Volume xx (200y), Number z, pp. 1–10

CS4365 Applied Image Processing: Toonification Final Report

R. Kargul (4770625), I. Hashmi (4829360), and R. Huizer (4847652)

Delft, Netherlands
Delft University of Technology

1. Introduction

Images can be abstracted in a way to resemble a toon version of them. This can be done in quite some ways, where three of which are described in this report. The problem in this research is to abstract an image, to give it a cartoon-like look while preserving its visual details and features.

Method 1 approaches this by first smoothing the image while preserving details (such as edges). Next, edges are detected in order to establish lines and extract them from an image. Furthermore, colors in the smoothed image are quantized to give the image a more abstract aspect, almost paint-like. Finally, the quantized image is combined with extracted lines to give the final toonified result. Method 2 creates stylized images by recovering the depth of the image by decomposing the image into base and detail layers, combining the both layers with weighted factors, and applying different non-photorealistic rendering (NPR) techniques, such as halftoning. Method 3 creates an image pyramid of segmented images at various resolutions, after which it blends segments together across different resolutions based on various rules, including eye-tracking data of people looking at the image.

The solutions presented in the report replicate the implementations described in the papers. Some steps are vague, hence the implementations are not exact replicas. Implementation of some steps is difficult (e.g., method 1 image-based warping is based on a paper written in German (from 1999)). Additionally, it is worth keeping in mind that the papers investigated are close to two decades old, which in the computer graphics domain is quite significant.

The methods mentioned above are compared in two ways: the runtime of each algorithms pipeline, and the depiction of "toonification" from the final results. The runtimes can be quantitatively compared, given that all solutions were implemented in Python, only adding bias from running each method estimate on different machines. The visual differences between the three methods are compared by visual appeal and accuracy (perceived correctness).

2. Literature Survey

Toonification, as a research area, has been explored previously. The three main approaches discussed in this paper have been discussed and implemented in other works. Table 2 shows a list of methods with their pros and cons.

Approach / Pros (+) / Cons (-)
Real-time abstraction [WOG06] + Quick processing time + Preserves edges + Lightweight line extraction + Multiple levels of quantization + Step function for edge detection (different weights) + Image-based warping - Low-contrast images abstract too much and lose too much detail
Toonify: Cartoon photo effect application [Dad] + Preserves edges + Quantizes colors (simple) - Toonified images do not contain lines - Canny edge detection (rather costly)
Shape-simplifying image abstraction [KL08] + Direct shape simplification + Mean Curvature Flow (MCF), enhances edges preservation + Shock filtering: blurring along edge directions (adds edges) + Many levels of abstraction (iterative approach) + Incorporating user input to assign weights to certain parts of an image - Toonified images do not contain lines - Advanced/complex implementation
Flow-based image abstraction [KLC09] + Edge Tangent Flow (ETF): takes into account direction of local image structure and color + ETF filter algorithm acceleration (separate filter into x and y directions, to reduce time) + Fast running time - Advanced / complex implementation
Stylization and Abstraction of Photographs [DS02] + Utilizes eye-tracking data to abstract away irrelevant details of the image + Important edges are preserved + Uses various tweakable parameters to determine relevant regions + Can be used to compress images as unneccesary details are removed - Complex algorithms making it slow - Need to gather eye-tracking data for each image - No code implementation available
Abstracted Painterly Renderings Using Eye-Tracking Data [SD02] + Uses eye-tracking data to render abstraced images + Various tweakable parameters to influence rendering - Need to obtain eye-tracking data points from people - Edges are not preserved

Stylized depiction of images based on depth perception [LMJH*10]
+ Uses depth recovery to create stylized depictions
+ Tweakable parameters to influence the small and large features
+ Computes the visibility of new light sources if added by the user
+ High degree of artistic freedom and flexibility
+ Algorithm works for paintings
- Depth recovery is an approximate, not accurate
Stylized Black and White Images from Photographs [MG08]
+ Uses different halftoning techniques
+ Preserves detail as much as possible
+ Different variants of the algorithm
- Complex implementation of certain algorithms

Table 1: Approaches related to "toonification" for all methods

3. Methods

3.1. Method 1: Real-time video abstraction (Radek)

The first method is based on a paper from 2006 [WOG06]. The paper abstracts images by applying a smoothing filter iteratively. It detects edges to preserve contours in the images. The cartoon-like effect is achieved with color quantization. All these steps are combined to create a "toonified" version of an input image. The following steps describe the method implementation:

1. Abstraction (bilateral): using a bilateral filter, which removes noise but preserved edges. The filter with $\sigma_r = 3$ (range weight; intensity) and $\sigma_s = 4.25$ (spatial weight; distance) is used in the work. The filter is used 5 times (iteratively, see Figure 2). Different versions of bilateral filters are also used in different works [KLC09] [KL08] [Dad].
2. DoG Edges: Difference of Gaussians(DoG) applies Gaussian filter at two distinct scales (where $scale_1 = \sqrt{1.6} \times scale_2$) to two images; the difference returns detected edges. This step uses a smooth-step function to increase temporal coherence in animations. It uses a parameter τ that controls the amount of center-surround difference required for cell activation, as well as ϕ_e which controls the sharpness of the activation fall-off. The function gives more control over the edges, where some are stronger than others (see Figure 3).
3. Luminance quantization: a color quantization that maps the color palette into N bins (colors). This adds more paint/cartoon-like effects to an image. The paper uses a pseudo-quantization, which introduces the ϕ_q parameter controlling the sharpness of the transition from one bin to another.
4. Image-based warping: optionally performed by the [WOG06] authors. It is based on a paper written in German [Lov99], and not easily accessible, hence not implemented.
5. Combination: adds extracted lines to the quantized image to give the final result (see Figure 5).

3.2. Method 2: Stylized depiction of images based on depth perception (Irtaza)

The second method is based on the paper from 2010 by Lopez-Moreno et al. [LMJH*10]. This paper extracts the approximate depth information from the input image and highlights certain features for non-photorealistic rendering (NPR) to produce stylized depiction of images. The information we have is the pixel values, therefore we can't recover the depth accurately and the results may contain large errors. However, since our essential goal is to produce stylized images, the recovered depth doesn't need to be physically accurate. Even approximate values of the depth can yield pleasing results. The paper provides multiple methods which were implemented, each one defining a different technique to relight the original image on the basis of the recovered depth, however, we will focus on the halftoning technique in this report. The following steps were taken to implement the algorithm:

1. Compute the luminance values of the input image based on the pixel input using the following formula [I.T90]:
$$L(x,y) = 0.212R(x,y) * 0.715G(x,y) * 0.072B(x,y)$$
2. Implement bilateral filter, an image processing technique that removes noise but preserves edges. This algorithm will be utilized later for the bilateral decomposition step. The bilateral filter is applied with $\sigma_r = 0.2$ (range weight; intensity) and $\sigma_s = min(width, height)/16$ (spatial weight; distance), which are the values suggested by Bae et al. 2006 [BPD06]
3. We decompose the image into a base layer $B(x,y)$ and detail layer $D(x,y)$. The base layer and detail layer are defined as follows:
$$B(x,y) = applyBilateralFilter(InputImage)$$
$$D(x,y) = InputImage - B(x,y)$$
4. The base and detail layers have different distributions, which is problematic as they will be combined with different weights in the next steps. To solve that, tone mapping is applied. One way of doing it is by scaling the base layer down as done by Durand and Dorsey in 2002 [DD02], however, want to enforce the large-scale distribution of tones to match the model image [BPD06]. Therefore, we perform histogram matching to the base layer and transfer its histogram.
5. The final approximate depth is calculated using the following formula:
$$Z(x,y) = Fb * B(x,y) + Fd * D(x,y)$$
Fb and Fd are user-defined weighting factors between 0 and 1, that control the presence of small and large features.
6. The calculated approximation of the depth map is passed as input to halftoning techniques to get the final result. Thresholding and adaptive thresholding [BR07] were implemented to produce stylized black and white images from input images [MG08].

The steps taken above to implement the algorithm don't follow one paper but multiple ones. This was due to my personal interest in applying different techniques rather than following one paper in a sequential manner.

3.3. Method 3: (Rick)

The third method is based on a paper from 2002 by DeCarlo and Santella [DS02]. This paper toonifies images whilst taking into account regions of the image a person looks at. It uses image segmentation and an analysis of eye movements of a person to create a toonified version of an image. The algorithm consists of the following steps:

1. Calculate edge chains that are used later in the algorithm using Canny edge detector.
2. Create an image pyramid where each layer contains a segmented image, which is computed using the algorithm in [CGM02], at a different resolution, with the lowest resolution being the top of the pyramid.
3. Create a hierarchy tree containing image regions at each layer, where parents and children are related by the overlap the segmented regions have across different resolutions.
4. Use fixation data to determine for each fixation point the corresponding region the person was looking at. Fixation data was generated using an online tool to select pixel locations.
5. Calculate the frontier regions from the hierarchy tree using contrast and frequency of regions, along with duration of the fixations.
6. Smooth the borders of the regions, and draw the regions.
7. Add black lines on top of the drawn regions using the edge chains calculated earlier.

Unfortunately, as the paper stems from 2002 the links used in the paper that refer to existing helper algorithms that were used in the paper are no longer online. Therefore, all the steps had to be implemented by hand or with other available solutions online. This results in outputs that unfortunately do not quite look the same as the outputs in the paper. After many hours of debugging I decided to present you with these results, even though they are not as nice as the ones from the paper. I think they are still relevant as all the key ideas and steps from the algorithm are implemented and used to generate the outputs.

4. Evaluation

Comparison is done using a qualitative assessment of the images, and a quantitative assessment of the timings. The timings can be found in Table 2. Based on the timings, we can conclude that all methods have relatively similar runtime, with method 2 taking slightly longer.

It is worth noting, that Method 1 works better on higher-resolution images (Figure 28), but also takes 16x longer (approx. 320 seconds). This however makes sense, as image is 4x larger (in both directions). The iterative smoothing makes a much bigger difference on higher resolutions as well, as the regions get flattened almost to uniform pixel intensities, whereas edges get preserved well. Similarly, color quantization can be better perceive on those images too

Method	Time
Method 1	20.38 (0.92)
Method 2	30.72 (0.81)
Method 3	21.94 (0.12)

Table 2: Average runtime in seconds of 10 iterations. Standard deviation is shown in brackets.

Overall, each method tested returns a different result. The most closely related are methods 1 and 3, where both use lines (borders/edges) and colors to represent toonification. On the other hand, method 2 represents the image in a black-white setting. Based on the input image used, Method 1 gives a cartoon-like result. In conclusion, each method approaches toonification rather differently and therefore results in distinct outputs.

4.1. Method 1: Real-time video abstraction

Figure 1: *Method 1: 1-Input image.*

Figure 2: *Method 1: 2-Iterative bilateral filtering (with* $\sigma_r = 3$ *(range weight; intensity) and* $\sigma_s = 4.25$ *(spatial weight; distance)).*

Figure 3: *Method 1: 3-Line extraction using DoG with the smooth step function ().*

Figure 4: *Method 1: 4-Pseudo quantization with 32 bins and transition sharpness 0.1.*

Figure 5: *Method 1: 5-Combined result image.*

4.2. Method 2: Stylized depiction of images based on depth perception

Figure 6: *Method 2: 0-Input image.*

Figure 7: *Method 2: 1 - Luminance image.*

Figure 8: *Method 2: 2 - Applying custom bilateral filter on the image.*

Figure 9: *Method 2: 3 - Bilateral decomposition, base layer.*

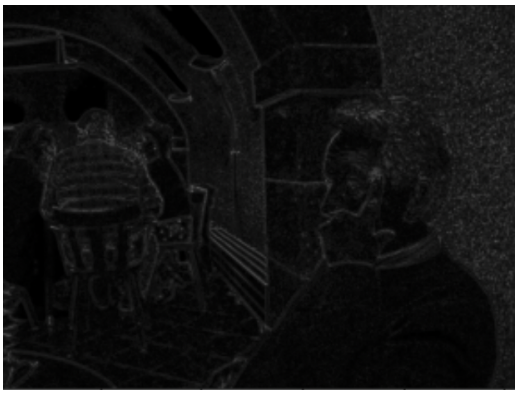

Figure 10: *Method 2: 3 - Bilateral decomposition, detail layer.*

Figure 11: *Method 2: 5 - Combination of the detail and base layer where Fb = 1, Fd = 0*

Figure 12: *Method 2: 5 - Combination of the detail and base layer where Fb = 1, Fd = 0.25*

Figure 13: *Method 2: 5 - Combination of the detail and base layer where Fb = 1, Fd = 0.5*

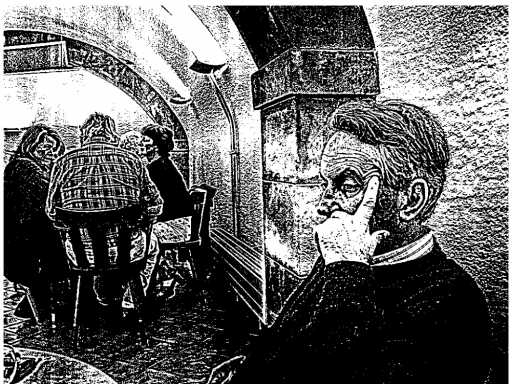

Figure 14: *Method 2: 5 - Combination of the detail and base layer where Fb = 1, Fd = 0.75*

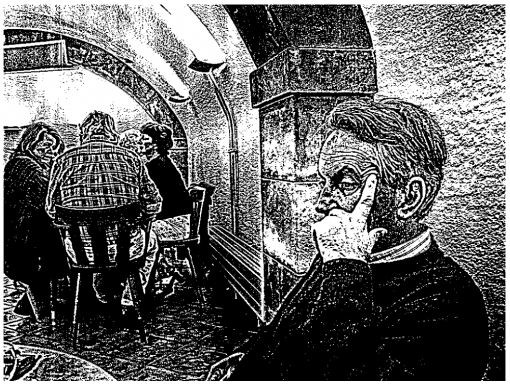

Figure 15: *Method 2: 5 - Combination of the detail and base layer where Fb = 1, Fd = 1*

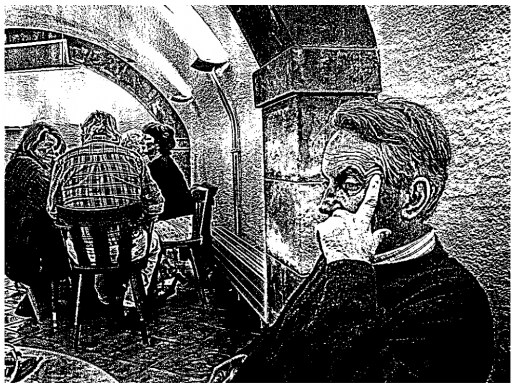

Figure 16: *Method 2: 6 - Halftoning: thresholding.*

Figure 17: *Method 2: 6 - Halftoning: Adaptive thresholding.*

4.3. Method 3:

Figure 18: *Method 3: Highest resolution segmented image using the mean shift algorithm from [CGM02]. $h_s = 7$, $h_r = 6.5$ and $M = 20$.*

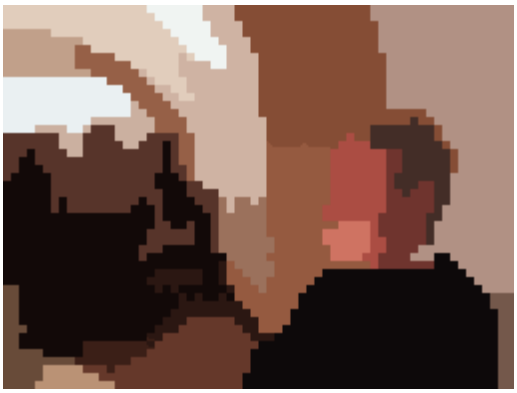

Figure 19: *Method 3: Lower resolution (downsampled by factor 4) segmented image using the mean shift algorithm from [CGM02]. $h_s = 7$, $h_r = 6.5$ and $M = 20$.*

Figure 20: *Method 3: Line image obtained using the edges found by the Canny edge detector. Here $l_{min} = 15$ and $l_{max} = 500$, $t_{min} = 1$ and $t_{max} = 6$. These were obtained by some trial and error, giving better results than the recommended values from the paper.*

Figure 21: *Method 3: The final image where all frontier regions are drawn in coarse to fine order, along with the line image on top of them. Unfortunately due to implementation differences and difficulties this does not look that similar to the papers' results.*

5. Appendix: results on other images

Figure 22: *Method 1: "The man" higher resolution image generates a better-looking result (bilateral filter with σ_s=3, σ_r=4.25, and 15 iterations; quantization with q=24, ϕ_q=2.5)*

Figure 23: *Method 1: "Apples" (final result)*

Figure 24: *Method 1: "Cookie Monster" (final result)*

Figure 25: *Method 1: "Elephant" (final result)*

Figure 26: *Method 2: Halftoning: thresholding (above) and adaptive thresholding (below) applied on Lenna.*

Figure 27: *Method 2: Different combinations of the detail and base layer yield different depictions (here shown for the halftoning technique). The Fb is fixed at 1.0 and Fd is increasing from 0 to 1 in 0.25 increments starting from the top image.*

Figure 28: *Method 2: Halftoning: thresholding (above) and adaptive thresholding (below) applied on old man.*

References

[BPD06] BAE S., PARIS S., DURAND F.: Two-scale tone management for photographic look. *ACM Trans. Graph. 25* (07 2006), 637–645. doi:10.1145/1141911.1141935. 2

[BR07] BRADLEY D., ROTH G.: Adaptive thresholding using the integral image. *Journal of Graphics Tools 12*, 2 (2007), 13–21. URL: https://doi.org/10.1080/2151237X.2007.10129236, arXiv:https://doi.org/10.1080/2151237X.2007.10129236, doi:10.1080/2151237X.2007.10129236. 2

[CGM02] CHRISTOUDIAS C. M., GEORGESCU B., MEER P.: Synergism in low level vision. In *Object recognition supported by user interaction for service robots* (2002), vol. 4, IEEE, pp. 150–155. 3, 6

[Dad] DADE K.: Toonify: Cartoon photo effect application. 1, 2

[DD02] DURAND F., DORSEY J.: Fast bilateral filtering for the display of high-dynamic-range images. *ACM Trans. Graph. 21*, 3 (jul 2002), 257–266. URL: https://doi.org/10.1145/566654.566574, doi:10.1145/566654.566574. 2

[DS02] DECARLO D., SANTELLA A.: Stylization and abstraction of photographs. *ACM transactions on graphics (TOG) 21*, 3 (2002), 769–776. 1, 2

[I.T90] I.T.U: Basic parameter values for the hdtv standard for the studio and for international programme exchange. URL: https://www.itu.int/rec/R-REC-BT.709-1-199311-S/en. 2

[KL08] KANG H., LEE S.: Shape-simplifying image abstraction. In *Computer Graphics Forum* (2008), vol. 27, Wiley Online Library, pp. 1773–1780. 1, 2

[KLC09] KANG H., LEE S., CHUI C. K.: Flow-based image abstraction. *IEEE Transactions on Visualization and Computer Graphics 15*, 1 (2009), 62–76. doi:10.1109/TVCG.2008.81. 1, 2

[LMJH*10] LOPEZ-MORENO J., JIMENEZ J., HADAP S., REINHARD E., ANJYO K., GUTIERREZ D.: Stylized depiction of images based on depth perception. In *Proceedings of the 8th International Symposium on Non-Photorealistic Animation and Rendering* (New York, NY, USA, 2010), NPAR '10, Association for Computing Machinery, p. 109–118. URL: https://doi.org/10.1145/1809939.1809952, doi:10.1145/1809939.1809952. 2

[Lov99] LOVISCACH J.: Scharfzeichner: Klare bilddetails durch verformung. *Computer Technik 22*, 236-237 (1999), 74. 2

[MG08] MOULD D., GRANT K.: Stylized black and white images from photographs. In *NPAR* (2008). 2

[SD02] SANTELLA A., DECARLO D.: Abstracted painterly renderings using eye-tracking data. In *Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering* (2002), pp. 75–ff. 1

[WOG06] WINNEMÖLLER H., OLSEN S. C., GOOCH B.: Real-time video abstraction. *ACM Transactions On Graphics (TOG) 25*, 3 (2006), 1221–1226. 1, 2