Stylized Black and White Images from Photographs

David Mould*
University of Saskatchewan

Kevin Grant†
University of Saskatchewan

Figure 1: *Black and white image conversion. Left to right: original image; belief propagation result; graph cuts result; base+detail result.*

Abstract

Halftoning algorithms attempt to match the tone of an input image despite lower color resolution in the output. However, in some artistic media and styles, tone matching is not at all the goal; rather, details are either portrayed sharply or omitted entirely.

In this paper, we present an algorithm for abstracting arbitrary input images into black and white images. Our goal is to preserve details while as much as possible producing large regions of solid color in the output. Our algorithm involves composing a base layer, consisting of large flat-colored regions, with a detail layer containing the small high-contrast details. The base layer is computed using graph cuts, while thresholded local histogram equalization gives the detail layer. The final labeling is tidied by removing small components, vectorizing, and smoothing the region boundaries. The output images satisfy our goal of high spatial coherence with detail preservation.

CR Categories: I.3.3 [Computer Graphics]: Picture/Image Generation—Display Algorithms; J.5.0 [Computing Applications]: Arts and humanities—Arts, fine and performing

Keywords: halftoning, image filtering, black and white images

1 Introduction

Conversion of grayscale images into binary images is a task long studied in computer graphics, and various halftoning algorithms have been proposed over the years. Oftentimes, such algorithms exchange spatial resolution for intensity resolution: a solid-colored region of the input image is transformed into a pattern of black and white primitives such that the area integral of intensity in the output is the same as the intensity level of the input.

*e-mail: mould@cs.usask.ca
†e-mail:kjg658@mail.usask.ca

However, there are contexts in which low spatial and low color resolutions occur simultaneously. Mobile devices usually have small screens and may have limited color resolution, so images intended for such displays out to be as simplified as possible. Icon design provides another example: icons should be recognizable, but small, so few pixels are available; and often, color is used as a separate information channel, so is not available for image reproduction.

Further, faithful reproduction of tone is not always desirable, even when possible. Some artistic media, such as scratchboard [Lozner 1990], inherently result in black and white images. Also, some artists employ other media such as ink to obtain similar effects. For example, Frank Miller's *Sin City* graphic novels [Miller 2005] use solid blacks and whites without mid-gray levels (other colors are occasionally introduced, their effectiveness heightened by their rarity).

In this paper, we propose an automated algorithm for converting a grayscale image into a black and white image; the intended output image retains sharp features while areas lacking sharp features are converted to solid color. Example output from our algorithm is shown in Fig 1. The results are an instance of abstraction, the process of removing irrelevant information while preserving salient detail; even salient detail is sometimes simplified.

Binarization of grayscale images is in some sense a trivial operation. One method is to choose a threshold, so that pixels brighter than the threshold become white, and darker pixels become black. Unfortunately, it is not easy to find a threshold that does a good job for a particular image; for most images, no single threshold will do a reasonable job. On the other hand, adaptive thresholding techniques suffer from overamplification of noise.

We use adaptive thresholding to provide an initial guess about the fate of individual pixels, and then use energy minimization to obtain a binary labeling; we first approached the minimization problem using loopy belief propagation, and then, using our observations from this first attempt, we designed a new energy formulation and used graph cuts to obtain the labeling. The difficulty of constructing an appropriate energy term that both captured local details and large-scale coherence prompted us to devise a modified method that produced a labeling based on a composition of a detail layer and a base layer; graph cuts was used in building and composing the layers. Following the labeling, the binary image is polished by removing small isolated elements and converted to a vector representation.

The contribution of this paper is to present a trio of related meth-

ods for creating stylized black and white images from input photographic images. The desired labelings retain salient details but have substantial regions of constant color where no details were present. Graph cuts and loopy belief propagation were applied to the problem framed as an energy minimization exercise, and produced unsupervised labelings of an arbitrary input image. With the lessons learned from these, coupled with the observation that local adaptive thresholding preserves detail, we propose a base plus detail algorithm for black and white stylization, where graph cuts are used to obtain a coarse (base) level, local adaptive segmentation is usd to obtain a fine (detail) level, and the two are combined to produce a final image that both preserves details and has large flat-colored regions where warranted.

The paper is organized as follows. In the following section, we review some artistic and scientific work related to our goal. Next, we describe the details of our proposed algorithms. We show results of applying the method to different input images in the following section, and give some commentary on the effectiveness and robustness of the techniques. This is followed by a final section in which we conclude and provide some suggestions for future work.

2 Previous Work

Previous work related to this problem follows three threads: algorithms for halftoning; methods for artistic depictions of input images; and schemes for foreground extraction from images. We consider each of these in turn.

The goal of halftoning is to convert a grayscale (continuous tone) image into a binary (two tone) image, often for reproduction on a low color resolution output device such as a printer. Techniques including error diffusion [Floyd and Steinberg 1977] and ordered dither are commonplace. Both error diffusion and ordered dither use the maximum available spatial resolution; the intent is that over a region, the average intensity of the two-tone output image equals the average intensity of the continuous-tone input image. When the output image has higher spatial resolution than the input, halftoning algorithms can match tones very effectively.

When the output image resolution is the same as or lower than the input resolution, however, halftoning does not always reproduce the input effectively. Sharp features may become less clear, and spurious textures may be introduced (although it is possible to control such textures for artistic effect, as shown by Veryovka and Buchanan [Veryovka and Buchanan 1999]).

Close tone matching of the input image is not always necessary in image abstraction contexts. Various artistic media, including pen and ink [Guptill 1976], engraving, and scratchboard [Lozner 1990], produce images in black and white.

Both pen-and-ink and engraving have seen some attention in past years. Winkenbach and Salesin [Winkenbach and Salesin 1994] gave a scheme for imitating artistic pen and ink illustration and introduced stroke textures; Salisbury et al. [Salisbury et al. 1994] made use of the stroke textures in an interactive application, creating halftoned images which look as though made with pen and ink. Ostromoukhov [Ostromoukhov 1999] presented a method for creating beautiful digital engravings of an input image, but one that requires considerable user input to achieve its effects.

Also worth mentioning in this context is stippling, a particular style of drawing in ink in which the ink is distributed through the output image as tiny discrete dots. Stippling has fascinated computer graphics practitioners in part because of its connections with computational geometry; algorithms for stippling [Deussen et al. 2000; Secord 2002] have often drawn on Lloyd's algorithm [O'Rourke 1990] for centroidal Voronoi diagrams. Usually, problems of stipple distribution have been cast as halftoning problems, deploying stipples as halftoning primitives with the intent of matching tone.

Other researchers have previously proposed automated abstraction problems. Gooch et al. [Gooch et al. 2004] presented a method for creating black and white caricatures of human faces out of photographs. Their method is automatic, but specialized to the case of human faces. While Winnemöller et al. [Winnemöller et al. 2006] perform abstraction on arbitrary photographs and video, they do not abstract all the way to black and white.

Lastly, methods for image segmentation have some relevance to the current problem. We have used the existing frameworks of *loopy belief propagation* (LBP) and *graph cuts* to perform foreground extraction for our stylized images.

In the following subsections, we describe LBP and graph cuts, but first, we give a brief discussion of image segmentation as an energy minimization problem. The energy cost function to be minimized is usually written as a Gibbs energy with a *data cost* component D (the cost of assigning a label to a pixel) and a *smoothness cost* component V (the cost of assigning labels to adjacent pixels).

We write the label of a pixel p as f_p. The data term of the energy is $D = \Sigma_p d_p(f_p)$; that is, the total data energy is the sum over all pixels of individual pixel energies. The smoothness term is $V = \Sigma v_{pq}(f_p, f_q)$, where the sum is taken over all neighboring pixel pairs (8-connected, in our implementations), and $v_{pq}(f_p, f_q)$ gives the smoothness energy of assigning label f_p to pixel p at the same time as assigning f_q to q. The particular functions $d_p(\cdot)$ and $v_{pq}(\cdot, \cdot)$ depend on the application. The goal is to choose a set of labels over all pixels that minimizes the total energy $E = D + \lambda V$. The particulars of the different minimization processes are given next.

2.1 Loopy Belief Propagation

Loopy Belief Propagation is a popular algorithm for performing energy minimizations in image processing. The technique is borrowed from inference in graphical models. In particular, *max-product belief propagation* computes a labeling for any unlabeled variables given an assignment to the other variables in the model. If the graphical model is a tree (no loops), then the computed labeling is optimal (it is the maximum a posteriori assignment, or the labeling with the highest probability). In general graphs, optimality is not guaranteed, but the algorithm has demonstrated itself to compute a reasonable approximation in many circumstances. Here, we focus on belief propagation as it pertains to image analysis; a discussion of loopy belief propagation in general graphs is given by Weiss and Freeman [Weiss and Freeman 2001]. A more thorough description of loopy belief propagation for image analysis is given by Szeliski et al. [Szeliski et al. 2006].

The loopy belief propagation algorithm is easily modified to compute a labeling of minimum energy over an image rather than a maximum probability (the energy is the negative logarithm of the probability); hence, it can be used to compute pixel assignments of low energy. In this framework, each pixel is treated as a random variable whose possible labelings are the potential colours of the pixel. Each pixel also has a neighborhood, which is the 4-neighborhood or the 8-neighborhood of the pixel (in our case, the 8-neighborhood was used). A pixel receives messages from each of its neighbors; these messages influence the likelihood of a pixel taking on a particular label. The message passed from a pixel p to a neighboring pixel is a summary of p's local data and smoothness terms, plus the messages p received from all other neighbors in previous iterations. The algorithm terminates after a fixed number of

iterations, or if it converges to a solution, where convergence means that the likelihoods at each pixel are no longer changing. When the computation is complete, each pixel is assigned the label with the highest likelihood.

2.2 Graph Cuts

Graph cuts are another framework for energy minimization, and in recent years graph cuts have seen widespread use in computer vision and computer graphics [Boykov et al. 2001; Boykov and Jolly 2001; Li et al. 2004; Rother et al. 2004]. The basic idea is to treat the image as a graph, typically a 4-connected or 8-connected grid, and to connect each node (pixel) to two additional *terminal* vertices, the source and sink. Edge weights are computed based on relevant image properties, and the *minimum cut* is determined: the set of edges with minimum total cost whose removal will separate source and sink. After cutting, each pixel will be connected to only one terminal, giving it its label.

The minimum cut for the two-terminal case can be straightforwardly calculated based on max-flow. In our case, we are attempting to partition the graph into two pieces, so we can employ this method. We used an implementation of min-cut/max-flow provided by Kolmogorov [Boykov and Kolmogorov 2004]. Our remaining task is to decide what weights to assign to the edges.

Typically, terminal weights are assigned according to the data term of the Gibbs energy, while neighborhood weights are determined by the smoothness energy. The neighborhood weights may be given by something as simple as a linear energy term, such as

$$V_{pq}(f_p, f_q) = 0, f_p = f_q \quad (1)$$

$$V_{pq}(f_p, f_q) = |i_p - i_q|, otherwise \quad (2)$$

where i_p is the intensity of pixel p. In our application, we employ a variation of this energy term, and derive terminal weights from a comparison of a pixel's value to the mean value in its neighborhood, as detailed in Section 3.2.

3 Algorithms

Figure 2: *Thresholding and adaptive thresholding.*

This section describes the system we built to convert images to black and white. First, though, we seek to supply some insight into the problem and some additional motivation for the need for a complex algorithm to solve it.

Fig. 2 shows the perils of thresholding. The left-hand result used a single threshold (the mean image intensity) over the entire image, while the right-hand result used a variable threshold (the mean intensity within a window). The problems with such thresholding schemes are clear. Global thresholds ignore local contrasts, while local thresholds preserve local contrast but amplify noise in relatively flat regions of the image.

At first, we thought that the outcomes from local and global thresholding might be combined in some way to produce an adaptive thresholding algorithm that suited our needs. However, after spending some effort attempting to create a more sophisticated thresholding scheme, we came to the conclusion that thresholding alone was not a good approach to the problem, and we turned to other segmentation methods. We present here two variants of an algorithm for converting grayscale images to black and white, one using loopy belief propagation and the other using graph cuts. The surrounding processing is the same in each case, but LBP and graph cuts approach the core segmentation problem differently; also, we used different energy frameworks for the two algorithms, using the experience of LBP to design better energy terms for use with graph cuts.

The segmentation-based black and white conversion system consists of four stages. First, image statistics are extracted from the original image: standard deviation, local and global means, and gradient magnitudes. Second, the image is separated into black and white regions using either graph cut or loopy belief propagation. Third, the black and white image is smoothed by removing small isolated regions. Finally, boundaries of regions in the raster image are extracted and drawn with splines; optionally, the vector boundaries can be further smoothed at this stage.

Although adaptive thresholding is not suitable for the segmentation, we can use the outcome of adaptive thresholding as a set of priors we can use as input to the more sophisticated segmentation algorithm (energy minimization, using one of loopy belief propagation or graph cuts). In particular, the distance of a pixel from its neighborhood's mean was found to be an energy term well-suited to representing local detail. Our first attempt to solve the problem combined global and local thresholding results and labeled the output using loopy belief propagation. In our second attempt, we obtained a data energy term using only local thresholding and minimized the energy using graph cuts. As we will see, graph cuts did a better job preserving detail while flattening the low-contrast regions. However, the images resulting from LBP are not without their charm.

The progression of an image through this pipeline is shown in Fig 3. The top image shows a visualization of the data energy term used for graph cuts, where only the local neighborhood is considered; it is actually an alternative visualization of the image in its own right, akin to "embossing" effects obtained by gradient-based edge detection. The second image is the outcome from the segmentation, in this case graph cut. The third image shows the result of despeckling by eliminating connected components smaller than a certain size (25 pixels, here) and the bottom image is the final vectorized version of the black and white raster image. Connected components labeling and boundary extraction can be done using well-known algorithms such as those found in textbooks [Shapiro and Stockman 2001]. The source image for this sequence is shown in the teaser in Fig. 1.

The heart of the algorithm is the segmentation in the second stage. We present here a brief overview of the idea; additional details for LBP are given in section 3.1 and details for the graph cut are given in section 3.2.

Our segmentation-based algorithms preserves most details, although very fine scale details such as hair texture are often merged. However, large regions of the image can change color based on a desire for coherence with local details; ideally, we would want the sky region in the sheep texture to become a single color. We found it difficult to manage both coherence and detail in a single image. Accordingly, we split the problem into a base layer (promoting coherence) and a detail layer (containing local features). The details

Figure 3: *Progress through the pipeline: graph cuts. Top: visualization of data energy. Middle: initial segmentation. Bottom: result after despeckling.*

of this algorithm are given in section 3.3.

3.1 Segmentation using LBP

We first approached the labeling problem using loopy belief propagation. In this section, we describe the energy term that we sought to minimize.

As before, we associate a cost with a label f_p for a given pixel p; the LBP scheme will attempt to determine the minimum cost configuration of labels for all pixels in the image. A label can be either black or white; we define the function $s(f)$ such that $s(black) = -1$ and $s(white) = 1$. We will also use the two-argument function $s(f_p, f_q)$ which returns 1 if its arguments are the same, -1 otherwise.

We compose the data cost function for pixel p from a local energy component and a global energy component. The global energy component for a given pixel is determined by comparing the pixel's value i_p with the mean image intensity μ. Let $\delta_G = (\mu - i_p)/(M+1)$, where M is the maximum possible pixel value (in our case, 255). The division by $M+1$ puts the value in the range $(-1, 1)$. Finally, the the *global energy cost* of labeling the pixel with label f_p is given by

$$G(f_p) = -ln(0.5 + 0.5 \cdot s(f_p) \cdot (\delta_G)^{\gamma_D}). \quad (3)$$

Similarly, we compute the local energy component for a pixel by comparing the pixel's value i_p with be the mean intensity of its neighborhood μ_p. We first compute $\delta_L(p) = (\mu_p - i_p)/(M+1)$. The *local energy cost* of labeling the pixel with f is then

$$L(f_p) = -ln(0.5 + 0.5 \cdot s(f_p) \cdot (\delta_L)^{\gamma_D}) \quad (4)$$

where $s(\cdot)$ is defined as before. Functions 3 and 4 punish (with a higher energy) a pixel label that falls on the opposite side of the computed mean as the original pixel value, while rewarding labelings that fall on the same side of the mean as the original pixel value. The parameter γ_D affects the magnitude of the energy; the higher the value of γ_D, the more severe the punishment and rewards are for even small values of δ_G and δ_L. We experimented with different values of γ_D, and obtained the most pleasing images with $\gamma_D = 0.3333$ (cube root).

The global energy and local energy term are combined to give the data term through a simple weighting function:

$$D(f_p) = (1 - \alpha)G(f_p) + \alpha L(f_p), \quad (5)$$

giving the data term for pixel p.

The parameter α allowed us to tune the amount of local features in the resulting images. Higher values of α retained much of the spatial frequency of the original image, while lower values created larger segments of one colour. We explored different settings for this parameter; values around 0.8 seemed to give the best result.

To compute the smoothness cost between two neighboring pixels p and q, we used the gradient magnitude values of the image. Let g_p represent the gradient magnitude at pixel p, and define GM to be a constant larger than the gradient magnitude of any pixel. Let $\delta_V(p, q) = (GM - max(g_p, g_q))/GM$. Given these definitions, the smoothness energy is

$$V_{pq}(f_p, f_q) = -ln(0.5 + 0.5 \cdot s(f_p, f_q) \cdot \delta_V(p, q)^{\gamma_V}) \quad (6)$$

where as stated previously, $s(\cdot, \cdot)$ returns 1 if its inputs are the same and -1 otherwise. This function punishes (rewards) neighboring

pixel labelings that are different (the same) with a high (low) energy value. The parameter γ_V governs the effect of the gradient. The higher the value of γ_V, the less severe the punishment and rewards are for values of δ_V. As with parameter γ_D, we experimented with different values of γ_V, and found the most pleasing images occurred with $\gamma_V = 8$.

Figure 4: *Iterations from LBP. Top: one iteration; 20 iterations. Bottom: 40 iterations; 60 iterations.*

Fig. 4 shows the results of the first 60 iterations of LBP on the subway image. The image is still changing slightly at this point; the final image is shown in Fig. 1. We consider LBP results to be final after 100 iterations. The initial labeling is dominated by the data term, and as time goes on and messages transmit information through the graph, the smoothness term exerts a larger and larger influence over the results.

We were not completely satisfied with our LBP results. Although we were usually able to get a somewhat reasonable segmentation, the features in the final image appeared blurred relative to the original image content. Also, and critically for our goal of unsupervised segmentation, we often had to resort to a trial and error process to find good parameters for a given image (including iteration count). Accordingly, we took the lessons from LBP and designed a new energy formulation which we minimized using graph cuts. Graph cuts were found to be a better energy minimization framework than LBP on a set of benchmark problems [Szeliski et al. 2006] and we were hopeful that using graph cuts to minimize the new energy terms would produce better images overall. Details of this endeavour are given next.

3.2 Segmentation using Graph Cut

Our initial experiments with LBP suggested that the global data energy term was not helpful, and although we were not able to obtain reasonable results by eliminating it from LBP, we decided to design a new energy formulation for graph cuts which excluded this term.

Using graph cuts, we exercise control over the segmentation by setting edge weights in the graph. We have terminal weights T and neighborhood weights N; the terminal weights represent the data term, and the neighborhood weights represent the smoothness term. We consider each of these in turn.

We determine for each pixel a signed energy value E describing its distance from the local mean:

$$E = \mu_p - i_p, \tag{7}$$

where i_p is the pixel value and μ_p is the mean in a local neighborhood surrounding p (in our implementation, a square neighborhood with radius 10). A positive E indicates a tendency towards black and a negative E a tendency towards white. We allocate a probability r to a pixel as follows:

$$r = \frac{1}{2} e^{-zE^2/2\sigma^2}, \tag{8}$$

where σ is the standard deviation of intensity values in the input image. For normally distributed pixel intensities, comparing E to σ provides a good characterization of whether an intensity difference is strong or not. The parameter z allows us to adjust the tradeoff between the data energy and the neighborhood energy; larger z means more attention to the data, while a smaller z leads to weaker terminal links and correspondingly greater emphasis on coherence. We used $z = 1$ in all single-stage graph cuts segmentation; however, we will use this parameter later, in the base plus detail algorithm presented in section 3.3.

The value r from equation 8 is the probability of the less likely terminal; $(1 - r)$ is the probability of the more likely terminal. Notice the factor $1/2$, needed because E is signed; an E of zero means that both terminals have likelihood 0.5.

If we use the above probabilities as weights directly, the resulting labelings are too noisy, since outlier pixels will have one terminal assigned a large absolute weight. In consequence, we compute an attenuation factor λ:

$$\lambda = 4e^{-E^2/16\sigma^2}. \tag{9}$$

The value of λ decreases with larger energy, meaning that the greater the bias of a pixel towards one label, the more likely we are to disregard that bias. This is counterintuitive, but has the effect of ensuring that individual pixels, even with large energies, still need the cooperation of their neighbors (i.e., the data weight of an isolated pixel cannot swamp the neighborhood weight). Small regions of pixels, and (especially) step edges, will still have a large total data weight, so that small and medium features can be shown in the labeling. The terminal weights are set as follows:

$$T_{better} = \lambda(1 - r), \tag{10}$$

and

$$T_{worse} = \lambda r, \tag{11}$$

where "better" and "worse" are with respect to the sign of E.

The neighborhood weights are computed conventionally, based on image gradient. For gradient magnitude g at the location in question, the edge between the pixels is assigned weight

$$N = e^{-g^2/2\sigma^2}. \tag{12}$$

This is the neighborhood weight suggested by Boykov and Jolly [2001].

3.3 Base plus Detail

It has been difficult to get both the appropriate level of noise reduction and the desired amount of detail preservation simultaneously from a unified segmentation algorithm. However, the energy terms we used previously had a natural division into global and local terms. In this section, we discuss a method that tries to obtain

better results by creating a base-layer image in parallel with a detail-layer image, and then combines the two to create the final image. This design philosophy follows the successful photo manipulation work done by Bae, Paris, and Durand [Bae et al. 2006], in which the bilateral filter provides a base/detail separation and the two layers are processed separately before being merged into a single final image.

In Figure 2, we can see that adaptive thresholding is fairly effective at capturing details. Here, we propose a three-level classification scheme: a pixel is classed as "definitely black", "definitely white", or "don't know". We use the intensity variance over the entire image to obtain our threshold for "don't know": pixels further than a threshold away from the average value in their neighborhood are known, while those within the tolerance are classed as "don't know". The known values from this process form a detail layer, while a base layer is obtained by running graph cuts as above but with stronger neighborhood links ($N = e^{-kg^2/2\sigma^2}$, with $k = 4$) and weaker terminal links ($z = 1/50$, in equation 8). Our approach is summarized in Figure 5.

The stronger neighborhood links produce a base layer with high spatial coherence, but the details have been obliterated. We then use the adaptive threshold technique to give us a detail layer. For the results shown in this paper, we used threshold $\tau = q \times \sigma$, where σ is the standard deviation of the intensities over the entire image, and q is a parameter. We used $q = 0.4$ for all images. Note that the pixel intensities are being compared against the mean value in a local neighborhood, where we would expect the values not to vary much; our choice of q represents a rather conservative threshold, in the sense that only quite dramatic differences are preserved as details. If preferred, the user can change q, although we did not need to for any of the examples we show.

Once the base and detail layers are built, we combine them with a final application of graph cuts. Care must be taken with this final cut not to destroy the detail we have collected; We therefore restrict the region to be cut, as follows.

Where the detail layer provides us a known result, we take the terminal energy from the local value. Where the detail layer has a "don't know" label, we take our energy term from the base layer. The neighborhood links are taken from the intensity differences (equation 12). Then, graph cuts segmentation is only applied to the detail region and a narrow band around it (5 pixels wide in our implementation); this is shown as a blue region in step 3 of the pipeline in Figure 5. Further from the detail regions, the final values are taken directly from the base layer. This composition step usually does not change the details that were in the detail layer, but does in some cases slightly improve the region shapes. A particular form of improvement is when a potential feature has a very weak gradient with the surrounding flat-colored region; in such cases, these regions can be thinned to lines, such as the line indicating the top of the sheep's head and horns in the final image in Figure 5.

Figure 6 shows base and detail examples for some test images. The detail layer reintroduces the small features that were eliminated to create the base layer, while the base layer provides sensible defaults in regions where there are no details present. Even when details do populate a region, the base layer gives a sensible background color; the shape of the man's face, and the structure of the mustache, are emphasized by the differences in the base layer.

4 Results

In this section we show some additional results from applying our algorithm to input images. While there are some parameters available to tune our algorithm, such as neighborhood size, we have

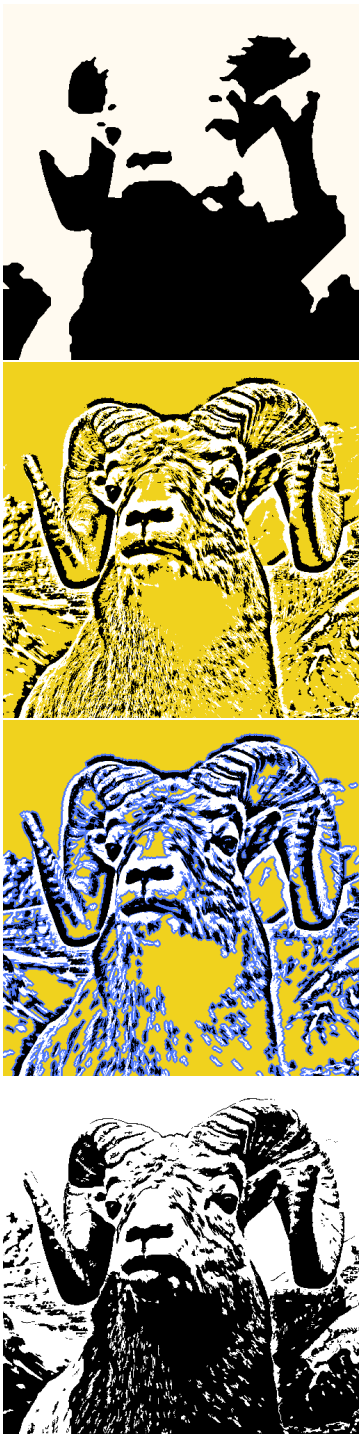

Figure 5: *Base plus detail overview. Top: base layer from graph cut segmentation. Next: detail layer from 3-way threshold. Next: extended detail regions for composition. Bottom: composed base and detail.*

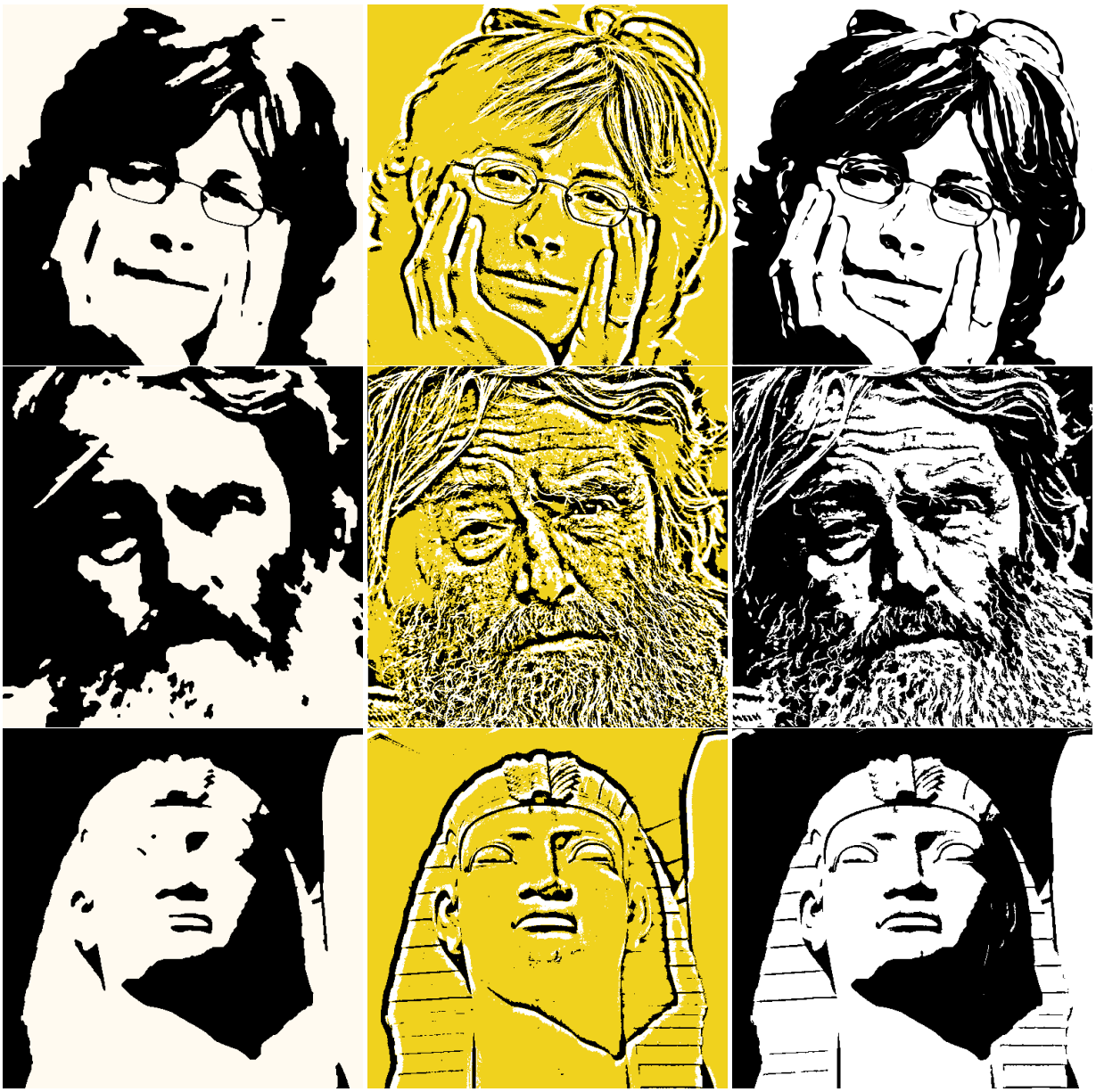

Figure 6: *Base plus detail. Left to right: base layer; detail layer; combined base and detail.*

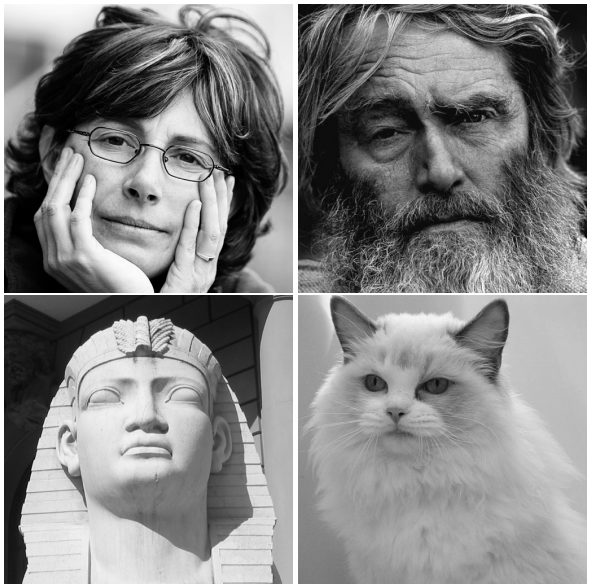

Figure 7: *Original images.*

found that the default settings are robust for a variety of images. The user may want to adjust the scale parameters in the final speckle removal step. Also, the scheme allows the usual lightweight interaction method of user-assisted segmentation with graph cuts (setting hard constraints), both in the graph cuts version and the base plus detail version.

We can assess our results by evaluating to what extent we met our initial goals. Recall that we wanted to preserve sharp features while flattening out noise and background areas into single-color regions. We ran our algorithm on a variety of images with different subjects, different scales of details, and different contrast levels.

A collection of black and white image conversions is shown in Fig. 8. We consider these to be success cases; the figures or objects in the scene are clearly recognizable, and subtle details have been preserved. We are particularly pleased with the outcome of the cat image, a difficult case because of the extremely low contrast between the cat's body and the wall behind it. High frequency details have been simplified, but remain present (such as in the beard of the old man); where the image is not changing much, the results from both graph cuts and base plus detail have a single solid color.

Our completely automated approach leaves us with some limitations. Because we have been fairly cautious in setting the size of regions to be removed (keeping even quite small regions so as to preserve small high salience details such as people's eyes) we sometimes retained spurious regions, such as in the top left of the sphinx image. The images also excessively preserve background detail, as seen in the mountains of the sheep image. An artist would doubtless make different decisions about which parts of the image are important, and indeed, minor human-assisted postprocessing can eliminate the specific problems we mention. However, in the absence of human intervention, these results are credible.

Because we implemented two variants of energy minimization, we can also compare the LBP results to the graph cuts results. From the standpoint of our initial goal, preserving details while eliminating background, we believe the results of graph cuts superior to LBP. Details are more sharply resolved by graph cuts. The LBP images possess a uniform style which observers have characterized as "soft", "dreamy", or "melted". While not all of these terms are entirely flattering, the consistent style of the LBP processed images is one virtue of the technique.

One reason for the difference between the graph cuts and LBP results is the use of a global energy term in LBP and the absence of this term in graph cuts. In the Lena image, for example, the top central area is marginally lighter than the average, so LBP pushes it towards white; we have no such pressure from the graph cuts edge weights, so the neighborhood flow allows it to be flat colored. Although we attempted to remove the global term from LBP, we found it useful in enforcing local coherence (noise amplification was worse when the global term was eliminated).

Lastly, we believe that the base plus detail algorithm produces the best quality results of all. Both coherence and detail preservation are better than in either of the other two algorithms. The main limitation of our methods, common to all three algorithms presented, is in their application to highly textured areas. The base plus detail algorithm faithfully preserves the textures, such as the hair of the sheep and the old man's beard; the two energy minimization approaches attempt to simplify the texture by merging regions to reduce boundary length, with not entirely satisfactory results. Texture simplification is an outstanding problem in image abstraction and we have not solved it here.

We report graph cuts and base plus detail times relative to a 3.0GHz P4 with 1GB RAM, while LBP times are relative to an Intel Core Duo 2 E6300 with each core at 1.86 GHz, and 2 GB RAM. LBP was terminated after 100 iterations of message passing. All images were of size 512×512. Under these conditions, graph cuts produces results in about 2 seconds per image, while LBP takes about 20 seconds. Graph cuts, which usually produced better quality images, is thus roughly an order of magnitude faster than LBP, typically requiring about 2 seconds per image. The results from the base plus detail algorithm require two iterations of graph cuts, executing in about 4 seconds, although a good approximation of the final image (doing a trivial compositing step) is available after 2 seconds.

5 Conclusion

In this paper, we have presented an algorithm for converting input photographic images into stylized black and white images. The core of the algorithm is the segmentation from energy minimization, for which we have implemented graph cuts and loopy belief propagation. We were able to achieve superior results using graph cuts. Following this success, we devised a base plus detail formulation that uses local thresholding to produce a detail layer, and graph cuts to get a base layer and to merge the base and detail layers together. The spatial coherence and detail preservation capabilities of this final algorithm are better than either of the single-stage algorithms.

The images our method produces have considerable detail where there were high or medium contrasts in the original image, and are solid colored elsewhere. This provides an abstracted black and white version of the input, and allows us to automatically produce images reminiscent of some scratchboard or inked images. Our algorithms were tested on images with different levels of contrast and different quantities of texture, and found to be robust over a range of image characteristics.

There remain avenues for future work. Likely the most significant of these is texture indication, a longtime problem in image abstraction. Reliance on contrast produces overrepresentation of texture edges (low salience details). In the context of this work, the question may be framed thus: can we construct energy functions that distinguish between texture edges and feature edges?

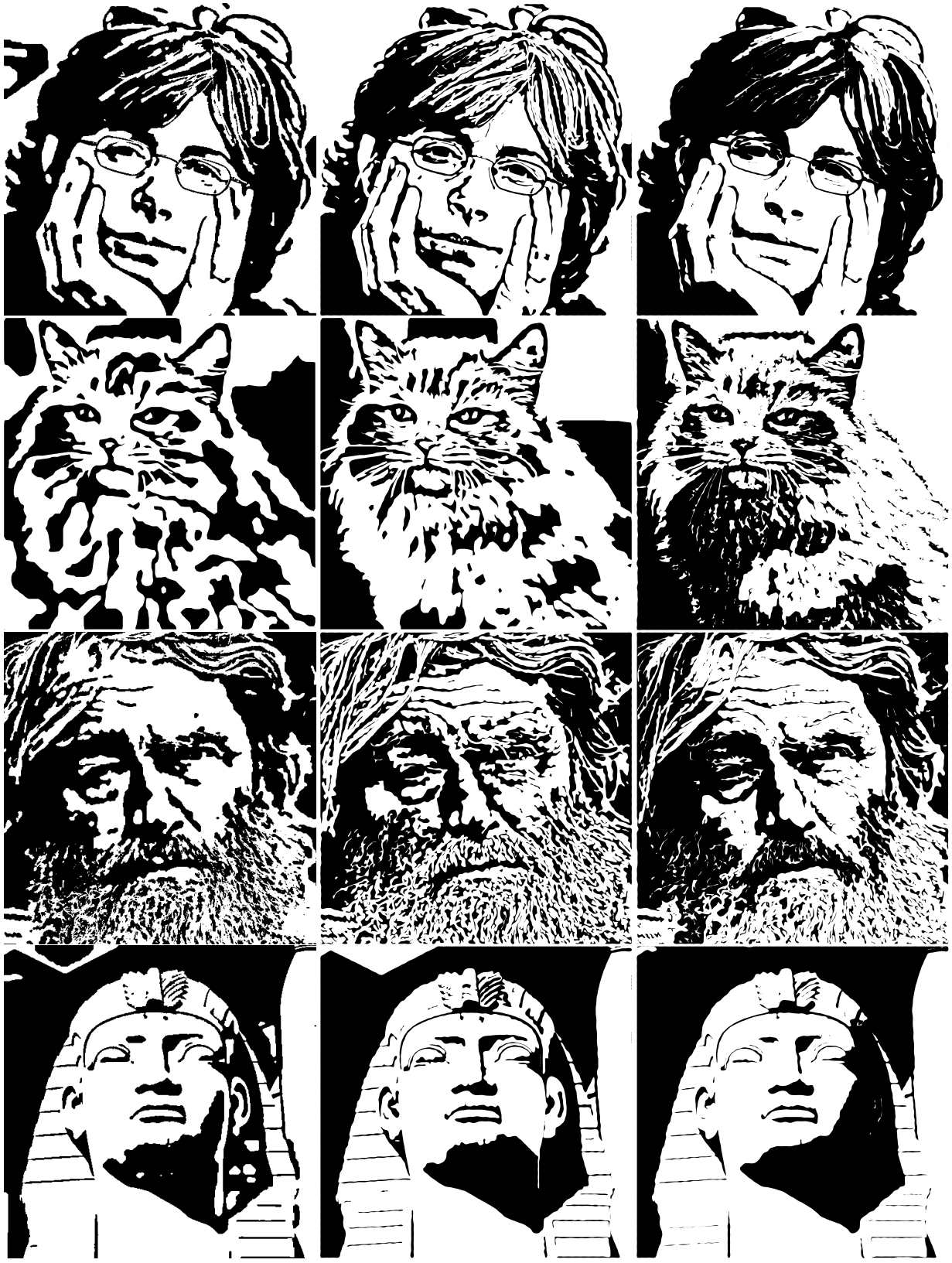

Figure 8: *Image conversion results. Left to right: labeling with LBP; labeling with graph cuts; base plus detail. Top to bottom: portrait; cat; old man; sphinx.*

We would like to improve our discrimination of foreground and background regions, in part to eliminate background textures from the final image, and one cue to doing this is blur. Professional photographs often use a relatively narrow depth of field, so that only the subject is in focus. If we can assume that the background is blurred, we may be able to consider that when creating our detail layer and thus avoid preserving less-salient background details.

Furthering the goal of converting images to black and white, we can consider converting color images. Color opponency can give us additional contrast information and potentially provide a labeling with better detail. We might also consider image sequences. The present system processes individual images independently, and minor changes to the image content can cause entire sections to flip from black to white. For coherent image sequences, it is necessary to consider information from surrounding frames, perhaps biasing the energy locally by the labeling at the previous frame.

Acknowledgements

Thanks to Mark Eramian for useful discussions concerning graph cuts and image filtering. This work was supported in part by NSERC RGPIN-299070-04.

References

BAE, S., PARIS, S., AND DURAND, F. 2006. Two-scale tone management for photographic look. *ACM Trans. Graph. 25*, 3, 637–645.

BOYKOV, Y., AND JOLLY, M.-P. 2001. Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images. In *International Conference on Computer Vision*, 105–112.

BOYKOV, Y., AND KOLMOGOROV, V. 2004. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. Pattern Anal. Mach. Intell. 26*, 9, 1124–1137.

BOYKOV, Y., VEKSLER, O., AND ZABIH, R. 2001. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell. 23*, 11, 1222–1239.

DEUSSEN, O., HILLER, S., VAN OVERVELD, C., AND STROTHOTTE, T. 2000. Floating points: A method for computing stipple drawings. *Computer Graphics Forum 19*, 3, 40–51.

FLOYD, R. W., AND STEINBERG, R. 1977. An adaptive algorithm for spatial grey scale. *Proceedings of the Society for Information Display 15*, 75–77.

GOOCH, B., REINHARD, E., AND GOOCH, A. 2004. Human facial illustrations: Creation and psychophysical evaluation. *ACM Trans. Graph. 23*, 1, 27–44.

GUPTILL, A. 1976. *Rendering in Pen and Ink*. Watson-Guptill Publications, New York.

LI, Y., SUN, J., TANG, C.-K., AND SHUM, H.-Y. 2004. Lazy snapping. In *Proceedings of SIGGRAPH 2004*, 303–308.

LOZNER, R. 1990. *Scratchboard for Illustration*. Watson-Guptill Publications, New York.

MILLER, F. 2005. *The Hard Goodbye (Sin City, Book I)*. Dark Horse, Milwaukie, Oregon.

O'ROURKE, J. 1990. *Computational Geometry in C*. Cambridge University Press, Cambridge.

OSTROMOUKHOV, V. 1999. Digital facial engraving. *Proceedings of SIGGRAPH 1999* (August), 417–424. ISBN 0-20148-560-5. Held in Los Angeles, California.

ROTHER, C., KOLMOGOROV, V., AND BLAKE, A. 2004. "GrabCut" – interactive foreground extraction using iterated graph cuts. In *Proceedings of SIGGRAPH 2004*, 309–314.

SALISBURY, M. P., ANDERSON, S. E., BARZEL, R., AND SALESIN, D. H. 1994. Interactive pen-and-ink illustration. In *Proceedings of SIGGRAPH 1994*, ACM Press, New York, NY, USA, 101–108.

SECORD, A. 2002. Weighted Voronoi stippling. In *NPAR '02: Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering*, 37–43.

SHAPIRO, L., AND STOCKMAN, G. 2001. *Computer Vision*. Prentice-Hall, Inc., Upper Saddle River.

SZELISKI, R., ZABIH, R., SCHARSTEIN, D., VEKSLER, O., KOLMOGOROV, V., AGARWALA, A., TAPPEN, M. F., AND ROTHER, C. 2006. A comparative study of energy minimization methods for Markov random fields. In *Proceedings of the Ninth European Conference on Computer Vision*, 16–29.

VERYOVKA, O., AND BUCHANAN, J. 1999. Halftoning with image-based dither screens. In *Proceedings of Graphics Interface '99*, Canadian Human-Computer Communications Society, 167–174.

WEISS, Y., AND FREEMAN, W. 2001. On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs. *IEEETIT: IEEE Transactions on Information Theory 47*.

WINKENBACH, G., AND SALESIN, D. 1994. Computer-generated pen-and-ink illustration. In *Proceedings of SIGGRAPH 1994*, 163–170.

WINNEMÖLLER, H., OLSEN, S. C., AND GOOCH, B. 2006. Real-time video abstraction. In *Proceedings of SIGGRAPH 2006*, ACM Press, 1221–1226.