Report

Overview

Through the implementation of all kinds of filters and effects, we can have a deeper understanding of images, the basics of computer vision, which is also the fundamental to master more complicated tools and techniques of computer vision (CV). After applying several basic filters on one image, we can find more information hidden behind the colorful pixels.

Although the final goal of computer vision is to give computers (super) human-level perception, before achieving that long-term goal, we need to empower computers to deal with some problems with digital "eyes" in certain situations/conditions; that is the objective of this assignment.

Task 1 & 2 Display image and live video

Key	Action	Additional Information
q / esc	Quit	
s	Save the image	The format of image name fIlterName_timestamp.format
g	Grayscale	Using cvtColor
h	Grayscale	Customized
b	Blur	Using separable Gaussian filter
x	Sobel X	
y	Sobel Y	
m	Magnitude	
i	Blur and Quantize	
c	Cartoon	
↑ (up arrow)	Increase brightness	Can be applied with one other filter
↓ (down arrow)	Decrease brightness	
→ (right arrow)	Increase contrast	Can be applied with one other filter
← (left arrow)	Decrease contrast	
‘ ’ (space)	Remove all filter	Also reset the brightness and contrast

Task 3 Display greyscale live video

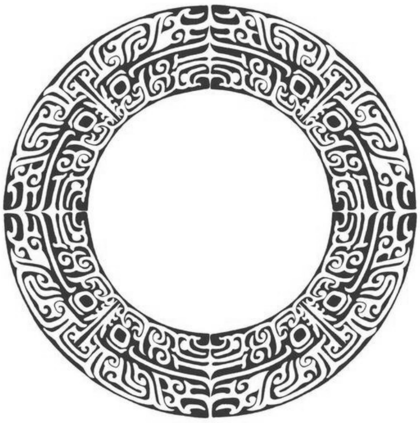
Original Image

Grayscale Image using cvtColor

According to the opencv document, RGB have different weights when calculating the grayscale value Y, which is calculated using the formula $Y←0.299·R+0.587·G+0.114·B$. This formula takes into account the contributions of the red, green and blue channels and combines them into a single channel of grayscale information. This allows us to effectively condense the three channels of color information into one monochromatic channel and eventually get our Grayscale Image. This process of condensing the three color channels into one is a very effective way to transform a full-color image into a monochrome one.

Task 4 Display alternative greyscale live video

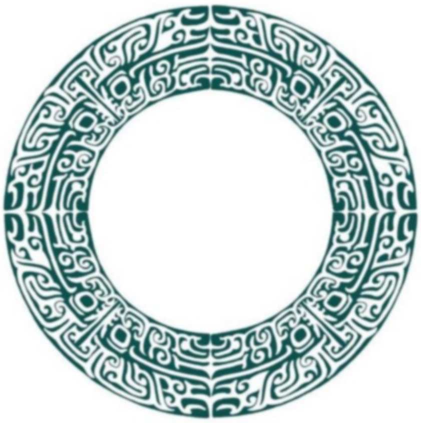
Similar to cvtColor, I used the values of RGB to get my grayscale value, but each color had the equal weight. In order to achieve this, I calculated the average of the three colors, Red, Green and Blue, and then assigned the same value to each of the channels. This allowed me to keep the original colors in the image while still obtaining a grayscale image. The formula for this was simple: average = (R+G+B) / 3, with R = G = B = average. This method of obtaining a grayscale image by assigning the same values to each of the colors proved to be quite effective, as it gave me a good representation of the original image in a single-toned format.



Original Image

Customized greyscale image

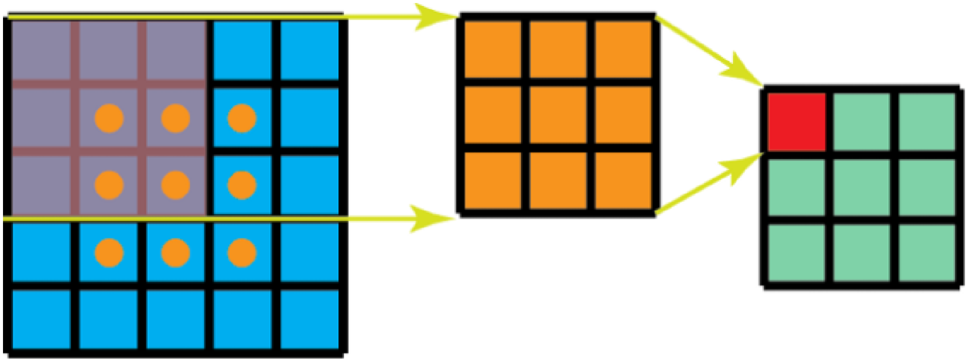
Task 5 Implement a 5x5 Gaussian filter as separable 1x5 filters



Original Image

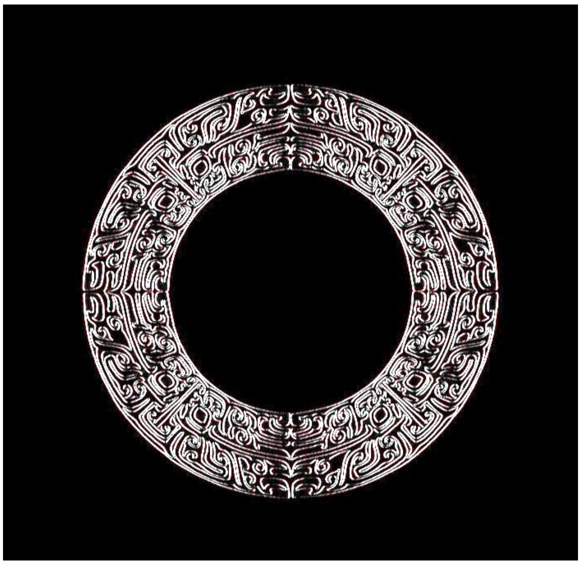
Blurred image

We used Gaussian Filter to blur images, but to be more time efficient, we use separable filter to decrease time complexity.

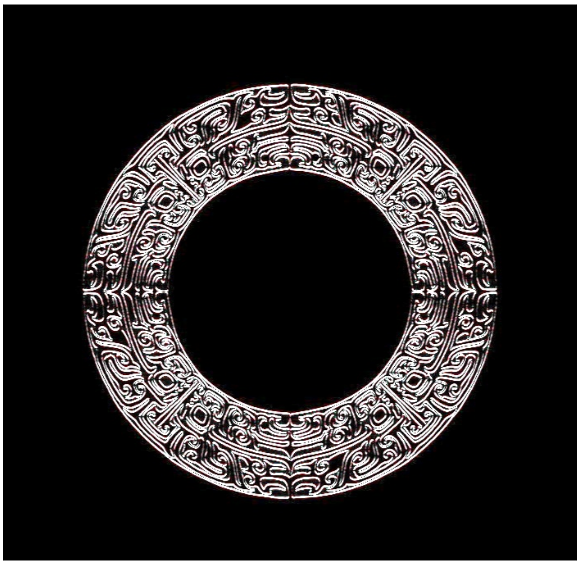


A simple example of implementation

Task 6 Implement a 3x3 Sobel X and 3x3 Sobel Y filter as separable 1x3 filters



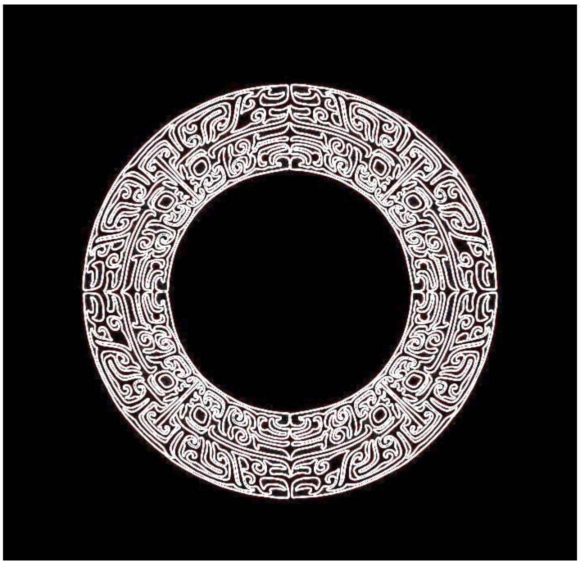
Sobel X



Sobel Y

Task 7 Implement a function that generates a gradient magnitude image from the X and Y Sobel images

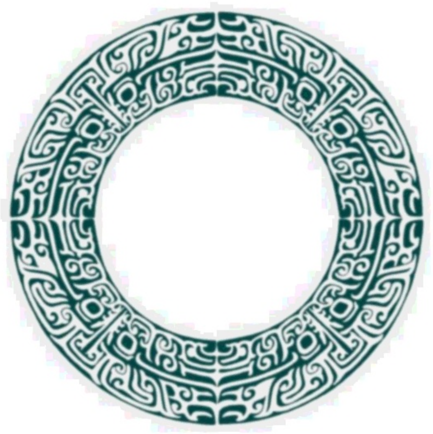
Original Image



Gradient magnitude image

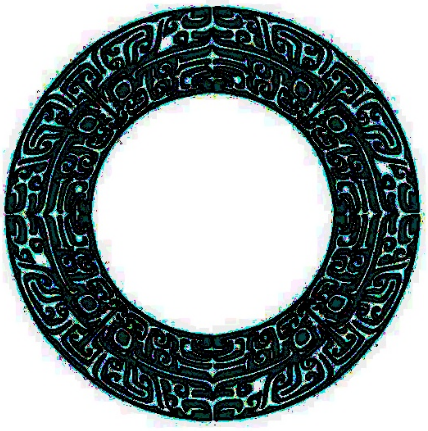
Task 8 Implement a function that blurs and quantizes a color image

Original Image



Blurred/quantized image

Task 9 Implement a live video cartoonization function



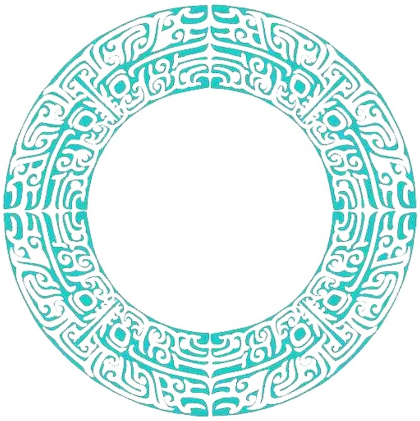
Original Image

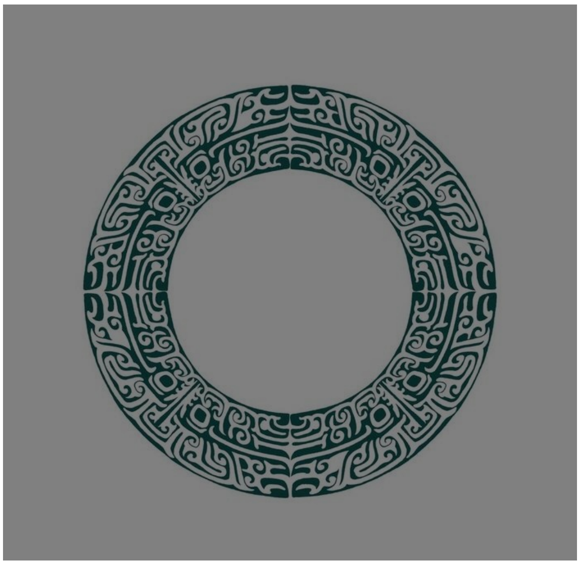
Cartoon Image

Task 10 Allow the user to adjust brightness or contrast

I use the `convertScaleAbs()` function to control brightness and contrast. It takes four parameters: `src`, `dst`, `alpha`, and `beta`. Within the parameters, alpha controls brightness and beta controls contrast.

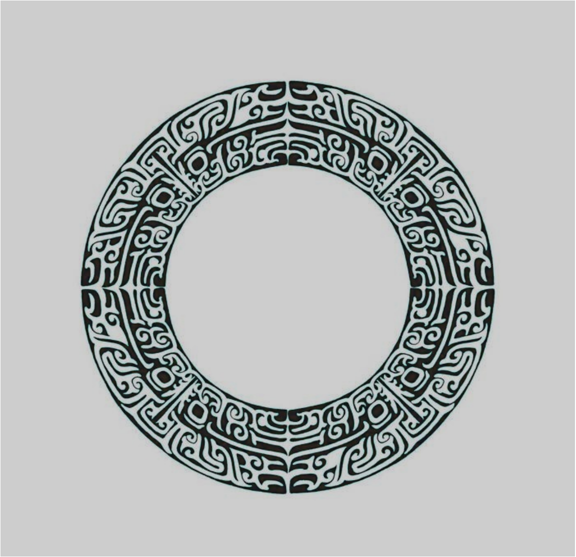
I equip the four directional arrows with the capability to adjust the values of alpha and beta, thereby allowing you to control the brightness or contrast of the image. Furthermore, this power is independent of any other filters that may have been applied beforehand, meaning that you can still make alterations to the brightness or contrast even if a filter has already been applied to the image. This provides you with the flexibility to make further adjustments to the brightness and contrast without having to undo any of the other filters you may have used.





Brightest

Darkest



Most contrast

Less contrast

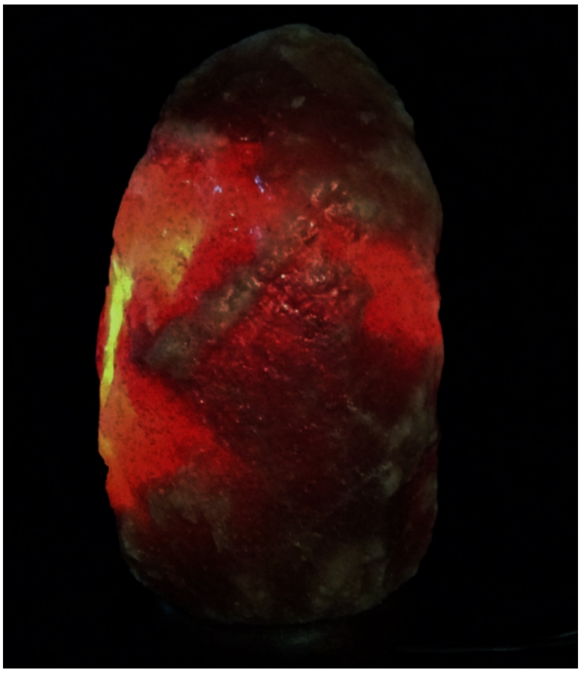
Extensions

1. Implement your effects for still images and enable the user to save the modified images.

   When you press the "q" key, the current image with the applied filter, brightness, and contrast is saved. The image is automatically renamed with the format "filterName_timestamp.png", so you can easily identify which filter was applied to the saved image. To make sure the file is properly saved, the timestamp is automatically added to the filename, so that you can easily find the newly created image in the directory. This also ensures that no previous image is accidentally overwritten by the new image. Additionally, the filter name is included in the filename to make it easier to distinguish between different images with the same timestamp.

2. Automatic brightness and contrast optimization

   Utilize the histogram to find the optimal values of brightness and contrast to make the image appear more vivid and visually appealing. By adjusting the brightness and contrast, you can make the colors of the image pop and create a more dynamic look. Additionally, by increasing the brightness and contrast, you can help to make the details of the image stand out more, creating a more defined look.



Original image

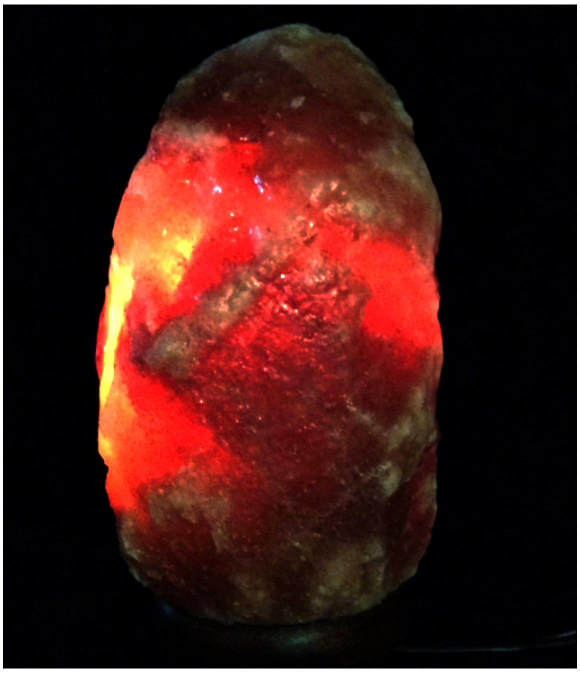


Image after automatic brightness and contrast optimization

Reflection

By completing project 1, I have developed a comprehensive understanding of the foundations of opencv and have acquired a deep knowledge of computer vision. I have been able to learn the basics of opencv, such as image processing, object detection, and feature extraction, as well as gain a better understanding of the potential of computer vision and its practical applications. Through this project, I have achieved a greater mastery of opencv, and I am confident that I can confidently apply the concepts to future projects.

Acknowledgement

1. "Color Conversions." *OpenCV*, https://docs.opencv.org/3.4/de/d25/imgproc_color_conversions.html#color_convert_rgb_gray.

2. Atam, Akshay. "Separable Convolution in Machine Learning." *OpenGenus IQ: Computing Expertise & Legacy*, OpenGenus IQ: Computing Expertise & Legacy, 8 Aug. 2021, https://iq.opengenus.org/separable-convolution/.