DynamicManga: Animating Still Manga via Camera Movement

Ying Cao, Xufang Pang, Antoni B. Chan, *Member, IEEE*, and Rynson W.H. Lau, *Senior Member, IEEE*

***Abstract*—We propose a method for animating still manga imagery through camera movements. Given a series of existing manga pages, we start by automatically extracting panels, comic characters and balloons from the manga pages. Then, we use a data-driven graphical model to infer per-panel motion and emotion states from low-level visual patterns. Finally, by combining domain knowledge of film production and characteristics of manga, we simulate camera movements over the manga pages, yielding an animation. The results augment the still manga contents with animated motion that reveals the mood and tension of the story, while maintaining the original narrative. We have tested our method on manga series of different genres, and demonstrated that our method can generate animations that are more effective in storytelling and pacing, with less human efforts, as compared with prior works. We also show two applications of our method, mobile comic reading and comic trailer generation.**

***Index Terms*—Comics, 2D animation, semantic estimation, camera movement**

I. Introduction

DURING the last few decades, manga, i.e., Japanese comics, has grown to be one of most popular storytelling mediums, consumed by an increasingly larger number of audiences across the world. However, as our viewing experience is shifting to smartphones and tablets, the way to consume comics remains traditional, mainly turning digital comic pages that are converted from print. In an attempt to evolve the medium of comic storytelling, new digital comic formats, such as Infinite Comics and DC Squared [6], have been unveiled recently, which introduces dynamics to static comics by moving comic characters and balloons to tell a story. However, they are often manually created by professional artists using special-purpose tools (e.g., Manga25 [21]) with a lot of authoring efforts. The high production cost makes it impractical to deploy such dynamic formats at a large scale.

Our goal of this work is to create a *low-cost* system that enables *arbitrary users* to produce a dynamic format of manga with compelling motion from existing manga pages. Low-cost in our context means minimal user intervention, which allows for large-scale application of our method in practice. However, due to the low-cost requirement, it is infeasible to target for the complex animations used by Infinite Comics and DC Squared because of two technical aspects. First, moving comic characters individually requires accurate masks of every character, which is difficult to achieve without manual efforts since manga drawings mainly consist of monochromatic line drawings. Second, to create compelling motions of individual characters, it is essential to understand the semantic relations among the characters, which are hard to obtain automatically. Hence, instead of moving the foreground contents, we focus on creating the illusion of motion by mainly moving a virtual camera while keeping the contents still. This is, in spirit, similar to the Ken Burns effect [15] widely used in documentary filmmaking, where the illusion of motion is created by panning over and zooming into or out of still photographs over time.

Although it is based on simple camera movements, producing an animation that effectively tells a story is challenging. First, to properly show important contents of manga, we need to automatically detect semantically important comic characters from the monochromatic line drawings. Second, camera movements need to be guided by high-level semantics of the contents (e.g., motion states or emotional states of comic characters), which are not available a priori. Third, camera movements must be well planned, such that they are meaningful and consistent throughout the animation, while also properly aligning with the narrative in the manga pages.

To address these challenges, we present a novel approach for automatically generating a storytelling animation from existing manga pages via camera movement mainly comprising of zooming and panning. Our method also simulates a set of special effects (e.g., shaking camera, motion blurring, and moving motion lines) automatically or with little user intervention. Given a sequence of manga pages in raster format, we begin by automatically segmenting panels and balloons, and detecting comic characters based on contour grouping (see Section IV). With the extracted elements, we infer the motion and emotion states of each panel using a context-aware graphical model based on a set of low-level background visual cues that can be easily and reliably detected. Finally, we use the extracted elements and the inferred semantics to determine the type, path and speed of camera movements over time, based on domain knowledge of film production and characteristics of manga (see Section VI).

We have tested our approach on manga series with different story tempos and visual styles. Our results show that our method simplifies the animation creation from still manga contents, and can produce more effective animations for storytelling than existing alternatives. We also demonstrate two applications of our approach to facilitating mobile comic reading and rapid creation of comic trailers.

In summary, we make the first attempt to animate still manga via camera movements for effective storytelling, with the following technical contributions: 1) an automatic method

Ying Cao, Xufang Pang, Antoni Chan, and Rynson Lau are with the Department of Computer Science, City University of Hong Kong, Hong Hong. E-mail: {yingcao4, pangxufang}@gmail.com, {abchan, rynson.lau}@cityu.edu.hk.

for detecting comic character regions on manga line drawings; 2) a context-aware graphical model that can robustly infer motion and emotion states in each panel, from a range of low-level visual patterns; 3) a method for planning camera movements, by combining domain knowledge of film production and characteristics of manga.

II. Background

Animating still imagery. Adding animated motions to a single still image has long been the interest of computer graphics community. Early work by Freeman et al. [11] used a band-pass filter on a single image to produce a compelling illusion of motion. To animate a 2D still picture, researchers have attempted to drive the motion of some target objects using a wide spectrum of sources, including hand-drawn sketches [19], motion parameters estimated from videos [31], as well as 3D motion capture data [12], [14]. Several methods [29], [7] augment scene photos by synthesizing stochastic dynamics of natural phenomena (e.g., rippling water and swinging trees) using physically-based simulations. Xu et al. [34] generated a motion sequence of an animal from a still picture of an animal group, by ordering snapshots of individuals and morphing the ordered snapshots to smooth the animation. Shlizerman et al. [30] cross-faded well-aligned face images to generate a face animation from a face image collection of the same person. To convincingly tell the story, our targeted motion should be well planned based on high-level semantics of image contents. This essentially distinguishes our work from all previous ones, where synthesized motion, either stochastic or designed by users, is not meant to convey a storyline to the audience.

Ken Burns effect. Several commercial tools, such as Microsoft Photo Story [24], support the creation of simple Ken Burns effects from a photo collection, but rely on manual editing on every keyframe. Most closely related to our work is "moves-on-stills" by Jain et al. [14], which makes use of eye movement data recorded from multiple viewers on comic panels to animate still comics with the Ken Burns effect. While sharing a similar goal, our method has two major differences from theirs: First, their method does not consider the semantics of the contents, whereas we take advantage of inferred semantics to determine temporal behaviors (e.g., speed) of the virtual camera, leading to more sophisticated effects that benefit storytelling. Second, their method requires tedious process of collecting eye-tracking data on every comic panel to be animated, while our method is low-cost, only requiring little manual efforts.

Computational Manga. Recent works on computational manga automatically arrange input images or picture subjects and text balloons to synthesize professional-looking manga pages [3], [4]. In this work, instead of assisting in manga creation process, we enhance existing manga pages with animated motion, resulting in an effective way of manga consumption.

III. Overview

As shown in Figure 1, our method consists of three main stages: *element extraction*, *semantic estimation* and *camera movement simulation*. Given a manga page in raster format, the element extractor begins by automatically labeling all the panels, subjects and balloons. Then, taking the labeled manga page as input, the semantic estimator employs a context-aware graphical model to infer the motion and emotion states of each panel. Finally, the camera movement simulator uses the estimated semantics and the labeled elements to compute the type, motion path and speed of camera movements, by combining domain knowledge of film production and manga characteristics.

IV. Element Extraction

Given a sequence of manga pages, our element extractor analyzes each manga page and segments the elements of interest, including panels, balloons and comic characters.

A. Panel and Balloon Extraction

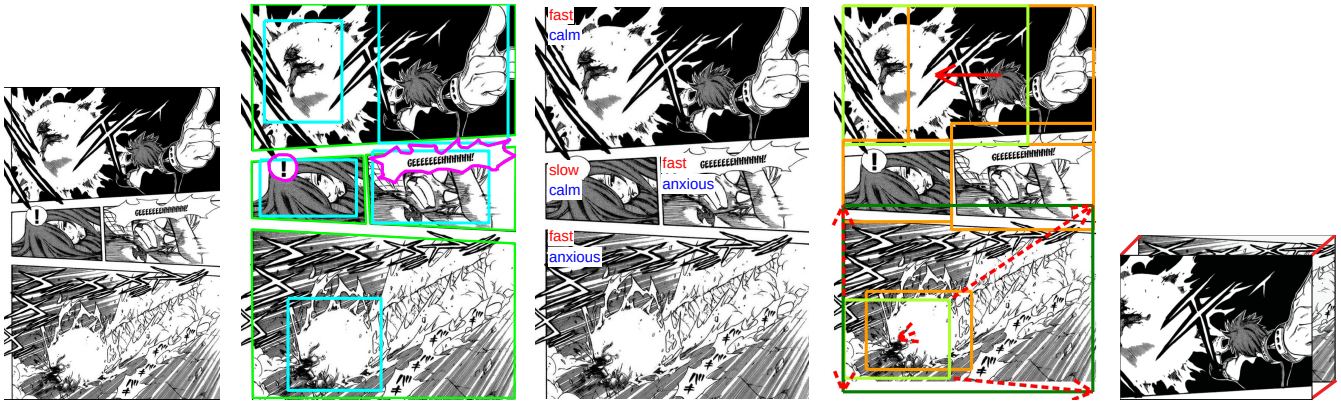
We use the automatic method of [26] to extract panels and then compute reading order of the extracted panels based on their bounding boxes [1]. To extract balloons, we first detect text regions in the panel using the method of [18], which helps localize text lines in comic images. The text regions are then filled with white color, and the balloon regions are identified by running the trapped-ball method [35] using a seed point randomly selected within the text region. For the borderless balloons where the trapped-ball method would output quite irregular and large regions, we simply treat the bounding boxes of the text regions as the detected balloons.

B. Comic Character Detection

Automatically extracting *accurate* boundaries of foreground characters in manga drawings can be challenging since the characters are often drawn with open boundaries and no color information. Thus, given a segmented panel, we aim only to detect bounding boxes of characters, which are sufficient for our purpose of framing them within a rectangular window, i.e., virtual camera (as discussed in Section VI).

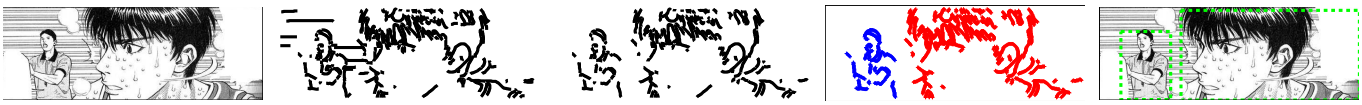
Our observation is that to emphasize the comic characters, manga artists often draw them using percetually salient contours, i.e., *long*, *curved* and *thick* edges (e.g., contours of human body, face, and hair) [25], while the background objects using short curved edges (e.g., tree leaves in natural images or decoration patterns on buildings) or long straight lines (e.g., boundaries of buildings). Therefore, we propose to detect comic character regions by detecting and grouping the long, curved and thick contours. Note that operating on low-level image features would make our detection method quite general and independent of character type (e.g., human, animal or robot), albeit at the cost of its robustness for highly cluttered scenes. Nevertheless, we have found that such cluttered scenes are only occasionally used by artists for some special effects, since frequent appearance of overwhelming visual complexity would impact visual clarity of their artworks.

Given an input panel image $\mathbf{I}$, our character detection algorithm proceeds in three steps: thick contour extraction, contour selection, and contour clustering. To prevent false-positives on balloons and text, we remove balloon regions



(a) Input page (b) Element extraction (c) Semantic estimation (d) Camera simulation (e) Animation

Fig. 1: Overview of our method. Given a manga page ("*Fairy Tail*") (a), our approach automatically extracts panels, comic characters, and balloons (b), and infers per-panel motion state (slow, medium, fast) and emotion state (sad, calm, anxious, excited) (c). It then generates 2D camera movements (e.g., zooming and panning) of varying speeds across the page based on the extracted elements and the estimated semantics (d), resulting in an animation video that can effectively tell the story (e).



(a) Input panel (b) Contour extraction (c) Contour selection (d) Contour clustering (e) Detected regions

Fig. 2: Comic character detection. The contour lines are thickened to improve their visibility in this figure.

from $\mathbf{I}$ by filling them with white (see Figure 2(a)) before comic character detection.

Contour extraction. In this step, we detect all the thick contours in $\mathbf{I}$, which are used as character contour candidates in subsequent steps. We first extract the main structures from $\mathbf{I}$ by removing texture details using [33], and then enhance edges by applying adaptive histogram equalization. Next, we extract a set of thick line segments using an edge-enhanced isotropic nonlinear filter [13]. This set of line segments may have gaps between them due to misdetection or occlusion. Hence, we finally use a contour completion algorithm [27] to connect the fragmented line segments, forming perceptually coherent contours $\mathbf{C} = \{\mathbf{c}_i\}_{i=1}^{N}$ (Figure 2(b)).

Contour selection. The goal of this step is to select a set of contours from $\mathbf{C}$ that belong to the foreground characters. We formulate the selection process as a label assignment problem. Each contour $\mathbf{c}_i$ is associated with a binary variable $y_i \in \{1, 0\}$, which indicates whether the contour is selected ($y_i = 1$) or not ($y_i = 0$). The optimal set of labels $\mathbf{Y} = \{y_i\}_{i=1}^{N}$ are then found by minimizing an energy function, consisting of a *likelihood* term $E_L(\mathbf{Y})$ and a *regularization* term $E_R(\mathbf{Y})$.

$E_L(\mathbf{Y})$ prefers selecting long, curved contours that lie near regions of densely-distributed contours, and is defined as:

$$E_L(\mathbf{Y}) = \sum_{i=1}^{N} \frac{\Phi_i^0}{\Phi_i^0 + \Phi_i^1}(1 - y_i) + \frac{\Phi_i^1}{\Phi_i^0 + \Phi_i^1} y_i, \qquad (1)$$

where $\Phi_i^1 = (1 - l_i)\exp(-\kappa_i)A_i s_i$ and $\Phi_i^0 = l_i \exp(\kappa_i)(1 - A_i)(1 - s_i)$. l_i is the length of $\mathbf{c}_i$, normalized w.r.t all the contours within the panel. κ_i is the mean absolute curvature of $\mathbf{c}_i$, calculated from the the analytical curvatures of a 3rd order polynomial fit to the points of $\mathbf{c}_i$. A_i is the area of the bounding circle of $\mathbf{c}_i$'s 5 nearest neighbors, normalized to the bounding circle of the panel. $s_i \in [0, 1]$ is the average motion-line response of the points on $\mathbf{c}_i$, where the motion-line response is 1 if the point is detected as part of a motion line (see Section V-A) and 0 otherwise. s_i suppresses motion lines while preserving the character contours that look similar to motion lines. For long curved contours that are close to other contours and not on motion lines, Φ_i^1 will have a small value and Φ_i^0 will have a large value, which favors selecting the contour ($y_i = 1$).

$E_R(\mathbf{Y})$ prefers nearby contours having the same labels as:

$$E_R(\mathbf{Y}) = \sum_{i=1}^{N} \exp(-d(\mathbf{c}_i, \mathbf{c}_{n_i}))I[y_i \neq y_{n_i}], \qquad (2)$$

where n_i is the index of the nearest contour to $\mathbf{c}_i$ according to Hausdorff distance $d(\cdot, \cdot)$ between two contours. $I[y_i \neq y_j]$ is an indicator function that is set to 1 if $y_i \neq y_j$, and 0 otherwise.

We use the graph-cut algorithm [2] to minimize the total energy function $E_L(\mathbf{Y}) + E_R(\mathbf{Y})$, thus obtaining a set character contours (Figure 2(c)).

Contour clustering. In this step, we cluster the character contours into groups (Figure 2(d)) using single-linkage agglomerative hierarchical clustering. We set the maximum number of clusters to 3 since 92% of panels in our training dataset contain at most 3 characters. Small groups with less than 4 contours are merged to their closest large group, and groups with only one contour are discarded as outliers. Finally,

we compute the bounding box of each group to represent a character region (Figure 2(e)). Our clustering method may merge multiple spatially adjacent characters together to form a single region. However, it is worth emphazing that, because each region is used to define a framing window during animation generation (Section VI), such grouping behaviour is not problematic in our context since it results in spatially adjacent characters being framed into a single shot.

C. Evaluation

We evaluated the effectiveness of our element detectors on 220 manga pages from different manga series ("*Fairy Tail*", "*Naruto*", "*Slam Dunk*", "*Binetsu Shoujo*" and "*Strobe Edge*"), containing 1041 panels, 1221 characters, and 1124 balloons. For each page, we manually labeled the boundaries of the panels and balloons, and the bounding boxes of the characters as ground truth regions. We measure the detection accuracy using the Jaccard index, $J = \frac{|R \cap G|}{|R \cup G|}$, where R and G are the automatically detected and the ground truth element regions. A higher Jaccard index indicates better detection performance. We define an element (i.e., panel, balloon or character) to be correctly detected if $J \geq 0.8$.

On our test dataset, the panel extraction method successfully detected 90% of all panels, and failed on borderless panels or panels with a large percentage of open boundaries. The balloon extraction method correctly detected 83% of the balloons,but typically failed on borderless balloons or very small balloons with little text. Since clustering nearby characters is valid in our character detection as discussed in contour clustering step, for the ground-truth in character detection, we combined multiple overlapping characters into a single region. Our character detector correctly identified 85% of the character regions, and would malfunction when the scene is extremely cluttered.

V. Semantic Estimation

Using the automatically labeled panels, balloons and comic characters, we next estimate semantic information for each panel. We consider two basic types of semantics, motion and emotion, which are crucial for understanding the tempo and tone of the story. The motion state describes how fast an action is taking place, and has three possible values: *slow*, *medium*, and *fast* (denoted as $\{m_1, m_2, m_3\}$), as shown in Figure 3(left). The emotion state describes how a character is feeling, and has four possible states, *sad*, *calm*, *anxious*, and *excited* (denoted as $\{e_1, e_2, e_3, e_4\}$), which are derived from a 2D arousal-valence space that is commonly used in the field of emotion study [16] (see Figure 3(right)).

One possible approach to estimate the motion and emotion states of each panel is to recognize facial expressions and poses of comic characters. Unfortunately, as the face and body of a comic character may have exaggerated proportions, with inconsistent transformations across drawings (viewpoints), existing computer vision techniques, which assume that the human face and body deform slightly and consistently, cannot be easily amended to address our problem.

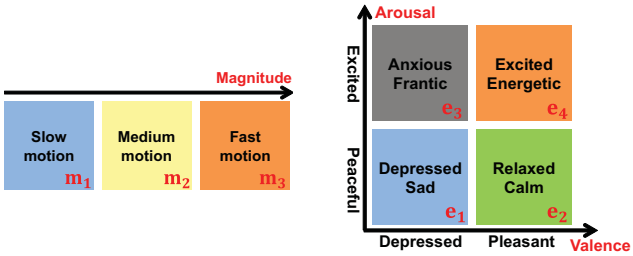


Fig. 3: Motion and emotion space.

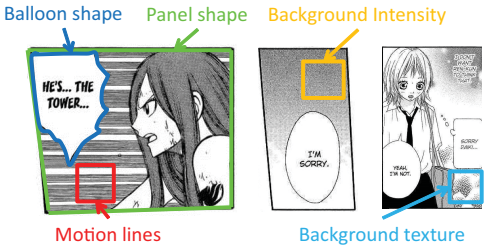


Fig. 4: Features used for semantic estimation. The left panel is from "*Fairy Tail*", and the others are from "*Strobe Edge*".

We note that manga artists often use a number of visual patterns (e.g., motion lines and background texture) to enhance readers' perception of motion and emotion states depicted in the panels [22]. For example, the use of streaked backgrounds in the left most panel of Figure 4 can make readers feel that they are moving together with the characters. In addition, emotionally expressive background textures (e.g., screen tones in the right most panel of Figure 4) can help readers experience the characters' feeling. This motivates our insight that motion and emotion states are correlated with these non-foreground visual patterns, and thus can be estimated reliably from these patterns without using high-level attributes of the characters.

Hence, we propose a context-aware probabilistic model to estimate the motion and emotion in the panel from its low-level visual patterns. The model infers a panel's states by considering all the visual patterns in a panel jointly. This allows the model to better handle conflicting evidence, for example, when the same panel contains both motion lines (suggesting fast motion) and regular panel shape (suggesting medium motion). The local context model, which connects each panel to its adjacent panels, allows a panel's semantics to be inferred from its neighbors when the visual patterns in the panel are insufficient to support direct inference. This is motivated by the fact that a sequence of panels depicts a continuous event, and thus adjacent panels will likely have coherent semantics. For example, a comic character that is crying in one panel is more likely to be peaceful in the adjacent panel than laughing.

A. Features of Visual Patterns

To describe the aforementioned visual patterns, we compute a set of features for each panel (Figure 4). During feature extraction, we first remove all comic characters and balloons detected in Section IV by filling their regions with white.

- **Presence of motion lines** I^m: Motion lines in the background region can indicate that the character inside a panel is moving fast. The motion lines typically appear as a group of parallel lines, or as lines converging towards one point. We use a binary variable I^m to indicate the presence or absence of motion lines. In addition, we also classify motion lines into parallel and converging for use in the camera simulation stage. Please refer to the supplementary material for our motion line detection algorithm.
- **Background texture histogram** $\mathbf{H}^t$: Manga artists emphasize a comic character's feeling by using a variety of background texture patterns (i.e., screentones), from basic dotted tones to fancy effects. Therefore, textures of different styles can also reflect the emotion states of the character. We describe the texture in a background region with a histogram of 64 texton codewords [20], which are based on Gabor filters at 8 orientations and 8 scales.
- **Background intensity** $\mathbf{H}^I$: Manga artists often adjust the intensity level of the background region to highlight the emotion of the scene. For example, a darker background may convey a feeling of depression or suspense. We compute three features: the mean, standard deviation, and entropy of the intensity values of the background pixels.
- **Regularity of balloon shape** R^b: The shape of the balloon can also provide cues about the character's feeling and action. In scenes where comic characters are talking calmly, the balloons are typically ellipses. When comic characters are fighting or excited, the balloons will likely have jagged edges (e.g., see Figure 4). We compute the orientations of the normal vectors sampled uniformly along the balloon contour, and use the entropy of the orientation to describe the balloon shape.
- **Regularity of panel shape** R^p: An irregular panel shape can give a strong sense of dynamics, suggesting the presence of strong motion or emotion. To describe the panel shape, we compute the standard deviation of the relative orientations between any two adjacent boundaries of the panel.

Given a panel, its motion-related and emotion-related feature vectors are defined as $\mathbf{x}^m = (I^m, R^p, R^b)^T$ and $\mathbf{x}^e = (R^p, R^b, \mathbf{H}^I, \mathbf{H}^t)^T$, respectively.

B. Context-aware Graphical Model

To infer the motion or emotion states, we propose a context-aware probabilistic model that optimally combines a variety of visual features as local evidence, while considering contextual information from neighboring panels. We use two separate models, one for motion and another for emotion. Here, we describe the model used to estimate the motion state $y_i \in \{m_1, m_2, m_3\}$ from input features $\mathbf{x}_i^m$. The model for estimating the emotion state $y_i \in \{e_1, e_2, e_3, e_4\}$ can be obtained similarly from the input emotion-related feature vector $\mathbf{x}_i^e$.

To infer the motion states of the panels, for each manga page, we build a temporal chain of the panels, $\mathcal{C} = (\mathcal{V}, \mathcal{E})$, according to the reading order, where nodes $\mathcal{V}$ represent the panels, and edges $\mathcal{E}$ represent connectivity between the consecutive panels. We associate a discrete variable $y_i \in \{m_1, m_2, m_3\}$ with each node, representing its motion state. Let $\mathbf{Y} = \{y_i | \forall i \in \mathcal{V}\}$ be the states of the entire page and $\mathbf{X} = \{\mathbf{x}_i^m | \forall i \in \mathcal{V}\}$ be the observed features of all the panels. This gives rise to a chain-structured probabilistic graphical model (also called a conditional random field, CRF [17]), which captures local evidence $\mathbf{x}_i^m$ of motion state y_i, as well as interactions between neighboring motion states. Figure 5 shows a manga page along with its graphical model.

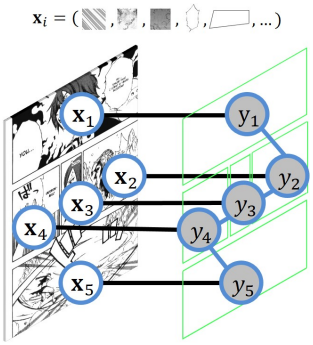


Fig. 5: Context-aware graphical model used to estimate the semantics (i.e., motion and emotion states) of all the panels $\{y_i\}$ from low-level visual patterns $\{\mathbf{x}_i\}$.

Using the graphical model, we define the probability of $\mathbf{Y}$ conditioned on $\mathbf{X}$ as:

$$p(\mathbf{Y}|\mathbf{X}) = \frac{1}{Z(\mathbf{X})} \exp[\sum_i \phi(\mathbf{x}_i^m, y_i) + \sum_{(i,j)\in\mathcal{E}} \psi(y_i, y_j)], \quad (3)$$

where $Z(\mathbf{X})$ is a partition function ensuring that the probabilities sum to one. $\phi(\mathbf{x}_i^m, y_i)$ is the node energy denoting the potential of assigning y_i to node i, having seen evidence $\mathbf{x}_i^m$. $\psi(y_i, y_j)$ is the edge energy measuring the potential of jointly assigning y_i to node i and assigning y_j to node j. Thus, our problem amounts to finding the most likely configuration of $\mathbf{Y}$ given the evidence $\mathbf{X}$, i.e., $\hat{\mathbf{Y}} = \arg\max_{\mathbf{Y}} p(\mathbf{Y}|\mathbf{X})$. The inference can be performed efficiently using the Viterbi algorithm [10].

Node energy. The node energy $\phi(\mathbf{x}_i^m, y_i)$ is the potential of motion state y_i and observed visual features $\mathbf{x}_i^m$ occurring together. We define it as a linear function of $\mathbf{x}_i^m$,

$$\phi(\mathbf{x}_i^m, y_i) = \mathbf{w}_{y_i}^T \mathbf{x}_i^m, \quad (4)$$

where $\mathbf{w}_{y_i}$ is a weight vector for each motion state $y_i \in \{m_1, m_2, m_3\}$. We use a different weight vector for each motion state since different features (or combinations thereof) are useful for predicting a particular motion state.

Edge energy. The edge energy allows the motion states of adjacent panels to influence each other, and is defined as

$$\psi(y_i, y_j) = \beta_{y_i, y_j}, \quad (5)$$

where β_{y_i,y_j} is the edge weight, which measures the potential of two neighboring nodes being assigned to y_i, y_j. Figure 6 visualizes the edge weights estimated from the "*Naruto*" manga series.

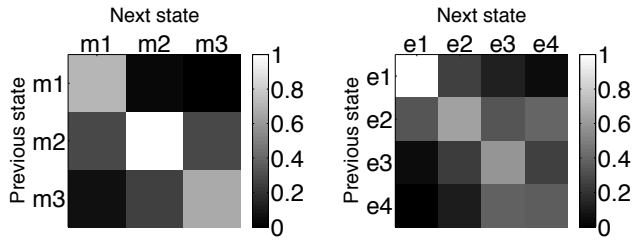


Fig. 6: The edge weight matrix for motion (left) and emotion (right) estimation. The intensity of each entry indicates the probability of observing two specific states at adjacent panels. Note that for both matrices, the entries closer to the diagonal tend to have higher values than those farther away from the diagonal. This indicates that the motion/emotion states change gradually between adjacent panels, and that dramatic changes are less frequent.

C. Weight Learning

The parameters of our graphical model are $\mathbf{\Theta} = \{\mathbf{w}_l, \beta_{l,k} | \forall l, k \in \{m_1, m_2, m_3\}\}$. We learn the parameters $\mathbf{\Theta}$ by maximizing the log-likelihood of a training dataset (details below) using the implementation in [28]. Our training dataset contains 125 manga pages from 3 manga series of different genres, 30 from "*Naruto*" (action), 56 from "*Slam Dunk*" (sports), and 39 from "*Binetsu Shoujo*" (romance). For each page, we first manually annotated all the panels, comic characters and their corresponding balloons. Then, the motion and emotion states for each panel were labeled by five people with significant experience in reading manga. Finally, each panel was assigned the motion/emotion states with the largest consensus among the labelers.

D. Evaluation

We evaluated the effectiveness of our motion/emotion estimators on the training dataset. We used 5-fold cross validation where 80% of the data is used for training and 20% held out for testing. We compared our estimator (using all features and context) against several variants, in which the edge energy is disabled (no context is used) or only one of the features is used as input. The estimators were evaluated using the average accuracy rate on the five validation folds, and the results are shown in Table I. Our estimator outperforms the variant that uses only node energy by a large margin, which confirms the importance of using the context of neighboring panels. It also outperforms the estimators that use only individual features, thus showing the importance of considering multiple features.

Contribution of foreground features. To investigate how well the visual features of foreground characters affect the semantic estimation, we compare our estimator with two alternative estimators, the *fg* estimator and the *fg-bkg* estimator. The alternative estimators are constructed and learned in the same way as our estimator except that they use different sets of visual features. The *fg* model is based on only foreground features, while the *fg-bkg* model combines foreground features and all the background features in Section V-A. We compute the following features on the foreground region of each panel: 1) foreground intensity computed in the same way as background intensity in Section V-A; 2) foreground texture computed in the same way as background texture histogram in Section V-A; 3) foreground shape described by the HOG (Histogram of Oriented Gradient) feature [8].

	Motion	Emotion
Our estimator (all features, w/ context)	**86.3%**	**85.9%**
Our estimator (w/o context)	70.3%	72.0%
Our estimator (w/ motion lines)	58.9%	-
Our estimator (w/ panel shape)	57.8%	60.3%
Our estimator (w/ balloon shape)	52.7%	65.9%
Our estimator (w/ bkg intensity & texture)	-	53.0%
The *fg* estimator	51.7 %	62.7 %
The *fg-bkg* estimator	65.6 %	68.0 %

TABLE I: Average accuracy of our motion/emotion estimators and its variants. For both motion and emotion, our context-aware estimator using all features outperforms the variants.

The results are shown in Table I. The two estimators with foreground features have poorer performance than our background-based estimator. This implies that the foreground features are not indicative of the motion and emotion states of the panel, and would confuse our estimator (poor performance of the *fg-bkg* estimator). Hence, we do not incorporate the foreground features into our semantic estimator.

VI. Camera Movement Simulation

Given the extracted elements and inferred per-panel sematnics, the goal of camera movement simulation is to move a *window* (viewport) of fixed aspect ratio over the manga pages to generate an animation that effectively tells the story. Working in the 2D image domain, we primarily focus on 2D camera movements, panning and zooming. In our context, panning and zooming are implemented by translating and scaling the window, respectively. We first determine, for each panel, the type and path of camera movement according to the panel's contents and semantics. Then, the speed of the camera movement and shot duration are computed based on the inferred motion and emotion states.

To begin with, we group all the comic character and balloon elements (represented by a set of discrete points along their contours) using the mean-shift algorithm with the Hausdorff distance, so that adjacent subjects and their balloons are grouped into the same cluster. We refer to the bounding box of each cluster as a *region of interest* (ROI). Then, for each ROI and panel, we determine a window to properly compose its contents. These windows will serve as the key windows for computing camera movement path as described later. For a panel, the window is defined as the smallest window enclosing the panel. For a ROI, the window is selected so that it should be tight around the ROI, contain as many ROI pixels as possible and lie inside the current panel. Please refer to the supplemental for more details.

A. Computing Type and Path of Camera Movements

In film production, basic panning and zooming are typically motivated as follows [32]:

- **Panning** is usually used to shift the audience's focus from one subject or location to another.
- **Zooming in** can make the audience focus on a comic character, showing the character's reaction or emotions.
- **Zooming out** can pull the audience away from the character, putting the character in context.

We propose a method to determine the type and path of camera movement, by combining the filming techniques above with manga characteristics. Our method is guided by the following principles (illustrated in a binary decision tree in the supplemental):

- **No ROIs.** If a panel has no ROIs, then the panel is likely an establishing shot with only a background scene. We translate the window from right to left (if the panel is a horizontal rectangle) or from top to bottom (if the panel is a vertical rectangle), to establish the whole scene [9].
- **Strong motion with motion lines.** If a panel has a single ROI with motion lines present fast motion detected, this indicates that the character is moving fast. To create a strong illusion of the character motion, we start with the ROI and move the window along the motion lines, by either panning across parallel motion line (in the direction that is consistent with the reading order), or zooming into the convergence point of converging motion lines.
- **Weak or strong emotion.** If there is a single ROI with weak (i.e., sad) or strong emotion (i.e., anxious and excited), we zoom to highlight the emotions of the character [32]. If the background region is large, then we start with the ROI and zoom out to the panel window, putting the character into its context (i.e., background). Otherwise, we start with the panel window and zoom into the character (ROI window) to reveal its inner thoughts and emotions.
- **Multiple ROIs.** When there are more than one ROI inside a panel, we use a series of pans to visit each of the ROIs, in order to transfer focus from one ROI to another [32]. To determine the visiting order, we use the manga reading convention, which is to read contents within a panel from right to left and then top to bottom. The windows of the ordered ROIs are then used as the set of key windows.
- **Other cases.** If there is a single ROI with calm emotion or weak motion (slow/medium), we pan the window over the panel to create subtle motion based on its shot type. The distance of the panning is proportional to the panel's shot duration as computed in Section VI-B. In particular, we detect a close-up shot if the ROI occupies nearly the entire panel (i.e., ROI to panel area ratio $\geq 70\%$). For a close-up shot, we start by showing all the contents, and pan down (if panel is a vertical rectangle) or left (if panel is a horizontal rectangle). Otherwise, the panel is a medium or long shot, and we start with the ROI and pan the window in the direction that captures the most panel contents, to put the character in the context.

Note that, in all the cases, if the start and end key windows are similar in position and size, we just show the start key window without any movement to avoid any visually abrupt short pan or zoom.

Adding establishing camera movement. When panning across the panel of a long shot, it is essential to show the entire scene at the end, so that the audience can see the relationships among the characters that have been presented. Therefore, at the end of panning over a long shot, we append a zoom out, which is from the end window of the panning to the window of the panel. A panel is regarded as a long shot, when all the comic characters take up less than 20% of the panel area.

Transition between panels. After computing the window movements for each individual panel, we link these movements together to form a coherent motion. Given two panels that are adjacent in reading order, we smoothly transition from the last key window of the first panel to the first key window of the second panel. We support three transition modes: panning, cross-blurring, cross-fading. Panning is used when the two panels are well-aligned horizontally in the same row on the page and the ROIs in the two key windows have similar aspect ratios. This avoids visual discontinuities induced by diagonal camera movements and abrupt changes in ROI shape. Cross-blurring (i.e., blurring the first panel, transitioning to a blurred second panel, and then deblurring) is used for moving from one page to another, to imply scene or event transition between the pages. Otherwise, cross-fading is used.

B. Computing Speed and Shot Duration of Camera

To fully express the tempo and tension of a story, for each panel, we define the speed s of its camera movements as a function of both its motion state m and emotion state e,

$$s = \rho[\lambda \tfrac{m}{3} + (1-\lambda)\tfrac{e}{4}], \tag{6}$$

where $m \in \{m_1 = 1, m_2 = 2, m_3 = 3\}$ and $e \in \{e_1 = 1, e_2 = 2, e_3 = 3, e_4 = 4\}$. The parameter λ balances the contribution of the motion and emotion states. In our implementation, we set $\lambda = 0.6$ to give the motion state a higher weight in affecting the speed. ρ is a scalar value that varies for different types of camera movements: w/30 pixels per second for panning (w is the width of a manga page), and 1.4 scales per second for zooming.

To offer the audience with enough time to understand the contents in the window, we hold camera over a key window for a certain amount of time, which we denote as the *shot duration*. It is defined as a content-dependent function,

$$t = \tau + t_c \mathcal{A}_c + t_b \mathcal{A}_b, \tag{7}$$

where $\tau = 2$ is a bias factor. $\mathcal{A}_c$ and $\mathcal{A}_b$ are the areas of comic characters and balloons, normalized to the area of key window. t_c and t_d are weighting coefficients, balancing contributions of comic characters and balloons. Since more time is usually required for readers to read text in balloons than to view comic characters. Our implementation uses $t_c = 1$ and $t_b = 3$. These parameters can be adjusted according to the user's preference.

C. Adding Special Effects

To produce more dynamic animations, we add three special effects to the animations, shaky effect, motion blur and moving motion lines. Shaky effect results from simulating movements of a handheld camera, to increase the tension in the animation [32]. To do this, we track the camera offsets in video footage with a shaking camera, and apply the offsets to the window when panning or zooming in/out within the panel. This effect is applied when the *anxious* emotion state is detected. Motion blur is achieved by applying a zoom radial blur filter on a foreground character, in cases of zooming in and out when *fast* motion is detected. Moving motion lines are simulated using user-provided foreground and background masks to enhance the perception of the comic character's motion, when parallel motion lines are detected. Implementation details of the effects are included in the supplemental.

VII. Results and Evaluation

A. Comparison to Prior Works

We compared our approach against two previous works that use the Ken Burns effect on still images: Microsoft Photo Story [24] (MPS) and Moves-on-Stills [14] (MoS). MPS generates an animation, like ours, by randomly panning and zooming over photos. MoS animates comics using eye movement recordings from multiple users.

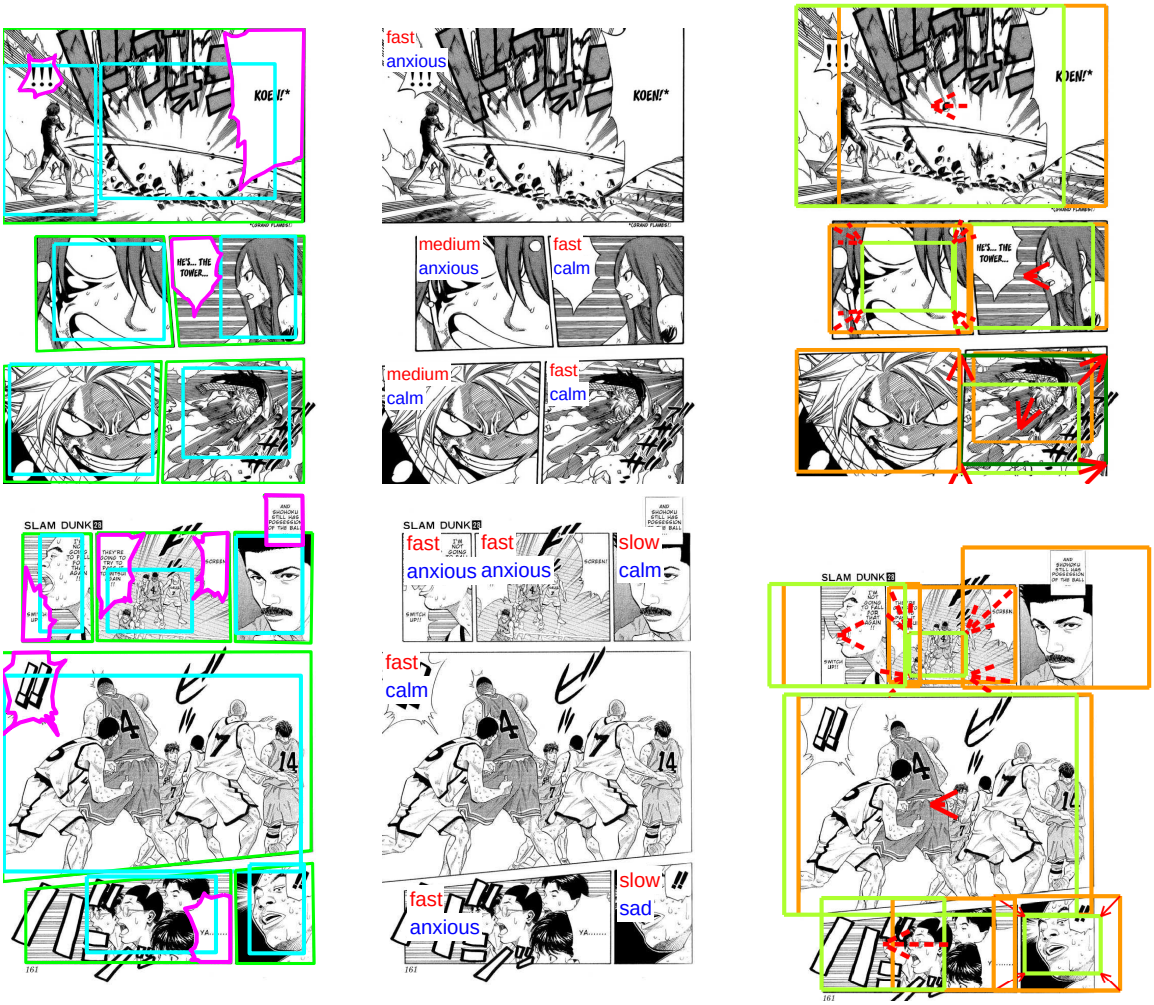
We tested the three approaches on three manga series of different genres, 18 pages from "*Fairy Tail*" (action), 16 pages from "*Slam Dunk*" (sports), and 21 pages from "*Strobe Edge*" (romance). For each manga series, we generate one animation using each of the three methods. Note that these pages are not part of the training set used in Section V-C. We have chosen these manga series because they are representative and stylistically distinctive: "*Fairy Tail*" comprises many visual effects (e.g., motion lines and sound words); "*Slam Dunk*" contains detailed foreground characters and cluttered background; "*Strobe Edges*" is characterized by abstract characters and elaborate background textures.

To create animations using MPS, we automatically segmented the panels on all pages into individual photos, imported them into MPS, and created a video story by following the instructions of its wizard. For one manga series, this process only took about 10 seconds, with the motion and duration of the camera set automatically. To create animations with MoS, we recorded eye gaze data from 5 experienced manga readers viewing the testing pages, using a Tobii infrared eye tracker with the same experiment setup as in [14]. Acquiring eye tracking data from 5 viewers on one manga series ($\sim$20 pages) took about 1 hour, including device setup and recording time. For comparison, our approach took an average of 3.5 minutes (including 2.5 minutes for user assistance in correcting erroneously detected elements before the semantic estimation) to process one manga series ($\sim$20 pages). The precision rates of our panel, balloon and character detectors on the testing dataset are 87%, 82%, and 83%, respectively. While the accuracies of our element extraction are not considerably high, user intervention is quite minimal, especially compared to the efforts required to manually segment all the elements, which typically takes 25 minutes for about 20 pages. In addition, errors in the element extraction step are fixed by users before the semantic estimation, and therefore would not affect the quality of final animation.

For a fair comparison in the user study, some special effects, including motion blur and moving motion lines, are not used in our approach as other methods do not naturally support these effects. Moreover, all the methods use the same panel display order (i.e., reading order of the panels as computed in Section IV-A). For transitions between panels, only cross-fading is used. Figure 7 shows some results from element extraction, semantic estimation and camera movement simulation. Note that all our results shown in this paper and in the accompanying videos are generated using the semantics that are automatically inferred by our semantic estimator.

User Study. We conducted a perceptual study to evaluate the effectiveness of our method versus the other two methods (MPS and MoS). Our study includes 20 participants, consisting of 10 experienced manga readers and 10 with limited experience in reading manga. Each participant compared the animation methods on three manga series of different genres. In each comparison, the participants were presented the three animations (ours, MPS, MoS) one-by-one with randomized presentation order, and asked to rank their preferences on the animations and rate various aspects of the animations, including storyline and pacing. To avoid participant frustration, we limited the length of each animation to be within 3 minutes by using only 5 contiguous pages of each manga series. See the supplemental materials for some example animations used in the study. After the study, we obtained 60 comparisons in total (3 manga series by 20 participants). At the end of viewing sessions, the participants were also asked to rate how well they appreciated such manga animations compared to traditional manga pages, on a scale of 1 to 5 with 1 being the least and 5 being the most. We end up with an average rating of 4.3, implying that the participants are in favor of the dynamic form of manga.

Results. As shown in Figure 8, the participants are, in general, very enthusiastic about the animations generated by our approach, and rank them as the most effective in storytelling 88% of the time (53 out of 60), with only 7 cases where the MoS animations are rated higher. Also, as reflected in the ratings, the main comment about MoS is that it cannot express the pace of the story, mainly due to the lack of semantic understanding of the contents being animated. The 7 cases where MoS was ranked higher than our approach were all on the animations of "Strobe Edge", which is a romantic manga series. The slow pace of this romantic manga series leads to animations comprising slow camera movements with constant speed. In this case, the advantage of our approach in varying the camera speed and movements according to the semantics is not obvious, and thus the participants are likely to randomly select between the two methods. MPS is always ranked third in our study. The random nature of MPS breaks the structure of the manga contents, which are designed by artists to develop the storylines. Thus, MPS fails to generate meaningful animations from manga. In contrast to MPS and MoS, our approach faithfully reproduces the



Element Extraction Semantic Estimation Camera Movements

Fig. 7: Results of element extraction, semantic estimation and camera movement simulation on two manga pages (one page per row). For each page, extracted panels, comic characters and balloons are outlined in green, cyan and magenta, respectively. Estimated motion/emotion states are in the upper-left corner of each panel. For camera movement simulation, each rectangle represents a key window, with the color indicating its temporal order (orange = 1st, green-yellow = 2nd, green = 3nd). Arrows indicate the motion trajectories of the window, with a larger size indicating a higher speed. Dashed lines suggest the presence of shaky effect. Note that to reduce clutter, cross-panel movements of the framing window are not visualized in these examples. The manga pages are from "*Fairy Tail*" (top) and "*Slam Dunk*" (bottom).

story pace by using the inferred motion and emotion states, while generating meaningful camera movements consistent with film production. When rating the animations of three different manga series, the two groups of readers (expert and inexperienced) have the same ranking. On "*Strobe Edge*" (romance), the rankings of the two groups only differed in one comparison. This shows that our animations are appreciated by both experienced and inexperienced manga readers.

B. Effect of Semantic Estimation and Camera Movement

We evaluate the contribution of our semantic estimation to the final animations, the effect of its errors on the perceived quality of the animations, and the effectiveness of our camera movement simulator. For these purposes, we compare our method (method A) to three variants of our method. Method B: we use ground-truth semantic labels rather than the estimated ones. Method C: we disable the semantic estimation, and generate animation using elements detected by our element extractor and a naive camera movement technique with constant speed. For the naive camera movements, we simply pan the framing window across ROIs in their reading order. Method D: we replace our camera movement simulation (used in method A) with the naive camera movement strategy.

We used the same procedure and set of manga series as in Section VII-A to perform a user study with 20 participants. Figure 9 shows the results from the user study. There is no significant difference between the ratings of our method and the variant with ground-truth semantic labels (A vs. B, Friedman test, $p > 0.05$). This implies that the errors in the semantic estimation step would not significantly affect the quality of final animations. On the testing dataset, our motion and emotion estimators achieve the accuracy rates of 87% and 88%, respectively. In most failure cases, the motion estimator misclassified "slow" m_3 as "medium" m_2, while the emotion

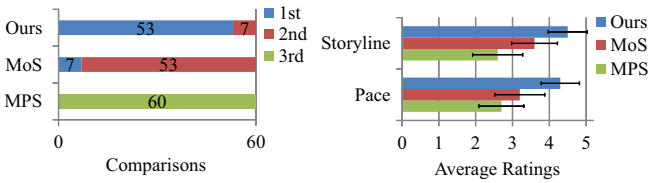


Fig. 8: Results from the user study. Left: preference ranking of our approach versus MPS and MoS. Ours are consistently ranked higher than others (Friedman test, p $<$ 0.05). Right: participant ratings for the effectiveness in revealing storyline and story pace. The ratings of the three methods in storyline and pace have statistically significant differences (one-way repeated measures ANOVA, $p < 0.05$). Our method is rated higher in all aspects than the other two methods with statistical significance (paired t-test, $p < 0.05$).

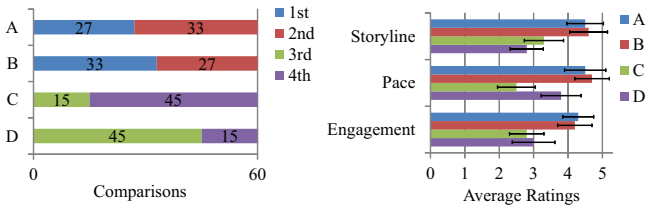


Fig. 9: Results from the user study. Left: preference ranking of our method (method A) versus other competitors, i.e., our method with ground-truth semantic labels (method B), our method without semantic estimation (method C), and our method with naive camera movements (method D). Right: participant ratings for the effectiveness in presenting storyline and story pace, and engaging viewers. For all the aspects, the ratings for the different methods have statistically significant differences (one-way repeated measures ANOVA, $p < 0.05$). The ratings of our method are similar to those of method B (paired t-test, $p > 0.15$), which are higher than method C and D (paired t-test, $p < 0.05$)

estimator misclassified "excited" e_4 as "anxious" e_3, largely because the two motion or emotion states have similar visual features. These errors do not lead to obvious artifacts in the final animations, perhaps because people tend to notice high contrast between two obviously different states (e.g., "fast" and "slow"), and are less sensitive to differences between two similar states (e.g., "medium" and "slow"). From the participants ratings, we can see that our estimated semantics are important for expressing the pace of the story (A vs. C), while our well-designed camera movements are essential for revealing the storyline (A vs. D). Moreover, well-designed camera movements are important for engaging viewers, since they change with contents and semantics, resulting in more versatile perceived motion, as compared with the pan-based naive camera movements.

C. Applications

Automatically converting manga contents into an animation will benefit many approaches to manga consumption. To show this potential, we demonstrate two applications of our method.

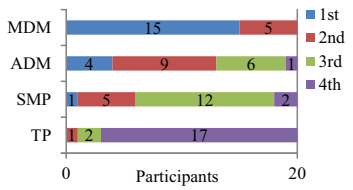


Fig. 10: Preference ranking of 4 mediums for mobile manga reading. Manually-advanced dynamic manga (MDM) is consistently rated higher than automatically-advanced dynamic manga (ADM), sequential manga panels (SMP) and traditional pages (TP) (Friedman test, p $<$ 0.05).

First, our animation can be used to create a dynamic comic reading experience on mobile devices. Second, our method can be used to create comic trailers with little post-process authoring.

1) Mobile Comic Reading: To read comic pages that do not fit within the small screen of smartphones or tablets, readers are forced to frequently pan and zoom on the screen. Our animation displays a local portion of the entire artwork at one time, and contains a sequence of automatically performed panning and zooming operations on comic pages. Hence, our animation can be naturally exploited to facilitate comic reading on mobile devices. Instead of directly playing an animation, users manually advance the animation so that they can control their own reading speed. The accompanying video shows a demo of this manually-advanced dynamic manga on a smartphone.

To evaluate whether our manually-advanced dynamic manga (MDM) improves the mobile reading experience, we conducted a user study, where subjects were asked to compare MDM to automatically-advanced dynamic manga (ADM, i.e., animation generated by our method), sequential manga panels (SMP, i.e., a sequence of static panels that are displayed one-by-one in their reading order, which is usually adopted to facilitate comic reading on mobile devices [5], [1]), and traditional pages (TP, i.e., manual pan and zoom). The study included 20 participants, all of whom have prior experience in reading manga or other types of comics. Each participant viewed 6 manga pages using the 4 comic mediums (MDM, ADM, SMP, and TP) on a smartphone with a 5.7" screen of resolution 1920×1080, and then answered three comprehension-based questions about the manga contents (e.g., what is the relationship between comic characters A and B: friend or enemy?).

For a fair comparison, the mediums were presented in a randomized manner. The participants were allowed to adjust the playing speed for MDM and ADM to reflect their own reading speed. At the end of each viewing session, we asked the participant to rank the four mediums in terms of their preference for manga viewing on mobile devices, and leave open comments regarding their ranking.

Figure 10 shows the preference ranking by the participants. Overall, MDM is rated higher than the other options. By analyzing the open comments, we found that most participants liked the animated motion in MDM and ADM, thinking that the motion gave them an interesting and engaging reading

experience and minimized the interaction overhead of reading traditional pages on a smartphone of limited display size.Most participants preferred MDM over ADM because it allowed them to interactively control the reading pace, which was commented as being very helpful for addressing individual reading speeds. The detailed user comments are included in the supplemental material.

2) Comic Trailer: A manga trailer is a short movie of selected manga contents, which is designed especially for advertising a manga series [23]. Given an animation generated by our method, a manga trailer can be easily produced using video editing software for post-processing, e.g., varying the playback speed and incorporating a soundtrack. In the supplemental materials, we show a comic trailer produced in this way. Our method reduces the cost of manga trailer production, making it practical for manga retailers to create *customized* trailers based on individual user preferences.

D. Application to Western Comics

The increased popularity of manga has motivated comic artists across the world to adopt the visual language of manga in their own artworks, blurring the boundary between manga and other comics. Hence, our approach is directly applicable to animating other types of comics that exhibit similar characteristics to manga (e.g., using motion lines to depict fast motion and using oddly shaped balloons and panels to stress the degree of motion and emotion). We show an animation generated from western comics in the supplemental.

VIII. Conclusion

We have presented an approach to convert manga pages to animation via camera movement. This is made possible by automatic extraction of basic comic elements, the proposed graphical model that reliably estimates per-panel motion and emotion states from low-level visual features, and camera movement simulator based on filming techniques. The result is a plausible and consistent set of camera movements, which yields an animation that effectively tells the story. Our experiments on manga series of different genres show that, in comparison to previous methods, our approach can quickly generate more effective storytelling animations. Futhermore, we have also demonstrated the usefulness of our animations in improving mobile comic reading and enabling quick production of comic trailers.

Limitation and future work. Our approach has several limitations, and thus can be improved in a number of ways. First, in this work, we do not animate the characters themselves. This can be done by segmenting a comic character into different parts that can be properly deformed, which we would like to explore in the future. Second, our camera movement simulation currently relies on some heuristic rules. One interesting direction is to understand how animators set up camera movements when interpreting storyboards into motion pictures, and use a data-driven strategy to learn how various visual and semantic factors are correlated with camera settings during animation production. Such knowledge can be exploited to enable simulations of more nuanced camera movements in our animations. Third, deploying our system as a web service would allow any users to convert their favorite comics into animations online. Because creating animations with our system requires users to semi-automatically label foreground objects, our online system can serve as platform for collecting a database of foreground objects labeled by different users. Analyzing such a large-scale database can shed light on what features characterize foreground objects in line drawings, and thus enables design of better algorithms for foreground segmentation in line drawings.

Acknowledgment

We would like to thank Takeo Igarashi for his useful comments and suggestions on the application of our method to comic trailer generation. This work was partially supported by two GRF grants from the Research Grants Council of Hong Kong (CityU 123212 and CityU 115112).

References

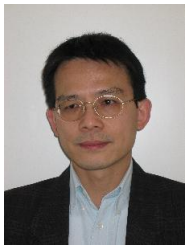
[1] K. Arai and T. Herman. Method for automatic e-comic scene frame extraction for reading comic on mobile devices. In *Proc. ITNG*, 2010.
[2] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE TPAMI*, 23(11):1222–1239, 2001.
[3] Y. Cao, A. Chan, and R. Lau. Automatic stylistic manga layout. In *Proc. ACM SIGGRAPH Asia*, 2012.
[4] Y. Cao, R. Lau, and A. Chan. Look over here: Attention-directing composition of manga elements. In *Proc. ACM SIGGRAPH*, 2014.
[5] C. Chan, H. Leung, and T. Komura. Automatic panel extraction of color comic images. *LNCS, Advances in Multimedia Information Processing*, 4810:775?784, 2007.
[6] T. Cheredar. http://venturebeat.com/2013/06/06/how-dc-plans-to-evolve-digital-comic, 2013.
[7] Y. Chuang, D. Goldman, K. Zheng, B. Curless, D. Salesin, and R. Szeliski. Animating pictures with stochastic motion textures. In *Proc. ACM SIGGRAPH*, 2005.
[8] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. IEEE CVPR*, 2005.
[9] S. Digicams. http://steves-digicams.com/knowledge-center/how-tos/photography-techniques/camera-movement-understanding-the-pan.html, 2014.
[10] G. D. Forney. The viterbi algorithm. *Proc. IEEE*, 61(3):268–278, 1973.
[11] W. Freeman, E. Adelson, and D. Heeger. Motion without movement. In *Proc. ACM SIGGRAPH*, 1991.
[12] A. Hornung, E. Dekkers, and L. Kobbelt. Character animation from 2D pictures and 3D motion data. *ACM TOG*, 26(1), 2007.
[13] M. Huang, M. Yang, F. Liu, and E. Wu. Stroke extraction in cartoon images using edge-enhanced isotropic nonlinear filter. In *Proc. ACM VRCAI*, 2010.
[14] E. Jain. *Attention-guided Algorithms to Retarget and Augment Animations, Stills and Videos*. PhD thesis, CMU, 2012.
[15] KenBurnsEffect. https://en.wikipedia.org/wiki/Ken_Burns_effect, 2013.
[16] E. Kensinger. Remembering emotional experiences: the contribution of valence and arousal. *Reviews in the Neurosciences*, 15(4):241–251, 2004.
[17] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. ICML*, 2001.
[18] L. Li, Y. Wang, Z. Tang, X. Lu, and L. Gao. Unsupervised speech text localization in comic images. In *Proc. ICDAR'13*, 2013.
[19] P. Litwinowicz and L. Williams. Animating images with drawings. In *Proc. ACM SIGGRAPH*, 1994.
[20] J. Malik, S. Belongie, J. Shi, and T. Leung. Textons, contours and regions: Cue integration in image segmentation. In *Proc. IEEE ICCV*, 1999.
[21] Manga25. http://manga25.com/en/, 2015.
[22] S. McCloud. *Making Comics: Storytelling Secrets of Comics, Manga and Graphic Novels*. William Morrow, 2006.
[23] MetaKiller. https://www.youtube.com/watch?v=OURyGa2g0o0, 2009.
[24] MicrosoftPhotoStory. http://en.wikipedia.org/wiki/Photo_Story.

[25] G. Office. *More How To Draw Manga Volume 1: The Basics Of Character Drawing*. Graphic-Sha, 2004.
[26] X. Pang, Y. Cao, R. Lau, and A. B. Chan. A robust panel extraction method for manga. In *Proc. ACM Multimedia*, 2014.
[27] X. Ren, C. Fowlkes, and J. Malik. Scale-invariant contour completion using conditional random fields. In *Proc. IEEE ICCV*, 2005.
[28] M. Schmidt. http://www.di.ens.fr/~mschmidt/Software/UGM.html, 2007.
[29] M. Shinya, M. Aoki, K. Tsuisuguchi, and N. Kotani. Dynamic texture: physically based 2D animation. In *Proc. ACM SIGGRAPH*, 1999.
[30] I. Shlizerman, E. Shechtman, R. Garg, and S. Seitz. Exploring photobios. In *Proc. ACM SIGGRAPH*, 2011.
[31] M. Sun, A. Jepson, and E. Fiume. Video input driven animation. In *Proc. IEEE ICCV*, 2003.
[32] N. Tibbetts. http://blog.teamthinklabs.com/index.php/2012/04/25/the-camera-and-your-emotions/, 2012.
[33] L. Xu, Q. Yan, Y. Xia, and J. Jia. Structure extraction from texture via relative total variation. In *Proc. ACM SIGGRAPH Asia*, 2012.
[34] X. Xu, L. Wan, X. Liu, T. Wong, L. Wang, and C. Leung. Animating animal motion from still. In *Proc. ACM SIGGRAPH Asia*, 2008.
[35] S. Zhang, T. Chen, Y. Zhang, S. Hu, and R. Martin. Vectorizing cartoon animations. *IEEE TVCG*, 15(4):618–629, 2009.

Ying Cao received the Ph.D. degree in computer science from the City University of Hong Kong in 2014, and the M.Sc. and B.Eng. degrees in software engineering from Northeastern University, China, in 2010 and 2008, respectively. He is now a Postdoctoral Fellow in the Department of Computer Science at the City Univerity of Hong Kong. His primary research interest lies in data-driven computational manga and graphic design.

Xufang Pang received her B.Sc. degree from Jiamusi University and her M.Sc. degree from Nanjing Normal University, in 2007 and 2010, respectively. She is now a Ph.D. student in the Department of Computer Science of City University of Hong Kong. Her research interests include computer graphics, machine learning and digital geometry processing.

Antoni B. Chan received the B.S. and M.Eng. degrees in electrical engineering from Cornell University, Ithaca, NY, in 2000 and 2001, respectively, and the Ph.D. degree in electrical and computer engineering from the University of California, San Diego (UCSD), San Diego, in 2008. From 2001 to 2003, he was a Visiting Scientist with the Vision and Image Analysis Laboratory, Cornell University, Ithaca, NY, and in 2009, he was a Postdoctoral Researcher with the Statistical Visual Computing Laboratory, UCSD. In 2009, he joined the Department of Computer Science, City University of Hong Kong, and is now serving as an Associate Professor. His research interests include computer vision, machine learning, pattern recognition, and music analysis. Dr. Chan was the recipient of an NSF IGERT Fellowship from 2006 to 2008, and an Early Career Award in 2012 from the Research Grants Council of the Hong Kong SAR, China.



Rynson W.H. Lau received his Ph.D. degree from University of Cambridge. He was on the faculty of Durham University and Hong Kong Polytechnic University. He is now with City University of Hong Kong. Rynson serves on the Editorial Board of Computer Animation and Virtual Worlds, and IEEE Trans. on Learning Technologies. He has served as the Guest Editor of a number of journal special issues, including ACM Trans. on Internet Technology, IEEE Multimedia, IEEE Trans. on Multimedia, IEEE Trans. on Visualization and Computer Graphics, and IEEE Computer Graphics & Applications. In addition, he has also served in the committee of a number of conferences, including Program Co-chair of ACM VRST 2004, ACM MTDL 2009, IEEE U-Media 2010, and Conference Co-chair of CASA 2005, ACM VRST 2005, ACM MDI 2009, ACM VRST 2014. His research interests include computer graphics and image processing.