Advanced Edge Preserving Smoothing and Cartooning with Artistic Filtering ”AESCA”

Amit H Patel
The University of Adelaide
Adelaide, SA 5000
a1697700@student.adelaide.edu.au

Abstract

After analysing OpenCv functions and other methods like:MEDIAN[1], BILATERAL[2] and Kuwahara filter[3] for edge preserving smoothing , it was observed that when one function is used for different kind of images(specifically, object focused image or wide scale nature image), it provides same standard function that might not be appropreate.Thus, proposed algorithm ”AESCA” tries to come out from those drawback of specific functions and provides more flexibility to change parameters for detail works in the algorithm steps. Not only this, Algorithm includes the two major outcomes from it: 1) Artistic Image(painting) and 2) Caroonified Image. To achieve needed outcomes it follows 4 steps: 1) Advanced Edge Preserving smoothing 2) Edge detection 3) Imfusing 4) Quantization.

1. Introduction

Numerous functions are availed to smoothing the image. Usually, achieving a single functionality like converting image in cartoon or artistic fork, is widely opt for. Although, the proposed algorithm in this paper(AESCA) is combining various functions that can provide a user more flexibility to change the parameters according to different images. It provides two outputs: Advanced artistic image and cartooning image too.The algorithm's 1st and last step is highly weighted due to hand made functionality that contributes more in this paper. The algorithm consist main four processes:

1. Advanced Edge Preserving Smoothing.

2. Edge Detection.

3. Imfusing(Outcome: Artistic Image).

4. Quantization(Outcome: Cartoonified image).

2. Overview of the algorithm ”AESCA”

The algorithm is followed in the sequence of specified processes to achieve needed results.

The most important part in the algorithm is **Advanced Edge Preserving Smoothing** due to results are heavily dependent on that. Normally in smoothing process, edges also get blurred, which unable identification of the captured object. Therefore, edge preserving smoothing was proposed.There are many inbuld filters to do so. For instance, Median Filters[1],Bilateral Filters[2] and Kuwahara Filter[3].Comparing these filters, Kuwahara provides more efficient artistic filter for preserving edges. It acts almost like taking out average of the matrix and forming a blurred image, where shadowed areas are usually distracts the image. Whereas, advanced edge preserving smoothing operates differently instead of taking first standard deviation($\mathrm{S_i}$) of the matrix. It puts the use of weighting windows, that can be accumulated through Guassian mask. Depending on weighted sectors, standard deviations($\mathrm{S_{in}}$) are computed in a loop. Moreover, it provides more sharpen edges by considering circular areas. This process only provides artistic image. Although, there is lot more ahead in the algorithm depending on outcome of this artistic image.

The second considerable contribution is the abstraction of the edges from the artistic image in a binary matrix(for Black and White image). After advanced smoothing, artistic image would have obtained the smooth texture with essential edges. Those edges will give better image ahead in the algorithm to get artistic image combining sharp black edges and cartoonified image. In the process of **Edge Detection**, first of all image gets converted in gray colored image. After graying the image, the pixel's densities are fetched based on which, grayed pixels are replaced to binary pixels with the help of threshold, what lefts us with black sharp edges like borders with white background. In result of the process, these edges/ borders will be used for enhancing the artistic filtering to the sophisticated standards.

The third part is **imfusing**. This process is named after matlab function imfuse, in which two images can be combines with a blend and joint attribute. Imfusing is used to generate sophasticated artistic image. With the experiment, artistic images normally looses edges and borders. Thus, placing an extracted edges on the artistic image can separately emboss the features of the object with a smooth texture. Although, joining two images through imfuse function causes pixels to become lighter. That's why, imfusing step needs to involve function that upgrade saturation of the image. this is achieved by getting standard deviation($\mathrm{S_{in}}$) and forming a new image matrix with a hand made formula. By the process, algorithm breakthrough the first goal of creating artistic image with embossed black edges(borders).

The last and turning point of the algorithm is the **Quantization**. This process helps to achieve cartooning the image with the help of former step and edge detection. The basic idea of quantization is limiting number of colors used for pixels with the quantified limitation of the colors. The colors are chosen based on majority pixels. Each pixel colors are replaced with the nearest similar pixel value that are included in chosen limited numbers before.This functionality is achieved by K-means algorithm. This process is based on manipulation of the RGB values of each pixels. After getting needed outcome, algorithm adds extracted edges from the second step, that again results in dullness in saturation of the image. Thus, it again follows last step of imfusing, where up-gradation in saturation is achieved through mean($\mathrm{S_i}$) and standard deviation($\mathrm{S_i}$).
In the result of the above step, a real caroonified image can be formed from any image that can be more flexible with the needs of the user.

3. Related Work and Tested Different Approaches

For edge preserving smoothing, I have tried the below formula instead of using direct implemented formulas to understand more detail works. Thus, it turned out, it was lacking many information of the object and slow too.
The tried formula is:

$$H(\bar{x}, \sigma_d, \sigma_r) = \frac{\int e^{-\frac{1}{2}(\frac{||\overline{x}-x||}{\sigma_d})^2} w(x,\overline{x}) f(x) dx}{\int e^{-\frac{1}{2}(\frac{||\overline{x}-x||}{\sigma_d})^2} w(x,\overline{x}) dx}$$

$$w(x,\overline{x},\sigma_r) = (1-m(\overline{x})) \times w'(x,\overline{x},\sigma_r) + m(\overline{x}) \times u(\overline{x})$$

$$w'(x,\overline{x},\sigma_r) = e^{-\frac{1}{2}(\frac{||f(\overline{x})-f(x)||}{\sigma_r})^2}$$

OpenCv functions are really useful for such work more quicker and faster. Thus, I used MEDIAN[1] and BILATERAL[2] functions. Beside getting edge preserving smoothing, I was opting for paint like smoothing. For which, I tried Kuwahara filter[3]. In Kuwahara filter, it uses Gaussian weighted local average instead of local average. In last, the used technique for edge preserving edging is somewhat modification of Kuwahara technique. The details are explained in the following section.

4. The Approach

The main approach of the algorithm is increasing edge details with smoothing the texture of the image. While, combining Imfusing and Quantization technique aim for advanced artistic filtering and cartooning image respectively. For edge preserving smoothing, bilateral filter but it was unnecessarily creating the shadows in the needed output image. So, the algorithm includes masking method according to Sigma value.

$$g_\sigma(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

After separating the group of pixels/channels of the image, the described function runs in loop. Each channels are converted to frequency transfor.

For weighting channels, Gaussian mask are divided into circular regions. The cutting functions are used for making sectors. Thus, weighting is a multiplication of cutting function and Guassian mask.

$$w_i = g_\sigma v_i$$

With the help of weight, local mean(m) and standard deviations ($\mathrm{S_i}$) are calculated for each sectors.The parameter **quan** is used to control sharpness of the sectors. Due to circular sectors of the image, edges does not get blurred as it exaggerates the edge in that area and surrounding area gets blurred. Here, normalisation operation depends on the following formula:

$$\theta_q(x,y) = \frac{\sum_i m_i(x,y) s_i(x,y)^{-q}}{\sum_i s_i(x,y)}$$

In color images, operators take the three set's local averages and local standard deviation. this is repeated for each color pixels/ channals. To sum up first step, if image obtains sharp edged in some part, then those edges are considered as high weighted sectors with the help of local standard deviation value(s). By this, edges are preserved more efficiently than using other implemented functions, while smoothing the image. This provides effect like painting.

In the process of edge detection, some modifications are done on versions of DoG.When image is blurred from smoothing process, still it lacks the detail features of the object. Edge detection help to specify the boundaries and embosses the features of the object by placing black border and edges on the artistic image. The value of 255, is considered the highest value as the strongest part of the edge. Instead of having 255 and 0 values for making image with edges, binary image is used with the specified threshold, in which artistic image is used as a normalized image(Outcome from the former step). It provides boundaries that enhances the details of the object with having more user control through changing threshold parameter in edge detection.

For imfusing process, it directly uses imfuse function of the matlab, where both the results from the former step and edge detection, are merged. Although, blending reduces saturation of the combined image. For this consequence, mean and standard deviation are used to put high in and out values as parameters in imadjust function of matlab. The values are defined by m_i- n*s; and m_i +n*s;. Here, n can be adjusted to get appropriate saturation of the output image.

Whereas, Quantization is proceesed with the help of K-means algorithm. Process includes extracting RGB color values from each pixels and use it in K-means function of the matlab with defined number of colors. After forming new matrix with K-means, it used output of edge detection and merged with the same technique of imfusing.

The algorithm mainly aims for more user controlled artistic image with edges overlapped on it. Not only sophisticated artistic image, but also with the help of quantization technique, it provides cartoonified image, which can be more adjustable by the user.

5. Results

Observing algorithm's working on different picture, I classified any kind of images in two types of images due to it gives a huge difference by using same parameters of the algorithm.

5.1. Results on Nature Images

Running algorithm on natural images, I found better adjustments in the different parameters of different process of the algorithm. the results are shown below each process and its defined parameters.

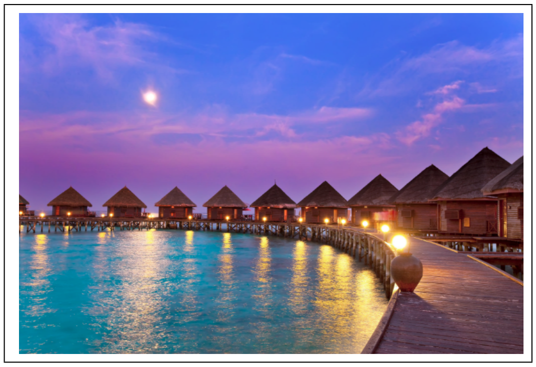

Figure 1. The image is the input image.

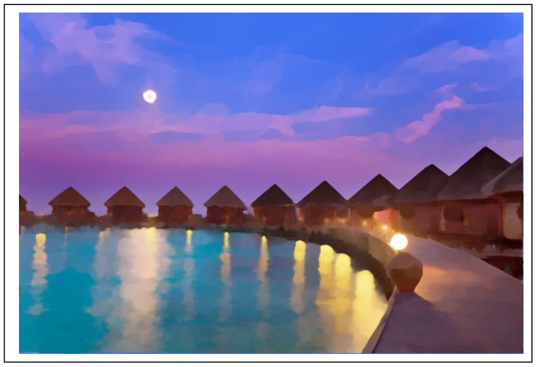

Figure 2. Result from Advanced Edge preserving smoothing with the followed parameters:

sigma = 6
N=4
quan=8

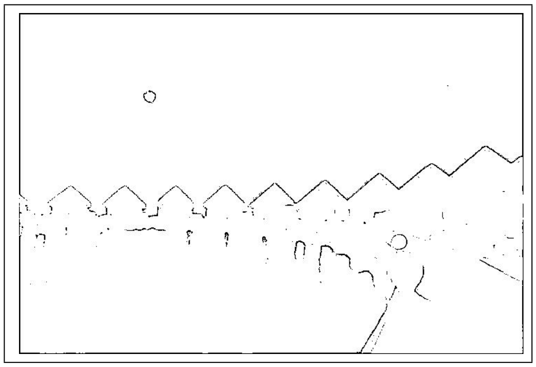

Figure 3. Result from Edge Detection with followed parameters:

Threshold = 0.05

Dark index for next step = 2

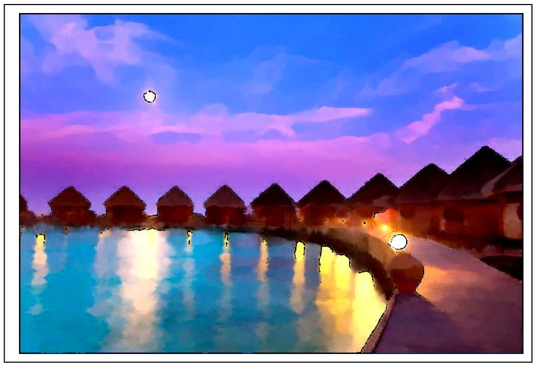

Figure 4. Result from Imfusing process, where edges are overlapped on artistic filtering to enhance features of the objects in the image.

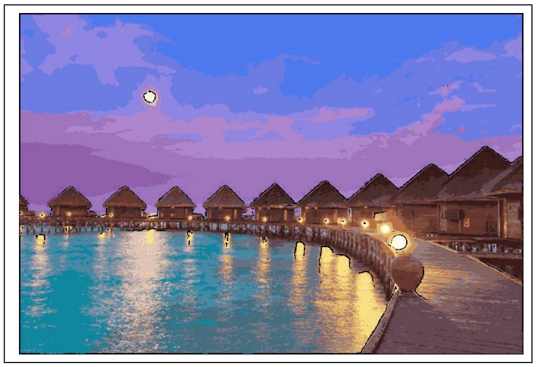

Figure 5. Result from Quantization to make cartoonified effect of the image with specified parameters:NumberofColors = 25

5.2. Results on specific Objects/living objects

While considering the images with the human objects, that needs to be more carefully transferred due to have minor details than nature images, where objects usually have specific shapes.

There are many factors/parameters should be changed like: Image sharpness should be lesser tha the nature images because, natural images probably have thousands of essential edges, so we try to obtain as many possible. Thus, quan parameter should be around 7-10. While, for faces and other specific objects it is better to have sharpness around 4-6, this can give most of the important edges.

Another one is number of limiting colors for Quantization. Here, it has almost same theory to sharpness. That's why, we try to make cortoonified image with more colors which can give more functionalists of the objects in the image.Therefore, parameter(numofcolors) should be set between 25-40. Whereas, due to having less colors in specific object photo, itis suggested to have 7-12 limited colors.

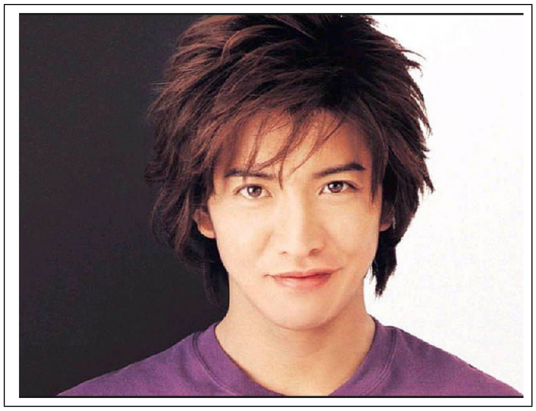

Figure 6. The image is the input image.

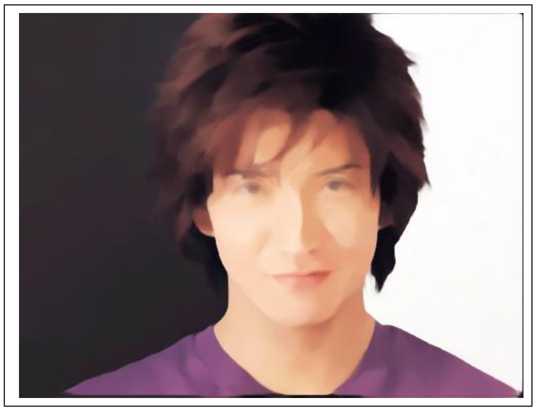

Figure 7. Result from Advanced Edge preserving smoothing with the followed parameters:

sigma = 6
N=5
quan=4

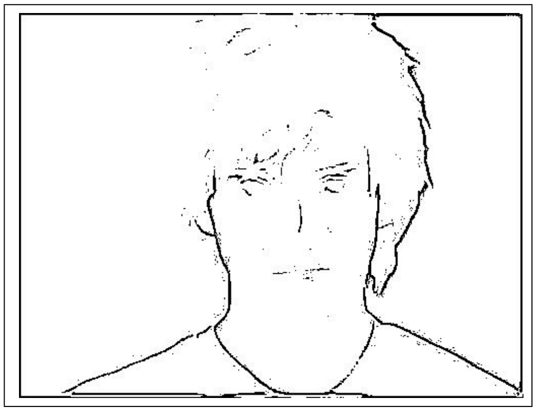

Figure 8. Result from Edge Detection with followed parameters:

Threshold = 0.02

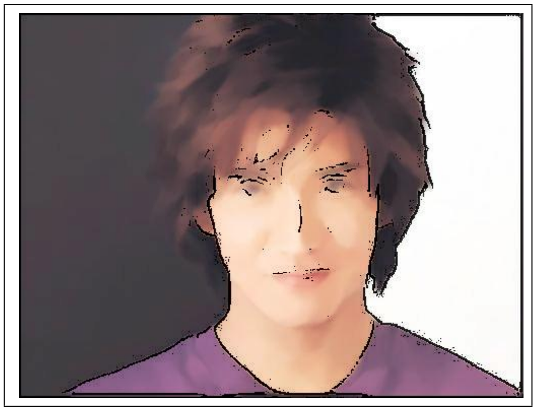

Figure 9. Result from Imfusing process, where edges are overlapped on artistic filtering to enhance features of the objects in the image.

Dark Index = 1.4

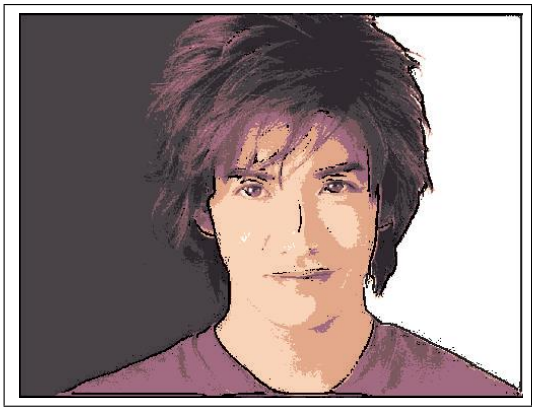

Figure 10. Result from Quantization to make cartoonified effect of the image with specified parameters:

NoOfcolors=7

As described Quan and noOfColor parameters are needed to be different based on different objects. The same way, DarkIndex also need to be set according to image size. If height is more than width, then Darkindex is suggested to be more less than 1.5. while index should be above 1.9 for longer width of the image.

In last edge threshold also matters depending different type of images. Here, If input image is nature, where we might need less edges to be covered due to many sharp edges exists in the image, then edge threshold should be above 0.5 that gives less detailed edges in binary image. While, for specific object oriented images might have some details like hair in given example, that needs more detail to get the edges in binary image. That's why, it is better to have threshold lesser than 0.5.

6. Conclusion

To conclude, the whole process of ”AESCA” algorithm, it is developed after many iteration of learning different methods, where it tries to overcome negative points of other function. It is concluded that there are plenty of ready made functions, although when we put different images like: object specified image or wide scenario images, functions give the same results for each type of images. Thus, following the whole procedure of ”AESCA” algorithm with defining parameters depending on the image object, it can give more sophisticated artistic image and cartoonified images. It also provides wide flexibility from inside of programming to get needed outcome.

7. references

[1] W. K. Pratt, Digital Image Processing. New York: Wiley, 1978

[2] C. Tomasi and R. Manduchi, Bilateral filtering for gray and color images, in Proc. Int. Conf. Comput. Vision, 1998, pp. 839-846

[3] M. Kuwahara, K. Hachimura, S. Ehiu, and M. Kinoshita, Processing of ri-angiocardiographic images, in Digital Processing of Biomedical Images. New York: Plenum, 1976, pp. 187-203.

[4]Kuwahara Filter - Wayne Rasband, 2005 https://imagej.nih.gov/ij/plugins/kuwahara.html

[5] M. Nagao and T. Matsuyama, Edge preserving smoothing, Comput. Graph. Image Process., vol. 9, pp. 394-407, 1979.

[6] R. van den Boomgaard, Decomposition of the Kuwahara-Nagao operator in terms of linear smoothing and morphological sharpening, in Proc. ISMM, pp. 283-292, in 2002

[7] Hacettepe University Department Of Computer Engineering- BBM 413: Fundamentals of Image Processing Lecture Slides http://web.cs.hacettepe.edu.tr/erkut/bbm413.f16/

[8] M.A. Schulze and J.A. Pearce. ”A morphologybased filter structure for edge-enhancing smoothing,” 1994 IEEE International Conference on Image Processing (ICIP-94), v. 1, pp. 530-534, Austin, Texas, 1994.

[9] H. J. A. M. Heimans, Connected morphological operators for binary images, Comp. Vis. Image Understand., vol. 73, pp. 99-120, 1999.

[10] A. Meijster and M. H. F. Wilkinson, A comparison of algorithms for connected set openings and closings, IEEE Trans. Pattern Anal. Mach. Intell., vol. 24, no. 4, pp. 484-494, Apr. 2002

General Reference: Giuseppe Papari, Nicolai Petkov, and Patrizio Campisi. Artistic edge and corner preserving smoothing. IEEE Transactions on Image Processing, 29, 2007.
https://au.mathworks.com/help/comm/ug/quantization.html