**Mini Project Report on**



**Cartoonifying an Image using OpenCV**



**Submitted in partial fulfillment of the requirement for the award of the degree of**

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE & ENGINEERING**

**Submitted by:**

**Student Name**  **University Roll No.**

**Harshit Bajeta 2016765**

***Under the Mentorship of***

**Dr. Manoj Diwakar**

**Associate Dean (R&D) Graphic Era Deemed to be University**



**Department of Computer Science and Engineering**

**Graphic Era (Deemed to be University)**

**Dehradun, Uttarakhand**

**July-2023**

A picture containing text

Description automatically generated

**CANDIDATE’S DECLARATION**

I hereby certify that the work which is being presented in the project report entitled **“Cartoonifying an Image using OpenCV”** in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science and Engineeringof the Graphic Era (Deemed to be University), Dehradun shall be carried out by the under the mentorship of **Dr. Manoj Diwakar, Associate Dean (R&D)**, Department of Computer Science and Engineering, Graphic Era (Deemed to be University), Dehradun.

**Name** **University Roll no**

Harshit Bajeta 2016765

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Chapter** | **Description** | **Page No** |
| Chapter 1 | Introduction   * 1. Introduction   2. Problem Statement | **1-2** |
| Chapter 2 | Literature Review | **3-4** |
| Chapter 3 | Methodology | **5-8** |
| Chapter 4 | Result and Discussion  4.1 Result and Discussion  4.2 Challenges Encountered | **9-10** |
| Chapter 5 | Conclusion and Future Work  5.1 Conclusion  5.2 Future Scope | **11** |
| Chapter 6 | References | **12** |

**Chapter-1**

**Introduction**

**1.1 Introduction**

In the age of social media, standing out and capturing attention has become crucial for users on various platforms. Whether it's sharing images, blog posts, artwork, memes, or opinions, individuals strive to create an impact and connect with their followers and friends. To address this need for creative expression, we aim to provide a solution that applies cartoon-like effects to images, allowing users to share them on social media platforms, messaging apps, or keep them for personal enjoyment.

The field of image processing plays a significant role in various computer-related applications. It is utilized in home security systems, banking, education, defense, transportation, and more. The ability to accurately capture the essence of an image involves factors such as contrast, color mixing, pixel matching, and precise object placement. These advancements in image processing have contributed to the overall betterment of society.

The algorithm underlying our approach focuses on converting RGB color images into accurate and visually appealing cartoon-like representations without the need for multiple filters or blurred images lacking proper edge detection. Through an intuitive user interface, users can apply animation effects that provide an artistic touch and transform images into comic-like representations.

Creating a cartoon-like effect can be a time-consuming and resource-intensive process. Existing solutions often involve the installation of complex photo editing software like Photoshop or require users to perform multiple manual tasks. Our research indicates that a user-friendly website would be a more suitable and efficient solution, minimizing user efforts while providing a wide range of options for applying effects to images.

While there are some existing websites, such as Toony Photos, that offer similar services, they often lack user-friendliness and have limited options. This highlights the need for an improved website that offers a seamless and intuitive user experience, allowing individuals to easily apply effects to their images and achieve the desired cartoon-like transformation.

So, we aim to address these challenges by developing a user-friendly website that efficiently applies cartoon-like effects to images. By providing a wide range of options and simplifying the process, users can easily enhance their images and share them on various social media platforms, thereby standing out in the online crowd and fulfilling their creative aspirations.

Cartoons have widespread application in various industries, requiring exceptional artistic skills to create visually appealing and high-quality animations. However, the process of manually sketching cartoons for animated movies is time-consuming and demanding. Additionally, individuals often seek to convert their own images into cartoon-like representations to explore their appearance in a different style. Nevertheless, this conventional approach is time-consuming and resource-intensive, making it challenging to gather the necessary resources and execute the process effectively.

To overcome these limitations, a technique known as "cartoonifying" has emerged, which significantly reduces the time and effort required for artists in the film industry. By employing this technique, artists can achieve comparable results to traditional sketching methods but in a more efficient and expeditious manner. The implementation of this technique utilizes the Python programming language, leveraging various libraries, such as OpenCV, to obtain highly accurate outputs.

By adopting this automated approach, the process of cartoonifying images becomes more streamlined and accessible. The use of Python and specialized libraries enables artists to achieve the desired cartoon-like effect without the need for extensive manual sketching. This not only saves time but also improves efficiency in the production of cartoons.

Overall, the introduction of the cartoonifying technique in the industry represents a significant advancement in the field. It offers a faster and more efficient alternative to manual sketching, opening up possibilities for artists to explore different styles and transform images into visually appealing cartoon-like representations.

**1.2 Problem Statement**

The problem addressed in this research is the development of an efficient and accurate ML-based method for cartoonifying images. While traditional manual techniques for creating cartoons require significant artistic skills and effort, ML approaches offer a more automated and streamlined process. The aim is to overcome the limitations of existing methods by leveraging the power of ML algorithms to generate high-quality cartoon-style representations that faithfully capture the essence of the original image.

Specific challenges to address include identifying and extracting relevant features from the input image, such as edges, textures, and color gradients, and transforming them in a manner that achieves a visually pleasing and coherent cartoon-like appearance. Furthermore, the proposed method should be flexible and customizable, allowing users to adjust various parameters to achieve different cartoon styles and effects.

By addressing these challenges, the goal is to provide a robust ML-based solution for cartoonification that can be applied to a wide range of images, enabling users to effortlessly create captivating and stylized cartoon representations.

**Chapter-2**

**Literature Survey**

The cartoonification of images using machine learning (ML) techniques has gained significant attention in recent years. Several studies have explored different approaches and methodologies to achieve visually appealing and artistically stylized cartoon-like representations.

A prominent study conducted by MD.Salar Mohammad, Bollepalli Pranitha, et.al titled “Cartoonizing the Image” focuses on the application of image processing techniques to cartoonize images. Image processing plays a crucial role in identifying objects, determining dimensions, and applying effects like blurring, which are highly valued in today's media and communication landscape. The properties of image processing are explored to produce sharper and more refined images, with each picture element represented as a 2-D matrix containing pixel values. The research delves into various aspects of image analysis using these techniques.[1]

" Auto-painter: Cartoon Image Generation from Sketch by Using Conditional Generative Adversarial Networks” by Zengchang Qin, Zhenbo Luo, Hua Wang. Realistic image generation using deep neural networks has become a hot topic in machine learning and computer vision. In this paper, the sketch-to-image synthesis problem is investigated by using Conditional Generative Adversarial Networks (CGAN). The new model is not only capable of painting hand-draw sketch with proper colors , but also allowing users to indicate preferred colors .Experimental results on two sketch datasets show that the auto-painter performs better that existing image-to-image methods. Auto-painter is a supervised learning model, given a black-and-white sketch , the model can generate a painted colorful image based on given sketchimage pairs in the training data.[2]

In the research conducted by Akanksha Apte, Ashwathy Unnikrishnan, Navjeevan Bomble, and Prof. Sachin Gavhane titled "Transformation of Realistic Images and Videos into Cartoon Images and Video using GAN". This framework enables the generation of high-quality cartoonized images and videos from real-world photos and videos. The proposed method involves decomposing the input images into three cartoon representations: surface, structure, and texture. To train the network, image processing models like OpenCV, PIL, and others are employed to extract these three representations. By adjusting the weight assigned to each representation, the resulting cartoon style can be controlled and customized.[3]

The research paper titled "A Neural Algorithm of Artistic Style" by Gatys, Ecker, and Bethge introduces a system based on a Deep Neural Network for creating high-quality artistic images. The method involves using a pre-trained neural network and a 3-component loss function to achieve style transfer. By optimizing over the loss function, the system can generate images that combine the content of one image with the artistic style of another. This approach utilizes deep learning to mimic and reproduce artistic styles, resulting in visually appealing and perceptually accurate images.[4]

Anusha Pureti, Ch. Sravani, Y. Pavankumar, T. Venkateswarlu, and G. Jahnavi A. Hema proposed an efficient technique for extracting objects from animation pictures. The technique relies on assumptions regarding the colors and spatial distribution of objects in animation pictures. Typically, objects are centered in the image, while background colors tend to be more concentrated towards the edges. The process involves color quantization, seed filling, and object segmentation. Experimental results demonstrated the system's effectiveness in extracting both single and multiple objects from simple and complex backgrounds in animation pictures. The research was published in the International Research Journal of Engineering and Technology (IRJET), showcasing promising efficiency in object extraction from animation images.[5]

Early used methods are optimization-based methods and filtering-based [6]. Xu and Fan [7, 8] proposed end-to-end networks for image smoothing. Bi et al. proposed an L1 transformation for the image flattening problem. Min et al. introduced image smoothing by reducing a quadratic energy function.

Image Smoothing [9, 10, 11, 12] is used to produce an image with less pixel and reduce the noise of an image. Smoothing is also usually based on a single value representing the image, such as the average value of the image or the middle value.

In their research on animated cartoon faces, Ruttkay and Noot explore the significance of human faces in everyday communication and the challenges of modeling and animating realistic 3D faces in human-computer interaction. To overcome these limitations, they propose 2D cartoon faces as viable alternatives with the added advantage of "beyond realism" features. They introduce "Cartoon," an interactive system designed to create and animate 2D cartoon faces. The researchers provide visual illustrations and accompanying movies on CD to showcase the expressive and artistic effects achievable with the system. Implemented in Java, Cartoon enables real-time animation on PCs and web platforms, making it accessible to a wide range of users who have successfully utilized the system.[13]

**Chapter-3**

**Methodology**

**3.1 Importing the required modules**

**Python:**We use python as a programming language for building the application.

**cv2:** We use cv2 for image processing.

**Numpy:** Mainly NumPy is used for dealing with arrays. Here the images that we use are stored in the form of arrays. So, for that, we use NumPy.

**easygui:** easygui is a module used for GUI programming in python. In our application easygui is used to open the file box to upload images from the local system.

**Imageio:** Imageio is a python library that reads and writes the images.

**Matplotlib:** Matplotlib is used for visualization purposes. Here we plot the images using matplotlib.

**OS:**Here in our application os is used for dealing with paths like reading images from the path and saving the image to the path.

**Tkinter:** Tkinter is a standard Graphical User Interface (GUI) package.

**SYS**: This module provides access to some variables used or maintained by the interpreter and to functions that interact strongly with the interpreter. e.g., exit ().

A screenshot of a computer program

Description automatically generated

**Fig-3.1**

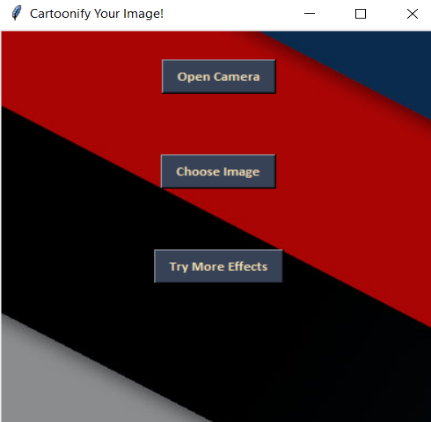
**3.2 Building a File Box to choose a particular file**

Create the main window for our application, containing buttons, labels, and images. Set a title for the window using the title() function.

**3.3 Loading & Storing Image**

We will transform our image into a NumPy array since computers interpret all data as numerical values.

A close up of a dog

Description automatically generated

**Fig-3.2 Fig-3.3**

**3.4 Transforming an image to grayscale**

In the cv2 library, the cvtColor(image, flag) method is employed to transform an image into the specified color space indicated by the 'flag' parameter. In this case, we aim to convert the image into grayscale, which can be achieved by using the BGR2GRAY flag. By applying this flag, the method returns the image representation in grayscale, with color information removed.

**3.5 Smoothening a grayscale image**

To achieve a smoothing effect on an image, we can apply a blur operation. In this case, the medianBlur() function is used. This function calculates the mean value of all the pixels within a kernel centered on each pixel and assigns this mean value to the center pixel. As a result, a blur effect is created, reducing noise and enhancing image smoothness.

**A close up of a dog

Description automatically generated**Close-up of a dog's face

Description automatically generated

**Fig-3.4 Fig-3.4**

**3.6 Retrieving the edges of an image**

To achieve the first specialty of the cartoon effect, which is highlighted edges, an adaptive thresholding technique is employed. This technique calculates a threshold value based on the mean of the neighboring pixel values minus a constant, C. By applying this threshold, the edges are emphasized. The specific type of threshold used is Thresh\_binary, and the block size for analysis is determined by additional parameters. This step helps retrieve and accentuate the edges, contributing to the overall cartoon effect.

**3.7 Preparing a Mask Image**

The second specialty of the cartoon effect is implemented by preparing a lightened color image and applying a mask with edges. To achieve this, the bilateralFilter function is utilized for noise removal and image smoothing. Parameters such as diameter, sigmaColor, and sigmaSpace are used to create a watercolor-like effect, resulting in smoother colors and a visually appealing cartoon image. This process resembles the beautify or AI effects found in modern mobile phone cameras, enhancing the overall artistic appearance of the image.

.

A drawing of a lion

Description automatically generatedA close up of a dog

Description automatically generated

**Fig-3.6 Fig-3.7**

**3.8 Perform color quantization technique**

For color quantization, the K-Means clustering algorithm available in the OpenCV library is utilized. By adjusting the value of k, the number of colors desired for the image can be determined. The K-Means algorithm assigns each pixel to one of the k clusters based on their color values. This process effectively reduces the number of colors in the image, resulting in a quantized color palette. By manipulating the value of k, different levels of color reduction can be achieved, allowing for customization of the final image's color scheme.

**3.9 Giving a Cartoon Effect**

To combine the two specialties of the cartoon effect, masking is employed. This is achieved by performing a bitwise "AND" operation on two images, utilizing their numerical representations. The edged image is masked onto the "BEAUTIFY" image, resulting in the final cartoonified image. By applying this masking technique, the highlighted edges are blended with the smoothed and quantized color image, producing the desired cartoon effect.

A close up of a dog

Description automatically generatedA close up of a dog

Description automatically generated

**Fig-3.8 Fig-3.9**

**3.10 Plotting all the transitions together**

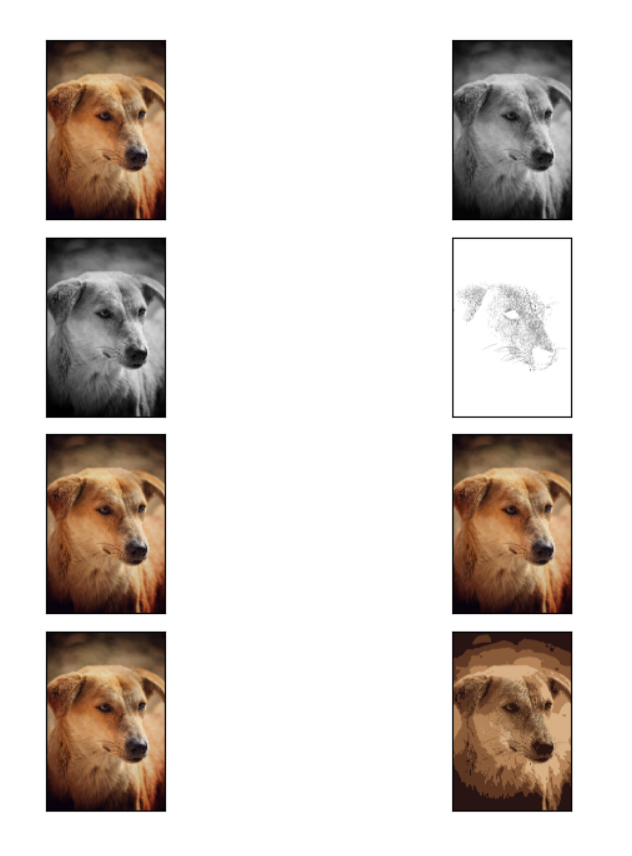
To display multiple images, we begin by creating a list called "images" that contains all the resized images. Next, we create axes using the subplot function to set up a grid for displaying the images. Each image is then displayed on its corresponding axis using the imshow() method.

Finally, to show the entire plot with all the images, we use the plt.show() function. This function displays the complete plot once we have plotted all the images on their respective subplots.

**Chapter-4**

**Result and Discussion**

**4.1 Result and Discussion**

In this OpenCV project, we’ve developed an image cartoonizer application using python. From this project, we’ve learned about thresholding and edge finding from an image, K-means clustering algorithm, color quantization technique, and some other basic image processing techniques.

**Fig-4.1**

Firstly, altering the filter type can significantly affect the final output. Different filter types, such as Gaussian, bilateral, or median filters, offer distinct smoothing properties and noise reduction capabilities. By experimenting with different filter types, we can achieve varying levels of image smoothness and control over noise reduction. This flexibility allows for customization based on the desired artistic style and visual effect.

Additionally, adjusting the filter size can have a significant impact on the cartoonization outcome. Larger filter sizes tend to create more pronounced smoothing effects, resulting in a more simplified and cartoon-like appearance. On the other hand, smaller filter sizes can retain more fine details and textures in the image, offering a different artistic interpretation. By varying the filter size, we can strike a balance between preserving important image details and achieving the desired cartoon effect.

Furthermore, changing the technique employed for edge detection and highlighting can significantly influence the final cartoonized image. Different edge detection algorithms, such as Canny, Sobel, or Laplacian, offer varying levels of edge emphasis and precision. By selecting the most appropriate edge detection technique, we can control the appearance and clarity of edges in the cartoonized image. This decision is crucial as it directly impacts the overall visual style and artistic appeal.

**4.2 Challenges Encountered**

While developing an image cartoonizer application, there are several challenges that may be encountered:

4.2.1. Edge Detection: Accurately identifying and extracting edges from an image can be challenging, especially in complex or cluttered scenes. Different edge detection techniques may yield varying results, and finding the most suitable method for the desired cartoon effect can be a challenge.

4.2.2. Color Quantization: Achieving optimal color quantization requires determining the appropriate number of colors and effectively assigning them to the image. Striking a balance between preserving important details and simplifying the color palette can be a challenging task.

4.2.3. Smoothing and Noise Reduction: Balancing the degree of smoothing and noise reduction is crucial to achieve the desired cartoon effect. Finding the right filter type, size, and parameters can be challenging, as excessive smoothing can lead to loss of important details, while inadequate smoothing may result in a noisy appearance.

4.4.4. Preserving Image Content: Retaining the essential content of the image while applying the cartoon effect is essential. Ensuring that important features, such as facial expressions or recognizable objects, are not lost or distorted during the transformation requires careful consideration and adjustment.

5. User-Defined Parameters: Determining optimal parameter values for user-defined settings, such as edge thickness, color saturation, or cartoon effect intensity, can be challenging. Providing a user-friendly interface and intuitive control over these parameters is crucial for an effective and enjoyable user experience.

**Chapter-5**

**Conclusion and Future Work**

**5.1 Conclusion**

This project has provided insights into the origin and history of image processing, discussed uncertain environments, and explored existing methods for cartoon imaging. It introduced an efficient method for extracting cartoon effects, which holds promise in areas such as signature recognition, digital video processing, remote sensing, and finance.

The proposed method successfully extracts meaningful objects from diverse characters and backgrounds. This application has the potential to empower individuals with visual impairments, allowing them to live more independently and interact with visual content effectively.

Cartoonizing an image offers the advantage of transforming it into a visually appealing representation, resembling watercolor paintings and eliminating color roughness.

The significant contribution of this project lies in addressing the challenges posed by different image types containing one, two, or three objects. While existing methods struggle with such cases, the proposed method demonstrates its effectiveness in solving these scenarios.

Research advances the field of image processing by presenting an efficient method for cartoon image extraction. The results demonstrate its potential in enhancing visual content and overcoming challenges in various image scenarios. This work opens doors for further developments in cartoon imaging and its applications across different domains.

**5.2 Future Scope**

It would be interesting to explore the integration of generative adversarial networks (GANs) into the cartoonification process. GANs have shown promise in generating realistic and visually appealing images, and incorporating them into the proposed method could enhance the quality and artistic style of the generated cartoon images. Additionally, further research can focus on improving the efficiency and scalability of the cartoonification process, allowing for real-time application and handling of larger datasets. Exploring the potential applications of cartoonification in emerging technologies such as augmented reality (AR) and virtual reality (VR) would also be a valuable avenue for future investigation

**Chapter-6**

**References**

[1]. MD.Salar Mohammad, Bollepalli Pranitha,Shivani Goud Pandula, Pulakanti Teja Sree.(2021). “Cartoonizing the Image” an International Journal of Advanced Trends in Computer Science and Engineering, Volume 10, No.4, July – August 2021

[2]. Zengchang Qin, Zhenbo Luo, Hua Wang, " Auto-painter: Cartoon Image Generation from Sketch by Using Conditional Generative Adversarial Networks”, International Conference on Image Processing, 2017.

[3]. Akanksha Apte, Ashwathy Unnikrishnan, Navjeevan Bomble, Prof. Sachin Gavhane, “Transformation of Realistic Images and Videos into Cartoon Images and Video using GAN” an International Journal of Research in Engineering, Science and Management Volume 4, Issue 7, July2021.

[4]. A Neural Algorithm of Artistic Style (Leon A. Gatys, Alexander S. Ecker, Matthias Bethge) , 2015.

[5]. Zengchang Qin, Zhenbo Luo. Hua Wang. "scikit-image processing in python", International Conference on Image Processing, 2014

[6]. Yang Chen, Yu-Kun Lai, and Yong-Jin Liu. Cartoongan: Generative adversarial networks for photo cartoonization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 9465–9474, 2018.

[7]. Lvmin Zhang, Chengze Li, Tien-Tsin Wong, Yi Ji, and Chunping Liu. Two-stage sketch colorization. In SIGGRAPH Asia 2018 Technical Papers, page 261

[8]. Kaiming He, Jian Sun, and Xiaoou Tang. Guided image filtering. In European Conference on Computer Vision, pages 1–14. Springer, 2010.

[9]. J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum. Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. In Advances in Neural Information Processing Systems, pages 82–90, 2016.

[10]. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In Advances in Neural Information Processing Systems 27, pages 2672–2680. 2014

[11]. Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2536–2544, 2016.

[12]. ArtsiomSanakoyeu, Dmytro Kotovenko, Sabine Lang, and Bjorn Ommer. A style-aware content loss for real-time HD style transfer. In Proceedings of the European Conference on Computer Vision (ECCV), pages 698–714, 2018

[13]. Ruttkay, Zsófia&Noot, Ha: Animated Cartoon faces an International Symposium on Non-Photorealistic Animation and Rendering.