A Robust Panel Extraction Method for Manga

Xufang Pang, Ying Cao, Rynson W.H. Lau, and Antoni B. Chan
Department of Computer Science, City University of Kong Kong
{pangxufang,caoying59}@gmail.com, {rynson.lau,abchan}@cityu.edu.hk

ABSTRACT

Automatically extracting frames/panels from digital comic pages is crucial for techniques that facilitate comic reading on mobile devices with limited display areas. However, automatic panel extraction for manga, i.e., Japanese comics, can be especially challenging, largely because of its complex panel layout design mixed with various visual symbols throughout the page. In this paper, we propose a robust method for automatically extracting panels from digital manga pages. Our method first extracts the panel block by closing open panels and identifying a page background mask. It then performs a recursive binary splitting to partition the panel block into a set of sub-blocks, where an optimal splitting line at each recursive level is determined adaptively. Finally, it recovers accurate panel shapes from the computed sub-blocks. Our experiments show that the proposed method can robustly segment panels on the manga pages with various styles.

Categories and Subject Descriptors

I.4 [**Computing Methodologies**]: Image Processing and Computer Vision

Keywords

Comics processing, computational manga, panel extraction, adaptive page partitioning

1. INTRODUCTION

Comics is one of most successful storytelling mediums. With rapid development of smartphones and tablets, reading digital comics on mobile devices is becoming increasingly popular. To improve the mobile comic reading experience, one fundamental technical challenge is to automatically partition each comic page into a set of panels, which are then displayed in correct order on the small screen of a mobile device [1]. Recently, manga, as a type of comics, has gained massive popularity and is consumed by a huge audience across the world. Manga is mainly characterized by its unique style in panel composition and layout (such as irregular and borderless panels) [2], as well as extensive use of complex visual symbols (such as sound verbs and background texture). Figure 1 illustrates the stylistic effects common in manga layouts, including: (a) irregular panels (i.e., non-rectangular panel shape), (b) fourth wall break effect (i.e., some part of the foreground object lying outside of the panel boundary), (c) joined panels (i.e., two panels connected by some foreground objects, balloons or other visual effects), (d) unclosed panels (i.e., missing part of the panel boundary), and (e) borderless panels (i.e., no explicit panel boundary). While being able to effectively enhance the expressiveness of manga, these stylistic effects pose unique challenges to the task of automatically extracting panels from manga pages.

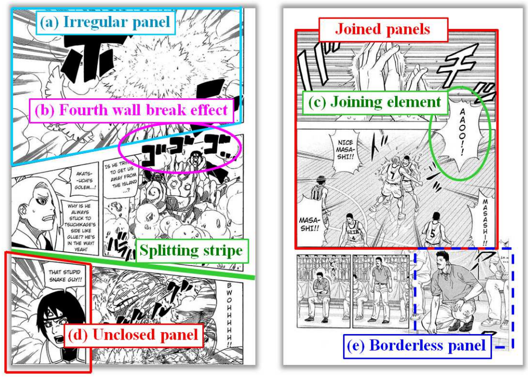

Figure 1: Some stylistic effects used in manga.

There are several works for automatic extraction of panels from comic pages, but none of them is capable of dealing with all the aforementioned stylistic effects. Some of the methods detect division lines between adjacent panels using recursive X-Y cut [4], exhaustive search [3] or density gradient [8]. However, the method based on X-Y cut [4] cannot robustly segment irregular panels, and none of these methods can handle unclosed and borderless panels. The method based on the connected component labeling (CCL) algorithm [1] can identify irregular panels as individual components, but fails to separate panels joined by some elements such as sound verbs or balloons. To handle joined panels, a sequence of N erosions followed by N dilations [5] are performed on the CCL mask to cut the connecting elements. However, the results of their algorithm are very sensitive to the value of N. The CCL mask for an unclosed panel may includes a group of fragmented borders as individual components,

ACM MM '14 November 3-7, Orlando, Florida, USA

http://dx.doi.org/10.1145/2647868.2654990.

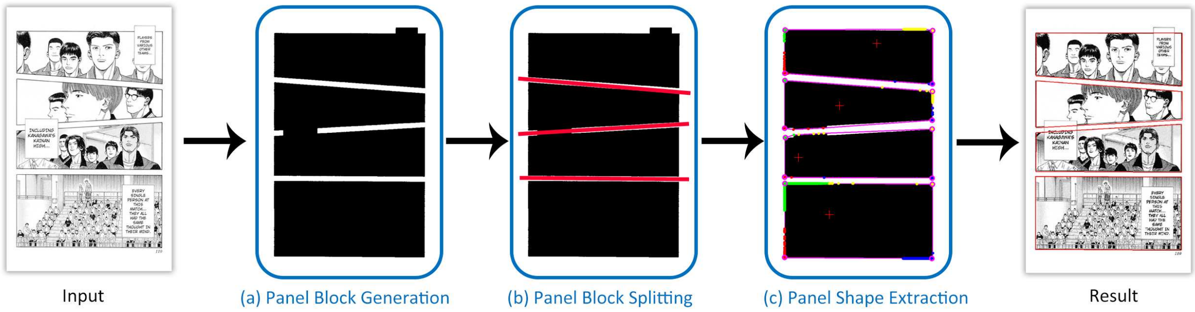

Figure 2: The pipeline of our method.

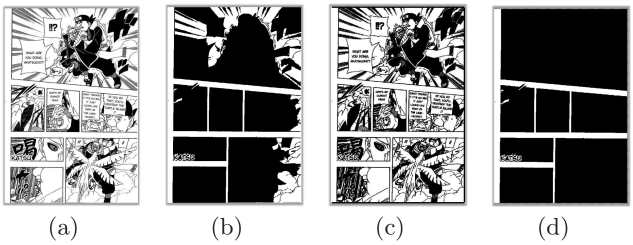

Figure 3: Generating the panel block without/with repairing unclosed panels. (a) Input manga page, and (b) its panel block. (c) Repaired manga page where the open boundaries of the unclosed panels are closed, and (d) its panel block.

making it impossible to obtain a complete panel shape from the erosion process. Although clustering the bounding boxes of the connected components [6] or fitting rectangles to the detected potential border lines and corners [7] may recover the shape of unclosed panels, they are only applicable to regular panels.

In this paper, we propose a robust method to extract panels from manga pages. Our method is composed of three steps: *panel block generation*, *panel block splitting* and *panel shape extraction*. Taking a digital manga page as input, our method first generates a panel block (Fig. 2(a)) by closing open panels and applying the CCL algorithm on the page background. It then adaptively places splitting lines in a recursive manner, which divides the panel block into a list of sub-blocks, each representing a single panel region (Fig. 2(b)). Finally, it recovers accurate panel shapes from the panel regions obtained (Fig. 2(c)).

2. PANEL EXTRACTION METHOD

2.1 Panel Block Generation

Given a manga page, we aim at extracting a panel block, representing a set of candidate panel regions. For this purpose, a common approach is to detect the connected components of black pixels on the foreground characters using the CCL method [1], and group them as panel regions. However, these methods may fail to preserve the panel shape, and could generate a labeling mask with many fragmented pieces. By inspecting manga pages, we observe that a panel may be composed of many disconnected components, but the pixels of the page background are often connected as a whole. Thus, we apply CCL on the page background (i.e., white pixels) to identify connected components as the background mask, and treat the remaining regions in the page as the panel block (Fig. 2(a)).

Handling unclosed panels. Our background mask obtained above may leak into the inside of the unclosed panels with open boundaries, leading to noises inside the panel block (see Fig. 3(b)). To avoid this issue, before computing the background mask, we repair the unclosed panels by adding boundaries to them to close the panels. We observe that the unclosed panels typically appear at the boundaries of a page. Therefore, we insert the missing boundaries by analyzing the intensity values of the pixels along the page boundaries. We first binarize the input manga page by setting the pixels with intensity values lower than 235 to 0, and 1 otherwise. Then, we find the page boundary that contains missing panel boundaries and replace it with a black line. When panel boundaries are missing, the visual contents flow outside the panels, resulting in greater intensity variation along the page boundary. Let $\varphi(x)$ be an intensity function of the binarized image, where x is the position of the pixel along the page boundaries. For each of the four page boundaries, a missing panel boundary is detected if the following conditions are satisfied: $\max(\varphi(x) * x) - \min(\varphi(x) * x) > 0.7 * L_e$ and $\sum_{x=1}^{L_e} \varphi'(x) > 10$, where $\varphi'(x) = \varphi(x) - \varphi(x-1)$ is the intensity gradient between neighboring pixels and L_e is the length of the boundary. A repaired image is shown in Fig. 3(c).

Open panel boundaries can also be caused by the fourth wall break effect, where characters or text break the panel boundaries and leaves part of the boundaries open. This can introduce holes formed by disjoint components inside the panel blocks. To fill the holes, we perform a grouping on the obtained connected components. Specifically, if more than half of a component's convex hull overlaps with another component's, the two components are considered to belong to the same panel, and therefore are merged together.

2.2 Panel Block Splitting

The aim of this step is to split the panel block from the previous step into a list of sub-blocks, each representing an individual panel region. This step is necessary since multiple panel regions may be joined together in the panel block, due to the existence of joined panels or fourth wall break effect (see the second and third panels in Fig. 2(a)). Since the comic layout structure can be formed by recursive partitioning [2], we separate joined regions by recursively applying a binary splitting procedure (see Fig. 2(b)). At each recursion level, we determine an optimal splitting line, which divides the current panel block into two sub-blocks. For example, starting with the whole panel block as the current block,

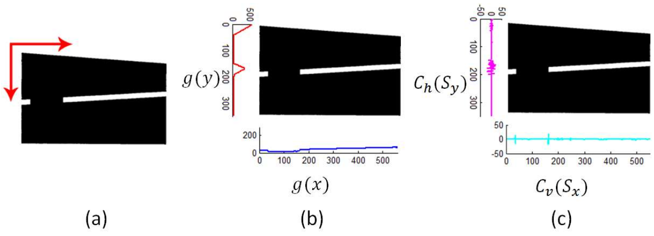

Figure 4: Computing splitting line. (a) current block to be split. (b) accumulated pixel values along horizontal and vertical directions. (c) the cost functions of placing a splitting line along horizontal and vertical directions.

we first split the current block into two sub-blocks using a horizontal or vertical splitting line. Then, each sub-block is regarded as the current block, which is further partitioned in the same binary manner. This process is performed recursively until the current block cannot be divided further.

To determine the locations of splitting lines, previous methods search for black boundaries of panels [7, 8, 1, 4] or white division stripes between the panels [3] on the original comic page, which is time-consuming and susceptible to noise caused by panel contents. Operating on the panel block, we first detect a set of candidate splitting lines (both horizontal and vertical) by evaluating a cost function of inserting a splitting line at a certain place. Then, an optimal splitting line, either horizontal or vertical, is adaptively determined by measuring the confidence values of the candidate splitting lines. For brevity, we focus our discussion on computing horizontal splitting lines. The vertical splitting lines can be computed in a similar way.

Computing candidate splitting lines. We determine the locations of candidate splitting lines based on the prior knowledge of the comic layout: two adjacent panels are often separated by white spacing between them. Thus, a horizontal splitting line should be placed close to the boundary between the panel and white spacing of the current block. To incorporate such knowledge in our approach, we define a *preference function* of placing a splitting line. The current block is first embedded in a 2D Cartesian coordinate system, with row and column corresponding to y-axis and x-axis, respectively (see Fig. 4(a)). Let $I(\mathbf{p})$ be the intensity function of the current block, where $\mathbf{p} = (\mathbf{p}_x, \mathbf{p}_y)$ is the pixel location. The preference function for placing a horizontal splitting line $S_y = \{\mathbf{p}|\mathbf{p}_y = y\}$ is then formulated as

$$C_h(S_y) = 2 * g(y) - g(y+1) - g(y-1), \qquad (1)$$

where $g(y)$ is the accumulated pixel values along S_y, i.e., $g(y) = \sum_{\mathbf{p} \in S_y} I(\mathbf{p})$. $C_h(S_y)$ has larger value at places where the horizontal splitting line is near to the boundary between the panel and white spacing. The horizontal splitting lines with $C_h(S_y) > 0.1 * W$ (W is the width of the current block) are referred to as candidate splitting lines. However, evaluating $C_h(S_y)$ alone cannot effectively detect diagonal white spacing, where $g(\cdot)$ may vary smoothly in a local neighborhood (as shown in Fig. 4(b)). As a result, we also include the horizontal splitting lines with $g(y) > 0.1 * W$, given the fact the splitting lines passing through the diagonal white spacing should include many white pixels. Since multiple horizontal splitting lines may be detected for one white spacing because of the white spacing's thickness, the splitting lines are finally grouped to form splitting stripes, and center lines within the groups are used as the final candidate splitting lines. We group the horizontal splitting lines if their distance is smaller that 30 pixels, and group the diagonal splitting lines if their distance is smaller than 50 pixels.

Eliminating false detection. In the resulting splitting stripes, some horizontal and diagonal stripes may overlap with each other, leading to false detection. We identify a diagonal splitting stripe as a false detection if it overlaps with any other horizontal splitting stripes. In addition, white holes on the current block, which are caused by open boundaries of the unclosed panels, can also give rise to false detection. To address this problem, for horizontal splitting stripes, we remove them if the distance between its leftmost and rightmost black pixels (if any) is larger than $0.7 * W$, i.e., the splitting stripes cut the black panel region on the current block. For each diagonal division stripe, we apply CCL to its local region on the current block. If the region contains more than 5 disjointed components, the stripe is discarded. Otherwise, we fit an ellipse to the component with longest major axis. If the ellipse approximates a long stripe (ratio of the ellipse's minor axis to major axis is smaller than 0.2), we consider the stripe to be a diagonal stripe, and fit a line to it as follows. We denote the stripe's local region as $R(x, y)$. For each column line of $R(x, y)$, we first compute a centroid point that divides the continuous white pixels into equal halves, i.e., $\mathbf{c}_x = \frac{\sum_{y=1}^{h} y * R(x,y)}{\sum_{y=1}^{h} R(x,y)}$. Then, a straight line is fitted to the centroid points, and used as a diagonal splitting line.

Determining an optimal splitting line. To perform binary partitioning adequately at each recursive level, we need to select an optimal splitting line from the computed candidate splitting lines. Take the results in Fig. 5 as an example. Properly partitioning panels in the fourth example should start by splitting the whole page *horizontally* into rows rather than vertically into columns. To this end, after the candidate splitting lines have been computed along both horizontal and vertical directions for the current panel block, we first assign each splitting line a confidence value, which is calculated as the ratio of the number of white pixels to its length. The splitting line that travels across the whole page without cutting the panel block has the highest confidence value of 1. Then, the optimal splitting line is selected as the one with the highest confidence value.

2.3 Panel Shape Extraction

Given a set of panel regions obtained in the previous step, this step aims at recovering accurate panel shape by detecting the four corners of each panel. We begin by computing a convex hull for each panel region and the centroid of the convex hull's vertices C (red crosses in Fig. 2(c)). Then, a 2D Cartesian coordinate system, centered at C, is constructed, thereby separating all the vertices of the convex hull into four quadrants. In each quadrant, we select a vertex that is furthest away from C as the initial corner point. Finally, we optimize the four corner points by performing a local search around each initial corner point so that the area of quadrilateral formed by the four points are maximized. Our implementation uses a circle of 10 pixels in radius as the local search region. This leads to the final panel shape as shown in Fig. 2(c).

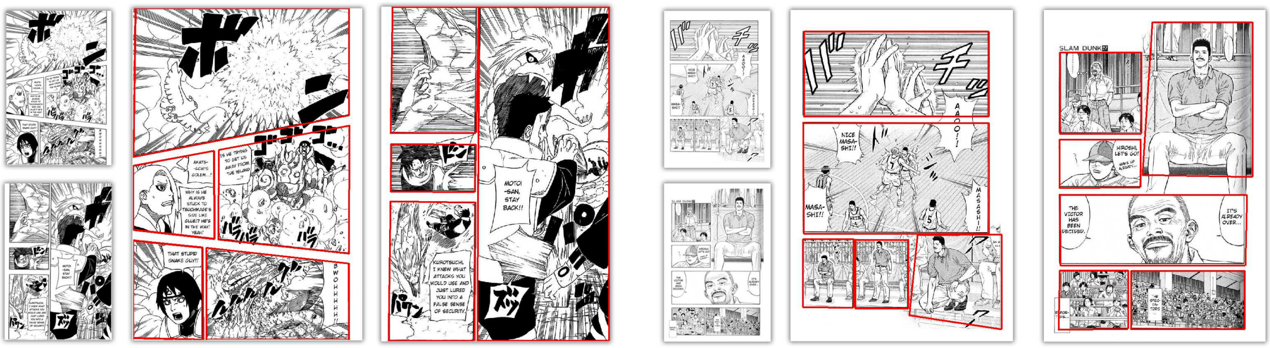

Figure 5: Panel extraction results on four manga pages with different stylistic effects. The left two pages are from "Naruto", and contain the fourth wall break effect, irregular panels and unclosed panels. The right two pages are from "Slum Dunk", and have borderless and joined panels.

3. EXPERIMENTS AND ANALYSIS

To evaluate the effectiveness of the proposed method, we have tested it on numerous manga pages. As shown in Fig. 5, the proposed method can accurately segment out the panels on the manga pages with various stylistic effects.

We have also compared our method with [5], which segments joined panels by performing an erosion-dilatation operation on the labeling mask generated from CCL. We apply both methods to a database of 104 manga pages from "Slum Dunk" and "Naruto". For each page in the database, we manually annotate panels as the ground truth. To evaluate the performance of the panel extraction methods, we calculate segmentation accuracy by computing the overlap ratio (Jaccard index) between the segmented panel and the ground truth. Let $G(x, y)$ and $S(x, y)$ be the masks of the annotated panel and the segmented panel. The overlap ratio ρ is computed as

$$\rho = \frac{|G \cap S|}{|G \cup S|}, \tag{2}$$

where $|M|$ is the number of non-zero pixels in the mask M. If ρ is larger than 90%, the panel is deemed to be successfully segmented. If all panels in a manga page are correctly segmented, it is defined as a successfully segmented page. The results are presented in Table 1. Our method achieves higher rates of successfully segmented panels and pages, and our average overlap ratio for correct segmentation is also higher than [5]. This is largely because [5] cannot properly deal with unclosed panels.

Our method may fail to detect the panel shape correctly when a large visual symbol cuts across the panel and breaks the panel boundary. In this case, our panel block generation cannot merge the two disjoint parts caused by the visual symbol, resulting in an incomplete panel shape (as shown in the figure on the right).

4. CONCLUSION

In this paper, we have proposed a robust method for automatically extracting panels from manga pages, which are characterized by various stylistic effects. Our method starts by extracting a panel block, i.e., a set of panel regions. It then employs a recursive binary splitting strategy to partition the panel block into disjoint panel regions, where we dynamically determine the location of an optimal splitting line at each recursion level. Accurate panel shapes are finally obtained from the panel regions. Our results show that our method is insusceptible to various stylistic effects and complex layout style of manga, achieving a higher successful segmentation rate, as compared with previous work. We are interested in further exploring applications that can benefit from our method, such as converting scanned comic pages into a format that is readable on mobile devices, and an automatic comic annotation system for comic analysis.

	Page rate %	Panel rate %	Average Overlap Ratio
Our approach	**87.9**	**91.3**	**0.97**
[5]	73.9	74.1	0.94

Table 1: Comparison of our method and [5]. The successful segmentation rates for both pages and panels, and the overlap ratio for successfully segmented panels are given.

5. REFERENCES

[1] K. Arai and T. Herman. Method for automatic e-comic scene frame extraction for reading comic on mobile devices. In *Proc. ITNG*, pages 370–375, 2010.

[2] Y. Cao, A. Chan, and R. Lau. Automatic stylistic manga layout. *ACM Trans. on Graphics*, 31(6), 2012.

[3] C. Chan, H. Leung, and T. Komura. Automatic panel extraction of color comic images. In *Proc. PCM*, pages 775–784, 2007.

[4] E. Han, K. Kim, H. Yang, and K. Jung. Frame segmentation used mlp-based x-y recursive for mobile cartoon content. In *LNCS: Human-Computer Interaction*, volume 4552, pages 872–881, 2007.

[5] A. Ho, J. Burie, and J. Ogier. Comics page structure analysis based on automatic panel extraction. In *Proc. IAPR GREC*, 2011.

[6] C. Rigaud, N. Tsopze, J. Burie, and J. Ogier. Robust frame and text extraction from comic books. In *Proc. IAPR GREC*, pages 129–138, 2013.

[7] M. Stommel, L. Merhej, and M. Müller. Segmentation-free detection of comic panels. In *LNCS: Computer Vision and Graphics*, volume 7594, pages 633–640. 2012.

[8] T. Tanaka, K. Shoji, F. Toyama, and J. Miyamichi. Layout analysis of tree-structured scene frames in comic images. In *Proc. IJCAI*, pages 2885–2890, 2007.