16 de marzo de 2022

**Nombre y número de registro:** Roberto Carlos Guzmán Cortés  **Calificacion:** \_\_\_\_\_\_\_

Notas: Tienes 60 minutos, como máximo, para resolver este examen. Una vez que haya terminado, cargue su solución en Google Drive en su carpeta de tareas, guárdalo en la carpeta que creaste llamada "E01\_First Partial Exam".

Problema 1 (70%): Filtrado de medianas en el procesamiento de imágenes.

Modifique la plantilla C incluida en esta carpeta comprimida para implementar un filtro de mediana que limpiará una imagen de entrada contaminada con ruido de "sal y pimienta" usando un filtro de mediana.

Definición: El **filtro mediano** es una técnica de filtrado digital no lineal, que se utiliza a menudo para eliminar el ruido. Dicha reducción de ruido es un paso típico de preprocesamiento para mejorar los resultados del procesamiento posterior (por ejemplo, detección de bordes en una imagen).

|  |  |
| --- | --- |
|  |  |
| Lenna contaminada con ruido “salt & pepper” | Resultado después de aplicar el filtro de mediana |

**El algoritmo de filtro mediano:**

El filtro mediano considera cada píxel de la imagen por turnos y mira a sus vecinos cercanos para decidir si es o no representativo de su entorno. En lugar de simplemente reemplazar el valor de píxel con la *media* de los valores de píxel vecinos, lo reemplaza con la *mediana* de esos valores. La mediana se calcula clasificando primero todos los valores de píxel del vecindario circundante en orden numérico y luego reemplazando el píxel que se está considerando con el valor de píxel medio. (Si la vecindad bajo consideración contiene un número par de píxeles, se usa el promedio de los dos valores de píxeles del medio). La figura 1 ilustra un ejemplo de cálculo.

|  |
| --- |
|  |
| **Figura 1** Cálculo del valor medio de una vecindad de píxeles. Como se puede ver, el valor del píxel central de 150 es bastante poco representativo de los píxeles circundantes y se reemplaza con el valor medio: 124. Aquí se usa un vecindario cuadrado de 3×3; los vecindarios más grandes producirán un suavizado más severo.  Hacer el borde de la imagen filtrada negro (valor de píxel 0) .  Sube lo siguiente a Google Drive:   1. Una imagen de su diagrama de flujo que muestre claramente cómo planteó el problema que le permitió resolverlo. 2. Un archivo de código C perfectamente comentado. 3. Las tres imágenes resultantes obtenidas después de aplicar el filtrado mediano a cada una de las 3 imágenes ruidosas proporcionadas con el examen (lenaNoise10Percent.pgm, lenaNoise20Percent.pgm y lenaNoise30Percent.pgm). |

**FlowChart**

1. **Leer imagen y vaciarla en matriz original**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 49 | 50 | … | 179 | 88 |
| 78 | 78 | … | 56 | 45 |
| … | … | … | … | … |
| 189 | 56 | … | 344 | 68 |
| 99 | 190 | … | 500 | 71 |

1. **Inicializar Matriz de salida a 0s (porque habrá pérdida de info en row y column 0, row y column 511)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0 | 0 | … | 0 | 0 |
| 0 | 0 | … | 0 | 0 |
| … | … | … | … | … |
| 0 | 0 | … | 0 | 0 |
| 0 | 0 | … | 0 | 0 |

1. **Recorrer matriz original (pixel actual en naranja obscuro, vecinos en naranja tenue)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0 | 0 | … | 0 | 0 |
| 0 | 0 | … | 0 | 0 |
| … | … | … | … | … |
| 0 | 0 | … | 0 | 0 |
| 0 | 0 | … | 0 | 0 |

* 1. **Vacío temporalmente los vecino y el pixel actual en array[9]**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 1 | 49 | 177 | 0 | 0 |

* 1. **Ordeno array[9] (mediana en amarillo)**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 49 | 177 |

* 1. **Hago que el pixel actual valga arr[4] porque es la mediana**

Mat\_salida[col][row] = array[4]

1. **Guardo la matriz de salida como una imagen**