Laboratorio 1: Comenzando con OpenCV

Alumno: Josue Samuel Philco Puma

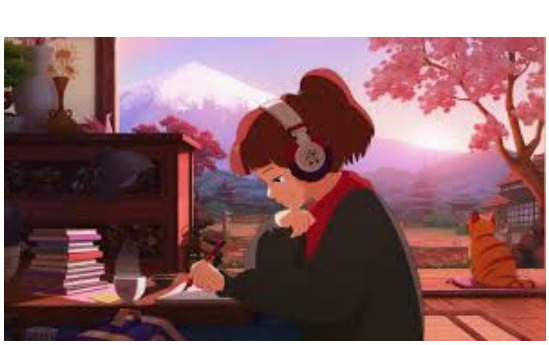
6 de mayo de 2025

1. Mostrando la imagen

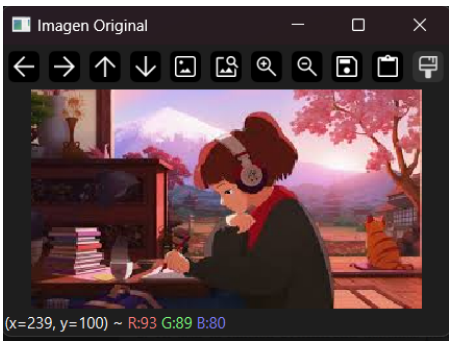
Para mostrar la imagen desde OpenCV simplemente usamos las funciones OpenCV que ofrece, se usa **cv::imread** para leer la imagen y **cv::imshow** para mostrarla en pantalla.

```cpp
#include <iostream>
#include <opencv2/opencv.hpp>
using namespace std;
using namespace cv;

int main() {
    string image_file;
    cout << "Ingrese el nombre de la imagen (con extension): ";
    cin >> image_file;
    string image_path = "D:/UNSA EPCC/7mo semestre/Computacion Grafica/Unidad 1/Imagenes con OpenCV/Imagenes/" + image_file;

    Mat image_original = imread(image_path);
    if (image_original.empty()) {
        cout << "No se pudo cargar la imagen." << endl;
        return -1;
    }

    imshow("Imagen Original", image_original);
    waitKey(0);
    return 0;
}
```

Entonces, con el código anterior veremos el resultado que es mostrar la imagen que le demos a OpenCV:



(a) Imagen original.



(b) Imagen mostrada con OpenCV.

Figura 1: Mostrando la imagen original (a) y la imagen con OpenCV (b)

2. Función para modificar los pixeles para hacer una rotación

Ahora que tenemos la imagen cargada y mostrándose debemos manipular los píxeles de la imagen para que se reordenen al realizar una rotación, en este caso la rotación es antihoraria por lo que debemos saber que vamos a hacer:

1. **Primero debemos encontrar la forma de acceder a las filas y columnas de la imagen**: Esto lo haremos usando lo siguiente: **.rows** y **.cols**, esto lo que hace es obtener el número de filas y columnas de una imagen.

2. **Crear una nueva imagen para el resultado:** Como vamos a manipular los píxeles, lo mejor es almacenarlo en una nueva imagen, pero debemos tener en cuenta que al rotar nuestra imagen intercambiamos las filas por las columnas y viceversa, y también al crearlo que sea de forma en que esté en los 3 canales de colores (RGB).

3. **Recorrer las filas y columnas y acceder a cada píxel para intercambiar los píxeles:** Ahora que tenemos las filas y columnas podemos recorrerlas con dos bucles for, como vamos recorriendo debemos intercambiar los píxeles, intercambiar las filas para que sean columnas no es un gran problema, pero al intercambiar las columnas para que sean filas debemos ir reordenando los píxeles. que sería la intrucción de intercambio **image.at<Vec3b>(columns - 1 - j, i)**

4. **Retornar la imagen rotada:** Una vez terminado los dos for se retorna la nueva imagen con los píxeles cambiados.

Ahora con los pasos numerados que se debe realizar se muestra el código que lo realiza:

```
Mat rotate_image_antihorary(const Mat& image) {
    int rows = image.rows;
    int columns = image.cols;

    Mat image_rotate(columns, rows, CV_8UC3);

    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < columns; j++) {
            image_rotate.at<Vec3b>(columns - 1 - j, i) = image.at<Vec3b>(i, j);
        }
    }

    return image_rotate;
}
```

Ahora se muestran los resultados obtenidos de la imagen con la rotación antihoraria:

Figura 2: Imagen original

Figura 3: Imagen con rotación

3. Solicitar el número de rotaciones

Para realizarlo simplemente en el main se puede agregar una variable para que el usuario ingrese la cantidad de veces que va a hacer la rotación, este puede ser hasta 3 veces, ya que si hacemos 4 simplemente haría una rotación de 360°. También para visualizar la imagen debemos hacer tener una variable **Mat** para almacenar la imagen, entonces con un bucle for que itera la cantidad de rotaciones va actualizando la imagen con la función hecha en la actividad 2, terminando muestra la imagen rotada.

```
int main() {
    int number_of_rotations;
    string image_file;
    cout << "Ingrese el nombre de la imagen (con extension): ";
    cin >> image_file;
    cout << "Cuantas rotaciones desea hacer? (1-4): ";
    cin >> number_of_rotations;

    string image_path = "D:/UNSA EPCC/7mo semestre/Computacion Grafica/Unidad
        1/Imagenes con OpenCV/Imagenes/" + image_file;

    Mat image_original = imread(image_path);
    if (image_original.empty()) {
        cout << "No se pudo abrir la imagen." << endl;
        return -1;
    }

    imshow("Imagen Original", image_original);

    if (number_of_rotations < 1 || number_of_rotations > 3) {
        cout << "Numero de rotaciones no valido." << endl;
        return -1;
    }

    Mat image_rotated_antihorary = image_original;
    for (int i = 0; i < number_of_rotations; i++) {
        image_rotated_antihorary = rotate_image_antihorary(
            image_rotated_antihorary);
    }

    imshow("Imagen Rotada Antihorario", image_rotated_antihorary);

    waitKey(0);

    return 0;
}
```

Con esto podremos ver las imagenes con rotación de 1 a 3:

(a) Rotación 1

(b) Rotación 2

(c) Rotación 3

Figura 4: Mostrando las imágenes con las rotaciones.

4. Código completo

Entonces con todas las actividades desarrolladas se presenta el código completo con todos los pasos indicados:

```cpp
#include <iostream>
#include <opencv2/opencv.hpp>
using namespace std;
using namespace cv;

Mat rotate_image_antihorary(const Mat& image) {
    int rows = image.rows;
    int columns = image.cols;
    Mat image_rotate(columns, rows, CV_8UC3);

    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < columns; j++) {
            image_rotate.at<Vec3b>(columns - 1 - j, i) = image.at<Vec3b>(i, j);
        }
    }

    return image_rotate;
}

int main() {
    int number_of_rotations;
    string image_file;
    cout << "Ingrese el nombre de la imagen (con extension): ";
    cin >> image_file;
    cout << "Cuantas rotaciones desea hacer? (1-3): ";
    cin >> number_of_rotations;

    string image_path = "D:/UNSA EPCC/7mo semestre/Computacion Grafica/Unidad 1/Imagenes con OpenCV/Imagenes/" + image_file;

    Mat image_original = imread(image_path);
    if (image_original.empty()) {
        cout << "No se pudo abrir la imagen." << endl;
        return -1;
    }

    imshow("Imagen Original", image_original);

    if (number_of_rotations < 1 || number_of_rotations > 3) {
        cout << "Numero de rotaciones no valido." << endl;
        return -1;
    }

    Mat image_rotated_antihorary = image_original;
    for (int i = 0; i < number_of_rotations; i++) {
        image_rotated_horary = rotate_image_horary(image_rotated_horary);
        image_rotated_antihorary = rotate_image_antihorary(
            image_rotated_antihorary);
    }

    imshow("Imagen Rotada Antihorario", image_rotated_antihorary);

    waitKey(0);

    return 0;
}
```

Referencias

[1] Insightful TScript. *Image Rotation in OpenCV*. Disponible en: `https://insightfultscript.com/collections/programming/cpp/opencv/image-rotation/`

[2] TutorialsPoint. *How to rotate an image in OpenCV using C++*. Disponible en: `https://www.tutorialspoint.com/how-to-rotate-an-image-in-opencv-using-cplusplus`

[3] StackOverflow. *Rotate an image without cropping in OpenCV*. Disponible en: `https://stackoverflow.com/questions/22041699/rotate-an-image-without-cropping-in-opencv-in-c`

[4] OpenCV Forum. *Rotate frame in C++*. Disponible en: `https://forum.opencv.org/t/rotate-frame-in-cpp/9787`