Project Report
Flow-Based Image Abstraction

Mayank Meghwanshi 110050012

Vivek Atulkar 110050039

November 8, 2014

1 Problem Statement

We implemented image toonificaton technique based on the paper Flow-Based Image Abstraction[1]. The approach is based on shape/color filtering guided by edge tangent field which improves abstraction performance in terms of feature enhancement and stylizations.
We implemented following two subproblem of image abstraction

1. **Line Extraction** - Using Flow-based Difference of Gaussian (FDoG).
2. **Region Smoothing** - Using Flow-based Bilateral Filter (FBL).

2 Algorithms Implemented

2.1 Flow Construction - Edge Tangent Flow

Edge Tangent Flow(ETF) is smooth, feature preserving edge flow field which is used as guiding map for further steps of the filter. *Edge tangent* is defined as vector perpendicular to image gradient, and it represents tangent of local edge flow.
Input - Gray-scaled image, Kernel radius(μ), Number of Iterations
Output - Tangent direction vectors at each pixel
Steps

1. Initialize tangent flow using initial gradient map calculate using sobel operators.
2. Now iteratively update the tangent flow using ETF construction filter described in the paper.
   ETF construction filter is combination of *spatial weight function*, *magnitude weight function* and *direction weight function*
3. Figure 1 shows 3 iterations of ETF filter.

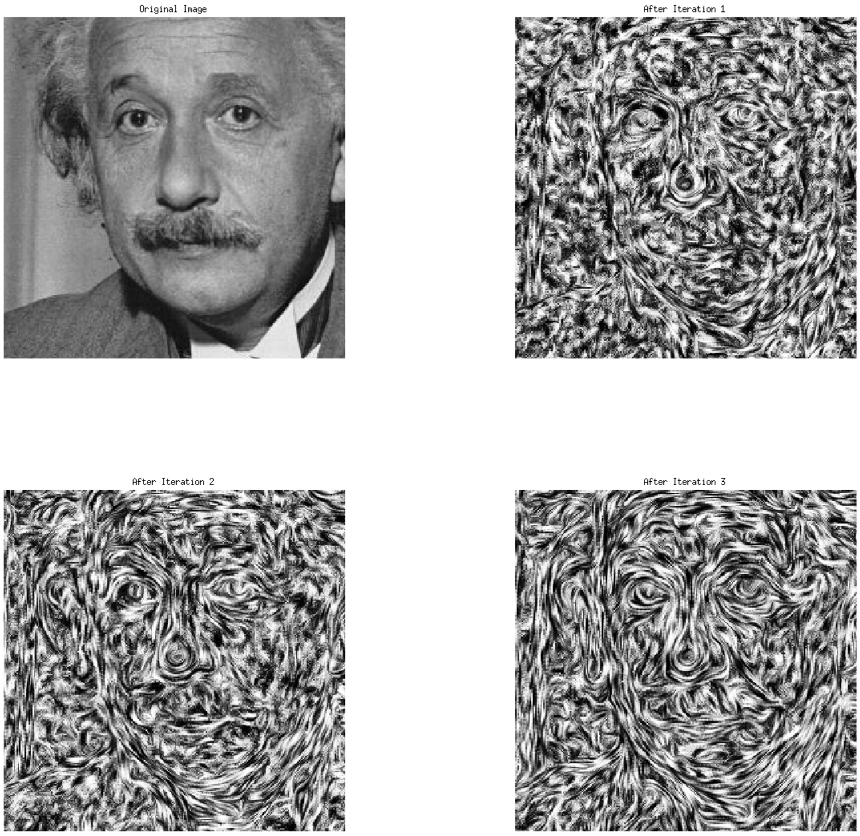

Figure 1: Edge tangent flow. It grows progressively smoother with each iteration.

2.2 Line Extraction - Flow-based Difference of Gaussian Filter

ETF gives us the local tangent direction of the image. In FDoG filter we use this information and apply Difference of Gaussian filter along the perpendicular direction which is more likely to give highest contrast. Then we move along the flow to collect enough information to find genuine edges.
Input - Gray-scaled image, Tangent directions, σ_m, σ_c, ρ, τ, Number of iterations

- σ_m - Represents Gaussian along the tangential direction controls degree of line coherence.
- σ_c - Represents one of the Gaussian used in DoG filter determines width of resulting line.
- ρ - Controls the noise detected.
- τ - Serves as the threshold for binarization of the image.

Output - Binary Edge Image
Steps

1. First we take 2β points along the perpendicular direction of tangent and apply DoG filter on it, do this for all pixels.
2. Then take 2α points along the direction of tangent and apply Gaussian filter on it, do this for all pixels.
3. Use thresholding to generate binarized image.
4. To apply FDoG iteratively first merge edge image to original image and then do above steps again. Figure 2 shows 3 iterations.

2.3 Region Smoothing - Flow-Based Bilateral Filter

We use two filters in FBL, first along the tangent direction and next along the gradient direction(perpendicular to tangent). First operates in edge directions and cleans out shape boundaries and the other one acts perpendicular and smooths out region interiors.
Input - Original Image, Tangent directions, σ_e, r_e, σ_g, r_g

- σ_e - Spatial weight function(Gaussian) along the edge direction
- r_e - Similarity weight function along the edge direction on difference of color
- σ_g - Spatial weight function(Gaussian) along the gradient direction
- r_g - Similarity weight function along the gradient direction on difference of color

Output - Smoothened Image
Steps

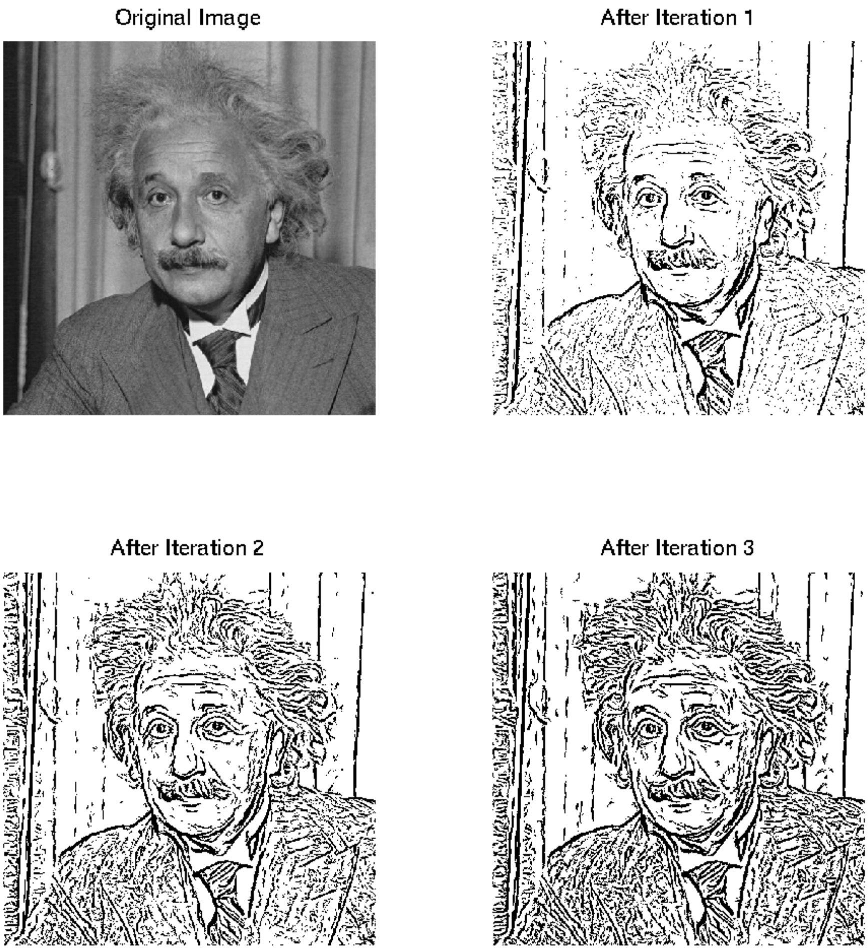

Figure 2: Flow-based DoG. Progressively improves edge coherence

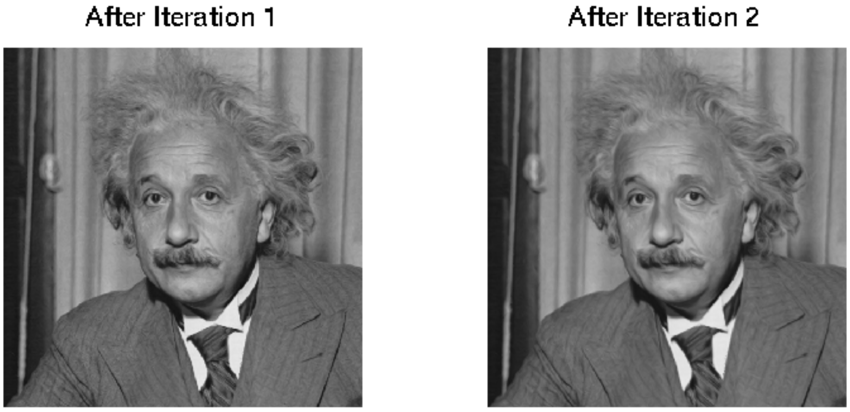

Figure 3: Flow-based Bilateral Filter.

1. First we take 2α points along the edge direction and apply linear bilateral filter on it.
2. Then we take 2β points along the gradient direction and apply linear bilateral filter on it.
3. Apply the mask iteratively. Figure 3 shows 2 iterations of FBL.

2.4 Region Flattening - Luminance Quantization

We quantize the filtered image by uniform-sized-bin luminance quantization. For gray-scale image we quantize directly according to the intensity value. For RGB image first we convert it into Cie-LAB color-map and apply the quantization on luminance part of the image. Then we convert back to RGB image. Figure 4 shows quantized image.

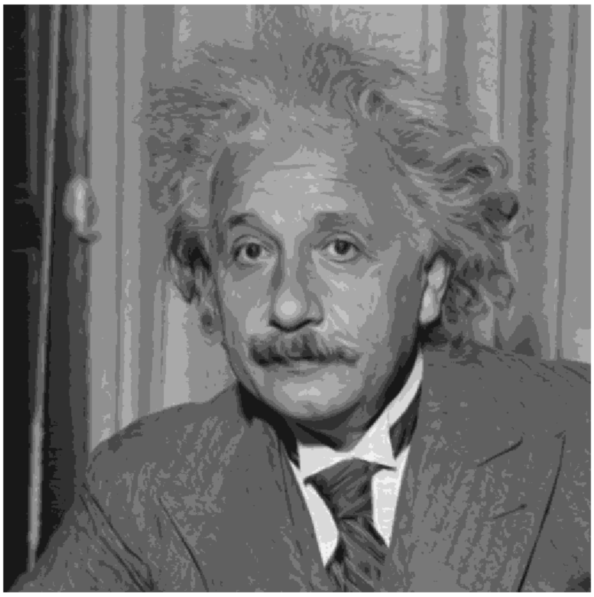

Figure 4: Quantized Image.

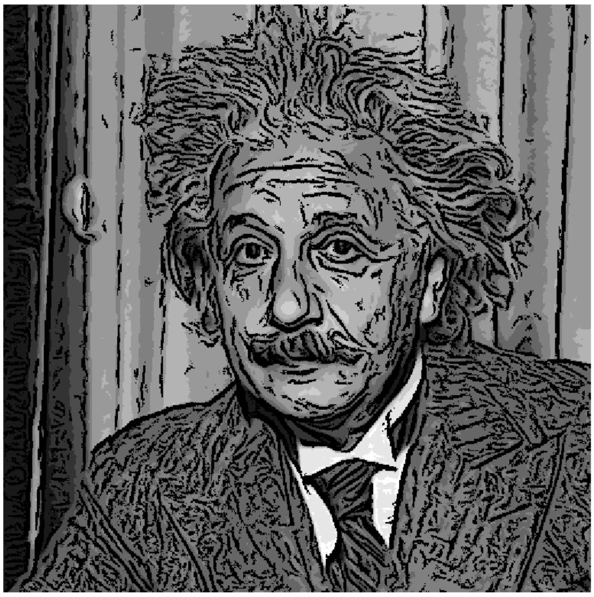

Figure 5: Final Merged Image.

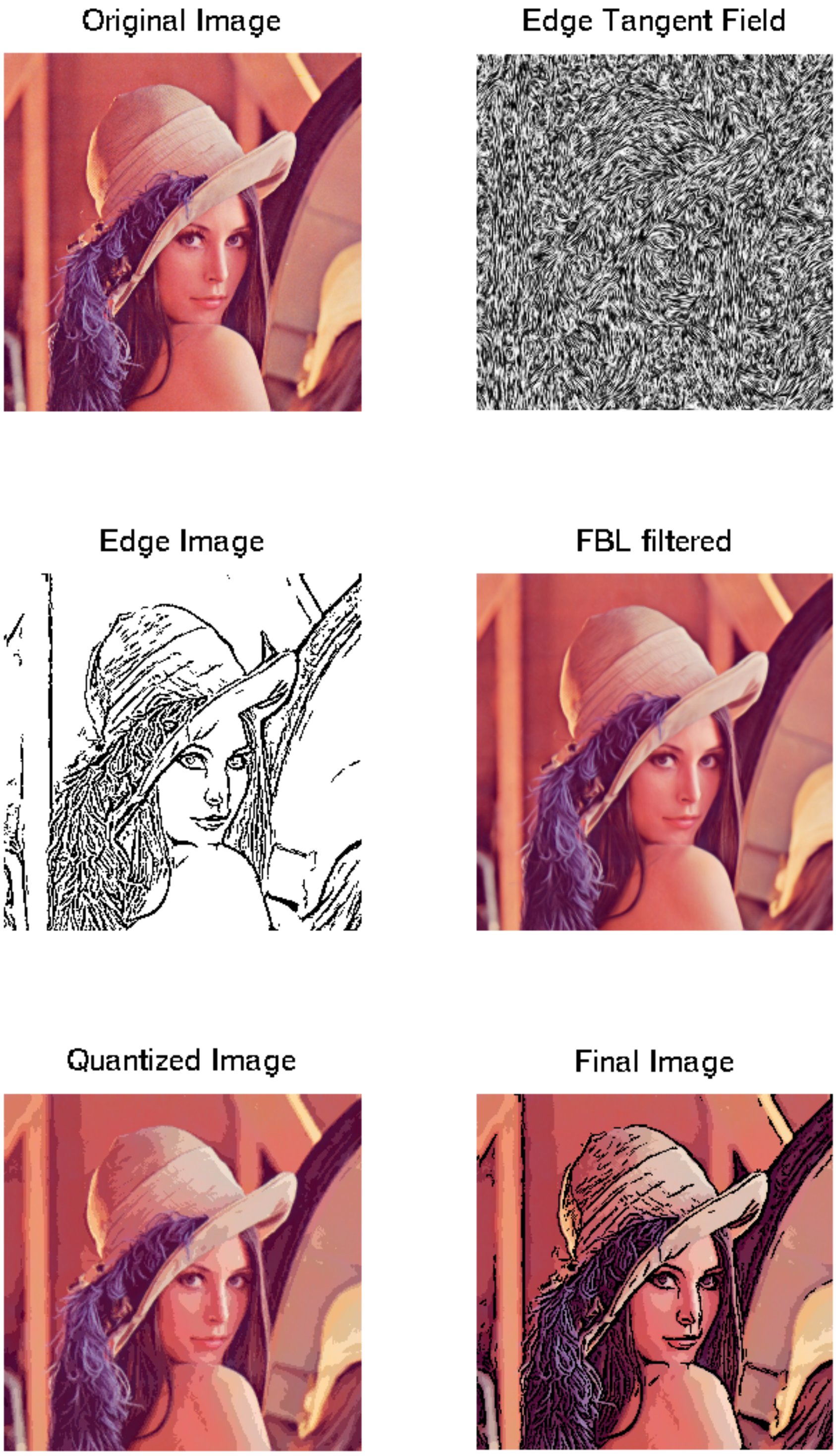

Figure 6: Lenna Image

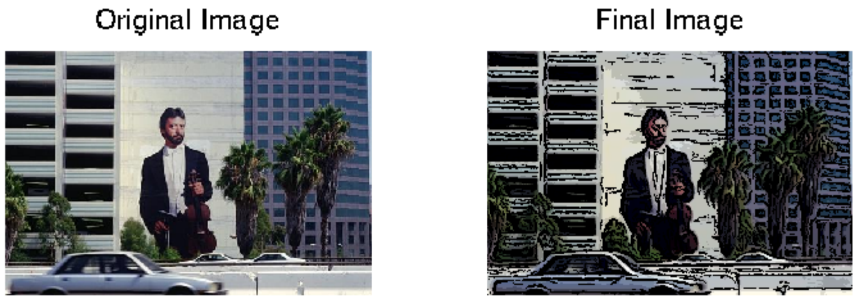

Figure 7: Sample Image 1

Figure 8: Sample Image 2

Figure 9: Sample Image 3 - Original

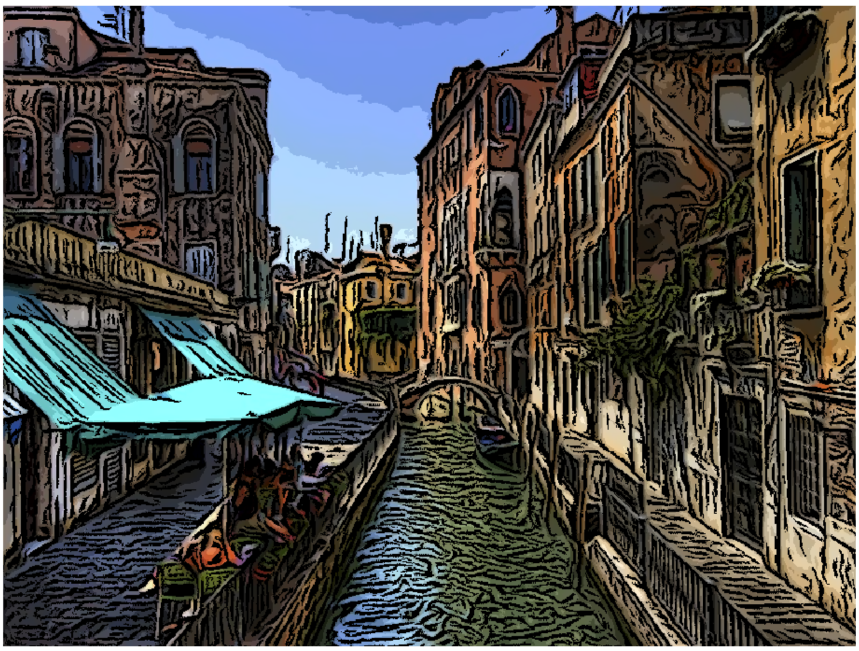

Figure 10: Sample Image 3 - Final

2.5 Limitations

- This approach depends completely upon accuracy of underlying vector field ETF, if that is not properly represent local features then it is not possible to abstract that detail.
- ETF is also not capable of preserving finer texture of the image, which requires it to have small kernel radius.
- This approach have a lot of parameters at each step, for each image the best parameters to set might be different and is subjective in nature.
- Our implementation of FDoG filter does not give as good results as shown in the paper, and shows too many unnecessary edges.

2.6 Advantages

- This algorithm uses two linear bilateral filters instead of single 2-D bilateral filter, which speeds it up relative to traditional bilateral filter.
- This algorithm give thicker and more coherent edges, which are perceptually more meaningful then given by other edge detection algorithms.
- ETF filters enables it to smoothen along the edge and perpendicular to edge separately. Which protects edges and smooths out interior.

2.7 Work Division

Most of the time we worked together while working on the project and each of us have equal contribution to the project.

References

[1] Henry Kang, Seungyong Lee, and Charles K. Chui. Flow-based image abstraction. *IEEE Transactions on Visualization and Computer Graphics*, 15(1):62–76, 2009.