Mejoramiento de imágenes

Felipe Sánchez Soberanis

12 de septiembre de 2022

Índice

Punto 1

Resultados

Figura 1: Resultado de la ecualización del histograma en la imagen.

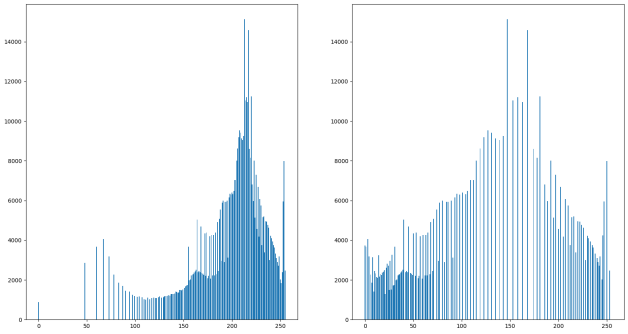

Figura 2: Resultado de la ecualización del histograma.

Como se puede observar, el resultado de ecualizar el histograma es que la imagen modificada tiene un mejor contraste, comparada la imagen original. Esto es debido a que, como se puede observar en el segundo histograma, los valores de brillo son mejor distribuidos a lo largo del rango 0 a 255.

Bibliografía

N/A.

Librerías

- opencv-contrib-python
- numpy
- matplotlib

Algoritmos propios

Calcular histograma de una imagen

```
imagen = ingresar imagen

ancho = imagen.ancho
alto = imagen.alto

x = 0 a 255, de 1 en 1
```

```
y = 256 0s

por cada w en ancho:
    por cada a en alto:
        v = imagen[w][a].intensidad_del_pixel
        y[v] += 1

regresar x, y
```

Ecualizar histograma

```
imagen = ingresar imagen

ancho = imagen.ancho
alto = imagen.alto

y = calcular_histograma(imagen)

k = 255 / (ancho * alto)
suma = 0

imagen_ecualizada = zeros, ancho y alto igual al de imagen

por cada w en ancho:
    por cada a en alto:
        por cada s en intensidad_pixel(imagen[w][a]):
            suma += y[s]
        imagen_ecualizada[w][a] = k * suma
        suma = 0

regresar imagen_ecualizada
```

Problemas

N/A.

Punto 2

Resultados

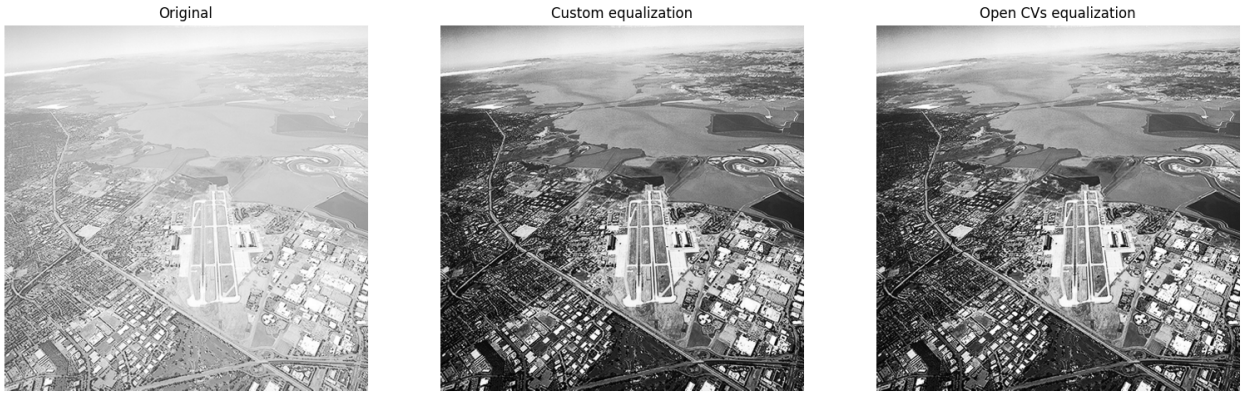

Figura 3: Comparación de la ecualización hecha por mi contra la ecualización de la librería OpenCV.

Visualmente, los resultados obtenidos por el algoritmo hecho desde 0 contra el algoritmo implementado en la librería de OpenCV, no es aparente. La mayor diferencia se encuentra en la velocidad con la que ambos algoritmos se llevan a cabo, ya que el propio, al estar hecho en Python, es mucho más lento que el de OpenCV, que se encuentra escrito en C y C++.

Bibliografía

Figura 4: Documentación de la función equalizeHist (click aquí para ir).

Librerías

- opencv-contrib-python
- matplotlib

Algoritmos propios

N/A.

Problemas

N/A.

Punto 3

Las transformaciones afines son transformaciones que están compuestas de una transformación lineal seguida por una traslación o desplazamiento. Es decir, mantiene las líneas y el paralelismo, más no las distancias ni los ángulos.

Identidad

En matemáticas una función identidad es una función matemática, de un conjunto M a sí mismo, que devuelve su propio argumento. Es decir, mantiene sin cambios la imagen que se use como entrada.

Escalamiento

Es una aplicación lineal que aumenta (incrementa) o contrae (disminuye) el tamaño de distintas entidades (como formas o figuras geométricas) mediante un factor de escala que es el mismo en todas direcciones. El resultado de un escalado uniforme es una relación de semejanza (en el sentido geométrico) entre la figura original y su imagen.

Rotación

Es el movimiento de cambio de orientación de un cuerpo o un sistema de referencia de forma que una línea (llamada eje de rotación) o un punto permanece fijo.

Reflexión

Es un mapeo desde un espacio euclídeo a sí mismo que es una isometría con un hiperplano como un conjunto de puntos fijos; este conjunto es llamado eje (en 2 dimensiones) o plano (en 3 dimensiones) de reflexión. La imagen de una figura por una reflexión es su imagen especular, en el eje o plano de reflexión. Por ejemplo, la imagen especular de la letra minúscula p por una reflexión con respecto a un eje vertical se vería como la letra q. Su imagen por una reflexión en un eje horizontal se vería como la letra b.

Shear

Es un tipo de matriz elemental que implica la suma del múltiplo de una fila (o de una columna) a otra. Esta matriz se puede generar a partir de la matriz identidad, reemplazando algunos elementos nulos por uno o más valores distintos de cero.

El nombre de matriz de cizallamiento o de corte se refiere al hecho de que la matriz representa una transformación asociada a una deformación lateral elástica de un objeto, similar a la que se produce en un cubo cuando se ejerce un esfuerzo tangencial (o cortante) sobre dos de sus caras opuestas.

Punto 4

Resultados

Figura 5: Resultados de la aplicación de un mejoramiento de imagen por estadísticas locales con un cuadro de 3x3.

Se puede observar cómo el mejoramiento de imagen por medio de estadísticas locales permite que la imagen sea suavizada. En la imagen mostrada arriba, el ejemplo es exagerado por motivos demostrativos.

Bibliografía

N/A.

Librerías

- opencv-contrib-python
- numpy
- matplotlib

Algoritmos propios

Estadísticas locales

```
imagen = ingresar imagen
kernel = ingresar kerel, entero mayor a 0

ancho = imagen.ancho
alto = imagen.alto

imagen_suave = copiar(imagen)

pixeles = lista vacía

por cada x en ancho:
    por cada y en alto:
```

```
        por cada pixel alrededor de imagen[x][y] en un radio igual a kernel:
            pixeles.agregar(pixel)
        promedio = promedio(pixeles)
        imagen_suave[x][y] = promedio
        pixeles = vaciar lista

regresar imagen_suave
```

Problemas

N/A.

Punto 5

Resultados

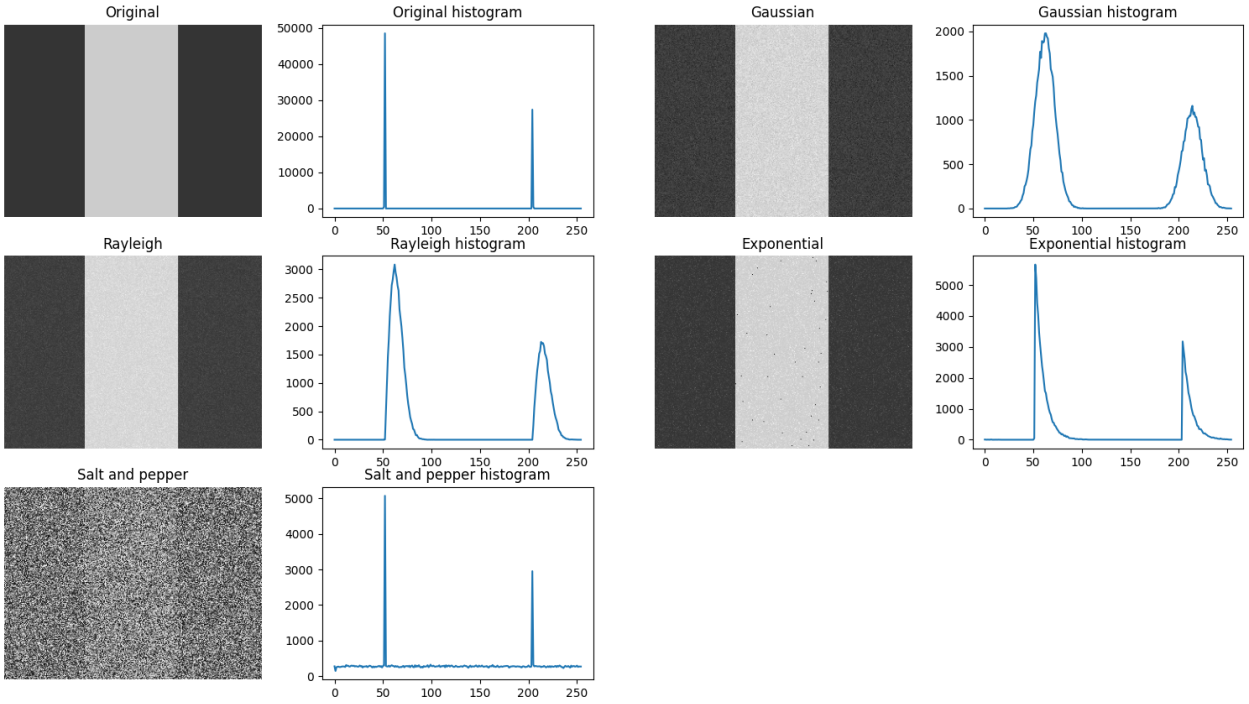

Figura 6: Resultados de la aplicación de diferentes tipos de ruido a una imagen con 2 tonos de gris.

A pesar de que, visualmente, el ruido aplicado no se ve tan diferente entre las imágenes, los histogramas de cada caso permiten ver cómo obedecen la distribución de la función de probabilidad utilizada para generar el ruido.

Bibliografía

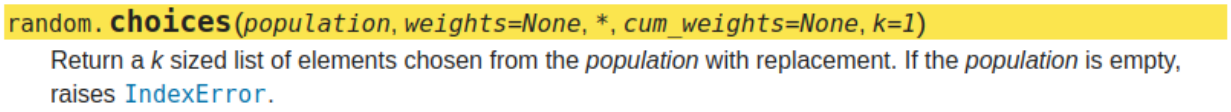

Figura 7: Documentación de la función choices del módulo random de Python (click aquí para ir).

Librerías

- opencv-contrib-python
- numpy
- matplotlib

Algoritmos propios

Agregar ruido a una imagen

```
imagen = ingresar imagen
imagen_ruido = copy(imagen)
```

```
ancho = imagen.ancho
alto = imagen.alto

x_eje = 0 a 255
y_eje = probabilidad de x según el método usado para el ruido por cada x en x_eje

por cada x en alto:
    por cada y en ancho:
        imagen_ruido[x][y] += eleccion_al_azar_segun_pesos(valores=x_eje, pesos=y_eje)

regresar imagen_ruido
```

Problemas

N/A.

Conclusiones

Esta tarea hizo que trabajaramos con, a mi parecer, varios de los procesamientos de imágen más importantes que hemos visto en la materia. En el caso de la ecualización del histograma, esto permite que las imagenes tengan una mejore visualización, lo que quiere decir que, al aplicar algún tipo de visión por computadora, los resultados serán mucho mejores.

De igual manera, en el caso de la aplicación de diferentes ruidos, demuestra cómo reconocer el ruido que tiene cada imagen. En la vida real, la intención no es aplicar ruido, si no lograr reconocerlo por medio del histograma y aplicar la operación inversa para poder tener una imagen más clara, logrando mejorar los resultados de cualquier proceso que se aplique seguidamente.

Repositorio

https://github.com/FelipeSanchezSoberanis/vision-por-computadora/tree/main/mejoramiento_de_imagenes