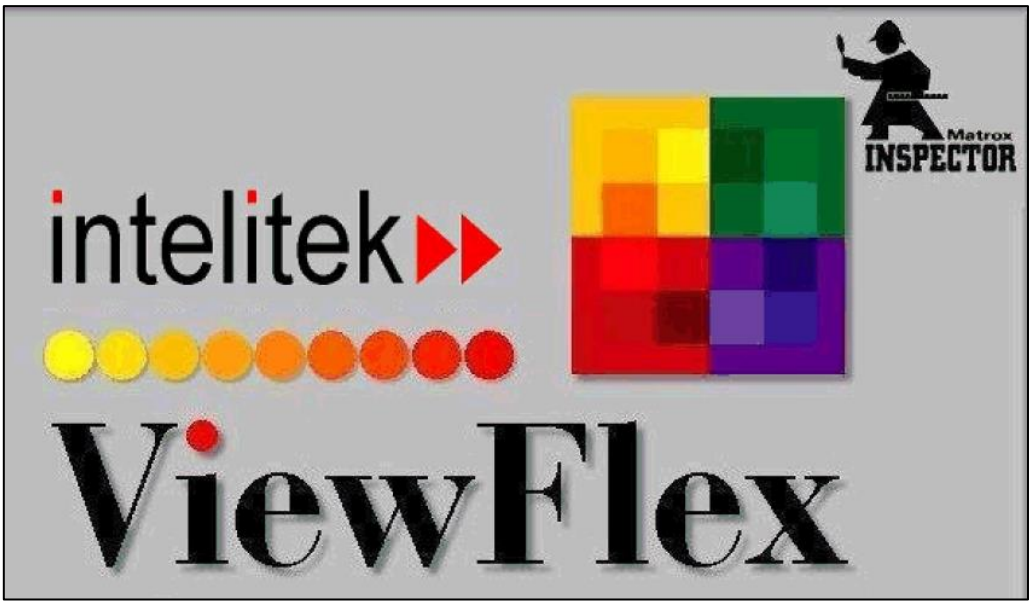
Universidad de Sevilla

Escuela Politécnica Superior

Trabajo de Fin de Grado en Ingeniería Electrónica Industrial



ANÁLISIS Y PRUEBAS DE VISIÓN ARTIFICIAL EN UN ENTORNO INDUSTRIAL USANDO CÁMARA ACOPLADA AL BRAZO DE UN ROBOT.

Autor: Lydia Gómez Franco

Tutor: Fernando Díaz del Río

Fecha: Noviembre de 2021

ANÁLISIS Y PRUEBAS DE VISIÓN ARTIFICIAL EN UN ENTORNO INDUSTRIAL USANDO CÁMARA ACOPLADA AL BRAZO DE UN ROBOT.

Autor: Lydia Gómez Franco

Tutor: Fernando Díaz del Río

Índice

1. RESUMEN

El presente proyecto se basa en el estudio de la identificación de piezas con diferentes formas mediante el uso de tecnología de visión artificial y un brazo robótico, para su posterior clasificación.

Este se realiza a través de la tecnología ofrecida por la empresa Intelitek, tanto para el software de visión artificial, ViewFlex; el robot, Rocobell-4U, y el software de control del robot, SCORBASE.

Por lo que, este estudio se basa en la captura de imágenes mediante una cámara de visión, con el que se genera un patrón de tal imagen, que es guardado, y que a través de él es posible encontrar las mismas figuras en diferentes posiciones. Estos patrones pueden ser obtenidos antes de ejecutar el programa o mientras este está ejecutándose.

2. TABLA DE FIGURAS

3. INTRODUCCIÓN

3.1. Robótica industrial

Desde el principio de la historia se observa una inquietud por crea máquinas y dispositivos capaces de imitar los movimientos y funciones de los seres vivos, para así realizar las tareas que resultan más complicadas.

Ya en la antigua Grecia se empezaron a desarrollar los primeros automatismos para satisfacer las diferentes tareas de la época, mediante tecnología hidráulica y mecánica. Debido a ello, el término “automatos” deriva actualmente en la palabra conocida en el sector como “autómata”.

Los árabes a través de los conocimientos griegos crearon mecanismos para aplicaciones prácticas para la vida cotidiana. Un ejemplo de ello son los dispensadores automáticos para beber agua o lavarse.

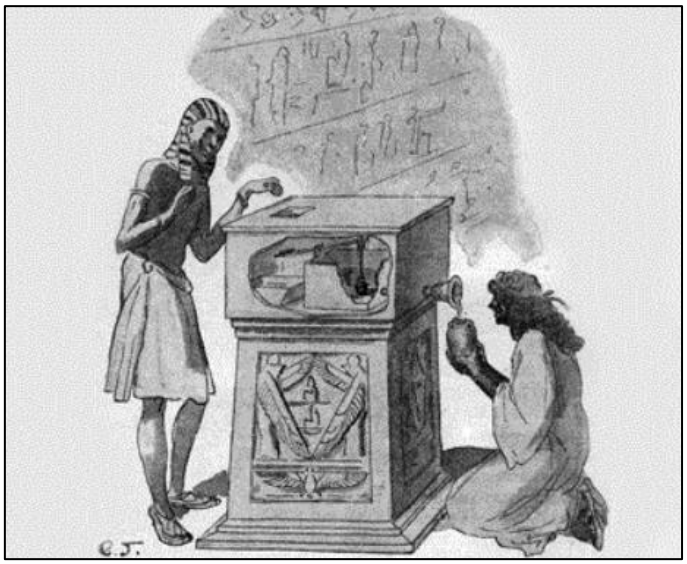


Figura 1. Dispensador de agua egipcio.

Durante los siglos XV y XVI Leonardo Da Vinci (1452-1519) construyó el conocido *León mecánico* para el rey LUIS XII. Mientras que, en el siglo XVII, Juanelo Turriano construyó el *Hombre de palo,* este autómata con forma de monje movía la cabeza, ojos, boca, brazos e incluso andaba.

Ya en los siglos XVII y XVIII los nuevos mecanismos iban adquiriendo más parecido a lo que actualmente conocemos como robots. Un claro ejemplo de los mecanismos de la época es el famoso Pato de Jacques Vaucanson (1709-1782) que hizo ha este pato capaz de graznar, beber, comer, digerir y evacuar comida.

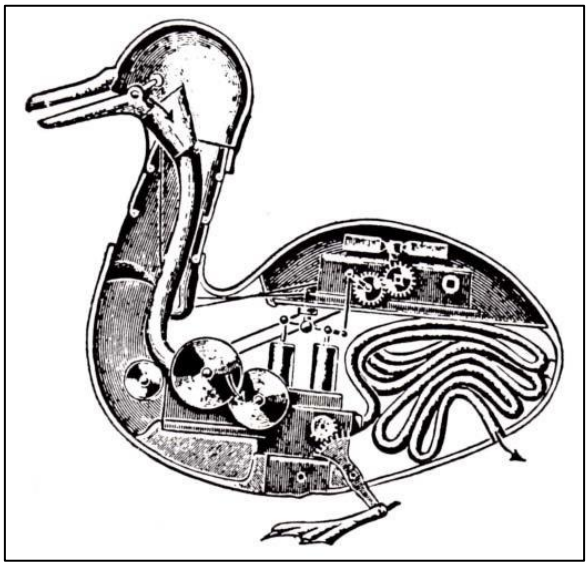


Figura 2. Pato de Vaucanson.

La palabra robot proviene de “robota”, de origen eslava, que significa el trabajo realizado de manera forzado. Se implanta dicha palabra en el año 1921 gracias a un escritor checo llamado Karel Capek (1890-1938) con su obra de teatro *Rossum’s Universal Robot (R.U.R)*, donde aparecen unas máquinas humanoides que servían a sus jefes humanos desarrollando trabajos físicos.

El término podría haberse olvidado si no fuera gracias a los escritores del género de ciencia ficción. Su máximo propulsor fue Isaac Asimov (1920-1992), ya que en una historia aparecen reflejada por primera vez las tres leyes de la robótica:

1. Un robot no puede perjudicar a un ser humano, ni con su inacción permitir que un ser humano sufra daño.
2. Un robot ha de obedecer las órdenes recibidas de un ser humano, excepto si tales órdenes entran en conflicto con la primera ley.
3. Un robot debe proteger su propia existencia mientras tal protección no entre en conflicto con la primera o segunda ley.

Aunque en la actualidad el robot como máquina dista mucho del término robot, ya que los progenitores de los robots fueron los telemanipuladores para la manipulación de elementos radiactivos. Luego en 1954 se cambia la transmisión mecánica del telemanipulador por transmisión eléctrica y desarrollando el primeo con servocontrol bilateral.

El paso de transformación del manipulador al robot se dio gracias a la sustitución del operador que vigilaba el proceso por un programa de ordenador que controlase los movimientos del manipulador.

La primera patente robótica se solicitó en 1954 por el inventor C.W. Kenward, fue emitida en 1957, aunque el inventor que estableció las bases del robot industrial moderno fue George C. Devol. Este junto con otro científico fundaron la *Unimation* e instalaron su primera máquina Unimate 1960, similar a la de la siguiente figura:

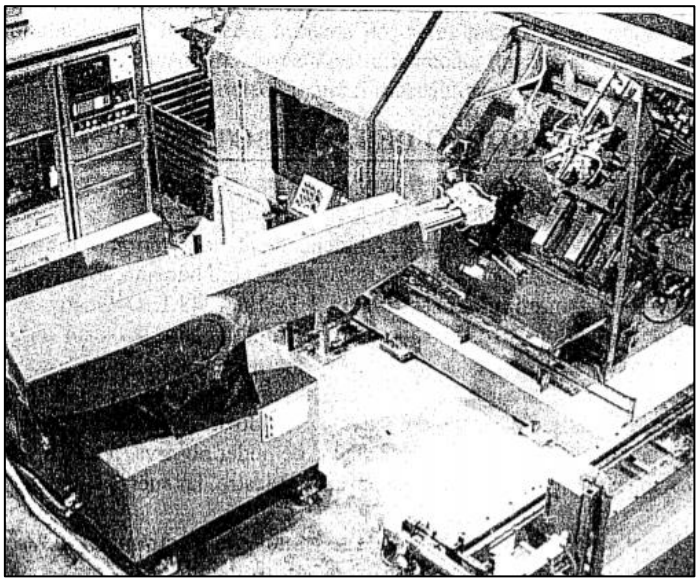


Figura 3. Robot Unimate 2000.

La empresa fue creciendo exponencialmente debido a los acuerdos firmados con Kawasaki, y de esta manera Japón fue creciendo en el ámbito de la robótica hasta ponerse a la cabeza e incluso formando la primera asociación robótica del mundo, la Asociación de Robótica Industrial de Japón (JIRA) en 1972.

En cambio, en Europa fue todo más tardío, ya que el primer robot de accionamiento eléctrico llegó de la mano de la firma sueca ASEA en 1973.

Figura 4. Robot IRb6, primer robot europea de accionamiento eléctrico.

Las primeras configuraciones de los robots fueron la esférica y la antropomórfica, de uso especialmente válido para la manipulación. En 1982, Maino desarrolla el concepto de robot SCARA (*Selective Complaince Assembly Robot Arm*) buscando un número reducido de grados de libertad ,3 o 4, un coste limitado y una configuración orientada al ensamblado de piezas.

La evolución de los robots industriales ha sido muy grande desde que se desarrolló el primero, ya que en 30 años los robots toman posición en casi todos los tipos de industria y sus diferentes áreas.

En la actualidad la mayoría de los robots son de base estática, pero ya han empezado a desarrollarse aquellos que son móviles, que se utilizan sobre todo para el ámbito no industrial. Así que, al modo en el que avanza esta tecnología, puede que en un futuro los robots industriales se parezcan a los robots de las novelas de Asimov o Capek, que les dieron nombre a los actuales herederos de Unimate. (1)

3.2. Visión artificial

Según la definición dada por *Automated Imaging Association (AIA),* la visión artificial abarca todos las aplicaciones industriales y no industriales en las que una combinación de hardware y software brindan un guiado operativo a los diferentes dispositivos en la ejecución de sus funciones de acuerdo con la captación y procesamiento de imágenes.

Para el ámbito industrial se exige mayor robustez, fiabilidad y estabilidad que para otros ámbitos, debido al entorno en el que son usados. Las características para este sector es que deben tener una precisión aceptable, resistencia elevada, alta fiabilidad y una gran estabilidad mecánica y de temperatura.

Estos sistemas constan con sensores digitales protegidos en el interior de cámaras industriales con ópticas especializadas para adquirir imágenes, y a parte pueden realizar medidas objetivas.

Por tanto, un sistema de visión artificial se caracteriza por la resolución de la cámara y la óptica, ya que son las que permiten la inspección de los objetos de una forma más detalla que la que se lograría con el ojo humano.

Una de las mayores ventajas es la aportación de beneficios operativos y seguridad adicional al reducir la participación humana en los procesos industriales, ya que protege a las personas frente a entornos peligrosos y elimina la contaminación humana en el proceso.

Normalmente, el primer paso en cualquier aplicación de visión artificial es la realización de patrones para poder buscar la característica u objeto dentro del campo de visión, esto suele marcar el éxito o fracaso. Por ello, las herramientas de localización del sistema debe ser lo suficiente mente eficaz para comparar de forma rápida y precisa los patrones de formación con los objetos captados. A continuación, se nombrar algunos de los procesos industriales donde se hace uso de los sistemas de visión:

- Guiado: el sistema de visión puede localizar la posición y orientación de un objeto, compararla con una tolerancia especificada, y garantizar el ángulo.

- Identificación: el sistema de visión lee código de barras, códigos de matrices de datos, marcajes directos en piezas o caracteres impresos en ellas.
- Medición: el sistema de visión calcula la distancia entre dos o más puntos o ubicaciones geométricas de un objeto y determina si esas medidas cumplen las especificaciones.
- Inspección: el sistema de visión detecta imperfecciones o defectos en las piezas que capta la óptica. (2)

3.3. Protocolo USB

El *Universal Serial Bus* (USB) se trata de un sistema de comunicación entre dispositivos electrónicos-informáticos que solo transmiten una unidad de información a la vez, a parte, de permitir la conexión sin configuración y sin reinicio del sistema de dispositivos externos al PC de forma transparente al usuario.

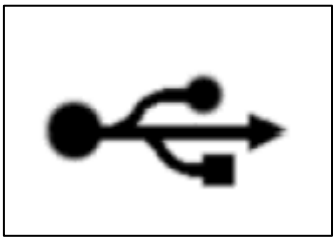


Figura 5. Logotipo de USB

El bus sigue una estructura de árbol descendente, con múltiples dispositivos conectados a un mismo bus, en la que unos elementos llamados *hubs* encaminan las señales desde un dispositivo al *host* y viceversa.

Dentro de la terminología USB se define *Host* al PC que soporta el bus, *Hubs* como elementos para conectar y utilizar USB, mientras que *Function* son todos los periféricos.

Solo existe un único *Host* que es mismo PC, aunque en específico la parte que controla al USB y hace de interfaz entre el PC y los distintos positivos externos. Este se provee de uno o dos puntos de conexión a partir de los cuales y de forma ramificada se conectan los periféricos. Otra de sus funciones es detectar si hay un puerto en uso

o no y notificándoselo al controlador del *Host* para que este remueva las estructuras de datos y programas de los *drivers* del dispositivo retirado.

Por ello se pasa a la definición del controlador del *Host* que es la interfaz entre el bus USB y el bus del PC, del que cuelgan los dispositivos USB, como máximo se pueden conectar 127 dispositivos a un solo controlador. Este se hace responsable, a nivel hardware, de los siguientes aspectos dentro del sistema USB:

- Detección de la conexión/desconexión de los dispositivos, a parte de sus respectivas configuraciones.
- Administrar y controlar el flujo de datos.
- Administrar y regular los flujos de control.
- Recolectar y resumir estadísticas de actividad y estado de los elementos del sistema.
- Proveer de energía eléctrica a los dispositivos.

Mientras que a nivel software se hace responsable de:

- Enumeración y configuración de los dispositivos conectados al sistema.
- Administración y control de transferencias isocrónicas y asincrónicas de información.
- Administración avanzada de suministro eléctrico a los diferentes dispositivos.
- Administrar la información del bus y de los dispositivos USB.

Los *Hubs* simplifican la interconexión de dispositivos al PC. Por tanto, se tratan de concentradores cableados que permiten múltiples conexiones simultáneas, de est forma se puede ampliar el número de puertos disponibles. Un PC normalmente tiene dos puertos, para llegar a los 127 máximos se debe emplear el uso de esta tecnología.

Los *Function* son todos los dispositivos, diferentes de los *hubs*¸ que son conectados al bus y estos son capaces de recibir y transmitir información. Todas las funciones USB tienen un cable y un conector, fabricado y diseñado siguiendo las especificaciones del bus permitiendo así la compatibilidad.

El USB está formado por un cable de cuatro hilos donde dos de ellos son para datos y los restantes para alimentación, permitiendo un ahorro tanto espacial como de

material (Figura X). Cada dispositivo puede tener un cable de hasta 5 metros, frente a 1 metro para los puertos serie y 4 metros para el puerto paralelo, si se conecta todo en cadena la máxima distancia que se puede alcanzares de 635 metros.

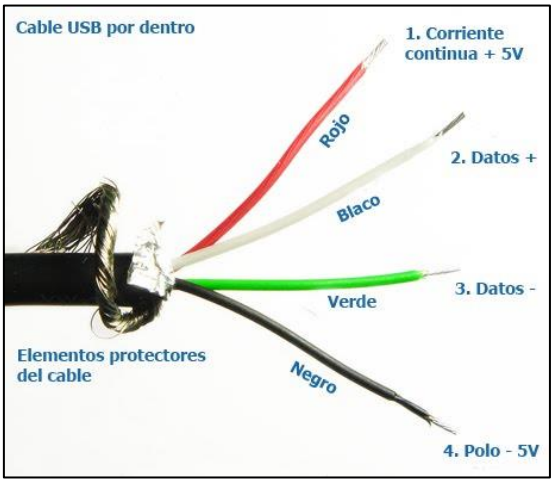


Figura 6. Cable USB.

Luego estos cables tienen unos conectoras, que dependiendo de la función que se quiera desarrollar tienen las siguientes formas: (3)

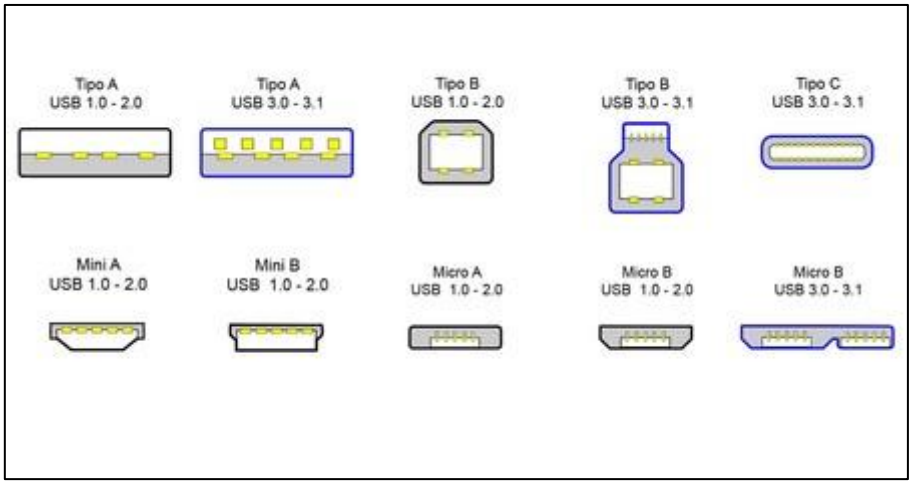


Figura 7. Tipos de puertos USB.

La comunicación se produce cuando el dispositivo externo se conecta por primera vez iniciando un proceso llamado “proceso de enumeración”. Este consiste en que el *host* se comunique con el dispositivo para conocer su identidad y ver qué tipo de controlador se requiere.

La enumeración empieza mandando una señal de reinicio al dispositivo USB y mediante la señalización de reinicio se determina la velocidad de este. Después del reinicio el host lee la información y le asigna al dispositivo una identificación de 7 bits única. Si el dispositivo es compatible con el *host* se cargan los controladores necesarios para la tarea de comunicación y se establece un estado de configuración. Este proceso se lleva a cabo tanto para la conexión como para la desconexión del periférico.

La comunicación va desde el host hasta un punto final del dispositivo periférico que su dirección es definida por 4 bits. Este punto final se trata de una parte únicamente direccionable por el periférico que es la fuente o el receptor de los datos y el punto final 0 está reservado para las transferencias de control, todos los dispositivos deben permitir este punto 0.

Cada comunicación con el periférico devuelve un descriptor que es una estructura de datos que informa al host sobre la configuración y las expectativas del punto final.

Los dispositivos solo pueden transmitir con a orden del controlador *host*, que este siempre está buscando tráfico.

USB puede admitir cuatro modos de transferencia:

- Control: transfiere la configuración de intercambio, la configuración y la información de comando entre el *host* y el dispositivo externo.
- Isócrono: para dispositivos de tiempo crítico. Se garantiza el tiempo al bus debido a que es información de tipo sensible, ya que depende del tiempo. Los datos son transmitidos en tiempo real y no se comprueban los errores.
- *Bulk* o masiva: se reciben paquetes muy grandes de datos, por tanto, reclaman todo el ancho de banda disponible.
- Interrupción: solo para dispositivos que intercambian pocos datos pero que necesitan atención inmediata.

El *host* solo permite una utilización del ancho de banda del 90% para las transferencias de tipo isócrono o de interrupción, llegando a este límite no se pueden configurar más.

El *host* es el que inicia la comunicación mediante un primer paquete denominado Token, que se trata de la configuración del enlace. Luego va precedido por un paquete de datos que lleva la información de contenido y va seguido por un paquete de reconocimiento, que informa si se han recibido con existo.

Estos paquetes USB pueden estar formados por los siguientes campos:

- Sync (campo de sincronización): lo incluyen todos los paquetes al principio. Pueden tener una longitud de 8/32 bits según su velocidad y es utilizado para la sincronización de los relojes. Sus últimos 2 bits indican cunado empieza el siguiente campo.
- PID: identificador del paquete, 8 bits de los cuales los que identifican son 4 los otros están para detectar errores.
- ADDR: dirección específica del receptor del paquete, tiene 7 bits (127 dispositivos).
- ENDP: puntos finales, formado por 4 bits.
- CRC: verificación de redundancia cíclica para la detección de errores.
- EOP: fin de paquete.

Los tipos de paquete USB son:

- Paquetes de protocolo de enlace, se envían en respuesta a paquetes de datos.

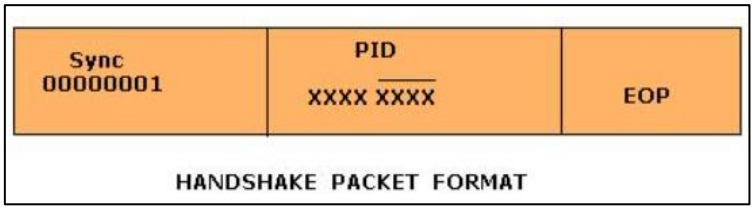


Figura 8. Estructura para paquetes de protocolo enlace.

- Paquete de token, permite la definición de los datos transmitidos, es generado por el *host*, formado por 11 bits de dirección y 5 bits para el CRC. Hay tres tipos diferentes de t*okens*:
    - *In token*: informa de que se desea leer información.
    - *Out token*: informa de que se desea enviar información.
    - *Token* de configuración: para comenzar las transferencias de control.

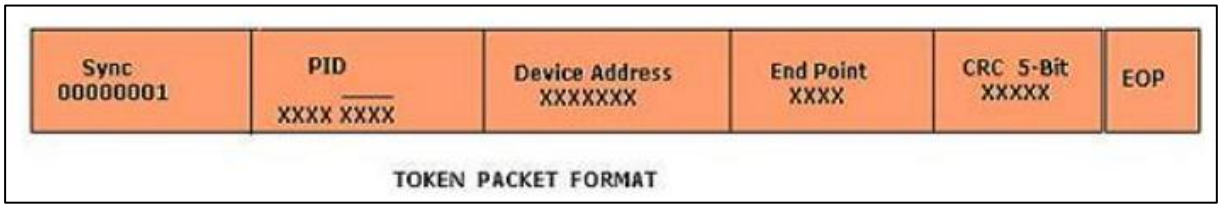


Figura 9. Estructura para paquetes de token.

- Paquete de datos: formado por dos paquetes, DATA0 y DATA1. Tienen disponibles 1023 bits de carga útil de datos y 16 de CRC y siempre precedidos por un token de dirección.

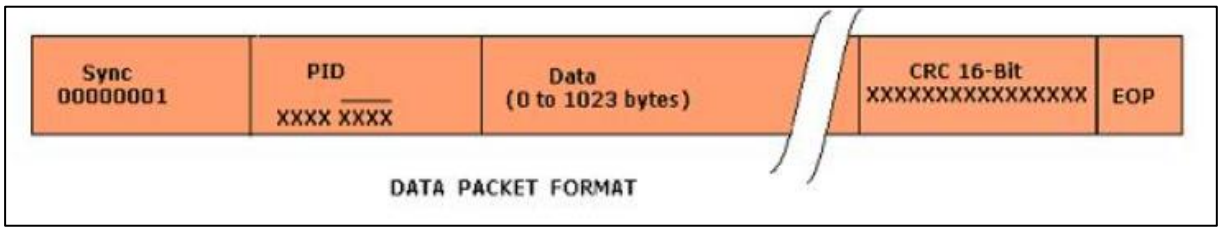


Figura 10. Estructura de paquetes de datos.

- Paquete PRE: exclusivo para paquetes de poca velocidad.
- Inicio de paquetes trama: cada 1ms el *host* envía un token especial que sirve para sincronizar los flujos de datos isócronos. (4)

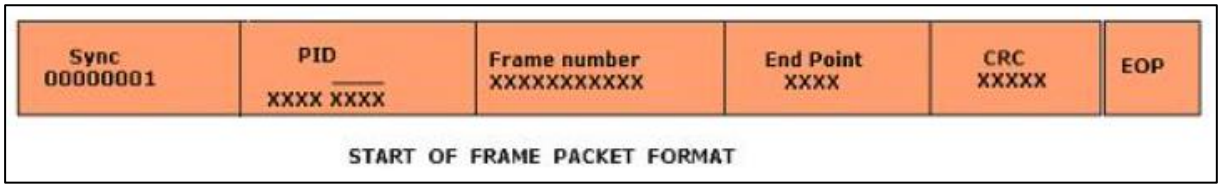


Figura 11. Estructura de paquetes PRE.

3.4. Objetivos

El objetivo principal a alcanzar en este estudio es la utilización flexible y robusta de visión artificial en un robot industrial demostrándolo con algún proceso que, aunque se ha desarrollado en un entorno prototipo, sea similar al de cualquier entorno industrial.

Para ello se realizan varias pruebas donde se muestra el funcionamiento de la identificación y captación de piezas de diferentes formas en un entorno controlado. Estas pruebas subdividen el objetivo principal en los siguientes:

- Calibración del brazo robótico respecto a la cámara para devolver las coordenadas de las piezas detectadas respecto a la posición del robot.
- Procesamiento de imágenes para la realización de modelos para la búsqueda de piezas.
- Movimiento del robot a la posición de la pieza.
- Creación de nuevos modelos no realizados y guardados previamente.
- Recolección de varias piezas captadas por la cámara acoplada al brazo robótico.

Finalmente, el estudio se pondrá a prueba con la realización de un puzle mediante una agrupación de los objetivos anteriores y sus respectivas pruebas, logrando demostrar el objetivo principal del estudio. Esta última prueba se asemeja a la clasificación de piezas de diferentes formas en un entorno industrial.

4. TECNOLOGÍA Y HERRAMIENTAS

A lo largo de este apartado se desarrollan los diferentes elementos como las aplicaciones software usadas durante el estudio del proyecto, para alcanzar los diferentes objetivos propuestos.

4.1. Tecnología usada:

4.1.1. Robot SCORBOT ER-4u de Intelitek

Se trata de un robot educativo, creado por la empresa Intelitek, con forma de brazo robótico y controlado mediante USB que puede ser montado en una mesa, un pedestal o una base deslizante. Es un robot apto tanto para uso integrado en celda automatizadas como para operaciones independientes.

Figura 12. Robot SCORBOT ER-4U

Este es controlado mediante un software de control y programación llamado SCORBASE, explicado en el siguiente apartado, y permite la simulación dinámica del

robot y las células de trabajo durante la enseñanza de posiciones y ejecución del programa.

El brazo robótico conta de 5 ejes más una pinza que son controlados mediante servomotores DC, con control de lazo cerrado y encoders ópticos. Este brazo está diseñado para una máxima seguridad, poseyendo características de seguridad multipunto tales como detección, protección de impactos, protección de sobrecarga de motores, velocidades variables y paradas de emergencia, que sirven para evitar daños en el usuario o al propio robot.

Las características técnicas del robot Scorbot ER-4u son: (5)

Brazo mecánico	
Estructura mecánica	Brazo vertical articulado; estructura abierta.
Grados de libertad	5 ejes rotacionales y pinza.
Capacidad de carga	2.1 Kg
Movimiento de los ejes: Eje1: Base Eje 2: Brazo inferior Eje 3: Brazo superior Eje 4: Elevación pinza Eje 5: Giro pinza	 310º +135º/-35º +130º +130º ± 570º
Alcance	610mm con pinza
Velocidad	700 mm/seg
Repetibilidad	±0.18 mm
Realimentación	Enconders ópticos incrementales
Home	Microinterruptor en cada eje
Actuadores	Servomotes 12V DC en cada eje
Pinza	Servomotor DC, 2 dedos paralelos
Abertura de la pinza	65/75 mm con/sin almohadillas de goma
Transmisión	Engranaje correa dentada
Peso	10.8 Kg

Figura 13. Características del brazo robótico Scorbot ER-4u.

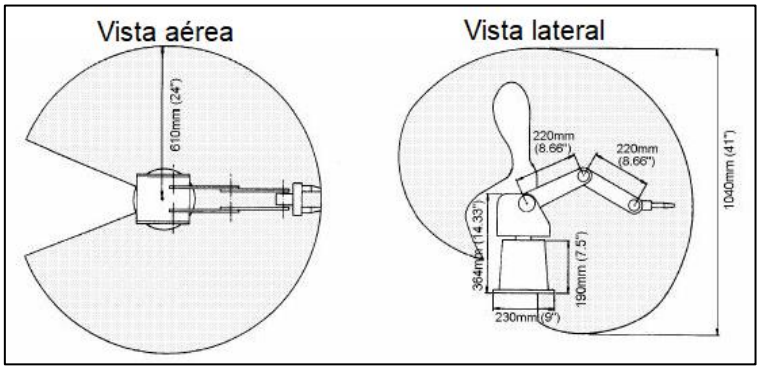


Figura 14. Movimiento de los ejes en Scorbot ER-4u.

Controlador	
Comunicación	USB tipo A cable conexión al PC *Plug and play* sin arrancar.
Entradas/Salidas	8 entradas digitales; 4 entradas analógicas; 8 salidas digitales (4 relés, 4 colector abierto); 2 salidas analógicas.
Microcontrolador	NEC V853 RISC 32-bit
Servo Control de Ejes	Tiempo-real, PID, PWMN
Número de servo ejes	8 (brazo del robot, pinza y 2 periféricos)
Memoria de usuario	Programas, líneas de programa, variables y posiciones ilimitadas.
Definición de Posición	Absoluta, Relativa, Cartesiana, Ejes (Joints), Encoders
Control de Trayectoria	Ejes (Joint), lineal, circular
Definición de Velocidad	10 configuraciones de velocidad, definición del tiempo de movimiento.
Parámetros de Control	160 parámetros accesibles para el usuario
Características de seguridad.	Interruptor de emergencia, protección de impacto, cierre automático en detección de impacto, sobrecalentamiento, fallo del PC o error de comunicación

Figura 15. Característica de los controladores del Scorbot ER-4u.

4.1.2. Cámara de visión NGS SwiftCam 300

Se hace uso de una cámara de la marca NGS con sensor CMOS de 300Kpx y conexión USB 2.0. También dispone de micrófono incorporado y capturador de imágenes y vídeos lo que la hace ideal para el estudio de este proyecto. Dispone de una base flexible ajustable para poder colocarla en el brazo robótico.

A continuación, se exponen sus características más técnicas:

Detalles técnicos	
Marca	NGS
Fabricante	NGS
Series	Swiftcam300
Dimensiones	22.8 x 17.8 x 8.8 cm
Peso	272 g
Color	Negro, plata
Resolución de pantalla	5MP
Sistema operativo	Windows Vista, XP, 2000

Figura 16. Detalles técnicos de la cámara NGS SwiftCam 300.

Figura 17. Webcam NGS SwiftCam 300.

4.2. Softwares utilizados:

4.2.1. SCORBASE de Intelitek

El SCORBASE para SCORBOT-ER 4u se trata de un software de control de robótica para la programación y la operación del robot. Este ofrece numerosas funciones:

- Comunicación mediante canal USB con el controlador del robot.
- Control y visualización del estado, en tiempo real, de los ejes del robot, la pinza y los dos ejes periféricos.
- Control y visualización del estado, en tiempo real, de las ocho entradas y salidas digitales y sus cuatro entradas y salidas analógicas.
- Definición y visualización de la posición y del movimiento manual del robot del robot en base al Sistema de Coordenadas Articulares.
- Sistemas de coordenada cartesianas.
- Definición del movimiento de robot con configuración de velocidad.

Una vez instalado el programa se puede observar una primera interfaz gráfica (Figura X) donde se ven las siguientes funciones:

- Barra de menú que contiene todos los menús y opciones.
- Barra de herramientas de iconos de las funciones más utilizadas.
- Barra de estado para visualizar la información del software de SCORBASE.

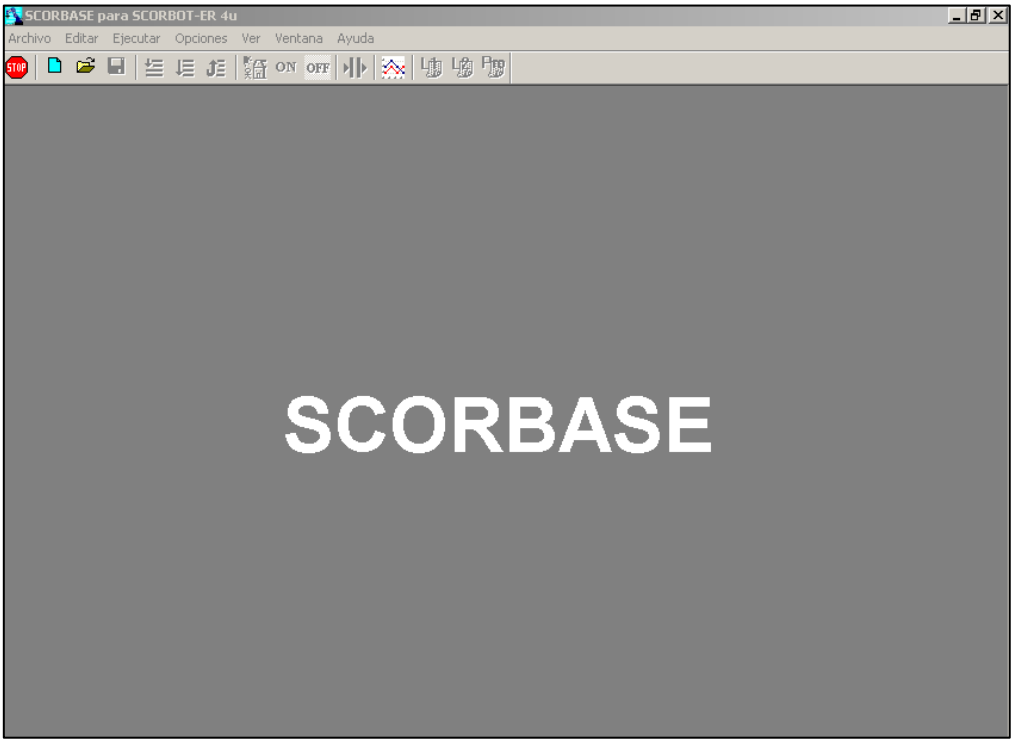


Figura 18. Interfaz inicial del SCORBASE.

Para la calibración del robot con este programa se realiza mediante un procedimiento de *homing* que se encarga de realizar todos los movimientos y posiciones permitidas de este.

Esta función se realiza por el comando *Search Home All Axes,* donde se abre una interfaz que muestra todos los ejes e indica cuando el eje ha sido retornado con éxito.

Figura 19. Opción de *Search Home All Axes*

Figura 20. Interfaz de *Search Home All Axes*

Este robot, como ha sido explicado en el apartado anterior, consta de una serie de elementos que permiten su movimiento en coordenadas articulares y cartesianas. Estos son definidos a través de cinco parámetros derivados de los datos suministrados por los codificadores de los cinco ejes.

En el sistema de coordenadas articulares la posición del robot se define por valores de ángulos y que representan cada articulación de este.

A continuación, se ponen los grados que tiene cada uno de ellos en la posición de *homing:*

- Base: 0º
- Hombro: -120º
- Codo: 95º
- Inclinación: 88º
- Rotación: 0º

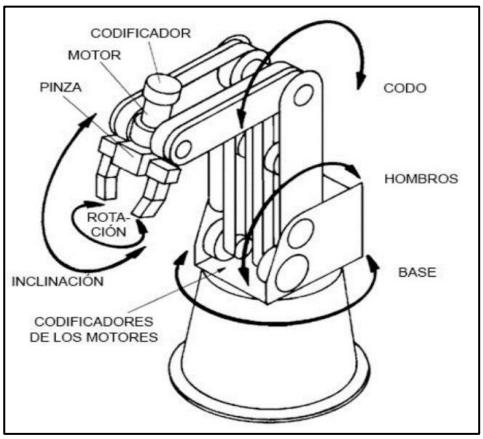


Figura 21. Robot – coordenadas articulares

En cambio, para el sistema de coordenadas cartesianas la posición viene definida en por un espacio tridimensional (X, Y, Z) colocado en la base central del robot. A parte, tiene el elemento Pitch (P) y Roll (R) de la pinza que son definidos en ángulos.

A continuación, se ponen los grados que tiene cada uno de ellos en la posición de *homing:*

- X: 169 mm
- Y: 0 mm
- Z: 503 mm
- P: -63º
- R: 0º

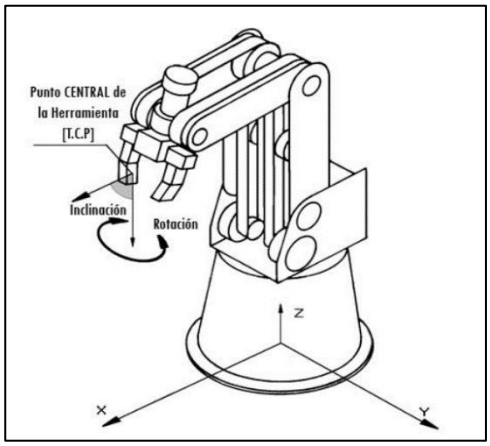


Figura 22. Robot – Coordenadas Cartesianas

Una vez definidas el tipo de coordenadas se pasa a definir qué tipo de posiciones ofrece el software, que en este caso son posición relativa y posición absoluta.

La primera de ella se define usando los cinco parámetros del robot y se trata de una posición fija en el espacio mundial. Mientras que el segundo tipo de posición se trata de un desplazamiento de coordenadas sobre una que están ya predefinidas, por tanto, si la posición de referencia cambia esta posición relativa también lo hará.

El robot puede ser movido a través de una interfaz que ofrece el software SCORBASE, mediante el teclado del ordenador, o puede aprenderlo mediante una instrucción en el programa (se verá más adelante).

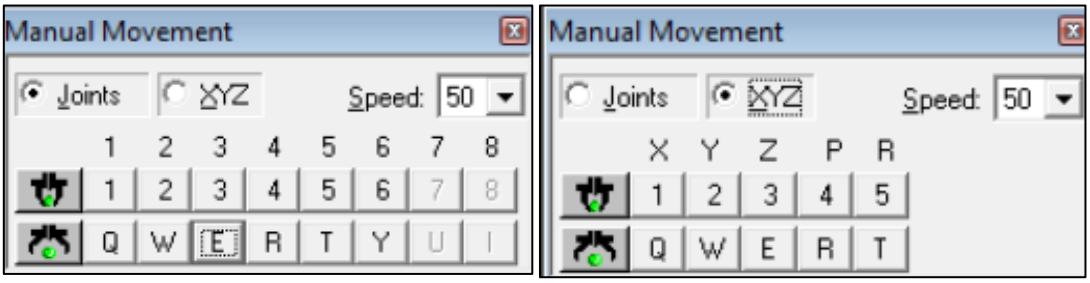


Figura 23. Interfaz de movimiento manual del robot.

Teclas	Movimiento XYZ	Movimiento Joints
1/Q	La herramienta se mueve a lo largo del eje X.	Gira la BASE.
2/W	La herramienta se mueve a lo largo del eje Y.	Mueve el hombro.
3/E	La herramienta se mueve a lo largo del eje Z.	Mueve el codo.
4/R	Los ejes se mueven para cambiar el ángulo de inclinación de la pinza.	Mueve la muñeca.
5/T	La pinza gira.	Gira la muñeca
6/Y	---	Control de la pinza.

Figura 24. Definición de código para el movimiento manual del robot

Opciones de la pinza		
Icono	**Opción**	**Descripción**
	*Open Gripper*	Abre la pinza completamente.
	*Close Gripper*	Cierra la pinza completamente.

Figura 25. Definición de código para el movimiento manual de la pinza.

Ya una vez elegida la posición del robot se pasa a guardarla a través de la siguiente interfaz, aunque también es posible guardar una posición mediante una instrucción:

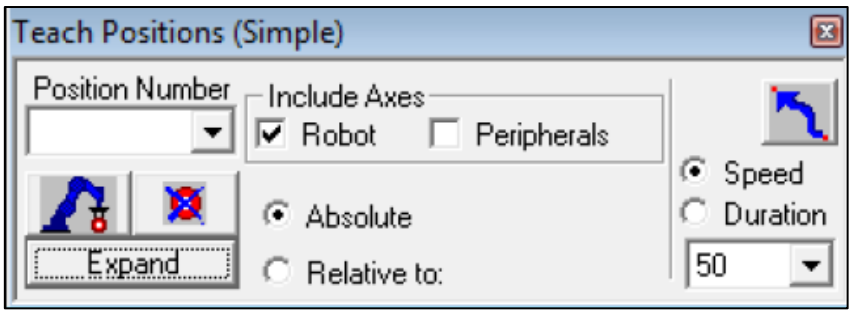
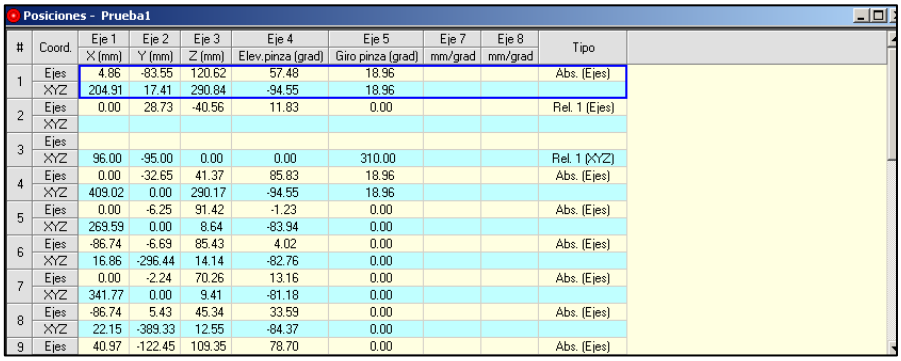


Figura 26. Interfaz para guardar las posiciones manualmente.

Opción	***Descripción***
Position Number	Nombre numérico para la posición
Record 	Registra la posición actual del en el cambo numérico.
Delete 	Borra de la memoria la posición del campo numérico
Go to position 	Envía al robot desde su posición actual hasta la mostrada en el campo numérico.
Speed	Selecciona la velocidad de movimiento del robot.

Figura 27. Definiciones de los comandos para guardar una posición.

Existe una opción que te muestra una interfaz para visualizar la lista de posiciones, esta se presenta en formato de tabla, esta se encuentra en *menú >> ver >> posiciones*.



#	Coord.	Eje 1 X (mm)	Eje 2 Y (mm)	Eje 3 Z (mm)	Eje 4 Elev.pinza (grad)	Eje 5 Giro pinza (grad)	Eje 7 mm/grad	Eje 8 mm/grad	Tipo
1	Ejes	4.86	-83.55	120.62	57.48	18.96			Abs. (Ejes)
	XYZ	204.91	17.41	290.84	-94.55	18.96			
2	Ejes	0.00	28.73	-40.56	11.83	0.00			Rel. 1 (Ejes)
	XYZ								
3	Ejes								
	XYZ	96.00	-95.00	0.00	0.00	310.00			Rel. 1 (XYZ)
4	Ejes	0.00	-32.65	41.37	85.83	18.96			Abs. (Ejes)
	XYZ	409.02	0.00	290.17	-94.55	18.96			
5	Ejes	0.00	-6.25	91.42	-1.23	0.00			Abs. (Ejes)
	XYZ	269.59	0.00	8.64	-83.94	0.00			
6	Ejes	-86.74	-6.69	85.43	4.02	0.00			Abs. (Ejes)
	XYZ	16.86	-296.44	14.14	-82.76	0.00			
7	Ejes	0.00	-2.24	70.26	13.16	0.00			Abs. (Ejes)
	XYZ	341.77	0.00	9.41	-81.18	0.00			
8	Ejes	-86.74	5.43	45.34	33.59	0.00			Abs. (Ejes)
	XYZ	22.15	-389.33	12.55	-84.37	0.00			
9	Ejes	40.97	-122.45	109.35	78.70	0.00			Abs. (Ejes)

Figura 28. Interfaz de posiciones guardadas.

A continuación, se muestra la interfaz donde se realiza la programación del robot, llamado "*Workspace*":

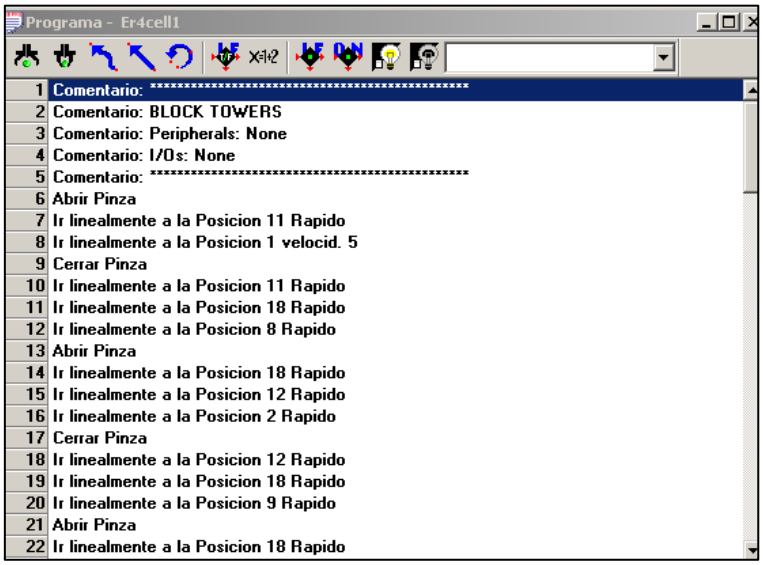


Figura 29. Interfaz del Espacio de Trabajo.

En la parte superior de la Figura 15 se puede observar una serie de figuras en forma de flecha. Estas son las que permiten mandar la instrucción de movimiento y cada una de ellas realiza el movimiento de diferentes formas:

-  : esta opción permite mandar el robot la posición seleccionada en el menor tiempo posible.

-  : esta opción hace que el movimiento del robot tenga una trayectoria lineal hasta llegar a la posición deseada.
-  : esta opción hace que el movimiento del robot tenga una trayectoria circular hasta llegar a la posición deseada.

Ya por último se muestra y define las instrucciones aplicables en este software, pero antes hay que saber que existen tres niveles de control según él nivel de conocimiento que se tenga. En este estudio todo se realiza a nivel profesional, ya que es únicamente donde aparecen las instrucciones para el control de Viewflex.

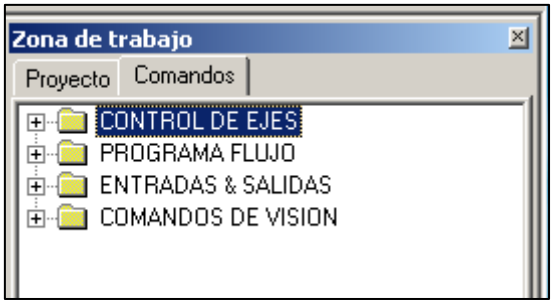


Figura 30. Interfaz de zona de trabajo.

Las instrucciones que a continuación se definen son las usadas para este estudio, aunque se muestran todas las disponibles, pero las que tienen que ver con el software Viewflex se definirán en el siguiente apartado:

- Instrucciones de la carpeta CONTROL DE EJES:

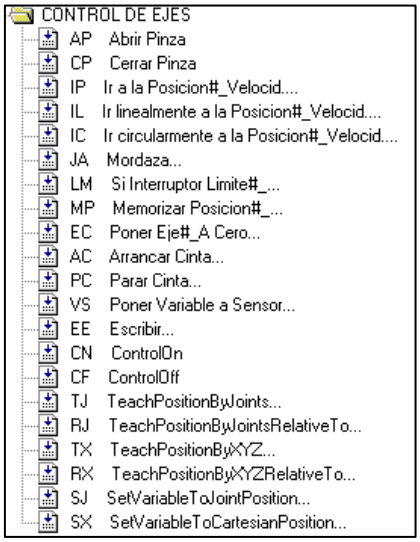


Figura 31. Instrucciones del “Control de Ejes”.

A continuación, se pasa a definir la instrucción que se utiliza de este bloque:

- RX *TeachPositionByXYZRelativeTo:* enseña una posición definida numéricamente en la coordenada cartesiana relativa a ya otra definida.

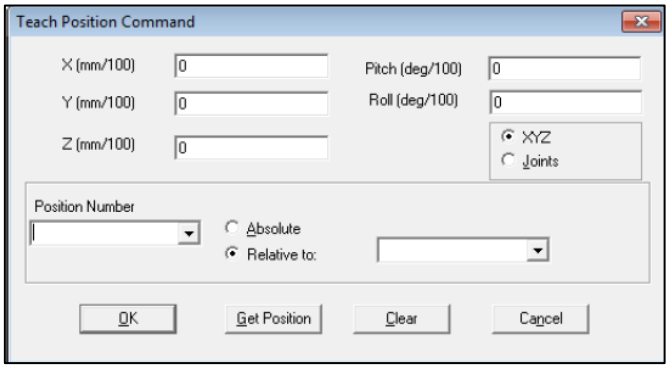


Figura 32. Interfaz – RX *TeachPositionByXYZRelativeTo.*

- Instrucciones de la carpeta CONTROL DE FLUJO:

Figura 33. Instrucciones de la carpeta “Programa de Flujo”

A continuación, se definen los comandos usados de este bloque:

- SA Salta A: comando de salto incondicional para que el puntero del programa salte a una línea especificada.

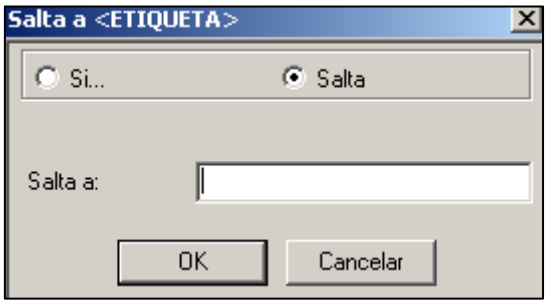


Figura 34. Interfaz – Salta A

- CO Comentario: pueden incorporarse hasta 47 caracteres de texto.

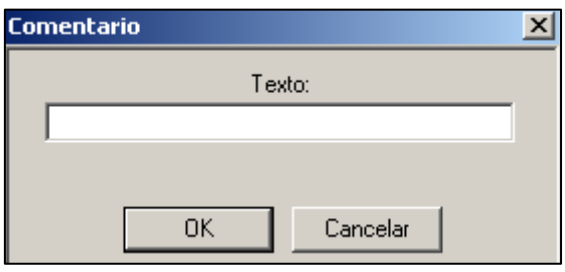


Figura 35. Interfaz – Comentario

- VC Poner Variable a Computar: permite asignar un valor o expresión a una variable. Se debe tener en cuenta que el nombre dela variable deben ser letras.

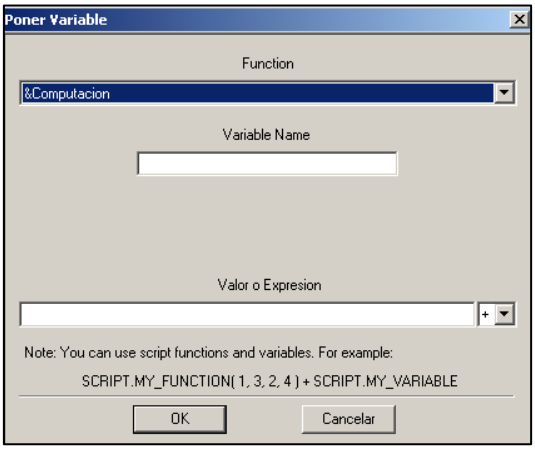


Figura 36. Interfaz – Poner Variable a Computar

- SS Si Salta: se utiliza para determinar el flujo de un programa en relación con el valor de las variables, si se cumple el campo “Si” el programa salta a la etiqueta del campo “Salta a”:

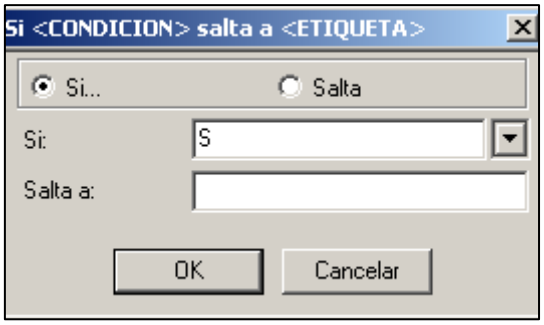


Figura 37. Interfaz – Si Salta

- ET Etiqueta: para crear una referencia para el comando “Saltar”.

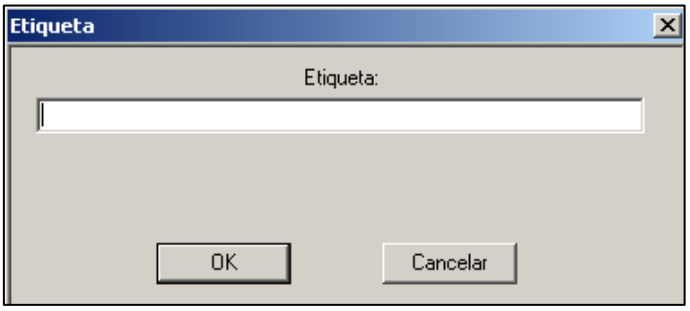


Figura 38. Interfaz – Etiqueta

- PL Imprimir a pantalla Log: imprimir los datos que contienen mensajes, secuencias y valores de variables en un archivo de registro o ventana de mensaje, o en ambos. (6)

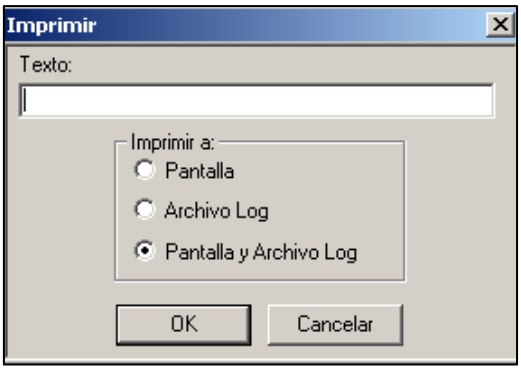


Figura 39. Interfaz – Imprimir a pantalla Log

4.2.2. Viewflex de Intelitek

Viewflex se trata de un software de visión utilizado en el SCORBASE para el análisis de imágenes a través del uso de la herramienta Matrox. En este estudio se utiliza la versión Viewflex para SCOBASE USB, que es el necesario para el uso del robot SCORBOT ER-4U.

Al iniciar el software la primera interfaz que aparece es una barra de herramientas (Figura 39) y que viene definida con diferentes funciones:

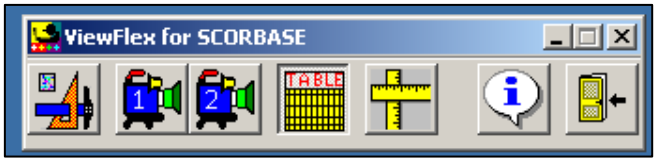


Figura 40. Barra de herramientas inicial de ViewFlex.

- Herramienta de procesamiento de imagen

  Con ella se abre la herramienta Matrox Inspector 8.0 que es la encargada de realizar todas las operaciones sobre la imagen.

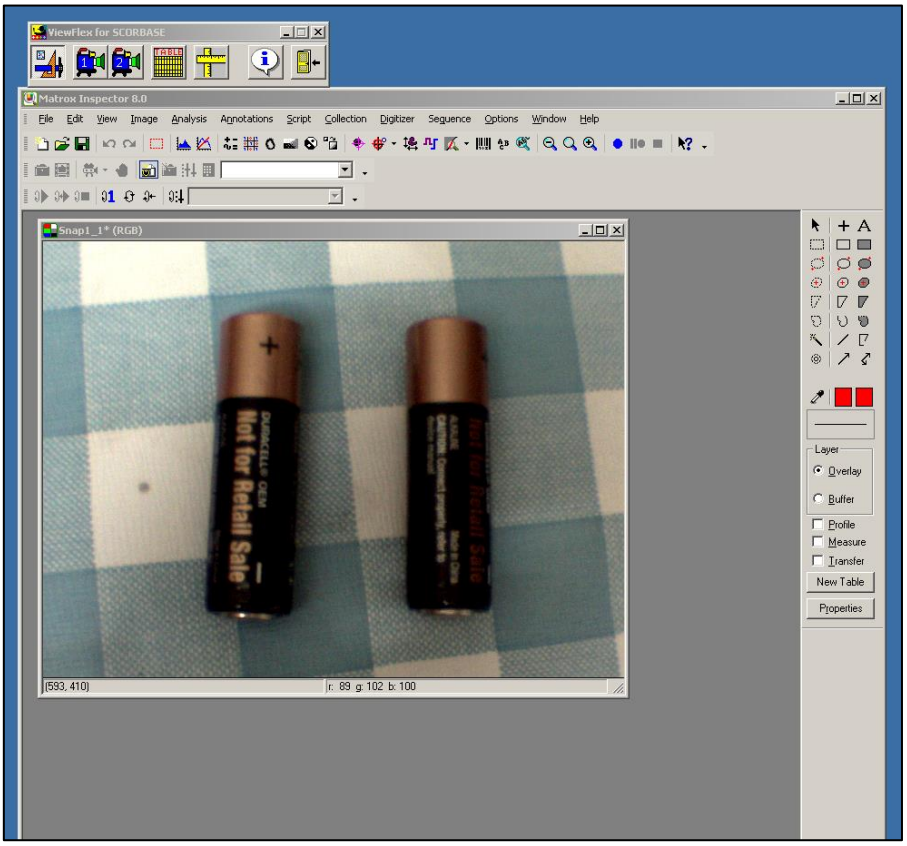


Figura 41. Interfaz del Matrox Inspector 8.0 con una captura de imagen.

En la figura anterior se puede observar la interfaz de Matrox que contiene muchas opciones referentes al procesamiento de imágenes. En la barra de herramientas superior se encuentran las opciones más utilizadas, que a lo largo de este documento se detalla más en profundidad las que serán utilizadas en el estudio. La barra de herramientas de la derecha se trata de herramientas de dibujo específicas para la realización de las secciones de los patrones.

- Cámara 1 y 2:

  Se abre una interfaz donde se ve reflejado la imagen captada por la cámara. En “Cámara 1”, utilizada en este estudio, da una visión normal de la imagen y permite la captura de la imagen, mientras que en “Cámara 2” o cámara cliente

se conecta al servidor mostrando en la interfaz la dirección IP del PC donde se encuentra la ServeCam.

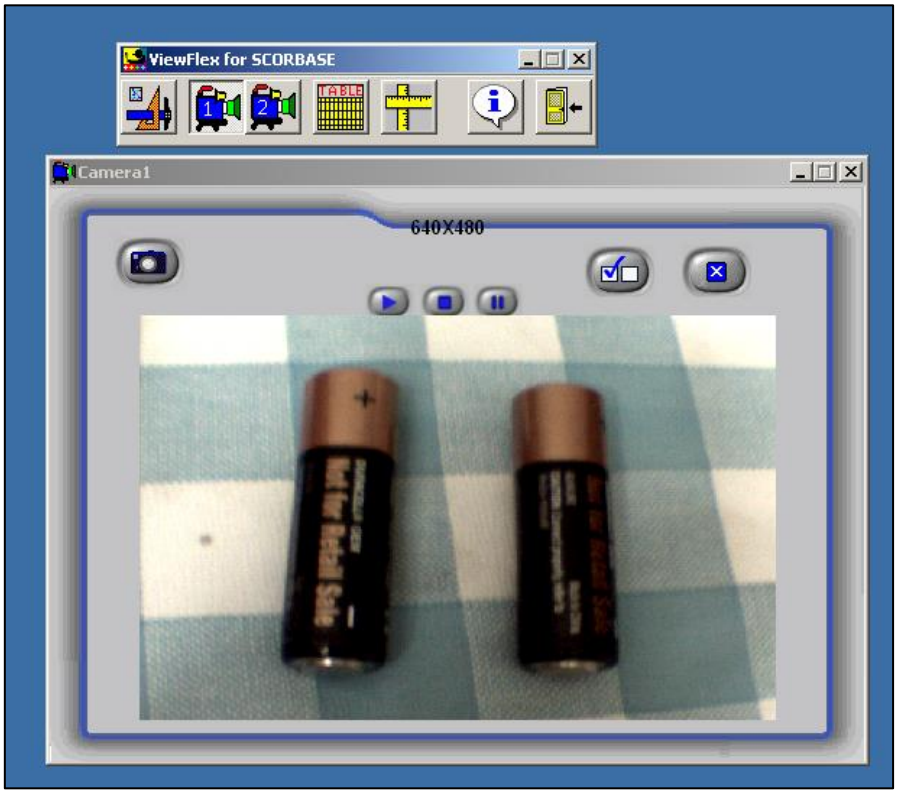


Figura 42. Interfaz de “Cámara 1” de ViewFlex

Se puede observar en la esquina superior izquierda de la interfaz la opción de una cámara, que es la que permite hacer las capturas manuales a la imagen captada por la cámara para posteriormente hacer el procesamiento de esta.

- Tabla de resultados:

  Se divide en una sección en forma de árbol donde se representan todas las carpetas de ViewFlex para su más fácil localización y otra parte donde aparece una tabla con las coordenadas del objeto en el entorno del robot, incluidas las características de la región conexa.

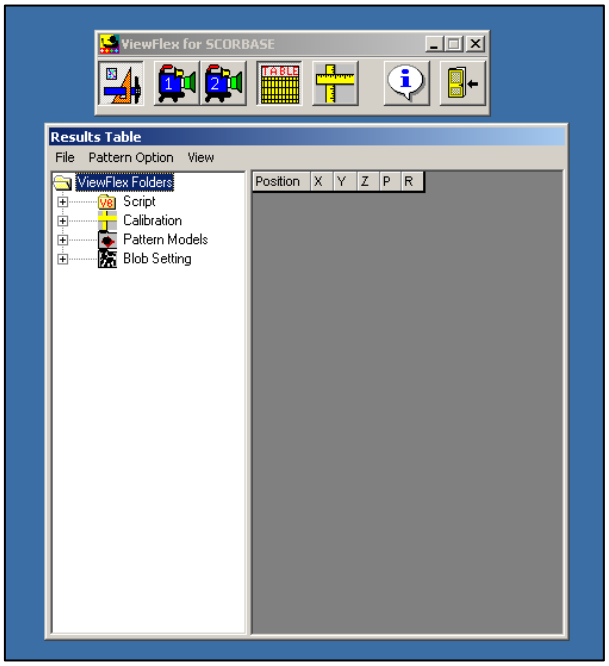


Figura 43. Interfaz de la Tabla de Resultados en ViewFlex

Cada vez que se guarda algún archivo nuevo en alguna de estas carpetas el árbol se actualiza con las nuevas informaciones. Las carpetas que conforman el árbol tratan de lo siguiente:

- *Scripts:* se ejecuta la primera función o subrutina que se encuentra en el *script.*
- *Calibration:* aparecen todas las calibraciones que se han realizado y guardado.
- *Pattern Models:* se trata de buscar un objeto mediante un patrón ya definido en la captura de imagen actual.
- *Blob Stting:* se trata de buscar las regiones conexas con el Blobset de la imagen actual.

- Calibración:

  Inicialmente se abre una interfaz donde se definen dos posiciones del robot en el mundo real, el proceso de calibración se explica en la fase de “BANCO DE PRUEBAS Y DISCUSIÓN”.

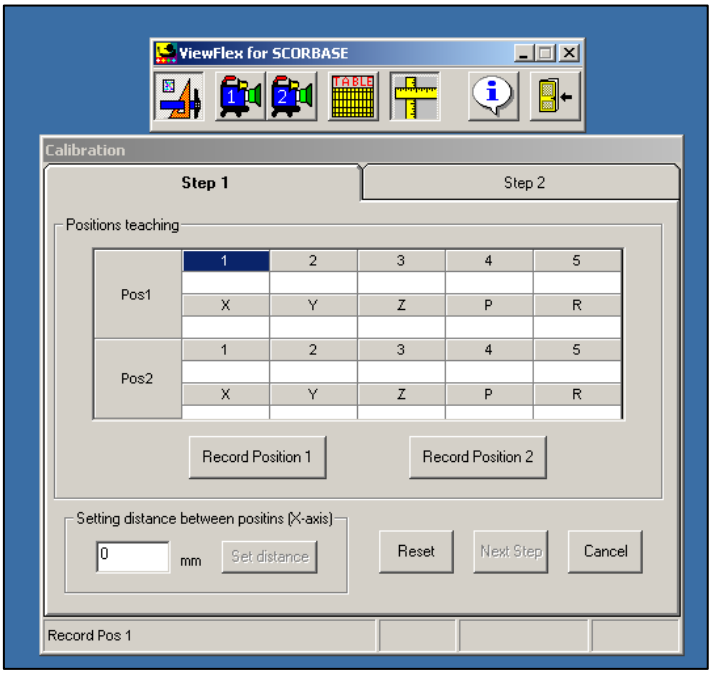


Figura 44. Interfaz de calibración en ViewFlex

- Ayuda: buscar ayuda acerca de algo referente al software.
- Salida: para cerrar el software.

Existen unos comandos específicos para el control de ViewFlex a través del software SCORBASE, estos comandos son los siguientes:

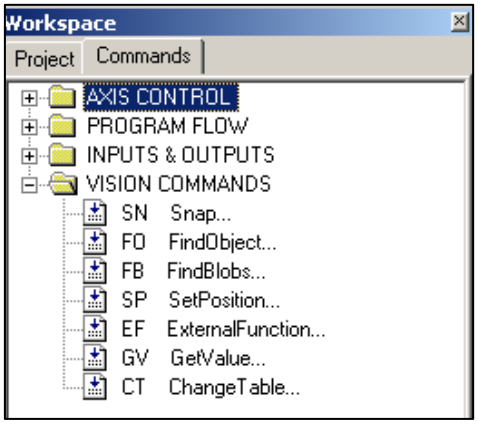


Figura 45. Instrucciones de "*Vision Commads*"

- SN *Snap*: permite hacer una captura de imagen, dependiendo de la versión del software de SCORBASE que se utilice se debe de especificar el número de la cámara con el que se desea realizar la captura.

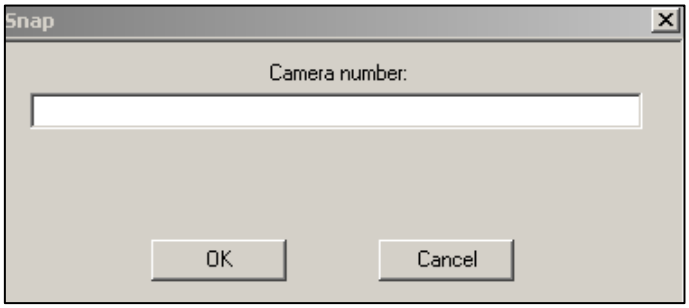


Figura 46. Interfaz – *Snap*

- FO *Find Object*: busca un objeto según el nombre del modelo de diseño que ha sido guardado en la carpeta "*Pattern"* y devuelve el número de instancias del objeto.

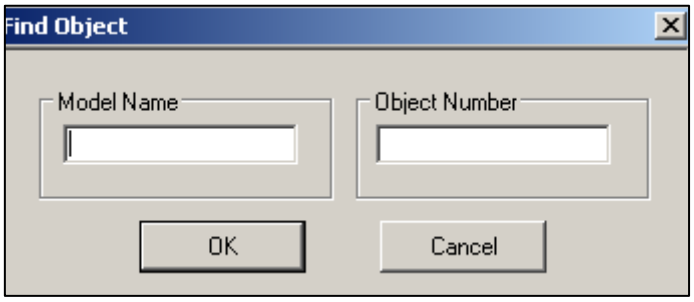


Figura 47. Interfaz – *Find Object.*

En la parte de "*Object Number*" se debe indicar el nombre de la variable donde se desea guardar el número de ocurrencias, mientras que en el campo "*Model Name"* se indica el nombre del modelo a buscar.

- FB *Find Blobs*: busca regiones conexas según el nombre del análisis de regiones conexas que se ha guardado en la carpeta "*Blobs"* y devuelve el número de ocurrencias.

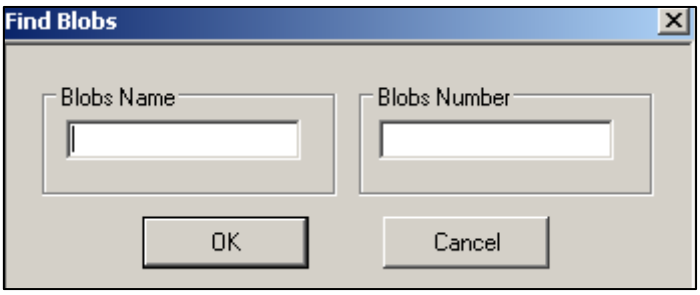


Figura 48. Interfaz – *Find Blobs*

El orden de las casillas para colocar los datos para utilizar esta instrucción se asemeja a la explicada en la instrucción anterior (*Find Objects*)

- SP *Set Position*: coloca el valor de la coordenada de la Tabla de Resultados en el número de posición deseado en el SCORBASE.

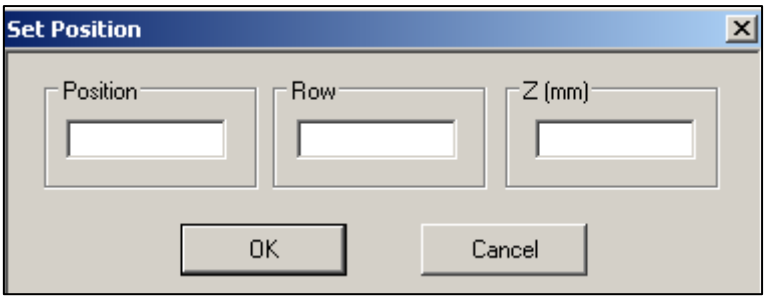


Figura 49. Interfaz – *Set Position*

En el primer campo se introduce la posición donde se quiere guardar los valores obtenidos, en el siguiente campo se introduce el número de ocurrencia del que se desean conocer los datos y por último se introduce el valor de la coordenada Z, que por defecto es el mismo que el de la calibración.

- EF *External Function:* ejecuta la función o subrutina que el usuario ha creado en un archivo *script* de Matrox.

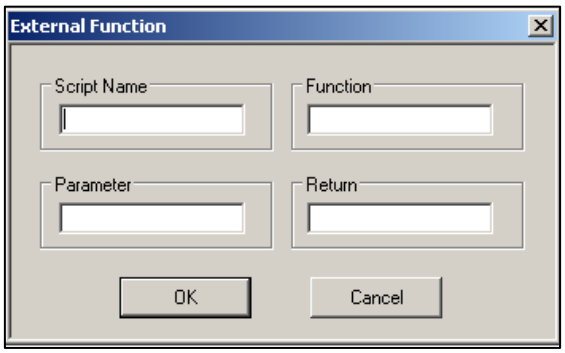


Figura 50. Interfaz – *External Function*

En los dos campos superiores se introduce el nombre del *script* y la función, mientras que en los dos campos inferiores se escribe la variable de la función y el parámetro donde desea ser devuelto el resultado.

- GV *Get Value*: recibe un valor de cualquier celda de la tabla de resultados.

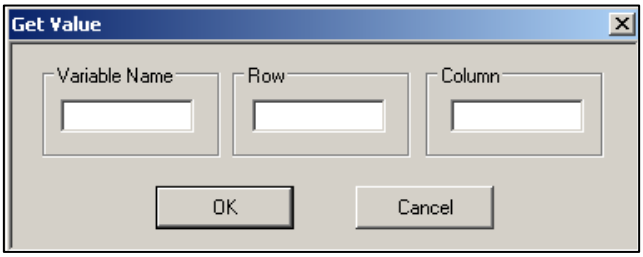


Figura 51. Interfaz – *Get Value*

Los campos se rellenan en el siguiente orden: nombre de la variable donde se devuelve el valor de la celda, insertar el número de la fila de la celda de la Tabla de Resultados y por último se inserta el nombre de la columna de la Tabla de Resultados.

- CT *Change Table*: cambia el valor en cualquier celda de la tabla de resultados.

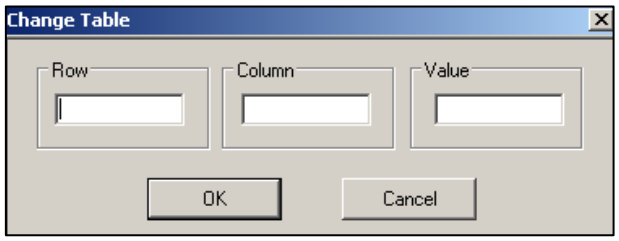


Figura 52. Interfaz – *Change Table*

Los campos se rellenan de la siguiente manera: en el primero se inserta la fila de la celda de la Tabla de resultados, en el siguiente se inserta el nombre de la columna de la Tabla de Resultados y por último se inserta la variable que se desee obtener de la tabla.

Una vez explicado el uso de programa ViewFlex solo hace falta añadir que las imágenes capturadas aparecen por defecto en el modelo de color RGB y la realización de modelos de patrones se hacen en blanco y negro. (7)

5. METODOLOGÍA Y PLANIFICACIÓN

5.1. Metodología:

Este apartado se subdivide en la definición de las diferentes tareas que han sido necesarias realizar para alcanzar los objetivos de este estudio, los resultados de estos

serán explicados más detalladamente en siguiente capítulo llamado “BANCO DE PRUEBAS Y DISCUSIÓN”.

Las pruebas realizadas siguen el siguiente flujo de trabajo:

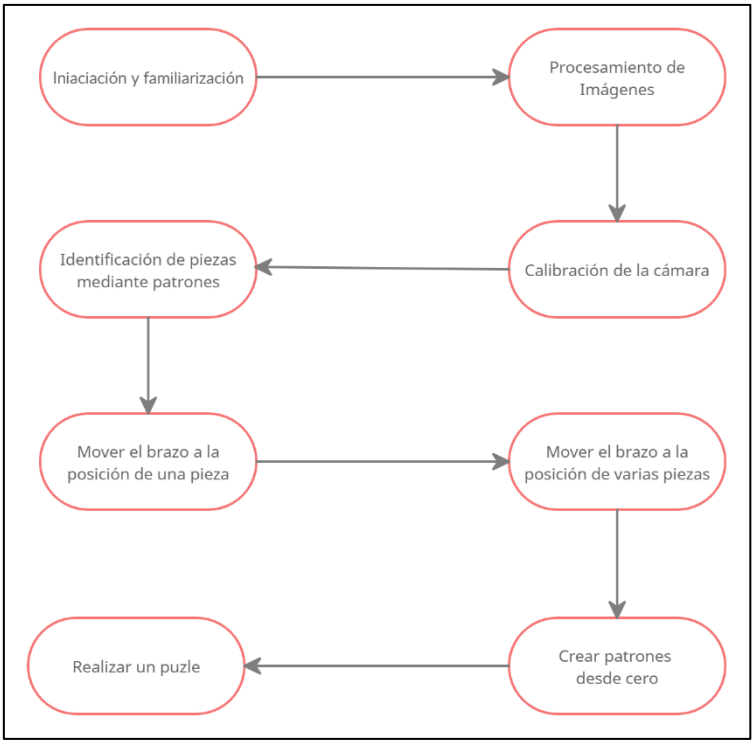


Figura 53. Diagrama de flujo de las tareas del estudio

Se debe de tener en cuenta que a lo largo de todas las pruebas realizadas se debe de tener que todas sus capturas se hacen en una posición estática de la cámara.

5.1.1. Iniciación y familiarización con el entorno ViewFlex:

El estudio del proyecto empieza por una familiarización de los entornos software que van a ser utilizados, para ello se hace uso de los manuales correspondientes y se van probando las diferentes funciones que ofrecen los programas.

5.1.2. Procesamiento de imágenes para modelos con cámara desacoplada del robot:

Se realizan varias pruebas de captura de imágenes en un entorno controlado y con la cámara de visión desacoplada del brazo robótico, estas pruebas no se llevan a cabo en el laboratorio.

La cámara se encuentra acoplada a un trípode estático, tal como muestra la siguiente figura:

Figura 54. Cámara acoplada en posición estática

Se capturan imágenes de diferentes formas y se realiza los patrones correspondientes. A parte se comprueba qué a diferentes posiciones de la misma forma el modelo hecho es capaz de reconocerlo y también que reconozca más de una pieza.

5.1.3. Procesamiento de imágenes para modelos con cámara acoplado al robot:

Se realizan varias pruebas de captura de imágenes en un entorno controlado y con la cámara de visión acoplada al brazo robótico, estas pruebas se llevan a cabo en el laboratorio.

La cámara es acoplada al brazo en la posición mostrada en la siguiente figura:

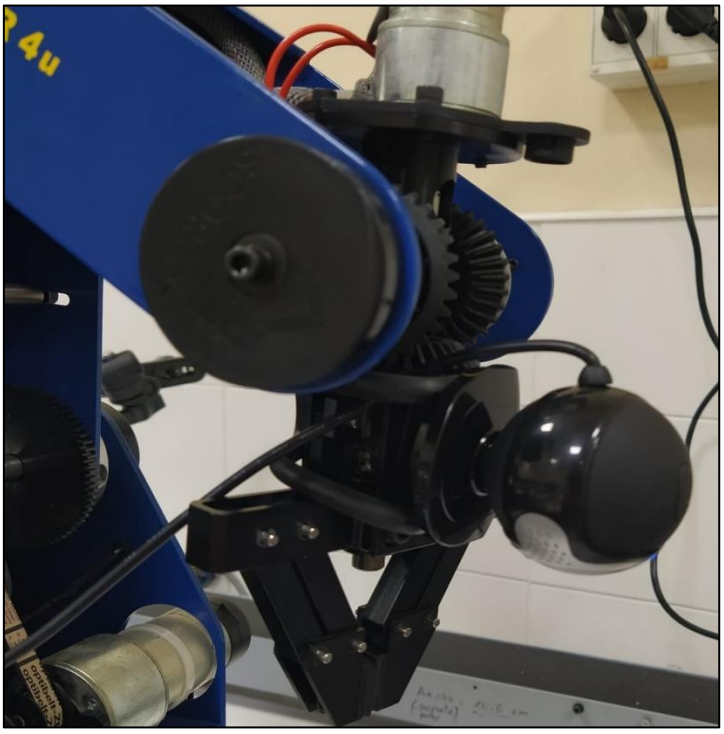


Figura 55. Posición de la cámara acoplada al brazo robótico

En esta prueba se capturan imágenes de diferentes formas y se realiza los patrones correspondientes. A parte se comprueba qué a diferentes posiciones de la misma forma el modelo hecho es capaz de reconocerlo y también que reconozca más de una pieza.

5.1.4. Calibración del brazo robótico con cámara desacoplada del robot:

Se lleva a cabo la calibración del brazo robótico con la cámara fuera del entorno de movimiento del robot.

5.1.5. Calibración del brazo robótico con cámara acoplada al robot:

Se lleva a cabo la calibración del brazo robótico con la cámara acoplada al brazo robótico, para ello, es necesario guardar una posición de captura de imágenes para que no afecte a la calibración.

5.1.6. Identificación de la forma de las piezas por patrones mediante SCORBASE:

Se realiza un programa en el software SCORBASE que sea capaz de identificar diferentes modelos de patrones en una posición exacta, y que el brazo robótico coja la pieza y la suelte en una posición ya predefinida dependiendo de su forma.

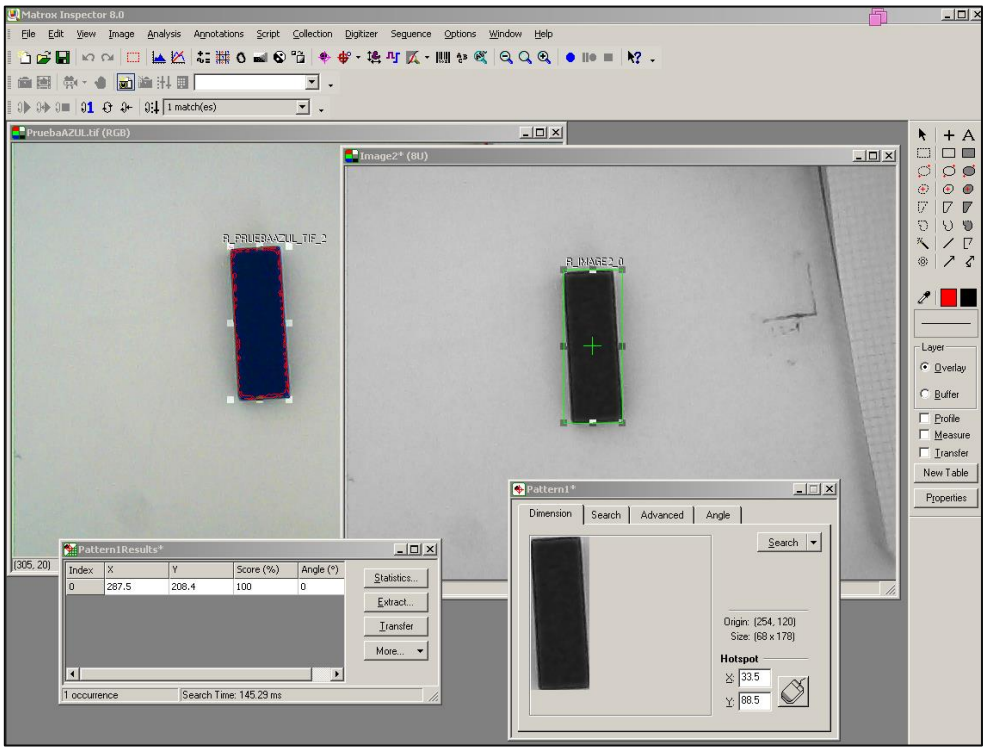


Figura 56. Creación de modelos con ViewFlex

5.1.7. Movimiento del robot a la posición de una pieza:

Se crea un programa en SCOBASE para que el robot vaya a la posición de la pieza. Este programa se hace con una única pieza en la visión de la cámara.

Primero se hace con la cámara fuera del entorno de trabajo y luego el mismo con la cámara acoplada al brazo robótico, para diferencias.

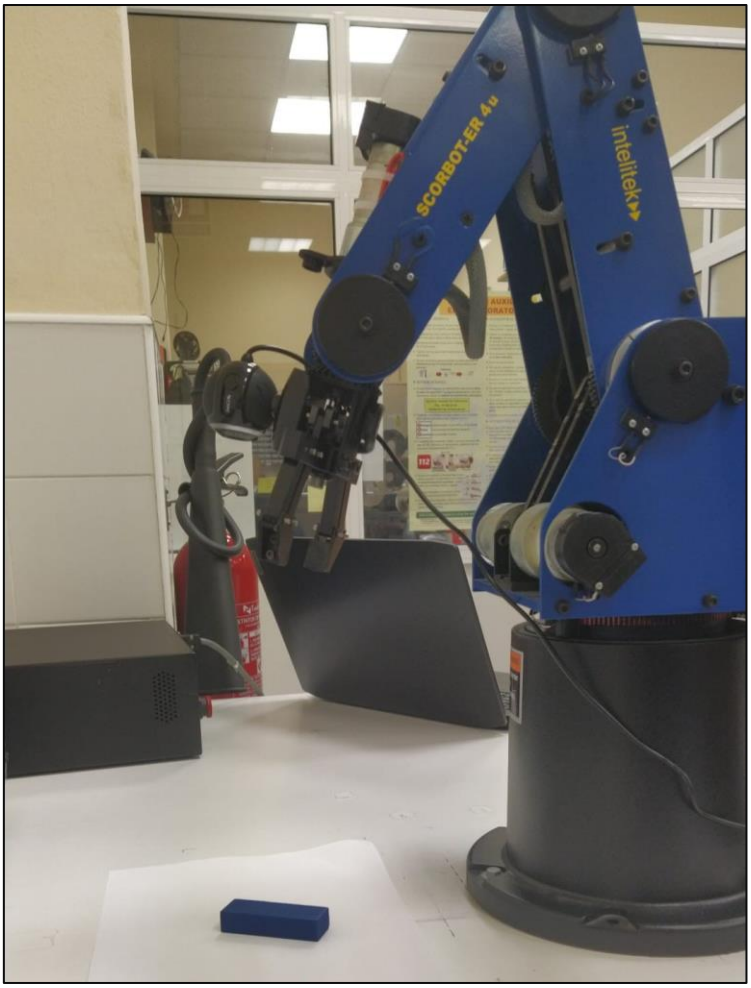


Figura 57. Robot para buscar una pieza

5.1.8. Movimiento del robot a la posición de varias piezas.

Se crea un programa en SCORBASE para que el brazo robótico vaya a la posición de la primera pieza detectada, luego va cogiendo el resto de las piezas en un orden específico.

Esta prueba se realiza directamente con la cámara acoplada al brazo del robot.

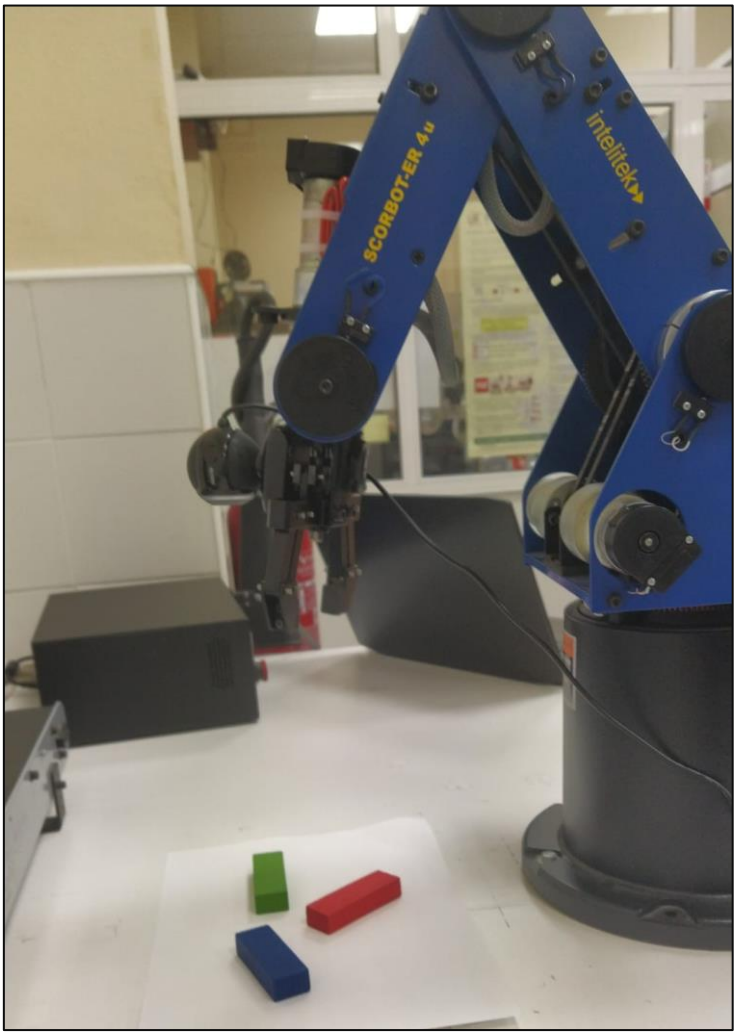


Figura 58. Robot para buscar varias piezas

5.1.9. Creación de nuevos modelos de formas de pieza:

Se realiza un programa donde se puedan crear nuevos patrones sin la necesidad de que estos estén previamente creados.

Esta prueba se realiza directamente con la cámara acoplada al brazo del robot.

5.1.10. Puesta en común de todos los apartados:

Se crea un nuevo programa que ponga en práctica todos los puntos anteriores, para ver un funcionamiento completo de la visión artificial en el brazo robótico.

Se lleva a prueba mediante la utilización de un puzle de formas. Primero debe crear los modelos de patrones de las piezas para posteriormente ir a buscarlas, por último, debe colocarlas en la posición correcta.

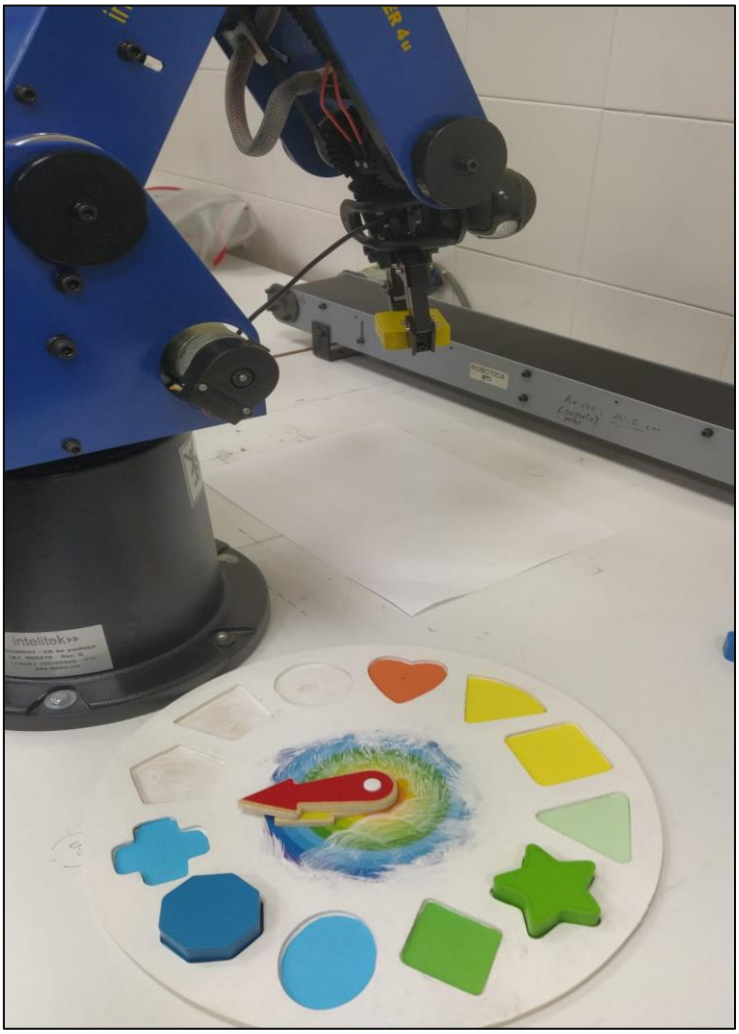


Figura 59. Robot montando el puzle

5.2. Planificación

El tiempo estimado para este estudio son de 300 horas donde inicialmente se creó una planificación con tiempos estimados para un más fácil desarrollo de este. Las tareas se dividen en las diferentes pruebas a realizar y en un tiempo de margen para la resolución de las diferentes dificultades que puedan aparecer en el desarrollo.

La planificación mencionada anteriormente es la que se muestra a continuación:

PLANIFICACIÓN INICIAL	
Tarea	Horas adjudicadas
Familiarización con el entorno	25
Procesamiento de imágenes	50
Identificación de diferentes formas	75
Prueba final	100
Tiempo de margen	50

Figura 60. Tabla de planificación inicial

Una vez se empieza con las diferentes pruebas se observa que no se puede seguir con la planificación inicial, y por tanto se debe retocar esta tanto en tiempos de realización, debido a las dificultades que se presentan en él, como en el de incluir nuevas pruebas que se han ido pensando y razonando a lo largo del estudio y que se desarrollarán en los apartados posteriores.

PLANIFICACIÓN FINAL	
Tarea	Horas adjudicadas
Familiarización con el entorno	25
Procesamiento de imágenes	50
Calibración de la cámara	15
Identificación de piezas	25
Ir a buscar las piezas en diferentes posiciones	50
Crear modelos de piezas desde cero	35
Prueba final	100

Figura 61. Tabla de planificación final

6. BANCO DE PRUEBAS Y DISCUSIÓN

En este apartado se van a mostrar todo el conjunto de pruebas, ejemplos y ensayos que han sido realizados en este estudio y se van a ir analizando resultados obtenidos. Estas pruebas se van a ir siguiendo el orden predefinido que ha sido diseñado en el apartado de metodología.

Para las pruebas que se deban realizar con la cámara acoplada al brazo robótico, esta se dispondrá de la siguiente manera:

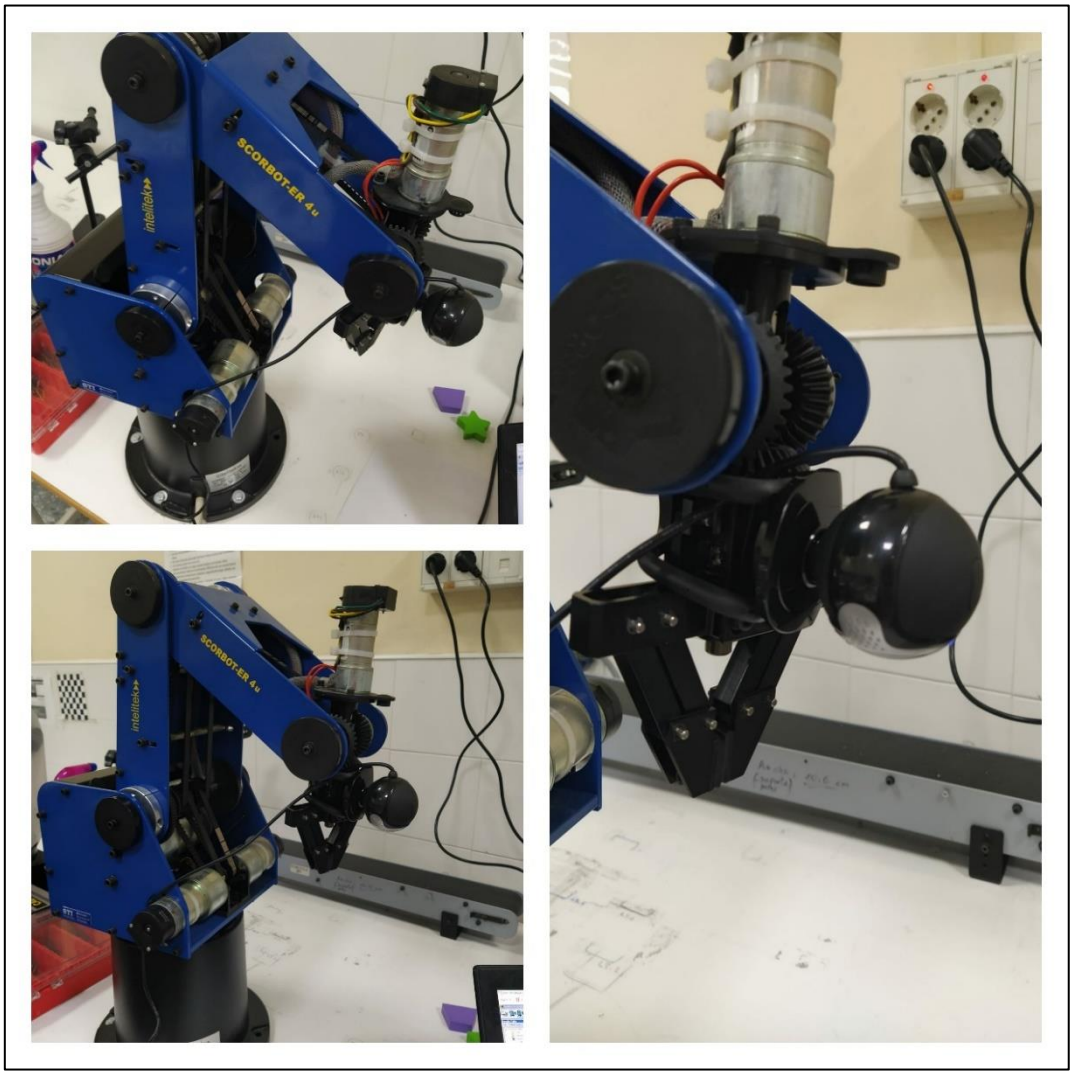


Figura 62. Posición de la cámara acoplada al brazo robot.

6.1. Procesamiento de imágenes para modelos con cámara desacoplada del robot

En este apartado se realiza una prueba para la creación de un modelo sobre una pieza de forma rectangular y de color azul. También se muestra que al cambiar dicha pieza de posición el modelo creado es capaz de seguir detectándola.

El entorno donde se realiza la prueba no está controlado, ya que depende de la luz que incida sobre la zona de visión y el fondo utilizado en esta prueba es de color blanco.

Para la creación del modelo solo es necesario utilizar el software ViewFlex y sus funciones de cámara, Matrox Inspector y Tabla de Resultados, esta última es necesaria para comprobar que se encuentran las ocurrencias una vez creado el modelo.

El primer paso necesario en este proceso es colocar la cámara en una posición estática y que tenga una buena visibilidad de la zona donde se van a colocar las piezas para posteriormente realizar la captura de la imagen.

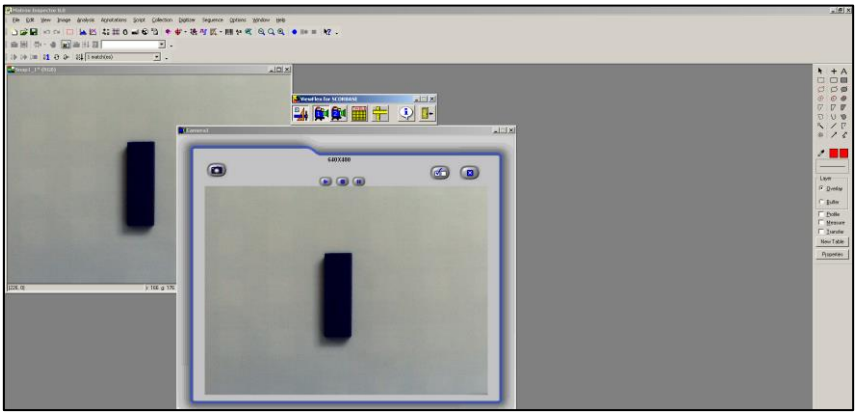


Figura 63. Captura de imagen para crear un modelo

Una vez esta haya sido capturada se pasa a crear el modelo, para ello se necesita enmarcar la zona interesada. Esta zona se marca con el uso de las herramientas que aparecen a la derecha en la interfaz de Matrox Inspector.

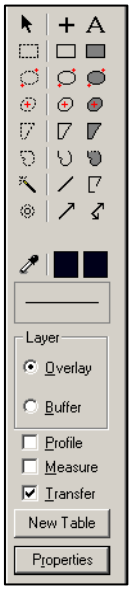


Figura 64. Herramientas para crear la zona del modelo

En este estudio las herramientas que se utilizan son únicamente la de la primera columna. Por tanto, se selecciona la herramienta de rectángulo y se pasa a encasillar la zona deseada, esto es llamado crear un ROI.

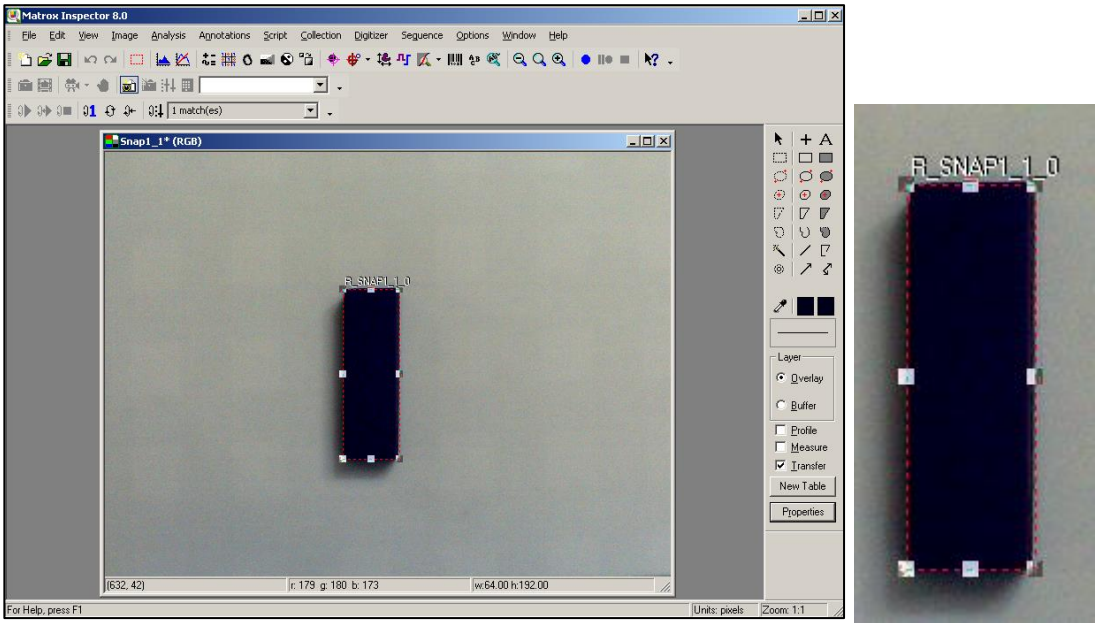


Figura 65. Zona a la que se le desea hacer el modelo

Lo siguiente es presionar la opción de "*Pattern Matching*" que se puede localizar tanto en *Analysis>>Pattern>>New Model* o a través de la misma opción que aparece en la barra de herramientas superior.

Al presionar esta opción se abre la misma imagen, pero en escala de grises debido a que los modelos solo pueden ser creados mediante estos colores. Por tanto, solo se pueden definir modelos para la captación de formas y no para diferenciar colores.

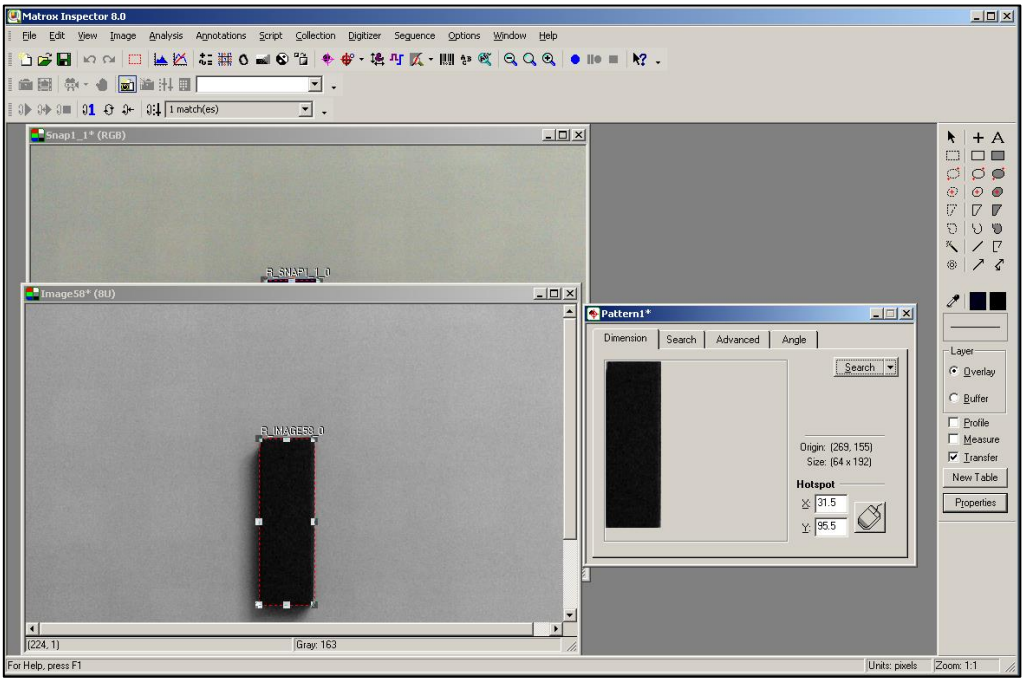


Figura 66. Creación del modelo de la pieza

Se abre una interfaz para poder configurar las diferentes características disponibles en la creación del modelo.

En la opción "*Search*" se debe de elegir la opción que permita buscar todas las ocurrencias, y si fuera necesario cambiar el valor en % de la aceptación para la búsqueda, esta inicialmente siempre es del 70%.

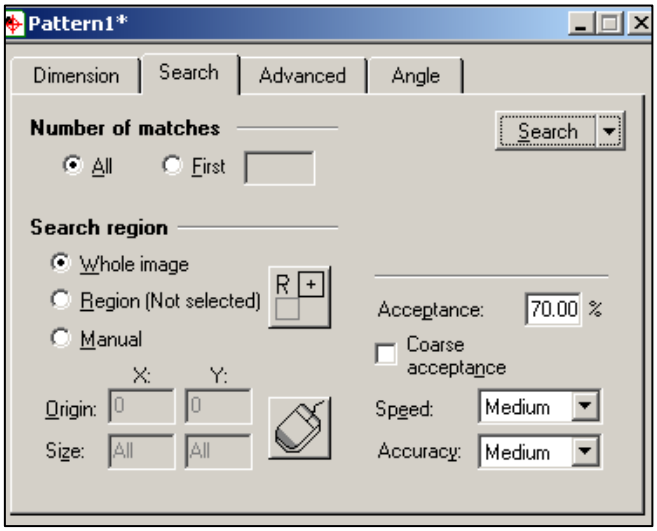


Figura 67. Modelo – *Search*

Luego la otra opción que se configura en la de "*Angle*", donde se debe activar la opción de rotación y configurar para que busque las piezas en los 360º.

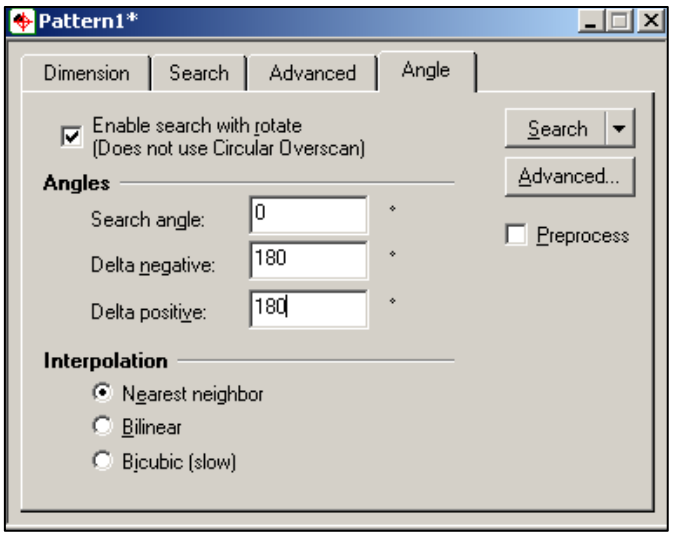


Figura 68. Modelo – *Angle*

Una vez configurado estas características se pasa a guardar el modelo bajo el nombre de "Prueba1" y se realiza una prueba de detección de piezas en diferentes posiciones.

Para poder comprobar si se han producido ocurrencias con este modelo, se abre la interfaz de la función de Tabla de Resultados y se elige el *Pattern* guardado.

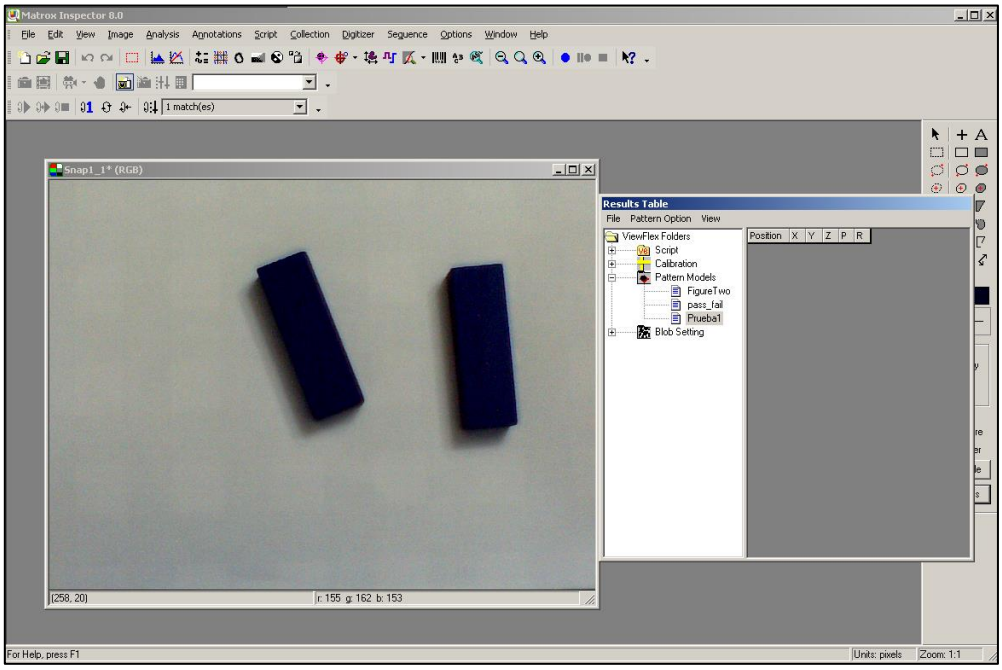


Figura 69. Ocurrencias del modelo “Prueba1”

En la figura anterior se puede observar que este modelo no ha sido capaz de reconocer ninguna de las dos piezas, por lo que se crea un nuevo modelo llamado “Prueba2” en el que el ROI no se hace en el borde de la pieza.

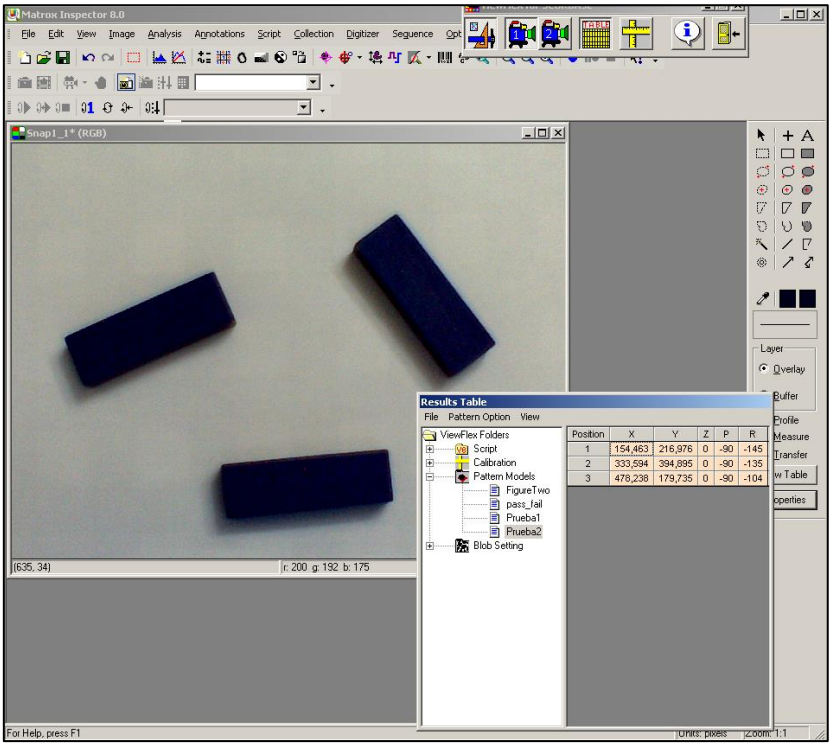


Figura 70. Ocurrencias del modelo “Prueba2”

Ahora se muestran todas las ocurrencias, esto se debe a que en el último modelo se tiene en cuenta las sombras que se producen por la incidencia de la luz, mientras que en el modelo “Prueba1” no se tienen en cuenta y, por tanto, estas no son detectadas correctamente impidiendo encontrar las ocurrencias correctamente.

En ambos casos los modelos se han realizado con la pieza totalmente en vertical, pero esto no quiere decir que no se puedan crear nuevos modelos con las piezas en posición oblicua o en horizontal. Solo es necesario no crear un ROI al borde de la pieza, sino que hay que estar conscientes de las sombras que se producen.

Por otro lado, la creación del ROI en este apartado del estudio se hace mediante una herramienta de recorte rectangular. En cambio, a lo largo del estudio la realización de los ROIs se hace mediante la herramienta “*Magic Wand*” que se encarga de detectar de manera automática los bordes de la imagen.

Es conveniente que al utilizar esta herramienta la pieza se encuentre en una posición oblicua, ya que el ROI creado solo detecta los bordes de la pieza y al crear el modelo se pasa a una forma rectangular, por tanto, si se encuentra en vertical u horizontal no se tienen en cuentas las sombras producidas y pasa lo mismo que en el modelo “Prueba1”.

6.2. Procesamiento de imágenes para modelos con cámara acoplada al robot

En este apartado se realiza una prueba para la creación de varios modelos para piezas de forma rectangular y circular de varios colores diferentes. También se muestra que al cambiar dichas piezas de posición y lugar los modelos creados son capaces de seguir detectándolas.

El entorno donde se realizan las pruebas está controlado y se hacen con diferentes fondos de colores y texturas para ver cómo afectan a la creación de los modelos y de la detección de estas piezas.

6.2.1. Fondo de papel blanco

La primera prueba que se realiza con este tipo de fondo es la de una pieza de forma rectangular en color azul. A continuación, se muestra la creación del modelo:

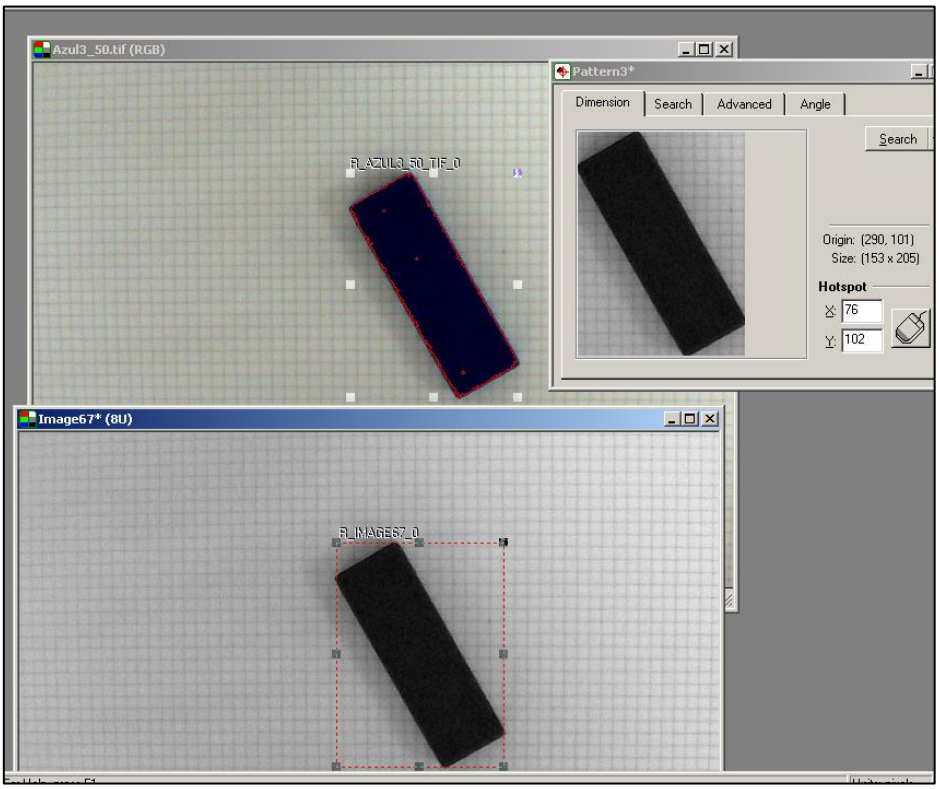


Figura 71. Papel blanco – Modelo de pieza rectangular y azul

Este nuevo modelo es guardado bajo el nombre “Azul_2”, y busca todas las ocurrencias que se produzcan con una aceptación del 70% en todas las posiciones y ángulos posibles.

A continuación, se hacen varias pruebas con piezas de la misma forma que la usada para la creación del modelo “Azul_2” pero con diferentes colores.

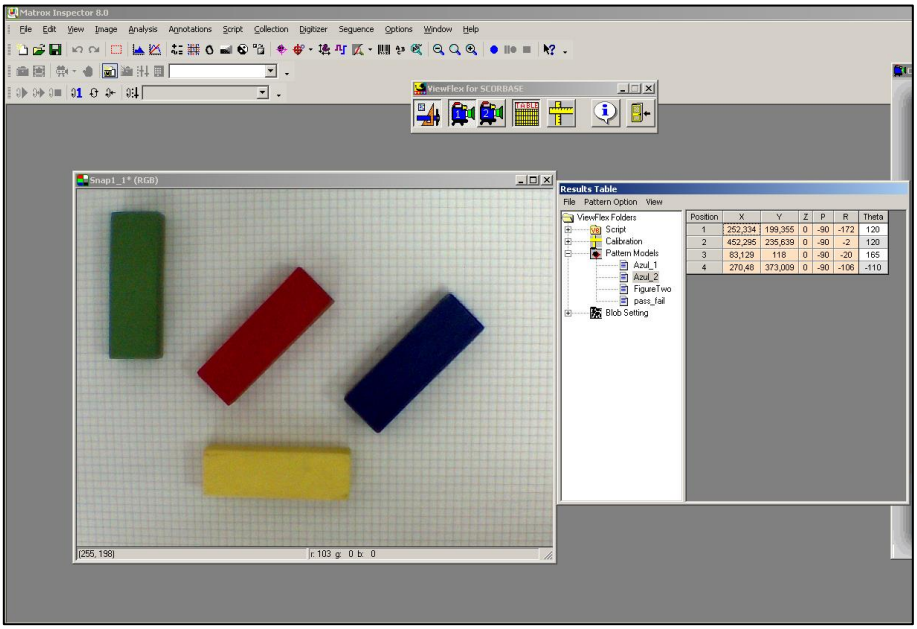


Figura 72. Papel blanco - Ocurrencias del modelo “Azul_2”

Se comprueba que la mayoría de estas piezas son detectadas, pero ocurren algunas incidencias con piezas de colores más claros. Las piezas de color rosa no siempre son detectadas y las que tienen un color madera claro no son detectadas en ninguna ocasión.

Por ello, se procede a crear un nuevo modelo con la misma captura de imagen utilizada en el modelo “Azul_2” pero con un porcentaje de aceptación más bajo (50%), este nuevo modelo pasa a ser llamado “Azul_3”.

En este modelo se comprueba que es capaz de detectar los diferentes tipos de colores sin ningún problema aparente, exceptuando la pieza de color madera claro que sigue sin poder reconocerse.

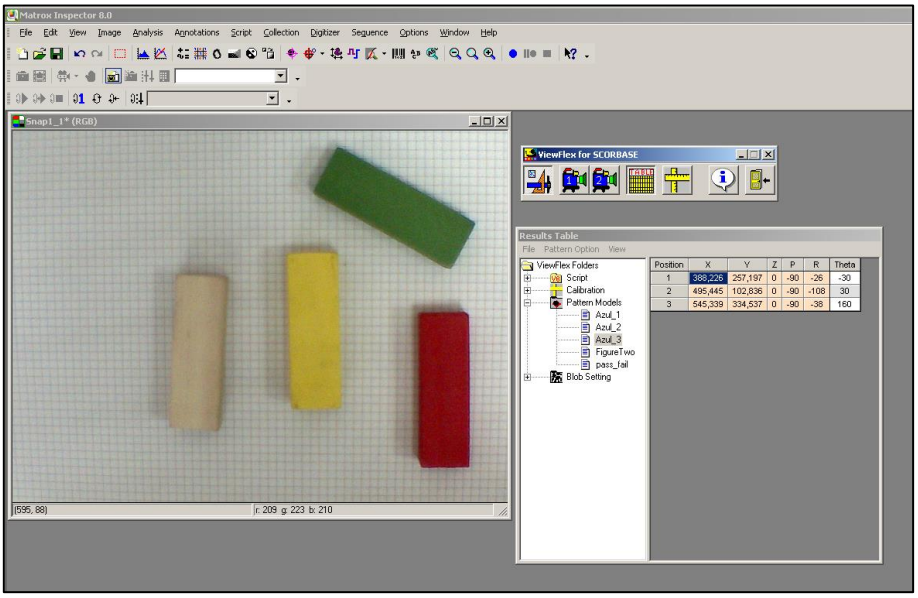


Figura 73. Papel blanco - Ocurrencias del modelo “Azul_3”

Se puede llegar a pensar en bajar más el porcentaje de aceptación, pero no es recomendable ya que pueden aparecer ocurrencias no deseadas y dar errores referentes a las posiciones que se muestran en la tabla de resultados.

Por tanto, se procede a crear un modelo exclusivamente para la pieza rectangular en color madera claro.

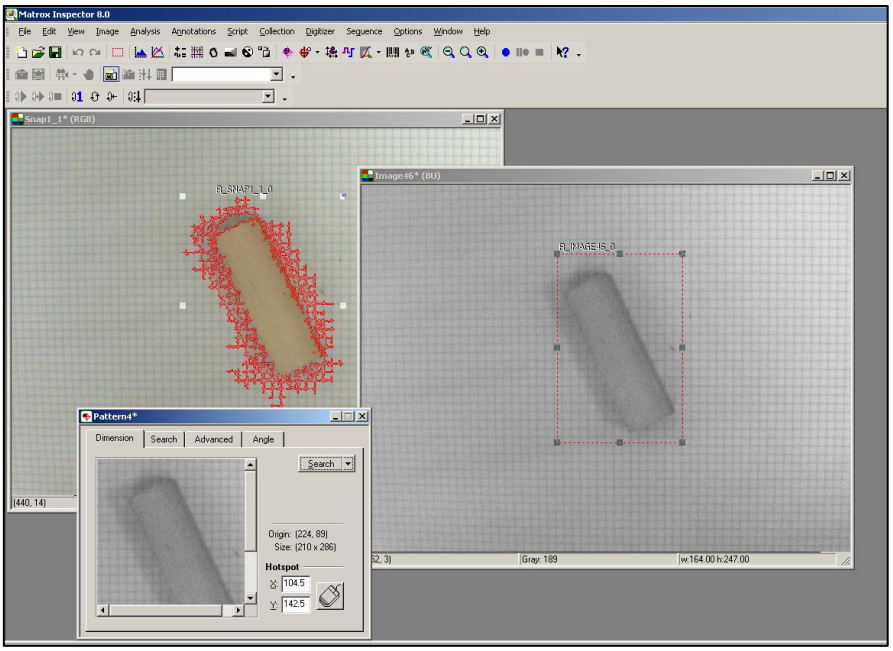


Figura 74. Papel blanco - Creación del modelo con pieza rectangular en madera clara

Como se observa en la figura anterior al pasar la imagen a una escala de grises este tipo de color de pieza se difumina con el fondo banco, por tanto, no se puede crear el modelo de la pieza con este tipo de fondo.

Una vez terminado de realizar las pruebas y análisis para las piezas de forma rectangular se pasan a las de forma circular.

Primero se hace una captura de una pieza circular de color negra y a partir de este se crea un modelo, llamado “DamaNegra_1”, que capta todas las ocurrencias con una aceptación del 70%. Se tiene en cuenta que al ser piezas de forma circular no hace falta configurar el apartado de ángulo, ya que ese siempre se mantiene constante.

Luego se pasa a hacer una captura con varias piezas de la misma forma y color y se comprueba que son detectadas sin ningún tipo de fallo.

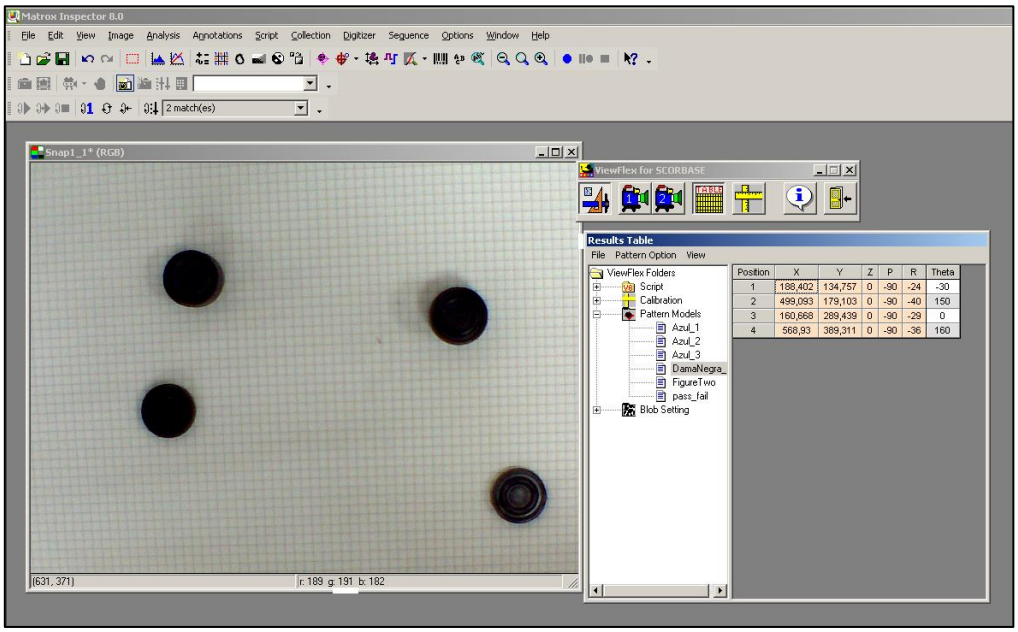


Figura 75. Papel blanco - Ocurrencias del modelo “DamaNegra_1”

En cambio, cuando se introduce una pieza de la misma forma, pero de color blanco este modelo no es capaz de detectarlo.

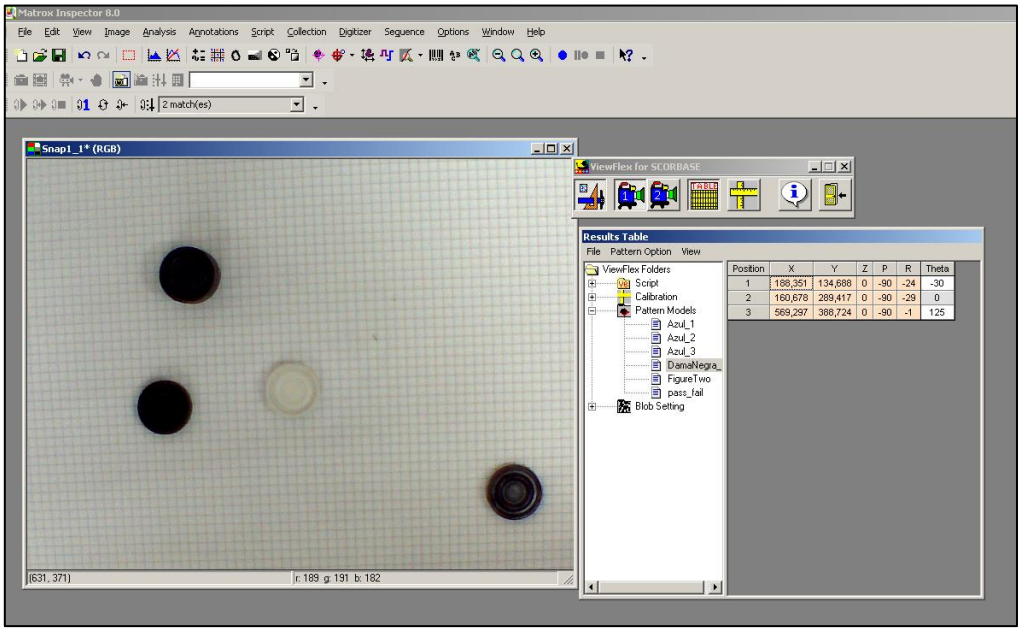


Figura 76. Papel blanco - Ocurrencias del modelo “DamaNegra_1” con piezas de diferentes colores

Esto se sigue produciendo también con modelos que tengan una aceptación del 50%, y como se ha comentado con anterioridad no es conveniente bajarlo más. Por tanto, a este tipo de piezas blancas le ocurre lo mismo que a las piezas de madera clara y es que se difuminan con este tipo de fondo.

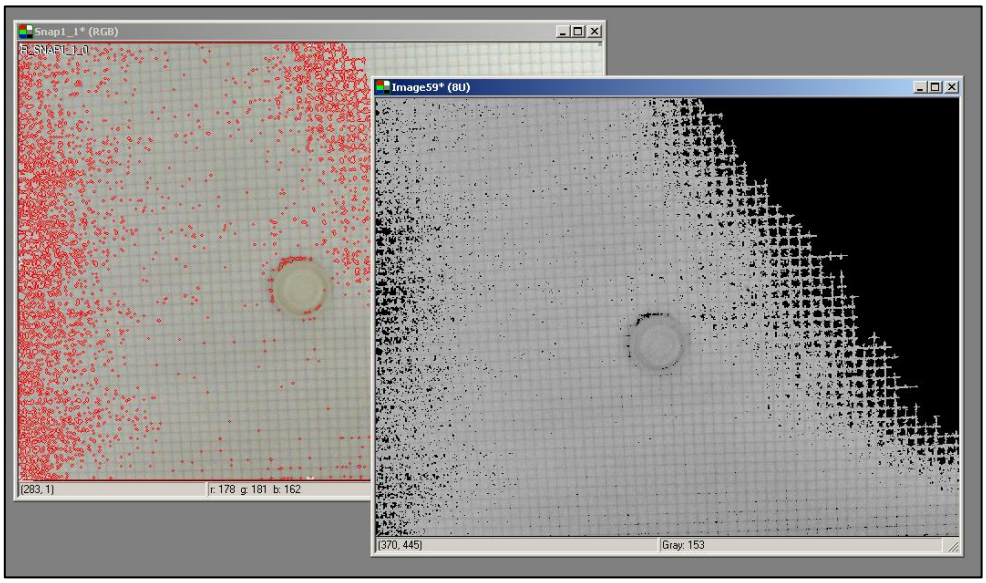


Figura 77. Papel blanco - Creación de modelo con pieza circular blanca

Por tanto, los resultados obtenidos utilizando este tipo fondo son los siguientes:

Tipo de fondo	Tipo de pieza	Conclusión
Papel blanco	Rectangular azul	Se reconocen fácilmente, no producen ningún problema.
	Rectangular de colores oscuros	Se reconocen fácilmente, no producen ningún tipo de problema.
	Rectangular de colores claros	Se reconocen con una aceptación del 60-50%.
	Rectangular madera clara	Imposible de reconocer, se difuman con el fondo.
	Circular negra	Se reconocen fácilmente, no producen ningún problema.
	Circular blanca	Imposible de reconocer, se difuminan con el fondo.

Figura 78. Tabla resumen del fondo de papel blanco

6.2.2. Fondo de cartulina negra

La primera prueba que se realiza con este tipo de fondo es con la pieza de forma rectangular en color madera clara, para comprobar que con este tipo de fondo si se le puede hacer un modelo.

Inicialmente se crea un modelo llamado “Madera_1” que busca todas las ocurrencias que se produzcan con una aceptación del 70% en todas las posiciones y ángulos posibles.

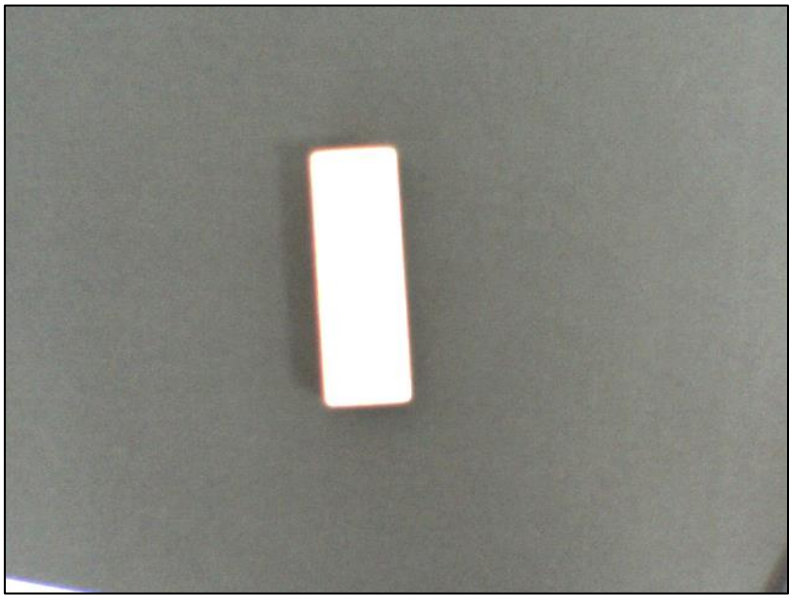


Figura 79. Cartulina negra - Modelo para figura rectangular de madera clara

Se comprueba que al buscar ocurrencias en la Taba de Resultados este modelo es capaz de encontrar todas las piezas de la misma forma y diferente color exceptuando la pieza de color azul, pasa el caso contrario a la prueba con fondo de papel blanco.

Pero en cambio, si al mismo modelo, llamado “Madera_2”, se le pone una aceptación del 50% si es capaz de reconocer las piezas de color azul.

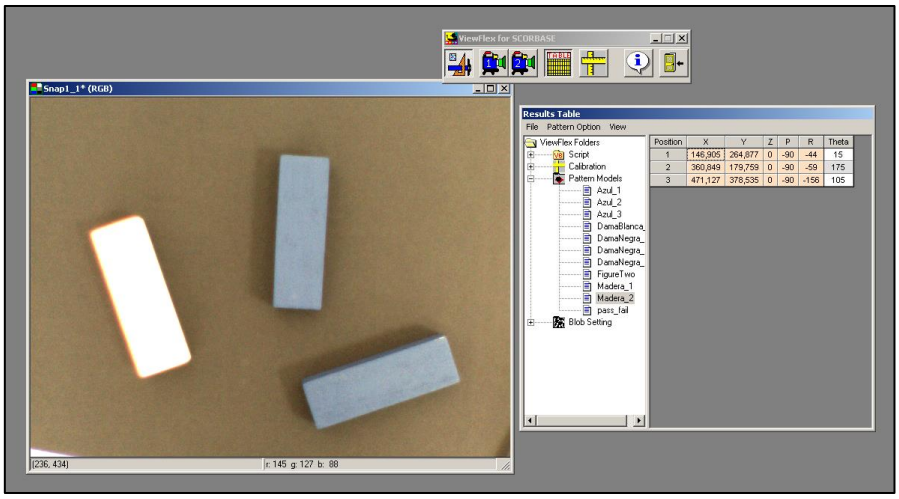


Figura 80. Cartulina negra – Ocurrencias del modelo “Madera_2”

Por tanto, con ese tipo de fondo y un único modelo de la figura rectangular en color madera clara se pueden detectar todas posiciones y colores de este tipo de piezas.

En cambio, para las piezas circulares se debe de hacer dos modelos, uno para cada color, ya que incluso con una aceptación del 50% no son capaces de reconocerse.

En primer lugar, se hace una captura de una pieza circular de color blanca y a partir de este se crea un modelo, llamado “DamaBlanca_1”, que capta todas las ocurrencias con una aceptación del 70%.

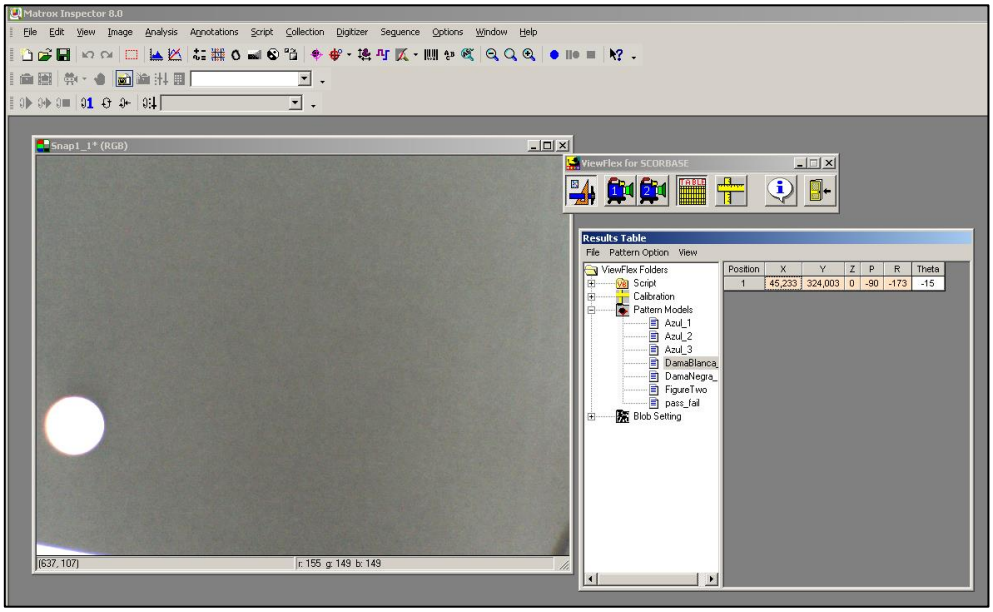


Figura 81. Cartulina negra – Ocurrencias del modelo “DamaBlanca_1”

Luego se procede a la creación del modelo para la pieza circular negra, llamada “DamaNegra_2”, que tiene las mismas condiciones que las de “DamaBlanca_1”. Este modelo es incapaz de reconocer a las demás piezas circulares negras, por tanto, se crea un nuevo modelo nombrado por “DamaNegra_3” con una aceptación del 50% que sí es capaz de reconocer todas las piezas.

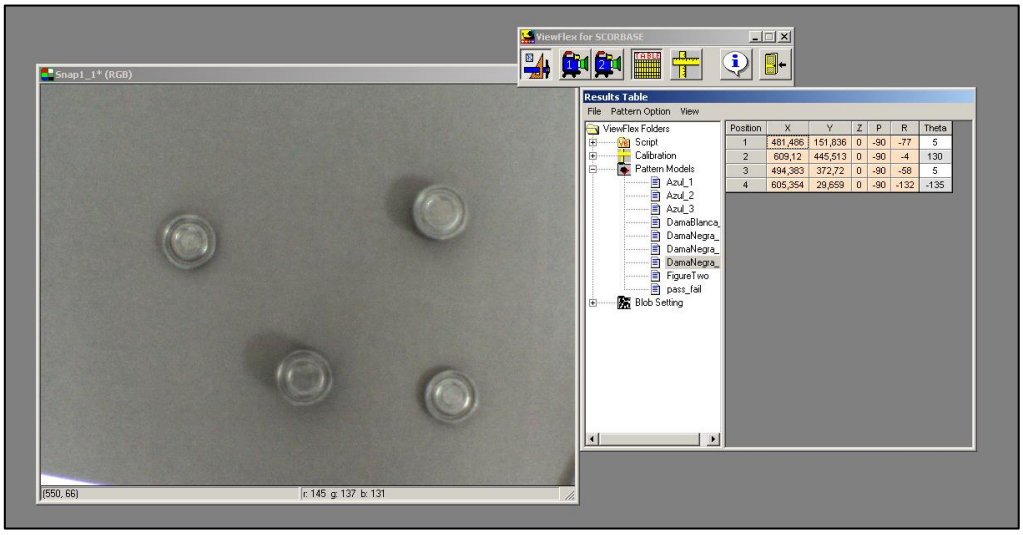


Figura 82. Cartulina negra – Ocurrencias del modelo “DamaNegra_3”

Por tanto, los resultados obtenidos utilizando este tipo fondo son los siguientes:

Tipo de fondo	Tipo de pieza	Conclusión
Cartulina negra	Rectangular azul	Se reconocen con una aceptación del 50%.
	Rectangular de colores oscuros	Se reconocen fácilmente, no producen ningún problema.
	Rectangular de colores claros	Se reconocen fácilmente, no producen ningún problema.
	Rectangular madera clara	Se reconocen fácilmente, no producen ningún problema.
	Circular negra	Se reconocen con una aceptación del 50%.
	Circular blanca	Se reconocen fácilmente, no producen ningún problema.

Figura 83. Tabla resumen del fondo de cartulina negra

6.2.3. Madera blanca

La primera prueba que se realiza con este tipo de fondo es con una pieza de forma rectangular en color azul y se crea un modelo llamado “AzulM_1” con una aceptación del 70%.

También se crea el modelo para que detecte todas las ocurrencias que sea capaz de identificar y en todos los ángulos posibles.

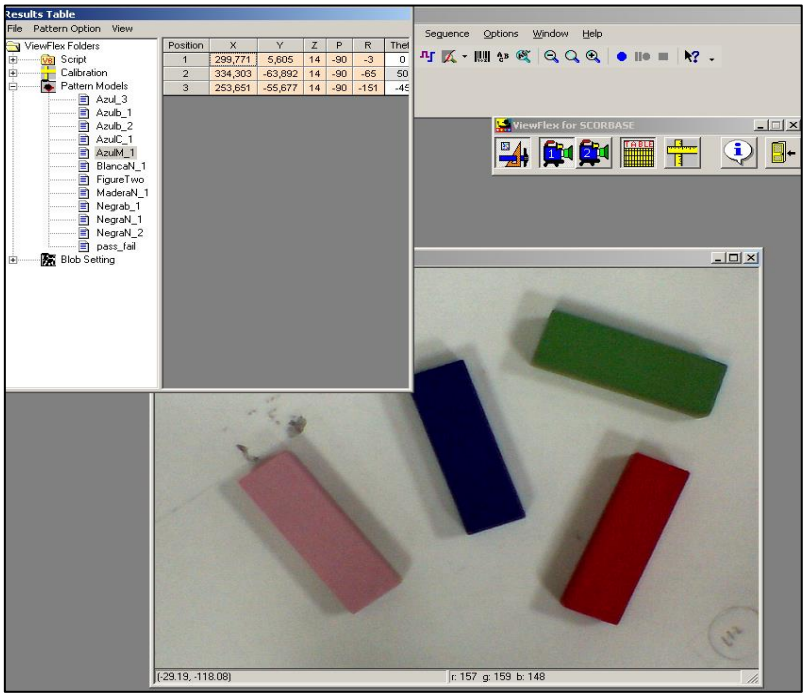


Figura 84. Madera blanca – Ocurrencias del modelo “AzulM_1”

Se observa que las piezas de color más claro no son capaces de encontrarlas, en este caso es la pieza rosa. Por tanto, se procede a crear un nuevo modelo llamado “AzulM_2” con una aceptación del 50%.

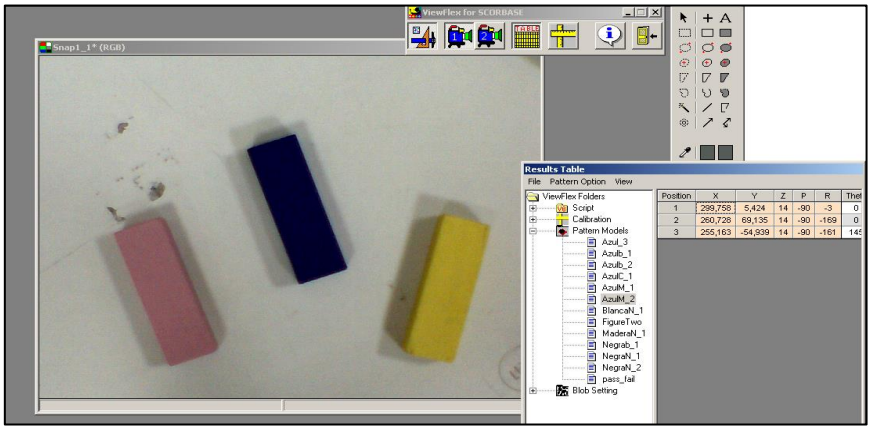


Figura 85. Madera blanca – Ocurrencias del modelo “AzulM_2”

Con este modelo si encuentra todas las piezas de colores claros, aunque la pieza de color madera clara es la única que no reconoce, pasa igual que en la prueba de fondo con papel blanco.

Por tanto, se pasa a crear un modelo para la pieza de color madera claro, pero igual que pasa con los fondos más claros no es posible realizar el modelo ya que se difumina con el fondo.

A continuación, se muestra una imagen de cómo se difumina con el fondo:

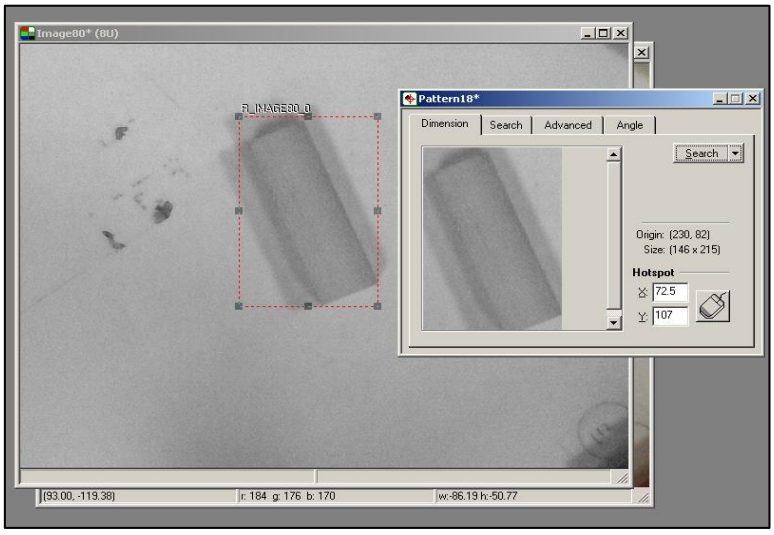


Figura 86. Madera blanca – Creación del modelo con pieza rectangular en madera clara

Ahora se pasa a las piezas circulares negras, se crea un modelo llamado “NegraM_1” con una aceptación del 70% que es capaz de reconocer todas las piezas negras, pero no la de color blanco.

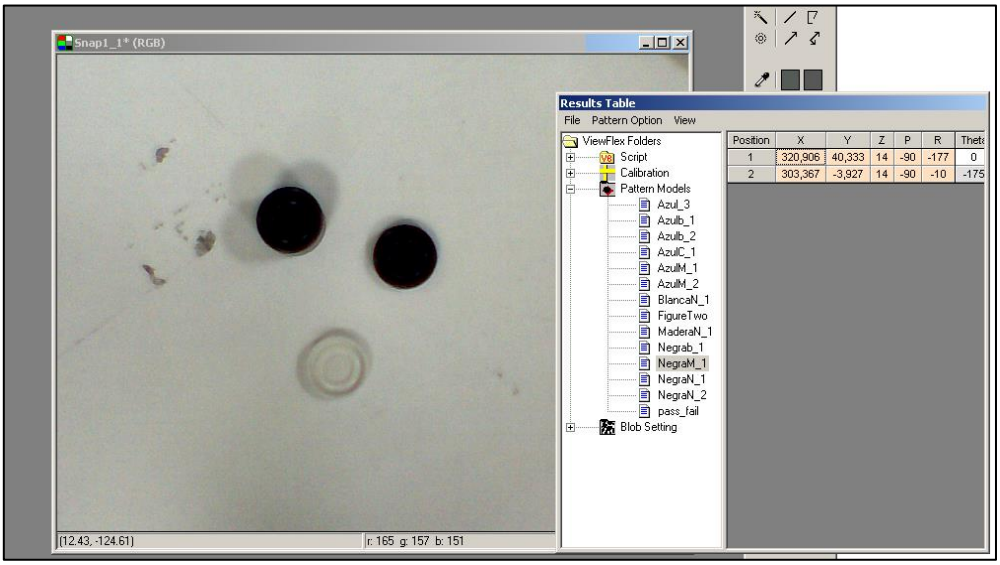


Figura 87. Madera blanca – Ocurrencias del modela “NegraM_1”

Por tanto, se pasa a crear un modelo que detecte las piezas circulares blancas, pero como llevan pasando a lo largo del estudio se sigue comprobando que las piezas claras en fondos blancos se difuminan con este.

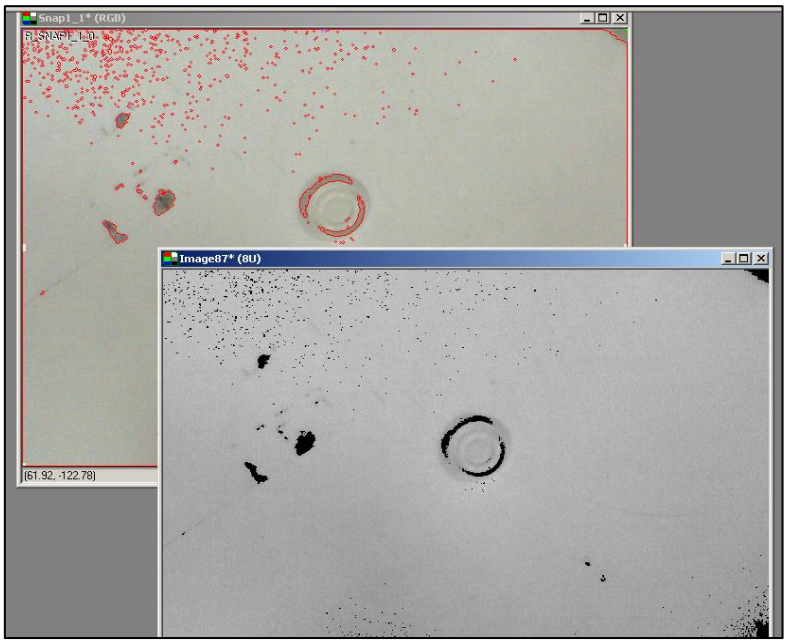


Figura 88. Madera blanca - Creación de modelo con pieza circular blanca

Por tanto, los resultados obtenidos utilizando este tipo fondo son los siguientes:

Tipo de fondo	Tipo de pieza	Conclusión
Madera blanca	Rectangular azul	Se reconocen con una aceptación del 50%.
	Rectangular de colores oscuros	Se reconocen fácilmente, no producen ningún problema.
	Rectangular de colores claros	Se reconocen fácilmente, no producen ningún problema.
	Rectangular madera clara	Se reconocen fácilmente, no producen ningún problema.
	Circular negra	Se reconocen con una aceptación del 50%.
	Circular blanca	Imposible de reconocer, se difuminan con el fondo.

Figura 89. Tabla resumen del fondo de madera blanca

6.2.4. Cartón

Se inician las pruebas con este tipo de fondo creando un modelo de la pieza rectangular azul con una aceptación del 70% y que le sea posible encontrar todas las ocurrencias en todas sus posiciones, este modelo es llamado “AzulC_1”.

Al buscar ocurrencias con otras piezas de la misma forma, pero de diferente color se puede observar que no es capaz de detectarlas.

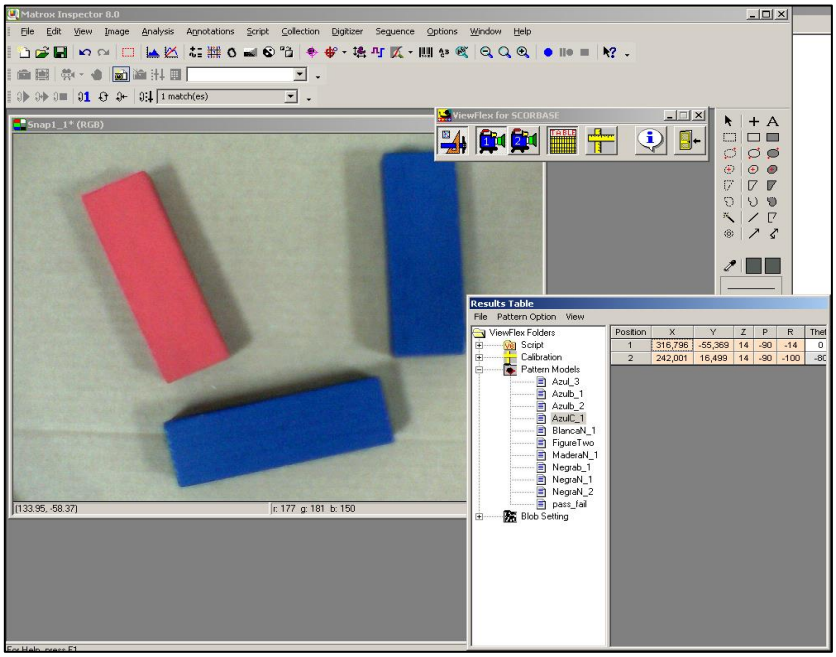


Figura 90. Cartón – Ocurrencia con el modelo “AzulC_1”

Por tanto, se procede a crear un nuevo modelo, pero variando el porcentaje de aceptación al 50%. Pero en la siguiente imagen se puede observar que aún bajando el porcentaje la pieza es detectada de formar incorrecta, por lo que se deduce que este tipo de fondo es apto si se quieren realizar ensayos de piezas de colores, más que de formas.

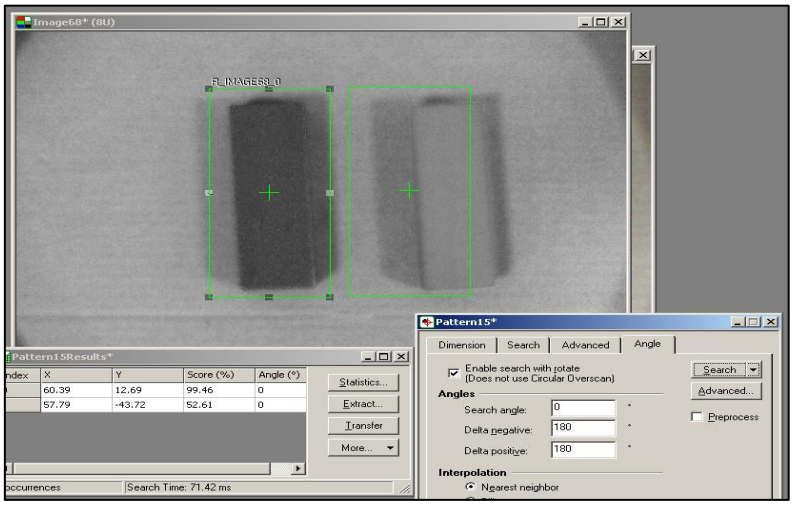


Figura 91. Cartón – Ocurrencias con el modelo de pieza azul al 50%

Si se quisiera realizar el mencionado estudio de las piezas por colores, es necesario poner el porcentaje de aceptación mayor o igual al que viene por defecto que es del

70%. Esto es necesario para que no se produzca el error de sombras que se produce en la figura 83, también se recomienda que en las piezas de color más claro o que se asemejen más al color del fondo se utilice un porcentaje de aceptación mayor.

Por tanto, los resultados obtenidos utilizando este tipo fondo son los siguientes:

Tipo de fondo	Tipo de pieza	Conclusión
Cartón	Rectangular azul	Se reconocen con una aceptación del 50%.
	Rectangular de colores oscuros	Se reconocen con una aceptación del 50%.
	Rectangular de colores claros	Se reconocen con una aceptación del 50%.
	Rectangular madera clara	Se reconocen con una aceptación del 50%.

Figura 92. Tabla resumen del fondo de cartón

6.2.5. Algodón blanco

Inicialmente se crea un modelo, llamado “Azulb_1”, de la pieza rectangular de color azul y con una aceptación del 70%.

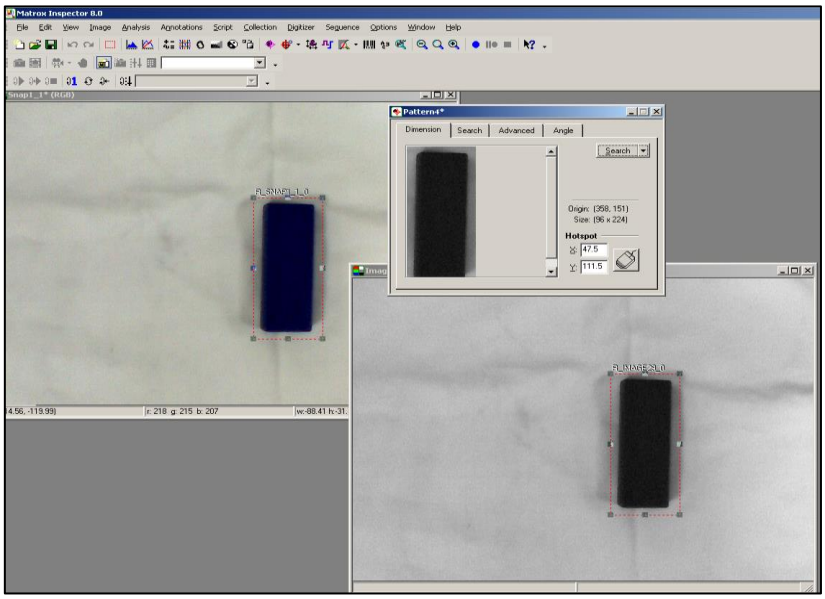


Figura 93. Algodón blanco – Creación del modelo “Azulb_1”

Se puede comprobar que detecta casi todas las piezas, y las que son de color un poco más claro dependiendo de la zona donde sean colocadas.

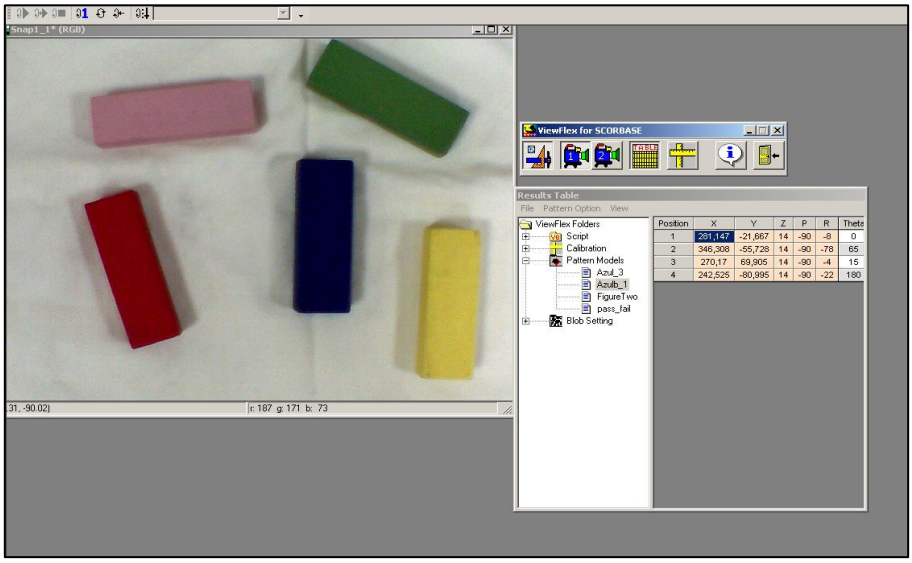


Figura 94. Algodón blanco – Ocurrencias del modelo “Azulb_1”

Lo siguiente es hacer como en los casos anteriores, crear un modelo de la misma pieza azul, pero con un porcentaje de aceptación menor.

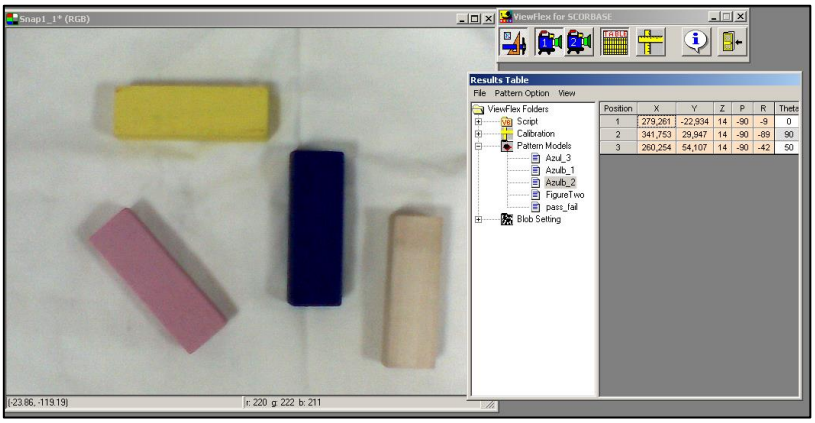


Figura 95. Algodón blanco – Ocurrencias con el modelo “Azulb_2”

Donde se vuelve a comprobar que la pieza que no es capaz de detectar sigue siendo la de color madera clara y tampoco se puede crear el modelo porque se difumina con el fondo.

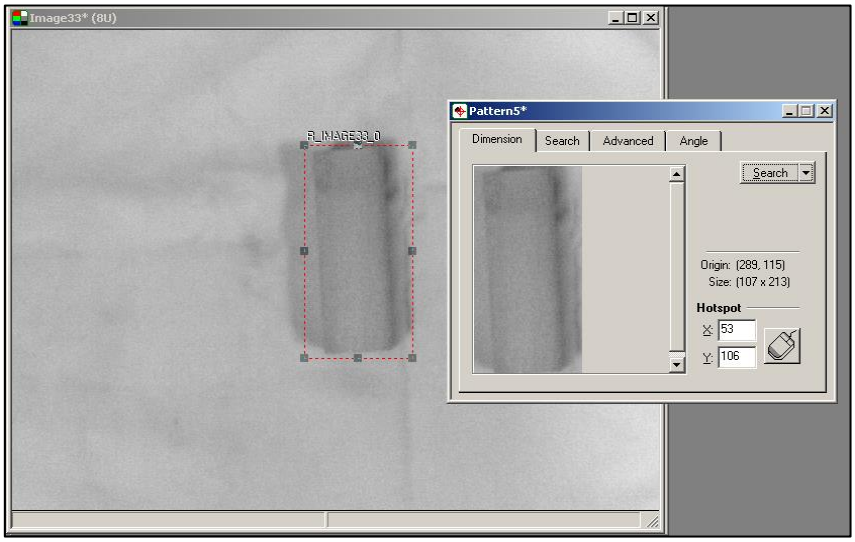


Figura 96. Algodón blanco – Creación del modelo para la pieza madera clara

A continuación, se pasa a la creación del modelo para las piezas circulares, que pasa igual que con los otros fondos de colores claros. Las piezas negras se realizan con una aceptación del 70%, modelo llamado “Negrab_1” se detectan sin ningún tipo de problema, pero con las blancas no ocurre lo mismo.

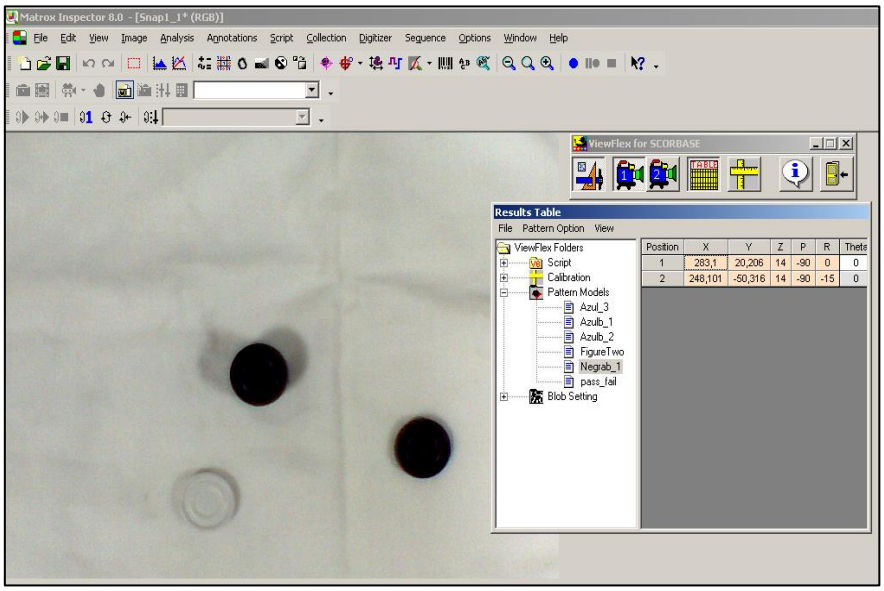


Figura 97. Algodón blanco – Ocurrencias del modelo “Negrab_1”

Si se intenta realizar el modelo de la pieza circular blanca ocurre lo mismo que en los otros casos, al convertir la imagen a una escala de grises los colores de las piezas y el fondo se difuminan impidiendo realizar el modelo.

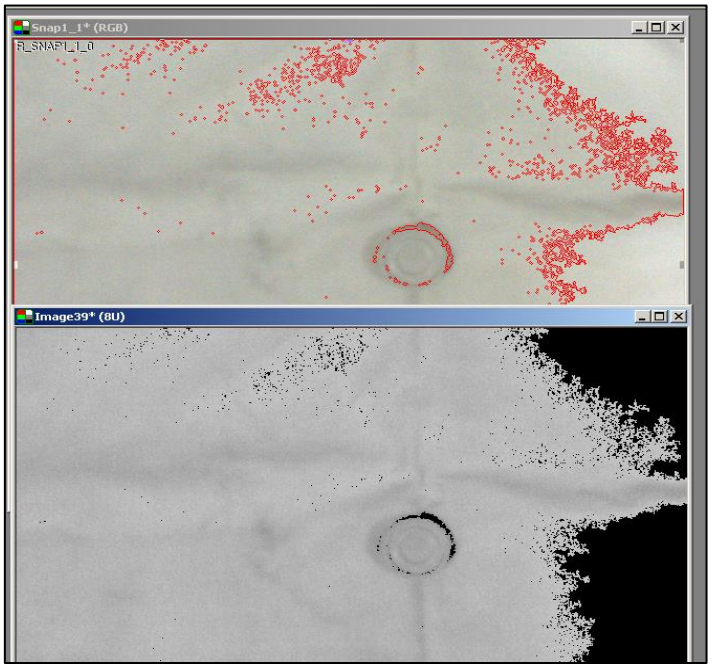


Figura 98. Algodón blanco – Creación del modelo para la pieza circular blanca

Por tanto, los resultados obtenidos utilizando este tipo fondo son los siguientes:

Tipo de fondo	Tipo de pieza	Conclusión
Algodón blanco	Rectangular azul	Se reconocen fácilmente, no producen ningún problema.
	Rectangular de colores oscuros	Se reconocen fácilmente, no producen ningún problema.
	Rectangular de colores claros	Se reconocen dependiendo de la zona donde son colocadas, peor identificación contri más al borde se coloquen.
	Rectangular madera clara	Imposible de reconocer, se difuminan con el fondo.
	Circular negra	Se reconocen fácilmente, no producen ningún problema.
	Circular blanca	Imposible de reconocer, se difuminan con el fondo.

Figura 99. Tabla resumen del fondo de algodón blanco

6.2.6. Algodón negro

Primero se crea un modelo de la pieza de color madera clara con un porcentaje de aceptación del 70%, llamado “MaderaN_1”. Con este modelo y con el grado de aceptación inicial se reconoces todas las piezas rectangulares de diferentes colores.

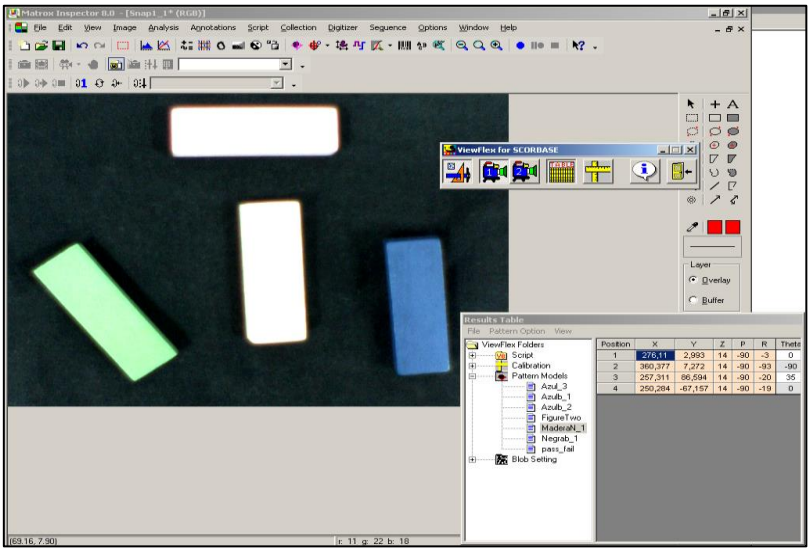


Figura 100. Algodón negro – Ocurrencias con el modelo “MaderaN_1”

Para este fondo hay que esperar unos cuantos de segundo hasta que la lente pueda adaptarse a los cambios de luz que se producen.

Para las piezas circulares blancas se crea un modelo llamado “BlancaN_1” que inicialmente tiene una aceptación del 70%, aunque se vuelve a cambiar este modelo para ver si es capaz de detectar las piezas circulares negras.

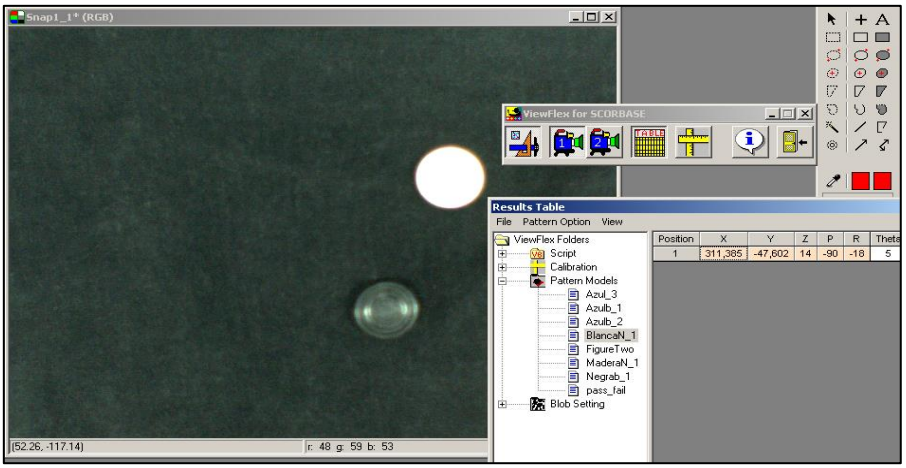


Figura 101. Algodón negro - Ocurrencia del modelo “BlancaN_1”

Como se puede observar aún cambiando el porcentaje de aceptación este modelo no es capaz de captar las piezas negras.

Para reconocer las piezas circulares negras se debe crear un modelo, llamado "NegraN_2", con un porcentaje de aceptación del 50%, ya que al 70% no es capaz de detectar más piezas.

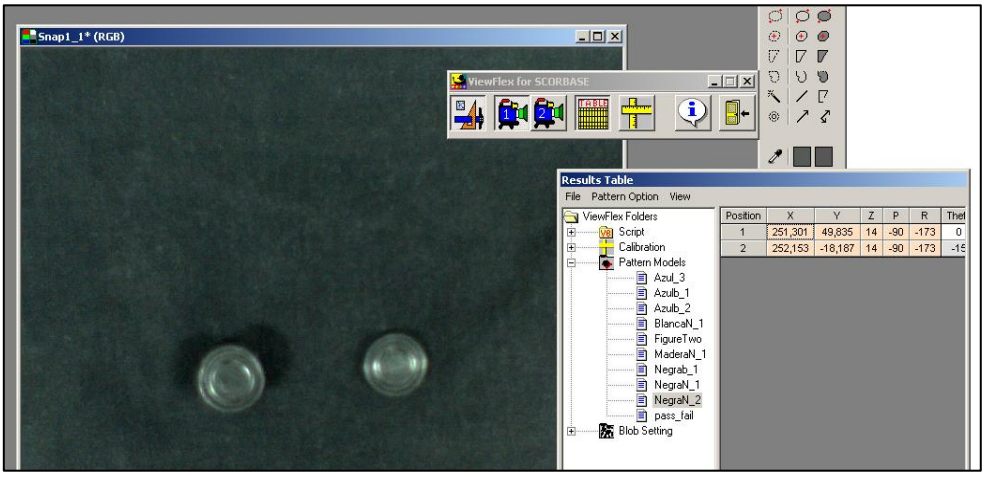


Figura 102. Algodón negro – Ocurrencias del modelo "NegraN_2"

Por tanto, los resultados obtenidos utilizando este tipo fondo son los siguientes:

Tipo de fondo	Tipo de pieza	Conclusión
Algodón negro	Rectangular azul	Se reconocen fácilmente, no producen ningún problema.
	Rectangular de colores oscuros	Se reconocen fácilmente, no producen ningún problema.
	Rectangular de colores claros	Se reconocen dependiendo de la zona donde son colocadas.
	Rectangular madera clara	Se reconocen fácilmente, no producen ningún problema.
	Circular negra	Se reconocen con una aceptación del 50-55%.
	Circular blanca	Se reconocen fácilmente, no producen ningún problema.

Figura 103. Taba de resumen del fondo de algodón negro

6.3. Calibración del brazo robótico con cámara desacoplada y acoplada al robot

El proceso de calibración es prácticamente igual para las dos posiciones de la cámara, por lo que se pasa a explicar el proceso para la cámara desacoplada del brazo robótico y al final de este apartado se hacen los apuntes necesarios para la opción de cámara acoplada al brazo robótico.

Este proceso de calibración sirve para que el brazo robótico se mueva con en consecuencia a lo que la cámara de visión está capturando y así el robot poder moverse a la coordenada exacta de la pieza. Este proceso e divide en dos pasos:

- Primer paso: guardar posiciones del robot.

Para ello, lo primero es colocar el robot en una posición dentro del campo de visión de la cámara, esta posición debe de ser guardada pulsado *"Record Position 1"* y hacer una marca en un papel. Luego se debe desplazar el robot a lo largo del eje X y hacer el mismo proceso, se guarda la posición en *"Recod Position 2"* y se hace una marca en el papel.

Una vez guardada las dos posiciones aparece en el recuadro "*Setting distance between positins (X-asis)*" una distancia en milímetros calculada por el programa. Esta es la diferencia en el eje X entre las dos posiciones guardadas, pero se debe medir manualmente con una regla las dos marcas realizadas por si se produjera algún error de medida.

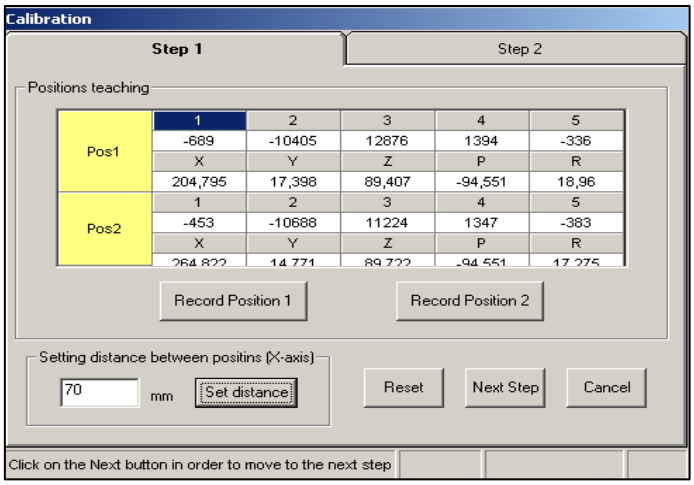


Figura 104. Paso 1 del proceso de calibración

Una vez comprobada la medida, en el caso de que la medida manual no fuera igual a la calculada esta se cambiaría por la obtenida en el proceso manual. Por último, se presiona la opción "*Set distance*" y se pasa al siguiente paso.

- Segundo paso:

Se pasa a realizar una captura de imagen de las marcas realizadas en el paso anterior. Una vez aparezca la imagen con las marcas se procede a utilizar la herramienta de lápiz que aparece a la izquierda de la imagen capturada, con ella se dibuja una recta desde la marca de la posición 1, hasta la marca de la posición 2.

Luego se pulsa otra vez la opción del lápiz, y para comprobar que ha sido calibrado correctamente se pasa el ratón sobre la primera marca y en ella se debe apreciar las coordenadas (0,0), convirtiéndose en el punto de origen.

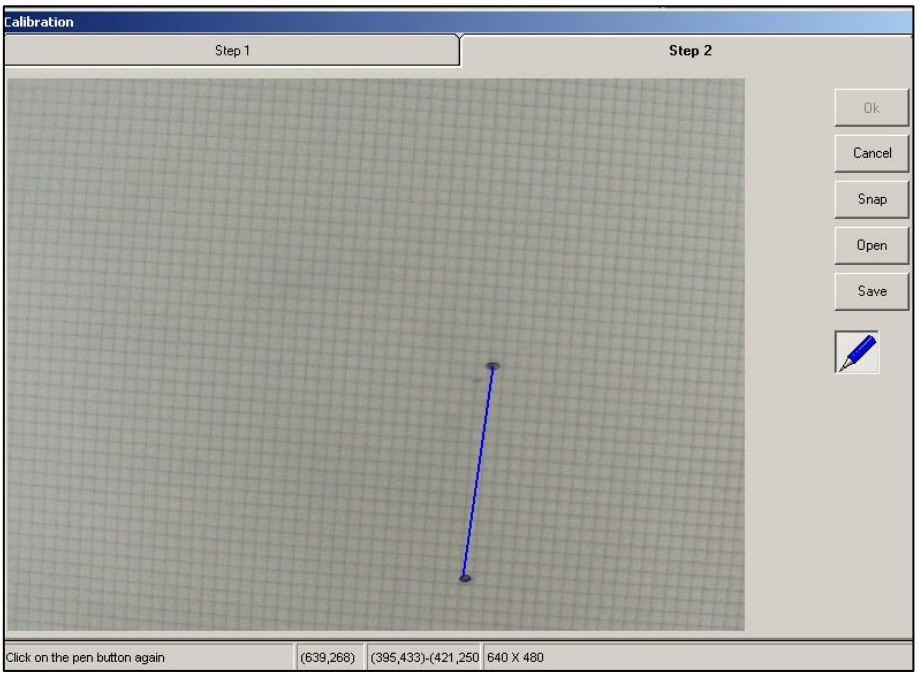


Figura 105. Paso 2 del proceso de calibración

Una vez realizado este ya se ha terminado de calibración del brazo robótico ya solo queda guardarlo y pulsar "*Ok*".

Una vez definido el proceso, la única diferencia notable para cuando la cámara este acoplada al brazo del robot es que inicialmente en el SCORBASE se debe de guardar

una posición de captura de imagen para que al pasar al paso 2 se pueda realizar la captura de las marcas.

En teoría, en un entorno industrial, la posición de la cámara esta fija y es inalterable por lo que este proceso de calibración solo debe hacer una única vez. Solo se debe volver a realizar si la posición de la cámara se va a cambiar para adaptarse a un nuevo proceso.

Pero en este estudio, cada vez que se coloque la cámara tanto acoplada al brazo robótico como desacoplada a él, se debe realizar el proceso de calibración. Esto se debe a que la cámara no siempre se encuentra en la misma posición y por factores humanos, que son difícil de controlar, la visión de la cámara puede variar.

6.4. Identificación de forma de las piezas por patrones mediante SCORBASE

Para esta prueba se ha desarrollado un programa mediante la utilización del software SCORBASE que permite detectar las piezas de los modelos “Azul_3” y “DamaNegra_1” creados en el apartado 5.2.

El programa que ha sido creado es el siguiente:

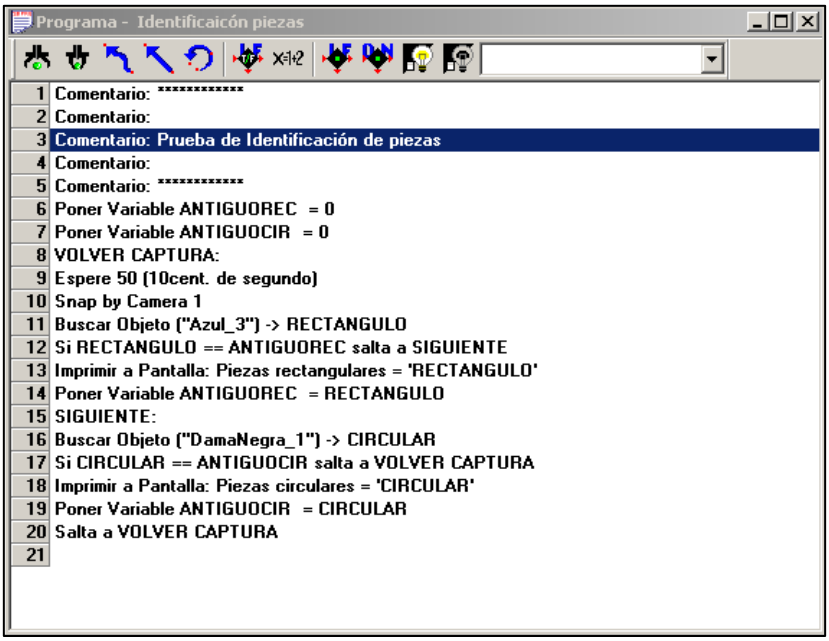


Figura 106. Programa – Identificación de piezas

Este consiste en mostrar en la pantalla de mensajes del software el número de ocurrencias producidas por cada tipo de patrón que son captadas por la cámara, pero si el número de ocurrencia sigue el mismo, este mensaje no se vuelva a mostrar.

Para una mayor comodidad a la hora de poder cambiar las piezas sin tener que detener la ejecución del programa en la línea 9 del código mostrado anteriormente se ha introducido una espera de 5 segundos.

A continuación, se muestran diferentes ejemplos del programa donde se comprueba la explicado anteriormente, probando así su correcto funcionamiento.

En esta primera figura se observa que en los mensajes aparece el reconocimiento de una única pieza rectangular.

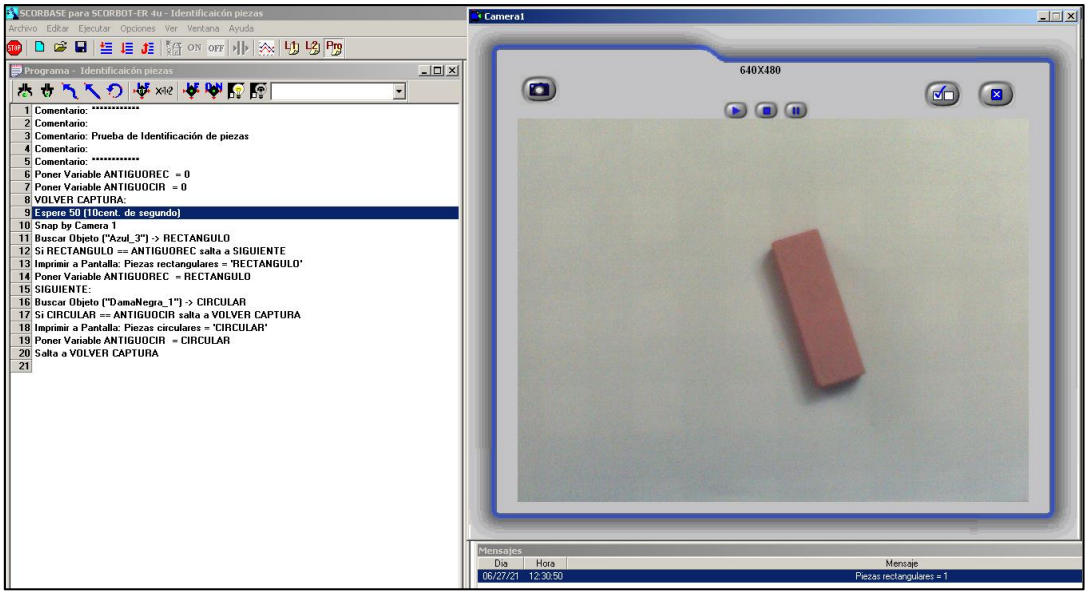


Figura 107. Ejemplo - Identificación de piezas (1)

En la siguiente figura se cambia la pieza rectangular por otra diferente y además se añade una nueva pieza de forma circular, viendo que solo es esta última la que aparece reflejada en los mensajes.

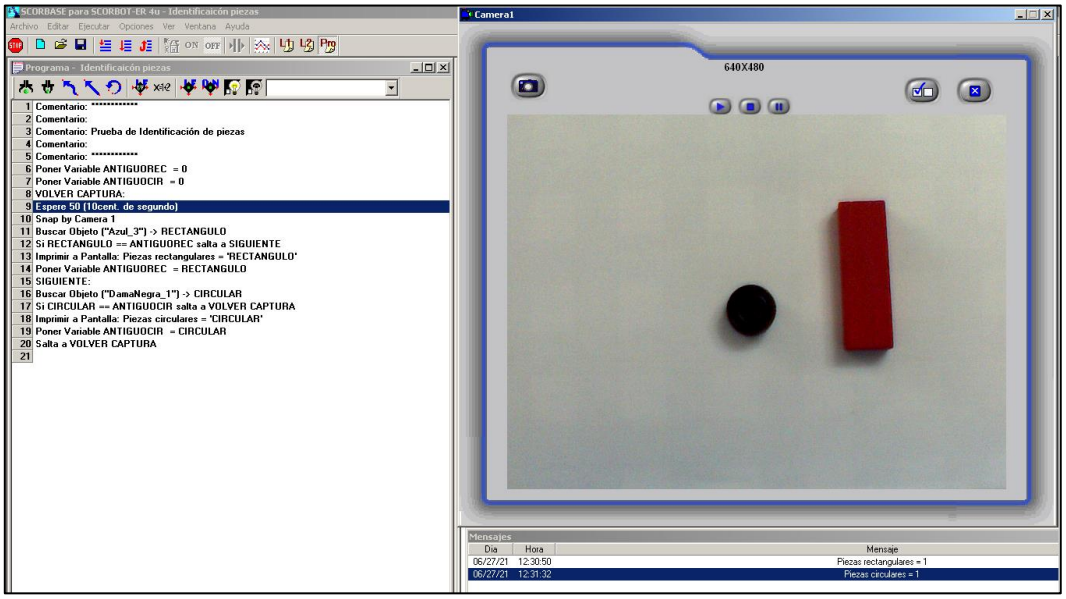


Figura 108. Ejemplo – Identificación de piezas (2)

A continuación, en la próxima figura se muestran únicamente cuatro piezas de forma rectangular y viéndose reflejado en el apartado de mensajes que la cámara solo es capaz de captar estas cuatro piezas y ninguna de forma rectangular.

Por tanto, con este ejemplo se comprueba que las piezas se van actualizando de una forma correcta.

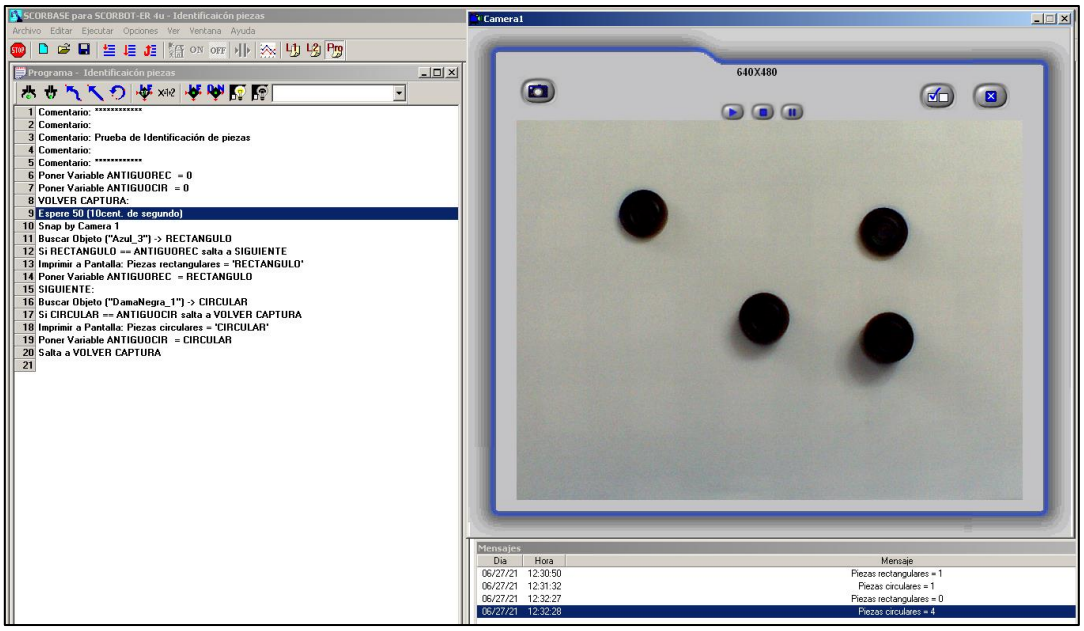


Figura 109. Ejemplo - Identificación de piezas (3)

Por último, en la siguiente figura se muestran dos piezas rectangulares de diferentes colores y una circular. Como en los ejemplos anteriores, en los mensajes se muestra que todas las piezas son reconocidas con exactitud y, a parte, se vuele a comprobar que los patrones creados en el apartado 5.2 son capaces de detectar las piezas y colores correctamente.

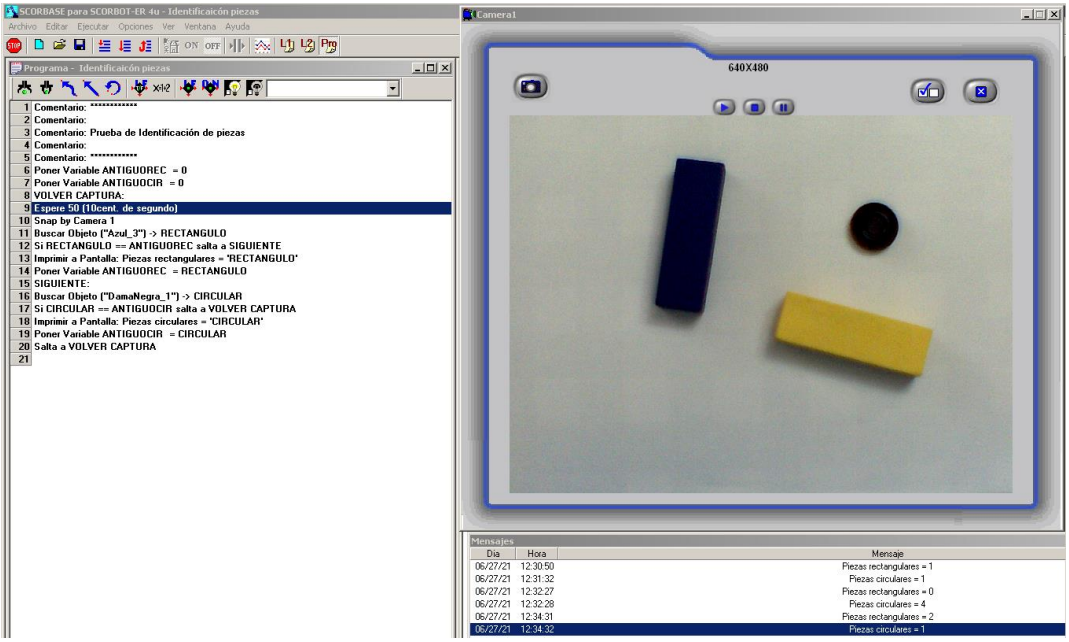


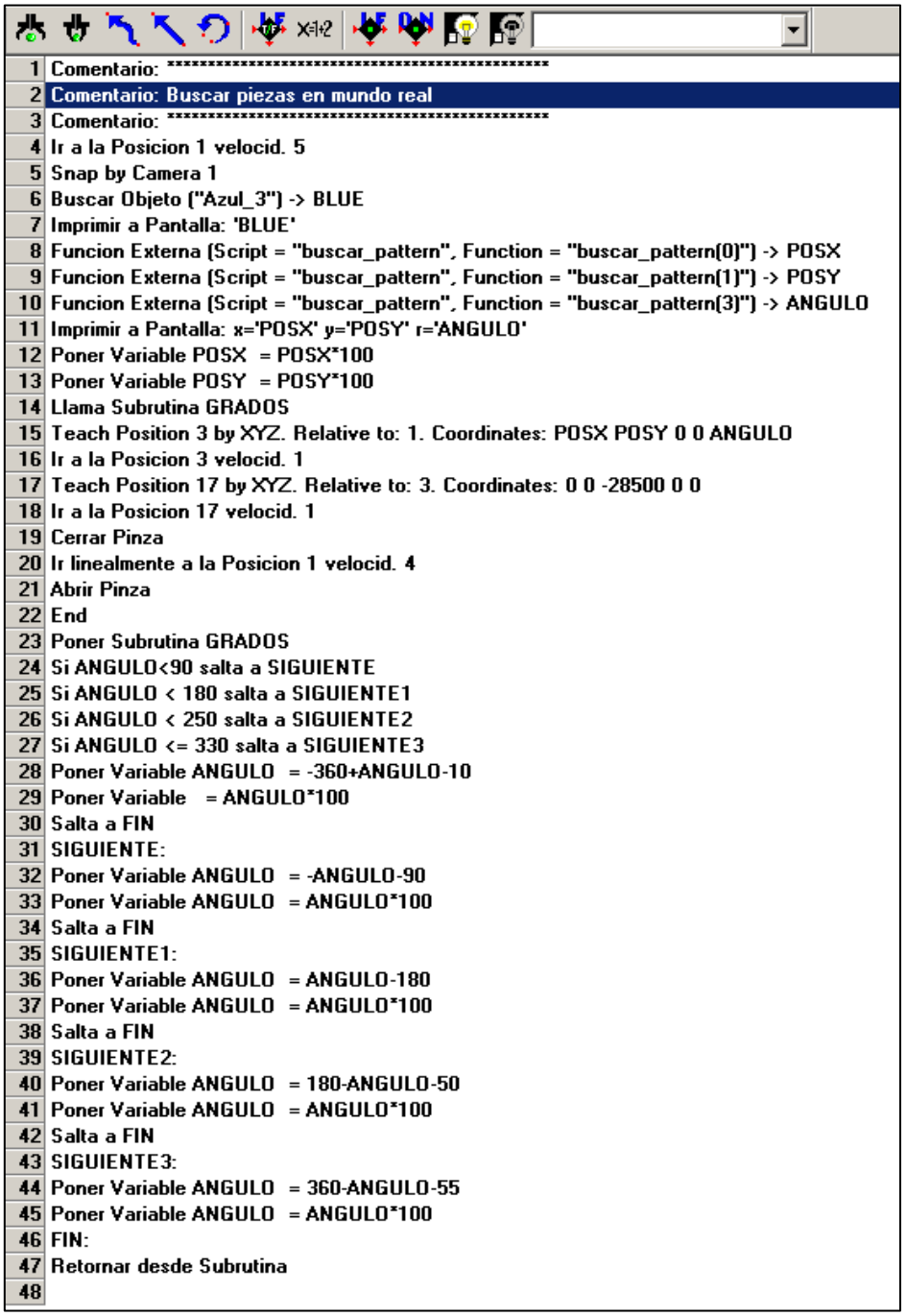
Figura 110. Ejemplo - Identificación de piezas (4)

6.5. Movimiento del robot a la posición de una pieza

En este apartado se ha desarrollado un programa que mediante la utilización del software SCORBASE se detecten las piezas del modelo “Azul_3” y que el brazo robótico vaya a por la pieza a su posición exacta.

En esta prueba solo se buscan ocurrencias con piezas rectangulares debido a que la pinza del brazo robótico no es apta para poder coger las piezas circulares de las que se dispone en este estudio, pero esto no implica que no se pueda hacer con otro tipo de piezas.

A continuación, se muestra el programa desarrollado:



```
1  Comentario: ***********************************************
2  Comentario: Buscar piezas en mundo real
3  Comentario: ***********************************************
4  Ir a la Posicion 1 velocid. 5
5  Snap by Camera 1
6  Buscar Objeto ("Azul_3") -> BLUE
7  Imprimir a Pantalla: 'BLUE'
8  Funcion Externa (Script = "buscar_pattern", Function = "buscar_pattern(0)") -> POSX
9  Funcion Externa (Script = "buscar_pattern", Function = "buscar_pattern(1)") -> POSY
10 Funcion Externa (Script = "buscar_pattern", Function = "buscar_pattern(3)") -> ANGULO
11 Imprimir a Pantalla: x='POSX' y='POSY' r='ANGULO'
12 Poner Variable POSX  = POSX*100
13 Poner Variable POSY  = POSY*100
14 Llama Subrutina GRADOS
15 Teach Position 3 by XYZ. Relative to: 1. Coordinates: POSX POSY 0 0 ANGULO
16 Ir a la Posicion 3 velocid. 1
17 Teach Position 17 by XYZ. Relative to: 3. Coordinates: 0 0 -28500 0 0
18 Ir a la Posicion 17 velocid. 1
19 Cerrar Pinza
20 Ir linealmente a la Posicion 1 velocid. 4
21 Abrir Pinza
22 End
23 Poner Subrutina GRADOS
24 Si ANGULO<90 salta a SIGUIENTE
25 Si ANGULO < 180 salta a SIGUIENTE1
26 Si ANGULO < 250 salta a SIGUIENTE2
27 Si ANGULO <= 330 salta a SIGUIENTE3
28 Poner Variable ANGULO  = -360+ANGULO-10
29 Poner Variable  = ANGULO*100
30 Salta a FIN
31 SIGUIENTE:
32 Poner Variable ANGULO  = -ANGULO-90
33 Poner Variable ANGULO  = ANGULO*100
34 Salta a FIN
35 SIGUIENTE1:
36 Poner Variable ANGULO  = ANGULO-180
37 Poner Variable ANGULO  = ANGULO*100
38 Salta a FIN
39 SIGUIENTE2:
40 Poner Variable ANGULO  = 180-ANGULO-50
41 Poner Variable ANGULO  = ANGULO*100
42 Salta a FIN
43 SIGUIENTE3:
44 Poner Variable ANGULO  = 360-ANGULO-55
45 Poner Variable ANGULO  = ANGULO*100
46 FIN:
47 Retornar desde Subrutina
48
```

Figura 111. Programa – Buscar una pieza en el mundo real

Para poder realizar esta prueba ha sido necesario crea un documento *script* para que el software ViewFlex sea capaz de devolver las posiciones de las piezas detectadas de forma calibrada, ya que si se usa el comando "*Set position*" nos devuelve los valores de la Tabla de Resultados y estos no están calibrados.

El documento *script* que ha sido desarrollado se llama “buscar_pattern.bas”. Es un archivo en *basic* que devuelve los valores de las coordenadas X, Y o el giro de la pieza dependiendo del valor numérico de la variable de entrada “COOR” que se le haya asignado.

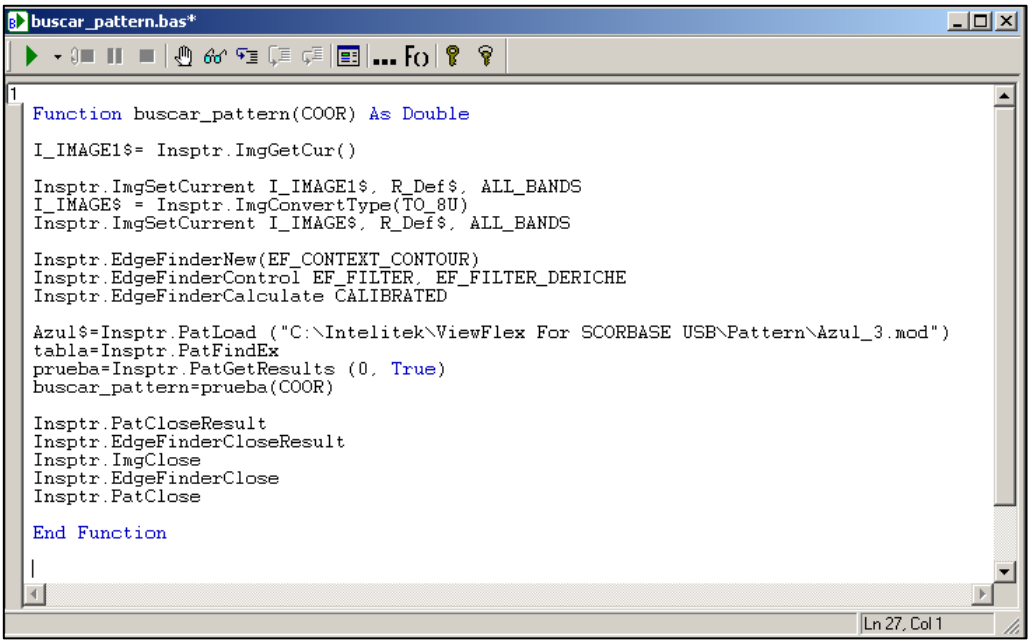


Figura 112. *Script* – Buscar_pattern

En este primeramente se pasa la imagen previamente capturada por la cámara a una escala de colores de grises, ya que el software Viewflex solo es capaz de crear patrones de esta forma. Posteriormente se le paso un filtro para definir exactamente todos los bordes que aparecen en la imagen de una mejor calidad, para luego realizar la búsqueda de n patrón previamente creado y devolver su coordenada.

Una vez explicado la script, en el programa principal de SCORBASE se puede observar que se ha creado una subrutina llamada “Grados”. Esta se encarga de hacer los ajustes necesarios para configurar el valor del eje R, giro de la pinza, sobre la posición inicial, que es la posición de captura de imagen.

Esto se debe a que cuando se hace la calibración del brazo robótico este solo afecta a los ejes X e Y, por tanto, el eje R no está en consecuencia a los del brazo del robot.

La posición donde se realizan las capturas de imágenes es en la posición número 1 y tiene las siguientes coordenadas:

#	Coord.	Eje 1	Eje 2	Eje 3	Eje 4	Eje 5	Eje 7	Eje 8	Tipo
		X (mm)	Y (mm)	Z (mm)	Elev.pinza (grad)	Giro pinza (grad)	mm/grad	mm/grad	
1	Ejes	4.86	-83.55	120.62	57.48	18.96			Abs. (Ejes)
	XYZ	204.91	17.41	290.84	-94.55	18.96			

Figura 113. Buscar piezas – Coordenadas de la posición 1

Como se puede observar, en la figura anterior, la coordenada del eje 5, que es la igual a la del eje R, no es cero y esto provoca un desfase a la hora del giro de la pinza. También se debe tener en cuenta que el 0º en el software ViewFlex no es el mismo que el del SCORBASE, entre ambos ceros hay una diferencia de 30º.

Por tanto, para los valores de ángulo del giro de la pinza del robot se debe de tener en cuenta el desfase total producido que en total está entre 50º y 55º.

Otro punto interesante en el estudio sobre el giro de la pinza es que los ángulos de operación en los que se puede mover la pinza del robot para que no golpee la cámara es limitado, este intervalo va desde los -56º hasta los 55º. Por tanto, se tienen que realizar las conversiones necesarias para que solo se opere en este intervalo.

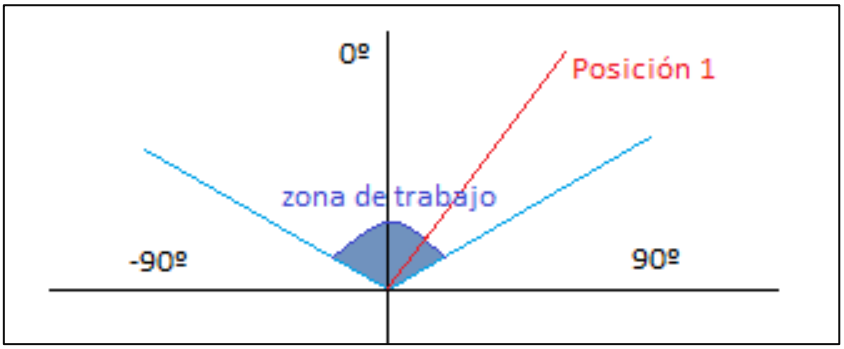


Figura 114. Zona permitida para el giro de la pinza

Se utilizan los ángulos negativos en vez de los positivos para que la pinza gire en sentido horario y antihorario. Esto se hace para evitar el choque de la cámara con alguna parte del brazo robótico.

En este caso, no se ha podido encontrar una mejor posición para colocar la cámara ya que esta es la mejor para poder realizar las capturas de las imágenes, pero en un entorno industrial el ángulo de operación para el giro de las pinzas debería de ser completo, desde 0º hasta 360º, ya que la cámara puede estar colocada en una zona donde no interfiera con la movilidad del robot.

Por último, solo queda destacar la línea 17 del código que muestra una coordenada del eje Z de -285mm. Esta coordenada se introduce de forma manual, pero no de forma aleatoria ya que está es la altura a la que está el brazo robótico con la cámara sobre el plano horizontal de la mesa donde se colocan las piezas (posición 1, figura 106).

Esto se debe hacer de esta manera porque el software Viewflex solo tiene instrucciones que devuelven las ocurrencias de los siguientes valores:

- Coordenada del eje X
- Coordenada del eje Y
- Valor del parecido en % entre el modelo y la ocurrencia
- Ángulo de giro

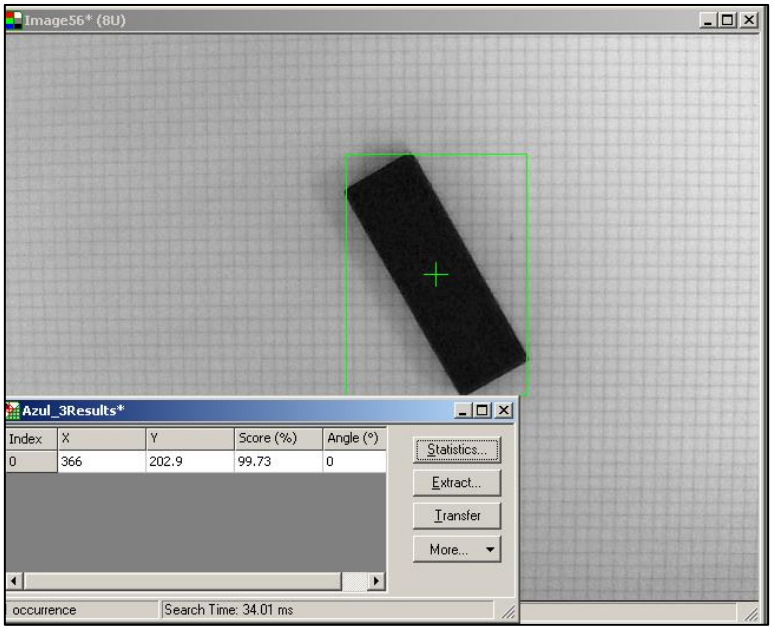


Figura 115. Tabla de resultados de la ocurrencia en Matrox

6.6. Movimiento del robot a la posición cuando hay varias piezas

Para esta prueba se realiza un programa con el software SCORBASE que permite al robot moverse para buscar la pieza que sea identificada mediante la captura realizada con la cámara colocada en el brazo robótico.

Se colocan un máximo de tres piezas en la visión de la cámara, y el robot va a ir recogiendo cada una de las piezas desde la primera ocurrencia hasta la última.

El programa que ha sido creado para realizar esta prueba es el siguiente:

```
1  Comentario: ***********************************************
2  Comentario: Buscar varias piezas
3  Comentario: ***********************************************
4  Ir a la Posicion 1 velocid. 5
5  Snap by Camera 1
6  Buscar Objeto ("Azul_3") -> BLUE
7  Imprimir a Pantalla: 'BLUE'
8  Si BLUE == -1 salta a FINN
9  VOLVER:
10 Comentario: ************ Cuando hay 1 pieza ************
11 Si BLUE == 0 salta a FINN
12 Si BLUE == 2 salta a MAS
13 Poner Variable BLUE  = BLUE-1
14 Funcion Externa (Script = "VariasPiezas", Function = "CoorX(0)") -> POSX
15 Funcion Externa (Script = "VariasPiezas", Function = "CoorY(0)") -> POSY
16 Funcion Externa (Script = "VariasPiezas", Function = "CoorR(0)") -> ANGULO
17 Imprimir a Pantalla: x='POSX' y='POSY' angulo='ANGULO'
18 Poner Variable POSX  = POSX*100
19 Poner Variable POSY  = POSY*100
20 Llama Subrutina GRADOS
21 Teach Position 3 by XYZ. Relative to: 1. Coordinates: POSX POSY 0 0 ANGULO
22 Ir a la Posicion 3 velocid. 1
23 Teach Position 17 by XYZ. Relative to: 3. Coordinates: 0 0 -28500 0 0
24 Ir a la Posicion 17 velocid. 1
25 Cerrar Pinza
26 Ir a la Posicion 2 velocid. 1
27 Abrir Pinza
28 Ir a la Posicion 1 velocid. 4
29 Salta a VOLVER
30 Comentario: ************ Cuando hay 2 piezas ************
31 MAS:
32 Si BLUE == 3 salta a MAS1
33 Poner Variable BLUE  = BLUE-1
34 Funcion Externa (Script = "VariasPiezas", Function = "CoorX(1)") -> POSX
35 Funcion Externa (Script = "VariasPiezas", Function = "CoorY(1)") -> POSY
36 Funcion Externa (Script = "VariasPiezas", Function = "CoorR(1)") -> ANGULO
37 Imprimir a Pantalla: x='POSX' y='POSY' angulo='ANGULO'
38 Poner Variable POSX  = POSX*100
39 Poner Variable POSY  = POSY*100
40 Llama Subrutina GRADOS
41 Teach Position 3 by XYZ. Relative to: 1. Coordinates: POSX POSY 0 0 ANGULO
42 Ir a la Posicion 3 velocid. 1
43 Teach Position 17 by XYZ. Relative to: 3. Coordinates: 0 0 -28500 0 0
44 Ir a la Posicion 17 velocid. 1
45 Cerrar Pinza
46 Ir a la Posicion 2 velocid. 1
47 Abrir Pinza
48 Ir a la Posicion 1 velocid. 4
49 Salta a VOLVER
50 Comentario: ************  Cuando hay 3 piezas ************
51 MAS1:
52 Si BLUE == 4 salta a FINN
```

Figura 116. Programa – Buscar varias piezas (1)

```
50 Comentario: ********** Cuando hay 3 piezas **********
51 MAS1:
52 Si BLUE == 4 salta a FINN
53 Funcion Externa (Script = "VariasPiezas", Function = "CoorX(2)") -> POSX
54 Funcion Externa (Script = "VariasPiezas", Function = "CoorY(2)") -> POSY
55 Funcion Externa (Script = "VariasPiezas", Function = "CoorR(2)") -> ANGULO
56 Imprimir a Pantalla: x='POSX' y='POSY' angulo='ANGULO'
57 Poner Variable POSX  = POSX*100
58 Poner Variable POSY  = POSY*100
59 Llama Subrutina GRADOS
60 Teach Position 3 by XYZ. Relative to: 1. Coordinates: POSX POSY 0 0 ANGULO
61 Ir a la Posicion 3 velocid. 1
62 Teach Position 17 by XYZ. Relative to: 3. Coordinates: 0 0 -28500 0 0
63 Ir a la Posicion 17 velocid. 1
64 Cerrar Pinza
65 Ir linealmente a la Posicion 1 velocid. 4
66 Abrir Pinza
67 Ir a la Posicion 2 velocid. 1
68 Abrir Pinza
69 Ir a la Posicion 1 velocid. 4
70 FINN:
71 End
72 Poner Subrutina GRADOS
73 Si ANGULO<90 salta a SIGUIENTE
74 Si ANGULO < 180 salta a SIGUIENTE1
75 Si ANGULO < 250 salta a SIGUIENTE2
76 Si ANGULO <= 330 salta a SIGUIENTE3
77 Poner Variable ANGULO  = -360+ANGULO-10
78 Poner Variable  = ANGULO*100
79 Salta a FIN
80 SIGUIENTE:
81 Poner Variable ANGULO  = -ANGULO-90
82 Poner Variable ANGULO  = ANGULO*100
83 Salta a FIN
84 SIGUIENTE1:
85 Poner Variable ANGULO  = ANGULO-180
86 Poner Variable ANGULO  = ANGULO*100
87 Salta a FIN
88 SIGUIENTE2:
89 Poner Variable ANGULO  = 180-ANGULO-50
90 Poner Variable ANGULO  = ANGULO*100
91 Salta a FIN
92 SIGUIENTE3:
93 Poner Variable ANGULO  = 360-ANGULO-55
94 Poner Variable ANGULO  = ANGULO*100
95 FIN:
96 Retornar desde Subrutina
```

Figura 117. Programa – Buscar varias piezas (2)

Se puede observar una estructura similar a la usada en el apartado anterior, ya que se van a utilizar el mismo tipo de piezas rectangulares y el modelo ya creado “Azul_3”.

El código ha sido desarrollado de esta forma debido a limitaciones que se presentan en el software Viewflex a la hora de indicar las variables de inicio de la *script.*

Anteriormente se probó crear una variable en SCROBASE que detectará el número de ocurrencias totales, y que de forma decreciente esta fuera buscando cada pieza detectada de una en una, de está forma el código estaría más optimizado. Pero, al llamar a una función *script* de Viewflex y que la variable declarada en el encabezado de esta sea otra variable, y no un número entero, provoca que Viewflex solo sea capaz de detectar un 1, aunque esta variable sea actualizada en el SCORBASE.

Por otro lado, se observa el uso de una *script* llamada “VariasPiezas” en el que según la función que se llame se devuelve el valor de la coordenada X, el valor de la coordenada Y o el ángulo de giro.

```
Function CoorX (OCURRENCIA) As Double

I_IMAGE1$= Insptr.ImgGetCur()

Insptr.ImgSetCurrent I_IMAGE1$, R_Def$, ALL_BANDS
I_IMAGE$ = Insptr.ImgConvertType(TO_8U)
Insptr.ImgSetCurrent I_IMAGE$, R_Def$, ALL_BANDS

Insptr.EdgeFinderNew(EF_CONTEXT_CONTOUR)
Insptr.EdgeFinderControl EF_FILTER, EF_FILTER_DERICHE
Insptr.EdgeFinderCalculate CALIBRATED

Azul$=Insptr.PatLoad ("C:\Intelitek\ViewFlex For SCORBASE USB\Pattern\Azul_3.mod")
Insptr.PatFindEx
prueba=Insptr.PatGetResults (OCURRENCIA, True)
CoorX=prueba(0)

Insptr.PatCloseResult
Insptr.EdgeFinderCloseResult
Insptr.ImgClose
Insptr.EdgeFinderClose
Insptr.PatClose

End Function
```

Figura 118. *Script* – CoorX

```
Function CoorY(OCURRENCIA) As Double

I_IMAGE1$= Insptr.ImgGetCur()

Insptr.ImgSetCurrent I_IMAGE1$, R_Def$, ALL_BANDS
I_IMAGE$ = Insptr.ImgConvertType(TO_8U)
Insptr.ImgSetCurrent I_IMAGE$, R_Def$, ALL_BANDS

Insptr.EdgeFinderNew(EF_CONTEXT_CONTOUR)
Insptr.EdgeFinderControl EF_FILTER, EF_FILTER_DERICHE
Insptr.EdgeFinderCalculate CALIBRATED

Azul$=Insptr.PatLoad ("C:\Intelitek\ViewFlex For SCORBASE USB\Pattern\Azul_3.mod")
Insptr.PatFindEx
prueba=Insptr.PatGetResults (OCURRENCIA, True)
CoorY=prueba(1)

Insptr.PatCloseResult
Insptr.EdgeFinderCloseResult
Insptr.ImgClose
Insptr.EdgeFinderClose
Insptr.PatClose

End Function
```

Figura 119. *Script* – CoorY

```
Function CoorR(OCURRENCIA) As Double

I_IMAGE1$= Insptr.ImgGetCur()

Insptr.ImgSetCurrent I_IMAGE1$, R_Def$, ALL_BANDS
I_IMAGE$ = Insptr.ImgConvertType(TO_8U)
Insptr.ImgSetCurrent I_IMAGE$, R_Def$, ALL_BANDS

Insptr.EdgeFinderNew(EF_CONTEXT_CONTOUR)
Insptr.EdgeFinderControl EF_FILTER, EF_FILTER_DERICHE
Insptr.EdgeFinderCalculate CALIBRATED

Azul$=Insptr.PatLoad ("C:\Intelitek\ViewFlex For SCORBASE USB\Pattern\Azul_3.mod")
Insptr.PatFindEx
prueba=Insptr.PatGetResults (OCURRENCIA, True)
CoorR=prueba(3)

Insptr.PatCloseResult
Insptr.EdgeFinderCloseResult
Insptr.ImgClose
Insptr.EdgeFinderClose
Insptr.PatClose

End Function
```

Figura 120. *Script* - CoorR

6.7. Creación de nuevos modelos de formas de piezas

En esta prueba se crea un programa en el software SCORBASE que sea capaz de crear un modelo nuevo y como no existe ninguna función que sea capaz de realizar esto se crea una *script* capaz de crearlo desde cero.

Este archivo escrito en *Basic* se llama “Crea_modelos_central.bas”, y es capaz de crear el nuevo modelo a partir de una captura de imagen realiza previamente.

A continuación, se muestra el código capaz de realizar dicha función:

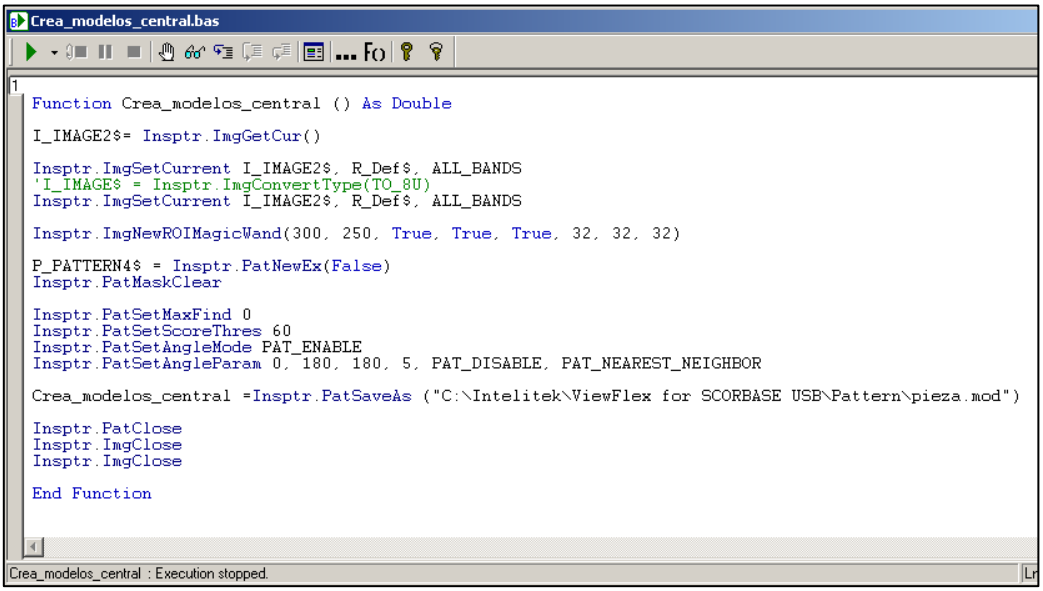


Figura 121. *Script* – Crea_modelos_central

Para poder crear el modelo deseado es necesario que previamente se coloque la pieza en el centro de la imagen captada por la cámara, de no seguir este paso el programa no será capaz de realizar el modelo.

Las características indicadas en el modelo son con una aceptación del 70%, que busque todas las ocurrencias posibles y en todos los ángulos de giro (360º).

Por otro lado, se ha comprobado que es mejor utilizar la función *MagicWand* en la imagen inicial a color que en una de escala de grises, ya que en esta última se captaban sombras que empeoraban la calidad del modelo.

Este programa solo realiza un modelo llamado “pieza”, por tanto, solo se puede guardar un único modelo. Esto implica que se va a ir sobrescribiendo el modelo cada vez que se utilice la función.

El programa creado en SCORBASE es el siguiente:

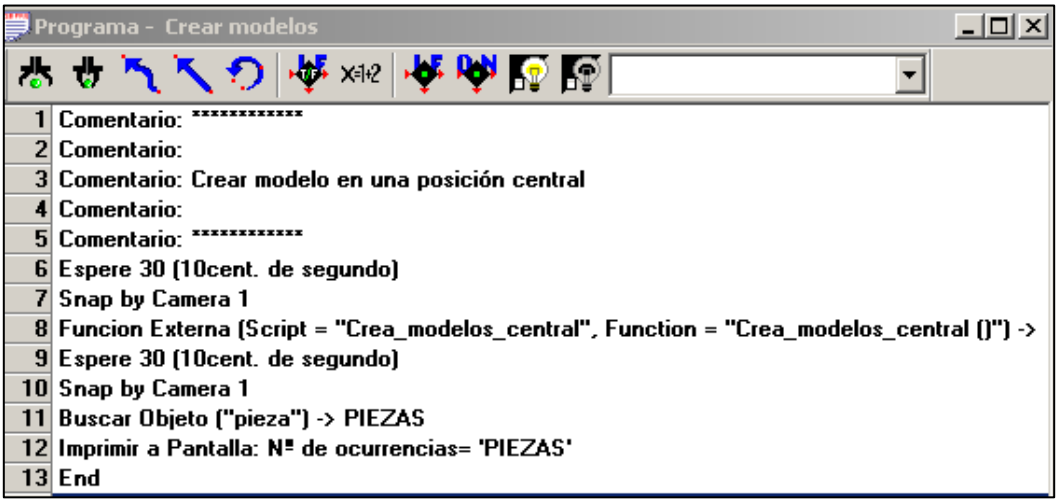


Figura 122. Programa – Crear un modelo nuevo desde una posición central

Este programa hace una captura inicial para crear el modelo de la pieza mediante la función que aparece en la Figura 101. Luego pasa a una espera de 3 segundo para poder cambiar la pieza de posición o introducir otra y se realiza otra captura para luego pasar a buscar las ocurrencias del modelo creado y posteriormente mostrar en la interfaz de mensajes.

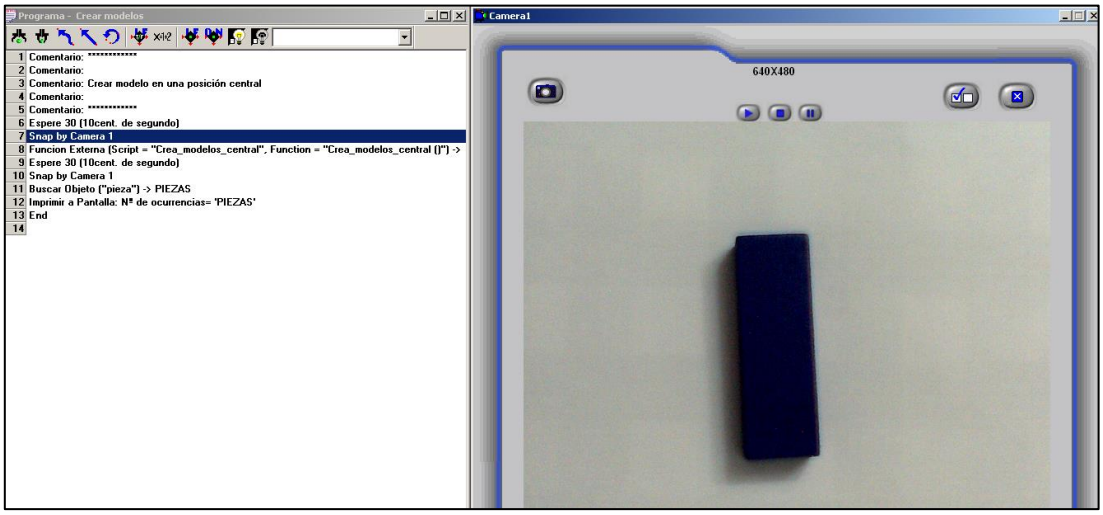


Figura 123. Ejemplo – Crear un modelo nuevo desde una posición central (1)

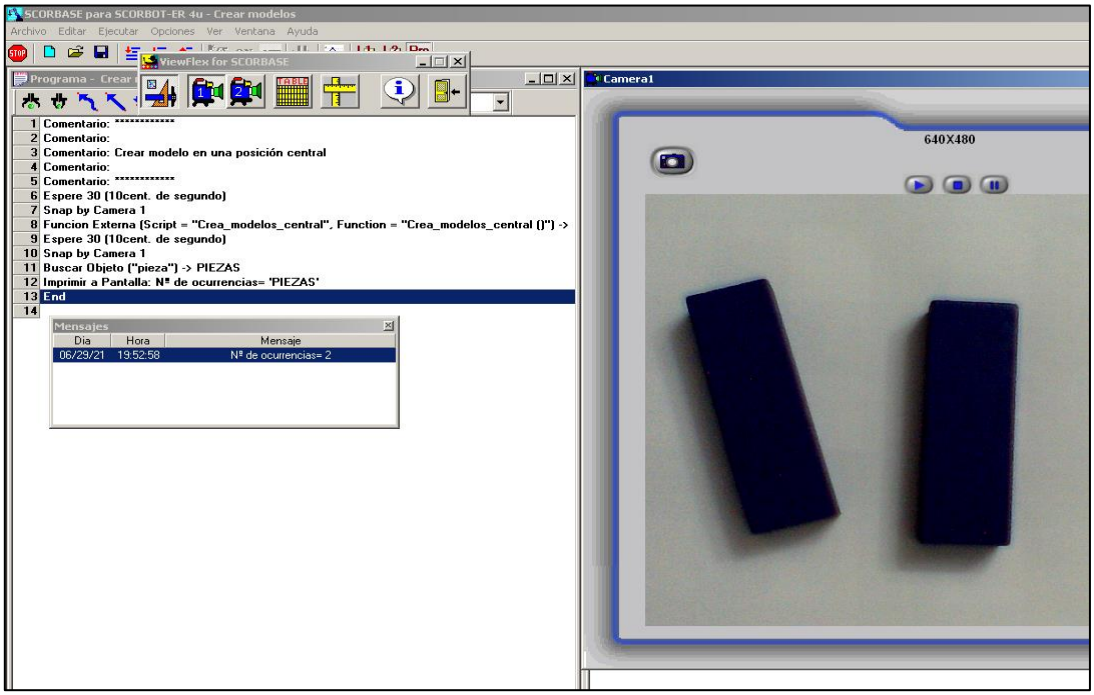


Figura 124. Ejemplo – Crear un modelo nuevo desde una posición central (2)

6.8. Realización de un puzle

Para la realización de esta prueba es necesario disponer de un puzle con piezas de formas y colores diferentes para que el robot cree un nuevo modelo sobre cada una de las piezas para, posteriormente, buscar el modelo en la zona donde debe ser encajada. Esta prueba se realiza con el siguiente puzle:

Figura 125. Puzle de formas y colores.

Se observa que el puzle consta de doce piezas en total, pero para este estudio solo se utilizan nueve de ellas, debido a las limitaciones físicas del robot. También, se ha tenido que pintar la base del puzle de color blanco. Esto se debe a que la captura de imagen de la pieza que se va a colocar se realiza sobre un fondo blanco, mientras que el color de la zona donde se va a encajar la pieza es de color madera claro, esto provoca que la tarea de encontrar el hueco sea más dificultosa.

Otra modificación realizada es que el fondo del hueco también ha sido pintado, pero de tonos parecidos al de las piezas para que la búsqueda del modelo sea más exacta y de esta manera no tener que bajar el tanto por ciento de aceptación al buscar dicho hueco.

Figura 126. Puzle con modificaciones para mejorar la precisión

Para controlar toda la prueba se necesita crear un programa en SCORBASE que hace uso de los diferentes bloques creados en los apartados anteriores y que funciona siguiendo el siguiente diagrama de flujo:

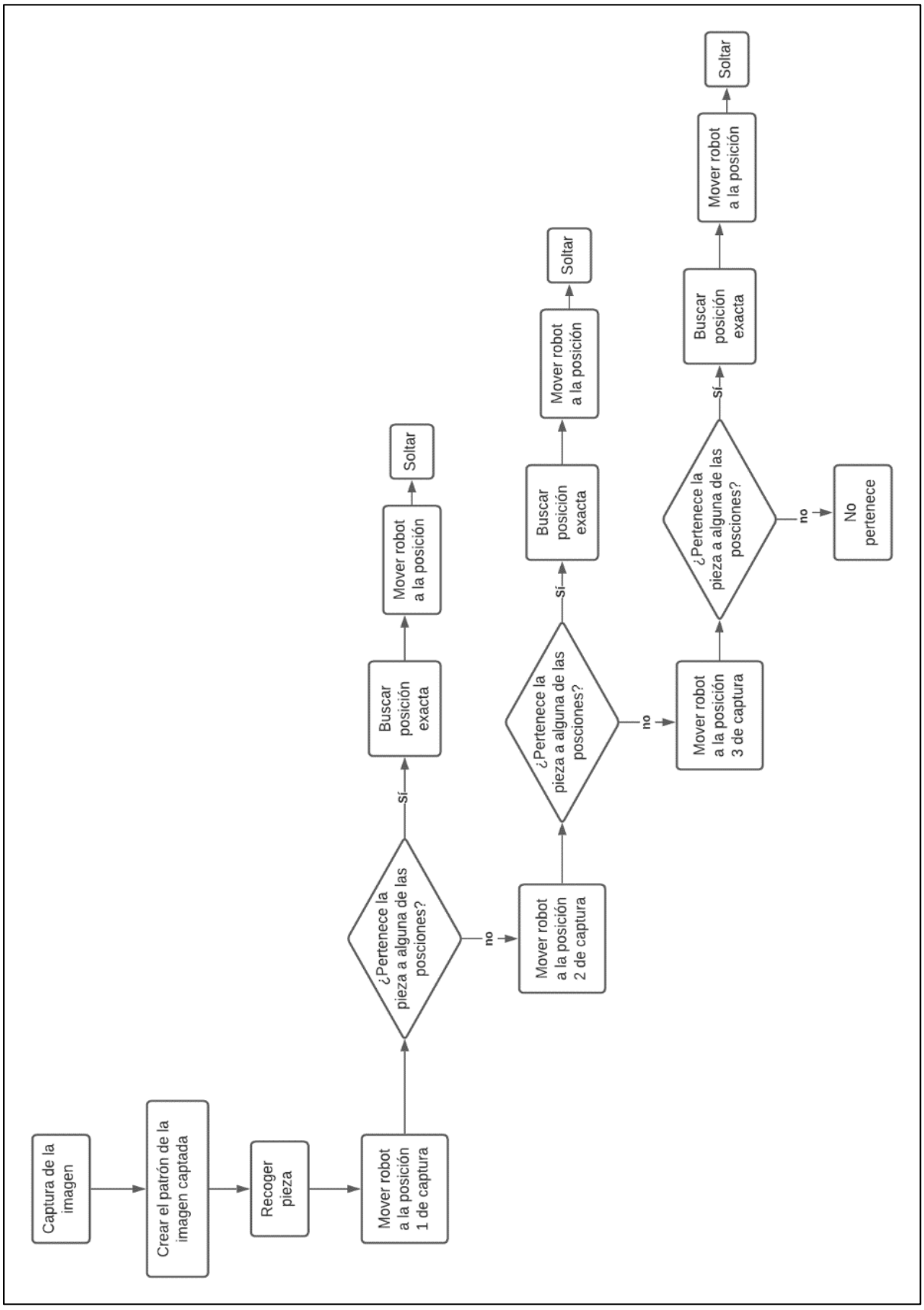


Figura 127. Diagrama de flujo del programa "Puzle"

En este diagrama se puede observar que el robot hace una captura inicial de la imagen de la pieza que se desea colocar, para posteriormente crear un nuevo modelo de esta y así buscar la posición exacta donde colocarla.

Este modelo busca en tres posiciones predefinidas en las que la cámara capta solamente tres huecos posibles para colocar la pieza. Al hacer la comparación del modelo y no encontrar ninguna similitud se pasa a la siguiente posición predefinida. Si en la última posición no se detecta ninguna similitud, aparece un mensaje indicando que la pieza cogida no se puede colocar y es soltada en otra posición diferente.

Debido a que la captura del modelo y la búsqueda de la pieza se hacen en posiciones diferentes es necesario realizar cuatro calibraciones diferentes y estas deben de ser inicializadas cada vez que esté en su posición correspondiente.

Esta inicialización se debe de hacer de forma manual debido al modelo de ViewFlex que se utiliza en el estudio, ya que no permite realizarlo de otra manera, ya que inicialmente se creó un código en una *script* llamada “Calibra.CAL” que trataba de inicializar la calibración dependiendo de la posición en la que se encontrara el robot.

La *script* mencionada tendría la estructura mostrada en la siguiente figura. Se debe de crear una función por cada calibración, previamente realizada, que se desee utilizar.

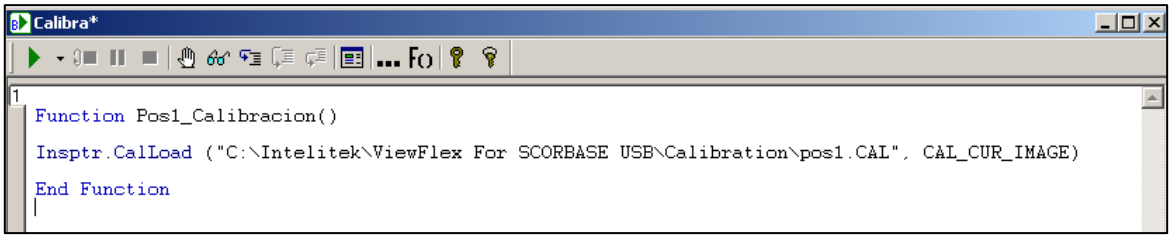


Figura 128. *Script* – Calibra

Esta solución fue creada debido a la imposibilidad de abarcar las diferentes posiciones de captura con una única calibración, pero, como se ha comentado con anterioridad no es la solución válida. Por tanto, el proceso de calibración debe de hacerse de forma manual y abrirse antes de detectar las posiciones de las figuras/huecos que se deseen conocer.

También, referente a la calibración, hay que tener en cuenta que cada vez que se usa el robot por primera vez se debe hacer el proceso de calibración de las posiciones de captura. Esto se debe a que cada vez que se coloque la cámara nuevamente en la pinza del robot, la imagen ha podido cambiar y esto puede llegar a provocar errores no deseados para encontrar las piezas/huecos.

Otro inconveniente que ha tenido esta prueba han sido los ángulos de giro de la pinza del robot. Anteriormente se hizo el estudio del giro para el apartado de buscar una pieza, desarrollado en el apartado 6.5., que también es válido para la primera posición de captura de imagen para la creación del modelo.

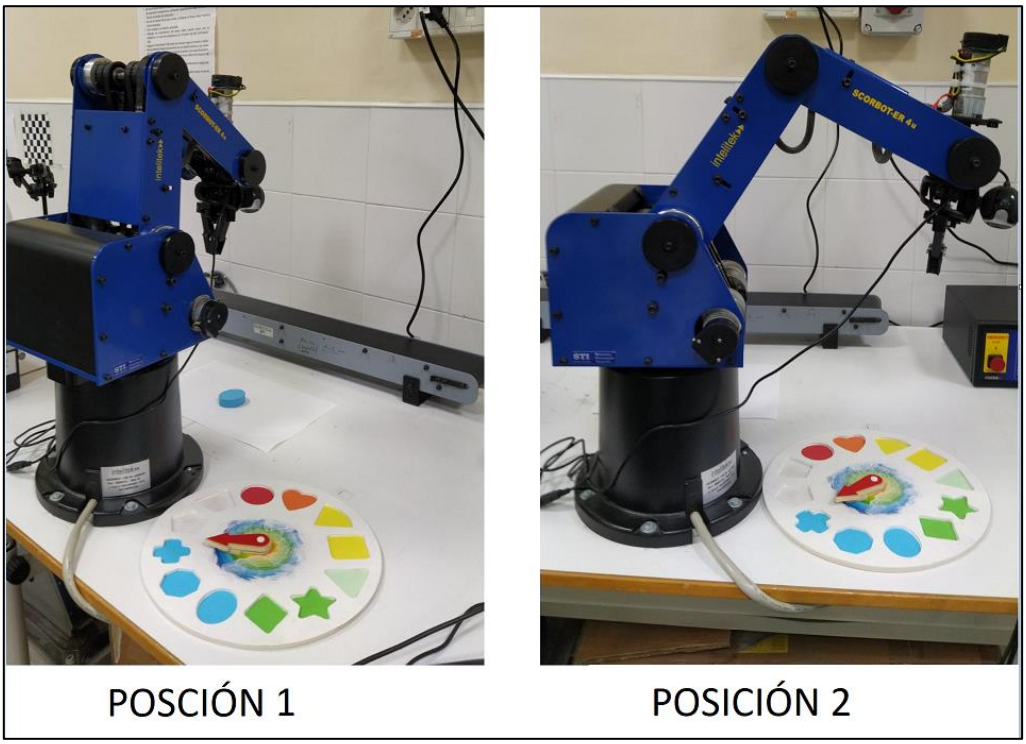


Figura 129. Diferentes posiciones para hacer la captura de imagen

En cambio, para la posición de captura de imagen de buscar el hueco donde se debe colocar la pieza, esta solo varía en el eje del hombro del robot en coordenadas articulares (W) y hace que los valores de los ángulos de giro de la pinza del robot calculados en el apartado 6.5. no sean válidos, provocando un nuevo estudio sobre estos de la misma forma que se ha sido desarrollado para la posición 1.

De otra forma, también hay que destacar que al usar un robot educativo para este estudio la precisión no es excelente y como se puede observar en la siguiente figura, los huecos donde se deben colocar son justos.

Figura 130. Piezas encajadas en sus huecos

Entonces, dependiendo de la buena calibración que se realice previamente se puede obtener mejores o peores resultados a la hora de colocar la pieza en su posición exacta.

Otro factor que provoca que la pieza no pueda ser colocada correctamente y hacerla encajar es debido al movimiento para recoger la pieza desde la primera posición, ya que se puede producir un leve golpe a la pieza en la instrucción de cerrar pinza. Este movimiento no se contempla en los cálculos realizados para traspasar las coordenadas desde la primera posición a la segunda y por tanto hace que se vaya acumulando el error.

A continuación, se muestran dos piezas colocadas por la pinza del brazo robótico donde se puede apreciar el fallo a la hora de colocar la pieza en su sitio exacto.

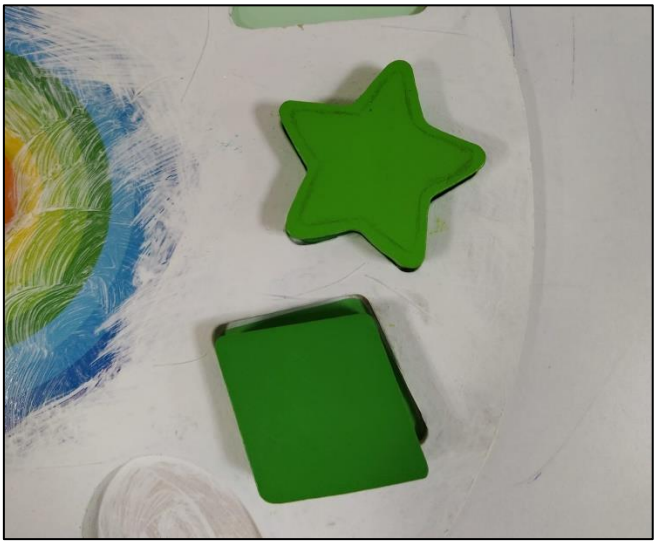


Figura 131. Piezas soltadas por el robot.

7. CONCLUSIÓN

Como conclusión, se ha logrado el objetivo principal propuesto mediante las pruebas llevadas a cabo en este estudio, por ello se concluye también que el sistema de visión artificial utilizado es apto para realizar tareas de clasificación de formas en entornos industriales.

Siguiendo la estructura de las diferentes pruebas se obtiene que:

- Una cámara incrustada totalmente en el brazo robótico permite una mayor flexibilidad a la hora de poder detectar objetos en distintas partes de la célula de fabricación (siempre que la calibración sea la adecuada).
- La calibración del robot tiene que ser lo más exacta posible para que devuelva las coordenadas de la manera más precisa posible.
- Para el modelo de piezas, lo más recomendable es una aceptación igual o superior al 70%. Cuando la aceptación es menor de 50% se empiezan a producir errores y no detecta la posición central de esta. Si se desea subir el porcentaje de aceptación se recomienda hacerlo en un entorno controlado donde la pieza no produzca ninguna sombra y la cámara se coloque cerca de la pieza.
- El material de fondo más apto probado en este estudio es el de cartón, ya que este es el que menos brillos y sombras devuelve con la refracción de la luz. Esto permite que a través de un único modelo de pieza rectangular se detecten todas las piezas de la misma forma, pero de diferente color y aunque hay que tener en cuenta que para los colores extremos blanco y negro siempre se debe de hacer dos modelos diferentes.

Tipo de fondo	Tipo de pieza	Reconocimiento
Papel blanco	Rectangular	Todos los colores, al 50% de aceptación, menos la de color madera clara que es incapaz de reconocerlas.
	Circular	Solo reconoce las de color negro, las blancas se difuminan con el fondo.

Cartulina negra	Rectangular	Todos los colores.
	Circular	Todos los colores con una aceptación del 50%.
Madera blanca	Rectangular	Todos los colores con una aceptación del 50%.
	Circular	Solo de color negro con una aceptación del 50%, las de color blanco se difuminan.
Cartón	Rectangular	Todos los colores con una aceptación del 50%.
Algodón blanco	Rectangular	Solo colores oscuros, los colores claros no los reconoce o con dificultad.
	Circular	Solo las de color negro, las blancas se difuminan con el fondo.
Algodón negro	Rectangular	Todos los colores con una aceptación del 70%.
	Circular	El color negro con una aceptación del 50%, mientras que las blancas con un 70%.

Figura 132. Tabla resumen del reconocimiento de piezas.

- El sistema de visión artificial no detecta correctamente la variación de ángulo muy pequeños, esto se ve reflejado al colocar una pieza en una posición y figura exacta.

Tras realizar varios experimentos, que inicialmente no estaban pensados y que se han ido solventando de manera efectiva mientras se avanzaba con el estudio, se ha concluido que las pruebas realizadas del procesamiento de imágenes para entornos robóticos son beneficiosas para la optimización de una línea de producción industrial. Esto se debe a que se produciría una agilización del al conocer la posición y orientación de las piezas. Además, se podría mejorar la versatilidad de la línea de producción, ya que se modificaría la tarea del robot según que tipo de pieza se detecte.

Un proceso real donde se podría llegar a implementar este tipo de solución puede dar en alguno de los siguientes casos:

- Una producción donde sea necesario detectar las piezas que tengan una malen parformación y, de esta manera, proceder a sacarlas del proceso, o para saber cuánto se asemeja dicha pieza a una ideal.
- Una producción simultánea de diferentes tipos piezas y que mediante una cámara y software de visión artificial sea capaz de detectar que tipo es para su posterior clasificación, y a su vez llevar un conteo de cada una de ellas.
- Colocación de productos finales en diferentes localizaciones. Esto se puede realizar mediante el uso de marcas que indiquen la posición de almacenamiento y así los sitios disponibles para ser colocados, impidiendo así la acumulación de varios productos en el mismo sitio.

8. BIBLIOGRAFÍA

1. **Barrientos Cruz, Antonio.** *Fundamentos de robótica.* 2. Madrid : McGraw-Hill, Interamericana de España, 2017. ISBN: 9788448156367.

2. **Cognex Corporatión .** *INTRODUCCIÓN A LA VISIÓN ARTIFICIAL.* 2016.

3. **Barragán López, Antonio.** Intez.es. [En línea] 27 de febrero de 2003. http://platea.pntic.mec.es/~alopez1/web_Eldad/_gs_srca/CI/Puertos%20USB.pdf.

4. **Eeherald.** www.eeherald.com. [En línea] 30 de 12 de 2016. http://www.eeherald.com/section/design-guide/esmod14.html.

5. **Intelitek.** *Folleto Scorbot ER-4u.*

6. —. *Manual del usuario para SCORBASE USB.*

7. **Intelitek Inc.** *Manual del Usuario de ViewFlex.* 2014.