Optimized Comics-Based Storytelling for Temporal Image Sequences

Wei-Ta Chu, *Member, IEEE*, Chia-Hsiang Yu, and Hsin-Han Wang

***Abstract*—**We propose a system to transform any temporal image sequence into a comics-based presentation, as an effective and interesting storytelling manner**.** Three main components, including page allocation, layout selection, and speech balloon placement, are respectively formulated as optimization problems, and systematic approaches are proposed to find solutions. Page allocation is viewed as a labeling problem, and the best solution is determined by the genetic algorithm. Importance values of images and predefined layouts are both represented in vector forms, and the best layout is selected by finding the best match between vectors. Feasible solutions of speech balloons constitute a solution space, and the best solution that jointly describes the best locations of all balloons in a page is determined by the particle swarm optimization algorithm. Objective evaluation and subjective evaluation are designed from various perspectives to demonstrate effectiveness and superiority of the proposed system.

***Index Terms*—Comics-based storytelling, genetic algorithm, particle swarm optimization, page allocation, layout selection, speech balloon placement**

I. Introduction

Effective presentation and communication of data has been widely studied in various communities for a long time, such as information visualization in computer graphics, and image/video summarization in multimedia and computer vision. Recently, *storytelling* is viewed as an emerging research field in the visualization society [1], and has been well recognized in journalism [2] and scientific data visualization [3]. The multimedia society mainly puts efforts on building stories as a combination of multiple modalities, such as presenting a tour with photos and music [4] and presenting actions and interactions in a virtual world [5]. In the era where tremendous amounts of data in heterogeneous forms are shared and circulated, researches in different fields have revealed that a well-paced presentation with well-organized narrative structure not only effectively visualizes massive data, but also provides new perspectives for viewers to obtain insights.

Stories are sequences of causally related events presented in text, images, graphics, or videos, and storytelling can be manifested in various forms. Recently from the multimedia research field, comics-based presentation for movies [6], animation [7], photos [8], and gaming behaviors [5] has become an emerging type of presentation carrying important information and facilitating entertaining and highly interactive browsing experience. Comics are believed to be an ideal medium for visual storytelling because of rich expressivity, high interactivity, and high interoperability. For rich expressivity, a story's progression can be expressed by images with various panel sizes and shapes, according to story pace or images' importance. For high interactivity, reading stories from comic books is an efficient and broadly acceptable manner by just sliding our fingers to turn comic pages. Comparing with slideshows and videos, readers have full control on their reading pace as no temporal restriction is imposed. Comics can be displayed on both desktop PCs and mobile devices, or can even be physically presented on papers by printing. We call this characteristic high interoperability, which makes comics easily and widely spread on heterogeneous devices.

Comparing with text, comics present stories in visual forms; comparing with slideshow, comics present richer information such as captions, panels of various sizes, and layout showing story pace; comparing with videos, comics provide higher interactivity and higher interoperability. However, creating a comic from an image sequence is not an easy task, especially storyboarding, layout planning, and speech balloon placement. In this paper, we focus on developing a systematic approach associated with optimization algorithms to generate optimized comics-based storytelling from temporal image sequences.

Four ingredients should be included in all forms of stories: perspective, characters, imagery, and language [3]. In comics-based storytelling, the perspective ingredient is manifested by panel arrangement that builds reading pace and shows information at multiple granularities. Characters and imagery ingredients are inherently embedded in the input image sequence; and the language ingredient is expressed by speech balloons.

Figure 1 shows sample pages of the proposed comics-based storytelling. There may be different numbers of images (panels) in different comic pages, depending on visual content and the reading pace to be built. Varied space may be allocated to different panels. To show a large image in a small panel, less important part of the image may be removed so that the aspect ratio of the remaining region matches with that of the targeted panel.

Wei-Ta Chu, Chia-Hsiang, and Hsin-Han Wang are with National Chung Cheng University, Taiwan. (email: wtchu@cs.ccu.edu.tw, xneonvisionx@hotmail.com, xinghan177@gmail.com).

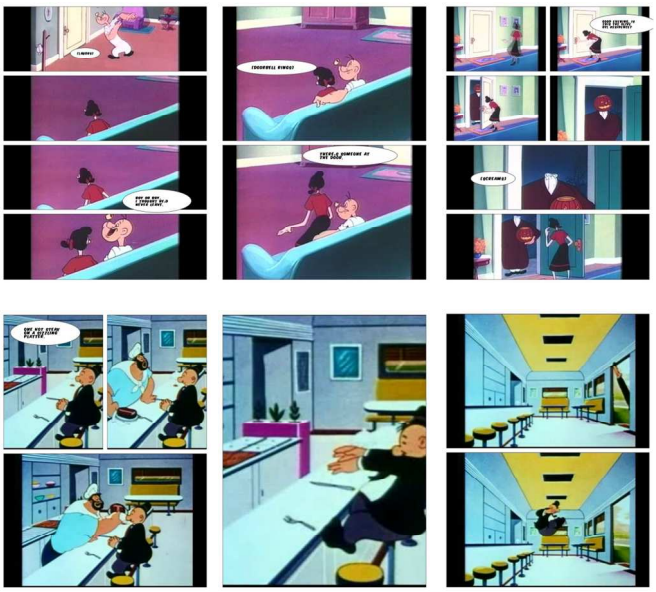


Figure 1. Top: Sample comic pages generated from "Popeye: Fright to the Finish (1954)" in the public domain. Bottom: Sample comic pages generated from "Popeye: Spree Lunch (1957)" in the public domain.

In summary, the following challenges should be addressed to build an automatic comics-based storytelling system.

- Page allocation: *How to segment a given temporal image sequence, so that images in the same subsequence present similar visual content and are better to be put into the same comic page?* Images showing similar visual content are better to be presented in the same page. Furthermore, by varying the number of panels in a page, we can build different reading paces. For example, we feel that the top row of Figure 1 shows a higher pace than the bottom row, because content in the former one changes more frequently. In this paper, we formulate this issue as an optimization problem that can be solved by the genetic algorithm.
- Layout selection: *Which layout is best to present a given image subsequence?* To present K images in a page, the best layout should be selected from a pool of predefined layouts containing K panels. Intuitively, we tend to allocate more space to more important images. In this paper, distributions of image importance and layouts are characterized in vector forms, and the best layout is determined by similarity measurement between vectors.
- Speech balloon placement: *How to place speech balloons, so that important content in images are not occluded, and balloons' positions direct viewer's gaze to build a pleasing reading trajectory?* Considering the comic design theory, we formulate this issue as an optimization problem that can be solved by the particle swarm optimization algorithm.

Contributions of this work are threefold. First, we propose a system framework that integrally consists of various aspects of automatic comics generation, including page allocation, layout selection, and speech balloon placement. Second, important components are formulated as optimization problems, and well-designed optimization algorithms are adopted to systematically solve these problems. Third, to more faithfully demonstrate effectiveness of the proposed system, in addition to subjective evaluation, we ask semi-professional editors to edit comics and quantitatively compare automatically generated comics with manual ones.

II. Related Works

Information visualization research has been developed for decades to visualize mass data collection, in order to efficiently express essence of data or even provide alternative perspectives to discover unknown characteristics. While data exploration and analysis have been widely studied, recently presenting information as a story has been proven as an effective way for information conservation and communication, and makes information more memorable [9].

Photographs are viewed as the best medium for storytelling, because they were often captured to record visual experience in daily life. Balabanovic et al. [12] developed a device to facilitate sharing photo-based stories locally and remotely. Given a text-based story, the story picturing engine [13] visualized this story by images through searching an annotated image database. Chen et al. [4] generated photo slideshow in a tile-like manner, with page changing according to pace of background music. Commercial software like Microsoft Photo Story[1], Cyberlink MagicDirector[2], and Muvee Reveal[3] provide highly interactive functions enabling users to manually generate photo-based storytelling, mainly in the representation of slideshow, animation, or videos.

To efficiently present a story embodied in videos, many researchers transform videos into static storyboards or image collages. For example, Ciocca et al. [14] transformed videos into multilevel storyboards constituted by hierarchical representation of keyframes. Storyboard presentation is also provided by commercial software, such as KMPlayer[4]. Wang et al. [15] proposed a video collage system that automatically constructed a compact and visually appealing synthesized collage from a video sequence. With similar concepts, Pritch et al. [16] proposed a video synopsis exhibiting static presentation showing essential activities of a video.

Slideshow, storyboard, or image collage can be used to tell stories from various perspectives, but their expressivity is limited because no language expression is provided. Comics-based presentation, therefore, emerges as a new type of storytelling. Yeung and Yeo [41] proposed one of the earliest comic-like video summaries, called pictorial summary of video posters. Many essential ideas originate from this work, such as considering image importance to allocate display space and generating grid-based layouts based on a basic unit. Uchihashi et al. [40] proposed the Video Manga system to generate pictorial summaries, with clearer statement about creating summaries resembling comic books. A frame packing algorithm was proposed to efficiently fill space by frames with varied sizes. Girgensohn [36] improved these early systems by proposing a fast frame packing algorithm with a few heuristic rules.

[1] Microsoft, http://www.micrisoft.com
[2] Cyberlink, http://www.cyberlink.com
[3] Muvee, http://www.muvee.com
[4] KMPMedia, http:// www.kmpmedia.net

The Movie2comics system [6][29] was proposed to produce comic-like presentation for movies based on keyframes, with consideration of script-face mapping and cartoonization. Given importance of keyframes, Calic et al. [30] proposed a dynamic programming approach to find optimal configurations of displayed widths and heights for a sequence of keyframes. This work was further improved to enable high interactivity in multi-scale comic-like summaries, considering the effect of layout disturbance [31]. Herranz et al. [32] made comic-like summaries more compact by appropriately overlaying regions of interest on panels, making an effect similar to picture in picture. Rather than selecting layouts from a pool or generating a layout consisting grids, Cao et al. [33] proposed a data-driven approach where a probabilistic model was built to describe panel arrangement. Myodo et al. [34] utilized a tree structure to describe splitting lines on the screen to generate non-grid panels. Toyoura et al. [10] designed a rule-based layout algorithm considering camera works like pan, zoom, and fade to arrange panels. The effects of camera works can thus be inferred from static comic pages. In addition to being a video summary, the DigestManga system [35] was proposed to show that comic-like presentation is able to facilitate video editing.

Wen et al. [8] developed an online comic composition system that automatically generated initial comic pages, which can later be refined by users through rich interactive functions, such as moving photos to display a region of interest and adding speech balloons or icons. Similar idea was adopted to show gaming behaviors in a virtual world as well [5].

In our previous work [7], animation videos were transformed into comics by a system that automatically determined how a keyframe sequence was segmented and composed to generate comic pages. Optimization techniques were employed to make comics-based storytelling generation a systematic practice. In [20], we further considered comic design principles in the optimization process, and added the language perspective, i.e., speech balloons, to comic pages. In this work, we improve our previous works from the following perspectives:

- Image selection: State of the art saliency analysis and image matching techniques are employed to estimate image importance and select keyframes, facilitating more promising layout selection.
- Multiple balloons: We modify the objective function in our previous optimization framework to allow displaying multiple balloons in a panel, facilitating more effective presentation and better reading experience.
- Evaluation: Various objective evaluations and subjective evaluations are designed to show effectiveness of the proposed system. We compare automatically generated comics with manually edited ones, and show how designs of objective functions influence comics generation results.

III. Preprocessing

The most common temporal image sequences come from keyframes extracted from videos or photos with shooting time in an album. For the former case, we detect shot change boundaries in a video, and then extract keyframes from each video shot. In [7], we considered color histogram distances between frames in a shot, and motion type of this shot, to select parts of frames as keyframes. This procedure generally worked well, but sometimes it selected near-duplicate frames as keyframes because of high motion dynamics. According to our experiments, redundant images in different panels annoy users. To avoid this annoying effect, in this work we further adopt the keypoint-based approach [21] to find near-duplicate frames and eliminate redundant keyframes. Finally, the method proposed in [22] was adopted to detect scene change boundaries as an important factor in page allocation.

For photos in an album, the keypoint-based approach [21] is also adopted to eliminate redundant photos. To detect photo scene change boundaries, we calculate the shooting time distance between temporally adjacent photos, and adopt the adaptive sliding window approach proposed in [23].

In the next section, we design an optimized page allocation scheme to find the optimal segmentation of a temporal image sequence, so that images in the same subsequence will be put into the same comic page.

IV. Optimized Page Allocation

Given a temporal image sequence where images may be associated with text annotation, the goal of the page allocation component is to automatically segment this sequence so that images in the same subsequence are appropriate to be put into the same comic page. To make discussion concrete, we take transforming videos into comics as the main exemplar application, though the proposed optimization scheme is general to any temporal image sequence. We will use the term *keyframes* and *images* interchangeably in the following.

Let $I_1, ..., I_N$ denote the temporal sequence consisting of N images. We would like to allocate appropriate number of comic pages that may include various numbers of panels and exhibit the following characteristics:

- Visual coherence: Consecutive or similar visual content is better to be put into the same comic page.
- Reading pace: A comic page consisting of more images builds *higher reading pace*. Therefore, photos densely took in a short period, or keyframes conveying high motion, are better to be put into the same page containing more panels.

Basically this task is a labeling problem, in which appropriate number of classes is to be determined to label these N images. There may be tremendous number of labelings. Fortunately, in our work temporal continuity of consecutive images should be maintained. For example, if five images need to be labeled, the labeling 11223 is legal, but the labeling 12213 is not allowed, where each number indicates the ID of a class. To efficiently and systematically solve this problem, we exploit the genetic algorithm (GA) to find the optimal labeling.

We first randomly generate initial chromosomes. Each chromosome represents a labeling $\boldsymbol{x} = (x_1, x_2, ..., x_N)$, where $x_i = n$ denotes the image I_i is assigned to the nth comic page ($i = 1, ..., N$, and n is a positive integer). Note that if $x_i \neq x_{i+1}$, then $x_i = x_{i+1} - 1$. The first image x_1 is always assigned to the first page, i.e., $x_1 = 1$. Because visual content

would be too messy or important objects would be critically shrunk if too many images are squashed into a page, we set an upper bound U to limit the maximum number of images being labeled as the same class. Therefore, an initial chromosome is generated by the following process.

(1) Randomly pick a number from 1 to U, say m_1, and assign the first m_1 images into the first page, i.e., $x_1 = x_2 = \cdots = x_{m_1} = 1$.

(2) Randomly pick a number from 1 to U, say m_2, and assigned the following m_2 images into the second page, i.e., $x_{m_1+1} = x_{m_1+2} = \cdots = x_{m_1+m_2} = 2$.

(3) The same process repeats until all images are assigned.

To set the upper bound U, we read ten Japanese and western comic books, respectively, and found that a comic page rarely contains more than eight panels. Therefore, in our work the upper bound U is set as 8.

Figure 2(a) shows eighteen keyframes extracted from an animation video. Each keyframe is viewed as a gene and a chromosome is constituted to show a sample allocation, as illustrated in Figure 2(b). This chromosome represents that the eighteen keyframes are arranged into four pages. The first four keyframes are put into the first page, followed by three, four, and seven keyframes putting into the second, the third, and the fourth pages, respectively.

To find the optimal page allocation situation, the objective function value (or fitness in GA) of a chromosome is described as follows. Let $\boldsymbol{x} = (x_1, ..., x_N)$ denote a chromosome where images are assigned to M pages $(P_1, P_2, ..., P_M)$, in which $P_j = \{I_i | x_i = j\}$ consists of images assigned to the jth page. Notice that $|P_1| + \cdots + |P_M| = N$, with $|P_j|$ denoting the cardinality of the set P_j. Considering the factor of visual coherence, we calculate the average color histogram similarity A_c corresponding to the chromosome $\boldsymbol{x}$ as

$$A_c = \frac{1}{M}\sum_{j=1}^{M}\left(1 - \frac{\min_{I_a \in P_j, I_b \in P_j} d_c(I_a, I_b)}{\max\limits_{\substack{I_a \in P_j \\ I_b \in P_{j-1} \text{ or } I_b \in P_{j+1}}} d_c(I_a, I_b)}\right), \quad (1)$$

where $d_c(I_a, I_b)$ is the Euclidean distance between the RGB color histograms of I_a and I_b. The value A_c is designed that images assigned to the same page are better to be visually similar.

If $I_1, ..., I_N$ are keyframes extracted from a video, we can further embed motion coherence in the fitness function. The average motion similarity A_m corresponding to the chromosome $\boldsymbol{x}$ is defined as

$$A_m = \frac{1}{M}\sum_{j=1}^{M}\left(1 - \frac{1}{Z}\sum_{I_a \in P_j, I_b \in P_j} MHD(I_a, I_b)\right), \quad (2)$$

where $Z = \binom{|P_j|}{2}$ is a normalization factor. The function $MHD(I_a, I_b)$ measures the motion histogram difference between two images. The motion histogram corresponding to I_i is obtained based on the motion from I_i to its next frame in the original video.

It is better that images in the same page belong to the same video scene. Based on results of scene boundary detection mentioned in Section III, the scene coherence characteristic A_s is designed as

$$A_s = \frac{1}{Z}\sum_{I_a \in P_j, I_b \in P_j} \delta(I_a, I_b), \quad (3)$$

where $\delta(I_a, I_b) = 1$ if keyframes I_a and I_b belong to the same scene, and $\delta(I_a, I_b) = 0$ otherwise. The value A_s of a page is equal to one if all keyframes come from the same scene, and is smaller than one if some of them come from different scenes.

If video shots change frequently, we tend to put more keyframes into the same page to build higher reading pace. In other words, the number of keyframes in a page is inversely proportional to the total lengths of their corresponding video shots. This factor is considered by allocating pages such that the total length of shots in each page is similar. It is quantitatively evaluated as the value A_τ:

$$A_\tau = 1 - std(T_1, T_2, ..., T_M), \quad (4)$$

$$T_j = \frac{\tau_j}{\max(\tau_1, ..., \tau_M)}, \quad (5)$$

where τ_j is the total length of shots in the page $P_j = \{I_i | x_i = j\}$. The value τ_j is obtained by summing the lengths of shots from which the keyframes in $\{I_i | x_i = j\}$ are extracted. The function $std(\cdot)$ calculates the standard deviation.

The fitness $f(\boldsymbol{x})$ of the chromosome $\boldsymbol{x}$ in the GA process is then defined as a linear combination of these four factors:

$$f(\boldsymbol{x}) = \alpha A_c + \beta A_m + \gamma A_s + \zeta A_\tau, \quad (6)$$

where α, β, γ, and ζ are empirically set as 0.4, 0.2, 0.2, and 0.2, respectively. The value α is set larger because color coherence often plays the most important role to achieve higher visual coherence.

In our work, forty chromosomes were generated at random as the initial population $\Omega(0)$. At the tth iteration of the GA process, the fitness $f(\boldsymbol{x}^{(t)})$ of each member $\boldsymbol{x}^{(t)}$ of the population $\Omega(t)$ is calculated. We then form a set $\Phi(t)$, called the mating pool, with the same number of elements as $\Omega(t)$ using the roulette-wheel scheme [24], i.e., chromosomes are selected from $\Omega(t)$ with the probability proportional to their fitness. Elements in the mating pool $\Phi(t)$ are then selected at random to conduct crossover and mutation operations to generate the next-generation population $\Omega(t+1)$. The GA process keeps running until 100 iterations have been completed. Finally, the chromosome with the highest fitness represents the best page allocation situation.

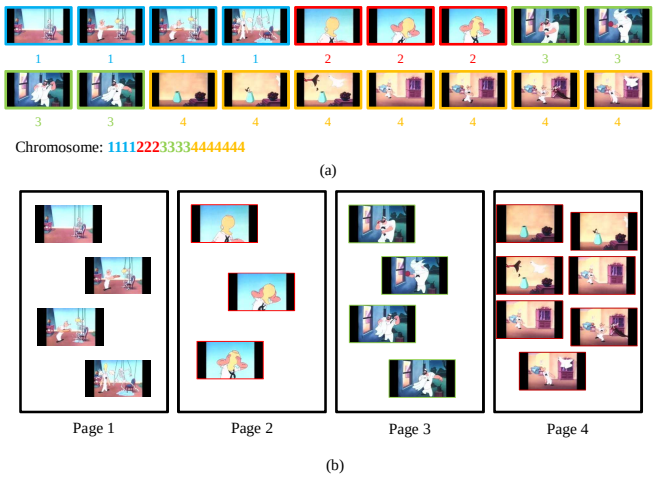
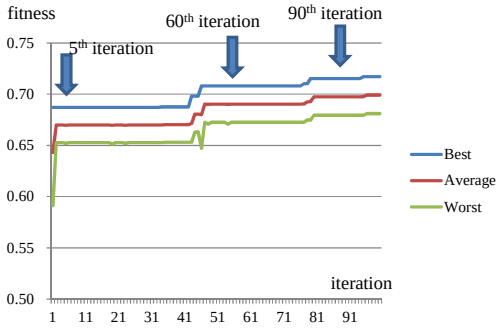
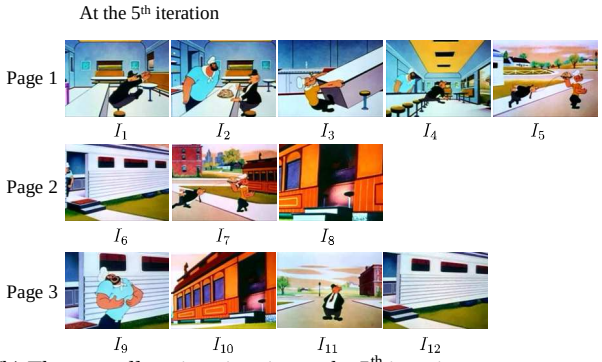


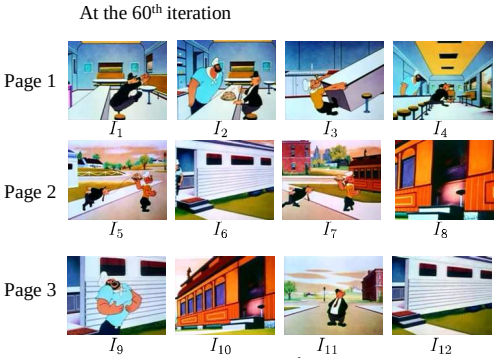
Figure 2. (a) Keyframes extracted from the video "Popeye: Fright to the Finish (1954)" in the public domain and a sample chromosome. (b) The allocation corresponding to the chromosome shown in (a).



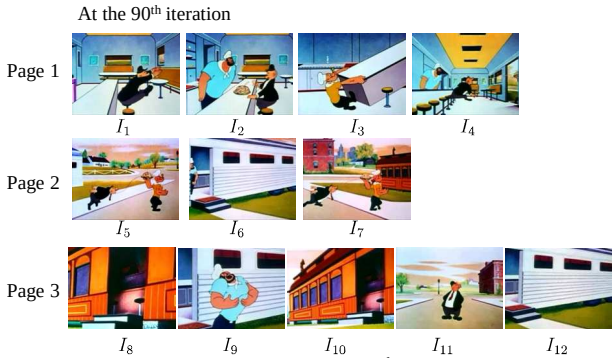
(a) The evolutions of the best, average, and worst objective function values in the GA process.



(b) The page allocation situation at the 5th iteration.



(c) The page allocation situation at the 60th iteration.



(d) The page allocation situation at the 90th iteration.

Figure 3. The evolutions of objective function values and their corresponding page allocation situations in the GA process. The input artwork is from "Popeye: Spree Lunch (1957)".

Note that the next-generation chromosomes generated by crossover and mutation operations should also conform to the temporal constraint. In our work, we simply check whether a generated chromosome conforms to the temporal constraint after every crossover or mutation operation. A chromosome is discarded if it does not meet the constraint. We keep randomly selecting parent chromosomes from the mating pool to generate child chromosomes until the number of elements in the next-generation population is the same as the parent population.

Figure 3(a) shows an example of evolutions of the best, average, and worst objective functions values in the GA process. The objective function value gradually increases as the process iterates more. Figure 3(b) shows the page allocation situation if we stop the GA process at the 5th iteration. Three pages were allocated. However, we clearly see that the keyframe I_5 appears in a scene different from I_1, I_2, I_3, and I_4, and thus I_5 is better to put into another page. This expectation was fulfilled when the process iterates sixty times, as illustrated in Figure 3(c). However, an event continuously takes place from I_8 to I_{12}, and it is better to put I_8 into the same page as I_9 to I_{12}. Finally, if we stop the GA process at the 90th iteration, the first four keyframes are put into the first page, followed by three keyfrmes in the second page and the last five keyframes in the third page, showing the best case as we expected.

V. Optimized Layout Selection

The next step is selecting the best layout to display images assigned to the same page. If K images are to be displayed, we would like to select the best layout from a predefined set that includes layouts with K varied-sized panels. Each panel is a room allocated to display an image. The desired best layout should have the following properties: 1) More important images should be allocated larger panels; 2) keyframes extracted from the same shot or photos consecutively taken in the same place are better to be put in the same row of panels; 3) keyframes with more subtitle words or photos with more annotation are to be allocated larger panels. Note that in this work, we assume that subtitle text is beforehand available in SubRip files (in .srt format) if the processed temporal image sequences come from videos. Text-based annotation is viewed as subtitle text if the processed temporal image sequences come from photo albums.

The basic idea of layout selection is as follows. We evaluate the "importance distribution" of panels in a layout, evaluate the "importance distribution" of images to be displayed, and then determine the images-layout pair that has the most similar distributions. Various layouts consisting of various numbers of panels were designed in advance. In our work, totally 119 layouts containing one to eight panels were manually designed according to our observations of common templates in popular comic books. Figure 4(a) shows three sample layouts consisting of five panels. The importance distribution of the ith layout is then described by a vector: $\boldsymbol{\ell}_i = (\ell_1, \ell_2, ..., \ell_K)$, where ℓ_k, $k = 1, ..., K$, is the ratio of the area of the kth panel to the area of the whole page.

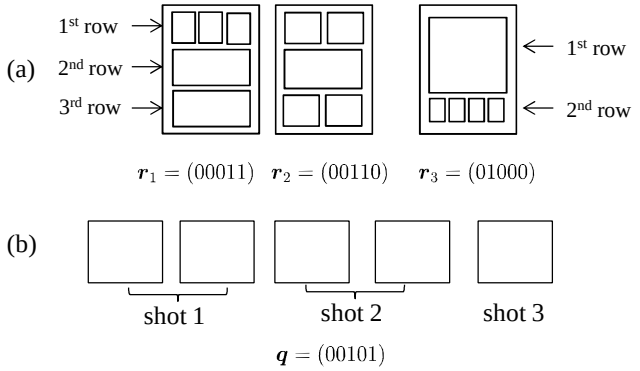


Figure 4. (a) Three sample layouts containing five panels and the corresponding binary vectors showing panel arrangement. (b) A sequence consisting of five keyframes and the corresponding binary vector showing the shot change situation.

The importance distribution of K images is also described by a vector $\boldsymbol{f} = (e_1, e_2..., e_K)$, where e_k is the importance value of the kth image f_k. To consider the first and the third desired properties, the importance value e_k of an image f_k is defined as

$$e_k = \mu_1 \times \frac{a_k}{A} + \mu_2 \times \frac{s_k}{S} + \mu_3 \times mCHD_k. \qquad (7)$$

The first term is the ratio of the area of the region-of-interest (ROI) in f_k, i.e., a_k, to the total area of ROIs in $\{f_1, ..., f_K\}$, i.e., $A = a_1 + \cdots + a_K$. This term measures relative importance of ROIs in these images. Note that if there are many ROIs in an image, the minimum bounding box covering all ROIs is found to calculate this term. ROIs can be determined by prior works such as [17] and [18]. In this work, we employ the method in [17] to find ROIs. The second term is the ratio of the number of subtitle words in f_k, i.e., s_k, to the total number of subtitle words in $\{f_1, ..., f_K\}$, i.e., $S = s_1 + \cdots + s_K$. The third term $mCHD_k$ is the minimum color histogram distance from f_k to other images in $\{f_1, ..., f_K\}$. That is, $mCHD_k = \min_j \|H_j - H_k\|$, for $j \neq k$ and $j = 1, ..., K$. The notation $\|H_j - H_k\|$ means the normalized Euclidean distance between the RGB histogram H_j of f_j and the RGB histogram H_k of f_k. Overall, the second term emphasizes the importance of subtitle, while the third term indicates that an image more similar to others is more important. The weights μ_1, μ_2, and μ_3 are empirically set as 0.3, 0.3, and 0.4, respectively, because visual coherence is generally the most important factor to evaluate image importance.

To measure how appropriately a layout matches with a given image sequence, we refer to the inner product matching scheme proposed in [4], and thus measure it as $\langle \boldsymbol{\ell}_i, \boldsymbol{f} \rangle$.

To consider the second property, we first describe how panels are arranged into rows and represent it as a K-dimensional binary vector. If the kth panel is in the ρth row, and the $(k+1)$th panel is in the $(\rho+1)$th row, the bit at the $(k+1)$th dimension is set as 1, otherwise 0. In Figure 4(a), reading from left to right and top to down, the panel arrangement for the first layout is represented by the vector $\boldsymbol{r}_1 = (00011)$, because the first three panels are located in the first row, the fourth panel is located in the second row, and the fifth panel is located in the third row. The panel arrangements for the second and third layouts are represented by vectors $\boldsymbol{r}_2 = (00110)$ and $\boldsymbol{r}_3 = (01000)$, respectively.

We also describe how keyframes belong to different shots as a binary vector. Assume we have five keyframes to be placed into a 5-panel layout. Suppose that the first two keyframes come from the first video shot, the third and the fourth keyframes come from the second video shot, and the fifth keyframe comes from the third video shot, as illustrated in Figure 4(b). If the kth keyframe is from the ξth shot, and the $(k+1)$th keyframe is from the $(\xi+1)$th shot, the bit at the $(k+1)$th dimension is set as 1, otherwise 0. Therefore, in Figure 4(b), how keyframes belong to different shots is represented by the vector $\boldsymbol{q} = (00101)$. The degree of matching m_i between the ith layout and the keyframe sequence is then evaluated as

$$m_i = 1 - \frac{|XOR(\boldsymbol{q}, \boldsymbol{r}_i)|_1}{K}, \qquad (8)$$

where $|\cdot|_1$ is the L1 norm of the vector. The value m_i ranges from 0 to 1, and a larger value means higher matching.

Finally, by jointly considering image importance and panel arrangement, the best layout $\boldsymbol{\ell}_{i^*}$ is determined by

$$i^* = \arg\max_i (\langle \boldsymbol{\ell}_i, \boldsymbol{f} \rangle \times m_i). \qquad (9)$$

Once the best layout is determined, the panel a keyframe should be put inside is also determined. Different panels may have significantly different aspect ratios, and undoubtedly blindly resizing a keyframe would largely distort visual information. Therefore, we would like to find an important region inside the keyframe that simultaneously covers salient objects and meets the aspect ratio of the targeted panel. This task is accomplished by a method similar to that proposed in [4], and the composition process is illustrated in Figure 5. First, based on global color contrast [17] a saliency map is constructed to represent saliency characteristics of a keyframe, and the centroid of the map is determined. Starting from the centroid, we expand the region towards four directions (top, down, left, right) according to the aspect ratio of the targeted panel. The expansion stops when at least two boundaries of this region reach boundaries of the keyframe. The selected region is then resized and stuck on the targeted panel. After sticking regions from keyframes on all panels, a comic page is generated.

Figure 6 uses two examples to show effectiveness of the proposed layout selection method. The layout in Figure 6(a) consists of two small panels in the top row and one large panel in the bottom row, and the layout in Figure 6(b) contains three equal-sized panels arranged in three rows. In Figure 6(a) the first two images presenting the same object are arranged in the same row, yielding better presentation. Figure 6(c)(d)(e) show another examples displaying six images. Figure 6(c) is the selected best layout, and Figure 6(d) and Figure 6(e) are two layouts where panels are derived by equally dividing the page in the horizontal direction, and dividing in both horizontal and vertical directions, respectively. Comparing with Figure 6(c), we would feel dazzled if images were displayed like Figure 6(d), and get bored if images were displayed like Figure 6(e). We can clearly see that Figure 6(c) more appropriately presents information in a more lively way.

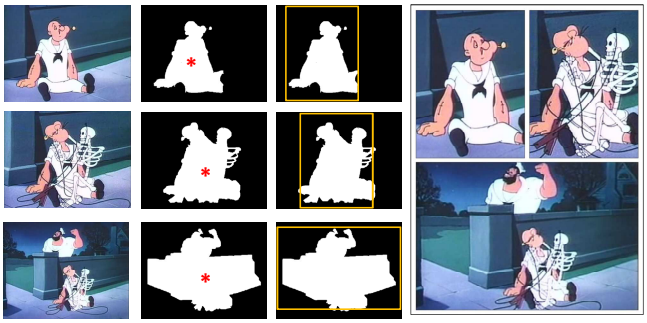


Figure 5. Illustration of the composition process. The input artwork is from "Popeye: Fright to the Finish (1954)".

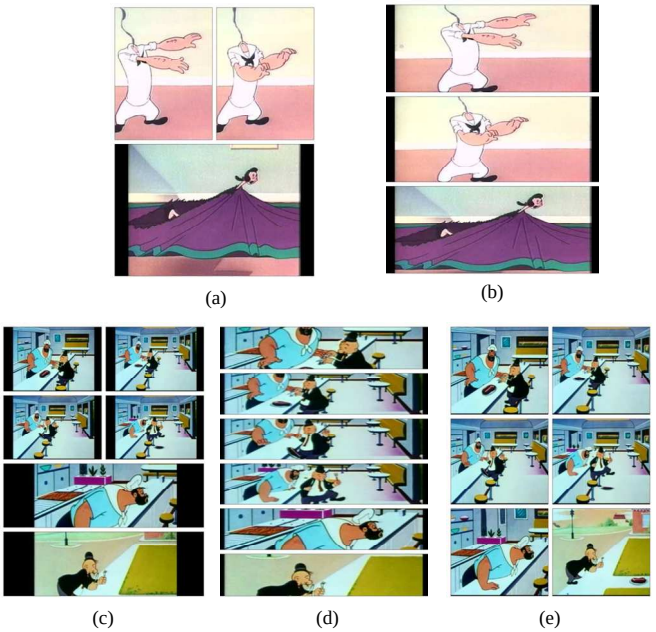


Figure 6. Comparison of layout selection results. Example 1: Layout selected by the proposed method (a) and by equal allocation (b). Example 2: Layout selected by the proposed method (c) and two different equally-allocated layouts (d)(e). The input artwork for (a)(b) is from "Popeye: Fright to the Finish (1954)", and that for (c)(d)(e) is from "Popeye: Spree Lunch (1957)".

Layout optimization is generally the main focus of previous works on comic-like presentation. Such works can be roughly categorized into three groups: 1) selecting layouts from a predefined set, such as [6] and [29]; 2) generating grid-based layouts, such as [30], [31], and [32]; and 3) generating non-grid layouts, such as [33] and [34]. In addition to generating comic-like video summaries, layout optimization is also important in presenting information on websites [37][38] and newspapers [39]. A variety of ways range from sequence matching [6], dynamic programming [30], probabilistic model [33], to genetic algorithm [37], were proposed to find/generate optimal layouts. In our work, we simply select the best layout from a predefined set with the consideration of subtitle information and panel arrangement. This simple method is especially efficient when a large number of keyframes need to be presented. The disadvantage is that only a limited number of choices can be made, and thus more elegant features, such as camera works [10], are hard to be incorporated. Overall, to keep this paper more focused, we put more efforts on building the optimization framework rather than elegant layout generation.

VI. Optimized Speech Balloon Placement

A. Balloon Generation

To constitute the language ingredient of storytelling, speech balloons are generated and placed in panels. Two factors are considered in balloon generation: balloon size and balloon shape. Obviously a balloon would be larger if it contains more words. In addition, words conveying exclamation or excitement can be enlarged to more faithfully represent emotion. Emotion can further be exhibited by shape of balloon [19]. Ellipses are most commonly used to show general emotion, while for speech containing excitation, exclamation, and violence words, jagged edged balloons can be used.

Assume that the default size of each English letter is $w_c \times h_c$, which means every letter is bounded by a box of the width w_c and the height h_c. For Chinese or Japanese, a *letter* here corresponds to a Chinese/Japanese character. Though not limited to any language, the idea of balloon generation described in the following is mainly for Chinese/Japanese speech balloons. Suppose that the keyframe I_i is determined to be put into the panel C_i, and there are some text corresponding to I_i. Width of a letter $\hat{w}_c$ in C_i is calculated as

$$\hat{w}_c = w_c + e(C_i) \times \Delta_w, \tag{10}$$

where $e(C_i)$ is calculated as

$$e(C_i) = \begin{cases} 0, & e(I_i) < \bar{e} + std(e), \\ 1, & \bar{e} + std(e) \le e(I_i) < \bar{e} + 2 \times std(e), \\ 2, & \bar{e} + 2 \times std(e) \le e(I_i). \end{cases} \tag{11}$$

The values $\bar{e}$ and $std(e)$ are the mean and standard deviation of sound energy derived from the video. When the sound energy corresponding to I_i is lower than the average energy plus one standard deviation, no emphasis is applied. When the energy is larger than the average energy plus one or two standard deviations, font size is enlarged by one or two Δ_w, which is set as one percent of the width of a page (in terms of pixels). Height of a letter is defined in the same way. The font emphasis process is adopted for keyframes extracted from videos. When generating comics for photo albums, letters are always displayed in default size.

Note that the sound energy corresponding I_i is computed from the average sound energy of the video shot where I_i was extracted from. The audio clip corresponding to this video shot is divided into overlapped audio frames, and the sound energy of each audio frame is calculated as the root mean square value [11]. The energies of all audio frames in the same shot are then averaged to represent the energy of this shot. The values $\bar{e}$ and $std(e)$ mentioned above are calculated from energies of all video shots where keyframes were extracted from.

Given a letter's width and height, the width and height of a rectangle able to include all letters corresponding to I_i are determined. Based on this rectangle, ellipse or other shapes of balloons are generated. In the western reading order, i.e., reading from left to right, and then top to down, the width w_R of the rectangle is determined as

$$w_R = \begin{cases} 0.8w(C_i) & \text{if } \hat{w}_c \times \hat{s}_i > 0.8w(C_i), \\ \hat{w}_c \times \hat{s}_i & \text{else}, \end{cases} \quad (12)$$

where $\hat{s}_i$ is the number of letters corresponding to the keyframe I_i, and $w(C_i)$ is width of the panel C_i. The height h_R of the rectangle is calculated as

$$h_R = \lceil \frac{\hat{w}_c \times \hat{s}_i}{w_R} \rceil \times \hat{h}_c, \quad (13)$$

where $\hat{h}_c$ is height of a letter, which is basically the same as $\hat{w}_c$.

Rectangles showing other languages can be obtained by slight adjustment. For example, an English word consisting of a number of letters cannot be divided into two separate parts and shown in two different lines. When the rectangle is not wide enough to completely show the last English word of a text line, we can divide this word into two parts and append a dash to the end of the first part. The first part is shown in the end of the text line, and the second part is shown in the beginning of the next text line. Another strategy is just showing the whole English word in the beginning of the next line. In this work, we mainly adopt the second strategy to generate English balloons.

Based on the $w_R \times h_R$ rectangle, a speech balloon with appropriate shape is generated. We construct a list in advance to store words commonly used to represent violence and excitation. If at least one of spoken words matches with the words in the list, this balloon is claimed to present violent emotion, and general emotion otherwise. Generally most balloons are categorized as general emotion, and are displayed as the minimum ellipses covering the rectangles. For subtitle containing excitation or violence words, a predefined jagged edged balloon is resized to include the rectangle to show emphasis.

B. Balloon Placement

After generating balloons, the next step is to determine their positions. The most important guideline for balloon placement is not to occlude important regions in images [19], which is quite natural and had been widely adopted to place balloons [6][25][26]. However, in most previous works, balloons positions in different panels were determined separately. The way to place balloons in different panels lead the reading order. Therefore, we argue that positions of balloons in different panels should be jointly determined so that the best reading experience can be provided.

In [20], we treated balloons in the same page as a whole, and formulated balloon placement as an optimization problem. Motivated by the formulation of particle swarm optimization (PSO), a balloon in a panel was viewed as a *particle*, and the set of particles in the same page was viewed as a *swarm*. After randomly initializing positions in each panel, the PSO algorithm was used to adjust positions of particles in multiple panels so that global optimal (from the aspect of the whole page) and local optimal (from the aspect of one panel) balloon positions can be determined simultaneously. However, in [20] only one balloon was allowed in each panel. When a character consecutively speaks many sentences, similar keyframes would be displayed in nearby panels, and significantly annoy viewers. Here we extend the framework to allow multiple balloons in a panel.

Assume that G balloons $\mathcal{B} = \{B_1, ..., B_G\}$ are to be respectively placed into K panels of a page. The optimal positions are determined by jointly considering the following factors:

- Balloons should not overlap with ROIs in images.
- Balloons should be placed as close as ROIs in images.
- When there are multiple balloons in a panel, the sentences spoken earlier should be placed closer to the left-top corner of the panel. This is to maintain correct reading order.
- Balloons should not overlap with each other.
- Reading trajectory should be built so that the reading order is not only correct but also vivid.

These factors are quantitatively evaluated to form the objective function for the PSO algorithm. Details of these factors are described as follows.

Overlapping between balloon and ROI. The degree of overlap V_1 between balloon and ROI is calculated as

$$V_1 = \sum_{k=1}^{K} \sum_{i=1}^{G} \frac{R_k \bigcap B_i}{R_k}, \quad (14)$$

where R_k denotes the area of ROI in the kth panel, $R_k \bigcap B_i$ means the area of the intersection (if any) between the ROI R_k and the balloon B_i. The value V_1 should be as small as possible.

Close to ROI. The overall distance V_2 between balloon and its closest ROI is evaluated as

$$V_2 = \frac{\sum_{k=1}^{K} \sum_{i=1}^{G} d(r_k, e_i)}{\sum_{k=1}^{K} \sum_{i=1}^{G} d(r_k, o_i)}, \quad (15)$$

where $d(r_k, o_i)$ is the spatial distance from the centroid o_i of the balloon B_i to the ROI centroid r_k in the kth panel. The value $d(r_k, e_i)$ is the spatial distance from the balloon centroid o_i to the closet point on the boundary of the ROI R_k. The value V_2 is smaller if balloons are placed closer to ROIs.

Order of balloons. We are able to know the temporal order of balloons according to the subtitle information. In the fashion of reading from left to right and then top to bottom, the balloon spoken earlier should be placed closer to the left-top corner of a panel. Assume that there are J balloons $\mathcal{B}_k = \{B_k^1, B_k^2, ..., B_k^J\}$ to be put into the kth panel, and the spoken order is B_k^1, B_k^2, etc. Let $b_k^j(x)$ and $b_k^j(y)$ denote the x and y coordinates of the centroid of the balloon B_k^j, $j = 1, ..., J$, respectively. We calculate the horizontal displacement between two consecutively spoken balloons as $\Delta x_k^j = \frac{b_k^{j+1}(x) - b_k^j(x)}{\max(C_W, C_H)}$, where C_W and C_H are width and height of the targeted panel, respectively. Similarly, the vertical displacement is calculated as $\Delta y_k^j = \frac{b_k^{j+1}(y) - b_k^j(y)}{\max(C_W, C_H)}$. When the balloon B_k^{j+1} is appropriately placed at the right bottom of B_k^j, the value $\Delta_k^j = \Delta x_k^j + \Delta y_k^j$ is larger than zero. The degree of how balloons placed in a correct order in the kth panel is calculated by $\bar{\Delta}_k = 1 - \frac{1}{J-1} \sum_{j=1}^{J-1} \Delta_k^j$. The overall degree V_3 for all the K panels in a page is then measured by

$$V_3 = \frac{1}{K} \sum_{k=1}^{K} \bar{\Delta}_k. \quad (16)$$

The value V_3 is smaller if later balloons are placed at the right bottom of earlier balloons.

Overlapping between balloons. For the J balloons $\mathcal{B}_k = \{B_k^1, B_k^2, ..., B_k^J\}$ to be put into the kth panel, the degree of overlapping between two balloons B_i and $B_{i'}$ is measured by

$X_{ii'} = \frac{B_i \cap B_{i'}}{B_i \cup B_{i'}}$. The average degree of overlapping in the kth panel is calculated as $\bar{\Lambda}_k = \frac{1}{Z} \sum_{\substack{1 \le i \le J-1 \\ i < i' \le J}} X_{ii'}$, where $Z = \binom{J}{2}$ is a normalization factor. The overall degree of overlapping V_4 for all K panels is defined as

$$V_4 = \frac{1}{K} \sum_{k=1}^{K} \bar{\Lambda}_k. \quad (17)$$

The value V_4 is smaller if balloons have less overlapping.

The four factors mentioned above are about balloons inside single panels. On the other hand, the fifth factor is concerned about the reading trajectory across panels.

Reading trajectory. When reading a comic page, we usually read the balloons in the first panel, and then the main character (ROI) in this panel, and then move to the balloons in the second panel, and so on. According to [27], the reading tempo in comics is embedded in the reading trajectory. To build lively reading tempo, therefore, we would like to place balloons so that the reading trajectory turns a lot. Let $Y_k = \{Y_k^1, ..., Y_k^{J+1}\}$ be the set of J balloons and one ROI in the kth panel, i.e., $Y_k^j \in \{\mathcal{B}_k, R_k\}$. Suppose that the elements in Y_k are sorted so that Y_k^{j+1} is on the right bottom of Y_k^j and is the nearest neighbor of Y_k^j (measured by Δ_k^j mentioned previously). The reading trajectory factor $\bar{\Upsilon}_k$ within the kth panel is calculated as

$$\bar{\Upsilon}_k = \frac{1}{J-1} \sum_{j=1}^{J-1} |\langle \boldsymbol{u}_j, \boldsymbol{u}_{j+1} \rangle|, \quad (18)$$

where $\boldsymbol{u}_j$ is the vector from the centroid of Y_k^j to the centroid of Y_k^{j+1}. The notation $\langle \boldsymbol{u}_j, \boldsymbol{u}_{j+1} \rangle$ denotes the Euclidean inner product between two vectors. Therefore, the value $\bar{\Upsilon}_k$ denotes the average absolute inner product between balloons (ROI) within the kth panel.

The reading trajectory factor z_k between the kth panel and the $(k+1)$th panel is calculated as

$$z_k = |\langle \boldsymbol{u}_J, \boldsymbol{v}_k \rangle|, \quad (19)$$

where $\boldsymbol{u}_J$ is the vector from Y_k^J to Y_k^{J+1}, i.e., the vector between the last two balloons in the kth panel. $\boldsymbol{v}_i$ is the vector from the last balloon of the kth panel to the first balloon (or ROI) of the $(k+1)$th panel.

The overall reading trajectory factor V_5 is defined as

$$V_5 = \frac{1}{K} \sum_{k=1}^{K} \bar{\Upsilon}_k + \frac{1}{K-1} \sum_{k=1}^{K-1} z_k. \quad (20)$$

The value V_5 is smaller if vectors between reading entities (balloons or ROIs) are approximately orthogonal.

Integration. Finally, the five factors are linearly combined to jointly evaluate the objective function value E:

$$E = \mu_1 V_1 + \nu_2 V_2 + \nu_3 V_3 + \nu_4 V_4 + \nu_5 V_5, \quad (21)$$

where $\nu_1, ..., \nu_5$ are weights for combination and $\sum_{i=1}^{5} \nu_i = 1$. With this objective function, the optimal balloon positions in the same page are determined in an integrated optimization framework, and can be efficiently solved.

Particle swarm optimization. Finding the best balloon placement is viewed as finding the best set of positions that causes the minimum objective function value in the parameter space constituted by all possible positions (in the representation of x-coordinate and y-coordinate). This problem can be intuitively mapped to the one efficiently solved by the particle swarm optimization algorithm (PSO) [24]. The PSO algorithm is an iterative randomized search algorithm that updates a population of candidate solutions at each iteration. Initially we randomize one particle for each balloon. The ith particle is represented as $\boldsymbol{p}_i = (x_i, y_i, v_i, \boldsymbol{d}_i)$, where x_i and y_i respectively denotes the x-coordinate and y-coordinate of the ith balloon, v_i is this particle's moving speed, and $\boldsymbol{d}_i$ is its moving direction. The set of particles forms a swarm (population) indicating positions of all balloons in a page. At the tth iteration, $t = 1, 2, ..., T$, we evaluate the objective function value of the current swarm by eqn. (21), and keep track of the best-so-far positions encountered by the entire population. At the $(t+1)$ th iteration, particles move towards updated directions with updated velocities, which are dynamically updated according to their best-so-far positions. The same procedure repeats until the stop condition meets, and the set of positions giving the optimal objective function value is found. Please refer to [24] for details of the particle swarm optimization algorithm. In this work, the iterative process stops when 200 iterations have completed.

Figure 7 illustrates how different factors affect final placement results. Figure 7(a) and Figure 7(c) are placement results if all factors are jointly considered. Figure 7(b) is the placement result without taking overlapping between balloons into account. In this case, balloons are occluded by other balloons, which significantly harms readability of the generated comics. Figure 7(d) is the placement result without taking overlapping between balloon and ROI into account. The important object, human face in this example, is occluded by balloons, causing the most unwilling result. Figure 7(e) shows ROIs of images shown in Figure 7(c) for reference.

VII. Evaluation

We start the evaluation from transforming animation videos into comics. Table 1 shows information of the evaluation dataset, which contains six animation clips with subtitles. The evaluation is divided into two parts: objective evaluation where automatically generated comics were quantitatively compared with manually edited comics, and subjective evaluation where subjects were asked to answer questions in questionnaires.

A. Objective Evaluation

We first attempt to quantify the difference between automatically generated comics and manually edited comics. We invited eight semi-professional editors, named from A to H, who are members of the comic club in our university and are quite familiar with comics design or even draw comics by themselves. From easy to difficult, these editors were asked to conduct the following three tasks:

- Task 1: Given a comic page where everything (layout and images) except for balloons has been composed, editors were asked to place balloons in panels. This task was designed to evaluate location difference between automatically placed balloons and manual ones.
- Task 2: Given a comic page and the image set to be displayed, editors were asked to design a layout to display these images. This task was designed to evaluate the difference between automatically selected layout and manual ones.

(a) (b) (c) (d) (e)

Figure 7. Comparison of balloon placement considering different factors. (a)(c) The placement results if all factors are jointly considered. (b) The placement result if overlapping between balloons is not taken into account. (d) The placement result if overlapping between balloons and ROIs is not taken into account. (e) The ROIs corresponding to each image in (c). The input artwork is from "Popeye: Spree Lunch (1957)".

Table 1. Information of the animation evaluation dataset.

ID	Name	Length	#shots	#subtitle
1	EVA_1*	9'37''	166	124
2	EVA_2	7'57''	176	99
3	Chu_1†	8'03''	171	148
4	Chu_2	10'02''	173	182
5	Summer Wars_1‡	9'36''	168	143
6	Summer Wars_2	9'13''	207	90

*EVA = "Neon Genesis Evangelion" (©Hideaki Anno/Gainax & Tatsunoko), a Japanese science-fantasy animation series

† Chu = "Love, Chunibyo & Other Delusions!" (©Tatsuya Ishihara/Kyoto Animation), a Japanese light novel animation series

‡ Summer Wars = "Summer Wars" (©Mamoru Hosoda/ Madhouse), a Japanese science fiction film

Table 2. Average placement difference between automatically placed balloons and manually edited balloons.

EVA_1	Avg. Dist.	Ratio	Chu_2	Avg. Dist.	Ratio
Sys. vs. A	132.7	0.17	Sys. vs. A	120.3	0.16
Sys. vs. B	119.4	0.15	Sys. vs. B	105.7	0.14
Sys. vs. C	120.9	0.15	Sys. vs. C	135.6	0.16
Sys. vs. D	140.4	0.18	Sys. vs. D	148.7	0.18
Sys. vs. E	123.6	0.15	Sys. vs. E	93.0	0.12
Sys. vs. F	109.2	0.14	Sys. vs. F	145.1	0.18
Sys. vs. G	99.1	0.12	Sys. vs. G	95.6	0.12
Sys. vs. H	112.3	0.14	Sys. vs. H	118.7	0.15
A vs. C	94.5	0.12	A vs. C	116.2	0.15
B vs. D	121.9	0.15	B vs. D	120.9	0.15

Table 3. Performance comparison between our method and [26], in terms of average placement difference.

EVA_1	Avg. Dist.	Ratio	Chu_2	Avg. Dist.	Ratio
[26] vs. editors	138.1	0.17	[26] vs. editors	121.2	0.15
Our Sys. vs. editors	120.3	0.15	Our Sys. vs. editors	120.3	0.15

- Task 3: Given a temporal image sequence and each image's associated spoken words (if any), editors were asked to freely compose appropriate number of comic pages, which includes processes of page allocation, layout design, image cropping and resizing, and speech balloon placement. This task was designed mainly to evaluate the difference between automatic page allocation results and manual allocation results.

In Tasks 1 and 2, each editor was given thirty comic pages, and the corresponding results were recorded. In Task 3, each editor was given 100 images as the materials to freely compose comic pages. We recorded how they allocated pages, designed layouts, and placed balloons.

Difference in balloon placement. Spatial difference between automatically placed balloons and manually edited balloons is measured by the Euclidean distance between balloon centroids. Table 2 shows the average Euclidean distances in terms of pixels between the system's results and each editor's edited results, respectively based on the materials extracted from "EVA_1" and "Chu_2". Normalized by width of the comic page, placement errors are averagely from 12% to 18%, indicating that the system's results are consistently similar to manual ones.

We also quantitatively compare a subset of editors' results, as shown in the last two rows of Table 2. Average differences between editors are from 12% to 15%. Comparing these results with that in the first eight rows, we see that differences between editors, and differences between editors and our proposed system, are similar. This also verifies effectiveness of the proposed speech balloon component.

Table 3 shows performance comparison between our method and [26], which was one of the few works on placing multiple balloons in an image. It also formulated balloon placement as an optimization problem, and a heuristic algorithm was designed to ensure a reasonable reading order in a single image. Based on ROI detection results, we implemented the method in [26] and separately placed balloons for each image. As can be seen in Table 3, the average displacement difference between our system's results and manually edited results is smaller than that obtained by [26]. Especially for "EVA_1" superiority of our method is statistically significant (p value < 0.005).

Difference in layout selection. We calculate the Hamming distance between binary vectors representing panel arrangement. Definition of a binary vector is illustrated in Figure 4(a). Table 4 shows accumulated and average distances between manual and system's results calculated based on thirty comic pages. Overall, the Hamming distance is from 0.90 to 1.23 per page. The last two rows show average distances between a subset of editors are from 0.83 to 1.13. Because panel arrangement involves personal preference and content understanding, there is still a gap between automatically selected layouts and manual ones.

Difference in page allocation. We quantitatively evaluate page allocation results from two perspectives: number of allocated pages and boundaries of pages. Figure 8 shows comparisons between automatic allocation and manual allocation based on "Chu_1" and "Summer Wars_2". The

horizontal axis denotes IDs of images, and the vertical axis denotes the number of allocated pages. We see that the result of automatic allocation is similar to most editors' results. For "Chu_1", except for the editor B, the number of allocated pages between our system's and editors' is less than 4. From this figure we can see that page allocation results of different editors could be significantly different, indicating that this task is very subjective even the same materials are given.

From the perspective of page boundary, we view an editor's allocation result as the ground truth, and evaluate overlap between automatic allocation and the ground truth by calculating the *purity* value [28]. A purity value ranges from 0 to 1, and a larger purity value means automatic allocation is more similar to manual allocation. Table 5 shows comparison results. Generally, purity values are around 0.45 on average for the two animation videos, with standard deviation around 0.08. These results again show that the page allocation task is quite subjective, and despite this our system's results are moderately correlated with manual results.

One may notice that the evaluated videos in Tables 2 to 5 are different. In fact, we purposely designed such experimental settings. Each editor was asked to conduct three tasks. In Task 1, we gave them composite comic pages, and only asked them to place speech balloons. In Task 2, we only gave them page allocation results and asked them to design the best layout. If visual content for Task 1 and Task 2 are the same, the layout given in Task 1 may influence the choice of layout design when he/she conducted Task 2. Same situation occurs between Task 2 and Task 3. Therefore, we separate the six evaluation videos into three groups for three Tasks, respectively. Because manual results were separately obtained, Tables 2 to 5 only show evaluation results corresponding to three separate groups.

Table 4. Accumulated and average Hamming distances between automatically selected layouts and manually edited layouts.

Summer War_1	Acc. Dist.	Avg. Dist	EVA_2	Acc. Dist.	Avg. Dist
Sys. vs. A	32	1.07	Sys. vs. A	27	0.90
Sys. vs. B	32	1.07	Sys. vs. B	35	1.17
Sys. vs. C	33	1.10	Sys. vs. C	35	1.17
Sys. vs. D	35	1.17	Sys. vs. D	37	1.23
Sys. vs. E	27	0.90	Sys. vs. E	29	0.97
Sys. vs. F	33	1.10	Sys. vs. F	24	0.80
Sys. vs. G	30	1.00	Sys. vs. G	32	1.07
Sys. vs. H	37	1.23	Sys. vs. H	32	1.07
A vs. C	25	0.83	A vs. C	34	1.13
B vs. D	27	0.90	B vs. D	28	0.93

Table 5. Purity between automatic allocation and manual allocation.

Chu_1	Purity	Summer War_2	Purity
Sys. vs. A	0.49	Sys. vs. A	0.54
Sys. vs. B	0.52	Sys. vs. B	0.49
Sys. vs. C	0.44	Sys. vs. C	0.48
Sys. vs. D	0.37	Sys. vs. D	0.41
Sys. vs. E	0.37	Sys. vs. E	0.31
Sys. vs. F	0.66	Sys. vs. F	0.45
Sys. vs. G	0.41	Sys. vs. G	0.42
Sys. vs. H	0.44	Sys. vs. H	0.41
A vs. C	0.41	A vs. C	0.60
B vs. D	0.30	B vs. D	0.39

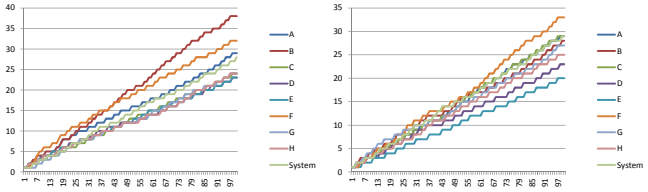


Figure 8. The relationship between number of allocated pages (vertical axis) and the image sequences. Left: Comparison between automatic allocation and manual allocation based on "Chu_1". Right: Comparison based on "Summer Wars_2".

B. *Subjective Evaluation*

We conducted subjective evaluation from various perspectives to demonstrate effectiveness and limit of the proposed method. Fifteen subjects (graduate students major in computer science, familiar with animation and comics, excluding the editors mentioned in the previous section) were invited in the following subjective experiments.

Influence of speech balloon. We first show the influence of speech balloons on comics-based storytelling, comparing with our previous work where the language ingredient (speech balloon) was missing [7]. Each subject was shown the comic pages generated by this work and by [7]. To show the power of comics-based storytelling, storyboards generated by the KMP player [5] were also shown to each subject. In storyboards, keyframes are presented in an array manner, and the perspective and language ingredients of storytelling are missing. After watching comic/storyboard presentations, each subject was asked to give a score from 1 to 5 (larger means better) from four aspects: *comprehensibility* (the degree about how a subject understands the presentation), *interestingness* (the degree of fun when reading the presentation), *coverage* (the degree of how the presentation covers important information of the original animation), and *layout* (the degree of appropriateness of presentation layout).

Figure 9 shows the mean opinion scores from the four aspects. We can clearly see that comics with balloons yield much higher comprehensibility and coverage, which mainly accredits to appropriate speech balloons presentation and placement. In terms of interestingness, the superiority of comics-based storytelling confirms that comics provide funnier presentation than storyboards. Appropriateness of layout arrangement depends on individual's preference, but from this figure we still can see superiority of the proposed method.

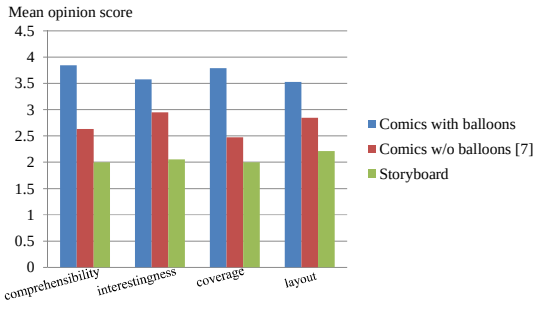


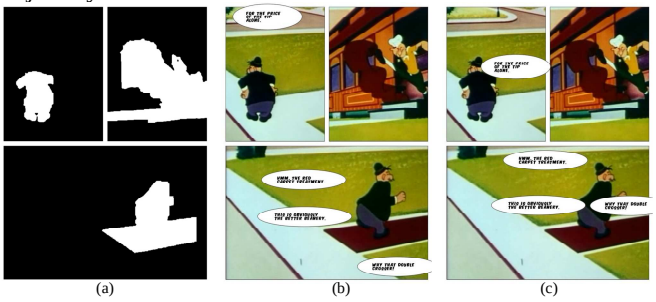
Figure 9. Mean opinion scores from the aspects of comprehensibility, interestingness, coverage, and layout.

[5] http://www.kmpmedia.net/

Figure 10. Comparison between a single-balloon version (top) and a multiple-balloon version (bottom). The input artwork is from "Popeye: Spree Lunch (1957)".

Single balloon vs. multiple balloons. Placing multiple balloons in a panel makes balloon placement more challenging, but facilitates more compact representation. Figure 10 shows comparison between a single-balloon presentation and a multiple-balloon presentation. If only one balloon is allowed in a panel, multiple panels will be needed to show multiple sentences spoken in the same scene, and thus more pages are needed. We can easily perceive redundancy in the single-balloon version. On the other hand, the third page of the multiple-balloon version compactly conveys content covered by the third, the fourth, and half of the fifth pages of the single-balloon version. The newly proposed multiple-balloon placement obviously facilitates compact comic presentation matching with our expectation in a professional comic book.

Comparison with an existing speech balloon placement work. Figure 11 shows a sample comparison between our method and [26]. Because balloon positions in images of a comic page are not jointly considered, results shown in Figure 11(c) are less attractive because of the relatively stiff reading trajectory.



(a) (b) (c)

Figure 11. Comparison between the proposed balloon placement method and [26]. (a) The detected ROIs; (b) results of the proposed method; (c) results of Chun et al. [26].

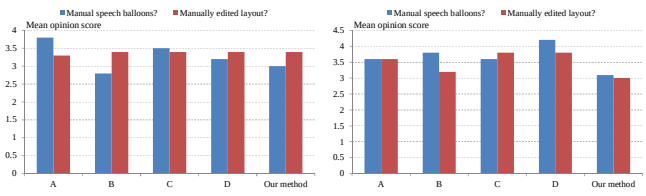


Figure 12. Experimental results showing how subjects distinguish automatic balloon placement and page allocation from manual ones. Left: results obtained based on Chu_1; right: results obtained based on Summer War_2.

Comparing automatic results with manual results. Based on the materials collected from Task 1 and Task 3 mentioned in the previous section, we randomly selected five manually-edited comic pages from editors A's, B's, C's, and D's results and five automatically generated comic pages, and randomly showed these pages to subjects. Each subject was asked to give a score from 1 to 5 to each page by judging the following questions:

- Q1: To what degree do you think the speech balloons in this page are manually placed?
- Q2: To what degree do you think the page is manually allocated?

Figure 12 shows how subjects distinguish automatic balloon placement and page allocation from manual ones. For "Chu_1", subjects didn't clearly distinguish automatic results from manual ones, indicating the proposed system generates comic pages very similar to manually edited pages. For "Summer Wars_2", subjects are relatively easier to recognize, but the difference between our result and manual ones is not much.

Based on the randomly selected comic pages, we also asked subjects to judge whether the results of balloon placement and page allocation are reasonable.

- Q3: To what degree do you think the speech balloons in this page are reasonably placed?
- Q4: To what degree do you think the page is reasonably allocated?
- Q5: To what degree do you like this page?

Figure 13 shows that, in terms of reasonability, there is no significant difference between automatic results and manual ones. In particular, for "Chu_1", the overall score of our system is slightly higher than that of editor B's results; but for "Summer Wars_2", our system yields slightly worse performance in terms of reasonability, which echoes the trend shown in Figure 12.

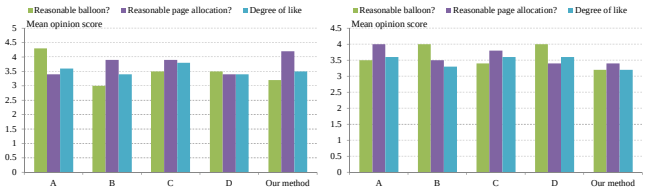


Figure 13. Experimental results showing how subjects think the results of automatic balloon placement and page allocation are reasonable. Left: results obtained based on "Chu_1"; right: results obtained based on "Summer Wars_2".

Development platform and execution time. The development platform is a PC with an Intel Core i7-870 2.93 GHz CPU and with 14GB RAM. This system was developed by C/C++ programming language associated with the OpenCV library. Generally, transforming a 10-min animation video into comic pages takes about 20 minutes. Around half of execution time was spent by the keypoint-based approach [21] mentioned in Section III to eliminate redundant keyframes.

C. Comics-based Storytelling for Photo Albums and Movies

Although we mainly show storytelling results for animation videos, it is worth noting that the proposed method is generally applicable to any temporal image sequence. Given a photo

album consisting of photos with timestamps and annotations, we can employ the same process to generate comic-style presentation, with the associated annotations displayed in speech balloons. Figure 14 shows some sample comic pages generated from a photo album[6].

To verify effectiveness of the proposed system to generate comics for photo albums, the three tasks mentioned in Section VII.A were also conducted by the editor B, and three objective evaluations with respect to speech balloon placement, layout selection, and page allocation were also conducted. For the same reason mentioned in the end of Section VII.A, we asked the editor B to conduct three tasks for three different albums, and the results were separately used in three evaluations. Tables 6 to 8 respectively show difference between the system's results and manually edited results. Comparing these tables with Tables 2, 4 and 5, we clearly see that the proposed system can be generalized to photo albums with similar or even better performance (Table 8, page allocation) performance. The reason of better page allocation result in Table 8 may be twofold. First, for photo albums we get rid of the noise or inaccurate results caused by keyframe selection. Second, photos in these albums are all well photographed and with clear themes, making our system and the editor easily divide them into groups.

Similarly, given a sequence of keyframes extracted from a movie, the same process can be employed to generate a comics-based presentation, with the associated subtitles as speech balloons. Figure 15 shows sample comic pages generated from the movie "The Boy in the Plastic Bubble (1976)".

Table 6. Average placement difference between automatically placed balloons and manually edited balloons for the comics generated from a photo album[7].

	System vs. Ground truth
Avg. Dist.	148.59
Ratio	0.18

Table 7. Accumulated and average Hamming distances between automatically selected layouts and manually edited layouts for the comics generated from a photo album[8].

	System vs. Ground truth
Acc. Dist.	17
Avg. Dist.	0.85

Table 8. Purity between automatic allocation results and manual allocation results for the comics generated from a photo album[9].

	System vs. Ground truth
Purity	0.69

[6] Image source: http://www.flickr.com/photos/rose_des_vents/. We totally downloaded 36 photos and generated 12 comic pages. Only three sample pages are shown here due to space limitation.

[7] Image source: https://www.flickr.com/photos/next-blessing/

[8] Image source: https://www.flickr.com/photos/jimmaybones/page6/

[9] Image source: https://www.flickr.com/photos/snoweeowl

Figure 14. Sample comic pages generated from a photo album.

Figure 15. Sample comic pages generated from the movie "The Boy in the Plastic Bubble (1976)" in the public domain.

VIII. Conclusion

We have presented a system that automatically transforms temporal image sequences into comics-based storytelling. Our first contribution is to formulate page allocation as a labeling problem and find the optimal solution by the genetic algorithm. The second contribution is representing distributions of image importance and panel importance by vectors, and finding the best match between them by inner product operation. As the third contribution, we consider multiple balloons in single panels and the interrelationship between balloons across panels, and cast this problem into a framework based on the particle swarm optimization. Finally, we design a series of subjective and objective evaluations to demonstrate effectiveness and superiority of the proposed system from various perspectives. Overall, we propose thorough and systematic approaches to generate comics-based storytelling.

In the future, user interaction can be added to enable interactive modification. Moreover, existing ROI analysis techniques are mainly designed for natural images rather than animation, and thus the system has limited information of important objects. Advanced ROI analysis techniques for animation may be studied in the future.

Acknowledgement

The authors would like to thank the reviewers for insightful comments and suggestions. The work was partially supported by the Ministry of Science and Technology in Taiwan under the grants NSC101-2221-E-194-055-MY2 and MOST103-2221-E-194-027-MY3.

References

[1] R. Kosara and J. Mackinlay, "Storytelling: The next step for visualization," *Computer*, vol. 46, no. 5, pp. 44-50, 2013.

[2] E. Segel and J. Heer, "Narrative visualization: Telling stories with data," *IEEE TVCG*, vol. 16, no. 6, pp. 1139-1148, 2010.

[3] K.-L. Ma, I. Liao, J. Frazier, H. Hauser, and H.-N. Kostis, "Scientific storytelling using visualization," *IEEE CG&A*, vol. 32, no. 1, pp. 12-19, 2012.

[4] J.-C. Chen, W.-T. Chu, J.-H. Kuo, C.-Y. Weng, and J.-L. Wu, "Tiling slideshow," In *Proc. of ACM MM*, pp. 25-34, 2006.
[5] C.-J. Chan, R. Thawonmas, and K.-T. Chen, "Automatic storytelling in comics: A case study on world of warcraft," In *Proc. of ACM CHI*, 2009.
[6] M. Wang, R. Hong, X.-T. Yuan, S. Yan, and T.-S. Chua. "Movie2Comics: Towards a lively video content presentation," *IEEE TMM*, vol. 14, no. 3, pp. 858-870, 2012.
[7] W.-T. Chu and H.-H. Wang, "Enabling portable animation browsing by transforming animations into comics," In *Proc. of ACM International Workshop on Interactive Multimedia on Mobile and Portable Devices*, pp. 3-8, 2012.
[8] M.-H. Wen, R. Thawonmas, and K.-T. Chen, "Pomics: A computer-aided storytelling system with automatic picture-to-comics composition," In *Proc. of 2012 Conference on Technologies and Applications of Artificial Intelligence*, pp. 314-318, 2012.
[9] M. Austin, *Useful Fictions: Evolution, Anxiety, and the Origins of Literature*. University of Nebraska Press, 2011.
[10] M. Toyoura, M. Kunihiro, and X. Mao, "Film comic reflecting camera-works," In *Proc. of MMM*, LNCS 7131, pp. 406-417, 2012.
[11] Y. Wang, Z. Liu, and J.-C. Huang, "Multimedia content analysis: Using both audio and visual cues," *IEEE Signal Processing Magazine*, vol. 17, no. 6, pp. 12-36, 2000.
[12] M. Balabanovic, L.L. Chu, and G.J. Wolff, "Storytelling with digital photographs," In *Proc. of ACM CHI*, pp. 564-571, 2000.
[13] D. Joshi, J.Z. Wang, and J. Li, "The story picturing engine – a system for automatic text illustration," *ACM TOMCCAP*, vol. 2, no. 1, pp. 68-89, 2006.
[14] G. Ciocca and R. Schettini, "Dynamic storyboards for video content summarization," In *Proc. of ACM International Workshop on Multimedia Information Retrieval*, pp. 259-268, 2006.
[15] Y. Wang, T. Mei, J. Wang, and X.-S. Hua, "Dynamic video collage," In *Proc. of MMM*, *LNCS* 5916, pp. 793-795, 2010.
[16] Y. Pritch, A. Rav-Acha, and S. Peleg, "Nonchronological video synopsis and indexing," *IEEE TPAMI*, vol. 30, no. 11, pp. 1971-1984, 2008.
[17] M.-M. Cheng, G.-X. Zhang, N.J. Mitra, X. Huang, S.-M. Hu, "Global contrast based salient region detection," In *Proc. of CVPR*, pp. 409-416, 2011.
[18] S. Goferman, L. Zelnik-Manor, and A. Tal, "Context-aware saliency detection," *IEEE TPAMI*, vol. 34, no. 10, pp. 1915-1926, 2012.
[19] G.S. Millidge. *Comic Book Design: The Essential Guide to Designing Great Comics and Graphic Novels*. Watson-Guptill, 2009.
[20] W.-T. Chu and C.-H. Yu, "Optimized speech balloon placement for automatic comics generation," In *Proc. of ACM International Workshop on Interactive Multimedia on Mobile and Portable Devices*, pp. 1-6, 2013.
[21] G. Guan, Z. Wang, S. Lu, J. Da Deng, and D.D. Feng, "Keypoint-based keyframe selection," *IEEE TCSVT*, vol. 23, no. 4, pp. 729-734, 2013.
[22] Z. Rasheed and M. Shah, "Scene detection in Hollywood movies and tv shows," In *Proc. of CVPR*, vol. 2, pp. 343-348, 2003.
[23] J.C. Platt, M. Czerwinski, and B.A. Field, "PhotoTOC: automatic clustering for browsing personal photographs," In *Proc. of PCM*, vol. 1, pp. 6-10, 2003.
[24] E.K.P. Chong and S.H. Zak, *An Introduction to Optimization*, Wiley, 2013.
[25] T. Sawada, M. Toyoura, and X. Mao, "Film comic generation with eye tracking," In *Proc. of MMM*, pp. 467-478, 2013.
[26] B.-K. Chun, D.-S. Ryu, W.-I. Hwang, and H.-G. Cho, "An automated procedure for word balloon placement in cinema comics," In *Proc. of International Conference on Advances in Visual Computing*, vol. 2, pp. 576-585, 2006.
[27] H. Tobita, "Comic engine: Interactive system for creating and browsing comic books with attention cuing," In *Proc. of the International Conference on Advanced Visual Interfaces*, pp. 281-288, 2010.
[28] A. Vinciarelli and S. Favre, "Broadcast news story segmentation using social network analysis and hidden Markov models," In *Proc. of ACM MM*, pp. 261-264, 2007.
[29] R. Hong, X.-T. Yuan, M. Xu, M. Wang, S. Yan, and T.-S. Chua. "Movie2Comics: A feast of multimedia artwork," In *Proc. of ACM MM*, pp. 611-614, 2010.
[30] J. Calic, D.P. Gibson, and N.W. Campbell, "Efficient layout of comic-like video summaries," *IEEE TCSVT*, vol. 17, no. 7, pp. 931-936, 2007.
[31] L. Herranz, J. Calic, J.M. Martinez, and M. Mrak, "Scalable comic-like video summaries and layout disturbance," *IEEE TMM*, vol. 14, no. 4, pp. 1290-1297, 2012.
[32] L. Herranz, H. Liu, and S. Jiang, "Effective comic-like representations with embedded regions of interest," *LNCS*, vol. 7674, pp. 464-475, 2012.
[33] Y. Cao, A.B. Chan, and R.W.H. Lau, "Automatic stylistic manga layout," *ACM TOG*, vol. 31, no. 6, Article 141, 2012.
[34] E. Myodo, S. Ueno, K. Takagi, and S. Sakazawa, "Automatic comic-like image layout system preserving image order and important regions," In *Proc. of ACM MM*, pp. 795-796, 2011.
[35] H. Tobita, "DigestManga: Interactive movie summarizing through comic visualization," In *Proc. of CHI*, pp. 3751-3756, 2010.
[36] A. Girgensohn, "A fast layout algorithm for visual video summaries," In *Proc. of ICME*, vol. 2, pp. 77-80, 2003.
[37] L. Troiano, C. Birtolo, R. Armenise, and G. Cirillo, "Web form page in mobile devices: Optimization of layout with a simple genetic algorithm," In *Proc. of ICEIS*, vol. 5, pp. 118-123, 2009.
[38] G.B. Brabrand, "Dynamic layout optimization for newspaper web sites using a controlled annealed genetic algorithm," Master Thesis, Gjovik University College, 2008.
[39] J.H. Ricketts, "Optimization of newspaper display advertising layout: Using a genetic algorithm with a novel rearrangement step," Master Thesis, Royal Melbourne Institute of Technology, 1997.
[40] S. Uchihashi, J. Foote, A. Girgensohn, and J. Boreczky, "Video manga: Generating semantically meaningful video summaries," In *Proc. of ACM MM*, pp. 383-392, 1999.
[41] M.M. Yeung and B.-L. Yeo, "Video visualization for compact presentation and fast browsing of pictorial content," *IEEE TCSVT*, vol. 7, no. 5, pp. 771-785, 1997.

Wei-Ta Chu received the B.S. and M.S. degrees in Computer Science from National Chi Nan University, Taiwan, in 2000 and 2002, and received the Ph.D. degree in Computer Science from National Taiwan University, Taiwan, in 2006. He is now an Associate Professor in the Department of Computer Science and Information Engineering, National Chung Cheng University, Taiwan. His research interests include digital content analysis, multimedia indexing, digital signal process, and pattern recognition.

He won the Best Full Technical Paper Award in ACM Multimedia 2006. He was awarded the Distinguished Alumni Award presented by National Chi Nan University in 2014, the K. T. Li Young Researcher Award presented by Institute of Information & Computing Machinery in 2012, and the Young Faculty Awards presented by National Chung Cheng University in 2011. He was a visiting scholar at Digital Video & Multimedia Laboratory, Columbia University, from Jul. to Aug. 2008. He organized ACM Workshop on Crowdsourcing for Multimedia in 2012 and 2013 at ACM Multimedia Conference.

Chia-Hsiang Yu received the B.S. and M.S degrees in Computer Science from National Chung Cheng University, Taiwan, in 2012 and 2014. His research interests include digital content analysis and multimedia systems.

Hsin-Han Wang received the B.S. and M.S degrees in Computer Science from Dayeh University and National Chung Cheng University, Taiwan, in 2010 and 2012. His research interests include digital content analysis and multimedia systems.