INSTITUTO TECNOLÓGICO DE COSTA RICA

ÁREA ACADÉMICA DE INGENIERÍA EN COMPUTADORES

CURSO: CE-4202 TALLER DE DISEÑO ANALÓGICO

TÍTULO: PROYECTO 3: “SOLUCIONADOR CUBO RUBIK”

INTEGRANTES:

CALVO PORRAS ALEJANDRO
CHINCHILLA SÁNCHEZ ARTURO
FERNÁNDEZ FERNÁNDEZ CRISPTOFER
MUÑOZ ALVARADO ÉRICK ALEJANDRO

I SEMESTRE, 2019

Tabla de contenidos

1. INTRODUCCIÓN

De acuerdo con Slocum y colaboradores (2009), el cubo Rubik fue desarrollado por el escultor y profesor de arquitectura húngaro Erno Rubik en 1974. El cubo posee seis caras con colores distintos y posee mecanismo de ejes permite a cada cara girar independientemente, mezclando así los colores de este. El problema a resolver se basa en diseñar un circuito que esté en la posibilidad de analizar el estado inicial del cubo en términos de colores, tomar dicha posición inicial para alimentarla a un sistema de procesamiento de datos, y con base a ello calcular los movimientos requeridos para resolver el cubo.

Para ello se utilizará una placa Arduino y servomotores para poder rotar las caras del cubo, además para analizar el cubo, se realiza la implementación de un escáner usando una cámara y la librería OpenCV en Python 3 para obtener el estado del cubo. Usando esta información se obtiene la solución usando un algoritmo computacional que traduce la solución a pasos físicos y por último, estos son ejecutados por el circuito el cual resolverá el cubo Rubik.

2. DESCRIPCIÓN DEL PROBLEMA

El problema se basa en la resolución de el clásico cubo Rubik, para ello es necesario implementar un sistema o “robot” capaz de interactuar físicamente con el dispositivo, es decir, que sea capaz de rotar cada una de sus caras para que de esta forma, sea posible “armar el cubo. Además de esto, es necesario integrar un sistema que esté en la capacidad de analizar el cubo Rubik para poder digitalizar el estado de este, es decir, su distribución de colores, esto porque es necesario contar con dicha información para poder encontrar los movimientos que resuelven el sistema.

Para encontrar estos movimientos, se debe desarrollar un algoritmo computacional que encuentre la solución óptima y la traduzca en pasos físicos que el “robot” pueda interpretar y ejecutar. El obstáculo más importante que debe atravesarse es el diseño del sistema o “robot” que interactúa con el cubo, dado que debe considerarse parámetros como simplicidad y eficacia, es decir, que el sistema sea lo suficientemente simple de implementar pero lo más eficaz posible para resolver el problema en un tiempo prudente.

Además, debe considerarse el costo de la implementación de este, dado que para realizar el escáner y el algoritmo que resuelve el problema, puede recurrirse a elementos o herramientas que ya se cuentan como computadoras, cámaras web o inclusive la cámara de un teléfono celular para implementarlos, pero para el sistema físico como tal, es necesario realizar la compra de dichos componentes y los materiales para realizar la implementación del diseño.

3. MARCO TEÓRICO

Python

Citando a Van Rossum (2009), Python es un lenguaje de programación de alto nivel, poderoso y fácil de aprender. Cuenta con estructuras de datos eficientes y un enfoque simple pero efectivo a la programación orientada a objetos. La elegante sintaxis de Python y su tipado dinámico, junto con su naturaleza interpretada, hacen de éste un lenguaje ideal para scripting y desarrollo rápido de aplicaciones en diversas áreas y sobre la mayoría de las plataformas.
En este lenguaje de programación se implementará el algoritmo que resuelve el cubo, partiendo de la lectura de cada cara del cubo (mediante la biblioteca de OpenCV) y terminando con la serie de movimientos que se le deben realizar al cubo.

OpenCV

Rodríguez Bazaga (2015) menciona que OpenCV es una biblioteca de visión por computador de código abierto, la cual está escrita en los lenguajes C y C++ y es compatible con Linux, Windows y Mac OS X. Cuenta con un desarrollo activo en interfaces para Python, Ruby, Matlab y otros lenguajes. Además afirma que OpenCV ha sido diseñado para ser eficiente en cuanto a gasto de recursos computacionales y con un enfoque hacia las aplicaciones de tiempo real, donde uno de los objetivos es proveer una infraestructura de visión por computador fácil de utilizar que ayude a los programadores a desarrollar aplicaciones 'sotisficadas' de CV (Computer Vision) rápidamente.
OpenCV proporciona la manera de poder identificar una a una las caras del cubo (el color de cada cuadro que conforma la cara) y con ello poder comprender la realidad de cómo se encuentra el cubo al momento de querer resolverlo.

Arduino

Según Ingeniería MCI Ltda, "Arduino es una plataforma de desarrollo basada en una placa electrónica de hardware libre que incorpora un microcontrolador re-programable y una serie de pines hembra, los que permiten establecer conexiones entre el microcontrolador y los diferentes sensores y actuadores de una manera muy sencilla (principalmente con cables dupont)".
Una vez resuelto el algoritmo para armar el cubo, se le transmiten al arduino las instrucciones/movimientos que cada motor debe realizar a las caras del cubo para su solución.

Servomotor de rotación contínua

Este tipo de servomotor posee la gran característica de que no tiene limitación en su ángulo de giro o la cantidad de vueltas que puede dar, a diferencia de los convencionales que solamente tienen un ángulo de rotación de 180°; gracias a ello es ideal para proyectos de robótica. Además otra de sus principales ventajas es no necesitar un driver (puente h) externo, pues por ser un servo incluye su propio driver interno, facilitando a gran medida su uso e implementación en nuestros proyectos.

Algoritmo de Kociemba

Es el utilizado para resolver el cubo en Python, es el algoritmo más veloz y eficiente, gracias a la implementación en computadora. Esto se utiliza a través del paquete de rubik_solver.

4. DESCRIPCIÓN DE LA SOLUCIÓN

Se implementará un sistema basado en servomotores que serán manejados por un controlador ATmega328p en una placa arduino. El arduino controla los servomotores mediante la biblioteca Servo.h que envía una señal PWM a la entrada de control de los servomotores. El arduino recibe una secuencia de caracteres mediante su puerto serial que le indican los movimientos a realizar para resolver el cubo rubik. Esta cadena de caracteres son las iniciales en mayúsculas de los movimientos en inglés (Up, Down, Left, Right, Front, Back) y en caso de querer realizar un movimiento hacia el otro sentido se le indica con un apóstrofe (‘) luego de la inicial mayúscula (U’, D’, ...). Cada movimiento de la cadena se separa con guiones (-). Un ejemplo de una cadena de movimientos es: “U-L-L-D’-F’”. Para el sistema que escaneado del estado del cubo se implementa utilizando python 3, OpenCV 4.1.0 y la cámara del teléfono celular, que empleando la aplicación IP Webcamera que transforma el teléfono en una especie de “Servidor” local en una red wifi que puede ser consumido mediante peticiones, este provee la imagen que es procesada por un algoritmo realizado en Python que toma esta imagen y le aplica una serie de filtros para poder extraer los colores de cada cara del cubo y crear una lista de matrices, una matriz por cada cara.

Esta información será utilizada por el algoritmo que traducirá este estado a una solución basada en pasos capaz de ser leída y ejecutada por el Arduino UNO y traducida a movimientos físicos ejecutados por los servomotores. Como parte de la estructura, se creará una diseño basado en acrilico para crear la estructura que sostendrá el cubo, servomotores y demás.

5. DIAGRAMA DETALLADO DE LA SOLUCIÓN

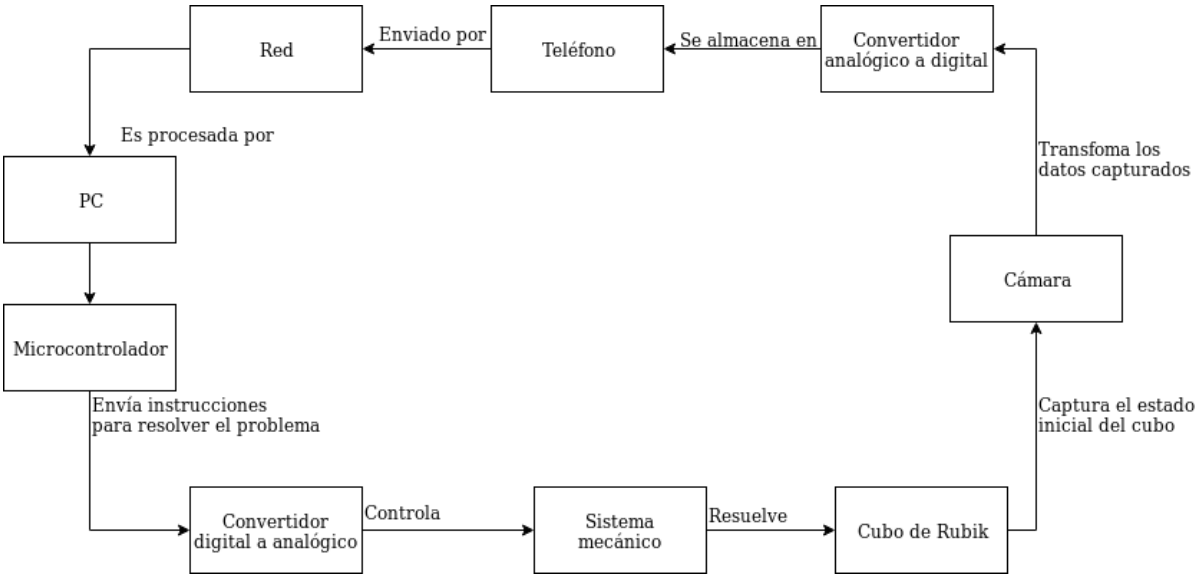

Figura 1. Diagrama de bloques de la solución.

El sistema mecánico se espera que quede similar a la figura 2, sin embargo los puntos de apoyo del cubo no serán iguales a los de la imagen sino más bien estarán ubicados en las piezas del centro, adaptados con acoples impresos en 3D.

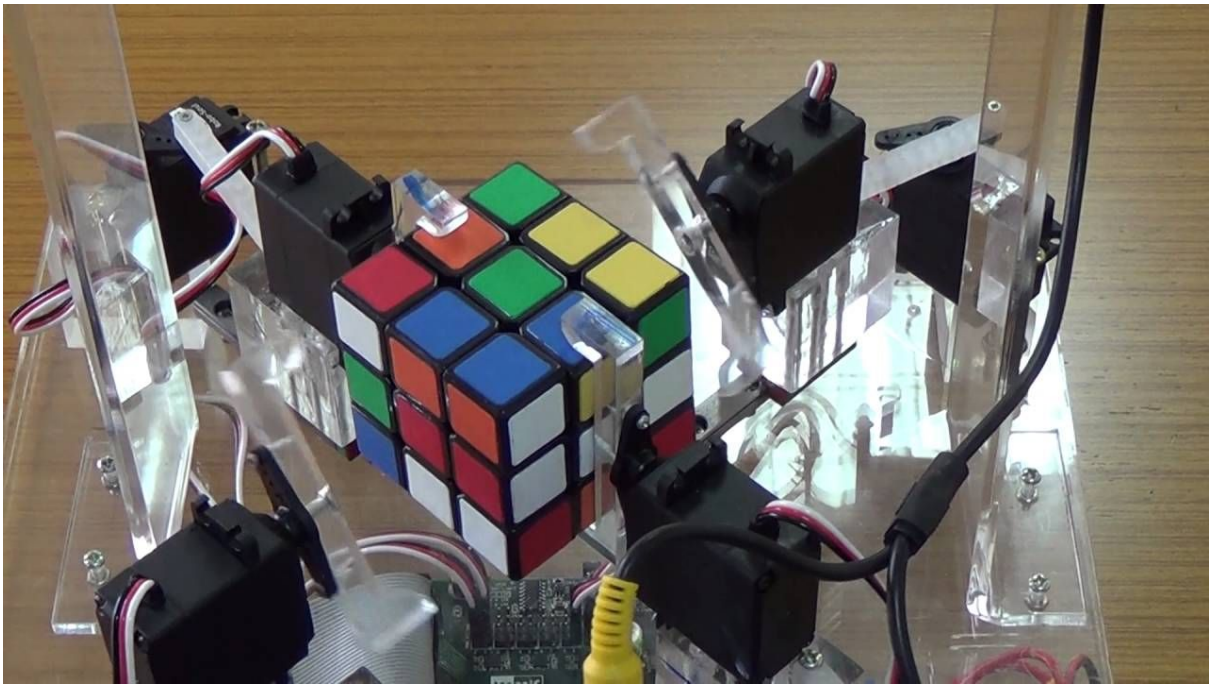

Figura 2. Estructura ejemplo para la construcción del sistema que resuelve el cubo.

Como microcontrolador se espera utilizar el ATMega 328P y si visualizará muy similar a la siguiente imagen:

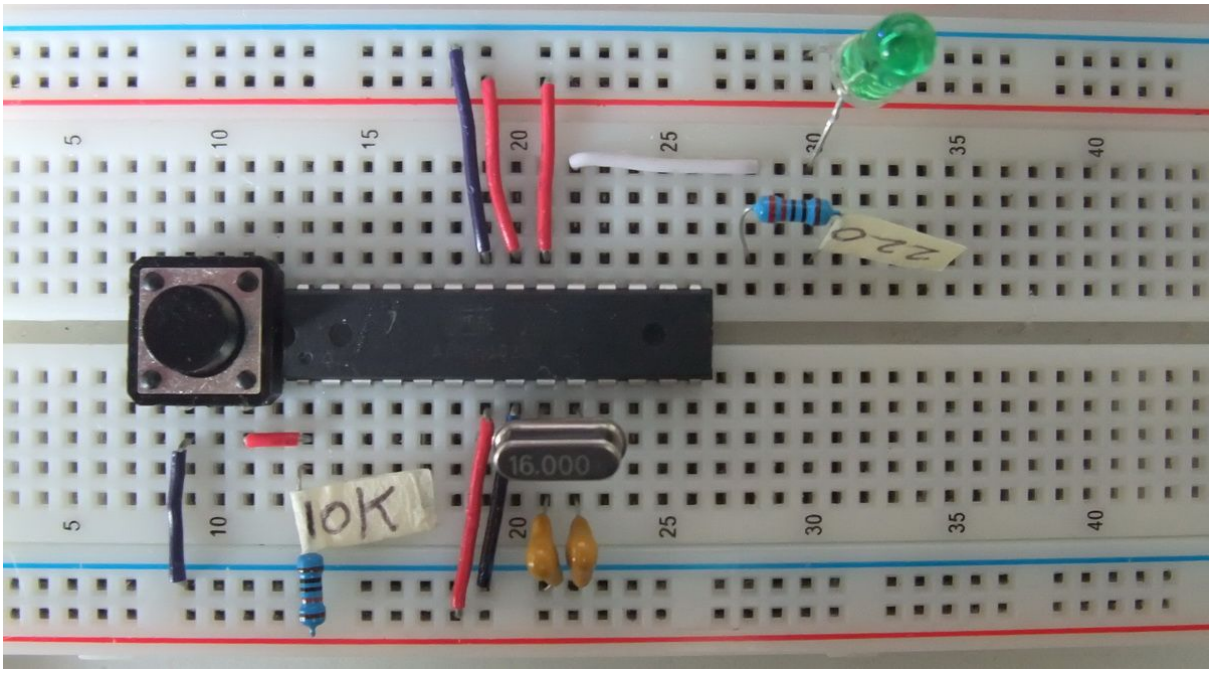

Figura 3. Ejemplo de un microcontrolador ATmega328P.

El teléfono podría ser cualquier smartphone moderno y el sistema de conversión A/D y D/A se encuentra en etapa de diseño por lo que no es posible representar el circuito que se espera utilizar para la implementación del proyecto.

6.Resultados obtenidos

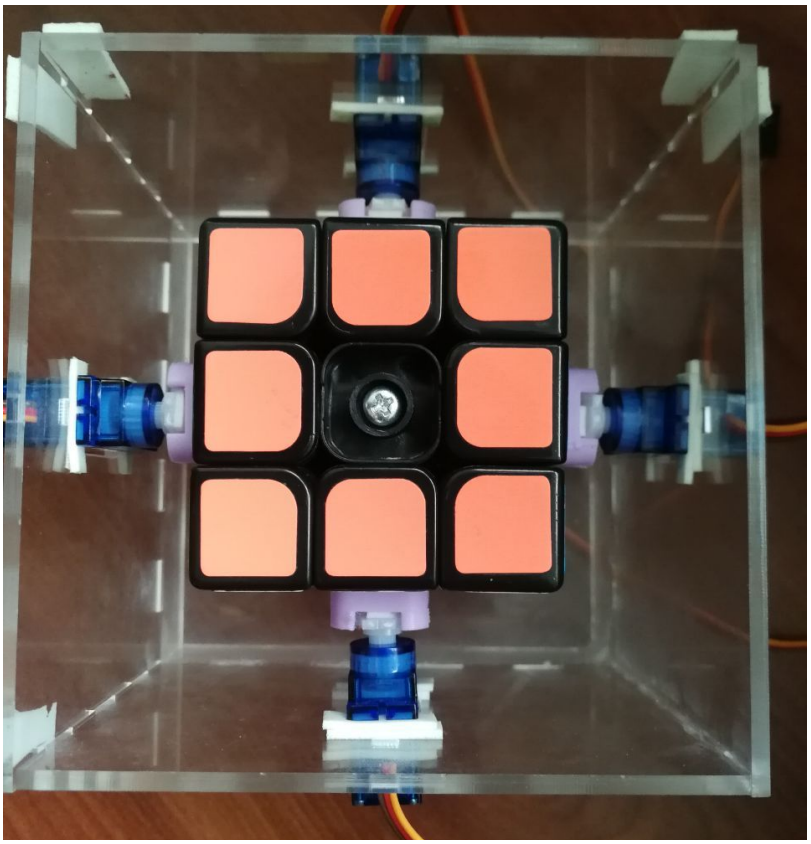

Figura 4. Vista superior de la estructura para resolver el cubo.

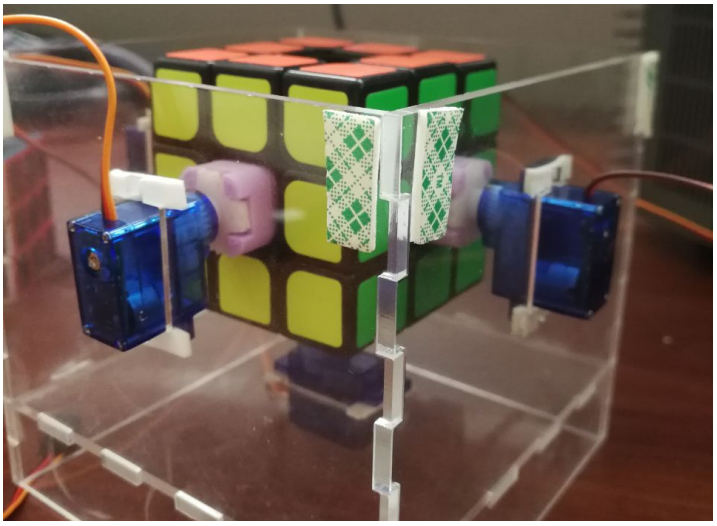

Figura 4. Vista lateral de la estructura para resolver el cubo.

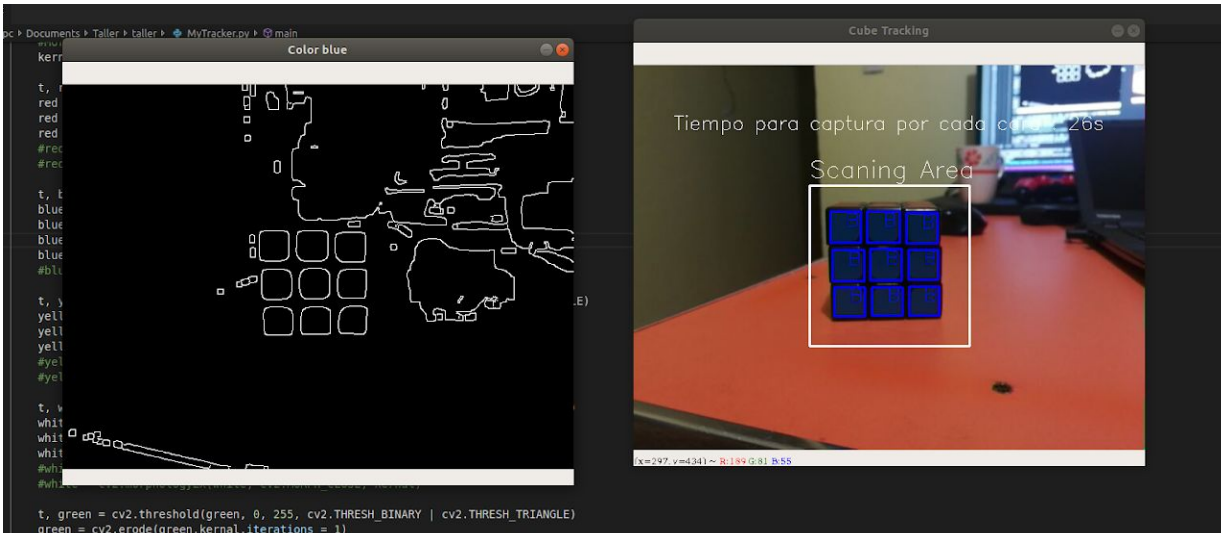

Figura 5. Lectura de las caras del cubo

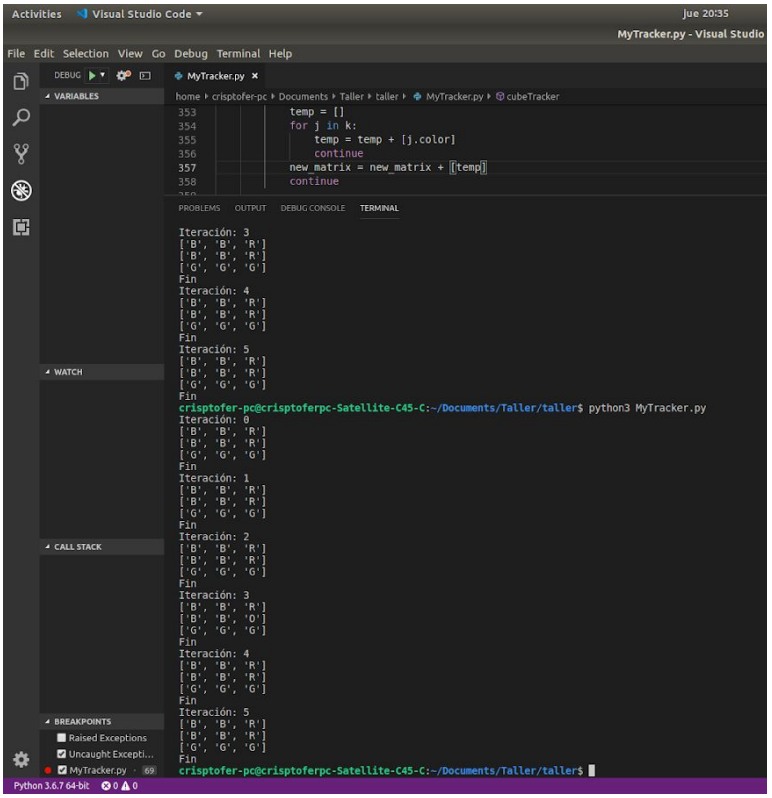

Figura 6. Creación de las matrices correspondientes a las caras del cubo

El servo motor FS90R no cuenta con la fuerza suficiente para mover una cara del cubo en caso de que el mismo no esté en la posición adecuada (trabado) provocando que al quedarse pegado el cubo/servo no se logre solucionar el cubo.
En caso de que el cubo se quede pegado se le debe dar una pequeña ayuda con la mano.

7. Análisis de los resultados

Para la solución del problema se tiene que hacer uso de servomotores de 360°, esto es por la naturaleza de los movimientos para resolver el cubo rubik, ya que eventualmente se tienen que hacer movimientos que no se pueden realizar con servomotores de 180°. Lo malo los servos de 360° es que estos no poseen un encoder que permite saber dónde se encuentra el servomotor, esto implica que se tiene que realizar los movimientos en un tiempo dado y si el servomotor, por alguna razón se traba o el cubo no está acomodado de correcta manera para realizar el movimiento en el tiempo establecido, el servomotor no se entera y piensa que sí realizó el movimiento. Por lo tanto esto hace que el cubo no dé los giros de manera correcta siempre y hace que se interrumpa la secuencia lógica de los movimientos para la resolución del mismo.

Para el sensado de los colores del cubo es necesario reconsiderar el método o optimizarlo de alguna forma, dado que cualquier cambio en el ambiente o la luz puede afectar en el reconocimiento de los colores. Esto también se produce porque en ambientes cerrados la luz proviene de una lámpara o bombillo y produce que se sature la luz en el objeto desde una sola dirección, lo que ocasiona que la cámara no tome los colores de la forma correcta.

Se utilizó acrílico para formar una caja como estructura para sostener los servomotores y el cubo, como se muestran en las Figuras 3 y 4. Sin embargo, al ser una caja cerrada, se dificulta la lectura y el ingreso y extracción del cubo en el sistema.
Además se utilizó impresión en 3D para crear las uniones entre los servomotores y el cubo.

8. Conclusiones y recomendaciones

Utilizando el método elegido, la extracción del estado del cubo es un poco tediosa, dado que los cambios en la intensidad de la luz o el entorno puede influir mucho en el resultado.
Se debe considerar emplear servos servos con encoder o algún tipo de sensado para asegurar que se realizan los movimientos.
Se debe considerar la fuerza de los servomotores o steppers para la construcción de un sistema mecánico, dado que si no cuentan con la fuerza suficiente, cabe la posibilidad que no sea capaz de mover las caras del cubo.
Es necesario analizar el consumo de corriente de los dispositivos a emplear, dado que dependiendo de esta, se debe elegir la fuente de alimentación.

La utilización de un sistema cerrado para la resolución del cubo, dificulta la lectura del mismo, debe considerarse diseños que permitan esto sin tener que retirar el cubo de la estructura.

9. BIBLIOGRAFÍA

Slocum, J., Singmaster, D., Huang, W. H., Gebhardt, D., & Hellings, G. (2009). The Cube: The Ultimate Guide to the World's Bestselling Puzzle—Secrets, Stories, Solutions. *Black Dog & Leventhal Publishers*.

Ingeniería MCI Ltda. ¿Que es Arduino? ~ Arduino.cl - Plataforma Open Source para el desarrollo de prototipos electrónicos", [Online]. Available: http://arduino.cl/quees-arduino/. [Accessed: 24- Mar- 2019].

Van Rossum, G. (2009). El tutorial de Python [PDF]. Retrieved from http://docs.python.org.ar/tutorial/pdfs/TutorialPython2.pdf

Rodríguez Bazaga, A. (2015). OpenCV: Librería de Visión por Computador - Oficina de Software Libre (OSL). Retrieved from https://osl.ull.es/software-libre/opencv-libreria-vision-computador/