Cartoon Transformation

INSERT PIC

Jake Garrison

EE 440 Final Project - 12/5/2015

Table of Contents

[Features 2](#_Toc437377055)

[Functions 3](#_Toc437377056)

[*Global Controls* 3](#_Toc437377057)

[*Effect Parameters* 4](#_Toc437377058)

[Implementation 5](#_Toc437377059)

[*Color Mode* 5](#_Toc437377060)

[*Binary Mode* 5](#_Toc437377061)

[User Guide 6](#_Toc437377062)

[Comments 6](#_Toc437377063)

[Future Research 6](#_Toc437377064)

# Features

The core of the program relies on a gradient minimization algorithm based the *L*0 gradient minimization concept. This filter generally flattens the color content and removes high frequency noise while still retaining the edges and primary color change. In my experience, the gradient minimization algorithm I implemented preforms better than ‘cartoon filters’ and ‘bilateral filters’ found online which simply combing a blurring filter or bilateral filter with and edge detection. After the gradient minimization, the edges of the original image are extracted and altered using a customizable morphological transformation that serves to thicken the edges and display them in a more ‘hand drawn stroke’ form. After applying the modified edges to the gradient minimization output, the final output is formed after a simple color quantization is applied, reducing the color depth to better represent a cartoon. These stroke and color settings, as well as the gradient minimization parameters are customizable through the GUI.

An additional mode inspired by old newspaper cartoons and the pop art style is also available in the GUI. This mode performs binary operations and color masking to output a simple, binary representation of the input that emphasizes the features present in the input image. As in the cartoon mode, this mode has morphological operations to better simulate human strokes, but the true elegance of this effect comes from its ability to accurately emphasize borders rather than all edges. The border is extracted with a function that scores all edges based off continuity and density to identify possible borders, then gaps in edges are filled in and the resulting fully closed edges are added to the output effectively emphasizing the borders.

The GUI provides a simple, but powerful interface for working with the effects outlined above. It is resizable (responsive design) and allows the user to load an image of choice and save the output to disk. There are also some radio buttons allowing a user to apply addition transformations like inverting or toggle controls such as the morphological transform.

# Functions

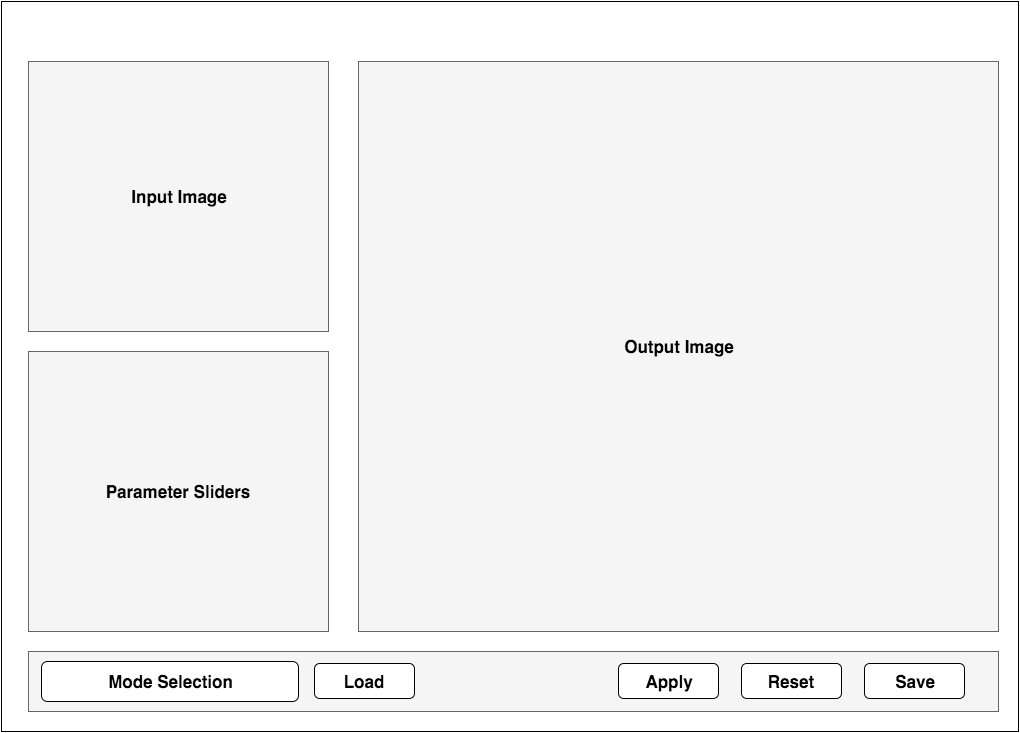


Figure 1: GUI wireframe

The GUI has many functions and parameters that can be set via sliders. There are also different selectable modes which influence the direct behavior of the sliders.

## Global Controls

The global parameters are located along the bottom row and are made up of global controls for working with the effects. The functionality is summarized below.

Mode

Toggles between the Color and Binary modes. Color is meant to transform the image into a color, hand drawn cartoon. All color information in the original image is retained and gradient minimization is used along with other edge related effects to produce the output.

In Binary mode, the output is meant to resemble that of a black and white newspaper graphic, or pop art from the ‘60s. Color information is lost; however, color masks can be applied using the Color Tweak slider.

Color Representation

In addition to the mode selection, the output color representation can be further modified with the Invert and BW radio buttons. These toggles are fairly straightforward and work in all modes. In color, BW converts to grayscale, and in binary, removes the colored mask. The Invert option inverts the colors. In binary mode, this essentially exposes the edges.

Load

Prompts the user to load a local image file.

Apply

Applies the current parameters to the image and displays the output.

Reset

Clears the input and output images

Save

Prompts the user to save the output to file

## Effect Parameters

There are a number of effect sliders that can parameterize the overall transformation applied to the output.

Color Tweak

In Color mode, this adjusts the color *quantization*, spanning from 1 to 255. Lowering this reduces the color gradients and available colors in the display palette.

In Binary mode, this adjusts the *hue* of the mask, spanning from 0 to 255. In BW mode, this slider has no effect.

Smoothness

This is the main parameter for the *gradient minimization*. Increasing this will consequently produce a smoother image with less detail and color gradient, and also require fewer iterations in processing.

Detail

Decreasing this parameter will remove finer details from the output creating a simpler output. Rather than filtering based of gradient and frequency, this parameter is based off feature density. Features that don’t meet the minimum area defined by this parameter are removed.

Morph Toggle

Toggles whether the morphological transformations are applied to the output. The sliders below this are enabled or disabled based on the toggle. The transformation attempts to add a more human style to the extracted borders using erosion for line border thickness, and dilation with a customizable line structuring element.

Stroke Thickness

Controls the thickness of the extracted borders using erosion with a disk structuring element.

Stroke Length

Modulates the length of the line segment used in the dilation process

Stroke Angle

Modulates the angle of the line segment used in the dilation process; from 0 to 360 deg.

# Implementation

The general implementation strategy is covered first, then the discussion is split into a section for both modes.

All code is developed in Matlab, and it has been tested on version 2015b. Some of the functionality relies on the Image Processing Toolbox. The *input* image must be of the form \*.bmp, \*.tif \*.jpg, \*.hdf. The *output* is saved as a \*.jpg.

The processing time is mainly dependent on the inut image size. In general, the Color mode takes between 3 to 8 seconds due to the FFT and IFFT required and the gradient optimization process. In Binary mode, processing usually takes less than a second. While this code isn’t optimized for speed, many considerations were accounted for in the implementation to help speed it up. For example, a simple Sobel edge detection is used rather than a slower more accurate Canny detection. Operations were vectorized as well to avoid loops as much as possible. Both modes rely on processing a binary representation of the input through the same Area Filter fed into the Morphological Transform block.

Area Filter

The area takes binary input and computes the pixel density of the black pixels. If the density for a cluster, or feature is below the value specified in the Detail parameter, that feature is erased. This effectively removes small details that frequently occur in nature, but are usually not included in a cartoon. In Matlab this is achieved with the bwareaopen command.

Morphological Transform

The morphological transform block aims to add human artifacts to the output such as stroke simulation and variable edge thickness. The thickness is achieved by applying erosion with a disc structuring element that takes user input for radius. This effectively thickens all black pixels of the binary input. The stroke pattern and variable thickness is achieved by using dilation with a line structuring element to erase black pixels. Depending on the angle and length specified by the user, the patter can have a variable effect. Often it adds stroke patterns that may happen when an artist has their hand at a constant angle. It also produces interesting variable thickness, especially on rounded edges.

## Color Mode

Gradient Smoothing and Sharpening

The [gradient minimization method](http://iosrjournals.org/iosr-jece/papers/Vol8-Issue3/J0835156.pdf?id=7651) is effective for sharpening major edges by increasing the steepness of transition while eliminating a significant degree of low-amplitude structures. The result is strengthened edges and flattened color gradient with high frequency details attenuated. This contradictory effect relies on an optimization routine that utilizes gradient minimization, which controls how many non-zero gradients are rejected in the transformation process. This method globally locates important edges and doesn’t rely on any local features, making it very universal. The runtime can be controlled by lowering the max\_beta since the while loop executes until the max\_beta value is reach. A larger max\_beta, however is a more optimized result.

Sobel Edge

Simple Sobel edge detection is used to extract some of the defining edges to thicken and add strokes to. Other methods could be used, but Sobel is ideal for speed. The Matlab command used is edge(Img, 'Sobel').

Quantization

Color quantization is used to reduce the color palette to be closer to what an artist may have. This effectively eliminates color gradients and instead has more abrupt color shifts. The Matlab command used is rgb2ind(Img,colordepth,'nodither'). The dither option produces mixed results and often adds dot or fuzziness.

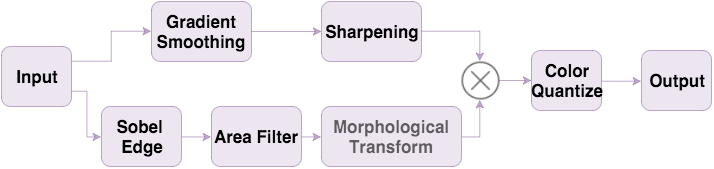
**

Figure : Block diagram for Color Mode

## Binary Mode

Binary

The first step here is to convert the input to binary. The binary threshold is specified with the graythresh(Img) command which uses Otsu's method.

Hue

A colored mask is created based off the *hue* user input. This mask is multiplied by the binary image to fill in the white pixels with the color mask.

Boundary Emphasis

Using the command bwboundaries(Img), the exterior boundaries of objects is traced, as well as boundaries of holes inside these objects. Due to the Area Filter applied before, most holes inside boundaries are removed, but any remaining ones are removed in this process. Only the exterior boundaries are kept and added to the image. This approach offers much more control over which edges are included.

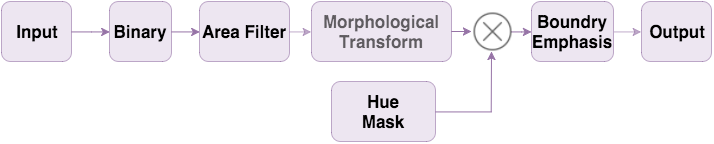


Figure : Block diagram for Binary Mode

# User Guide

# Comments

# Future Research