Dispositivo de escaneo para piezas de ajedrez

ChessPieceScanner

I CONCURSO DE HARDWARE

UNIVERSIDAD COMPLUTENSE DE MADRID

FACULTAD DE INFORMÁTICA

Ángel Molina Núñez

Madrid, 27 de mayo de 2021

Resumen

El entrenamiento de redes neuronales necesita de grandes cantidades de datos para conseguir buenos resultados, y, aunque existen muchos modelos previamente entrenados pueden no adaptarse a nuestras necesiades. En nuestro caso, no hay ningún modelo completo de inferencia de piezas del ajedrez, y es por eso que decidimos desarrollar un sistema que sea capaz de realizar fotografías de forma autónoma y eficiente. Se han realizado más de 13.500 fotografías de las distintas piezas de ajedrez con las que se puede entrenar un modelo que sea capaz de detectarlas con un alto porcentaje de acierto.

Palabras clave

Aceleración de redes neuronales, Arduino , Ajedrez, Visión artificial, Redes neuronales convolucionales, Deep learning, Webcam.

Agradecimientos

En primer lugar agradezco a todos los profesores que, durante la carrera, me han enseñado todo lo que sé y me han guiado durante todos estos años. Me gustaría además agradecer especialmente a Juan Carlos Fabero Jiménez y José Manuel Mendías Cuadros por haberme animado a presentar este proyecto. También quiero agradecer a Jennifer por su apoyo constante dureante los últimos años y finalmente, a mis padres por haberme apoyado, ayudado y animado a llegar donde estoy.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

El entrenamiento de redes neuronales se basa en grandes cantidades de datos preprocesados que permiten la inferencia de parámetros. En internet podemos encontrar una gran cantidad de redes preentrenadas que podemos utilizar, no obstante, pueden no ofrecer una solución para el problema que planteemos, en nuestro caso la inferencia de piezas de ajedrez en una imagen.

En estos casos la solución es reunir una gran cantidad de datos que deben estar preprocesados para después realizar el entrenamiento de nuestra propia red. Es por esto que, hemos decidido desarrollar un sistema semiautomático de fotografiar piezas de ajedrez que, además, está basado en hardware y software libre utilizando una webcam y un microcontrolador Arduino Uno.

El dispositivo se conecta mediante 2 puertos USB a un ordenador con sistema Linux y, tras colocar una pieza, comienza a hacer fotografías desde diferentes ángulos con el fin de cubrir el mayor número de posibilidades.

Todo el código fuente, modelos 3d y planos se encuentran disponibles en el repositorio GIT: ***https://github.com/angmolin/ChessPieceScanner***

1.1. Antecedentes

Durante el desarrollo de mi Trabajo de Fin de Grado, donde se tratan técnicas de aceleración para el reconocimiento de piezas de ajedrez, uno de los puntos clave es entrenar a los modelos de inferencia de modo que consigan un alto porcentaje de acierto y así poder desarrollar un sistema que mediante un dispositivo Android sea capaz de obtener la notación FEN de una imagen de un tablero en tiempo real.

1.2. Objetivos y plan de trabajo

El servomotor del dispositivo presenta un margen de error que sería conveniente corregir, bien sustituyéndolo por otro servomotor más preciso o por un motor paso a paso. Asimismo sería conveniente que el recorte de las imágenes se realizara de forma automática, detectando los bordes de la pieza y así poder eliminar la lista de recortes de la pieza para cada posición.

Aunque hemos realizado 3 iteraciones sobre el diseño del dispositivo se podría continuar desarrollando un dispositivo autónomo, mediante el uso de una Raspberry Pi ó reduciendo el tamaño del dispositivo con una webcam que sea capaz de enfocar desde un punto más cercano.

Además este dispositivo puede ser adaptado para realizar fotografías de otros objetos de los que necesitemos una gran cantidad de imágenes de forma sencilla.

Capítulo 2

Desarrollo

2.1. Planos

El dispositivo consta de dos bases circulares, una inferior de tamaño más grande que se sostiene mediante 4 patas y que sirve de base para todos los componentes y una superior sobre la que se coloca la pieza que deseemos escanear y que llamaremos eje Z. Véanse las *fig. 2.1.1 y 2.1.2*.

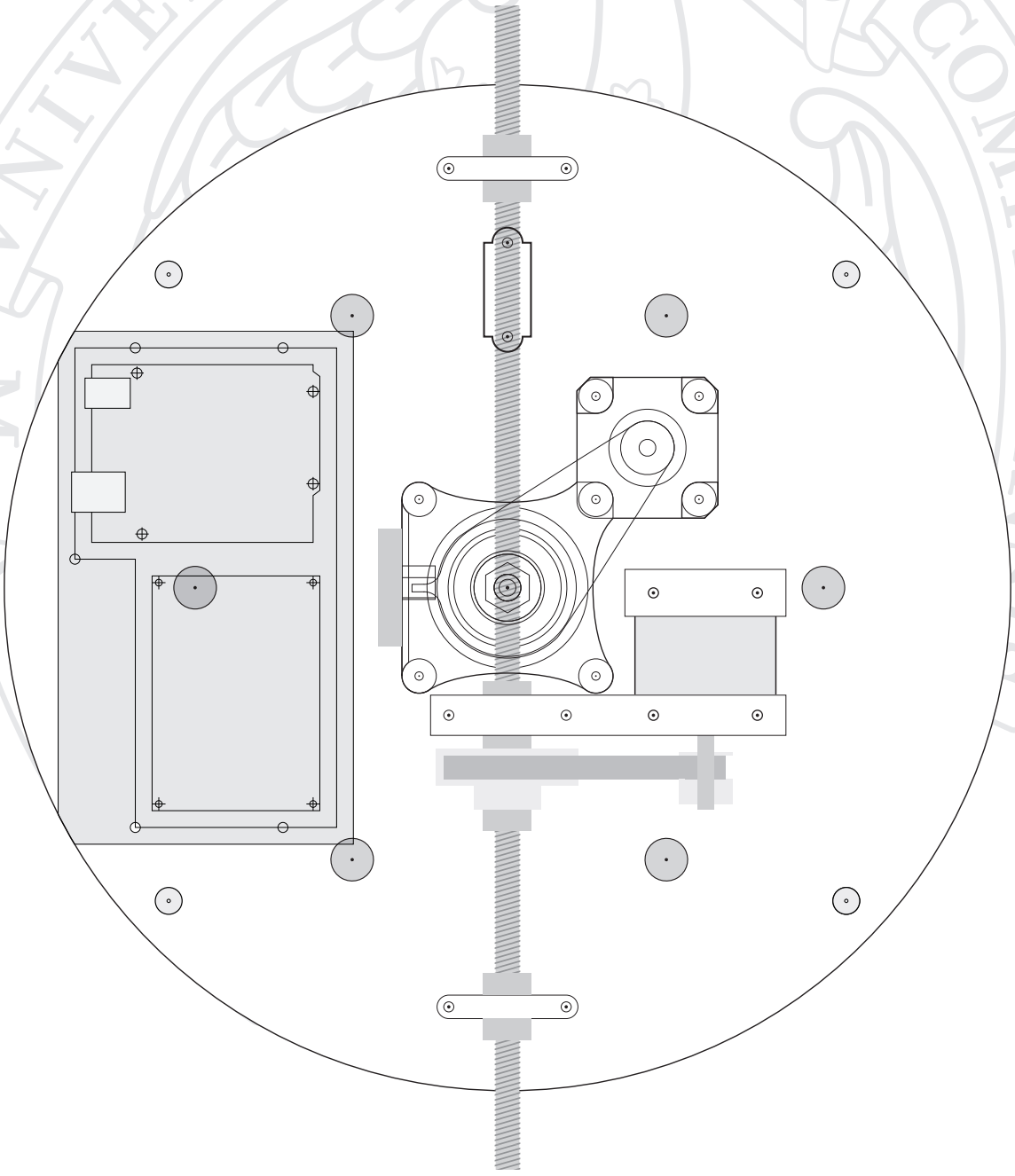

Figura 2.1.1: Base inferior del dispositivo a escala 1:2

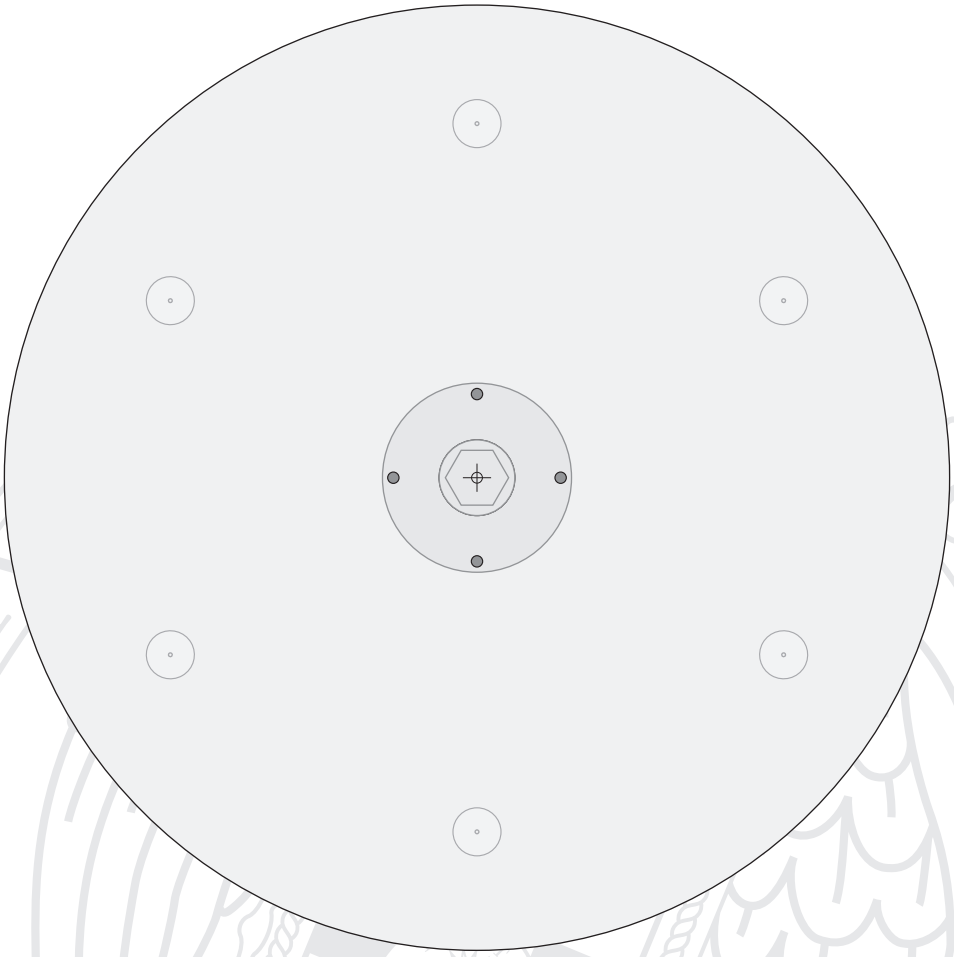

Figura 2.1.2: Base superior del dispositivo a escala 1:2

La webcam se monta en el interior de una pieza que debajo permite la instalación del servomotor que permitirá el giro de la misma. Como veremos más adelante a este eje de rotación le llamaremos eje J. Esta pieza se monta sobre otra por medio de tornillos de métrica 3 y esta otra pieza encaja con 4 varillas de metal *fig. 2.1.3* que llegan hasta el eje que llamaremos Y que consiste en una varilla roscada de métrica 8 que está montada sobre la base inferior. Véase la *fig. 2.1.4*.

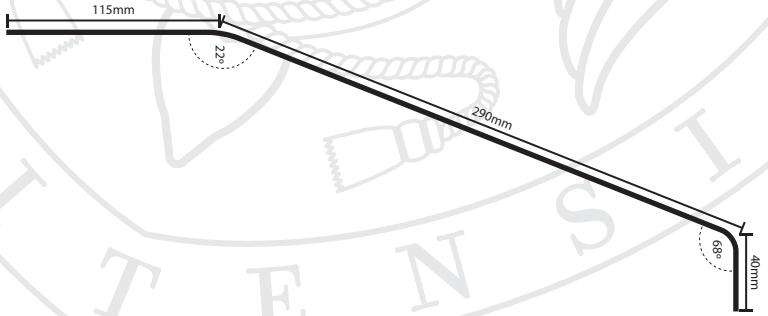

Figura 2.1.3: Varilla de metal de unión entre la webcam y la base a escala 1:4

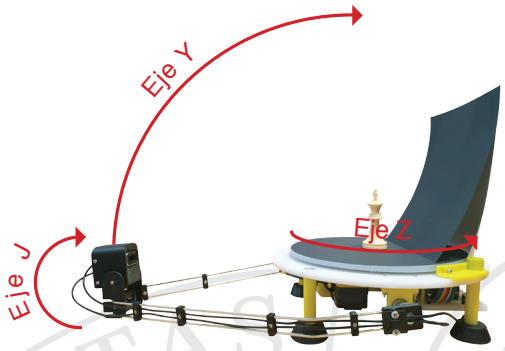

Figura 2.1.4: Lateral del dispositivo

Como mencionamos antes, existen 3 ejes de movimiento en el dispositivo, que hemos denominado como eje J, Y y Z. El eje J es un eje horizontal situado en la webcam y que permite que se mantenga tangente a la circunferencia que describe el eje Y que está situado debajo de la base inferior y cuyo centro es el centro de la varilla roscada. En cuanto al eje Z, es un eje vertical que permite la rotación de las piezas.

El microcontrolador Arduino Uno y la placa de control se montan sobre un soporte específico –véase la *fig. 2.1.5 y 2.1.6*– que también se ancla a la base inferior del dispositivo.

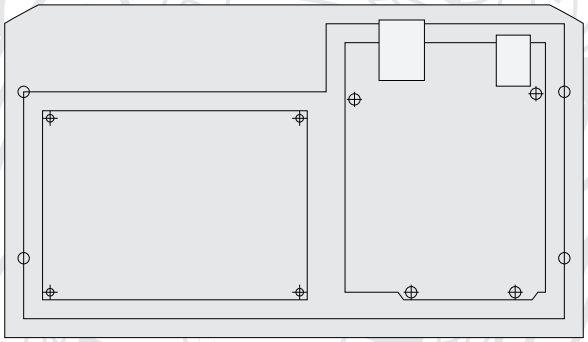

Figura 2.1.5: Soporte de Arduino y placa de control a escala 1:2

Figura 2.1.6: Soporte de Arduino y placa de control en el dispositivo

2.2. Piezas 3d

Para ensamblar todas las bases, barillas y componentes hemos desarrollado piezas que pueden ser impresas en cualquier impresora 3d. Aunque es recomendable que la impresión se realize en plástico ABS ya que es más resistente a la temperatura y después de un uso prolongado los motores pueden alcanzar temperaturas de entre 40 y 50 grados centígrados.

El montaje debe comenzar por el eje vertical o eje Z de la base inferior, cogeremos un tornillo de métrica 8 al que le colocaremos un rodamiento de diámetro interior 8mm, una arandela que actúe como separador, la polea, el cuentavueltas del eje Z *fig. 2.2.1*, un rodamiento de diámetro interior 20mm y una tuerca que fije todo. Una vez tengamos este eje encajaremos el exterior del primer rodamiento en la base inferior, dejando que la cabeza del tornillo sobresalga por la parte superior, colocamos la correa en la polea y encajaremos el segundo rodamiento sobre el soporte impreso del eje *fig. 2.2.2* después colocamos el otro extremo de la correa sobre la polea del motor y lo fijamos todo con tornillos a la base.Finalmente debemos colocar el sensor óptico de final de carrera y cubrirlo con su tapa *fig. 2.2.3*.

Figura 2.2.1: Cuentavueltas del eje Z

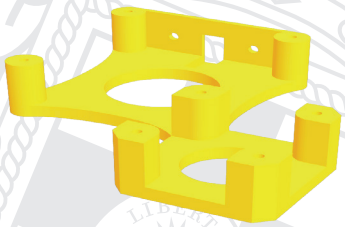

Figura 2.2.2: Soporte del eje Z

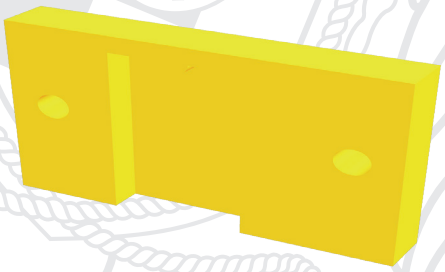

Figura 2.2.3: Tapa del sensor óptico de final de carrera del eje Z

Continuaremos después colocando el seguundo sensor óptico de final de carrera, el correspondiente al eje Y, para ello necesitaremos el soporte impreso *fig. 2.2.4*.

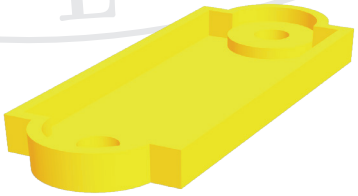

Figura 2.2.4: Soporte del sensor óptico de final de carrera del eje Y

Seguidamente montamos el eje horizontal o eje Y de la base inferior, necesitaremos una varilla roscada de métrica 8 sobre la que pondremos una polea con una tuerca a cada lado para impedir el movimiento y después encajaremos sobre esta un rodamiento de diámetro interior 20mm. Luego montaremos el rodamiento sobre el soporte impreso del eje *fig. 2.2.5*, al que también colocaremos el motor, y uniremos las poleas mediante la correa. En este punto la correa está destensada, pero no debemos preocuparnos. Ahora colocamos el cuentavueltas del eje Y *fig. 2.2.6* fijándolo a la varilla con una tuerca a cada lado y haciéndolo coincidir con el sensor que hemos colocado previamente. En este momento podemos colocar los soportes laterales *fig. 2.2.7* del eje –también con una tuerca en cada uno de sus lados– a los que previamente debemos colocar un rodamiento de diámetro interior 8 y los fijamos a la base colocando los separadores de estos soportes *fig. 2.2.8*.

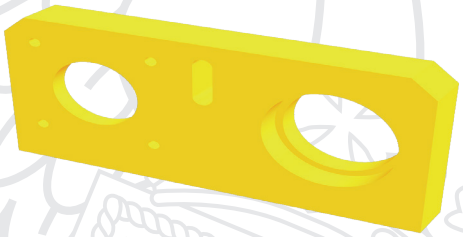

Figura 2.2.3: Soporte del eje Y.

Figura 2.2.6: Cuentavueltas del eje Y

Figura 2.2.7: Soporte lateral del eje Y

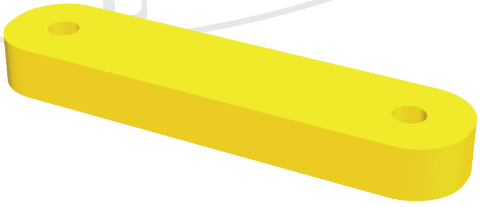

Figura 2.2.8: Separador del soporte lateral del eje Y

Una vez tengamos todo fijado es el momento de tensar la correa, para ello con la ayuda de un tornillo de métrica 6 colocamos un rodamiento en la ranura vertical que ha quedado libre del soporte del eje Y y lo arrastramos hacia abajo hasta que la correa quede tensa, una vez eso ocurra, lo apretamos.

En este momento podemos retirar la base inferior y ensamblar la parte de la webcam, para ello introducimos las varillas en el soporte de la webcam *fig. 2.2.9* y con ayuda de unos tornillos las apretamos para que queden fijas. Una vez hecho eso, fijamos al soporte los laterales *fig. 2.2.10 y 2.2.11* con ayuda de otros 2 tornillos e introduciendo los alambres por las guías. Después introducimos la webcam en la caja *fig. 2.2.12* y pasamos el cable por el interior del muelle, también introducimos el servomotor y lo fijamos con otros 2 tornillos a la caja. Más tarde colocamos la tapa trasera *fig. 2.2.13* teniendo cuidado de colorcar el muelle en su sitio y habiendo pasado los cables de la webcam y el servomotor por el hueco central y colocado la tapa de los cables *fig. 2.2.14*. Finalmente atornillamos la caja a los soportes laterales.

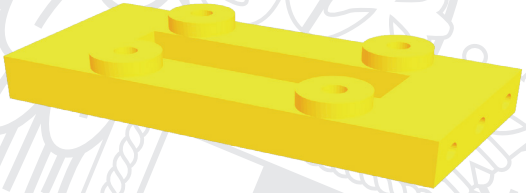

Figura 2.2.9: Soporte de la webcam

Figura 2.2.10: Soporte lateral derecho de la webcam

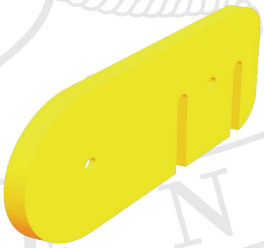

Figura 2.2.11: Soporte lateral izquierdo de la webcam

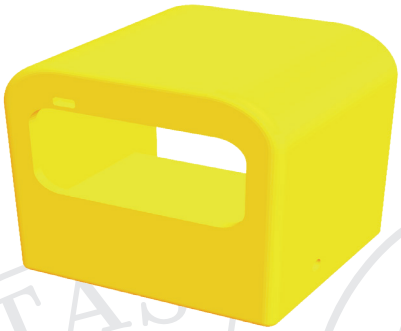

Figura 2.2.12: Caja de la webcam

Figura 2.2.13: Tapa trasera de la webcam

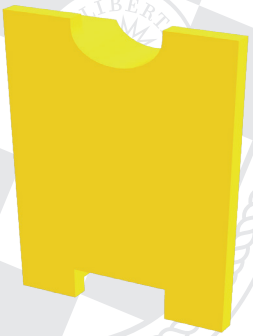

Figura 2.2.14: Tapa de los cables de la webcam

Llegados a este punto podemos colocar los soportes de los otros extremos de las varillas *fig. 2.2.15* que nos permitirán ensamblarlas con la varilla roscada de la base inferior o eje Y.

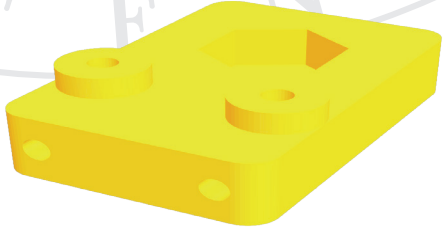

Figura 2.2.15: Soporte de ensamble de las varillas de la webcam y el eje Y

Para ensamblar ahora estas piezas al eje Y necesitaremos introducir una tuerca y una arandela en cada extremo de la varilla roscada hasta que choquen con las tuercas de los cojinetes. Después introducimos la varilla roscada por los agujeros de las piezas hasta que estas choquen con las tuercas. Una vez aquí ponemos otras dos tuercas hasta enrasarlas con los bordes de la varilla roscada y tiramos de los soportes hacia fuera hasta que las tuercas exteriores se embutan en el hueco. Finalmente aflojamos las primeras tuercas que introdujimos hasta que los soportes queden fijos.

Ahora debemos posicionar las varillas perfectamente paralelas a la base inferior, aflojar las tuercas del cuentavueltas del eje Y, colocar el cuentavueltas de modo que el sensor óptico quede tapado y volver a apretar las tuercas.

A continuación colocamos el soporte trasero del motor del eje Y *fig. 2.2.16* para evitar que el motor cabecee por la tensión de la correa.

Figura 2.2.16: Soporte trasero del motor del eje Y

En este momento podemos montar las patas de la base inferior cubriéndolas con los embellecedores *fig. 2.2.17* y la base del microcontrolador y la placa de control haciendo uso de los separadores *fig. 2.2.18*. Asimismo podemos guiar los cables de la webcam y del servomotor por las varillas utilizando las guías laterales para cables *fig. 2.2.19*, de este modo evitamos que con el movimiento puedan enredarse los cables provocando daños en el dispositivo y/o el ordenador.

Figura 2.2.17: Embellecedores de las patas

Figura 2.2.18: Separadores del soporte del microcontrolador y placa controladora

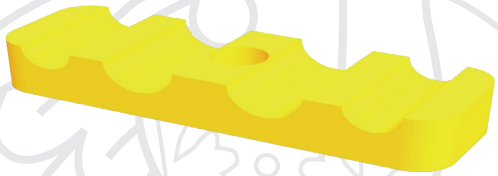

Figura 2.2.19: Guía lateral para cables

Llegados a este punto debemos ensamblar el soporte del fondo, para ello necesitaremos pegar con ayuda de un pegamento acrílico las piezas superior e inferior de los soportes izquierdo y derecho *fig. 2.2.20, 2.2.21, 2.2.22 y 2.2.23*. Después introducimos 2 pequeñas peltinas que apretamos con ayuda de un tornillo en cada extremo y colocamos el papel entre ellas para que quede recto. Finalmente debemos introducir unos imanes en los huecos laterales que harán que se peguen a los grandes tornillos que sujetan las patas y evitaran que nuestro fondo se caiga.

Figura 2.2.20: Soporte superior derecho del fondo

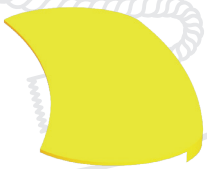

Figura 2.2.21: Soporte inferior derecho del fondo

Figura 2.2.22: Soporte superior izquierdo del fondo

Figura 2.2.23: Soporte inferior izquierdo del fondo

Finalmente colocamos las esferas de deslizamiento en la parte superior de la base inferior y atornillamos a la base superior el soporte de giro *fig. 2.2.24*

Figura 2.2.24: Soporte de giro de la base superior

Ahora tenemos nuestro dispositivo listo para funcionar. Véase la *fig. 2.2.25* para una vista general.

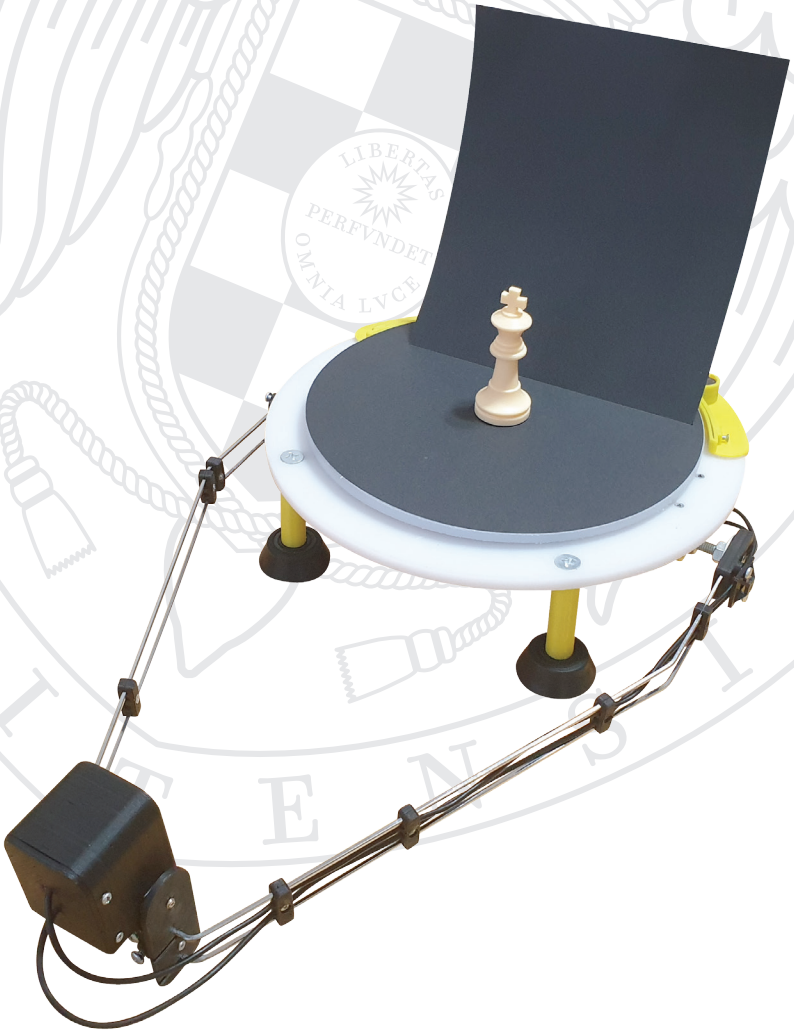

Figura 2.2.25: Vista general del dispositivo

2.3. Materiales utilizados

- Polietileno de color blanco para las bases superior e inferior
- Patas de compresor doméstico de aire acondicionado
- Filamento ABS para la impresión de las piezas 3d
- Varillas metálicas de métrica 4
- Varilla roscada de métrica 8
- Tornillería de métricas 3mm, 6mm y 8mm
- 2 rodamientos 6804ZZ
- 2 rodamientos F680ZZ
- 3 rodamientos ABEC-1
- 6 bolas transportadoras de acero de 8mm
- 2 juegos de poleas GT2 reducción 3:1
- 2 bridas metálicas

En un primer momento consideramos utilizar tableros de madera MDF como material para las bases superior e inferior y como base para el microcontrolador y la placa controladora, pero este material fue descartado por resultar demasiado flexible y al tensar la correa del eje Y se producia una ligera curvatura sobre la base. El polietileno sin embargo es un material más rígido que permite tensar la correa sin que se deforme.

2.4. Hardware

- Arduino Uno rev.3
- Motor NEMA-17 de 40mm
- Motor NEMA-14 de 28mm
- 2 drivers A4988
- 2 sensores ópticos de final de carrera
- 1 servomotor HD-1160A
- Placa de prototipado
- 5 resistencias de 1kΩ
- 1 resistencia de 330Ω
- 1 resistencia de 560Ω
- 1 condensador electrolítico de 200uF
- 2 diodos led
- 11 jumper
- Trasnformador de 12V y 1.5A

El microcontrolador se comunica con el ordenador por medio de un cable USB a través de un puerto serie con una velocidad de 115.200 baudios. La comunicación se realiza por medio de comandos, una vez recibido el comando y realizada la acción se envía un ACK al ordenador para indicar que se ha completado la acción. Podemos ver los comandos disponibles en la *tabla 2.4.1.*

Comando	Descripción
cps0	Imprime una breve descripción del objetivo del dispositivo.
cps100**$1**	Establece la posicion actual del eje **$1** del dispositivo como la posición inicial. • **$1** puede tomar los valores 'j', 'y' ó 'z'.
cps101**$1**	Calibra automáticamente el eje **$1** del dispositivo. • **$1** puede tomar los valores 'j', 'y' ó 'z'.
cps200**$1**	Libera el motor del eje **$1** cesando la entrega de corriente. • **$1** puede tomar los valores 'j', 'y' ó 'z'.
cps201**$1$2**	Mueve el eje **$1** hasta la posición **$2** en grados. • **$1** puede tomar los valores 'j', 'y' ó 'z'. • **$2** puede tomar los valores [0, 120] si **$1** es 'j', [0, 90] si **$1** es 'y' ó [0, 360) si **$1** es 'z'.
cps300**$1**	Devuelve el ángulo actual del eje **$1** en grados. • **$1** puede tomar los valores 'j', 'y' ó 'z'.
cps301**$1**	Devuelve el estado del dispositivo óptico de final de carrera del eje **$1**. • **$1** puede tomar los valores 'y' ó 'z'.
cps400**$1**	Establece el tiempo de espera entre pulsos del motor del eje **$1**. • **$1** puede tomar los valores 'y' ó 'z'.
cps401**$1**	Establece el número de pasos que debe avanzar o retroceder el motor del eje **$1** para moverse 1 grado. • **$1** puede tomar los valores 'y' ó 'z'.

Tabla 2.4.1: Comandos disponibles del microcontrolador

El microcontrolador se conecta a todos los demás componentes hardware a través de la placa controladora, en la que también están los drivers de los motores paso a paso. Esta placa además tiene una entrada de 12V para la alimentación de los motores y obtiene los 5V de alimentación del microcontrolador. La placa controladora dispone de 5 puertos híbridos que pueden ser usados para sensores y/o servomotores, cada puerto dispone de un cable de alimentación VDD, otro de tierra GND y otro de datos que puede ser usado como entrada o como salida. El diagrama de conexiones de la placa controladora está disponible en la *fig. 2.4.1.*

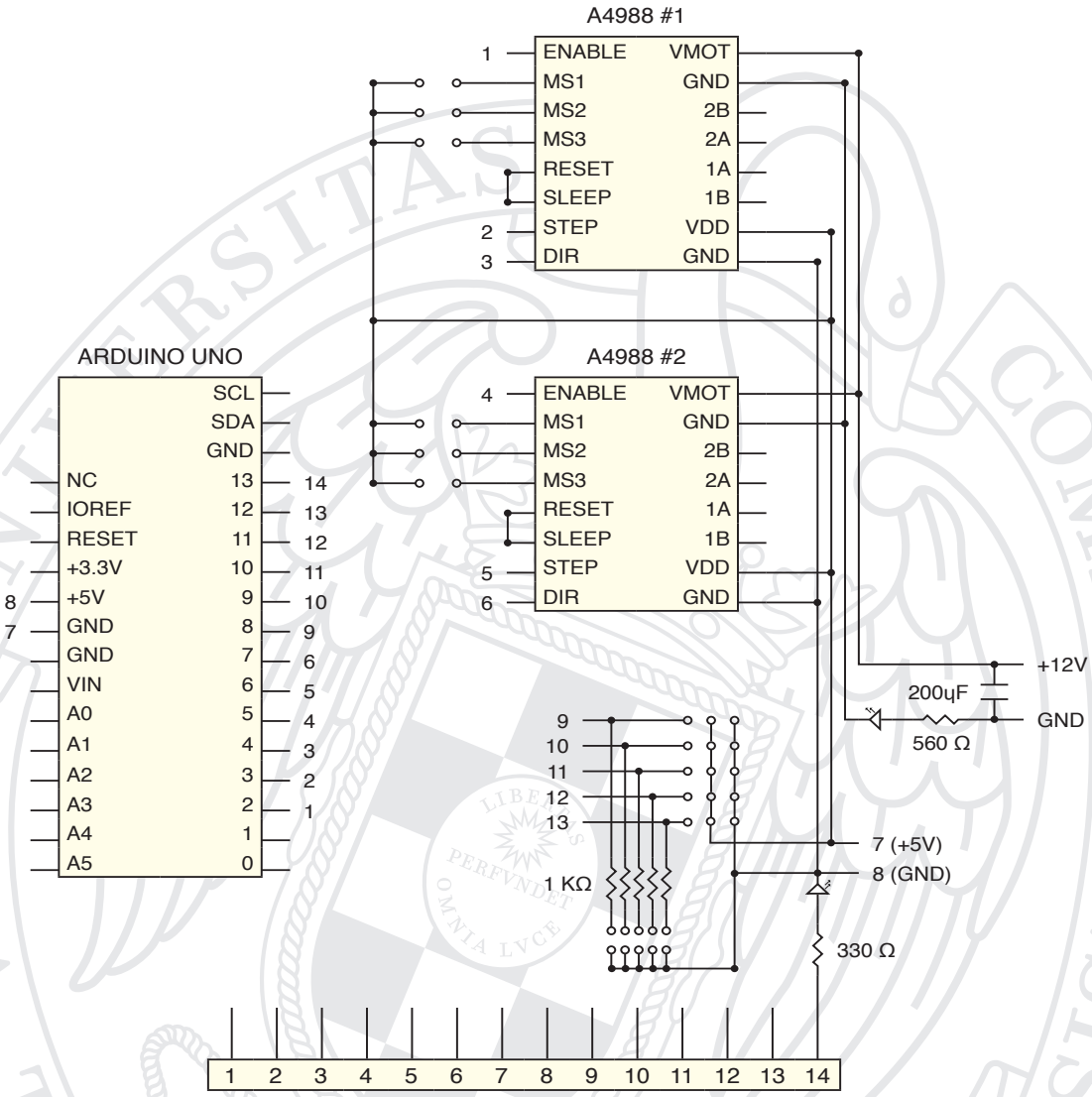

Figura 2.4.1: Diagrama de conexiones de la placa controladora

Antes de conectar los motores debemos calibrar los drivers para que entreguen la corriente necesaria al motor y que no resulte dañado. Para ello necesitamos conocer la corriente de funcionamiento para cada uno de nuestros motores, en nuestro caso tenemos un motor paso a paso del tipo NEMA-17 de 1,5A y otro del tipo NEMA-14 de 0,8A. Para calcular el Vref –voltaje de referencia– del driver aplicamos la siguiente fórmula:

$$Vref = Imáx * (8 * rs)$$

En nuestro caso, para el primer motor tenemos una Imáx –intensidad máxima– de 1,5A y para el segundo una Imáx de 0,8A. Ambas controladoras tienen una rs –resistencia de sensibilidad– de 0,1Ω por lo que este valor será constante en nuestros cáculos.

Sustituyendo los valores de Imáx y rs en la fórmula obtenemos los siguientes valores de Vref. Para el primer motor, con Imáx 1,5A:

Vref = 1,5A * (8 * 0,1Ω) = 1,2V

Para el segundo motor, con Imáx 0,8A:

Vref = 0,8A * (8 * 0,1Ω) = 0,64V

Una vez obtenidos los voltajes de referencia para el funcionamiento de los motores debemos comprobar en qué modo va a trabajar nuestro motor, pasos completos, medios pasos, etc. y adecuar el voltaje según indique el fabricante en la hoja de datos de nuestro motor paso a paso. Para nuestro primer motor, el fabricante incluye la *tabla 2.4.2.*

Full Step #	**Half Step** #	**1/4 Step** #	**1/8 Step** #	**1/16 Step** #	**Phase 1 Current**	**Phase 2 Current**
	1	1	1	1	100.00	0.00
				2	99.52	9.80
			2	3	98.08	19.51
				4	95.69	29.03
		2	3	5	92.39	38.27
				6	88.19	47.14
			4	7	83.15	55.56
				8	77.30	63.44
1	2	3	5	9	70.71	70.71

Tabla 2.4.2: Consumo máximo según el modo de trabajo del motor

Como nuestro primer motor está trabajando con pasos de 1/16 –los tres jumper de la placa controladora están cerrados– el Vref del motor es el 100% del Vref calculado, por lo que debemos configurar el driver con ese valor. Para el segundo motor, que trabaja con pasos de 1/8 –sólo los jumper MS1 y MS2 están cerrados– el Vref del motor es aproximadamente el 83% del Vref calculado lo que nos da un nuevo valor de Vref = 0,53V con el que debemos configurar el driver.

Por defecto el programa del microcontrolador está configurado para conectar el servomotor al puerto 10 del microcontrolador, el sensor óptico de final de carrera del eje Z al puerto 11 y el sensor óptico de final de carrera del eje Y al puerto 12. No obstante podemos configurar los números de los puertos de entrada/salida desde el fichero 'configuration.h' y después cargar el programa en el microcontrolador para aplicar los cambios.

2.5. Software

El microcontrolador recibe los comandos desde el ordenador y lleva a cabo las operaciones necesarias, una vez acaba de ejecutar el comando responde con un ACK para informar de que ha terminado y que está listo para ejecutar el siguiente comando. Además el microcontrolador almacena las posiciones de los ejes J, Y y Z para evitar inconsistencias y desde el ordenador se consulta el estado cuando se necesita.

El ordenador es el encargado de enviar las órdenes necesarias para la correcta colocación de los ejes y el manejo de la webcam.

La comunicación con el microcontrolador se realiza por medio de un puerto serie, que en Linux se realiza mediante un archivo. En nuestro caso está ubicado en '/dev/ttyUSB**$1**' donde **$1** toma un valor numérico comprendido en [0, ∞) y nos comunicamos leyendo y escribiendo como si se tratara de un fichero regular, aunque para la configuración del puerto (paridad, bits, control de flujo, velocidad, etc.) utilizamos la estructura termios –incluida en 'termios.h'– que permite establecer todos estos parámetros.

Por otro lado, la comunicación con la webcam se realiza a través de otro fichero, que en nuestro caso está ubicado en '/dev/video**$1**' donde **$1** toma un valor numérico comprendido en [0, ∞). En este caso la comunicación no se realiza mediante lecturas y escrituras, sino por medio de la librería V4L2 de Linux que permite el manejo de dispositivos de vídeo de manera sencilla. Con esta librería podemos configurar la resolución de la cámara, el formato de las imágenes y parámetros como el contraste, la saturación, la luminosidad, el brillo, etc. En nuestro caso, establecemos el formato de captura a MJPEG por lo que la cámara conectada debe dar soporte a ese formato.

2.6. Ejemplo de ejecución

Tras conectar el microcontrolador y la webcam al ordenador y comprobar que se crean los archivos correspondientes en el directorio '/dev' en nuestro caso '/dev/ttyUSB0' y '/dev/video0' respectivamente podemos inciar la aplicación con el comando '$ bin/chesspiecescanner /dev/ttyUSB0 /dev/video0'. Una vez iniciada nos aparecerá el menú principal de la aplicación *fig. 2.6.1.*

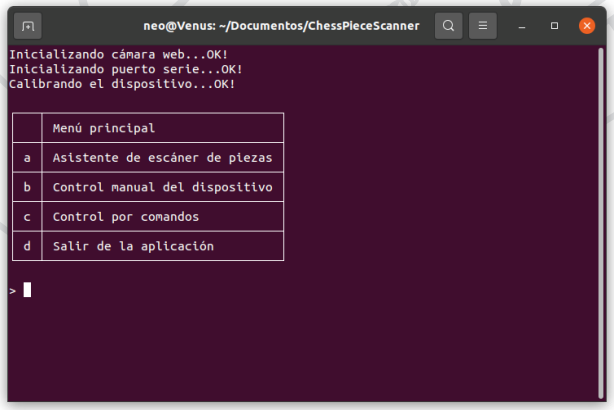

Figura 2.6.1: Menú principal de la aplicación chesspiecescanner

Desde este menú podemos seleccionar entre iniciar el 'Asistente de escáner de piezas', acceder al 'Control manual del dispositivo', acceder al 'Control por comandos' o 'Salir de la aplicación'.

Si seleccionamos la primera opción, pulsando la tecla 'a' y pulsando la tecla 'enter' a continuación, nos aparecerá el menú del asistente *fig. 2.6.2* desde donde podremos realizar fotografías de forma automática a una pieza. En este menú tenemos 3 opciones posibles, en la primera se realizarán fotografías moviendo únicamente los ejes J e Y del dispositivo para comprobar que las imágenes se capturan correctamente. En la segunda opción nos permite escanear 180 grados de la pieza, útil cuando las piezas son simétricas. En cuanto a la tercera opción escaneará la pieza completa. Finalmente disponemos de una cuarta opción que nos permite introducir los parámetros de escaneo manualmente.

Después de indicarle el modo de escaneo el programa nos preguntará el nombre de la pieza que va a escanear *fig. 2.6.3* y comenzará a capturar fotografías. Una vez acabe se volverá al menú principal *fig. 2.6.1* para que el usuario seleccione que desea hacer a continuación. Las imágenes se guardarán en la carpeta 'photos' dentro de un directorio con el nombre introducido anteriormente.

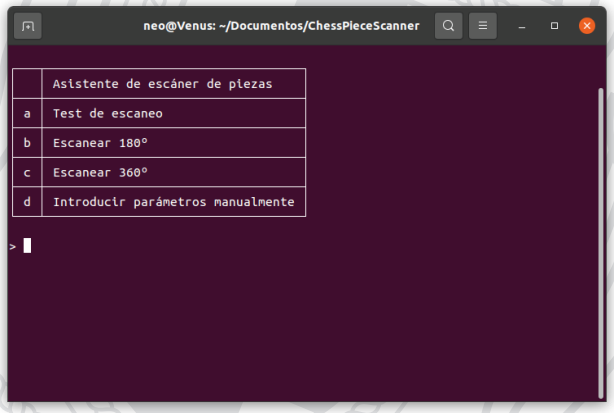

Figura 2.6.2: Menú del asistente de escáner de la aplicación chesspiecescanner

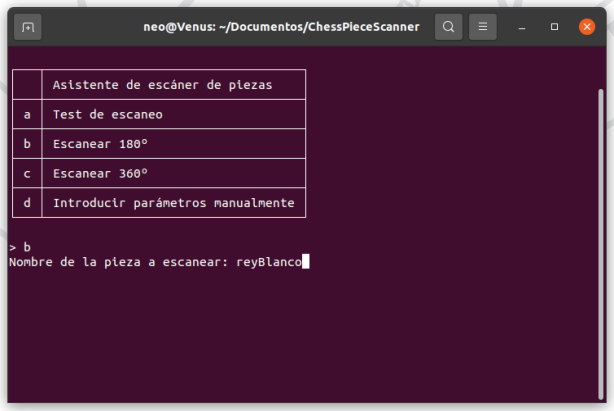

Figura 2.6.3: Solicitud del nombre de la pieza de la aplicación chesspiecescanner

Si seleccionamos la segunda opción el programa consultará al microcontrolador las posiciones de los ejes J, Y y Z y después nos mostrará una tabla con las opciones de control disponibles *fig. 2.6.4*. Si realizamos una fotografía con la letra 'P' se guardará en el directorio 'photos' con el nombre de 'preview.jpg'.

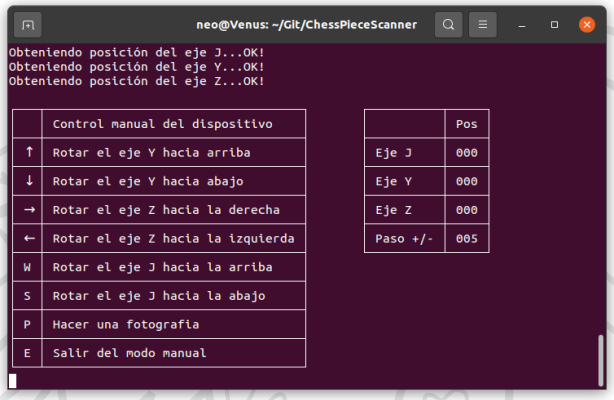

Figura 2.6.4: Menú de control manual de la aplicación chesspiecescanner

Al seleccionar la tercera opción el programa cargará una terminal que permite la comunicación por medio de comandos con el microcontrolador y mostrará el resultado de la ejecución *fig. 2.6.5*.

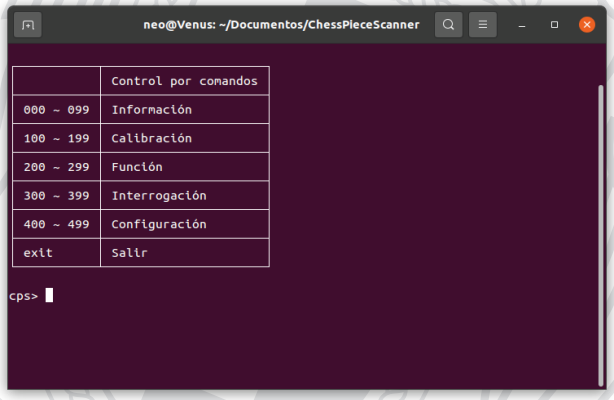

Figura 2.6.5: Terminal de la aplicación chesspiecescanner

La última opción finaliza la ejecución del programa desconectando el dispositivo y cerrando los archivos del microcontrolador y la webcam *fig. 2.6.6*.

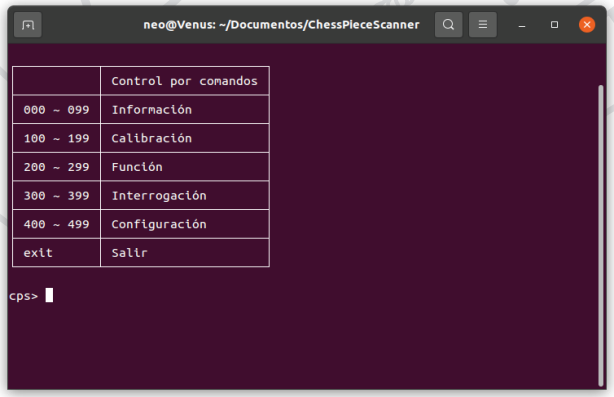

Figura 2.6.6: Cierre de la aplicación chesspiecescanner

2.7. Resultados

Tras realizar las fotografías de todas las piezas con fondo blanco y negro obtenemos un total de 13.824 imágenes de distintos ángulos que permiten inferir los detalles diferenciadores de cada una de las piezas. Véase la *fig. 2.7.1* dónde se muestran las 4 configuraciones utilizadas para realizar las fotografías.

Figura 2.7.1: Resultados de la ejecución de chesspiecescanner con distintos fondos

2.8. Configuración

Los parámetros utilizados para la toma de fotografías y posterior recorte son configurables por medio del archivo 'conf/images.conf' donde se encuentran todos los ángulos de los ejes J e Y desde los que el dispositivo debe realizar fotografías y el rectángulo de recorte para cada una. También podemos configurar si se deben recortar las imágenes cambiando el valor de 'crop_enabled' entre [0, 1].

Asimismo podemos configurar la webcam por medio del archivo 'conf/webcam.conf' donde podemos cambiar el brillo, el contraste, la saturación, la exposición y demás parámetros con los que se tomarán las fotografías.

La configuración de la webcam se aplica al iniciar la aplicación *fig. 2.6.1* por lo que si modificamos los valores debermos reiniciar la aplicación para que estos surjan efecto. Por otro lado, la configuración de la toma de fotografías se lee al iniciar el asistente de escaneo *fig. 2.6.2* por lo que si modificamos algún valor en la configuración se aplicará cuando entremos a este menú.