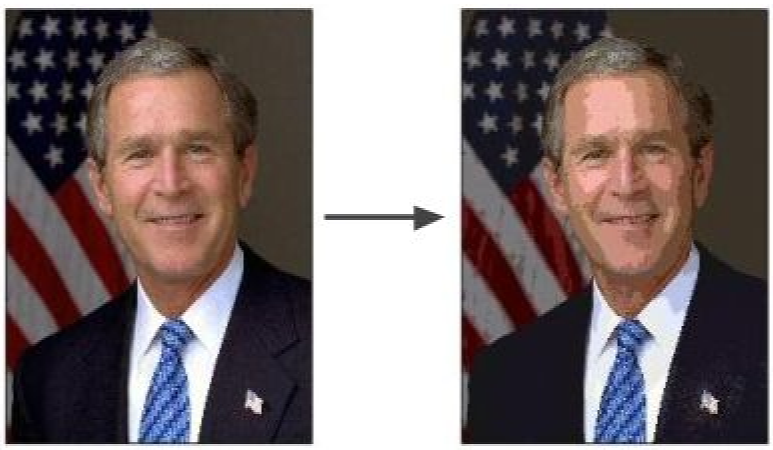
Image Toonification

Course Project

Introduction to Image Processing

2017



Resu Nikhil Reddy & Surya Prakash

14UEC080 & 14UEC109
LNMIIT Jaipur

INTRODUCTION

The process of converting a face image into a cartoon is called as toonification. We aim to convert an image into a cartoonish rendition of itself using MATLAB.

METHODOLOGIES USED

We have followed two different methodologies for toonification:

Approach 1 - Applying anisotropic diffusion to make the image look piecewise constant followed by edge detection. In this case, we could only work with grayscale images.

Approach 2 - Applying Bilateral Filtering. We could obtain colored toonified images here.

Approach 1

We used a two step process to get a toonified image: Anisotropic Diffusion followed by Edge detection. **Anisotropic diffusion** (by Perona–Malik**)** is a technique for image noise reduction without removing significant parts of the image content. Anisotropic diffusion can be used to remove noise from digital images without blurring edges.

We evolved the image towards a piecewise constant image with the boundaries between the constant components being detected as edges by running the diffusion with an edge seeking diffusion coefficient for a certain number of iterations.

Approach 2

We have obtained a cartoonish rendition of an image by performing Bilateral Filtering on it. A **bilateral filter** is a non-linear smoothing filter for images that replaces the intensity of each pixel with a weighted average of intensity values from nearby pixels, based on the Gaussian distribution. This filter preserves sharp edges, because the weights are not only determined by the spatial location in the neighborhood but also the intensity difference from the center value of the neighborhood.

Alternately, other edge preserving methods such as anisotropic diffusion could also be used.

IMPLEMENTATION

Approach 1

Anisotropic diffusion has been performed using the anis_diff.m function to perform anisotropic diffusion of an image following Perona and Malik's algorithm. This process smoothes regions while preserving, and enhancing, the contrast at sharp intensity gradients.

The *anis_diff()* function has the following components -

- kappa controls conduction as a function of gradient. If kappa is low small intensity gradients are able to block conduction and hence diffusion across step edges. A large value reduces the influence of intensity gradients on conduction.
- lambda controls speed of diffusion
- Diffusion equation 1 favours high contrast edges over low contrast ones.
- Diffusion equation 2 favours wide regions over smaller ones.

Reference: P. Perona and J. Malik. Scale-space and edge detection using ansotropic diffusion. IEEE Transactions on Pattern Analysis and Machine Intelligence,12(7):629-639, July 1990.

Based on original anisodiff.m function by Peter Kovesi, School of Computer Science & Software Engineering, The University of Western Australia.

This step was followed by edge detection, which was performed by using intuitionistic fuzzy set theory, based on original work of Krishna Prasad, IIT Delhi,New Delhi. Reference : Chaira T, Ray AK. Threshold selection using fuzzy set theory. Pattern Recognition Letters. 2004 Jun 30;25(8):865-74.

Approach 2

The project shows an application of bilateral filtering to image abstraction, employing a typical usage for the bilateral filter. Our MATLAB code uses the following functions to apply the bilateral filtering on the coloured image, while preserving its colour:

1. *cartoon()* - Cartoon Image abstraction using bilateral filtering. This function uses the bilateral filter to abstract an image following the method outlined in:
    - Holger Winnemoller, Sven C. Olsen, and Bruce Gooch. Real-Time Video Abstraction. In Proceedings of ACM SIGGRAPH, 2006.

    C = cartoon(A) modifies the color image A to have a cartoon-like appearance. A must be a double precision matrix of size NxMx3 with normalized values in the closed interval [0,1]. Default filtering parameters are defined in cartoon.m.

    [Douglas R. Lanman, Brown University, September 2006]

2. *colorspace()* - Convert a color image between color representations.

    B = colorspace(S,A) converts the color representation of image A where S is a string specifying the conversion. S tells the source and destination color spaces.

[Pascal Getreuer 2005-2006]

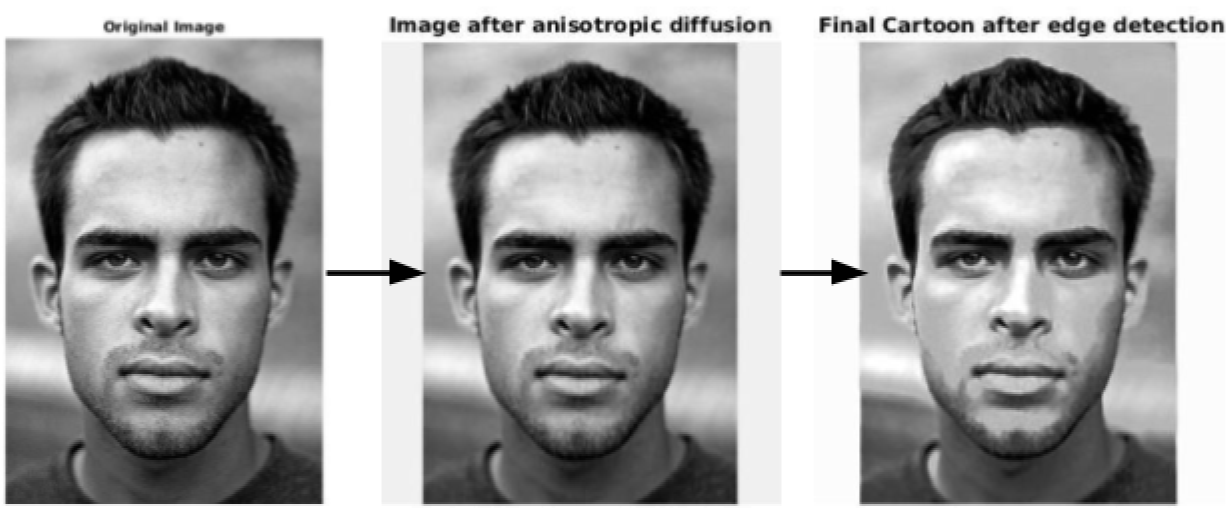
3. *bfilter2()* - Two dimensional bilateral filtering. This function implements 2-D bilateral filtering using the method outlined in:
    - C. Tomasi and R. Manduchi. Bilateral Filtering for Gray and Color Images. In Proceedings of the IEEE International Conference on Computer Vision, 1998.

   B = bfilter2(A,W,SIGMA) performs 2-D bilateral filtering for the grayscale or color image A. A should be a double precision matrix of size NxMx1 or NxMx3 (i.e., grayscale or color images, respectively) with normalized values in the closed interval [0,1]. The half-size of the Gaussian bilateral filter window is defined by W. The standard deviations of the bilateral filter are given by *SIGMA*, where the spatial-domain standard deviation is given by *SIGMA(1)* and the intensity-domain standard deviation is given by *SIGMA(2)*.

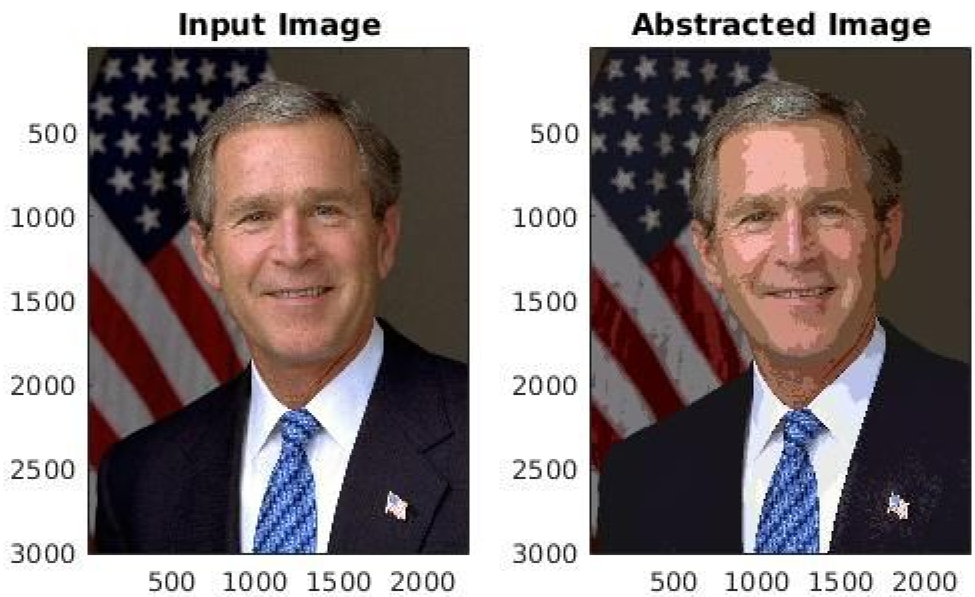
   [Douglas R. Lanman, Brown University, September 2006, dlanman@brown.edu, http://mesh.brown.edu/dlanman]

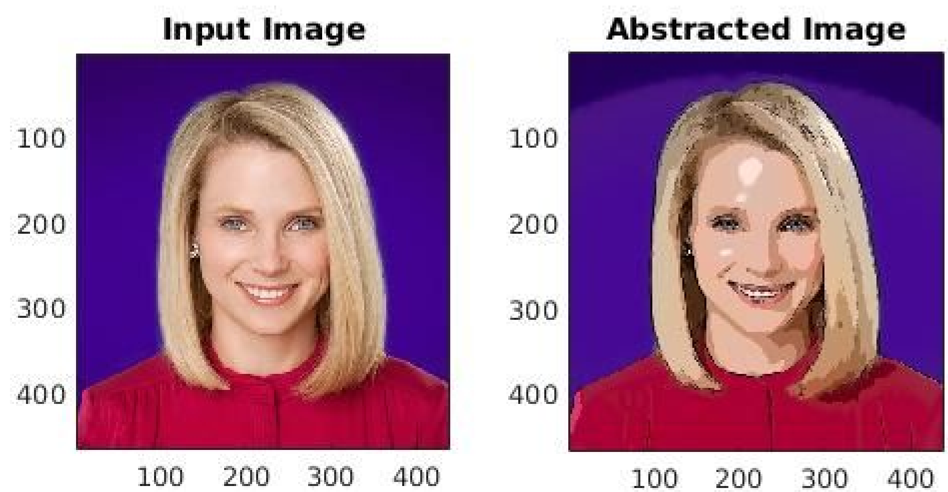
RESULTS

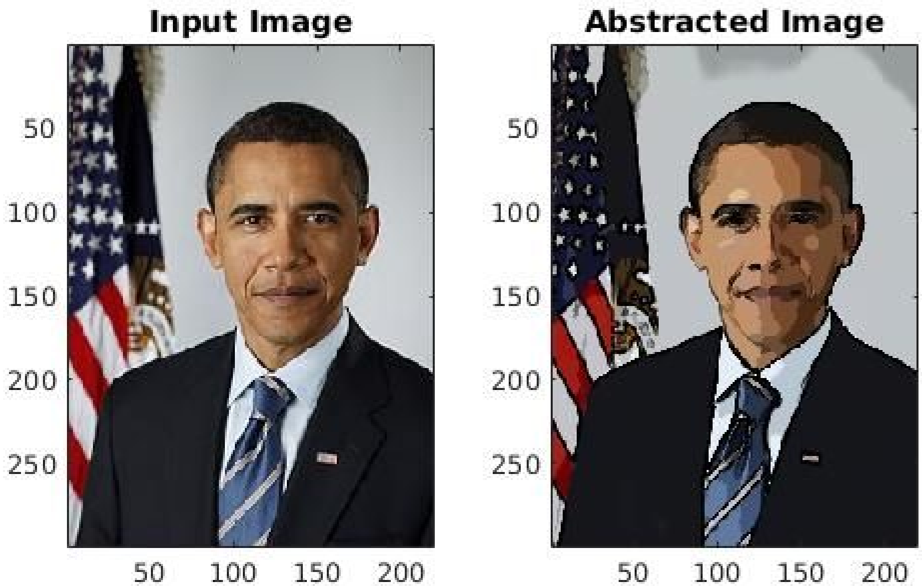
Approach 1

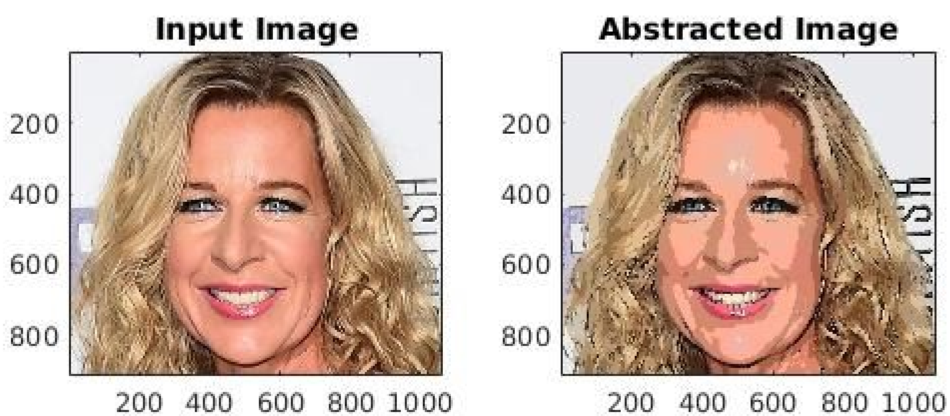


Approach 2


Input Image
Abstracted Image
500
1000
1500
2000
2500
3000
500 1000 1500 2000


Input Image
Abstracted Image
100
200
300
400
100 200 300 400


Input Image
Abstracted Image
50
100
150
200
250
50 100 150 200


Input Image
Abstracted Image
200
400
600
800
200 400 600 800 1000