UNIVERSIDAD AUTÓNOMA DE CIUDAD JUÁREZ

INSTITUTO DE INGENIERÍA Y TECNOLOGÍA

DEPARTAMENTO DE INGENIERÍA INDUSTRIAL Y MANUFACTURA,

"Sistema de servo-visión de 2 grados de libertad."

Visión

Francesco García Luna

Alumnos

Martín Iván Martínez Vela-180000

Maximiliano García Carreón-180026

Edgar Irán García Ojeda-180032

Fernando Durán Romero-180079

Allan Daniel González Antonio-180083

Ciudad Juárez, Chihuahua. 4 Mayo 2022

Índice general

Índice de figuras

Capítulo 1

Introducción.

En este presente documento se mostrarán los resultados del desarrollo de un algoritmo para apuntar al centro de un ArUco mediante un láser, así como la metodología empleada. El documento se encuentra seccionado por antecedentes, planteamiento del problema, objetivos, marco teórico, metodología, diagrama de flujo, resultados y conclusión.

1.1. Antecedentes.

En el articulo [1] nos explica que la visión por computadora es aquel conjunto de herramientas y métodos que permiten obtener, procesar y analizar imágenes del mundo real con la finalidad de que puedan ser tratadas por un ordenador.

Una de las primeras aplicaciones de las imágenes digitales aprecio en la industria de la prensa. Dichas imágenes eran enviadas a través de un cable submarino entre Londres y Nueva York. De ahí los equipos se encargaban de codificar las imágenes para transmitirla y luego se reconstruían al otro extremo. En un inicio los trabajos para mejorar la calidad visual de las imágenes se concentraban principalmente en la selección de los métodos de impresión y la distribución de los niveles de intensidad.

Para el procesamiento de imágenes se ha desarrollado bibliotecas referentes a la visión artificial, la más utilizada y popular hasta la fecha es la biblioteca abierta conocida como OpenCV que significa .ºpen Computer Vision (visión artificial abierta)". Es utilizada en la detección de movimiento, reconocimiento de objetos, reconstrucción 3D a partir de imágenes, estas son algunos ejemplos de sus aplicaciones. Si bien OpenCV está desarrollado en C++ también está disponible en el lenguaje de programación de Python.

1.2. Planteamiento del problema.

La precisión es una cualidad muy importante al momento de realizar tareas en la cuales el rango de error es muy pequeño, un ejemplo de este tipo de tareas se puede encontrar en el área

médica. Al momento de realizar una operación de retina por medio de rayo láser, en donde la falta de precisión puede afectar de manera negativa el resultado.

1.3. Objetivo.

Desarrollar un algoritmo de servo visión para un robot de dos grados de libertad.

1.3.1. Objetivos específicos.

- Obtener parámetros de la cámara.
- Construir estructura del robot manipulador.
- Integrar cámara en el robot.
- Integrar láser.
- Obtener marco de referencia del láser.
- Desarrollar algoritmo de lector serial, decodificación de datos e escritura en Arduino.
- Desarrollar algoritmo de control PI en Python
- Comunicar Python y Arduino.
- Validar algoritmo.

Capítulo 2

Marco Teórico.

2.1. Marcador ArUco

Un marcador ArUco es un identificador único compuesto por un borde y una matriz binaria que se utiliza como un sistema de referencia, el cual permite conocer la posición y orientación utilizando la librería de OpenCV. La codificación binaria permite su identificación y la aplicación de técnicas de detección de errores.

El tamaño del marcador determina el tamaño de la matriz interna. Los distintos tipos de arucos contienen matrices de 4x4, 5x5, 6x6, 7x7 y funcionan con un diccionario que son un conjunto de marcadores para una aplicación específica para cada uno de los valores de las matrices.

2.2. Controlador PI

Los controladores automáticos comparan el valor real de la salida de una planta con la entrada de referencia, es decir el valor deseado, determinan el error o desviación y producen una señal de control que reducirá el error de control a cero o a un valor pequeño. una de las representaciones para el controlador PI en función del tiempo esta dada por la ecuación 2.1.

$$u(t) = Kpe(t) + \frac{1}{t_i}\int_t^0 e(t)dt \tag{2.1}$$

2.3. Cámara

Es un dispositivo encargado principalmente de tomar la imagen de seguimiento, Esta diseñada para realizar el seguimiento de color. Dicha cámara se puede usar para la localización de líneas de referencia, sigue líneas, o el seguimiento de una luz en movimiento.

2.4. Servomotor

Es un actuador rotativo o motor eléctrico, se define básicamente por una regulabilidad extrema y sus correspondientes posibilidades de control. Por lo tanto, permite el control preciso

de posición de ángulo, aceleración y velocidad. La estructura básica del servomotor incluye un sensor de posición del rotor.

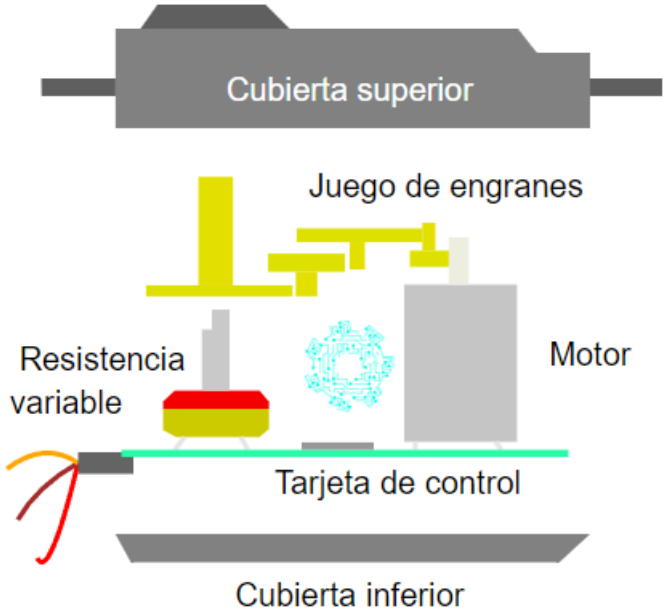

Figura 2.1: Estructura de un servomotor MG995.
Imagen tomada de Mecatrónica LATAM.

El servomotor cuenta con tres terminales de conexión:

- VCC: el cual corresponde al cable de alimentación.
- GND: corresponde al cable de tierra.
- Señal: corresponde al cable de entrada de señal (PWM)

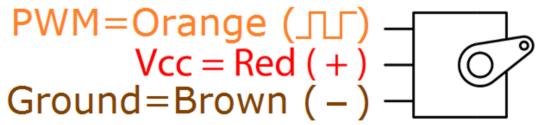

Figura 2.2: Diagrama de conexiones de servomotor MG995.
Imagen tomada de Electrónicos Caldas.

La tensión de alimentación de los servomotores suele ser de 4 a 8 voltios, en nuestro caso, lo servos modelo MG995 utilizan una tensión de 4.8 a 7.2, el control del servomotor consiste básicamente en indicarle la posición en la que se debe situar.

2.5. Sistema de control en lazo cerrado

Donde u(t) es la variable de control y e es el error de control. El controlador PI. es la suma de tres términos donde tenemos que el termino (P) que es proporcional al error, el termino (I) que es proporcional a la integral del error.

Los parámetros del controlador son ganancia proporcional K_p, tiempo integral representado por T_i.

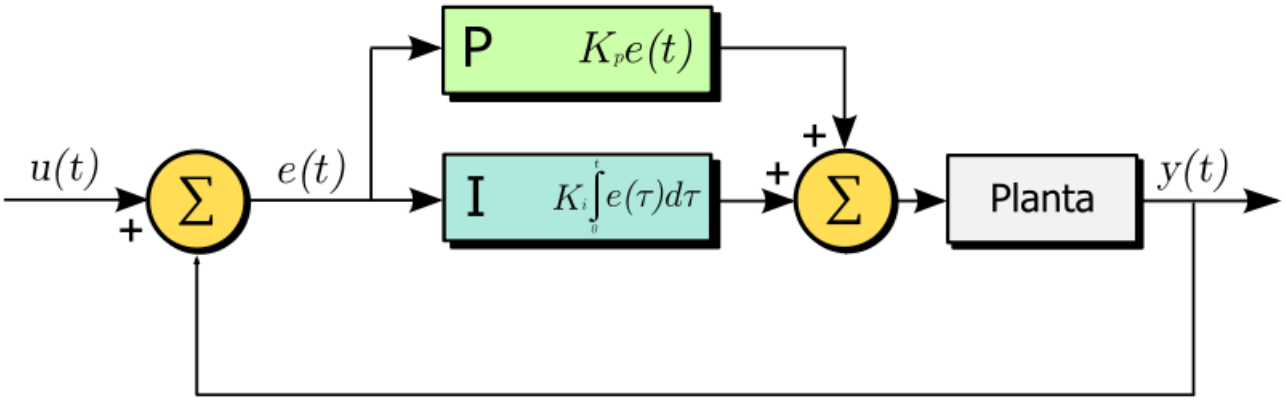

Figura 2.3: Representación del algoritmo proporcional integral.

2.6. OpenCV.

Biblioteca de código abierto para aplicaciones que utilicen visión artificial, es utilizada en varias aplicaciones como control de procesos, sistemas de seguridad con detección de movimiento, reconocimiento de objetos, robótica avanzada, etc.

Esta librería cuenta con una buena eficiencia computacional y un fuerte enfoque en aplicaciones de tiempo real. Opencv contiene también varias funciones utilizadas en distintas áreas de la visión por computadora.

Algunos ejemplos de sus aplicaciones son:

- Identificación de objetos
- Reconocimiento de rostros
- Interfaces de usuario
- Representación 3D.

2.7. Matriz de rotación.

Matriz que representa la rotación en grados del plano en sentido antihorario. La orientación de un punto puede expresarse con los tres ángulos de Euler que corresponden a la rotación

respecto a los ejes x, y, z.

La rotación en cada eje puede expresarse como una matriz para cada ángulo, como se muestra a continuación:

$$Rz(\psi) = \begin{bmatrix} cos(\psi) & -sin(\psi) & 0 & 0 \\ sin(\psi) & cos(\psi) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

$$Ry(\theta) = \begin{bmatrix} cos(\theta) & 0 & sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ -sin(\theta & 0 & cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

$$Rx(\phi) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\phi) & -sin(\phi) & 0 \\ 0 & sin(\phi) & cos(\phi) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

Capítulo 3

Metodología.

Para desarrollo de este proyecto tuvieron que realizar los siguientes pasos.

Primero se tuvo que calibrar la cámara que se iba a utilizar, posteriormente se monto la cámara en el robot de 2 grados de libertad y también se le implemento un láser. Una vez hecho esto, utilizamos un arduino como driver para controlar los movimientos del robot y el estado del láser.

Establecemos una comunicación serial entre Python y el Arduino para poder enviar la señal de control que se calcula en Python y que el Arduino la reciba para así poder mover los servomotores y láser. Para calcular la señal de control usamos un control PI (Proporcional e Integral), obteniendo el error en relación de la distancia que se encuentre el centro del ArUco al centro del frame que captura la cámara.

Para obtener el error, debemos de calcular el error proporcional y el error integral. Para obtener el error proporcional, obtenemos la diferencia de píxeles entre el centro de la imagen y el centro del ArUco, con la siguiente ecuación (3.1):

$$Ep = E * kp \tag{3.1}$$

Donde E es el error.

Una vez obtenido esto, calculamos el error integral, el cual es el área bajo la curva multiplicada por una ganancia integral. Para conseguir el área bajo la curva, lo hacemos con esta ecuación (3.2):

$$AreaBC = AreaBC + E \tag{3.2}$$

Una vez obtenemos el error integral, lo multiplicamos por una ganancia integral (3.3).

$$Ei = AreaBC * ki \tag{3.3}$$

Después de calcular el error proporcional y el error integral, ya sacar la señal de control con la ecuación (3.4).

$$Señal = Ep + Ei \tag{3.4}$$

3.1. Diagrama de flujo.

En la figura 3.1 se muestra el diagrama de flujo que se utilizó para desarrollar nuestro algoritmo en Python:

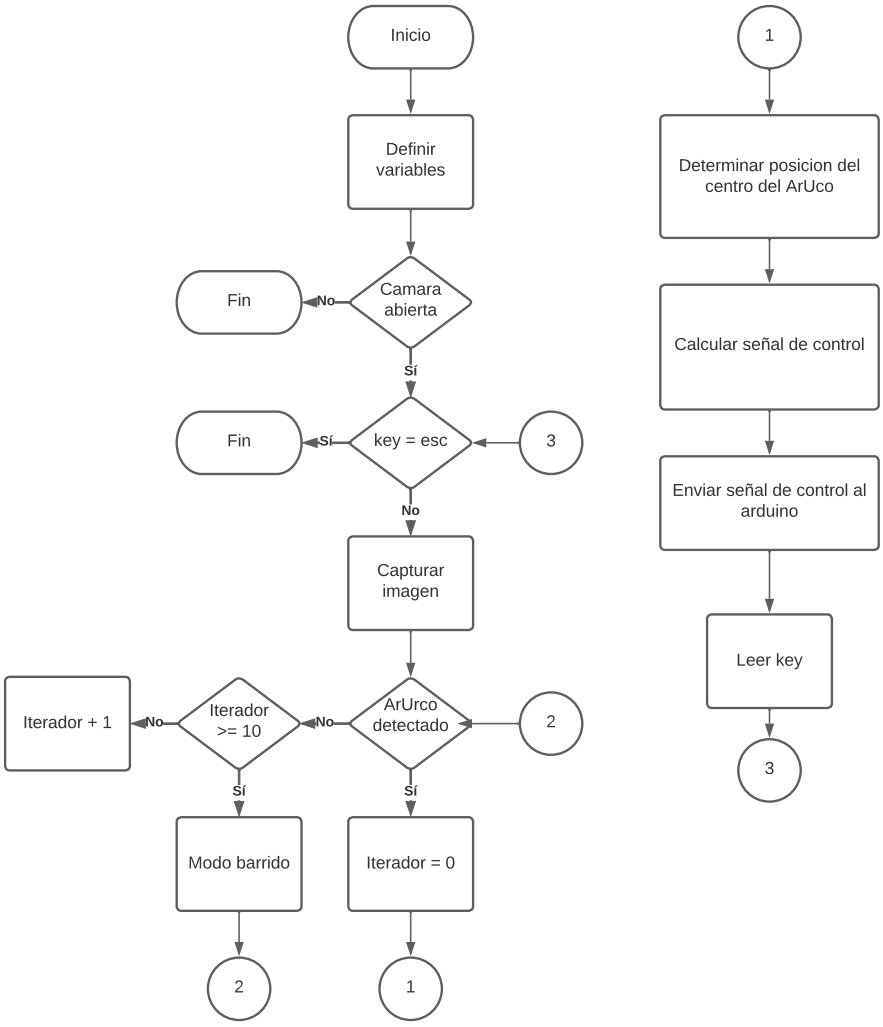

Figura 3.1: Diagrama de flujo de algoritmo de Python.

En la figura 3.2 se muestra el diagrama de flujo que se utilizó para desarrollar nuestro algoritmo en Arduino:

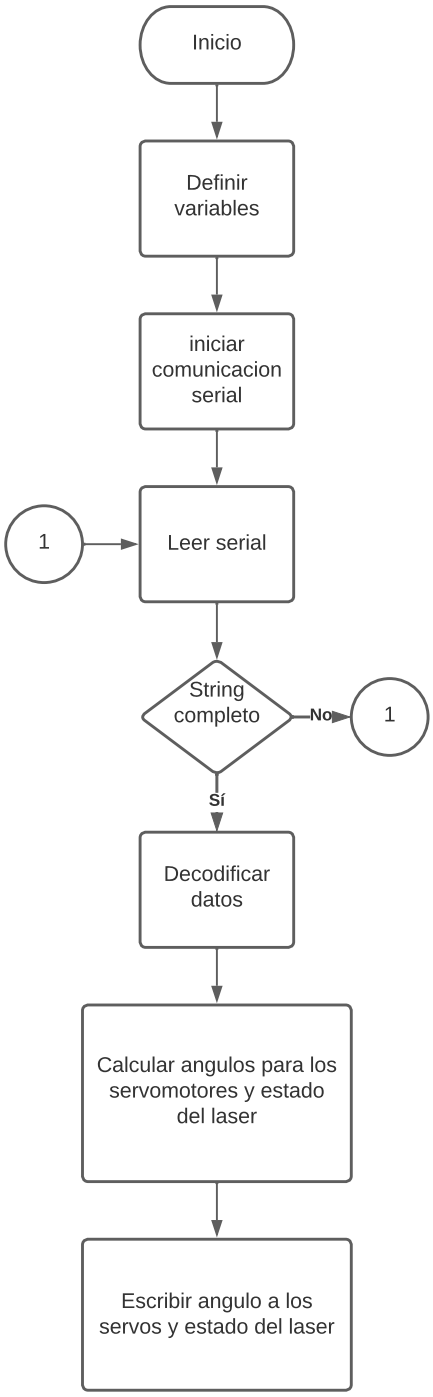

Figura 3.2: Diagrama de flujo de algoritmo de Arduino.

3.2. Modelo 3D del sistema de servo-visión.

En la siguiente figura se puede observar el modelo 3D del brazo manipulador de dos grados de libertad, donde encima de este, cuenta con una cámara y encima de este con un apuntador láser.

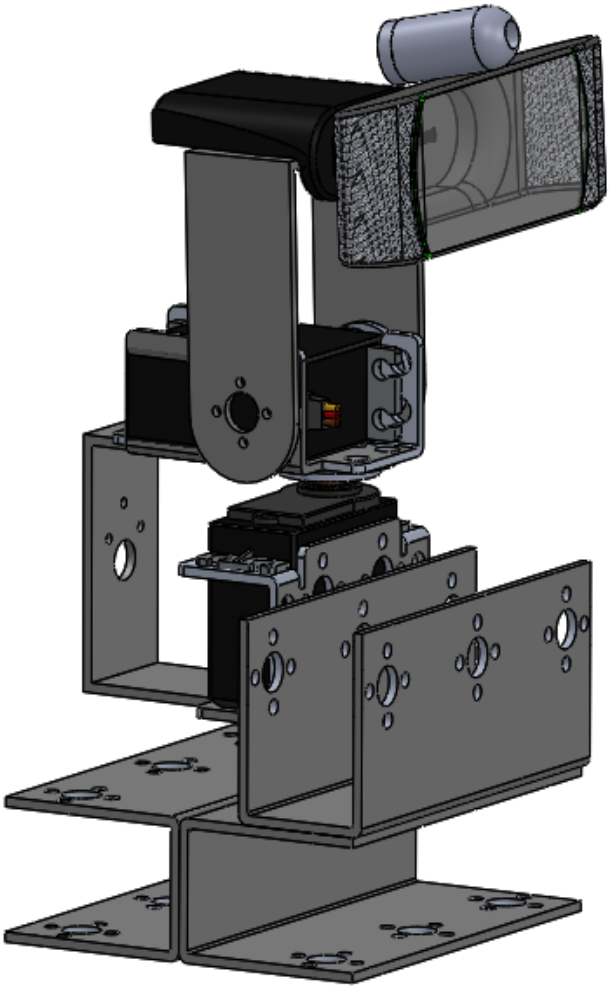

Figura 3.3: Modelo 3D de robot, cámara y láser.

3.3. Circuito Eléctrico del sistema de servo-visión.

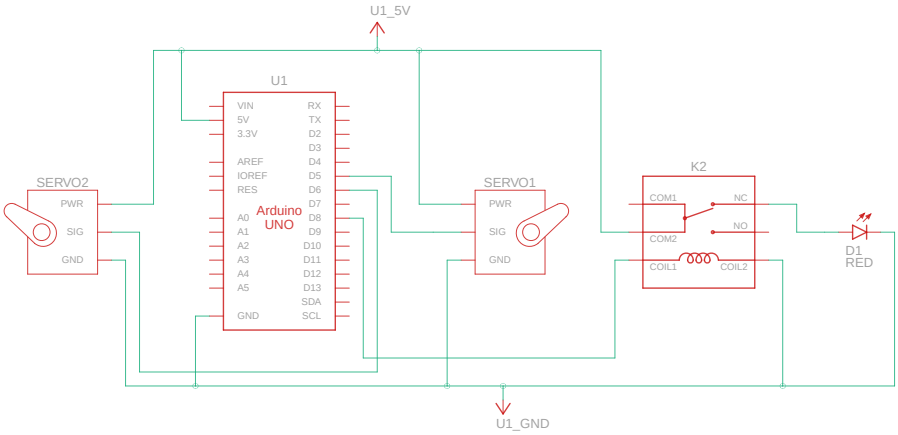

Figura 3.4: Circuito eléctrico.

En la fig. (3.4), se muestra el circuito eléctrico que se desarrolló en este proyecto. En el circuito se muestran 2 servomotores, uno de ellos está conectado al Pin 5 y el otro al Pin 6 del Arduino, los pines de alimentación de los servomotores se conectaron a 5 volts. También como elemento de salida se utilizó un láser, para hacer un poco más complejo el circuito se utilizó un relevador este se encarga de encender el láser cada vez que se le mande una señal de control al solenoide del relevador.

3.4. Resultados

Como resultado obtuvimos un sistema que logra apuntar hacia un ArUco de manera satisfactoria, corrigiendo su posición cada vez que el error sale de la tolerancia. En la figura 3.5, observamos la imagen capturada y procesada por el sistema de servo-visión.

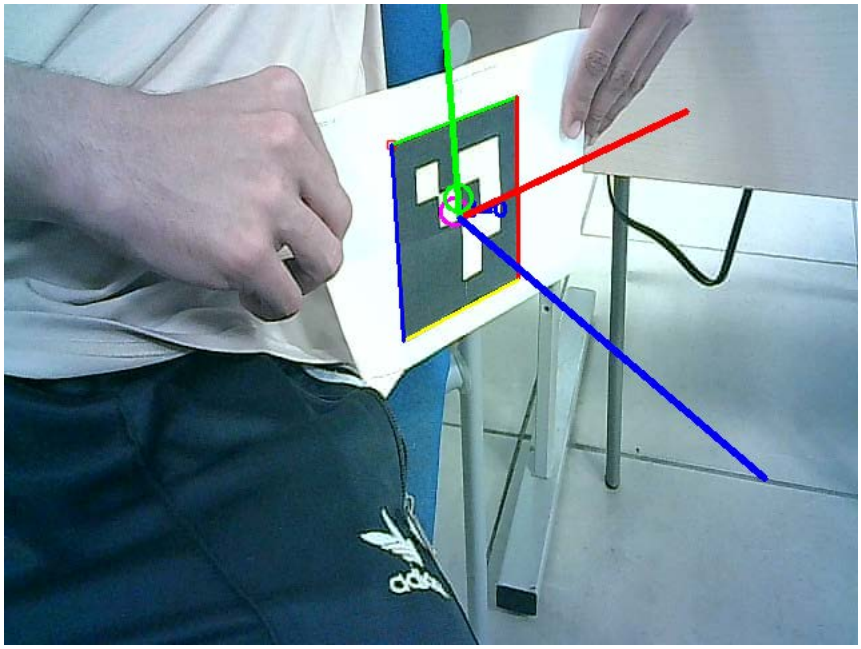

Figura 3.5: Imagen capturada por la cámara del sistema.

En la figura 3.6, podemos observar la gráfica del error que genera este sistema, observamos como al principio como tiene un overshoot debido al gran tamaño del error, pero conforme el sistema trabaja, este mismo se va a estabilizar poco a poco.

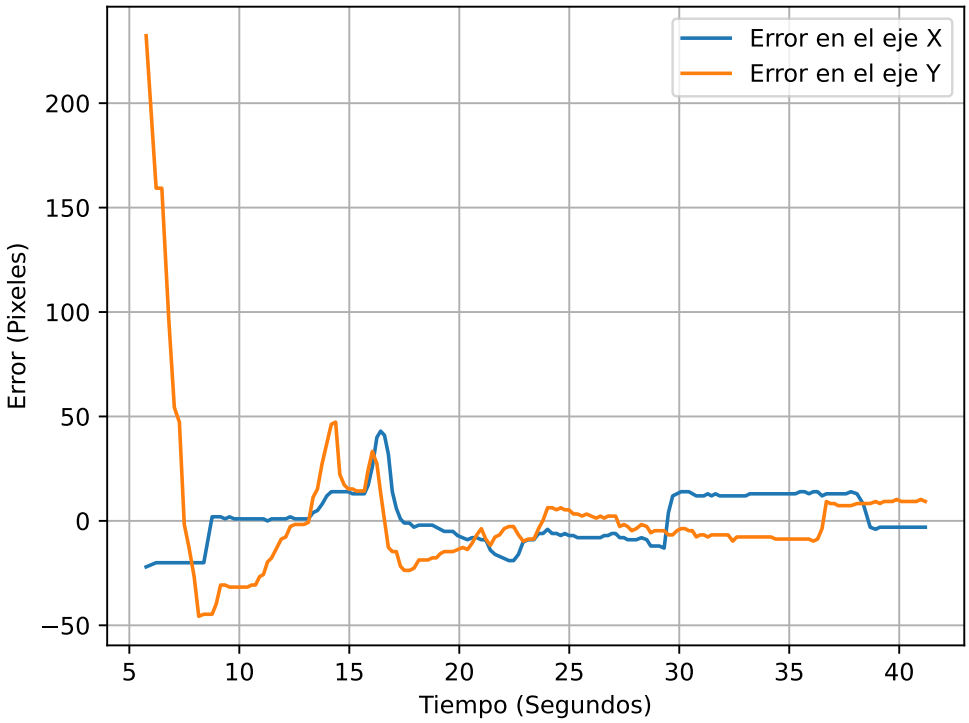

Figura 3.6: Gráfica que representa la corrección del error.

Podemos ver dos lineas independientes, cada linea representa el error de cada grado de libertad del robot.

Capítulo 4

Conclusión.

El sistema cumple con la tarea que fue definida, pero la precisión con la que trabaja el sistema depende de muchos factores, como la iluminación, la calidad de la cámara, la posición en donde se encuentra el ArUco, la resolución de los servomotores, etc. Es por ello que un trabajo a futuro, se buscaría de trabajar en los factores que afectan al sistema para que ofrezca una mejor precisión y mejorar el sistema de control agregando el control derivativo.

Capítulo 5

Referencias.

[1] About-OpenCV" . OpenCV (8 Oct 2021). [online], disponible: https://opencv.org/about/

[2] R. Jiménez, "Sistema de seguimiento de objetos usando OpenCv, ArUco y Filtro de Kalman extendido", tesis, Universidad de Sevilla, SE, España, 2018.

[3] J. N. A. Rivero, «Robot Dirigido Por Voz,» Universidad de las Américas Puebla, Puebla, 2007.

[4] E. A. Prieto y P. A. G. Quintero, «Diseño e implementación de un robot móvil autónomo con detección de color,» Universidad De San Buenaventura, Bogotá, 2006.

[5] G. Viera-Maza, «Procesamiento de imágenes usando OpenCV aplicado en Raspberry PI para la clasificación del cacao,» Universidad De Piura, Piura, 2017.

[6] USON, «USON,» 2010. [En línea]. Available: http://tesis.uson.mx/digital/tesis/docs/22170/Capitulo2.pdf. [Último acceso: 6 Abril 2022].

[7] OpenCV, «OpenCV,» 6 Abril 2022. [En línea]. Available: https://docs.opencv.org/4.x/d5/dae/tutorial$_a ruco_d etection.html.[Últimoacceso : 6Abril2022]$.

[8] A. E. Benavides, «Sistema de control dinámico del manipulador de un robot industrial en empresas troqueladoras de 4 grados de libertad,» Universidad Rafael Belloso Chacín, Maracaibo, 2013.

[9] M. E. P. Chin, «Desarrollo de un sistema de visión computacional para el vuelo autónomo de un Vehículo Aéreo No Tripulado,» Universidad Autónoma de Yucatán, Mérida, 2019.