



LAB 1: EFECTOS

VISIÓN POR COMPUTADOR

06/03/2022

ÍNDICE

LAB 1: EFECTOS

Resumen	1
Contraste y brillo	1
Alien	2
Póster	2
Distorsión	2

Aarón Ibáñez Espés - 779088
Sergio Gabete César - 774631

1. Resumen:

Este documento es una memoria de la primera práctica de la asignatura de visión por computador, en ella se han implementado diferentes efectos a fotogramas de un video en tiempo real capturado con la cámara interna del ordenador. El objetivo es aprender a usar openCV y entender los resultados obtenidos tras el procesado de la imagen.

2. Contraste y brillo:

Para modificar el contraste y el brillo se va a utilizar la siguiente fórmula:

$$g(x) = f(x)a + b$$

$g(x)$ será la matriz de píxeles resultante, $f(x)$ la matriz original y a y b dos números. De este modo se pueden elegir una a y b tal que se modifique el rango de valor de intensidad de grises de la imagen. Si se elige una a mayor que uno y una b mayor que cero se aumentará el rango de valores de la imagen, en caso contrario se hará más pequeño. Eligiendo un $a = 2$ y $b = 3$ se puede observar como aumenta la intensidad de la imagen.

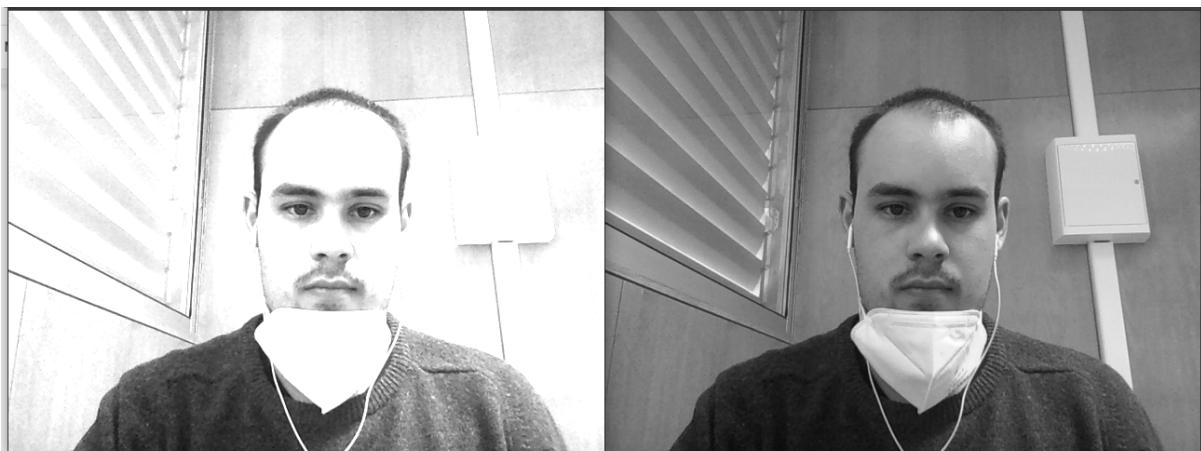


Figura 1. Efecto de ajuste de contraste

A continuación se debe buscar los mejores valores para a y b de tal forma que haya el mejor brillo y contraste. Para encontrar los mejores valores se debe seleccionar un a y b tal que se genere un histograma acumulado en el que sus valores aumentan de forma lineal. Se puede realizar con algoritmo de k-medias pero openCV tiene una función llamada equalizeHist que permite ecualizar el histograma obteniendo una a y b óptimas.

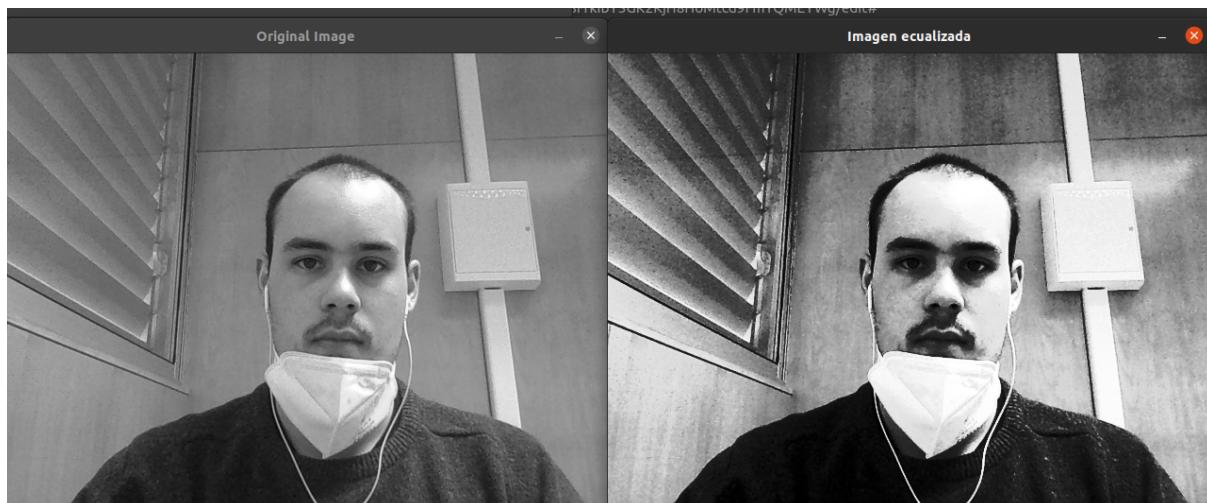


Figura 2. Resultado tras ecualización del histograma

Se puede observar como se ha obtenido una imagen con valores óptimos de brillo y contraste.

3. Alien:

El efecto alien se caracteriza por permitir cambiar el color de la piel a color rojo, verde o azul. Para ello se ha implementado la función `getSkin()`, la cual a partir del fotograma capturado por la cámara, lo transforma a formato `YCrCb` y haciendo uso de la función `inRange()` crea una máscara donde se encuentran los tonos detectados como piel.

La función `inRange` comprueba si los elementos de la matriz pasada como primer argumento se encuentran entre los valores establecidos como rango, en caso de no encontrarse entre dichos números, el pixel se pone a 255, blanco, y si se encuentra fuera de rango, se pone a 0, negro. De esta manera se crea una máscara con la piel detectada en la imagen, que posteriormente se utiliza junto a la función `add()` para modificar el valor del canal deseado y conseguir cambiar el color de la piel.

La función `add()` es la encargada de añadir a los pixeles de la matriz original seleccionados por la máscara, los valores de rojo, verde y azul para así realizar el cambio de color de la piel.

Para poder modificar los valores RGB se han añadido 3 trackbars que permiten al usuario establecer los valores deseados de cada uno de los colores para ser capaces de cambiar el color de la piel a su antojo.

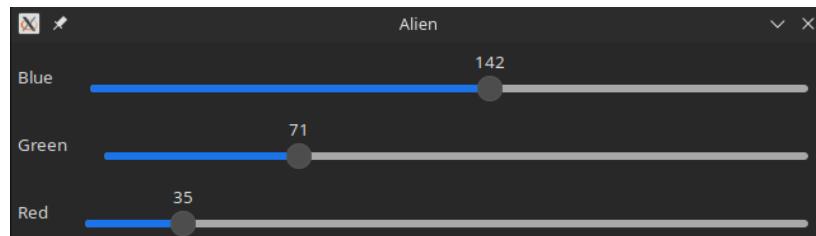


Figura 3. Trackbars de colores

A continuación se muestran tres ejemplos de colores verde, rojo y azul:

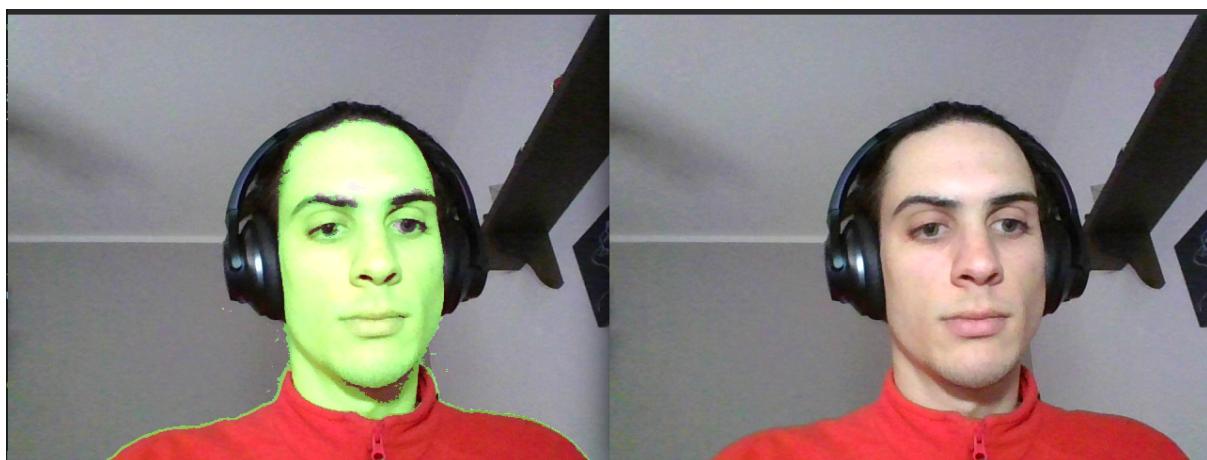


Figura 3. Alien verde



Figura 4. Alien rojo

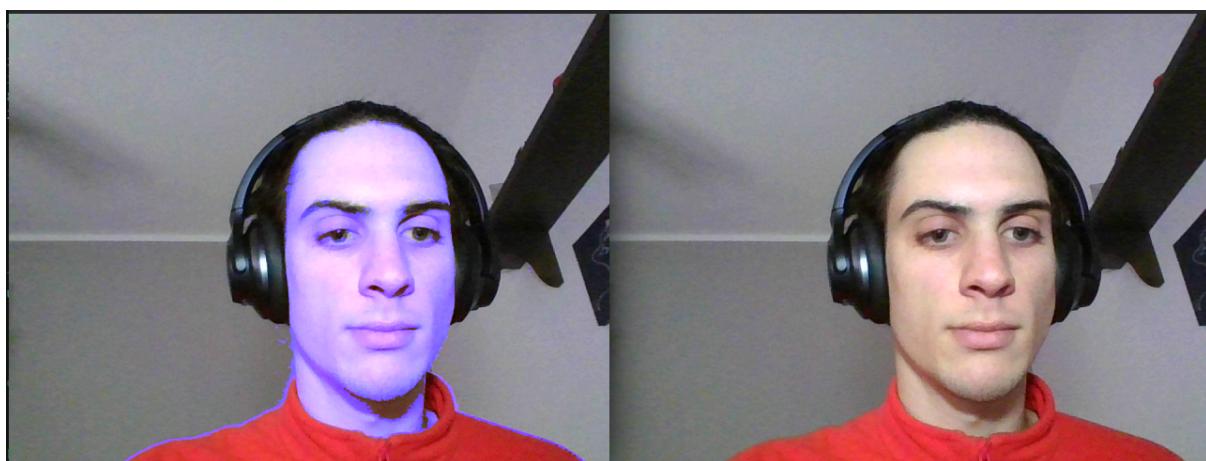


Figura 5. Alien azul

4. Póster:

El efecto póster se basa en reducir el número de colores que tiene una imagen. La idea principal se basa en modificar los valores de los píxeles de la imagen haciendo que los distintos colores tengan el mismo valor, por ejemplo un verde claro y uno oscuro acabarán siendo iguales. Esto se puede realizar dividiendo el valor de un píxel entre un determinado valor, como los valores son enteros entonces se redondeará al valor más próximo. Si se dividen los píxeles entre 64 por ejemplo el píxel con valor 65 y el píxel con valor 90 acabarán con el mismo valor.



Figura 6. Efecto póster

Se puede observar como colores parecidos acaban convertidos en el mismo color.

5. Distorsión:

En la distorsión se pueden encontrar dos efectos: el de barril y el de almohada. Como en openCV no se puede cambiar la posición de los píxeles lo que se hará será asignarle el color de la imagen original al nuevo píxel correspondiente en la nueva imagen. El efecto de barril hace que los píxeles más alejados del centro de la imagen acaben más cerca del centro de la imagen y el efecto de almohada hace que los píxeles más alejados del centro de la imagen acaben más lejos del centro de la imagen.

Se han utilizado las siguientes fórmulas para calcular la nueva posición de los píxeles:

$$x_u = x_d + (x_d - x_{cen}) * K1 * r^2 + (x_d - x_{cen}) * K2 * r^4$$

$$y_u = y_d + (y_d - y_{cen}) * K1 * r^2 + (y_d - y_{cen}) * K2 * r^4$$

$$r^2 = (x_d - x_{cen})^2 + (y_d - y_{cen})^2$$

Si $K1$ o $K2$ son mayor que 0 entonces se aplica un efecto de barril y en caso contrario se aplicará un efecto de almohada.

Dadas las fórmulas se puede observar que la compresión depende de la distancia al cuadrado del centro de la imagen, por tanto cuanto más alejado esté el píxel más se comprime.

En la distorsión de barril se ha usado una $K1 = 4*10^{-6}$ y $K2 = 0$

En la distorsión de almohada se ha usado una $K1 = 2*10^{-6}$ y $K2 = 0$



Figura 7.

6. Dibujo:

Este efecto de dibujo se ha implementado como efecto opcional. Para conseguir el efecto se han realizado de la siguiente manera.

Inicialmente, se convierte la imagen a escala de grises y tras ello se aplica un filtro de desenfoque gaussiano *GaussianBlur()*, para suavizar ligeramente la imagen. Tras esto se utiliza el operador Laplaciano, *Laplacian()*, para detectar los bordes y se aplica el valor absoluto sobre el resultado obtenido. Tras estos pasos se obtiene una imagen de este estilo:



Figura 8. Detección de bordes

Tras esto se utiliza la función *edgePreservingFilter()* para aplicar un filtro de desenfoque a la imagen pero manteniendo los bordes sin desenfocar, así se conserva el contorno del dibujo y finalmente se combinan la matriz de bordes y la que se obtiene de aplicar el filtro de *edgePreservingFilter()* para obtener el dibujo animado en tiempo real.



Figura 9. Resultado final del dibujo