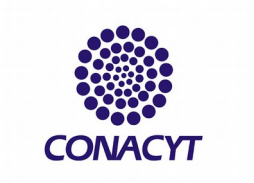
Instituto Nacional de Astrofísica, Óptica y Electrónica



Desarrollo de Talento Especializado 2021: Ciberseguridad

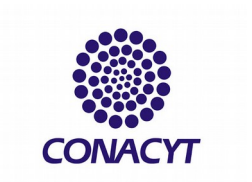
C&C++

Dra. Kelsey Ramírez y Dr.
Maikel Lázaro Pérez Gort

Proyecto 9

Fernando Rojas Ramos
Juan Jose Rosas Varela

Puebla, Puebla a 18 de Mayo de 2021



Indice general

Introducción

El procesamiento de imágenes es un método para realizar algunas operaciones en una imagen, con el fin de obtener una imagen mejorada o extraer información útil de ella. Es un tipo de procesamiento de señales en el que la entrada es una imagen y la salida puede ser una imagen o características / características asociadas con esa imagen. Hoy en día, el procesamiento de imágenes se encuentra entre las tecnologías de rápido crecimiento. También forma un área de investigación central dentro de las disciplinas de ingeniería e informática. El procesamiento de imágenes incluye básicamente los siguientes tres pasos: Importación de la imagen a través de herramientas de adquisición de imágenes; Analizar y manipular la imagen; Salida en la que se puede alterar el resultado de la imagen o informe que se basa en el análisis de la imagen.

Enunciado

Teniendo en cuenta que una imagen se puede representar en una estructura matricial, entonces se puede entender que cada pixel pertenece a dos dimensiones (alto y ancho). Imaginando dicha imagen acostada, cada valor del pixel se puede interpretar como el desplazamiento del mismo en otra dimensión, donde el menor valor posible estará justo en la base y el mayor será el más separado de la misma. Esto puede ser comparado con una vista nocturna de una ciudad en el horizonte, donde los puntos sobre la línea del horizonte representan los menores valores y los puntos sobre los edificios más altos, los valores mayores.
Considerando esta idea, realice el corte de una imagen en esta tercera dimensión. Tenga presente que la imagen debe estar en escala de grises. El nivel del corte será variable entrado por parámetro, cuando los valores de los píxeles son menores a ese nivel, se representarán con el valor 0, y los superiores con el valor 1. Debido a ello, el parámetro que representa los valores de corte tomará valores en el rango entre 0 a 255; y la imagen, resultado del corte consistirá en una imagen binaria (en blanco y negro).

Análisis y Diseño

El procesamiento de imágenes nos ayuda a obtener información relevante de las imágenes y a tomar decisiones basadas en esa información. El procesamiento de imagenes esta ligada fuertemente con los pasos básicos para una aplicación típica de visión por computadora.
Adquisición de imagen, Manipulación de imagen, Obtener información relevante y Toma de decisiones.
A continuación mostramos nuestra metodología a seguir para la solución de nuestro proyecto

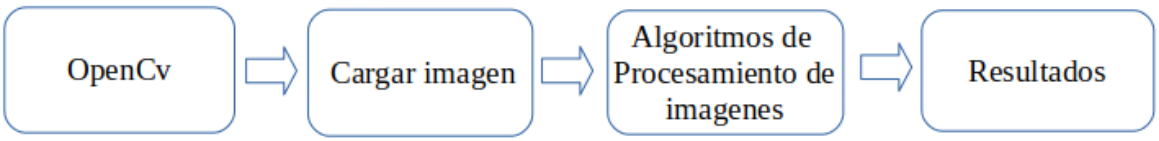


Figura 1.- Metodología.

De manera resumida mostramos la metodología que se siguió para el desarrollo de nuestro proyecto, Se pretende usar la librería de OpenCV por la facilidad que nos brinda en el uso de funciones de procesamiento de imágenes. Una fase importante es la importación de la imagen para poder iniciar con los siguientes pasos. Los algoritmos que se aplicaran son, convertir la imagen a escala de grises, normalizar y binarizar. Por ultimo mostramos para cada caso las imágenes con el resultado del procesamiento hecho con las funciones integradas en OpenCV.

Diagrama de Flujo

A continuación se muestra la secuencia de pasos para el desarrollo de nuestro proyecto, el Diagrama de flujo 1 que se presenta contiene los pasos a seguir para lograr con éxito su implementación.

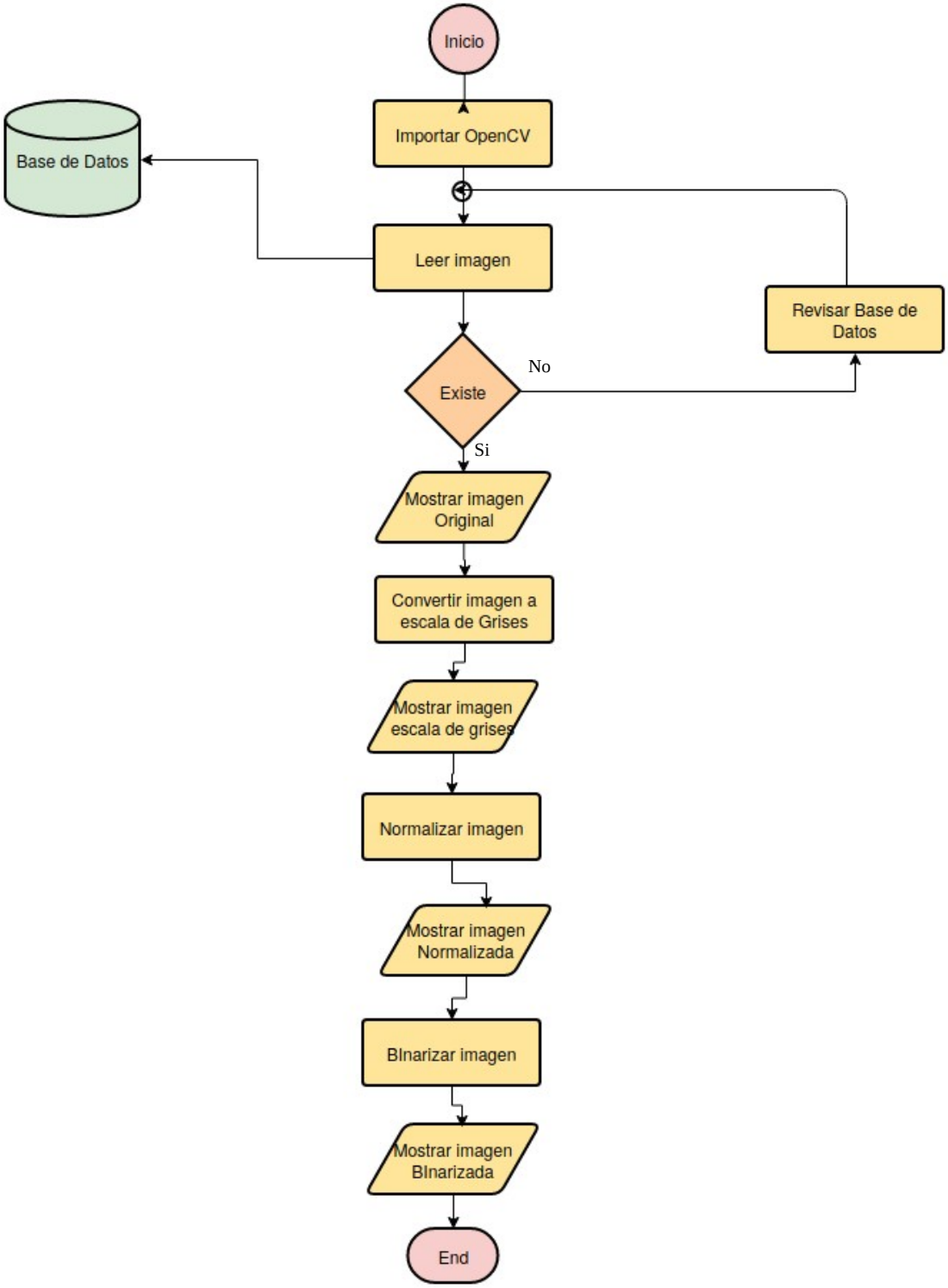


Diagrama de flujo 1: Representa la secuencia de pasos de nuestro proyecto, como también el orden de ejecución de nuestro código.

Imagen

Una imagen se define como una función bidimensional, F (x, y), donde x e y son coordenadas espaciales, y la amplitud de F en cualquier par de coordenadas (x, y) se denomina intensidad de esa imagen en ese punto. Cuando los valores de x, y y amplitud de F son finitos, lo llamamos imagen digital.

Imagen como matriz

Como sabemos, las imágenes se representan en filas y columnas tenemos la siguiente sintaxis en la que se representan las imágenes:

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & f(0,2) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & f(1,2) & \dots & f(1,N-1) \\ . & . & . & & . \\ . & . & . & & . \\ . & . & . & & . \\ f(M-1,0) & f(M-1,1) & f(M-1,2) & \dots & f(M-1,N-1) \end{bmatrix}$$

Figura 2.- Representación de una imagen en Matriz.

Imagen normalizada

La normalización se hace necesaria, para tener una cierta independencia de las propiedades de la imagen, como lo son el brillo y el contraste, y así poder comparar huellas por su índice de calidad, que más adelante definiremos.

$$N(x,y) = \frac{(N^o\, Niveles - 1)}{(max(I) - min(I))}.(I(x,y) - min(I)).$$

Donde:

- I(x,y), nivel de gris de la imagen en la coordenada (x,y).
- min(I),max(I): mínimo y máximo nivel de gris en la imagen respectivamente.
- N(x,y), nivel de gris de la imagen normalizada en la coordenada (x,y).

Imagen Binaria

La imagen binaria, como su nombre indica, contiene solo dos elementos de píxeles, es decir, 0 y 1, donde 0 se refiere al negro y 1 al blanco. Esta imagen también se conoce como monocromática.
En el caso más simple, una imagen puede consistir en un solo objeto o varios objetos separados de intensidad relativamente alta, vistos sobre un fondo de intensidad relativamente baja. Esto permite la separación figura / suelo mediante umbrales. Para crear la imagen binaria de dos valores, se puede aplicar un umbral simple para que todos los píxeles en el plano de la imagen se clasifiquen en píxeles de fondo y de objeto. Luego, se puede construir una función de imagen binaria de manera que los píxeles por encima del umbral estén en primer plano "1" y por debajo del umbral estén en el fondo "0".

Esta operación de binarizacion se puede expresar como:

$$\texttt{dst}(x,y) = \begin{cases} \texttt{maxVal} & \text{if } \texttt{src}(x,y) > \texttt{thresh} \\ 0 & \text{otherwise} \end{cases}$$

Entonces, si la intensidad del píxel src (x, y) es mayor que el umbral, entonces la nueva intensidad de píxel se establece en un MaxVal. De lo contrario, los píxeles se establecen en 0.

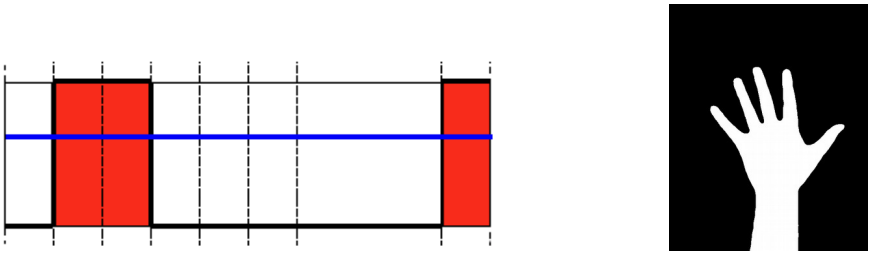


Figura 3.- Ejemplo de Binarización y un valor de corte.

Detalles de implementación

El desarrollo de este trabajo esta totalmente enfocado en el uso de la librería OpenCV, sin embargo es importante mencionar el proceso de instalación debido a que dependiendo a el sistema operativo cambia el proceso de instalación. Se anexa el link con las instrucciones de instalación de la librería de OpenCV: http://www.codebind.com/cpp-tutorial/install-opencv-ubuntu-cpp/

OpenCV es una biblioteca libre de visión artificial originalmente desarrollada por Intel. OpenCV significa Open Computer Vision (Visión Artificial Abierta). Desde que apareció su primera versión alfa en el mes de enero de 1999, se ha utilizado en una gran cantidad de aplicaciones, y hasta 2020 se la sigue mencionando como la biblioteca más popular de visión artificial. Detección de movimiento, reconocimiento de objetos, reconstrucción 3D a partir de imágenes, son sólo algunos ejemplos de aplicaciones de OpenCV.

Su popularidad se debe a que es:

- libre, publicada bajo licencia BSD, que permite que sea usada libremente para propósitos comerciales y de investigación
- multiplataforma, para los sistemas operativos GNU/Linux, Mac OS X, Windows y Android, y para diversas arquitecturas de hardware como x86, x64 (PC), ARM (celulares y Raspberry Pi)
- documentada y explicada: la organización tiene una preocupación activa de mantener la documentación de referencia para desarrolladores lo más completa y actualizada posible, ejemplos de uso de sus funciones y tutoriales accesibles al público no iniciado en visión artificial, además de difundir y fomentar libros y sitios de formación.

Pruebas

Como primer paso, leemos la imagen "Horizonte" esta imagen es un ejemplo acorde a la descripción del planteamiento del problema y asemejar el comportamiento con una imagen similar Figura 1. Para hacerlo, una llamada a la función cv :: imread carga la imagen usando la ruta del archivo especificada por el primer argumento. El segundo argumento es opcional y especifica el formato en el que queremos la imagen. Esto podría ser: IMREAD_COLOR carga la imagen en formato BGR de 8 bits. Este es el valor predeterminado que se usa aquí. IMREAD_UNCHANGED carga la imagen tal cual (incluido el canal alfa, si está presente) IMREAD_GRAYSCALE carga la imagen como una de intensidad Después de leer la imagen, los datos se almacenarán en un objeto cv :: Mat.
A continuación se muestra el código que nos ayuda acceder a la imagen en nuestro directorio donde esta almacenado.

```
std::string image_path = samples::findFile("Horizonte.jpg");
Mat img = imread(image_path, IMREAD_COLOR);
```

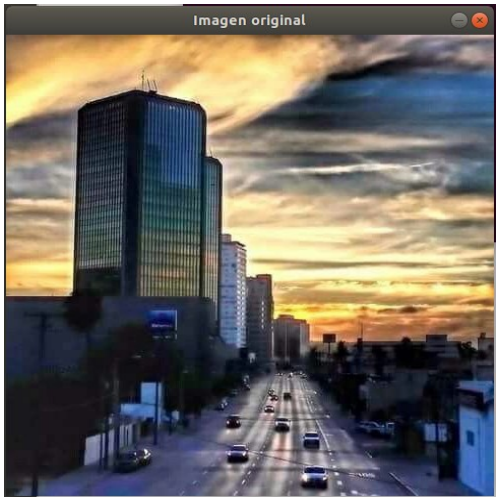


Figura 1.- Se realiza la lectura de una imagen a color desde nuestro directorio, la lógica de nuestro programa revisa si la imagen existe o no, con un mensaje a consola si no se encontró imagen con el nombre correspondiente en su llamada.

La librería de OpenCV con c++ nos permite hacer la lectura de diferentes formatos de imagen por ejemplo: mapa de bits de Windows (bmp), formatos de imagen portátiles (pbm, pgm, ppm). También puede cargar formatos de imagen como JPEG (jpeg, jpg, jpe), archivos TIFF (tiff, tif) y gráficos de red portátiles (png).

Luego, la imagen se muestra usando una llamada a la función cv :: imshow. El primer argumento es el título de la ventana y el segundo argumento es el objeto cv :: Mat que se mostrará. Debido a que queremos que nuestra ventana se muestre hasta que el usuario presione una tecla (de lo contrario, el programa terminaría demasiado rápido), usamos la función cv :: waitKey cuyo único parámetro es cuánto tiempo debe esperar una entrada del usuario (medido en milisegundos). Cero significa esperar para siempre. El valor de retorno es la tecla que se presionó.

El siguiente paso de nuestro proyecto en C ++ fue convertir una imagen BGR a una imagen en escala de grises usando la función cvtColor presente en la biblioteca OpenCV como se puede ver en la Figura 2. Las imágenes en escala de grises son necesarias para cualquier función en OpenCV y también son útiles para distinguir la intensidad entre los píxeles, ya que no es necesario enviar información adicional. En los siguientes pasos el desarrollo del proyecto se trabajara con la imagen en escala de grises como lo indica el problema.

Figura 2.- La escala de grises es la representación de una imagen en la que cada pixel se dibuja usando un valor numérico individual que representa su luminancia, en una escala que se extiende entre blanco y negro.

Para cada instancia de imagen que se requiere implementar dentro del código se usa la siguiente instrucción utilizada **cv::Mat imagen_gris** y **cv::cvtColor(imagen_original, imagen_gris, cv::COLOR_BGR2GRAY)**; para obtener la imagen resultante en tonos de grises, posteriormente mostramos el resultado.
El siguiente paso que realizamos fue la normalización de la imagen en escala de grises en el rango de [0, 1], la imagen en escala de grises los pixeles se encuentran dentro del rango de [0, 255], entonces aplicamos una regla de tres que hace correspondencia a nuestro rango de tonalidades de grises de nuestra imagen obtenida anteriormente. **cv::Mat img_normalizada; imagen_gris.convertTo(imagen_normalizada, CV_32F, 1.0 / 255, 0);**

Figura 3.- En el procesamiento de imágenes, la normalización es un proceso que cambia el rango de valores de intensidad de píxeles. Las aplicaciones incluyen fotografías con poco contraste debido al deslumbramiento, por ejemplo. La normalización a veces se denomina estiramiento de contraste o estiramiento de histograma. En campos más generales del procesamiento de datos, como el procesamiento de señales digitales, se denomina expansión de rango dinámico.

La binarización de imágenes es el proceso de tomar una imagen en escala de grises y convertirla a blanco y negro, esencialmente reduciendo la información contenida en la imagen de 256 tonos de gris a 2: blanco y negro, una imagen binaria. Esto a veces se conoce como umbral de imagen, aunque el umbral puede producir imágenes con más de 2 niveles de gris. Es una forma o segmentación, mediante la cual una imagen se divide en objetos constituyentes. Esta es una tarea que se realiza comúnmente cuando se intenta extraer un objeto de una imagen. Sin embargo, al igual que muchas operaciones de procesamiento de imágenes, no es trivial y depende únicamente del contenido de la imagen. El truco es que las imágenes que pueden parecer fáciles de convertir a blanco y negro muchas veces no lo son.

El proceso de binarización funciona al encontrar un valor umbral en el histograma, un valor que divide efectivamente el histograma en dos partes, cada una de las cuales representa uno de dos objetos (o el objeto y el fondo). Para nuestra tarea buscamos que en determinado valor de corte(valor en rango de los pixeles en escala de grises 0-255), es decir si asignamos el valor de corte igual a 100 en ese punto todos los puntos por debajo de ese valor toman el valor de 0 y por encima el valor de 1. Como se muestra en las siguiente Figura 4 un conjunto de imagenes con diferentes valores de corte y la representación binaria después de realizar el procesamiento con funciones de OpenCV.

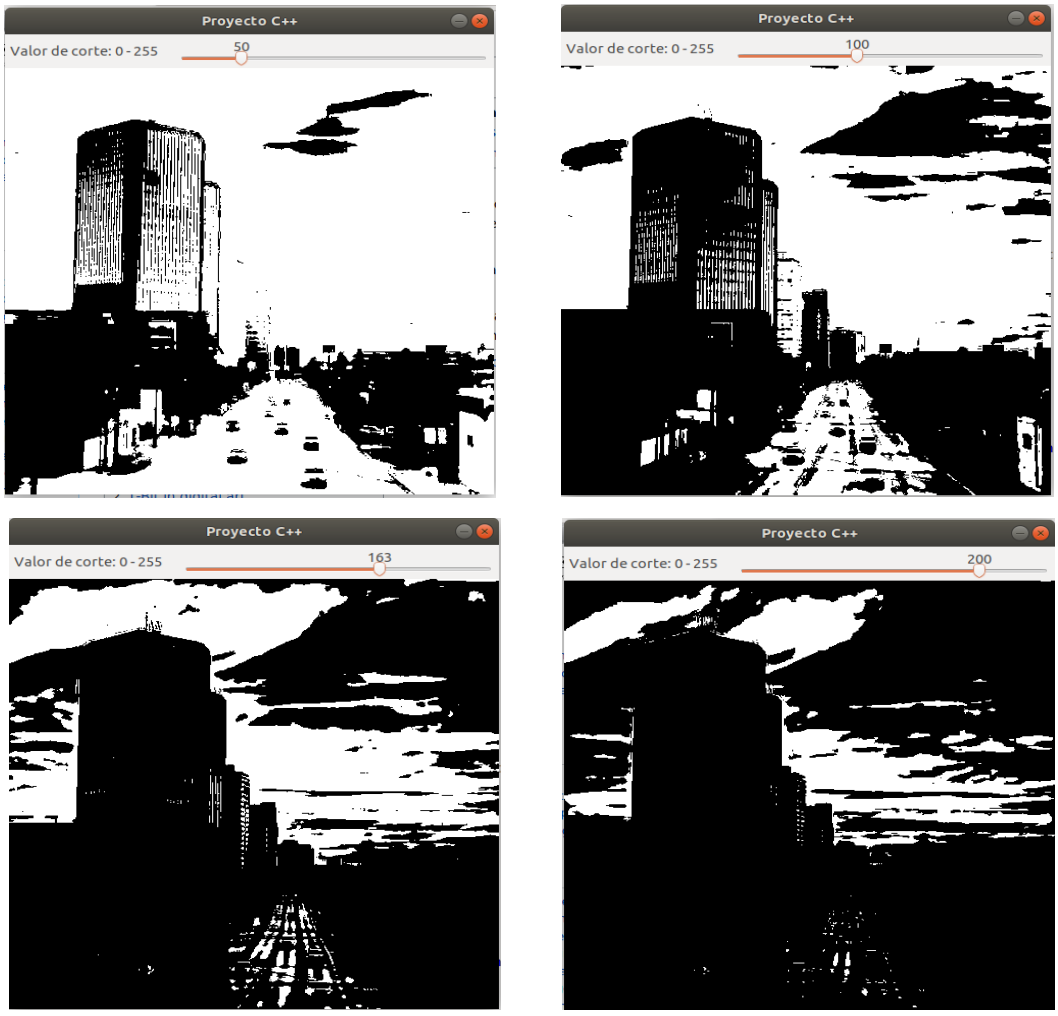


Figura 4.- De acuerdo a los requerimientos de problema, podemos observar los resultados obtenidos con diferentes valores de corte y la binarización correspondiente. Se puede ver la correspondencia que existe entre mas aumenta el valor de corte se va oscureciendo mas la imagen, con valores pequeños la imagen se colorea mas de blanco.

Pruebas de estrés

para nuestras pruebas de estrés usamos como entrada diferentes formatos de imagen para validar nuestro experimento y ver su funcionalidad.

La función imread() de OpenCV nos permite validad el formato de entrada de nuestra imagen, los formatos usados fueron: jpg, png, raw y bmp.

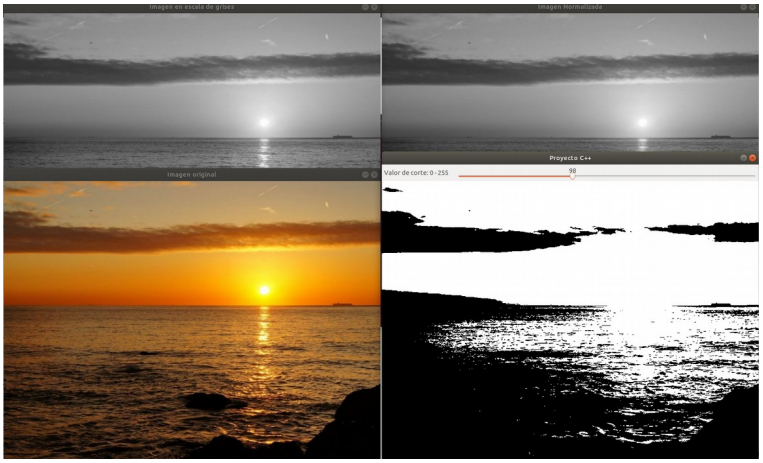


Figura 5.- Formato .png

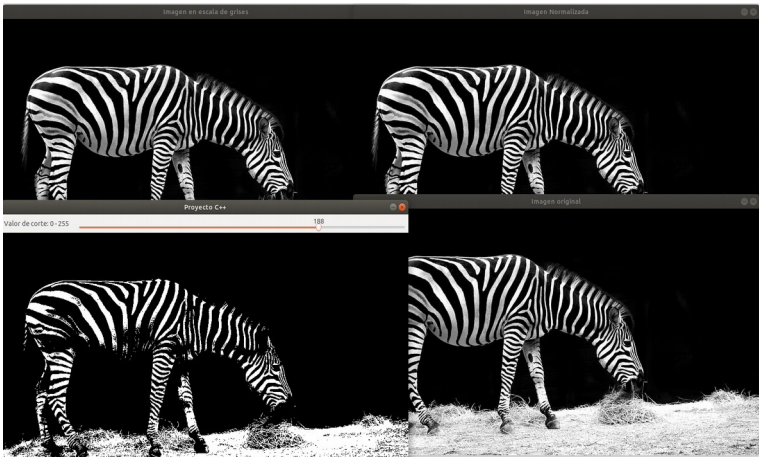


Figura 6.- Formato .raw

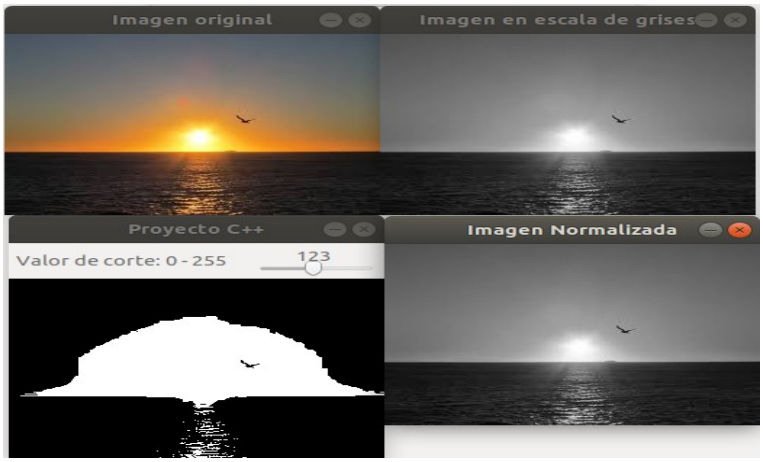


Figura 7.- Formato .bmp

La función imread() detecta cuando se le pasa un formato de archivo no valido, por ejemplo si se le pasa un nombre vacío o si es un archivo pdf, es decir la función es capaz de identificar si se le ha pasado un formato valido de imagen.

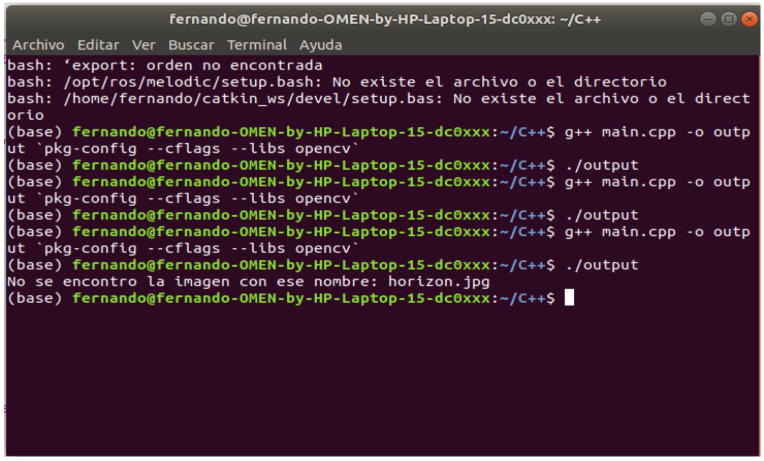


Figura 8.- Nombre de imagen que no existe

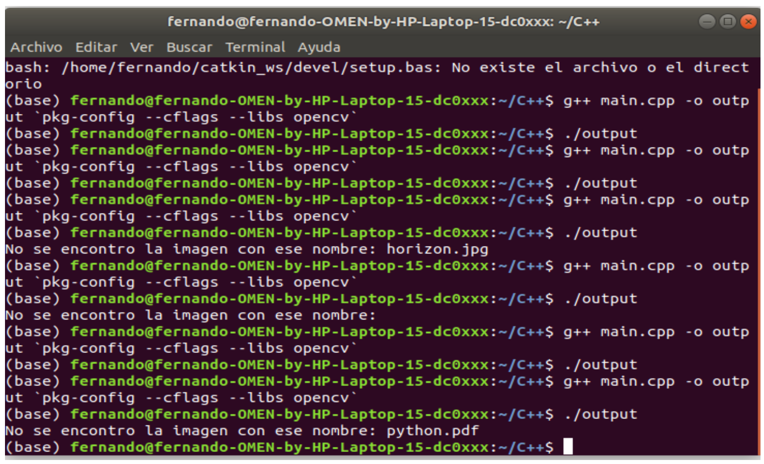


Figura 9.- Formato no valido de imagen

Conclusiones

Con el trabajo presente se implementaron funciones de la biblioteca de OpenCV para dar solución a nuestro proyecto, se logro la instalación, configuración y uso de la biblioteca para su correcta compilación y ejecución en nuestro entorno de trabajo, en nuestro caso en Linux.

Se decidió usar la biblioteca de OpenCV por su fácil integración de funciones parametrizadas para el desarrollo de prototipos que requieren el uso de procesamiento de imágenes, logrando así el aprendizaje de la biblioteca para realizar procesamiento de imágenes.

Es importante mencionar que se debe de conocer los algoritmos de procesamiento de imágenes para poder aplicar las funciones de OpenCV de manera correcta. De esta forma se pueden realizar proyectos complejos de manera eficiente y rápida.

Con este proyecto se aprende una nueva herramienta OpenCV disponible para el lenguaje de programación C+ + para el procesamiento de imágenes.