Cartoon-ify

Feziwe M Shongwe
melvin.shongwe@gmail.com

ABSTRACT

Before using images on various platforms, including social media platforms, different people prefer different image processing techniques. Some prefer to apply filters to images, change the contrast of an image, introduce shadows in an image, change an image to a cartoon, and so on. This paper focuses primarily on the technique of converting an actual image to a cartoon image. There are several methods for converting various types of images to cartoons, including the use of Photoshop websites and applications. This paper introduces a tool for converting images to cartoons that uses python libraries such as scikit-image, OpenCV, and others to segment the input image using k-means clustering.

I. INTRODUCTION

[5] defines cartoons as amusing drawings in newspapers or magazines. However, because of the vibrant colours, zany animation, and kid-friendly plots designed to capture and hold their attention, many adults regard cartoons as something for children. Despite popular belief, some adults enjoy cartoons as well, despite the theories of those who believe cartoons are only for children.

Cartoons are effective because they provide a quick way to smile when compared to actual images. That is why, for years, businesses and publications have used them successfully in advertising. Actual images add impact, but it is the sharing of a smile that makes cartoons such an effective tool. Cartoons are more effective at explaining complex issues, revealing simple truths that we all share, and providing much-needed visual interest and levity. [2] discussed the effectiveness of cartoons in blogs and related content in terms of engagement and sharing.

There are various platforms that convert an image to a cartoon image, but these platforms are costly and require users to purchase packages. To address this limitation, this paper will develop an open-source tool that will use Digital Image Processing techniques to convert actual images to cartoon images for various types of implementations. Users can then share these converted images on any social media platform, save them for later use, share them with loved ones, or do whatever they want with them. The conversion tool will be implemented by segmenting the image using K-means clustering, creating an edge mask of the actual image, determining dominant colours from the image, and combining all these processes to produce the cartoon image.

II. METHODOLOGY

In this section, we will go over the steps that were used and taken in this project to create an output image of a cartoon from a normal input image. Loading and creating an edge mask of the input image, extracting dominant colours, combining the edge mask and the image of the dominant colours extracted (Cartoon-ify), and reducing noise from the image formed after the combination of the two images using a smoothing filter are the steps taken in this project.

A. Loading and Creating an edge mask of the input image

I used Python libraries to complete these tasks (Loading the Image and Creating an edge mask). After loading an image, I converted it from a colour image to a greyscale image in order to prepare it for edge detection. Because of the limitations of global thresholding, having the same threshold value T may not be sufficient for different images as it may only work on a specific part of some images due to differences in lighting, shadowing, and so on. As a result, the edge mask is obtained by applying local thresholding to a greyscale image. The image below shows the effect of global and adaptive thresholding (local thresholding) on an image.

B. Extracting dominant colours (Colour Quantization)

I formed an edge mask of the input image in the previous subsection. As I was segmenting colours in this case, I used a regional segmentation approach (K-means Clustering) based on looking for clusters in related data. I segmented the input image to obtain the dominant colours in the image; this process is known as colour quantization. Because K-Means clustering is an iterative method, I had to perform 20 iterations for the epsilon of 1 to converge. Finally, I used different clusters (4 and 8) to examine the effect of clustering, as some images may have fewer colours than others.

C. Formation of the Cartoon Image (Cartoon-ify)

Bringing everything together, I combined the previously created edge mask and the colour quantized image. After combining the two (edge mask and colour quantized), the output image has noise that needs to be smoothed to reduce the noise of the sharp transitions of colour intensities. I used a median filter because it provides better noise reduction for different types of random noise with less blurring than linear smoothing filters of similar size (e.g., Gaussian filter).

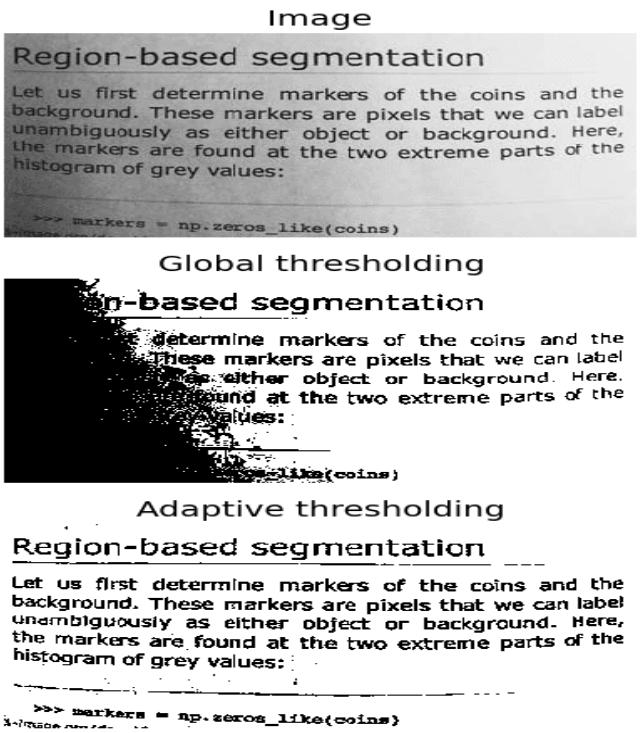

Fig. 1. [9] illustrated the effect of different thresholding methods on an image.
Top: Original Image, **Middle**: Global thresholding and **Bottom**: Adaptive thresholding.

III. Experimental setup

In this section, I'll go over how I successfully implemented the image converter (Carton-ify) tool. The section will primarily concentrate on the data (images) used to test the tool's effectiveness and the fundamental tools used to implement the image conversion tool.

A. Data

As the aim of this project is to implement a tool for image conversion from an actual image to a cartoon, the data I used in the implementation of the tool are images and were collected online from different open-source websites (Unsplash and iStock photo).

B. Toolbox

The primary tools in this project are Python libraries that perform all the Digital Image Processing techniques. For loading and converting the image to grayscale format, and thresholding the input image to obtain the edge mask, the scikit-image library was used and for noise reduction of the output image, scipy was used. OpenCV, on the other hand, was used to segment the image using K-means clustering. Finally, numpy and matplotlib were used for basic operations such as image reshaping and visualization.

IV. Results and Discussion

In this section, I tested the implemented tool's ability to convert images using various images. I compared the results from the different clusters (4 and 8) used for a specific image to see which one produced the best results. I used various types of images to determine the cartoon-ify effect on various types of images containing people's faces (different skin tone), objects, and animals since this tool will be used for a variety of applications depending on the users. I only included one figure illustrating the results of the converted image for test case 1, where I used a human (male) input image, and the other cases are in the appendix.

A. Test case 1 : Human(Male) Image

Fig. 2. Input Image [1]

Fig. 3. Edge mask of the input image

Fig. 4. Converted image using 4 Clusters

Fig. 5. Converted image using 8 Clusters

After evaluating the effect of the various clusters used in this project, depending on the intensity variation in the image. The tool performs well in images with multiple colours when using 8 clusters, and it also performs well in images with fewer colours when using 4 clusters for image quantization.

As previously stated, I used various images to test the tool's effect. After converting human images, the tool appears to be agile and works well with different skin tones. This demonstrates that the implemented tool does not favour a specific skin tone or image background. Similarly, to the object images, the results show that the tool will be useful to users because it not only converts human images but also converts images of houses and cars. The appendix contains the results of the remaining test cases, which show the actual image, edge mask created, converted image by 4 clusters, and converted image by 8 clusters.

V. Conclusion

Converting an image to a cartoon may seem to be an impossible or difficult task to perform. The proposed tool makes it more possible and less complex than most current tools for this task, which require the user to adjust various effects that eventually turn the image into a cartoon. In this project, I created a tool that will accept images of any shape that a user wants to cartoon-ify and save them for personal use or share them across multiple platforms. The proposed tool appears to work well in images where the object(s) are not affected by noise or other factors. Furthermore, the technique of implementing two (2) clusters aids in producing a well-converted output based on how the colours are distributed in the input image.

APPENDIX

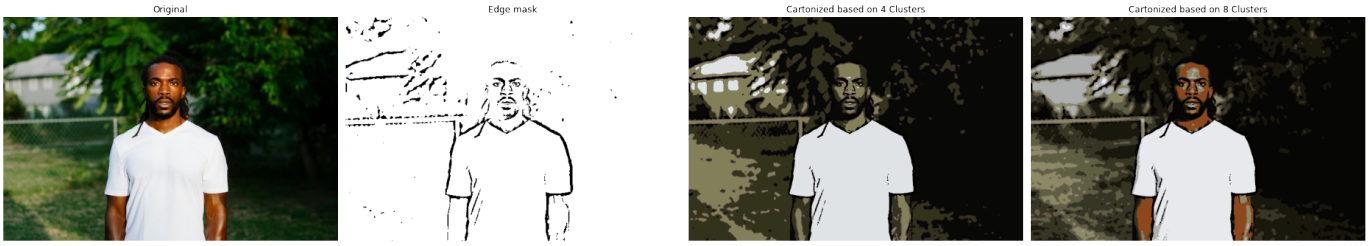

Fig. 6. Test case 2 : Human (Male) [11]

Fig. 7. Test case 3: Human (Female) [10]

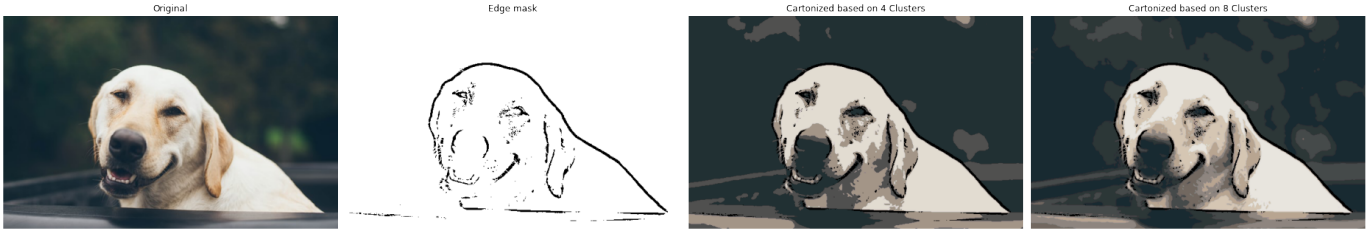

Fig. 8. Test case 4 : Animal (Dog) [3]

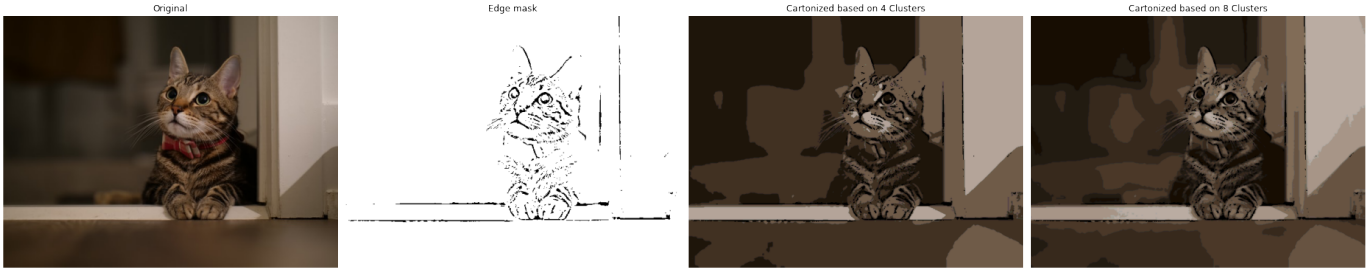

Fig. 9. Test case 5: Animal (Pet) [6]

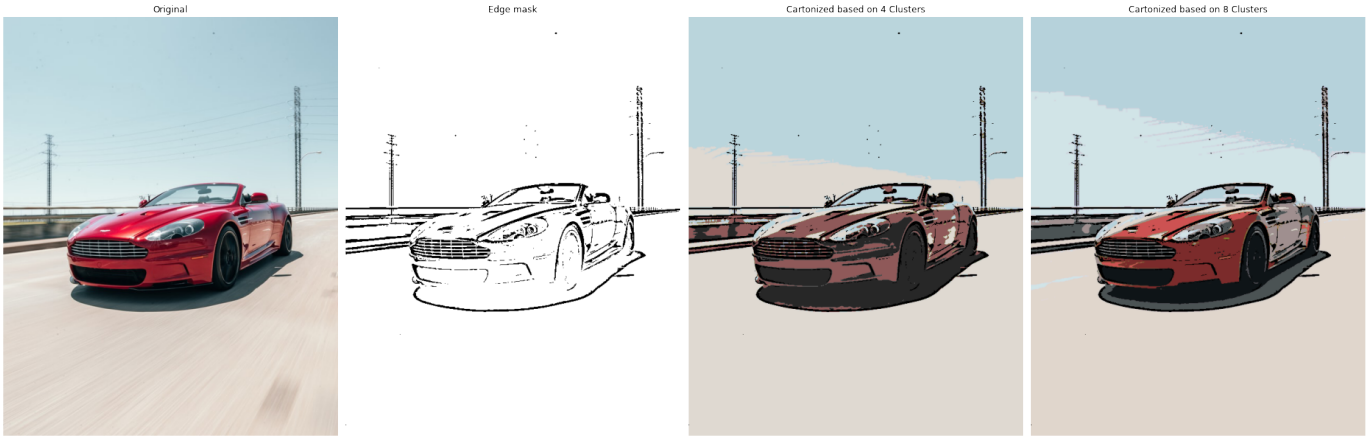

Fig. 10. Test case 6:Car [4]

Fig. 11. Test case 7: House [7]

Fig. 12. Test case 8: Crowd [8]

REFERENCES

[1] alvarez. Portrait of mature hispanic man stock photo, July 2017. Last accessed 29 may 2022.

[2] Mark Anderson. Why cartoons are better than stock photos and infographics, 2013. Last accessed 16 June 2022.

[3] Jonathan Daniels. Unsplash caption: Adult yellow labrador inside black basin, October 2017. Last accessed 15 June 2022.

[4] Spencer Davis. Unsplash caption: Red astorn martin vantage running on road, August 2019. Last accessed 12 June 2022.

[5] Margaret Deuter. Élégie. In *Oxford Advanced Learner's Dictionary*, page 220. Oxford University Press, 1948.

[6] River Kao. Unsplash caption: Brown tabby cat on white wooden table, July 2018. Last accessed 11 June 2022.

[7] Todd kent. Unsplash caption: Gray wooden house photo, October 2019. Last accessed 2 June 2022.

[8] Todd kent. Unsplash caption: Group of people standing on gray concrete floor during daytime, May 2020. Last accessed 30 May 2022.

[9] Adrian Rosebrock. Adaptive thresholding with opencv, 2021. Last accessed 10 June 2022.

[10] ErnAn Solozábal. Unsplash caption : Woman in turtleneck swearter, April 2021. Last accessed 12 June 2022.

[11] Dorrell Tibbs. Unplash caption:man in white polo shirt standing near green during daytime, June 2020. Last accessed 9 June 2022.