Escuela Politécnica Superior de Jaén

Grado en Ingeniería Electrónica Industrial

UNIVERSIDAD DE JAÉN
Escuela Politécnica Superior de Jaén

Trabajo Fin de Grado

SISTEMA DE ALINEACIÓN DE PCBS MEDIANTE EL USO DE VISIÓN POR COMPUTADOR

Alumno: Carlos Cañuelo Hernández

Tutor: Prof. D. Alejandro Sánchez García
Dpto: Ingeniería Electrónica y Automática

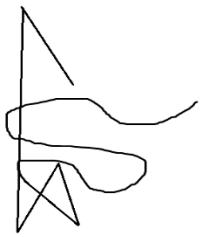
Septiembre, 2023

Universidad de Jaén
Escuela Politécnica Superior de Jaén
Departamento de Informática

Don Alejandro Sánchez García, tutor del Proyecto Fin de Carrera titulado: Sistema de alineación de PCBs mediante el uso de visión por computador, que presenta Carlos Cañuelo Hernández, autoriza su presentación para defensa y evaluación en la Escuela Politécnica Superior de Jaén.

Jaén, SEPTIEMBRE de 2023

El alumno: Los tutores:



Carlos Cañuelo Hernández Alejandro Sánchez García

RESUMEN

El presente proyecto tiene como finalidad, el desarrollo de una aplicación, que permita el alineamiento de una PCB, con la máquina encargada de realizar el ensamblaje de la misma, mediante visión por computador, concretamente, a través de un dispositivo de adquisición de imágenes, que permita detectar marcas fiduciarias en la PCB, para a partir de ellas, calcular la matriz de transformación, que determina la posición y orientación de la placa, y así, poder determinar la localización de todos los lugares, en los que hay que ensamblar componentes electrónicos en la placa.

Palabras clave: Visión por computador, Opencv, nube de puntos, robótica.

ABSTRACT

The purpose of this Project, is the development of an application, which allows the alignment of a PCB, with the machine in charge of assembling it, through computer visión, specifically, through an image acquisition device, that allows detecting fiduciary marks on the PCB, so that from them, compute the transformation matrix, which determines the position and orientation of the PCB, and thus, to be able to determine the location of all the places, in which the machine has to assemble electronic components on the board.

Keywords: computer vision, Opencv, point cloud, robotics.

Índice

Índice de figuras

1. INTRODUCCIÓN

Con el avance vertiginoso de la tecnología, cada año, los componentes electrónicos, que se ensamblan en una placa de circuito impreso (PCB), se vuelven de menor tamaño, y, por consiguiente, una PCB permite cada vez más componentes en ella, estos son dos factores que dificultan su instalación manual. Todo esto junto con la automatización de los procesos de fabricación, ha provocado que esta tarea, recaiga en las máquinas, capaces de realizar tareas repetitivas, de manera precisa, y sin la necesidad de intervención humana, concretamente, se hace referencia, a las máquinas de ensamblaje de PCB, encargadas de suministrar, e instalar, los diferentes componentes electrónicos, requeridos en una placa.

Figura 1.1 Máquina de ensamblaje de PCB de la empresa VEXOS

Sin embargo, una de las grandes dificultades que se presentan, en el proceso de montaje de placas de circuito impreso, es el alineamiento de la misma, con respecto al sistema de referencia de la máquina, encargada de ensamblar los diferentes componentes electrónicos en ella, esto se debe a que, cada PCB tendrá diferentes dimensiones, y no siempre se posicionará de manera precisa, en el mismo punto. Debido a que este es un proceso automático, y carece de un sistema de realimentación, que indique que el proceso se está realizando correctamente, una leve desviación, en el proceso de ensamblaje de componentes, podría provocar que se desestime la placa tratada, por una incorrecta instalación de sus componentes, lo cual, en una cadena de producción, provocaría pérdidas materiales, económicas, y retrasos. Por este motivo, es imprescindible que la máquina, realice un alineamiento de la PCB, con respecto al sistema de referencia de la máquina, para poder

proceder al ensamblaje de manera precisa y correcta, del conjunto de componentes, a instalar en la misma.

Figura 1.2 Máquina de ensamblaje de PCB de la empresa QIHE

Es aquí donde entra la visión por computador, que nos posibilita realizar el alineamiento necesario, ya que, por medio de un sistema de adquisición de imágenes, un algoritmo puede detectar, la localización y orientación de la PCB a ensamblar, y calcular la posición, en las que se encuentran el resto de componentes electrónicos de la placa, de manera, que se pueda proceder a su correcto ensamblaje, independientemente de la situación de la PCB.

1.1. Motivación

La motivación de este proyecto, viene dada, por la aspiración de realizar la automatización, de un proceso productivo, como es el ensamblaje de componentes electrónicos en PCBs, de manera económica y sencilla, utilizando conceptos de programación aprendidos a lo largo de la carrera. Además de su automatización, otro aspecto que aporta valor a este proyecto, es la implementación de un sistema, que permita el correcto funcionamiento de la máquina, independientemente de la situación posicional, en la que se encuentre la PCB, lo cual mejoraría el rendimiento de la máquina, facilitaría el proceso de ensamblaje, y dependería menos del factor humano a la hora de colocar la PCB.

1.2. Objetivos

El propósito del presente trabajo reside, en la programación de una máquina de ensamblaje, de dispositivos de montaje en superficie (SMD) sobre PCBs,

concretamente, en la fase de identificación de estas últimas, es decir, localizar una placa dentro del área de trabajo de la máquina, y realizar el alineamiento de la misma, de manera que, la máquina conozca las coordenadas, donde se instalarán los distintos componentes electrónicos en dicha placa, a partir de unas marcas fiduciarias, que servirán para identificar, la posición y orientación de la placa, y de los ficheros Gerber, que contienen las coordenadas en las que se instalarán los componentes.

Para ello, se desarrollará una aplicación de ordenador que, a través de una interfaz gráfica, permita al usuario realizar las siguientes funciones:

- Obtener una imagen que contenga una PCB, dentro del área de trabajo de la máquina.
- Seleccionar la marca fiduciaria, que se desea utilizar para el alineamiento de la PCB, y detectar aquellas, que se encuentran en la placa.
- Seleccionar y leer un archivo, que contenga las coordenadas de los componentes de la PCB, en su propio sistema de referencia.
- Establecer relaciones entre coordenadas, unas en el sistema de referencia máquina, y otras en el sistema de referencia de la PCB, para calcular la matriz de transformación y así, obtener la posición de todos los componentes en el sistema máquina.
- Visualizar dentro de la imagen obtenida, dónde se encuentran cada uno de los componentes de la PCB.
- Mover el cabezal de la máquina, en el que se encuentra el dispositivo de adquisición de imágenes, para poder desplazarse dentro del área de trabajo.

1.3. Estructura

La organización del presente documento, consta de 7 capítulos, comenzando el primero, con una introducción acerca de la cuestión a tratar, los objetivos marcados y la solución planteada para remediar dicho conflicto. En el segundo apartado, se trata la base teórica de la cuestión, y los conocimientos requeridos en ella. Seguidamente, el tercer apartado, consiste en la exposición de los elementos materiales, que constituyen, el conjunto de la situación y la solución adoptada. El

cuarto apartado estará compuesto por, la explicación de la interfaz gráfica del usuario y sus elementos, y la codificación detallada, del algoritmo encargado de llevar a la práctica la solución. Seguidamente, en el apartado cinco, se recopilarán y comentarán, el conjunto de resultados obtenidos, para las distintas pruebas realizadas, dentro de la variedad de escenarios planteados. En el sexto apartado, se propondrán mejoras para futuros proyectos. El último apartado, contendrá un anexo con el manual de uso. Y para finalizar, se presentará la bibliografía utilizada, para el desarrollo del trabajo.

2. MARCO TEÓRICO

La índole del problema reside, en que se trabaja en dos sistemas de referencia. Por una parte, se encuentra el sistema de referencia de la PCB, y, por otro lado, el sistema de referencia de la máquina. El primero, se trata de un sistema escalado en milésimas de milímetro, con origen de referencia, en una de las esquinas de la placa, mayoritariamente, en la esquina inferior izquierda. El segundo, está escalado en píxeles, al ser la unidad fundamental, en la que se conforma la imagen, y su origen se encuentra, en nuestro caso, en la esquina superior izquierda, sin embargo, este ha sido modificado, para situarse, como el primer sistema de referencia, en la esquina inferior izquierda de la imagen.

Para el objetivo planteado, es imprescindible poder saltar entre ambos sistemas, es decir, transformar las coordenadas de un punto concreto, en un sistema de referencia, al otro, para ello se hará uso, de la matriz de transformación. Este, es un método empleado en robótica, que permite, representar transformaciones geométricas, dentro de un espacio de tres dimensiones, posibilitando la manipulación, de la posición y orientación, en la que se encuentra los puntos en un sistema de coordenadas.

Su mecanismo consiste, en una matriz cuadrada, de dimensión 4, formada por dos submatrices, una de dimensión 3x3, encargada de la rotación, y representada por el carácter R, y otra de dimensión 3x1, encargada de la traslación, y representada por el carácter “T”. Se añadirá una fila de 4 columnas debajo de estas, para mantener su característica de matriz cuadrada y hacer las operaciones más sencillas. Con esto, se pueden representar, traslaciones, rotaciones y/o cambios de escala.

$$\begin{pmatrix} R_{11} & R_{12} & R_{13} & T_x \\ R_{21} & R_{22} & R_{23} & T_y \\ R_{31} & R_{32} & R_{33} & T_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Figura 2.1 Matriz de transformada homogénea

Para el caso de estudio, el robot no se moverá en el eje Z, sino que lo hará en el plano horizontal XY. Debido a esto, la matriz de transformación se ve reducida

para el caso de dos dimensiones. En la Figura 2.2, se ve el resultado de reducir la matriz a dos dimensiones, sustituyendo, además, los parámetros de rotación, por sus correspondientes valores trigonométricos, donde θ representa el ángulo de rotación.

$$\begin{pmatrix} \cos\theta & -\sin\theta & T_x \\ \sin\theta & \cos\theta & T_y \\ 0 & 0 & 1 \end{pmatrix}$$

Figura 2.2 Matriz de transformada homogénea en 2 dimensiones

A modo de ejemplo, se plantea una situación en dos dimensiones, con dos puntos en, A=(1, 1), y en B=(1, 2), a los que se les ha aplicado una traslación de 3 unidades en el eje x, y 2 unidades en el eje Y, además de una rotación de 10º. El planteamiento que se expone es como el que se muestra en la Figura 2.3.

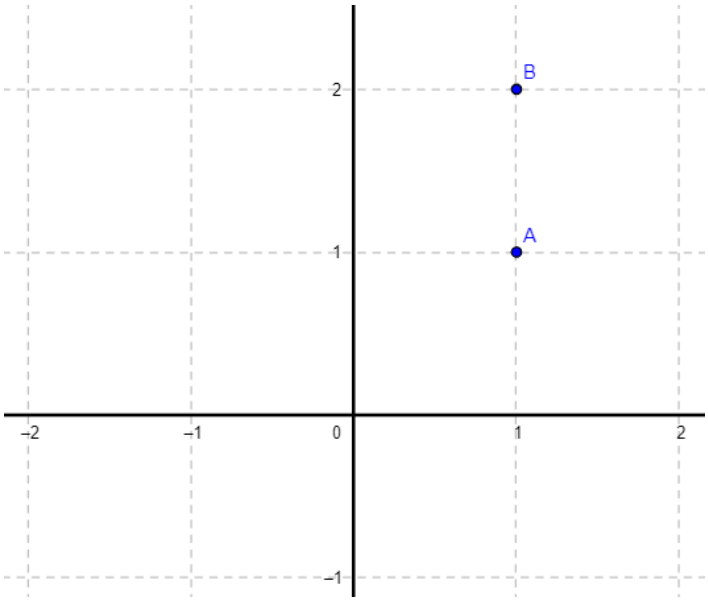


Figura 2.3 Situación de origen (Ejemplo)

Convirtiendo los 10º de la rotación a radianes, multiplicando por π/180º, se obtiene que el ángulo de rotación θ, son 0.1745 radianes. Realizando los senos y cosenos de dicho ángulo, se obtiene la siguiente matriz de transformación.

$$\begin{pmatrix} 0.9848 & -0.1736 & 3 \\ 0.1736 & 0.9848 & 2 \\ 0 & 0 & 1 \end{pmatrix}$$

Figura 2.4 Matriz de transformación (Ejemplo)

Aplicando esta matriz, a los dos puntos que se planteaban, al inicio del ejemplo, se obtendría la posición de estos, tras la traslación y rotación.

$$M * A = \begin{pmatrix} 0.9848 & -0.1736 & 3 \\ 0.1736 & 0.9848 & 2 \\ 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \rightarrow AA = (3.81, 3.16)$$

$$M * B = \begin{pmatrix} 0.9848 & -0.1736 & 3 \\ 0.1736 & 0.9848 & 2 \\ 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix} \rightarrow BB = (3.64, 4.14)$$

Figura 2.5 Cálculo de los puntos transformados (Ejemplo)

Representando, la nueva localización de ambos puntos, se obtendría la situación final mostrada en la Figura 2.6, en la que se ve que los puntos A y B, han sido trasladados 3 unidades, en el sentido positivo del eje Y, y 2 unidades en el sentido positivo del eje Y, además, tras haber sufrido dicha traslación, se les ha aplicado, una rotación de 10º. Estos nuevos puntos se han representado como los puntos AA y BB.

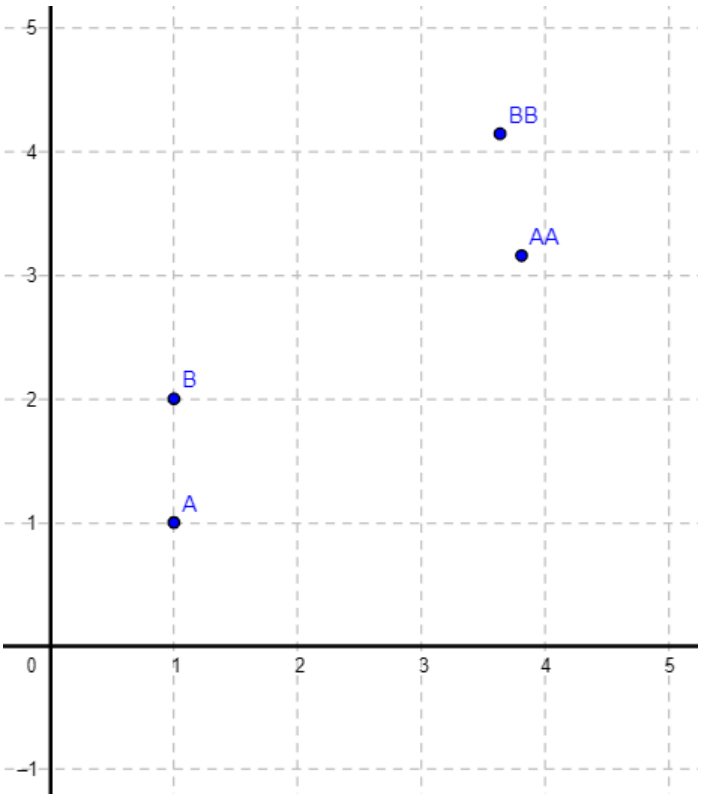


Figura 2.6 Situación final (Ejemplo)

Con este ejemplo, queda de manifiesto, la funcionalidad de la aplicación a desarrollar, mediante la cual, sabiendo la posición del punto A, que representa las

coordenadas proporcionadas por el fichero Gerber, y del punto AA, que representa las coordenadas del punto A, tras sufrir una transformación, se puede calcular la transformación que ha llevado el punto A, al punto AA, mediante el uso de la matriz de transformación homogénea. Así pues, habiendo obtenido la transformación sufrida, y conociendo la posición del punto B, dada de nuevo por el fichero Gerber, se puede aplicar dicha matriz de transformación, y obtener la posición final en el punto BB. En este ejemplo los puntos A y B, vienen dados por el fichero Gerber, y el punto AA, actúa como marca fiduciaria.

3. ELEMENTOS HARDWARE

Antes de entrar en la parte software del proyecto, se justificarán los elementos materiales, o hardware, utilizados en la realización del trabajo, así como una explicación, de en qué consisten. Se tratan de, el dispositivo de visón, utilizado para obtener las imágenes necesarias, para realizar el alineamiento, el robot y sus elementos, encargado de desplazar el dispositivo anterior, a lo largo del área de trabajo, y comunicarse con el usuario a través de la GUI, recibiendo las instrucciones que este le mande, y las PCBs utilizadas, que se colocarán dentro del área de trabajo del robot, para que el dispositivo de visión las detecte, y el programa se encargue de realizar el alineamiento de las mismas.

3.1. Dispositivo de adquisición de imágenes

Para la obtención de imágenes, se requerirá de un elemento, que visualice una zona dentro del área de trabajo del robot, y pueda observar la situación posicional, en la que se encuentra la PCB a alinear. Para esta tarea se barajaron dos opciones, primeramente, se planteó con el uso, de un microcontrolador ESP32, con una pequeña cámara integrada, y más adelante, se pasó al uso de una webcam.

Para el primero, se desarrolló un programa en Arduino, en el que, una vez se definían los pines del microcontrolador, y se habían ajustado los parámetros, para el modelo de la cámara (OV2640), se hacía que el microcontrolador, actuase como servidor WiFi, y el portátil actuaba como cliente, conectándose a la señal WiFi generada por el ESP32, para así, obtener la captura de imagen realizada, a través de la cámara integrada en el microcontrolador.

Figura 3.1 Microcontrolador ESP32 con cámara

La segunda opción, consiste en una webcam conectada vía USB al portátil. Con este método no es necesario el uso de Arduino, sino que directamente, en el programa realizado en Python, se realiza la orden de captura de imagen, y la guarda para luego ser utilizada en la interfaz gráfica, también generada desde este lenguaje.

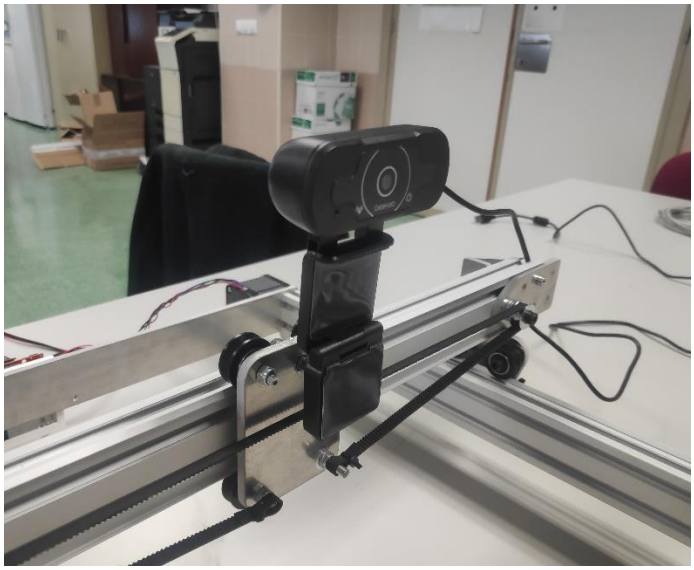


Figura 3.2 Webcam

3.2. Robot

El robot, es la máquina encargada de, una vez se sitúa una PCB dentro de su zona de trabajo, es capaz de moverse dentro de dicha área, para situar al dispositivo de visión en una posición adecuada, para observar la placa. Para ello, el que se ha

utilizado para el proyecto, es uno creado en el Grupo de Robótica, Automática y Visión por Computador (GRAV) de la Universidad de Jaén, a partir de perfiles de aluminio, y componentes proporcionados por FASTEN SISTEMAS. Su mecanismo, es el mismo que el de un robot cartesiano de dos ejes en el plano horizontal, en el que el elemento encargado de capturar imágenes, se encuentra en el cabezal móvil, para moverse dentro del área de trabajo y realizar las capturas necesarias. El movimiento del robot se realiza mediante 3 motores paso a paso, 1 encargado del movimiento del cabezal en el eje X, en el perfil en el que se encuentra la cámara, y 2 que realizan el movimiento en el eje Y.

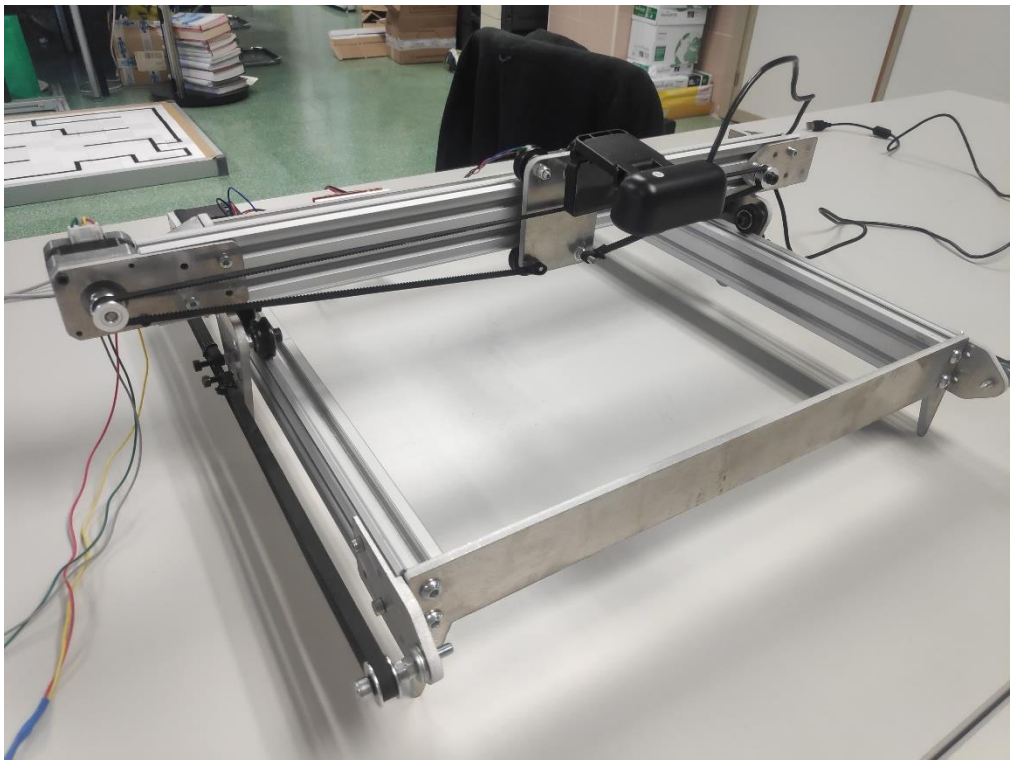


Figura 3.3 Robot

Para mandar las instrucciones de movimiento al robot, se utilizará una tarjeta CNC Shield, unida a una placa de Arduino, encargada de recibir las instrucciones en código GRBL, enviadas desde la interfaz gráfica. Gracias a los tres drivers incorporados, en la tarjeta CNC Shield, se podrán mandar las instrucciones, a sus correspondientes tres motores paso a paso, y los dos cables (rojo y negro), que se ven abajo a la izquierda de la Figura 3.4, proporcionarán la alimentación eléctrica. La comunicación entre la interfaz y la máquina se realizará mediante GRBL, y se explicará detalladamente en el apartado 4.3.

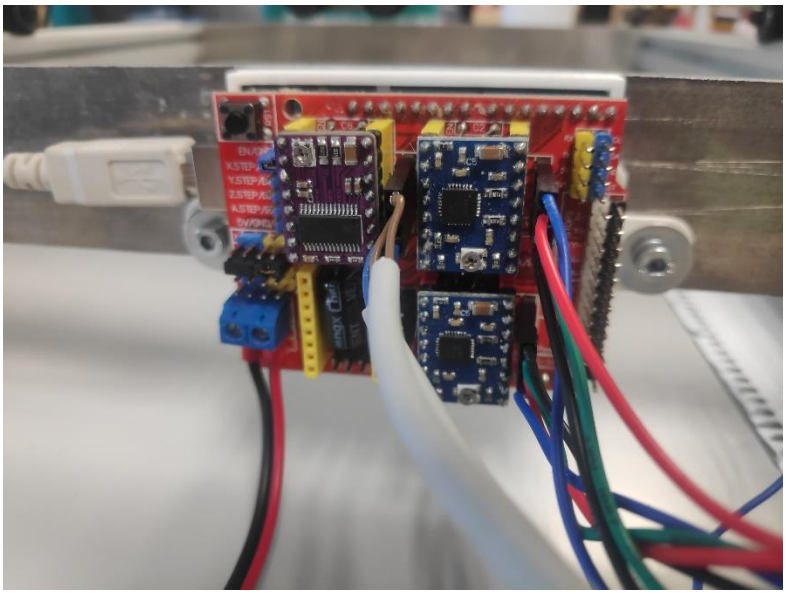


Figura 3.4 Tarjeta CNC Shield

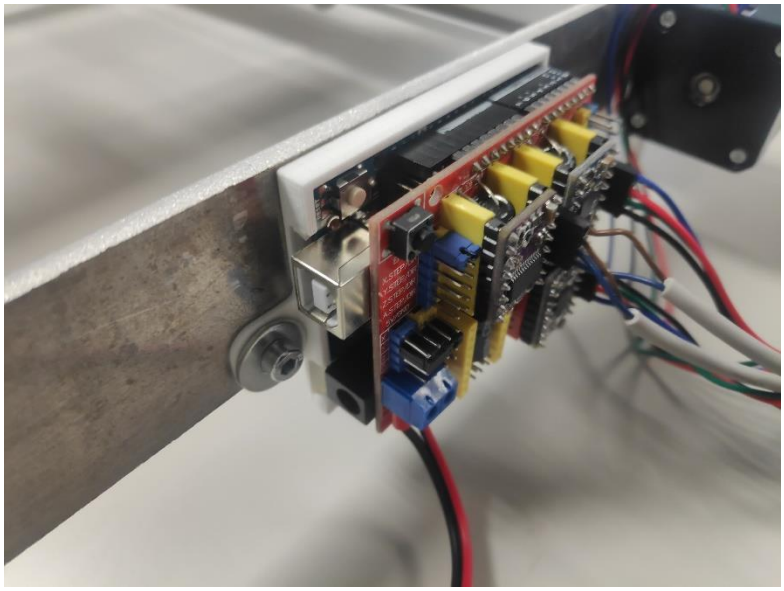


Figura 3.5 Tarjeta CNC Shield + Tarjeta Arduino

3.3. Placas de circuito impreso

Una placa de circuito impreso, o PCB, por sus siglas en inglés, printed circuit board, es un circuito electrónico, en el que sus componentes y elementos conductores, están situados en una estructura metálica con forma de tarjeta. La PCB está formada por varias capas, unas compuestas por cobre conductor, y otras por material no conductor. En el proceso de fabricación de una PCB, las capas conductoras se graban, dejando trazas de cobre al descubierto, para conectar los

componentes electrónicos. Estos componentes se añadirán en las capas externas, una vez se ha grabado y laminado el conjunto de las capas que conforman la placa.

Además, a cada diseño de PCB, le corresponde su respectivo fichero Gerber. Estos archivos son documentos vectoriales, que contienen descripciones acerca de, las conexiones eléctricas, los parámetros de su configuración, cuáles son las coordenadas en un plano XY de sus componentes, las pistas y vías, además de los pasos para su fabricación. Tras haber realizado el diseño de una PCB, antes de proceder a su fabricación, será imprescindible la generación de estos ficheros, ya que los fabricantes hacen uso de estos archivos para elaborar el grabado de las placas. Para desarrollar dichos ficheros, se hará uso de herramientas de fabricación asistida por ordenador (CAM), la cual, no tiene por qué ser el mismo software, utilizado para el diseño de la misma PCB.

Para la realización de las pruebas requeridas, durante el desarrollo del proyecto, se hizo uso de dos modelos de PCBs, facilitadas por la empresa ISR, junto con sus respectivos ficheros Gerber. A continuación, se muestra en la Figura 3.6 y en la Figura 3.7 las dos placas utilizadas.

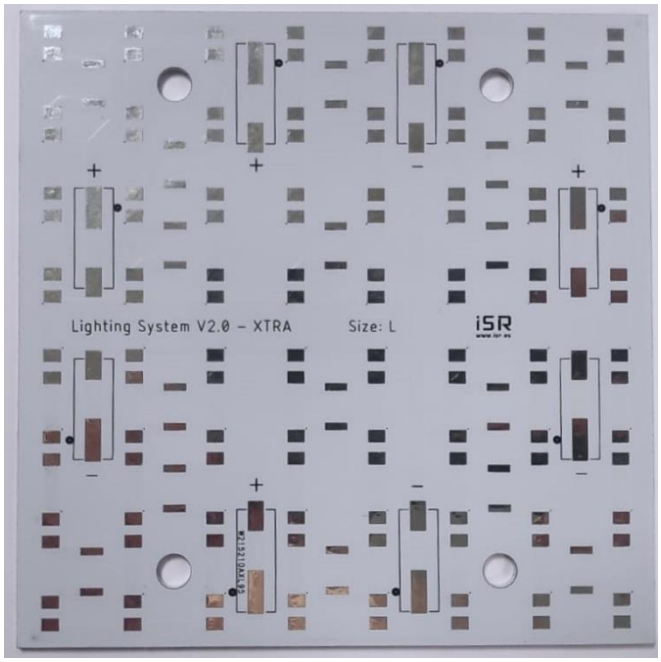


Figura 3.6 PCB backlight

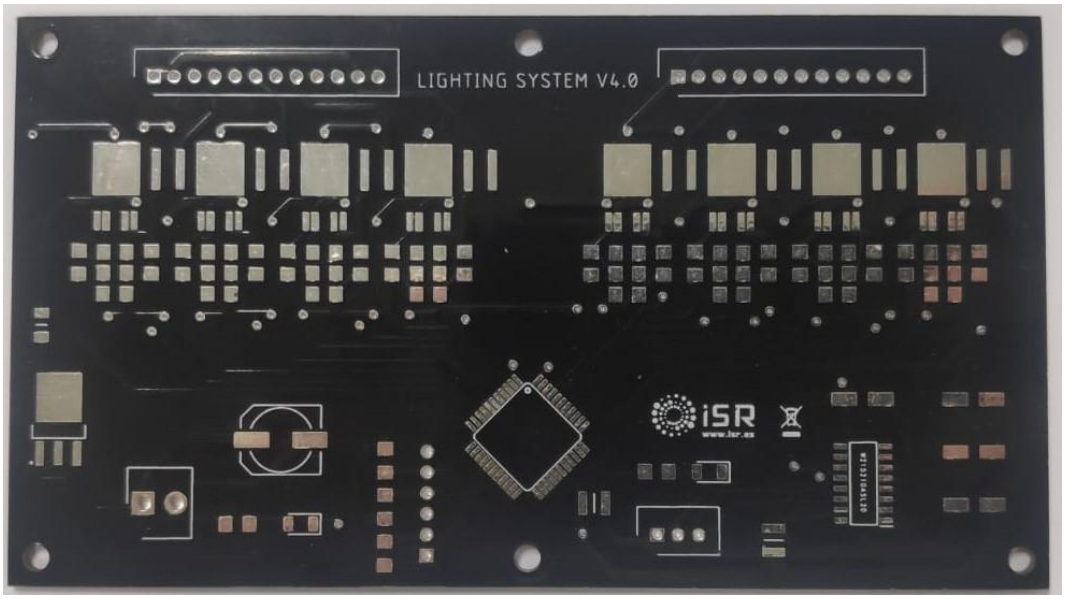


Figura 3.7 PCB control

4. Programación

Para la programación de la interfaz gráfica y los algoritmos necesarios, se plantearon dos opciones de lenguaje, Java y Python. El primero es un lenguaje compilado, mientras que el segundo es interpretado, esto hace que Java sea más rápido en tiempo de ejecución y fácil de depurar. Por otro lado, Python es más fácil de leer, y cuenta con una amplia gama de librerías, dedicadas a la creación de una interfaz gráfica de usuario (GUI). Es por esto que finalmente, se decidió utilizar Python como lenguaje de programación, a través del editor de texto Sublime Text 3.

Tras la instalación de Python y de Sublime Text 3, se deberá configurar este último, indicándole que se trabajará como “build system”, el lenguaje Python. Además, el archivo creado, para la programación de la aplicación, deberá incluir la extensión “.py” al final del nombre con el que se guarde.

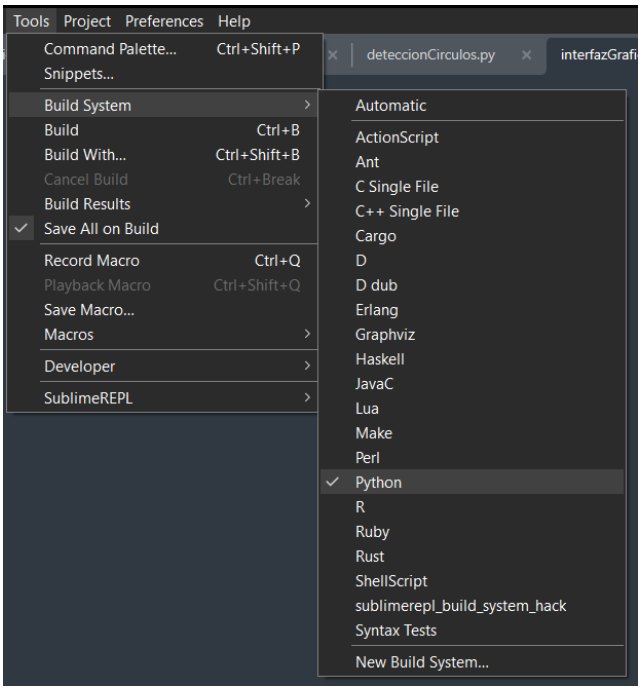


Figura 4.1 Selección del build system para la programación

Una vez instalado, y configurado, el editor de texto, se podrá comenzar a con la programación del código, encargado de realizar la interfaz gráfica, y los algoritmos requeridos, para cumplir la función de alineamiento de PCBs.

4.1. Interfaz gráfica

El siguiente paso, fue la elección de la librería encargada, de la implementación de la GUI. De entre todas las opciones, Tkinter fue la opción escogida, debido a que es una librería de código abierto, es conocida por su sintaxis sencilla y tener un fácil manejo, contiene todos los elementos necesarios para la interfaz deseada, y viene preinstalada en Python. Para su instalación, se deberá abrir la ventana de comandos de Windows, o CMD, e introducir el comando “pip install tk”.

De esta librería, se importarán, además, la clase “filedialog”, que permitirá mostrar cuadros de diálogo para abrir y guardar archivos en la interfaz, la clase “messagebox”, que permite mostrar cuadros de mensaje emergente, las clases “ImageTk” e “Image” del módulo “PIL” (Python Imaging Library), el cual es una biblioteca para trabajar con imágenes en el lenguaje de Python. Con “ImageTk” se permite mostrar imágenes en la interfaz de usuario, y con “Image” se pueden manipular las imágenes, para poder tratarlas. El módulo “c2v”, que es la biblioteca de OpenCV, realizará el procesamiento de las imágenes. Con el módulo “numpy” se permite realizar cálculos numéricos en Python, y también se utiliza para el procesamiento de imágenes. Y, por último, el módulo “serial”, se utilizará, para permitir la comunicación entre dispositivos serie, en este caso, con Arduino para mandar las instrucciones de movimiento de la máquina.

Ya con las librerías importada en el programa, se procederá al primer paso para generar la GUI, el cual consiste en generar la ventana principal (raíz). En esta, se añadirán los distintos elementos (widgets), que servirán para que el usuario, pueda mandar órdenes al programa, o modificar parámetros, en función de su necesidad. A continuación, se explicará la funcionalidad de cada uno de estos widgets, y cómo debe actuar el usuario sobre ellos.

4.1.1. Capturar / Volver a emisión

Para la toma de imágenes, al pulsar el botón “Capturar”, primeramente, se comprueba que haya una cámara conectada, en función de si está seleccionada la opción de “Webcam” o la de “ESP32”, y una vez se ha confirmado esto, realiza la captura de imagen, la cual, tras comprobar que se ha realizado correctamente, la guarda, con el nombre “captured_photo.jpg”, en la carpeta en la que está guardado

el proyecto, y la redimensiona al tamaño necesario, para que se muestre en el espacio reservado en la GUI, ya que cada cámara tendrá una dimensión diferente, a la hora de realizar capturas, y en la interfaz interesa mostrar las imágenes, siempre con el mismo tamaño, para no descuadrar la estructura de la GUI.

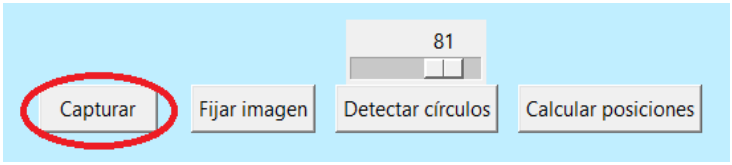


Figura 4.2 Botón “Capturar”

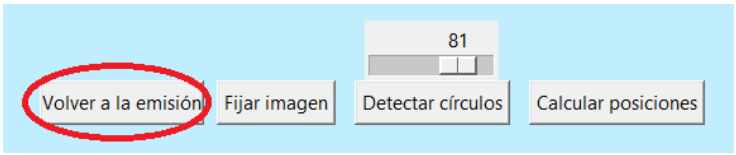


Figura 4.3 Botón “Volver a la emisión”

4.1.2. Fijar imagen

Tras obtener la imagen deseada de la PCB, a través del botón de captura de imagen, mencionado anteriormente, se requerirá fijar dicha imagen, para poder trabajar con ella, y poder realizar el alineamiento. Para ello, se pulsará el botón “Fijar imagen” situado a la derecha de los anteriores. Tras esto, no se apreciará ningún cambio a simple vista, sin embargo, se habrá habilitado la opción de seleccionar un área, dentro de la imagen, para determinar la marca fiduciaria a utilizar.

Esto se hace para que, en caso de que la imagen tomada, no cumpla los requisitos deseados por el usuario, independientemente de cuáles sean, se pueda realizar una nueva captura de imagen, pulsando el botón “Volver a la emisión”, y nuevamente el botón “Captura”. Así, se podrán realizar las capturas necesarias, hasta obtener una que satisfaga las necesidades del usuario.

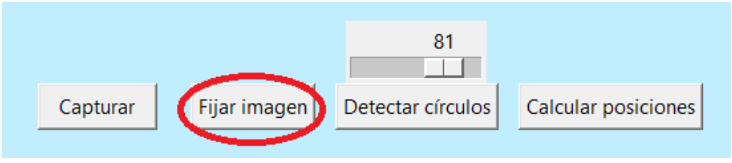


Figura 4.4 Botón “Fijar imagen”

4.1.3. Detectar círculos

Al momento de pulsar el botón “Detectar círculos”, ocurrirán tres eventos de manera simultánea. Por un lado, se mostrará, en la parte inferior izquierda de la imagen visualizada, una nube de puntos azules, que representan la estructura, en la que se distribuyen los distintos componentes de la PCB, estas vendrán dadas, por las coordenadas del fichero Gerber. Por otro lado, se señalizará sobre la imagen mostrada en la GUI, las zonas en las que se ha detectado, la marca fiduciaria seleccionada, y las similares, por medio de círculos verdes que rodeen dichas áreas. Por último, en el cuadro de texto situado a la izquierda de la GUI, aparecerán las coordenadas, de cada marca encontrada en la imagen, medidas en píxeles, y con origen de referencia, en la esquina inferior izquierda de la imagen. Haciendo clic en cada una de estas coordenadas, aparecerá en la imagen, un círculo rojo, en el centro del círculo verde, correspondiente a dicha marca, para poder reconocer, a cuál corresponde cada coordenada. En la Figura 4.6 se muestra, como, una vez detectadas las marcas fiduciarias, se selecciona una de ellas en el cuadro de texto, y se marca dicha coordenada en la imagen, con un punto rojo, para así, saber a cuál hace referencia.

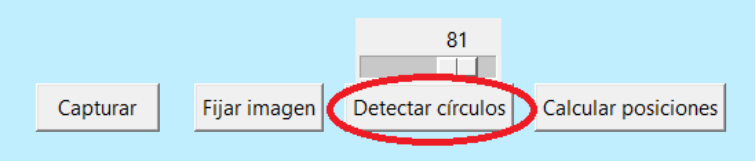


Figura 4.5 Botón “Detectar círculos”

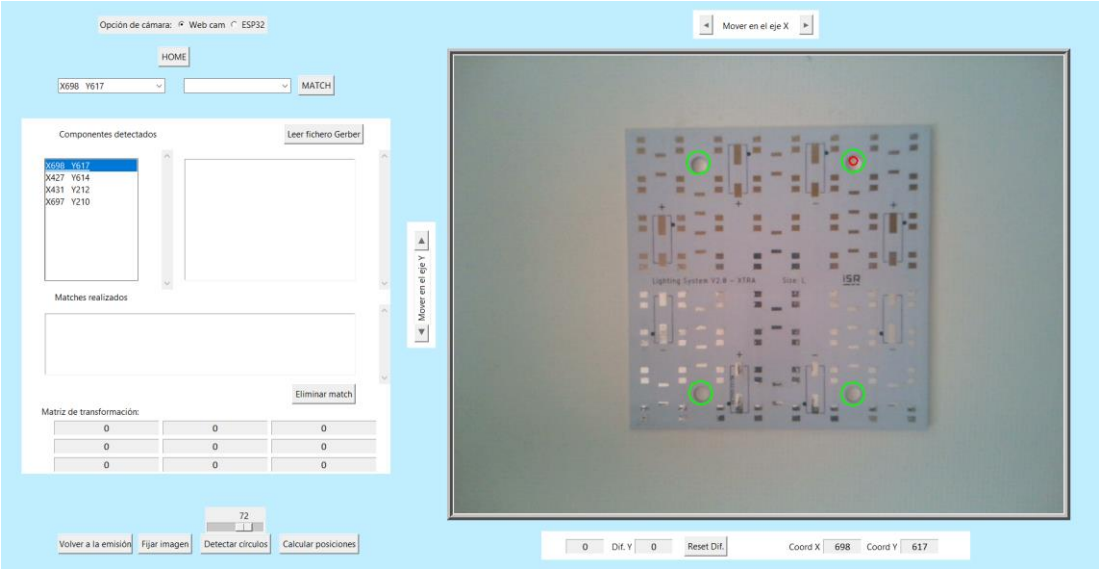


Figura 4.6 Marca fiduciaria seleccionada y marcada en la interfaz

En esta última imagen, se puede apreciar que no se ve la nube de puntos que representan la estructura de la PCB, esto es debido, a que se ha pulsado el botón “Detectar círculos” sin haber leído el fichero Gerber. Para que se muestren estos puntos, una vez se ha leído el fichero Gerber, se pulsará de nuevo el botón “Detectar círculos” y estos aparecerán.

4.1.4. Leer fichero Gerber

Para la obtención de las coordenadas, de los componentes de una PCB, se pulsará el botón “Leer fichero Gerber”, este abrirá una pequeña interfaz, gracias a la clase “filedialog”, para seleccionar el archivo que se desee leer. Al seleccionar el fichero, en el cuadro de texto, situado debajo de este botón, aparecerán las coordenadas, de los componentes de dicha tarjeta, en milésimas de milímetros, respecto al sistema de referencia de la propia placa.

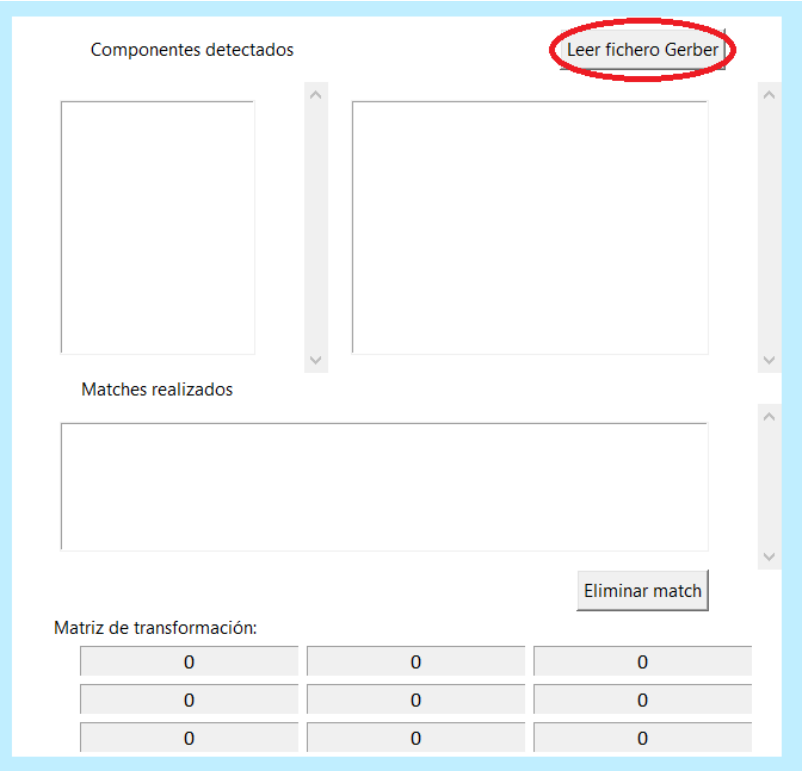


Figura 4.7 Botón "Leer fichero Gerber"

Como se mencionó en el apartado anterior, al pulsar el botón "Detectar círculos", aparecerá una nube de puntos, que representa la distribución de la placa a alinear, y también se mencionó, que esto ocurrirá, siempre y cuando se haya leído el respectivo fichero Gerber. A continuación, se mostrarán dos ejemplos, uno para cada placa, en los que se representa dicha nube de puntos.

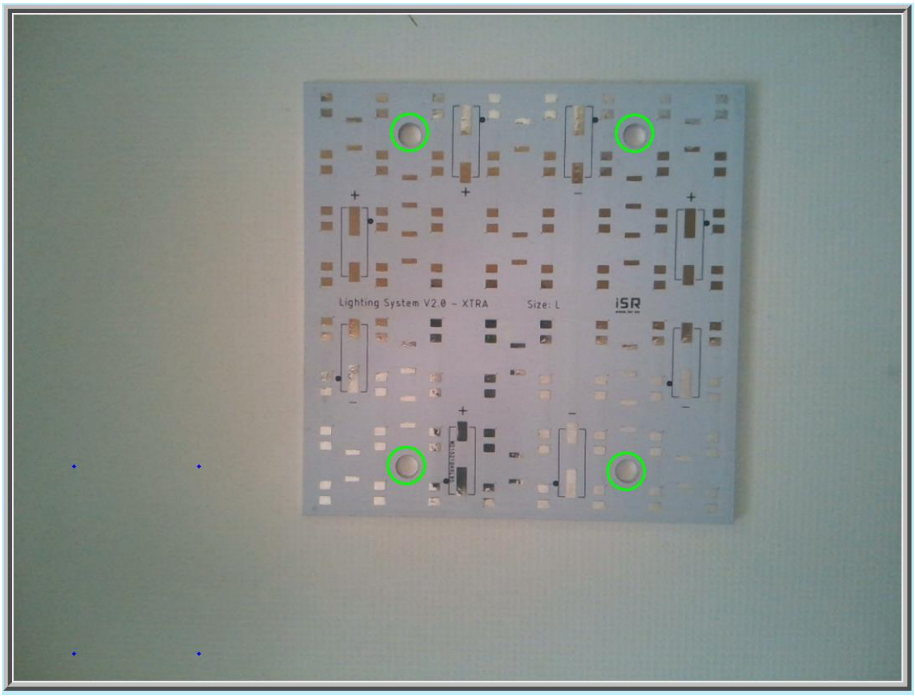


Figura 4.8 Nube de puntos para la PCB backlight

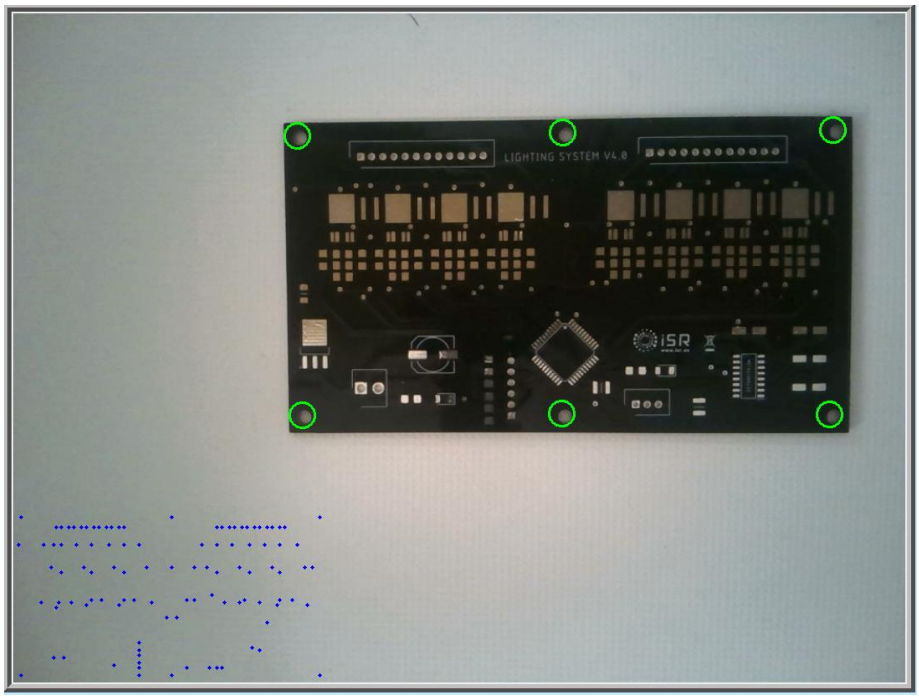


Figura 4.9 Nube de puntos para la PCB control

4.1.5. MATCH

En la parte superior izquierda de la GUI, justo debajo del botón “HOME”, se encuentran, dos desplegables y el botón “MATCH”. Habiendo pulsado los dos

anteriores botones, “Detectar círculos” y “Leer fichero Gerber”, no sólo se habrán escrito las coordenadas en sus respectivos cuadros de texto, sino que también se encontrarán en los dos desplegables mencionados. El primer desplegable contendrá, las coordenadas obtenidas al pulsar “Detectar círculos”, y el segundo, las leídas por el fichero Gerbe.

Con el botón “MATCH”, se establece una relación entre dos coordenadas, esta relación, viene dada por las coordenadas que se encuentren seleccionadas, en cada desplegable, y, al formarse dicha unión, se guardará en el cuadro de texto “Matches realizados”. Gracias a estas relaciones de coordenadas, el programa podrá calcular más adelante, la matriz de transformación necesaria, para calcular las coordenadas, del resto de componentes de la PCB de la imagen. Serán necesarios 4 matches, para el cálculo del resto de posiciones, y es imprescindible, que la disposición de estos 4 puntos, formen un cuadrado o un rectángulo.

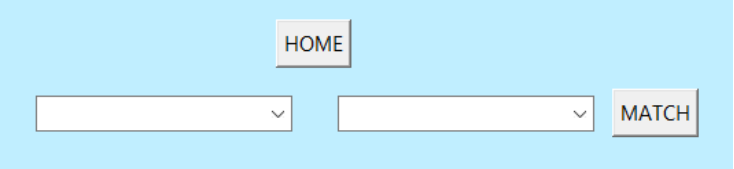


Figura 4.10 Botón “MATCH”

4.1.6. Eliminar match

En caso de producirse una equivocación, en el momento de realizar un match, se ha considerado el botón “Eliminar match”. Para su uso, será necesario hacer clic, dentro del cuadro de texto “Matches realizados”, en la relación de coordenadas que se desee eliminar. Una vez seleccionada, se pulsa este botón, y desaparecerá del cuadro de texto, para no tenerse en cuenta en posteriores cálculos.

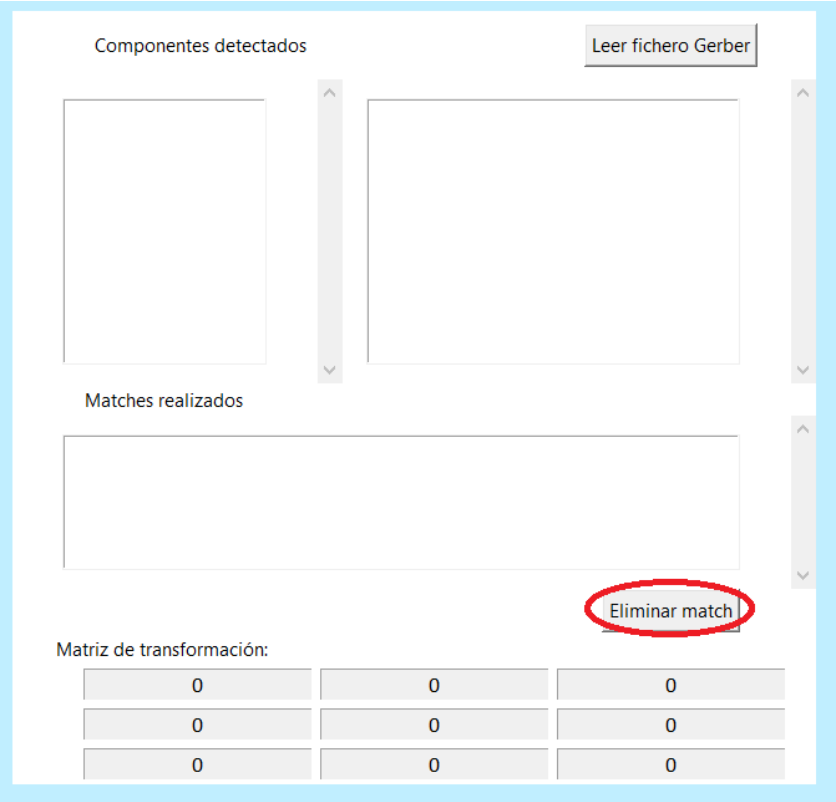


Figura 4.11 Botón "Eliminar match"

4.1.7. Calcular posiciones

Ya con los 4 matches realizados, se procede al cálculo de la matriz de transformación. Pulsando el botón "Calcular posiciones", se realizarán las operaciones necesarias y, una vez finalizadas, se podrá visualizar cada una de las nuevas localizaciones obtenidas. Tras haber realizado los cálculos pertinentes, aparecerá una ventana emergente, indicando que el procedimiento, se ha realizado correctamente, y, por tanto, se estarán visualizando en la imagen, todas las posiciones calculadas, mediante círculos azules. Además, al pulsar sobre una de las coordenadas, que contiene el cuadro de texto del fichero Gerber, se podrá observar que, en la imagen, se dibujará un punto rojo, en el lugar en el que dicha coordenada, se encuentra, para así saber a cuál se está refiriendo, y el robot, se desplazará a dicha posición.

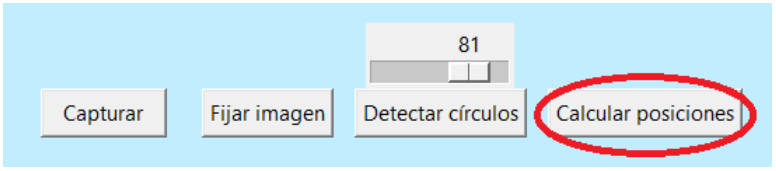


Figura 4.12 Botón "Calcular posiciones"

4.1.8. HOME

Para poder realizar movimientos con el robot, será necesario, al inicio del proceso, que se establezca la posición de "home", de manera que el robot, se oriente a partir de ella, y pueda moverse dentro del área de trabajo. A través de este botón, el programa envía a la máquina, la instrucción necesaria, para que se establezca dicha posición, en base a la localización actual del robot, y, una vez hecho esto, habilite el resto de botones de la interfaz, que permiten el movimiento del mismo. Además, este botón enviará la orden, de que el movimiento del robot sea relativo, es decir, que, si se envía la orden "X2", se mueva 2 unidades en el eje X, en lugar de moverse a la posición X=2.

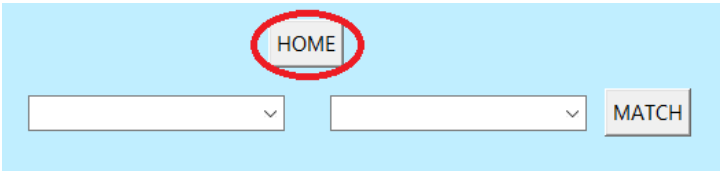


Figura 4.13 Botón "HOME"

4.1.9. Botones de movimiento de la máquina

Tras haber establecido la posición de home del robot, y haber indicado que el movimiento será relativo, los botones de movimiento, dejarán de estar deshabilitados. A través de estos botones, se podrá mover el robot una cantidad predefinida en el eje X (en ambos sentidos) y en el eje Y (en ambos sentidos), si alguno de estos botones es pulsado, se podrá comprobar, dentro de la GUI, el desplazamiento producido en cada eje, por medio de los cuadros de texto "Dif. X" y "Dif. Y", situados debajo de la imagen visualizada en la interfaz.

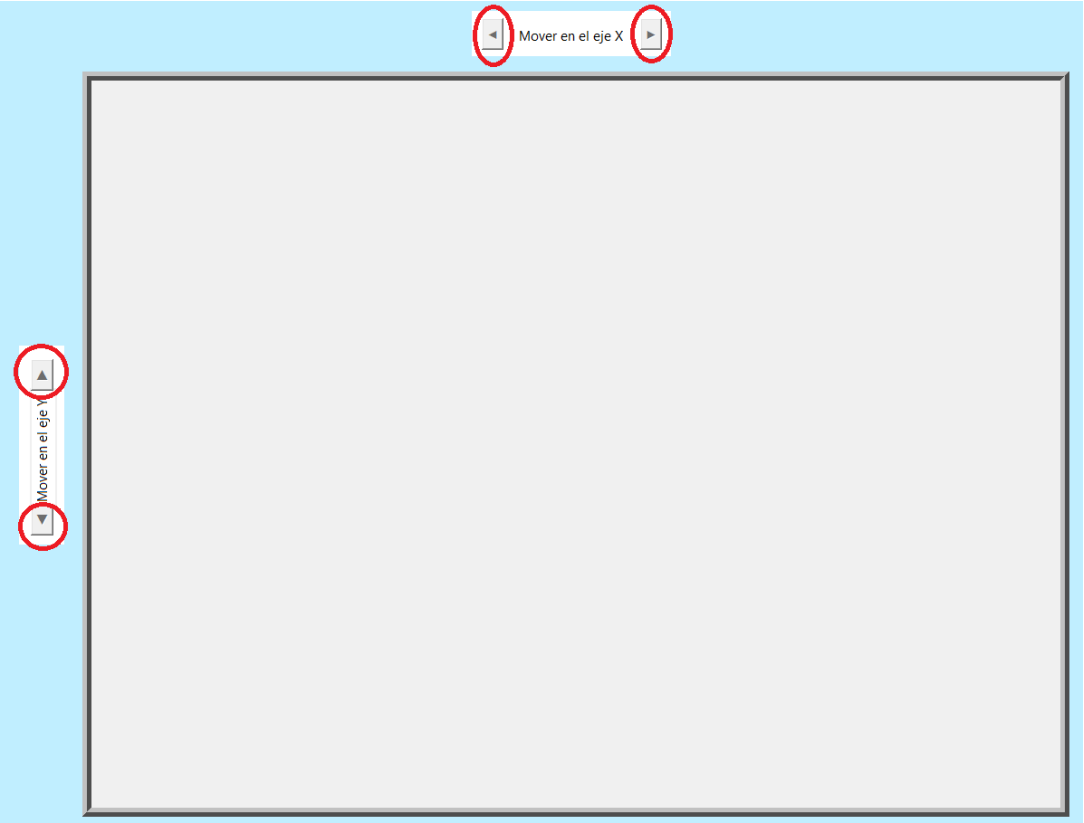


Figura 4.14 Botones de movimiento

4.1.10. Reset Dif.

Como se mencionó en el punto anterior, se podrá mover el robot, una cantidad determinada en ambos ejes, y, este desplazamiento se guardará en los cuadros “Dif. X” y “Dif. Y”. No obstante, para el cálculo de la matriz de transformación, es necesario que, la primera imagen obtenida de la PCB, tenga ambos valores a cero, esto es debido a que estos parámetros se utilizarán, para detectar el movimiento realizado, en caso de requerir de varias capturas de imagen, para visualizar todas las marcas fiduciarias necesarias en la PCB. Por lo que, al pulsar este botón, los valores de los desplazamientos en X e Y se reestablecerán a cero.

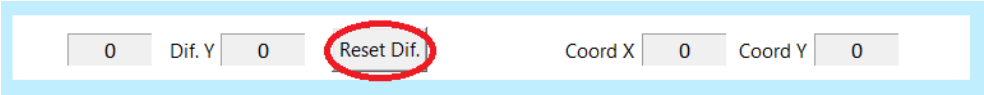


Figura 4.15 Botón “Reset Dif.”

4.2. Cáclculo de las coordenadas en el sistema máquina

En el apartado previo, se ha mencionado la funcionalidad, de cada elemento en la interfaz gráfica, y, por tanto, se ha comentado resumidamente, la forma de obtener las coordenadas del sistema PCB, en el sistema máquina. No obstante, a continuación, se profundizará en dicho proceso, explicando más detalladamente, la manera en la que se ha realizado.

4.2.1. Obtención de las coordenadas de las marcas fiduciarias de la PCB en el sistema máquina

Una vez la imagen es visualizada en la GUI, se procede a seleccionar la marca fiduciaria, la cual se desea utilizar, que se corresponden, con los taladros de mayor diámetro de la placa. Para ello, manteniendo el clic izquierdo del ratón pulsado, y arrastrando a lo largo de la imagen, se generará un recorte del área escogida, que será utilizada como plantilla, y será buscada en el resto de la imagen, de manera que, si se encuentra otra área de la imagen, cuyos píxeles guarden similitud con la plantilla, se contabilizará como marca fiduciaria encontrada.

El grado de similitud se medirá en tanto por ciento, siendo 81% el valor prestablecido, pudiendo modificarse en la GUI, por medio de un regulador, o slider, situado encima del botón “Detectar círculos”. Si se necesitase reducir el porcentaje de similitud, a fin de detectar todas las marcas fiduciarias situadas en una PCB, podrían aparecer marcadas áreas erróneas, que el programa detecta como zonas de interés, pero que no lo son. Dichas áreas no generarían conflicto, ya que es el usuario el encargado de seleccionar, qué marcas desea almacenar.

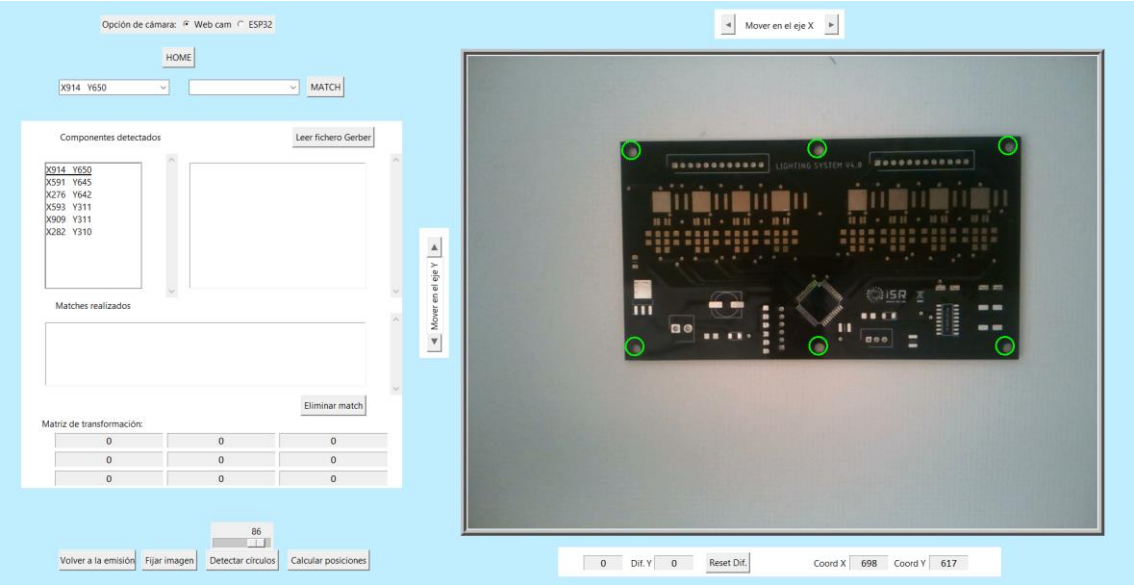


Figura 4.16 Detección de las marcas fiduciarias de la PCB en el sistema máquina

4.2.2. Obtención de las coordenadas de la PCB en el sistema PCB

Como se mencionó en el apartado 4.1.4, a través del botón “Leer fichero Gerber”, la aplicación permite la lectura de un archivo, de tal forma que, al seleccionar un fichero Gerber, se leerá dicho archivo por medio del comando “with open(fichero, 'r') as f”, mediante el cual, se procederá a la lectura del mismo, línea a línea, por lo que lo siguiente, será inicializar un bucle en el cual, por cada iteración, en la que se lee una línea, se comprobará si dicha línea comienza por el carácter X, mediante el siguiente comando “if línea.startswith(“X”)”, en caso afirmativo, se guardará en un vector a través de “arrayLineas.append(línea)”, de modo que, cada vez que se detecte que una línea del archivo, que comience por el carácter “X”, almacena dicha línea, en el cuadro de texto de la GUI, destinado a almacenar las coordenadas de la PCB, en el sistema de referencia de la propia PCB, así, por cada línea que lea, obtendremos un conjunto de coordenadas.

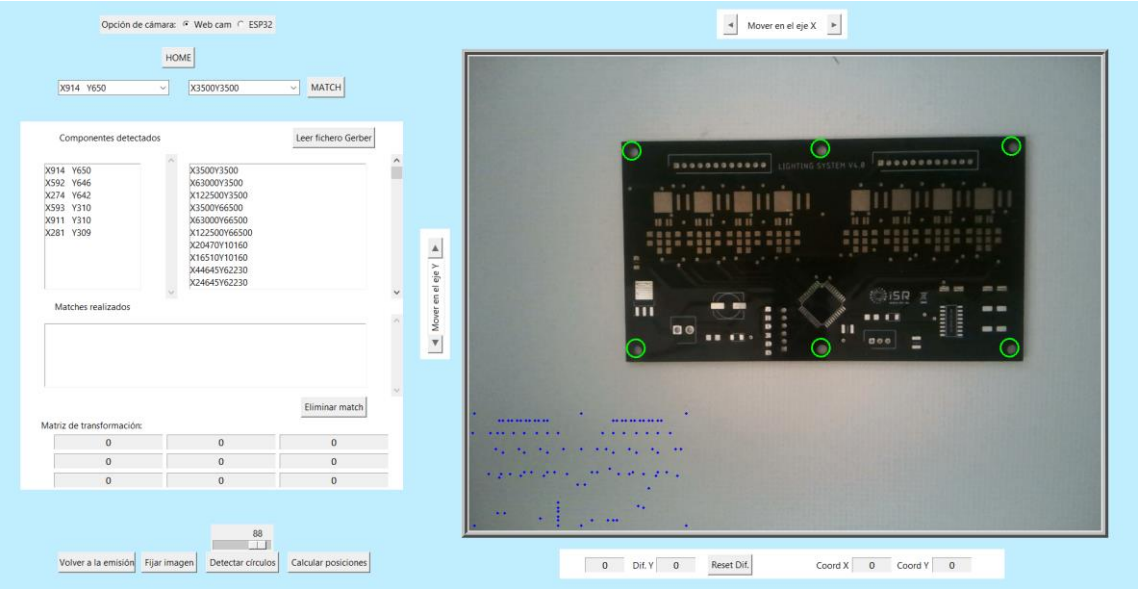


Figura 4.17 Obtención de las coordenadas de la PCB en el sistema PCB

4.2.3. Cálculo de la matriz de transformación

Para el cálculo de la matriz de transformación, se comenzará obteniendo los matches realizados, los cuales se encuentran guardados en el respectivo cuadro de texto. Estos, se desglosan en coordenadas X e Y, del punto A, y el punto B, de cada pareja de puntos, que conforman un match. Tras esto, las coordenadas del punto A, correspondientes a los círculos detectados por el programa, se guardan en el vector “puntosTransformados”, y las coordenadas del punto B, que representan los puntos del fichero Gerber, se almacenan en el vector “puntosOriginales”. Seguidamente, ambos vectores se mandan a la función “calcularMatrizTransformacion”, que, a partir de ellos, obtendrá la matriz de transformación, para el caso en el que se encuentra la PCB.

Este procedimiento se hará de la siguiente manera, primeramente, se crearán dos matrices de ceros, la matriz A que se utilizará para construir el sistema de ecuaciones lineales, y la matriz B, que almacenará los valores de “puntosTransformados”. Seguidamente se pasa a un bucle de 4 iteraciones, una por cada “match” realizado, en la que, por cada iteración, se extraen los valores X e Y de cada valor en los vectores de “puntoOriginal” y “puntoTransfromado”, y se construye la matriz A para representar el sistema de ecuaciones lineales para la transformación. Después de haber rellenado las dos matrices, con sus

correspondientes valores, se resuelve el sistema de ecuaciones utilizando el comando “np.linalg.solve(A,B)” que dará como resultado la matriz de transformación. Por último, se añaden los valores de la matriz creada a unas variables de texto, que permitirán mostrar los valores de la matriz en la interfaz gráfica. Al finalizar los cálculos, se mostrará una ventana emergente que indica, que la matriz de transformación se ha calculado correctamente, y está lista para ser usada en otros puntos.

Tras esto, en la imagen mostrada en la interfaz, se mostrarán, por medio de círculos azules, cada una de las posiciones calculadas en su correspondiente localización. Esto se habrá hecho recorriendo el cuadro de texto, que contiene las coordenadas del fichero Gerber, y, una a una, aplicarle la función “aplicarTransformacion” a cada línea de pareja de coordenadas, al finalizar esto, se habrán calculado todas las posiciones de cada componente.

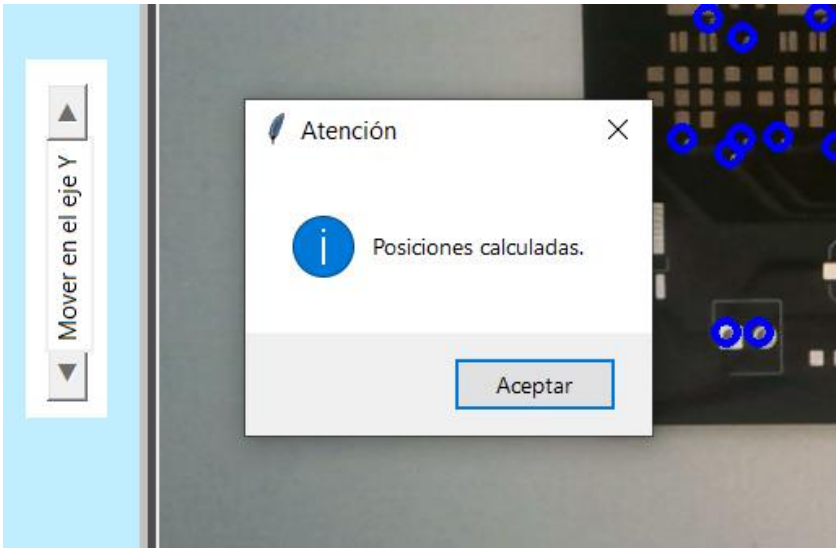


Figura 4.18 Mensaje de matriz de transformación calculada correctamente

4.3. Comunicación con la máquina

Para poder mandar instrucciones a la máquina, será necesario, una placa de Arduino junto con una tarjeta CNC shield, además del cable que conecte estas con el ordenador, y utilizar una serie de comandos, que permitan mover al robot dentro del área de trabajo, y así realizar correctamente las capturas de imagen. Esto se ha llevado a cabo mediante GRBL, un firmware de código abierto empleado, en manejo

de máquinas de control numérico por computador (CNC). Al cargar el software en el microcontrolador Arduino, posibilita la interpretación, y ejecución de instrucciones en G-Code, un lenguaje de programación, formado por comandos alfanuméricos, empleado para controlar máquinas CNC. Cada línea de este código representa una función concreta, como mover un eje, o activar un actuador.

Los comandos requeridos para el proyecto son, las dos órdenes para establecer el movimiento del robot en, movimiento relativo (G91), el origen de coordenadas se traslada, a la posición del robot, cada vez que este se mueve, o movimiento absoluto (G90), el origen de coordenadas es fijo, e instrucciones básicas de movimiento en los dos ejes del plano horizontal XY, tales como desplazarse en el eje X con sentido positivo (G1 X10 F100) o negativo, desplazarse en el eje Y con sentido negativo (G1 Y-10 F100) o positivo, moverse a la posición de home (G28), o moverse a una posición específica, haciendo clic en una de las coordenadas, almacenadas en el cuadro de texto del fichero Gerber, una vez se ha realizado el alineamiento (G0 X{coordenadaX} Y{coordenadaY}).

4.4. Secuencia del proceso de alineamiento de PCBs

Habiendo explicado ya la programación de la aplicación, y la comunicación entre esta y la máquina, se procederá a mostrar la secuencia a seguir, para realizar el alineamiento de una PCB, por medio de un flujograma. En este, se ha tenido en cuenta el principal dilema a la hora de realizar dicha tarea, si el conjunto de marcas fiduciarias, se observan en una sola imagen, o si, por el contrario, serán necesarias más de una imagen, para obtener los 4 puntos necesarios.

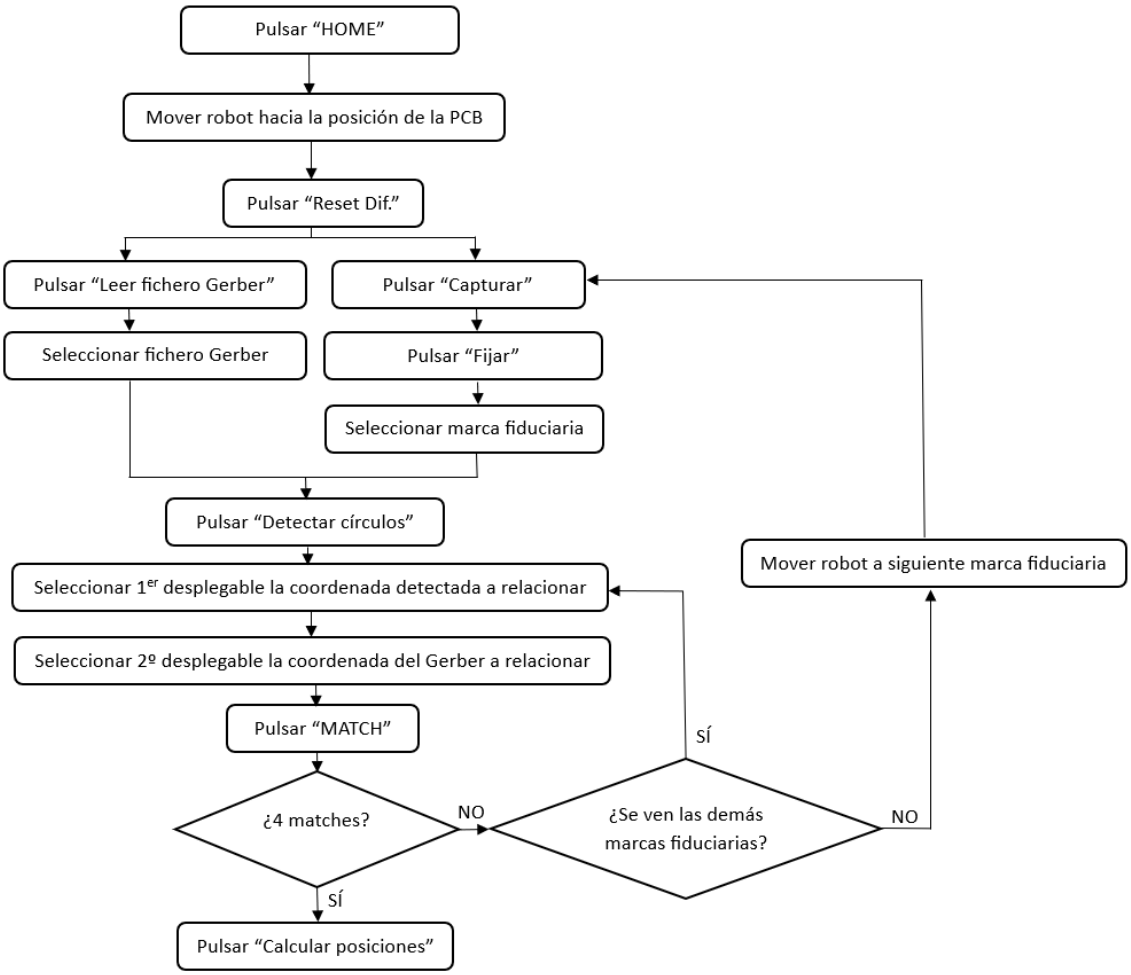


Figura 4.19 Flujograma del proceso de alineamiento

Tras esta secuencia, el programa tendrá calculada la matriz de transformación, y mostrará en la imagen, cada una de las posiciones que ha calculado, además de que permitirá, al seleccionar cada una de ellas, en el cuadro de texto del fichero Gerber, mover al robot a dicha posición, y remarcar en la imagen, dicha localización, con un círculo rojo.

Cabe destacar que, en el proceso mostrado, en caso de necesitar más de una imagen, para ver todas las marcas fiduciarias, se ha indicado que, tras pulsar "Capturar imagen", se requerirá seleccionar la marca fiduciaria. Este paso no es estrictamente obligatorio, tras realizarse por primera vez, ya que el programa, almacenará la marca fiduciaria seleccionada en el paso anterior, por lo que, al ser la misma marca, estará capacitado para detectarla. No obstante, se ha hecho de tal forma, ya que, debido a una diferente iluminación, podría no interpretarlo como el mismo tipo de marca, por lo que seleccionando de nuevo la marca fiduciaria, en este

nuevo punto, lo haría sin problemas, pero también, podría no seleccionarse esta nueva marca, y simplemente, reducir el porcentaje de similitud, situado encima del botón “Detectar círculos”.

5. RESULTADOS Y CONCLUSIONES

En el presente apartado, se mostrarán, y comentarán, los resultados obtenidos, tras realizar el alineamiento de PCBs, a través de la aplicación desarrollada, enfrentando a esta, a distintos escenarios, en los que se puede encontrar la placa, de manera que se pueda saber, la robustez y versatilidad, del sistema que se ha obtenido. Para estos casos de ensayo, se mostrarán las marcas fiduciarias detectadas, a través de círculos verdes como es habitual, y se mostrarán en la misma imagen, la posición de todos los componentes, por medio de círculos azules, más pequeños que los de las marcas fiduciarias.

Primeramente, se ha realizado un alineamiento de la PCB, en un escenario nominal, en el que la placa se encuentra, prácticamente con una orientación horizontal, centrada en la imagen, y observando la placa en su conjunto. En este caso, el alineamiento se ha realizado, de manera correcta, y sin la presencia de errores o inconveniente, y se puede observar, que la distribución de los puntos calculados en la PCB, es la misma que la dispuesta en la nube de puntos, en la esquina inferior izquierda de la imagen.

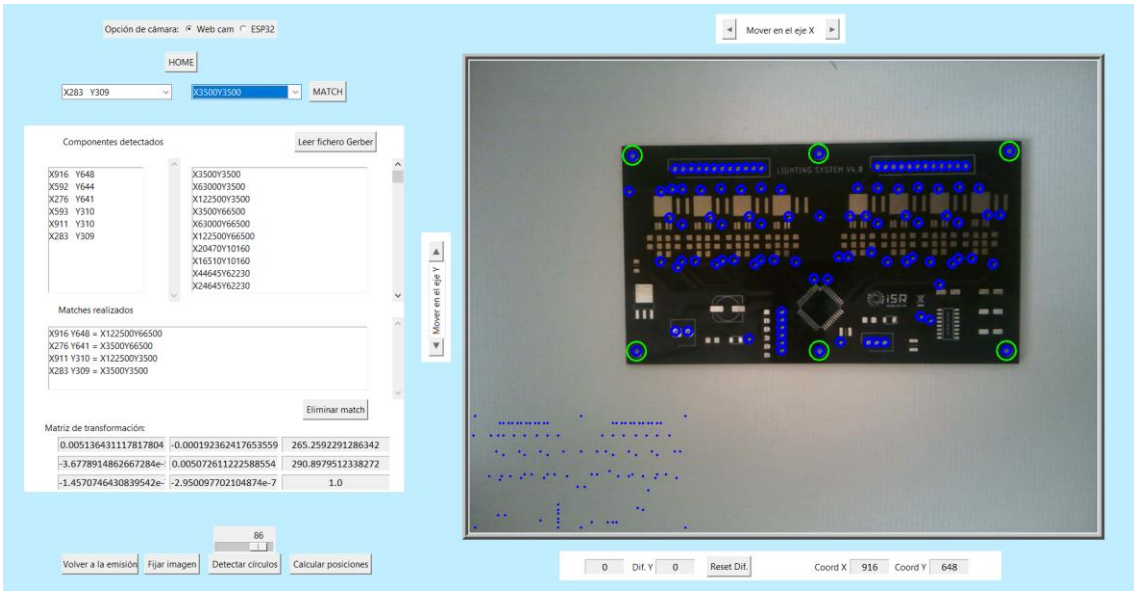


Figura 5.1 Resultado de alineamiento nominal

A continuación, se ha tratado uno de los escenarios, que ha originado la cuestión del trabajo, y es el caso de la rotación. El alineamiento de una PCB, no sólo depende de su posición, sino también de su rotación, que hace que el sistema de referencia de la máquina, difiera en mayor medida con el de la PCB. En este caso,

se ha tratado una rotación exagerada, y fácilmente perceptible al ojo humano, de manera que, si es capaz de realizar el alineamiento, con una rotación así, será capaz de realizarlo en escenarios, con menor rotación de la placa, y, por tanto, más favorables. Para demostrar su funcionamiento, este escenario se ha replicado en dos casos, el primero en el que se ha rotado la PCB en sentido horario, y el segundo, en el que la rotación se ha producido en sentido antihorario. En ambos casos, se puede apreciar que la rotación de la placa, independientemente del sentido de dicha rotación, no perjudica el alineamiento de la misma, y las posiciones se calculan con normalidad.

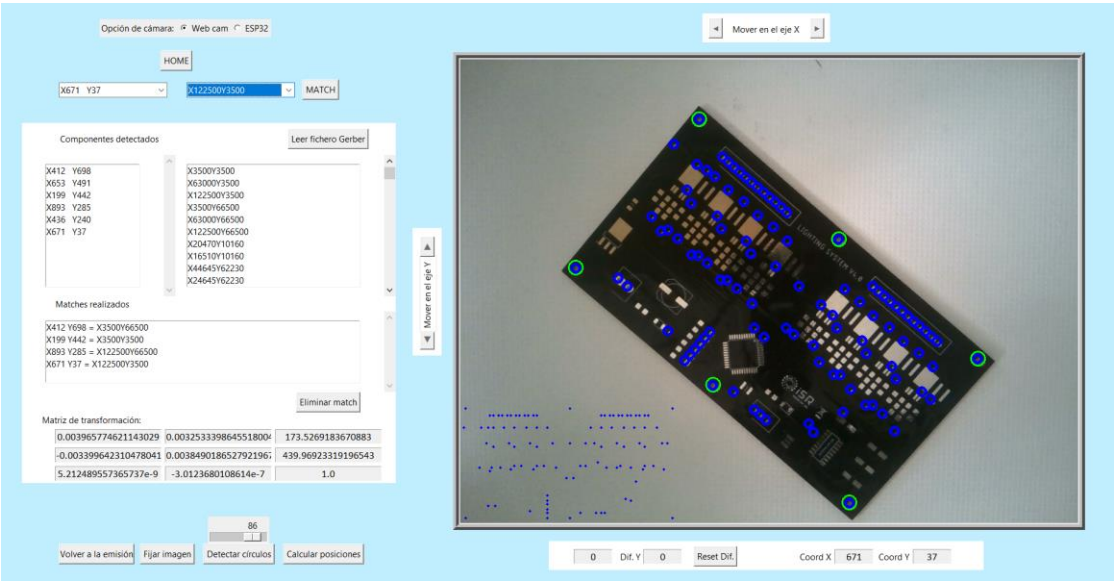


Figura 5.2 Resultado de alineamiento con rotación en sentido horario

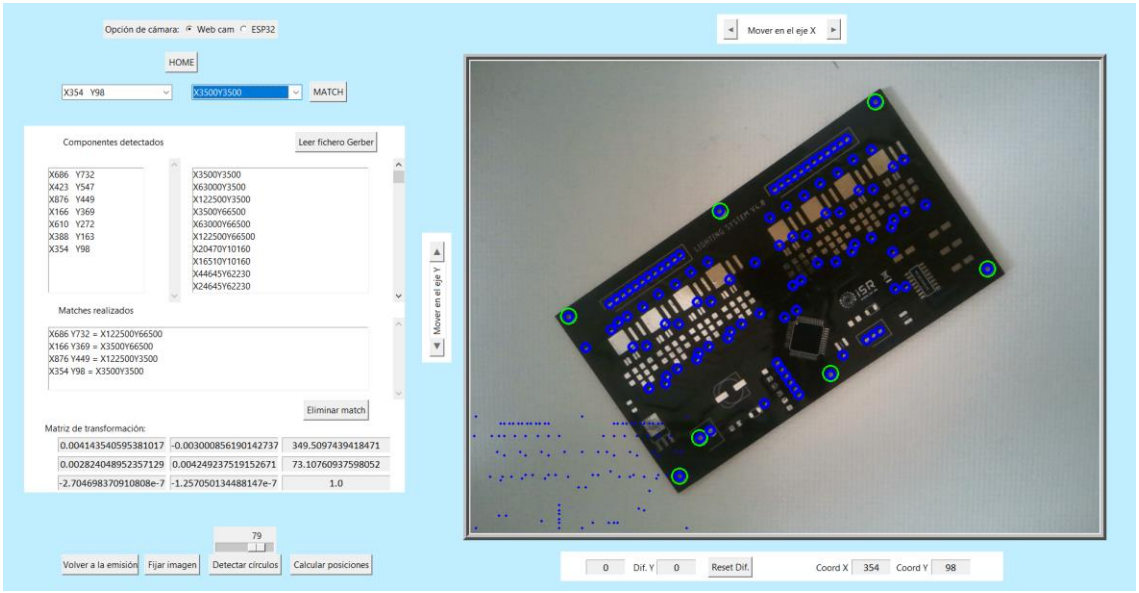


Figura 5.3 Resultado de alineamiento con rotación en sentido antihorario

Para finalizar con los casos de rotación, se ha colocado la PCB completamente en vertical, o lo que es lo mismo, con una rotación de 90º en sentido horario. Una vez más, se pone de manifiesto que, la rotación no es un inconveniente en el sistema.

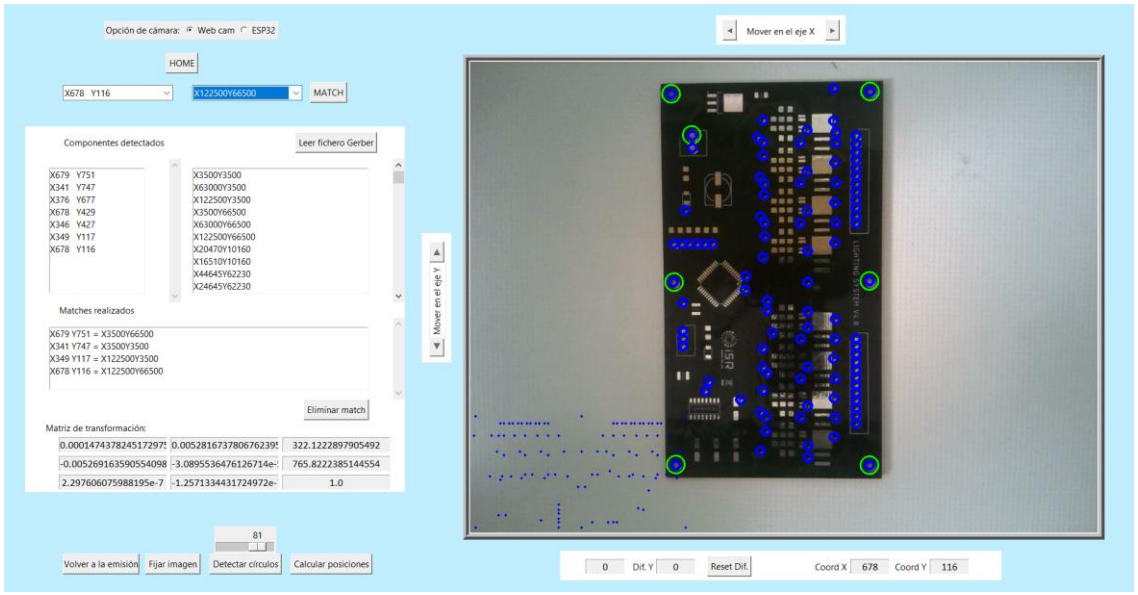


Figura 5.4 Resultado de alineamiento con orientación vertical

Seguidamente, se ha tratado un escenario poco probable, pero posible, por lo que, para descartar fallos de funcionamiento, se ha llevado a cabo. Se trata de un escenario, en el que la PCB se ha colocado boca abajo, es decir, con la cara principal orientada hacia la superficie en la que se apoya la placa, en lugar de hacia el exterior, además, se ha añadido una pequeña rotación a la PCB. La complejidad

de este escenario, está focalizada en la tarea del usuario de realizar el alineamiento, ya que este, debe ser consciente de que, la coordenada del fichero Gerber, que representaría la marca fiduciaria, de la esquina superior izquierda, debe alinearla en este caso, con la coordenada de la marca situada, en la esquina superior derecha, al estar las posiciones de los componentes de la placa, invertidas en el eje X.

Pese a esta complicación, si los 4 matches se realizan adecuadamente, se puede observar en la imagen, que el alineamiento se produce correctamente, y sin ninguna diferencia con los realizados anteriormente, por lo que, en conclusión, mientras los matches se realicen debidamente, el alineamiento no dependerá de qué cara de la PCB se esté observando.

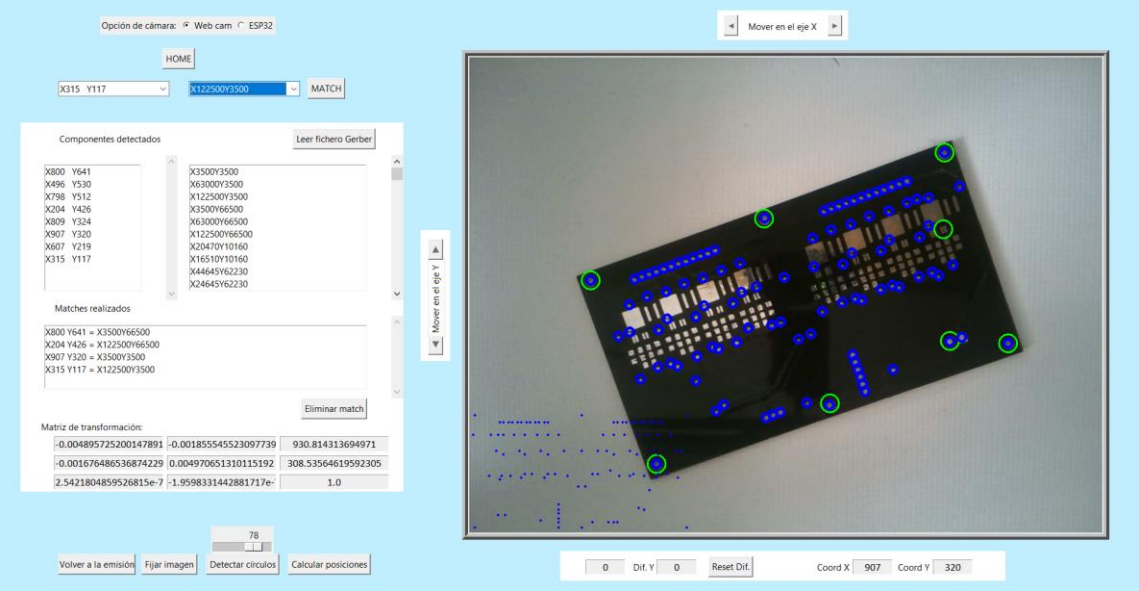


Figura 5.5 Resultado de alineamiento con PCB boca abajo

La PCB que se está utilizando para los casos, consta de 6 puntos que se utilizan a modo de marca fiduciaria, orientados a modo de matriz de 2 filas y 3 columnas, de manera que se encuentran 4 puntos, en las esquinas de la placa, y dos en la columna central de la matriz, y situados en los extremos superior e inferior de la placa. En los casos anteriores, se ha realizado el alineamiento mediante las marcas fiduciarias, situadas en las esquinas de la placa, es decir, la marca situada en la esquina superior izquierda, la situada en la esquina inferior izquierda, la situada en la esquina superior derecha, y la que se encuentra en la esquina inferior derecha.

Para el siguiente caso, se ha realizado el alineamiento, a través de las 4 marcas situadas en los primeros 4 puntos más a la izquierda, es decir, la marca situada en la esquina superior izquierda, la situada en la esquina inferior izquierda, la situada en la parta superior de la zona central, y la que se encuentra en la parte inferior de la zona central. Tras esto, se ha comprobado el resultado obtenido, del que se puede sacar que, el alineamiento no se ve perjudicado, en función de las marcas utilizadas, siempre y cuando estas formen un cuadrado, o rectángulo.

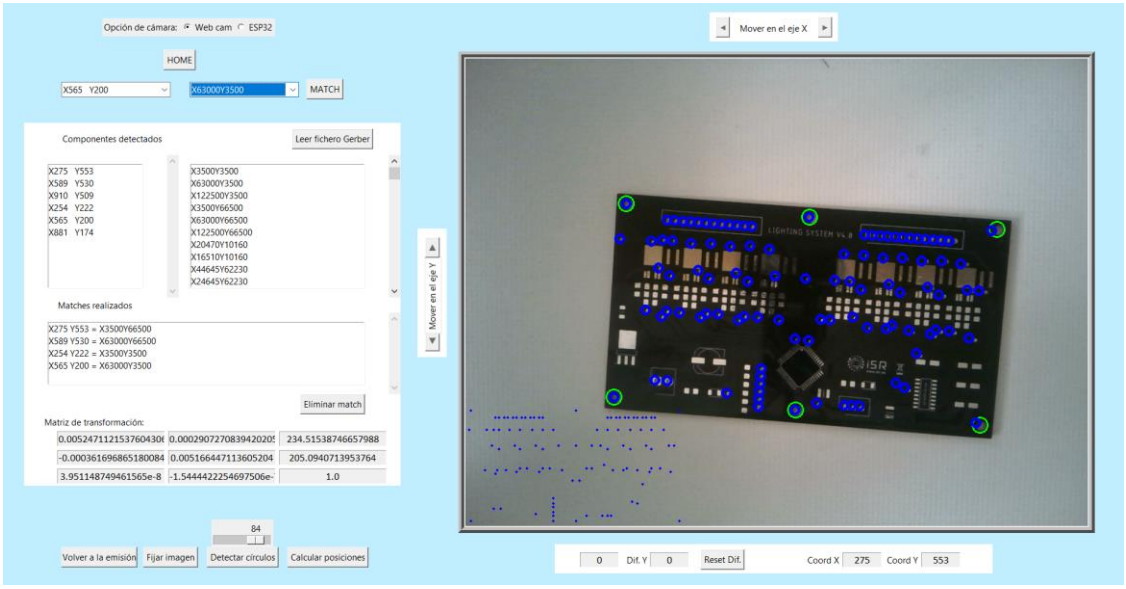


Figura 5.6 Resultado de alineamiento con 4 matches en los 4 puntos más a la izquierda de la PCB

Por último, se plantea otro de los casos que se planeaba solucionar, al inicio del trabajo, en el cual, no se obtiene una visual completa de la PCB a alinear. Este finalmente, no ha sido un inconveniente, ya que la distancia entre el dispositivo de adquisición de imágenes, y la superficie de apoyo de las placas, es suficiente, para que estas, se vean por completo en la imagen obtenida, sin embargo, para demostrar que este problema, también ha sido solventado, se hará una simulación de dicho caso. Para ello, se ha tomado una primera imagen de la PCB, en la que esta, no se ve al completo, y, por tanto, solo se han realizado los 2 matches de la izquierda, que permitía dicha visual. En la siguiente figura, se muestra dicha situación.

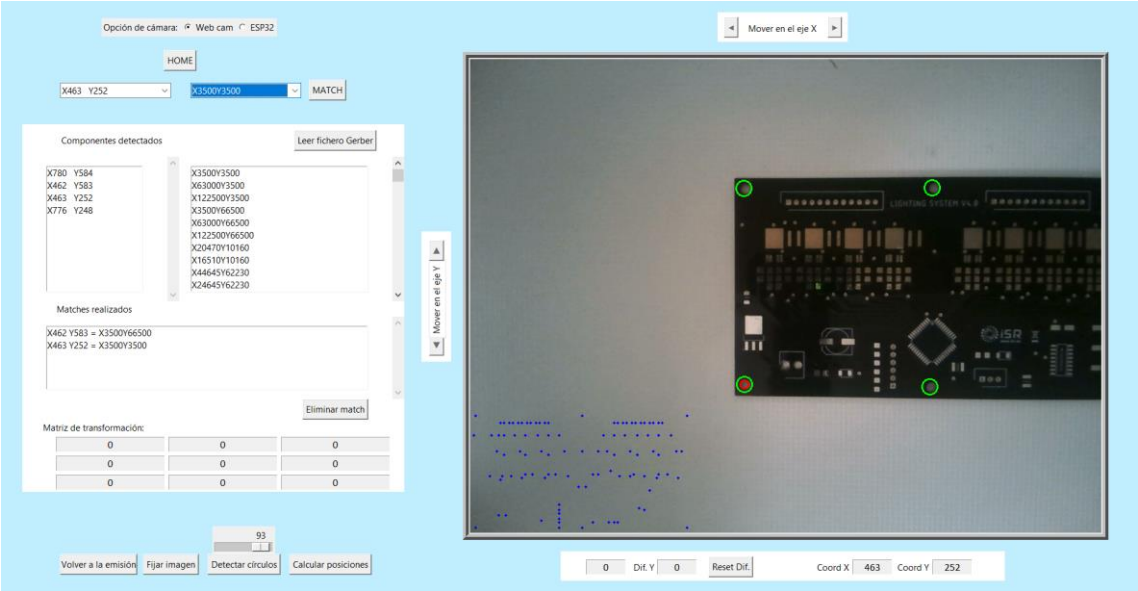


Figura 5.7 Resultado de alineamiento sin ver la PCB al completo (Parte 1)

Como se vio en el caso anterior, el alineamiento de la placa, se podría realizar con las 4 marcas que sí se ven en la imagen, pero para demostrar el funcionamiento del caso actual, solo se utilizaron, las dos marcas de la izquierda en esta imagen.

A continuación, para poder ver el resto de la PCB, se ha desplazado el robot hacia la derecha, por medio del botón "▶" de mover en el eje X. Tras haber realizado este movimiento, se pulsa el botón "Volver a la emisión", "Capturar" y "Fijar imagen", en la cual, ya se observaría la PCB al completo. Seguidamente, se han realizado los 2 matches que faltan, estos son, los dos situados más a la derecha, que no se veían en la primera imagen. Tras este proceso, se calculan las restantes posiciones, y el resultado permite ver que, aunque los matches se realizan en distintas imágenes, el alineamiento se producirá igualmente, como se muestra a continuación.

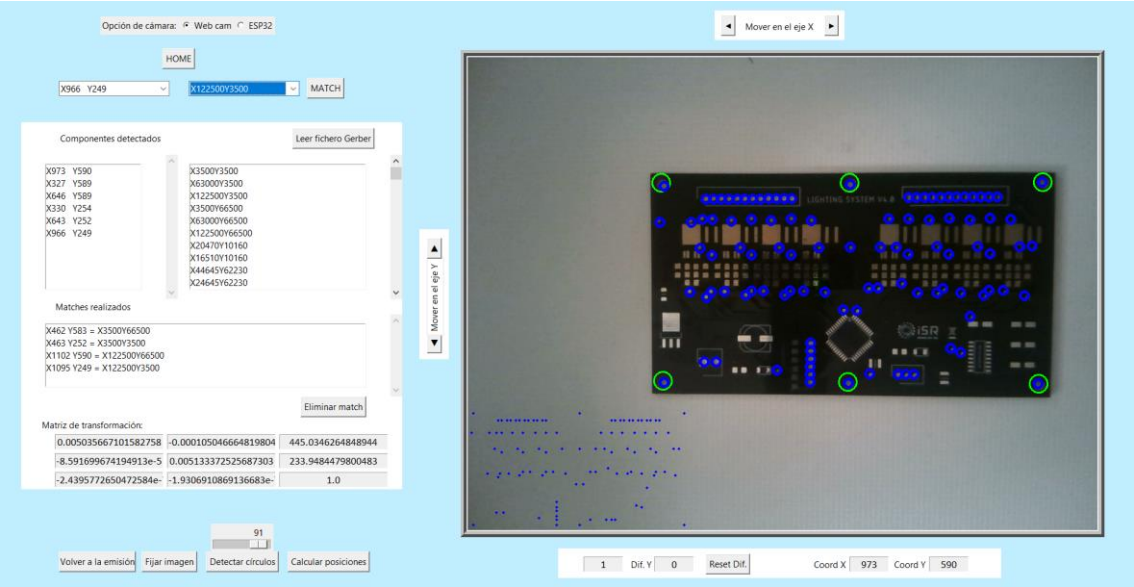


Figura 5.8 Resultado de alineamiento sin ver la PCB al completo (Parte 2)

Tras ver todos estos casos, en los que se dispone la PCB a alinear, en distintas posiciones y orientaciones, se puede determinar, que el sistema funciona correctamente, y realizará el alineamiento debidamente, siempre y cuando, se cumplan dos premisas, que los 4 matches a realizar, tengan una disposición en forma cuadrada, o rectangular, y que estos matches, se realicen correctamente, es decir, que cuando se establezca una de estas relaciones de coordenadas, entre A y B, realmente, esa coordenada A, en uno de los sistemas de referencia, represente al mismo punto, que la coordenada B, en el otro sistema de referencia.

6. FUTURAS MEJORAS

Como se mencionó al principio de la memoria, el presente trabajo se encarga, de la programación de una aplicación, que permita la identificación, y alineamiento de PCBs, dentro del área de trabajo de una máquina. Esta es la primera parte de un proyecto más grande, en el que se busca, crear una máquina de ensamblaje, de componentes SMD, en PCBs. En consecuencia, las futuras mejoras, y proyectos venideros, constituyen las siguientes partes de dicho objetivo, hasta finalmente, tener la máquina de ensamblaje al completo.

A la finalización de este proyecto, el cabezal del robot solo tendrá el dispositivo de adquisición de imágenes. Sin embargo, para poder realizar las tareas de ensamblaje, será necesario instalar en dicho cabezal, una serie de actuadores, encargados de la instalación de componentes en la placa. Una de las principales mejoras, que se requerirá en el futuro el proyecto, para finalmente convertirse en una máquina de ensamblaje, será la instalación de sendos actuadores.

Junto a la mejora mencionada anteriormente, será necesario un elemento suministrador de componentes SMD, para que el robot, cada vez que requiera ensamblar uno de estos elementos, se mueva a dicho suministrador, recoja el componente, y se mueva para proceder a su instalación. Gracias a este dispositivo, sólo será necesario, proporcionarle de suficientes componentes, y el robot se encargará de administrárselos.

Con estas dos mejoras anteriores, se podría implementar un programa de ensamblado automático, mediante el cual, una vez realizado el alineamiento de la PCB, y sabiendo las posiciones donde se debe instalar cada componente, el robot sea capaz de colocar, cada uno de estos componentes, en sus respectiva posición dentro de la PCB, de manera que el robot de manera autónoma, recoja los componentes del elemento suministrador, los coloque en posición, y los ensamble, de manera continuada, hasta realizar el ensamblaje completo de la placa.

Otra mejora, sería la implementación de una estación de recogida de PCBs y otra para depositarlas, es decir, una zona reservada a, dejar un conjunto de PCBs para que el robot, cada vez que termina su tarea, coja una nueva placa para realizar

de nuevo el proceso de ensamblado, en lugar de requerir de un operario que se las entregue una a una, y la otra, sería una estación en la que el robot, deposite las PCBs una vez las ha realizado por completo el ensamblaje de estas.

Tras la aplicación de las anteriores mejoras, se pasaría de tener una máquina semiautomática, a una completamente automática, por lo que, en consecuencia, será necesario dotar al robot, de sensores de final de carrera, de modo que, en ningún momento, el robot corra riesgo de descarrilar, ya que será consciente de dónde se encuentran sus límites, además de que pueda realizar el homming de manera correcta.

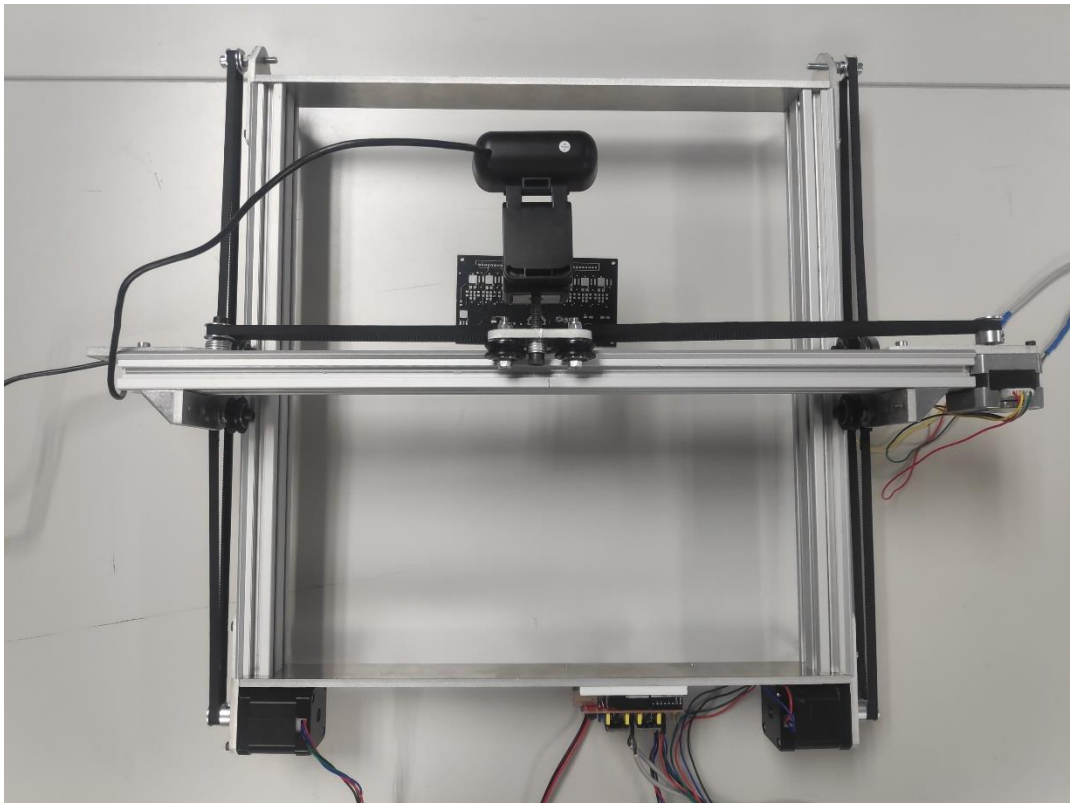
Una mejora no tan requerida en las futuras tareas, pero que sí podría mejorar el funcionamiento, de la actual tarea del robot, es dotarle de una correcta iluminación. Como se mencionó en el flujograma, del apartado 4.4, en ciertas ocasiones, el algoritmo puede detectar, cierta marca fiduciaria, con una iluminación distinta a la marca que está comparando. Para esto se comentó, que se podría reducir el porcentaje de similitud, con el que se comparan las marcas, y efectivamente da resultado, no obstante, esto se podría evitar, colocando un elemento que proporcione, una iluminación homogénea, a lo largo de toda el área de trabajo, para que, en todas las zonas, donde se encuentra una marca fiduciaria, el algoritmo no pueda confundirse, debido a una variación en la iluminación.

7. ANEXOS

7.1. Manual de uso

SISTEMA DE ALINEAMIENTO DE PCBs MEDIANTE EL USO DE VISIÓN POR COMPUTADOR

Manual de uso



1. INTERFAZ GRÁFICA

La representación de la interfaz gráfica presente en la Figura 1.1, muestra una serie de números que representan, al conjunto de elementos de interés en la interfaz. A continuación, por orden numérico, se detallarán cada uno de ellos y su funcionalidad.

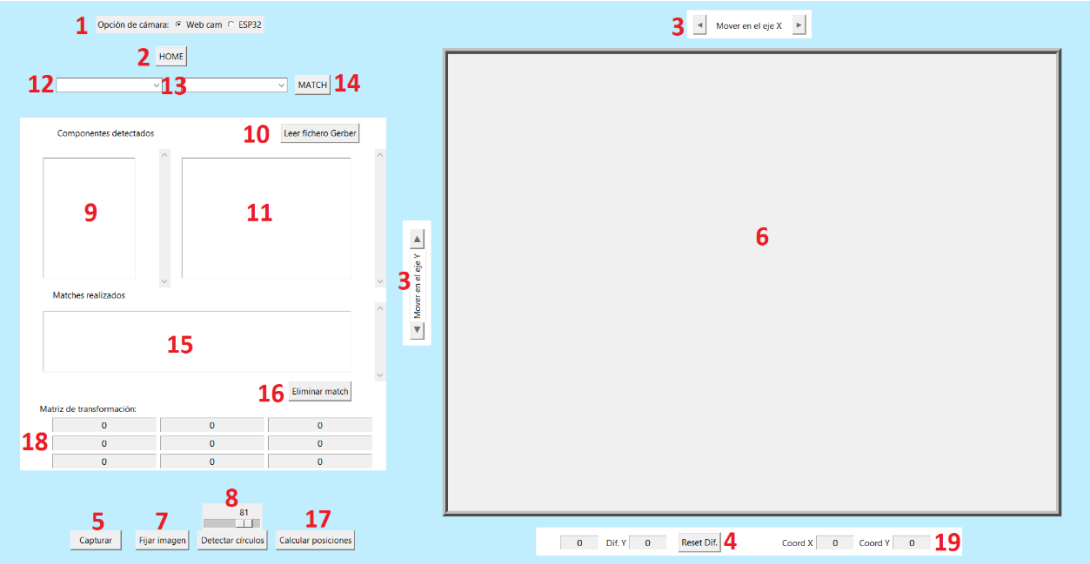


Figura 1.1 Distribución de la interfaz gráfica de usuario

- Punto 1. Se trata de un seleccionador de dispositivo de adquisición de imágenes, con él, se escogerá el elemento empleado para la toma de imágenes. Está programado para dos casos, el primero en el que se utiliza una webcam, y el cual, viene marcado de serie, al inicializar el programa, y el segundo, en el que se utiliza el microcontrolador ESP32 con cámara integrada.
- Punto 2. Con este botón, se establece la posición actual del robot como la posición de home, establece el movimiento en coordenadas relativas, y habilita el resto de botones de movimiento. Además, si se mueve el robot a otra posición, al pulsar este, volverá a la posición de inicio.
- Puntos 3. Consisten en un conjunto de 4 puntos, que permiten el movimiento en los ejes X e Y del robot. Estos serán necesarios para mover el cabezal del robot, a la posición en la que se encuentra la PCB a alinear.
- Punto 4. Es un botón que resetea la distancia, que se ha desplazado el robot en ambos ejes, almacenada en los cuadros de texto a la izquierda de

este botón, “Dif. X” y “Dif. Y”. Este será necesario ya que, en caso de requerir de más de una imagen, para obtener las 4 marcas fiduciarias, necesarias para realizar el alineamiento, se necesitará la distancia desplazada, para obtener cada imagen respecto a la primera imagen capturada.

- Punto 5. Botón que ejecuta la captura de imagen. Tras pulsar este botón, se mostrará la imagen en la interfaz, y se convertirá este botón, en el botón “Volver a la emisión”.
- Punto 6. Es el área en el que se mostrará la imagen capturada. Además, tras mostrar la imagen, es donde se seleccionará la marca fiduciaria a detectar, pulsando el clic izquierdo, arrastrando hasta generar el área en la que se encuentra dicha marca, y soltando el clic izquierdo.
- Punto 7. Botón para fijar la imagen actual, para poder trabajar sobre ella. Tras pulsar este botón, se habilitará la opción de seleccionar un área de la imagen, para la marca fiduciaria.
- Punto 8. Representa al conjunto del botón de “Detectar círculos” y el regulador del propio botón. Con el regulador se establece el porcentaje de similitud con el que compara la marca fiduciaria seleccionada, y las que detecte en la imagen, y con el botón, busca dentro de la imagen, zonas que guarden similitud en los píxeles, con la marca fiduciaria seleccionada.
- Punto 9. Cuadro de texto en el que se mostrarán las coordenadas X e Y de las marcas fiduciarias detectadas en la imagen. Haciendo clic en cada una de ellas, se marcará en la imagen, el respectivo punto, mediante un círculo rojo.
- Punto 10. Botón para seleccionar el fichero Gerber de la PCB. Abrirá una ventana para buscar un archivo en el ordenador, y al seleccionar el respectivo documento, lo leerá línea a línea.
- Punto 11. Cuadro de texto en el que se mostrarán las coordenadas X e Y leídas en el fichero Gerber. Por cada línea del fichero que comience por “X”, la escribirá aquí.
- Punto 12. Desplegable en el que se muestran las mismas coordenadas que en el cuadro de texto del punto 9. Con este, se seleccionará una de estas coordenadas para, más adelante, realizar un “match”.

- Punto 13. Desplegable en el que se muestran las mismas coordenadas que en el cuadro de texto del punto 11. Con este, se seleccionará una de estas coordenadas para, más adelante, realizar un “match”.
- Punto 14. Botón que realiza el “match” entre las coordenadas marcadas en el desplegable del punto 12, y las coordenadas marcadas en el desplegable del punto 13.
- Punto 15. Cuadro de texto donde se muestran los “matches” realizados.
- Punto 16. Botón que permite eliminar un “match” determinado. Para ello, se selecciona el “match” a eliminar en el cuadro de texto del punto 15, y se pulsa este botón.
- Punto 17. Botón que, una vez se han realizado los 4 “matches” necesarios, calcula la matriz de transformación. Tras pulsar este botón, se mostrará una ventana emergente, indicando que se ha realizado correctamente, y se dibujarán todos los puntos calculados, en la imagen de la interfaz.
- Punto 18. Consiste en 9 casillas, que muestran el contenido de la matriz de transformación, utilizada para calcular el resto de posiciones. La disposición de cada casilla, es la misma que la disposición de cada valor en la matriz.
- Punto 19. Área en la que se muestran las coordenadas X e Y, de donde se ha realizado un clic en la imagen, o de donde se encuentra el elemento que se ha seleccionado en el cuadro de texto del fichero Gerber, ya con la matriz de transformación calculada.

2. PARÁMETROS MODIFICABLES

En el presente apartado, se comentarán los parámetros de la aplicación, que pueden ser modificados, en caso de que usuario lo requiriese. Estas modificaciones, no supondrían en principio ninguna alteración, en el funcionamiento de la aplicación, ya que serían pequeños ajustes, que no afectan al algoritmo del programa, para realizar la tarea de alineamiento, para la que ha sido programado.

2.1. Movimiento relativo en los ejes

El movimiento relativo del robot, es decir, mover el robot, poco a poco en cada eje, se realiza mediante los 4 botones, representados con flechas en forma de triángulo, situados encima de la imagen, para el eje X, y a la izquierda de la imagen, para el eje Y. Si se deseara modificar el desplazamiento o velocidad, de estos movimientos habrá que modificar las funciones “aumentarEjeX”, “disminuirEjeX”, aumentarEjeY” y “DisminuirEjeY”, donde la segunda línea de código, de cada una de ellas, establecerá el tipo de movimiento a realizar.

En el primer caso, por ejemplo, aparecerá la línea “comando_gcode = 'G1 X1 F100'”, por lo que bastaría con modificar el valor “X1” a “X3”, en el caso de querer aumentar en 3 unidades el desplazamiento por cada clic, y cambiar el valor “F100” a “F50” si se deseara reducir a la mitad la velocidad de desplazamiento. Para el caso del eje Y, se realizaría la misma modificación, salvo que con el carácter “Y” en lugar del carácter “X”. El término “G1” o “G0” representa el tipo de movimiento a realizar, el primero indica un movimiento controlado, en el que se podrá modificar la velocidad del desplazamiento, a través del carácter “F”, por otro lado, el segundo es un movimiento rápido, en el cuál, no se indicará la velocidad del movimiento, ya que este, vendrá predeterminado, en el archivo de la configuración GRBL.

2.2. Selección del dispositivo de adquisición de imágenes

Para la selección del dispositivo de adquisición de imágenes, se creó un seleccionador de dispositivo, situado en la esquina superior izquierda de la interfaz gráfica. Este, aparece predefinido en WebCam, pero aparece desmarcada la opción del microcontrolador ESP32, el cual puede ser modificado por otro dispositivo. Para ello, habría que crear una condición, en la que, a partir de la variable ya creada, del

seleccionador de dispositivo, “radioButtonVar”, se ejecute la inicialización de uno u otro dispositivo.

3. EJEMPLO DE USO

Con la interfaz gráfica cargada, y habiendo comprobado que todas las conexiones se encuentran en correcto estado, se comienza pulsando el botón “HOME”, para establecer la posición de home en las coordenadas actuales, y permitir el movimiento de este. Seguidamente, se moverá el robot, a partir de los botones “▶”, “◀”, “▼” y “▲” hasta la posición en la que se encuentre la PCB a alinear, de manera que el cabezal del robot, en el que se encuentra el dispositivo de adquisición de imágenes, se sitúe encima de la placa. Una vez logrado esto, se hará clic en el botón “Reset Dif.” para poner a cero, los valores X e Y, en los que se ha trasladado el robot, y tomar la posición actual, como origen entre esta, y las posibles futuras capturas de imagen necesarias.

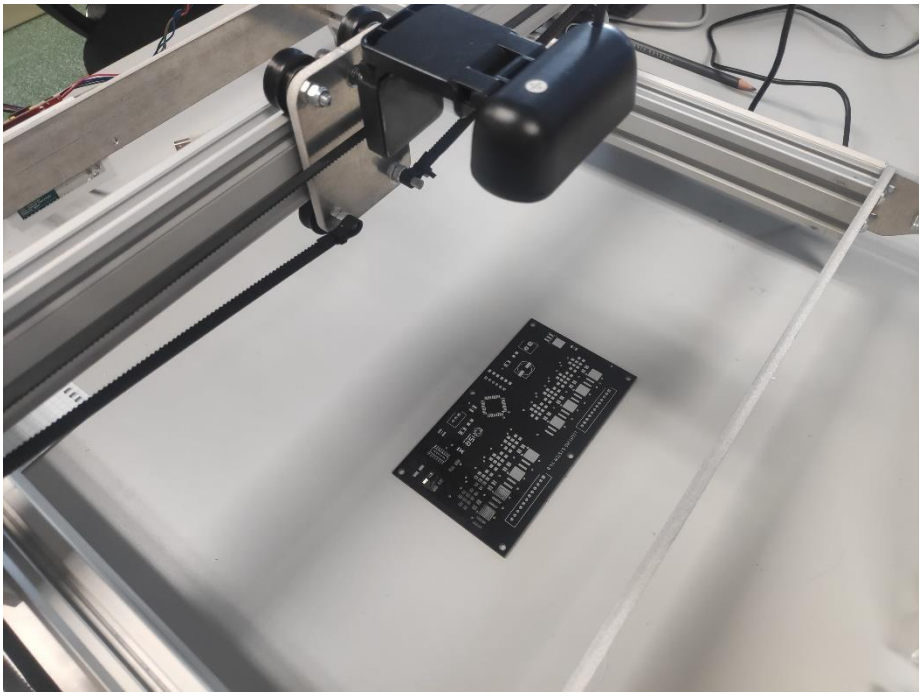


Figura 3.1 Situación de la WebCam encima de la PCB a alinear

Tras esto, se procederá a la captura de imagen, para ello, se pulsará el botón “Captura” y, si se ha realizado correctamente, se mostrará la captura, en el espacio reservado a la derecha de la interfaz gráfica. Con la imagen representada, existen dos opciones, que se fije dicha imagen, pulsando el botón “Fijar imagen”, o realizar una nueva captura, pulsando “Volver a emisión” y nuevamente el botón “Captura”. Cuando finalmente se ha fijado la imagen, que se planea utilizar, se seleccionará la marca fiduciaria que se desea usar.

Esto se hará, pulsando el botón izquierdo del ratón dentro de la imagen y arrastrando, hasta integrar dentro del área creada, la marca a utilizar. Al soltar el botón izquierdo del ratón, no se verá nada especial, pero se habrá guardado un recorte de dicha área, en la carpeta del proyecto, con el nombre “plantilla0.jpg”. Cada vez que se hace este proceso, se sobrescribirá el archivo mencionado, por lo que, si se seleccionase incorrectamente la marca fiduciaria, solo habría que volver a seleccionarla de nuevo, y la anterior se desestimaría.

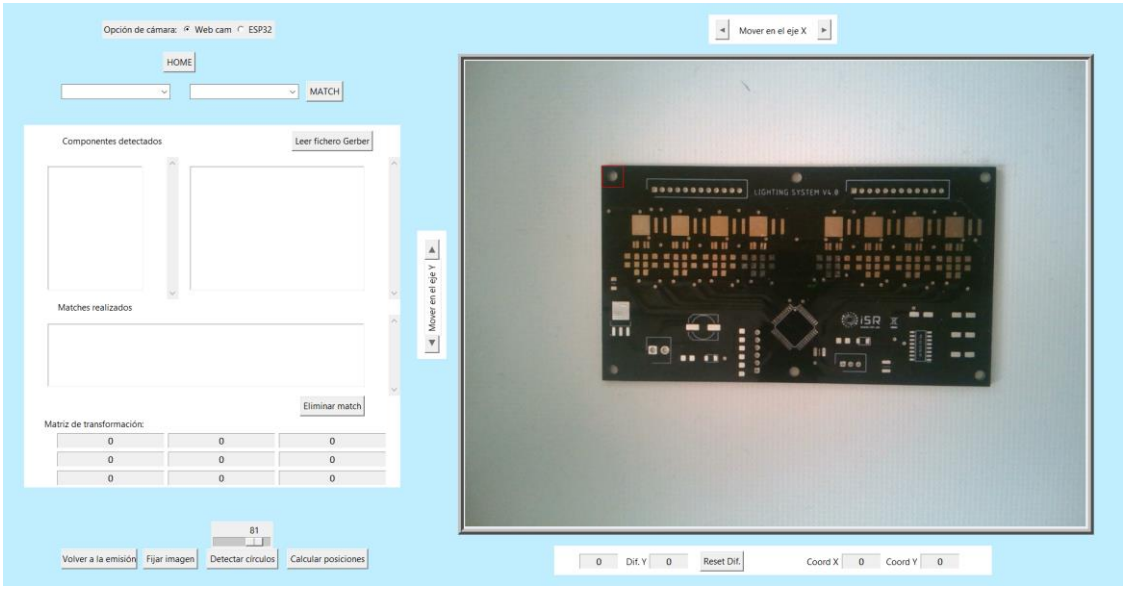


Figura 3.2 Selección de la marca fiduciaria

La lectura del fichero Gerber, se puede realizar en cualquier momento, incluso antes de pulsar el botón “HOME”. Al pulsar el botón “Leer fichero Gerber”, se mostrará una ventana, para seleccionar un archivo dentro del ordenador, al seleccionar el fichero Gerber, se mostrará en el cuadro de texto inferior, todas las coordenadas XY almacenadas en él. Además, habiendo leído ya el fichero Gerber, mostrará más adelante, abajo a la izquierda de la imagen, una nube de puntos que representa la disposición de la PCB.

Cuando se haya obtenido, la marca fiduciaria deseada, se procederá a la búsqueda de marcas similares en la imagen, para ello se pulsará el botón “Detectar círculos”, que señalizarán, mediante un círculo verde dentro de la imagen, todas las marcas fiduciarias similares a la elegida anteriormente. Si alguna de estas marcas no se hubiese detectado, bastaría con reducir el porcentaje de similitud, situado encima del botón “Detectar círculos”, y una vez se ha reducido, volver a pulsar este

botón. Cabe mencionar que, la detección de marcas incorrectas, es decir, que detecte una marca donde no la hay, no supone ningún problema, ya que es el propio usuario el que luego elegirá, con qué marcas se queda y con cuáles no. Lo importante es que como mínimo, detecte las que sí hay.

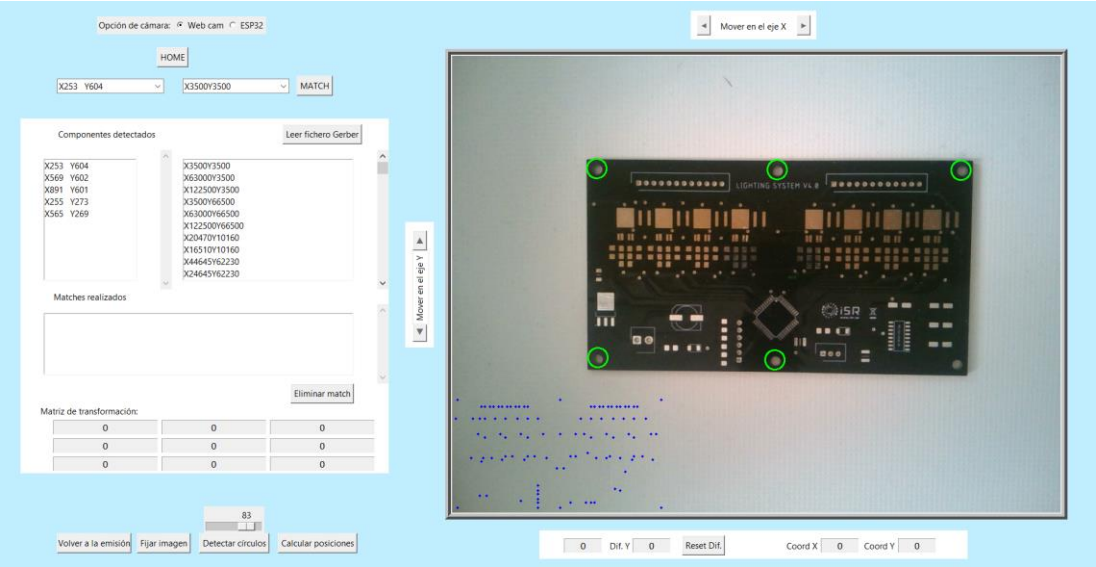


Figura 3.3 Detección de 5/6 marcas fiduciarias

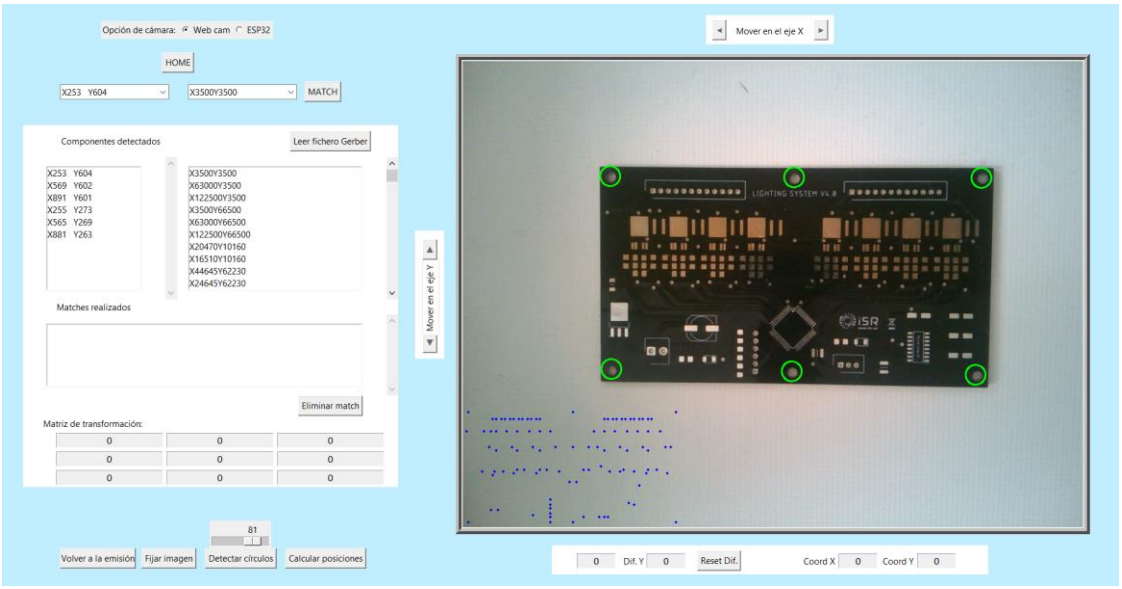


Figura 3.4 Detección de 6/6 marcas fiduciarias reduciendo el porcentaje de similitud

Con las coordenadas del fichero Gerber almacenadas, y habiendo detectado el conjunto de marcas fiduciarias que se encuentran en la imagen, se comenzará a realizar los “matches” o relaciones entre coordenadas de distintos sistemas. Para esta tarea, se disponen de dos desplegables, el primero tendrá las mismas coordenadas que se muestran en el primer cuadro de texto, son las que representan

la localización de las marcas fiduciarias en el sistema máquina, y en el segundo, las coordenadas del fichero Gerber. Para realizar un match, bastará con seleccionar en cada desplegable las coordenadas a relacionar, y posteriormente, pulsar el botón “MATCH” situado a la derecha. Estos matches, deben ser acordes a la realidad, es decir, por cada match, la coordenada del desplegable 1 debe coincidir, con la que representa la coordenada del desplegable 2, ya que son los encargados de realizar el alineamiento. Para comprobar, a qué marca fiduciaria corresponde cada coordenada del desplegable 1, se podrá hacer clic en dicha coordenada dentro del cuadro de texto en el que están almacenadas y, al hacer clic en esta coordenada, en la imagen, dicha marca se verá señalizada mediante un círculo rojo.

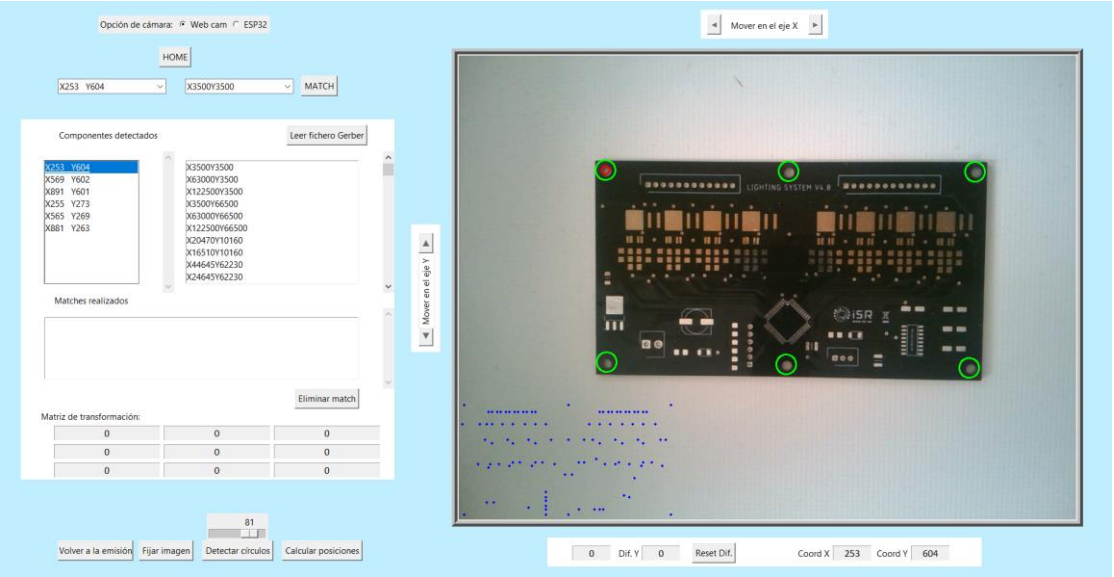


Figura 3.5 Remarque de la marca fiduciaria de la imagen clicando en el cuadro de texto

Tras realizar un match, este se verá en el cuadro de texto inferior de la interfaz, en el que se verá que la coordenada 1 es igual a la coordenada 2. Si se necesitase eliminar un match realizado, se haría haciendo clic en dicho match, dentro del cuadro de texto que los almacena, y pulsando el botón “Eliminar match”.

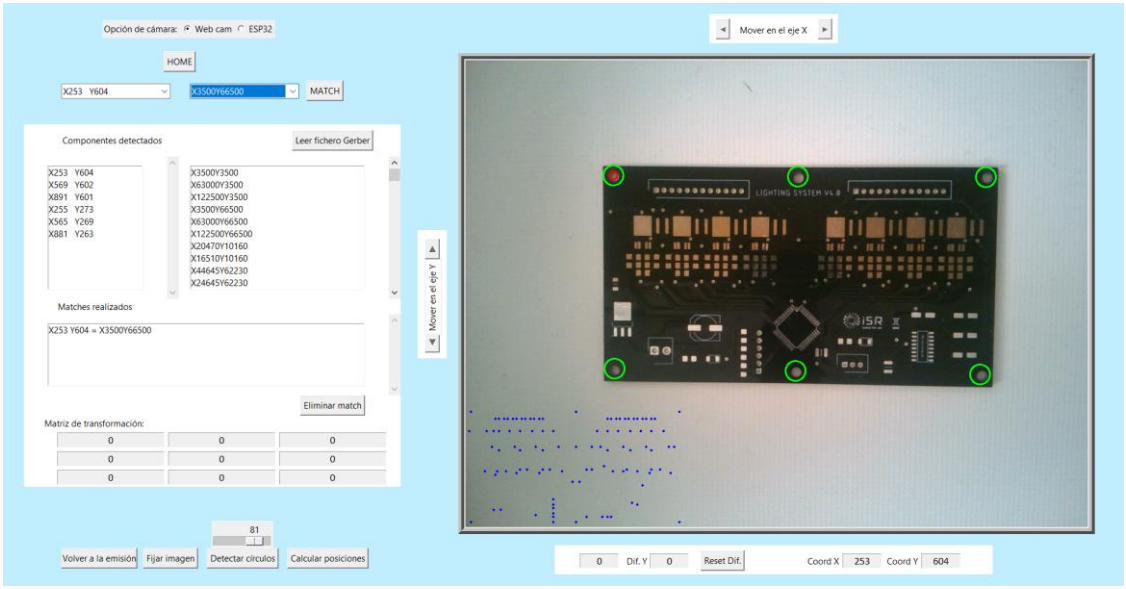


Figura 3.6 Realización de un match

Para poder realizar el alineamiento, serán necesarios 4 matches. Estos se pueden encontrar en una sola imagen, porque se viesen las 4 marcas fiduciarias en la imagen, o puede darse el caso de que en la imagen solo se vea 1 marca fiduciaria y, por tanto, se necesitasen más imágenes. En el primer caso no habría complicación, ya que consistiría en realizar los 4 matches a partir de las 4 marcas detectadas. En el segundo caso, tras realizar el match de la marca fiduciaria detectada en la primera imagen, se requeriría mover el robot a una nueva posición, en la que sí se viese la siguiente marca. Este movimiento se haría mediante las flechas de movimiento mencionadas al principio del apartado, y el resto del proceso se haría igual, ya que la única diferencia, sería la distancia desplazada, y esta se almacena al realizar el movimiento del robot.

Al cambiar la posición del robot, es posible que, debido a una distinta iluminación, el algoritmo no identifique la nueva marca fiduciaria, con la seleccionada en la primera imagen. Para solucionar esto hay dos opciones, la primera sería reducir el porcentaje de similitud, y volviendo a pulsar el botón “Detectar círculos”, o, por el contrario, volver a seleccionar la nueva marca fiduciaria en la nueva imagen, para ello habrá que volver a pulsar “Capturar imagen”, ya que una vez pulsado “Detectar círculos”, no permite seleccionar áreas dentro de la imagen.

Cuando se hayan realizado los 4 matches necesarios, se pulsará el botón “Calcular posiciones”, que realizará los cálculos necesarios para obtener la matriz de

transformación. Tras esto, se mostrará una ventana emergente indicando que los cálculos se han realizado correctamente. En caso de no aparecer dicha ventana, será porque se haya detectado un error, muy posiblemente en la creación de los matches, habría que eliminar el match erróneo, y crear el correcto. Con la obtención de la matriz de transformación, el algoritmo recorrerá el cuadro de texto, que contiene las coordenadas del fichero Gerber, y aplicará dicha transformación, una a una, a cada pareja de coordenadas, a la finalización de esto, se habrán calculado y mostrado, en la imagen, todas las coordenadas de la PCB, en las que se instalarán componentes electrónicos. Además, al hacer clic en una de las coordenadas, del cuadro de texto del fichero Gerber, se mostrará en la imagen un círculo rojo, donde se encuentre dicha coordenada transformada, para resaltar a cuál se refiere, y el robot, se moverá automáticamente, a dicha posición.

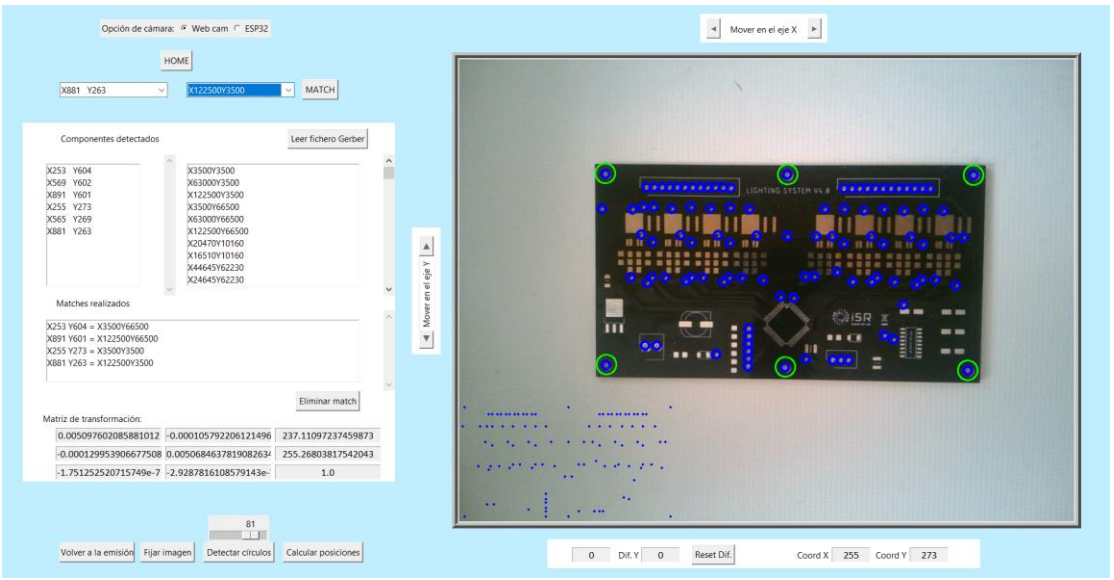


Figura 3.7 PCB alineada y posiciones calculadas y mostradas

Bibliografía

3D natives. (s.f.). *Qué es un G-Code y cuál es su función en el proceso de impresión 3D*. Obtenido de https://www.3dnatives.com/es/g-code-proceso-impresion-3d-230920212/.

3dfils. (s.f.). *¿Qué es el GCODE? Transformando tus modelos 3D al lenguaje de las impresoras 3D*. Obtenido de https://www.3dfils.com/es/blog/entrada/que-es-el-gcode.html.

Altium. (s.f.). *Qué es una PCB*. Obtenido de https://resources.altium.com/es/p/what-is-a-pcb.

Arduino, A. (s.f.). *Máquinas CNC en Arduino*. Obtenido de https://aprendiendoarduino.wordpress.com/tag/grbl/.

COGNEX. (s.f.). *Alineación de fiducial de PCB*. Obtenido de https://www.cognex.com/es-es/industries/electronics/pcb-assembly/pcb-fiducial-alignment.

Configuración de ESP32 en Arduino. (s.f.). Obtenido de https://estudiomiranda.github.io/ESP32/guia/#configuracion.

Crisalctime. (s.f.). *Uso y experiencia GRBL*. Obtenido de https://crisalctime.com/aprendiendo-a-configurar-grbl-1-1f/.

Duino pro. (s.f.). *Como instalar ESP32 en el IDE de Arduino*. Obtenido de https://duino.pro/como-instalar-esp32-en-el-ide-de-arduino-tutorial/.

GCODE. Qué es y cómo puede usarse. (s.f.). Obtenido de https://sites.google.com/site/mitevotarantula/consejos-de-impresion/gcode.

github. (s.f.). *GRBL*. Obtenido de https://github.com/grbl/grbl.

How To Mechatronics. (s.f.). *Cómo configurar GRBL y controlar la máquina CNC con Arduino*. Obtenido de https://howtomechatronics.com/tutorials/how-to-setup-grbl-control-cnc-machine-with-arduino/?utm_content=cmp-true.

Luis Llamas. (s.f.). *Cómo se fabrica una PCB*. Obtenido de https://www.luisllamas.es/como-se-fabrica-una-pcb/.

Luis Llamas. (s.f.). *Qué es el G-Code y su importancia en la impresión 3D*. Obtenido de https://www.luisllamas.es/que-es-el-g-code-y-su-importancia-en-la-impresion-3d/.

mbrobotics. (s.f.). *GRBL – Configuración*. Obtenido de https://mbrobotics.es/blog/grbl-configuracion/#:~:text=GRBL%20es%20un%20firmware%20para,y%20desplazamiento%20de%20nuestra%20maquina.&text=Los%20controladores%20paso%20a%20paso,de%20impulso%20m%C3%ADnima%20de%20paso.

Random Nerd Tutorials. (s.f.). *Instalación de la placa ESP32 en Arduino IDE*. Obtenido de https://randomnerdtutorials.com/installing-the-esp32-board-in-arduino-ide-windows-instructions/.

Random Nerd Tutorials. (s.f.). *Introducción a la placa CAM Freenove ESP32-Wrover*. Obtenido de https://randomnerdtutorials.com/getting-started-freenove-esp32-wrover-

cam/?unapproved=837950&moderation-hash=c41d449c8ddbc0f2b84dd033177555aa#comment-837950.