

Centro de Enseñanza Técnica Industrial

Visión Artificial.

Ingeniería Mecatrónica.



Practica# 10: ROI en Fondo y detección de esquinas

Platel: CETI Colomos.

Nombre: Ruiz Macías Luis Enrique - 21310196

Grado/Grupo: 6°G

Objetivo: Cortar un segmento del fondo de una imagen con ROI y a la región de interés, sacarle las esquinas, de modo que estas sean marcadas en la misma.

Este código usa funciones de CV2.

En el presente código se muestra una función de ROI en la cual se extrae la región de interés deseada y a la misma se le sacan las esquinas, puede ser con cualquier imagen, pero en esta ocasión utilizamos un ejemplo de cuadro cubista, esto para poder mostrar de forma mas firme y efectiva la función de detectar las esquinas.

Esta imagen la toma el programa y les aplica una conversión a grises, de modo que se trabajara sobre un solo canal y hace más fácil la detección de las esquinas, destacar que dependiendo el umbral y la diferencia de color entre grises estas pueden ser o no discriminadas

La siguiente función es la que nos ayuda a esto:

```
corners=cv2.goodFeaturesToTrack(gray,maxCorners,30,qualityLevel=0.1,minDistance=1)
```

Gray se refiere a la escala de blancos.

Max corners es el limite de esquinas que vas a detectar en tu ROI.

Quality Level se refiere a solo se consideran esquinas con una calidad al menos del 10% de la mejor encontrada esto por que usamos 0.1 siendo el umbral de 1 a 0.

minDistancia es la distancia mínima (en píxeles) entre las esquinas detectadas. Si dos esquinas están muy cerca, solo se queda con la mejor.

Esta es la función que nos ayuda a detectar las esquinas.

Posteriormente se muestran las imágenes en la pantalla, con las esquinas mostradas y con el ROI deseado.

A continuación, se presenta el Código:

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

1. Cargar imagen img =

cv2.imread('cubista.jpg') if img

is None:

print("No se encontró la imagen.")

exit()

2. Seleccionar ROI manualmente

roi = cv2.selectROI("Imagen a segmentar ",img,fromCenter=False,
showCrosshair=True)

x, y, w, h = roi

3. Crear una imagen negra del mismo tamaño que la original

roi_only = np.zeros_like(img)

4. Copiar solo el ROI seleccionado roi_only[y:y+h,

x:x+w] = img[y:y+h, x:x+w]

5. Convertir a escala de grises para detectar esquinas

gray = cv2.cvtColor(roi_only, cv2.COLOR_BGR2GRAY)

gray = np.float32(gray)

6. Detectar esquinas dentro del ROI (Shi-Tomasi)

```
corners = cv2.goodFeaturesToTrack(gray, maxCorners=
30,qualityLevel=0.1,minDistance=1)
```

if corners is not None:

```
    corners = corners.astype(int) #
```

```
    for i in corners:
```

```
        x_corner, y_corner = i.ravel()    cv2.circle(roi_only,
(x_corner, y_corner), 3, (0, 255, 0), -1)
```

```
# 7. Mostrar el resultado cv2.destroyAllWindows()
```

```
cv2.imshow("Solo ROI con esquinas detectadas",
roi_only) cv2.waitKey(0) cv2.destroyAllWindows()
```

```
# 8. Guardar si se desea cv2.imwrite("roi_con_esquinas.jpg",
roi_only)
```



Imagen a aplicar.


 Solo ROI con esquinas detectadas



imagen con esquinas
detectadas.