Creating Segments and Effects on Comics by Clustering Gaze Data

ISHWARYA THIRUNARAYANAN, KHIMYA KHETARPAL, and SANJEEV KOPPAL, University of Florida
OLIVIER LE MEUR, IRISA University of Rennes 1
JOHN SHEA and EAKTA JAIN, University of Florida

Traditional comics are increasingly being augmented with digital effects, such as recoloring, stereoscopy, and animation. An open question in this endeavor is identifying where in a comic panel the effects should be placed. We propose a fast, semi-automatic technique to identify effects-worthy segments in a comic panel by utilizing gaze locations as a proxy for the importance of a region. We take advantage of the fact that comic artists influence viewer gaze towards narrative important regions. By capturing gaze locations from multiple viewers, we can identify important regions and direct a computer vision segmentation algorithm to extract these segments. The challenge is that these gaze data are noisy and difficult to process. Our key contribution is to leverage a theoretical breakthrough in the computer networks community towards robust and meaningful clustering of gaze locations into semantic regions, without needing the user to specify the number of clusters. We present a method based on the concept of relative eigen quality that takes a scanned comic image and a set of gaze points and produces an image segmentation. We demonstrate a variety of effects such as defocus, recoloring, stereoscopy, and animations. We also investigate the use of artificially generated gaze locations from saliency models in place of actual gaze locations.

CCS Concepts:• **Information systems** → **Clustering**; • **Computing methodologies** → **Interest point and salient region detections**; **Image segmentation**; **Perception**; *Cluster analysis*; *Image processing;*

Additional Key Words and Phrases: Comics, effects

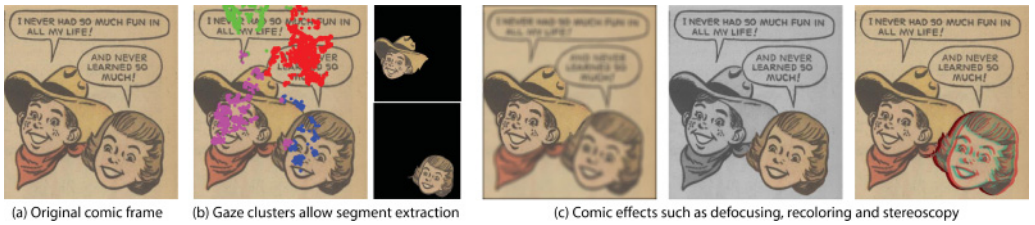
ACM Reference Format:
Ishwarya Thirunarayanan, Khimya Khetarpal, Sanjeev Koppal, Olivier Le Meur, John Shea, and Eakta Jain. 2017. Creating segments and effects on comics by clustering gaze data. ACM Trans. Multimedia Comput. Commun. Appl. 13, 3, Article 24 (May 2017), 23 pages.
DOI: http://dx.doi.org/10.1145/3078836

1. INTRODUCTION

Comics are a captivating medium of storytelling [Comichron 2015; ComicBookPlus 2015]. Increasingly, traditional comics are being augmented with digital effects (animations, stereoscopy, moves-on-stills, etc.) to appeal to a new generation of viewers who primarily use handheld devices. When creating such effects on comic panels, an open question is where to apply the effects? Applying effects to every object is unnecessary and could even be distracting to the viewer. In a fully manual comic digitization workflow, the digital artist guesses at what objects in the comic panel would

Authors' addresses: I. Thirunarayanan and K. Khetarpal, Department of Electrical and Computer Engineering, University of Florida, 216 Larsen Hall, Gainesville, FL 32611; emails: {iiyengarthir, kkhetarpal}@ufl.edu; S. Koppal, 437 New Engineering Building, Gainesville, FL 32611; email: sjkoppal@ece.ufl.edu; O. Le Meur, IRISA, Campus Universitaire de Beaulieu, 35042 Rennes Cedex, France; email: olivier.le_meur@irisa.fr; J. Shea, 439 New Engineering Building, P.O. Box 116130, Gainesville, FL 32611; email: jshea@ece.ufl.edu; E. Jain, E540 CSE Building, PO Box 116120, Gainesville, FL 32611; email: ejain@cise.ufl.edu.

DOI: http://dx.doi.org/10.1145/3078836



(a) Original comic frame (b) Gaze clusters allow segment extraction (c) Comic effects such as defocusing, recoloring and stereoscopy

Fig. 1. Eyetracking comics viewers allows us to extract the regions that are important to the story, thus enabling us to apply effects to the objects of interest.

be best highlighted via added effects. We address this challenge by leveraging eyetracking data collected from comic book readers. These data naturally highlight those regions in the comic panel that are the most important components for the readers (Figures 1(a) and (b)).

Even if we are able to identify regions that are important to the comic book narrative, the process of segmentation and applying effects can still be cumbersome. For example, most commercially available photo editing software provides highly controllable, but very labor-intensive, manual segmentation methods based on techniques such as intelligent scissors [Mortensen and Barrett 1995] or user-assisted matting [Levin et al. 2008]. We hypothesize that clusters extracted from eyetracking data could help an established off-the-shelf computer vision algorithm, such as color-based superpixels, to piece together plausible segments for effects generation (Figures 1(b) and (c)).

Our first contribution is to automatically predict the number of clusters for the spatial locations of gaze data using *relative eigen quality* (REQ) in a normalized cut framework. REQ has recently had success in the computer networks community [Shea and Macker 2013], and we have discovered that it is highly suited to gaze locations, achieving all three ideal eyetracking clustering characteristics, as outlined by Santella and colleagues [2004]. First, it does not require random initialization and has stable results. Second, it automatically detects the number of clusters. Third, it is outlier resistant.

Our second contribution is a semi-automated system to generate segments on a scanned comic panel for the purpose of effects generation. We demonstrate results on a variety of comic book images. We show that these segments can be used to create visual effects such as recoloring the background/foreground, adding a stereoscopic disparity to the important objects, defocusing the background, and simple animations.

Our third contribution is to explore the use of saliency-based visually attended locations as an alternative to eyetracking data. We use a Winner-Take-All network [Walther and Koch 2006] on our comic images to generate the visually attended locations. This network uses a saliency map as one of its inputs, and we demonstrate results with maps from two different saliency models: the model proposed by Judd et al. [2009] and the model proposed by Itti et al. [1998]. Clustering and segmentation are then performed on these saliency-based visually attended locations. We compare these generated locations to the scanpath locations generated by a saccadic model [Le Meur and Coutrot 2016; Le Meur and Liu 2015] that uses the saliency map from Judd et al. [2009] as input. We qualitatively compare the usefulness of the saliency-based visually attended locations generated by these two methods with actual recorded eyetracking data. All our results are generated on public domain comics.

2. RELATED WORK

Image segmentation: Automatic image segmentation has been studied extensively in computer vision. Individual works are too numerous to list, but there exist established benchmarks [Arbelaez et al. 2011] that compare many state-of-the-art algorithms with

different strengths. For computer graphics applications, and especially for comic and graphic novel data, not all objects are equally important. The importance of an object in a panel or frame depends on many complex factors, such as context, and narrative. Therefore, user-driven segmentation tasks [Mortensen and Barrett 1995; Levin et al. 2008] that allow for extraction of objects and mattes have been more successfully adopted by artists and casual users.

An active vision approach to segmentation by Mishra and colleagues [2009] proposed the idea of segmenting an object by finding the enclosing contour around any given fixation point. They suggest using a graph cuts framework where the edge weights are derived from cues such as color for monocular images, and stereo and motion for video data. The assumption here is that every fixation is informative, and there is no attempt to filter out stray fixation points or to look for consensus across fixations from multiple viewers. In our work, we leverage gaze locations from multiple viewers to resolve the following question: Among all the "segmentable" objects in a scene, which ones should be highlighted via digital effects? Because our contribution focuses on clustering gaze locations from multiple viewers, it complements this previous segmentation technique. Ramanathan and colleagues [2010] extend the former algorithm to use more than one fixation point as a seed for finding object boundaries. Although successful in finding a closed contour around the object of interest, the resultant contour does not always demarcate a semantic object. In contrast, our challenge is to segment an entire semantic object for the purpose of applying effects. If object boundaries are not followed exactly, then effects such as recoloring or animation will get applied to portions of the background as well as the desired object. We found color-based superpixels to be a good choice for the over-segmentation of a region of interest for comic images.

Clustering gaze data: Gaze data are clustered either as a preprocessing step, where the goal is to differentiate between fixations and saccades [Urruty et al. 2007; Blignaut 2009; Salvucci and Goldberg 2000], or as part of a gaze-driven algorithm, where the goal is to find regions of interest in an image [Ramanathan et al. 2010; Santella and DeCarlo 2004; Katti et al. 2010; Katti 2011]. Our method falls in the latter category. In this category, previous efforts have applied unsupervised clustering techniques such as k-means [Jain 2012], Gaussian mixture models [Santella and DeCarlo 2004], and mean-shift algorithms [Špakov and Miniotas 2015] on gaze data. Katti et al. [2010, 2011] proposed a binning algorithm to cluster gaze data based on the fixation sequence and a distance threshold that determines when a fixation has crossed over to the next region of interest. Typically, this method would require that each subject's gaze data be processed individually. In our case, we cluster multiple subjects' gaze data together as the consensus among individuals helps identify the semantically important objects. Additionally, the distance threshold in the binning method is dependent on the scale of the objects in the image. In a comics application, we regularly encounter different scales, in the form of wide, medium, and close-up shots. Setting a different threshold for every input comic panel would be a tedious process. Other efforts for clustering gaze data have focused on the scanpath in addition to the gaze locations [Goldberg and Helfman 2010].

A variety of online learning techniques have also been proposed for clustering gaze data on video [Tafaj et al. 2012], although these are less relevant to this work, since we focus on individual images in this article. We utilize normalized graph cuts on the gaze locations, and one of our key contributions is to address the issue of determining the number of clusters through the application of a recent breakthrough in the computer networks community [Shea and Macker 2013].

Eyetracking for image understanding: Gaze points have been previously used as a proxy for user priorities, for example, for cropping photographs [Santella et al. 2006]. For comics in particular, it has been shown that the comic artist is successful in leading

a viewer's gaze to specific objects in the image [Jain et al. 2012]. A number of hybrid techniques have been proposed to combine computer vision with gaze data for the specific goal of segmentation. In Karthikeyan et al. [2015], eyetracking data are used to select from the many outputs of a segmentation algorithm that operates on video. This helps with object tracking and detection. Papadopoulos et al. [2014] integrates gaze data into a machine-learning framework to do inference on object classes and in this sense is similar to Sugano et al. [2013], who build a network containing both pixel differences and regions weighted by their gaze importance to infer the order in which viewers looked at different regions. Our work differs from these prior works, because our goal is to oversegment the input image and then assemble these fragments using gaze data to recover the boundaries of an object of interest.

Effects on images: A number of previous works consider the problem of segmenting comics for the purpose of recoloring. Sykora and colleagues [2003] consider a style of comics where bold contours and homogeneous grayscale intensity demarcates regions to be recolored. Works that apply to more general comic art styles use some form of user assistance, such as seeds or scribbles [Qu et al. 2006; Zhang et al. 2009; Sỳkora et al. 2009; Aramaki et al. 2014]. Implicit user input in the form of gaze data has been demonstrated to be powerful for creating painterly renderings [DeCarlo and Santella 2002] and camera moves on comic panels [Jain 2012].

Additionally, there is a large body of work on augmenting photographs via visual effects such as recoloring [Chang et al. 2015; Levin et al. 2004], stereoscopy [Lo et al. 2010], background defocus [Bae and Durand 2007], simulation effects such as fluids [Sun et al. 2003; Okabe et al. 2011], and foreground object animation, such as objects reacting to wind [Chuang et al. 2005], user-guided animations [Kholgade et al. 2014], and animal motion [Xu et al. 2008]. In all cases where the effect needs to be applied to a particular object (rather than globally, with local constraints), the segmentation is provided by a user. This is the problem we primarily address in our work. The effects themselves are created as simple implementations of known methods to illustrate the quality of the segment created by our method.

3. METHOD

Our method has three main components. The first component is realized through our key contribution: a robust, reliable, and near-automatic method to cluster gaze locations from multiple viewers using REQ to drive normalized cuts. Next, we oversegment the input comic image into color-based superpixels using the well-known Simple Linear Iterative Clustering (SLIC) method [Achanta et al. 2010]. The third step involves assembling the superpixels to create semantic objects. While an artist could manually perform such an assembly using scribbles, for example, we attempt to use viewer gaze locations as an implicit, rather than explicit, form of user input.

3.1. Clustering Eyetracking Data

The inputs to our segmentation algorithm are a digitized comic image and the associated gaze locations from all users for that image. Our key idea is to predict the natural number of clusters in this data using REQ within the graph cuts framework. REQ is a signal-to-noise metric that has recently had success in the computer networks community [Shea and Macker 2013], and here we show that it is well suited to our problem.

Let us assume that each gaze location p_i is a vertex in a graph G, and the edge weights w_{ij} are proportional to the Euclidean distance between the ith and the jth gaze points. We construct a weighted adjacency matrix W for this graph and the diagonal matrix D, such that $d_{ii} = \sum_j w_{ij}$. Then, a partitioning of the graph G can be achieved by solving

the generalized eigenvector problem [Shi and Malik 2000],

$$(\mathbf{D} - \mathbf{W})x = \lambda \mathbf{D}x. \tag{1}$$

The eigenvector corresponding to the second smallest eigenvalue provides a bi-partitioning of the graph G, because it approximates the indicator vector $\mathbf{x_1^l}$, where $\mathbf{x_1^l} = 1$ if the vertex l belongs to one cluster and $\mathbf{x_1^l} = 0$ if the vertex l belongs to the other cluster. In the ideal case (edge weights that perfectly represent the similarity between two points), the eigenvector would be exactly the indicator vector. In practice, a splitting point needs to be selected based on the median values of the elements in the eigenvector (e.g., Shi and Malik [2000]).

After the graph has been divided into two clusters, it may be further sub-divided based on the eigenvector corresponding to the next smallest eigenvalue. This normalized cuts approach to clustering has been widely explored in the context of image segmentation, with problem-specific approaches to determine when to stop partitioning (thus determining the final number of clusters achieved). Thus, the problem of determining the natural number of clusters remains open.

In Shea and Macker [2013], it is shown that a measure of the noisiness of an eigenvector $\mathbf{v}_k$ can be obtained through its REQ, defined as a normalized ratio of its eigenvalue λ_k to the largest eigenvalue λ_{max},

$$REQ(\lambda_k) = \frac{1 - \lambda_k}{\sigma_r}, \tag{2}$$

where σ_r is

$$\sigma_r^2 = \frac{2}{N}\lambda_{max}^2. \tag{3}$$

This REQ value can be computed for all the generalized eigenvalues (Equation (1)). We note that there exist theoretical bounds on REQ values [Shea and Macker 2013],

$$REQ(\lambda_k) \leq REQ(\lambda_0) \leq \left(\frac{N}{2}\right)^{1/2}, \tag{4}$$

with λ_0 being the smallest eigen value and N being the total number of eigenvalues. The implication of the above equation is the following: *If* $\mathbf{v_k}$ *is the eigenvector corresponding to the eigenvalue* λ_k*, then the relative eigen quality provided by* $REQ(\lambda_k)$ *is a measure of how much better a partitioning based on* $\mathbf{v_k}$ *is, relative to a random cut.* To select the number of clusters, the REQs are compared to a threshold that represents the distance (in decibels) that we want to be from the bound in Equation (4).

We now summarize the clustering algorithm:

(1) For each pair of gaze data points, we compute the weight $w_{ij} = e^{\frac{-\sqrt{(y_i-y_j)^2+(x_i-x_j)^2}}{\sigma_x^2}}$ if $\sqrt{(y_i - y_j)^2 + (x_i - x_j)^2} < r$ and 0 otherwise. r and σ_x^2 together represent hard and soft penalties on distances between gaze points, and we set these to 300 and 20, respectively, for all our results.
(2) We solve the generalized eigenvalue problem of Equation (1) using the MATLAB solver *eig*.
(3) We compute the REQ for each eigenvalue as in Equation (2). The number of eigenvalues whose REQ is greater than the selected threshold is the number of clusters.
(4) We bi-partition the eigenvector corresponding to the second smallest eigenvalue to obtain the first two clusters. The mean value of the elements of the eigenvector is the splitting point; elements greater than this value are set to 1 and elements less

than this value are set to 0. This converts the eigenvector into an indicator vector that labels the vertices of the graph as belonging to one cluster or the other.
(5) If further clusters are required (because the REQ of the next larger eigenvalue is still greater than the selected threshold), then we sub-divide each of the above two clusters based on the eigenvector corresponding to this next larger eigenvalue.
(6) As in Shi and Malik [2000], we use a heuristic to avoid sub-dividing already-stable clusters. We do not sub-divide a cluster where the standard deviation of the eigenvector elements is less than 0.01.

Note that the REQ measure is set as an offset, in dB, from a baseline given by the highest possible value in Equation (4) that is determined by the smallest eigenvalue. Critically, although the baseline is data dependent, *the REQ offset does not change*. Therefore, the same REQ offset can specify a different number of clusters for different input data. Intuitively, the REQ offset or threshold can be thought of as the user's expectation of the signal-to-noise ratio in a particular class of signals. In our case, we used a single REQ offset for all comic panels in this article.

If the gaze locations are scaled by a factor λ, then the same REQ threshold can still be used for clustering as long as the two penalties used in computing the weights are also scaled accordingly. The value of r should be replaced by λr and the value of σ_x by $\sqrt{\lambda}\sigma_x$. The resulting clusters after scaling will then be identical to the original clusters.

3.2. Evaluation of the Relative Eigen Quality Metric

Before we describe the second and third components of our method, we discuss two experiments to evaluate REQ. Given a visual stimulus, gaze data contains a signal, that is, what objects the viewers intended to look at. However, there is noise in the measurement of this signal, due to calibration error, and individual differences. Because of this noise, clustering gaze locations is a non-trivial problem. The first experiment tests whether the clusters obtained from normalized cuts based on REQ are stable for different values of the REQ offset or threshold and compares this to the stability of the parameters for the baseline methods of classic k-means and Meanshift. The second experiment compares the parameter choices that would be needed to obtain meaningful clusters via our REQ enabled approach, again, when compared to k-means and Meanshift.

3.2.1. Threshold Sensitivity Evaluation with LabelMe Database. The LabelMe database [Russell et al. 2008] is an extensive collection of real images where objects have been marked with bounding boxes by users and labelled according to their category. Ninety-nine images from this database are also available in the dataset provided by Judd et al. [2009]. Thus, there are 99 images that have eyetracking data from 15 individuals, as well as user-annotated bounding boxes available. We use the LabelMe provided bounding boxes as the ground truth for experimentally testing the sensitivity of the REQ threshold. We perform this evaluation on real-world images rather than on comic images, because the LabelMe dataset is a well-tested benchmark in the computer vision literature.

The intuition is as follows: We assume that if a viewer was looking at an object (e.g., the blue storefront in Figures 2(a) and (b)), then their gaze locations would be inside the bounding box provided by LabelMe. Of course, even the most focused viewer may not be entirely inside the LabelMe provided bounding box all the time, but we treat the bounding box as the closest source of ground truth.

We run our experiment, as shown in Figure 2(c), for our REQ-based algorithm, as well as k-means and Meanshift. For each algorithm, we vary the associated parameter,

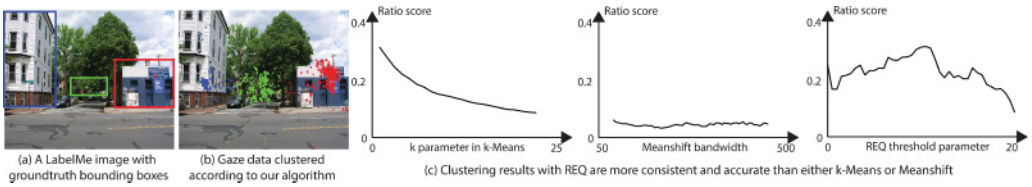


Fig. 2. Threshold sensitivity evaluation with the LabelMe database.

whether it is the number of clusters k for k-means, the bandwidth h for Meanshift or the REQ threshold for our technique, over values that are reasonable for this dataset. The range of values of these parameters for this LabelMe dataset was manually calculated for each algorithm.

For each image, given a particular value of the parameter for one of the three algorithms, we cluster the gaze locations. For each created cluster, we find a bounding box that encloses all the gaze points assigned to this cluster. The width of this bounding box is given by the minimum and maximum x-coordinates of this cluster's gaze points, and the height is computed similarly from the y-coordinates. Let us denote the area of this bounding box to be A. Then, we compute B, the area of the LabelMe provided bounding box inside which the centroid of this cluster falls. We compute the Jaccard index of these two areas, $\frac{\|A \cap B\|}{\|A \cup B\|}$. If the centroid falls in multiple LabelMe provided bounding boxes, then we pick the one that provides the highest Jaccard index, which implies the most meaningful $A - B$ combination.

We compute the average Jaccard index across all clusters for the given image and then across all images in the dataset from Judd et al. [2009]. We denote this average as the *ratio score* and its value across different algorithm parameters is shown in Figure 2(c). If the clustering method is robust, then we expect that the ratio score should be high and should vary smoothly as the algorithm's parameter takes on different values.

As this graph illustrates, the Jaccard ratio score for normalized cuts based on REQ stays in a band across wide range of values for the threshold. Compare this to the Jaccard ratio scores for k-means, as k is changed. We can conclude that the internal parameter for REQ is relatively stable. A possible reason for this could be that k-means creates many small clusters as k is increased, and this is not the most natural clustering for every one of the 99 images, resulting in a low value of the Jaccard ratio score on average. Similarly, while a particular Meanshift bandwidth parameter may be good for a given image, averaging all images results in a low score. This implies that the Meanshift parameter is best selected on a per-image basis and is not very robust across different types of scenes.

We note that there exist a variety of adaptive algorithms beyond the classic implementations of k-means and mean shift, where techniques have been proposed to extract the number of clusters or the bandwidth (sampling density) from the data directly. Our point is that all such algorithms *require some estimation*, and the final result may change based on the accuracy of this parameter. Even though our algorithm also requires the estimation of the REQ threshold, it is a more robust alternative because the Jaccard ratio score remains high over a fairly large range of REQ thresholds.

Our second experiment is a specific example on comic data to illustrate the general conclusions from the LabelMe experiment in the context of our domain. In Figures 3I(a) and (b) we show two examples of input comic panels. We cluster the gaze locations using three different methods. First, we use our approach, normalized cuts based on relative eigen quality. A single REQ threshold value can produce results for both the examples, as shown in Figure 3II. In contrast, classic k-means requires a different number of clusters to correctly capture the clusters in each of the examples, and Meanshift also

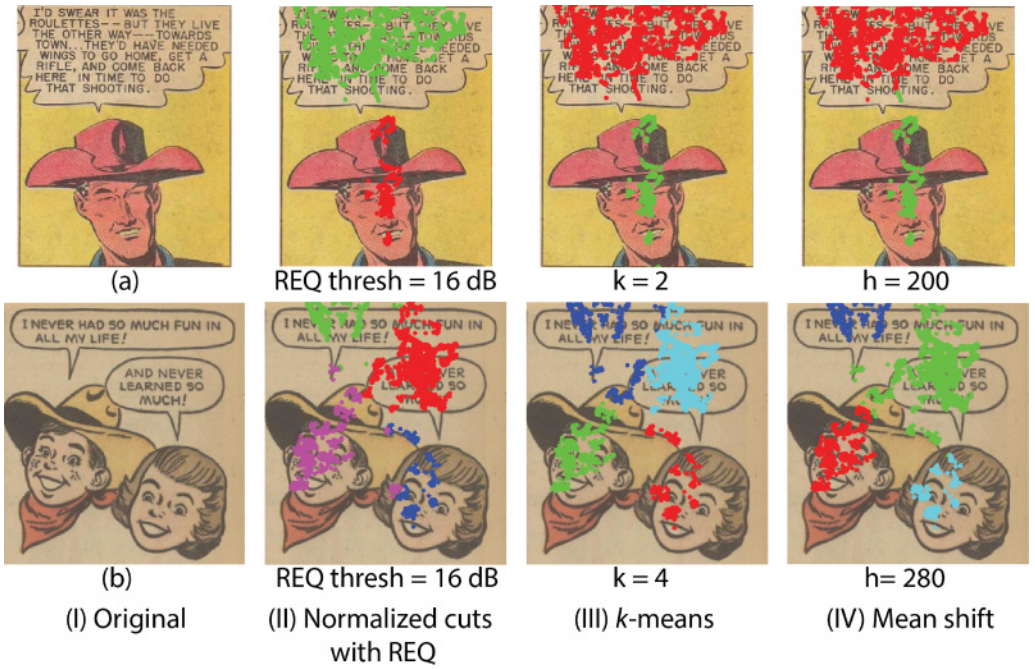


Fig. 3. These two example panels contain two and four natural clusters of gaze locations, respectively. While the same REQ threshold produced these clusters, we needed to set different parameters on k-means and Meanshift to obtain similar results.

requires different bandwidth parameter values for each of the panels. In practice, we use a single REQ threshold for all the comics shown in our results.

3.3. Combining Superpixels Using Clustered Gaze Locations

The second step in our method is to oversegment the input image into superpixels [Achanta et al. 2010]. The SLIC algorithm has two native parameters: the desired number of super pixels and the superpixel convexity measure. We expose these two parameters to the user in our Graphical User Interface (GUI) to allow for tweaking these parameters for different scene types. Our primary motivation for using superpixels is simplicity and speed, though any number of segmentation techniques can be used to create initial candidate segments, for example, recent work on extracting high-level image representations via convolutional neural networks [Farabet et al. 2013].

The third step in our method is to utilize the gaze clusters created by the REQ-based normalized cuts framework to extract the important objects of a comic panel. For each cluster on a given panel, we first perform outlier removal by discarding the gaze points that are M standard deviations away or more from the centroid of the cluster. We provide three options for M for the x- and y-axis; we remove outliers if they are 2, 1.5, or 1 standard deviations from the mean, representing low, moderate, and high removal of outliers. For all remaining gaze points, we compute the number of points that lie within a superpixel (similar to the bounding box test performed in the previous section). If this count is greater than a user-defined gaze density t, then that superpixel is marked as a fragment of an important object. All such superpixels are combined to create a final segment or object. Figure 4 shows how superpixels are combined to create the final segmentation. We encourage the reader to see that the objects extracted are indeed created from putting together fragments based on the gaze locations.

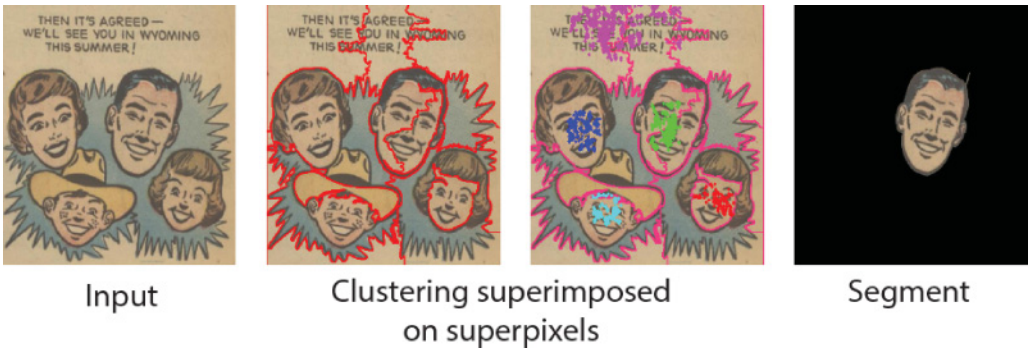


Fig. 4. Our method reassembles superpixels into segments.

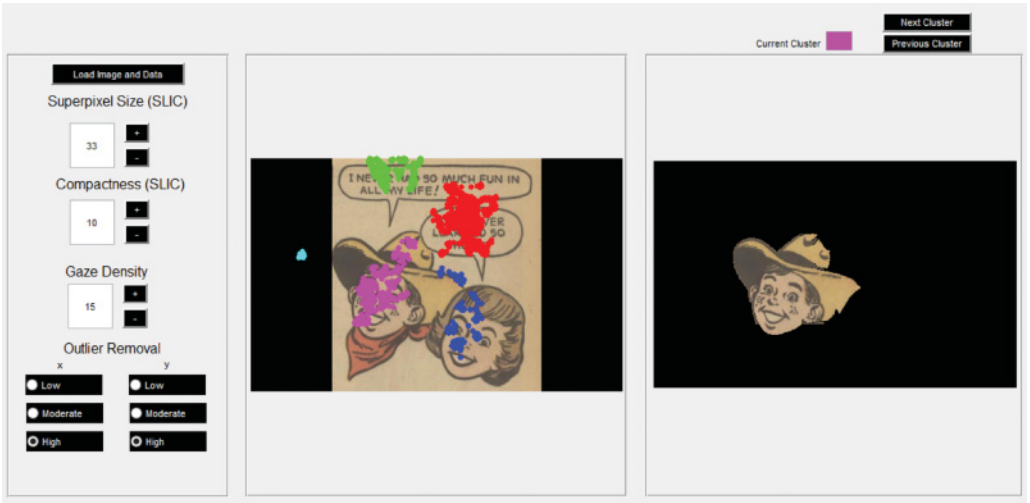


Fig. 5. Our GUI for Segmentation.

Our GUI works in near real time, and a user is able to specify the tunable parameters to intuitively recover the segments for any particular image in the set (Figure 5). The GUI exposes all five relevant parameters: the SLIC algorithm parameters, the gaze density, and the outlier removal bounds in x and y. In Section 3.4, the range of these values is discussed. We have found that most panels need moderate outlier removal, and the user does not need to deviate much from the default SLIC parameters that we pre-load into the GUI. The gaze density parameter t can change across results, and in practice we have seen that a good segment can be found quickly.

Though the segmentation step contains several user-specified parameters, the SLIC algorithm parameters, the outlier removal bounds, and the gaze density parameters are specific to the segmentation method we chose, primarily for its ease of implementation. These parameters could be eliminated with a different choice.

3.3.1. Comic Segmentation Results. In Figure 6, we show a few segmentation results of our approach. The figure contains results shown in sets of three, where the first column is the original input image, the second column shows the gaze locations processed by our REQ-based normalized cuts algorithm, and the color-coded gaze clusters. The third column displays the output segmentation.

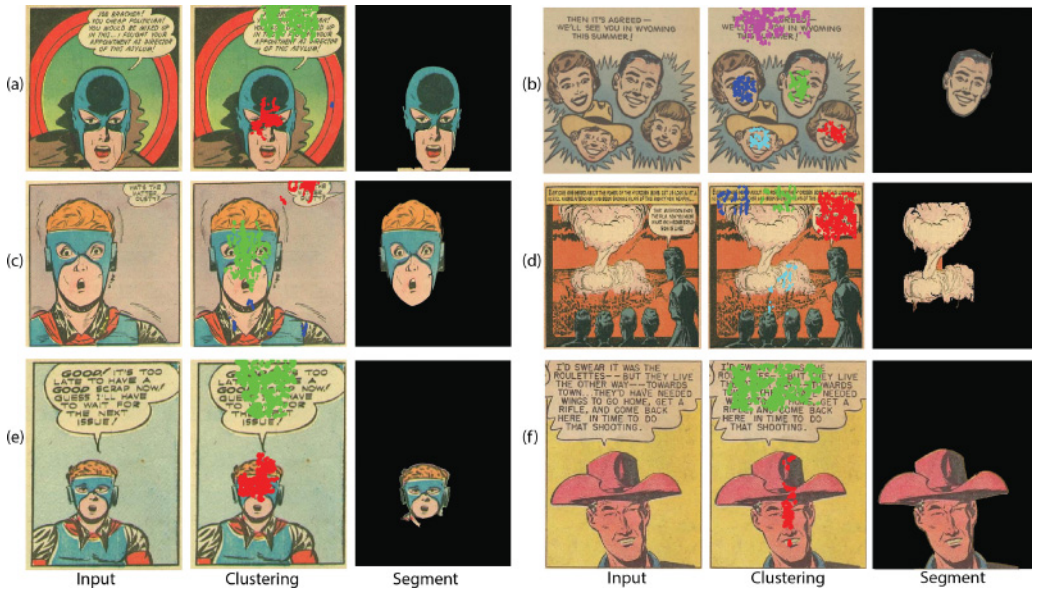


Fig. 6. Segmentation using eyetracking and REQ-enabled normalized cuts on six comic panels.

Figure 6 shows results on a few public domain comics from the Internet Archive [Archive 2015], some of which are many decades old. Though the color palettes in these comics are limited, naively segmenting regions based on color, or even a naive application of superpixels, does not result in segments that can be used for visual effects because the notion of a complete object is missing. Our method is able to extract compelling and complete objects from each of these images. In Figure 6(a), the extracted segment contains the character's mask and face. Figure 4 illustrates how gaze locations cause superpixels to be assembled into complete segments.

3.3.2. Evaluation of Segmentation Results. Figures 7(a) and (b) show the quantitative evaluation of the comic segments presented in this work. As in the evaluation in Section 3.2.1, we compute the Jaccard score for each of the comic segments shown in this article when compared with a ground-truth segment. We also repeat this comparison for different REQ thresholds (the threshold used to generate the results, ± 5 dB, ± 10 dB). As with the LabelMe evaluation, we see that the results are fairly consistent across a range of REQ thresholds. Lower scores occur when the change in REQ threshold results in a change in the number of gaze clusters, causing two or more objects in the image to become inseparable during segmentation. We also show the result segment and the ground-truth segment that earned the highest and lowest Jaccard score with our original parameters in the figure. We report the results when the REQ threshold is 1.5dB (cyan markers) as well, as this value was the chosen value for panels from a more modern comic style (reported in Section 3.4). This threshold does not work well for the comic style of the legacy comics.

In Figure 7(a), the ground-truth segments were generated by two of the authors. Although this ground truth is biased by the authors' knowledge of the final application, this graph allows us to evaluate the segments for the particular application that we demonstrate in our results. We also repeated the evaluation with ground-truth segments generated by two naive volunteers (Figure 7(b)). Their task was to pick the object in the image that they would like to see animated. These volunteers were not aware of the kind of animations that we intended to apply, for example, a bobbing head.

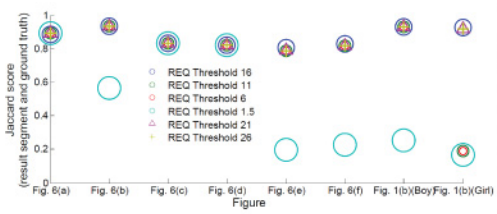
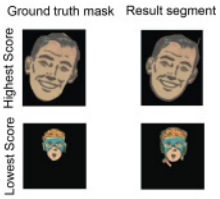


Figure	Jaccard Score (REQ threshold = 16)
Fig. 6(a)	0.8864
Fig. 6(b)	0.9327
Fig. 6(c)	0.8322
Fig. 6(d)	0.8196
Fig. 6(e)	0.8065
Fig. 6(f)	0.8306
Fig. 1(b)Boy	0.9317
Fig. 1(b)(Girl)	0.9256



(a) Ground truth annotations generated when annotators knew the final effects to be applied

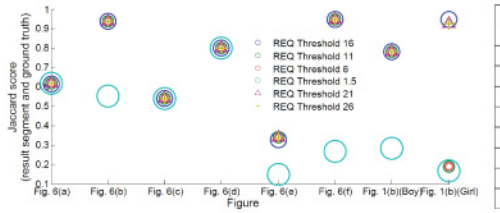
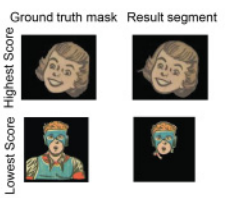


Figure	Jaccard Score (REQ threshold = 16)
Fig. 6(a)	0.6157
Fig. 6(b)	0.9381
Fig. 6(c)	0.5414
Fig. 6(d)	0.8020
Fig. 6(e)	0.3286
Fig. 6(f)	0.9473
Fig. 1(b)Boy	0.7820
Fig. 1(b)(Girl)	0.9474



(b) Ground truth annotations generated when annotators were unaware of the desired final effects

Fig. 7. Quantitative evaluation of segmentation results. The ground-truth masks and our results for the segments with the highest and lowest Jaccard scores are also shown here.

As a result, both volunteers often segmented out the entire object. The ground-truth segment was the intersection of their hand-drawn masks as seen in Figure 7(b). The Jaccard score is generally much lower, because the numerator (intersection of result with ground truth) is much smaller than the denominator. When the naive annotators picked only the head, for example in the girl's face, the score is comparable to the score achieved in Figure 7(a). We anticipate a workflow where the clustered gaze data would be used as a guide to finding effects-worthy objects. A comic artist can then make a creative decision on whether to use a part of the object or the complete object. In that sense, depending on the skill level of the artist (whether it is a novice user, an experienced amateur user, or a professional user), the expectation of the user will be somewhere in between the two types of users that generate the graphs in Figure 7.

3.4. Experimental Details

The gaze data on the public domain comics from Internet Archive [Archive 2015] was collected in our lab using an SMI RED-m eyetracker at 120Hz. Five viewers were asked to look at the comic panels, which were resized to be as large as possible on a 1680×1050 monitor. Each image was shown for 10s. Based on our pre-tests, this was the amount of time it took viewers to read the word bubbles in these older, dialogue-heavy comics. We chose to minimize the variability in the amount of gaze data with an auto-paced study rather than a user-paced study.

We used a total of 23 images from four comics available on Internet Archive for this article.[1] All our results were generated using the REQ threshold of 16dB. The outlier removal parameters were set to moderate ($M = 1.5$) for all but three of our results. For the two SLIC parameters, the compactness ranged from [10 40] while the superpixel size varied from [15 35] across the different examples. Finally, the gaze density parameter for all our results lies between [15 50]. For a more modern set of comics, the REQ threshold was 1.5dB for effective clustering. These results are not presented in this work.

[1]The images, associated gaze locations, and our code can be downloaded from the Downloads page on jainlab.cise.ufl.edu.

The visual angle subtended by our screen and eyetracker setup (18-inch monitor at a 24 to 26-inch viewing distance) is approximately 30° to 35°. The visual angle subtended by a typical tablet (10 inch monitor) held at a comfortable viewing distance (12–15 inches) is approximately 30° to 35°, too. This also holds for a phone (6-inch screen size), held at a distance of 8–10 inches. Comics are typically presented as single panels on small personal devices. Therefore, our data collection setup is a good approximation of the reading conditions in terms of the visual angle subtended. However, the absolute size of the screen differs in the cases of monitors, tablets, and phones. Previous literature has shown that even when humans view an image at a lower resolution, their fixation locations are consistent with the fixation locations measured while viewing the same image at a higher resolution [Judd et al. 2011]. In addition, scanpaths across different sizes of smart phones have also been reported to be similar in the literature [Al-Showarah et al. 2013]. This suggests that the scanpaths are more driven by the content rather than the screen sizes. For comics in particular, the work of Jain et al. [2012] shows us that comic artists deliberately lead the viewer gaze suggesting that readers will look at the same objects regardless of the screen size. Therefore, the scale at which the data were collected likely do not impact the segmentation results. We can obtain the same clusters as long as the internal parameters are scaled appropriately, as described in Section 3.1.

4. EFFECTS FOR COMICS

The output of the previous sections is a set of important objects (segments) extracted from a comic panel. Computer graphics researchers have recently created a large repertoire of effects for still images, such as recoloring [Chang et al. 2015; Levin et al. 2004], stereoscopy [Lo et al. 2010], and background defocus [Bae and Durand 2007]. We demonstrate these three applications on comic data using our segments as a mask for foreground/background separation in Figure 8.

In addition, there has been a spectrum of recent work on animating still images and comics, ranging from applying physical texture models [Chuang et al. 2005] to adding user-created dynamic objects [Kazi et al. 2014]. We demonstrate simple and compelling animations by applying periodic affine transformations onto the segments. While stills of these are shown in the figures, the full animations are available in this article's accompanying video. Finally, adding such animations requires realistic hole filling, and we apply Poisson editing-based techniques that are widely available and have many fast implementations.

The first effect that we describe is stereoscopy, where a stereo pair is generated from the two-dimensional (2D) illustrated comic. We apply a user-defined disparity to the objects of interest that are selected by the segmentation algorithm, although this disparity could be estimated by applying inference algorithms such as shown by Karsch et al. [2012]. The left image is created from the red channel of the segmentation result and the right image comes from a user-defined shift of the blue and green channels of the segmentation result. The left and right images are added to get the stereoscopic effect. Figure 8(a) shows this effect on public domain comics where the character that is currently speaking is perceived to be popping out of the comic panel. Please view these with red-cyan anaglyph glasses, with cyan on the left eye and red on the right eye.

The next effect that we describe is adding a defocus to the background that can lend a sense of depth-of-field to a 2D illustrated comic panel. Given an object segmented by our technique, we created this effect by simply applying a heavy Gaussian blur (standard deviation of about one-fifth of the image diagonal) to the pixels contained in the negative mask of the object. In Figure 8(b), we see this applied to comic panels

(a) Stereoscopic results on comic images

(b) Background defocus results on comic images

(c) Recoloring on still images

(d) Recoloring-based animations (please see accompanying video)

Fig. 8. Effects on public domain comics.

where the expression on the character's face is important to the viewer, since this was where most of the gaze points were present.

In frames where there are multiple characters, such as in the third column of Figure 1, defocusing each character in turn can mimic the gaze attention paths over time, giving the effect of a changing camera focus. In addition to defocus, recoloring comic images can add emphasis to interesting regions and to characters. In Figure 8(c) we show examples where the superhero mask is emphasized. In Figure 8(d) we show that recoloring over time creates animation effects, such as the brightness increase in the mushroom cloud and the alternative emphasis given to the characters as they speak.

In these results, the initial frame of the video is a frame from the comic or its grayscale version. The user specifies both the final desired colors and a required video length. For each pixel in the segmented region, we calculate the distance, in RGB space, from the pixels in the first frame to the desired colors, specified by the user. The video length determines a different step size in 3D color space for each pixel. Subsequent frames are created by adding the appropriate step size obtained above for each pixel in the R, G, and B channels of the segment.

In addition to these color animations, we also create simple affine transformations on some of the segments to create motion animations. These are best viewed in the accompanying video. In the public domain comics in Figure 8, we apply bobbing, chuckling, and shaking animations to the heads of different characters as they speak. These are created by applying a periodic signal, such as a square wave or a sinusoid, to the segment pixels, as the character speaks.

A potential criticism of the technique used here to identify "effects-worthy" regions is that the task of collecting gaze data from many viewers for each comic panel may be too onerous. We therefore evaluate the possibility of using gaze points generated automatically from computational models of visual attention. We investigate the quality of segments obtained from these generated gaze points as compared to the ones from the real gaze locations. In the sections that follow, we discuss the techniques used for gaze points generation and how these data can be incorporated into the framework discussed so far.

5. SALIENCY-BASED VISUALLY ATTENDED LOCATIONS

In this section, we explore two methods for the generation of visually attended locations. First, we employ saliency models coupled with a Winner-Take-all network for the computation of visually attended locations. Next, a more sophisticated approach termed "saccadic model" is used to generate plausible visual scanpaths.

5.1. Computation of Visually Attended Locations Based on a Winner-Take-All Network

To obtain visually attended locations for our comic dataset, we employ two saliency models coupled with a Winner-Take-All (WTA) network. First, we use Walther and Koch's implementation [2006] of the well-known Itti et al. [1998] computational visual attention method. This is a bottom-up approach based on low-level visual features resulting in a final saliency map. From the predicted saliency map, an ordered list of visually attended locations is computed using the MATLAB implementation provided as part of Walther and Koch's toolbox [2006]. As recommended by the authors, we keep the default parameters and generate 1,000 visually attended locations. Visually attended locations generated from this approach are concentrated in certain regions of the image, as shown in Figure 9(a)(left). This model generates a sparse saliency map, seen in Figure 9(a)(right), which in turn leads to the sparsity of the visually attended locations.

We seek an algorithm that results in visually attended locations that cover the image better and are not sparse. Khetarpal and Jain [2016] recently reported that

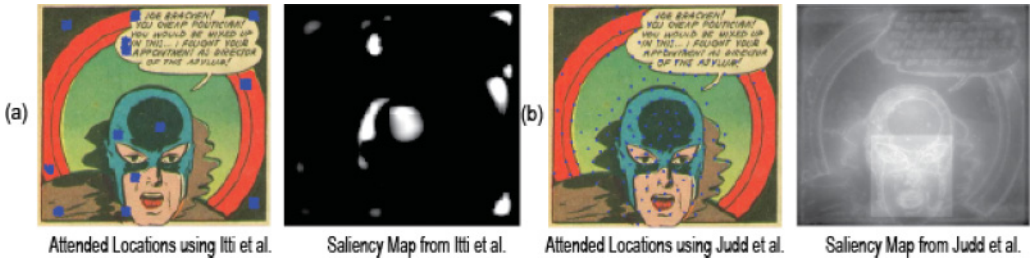


Fig. 9. Visually attended locations based on saliency algorithms and Winner-Take-All network. (a)(left) One thousand visually attended locations (overlaid in blue colored points) generated using Itti Koch's bottom-up visual attention system and (right) the saliency map obtained from Itti Koch's visual attention system. (b)(left) Two hundred visually attended locations using Judd et al's model of saliency and (right) the saliency map obtained from Judd et al's saliency model.

the method of Judd et al. [2009] performs well for predicting eyetracking data in the context of comic art. In addition, Judd et al. [2009]'s model of saliency is among the top 10 performing models for predicting eye fixations on the CAT2000 dataset [Borji and Itti 2015], as reported by the MIT Saliency Benchmark [Bylinskii et al. 2016]. Based on this evidence, we chose this as the second model to compute saliency maps for our experiments. By using an approach similar to the one described above, we generate 200 such attended locations in the decreasing order of saliency. We perform experiments with several variations of three main parameters, namely the resolution of the saliency map (mapLevel), the size of the inhibition of return applied to the winners (foaSize), and the decay rate of Inhibition Of Return (IOR) defined by IORdecay. Based on these experiments, we choose the values of the parameters mapLevel, foaSize, and IORdecay to be 2, −1, and 0.2, respectively.

Figure 10 illustrates the visually attended locations generated by the saliency map of Judd et al. [2009] and the Winner-Take-All network of Walther and Koch [2006] for six randomly chosen comic panels. We observe that using the saliency model of Judd et al. [2009] resulted in scattered locations with a much larger spread (Figure 9(b)(left)) as compared to the attended locations obtained from the saliency algorithm of Itti et al. [1998] in Figure 9(a)(left). The saliency map in Figure 9(a)(right) itself shows sparse regions being marked highly salient by the method of Itti et al. [1998]. As a result, the visually attended locations are concentrated in these sparse regions. On the other hand, Figure 9(b)(right) shows a lower relative change in saliency in the context of adjacent regions. This in turn produces attended locations that are better spread out.

Exploration of a More Sophisticated Saccadic Model. In addition to the classic Winner-Take-All approach, more sophisticated saccadic models have been recently proposed for modelling the way observers deploy their gaze while viewing a stimulus on the screen [Le Meur and Liu 2015; Le Meur and Coutrot 2016]. A saccadic model aims to generate plausible visual scanpaths, that is, the sequence of fixations and saccades. Predicted scanpaths result from the combination of three components: a bottom-up saliency map, memory mechanism, and viewing biases. The first component, a bottom-up saliency map, results from a classical saliency model. The memory mechanism is related to the inhibition of return. The third component is related to the viewing biases that reflect the way we look within a scene. Viewing biases can be represented by the joint distribution of saccade orientation and amplitudes and are learned from eyetracking data.

The generation of visual scanpaths using such a saccadic model adopts the following procedure: from an initial fixation point that has been randomly chosen, a Markovian stochastic process is used to define the next fixation point according to the probability

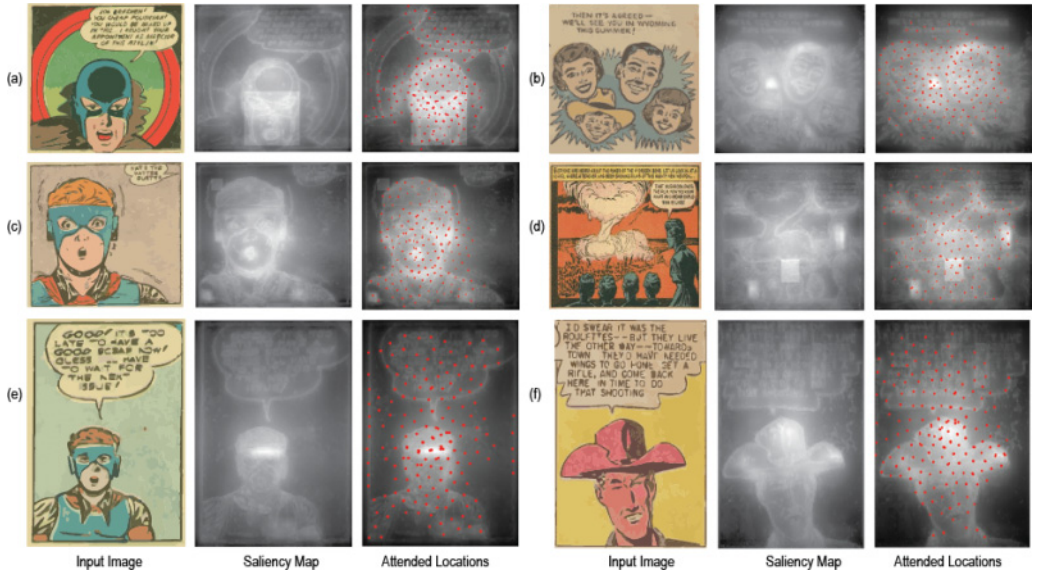


Fig. 10. Computation of saliency-based visually attended locations using Judd et al.'s saliency model coupled with Winner-Take-All network on six comic panels. In each comic panel, (left) the original input image, (middle) the saliency map generated using Judd et al.'s saliency model, and (right) the visually attended locations marked in red points have been overlaid on saliency maps.

Fig. 11. Saccadic model. From left to right: the first, the first four, and the first eight fixation points. (a) Combination of bottom-up saliency maps, viewing biases and memory map; (b) the generated scanpath. The first fixation point is represented by the blue circle. The radius of circle represents 1° of visual angle.

density function defined by the combination of the three aforementioned components. The stochastic sampling used in this saccadic model is fundamental to reflect the stochasticity of the visual perception [Merk and Schnakenberg 2002]. This procedure is illustrated in Figure 11.

We utilize the MATLAB implementation of the saccadic model proposed in the work of Le Meur et al. [2015, 2016] in the context of our study. This saccadic model uses the saliency model of Judd et al. [2009] as input. More importantly, a new joint distribution of saccades orientation and amplitudes was learned from eyetracking data collected on

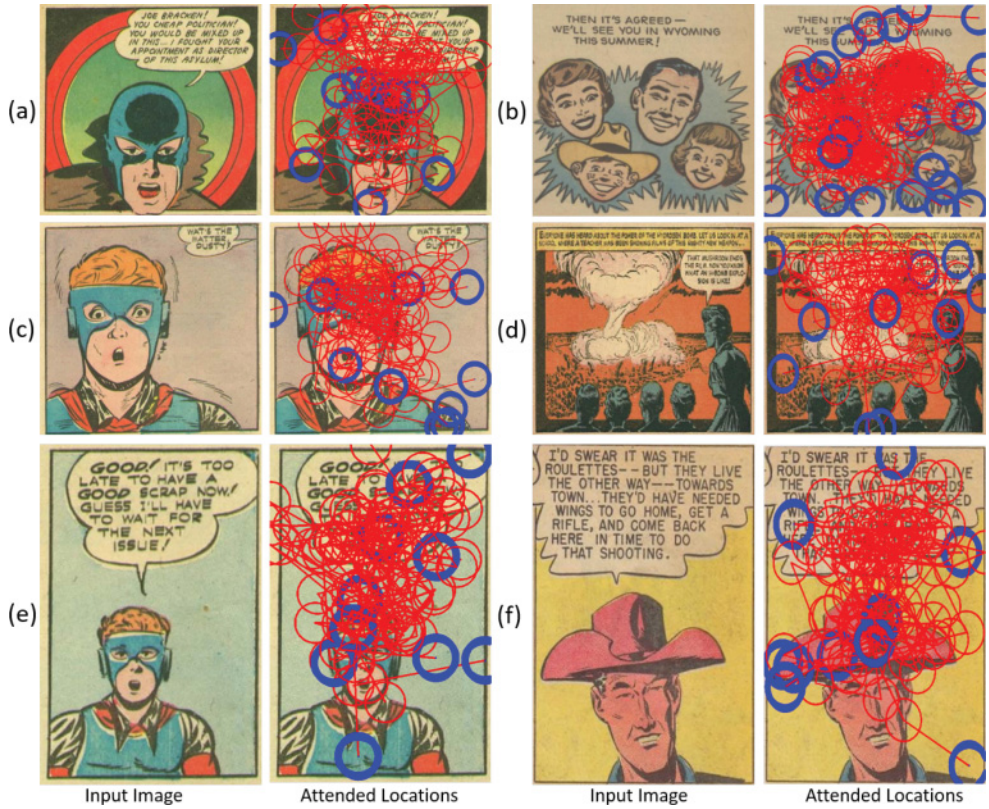


Fig. 12. Computation of saliency-based visually attended locations using the saccadic model proposed by Le Meur et al. [Le Meur and Coutrot 2016; Le Meur and Liu 2015]. Ten predicted scanpaths each composed of 10 fixations are generated for the six comic panels. The first fixations are represented by the blue circle. The radius of circle represents 1° of visual angle.

comics. We generate scanpaths, each composed of 10 fixations, and these are shown in Figure 12. Though the predicted scanpaths generated from the use of saccadic model share some similarities with eyetracking data, such as saccades amplitudes and orientations, the predicted fixation points do not fall within the most visually important areas (for example, the faces of the characters). This could be because the input saliency map is not well suited to the task. Based on this investigation, we generated all further results using the Winner-Take-All network.

5.2. Clustering and Segmentation Based on Generated Visually Attended Locations

For the input images shown in Figure 6, we qualitatively compare the visually attended locations generated using the saliency map from Judd et al. [2009] and the Winner-Take-All network, with the real gaze locations. We will henceforth refer to the visually attended locations thus obtained as generated gaze locations. We first apply the REQ-based clustering on the generated gaze locations and then create segments from these clusters. The motivation behind this investigation is that collecting gaze data from multiple viewers for each comic panel may be tedious and time consuming. If saliency maps are a sufficient substitute, then our method would be practically applicable at a much larger scale. The clustering and segmentation results obtained with the generated gaze locations are shown in Figure 13. The REQ offset for clustering is

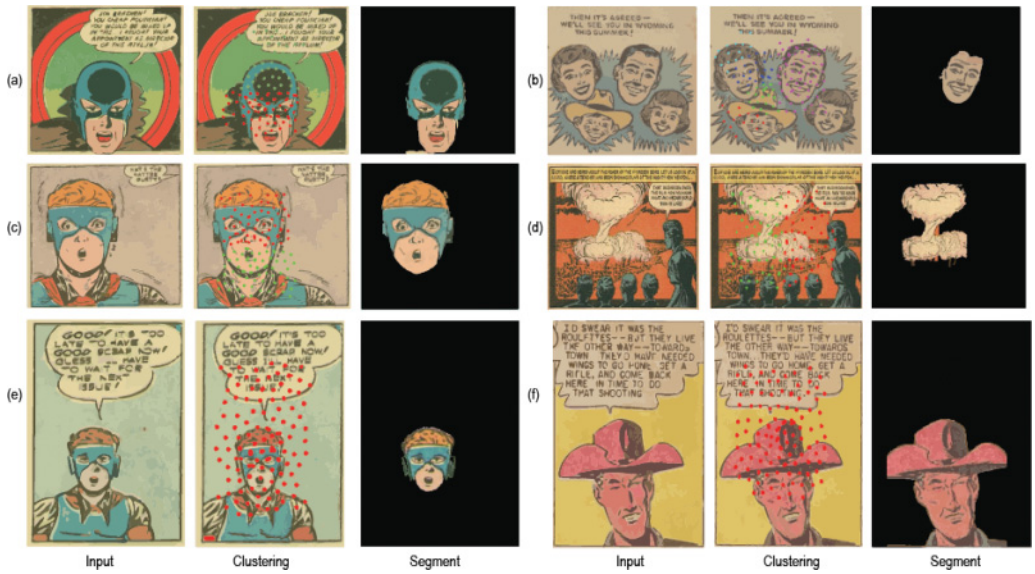


Fig. 13. Clustering and segmentation using visually attended locations.

set to 8 for all of the images. As before, the parameters exposed in the segmentation GUI are varied to produce the segments.

We find that the generated gaze locations do not resemble the real gaze locations. The 2D Winner-Take-All network used to generate the gaze points relies heavily on the saliency map input, which in turn depends on the model that produces it. Starting at the region marked as the most salient in the map, this network generates as many gaze points as the selected number of iterations. If the number of iterations is too high, then we force the network to pick more number of gaze points, and so they may lie far away from the salient region. If the number of iterations is too small, then we get too few points, highly focused near the salient region. This region is then too small to serve the purpose of segmenting a complete object. In our experiment, we selected 200 iterations as the tradeoff between a too-high and a too-low value for the number of gaze points.

The choice of the number of iterations and other parameters for the Winner-Take-All network makes the generated gaze locations for these set of comic images resemble uniformly generated points to some degree. A question that must then be addressed is whether the use of such uniform points eliminates the need for any gaze location generation. Figure 14(a) shows the result of clustering uniform points on three images using REQ. Clearly, the clusters of the uniformly spaced grid points convey no information about object boundaries or the importance of a region for effects. Therefore, they cannot substitute the real gaze points. Figure 14(b) shows the segments obtained from the uniform points. The presence of points at regular intervals damages the quality of the segments obtained by including some surrounding fragments. It is easy to see that even the outlier removal step will not improve the segments much. Adding effects to segments with these extra details may lead to visually unappealing results. Hence, we conclude that even though the generated gaze locations appear to be "uniformly distributed," they are, in fact, quite informative. We now proceed to discuss the segments obtained from these generated gaze points.

Interestingly, the segments obtained from the generated gaze locations shown in Figure 13 are comparable to those obtained from the actual gaze locations. The slight

(a) Clustering of uniform points using REQ

(b) Segmentation results from uniform points

Fig. 14. Clustering and segmentation using uniform points.

variations observed in the segments of Figure 6 and Figure 13 can be attributed to two factors. First, the parameters set for the SLIC algorithm decide what kind of superpixels are obtained. Second, there should be enough gaze points available in these superpixels in order for them to be considered a fragment of the object of interest. In Figures 13(b) and (c), the non-availability of gaze points is the reason for the missing hair of the man and the ear of the boy. However, in Figure 13(f), the combination of SLIC parameters result in the man's face and the part of the hat connected to it being one superpixel. Thus by setting the gaze density correctly, we are able to segment out the man and avoid the text. However, the clusters obtained from the real gaze locations are naturally concentrated on the different objects in the image (Figure 6). The points in a single cluster are tightly packed, leaving little room for ambiguity about the cluster to which a given point should belong. These clusters are hence easy to understand intuitively as compared to the clusters from the generated gaze locations.

6. DISCUSSION

In this work, we considered the problem of generating digital effects to augment traditional comics. We demonstrated a method to automatically determine the most "effects-worthy" regions by leveraging eyetracking data as a proxy for narrative importance. We clustered gaze points from multiple viewers by introducing relative eigen quality from the computer networks literature. We combined superpixels obtained from the SLIC algorithm on the basis of these clusters to yield the segmented object of interest. We showed a variety of applications in which such a segmentation plays a role. We also examined whether saliency-based visually attended locations can substitute real gaze locations. Two existing saliency map techniques were evaluated for their potential in generating visually attended locations. We converted the saliency maps from these techniques to artificial gaze locations using a Winner-Take-All network. Using these visually attended locations, we repeated our experiments of clustering and segmentation. We also studied the use of a saccadic model for generating artificial gaze points.

The segments obtained from the generated gaze points using a WTA network appear comparable to the ones from the real gaze locations on initial visual inspection. However, the saliency-based visually attended locations qualitatively differ from the more intuitively clustered actual gaze locations. A potential reason for this could be the role of central bias in saliency models. The saliency map model described in Itti et al. [1998] is based on features such as intensity, color, and so on. These may not be the best parameters to judge the importance of a region in a comic image. For example, in Figure 9,

the intensity of pixels in the superhero's mask is very low, and thus the saliency map does not include it. However, from the story and the reader's aspect, the superhero's mask is definitely significant. On the other hand, the saliency map from Judd et al. [2009] performs relatively better in terms of identifying salient regions in the comics. The model used in this work though was originally trained on natural images and not comic images. It is evident from Figure 10 that this model also misses some important regions in the image such as the text. Additionally, even a more sophisticated approach like the saccadic model [Le Meur and Liu 2015; Le Meur and Coutrot 2016] resulted in gaze locations that did not match eyetracking data for comic images as observed in Figure 12. These results are also evidence of the inadequacy of the classical saliency algorithms. We believe that the initial results look promising and it may be possible to close the gap by building saliency models specifically targeted towards comic art.

An alternative to our proposed approach could be to generate an importance map using gaze locations from comic readers and allow the artist to look at this map for guidance, while providing explicit scribble-based input to assemble superpixels. It would be interesting to compare such alternative interfaces with our approach of using gaze locations as an implicit form of input. A user interface comparison experiment could address questions such as reduction of effort, time taken to generate a desired effect, and the skill level required to generate a desired segmentation.

A limitation of this work is that the added effects shown in Figure 8 have not been evaluated by having actual viewers rate them as "Engaging" versus "Distracting." We note that selecting the particular effect to apply is an aesthetic/commercial choice, and so we present several possible effects that are enabled by our method, leaving the actual choice in the hands of the artist/editor. Our key insight is that REQ is a metric that works remarkably robustly in the context of gaze locations. Though we have a number of parameters that are exposed to the user via a GUI, these parameters influence the assembly of superpixels into segments, not the clustering process. A single REQ threshold is used to cluster gaze locations on all the comic panels in our results. We have found that obtaining similar results using classic k-means or mean shift clustering would require a user to set different parameters in each case. Further, two of the parameters for the segmentation come from the SLIC algorithm. Also, as seen in Figure 2, a large range of REQ thresholds produce similar clustering results, indicating that there is a very small chance of selecting a wrong threshold in this step. It would be interesting to apply the REQ-based segmentation to gaze locations on natural images and evaluate its performance.

Though our dataset is fairly small (23 images), we believe that the results nonetheless are a persuasive proof of concept for the key ideas proposed in this work. Extending this work to a wider variety of comics remains an interesting direction of future work. Recent works suggest that eyetracking-based techniques can be deployed in the real world by using cameras on phones and laptops instead of special-purpose eyetrackers [Choi et al. 2016; Krafka et al. 2016]. These could be employed to generate larger databases using crowdsourced comic readers.

7. FUTURE WORK

Hundreds of comics and graphic novels are released each month [Comichron 2015], and tens of thousands already languish in archives [ComicBookPlus 2015; Archive 2015]. These represent a massive amount of creative effort, and our focus in this article is a fast, semi-automatic method to update these comics with certain classes of digital effects. Fully resurrecting these comics into modern media is a fascinating topic of research. Artifacts such as spatial dithering offer both a challenge [Kopf and Lischinski 2012] and an opportunity for future work. Another interesting direction includes using temporal information to figure out the order in which multiple effects

should be applied. For example, temporal information may decide the order in which two faces are animated. We may also want to group faces and word bubbles and to animate the appropriate word bubbles as the comic is being read.

Yet another interesting direction could be the creation of a standardized database of comic images with associated eyetracking data and ground-truth bounding boxes, similar to the LabelMe dataset. Such a database will facilitate quantitative comparisons of algorithms specific to comic images.

Saliency-based computation of gaze points is one of the ways to make eyetracking-based algorithms more practical. The computation of visually attended locations by employing saliency models relies heavily on the quality of the predicted saliency map to match actual gaze locations. Therefore another potential direction for future work would be to investigate if recently developed deep-learning-based models could generate better visually attended locations for applications in comic art (for example, Kruthiventi et al. [2015], Jiang et al. [2015], Kümmerer et al. [2016], and Cornia et al. [2016]). These models have been trained on eyetracking data collected on natural scenes. This is of fundamental significance, because recent saccadic models have shown that the way we deploy our gaze depends on the type of content [Le Meur and Coutrot 2016]. Whether deep-learning models address this is thus an open question in the context of comics.

REFERENCES

Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. 2010. *Slic Superpixels*. Technical Report.

Suleyman Al-Showarah, Naseer Al-Jawad, and Harin Sellahewa. 2013. Examining eye-tracker scan paths for elderly people using smart phones. In *Proceedings of the 6th York Doctoral Symposium on Computer Science & Electronics*, Vol. 1. 7.

Yuji Aramaki, Yusuke Matsui, Toshihiko Yamasaki, and Kiyoharu Aizawa. 2014. Interactive segmentation for manga. In *Proceedings of ACM SIGGRAPH 2014 Posters*. 66.

Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. 2011. Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* 33, 5 (May 2011), 898–916. DOI:http://dx.doi.org/10.1109/TPAMI.2010.161

Archive. 2015. Internet Archive. Retrieved from http://archive.org/.

Soonmin Bae and Frédo Durand. 2007. Defocus magnification. In *Computer Graphics Forum*, Vol. 26. Wiley Online Library, 571–579.

Pieter Blignaut. 2009. Fixation identification: The optimum threshold for a dispersion algorithm. *Attention Percept. Psychophys.* 71, 4 (2009), 881–895.

Ali Borji and Laurent Itti. 2015. CAT2000: A large scale fixation dataset for boosting saliency research. In *Proceedings of the Computer Vision and Pattern Recognition 2015 Workshop on "Future of Datasets"*

Zoya Bylinskii, Tilke Judd, Ali Borji, Laurent Itti, Frédo Durand, Aude Oliva, and Antonio Torralba. 2016. MIT Saliency Benchmark. Retrieved from http://saliency.mit.edu/.

Huiwen Chang, Ohad Fried, Yiming Liu, Stephen DiVerdi, and Adam Finkelstein. 2015. Palette-based photo recoloring. *ACM Trans. Graph.* 34, 4 (2015), 139.

Jinsoo Choi, Tae-Hyun Oh, and In So Kweon. 2016. Human attention estimation for natural images: An automatic gaze refinement approach. *arXiv Preprint arXiv:1601.02852* (2016).

Yung-Yu Chuang, Dan B. Goldman, Ke Colin Zheng, Brian Curless, David H. Salesin, and Richard Szeliski. 2005. Animating pictures with stochastic motion textures. *ACM Transactions on Graphics (TOG)* 24, 3 (2005), 853–860.

ComicBookPlus. 2015. Comics in public domain. Retrieved From http://comicbookplus.com/.

Comichron. 2015. Comichron sales data. Retrieved from http://www.comichron.com/monthlycomicssales.html.

Marcella Cornia, Lorenzo Baraldi, Giuseppe Serra, and Rita Cucchiara. 2016. A deep multi-level network for saliency prediction. *arXiv Preprint arXiv:1609.01064* (2016).

Doug DeCarlo and Anthony Santella. 2002. Stylization and abstraction of photographs. *ACM Transactions on Graphics* 21, 3 (2002), 769–776.

Clement Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. 2013. Learning hierarchical features for scene labeling. *IEEE Trans. Pattern Anal. Mach. Intell.* 35, 8 (2013), 1915–1929.

Joseph H. Goldberg and Jonathan I. Helfman. 2010. Scanpath clustering and aggregation. In *Proceedings of the 2010 Symposium on Eye-tracking Research & Applications*. ACM, 227–234.

Laurent Itti, Christof Koch, and Ernst Niebur. 1998. A model of saliency-based visual attention for rapid scene analysis. *IEEE Trans. Pattern Anal. Mach. Intell.* 11 (1998), 1254–1259.

Eakta Jain. 2012. *Attention-Guided Algorithms to Retarget and Augment Animations, Stills, and Videos*. Ph.D. Dissertation. CMU.

Eakta Jain, Yaser Sheikh, and Jessica Hodgins. 2012. Inferring artistic intention in comic art through viewer gaze. In *Proceedings of the ACM Symposium on Applied Perception*. ACM, 55–62.

Ming Jiang, Shengsheng Huang, Juanyong Duan, and Qi Zhao. 2015. SALICON: Saliency in context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'15)*.

Tilke Judd, Fredo Durand, and Antonio Torralba. 2011. Fixations on low-resolution images. *J. Vis.* 11, 4 (2011), 14.

Tilke Judd, Krista Ehinger, Frédo Durand, and Antonio Torralba. 2009. Learning to predict where humans look. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Kevin Karsch, Ce Liu, and Sing Bing Kang. 2012. Depth extraction from video using non-parametric sampling. In *Proceedings of the Computer Vision–ECCV 2012*. Springer, 775–788.

S. Karthikeyan, T. Ngo, M. Eckstein, and B. S. Manjunath. 2015. Eye tracking assisted extraction of attentionally important objects from videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3241–3250.

Harish Katti. 2011. *Human Visual Perception, Study and Applications to Understanding Images and Videos*. Ph.D. Dissertation.

Harish Katti, Ramanathan Subramanian, Mohan Kankanhalli, Nicu Sebe, Tat-Seng Chua, and Kalpathi R. Ramakrishnan. 2010. Making computers look the way we look: Exploiting visual attention for image understanding. In *Proceedings of the 18th ACM International Conference on Multimedia*. ACM, 667–670.

Rubaiat Habib Kazi, Fanny Chevalier, Tovi Grossman, Shengdong Zhao, and George Fitzmaurice. 2014. Draco: Bringing life to illustrations with kinetic textures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 351–360.

Khimya Khetarpal and Eakta Jain. 2016. A preliminary benchmark of four saliency algorithms on comic art. In *Proceedings of the 2016 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*. IEEE, 1–6.

Natasha Kholgade, Tomas Simon, Alexei Efros, and Yaser Sheikh. 2014. 3D object manipulation in a single photograph using stock 3D models. *ACM Transactions on Graphics* 33, 4 (2014), 127.

Johannes Kopf and Dani Lischinski. 2012. Digital reconstruction of halftoned color comics. *ACM Transactions on Graphics* 31, 6 (2012), 140.

Kyle Krafka, Aditya Khosla, Petr Kellnhofer, Harini Kannan, Suchendra Bhandarkar, Wojciech Matusik, and Antonio Torralba. 2016. Eye tracking for everyone. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2176–2184.

Srinivas S. S. Kruthiventi, Kumar Ayush, and R. Venkatesh Babu. 2015. Deepfix: A fully convolutional neural network for predicting human eye fixations. *arXiv Preprint arXiv:1510.02927* (2015).

Matthias Kümmerer, Thomas S. A. Wallis, and Matthias Bethge. 2016. DeepGaze II: Reading fixations from deep features trained on object recognition. *arXiv Preprint arXiv:1610.01563* (2016).

Olivier Le Meur and Antoine Coutrot. 2016. Introducing context-dependent and spatially-variant viewing biases in saccadic models. *Vision Research* 121 (2016), 72–84.

Olivier Le Meur and Zhi Liu. 2015. Saccadic model of eye movements for free-viewing condition. *Vision Research* 116 (2015), 152–164.

Anat Levin, Dani Lischinski, and Yair Weiss. 2004. Colorization using optimization. In *ACM Transactions on Graphics (TOG)*, Vol. 23. ACM, 689–694.

Anat Levin, Dani Lischinski, and Yair Weiss. 2008. A closed-form solution to natural image matting. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30, 2 (2008), 228–242.

Wan-Yen Lo, Jeroen van Baar, Claude Knaus, Matthias Zwicker, and Markus Gross. 2010. Stereoscopic 3D copy & paste. In *ACM Transactions on Graphics (TOG)*, Vol. 29. ACM, 147.

I. Merk and J. Schnakenberg. 2002. A stochastic model of multistable visual perception. *Biological Cybernetics* 86, 2 (2002), 111–116.

Ajay Mishra, Yiannis Aloimonos, and Cheong Loong Fah. 2009. Active segmentation with fixation. In *IEEE International Conference on Computer Vision (ICCV)*. IEEE, 468–475.

Eric N. Mortensen and William A. Barrett. 1995. Intelligent scissors for image composition. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*. ACM, 191–198.

Makoto Okabe, Ken Anjyo, and Rikio Onai. 2011. Creating fluid animation from a single image using video database. *Computer Graphics Forum* 30 (2011).

Dim P. Papadopoulos, Alasdair D. F. Clarke, Frank Keller, and Vittorio Ferrari. 2014. Training object class detectors from eye tracking data. In *Proceedings of the Computer Vision–ECCV 2014*. Springer, 361–376.

Yingge Qu, Tien-Tsin Wong, and Pheng-Ann Heng. 2006. Manga colorization. *ACM Transactions on Graphics* 25, 3 (2006), 1214–1220.

Subramanian Ramanathan, Harish Katti, Nicu Sebe, Mohan Kankanhalli, and Tat-Seng Chua. 2010. An eye fixation database for saliency detection in images. In *Proceedings of the European Conference on Computer Vision*. Springer, 30–43.

Bryan C. Russell, Antonio Torralba, Kevin P. Murphy, and William T. Freeman. 2008. LabelMe: A database and web-based tool for image annotation. *International Journal of Computer Vision* 77, 1–3 (2008), 157–173.

Dario D. Salvucci and Joseph H. Goldberg. 2000. Identifying fixations and saccades in eye-tracking protocols. In *Proceedings of the 2000 Symposium on Eye Tracking Research & Applications*. ACM, 71–78.

Anthony Santella, Maneesh Agrawala, Doug DeCarlo, David Salesin, and Michael Cohen. 2006. Gaze-based interaction for semi-automatic photo cropping. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI)*.

Anthony Santella and Doug DeCarlo. 2004. Robust clustering of eye movement recordings for quantification of visual interest. In *Proceedings of the 2004 Symposium on Eye Tracking Research & Applications*. ACM, 27–34.

John M. Shea and Joseph P. Macker. 2013. Automatic selection of number of clusters in networks using relative eigenvalue quality. In *Proceedings of the Military Communications Conference, MILCOM 2013-2013 IEEE*. IEEE, 131–136.

Jianbo Shi and Jitendra Malik. 2000. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22, 8 (2000), 888–905.

Oleg Špakov and Darius Miniotas. 2015. Application of clustering algorithms in eye gaze visualizations. *Information Technology And Control* 36, 2 (2015).

Yusuke Sugano, Yasuyuki Matsushita, and Yoichi Sato. 2013. Graph-based joint clustering of fixations and visual entities. *ACM Transactions on Applied Perception (TAP)* 10, 2 (2013), 10.

M. Sun, A. D. Jepson, and E. Fiume. 2003. Video input driven animation (VIDA). In *IEEE International Conference on Computer Vision (ICCV)*. 96–103.

Daniel Sỳkora, Jan Buriánek, and Jiří Žára. 2003. Segmentation of black and white cartoons. In *Spring Conference on Computer Graphics (SCCG)*. 223–230.

Daniel Sỳkora, John Dingliana, and Steven Collins. 2009. LazyBrush: Flexible painting tool for hand-drawn cartoons. In *Computer Graphics Forum*, Vol. 28.

Enkelejda Tafaj, Gjergji Kasneci, Wolfgang Rosenstiel, and Martin Bogdan. 2012. Bayesian online clustering of eye movement data. In *Proceedings of the Symposium on Eye Tracking Research and Applications*. ACM, 285–288.

Thierry Urruty, Stanislas Lew, Nacim Ihadaddene, and Dan A. Simovici. 2007. Detecting eye fixations by projection clustering. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 3, 4 (2007), 5.

Dirk Walther and Christof Koch. 2006. Modeling attention to salient proto-objects. *Neural Networks* 19, 9 (2006), 1395–1407.

Xuemiao Xu, Liang Wan, Xiaopei Liu, Tien-Tsin Wong, Liansheng Wang, and Chi-Sing Leung. 2008. Animating animal motion from still. *ACM Transactions on Graphics* 27, 5 (2008), 117.

Song-Hai Zhang, Tao Chen, Yi-Fei Zhang, Shi-Min Hu, and Ralph R. Martin. 2009. Vectorizing cartoon animations. *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 15, 4 (2009), 618–629.

Received July 2016; revised March 2017; accepted March 2017