quebraderos de cabeza pues el centro de la pelota está en la misma posición X-Y para todas las alturas. El problema viene cuando se modifica la posición y salimos del eje de la cámara (imagen de la derecha). En este momento se crea un dx y un dy, entre el eje de la cámara y la pelota en función de la altura.

Para conocer la posición X e Y de la pelota con la altura, basta con intersectar la altura con la recta X-Y del plano corrigiendo el desfase producido por la altura en X e Y.

- Estudio de la posición X en función de la altura.

Para realizar este estudio se mantuvo una posición X fija y se varió la altura.

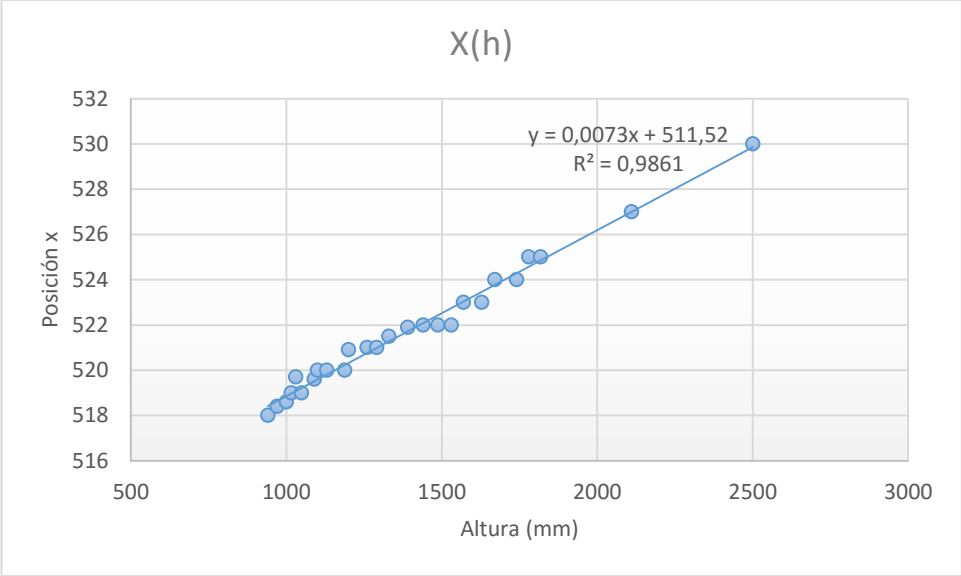


Figura 11 - eje X en función de la altura

- Estudio de la posición Y en función de la altura.

Para realizar este estudio se mantuvo una posición Y fija y se varió la altura.

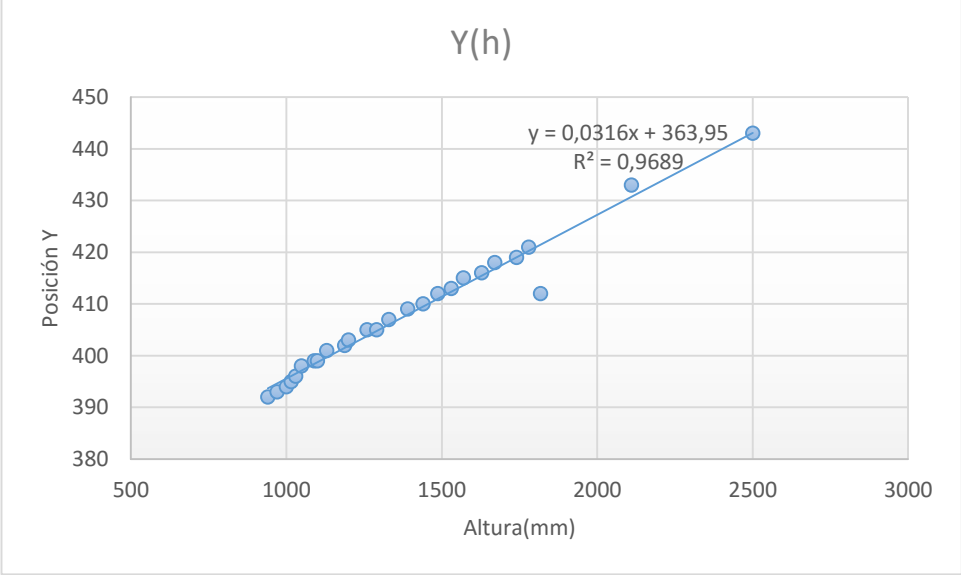


Figura 12 - eje y en función de la altura

Con los parámetros obtenidos de la cámara mediante el algoritmo de Sherlock se consigue obtener la posición del objeto dentro de un área determinada sobre la mesa. Siendo el coeficiente de determinación es válido en ambos estudios.

A continuación se encuentra el código del Script 2 Sherlock, en el están estas ecuaciones. Estas ecuaciones son las obtenidas de los parámetros de calibración mencionados en estos apartados.

Parametros de posicionamiento X e Y.

- Corrección en función de la altura en los ejes.

Vars.xh=0. 0073*Vars.Area_objeto+511,52;

Vars.yh=0. 0316*Vars.Area_objeto-363.2;

- Se resta la corrección.

Vars.x1=Vars.X-Vars.yh;

Vars.y1=Vars.Y-Vars.xh;

- Se cambia la coordenada X por la Y e viceversa y transforma de pixeles a coordenadas del robot.

Vars.Xmundo=1.0684*Vars.y1+273.06+20;

Vars.Ymundo=1.053*Vars.x1-522.56-10;

Los parámetros obtenidos se utilizan en un script de Sherlock en el apartado 8.3.4.

Parámetros Área , Altura.

- Cambio de Arte a la variable Z del robot.

Vars.Ah=134,42*Vars.Area_objeto-68739;

- Se añade 200 u de altura del robot, esta corresponde a la longitud de la herramienta en ese eje.

Vars.Zrobot=Vars.Ah+Vars.Ax-Vars.Ay+200;

8. ALGORITMO DE CONTROL SHERLOCK

En el mercado hay diferentes herramientas de procesamiento de imágenes. Cada una con diferentes características y todas ellas válidas para realizar el reconocimiento de objetos.

En este trabajo se ha utilizado Sherlock. Esta herramienta incluye algoritmos de filtrado y búsqueda utilizados en este proyecto. Puesto que el proyecto utiliza los algoritmos ya implementados en el software, no se hablara de su diseño. Pero si de cómo han sido utilizados y porque. También se ha establecido la comunicación con este Software.

Al inicio del proyecto se utilizó la guía "Anexo" para realizar la comunicación con sockets. La configuración de la comunicación viene detallada en el documento. Así que la mencionare brevemente. El programa de localización y los Scripts se realizaron con la experiencia obtenida durante el grado, en concreto la asignatura de Visión artificial.

8.1. INICIALIZACIÓN

Para realizar el programa empezamos por la inicialización de las variables. Para este programa son las mostradas a continuación en la Ilustración 15. Todas las variables se inicializan a 0.

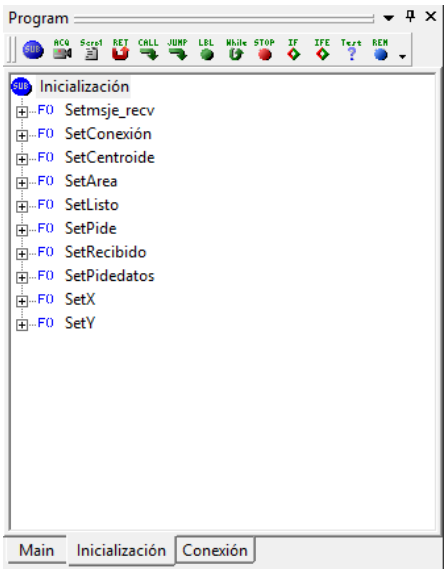


Ilustración 15 – Inicialización

Con los parámetros inicializados, se puede proceder ejecutar la instrucción Conexión en Sherlock.

8.2. COMUNICACIÓN ROBOTSTUDIO

En la guía explica como configurando los equipos Sherlock y Rapid, puedes establecer la comunicación e intercambiar variables.

En los primeros pasos se utilizó la configuración determinada en la guía para el conexionado de la tarjeta de entradas y salidas. Pues no era necesario modificar la configuración existente Ilustración 16 - Configuración E/S.

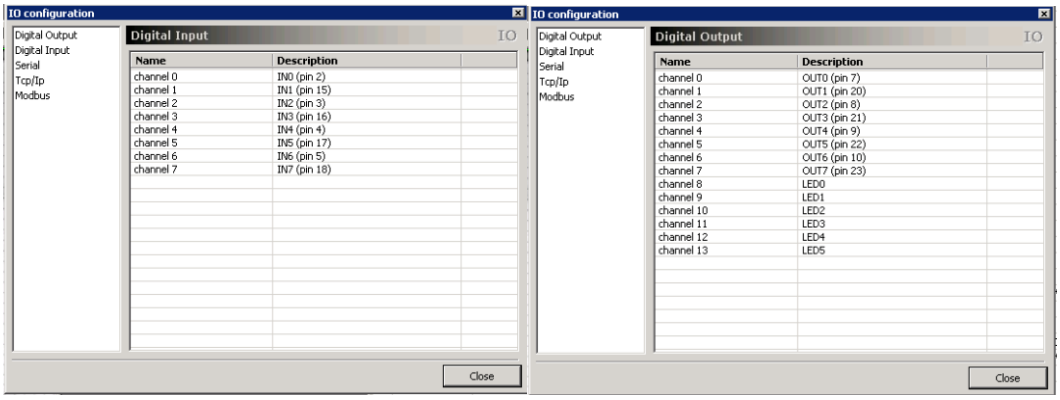


Ilustración 16 - Configuración E/S

Lugo se configuro el puerto serie Tcp/Ip, estableciendo la comunicación a través de la red. Para realizar esta comunicación se utilizó la IP 158.42.16.207 Puerto 1027.

El resto de la programación se realizó según las instrucciones del "Anexo" C.3 Comunicación con RobotStudio.

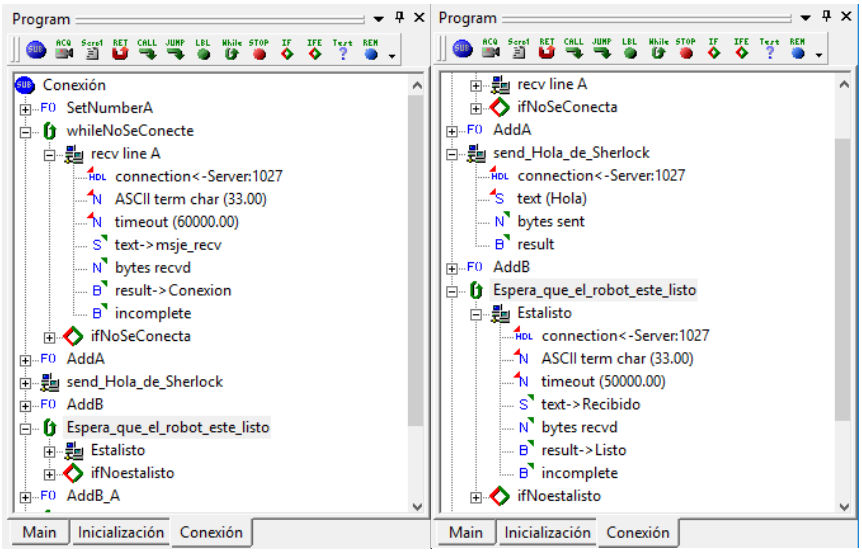


Ilustración 17 – Comunicación Sherlock

En la Ilustración 17 se muestra el proceso de conexión. En el proceso de conexión se establece la sentencia de iniciar la comunicación con RobotStudio. Esperando a recibir un mensaje de confirmación, "Msje_recv". A continuación se envía la palabra Hola y se espera la recepción por parte del robot. A continuación el software del robot espera la información que se envía un el String.

8.3. PROGRAMA LOCALIZACIÓN

Para localizar la pelota se ha utilizado la configuración de la cámara establecida mediante todos los cálculos teóricos del apartado 7.2. Si alguno de los parámetros de la calibración de la cámara, se modificaran, se deberían volver a realizar dichos cálculos y modificar este programa.

Para localizar la pelota en tiempo real, se ha utilizado la configuración de 'imagen Cámara', que proporciona la imagen a tiempo real de la cámara. Aunque en casa se ha trabajado con fotografías, que es lo que permite la versión de prueba.

Para describir la programación con el software Sherlock, se resumirán determinados pasos que cualquier operario que utilice este software sería capaz de hacerlos por sí mismo.

A continuación se muestra el bloque Main, donde se puede observar la estructura general del programa de visión artificial.

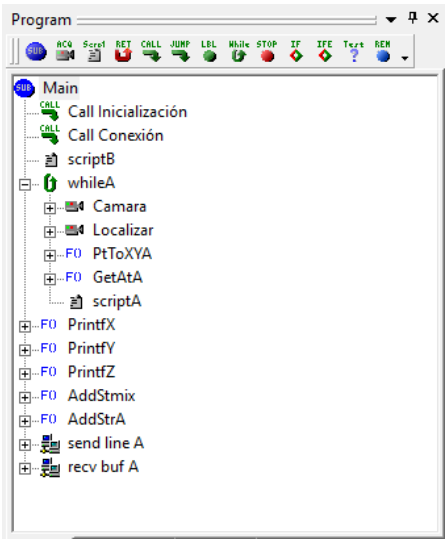


Ilustración 18 - Programa visión artificial

En la Ilustración 18 se puede observar el programa de visión artificial. Dentro del bloque Main se encuentran los dos bloques de inicialición y conexión mencionados en el apartado 8.2. Además de estos dos bloques En esta ilustración se muestra;

- Bloque While, que contiene el programa de localización.
- Bloque printf, que permite formar el string para enviar la información
- Bloque de envío y recepción de información
- Bloques script, donde se realizan cálculos y conversiones.

8.3.1. Bloque While

En este bloque se establecen los pasos debidamente ordenados, para obtener los parámetros necesarios para la localización del objeto "pelota". En la Ilustración 19 se muestra el bloque While desplegado para mejorar la comprensión.

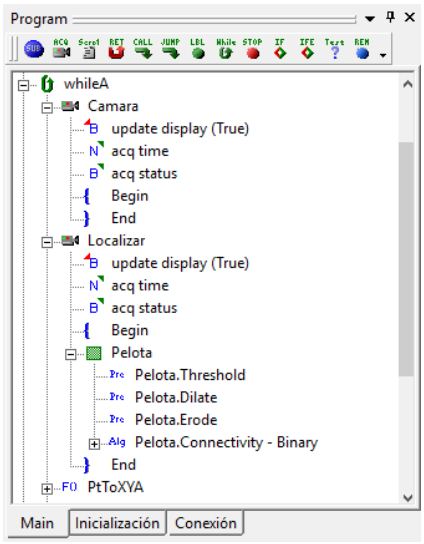


Ilustración 19 - Bloque While

En el bloque While se pretende;

- Obtener imagen.
- Tratar imagen.
- Localizar y posicionar objeto.

8.3.1.1. OBTENER IMAGEN

Para obtener la imagen vasta con seleccionar la cámara dentro de la pestaña en el bus de comunicación COM1.

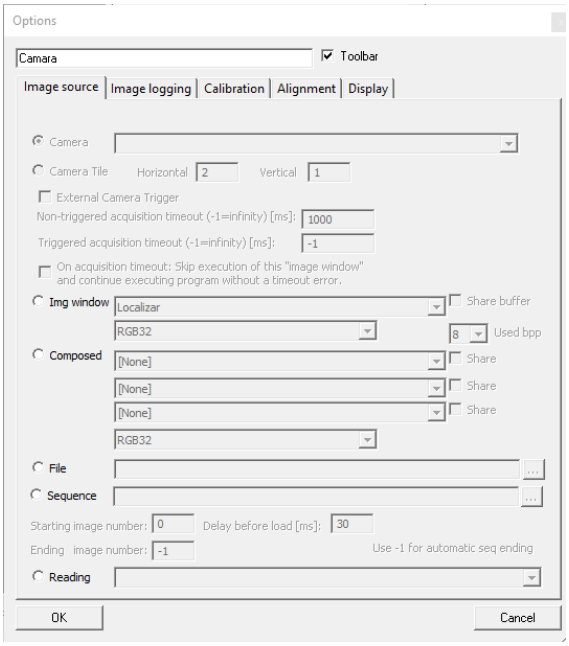


Ilustración 20 - Obtener imagen de la cámara

Una vez seleccionado, se obtuvo la imagen de la cámara.

8.3.1.2. TRATAR LA IMAGEN

Para tratar la imagen se creó una nueva imagen basada en la imagen de la cámara pero en MONO8[3]. Esta imagen es la que se modificó seleccionando el área de trabajo con la opción rectángulo. La opción rectángulo pretende delimitar con unas líneas la zona sobre la que se desea trabajar. En este rectángulo se intentó eliminar la zona que salía fuera de la mesa Ilustración 21.

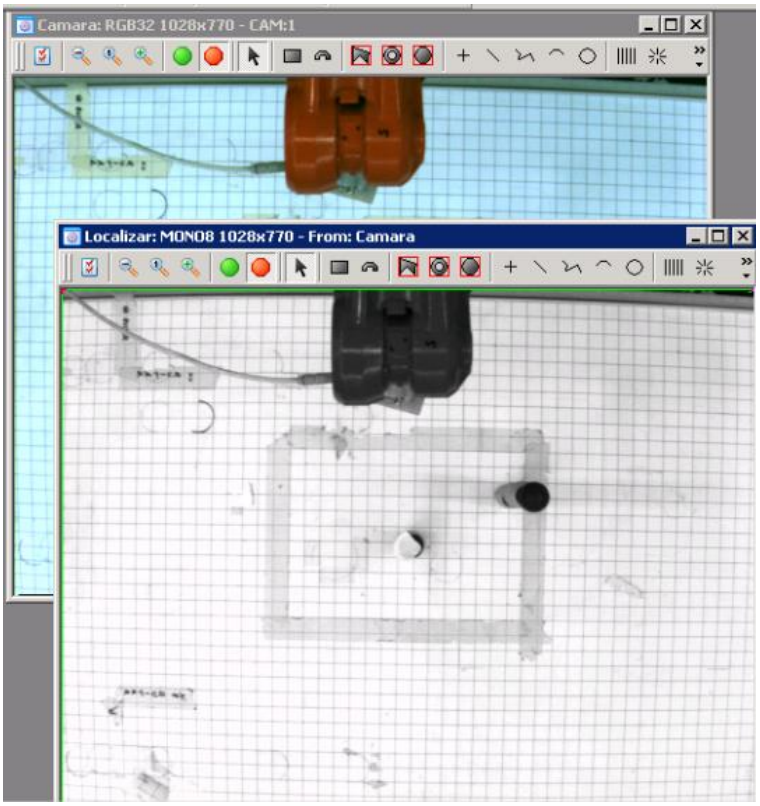


Ilustración 21 - Segunda imagen

A continuación se procedió a utilizar herramientas del propio software de visión artificial.

- Threshold que se ajustó de manera experimental. Filtrando todos los tonos de colores claros de la imagen.
- Erode que erosiona los puntos obtenidos para eliminar los puntos pequeños.
- Dilate los agranda tantas veces como se ha erosionado para obtener unas dimensiones similares a las originales.

El resultado de utilizar esas herramientas es una imagen en la cual solo se muestran los puntos de color oscuro.

[3] MONO8-Monocromo de un solo color

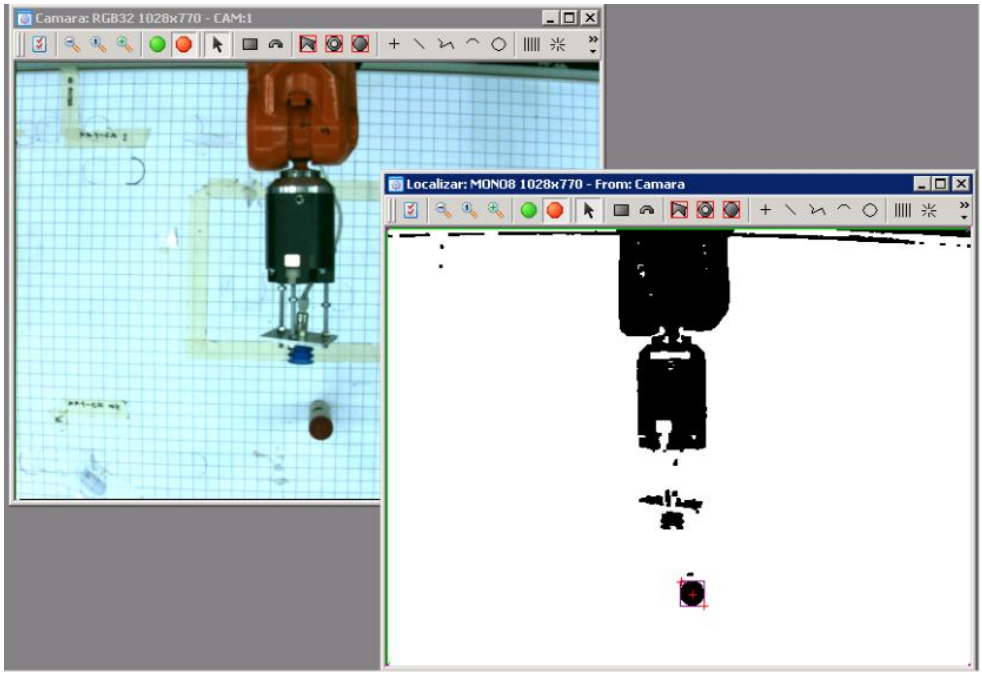


Ilustración 22 - Imagen tratada

Con la imagen de la Ilustración 22 ya se puede empezar utilizar otros comandos para la localización del objeto. Aunque en esta imagen se muestra la pelota localizada, con los pasos seguidos hasta ahora solo se vería la imagen como la ilustración, sin la localización.

8.3.1.3. LOCALIZAR Y POSICIONAR OBJETO

Para localizar el objeto como se observa en la Ilustración 22, es necesario establecer unos parámetros del objeto que se desea encontrar.

En este caso se ha utilizado el comando connectivity – binary. Este comando te permite establecer pequeñas conexiones entre los diferentes pixeles de la imagen. En este caso se buscaba un objeto redondo, por lo que se asignó la cualidad 31. Objetos redondos. Además, se estableció unas dimensiones máximas y mínimas experimentalmente de la pelota.

- Área mínima 700.

- Área máxima 2500.

Esto nos permitía diferenciar la pelota del resto de objetos de la imagen.

Con estos parámetros se acoto la distancia a la cual se podía distinguir la pelota. Siendo la distancia máxima la mesa y la mínima a 1200 mm de la mesa en dirección a la cámara. La zona de color rojo de la Ilustración 23 es la zona en la cual la pelota tiene un área tan grande que no se decide actuar y tenerla en cuenta como objeto de interés. Además el robot sería incapaz físicamente de llegar a este punto debido a su longitud.

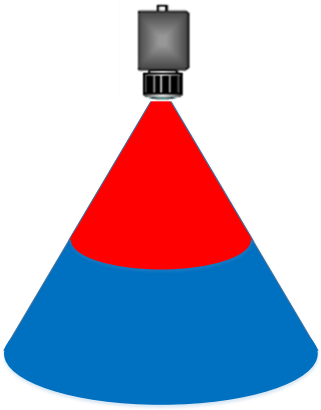


Ilustración 23 - Restricción de are por arriba

Con este algoritmo se localiza la pelota siguiendo los criterios mencionados. En el caso de poner más pelotas, el algoritmo detecta la pelota que mejor cumple las condiciones. Este algoritmo solo está diseñado para localizar un objeto de interés y extraer la información de interés en forma de números enteros.

En la Ilustración 24 vemos las variables obtenidas del algoritmo connectivity – binary Con las cuales se ha realizado la calibración.

Vars.Area_objeto= área en pixeles.

Vars.X=Posición X en la cámara.

Vars.Y=Posición Y en la cámara.

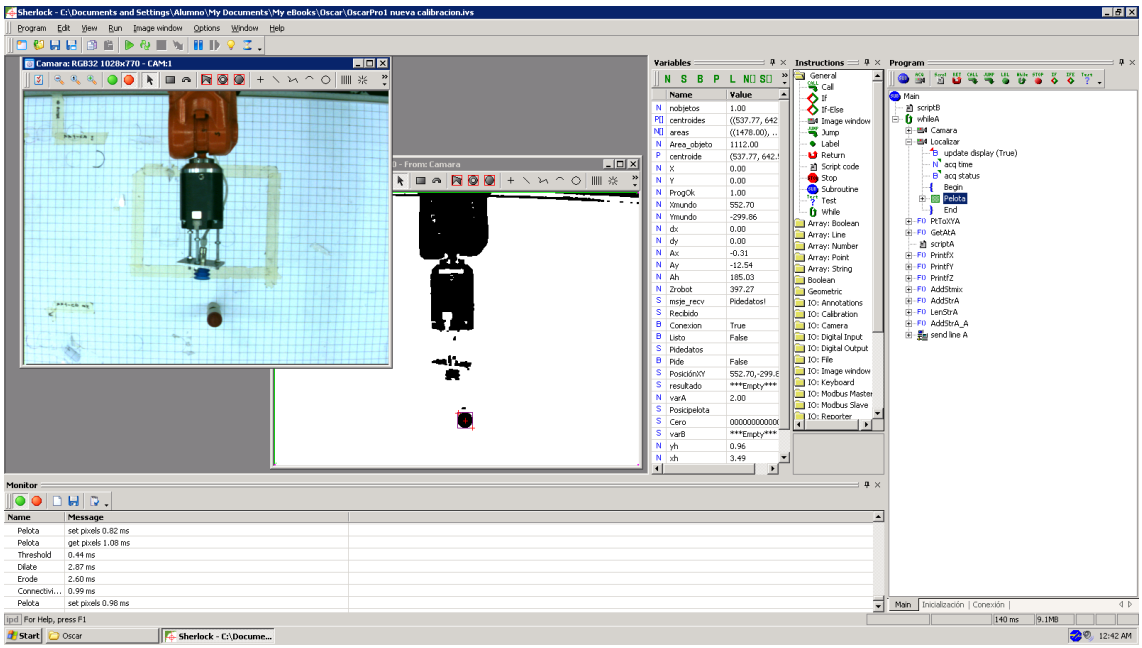


Ilustración 24 - Variables de Sherlock

8.3.2. BLOQUE PRINTF

Con estos bloques se cambia de formato las variables del sistema obtenidas en el punto anterior. Permitiendo así su futura agrupación para la lógica de programación a la hora de utilizar la comunicación por sockets.

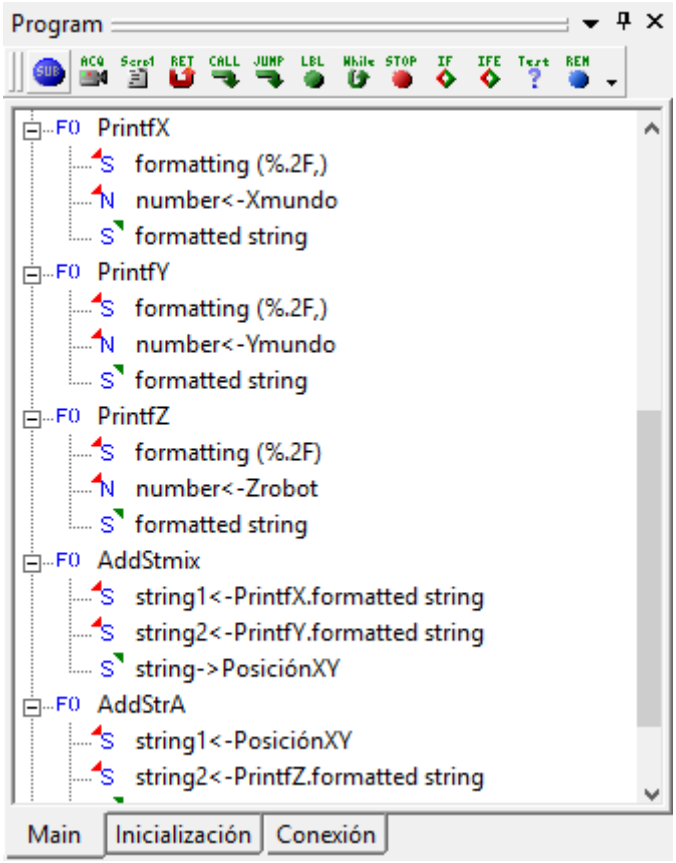


Ilustración 25 - Formato del string

En la Ilustración 25 los comandos printf" " se utilizan para al final de cada variable numero insertar un punto con dos decimales y dar un nuevo nombre a la variable.

Con el comando AddStmix se va agrupando cada una de las variables para formar un único string. Este será el string enviado a RobotStudio.

8.3.3. BLOQUE DE ENVÍO Y RECEPCIÓN DE INFORMACIÓN

Para enviar la información tenemos que haber realizado previamente la conexión. De esta manera se realiza la conexión a través del puerto 1024. Puesto que Sherlock es el Maestro y solo envía información a través del bus, el programa esta enviado constantemente la posición real de la pelota respecto al origen de coordenadas del robot.

8.3.4. BLOQUES SCRIPT

En estos bloques se establecen todos los cálculos necesarios para transformar los parámetros obtenidos con connectivity – binary a posiciones reales de la pelota respecto al origen de coordenadas del robot.

El scriptB, solo contiene una posición segura de la herramienta, por si el robot iniciara un movimiento sin pelota, que esta no intersecte con ningún objeto.

- scriptB;

```
Vars.X=384;                    Vars.Y=512;
```

El scriptA contiene los parámetros de las ecuaciones obtenidas en el apartado 7.2 Todas ellas unidas para obtener la posición de [X, Y, Z] en el mundo.

La variable altura como se menciona en el apartado 7.2.1 se corrige en función de la variable X e Y.

Por último se ha escrito una pequeña línea de código que establece la altura mínima que puede alcanzar el robot en el programa. Para que la herramienta del robot no choque con la mesa. Estableciendo que el área debe de ser superior a 1076.

- scriptA

```
Vars.xh=0. 0073*Vars.Area_objeto+511,52;
Vars.yh=0. 0312*Vars.Area_objeto-363.2;
Vars.x1=Vars.X-Vars.yh;
Vars.y1=Vars.Y-Vars.xh;
Vars.Xmundo=1.0684*Vars.y1+273.06-1;
Vars.Ymundo=1.053*Vars.x1-522.56;
Vars.Ah=134,42*Vars.Area_objeto-68739;
Vars.Zrobot=Vars.Ah+Vars.Ax-Vars.Ay+200;
if(Vars.nobjetos=1&Vars.Area_objeto>1076) {
Vars.ProgOk=1;
}
else {
Vars.ProgOk=0;
}
```

9. PROGRAMACIÓN ROBOTSTUDIO

RobotStudio es una herramienta proporciona por ABB que permite simular y programar el robot fuera de línea. Esta herramienta se ha utilizado para simular el movimiento del robot entre diferentes puntos en la asignatura del grado Sistemas Robotizados.

Durante el inicio en la toma de contacto con el robot se ejecutaron los programas realizados en la asignatura Sistemas robotizados, comprobando así su eficacia. Estas pruebas además ayudaron a comprender el funcionamiento del equipo.

Con los conocimientos adquiridos en la asignatura y la experiencia adquirida al llevar a práctica las prácticas de la asignatura se inicia el trabajo con el robot.

El lenguaje utilizado para programar en RobotStudio es Rapid. Rapid es un lenguaje de programación propio de ABB para programar robots de su marca. En este documento se mencionaran algunos comandos utilizados de Rapid.

EL programa generado con RobotStudio tiene la finalidad de recibir la información de Sherlock y mover la herramienta del robot hasta el punto calculado en Sherlock.

Proceso de programación con RobotStudio es:

- Preparación RobotStudio.
- Declarar variables.
- Declaración de los robtarget.
- Comunicación con Sherlock.
- Programa Rapid.

9.1. PREPARACIÓN ROBOTSTUDIO

Para realizar esta parte del trabajo también se disponía de un apartado en el documento “Anexo”. En esta guía te indica que hagas una copia de seguridad del robot, backup. La copia de seguridad se ha adjuntado en la documentación.

En la guía indica que se ejecute el proyecto a partir de un proyecto existente para obtener la configuración del robot. Te indica que una vez hecha la copia de seguridad utilices la estación de trabajo existente para editarla. En uno de estos pasos donde se supone que tiene que extraer el buckup, mostraba un error de incompatibilidad con la versión de RobotWare. Este apartado fue imposible hacerlo. El programa de RobotWare instalado en los equipos de la universidad, no era compatible con el programa del robot. Al parecer las versiones no coincidían. Después de hacer una tutoría con el profesor y varias reuniones con el encargado del laboratorio se decidió buscar una solución alternativa. Se creó un proyecto base desde la consola virtual y con la configuración predeterminada para el robot. Permitiendo así proseguir con el trabajo.

Con el RobotStudio configurado se procedió a realizar la programación en Rapid.

9.2. DECLARAR VARIABLES

En todo programa la inicialización de las variables es determinante para que el programa se ejecute correctamente. Cualquier variable no inicializada puede detener el programa o provocar una acción no deseada.

En este programa se inicializan todas las variables que se van a utilizar. El programa está en el apartado 2 de los anexos.

9.3. DECLARACIÓN DE LOS ROBTARGET

Los Robtarget son puntos que especifican tanto posición como orientación de una forma unívoca desde las coordenadas del robot. Los Robtarget pueden ser constante (CONST) o variables (VAR). Se han definido 4 Robtarget;

- Origen, posición inicial del robot al iniciar el programa.
- Origen1, inicializado a cero para sumar los parámetros en las posiciones [X, Y, Z] los valores de Sherlock.
- P2, punto donde se muestra la pelota.
- P3, punto donde se deposita.

De los Robtarget definidos, solo el Origen1 es un Robtarget variable el resto son constantes.

9.4. COMUNICACIÓN CON SHERLOCK

Para que el robot pueda establecer la comunicación se necesitan establecer los mismos parámetros que los establecidos en Sherlock. Por ellos se utilizan los mismos parámetros para comunicación por sockets (IP, puerto, variables.). La comunicación en con RobotStudio se realizó mediante las instrucciones proporcionados en la guía.

Instrucción	Descripción
SocketCreate	Crea un *socket* nuevo y lo asigna a la variable socketdev
SocketConnect	Realiza la conexión con un ordenador remoto
SocketSend	Envía datos vía *socket* al ordenador remoto.
SocketReceive	Recibe datos y los almacena en una cadena de datos
SocketClose	Cierra el *socket*

Tabla 3 – Instrucciones de comunicación

En la Tabla 3 se muestra la descripción de cada una de estas instrucciones.

En primer lugar se crea un canal, para posterior mente enviar y recibir las mismas cadenas de datos que las emitidas en Sherlock. (Viene explicado en la guía)

La complejidad de esta comunicación se basa en desfragmentar el string recibido y almacenar las variables enteras para su utilización.

9.4.1. RECONOCIMIENTO DEL STRING

El string es la cadena de información que envía Sherlock. Esta cadena de información una vez recibida, se trocea en tres trozos, PosX, PosY y PosZ. Esta troceado venia explicado junto con la comunicación por sockets. Solo se tenía que interpretar.

9.5. PROGRAMA RAPID

Con el programa de Rapid se pretende que el robot, a partir de un punto pre-determinado, establezca una trayectoria a un punto determinado por la cámara. Posteriormente muestre la pelota, y por último la deje en otro punto conocido.

La ejecución de este programa era sencilla, pues no era objeto de proyecto complicar el proceso. Los pasos del programa son;

- Ir al punto reposo del robot.
- Ir al punto establecido por Sherlock.
- Mostar pelota.
- Dejar pelota en otro punto determinado.

9.5.1. IR AL PUNTO REPOSO DEL ROBOT

Para ir al punto reposo del robot se le ordena al robot que valla al punto “origen”.

Para ello se pueden elegir dos comandos diferentes, MoveJ o MoveL estos comandos determinan como se va a generar la trayectoria en el algoritmo para ir de un punto a otro. Por lo general, en este proyecto se ha utilizado el comando MoveJ en todos los movimientos. Esto permite que la trayectoria se genere de forma natural. MoveL genera una trayectoria lineal de un punto a otro, que en este caso no era necesaria.

Mientras que el robot alcanza la posición se ordena al Flexpendant[4] que borre el contenido de la pantalla. Permitiendo así mostrar futuros mensajes al operador.

La orden de Rapid contiene cinco partes, el tipo de movimiento ”MoveJ”, el punto “Origen”, la velocidad “V100” (Velocidad moderada-lenta), precisión de seguimiento de la trayectoria”z5”, Herramienta”tool0”. A continuación se muestra un ejemplo de orden.

```
MoveJ origen,v100,z5,tool0;
```

En esta fase de programación no se creyó necesario aumentar la velocidad ni la precisión de los movimientos, pues es meramente ilustrativa.

9.5.2. IR AL PUNTO ESTABLECIDO POR SHERLOCK

En esta fase de la programación se prende ir a un punto cual sea de la mesa. Esta zona a la que pretendemos acceder está limitada por diferentes características;

- Características del programa de Sherlock.
- Características de la calibración.
- Características del robot.

Además de otras características externas (iluminación, pelota.). Por lo tanto diremos que iremos a un punto dentro de la zona de control. La zona de control se encuentra dentro de los puntos azules en la Ilustración 26.

[4] Flexpendant – Consola o manipulador del robot.

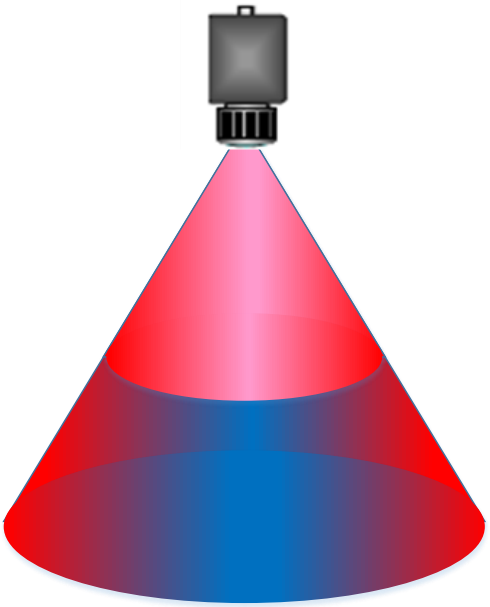


Ilustración 26 - Zona de control

Los puntos dentro de la zona de control se encuentran cerca de la línea que proyecta la cámara tangente a la mesa. A una distancia de la mesa de 1m como máximo. Cuando más nos alejamos del centro de la imagen de la cámara menos preciso es el posicionamiento de la pelota.

Para ir al punto donde se encuentra la pelota se ha establecido un punto "origen1" cuyas variables [X Y Z] están inicializadas a 0. De esta manera al escribir en estas variables la posición obtenida por Sherlock basta con hacer un MoveJ offs para ir a este punto. El comando offs, hace que el algoritmo añada los valores de Sherlock a los existentes.

MoveJ offs(origen1,posx,posy,posz),v50,z5,tool0;

Además de esto se muestra en el Flexpendant los valores obtenidos de Sherlock para que le operario pueda ver si son reales o no, y parar la ejecución del programa.

Junto a este movimiento se acciona la ventosa, que al final del movimiento, debe tener la pelota succionada.

9.5.3. MOSTAR PELOTA

El objetivo del trabajo era coger el objeto, pelota. Por lo tanto se estableció un parámetro que únicamente pretende mostrar la pelota.

Para ello, se estableció un punto dentro del rango del robot. El punto se estableció mediante el método experimental aprendido durante la elaboración del TFG.

El método experimental, consiste en posicionar el robot en un punto, obtener los parámetros de este y añadirlos al programa.

10. PRUEBAS

Para llegar a determinadas conclusiones se invirtieron muchas tardes en trabajo de análisis y estudio. Estas pruebas han formado parte de la experiencia de realizar el TFG con los equipos dispuestos.

A continuación se expondrán algunas de las pruebas realizadas a los diferentes equipos y las conclusiones de las mismas.

10.1. MÉTODO DE OBTENCIÓN DE PARÁMETROS

Para obtener los parámetros mencionados durante el trabajo se han seguid los siguientes métodos.

10.1.1. PARÁMETROS PARA LA CALIBRACIÓN EN 2D

Se ha utilizado un programa de localización para obtener los 5 puntos del patrón. Con estos cinco puntos se obtienen los cinco centros de las cinco esferas del patrón. Se ha anotado esta información y a continuación sin mover el patrón se ha llevado la herramienta del robot al centro de cada una de las esferas.

El robot se ha manejado en modo lineal. Lo cual facilitaba el posicionamiento del mismo sobre el patrón Ilustración 27.

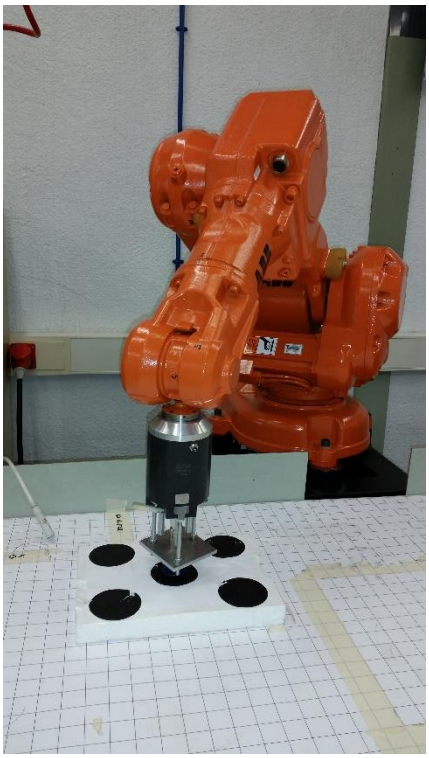


Ilustración 27 - Calibración 2D

Con la información obtenida se ha introducido en forma de matriz en el script de Matlab “homografía” y se han obtenido como resultado la matriz de transformación.

A continuación se han probado diferentes puntos al azahar para verificar la calibración.

10.2. PRUEBAS ROBOTSTUDIO

Durante la programación en Rapid se requería establecer determinadas posiciones del Robot para simular como se movería de un punto a otro. Estas simulaciones se hacían asignando dos puntos en el espacio y dejando que el robot se moviera en automático entre los dos puntos. Con estas simulaciones se consiguió comprender los errores de movimiento que daba el Robot al ejecutar el programa. Pues las articulaciones se retorcían y no era posible seguir ejecutando el programa.

A continuación se muestra una Ilustración 28, del robot en el punto de soltar la pelota.

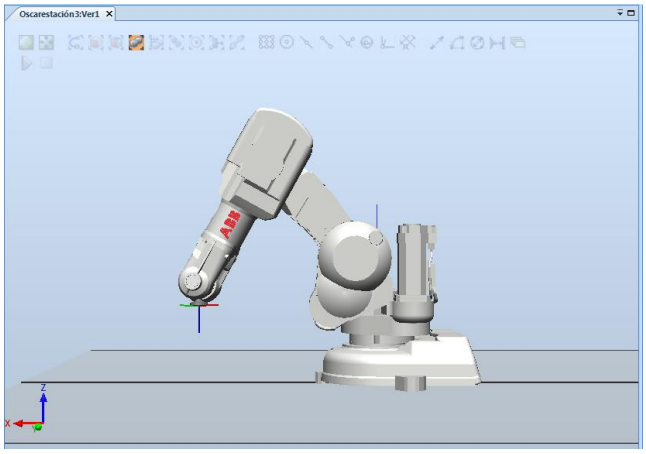


Ilustración 28 – RobotStudio

El problema se corrigió cambiando la configuración de los ejes del robot. De [0,0,-1,0] a [0,-1,0,1] Que pese a pesar de estar en el mismo punto, la segunda permite llegar a dejar la pelota.

11. CONCLUSIONES

Los conocimientos adquiridos y la información proporcionada al inicio del TFG y durante el Grado han permitido que este proyecto se pudiera realizar.

El objetivo propuesto inicialmente se cumplió. Localizar un objeto dentro de la zona, cogerlo, mostrarlo y depositarlo en otro punto. Cuando se propuso el trabajo durante una reunión, no sospechaba todo el trabajo que había detrás de esta tarea tan simple fuera a ser tan laboriosa.

Los equipos del laboratorio han sido suficientes para elaborar el TFG. Aunque necesitan una revisión.

La trayectoria generada entre los diferentes puntos era determinante para que el programa funcionara sin errores. Puesto que si el robot se retorcía, mostraba un error de movimiento. Este error se eliminado cuando al simular diferentes puntos de recepción de la pelota se vio que la posición del robot no era la idónea. Esto se modificó en el punto 10.2.

El método experimental de posicionamiento del robot resulto muy interesante pues parece ser un estándar para programar robots en la industria.

La herramienta utilizada para coger el objeto no era la más adecuada. Una herramienta para esta pelota mejoraría el éxito de las pruebas.

La visión artificial es una herramienta indispensable para conseguir que los procesos industriales sean dinámicos. Sin duda el trabajo invertido en estos equipos ha sido muy útil.

La mayor dificultad durante este trabajo fue conseguir una posición de la pelota en el espacio en 3D. La calibración en 3D resolvió el problema. Para conseguir más precisión se podría repetir el estudio e incorporar más información del entorno en los cálculos. Hay que tener en cuenta que este equipo requiere calibración precisa. Compartir el setup implica múltiples re-calibraciones.

Durante el TFG se ha aprendido muchísimo acerca de estos equipos, sin duda lo aprendido será de gran utilidad para un trabajo relacionado.

12. REFERENCIAS

Todas las licencias de los programas utilizadas durante el TFG han sido suministradas por la Universidad Politécnica de Valencia.

Los conocimientos adquiridos durante el grado han sido de gran utilidad. Han permitido identificar las diferentes partes del proceso a controlar y estudiarlas por separado.

Documentos utilizados para elaborar el TFG;

- Apuntes – Apuntes asignaturas – Sistemas Robotizados, Visión artificial etc.
- Documento – Anexo – Programación de un robot antropomórfico para la manipulación de elementos en una cinta transportadora.
- Manual - Datasheet_CV-M77 – Progressive Scan RGB Color Camera.
- Manual – Manual RobotStudio – Manual de operador.

Mis agradecimientos a la escuela y los profesores, en concreto Carlos Ricolfe que me han guiado durante este periodo. También agradecer a las personas que hacen posible que los alumnos dispongamos de estas herramientas que nos permiten poner en práctica nuestros conocimientos y desarrollarnos como profesionales.

1. ANEXOS

1. ANEXOS 42

2. PROGRAMA RAPID 43

2.1. TIEMPO UTILIZADO PARA ELABORAR EL TFG 47

3. ESTUDIOS SEGURIDAD 48

4. PARÁMETROS INTERNOS DE LA CÁMARA 49

4.1. RESULTADO CALIBRACIÓN 49

5. MATRIZ DENAVIT HARTENBERG IRB 140 50

6. RESUMEN ERRORES 51

6.1. CPU VISIÓN ARTIFICIAL 51

6.2. PROBLEMA CALIBRACIÓN ANEXOS 2D 51

6.2.1. Modificacion parametros de la camara 52

2. PROGRAMA RAPID

A continuación se incluye el programa copiado y pegado de Rapid para su reproducción en el futuro.

```
MODULE Oscar

 !Declaración de variables

VAR socketdev canal1;

VAR string cadena_de_datos_recibida:="";

VAR num longitud_trama:=0;

VAR num posicion_string_inicial:=0;

VAR num posicion_string_final:=0;

VAR num vuelta;

VAR string v:="";

VAR num posx:=0;

VAR num posy:=0;

VAR num posz:=0;

VAR string trozo:="";

VAR num Len2:=0;

VAR bool ok;

VAR num ChPos1:=0;

VAR num ChPos2:=0;

VAR num q1:=0;

VAR num q2:=0;

VAR num q3:=0;

VAR num q4:=0;

!Decalaracines Rob taget

!CONST robtarget origen:=[[515,0,612],[0,0,1,0],[0,2,2,1],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];

CONST robtarget origen:=[[454.43,4.76,644.14],[0,0,-1,0],[0,0,-1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];

VAR robtarget p1:=[[550,0,232],[0,0,-1,0],[0,0,-1,0],[9E9,9E9,9E9,9E9,9E9,9E9]];

CONST robtarget origen1:=[[0,0,0],[0,0,-1,0],[0,0,-1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];

CONST robtarget P2:=[[527.39,5.44,638.91],[0.712429,1.24725E-05,0.701705,0.00744453],[0,-1,0,1],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
```

```
    CONST                                                                robtarget
P3:=[[640.75,26.82,339.68],[0.0875311,0.429384,0.898182,0.0351644],[0,-
1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];


    !Programa principal
    PROC main()
      MoveJ origen,v100,fine,tool0;
       !Borrar Flexpendant
      TPErase;

      !Comunicación con Sherlock 7
      SocketCreate canal1;
      SocketConnect canal1, "158.42.16.207", 1027;
      SocketSend canal1\Str:="Hola!";
      SocketReceive canal1\Str:=cadena_de_datos_recibida;
      SocketSend canal1\Str:="Recibido";
      WaitTime 0.25;
      TPWrite "----------------------------";
      TPWrite "Conexión correcta";
      TPWrite "----------------------------";

      !Programa principal
      IF vuelta=0 then
        !set GRIPPER_CLOSE;
        posx:=454.43;
        posy:=4.76;
        posz:=644.14;
        SocketSend canal1\Str:="Pidedatos!";
        SocketReceive canal1\Str:=cadena_de_datos_recibida;
        TPWrite cadena_de_datos_recibida;
     longitud_trama:=StrLen(cadena_de_datos_recibida);

        !Lectura de la string
```

```
    IF longitud_trama>0 THEN
      posicion_string_inicial:=0;
      posicion_string_final:=1;
      vuelta:=0;
      WHILE vuelta<3 DO
        ChPos1:=posicion_string_inicial+posicion_string_final;
        v:=StrPart(cadena_de_datos_recibida,ChPos1,1);
        WHILE v<>"," DO
          posicion_string_final:=posicion_string_final+1;
          ChPos1:=ChPos1+1;
          v:=StrPart(cadena_de_datos_recibida,ChPos1,1);
        ENDWHILE
      Len2:=posicion_string_final-1;
      ChPos2:=posicion_string_inicial+1;
      trozo:=StrPart(cadena_de_datos_recibida,ChPos2,Len2);

      IF vuelta=0 THEN
        ok := StrToVal(trozo,posx);

      ELSEIF vuelta=1 THEN
        ok := StrToVal(trozo,posy);

      ELSEIF vuelta=2 THEN
        ok := StrToVal(trozo,posz);
ENDIF
posicion_string_inicial:=posicion_string_inicial+posicion_string_final;
posicion_string_final:=1;
vuelta:=vuelta+1;
WaitTime 1;
    ENDWHILE
    TPWrite "Coordenada X="\Num:=posx;
    TPWrite "Coordenada Y="\Num:=posy;
```

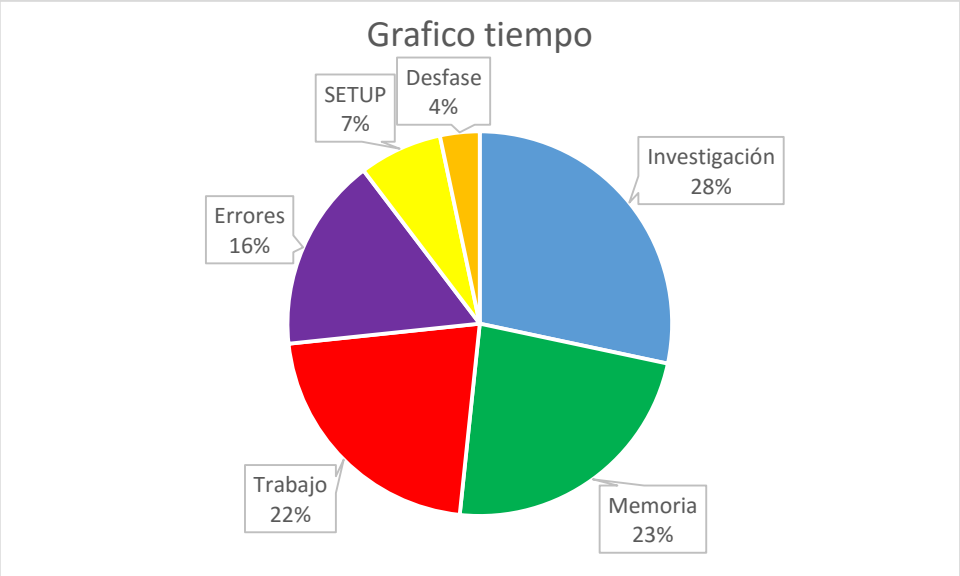
```
        TPWrite "Coordenada Z="\Num:=posz;
        TPWrite "----------------------------";
        TPWrite "------Final------";
        TPWrite "----------------------------";
        Reset GRIPPER_CLOSE;
        MoveJ offs(origen1,posx,posy,posz),v50,z5,tool0;
        WaitTime 3.25;
        MoveJ origen,v100,fine,tool0;
        !Borrar Flexpendant
        TPErase;
        longitud_trama:=1;
        MoveJ P2,v100,fine,tool0;
        WaitTime 3.25;
        MoveJ P3,v50,fine,tool0;
        WaitTime 3.25;
        vuelta:=0;
        set GRIPPER_CLOSE;
        ENDIF

    ENDIF

    SocketClose canal1;
ENDPROC
ENDMODULE
```

2.1. TIEMPO UTILIZADO PARA ELABORAR EL TFG

Durante el TFG se han encontrado muchas trabas que impedían que el TFG se realizara con normalidad. Debido que la cantidad de tiempo empleado para resolver estos inconvenientes se ha decidido incorporar un apartado para mencionar los inconvenientes encontrados e intentar justificar el tiempo empleado para elaborar el TFG.



Investigación – Tiempo dedicado a la formación e investigación de los asuntos que han surgido durante el TFG.

Memoria – Tiempo dedicado a la elaboración del escrito, memoria.

Trabajo – Tiempo dedicado a la adquisición de datos y pruebas.

Errores – Solución de errores de los equipos. En algunos casos los equipos no funcionaban correctamente o como era esperado. Estos errores se mencionaran en un apartado anexos 6.

Acceso - El acceso al laboratorio está condicionado pues no es de libre acceso. Además de esto modificar el software y hardware de la computadora o el setup en ocasiones no era viable. El tiempo dedicado a montar el setup está incluido en este punto.

Setup - Al inicio del TFG se indicó que los equipos disponibles eran un poco antiguos. Esto implica invertir tiempo en comprender el funcionamiento de los mismos, establecer comunicación con fabricante, resolver averías, buscar la manera de hacer que funcione, invertir tiempo en resetear los equipos, puesta en marcha lenta, etc.

3. ESTUDIOS SEGURIDAD

Las normas se seguridad, se basan principalmente en evitar el riesgo de contacto entre robot y operario. Por ello los robots trabajan en células robotizadas. El interior de la célula robotizada se descompone en diferentes áreas concéntricas según el riesgo

- Espacio controlado (zona de aislamiento): Es el espacio delimitado por protecciones. Se define en función de la evaluación de los riesgos.
- Espacio máximo: Espacio que puede ser barrido por las partes móviles del robot (definido por el fabricante), más el espacio que puede ser barrido por el terminal y la pieza.
- Espacio restringido: Porción del espacio máximo reducido por los dispositivos limitadores de recorrido, que establecen los límites que no pueden ser sobrepasados en caso de cualquier fallo previsible de la célula robotizada
- Espacio de operación: Porción del espacio restringido que es barrido por el robot cuando éste ejecuta un programa.

En este proyecto el robot se encontraba dentro de un espacio de operación con unas cristaleras que impedían el acceso al setup. Esto proporcionaba la máxima seguridad a la hora de ejecutar el programa en modo automático.

Para variar la posición de la pelota se utilizaron unos tubos de cartón que la sostenían.

Con todas estas medidas de seguridad cabe destacar que no hubo ningún incidente con ningún equipo.

4. PARÁMETROS INTERNOS DE LA CÁMARA

Calibración Matlab, cálculo de parámetros internos de la cámara. Durante el inicio que creyó conveniente obtener los parámetros internos de la cámara. El resultado obtenido de este estudio no fue concluyente, los parámetros no parecían realistas. El proceso se repitió unas veces y finalmente se desistió. Adjuntamos resultado.

Para realizar este estudio se tomó un patrón de cuadricula. Estos patrones son habituales en estas calibraciones. Se conocen sus características geométricas y se introducen en el algoritmo.

A continuación se muestran las imágenes obtenidas con Matlab. En estas imágenes se observa el patrón en diferentes ángulos.

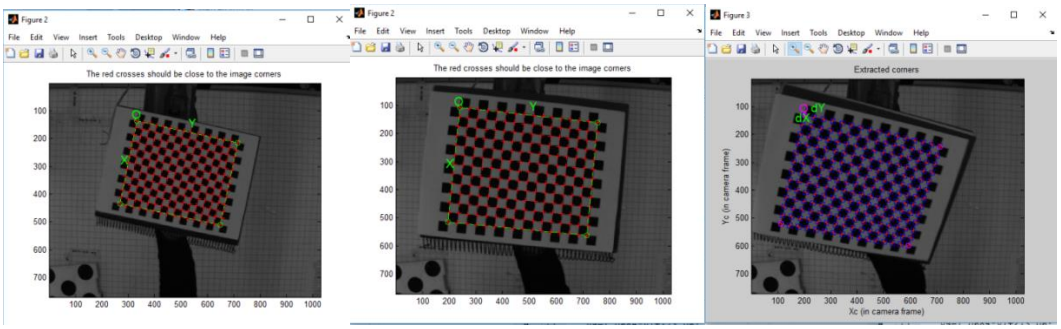


Ilustración 29 - Patrón calibración

4.1. RESULTADO CALIBRACIÓN

Se introdujeron seis imágenes del patrón.

- Initialization of the intrinsic parameters - Number of images: 6

A continuación se muestran los resultados obtenidos de la calibración.

- Calibration parameters after initialization:
  - Focal Length: fc = [1922.69685 1922.69685]
  - Principal point: cc = [513.500 384.500]
  - Skew: alpha_c = [0.000] => angle of pixel = 90degrees
  - Distortion: kc = [0.000 0.000 0.000 0.000 0.000]
- Calibration results after optimization (with uncertainties):
  - Focal Length: fc = [1737.54513 1698.40125]+/-[21.3567 20.4852]
  - Principal point: cc = [490.3185 394.5275]+/-[12.45210 11.02229]
  - Skew: alpha_c = [0.000] +/- [0.000] => angle of pixel axes = 90+/- 0.0 degrees
  - Distortion: kc = [-0.38640 -0.06190 -0.00179 -0.00069 0.00] +/- [0.03297 0.50356 0.00153 0.00103 0.00]
  - Pixel error: err = [0.32122 0.32507]

Puesto que estos parámetros parecían demasiado elevados a la realidad, 2cm frente a lo que parece ser medio en la realidad. Se desestimaron y se tomó otro rumbo en el proyecto.

A continuación se muestran las imágenes obtenidas con referencia la cámara. Esto muestra cómo se realizó la calibración. Ilustración 30 - Imágenes patrón calibración

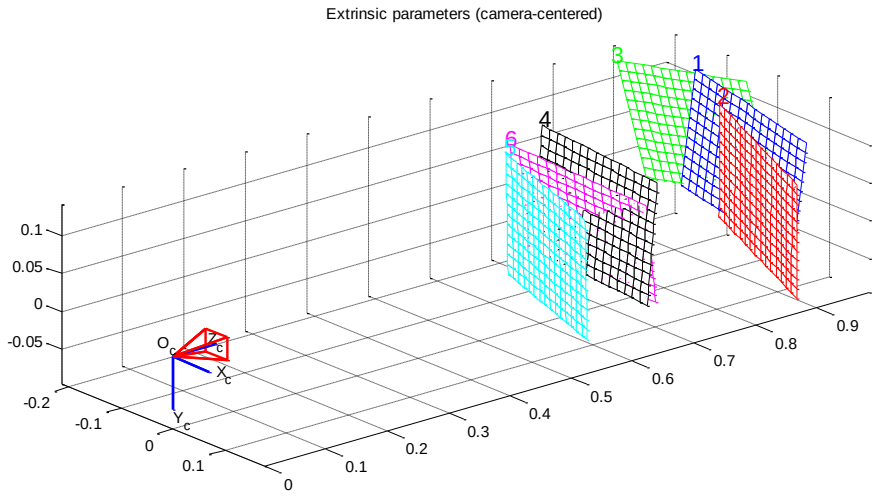


Ilustración 30 - Imágenes patrón calibración

5. Matriz Denavit Hartenberg IRB 140

Este robot tiene limitaciones de movimiento debido a su construcción y geometría. A continuación se muestra la ilustración proporcionada por el fabricante del robot a partir de la cual se pueden determinar los parámetros de q_x.

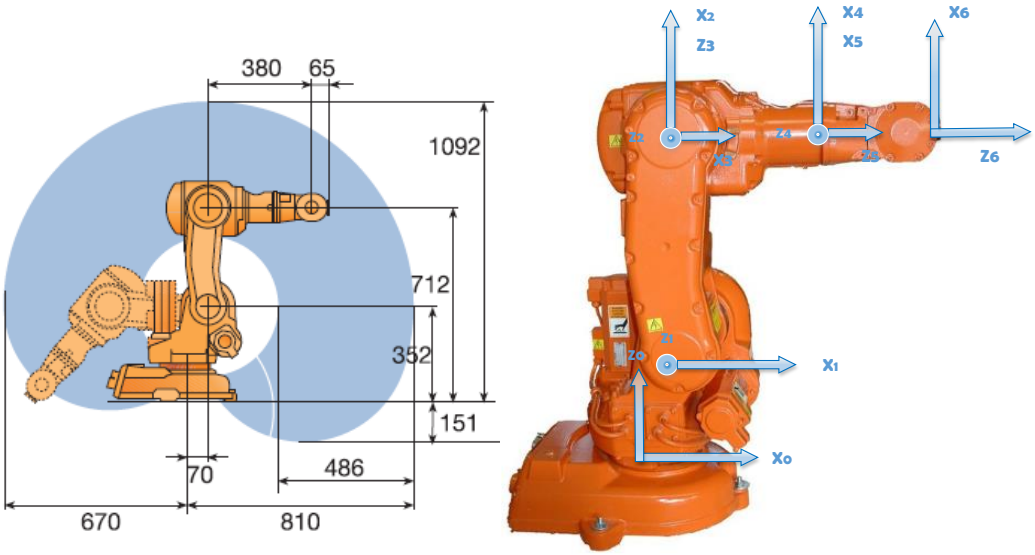


Ilustración 31 - IRB 140 Geometría

Durante inicio se realizó un breve estudio de los posibles puntos alcanzados con el robot, para ello se obtuvieron los parámetros de la matriz de matriz Denavit Hartenberg.

A posteriori se determinan los parámetros a partir de los cuales el robot permanecería en la posición determinada.

Articulación	β	d	a	α
1	q1	352	70	90º
2	q2+90º	0	360	0
3	q3	0	0	90º
4	q4	380	0	-90º
5	q5	0	0	90º
6	q6	65	0	0

Tabla 4 - Parámetros Denavit Hartenberg

Con estos parámetros se introdujeron en una práctica de la asignatura Sistemas Robotizados para simular el movimiento del robot fuera del laboratorio. Al final las simulaciones no eran reales, pues el robot tenía restricciones físicas que no estaban implementadas. Además en RobotStudio se podía simular mejor y era más realista.

6. RESUMEN ERRORES

A continuación se detallan algunos errores encontrados durante el TFG. Algunos errores se han ido solucionando y otros se han asumido.

Los que se han solucionado y se han dejado implantados.

6.1. CPU VISIÓN ARTIFICIAL

Durante la utilización de este equipo se han detectado varios inconvenientes;

- La necesidad de incorporar de un teclado externo. Pues debido a un error interno de la batería se necesita pulsar el botón F1 al cabo de un rato durante el inicio y posteriormente la tecla Enter. De esta manera se acepta el error y se inicia el sistema.
- Modificación de la posición de la CPU, pues el cable del teclado no debía estar colgando y de esta manera quedo encima de un armario. Este teclado era necesario porque al parecer el equipo da un error al arranque.
- Fallo en la comunicación, el equipo no podía conectarse a internet. Puesto que esta conexión solo era necesaria para extraer la información, se hizo un en lace web a la W y se de dejo puesto en el escritorio del equipo para futuros alumnos con un ejemplo.

6.2. PROBLEMA CALIBRACIÓN ANEXOS 2D

Una vez realizada la calibración en 2D y las pruebas para ver si podía ser utilizada. Se planteó si el algoritmo o sistema utilizado era el más apropiado para la localización de un objeto en el espacio.

Se estudió la geometría de la cámara mediante un programa de calibración de Matlab. Con este programa se calcularon los parámetros internos de la cámara. Los resultados de esta calibración se encuentran en El apartado anexos 4.

Con los parámetros internos de la cámara se conoció entonces la problemática. En Sherlock se observa una imagen en 2D. Cuando en realidad esta imagen, esta deformada por la geometría de la lente.

La distancia entre los pixeles del centro de la imagen y entre los de un una esquina de la imagen es diferente. Por ello se concluyó que el programa de calibración del fichero anexo podía funcionar en un punto y sin modificar el eje Z. Pero no se podía utilizar en toda la zona de visión. La lente de la cámara hace que la luz se plasme en la CCD con lo que se correspondería una esfera. Pro ello se realizo la calibracion en 3D 7.2.

6.2.1. MODIFICACION PARAMETROS DE LA CAMARA

Esta programación ha sido modificada tantas veces como se modificaron los parámetros del setup, un 25% de las veces que iba al laboratorio. Esto limito el tiempo de prueba del algoritmo pues cada vez que modificaba el brillo de la cámara me tocaba aumentar o reducir el Threshold para intentar que no afectara la modificación. Esto hacia muy tediosa la tarea de calibrar correctamente el equipo.