# Separation of Manga Line Drawings and Screentones

Kota Ito[1], Yusuke Matsui[1], Toshihiko Yamasaki[1] and Kiyoharu Aizawa[1]

[1]The Universitiy of Tokyo, Japan

**Abstract**

*Screentones are unique expressions of Japanese comics (manga), which enrich their visual expression. However, such screentones have a very different visual nature from that of line drawing areas; this prevents us from applying various kinds of image processing techniques to manga. We propose a method for extracting line drawings and removing screentones. We employ Laplacians of Gaussian filters and flow-based differences of Gaussian filters, one for removing screentones and the other for preserving lines, and make a binary mask for separating line drawings from manga by merging the results of the two filters. We show that the proposed method successfully separates line drawings and is better than existing methods in comparative studies.*

Categories and Subject Descriptors (according to ACM CCS): I.4.6 [Image Processing and Computer Vision]: Segmentation—Edge and feature detection

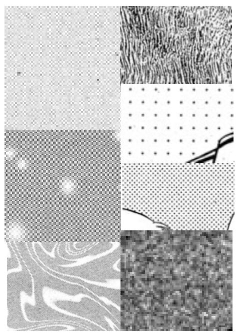
## 1. Introduction

Manga (Japanese comics) are popular all over the world. A unique trait of manga is their use of *screentones*, which are preprinted patterns that express visual effects such as textures and shadows (Fig. 1(a)). Manga authors draw manga by drawing lines and filling areas with screentones (Fig. 1(b)).

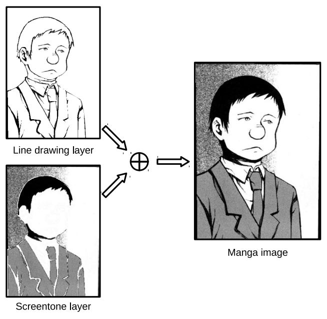
Currently, comic books are increasingly being published in electronic form as well as in print, and research on new applications of manga image processing has appeared, e.g., manga retrieval [MAJ14], interactive segmentation [AMYA14], and element composition for drawing assistance [CLC14].

However, it is not clear how to handle screentones in digital images. The nature of screentones is very different from that of line drawings, which might cause undesired effects if we do not take the screentones into consideration when manga images are processed. Therefore, we want to separate screentones and line drawings in advance before the main processing is applied, to make full use of the visual structure of the manga image. Further, the separation enables us to apply image processing methods developed for line drawings.

We propose a method for extracting line drawings and removing screentones. The method consists of applying several filters and merging the results. Because screentones have a wide variety of statistical natures, including large/small, isotropic/anisotropic, and low/high frequencies of textures



(a) Examples of various kinds of screentones.



(b) A manga image consists of a line drawing layer and a screentone layer.

**Figure 1:** *Manga components. Note that a black-filled area is considered to be a screentone in this paper. ©Junichiro Akabi*

(Fig. 1(a)), it is hard to extract all kinds of screentones with a single filter. To handle them, we create two masks: (1) a *screentone removal mask*, which deletes all screentone areas, and (2) a *line preserving mask*, which preserves lines as much as possible. By appropriate merging of the results from these two masks, we compute the final mask, which specifies line drawings. Our method does not require any parameter tuning and is very robust to the kinds of screen-

DOI: 10.2312/egsh.20151018

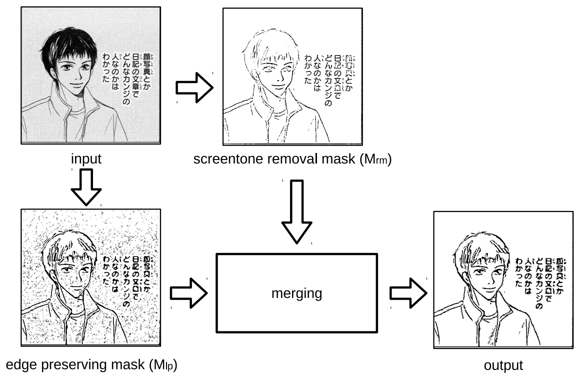


**Figure 2:** *Work flow.* ©*Hotaru Yoshizawa*

tones, whereas previous line extracting methods for manga are suitable only for a single kind of texture, e.g., clean filled areas [HYLW10], or simple dot patterns [KL12]. Note that Qu and colleagues proposed a robust extraction method for screentones [QWH06], but it requires manual interactions.

## 2. Method

We show the flow of our method in Fig. 2. First, we create a screentone removal mask $M_{rm}$, which removes screentone areas as much as possible, and a line preserving mask $M_{lp}$, which retains as many lines as possible. Then, we combine these masks, taking account of connected components.

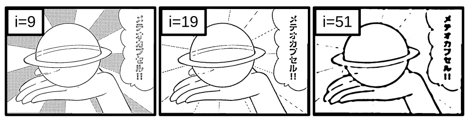
### 2.1. Screentone removal mask

The purpose of a screentone removal mask is to delete all screentone areas while allowing the removal of some portion of line drawings. To create a screentone removal mask, we employ a Laplacian of Gaussian (LoG) filter, which acts as a bandpass filter in the Fourier domain, and works as a line detector. Because screentones tend to have periodicity or high-frequency components, they can be removed by a LoG filter while preserving line drawings. We found two traits of outputs of LoG filtering: (1) the LoG value is relatively high at line drawings, and low in other areas; (2) a pixel that has a negative LoG value hardly ever belongs to a line drawing. By considering these observations, we create an LoG mask $M_{LoG_i}$ as shown below:

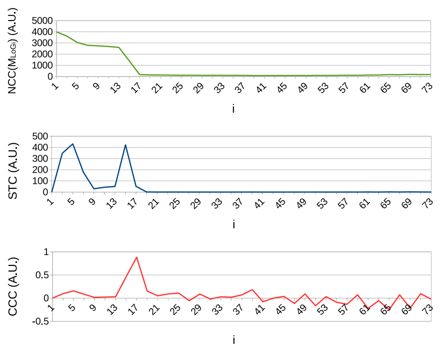
$$\hat{I}_{LoG_i}(\mathbf{x}) = \begin{cases} 0 & \text{if } I_{LoG_i}(\mathbf{x}) < 0 \\ I_{LoG_i}(\mathbf{x}) & \text{otherwise,} \end{cases} \tag{1}$$

$$M_{LoG_i} = Bin(\hat{I}_{LoG_i}), \tag{2}$$

where $I_{LoG_i}$ is the output of a LoG filtering of an input image $I$, and $i$ is the window size of a Gaussian filter. $i$ must be odd. Note that $i$ automatically determines a standard deviation parameter of the filter. $\hat{I}_{LoG_i}$ is an image in which each pixel $\hat{I}_{LoG_i}(\mathbf{x})$ is $I_{LoG_i}(\mathbf{x})$ if $I_{LoG_i}(\mathbf{x})$ is positive and 0 otherwise.



(a)



(b)

**Figure 3:** *Effect of LoG parameter: (a) LoG masks with various values of i* ©*Takuji. (b) Change of criteria with i: top: $NCC(M_{LoG_i})$; middle: STC, and bottom: CCC.*

$Bin(\hat{I}_{LoG_i})$ is a binary of $\hat{I}_{LoG_i}$ by Otsu's method [Ots75]. We omit a description of input $\mathbf{x}$ for readability.

The output of the LoG filter changes with $i$ as shown in Fig. 3(a). We must select an appropriate value for $i$ to remove screentones; $i = 19$ is the best in this case. To select $i$, we introduce two criteria, the *Connected Component Criteria (CCC)* and the *Stop Criteria (STC)*, which are defined as follows:

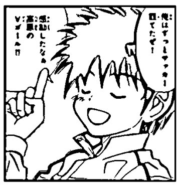
$$CCC_i = 1 - \frac{NCC(M_{LoG_i})}{NCC(M_{LoG_{i-2}})}, \tag{3}$$

$$STC_i = \frac{NCC(M_{LoG_i})}{NCC(M_{LoG_1})} \times |(NCC(M_{LoG_{i-2}}) - NCC(M_{LoG_i}))|, \tag{4}$$

where $NCC(M)$ is the number of connected components of black pixels in a mask $M$. $CCC_i$ is a ratio showing the decrease in the number of connected components when $i$ is incremented. If $CCC_i$ has its highest value at $i^*$, it means the most screentones are removed at $i^*$. We empirically found that $i$ values a little larger than $i^*$ are best for our purpose. If we increment $i$ much more, the lines become too blurred. Examples are shown in Fig. 3(a) and Fig. 3(b).

$STC_i$ is used for the stopping condition of the process. If $STC_i$ is smaller than a predefined threshold, the process stops. $STC_i$ denotes the amount of change of $NCC(M_{LoG_i})$ (compared with the ratio between $NCC(M_{LoG_1})$ and $NCC(M_{LoG_i})$, i.e., $NCC(M_{LoG_i})/NCC(M_{LoG_1})$), as shown in Fig. 3(b).

We show an algorithm for creating a removal mask $M_{rm}$ in


(a) Result without a base mask (bold).

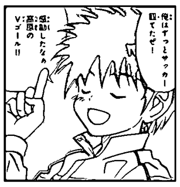

(b) Result with a base mask (thin).

**Figure 4:** *The effect of a base mask $M_{LoG_{i_{base}}}$.* ©*Motoi Takenaka*

Alg. 1. $\alpha$ is a threshold of $STC$, and set to 1. $\beta$ is used to control a search range. If $\beta = 1$, $i$ of max $CCC_i$ is simply found. Empirically, we found larger $i$ is better if there are several similar peaks of $CCC_i$. We therefore select a relaxation factor $\beta$, i.e., if there are similar peaks, we select the last one. $\beta$ is set as 0.8. By checking $CCC_i$ and $STC_i$ in the while loop, the best parameter $i_{LoG}$ is found. This procedure works well except when there are black-filled screentones and $i_{LoG}$ is too large. In such cases, lines around the black-filled areas tend to be bold as shown in Fig. 4(a). To handle such exceptions, we set $i_{base}$, which produces a relatively "thin" border mask (line 12), and take a pixel-wise logical AND of the two masks, computed from $i_{LoG}$ and $i_{base}$ (line 13). An example of the effect of $i_{base}$ is shown in Fig. 4(b).

**Algorithm 1** Create a removal mask.

```
 1: Input: $I$
 2: Output: $i_{base}$, $M_{rm}$
 3: $i \leftarrow 3,\ CCC_{max} \leftarrow 0,\ i_{LoG} \leftarrow 1$
 4: while $STC_i > \alpha$ do
 5:     if $CCC_i \geq \beta \times CCC_{max}$ then
 6:         $i_{LoG} \leftarrow i$
 7:         $CCC_{max} \leftarrow \max(CCC_{max}, CCC_i)$
 8:     end if
 9:     $i \leftarrow i + 2$
10: end while
11: $i_{LoG} \leftarrow i_{LoG} + 4$
12: $i_{base} \leftarrow \min(\frac{i}{2}, i_{LoG})$
13: $M_{rm} \leftarrow M_{LoG_{i_{LoG}}} \wedge M_{LoG_{i_{base}}}$
```

### 2.2. Line preserving mask

The line preserving mask is intended to extract line drawings. This is a difficult task, so this mask cannot avoid the capture of small amounts of screentones. Such unnecessary screentones will be deleted by combining the result from $M_{rm}$ in the merging step.

To create a line preserving mask $M_{lp}$, we use a Flow-based Difference of Gaussian (FDoG) filter [KLC07], which was originally proposed for non-photorealistic rendering to create visually plausible line drawings. FDoG extracts a set of coherent isotropic lines. It is useful for manga because they have clear isotropic line drawings.

We must select two parameters to compute FDoG; a threshold of binarization $\tau$, which is set as 0.05, and a window size for filtering $w_{DoG}$. If $w_{DoG}$ is smaller than the line width, the FDoG filter does not work well. Therefore, we select $w_{DoG}$ automatically, by using $i_{base}$, which denotes a scale space of the input image and was computed when making the screentone removal mask. Experimentally we selected $w_{DoG}$ as $w_{DoG} = i_{base} - 2$.

### 2.3. Mask merging for refining line extraction

After generating the two masks, we restore the line drawings from the line preserving mask by using the screentone removal mask as a guide. The algorithm is shown in Alg. 2. First, we make a combined mask $M_{comb}$ by taking a pixel-wise logical OR between $M_{rm}$ and $M_{lp}$ (line 3). $R$ is a set of pixels that belong to $M_{rm}$. Then we separate $M_{comb}$ into connected components. $CC_n(M)$ in line 8 represents the $n$th connected component of $M$. We consider a connected component as a unit for the restoration process. Each $CC_n$ is stored in $L$ if the intersection of $CC_n$ and $R$ is not empty. Finally, we obtain the final mask $M_{final}$ from $L$.

**Algorithm 2** Combined masks.

```
 1: Input: $M_{rm}, M_{lp}$
 2: Output: $M_{final}$
 3: $M_{comb} \leftarrow M_{rm} \bigvee M_{lp}$
 4: $R \leftarrow \{\mathbf{x} \mid M_{rm}(\mathbf{x}) = 1\}$
 5: $L \leftarrow \emptyset$
 6: $N \leftarrow NCC(M_{comb})$
 7: for $n \leftarrow 1 \ldots N$ do
 8:     if $CC_n(M_{comb}) \cap R \neq \emptyset$ then
 9:         $L \leftarrow L \cup CC_n(M_{comb})$
10:     end if
11: end for
12: $M_{final}(\mathbf{x}) \leftarrow \begin{cases} 1 & \text{if } \mathbf{x} \in L \\ 0 & \text{otherwise} \end{cases}$
```

## 3. Experiment

To evaluate our method, we manually created ground truth masks from 13 manga images. These were chosen from six different titles so that they have different screentones. The evaluation value, *overlap ratio*, is defined as $r_{overlap} = \frac{|M_{final} \bigwedge M_{gt}|}{|M_{final} \bigvee M_{gt}|}$, where $M_{gt}$ represents the ground truth. This is a standard criterion for evaluation in an object detecting task [EEG*14]. We compared our method with a binarizing operation [Ots75], Canny edge, LoG filter, FDoG filter [KLC07], and an edge-enhanced isotropic nonlinear filter [HYLW10], which is a stroke detection method for clear-background cartoon images. The parameters of these methods were selected to maximize the average *overlap ratio*,

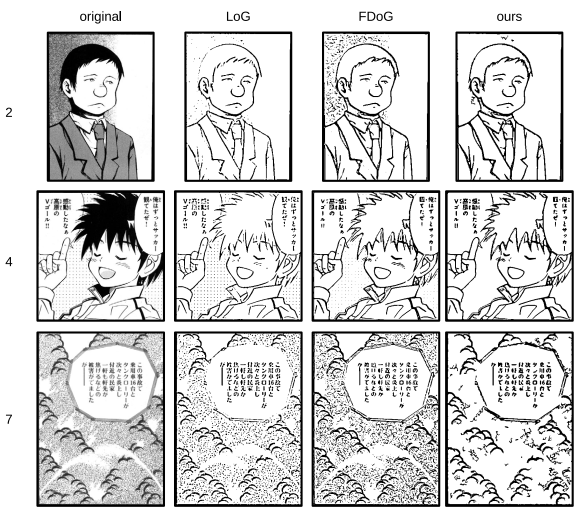


**Figure 5:** *Examples of comparative evaluation.*

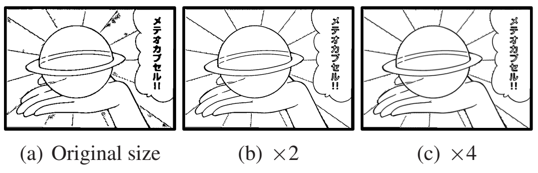


**Figure 6:** *An example of scale variation.* ©*Takuji*

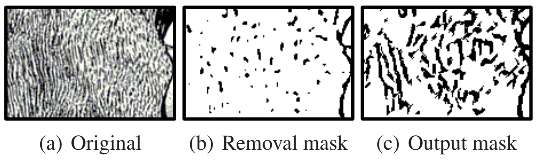


**Figure 7:** *An example of failure of merging.* ©*Tetsuya Kurosawa, Hidehisa Masaki*

and fixed for all images. The results are shown in Table 3 and Fig. 5. Our method outperforms all the other methods. Please see supplementary material for more details.

**Table 1:** *Results of the comparative studies: accuracy.*

| | Binarization | Canny | LoG |
|---|---|---|---|
| $r_o overlap$ | $0.41 \pm 0.15$ | $0.23 \pm 0.071$ | $0.47 \pm 0.068$ |
| | FDoG | [HYLW10] | Ours |
| $r_{overlap}$ | $0.51 \pm 0.081$ | $0.48 \pm 0.080$ | $\mathbf{0.56 \pm 0.073}$ |

LoG-based methods (LoG, [HYLW10]) and FDoG have different advantages. LoG-based methods can remove non-periodic or sparse screentones with large window sizes, but they cannot preserve lines in such cases. In contrast, the FDoG filter cannot remove such screentones, but it preserves line drawings strongly. Our method can make the best use of both advantages, and shows stable efficiency. In addition, because our method determines all parameters automatically, it is robust for both kinds of screentones and scale variation. An example of results with different scale images is shown in Fig. 6, where our method works well even when the scale of the image is changed. Our method is robust to the scale, especially for periodic screentones, but it cannot handle non-periodic screentones well. Please see the supplementary material for more examples. Note our run-time cost was 6.4 sec for $827 \times 1170$ images on average.

**Limitations:** If the screentone consists of dense long curves, the removal mask tends to fail to delete screentones (Fig. 7(b)). Because we consider connected components as a unit in the merging stage, our method restores pixels that do not exist in the removal mask (Fig. 7(c)).

## 4. Conclusion

We have presented a method for extracting line drawings from manga images. The algorithm is based on a combined LoG filter and FDoG filter. Experimental results show that our proposed method outperforms conventional approaches for 13 manga images.

**Acknowledgment**

We thank authors of manga for the use of their images. This work was partially supported by the Strategic Information and Communications R&D Promotion Programme (SCOPE) and JSPS.

**References**

[AMYA14] ARAMAKI Y., MATSUI Y., YAMASAKI T., AIZAWA K.: Interactive segmentation for manga. In *ACM SIGGRAPH 2014 Posters* (2014), ACM, p. 66. 1

[CLC14] CAO Y., LAU R. W. H., CHAN A. B.: Look over here: Attention-directing composition of manga elements. *ACM TOG 33*, 4 (2014), 1214–1220. 1

[EEG*14] EVERINGHAM M., ESLAMI S., GOOL L. V., WILLIAMS C. K., WINN J., ZISSERMAN A.: The pascal visual object classes challenge: A retrospective. *IJCV* (2014), 1–39. 3

[HYLW10] HUANG M., YANG M., LIU F., WU E.-H.: Stroke extraction in cartoon images using edge-enhanced isotropic nonlinear filter. In *Proc. VRCAI* (2010), ACM, pp. 33–38. 2, 3, 4

[KL12] KOPF J., LISCHINSKI D.: Digital reconstruction of halftoned color comics. *ACM TOG 31*, 6 (2012), 140. 2

[KLC07] KANG H., LEE S., CHUI C. K.: Coherent line drawing. In *Proc. Non-photorealistic Animation and Rendering* (2007), pp. 43–50. 3

[MAJ14] MATSUI Y., AIZAWA K., JING Y.: Sketch2manga: Sketch-based manga retrieval. In *ICIP* (2014), IEEE. 1

[Ots75] OTSU N.: A threshold selection method from gray-level histograms. *Automatica 11*, 285-296 (1975), 23–27. 2, 3

[QWH06] QU Y., WONG T.-T., HENG P.-A.: Manga colorization. *ACM TOG 25*, 3 (2006), 1214–1220. 2