Digital Reconstruction of Halftoned Color Comics

Johannes Kopf
Microsoft Research

Dani Lischinski
Hebrew University

Screened Input | Accurate Reconstruction | Painterly Rendering

Figure 1: *We reconstruct an accurate digital representation from a halftoned color print scanned at high resolution. Our result may be viewed at significant magnification, compresses well, and lends itself nicely to painterly re-rendering (shown on the right), as well as advanced editing, such as decomposition into layers for a tour-into-the-picture application (see supplementary video). Input image © Dupuis.*

Abstract

We introduce a method for automated conversion of scanned color comic books and graphical novels into a new high-fidelity rescalable digital representation. Since crisp black line artwork and lettering are the most important structural and stylistic elements in this important genre of color illustrations, our digitization process is geared towards faithful reconstruction of these elements. This is a challenging task, because commercial presses perform halftoning (screening) to approximate continuous tones and colors with overlapping grids of dots. Although a large number of inverse haftoning (descreening) methods exist, they typically blur the intricate black artwork. Our approach is specifically designed to descreen color comics, which typically reproduce color using screened CMY inks, but print the black artwork using non-screened solid black ink. After separating the scanned image into three screening grids, one for each of the CMY process inks, we use non-linear optimization to fit a parametric model describing each grid, and simultaneously recover the non-screened black ink layer, which is then vectorized. The result of this process is a high quality, compact, and rescalable digital representation of the original artwork.

Keywords: comics, descreening, inverse halftoning, vectorization, non-photorealistic rendering

Links: DL PDF WEB

1 Introduction

Emerging from newspaper comic strips in the early 20th century, comic books and graphic novels gradually evolved into a mass medium, and have become a major part of popular culture. Today, these books are purchased, collected, and read by millions of people around the globe. In fact, graphic novels are one of the last varieties of the printed form that are still gaining in popularity [Weiner 2010].

Despite the advent of electronic publishing and the increasing popularity of e-books, facilitated by mobile reading devices such as Amazon's Kindle and Apple's iPad, most classical comic books are still only available in their original printed form. Although some comic book publishers offer a limited amount of classical material as "special digital editions", it is apparent that the creation of these digital editions involved intensive manual intervention. This is not surprising, since as discussed below, simply scanning such books does not do justice to this medium, which often features intricate and detailed line artwork and masterful coloring.

Our goal in this work is to provide a new tool for automated conversion of scanned color comic books, as well as some other types of hand-drawn color illustrations, into a new, high-fidelity rescalable digital representation, suitable for today's high resolution digital reading devices. Black line artwork and lettering are the most important structural and stylistic element of comic art: black lines are used to draw outlines, boundaries of panels and speech bubbles, and as hatching strokes to convey shading and texture. Therefore, both our digitization process and the resulting representation strive to faithfully reconstruct the original line artwork and lettering, as well as to allow significant magnification, enabling full-screen viewing of individual panels on devices such as the 2048×1536 iPad.

The main challenge that we face is rooted in the printing process used to produce color comics. Most printing technologies are unable to produce continuous color or grayscale tones, resorting instead to *halftoning*: approximating continuous tones using various dot patterns. Color comic books, as well as virtually all books, newspapers, and magazines, are printed with AM halftoning, which attempts to simulate changes in tone by varying the size of the dots on a regular grid (*clustered dot screening*). Scanning such prints inevitably captures the halftone screen patterns. Viewing such scans on raster displays results in Moiré patterns due to the interaction between the halftone patterns and the display raster. The effectiveness of various image processing operations is also hindered by the presence of the halftone screen patterns. Thus, it is necessary to invert the halftoning process, an operation also known as *descreening*.

(a) Screened Input (b) Spatial filtering (Gaussian + Median) (c) Local extrema decomposition *[Subr et al. 2009]* (d) Vector Magic

(e) Fourier domain (Descreen 5.0) (f) GIMP plug-in (g) Training-based descreening *[Siddiqui and Bouman 2007]* (h) Our Result

Figure 2: *A comparison of existing descreening and vectorization results. Input image © Dupuis. The supplementary material contains a comparison to an even wider array of techniques on several inputs.*

Although a large number of inverse halftoning and descreening methods have already been proposed, we show that they are not well suited for our purposes. These techniques are typically based on either spatial or Fourier domain filtering, and the results they produce typically exhibit blurring of the intricate linework and sometimes introduce artifacts in the smoothly colored regions, as may be observed in Figure 2. In contrast, our approach is more specifically designed to descreen color comics. We take advantage of the fact that, in most color comic books, the non-black colors are printed using screened C, M, and Y inks, while the black line art is printed using non-screened solid black ink to better preserve its crispness and contrast [Grais 2012]. While not all comic books use this process (bi-tonal Japanese Manga being a notable exception), this is the process used by the majority of North American and European comic book publishers, including DC Comics and Marvel Comics, which together account for over 77 percent of the US market share [Diamond Comic Distributors 2012].

Our approach is to model the color comics printing process in a rigorous manner and to invert this model using non-linear optimization. We use the Neugebauer equations to predict the appearance of printed colors as a combination of the process CMY inks and the color of the paper. We introduce a simple, yet flexible, parametric model for modeling the shapes of the individual AM screening dots. The black ink layer is modeled as a binary Markov Random Field (MRF). Given a high-resolution scan of the printed artwork, we employ non-linear optimization to recover the CMY color screens, simultaneously with the non-screened black ink layer. We then apply vectorization to the black layer to extract a crisp, resolution independent representation of the linework. The reconstructed color densities of the CMY channels are interpolated to yield a continuous color image, which may be rendered as is, or by applying an NPR filter when appropriate, to support high-definition magnification. An example result is shown in Figure 1.

Our method produces a compact representation geared at capturing all of the information that can be extracted from the print, within the limitations imposed by the color channels screening grids sampling rate. The size of the representation depends on the number of printed screen dots, but not on the resolution of the scan. Thus, the representation is quite compact and may be effectively used for hi-fidelity compression of scanned comics. Furthermore, having a vectorized representation of the black layer supports editing and creatively manipulating digitized artworks, as demonstrated in the supplementary video.

2 Background and Related Work

Comics production process

In order to gain a better understanding of our goals and approach it is helpful to first briefly outline the traditional comics production process [Abel and Madden 2008]. The process starts with the artists producing line drawings for the panels on the page. Typically, the panels are first drawn with a pencil on a large sheet of paper. This initial drawing is then embellished or traced with black ink, using a pen or a brush. In the American studios this is typically done by two different artists, referred to as the *penciller* and the *inker*, while in Europe this is more commonly done by the same artist. Letters are typically added by another person, the *letterer*. The artwork is later photographically reduced to its intended printing size and colors are added by the *colorist*, originally using dyes or watercolors, and more recently also using a computer. The colored drawing is then separated into three (C,M,Y) color channels. These three channels are halftoned using AM screening, resulting in a screen with a different angle for each channel, and printed using the corresponding C, M, and Y process inks. The black artwork is printed using the fourth (K) process ink, but since this is essentially a binary image, it is printed without screening. This technique results in improved contrast and allows the outlines of the black artwork to remain very crisp, unlike the boundaries of the halftoned colorized regions.

Our primary challenge in this work is to reproduce the elements printed in non-screened black ink as accurately as possible, since the black artwork and lettering constitute the most important structural and stylistic element of comic art. However, in certain types of comic books, the color regions are also masterfully painted, exhibiting intricate shading and watercolor textures. Thus, our second challenge is to capture these details as well, while at the same time avoiding visible effects of the screening. Note, however, that the color regions have essentially been band-limited by the frequencies of the CMY screening grids, and therefore we do not attempt to reconstruct sharp features inside these regions.

Halftoning and Descreening

As mentioned earlier, continuous tone images must typically undergo a halftoning process before they can be printed. Halftoning methods may be classified into two main groups [Kipphan 2001]: amplitude modulation (AM) methods that respond to changes in tone by changing the size of the dots, and frequency modulation (FM) methods that locally adjust the density of the dots. AM halftoning is most commonly done using *clustered dot screening* [Stoffel and Moreland 1981], while FM halftoning typically em-

ploys *dispersed dot screening* [Bayer 1973] and *error diffusion* [Floyd and Steinberg 1976]. In this work, we focus on AM halftoning, which is widely used in the commercial offset printers on which virtually all books, newspapers, and magazines are printed.

The simplest approach to eliminate or suppress the halftone patterns is lowpass filtering, for example using a Gaussian filter. Unfortunately, this approach also has the unwanted side effect of smoothing the image details and blurring the sharp edges. According to online tutorials [Chastain 2012], somewhat better results may be obtained using a combination of Gaussian blurring and median filtering, followed by unsharp masking using fine-tuned parameters (Figure 2b). Another option, that might appear promising at first glance, is to apply a state-of-the-art edge-preserving filter (e.g., [Farbman et al. 2008; Subr et al. 2009]), but in our experiments we found the results unsatisfactory, as can be seen in Figure 2c. Attempting to vectorize the scanned image (either directly or after lowpass filtering) introduces contouring and also fails to faithfully reconstruct the original line art, as seen in Figure 2d.

The periodic dot grids employed in AM screening produce spikes at specific frequencies, suggesting that it might be more effective to perform the filtering in the frequency domain. For example, both the commercially available Descreen 5.0 plug-in for Photoshop [Sattva 2011], and the free plug-in for GIMP [2011] operate in the Fourier domain and claim to remove the screen pattern while preserving image detail (Figures 2e-f). Stanger et al. [2011] also operate in the frequency domain by constructing and applying a *swiss cheese* filter: band-reject filter with Gaussian-shaped holes around frequencies that arise from the halftone patterns of the different inks. Our technique, however, is capable of producing crisper, more accurate reconstructions, as shown in Figure 2h.

Many other research papers and patents address descreening and inverse halftoning. A good survey may be found in a recent paper by Siddiqui and Bouman [2010]. According to this survey, representative methods include inverse halftoning via projection methods [Wong 1995], via MAP estimation [Stevenson 1997], segmentation-based descreening [Jaimes et al. 1998], gradient-based adaptive filtering [Kite et al. 2000], wavelet-based descreening [Neelamani et al. 2000], and training-based descreening [Siddiqui and Bouman 2007]. However, most of these previous methods aim to avoid the Moiré patterns that emerge when attempting to display or print scans of halftoned prints, and do not account for the specific characteristics of the color comics printing process. In contrast, our goal is to produce a rescalable, resolution independent representation suitable for digital manipulation and viewing, and our approach is tailored to a specific printing process.

Stevenson [Stevenson 1997] posed the inverse halftoning problem as one of reconstruction. The problem is solved using nonlinear optimization, assuming knowledge of the halftoning process. A non-Gaussian MRF is used, which resembles the smoothness term used in edge-preserving filtering techniques [Farbman et al. 2008]. However, this approach is designed to reconstruct a grayscale image from a binary halftone, and is not suited for color halftone prints where several screening grids overlap.

The Training-Based Descreening method [Siddiqui and Bouman 2007] appears to be the state-of-the-art among the recently published descreening methods. This method essentially applies a modified bilateral filter [Smith and Brady 1997] using a denoised version of the grayscale channel to compute the range differences. The denoising stage involves training to compute the denoising parameters that perform optimally for each type of printer. This method produces good results for reprinting a scanned print (at the same size), but it does not attempt to extract resolution-independent primitives as we do. Thus, it is not well suited for our goals, as shown in Figure 2g.

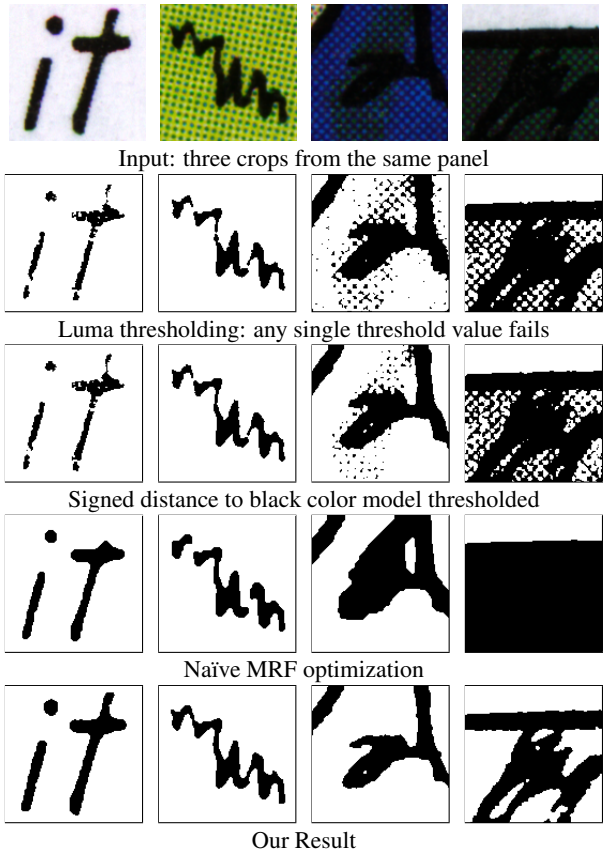


Figure 3: *A number of simpler approaches we tried fail to accurately extract the black ink layer (rows 2–4). The black color model was built from the colors found in a small fully black rectangle. The naïve MRF optimization uses the signed distance to the color model as data term and a Potts smoothness term. We obtain our results using a refined data term and interleaving black estimation with dot recovery (Section 6).*

3 Overview

Given an RGB scan of a printed comic book page, our goal is to reconstruct the original continuous artwork from which the print was produced. As explained earlier, this artwork actually consists of two separate parts: (a) the black artwork, including boundaries, contours, hatching strokes, and lettering, all of which is printed non-screened using the process black ink. (b) the colors that were added to the drawing by the colorist, all of which are printed using screened C, M, and Y inks.

Thus, our goal translates to the task of inverting the printing process and decomposing the RGB scan into four channels: the non-screened black ink layer, which may then be vectorized to yield a resolution-independent representation of the black artwork, and the CMY color screens from which continuous colors may be reconstructed in a number of ways. All of this should be done in a manner that is robust to the typical imperfections in both the printing and the scanning processes.

It should be noted that accurate extraction of the black ink layer is very challenging. The difficulty is rooted at the color ambiguities commonly found in printed materials. The density of the black ink across the print tends to be non-uniform, and it is often the case that pixels and small areas where all three CMY inks overlap are as dark, or even darker than some areas covered by black ink. This is particularly problematic in areas where such CMY overlaps are adjoining black regions, as demonstrated in Figure 3.

We have initially experimented with a number of approaches for first separating out the black ink and then descreening the underlying color channels. Specifically, we attempted thresholding, training classifiers for detecting black ink pixels, modeling the black ink layer as an MRF with simple data and smoothness terms, and using heuristics based on analyzing straight line profiles. However, none of the above succeeded to extract the black ink layer in a sufficiently accurate manner, as shown in rows 2-4 of Figure 3. As demonstrated in Figure 2, we also attempted to descreen the images without explicitly extracting the black ink layer using a variety of filters and existing descreening approaches.

Having failed to decouple the process of black layer extraction from color reconstruction, we gravitated towards a more sophisticated solution, which boils down to performing both tasks *simultaneously* using joint non-linear optimization, as described below.

Using the Neugebauer trilinear model for printed color formation [Neugebauer 1937], we first estimate the screened ink layers for the C, M, and Y inks, and detect the dot locations in each screen. Next, we apply numerical optimization in order to recover the centers and the shapes of the individual AM dots, while simultaneously recovering a binary mask indicating the presence of black (K) ink. Finally, we vectorize the binary black ink mask, and reconstruct a continuous image for each of the CMY channels by interpolation of the recovered dot sizes, since these dot sizes encode the local intensity of the corresponding ink.

In summary, the outline of our method is:

1. Estimate the primaries for the Neugebauer trilinear model and compute a CMY color separation by inverting the Neugebauer equation (Section 4).
2. Detect the dot grid for each of the CMY channels (Section 5).
3. Use optimization to jointly recover a binary black ink mask and the individual dot shapes (Section 6).
4. Vectorize the black ink mask and filter the color channels (Section 7).

4 Recovering the CMY separation

4.1 Estimating the Neugebauer primaries

The RGB color resulting from a combination of halftones printed using cyan, magenta, and yellow inks is commonly predicted using the Neugebauer equations [Kipphan 2001]:

$$\begin{aligned} c_{rgb} \;=\; & c_w(1-\alpha_c)(1-\alpha_m)(1-\alpha_y) + c_c\alpha_c(1-\alpha_m)(1-\alpha_y) \\ & + c_m(1-\alpha_c)\alpha_m(1-\alpha_y) + c_y(1-\alpha_c)(1-\alpha_m)\alpha_y \\ & + c_{cm}\alpha_c\alpha_m(1-\alpha_y) + c_{cy}\alpha_c(1-\alpha_m)\alpha_y \\ & + c_{my}(1-\alpha_c)\alpha_m\alpha_y + c_{cmy}\alpha_c\alpha_m\alpha_y \end{aligned} \quad (1)$$

The eight coefficients $c_w, c_c, c_m, c_y, c_{cm}, c_{cy}, c_{my}, c_{cmy}$, referred to as the *Neugenbauer primaries*, specify the RGB color of the paper, the individual inks, and their various combinations. The terms $\alpha_c, \alpha_m, \alpha_y$ denote the respective ink coverages.

We use a simple automatic procedure to estimate all eight Neugenbauer primaries for each page (in our experience there is some variability between different pages of the same book). The idea is to robustly estimate the extrema of the page's color gamut. For example, to estimate the color of the cyan primary we compute the maximal value of the product $(1-r)\cdot g\cdot b$ over the entire page, and average the colors of those pixels for which this product is within 1 percent from the maximal value. The other primaries are estimated in a similar fashion. In the supplementary material we include a quantitative evaluation of this technique on ground truth data.

4.2 Inverting the Neugenbauer equation

Our task is now to estimate the CMY color separation that could produce the scanned RGB image using the Neugenbauer equations with the recovered primaries. To facilitate this task we compute a three-dimensional lookup table that implements the inverse of Eq. (1), i.e., find $\alpha_c, \alpha_m, \alpha_y$ given c_{rgb}. Note that this lookup table has entries for both in-gamut and out-of-gamut colors, since the scanned input images always contain some out-of-gamut pixels (e.g., many of the pixels covered by black ink will be out of gamut).

Mahy and Delabastita [1996] show that Eq. (1) can be inverted by relating it to a hexic polynomial whose roots determine the possible values for α_c. The remaining coefficients, α_m and α_y, are then found through algebraic manipulation. The polynomial has up to six solutions, so, the inverse of Eq. (1) is not uniquely defined, which could result in non-smooth separations making the subsequent dot shape fitting difficult. Some of these solutions might also not be physically meaningful, e.g. by having a non-zero imaginary component, or $\alpha < 0$ or $\alpha > 1$. In practice, however, this does not seem an issue since for all scans we tried almost all non-black pixels had a unique physically meaningful solution.

We initialize the lookup table by setting all entries that have exactly one physically meaningful solution, and fill the remaining entries using a smooth membrane interpolation. This is accomplished by iteratively replacing every entry by the average of its six neighbors ("Laplacian smoothing"). The smoothing is performed in a coarse-to-fine manner. Computing a 100^3 lookup table takes about a second. Once the table is complete we use it to convert each of the pixels in the input RGB scan. An example color separation is shown in Figure 5.

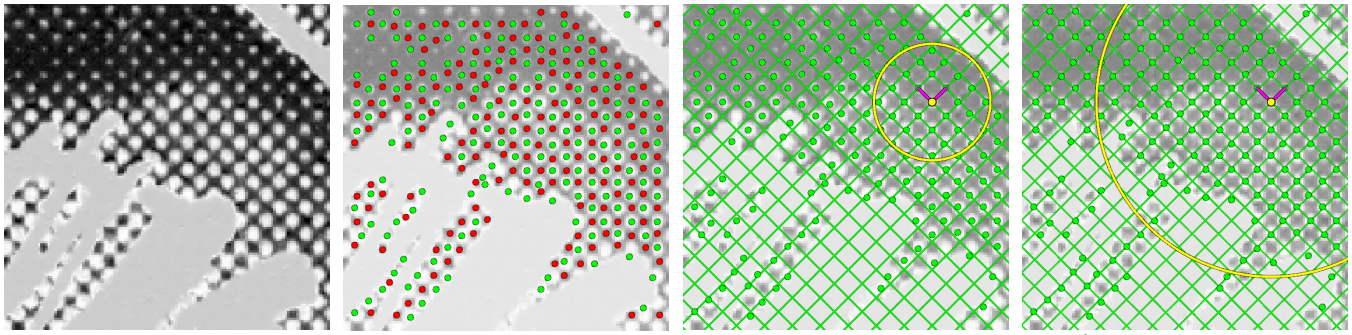
5 Grid detection

In this stage, our goal is to locate the centers of the dots in each of the separated CMY channels, taking into account that these dots were originally supposed to form a regular grid. Park *et al.* [2009] describe a method for detecting 2D wallpaper patterns in real photographs using mean-shift belief propagation. However, they face a much more challenging lattice reconstruction problem and the performance of their algorithm makes it impractical for recovery of large screening grids. Liu [1996] recovers the halftone lattice parameters in the Fourier domain, but his method does not seem to account for grid distortions. Other parametric solutions encounter the same difficulty. Thus, we propose our own simple and efficient image-space approach. We start by detecting local maxima and minima in each channel and then use least squares to recover the grid explicitly accounting for the presence of a smooth deformation field, which may arise in practice due to imperfections in both the printing and the scanning processes.

Local extrema detection: Since the input images (and hence the computed separations) contain a significant amount of noise, we first smooth them with a Gaussian filter. The filter width is set to 0.25 of the grid spacing. We detect both local minima and local maxima in each channel image, because in areas where the dots are large enough they blend with each other. In such areas it is easier to recover the grid by identifying the gaps between the dots (local minima), rather than the dot centers (local maxima).

Local extrema pruning: Local extrema might still fire in relatively flat regions. Therefore, we prune the detected extrema by fitting the region surrounding an extremum at (x_e, y_e) with a paraboloid of the form $A + B\left((x-x_e)^2 + (y-y_e)^2\right)$, and keeping only the extrema for which the paraboloid is sufficiently pronounced, i.e. $|B| > \tau^{\text{mag}} = 0.003$. Figure 4b shows the results of extrema detection and pruning.

Grid fitting: Next, we iteratively fit a grid to the extrema that survived the pruning stage. We must cope with noise that still remains in the extrema location, and with erroneous extrema that survived the pruning (e.g., in a gap between nearby black lines). Furthermore, we must be able to handle not perfectly regular grids, since smooth deformations occur across the paper due to imperfect print-



(a) Input (b) Extrema (c) 1st iteration (radius 25 pixels) (d) 2nd iteration (radius 75 pixels)

Figure 4: *Grid detection. (a) Input image. (b) Pruned extrema are shown as green (maxima) and red (minima) dots. (c–d) Two intermediate results from the grid optimization (minima are omitted here for clarity). The anchor $\vec{a}$ is shown as a yellow dot, and the spanning vectors $\vec{s}$ and $\vec{s}^{\perp}$ as purple lines. The horizon radius is shown in yellow. Note the better fitting of dots away from the anchor in (d).*

ing and scanning. Finally, the grids are not complete, since there may be large white or black regions.

To overcome these difficulties, we use a parametric model for the grid that explicitly incorporates a smooth deformation field, and formulate the fitting as an energy minimization problem. Given a current guess for the parameters of the grid and its deformation field we seek to optimize the following high level goals:

- Every maximum (dot) should be close to a grid point
- Every minimum (gap) should be close to a grid cell center
- The grid deformations should be small and smooth

We define a regular (undeformed) grid by an anchor point $\vec{a}$ and a spanning vector $\vec{s}$. A grid point $\vec{p}$ with grid coordinates (x,y) is then given by:

$$\vec{p} = \vec{a} + x\vec{s} + y\vec{s}^{\perp}, \tag{2}$$

where $\vec{s}^{\perp}$ is perpendicular to $\vec{s}$. We model the deformation field as a 2D offset $\vec{d}$ at every N-th dot in each dimension ($N = 50$ in our implementation), with bilinearly interpolated offsets in between. Therefore, (2) becomes

$$\vec{p} = \vec{a} + x\vec{s} + y\vec{s}^{\perp} + \sum_k w_k \vec{d}_k, \tag{3}$$

where w_k are the bilinear coefficients, at most four of which are non-zero for a given point (x,y).

We can now construct a linear system with the goals stated earlier expressed as soft constraints. For this, we assume that we know the grid coordinates (x_i, y_i) for every extremum i. Maxima (dots) have integer grid coordinates and minima (gaps) have half-integer coordinates. For every extremum we generate one linear equation of the form given in (3), with $\vec{a}$, $\vec{s}$, and $\vec{d}_k$ as the unknowns.

Next, we add an equation $\vec{d}_k = 0$ for every deformation grid point k within the image domain. To enforce smooth deformations we also add an equation $\vec{d}_k - \vec{d}_l = 0$ for every pair of neighboring deformation points k and l. This results in a rectangular (overdetermined) sparse linear system $A f = b$ that can be solved using standard numerical methods. In our implementation we solve the associated normal equation $A^T A f = A^T b$ using Cholesky decomposition.

Initialization: In order to find good initial values for $\vec{a}$ and $\vec{s}$, we find the maxima with the most "regular" neighborhood, i.e. the four closest other maxima/minima are at the same distance and form 90° angles. We set $\vec{a}$ to the most regular maximum, and $\vec{s}$ to the average distance to the neighboring maxima.

We should note that grid point assignment is extremely sensitive to the accuracy of the spanning vector $\vec{s}$. Initially, extrema far away from the anchor are likely to receive wrong coordinates. We avoid this issue by solving the problem in an iterative fashion, where at each iteration we only consider extrema within some limited radius around the anchor. Each iteration slightly improves the accuracy of the anchor, the spanning vector and the deformation grid. In our implementation we start with a radius of 25 pixels and increase it in steps of 50 until the entire image is covered (Figure 4c–d). The supplementary material contains a quantitative evaluation of the described grid detection algorithm.

Pseudocode: The grid fitting process is summarized below:

Input: initial $\vec{a}$ and $\vec{s}$ estimates; all pruned extrema
radius $\leftarrow 25$, $\vec{d}_k \leftarrow 0, \forall k$
repeat
  For each maximum (minimum) find closest grid point with integer (half-integer) coordinates (x,y), given the current $\vec{a}, \vec{s}, \vec{d}_k$ estimates
  Construct matrix A and solve for new $\vec{a}, \vec{s}, \vec{d}_k$
  radius $\leftarrow$ *radius* $+ 50$
until all extrema are inside *radius*

6 Descreening and black ink recovery

This section describes the core of our method: explicit and precise extraction of the black ink layer, simultaneously with the reconstruction of the continuous CMY color channels. This is done by using a detailed (forward) model of the AM screening process and then estimating the parameters of this model so as to match the input image. Our model comprises two parts:

1. Dot model. We use a radially symmetric model with 6 degrees of freedom to fit the shape of the dot at each grid location. Thus, a complete parametric model is obtained for each of the recovered CMY separations.

2. Black mask. A binary mask defined over the entire image specifies for each pixel whether it is covered by black ink.

We regularize the model in several ways enforcing basic constraints of the printing process. Specifically, the degrees of freedom of the dot profiles are constrained to produce non-negative monotonic curves, and the black ink layer has a smoothness term.

We alternate between black ink assignment (Section 6.1) and dot shape estimation (Section 6.2). Pixels classified as black are excluded from consideration in the dot profile estimation. This leads to accurately recovering even the shapes of dots that are partly covered by black ink. This is important because it enables recovering correct colors near black lines (pockets of color surrounded by black are particularly difficult for other algorithms). We initialize the black ink mask by conservatively detecting sufficiently large clusters of dark pixels (either disk-shaped or line-shaped). An example is shown in Figure 5.

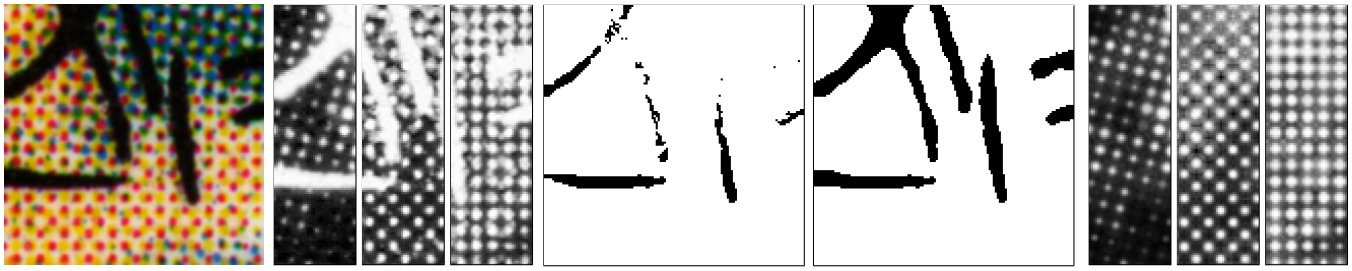


Figure 5: *Descreening and black ink recovery: given the separation of the input into CMY channels, we conservatively initialize the black ink mask. Alternating between black ink assignment and dot shape estimation, we converge to the final black ink mask and CMY dot grids. See the supplementary materials for additional visualizations.*

6.1 Black ink assignment

At each stage of the iterative optimization process outlined above we recompute the binary black ink mask using binary-label MRF optimization. The MRF is defined on a graph with the set of pixels P as nodes and all pairs of adjacent pixels on a 4-connected grid denoted by N^{pix} as edges. We solve the problem

$$\underset{\{\beta_i\}}{\operatorname{argmin}} \sum_{i \in P} E_i^{\text{k}}(\beta_i) + \lambda^{\text{sm}} \sum_{(i,j) \in N^{\text{pix}}} E_{ij}^{\text{k}}(\beta_i, \beta_j), \tag{4}$$

where β_i is a binary variable, whose value is 1 for a black pixel and 0 otherwise. The unary term E_i^{k}, defined below, measures the penalty of assigning the label β_i to pixel i, while the pairwise term E_{ij}^{k} encourages smoothness by penalizing assigning different labels to neighboring pixel: $E_{ij}^{\text{k}}(\beta_i, \beta_j)$ is 1 when $\beta_i = \beta_j$ and 0 otherwise. $\lambda^{\text{sm}} = 0.3$ is a regularization parameter.

The unary term takes the following form:

$$\begin{aligned} E_i^{\text{k}}(\beta_i) \;=\; & (1-\beta_i) \sum_{k \in \{\text{r,b,g}\}} \max\left(0, \tilde{c}_i^k - c_i^k\right) + \\ & \beta_i \sum_{k \in \{\text{r,b,g}\}} \max\left(0, c_i^k - c^{\text{black}}\right), \end{aligned} \tag{5}$$

where $\tilde{c}_i$ is the RGB color predicted by current dot model (computed using Eq. 1), while c_i is the scanned color. The first term penalizes color assignment for pixels where the scanned color is darker than what the dot model predicts. The second term penalizes assigning black to bright pixels. Since in some scans the black ink can take on fairly high values we subtract a baseline value c^{black}. We set it to the darkest channel value found within the initially detected black clusters mentioned earlier.

As shown by Kolmogorov and Zabih [2004], our MRF energy function (4) is *graph-representable*, meaning that its exact global minimum may be found by computing the minimum *s-t* cut in the corresponding graph, which we do using the well-known algorithm of Boykov *et al.* [2001].

6.2 Dot shape estimation

As mentioned earlier, our goal here is to fit a parametric model that describes the shape of the dot at each grid location. This is done independently for each of the CMY channels using the separations computed as described in Section 4.2.

The actual dot shapes we encountered in scanned prints are rather varied, and we were unable to obtain good results using a simple parametric model, such as a disk or a Gaussian. After some experimentation we chose modeling the dots as radially symmetric, with the profile consisting of two C^1 connected cubic Hermite splines. This model is able to approximate the observed dot shapes well using a total of six degrees of freedom.

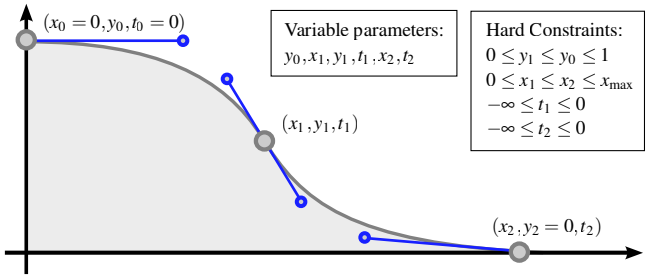


Figure 6: *Our model for the profile of a radially symmetric dot.*

As shown in Figure 6, each of the three control points has three parameters specifying its position and tangent. Some of these are locked leaving six free parameters. These parameters are further subject to hard constraints (see Figure 6) to enforce physically realizable dot shapes.

Our goal is now to recover the parameters for all dots such that each of the separated CMY channels is reproduced as accurately as possible, subject to a number of regularization constraints. Specifically, for each channel we minimize the following energy function:

$$E^{\text{dots}} = E^{\text{fit}} + \lambda^{\text{shape}} E^{\text{shape}} + \lambda^{\text{reg}} E^{\text{reg}} \tag{6}$$

The term E^{fit} captures how well we fit the pixels that are not covered by black ink:

$$E^{\text{fit}} = \sum_{i \in P} (1-\beta_i) \left\| s_i - \min\left(1, \sum_j f_j\left(d_{ij}\right)\right) \right\| \tag{7}$$

Here s_i is the value of the separation at pixel i and f_j is the profile function of dot j. d_{ij} is the distance between pixel i and the center of dot j. Contributions of overlapping dots saturate to one to keep things physically meaningful. The next term softly enforces physical plausibility:

$$\begin{aligned} E^{\text{shape}} = & \textstyle\sum_{k \in D} \max\left(0, \min_t f_k(t)\right)^2 + \\ & \textstyle\sum_{k \in D} \max\left(0, \max_t f_k(t) - 1\right)^2 + \\ & \textstyle\sum_{k \in D} \max\left(0, \max_t \frac{\mathrm{d}f_k}{\mathrm{d}t}(t)\right)^2, \end{aligned} \tag{8}$$

where D is the set of dots. The subterms enforce positivity, being smaller than one, and having a negative derivative, respectively. Note that it is not enough to enforce only the locations of control points using hard constraints, since the curve can take on negative values or positive tangents between the control points. However, simple analytical expressions exist for each of the constraints in Eq. 8. We found that making these constraints hard hurts the convergence of the optimization greatly and often gets points stuck behind a high energy barrier which prevents reaching a low energy configuration. Therefore, they are incorporated as soft constraints.

Finally, the E^{reg} term encourages spatial smoothness, which is particularly important for recovering dots that are covered by black ink:

$$E^{\text{reg}} = \sum_{(i,j)\in N^{\text{dots}}} \int_0^{x_{max}} \left\| f_i(t) - f_j(t) \right\| \mathrm{d}t, \tag{9}$$

where N^{dots} is the set of neighboring dot pairs. We set the balancing parameters in Eq. 6 to $\lambda^{\text{shape}} = 100$ and $\lambda^{\text{reg}} = 1$.

In summary, Eq. 6 gives rise to a non-linear system with six variables per dot. However, every dot is only coupled with its direct neighbors. Thus, instead of optimizing all of the dots at once, we solve a collection of small optimization problems: in each problem all the variables except those of a single dot are fixed. Since dots that are more than two grid locations away from each other have no interaction whatsoever, this computation can be trivially parallelized on multi-core machines.

To solve the system we use the BFGS algorithm [Nocedal and Wright 2000], as implemented in the GNU Scientific Library[1]. The hard constraints are enforced using logarithmic barrier functions [Nocedal and Wright 2000]. Even though the error surface might contain local minima, we have not observed problems with the process getting stuck in practice. In the accompanying video and the supplementary materials we demonstrate convergence from multiple starting points.

We generated all our results using 5 iterations of interleaved black ink recovery and dot shape optimization. Within each dot optimization phase, we iterate 10 times over all the dots, optimizing each dot individually.

7 Vectorization and Rendering

The result of the previous section is a binary black ink mask, and three angled grids of parameterized dot profiles.

To vectorize the binary black ink mask, we first convert the image into a 2D mesh by generating two triangles for each pixel. The triangles are augmented with an attribute that specifies whether it came from a black or non-black pixel. We simplify the mesh using the progressive meshes algorithm [Hoppe 1996] enforcing a maximum residual of one pixel. We prevent vertices on the mesh boundary from being collapsed to preserve the rectangular shape of the panel. Furthermore, we prevent any topology changes with respect to the black / non-black regions. Finally, we improve the smoothness of the obtained mesh by replacing edges between black / non-black regions with cubic spline curves. The resulting vector representation is resolution independent and can be rendered using any standard graphics package.

Although we recover accurate dot shapes in our descreening process, our eventual goal is to remove the screening pattern and render a continuous color image. We compute the magnitude of every dot by integrating its profile function. Depending on the quality of the scanned print, the result may still contain some residual noise, which we reduce by applying bilateral filtering to the dot magnitudes. A continuous CMY image can now be reconstructed by simply interpolating the computed dot magnitudes. Finally, this image is converted to RGB using the forward Neugenbauer equation.

The reconstructed color image may be displayed overlaid with the reconstructed black artwork. This results in a smooth color illustration look, which works well for certain kinds of comic books, such as the North American comics that often use piecewise constant colors or smooth color gradients. On the other hand, some of the Franco-Belgian comics we experimented with are more intricately and masterfully colored with watercolor. In this case, we found that applying a non-photorealistic painterly filter to the reconstructed image yields a more compelling result that is better suited for magnified viewing, which is one of our goals. To apply the painterly rendering filter we used the built-in Auto-Paint feature of Corel Painter 12. The filter is applied separately to the RGB color image and to the black artwork, which is then composited on top of the color image. This technique was used to generate the right image in Figure 1, as well as the top left result in Figure 10.

[1] http://www.gnu.org/software/gsl

8 Results

In order to validate and stress-test our method we scanned an original hand-drawn illustration which was then digitally color separated and screened. The screenings were subjected to a variety of degradations, such as the addition of various amounts of Gaussian noise, raising the grayscale level of the black channel, and applying a geometric distortion to the CMY screens. We applied our method to each of these inputs and measured the accuracy of the recovered primaries, dot grids, and the recovered CMY and K channels. Figure 7 demonstrates the robustness and graceful degradation of our method's accuracy. The full set of results is included in the supplementary materials.

We also applied our method to a large number of scanned pages from a variety of comic books, including North American comic books: Batman (by DC Comics), Iron Man (by Marvel Comics), and Bone, as well as several famous Franco-Belgian comics: Tintin, Jeremiah, and Thorgal. Due to space limitations we can only show a few crops in the paper (see Figure 10) in order to reveal the intricate details that are recovered by our algorithm. In Figure 2 we showed a comparison to alternative methods for removing the screening pattern. In the supplementary materials we include a more extensive comparison to a wider range of methods, and we also show several additional full panel results.

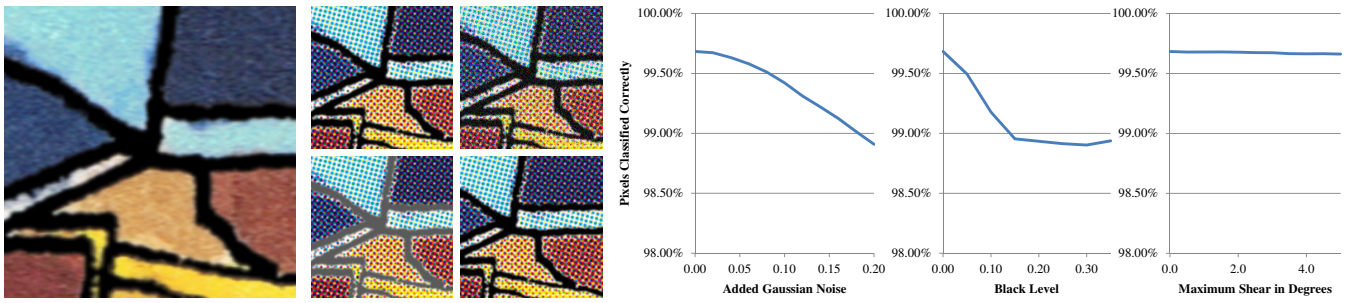
Our technique is applicable not only to comic books, but also to other types of color illustrations that are printed using a non-screened black ink layer. Figure 10 shows an example of a technical illustration from a psychology textbook, and a color illustration from an old children's book is included in the supplementary materials.

Our optimization needs the input scans to be of sufficient resolution such that the dots become clearly distinguishable. For all inputs we have tried a scan resolution of 1200dpi was sufficient. The dot spacing is around 6–9 pixels at this resolution. Our current implementation is not highly optimized; descreening a panel scanned at 3545×2010 pixels, for example, takes about 10-12 minutes on a 2.5 GHz Intel Xeon CPU. However, the long runtime is compensated by the high fidelity crisp reconstruction of the non-screened black ink channel. Since this channel usually contains much finer detail than the color channels and we are able to recover it faithfully, our vectorized results are suitable even for significant magnifications (Figure 10).

Compression: Our method recovers all the details present in the input in parametric form, e.g. the shape of every dot and every black line. Since the grid of screening dots is much sparser than the input pixel grid our representation can be highly compressed.

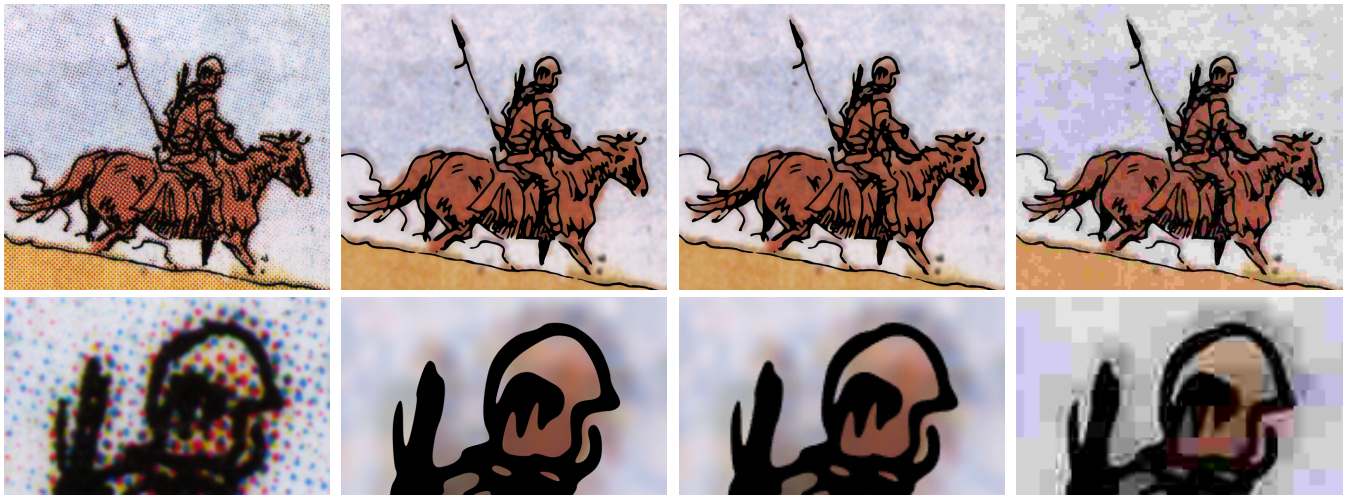
For rendering images we do not need the exact dot shapes, but require only their magnitudes (Section 7). We store these magnitudes as three grayscale images, one for each CMY channel. We quantize these images to 8 bits and use standard image coding techniques to compress them. Using the JPEG XR image coder, each of these images compresses to around 35–50 kb for a typical comic book panel (we achieve high compression since these images are typically very smooth). We store the control points for the black ink splines in a binary file which takes around 100–120k.

Our representation decompresses almost instantaneously and enables high fidelity rendering. In Figure 8 we compare to standard JPEG compression tuned to either similar quality or similar file size. The former yields a much larger file size compared to our representation, while the latter exhibits significant compression artifacts.



Ground truth | Degraded screenings | Percentage of pixels correctly classified as black/non-black

Figure 7: *Validation and stress testing our method with ground truth. A hand-drawn illustration was scanned and digitally screened simulating a variety of degradation conditions: adding noise, raising the black level, and introducing geometric distortions. The plots on the right show that our black ink recovery is robust and degrades gracefully. See the supplementary materials for a more exhaustive analysis.*



Input, 16-bit uncompressed TIF (102,023.6 kb) | Our Representation (230.1 kb) | High Quality JPEG (4,598 kb) | Low Quality JPEG (237.8 kb)

Figure 8: *Our representation is highly compressible without visible degradation. We compare to JPEG encoding tuned to either match our quality or file size. Input image © Rosinski – Van Hamme – Editions Du Lombard (Dargaud-Lombard s.a.)*

Editing: Our representation lends itself for creative manipulations of digitized comics art. Since the black channel is binary exhibiting crisp contours and the recovered colors are very smooth, region selection, which is the foundation for many editing tasks, becomes easy.

As *proof of concept* we implemented a scribble-based interface to drive a simple multi-label graph cut segmentation. The obtained segmentation requires only minor manual touch-ups since the recovered boundaries perfectly trace the strong black/color edges. We then moved the segments onto separate image planes where we completed them using Adobe Photoshop's image completion tools. The entire process took less than half an hour.

We assigned each layer a certain depth and implemented an interactive pan-and-zoom interface where we render the layers with parallax. Figure 9 illustrates this process and shows a novel view that was generated in this manner. Please refer to the supplementary material for a video capture from an interactive session.

Limitations: We have tried our method on about 100 different inputs and did not encounter any "catastrophic" failures. However, one limitation of our method is that it sometimes misses very faint black features, such as small dots or very thin lines. In this case the algorithm will compensate by adjusting the dots to recover the feature and it will appear more blurry in the result. This artifact can be seen when carefully comparing our results with the input (e.g. using the interactive viewer in the supplementary material). Also, on rare occasions our detected grids are locally not well aligned with the true grid, in which case the color and black reconstruction suffers.

Another problem (that we share with other methods) is that we rely on inputs to be printed on relatively high quality paper. In some cheap paperback books we experimented with, the printing on the back side of the page showed through significantly and ended up being reconstructed faintly. However, the bilateral denoising of the color channels (Section 7) helps reduce such artifacts. Another issue with low grade paper is that black lines appear less sharp and clean, sometimes leading to rougher looking reconstructions.

9 Conclusion

Over the last few decades, nearly all media (e.g., audio, images, video, text, CAD) have been transitioning to digital representations. There are of course countless works that predate this transition. One key challenge is that analog representations in physical media may be subject not only to noise but to other artifacts specific to each physical representation. In this work we considered the challenging case of printed color illustrations, and in particular, comics art. We have described a new tool that enables such illustrations to be digitized easily yet faithfully, resulting in a compact, rescalable representation.

In the future we plan to explore the editing possibilities that our work opens up. A particularly exciting direction are animations,

Input Layer Mask Inpainted Layers Novel View

Figure 9: *Proof of concept editing application. The layer mask image shows the graph cut based segmentation result. Input image © Rosinski – Van Hamme – Editions Du Lombard (Dargaud-Lombard s.a.)*

e.g., moving characters or ambient motions for trees or clouds. We also plan to look into semantic analysis of comic panels, which is much easier once we accurately segmented the lines, which are the main structural elements in this type of art.

Acknowledgements: The authors are indebted to Michel Kichka for providing invaluable explanations of the comics production process. This work was supported in part by the Israel Science Foundation founded by the Israel Academy of Sciences and Humanities.

References

ABEL, J., AND MADDEN, M. 2008. *Drawing Words, Writing Pictures: making comics from manga to graphic novels*. First Second, New York, NY.

BAYER, B. E. 1973. An optimum method for two-level rendition of continuous-tone pictures. *IEEE Intl. Conf. on Communications 1*, 2611–2615.

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

CHASTAIN, S., 2012. How to remove Moire patterns from scanned photos in Photoshop and Elements. http://graphicssoft.about.com/cs/photoshop/ht/apsremovemoire.htm, January.

DIAMOND COMIC DISTRIBUTORS, 2012. Publisher market shares: April 2010. http://www.diamondcomics.com/Home/1/1/3/237?articleID=94940, April.

FARBMAN, Z., FATTAL, R., LISCHINSKI, D., AND SZELISKI, R. 2008. Edge-preserving decompositions for multi-scale tone and detail manipulation. *ACM Trans. Graph. 27*, 3, Article 67.

FLOYD, R. W., AND STEINBERG, L. 1976. An adaptive algorithm for spatial grey scale. *Proc. Soc. Inf. Display 17*, 75–77.

GIMP, 2011. GNU Image Manipulation Program. http://www.gimp.org.

GRAIS, S., 2012. Comics color. http://facweb.cs.depaul.edu/sgrais/comics_color.htm, April.

HOPPE, H. 1996. Progressive meshes. *SIGGRAPH96*, 99–108.

JAIMES, A., MINTZER, F. C., RAO, A. R., AND THOMPSON, G. 1998. Segmentation and automatic descreening of scanned documents. *Proc. SPIE 3648*, 517–528.

KIPPHAN, H. 2001. *Handbook of Print Media: Technologies and Production Methods*. Springer-Verlag.

KITE, T. D., VENKATA, N. D., EVANS, B. L., AND BOVIK, A. C. 2000. A fast, high-quality inverse halftoning algorithm for error diffused halftones. *IEEE Trans. Image Proc. 9*, 1583–1592.

KOLMOGOROV, V., AND ZABIN, R. 2004. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell. 26*, 2 (Feb.), 147–159.

LIU, X. 1996. *Analysis and Reduction of Moire Patterns in Scanned Halftone Pictures*. PhD thesis, Virginia Polytechnic Institute and State University.

MAHY, M., AND DELABASTITA, P. 1996. Inversion of the Neugebauer equations. *Color Research & Application 21*, 6, 404–411.

NEELAMANI, R. N., NOWAK, R. D., AND BARANIUK, R. G. 2000. Model-based inverse halftoning with wavelet-vaguelette deconvolution. *Proc. IEEE ICIP*, Vol III: 973–976.

NEUGEBAUER, H. E. J. 1937. Die theoretischen grundlagen des mehrfarbenbuchdrucks. *Zeitschrift für wissenschaftliche Photographie Photophysik und Photochemie 36*, 4, 73–89.

NOCEDAL, J., AND WRIGHT, S. J. 2000. *Numerical Optimization*. Springer.

PARK, M., BROCKLEHURST, K., COLLINS, R., AND LIU, Y. 2009. Deformed lattice detection in real-world images using mean-shift belief propagation. *IEEE Trans. PAMI 31*, 10 (Oct.).

SATTVA, 2011. Descreen 5.0 plug-in for Adobe Photoshop. http://www.descreen.net/eng/soft/descreen/descreen.htm.

SIDDIQUI, H., AND BOUMAN, C. A. 2007. Training-based descreening. *IEEE Trans. Image Proc. 16*, 3, 789 –802.

SIDDIQUI, H., BOUTIN, M., AND BOUMAN, C. 2010. Hardware-friendly descreening. *IEEE Trans. Image Proc. 19*, 3, 746 –757.

SMITH, S. M., AND BRADY, J. M. 1997. SUSAN — a new approach to low level image processing. *Int. J. Comput. Vision 23*, 45–78.

STANGER, C. J., TRAN, T., AND BARNEY SMITH, E. H. 2011. Descreening of color halftone images in the frequency domain. *Proc. SPIE 7866, 78661H*.

STEVENSON, R. 1997. Inverse halftoning via MAP estimation. *IEEE Trans. Image Proc. 6*, 4, 574–583.

STOFFEL, J., AND MORELAND, J. 1981. A survey of electronic techniques for pictorial image reproduction. *IEEE Trans. Commun. 29*, 12, 1898–1925.

SUBR, K., SOLER, C., AND DURAND, F. 2009. Edge-preserving multiscale image decomposition based on local extrema. *ACM Trans. Graph. 28*, 5, Article 147.

WEINER, R. G., Ed. 2010. *Graphic Novels and Comics in Libraries and Archives*. McFarland & Company, Inc., Publishers, Jefferson, North Carolina.

WONG, P. W. 1995. Inverse halftoning and kernel estimation for error diffusion. *IEEE Trans. Image Proc. 4*, 4, 486–498.

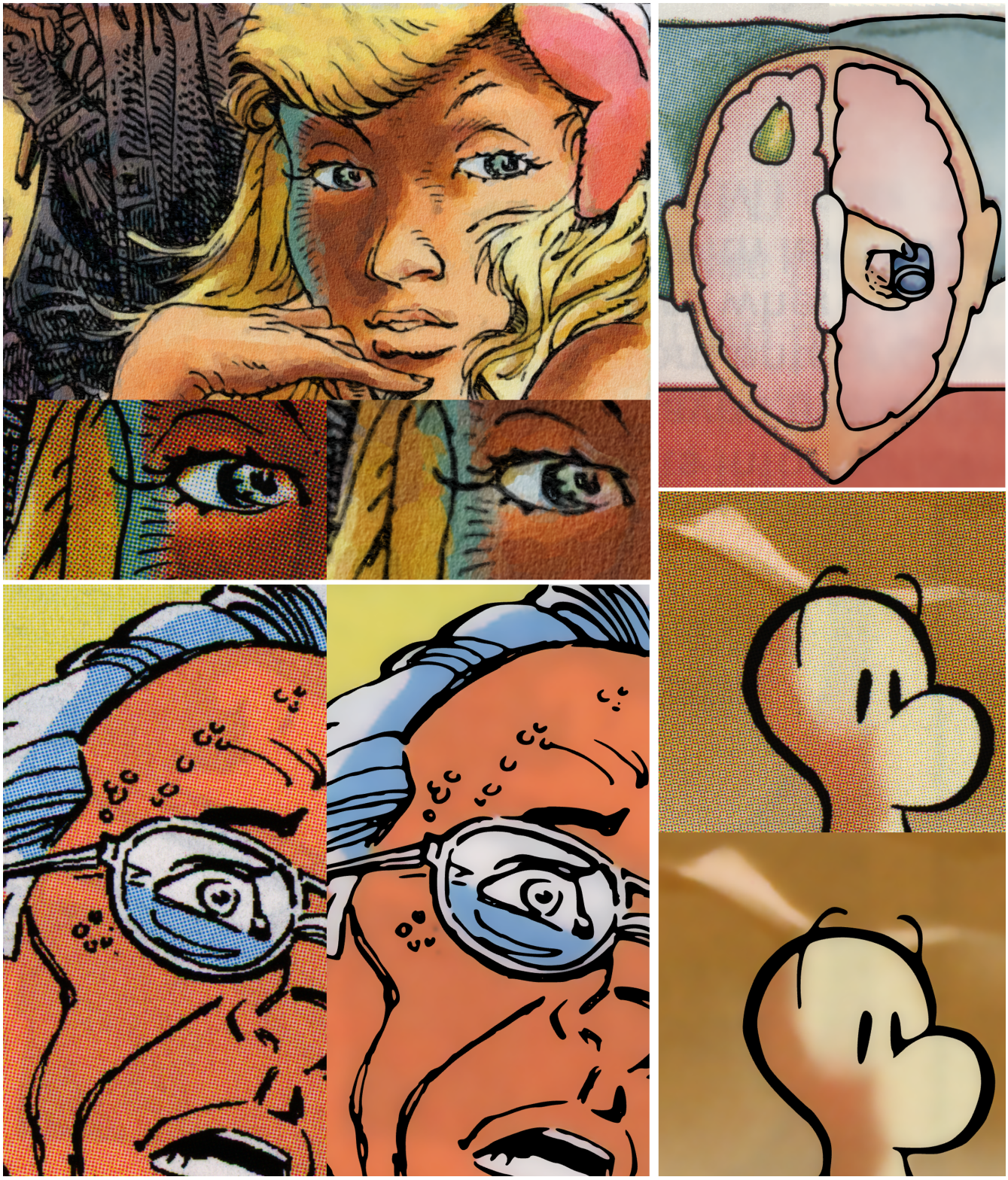


Figure 10: *A few sample crops from our many results (please zoom in to examine detail). Top left: a crop from Jeremiah rendered using an NPR filter (input © Dupuis). The quality of our reconstruction enables significant magnification (the top part is about* 4 *times larger than the input in the original scan, and the zoomed view below has even higher magnification). Top right: a crop from a psychology textbook diagram (input © Philip G. Zimbardo): the right half of the brain is halftoned, while the left is continuous. Bottom left: a crop from Iron Man (input © Marvel Comics). Bottom right: a crop from Bone (input © Jeff Smith).*