UNIVERSIDAD DE DARMSTADT
B. FRÖMMER, E. HERGENRÖTHER, B. MEYER

h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES
fbi
FACHBEREICH INFORMATIK

4. JUNIO DE 2023

VISUAL INFORMÁTIC SOSE 2023
A
TAREA 4

El objetivo de esta hoja de tareas es familiarizarse con la biblioteca OpenCV. OpenCV proporciona los fundamentos para trabajar con imágenes, así como muchas funciones utilizadas en el procesamiento de imágenes y
-análisis se utilizan con frecuencia. Está disponible como proyecto de código abierto en: http://opencv.org/

4.1 Instalación de OpenCV

Las instrucciones detalladas de instalación de C++ están disponibles en Moodle.
Puedes encontrar buena documentación en el libro Learning OpenCV de Gary Bradski y Adrian Kaehler o aquí: https://docs.opencv.org/4.7.0/d1/dfb/intro.html

(Recomendado) Alternativa: Instalar OpenCV bajo C++ puede ser un poco tedioso. Es mucho más fácil integrar OpenCV en un proyecto Python:

a) Instala un IDE de Python de tu elección (por ejemplo PyCharm Community de JetBrains, gratis como licencia de estudiante en https://www.jetbrains.com/pycharm/download/#section=windows).

b) Instale una versión actual de Python (en https://www.python.org/downloads/, asegúrese de que la variable PATH se establece en consecuencia durante la instalación).

c) Crea un proyecto Python en el IDE como prueba. En PyCharm, puedes usar *python --version en* la terminal para mostrar el número de versión de Python instalada. Si esto no funciona, su entorno Python no se encuentra.

d) Instalación de paquetes adicionales: Python viene con un asistente de paquetes integrado, PIP (Package Installer for Python). Si desea gestionar diferentes proyectos con diferentes versiones de Python, le recomendamos el gestor de paquetes *Anaconda* (no es necesario para estas prácticas).

e) Con pip install opencv-python se instala automáticamente la versión actual de OpenCV. Más información en https://pypi.org/project/opencv-python/.

f) Opcional: La edición de los datos de imagen leídos puede simplificarse con la ayuda de la biblioteca matemática Numpy. pip install numpy instala la biblioteca en consecuencia.

Copia la imagen yoshi.png en la carpeta de tu proyecto y finalmente prueba tu instalación añadiendo las siguientes líneas :

```
# import numpy as np ## optional import
cv2
print("Versión de OpenCV: " + cv2. version )
img = cv2.imread("yoshi.png")
cv2.imshow("image", img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

4.2 Primeros pasos

En primer lugar, veamos funciones sencillas de OpenCV para la manipulación y visualización de imágenes. Utiliza funcionalidades conocidas o recurre a recursos online como la documentación de la librería OpenCV para:

- para cargar el archivo adjunto "yoshi.png".
- muestra la anchura, la altura y el número de canales de color en la consola.
- cambiar el formato de los datos de la imagen a coma flotante.
- mostrar la imagen hasta que se pulse una tecla (tanto como uint8 como como float).
- dibuja un cuadrado rojo de 10x10 píxeles en el centro de la imagen.
- reemplazar cada 5 líneas con píxeles negros.
- guardar la imagen en el disco duro.

4.3 Espacios de color

En esta tarea queremos ocuparnos de las conversiones del espacio de color.

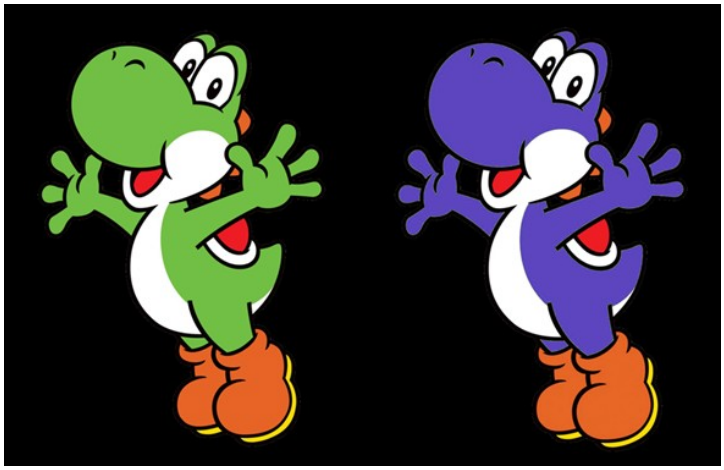
(1) Además de Yoshi, carga la máscara correspondiente "mask.png" en tu programa.

(2) Transfiere la imagen Yoshi al espacio de color HSV.

(3) Cambia el valor H en la imagen Yoshi para todos los píxeles blancos de la imagen de máscara.

(4) Muestre la nueva imagen en la pantalla (pero tenga en cuenta que imshow() sólo puede procesar correctamente imágenes BGR). El resultado debería ser el que se muestra.

(5) Opcional: Construya un control deslizante para ajustar manualmente el valor H a cualquier valor y actualizar la imagen mostrada.



4.4 Transferencia de color de Reinhard

En este ejercicio pondrás en práctica una variante de la transferencia de color de Reinhard. (Fuente: Transferencia de color entre imágenes de Reinhard, Ashikmin, Gooch y Shirley, disponible en Moodle). La idea de la transferencia de color es adaptar el esquema de color básico de una imagen (aquí: entrada) a un esquema de color de destino específico (aquí: destino). En la Fig. 1 se muestra un ejemplo.
Proceda como sigue:

(1) Cargue dos imágenes como "Entrada" y "Objetivo" y conviértalas en valores de coma flotante (en el rango de 0 a 1).

(2) Convierte ambas imágenes al espacio de color Lab.

(3) Para cada canal de color (por separado):

i) Restar el valor medio de la entrada de la entrada (el nuevo valor medio de la entrada se convierte en 0) .

ii) Divida la entrada por su desviación típica (la nueva desviación típica se convierte en 1).

iii) Multiplica la entrada por la desviación típica del objetivo.

iv) Añade el valor medio del objetivo a la entrada.

(4) Convierte la entrada de nuevo a BGR.

Prueba también el algoritmo con los espacios de color RGB y HSV que conozcas y con otras imágenes. ¿Qué espacios de color son adecuados para el algoritmo y cuáles no? ¿Con qué imágenes funciona bien la transformación del color? ¿Puedes explicar por qué?

Figura 1: **Izquierda: Imagen de entrada con cielo azul claro. Centro: Imagen de destino con la coloración deseada. Derecha: Imagen de entrada ajustada con la nueva coloración.**