

**Universidad Autónoma del Estado de México**

**Unidad Académica Profesional Tianguistenco**

**Ingeniería en software**

**Unidad de aprendizaje:**

**Técnicas y métodos de procesamiento de imágenes**

**Profesor:**

Rocio Elizabeth Pulido Alba

**Alumno:**

Saavedra Caballero Roberto Daniel

**Fecha de entrega:** 10/02/2023

|  |
| --- |
|  |

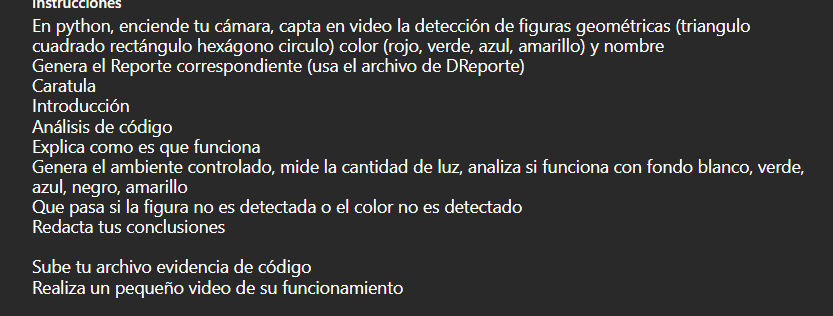
**Técnicas y Métodos de procesamiento de imágenes**

|  |  |  |  |
| --- | --- | --- | --- |
| Practica No # 5 | | Febrero Julio | 13/02/2023 |
| Título de la Práctica: Cámaras figuras y color | | 2023A | Fecha de Elaboración |
| Desarrollada por | | | |
| No. Control | Nombre del (los) Alumno(s) | | |
| 1645282 | Saavedra Caballero Roberto Daniel | | |

**Introducción del tema tratado**

**Las cámaras pueden ser accedidas y controladas en Python utilizando diferentes bibliotecas. Una de las bibliotecas más comunes es OpenCV (Open Source Computer Vision Library). OpenCV es una biblioteca de código abierto que proporciona una amplia variedad de algoritmos para procesamiento de imágenes y vídeo, incluido el acceso a cámaras web y dispositivos de captura de video.**

|  |
| --- |
| Definición de Problema |
| Codificar en Python para reconocer imágenes y colores |
| Objetivo General |
| Reconocer figuras y colores con Cámara |
| Objetivos Específicos |
| Reconocer figuras y colores con cámara de código en Python |

****

**Este código utiliza la biblioteca OpenCV (cv2) para capturar imágenes en tiempo real desde una cámara y detectar formas en las imágenes capturadas.**

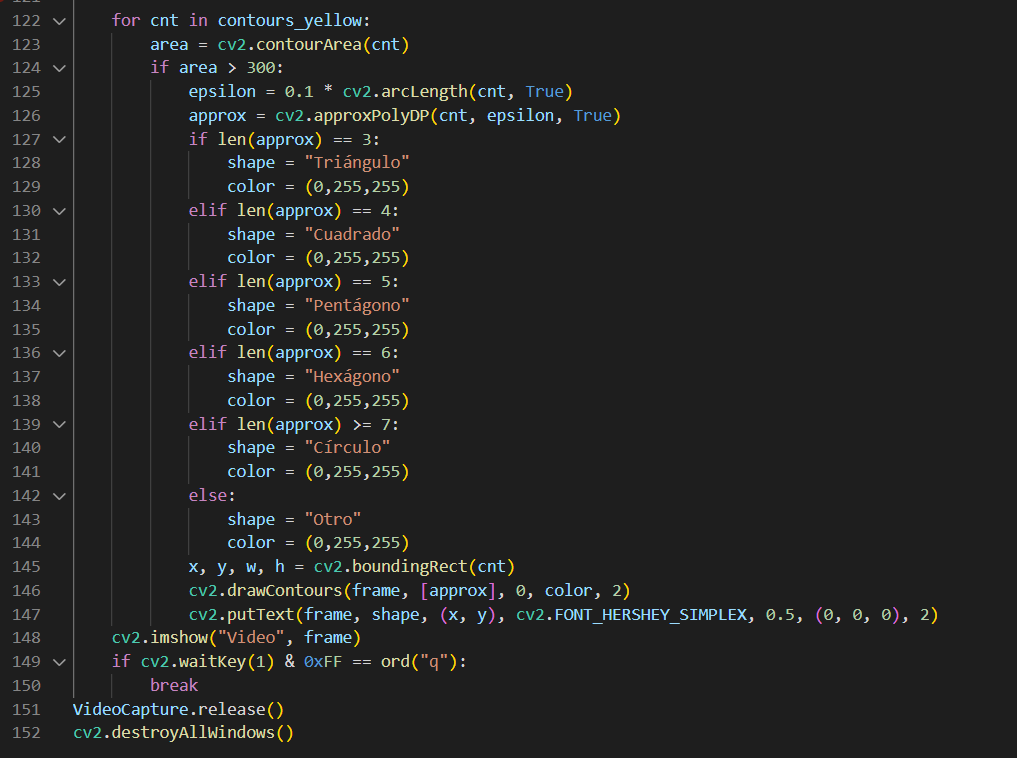
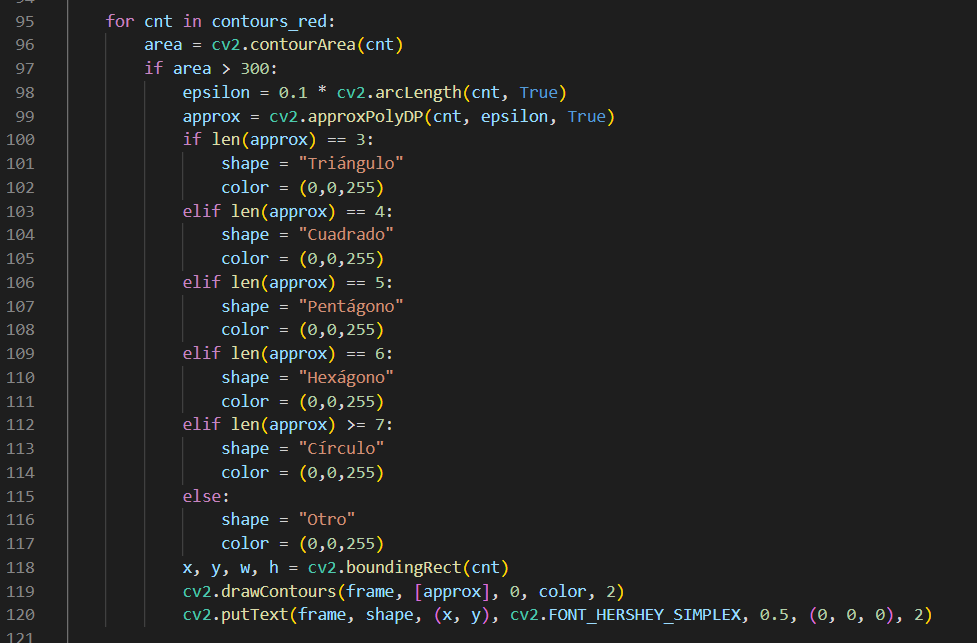
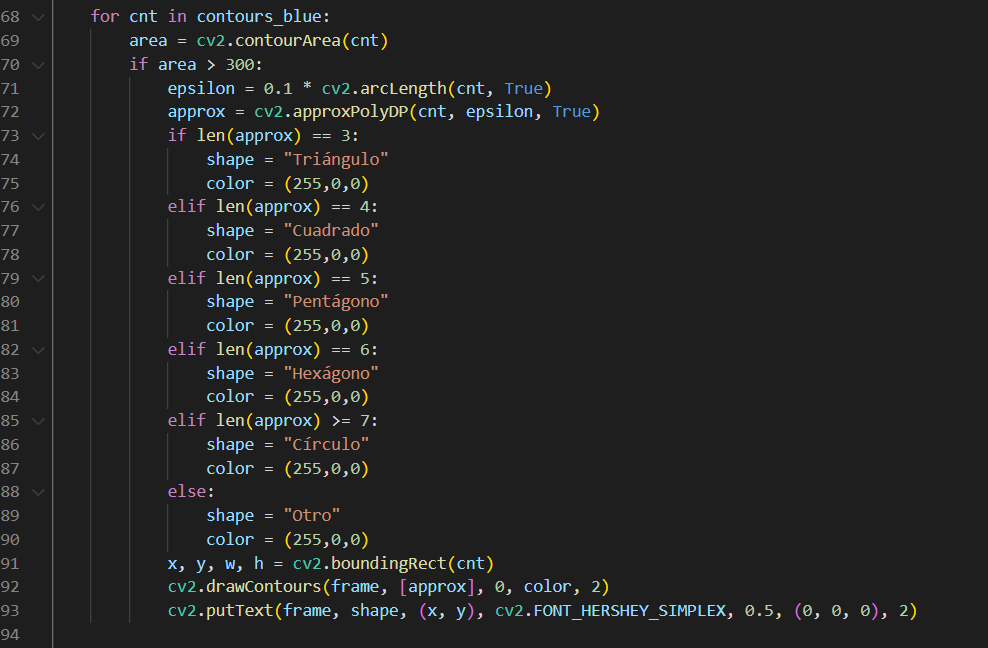
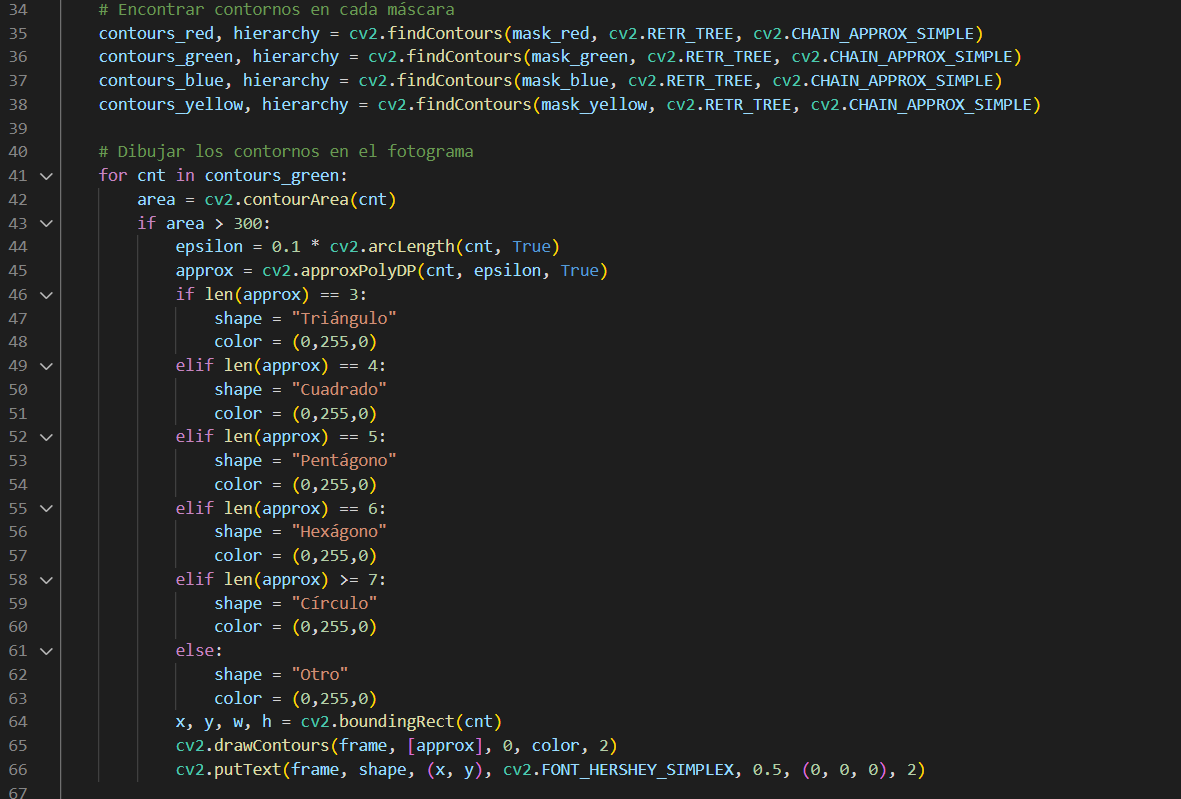
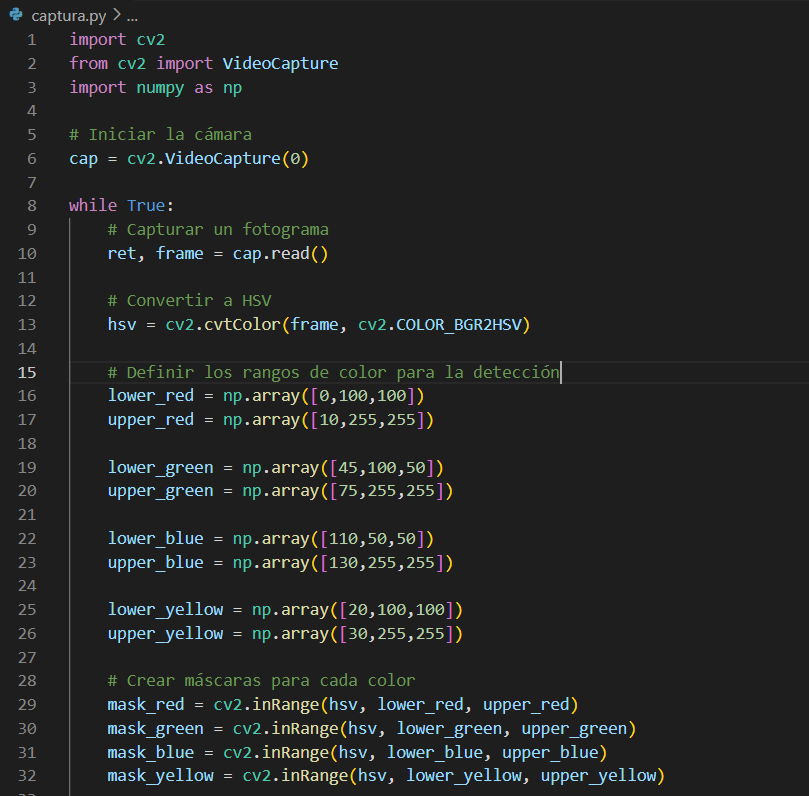
**La cámara se inicia con el siguiente comando: "cap = cv2.VideoCapture(0)". Luego, en el bucle "while True", se captura un fotograma de la cámara con "ret, frame = cap.read()".**

**A continuación, se convierte el fotograma a un formato HSV (Hue, Saturation, Value), que es un formato de representación de color que es más adecuado para la detección de colores.**

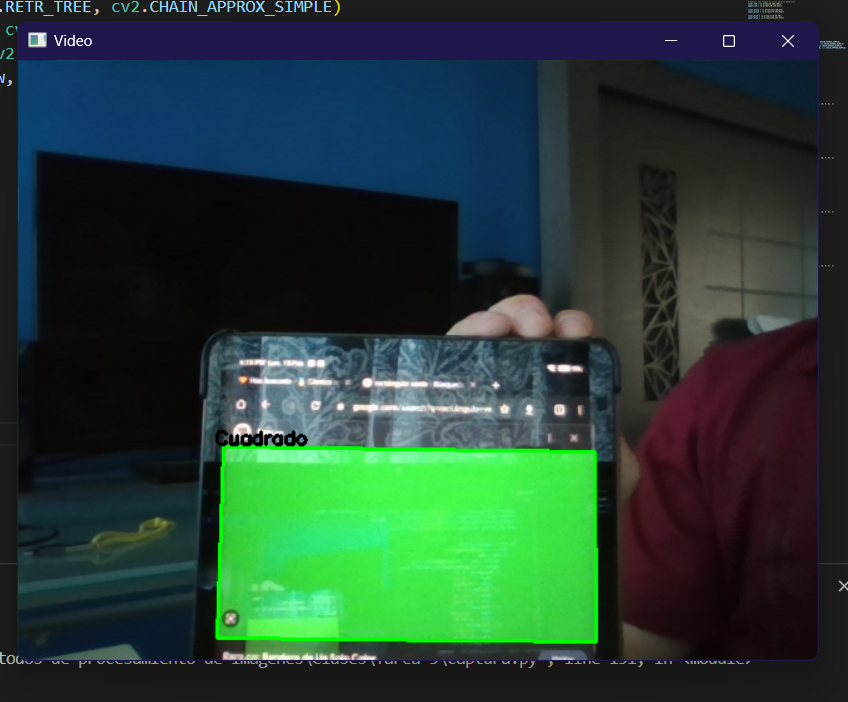
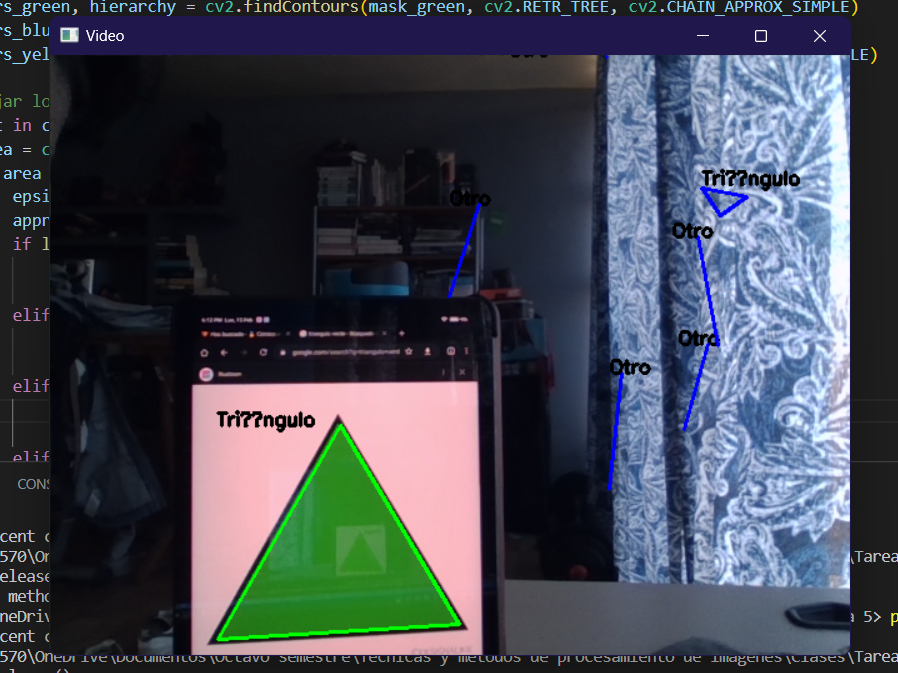
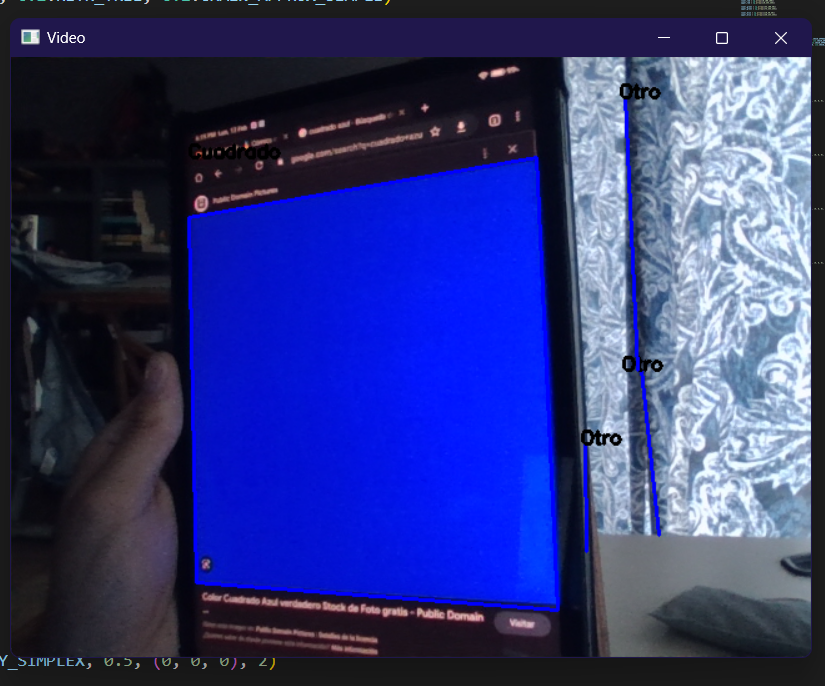
**Luego, se definen rangos de color para la detección de cuatro colores diferentes (rojo, verde, azul y amarillo). Para cada color, se crea una máscara que identifica los píxeles dentro del rango de color correspondiente en el fotograma.**

**Después, se encuentran los contornos en cada máscara y se dibujan en el fotograma. Además, se identifica el tipo de forma (triángulo, cuadrado, pentágono, hexágono o círculo) y se escribe en el fotograma.**

**En resumen, este código permite detectar formas en tiempo real en imágenes capturadas desde una cámara y representarlas en una ventana de visualización.**

****

**Prueba con luz poco saturada y figuras:**

****

**¿Qué pasa si la figura no es detectada o el color no es detectado?**

**Se muestra un texto diciendo otro, además de seguir tratando de encontrar imagen.**

**Conclusiones**

**En resumen, el código proporcionado muestra un ejemplo básico de cómo realizar el reconocimiento de imágenes y colores utilizando OpenCV en Python. Se utiliza la función cv2.inRange para filtrar los colores de la imagen y luego se calcula el contorno de los objetos que se encuentran dentro de los límites de color establecidos. Con esta información, es posible identificar y dibujar un rectángulo alrededor de los objetos y determinar su posición en la imagen. Sin embargo, este código es solo un ejemplo simple y puede requerir mejoras y ajustes adicionales para utilizarse en aplicaciones reales de reconocimiento de imágenes y colores.**

**Bibliografía**

*Ninguna referencia tomada para la asignación.*

‌

**Anexos**

No hay anexos