Actividad autocorregible: Segmentación y reconocimiento de objetos

Objetivos

El objetivo de esta actividad es utilizar técnicas de procesamiento de imágenes utilizando OpenCV para implementar un algoritmo de segmentación, así como el uso de estos algoritmos en tareas de reconocimiento para tareas específicas de visión por computadora.

Los objetivos específicos son:

* Utilizar técnicas de morfología matemática.
* Utilizar diferentes técnicas de binarización.
* Aplicar operaciones básicas de procesamiento de imágenes como la suma o resta.
* Identificar las etapas que pueden ser aplicadas en una tarea de reconocimiento de objetos.
* Identificar y entender la extracción de atributos para llevar a cabo un problema de clasificación específico.

Descripción de la actividad

El algoritmo de *watershed* es una técnica de segmentación de imágenes que se utiliza en el proceso de análisis de imágenes para dividir una imagen en diferentes regiones homogéneas. Algunos ejemplos de aplicaciones donde se puede utilizar el algoritmo de *watershed* son: segmentación de células en imágenes de células, de tejidos en imágenes médicas, de objetos para la detección de estos y en imágenes satelitales para segmentar diferentes tipos de terrenos.

A lo largo de la actividad implementaremos un proceso de segmentación para un tipo de imagen en particular y veremos como este y otros procesos del mismo estilo pueden ser utilizados para tareas de clasificación. También haremos énfasis en la etapa de extracción de atributos para llevar a cabo esta tarea de clasificación con éxito.

Preguntas

1. La función cv2.morphologyEx() en OpenCV es una función de morfología matemática que permite aplicar diversas operaciones morfológicas a una imagen binaria o de escala de grises. En el siguiente código estaremos haciendo las siguientes tareas: leer la imagen, transformarla a niveles de gris y binarizarla mediante Otsu:

Texto

Descripción generada automáticamente

Después de aplicar Otsu, la imagen resultante es la siguiente:

Imagen en blanco y negro

Descripción generada automáticamente con confianza media

Una vez binarizada la imagen, aplicaremos operaciones morfológicas, como se muestra en el siguiente código:

Texto

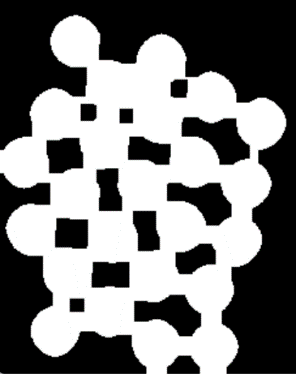
Descripción generada automáticamente con confianza media

A. Imagen resultante A.

Dibujo en blanco y negro

Descripción generada automáticamente con confianza baja

B. Imagen resultante B.



C. Imagen resultante C.

Imagen que contiene dibujo, luz

Descripción generada automáticamente

D. Imagen resultante D.

Un dibujo en blanco y negro

Descripción generada automáticamente con confianza baja

1. El operador cv2.distanceTransform() en OpenCV es una función que calcula la transformada de distancias de una imagen binaria, la cual proporciona una imagen de distancias que muestra la distancia de cada píxel a la frontera más cercana de un objeto en la imagen binaria. El siguiente código aplica esta transformada después de los operadores morfológicos.

Texto

Descripción generada automáticamente

Texto

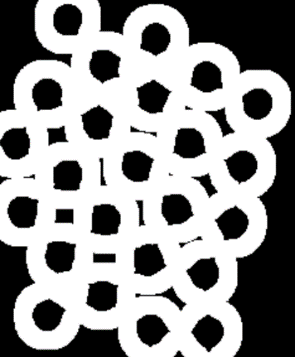
Descripción generada automáticamente con confianza media

Texto, Carta

Descripción generada automáticamente

Indica cual es la imagen resultante después de haber ejecutado el código:

A. La imagen resultante es:



B. La imagen resultante es:

Dibujo en blanco y negro

Descripción generada automáticamente con confianza baja

C. La imagen resultante es:

Un dibujo en blanco y negro

Descripción generada automáticamente con confianza baja

D. La imagen resultante es:

Imagen en blanco y negro

Descripción generada automáticamente con confianza media

1. En el código siguiente:

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente con confianza media



La línea:



Es equivalente:

A. Opción A:



B. Opción B:



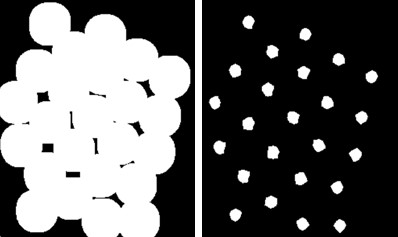
C. Opción C:



D. Opción D:



1. El siguiente paso a aplicar la transformada de *watershed* es la resta de la imagen de operadores morfológicos menos la imagen de distancia, esto es, la diferencia de las siguientes dos imágenes da como resultado:



Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

A. La imagen resultante es:

Dibujo en blanco y negro

Descripción generada automáticamente con confianza baja

B. La imagen resultante es:

Imagen en blanco y negro

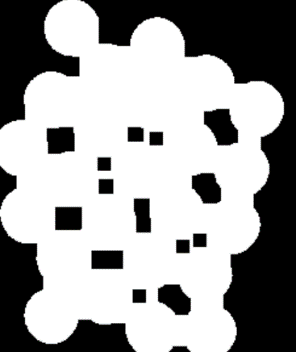
Descripción generada automáticamente con confianza media

C. La imagen resultante es:

Un dibujo en blanco y negro

Descripción generada automáticamente con confianza baja

D. La imagen resultante es:



1. Encontrar los componentes conexas o componentes conectados es otra tarea muy relevante en el procesamiento de imágenes. La función cv2.connectedComponents() en OpenCV devuelve un número entero que representa el número total de componentes conectados encontrados en la imagen proporcionada. Esta función asigna etiquetas a cada píxel de la imagen, cada etiqueta identifica un componente conectado diferente y los píxeles con la misma etiqueta pertenecen al mismo componente conectado.

**Nota.** La implementación de *watershed* en OpenCV utiliza la etiqueta -1 para representar las áreas desconocidas o no asignadas. Estas áreas se encuentran entre los límites de los objetos segmentados y se pueden visualizar de manera distinta a los objetos segmentados, por ejemplo, rellenándolas con un color determinado, en este caso de color rojo (255,0,0).

Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Imagen que contiene Texto

Descripción generada automáticamente

Imagen que contiene Diagrama

Descripción generada automáticamente



Después de ejecutar el código anterior, ¿cuántas etiquetas (cantidad de etiquetas) devuelve le siguiente código?

A. 20 componentes.

B. 25 componentes.

C. 24 componentes.

D. 26 componentes.

1. La función cv2.merge en OpenCV es una función que se utiliza para combinar varios canales de una imagen en una sola imagen multicanal. La función toma como entrada una lista de matrices (cada una representando un canal de la imagen multicanal) y devuelve una única matriz que representa la imagen completa con todos los canales combinados. En este ejemplo la utilizamos para mostrar las etiquetas que se han obtenido:

**Texto

Descripción generada automáticamente**

**Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente**

Imagen que contiene Texto

Descripción generada automáticamente

Imagen que contiene Diagrama

Descripción generada automáticamente

Texto

Descripción generada automáticamente



Después de haber ejecutado el código, la imagen de salida Componentes finales es:

A. Imagen de salida A.

Dibujo en blanco y negro

Descripción generada automáticamente con confianza media

B. Imagen de salida B.

Imagen que contiene Gráfico de burbujas

Descripción generada automáticamente

C. Imagen de salida C.

Patrón de fondo

Descripción generada automáticamente

D. Imagen de salida D.

Imagen en blanco y negro

Descripción generada automáticamente con confianza media

1. Requerimos hacer un OCR Optical Character Recognition de texto impreso. Algunas de las tareas para lograr esto son las siguientes:

A. Extracción de características.

B. Preprocesamiento (convertir imagen a niveles de gris, eliminar ruido).

C. Segmentación de cada uno de los caracteres (identificar una ROI región de interés de cada carácter).

D. Normalizar las imágenes (obtener imágenes del mismo tamaño para cada carácter).

¿Cuáles podrían ser el orden apropiado de las etapas para llevar a cabo este proceso?

A. B, C, A, D.

B. B, C, D, A.

C. B, D, A, C.

D. C, B, A, D.

1. Para un problema de reconocimiento de un conjunto reducido de caracteres de una fuente (Font) en particular, la proyección vertical y horizontal del histograma, podría ser una representación apropiada de atributos:

A. Sí.

B. No.