AUTOMATED SKETCHER USING EDGE DETECTION TECHNIQUES

Tissa Chandesa* and Michael Hartley**

Abstract

In this paper, the main objective was to determine whether edge details extracted from an image using edge detection techniques, with the help of image segmentation techniques, could be used to transform a particular image into a sketch. An automated sketcher system was developed and used to perform the transformation process, allowing three hypotheses to be tested and proven. In the end, it was shown that edge detection techniques can be used to obtain a sketch from an image. The success of the transformation process did not depend on the image type. Finally, a survey was conducted to determine which edge detection technique was best suited, to transform an image into a sketch.

Key Words

Edge detection techniques, automated sketcher system, images, sketch, transformation process

1. Introduction

An intensive amount of research and development have been carried out in the field of transforming an image into a sketch and also caricaturing a sketch from an image. Among them includes [1–6]. Most techniques used to transform an image into a sketch are techniques commonly used within the research field of face recognition. These techniques include Bayesian classifier [7], eigenfaces [8], principal component analysis (PCA) technique [9], cited in [7] and elastic graph matching [10]. Some of the technique or method developed utilizes a database that consists of parts of human face templates, which are later used for matching with the original facial image, prior to producing the facial sketch.

Hough transform [11] is another commonly used technique at detecting lines and curves in pictures apart from those described above. However, almost all automated sketcher systems and techniques developed thus far have focused on facial recognition and caricaturing a sketch of a face from an image. Not much research has been done on how to transform a general image scenario into a sketch. Besides that, using other techniques or methods such as edge detection techniques to transform an image into a sketch has not been thoroughly explored. In this paper, an automated sketcher system is described, which allows comparisons between nine different edge detection techniques, to answer these three hypotheses:

- (H1) Can edge detection techniques be used to help transform an image into a sketch?
- (H2) Can the automated sketcher system be applied onto any type of image?
- (H3) Which edge detection technique is best suited for transforming an image into a sketch?

Image segmentation techniques were also applied onto the edge-detected images to produce the sketch.

According to [12], sketches are perhaps the simplest form of drawings because they consist of only lines. However, transforming an image into a sketch is considered as a very complex process because images are in bitmap format but sketches are in vector format. The two formats have different 'modalities', [13]. Nonetheless, this transformation is often an extremely useful one [14].

Edge detection is a kind of image processing technique that is used to outline the boundaries of objects in the image plane [15], cited in [16]. Nine different edge detection techniques, namely, Frei and Chen, Kirsch, Laplacian, Laplacian of Gaussian (LOG), Prewitt, Roberts, Roberts Cross, Robinson and Sobel, were implemented in the automated sketcher. Apart from that, other filtering techniques were also applied onto the original image and edge-detected image, respectively, for the purpose of enhancing the edges in the image and, at the same time, to reduce any unwanted noise in the image. Image segmentation can be defined as a procedure to partition an image into its constituent parts or objects, page 27 of [17]. However, according to pages 27, 567 of [17], autonomous segmentation is one of the most difficult tasks in image processing and can determine the eventual success or failure of a computerized analysis procedure. Two image segmentation techniques were used in the automated sketcher system, a thinning algorithm [18], and an algorithm called 'bugwalk' [19].

The rest of this paper is organized as follows. The framework of the automated sketcher is presented in

* School of Computer Science, Faculty of Science, The University of Nottingham Malaysia Campus, Jalan Broga, Semenyih 43500, Selangor, Malaysia; e-mail: eyx6tc@nottingham.edu.my

** Faculty of Engineering and Computer Science, The University of Nottingham Malaysia Campus, Jalan Broga, Semenyih 43500, Selangor, Malaysia; e-mail: michael.hartley@nottingham.edu.my

Recommended by Dr. L. C. Monticone
(10.2316/Journal.202.2010.4.202-2160)

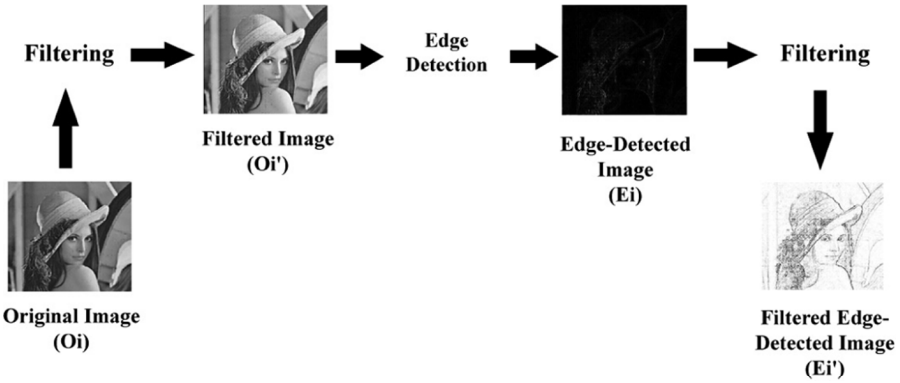

Figure 1. The edge detection phase.

Section 2. The edge detection phase is presented in Section 3. The image segmentation phase is presented in Section 4. Research findings and results from the survey are discussed in Section 5.

2. Framework of Automated Sketcher

The automated sketcher consists of an edge detection phase and an image segmentation phase. Figure 1 show the steps that are undertaken during the edge detection phase. The input to this phase is an original image, Oi, and the output is an edge-detected image, Ei′.

- Original image, Oi, is filtered to enhance the edges in the image and at the same time reduce any unwanted noise in the image. A filtered image, Oi′ will be produced at the end.
- The filtered image, Oi′, is used as input into the edge detection process, which will result in an edge-detected image, Ei, being produced.
- The edge-detected image, Ei, is then filtered and a grey-scale image, Ei′, is produced at the end.

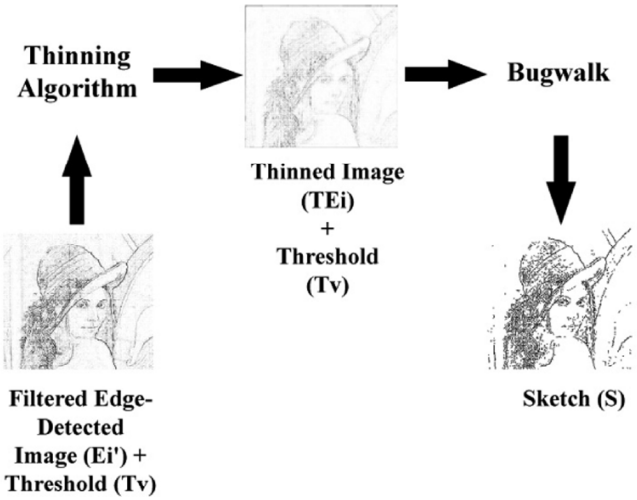

Figure 2. The image segmentation phase.

Meanwhile, Fig. 2 shows the steps carried out during the image segmentation phase. This stage takes in the edge-detected image, Ei′, and produces the sketch, S, at the end of the process.

- The edge-detected image, Ei′, and a threshold value, Tv, are used as input arguments for the thinning algorithm process. During this process, all thick lines and curves in the edge-detected image, Ei′, will be thinned down. At the end of this process, a thinned edge-detected image, TEi, will be produced.
- Then, the thinned edge-detected image, TEi, and the same threshold value, Tv, are used as input arguments for the Bugwalk process. After the completion of this process, the sketch, S, for that particular image is produced.

3. Edge Detection Phase

3.1 Filtering Process

The filtering processes that are applied onto the original image and edge-detected image consists of a brightening and sharpening process, an inversing process and a Gaussian process.

3.1.1 Brightening and Sharpening

The brightening process was applied to the image to enhance the brightness level of the original image. The sharpening process was used to make the original image sharper.

3.1.2 Inverse

Implementing edge detection techniques produced images with a black background and a white foreground which contains the image details. To display the edge-detected image better, an inverse process was applied, to inverse the colours of the image.

3.1.3 Gaussian

The Gaussian process is a technique used to smooth the image. Smoothing an image is a noise reduction technique [19]. The Gaussian used here is a two-dimensional

Gaussian, which was implemented as two orthogonal one-dimension Gaussian filters. The value of assigning $\sigma = 0.5$ was chosen after consideration of arguments of page 84 of [20] and [19].

3.2 Edge Detection Process

In the automated sketcher system, nine different edge detection techniques were used to extract the edge details from an image that will later be used to obtain the sketch. Each edge detection technique has its own unique way to extract these edge details, using predefined mask operators with the exception of Laplacian and LoG. These mask operators are used to extract the edge details in various orientations. For techniques where more than one mask operator was used, the outcome from the different masks is combined together, thus producing one edge-detected image at the end. For each technique, the masks are applied separately to the red, green and blue colour planes of the image before being combined and thresholded to form a monochrome edge-detected image. The formulae used are given in (1)–(9).

Equations (1)–(3) are used to gather the edge details for the horizontal orientation, while (4)–(6) are used to gather the edge details for the vertical orientation. The same mathematical operations were also applied when obtaining the edge details for the diagonal orientation. After all the values for each colour plane and each orientation have been calculated, the absolute values of each $newR$, $newG$, and $newB$ are combined. The result is then divided with the average of all the coefficient absolute values in the mask operators. This process is done to ensure that each method produces edge-detected images with similar contrast. Equations (7)–(9) are the mathematical operations that are used to produce the edge-detected images. $R(x+i, y+j)$, $G(x+i, y+j)$, $B(x+i, y+j)$ are operators used to extract the current colour intensity between the underlying image and the kernel overlapping it, in terms of each colour channel. $Ed(i, j)$, meanwhile, are the edge detection mask operator as illustrated in Table 1.

For Laplacian and LoG, the mask combines information about horizontal and vertical edges. Another difference between these and the other methods is that they are based on the second derivatives of the image $I(x, y)$ with respect to x and y. $I(x, y)$ are intensity values at x and y position of the image. x refers to the width, while y refers to the height of the image. The others rely only on first derivatives. Table 1 shows the mask operators that were used for each edge detection technique to obtain the edge-detected images. Each matrix represents the operator to measure the derivative in x, y and diagonal directions of the image.

$$R_h = \sum_{j=-1}^{1} \sum_{i=-1}^{1} R(x+i, y+j) * Ed_h(i, j) \quad (1)$$

$$G_h = \sum_{j=-1}^{1} \sum_{i=-1}^{1} G(x+i, y+j) * Ed_h(i, j) \quad (2)$$

$$B_h = \sum_{j=-1}^{1} \sum_{i=-1}^{1} B(x+i, y+j) * Ed_h(i, j) \quad (3)$$

$$R_v = \sum_{j=-1}^{1} \sum_{i=-1}^{1} R(x+i, y+j) * Ed_v(i, j) \quad (4)$$

$$G_v = \sum_{j=-1}^{1} \sum_{i=-1}^{1} G(x+i, y+j) * Ed_v(i, j) \quad (5)$$

$$B_v = \sum_{j=-1}^{1} \sum_{i=-1}^{1} B(x+i, y+j) * Ed_v(i, j) \quad (6)$$

$$newR = ([|R_h| + |R_v|]/(avg)) \quad (7)$$

$$newG = ([|G_h| + |G_v|]/(avg)) \quad (8)$$

$$newB = ([|B_h| + |B_v|]/(avg)) \quad (9)$$

Note: avg refers to the average for the coefficient values in the mask operator.

4. Image Segmentation Phase

4.1 Thinning Algorithm

Thinning is a very important technique extensively used in areas of pattern recognition, visual inspection and character recognition [24]. Furthermore, thinning is also used for fingerprint analysis [25], cited in [26], and in biomedical systems [27], cited in [26]. Thinning is a process that deletes dark points and transforms a pattern into a 'thin' line drawing known as a skeleton [28]. As a result, thinning is also essential in image processing and is used in the automated sketcher system to thin down the thick edges in the edge-detected image. The details of the thinning algorithm used can be found in [18]. The edge-detected image produced after the edge detection phase produces an image, which has an uneven mix of thin and thick lines and curves. As a result, the thinning algorithm was used to thin down all the thick lines and curves in the edge-detected image. However, the thinning algorithm will not thin down the already thin lines or curves in the image. The main idea is to make all the lines and curves in the edge-detected image to be one pixel thick. The thinned down edge-detected image produced is later used by the bugwalk operation.

4.2 Bugwalk

Bugwalk is a process used to perform edge tracing within the already thinned down edge-detected image. Edge tracing can be described as the process of following the edges and collecting the edge pixels into a list, page 2 of [29]. According to [30], edge tracing is one of the most fundamental subjects of image analysis, as it plays an important role in object recognition.

The concept of the bugwalk process can be found in page 323 of [16]. However, the original process implemented a three neighbour's concept, whereby, the program will only search for points in three directions, namely, east, southeast and south only. As a result, the original program

Table 1
Edge Detection Techniques Mask Operators

Edge Detection	Horizontal Angle (Ed_h)	Vertical Angle (Ed_v)	Diagonal Angle (Ed_d)
Frei and Chen	$\begin{bmatrix} -1 & -\sqrt{2} & -1 \\ 0 & 0 & 0 \\ 1 & \sqrt{2} & 1 \end{bmatrix}$ Page 87 of [21]	$\begin{bmatrix} -1 & 0 & 1 \\ -\sqrt{2} & 0 & \sqrt{2} \\ -1 & 0 & 1 \end{bmatrix}$ Page 87 of [21]	None
Kirsch	$\begin{bmatrix} 3 & 3 & 3 \\ 3 & 0 & 3 \\ -5 & -5 & -5 \end{bmatrix}$ [22]	$\begin{bmatrix} -5 & 3 & 3 \\ -5 & 0 & 3 \\ -5 & 3 & 3 \end{bmatrix}$ [22]	$\begin{bmatrix} 3 & 3 & 3 \\ -5 & 0 & 3 \\ -5 & -5 & 3 \end{bmatrix}$ [22]
Laplacian	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$ Page 129 of [17]		None
Laplacian of Gaussian (LoG)	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$, After additional Gaussian smoothing. Page 129 of [17]		None
Prewitt	$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$ Pages 578, 579 of [17]	$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$ Pages 578, 579 of [17]	$\begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{bmatrix}$ Pages 578, 579 of [17]
Roberts	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$ [23]	$\begin{bmatrix} 0 & 0 & 1 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$ [23]	None
Roberts Cross	$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$ Page 136 of [17]	$\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$ Page 136 of [17]	None
Robinson	$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & 1 \\ -1 & -1 & -1 \end{bmatrix}$ [22]	$\begin{bmatrix} -1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & 1 & 1 \end{bmatrix}$ [22]	$\begin{bmatrix} 1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & -1 & 1 \end{bmatrix}$ [22]
Sobel	$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$ Pages 578, 579 of [17]	$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$ Pages 578, 579 of [17]	$\begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix}$ Pages 578, 579 of [17]

(x, y)	$(x+1, y)$
$(x, y+1)$	$(x+1, y+1)$

Figure 3. Three neighbouring.

$(x-1,\ y-1)$	$(x, y-1)$	$(x+1,\ y-1)$
$(x-1, y)$	(x, y)	$(x+1, y)$
$(x-1,\ y+1)$	$(x, y+1)$	$(x+1,\ y+1)$

Figure 4. Eight neighbouring [30].

was modified to an eight neighbour's concept, which results in a more effective tracing of the edges. The neighbourhood concept is used to traverse between points within an image and perform the bugwalk process. The three neighbouring concept in Fig. 3 only considers the three nearest neighbours surrounding the x, y position of the image. The eight neighbouring concept in Fig. 4 meanwhile considers eight nearest neighbours surrounding the x, y position of the image.

5. Research Findings and Survey Result

After the completion of the automated sketcher system, the three hypotheses that were outlined earlier were tested. As a result, the following conclusion for each hypothesis was reached.

(H1) Can edge detection techniques be used to help transform an image into a sketch?

With the success of the automated sketcher system being developed, it has proven that edge detection techniques can be used to obtain a sketch from an image. The edge details of the image were traced by using each edge detection technique and the segmentation process were used to produce a sketch by using the edge details in the edge-detected image. Some examples are shown in Fig. 5. As a result, the first hypothesis has successfully been proven to be true.

(H2) Can the automated sketcher system be applied onto any type of image?

Different types of image were applied onto the automated sketcher system during the testing phase, as a way to justify this hypothesis. Regardless of the image type used, which could be JPEG (lossy), PNG (lossless), GIF (lossless), etc., the final outcome remained the same, as the sketch was successfully produced in the end. Furthermore, this also proves that the type of image encoding does not influence the sketch being produced. This shows that the automated sketcher system could be applied equally well to lossy and lossless images. Sample images in different type used to prove this hypothesis is shown in Fig. 6.

(H3) Which edge detection technique is best suited for transforming an image into sketch?

Early experiments did not provide significant evidence to conclude which edge detection technique is the best suited for transforming an image into a sketch. Therefore, a survey was conducted to gather statistical evidence from respondents that would help in answering this third and final question. The survey was conducted for about a month and 80 respondents took part in the survey. The respondents were asked to utilize the automated sketcher

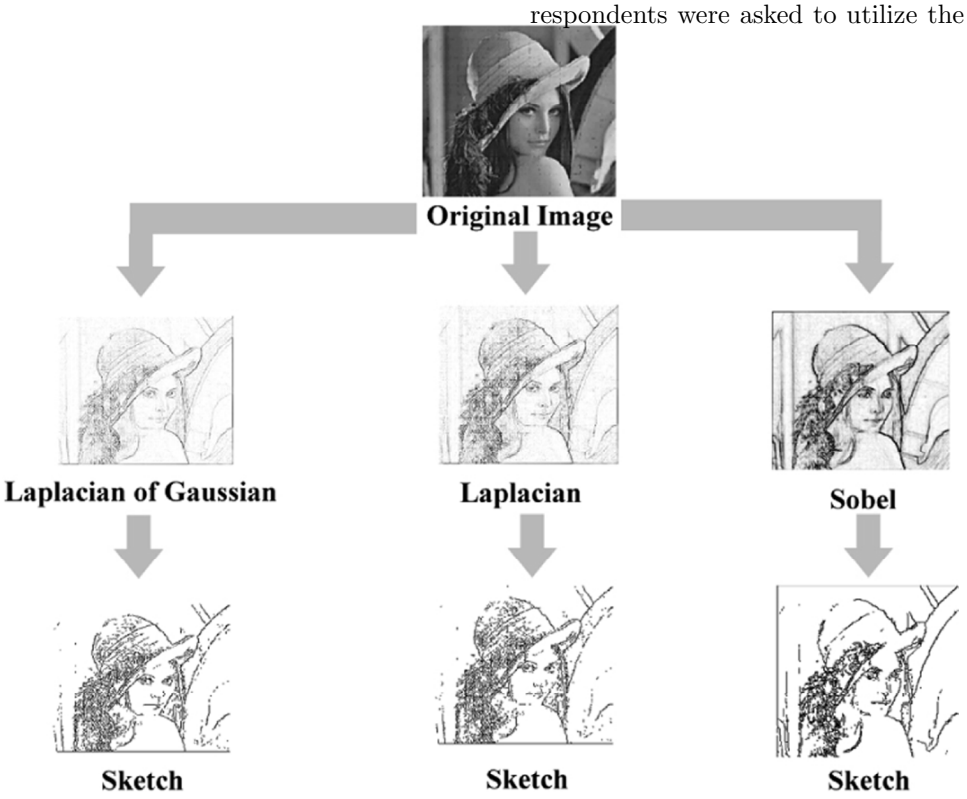

Figure 5. Examples of transforming an image into a sketch.

Figure 6. Sample images in different type used.

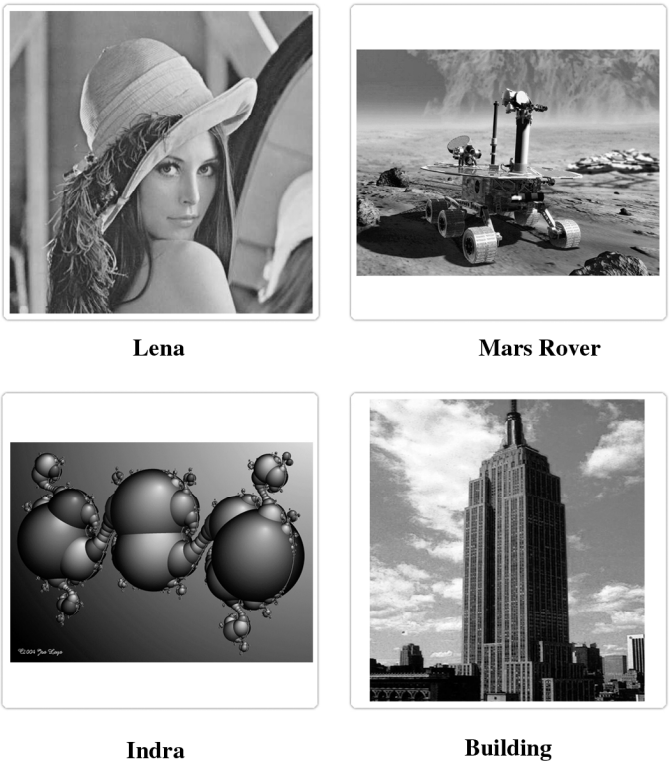

Figure 7. Test images used for the survey.

system and answer a questionnaire. The questionnaire was organized to examine five different objectives for each edge detection technique. The five objectives were:

Resemblance

- Used to determine whether or not the sketch had any resemblance to the original image.

General shape

- Used to determine whether or not the sketch had the general shape, if compared to the original image.

Preserves details

- Used to determine whether or not the sketch preserved the details, if compared to the original image.

Pleasing to the eye

- Used to determine whether or not the sketch produced was pleasing to the eye.

Threshold range

- Used to determine whether the default threshold value used was suitable. The defaults had been selected after a small pilot survey.

When the respondents utilized the automated sketcher system, they were asked to use one of the four different images that were selected as test data, as shown in Fig. 7, to answer the entire questionnaire. The nine edge detection techniques were applied one by one onto the image to obtain the respective edge-detected images. Next, the image segmentation techniques were applied onto the edge-detected images to obtain the sketches. Realistically, artists would normally sketch a scenario based solely on their perception of the scenario using only their eyes as a guide. With this in mind, no computation was included into the system to compute the threshold range. Computing the threshold range would make the system more automated instead of allowing the user using the software to decide the proper threshold. The proper threshold is measured when the sketch appears to the user to fulfil the criteria of resemblance, general shape, preservation of detail and pleasantness to the eye. Feedback obtained from the survey contributed to an appropriate threshold range for each edge detection technique as shown in Table 2.

Table 2
The Threshold Range for Each Edge Detection Techniques

Edge Detection Technique	Threshold Range	Default
Frei and Chen	502–654	610
Kirsch	464–553	561
Laplacian	573–665	630
Laplacian of Gaussian (LoG)	621–700	682
Prewitt	426–497	525
Roberts	485–565	588
Roberts Cross	539–603	633
Robinson	507–625	589
Sobel	400–541	511

The default threshold values that were used by the image segmentation phase varied between the nine different edge detection techniques. As the respondents utilized the system, they answered the questions about resemblance, shape, details and aesthetic value for each of the edge detection technique. Finally, after the respondents had tried all of the nine edge detection techniques, they were required to conclude which edge detection technique they felt was the best suited for transforming an image into a sketch.

From the 80 respondents who took part, 41% of them concluded that the LoG was the best-suited edge detection technique to transform an image into a sketch. The analysis charts produced from the survey for each objective, Figs 8 and 9, confirm the effectiveness of LoG as the best-suited edge detection technique for transforming an image into a sketch by obtaining significantly high scores as compared to other edge detection techniques. In Fig. 8, the x-axis represents the edge detection technique used, whereas the y-axis represents the mean values of a particular edge detection technique being chosen by the respondents. Figure 10 provides a pie chart that summarizes the respondent's choice of the best edge detection technique.

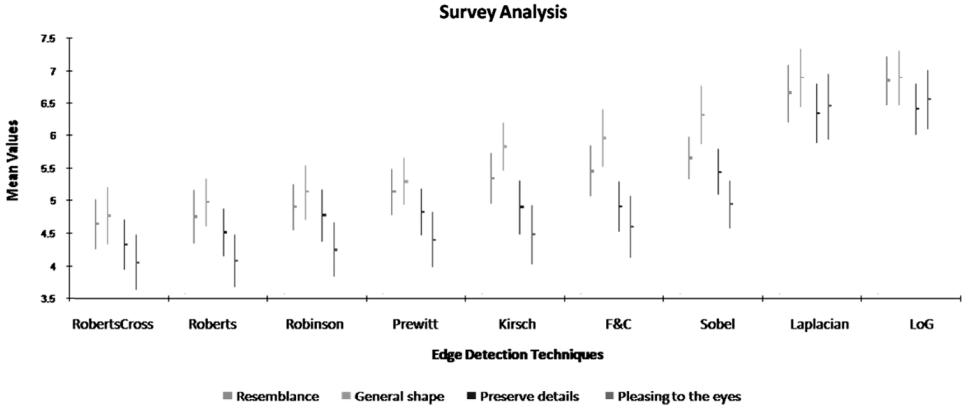

Figure 8. Analysis result for resemblance, general shape, preserve details, and pleasing to the eye.

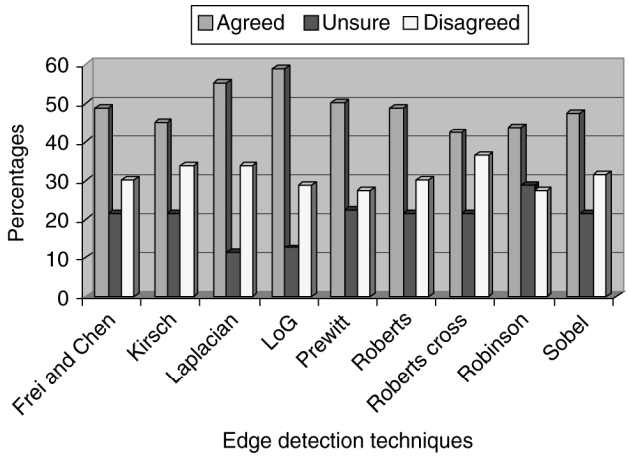

Figure 9. Analysis result for feedback on default threshold values.

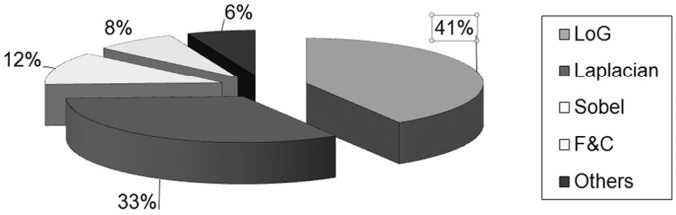

Figure 10. Best-suited edge detection technique.

6. Conclusion

Analysis results have proven that edge details extracted from an image using edge detection technique, with the help of image segmentation techniques, can be used to transform any type of image into a sketch successfully. Survey analysis have proven that edge detection techniques based on second derivatives such as Laplacian or LoG outperforms edge detection techniques based on first derivatives when rating the sketch as compared to the original image in terms of resemblance, general shape, details and pleasantness to the eye. In the future, it would be worthwhile to repeat this experiments using third or higher derivative techniques to see how well they perform.

References

[1] W. Konen, Comparing facial line drawings with gray-level images: a case study on PHATOMAS, *Proceedings of International Conference on Artificial Neutral Networks*, 1996, 727–734.

[2] H. Koshimizu, M. Tominaga, T. Fujiwara, & K. Murakami, On Kansei facial processing for computerized facial caricaturing system PICASSO, *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics*, 1999, 294–299.

[3] H. Chen, Z. Liu, C. Rose, C.Y. Xu, H.Y. Shum, & D. Salesin, Example-based composite sketching of human portraits, *Proceedings of the 3rd International Symposium on Non-Photorealistic Animation and Rendering*, 2004, 95–102.

[4] L. Liang, H. Chen, Y.Q. Xu, & H.Y. Shum, Example-based caricature generation with exaggeration, *Proceedings of the 10th Pacific Conference on Computer Graphics and Applications*, 2002, 386–393.

[5] P.Y. Chiang, W.H. Liao, & T.Y. Li, Automatic caricature generation by analyzing facial features, *Proceedings of the 2004 Asia Conference on Computer Vision (ACCV' 04)*, Korea, 2004.

[6] G.Z. Xu, M. Kaneko, & A. Kurematsu, Synthesis of facial caricature using eigenspaces, *Electronics and Communication in Japan, Part 3*, *87*(8), 2004, 43–54.

[7] B. Moghaddam, T. Jebara, & A. Pentland, Bayesian face recognition, *Pattern Recognition*, *33*, 2000, 1771–1782.

[8] M.A. Turk & A.P. Pentland, Face recognition using eigenfaces, *Proceedings of CVPR'91, IEEE Press*, 1991, 586–591.

[9] I.T. Jolliffe, *Principle component analysis* (New York: Springer-Verlag, 1986).

[10] L. Wiskott, J.M. Fellous, N. Kruger, & C. von der Malsburg, Face recognition by elastic bunch graph matching, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *19*(7), 1997, 775–779.

[11] R.O. Duda & P.E. Hart, Use of the Hough transformation to detect lines and curves in pictures, *Communication of the Association for Computing Machinery*, *15*(1), 1972, 11–15.

[12] H. Chen, Y.Q. Xu, H.Y. Shum, S.C. Zhu, & N.N. Zheng, Example-based facial sketch generation with non-parametric sampling, *International Conference on Computer Vision (ICCV'01)*, 2001, 433–438.

[13] X. Tang & X. Wang, Face sketch synthesis and recognition, *IEEE Transaction on Circuits & Systems for Video Technology*, *14*(1), 2004, 50–57.

[14] X. Tang & X. Wang, Face photo recognition using sketch, *Proceedings of ICIP*, *1*, 2002, I-257–I-260.

[15] S.H. Kwok & A.G. Constantinides, A fast recursive shortest spanning tree for image segmentation and edge detection, *IEEE Transactions on Image Processing*, *6*(2), 1997, 328–332.

[16] D.A. Lyon, *Image processing in java* (Upper Saddle River, NJ: Prentice-Hall Inc., 1999).

[17] R.C. Gonzalez & R.E. Woods, *Digital image processing,* Second Edition (Upper Saddle River, NJ: Prentice-Hall Inc., 2002).

[18] R. Wang, Thinning algorithm (Engineering Department, Harvey Mudd College, 2004), Available at: http://fourier.eng.hmc.edu/e161/lectures/morphology/node2.html (accessed Jan. 17, 2005).

[19] H. Chidiac & D. Ziou, Classification of image edges, *Vision Interface '99*, 1999, 17–24.

[20] M. Sonka, V. Hlavac, & R. Boyle, *Image processing, analysis, and machine vision*, second edition (Pacific Grove, CA: Brooks/Cole Publishing Company, A Division of International Thomson Publishing Inc., 1999).

[21] P.F. Whelan & D. Molly, *Machine vision algorithms in java: techniques and implementation* (London; New York: Springer, 2002).

[22] M. Sonka, Digital image processing, chapter 4, part III, image pre-processing: Local pre-processing (College of CSS Engineering, The University of Iowa, 2004), Available at: http://css.engineering.uiowa.edu/∼dip/LECTURE/PreProcessing3.html (accessed Dec. 3, 2004).

[23] H.E. Rhody, Edge detection by gradient calculation (RIT Center for Imaging Science, 2002), Available at: http://www.cis.rit.edu/people/faculty/rhody/EdgeDetection.htm (accessed Dec. 7, 2003).

[24] N.H. Han, C.W. La, & P.K. Rhee, An efficient fully parallel thinning algorithm, *In Proc. IEEE International Conference on Document Analysis and Recognition*, 1997, 137–141.

[25] B. Moayer & K.S. Fu, A tree system approach for fingerprint pattern recognition, *IEEE Transactions on Computing*, *25*(3), 1976, 262–275.

[26] P. Kwok, A thinning algorithm by contour generation, *Communications of ACM*, *31*(11), 1988, 1314–1324.

[27] A.R. Dill & M.D. Levine, Multiple resolution skeletons, *IEEE Transaction on Pattern Analysis and Machine Intelligence*, *9*(4), 1987, 495–504.

[28] P.S.P. Wang & Y.Y. Zhang, A fast and flexible thinning algorithm, *IEEE Transactions on Computing*, *38*, 1989, 741–745.

[29] J.R. Parker, *Algorithms for image processing and computer vision* (New York: John Wiley & Sons Inc., 1997).

[30] A. Arslan & I. Türkoglu, An edge tracing method developed for object recognition, *WSCG'99 – The 7th International Conference in Central Europe on Computer Graphics, Visualization & Interactive Digital Media '99, Vol. I–III*, Czech Republic, 1999.

Biographies

Tissa Chandesa completed his Diploma in Computer Studies from Informatics College, Malaysia and also obtained an International Diploma in Computer Studies, moderated by NCC Education in 2000. In 2002, he completed his Advanced Diploma in Computer Studies from Informatics College, Malaysia and also obtained an International Advanced Diploma in Computer Studies, moderated by NCC Education. In 2005, he obtained a B.Sc. (Hons) in Computer Science from the University of Nottingham, Malaysia Campus. Currently, he is pursuing a Ph.D. in Computer Science at the University of Nottingham, Malaysia Campus. His research interest centres on computer vision and image processing particularly visual tracking. During his high school years, he held a number of different posts, among them are President of the English Society, Secretary of the school prefect and facilitator of the Mathematics Clinic. He also represented the school for the English debate competition for the district level for two successive years.

Michael Hartley completed a B.Sc. and a Ph.D. from the University of Western Australia. He joined the private education industry in Malaysia in 1996, as a lecturer in Mathematics at Sepang Institute of Technology, later moving to the Information Technology department. In 2001, he became head of the school of information technology, and in 2002 Dean of the School of Multimedia and Engineering. He later spent a year as Director of Research at KDU College before joining the University of Nottingham Malaysia Campus as an associate professor in the School of Computer Science and Information Technology. He pursues research in Neural Networks, Genetic Algorithms, particularly applied to image processing. In addition, he has several journal publications in pure mathematics journals in an area of combinatorial geometry.