Universidad Nacional de Quilmes

Introducción ... 1er Semestre 2025

Primer Parcial

- **EN LA PARTE SUPERIOR DE CADA HOJA PONER:**
  - **NOMBRE Y APELLIDO**
  - **DNI**
  - **COMISIÓN**
  - **NÚMERO DE HOJA y CANTIDAD TOTAL CON LA FORMA <número de hoja>/<cantidad total de hojas>**
- Leer el examen completo antes de empezar a resolver los ejercicios, ya que ayuda a comprender mejor el dominio. Se puede consultar cualquier material (en papel) que haya sido escrito antes de comenzar el examen y usar sin definir todas las funciones y procedimientos vistos durante la cursada.
- Pensar bien la estrategia a seguir, consultar lo que no se entienda y recordar que el parcial es una instancia más de aprendizaje donde algún docente va a dar una devolución personalizada de todo lo que se escriba. Es necesario aprovechar esta instancia como algo más que solamente un momento para sacar una nota.

Filtros en imágenes

Una imagen digital es una representación de una imagen en una **matriz bidimensional** de puntos denominados *píxeles*, que se puede almacenar en código entendible por una máquina. Si queremos representar una imagen digital en **color** se debe codificar para cada píxel el color con el que debe mostrarse. Así, se puede usar alguna codificación de un espacio de colores como ***CMYK*** para determinar el color final que cada píxel tiene en la imagen. Esta codificación implica cuatro números del 0 al 255, donde cada uno identifica un **componente de color (Cyan, Magenta, Yellow y Key)**, que representan la intensidad de cada uno de los colores primarios en el modelo aditivo de color.

En esta evaluación vamos a modelar imágenes en Gobstones y se le solicitará que implemente distintos **filtros** de imágenes (entendidos como formas de alterar la imagen). En este sentido, se utilizará el **tablero** para representar la **matriz bidimensional**, donde cada **celda** representará un **píxel**. La intensidad de cada **componente de color** se representará con una cantidad de bolitas de entre 0 y 255; utilizando bolitas de diferentes colores para cada uno de los componentes.

Dispondrá para realizar los filtros ciertas operaciones que puede tomar como primitivas para llevar adelante las tareas solicitadas:

componenteC()
PROPÓSITO: Describe la componente "cyan" del píxel actual.
PRECONDICIONES: Ninguna.
TIPO: Color

componenteM()
PROPÓSITO: Describe la componente "magenta" del píxel actual.
PRECONDICIONES: Ninguna.
TIPO: Color

componenteY()
PROPÓSITO: Describe la componente "yellow" del píxel actual.
PRECONDICIONES: Ninguna.
TIPO: Color

componenteK()
PROPÓSITO: Describe la componente "key" del píxel actual.
PRECONDICIONES: Ninguna.
TIPO: Color

minComponente()
PROPÓSITO: Describe la componente más pequeña.
PRECONDICIONES: Ninguna.
TIPO: Color

maxComponente()
PROPÓSITO: Describe la componente más grande.
PRECONDICIONES: Ninguna.
TIPO: Color

siguienteComponente_(componente)
PROPÓSITO: Describe la componente siguiente a **componente**.
PRECONDICIONES: Ninguna.
PARÁMETROS:
- *componente*: Color - color del componente
TIPO: Color

intensidadDeComponente_Acá(componente)
PROPÓSITO: Describe la intensidad de la componente **componente** del píxel actual.
PRECONDICIÓN: Ninguna.
PARÁMETROS:
- *componente*: Color - color del componente
TIPO: Número.

IncrementarComponente_Acá(componente)
PROPÓSITO: Incrementar en 1 la intensidad de la componente *componente* del píxel actual.
PRECONDICIONES:
- La componente *componente* del píxel actual es menor a 255.
PARÁMETROS:
- *componente* : Color - color del componente

DecrementarComponente_Acá(componente)
PROPÓSITO: Decrementar en 1 el valor de la componente *componente* del píxel actual.
PRECONDICIONES:
- La componente *componente* del píxel actual es mayor a 0.
PARÁMETROS:
- *componente* : Color - color del componente

componenteDeMenorTemperaturaEnLaImagen()
PROPOSITO: Describe a la componente de menor temperatura en la imagen actual.
PRECONDICIÓN: Existe una componente de menor temperatura en la imagen actual.
TIPO: Color
OBSERVACIÓN: La componente de menor temperatura de una imagen es aquella para la cual la sumatoria de las intensidades de dicha componente en todos los pixeles es la menor.

Basados en este modelo y asumiendo que sobre el tablero hay una imagen bien representada, se pide:

Ejercicio 1)

Implementar la función **esPixelSaturadoAcá** que indica si el pixel actual está saturado en alguna de sus componentes. Un pixel está saturado en una componente cuando la intensidad de esa componente en dicho pixel es mayor a la suma de las intensidades de la misma componente en los pixeles lindantes existentes (en direcciones ortogonales).

Ejercicio 2)

Implementar el procedimiento **SaturarComponenteDeMayorTemperatura** que satura en todos los pixeles de la imagen la componente de mayor temperatura de la imagen representada en el tablero, usando como contraste la componente de menor temperatura de la imagen.

Saturar un pixel en una componente usando otra componente como contraste implica aumentar la intensidad de la componente a saturar tanto como la intensidad de la componente de contraste. Por ej. si el pixel tiene los siguientes valores de componente (C: 120, M: 200, Y: 55, K: 30), y estamos saturando M con K, el resultado será un pixel con los siguientes valores de componente: (C: 120, M: 230, Y: 55, K: 30), es decir, donde a M se le sumó la intensidad de K. **NOTA**: Recordar que la intensidad máxima de una componente es de 255, por lo cual si la saturación resulta en una superación de dicha intensidad, la componente deberá quedar con 255.

Por otro lado, la componente de mayor temperatura de una imagen es aquella para la cual la sumatoria de las intensidades de dicha componente en todos los pixeles es la mayor. Asimismo, la componente de menor temperatura de una imagen es aquella para la cual la sumatoria de las intensidades de dicha componente en todos los pixeles es la menor.

Ejercicio 3)

Un pixel se puede hacer negativo, invirtiendo la intensidad de sus componentes de colores. Esto se logra derivando el valor de cada componente de la resta entre la máxima intensidad posible (255) y el valor de intensidad actual de dicha componente.

Por ejemplo, el pixel con las siguientes componentes (C: 120, M: 200, Y: 55, K: 30) su pixel invertido tendría las componentes (C: 135, M: 55, Y: 200, K: 225).

Considere la siguiente implementación de **InvertirPixel()**, y asuma la existencia del procedimiento ***DibujarPixelConC_M_Y_K_*** que, como su nombre sugiere, deja el pixel actual con los valores de componentes dados.

```
procedure InvertirPixel() {
  /*
    PROPÓSITO: Invertir el píxel actual.
    PRECONDICIONES:
     * En la celda actual hay un píxel válido representado.
  */
  nuevoC:= (255 - nroBolitas(Azul))
  nuevoM:= (255 - nroBolitas(Rojo))
  nuevoY:= (255 - nroBolitas(Verde))
  nuevoK:= (255 - nroBolitas(Negro))
  DibujarPixelConC_M_Y_K_(nuevoC, nuevoM, nuevoY, nuevoK)
}
```

Se pide que determine si la implementación propuesta es adecuada (sigue los buenos criterios trabajado en la materia) o no. **JUSTIFICAR** brevemente su respuesta (no más de 5 renglones).