4. Detección de bordes y contornos

Resumen

Entre otras posibilidades, los bordes y contornos son estructuras importantes en una imagen ya que pueden facilitar la caracterización de objetos en una escena. Por tanto, una buena detección y representación de esta información puede ser crítica para problemas como estimar el movimiento, detectar la presencia de determinados objetos, localizarlos o incluso reconocer qué tipo de objetos son. En esta práctica veremos cómo detectar aristas con los detectores Sobel y Canny, y también cómo detectar líneas rectas con un mecanismo de votación, la Transformada de Hough.

Palabras clave Detectores de bordes Canny y Sobel • Transformada Hough

Contenido

1. Filtro Sobel

E1 Ejecute el código proporcionado para aplicar el filtro Sobel a imagen cuadros.png. Como puede observar, el resultado corresponde al gradiente de la imagen en una sola dirección. Ahora,

- Escriba el significado del nivel de gris asociado con los valores de gradiente en la salida mostrada.
- Modifique testSobel() para calcular el gradiente sólo en la otra dirección. Observe el resultado de ambos gradientes (horizontal y vertical) para imágenes cuadros y lena. ¿Sabrías en qué dirección se ha aplicado el gradiente sólo a partir de estas visualizaciones?
- Modifique testSobel() para calcular adicionalmente el gradiente también en la otra dirección y devolver la magnitud del gradiente. Comprueba el resultado.

Binarice la magnitud del gradiente para obtener un mapa E4 de borde binario .

Utilice la variable bAddNoise para comprobar el efecto en el mapa de borde para agregar (o no) ruido gaussiano a la imagen.

Descubra el significado del modo de parámetro opcional de la función sobel() y comprenda el comportamiento esperado para cada uno de sus posibles valores. No es necesario analizar sus respectivos efectos.

2. Detector de Canny

Prueba ahora el detector Canny con diferentes valores de la desviación estándar σ para el filtro gaussiano que este detector aplica como primer paso. Compara los resultados:

- para imágenes (cuadros y lena);
- con el instalador Sobel; y
- con/sin ruido (bAddNoise).

Anota tus observaciones e interpretaciones. Por ejemplo, ¿el mismo σ es bueno tanto para imágenes con ruido como para imágenes sin ruido?

Revise el significado de los dos umbrales que utiliza el detector Canny y analice el resultado en función de estos parámetros. E2

3. Gran transformación

Pruebe en cuadros de imágenes la Transformada de Hough (HT) para líneas (TH) según lo dispuesto en el código. Observe que el HT se aplica en los bordes, por lo que estos deben obtenerse primero.

El primer paso en el HT consiste en calcular el espacio del acumulador a partir de un mapa de bordes binario. E6

1. ¿ Cuál es la función que utilizamos para obtener este espacio, qué parámetros tiene y qué se devuelve?
2. Identificar en el código cuál es la variable correspondiente al espacio del acumulador, cuál es su tamaño y por qué.
3. En la figura mostrada, ¿qué representa cada eje?
   ¿ Cuáles son los valores de cada elemento en el espacio del acumulador? ¿Qué rasgos característicos del HT puedes notar? ¿A qué corresponden?

Una vez que se ha calculado el espacio del acumulador, podemos proceder a identificar sus picos. E3, E5

1. ¿Qué información nos aportan estos picos?
2. ¿De qué dependen sus valores mayores o menores?
3. Identificar en el código qué función utilizamos para localizar esos picos, qué parámetros tiene y qué devuelve .

Tabla 1. Cuatro combinaciones de parámetros del detector de bordes Canny. La imagen de entrada y los resultados de la detección en cada caso se dan en la Fig. 1. Su tarea es relacionarlos (E2) σ T1

		T2
un 1	0.1	0,2
b 1 0,4 0,6 c 3 0,1 0,2		
d 3 0,01 0,02		

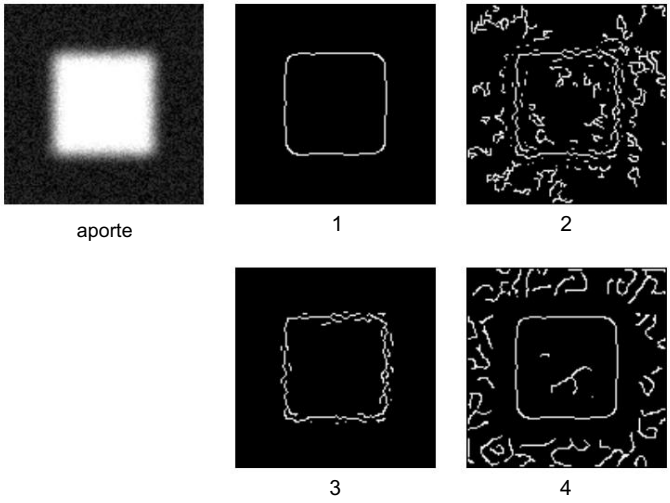

Figura 1. Cada detector Canny se ha aplicado con cuatro combinaciones de parámetros diferentes (Tabla 1) a la imagen de entrada.

4. Estudiar el código que dibuja los picos en el espacio del acumulador y que imprime en la salida estándar sus valores y las correspondientes coordenadas polares (θ,ρ).

4. Ejercicios

1. En lugar de utilizar la función sobel() de los filtros del módulo en scipy.ndimage, aplique el detector Sobel a través de las convoluciones correspondientes. Compruebe visualmente que los resultados de ambas convoluciones concuerden.

2. Dada la imagen de entrada de la Fig. 1, de tamaño 128×128, y con niveles de grises en el rango (aproximado) [0,1], se ha aplicado un filtro Canny, con cada una de las cuatro combinaciones de los parámetros (σ, T1 y T2) dadas en la Tabla 1. Relacionar cada combinación (a, b, cyd) con cada una de las salidas del detector (1, 2, 3 y 4) mostradas en la Fig. 1, con un argumento propio.

1 2 3 4			

El detector de bordes Canny se ha aplicado con la siguiente llamada:

```
característica.canny(estoy, sigma=sigma,
    umbral_bajo=T1, umbral_alto=T2, use_quantiles=False)
```

3. Habilite la variable booleana bRotate para rotar la imagen de entrada algunos grados α. Ejecute el HT para diferentes orientaciones y verifique si se encuentran el mismo (número de) picos en el espacio del acumulador (Hough) en todos los casos. Por otro lado, averigüe si existe una relación entre la coordenada θ del pico y la rotación aplicada α. ¿Por qué?

4. Los histogramas de gradientes se han utilizado para resolver problemas de detección y reconocimiento de objetos1 . Escriba una función HOG(im, nbins) que, dada una imagen im, devuelva un histograma de nbins de orientaciones de gradiente (histograma de gradientes orientados) de los niveles de gris de la imagen.

5. (Opcional) Escriba una función displayLines() para dibujar los picos detectados con el HT en la imagen de entrada. Muestra el resultado con Matplotlib. Decida qué parámetros de entrada debe tener esta función. Luego, llámelo en la ubicación deseada dentro del código proporcionado.

   Pista: La ecuación de línea recta utilizada en HT es, como sabes , x cosθ +y sinθ = ρ. En este caso conocemos θ y ρ (las coordenadas de cada pico detectado y seleccionado), y sólo necesitamos encontrar dos puntos concretos de la recta correspondiente, (x1, y1) y (x2, y2). Puede usar los extremos de la imagen para establecer una de las coordenadas de esos puntos (ya sea x = 0 o y = 0 o x = W o y = H, para una imagen W ×H), y luego obtener la otra coordenada resolviendo la ecuación. Una vez conocidas las cuatro coordenadas de los dos puntos, podemos trazar la recta de la siguiente manera:

```
plt.plot((x1,x2),(y1,y2))
```

6. (Opcional, avanzado) Hemos visto cómo detectar líneas rectas con la transformada de Hough, pero no hemos detectado los segmentos reales. Piensa o descubre cómo resolver esto y, si estás preparado para el desafío, implementa y prueba algún procedimiento simple.

1Como puedes leer en esta entrada de wikipedia, un trabajo de Dalal y Triggs en 2005 fue un hito decisivo para el desarrollo de este concepto. Por favor, tenga en cuenta que esto fue en el área previa al aprendizaje profundo para contextualizar este logro. El descriptor HOG es un ejemplo ilustrativo de cómo una idea relativamente simple, bien aplicada, puede resultar muy efectiva.