UNIVERSIDAD DEL BÍO-BÍO

18 Detección de rectángulos

Tutorial Procesamiento de Imagen con webcam

Estudiantes Practicantes:
Luis Pereira

Profesor:
Luis Vera

Laboratorio CIMUBB

2023-2

Importación de bibliotecas

Primero, debes importar las bibliotecas necesarias. En este código, utilizamos cv2 (OpenCV) y numpy.

```
import cv2
import numpy as np
```

cv2: Es la biblioteca OpenCV que utilizamos para capturar video desde la cámara, procesar imágenes y dibujar contornos.

numpy: Se usa para realizar operaciones matemáticas en matrices y arreglos.

Inicialización de la cámara y la ventana

```
# El 0 puede cambiar dependiendo de la cámara a utilizar
cap = cv2.VideoCapture(0)

# Crear una ventana para mostrar los resultados
cv2.namedWindow('Detención de Rectángulos', cv2.WINDOW_NORMAL)
```

cv2.VideoCapture(0): Inicializa la cámara predeterminada. Puedes cambiar el número entre paréntesis si tienes múltiples cámaras conectadas.

cv2.namedWindow('Detención de Rectángulos', cv2.WINDOW_NORMAL): Crea una ventana llamada 'Detención de Rectángulos' para mostrar los resultados. El segundo argumento **cv2.WINDOW_NORMAL** establece la ventana como redimensionable.

Bucle Principal

```
while True:
    ret, frame = cap.read()
    if ret:
        # ENCONTRANDO CONTORNOS DE LA IMAGEN
```

Un bucle **while** se ejecuta continuamente.

cap.read(): Lee un fotograma de la cámara. La variable frame contiene la imagen capturada.

Procesamiento de Imagen

```
    # Cambiar imagen de color a gris
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    # Volver la imagen gris a una binaria
    _, th = cv2.threshold(gray, 140, 240, cv2.THRESH_BINARY)  # Los números cambian (menor el número es más blanco, mayor el número más oscuro)
```

Convertimos la imagen a escala de grises y luego a una imagen binaria. Esto facilita la detección de contornos.

Detección de Contornos y Rectángulos

```
# Encontrar los contornos de la imagen binaria (solo encuentra en imágenes binarias)
contornos, hierarchy = cv2.findContours(th, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)

# CONTANDO OBJETOS
total = 0
for c in contornos:
    area = cv2.contourArea(c)
    if area > 1700:
        peri = cv2.arcLength(c, True)
        approx = cv2.approxPolyDP(c, 0.02 * peri, True)
        if len(approx) == 4:
            # Dibujar un contorno de un rectángulo alrededor del objeto
            cv2.drawContours(frame, [approx], -1, (255, 0, 0), 2, cv2.LINE_AA)
            total += 1
```

Usamos cv2.findContours para encontrar los contornos en la imagen binaria.

Filtramos los contornos que tienen un área mayor que 1700.

Identificamos los contornos que tienen 4 vértices (posibles rectángulos) utilizando cv2.approxPolyDP.

Dibujamos un contorno alrededor de cada rectángulo encontrado.

Mostrar la imagen

```
rectangulo = 'Rectangulo: ' + str(total)
# Mostrar el contador de rectangulos en la esquina superior izquierda
cv2.putText(frame, rectangulo, (10, 150), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2)

# Mostrar la imagen con los contornos en la ventana llamada 'Detención de Rectángulos'
cv2.imshow('Detención de Rectángulos', frame)
```

Agregamos un texto al fotograma para mostrar el número de rectángulos detectados.

Mostramos la imagen procesada en la ventana 'Detención de Rectángulos'.

Detener la ejecución

```
if cv2.waitKey(1) & 0xFF == ord('q'):
    break
```

Esperamos a que el usuario presione la letra ‘q’ para finalizar el programa.

Liberar recursos

```
# Liberar la cámara y cerrar la ventana
cap.release()
cv2.destroyAllWindows()
```

Liberamos la cámara y cerramos todas las ventanas cuando el programa finaliza.

Resultado Final