A System for Performance Evaluation of Arc Segmentation Algorithms

[1]Liu Wenyin [2]Jian Zhai [3]Dov Dori [2]Tang Long
[1]*Microsoft Research China, 49 Zhichun Road, Beijing 100080, PR China*
wyliu@microsoft.com
[2]*Department of Computer Science & Technology, Tsinghua University, Beijing 100084, PR China*
zhai98@mails.tsinghua.edu.cn
[3]*Faculty of Industrial Engineering and Management, Technion, Haifa 32000, Israel*
and Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, MA 02139, USA
dori@ie.technion.ac.il; dori@mit.edu

Abstract

Accurate segmentation of circular arcs from line drawings is essential for higher level processing in document analysis and recognition systems. In spite of the prevalence of arc segmentation methods, robust algorithms that perform well in complex graphic environments are scarce, and methods to evaluate such algorithms are even more rare. Extending our previous work, we propose a comprehensive system for evaluating arc segmentation. The system models three types of noise that can be found in real life drawing images: pixel, vector and context. To test the system, we applied it to evaluate an arc segmentation algorithm and obtained accurate numeric values for the various performance metrics of the algorithm. It has also been effectively used as the evaluation system in the IAPR Arc Segmentation Contest.

Keywords: *Performance Evaluation, Stress Testing, Arc Segmentation, Arc Segmentation Contest, Graphics Recognition, Engineering Drawings, Line Drawings Recognition*

1. Introduction

The problem of recognizing circular arcs from scanned line drawing, known as arc segmentation, is an open problem in the area of line drawings interpretation. The difficulty of arc segmentation stems from a number of sources. One difficulty is due to the relative complexity of the circle equation. Intersecting and tangent bars and arcs often smear the arc's image, creating larger black blobs that are hard to segment correctly. The raster image of an arc may be distorted disproportionately in different directions. Circular arcs are very frequently not full circles. Partial arcs, especially those with small angles, are more difficult to segment than full circles. Finally, various types of noise introduced by drafting, scanning, and binarization processes further increase the difficulty of arc segmentation.

In spite of the prevalence of arc segmentation methods, comprehensive approaches to the evaluation of arc segmentation algorithms and their robustness with respect to the problems reported above are scarce. Existing evaluation reports (e.g., Liu and Dori 1997, 1998, Chhabra and Phillips 1998, Phillips and Chhabra 1999) have focused on definition, selection and calculation of performance metrics of general line detection algorithms. Pixel type noise generation models include the document image degradation model proposed by Baird (1990, 1992, 1993) and Kanungo et al. (1993, 1994, 1995). They have been used by Haralick (1992), Kong et al. (1996), and Hori and Doermann (1996) in their protocols. Vector type noise has been studied to a much lesser extent. Madej and Sokolowski (1993) proposed a vector-based statistical model of variations of parcel (land registry) parameters, including length, angle, etc., in ground truthing cadastral maps for performance evaluation of map recognition and understanding systems.

These are necessary, but not sufficient for testing the performance of arc segmentation algorithms in complex graphic environments in general and in extreme situations in particular. Liu and Dori (1998), for example, tested their incremental arc segmentation (IAS) algorithm's performance only on arcs with varying thickness, open angles, and radii. While these are important basic parameters to be tested, they do not fully address many difficulties and obstacles in the way of arc segmentation in real life drawings. Current performance evaluation protocols are not designed to test the ability of arc segmentation algorithms to overcome difficulties of the types discussed above.

In this paper, we extend our previous work (Liu and Dori 1997, 1998) and develop a system that builds on a more comprehensive protocol for evaluating arc

segmentation algorithms. Our ground truth generating process models various noise types that cause arc segmentation difficulties. These are categorized into three types: pixel, vector, and context. We model several sub-types of pixel noise that may appear in real life drawings and add them to our synthetic ground truth drawing images. While modeling the vector type noises, we vary each parameter in the circle geometry equation to a tolerable extent. To account for context, we vary the extent and angles of intersecting straight lines and arcs with the arc being segmented. By calculating the arc segmentation performance on images that were generated using these interference models, we establish dependable metrics for the evaluation of the capabilities and the robustness of the tested arc segmentation algorithm.

The rest of the paper is organized as follows. In Section 2 we briefly present our previous performance evaluation protocol. In Section 3, we present the models of the three types of noise sources used in the ground truth generation process of our arc segmentation performance evaluation protocol. In Section 4, we apply the evaluation system on the IAS algorithm (Liu and Dori 1998) using generated images at different difficulty levels for each of the three noise types. In Section 5, we briefly present the IAPR Arc Segmentation Contest and how the system was used in contest. We conclude in Section 6 with remarks on the proposed system.

2. The Performance Evaluation Protocol

We define the performance of a graphics recognition algorithm as a set of metrics of interest on the output data that the algorithm produces with respect to the expected, ground truth data. Usually, the metrics are expressed in terms related to the difference between the expected output and the actual output of the algorithm. The metrics should be represented by quantitative indices based on accepted definitions to avoid subjectivity. Moreover, the entire performance of an algorithm should be reflected by a comprehensive metric. Appropriate analysis of relevant metrics should help compare, select, improve, and even design new algorithms to be applied in new systems targeted at specific applications.

Liu and Dori (1999) proposed that a performance evaluation protocol for a graphics recognition system consists of these three essential elements: (1) ground truthing (ground truth acquisition or generation); (2) matching; and (3) quantitative metrics. First, the expected output—the ground truth—must be known with certainty, such that it can be compared with the actual output—the recognition results. Therefore, a sound methodology of acquiring the appropriate ground truth data is required. Secondly, since both the ground truth data and the recognition results consist of many graphic objects, individual comparison of each ground truth object to its matching recognized object needs to be carried out. To do this, each ground truth graphic object must first be matched with one or more objects from the recognized objects set. Hence, a sound matching method is needed. Finally, representative metrics of interest should be selected, and quantitative indices that measure these metrics should be defined uniformly. The three elements are presented in detail in the following three sub-sections.

2.1 Ground Truthing

Ground truthing is the process that generates both the actual input image for the evaluated algorithm and the expected output (ground truth) for comparison with the actual output vector. Following the principles proposed by Liu and Dori (1999), we design the ground truthing process for arc segmentation algorithms as follows.

First, we generate in vector form ground truth arcs, as well as optional intersecting lines as sources of interfering noise. The generated vectors are saved into the ground truth vector file. For the same ground truth vector drawing, we then introduce various levels of noise for the various noise types while the vector format is converted into pixel format images. These synthetic images are used as input to the arc segmentation algorithm being evaluated, while the expected output is the same ground truth vector file. By modeling the various noise types and levels, the evaluated arc algorithm can be tested with the desired types and levels of noise that simulate disturbances in real life drawings. By calculating the arc segmentation performance on these images, we can evaluate the arc segmentation algorithm's robustness.

As noted, our system handles three types of noise. The first type is pixel noise that can be directly added onto the image generated from the ground truth vector drawing. The pixels are modified after the pixel image is generated. The second type is vector noise, which is generated by perturbations of the parameters in the circle equation while the drawing image is generated. The third type is context noise. Context noise concerns intersection and interference of arcs with other geometric vector lines and arcs that are added as part of the vector file before the images are generated. Modeling these three noise types is elaborated in Section 3.

2.2 Matching

To measure the difference between ground truth arcs and recognized arcs, ground truth arcs should be matched with the recognized ones. Problems associated with this procedure relate to the matching algorithm and what can be considered as a match. Following the principles proposed by Liu and Dori (1999), the matching degree of

a pair of arcs is based on the overlap area and the extent to which the endpoints match. Let $c = k \cap g$ denote the arc segment of the recognized arc k that overlaps the ground truth arc g, i.e., c is the intersection of k and g. The method of calculating c is presented and illustrated in Figure 1. A ground truth arc and a recognized arc are defined to be overlapping if the overlapping distance ($d_{overlap}$) is smaller than half the ground truth line width. As Figure 1 shows, the overlap distance, is defined as the Hausdorf distance—the maximum of all minimum distances between the points on k and g in the overlap area. The overlapping distance and the distances between the two arcs at the overlapping segment's two endpoints (denoted by d_1 and d_2) are used to calculate the performance metric, which is defined next.

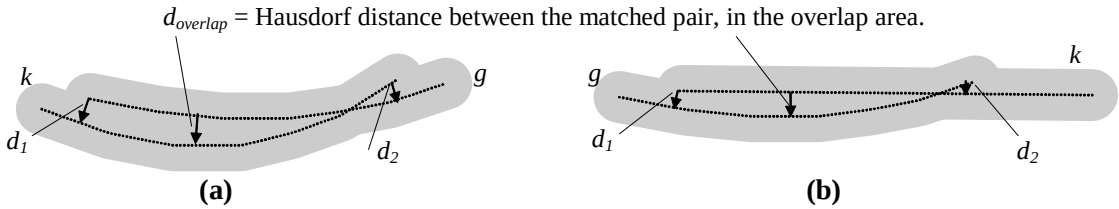


Figure 1. Illustration of the matching of arcs. (a) two arcs. (b) an arc and a line.

2.3 Performance Metrics

We use the set of performance metrics (Liu and Dori 1997) based on the vector detection quality of the overlapping segments of every recognized arc and the ground truth arc. In a nutshell, the vector detection quality is measured by the detection accuracy using a number of criteria, including the endpoints, the location offset, the line width, the geometry form, and the line style. The vector detection quality of these overlapping segments as well as their lengths are accumulated for both the ground truth arcs and the corresponding recognized arcs. The Basic Quality, (Q_b)is the length weighted sum of the vector detection qualities of overlapping segments, the Fragmentation Quality, (Q_{fr}), which is the measure of the detection fragmentation and/or consolidation, and the Total Quality, (Q_v), which is the product of Q_b and Q_{fr}, for both the ground truth arcs and their corresponding recognized arcs. The total Vector Detection Rate, D_v is length weighted sum of Q_v of all ground truth arcs, the Vector False Alarm Rate, F_v, which is the length weighted sum of $1–Q_v$ of all recognized arcs, and the resulting Vector Recovery Index, $VRI=(D_v+1–F_v)/2$.

3. Modeling Drawings Noise (Complexity)

Real engineering drawings frequently do not follow strictly the ideal geometrical shapes. Various types of noise at various levels are added to the drawings while they are drawn, stored and digitalized. The test drawing images are therefore similar to Non-Photorealistic Rendering (NPR) pictures. We generate the input images from the standard ground truth vector drawings. Vector and context noise types are introduced while generating the vector file while pixel noise is added later on.

For each noise type we generate a series of images with varying noise levels. Some of the noise types are further classified into noise subtypes. The evaluated arc segmentation algorithm is applied on each image and the arc recognition results are compared with the synthetic ground truth file. The algorithm is evaluated separately for each noise type to test its robustness with respect to this type of noise. Below we provide a model for each noise type.

3.1 Pixel Type Noise

The pixel type noise is classified into three subtypes: Gaussian noise, high frequency noise, and hard pencil simulation.

3.1.1 Gaussian noise

Gaussian noise is common in real life drawings. It is introduced into the digital image of the drawing by such factors as degrading paper quality and scanning effects. The Gaussian noise generation is accomplished as follows. We first generate a Gaussian random variable and map it onto the range [0, 255]. Gaussian distribution is generated using uniform distribution by assuming an n-dimensional random variable $\{Y_k, 1 \le k \le n\}$ of independently identical distribution (i.i.d.), and Y_k is a uniform random variable $U[0,1]$. We set $X_n = \sum_{k=1}^{n} Y_k$ and denote $f_n(x)$ as the probability density function (p.d.f.) of X_n.

The probability distribution function of a Gaussian random variable has the form

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{(x-\mu)^2}{2\sigma^2})$$
$$-\infty < x < +\infty$$

By standardizing Z_n, we get the form

$$Z_n = \frac{X_n - EX_x}{\sqrt{DX_n}}$$

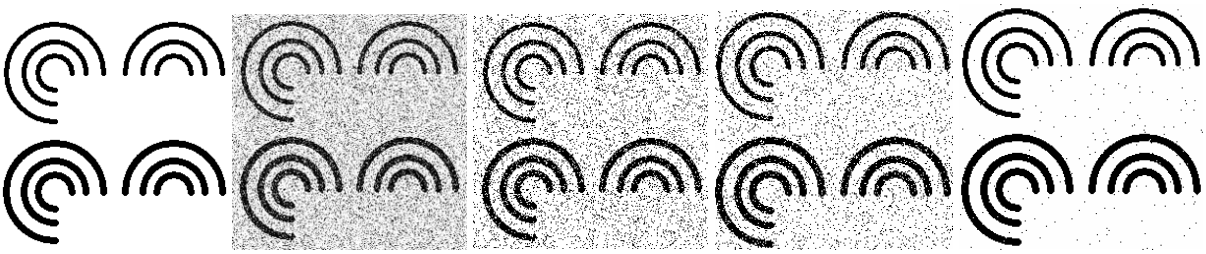
Using the Central Limit (Lindeberg-Levy) theorem (Heaton 1993) below, when $n \to \infty$, Z_n approaches the Gaussian distribution

$$P(Z_n \le x) \xrightarrow{w} \Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{u^2}{2}} du \quad ,$$
$$\forall x \in R, (n \to \infty),$$

When we set $n = 12$, the result of Z_n is a close approximation of the Gaussian distribution. Z_n is generated with value between 0 and 1. It is then multiplied by 255 and rounded. To get a grayscale image, we add this number to the original binary pixel value or subtract it as follows. If the original value is 0, we simply use the sum as the new pixel value. Otherwise, we use $255 - Z_n$ as the new value, because in black area, white pixel is noise, and the brighter the white pixel appears, the stronger the noise is.

Note that in the first step Y_k is a random variable between 0 and 1. If we change its range to [0, *L*], with *L* being a non-negative integer, then the distribution of Z_n changes accordingly. Since μ and σ describe the position and shape of a Gaussian random variable, σ changes along with *L*, so increasing *L* increases the Gaussian noise in the image. We use *L* to control the noise level at the Gaussian subtype of the pixel type noise. *L*=0 means that there is no noise. Finally, since arc recognition algorithms are usually designed to work with binary images, we apply a 50% threshold to convert the picture into a binary TIFF file.

Figure 2 illustrates the effect of Gaussian noise. Figure 2(a) shows an original drawing (*L*=0), to which Gaussian noise with *L*=8 is added and its gray scale image is shown in Figure 2(b). Figure 2(c) shows the noisy image (*L*=8) after Figure 2(b) has been binarized with a 0.5 threshold. Figure 2(d) shows the binary noisy image with *L*=5 and Figure 2(e), with *L*=2.



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

Figure 2. Illustration of Gaussian noise effect. (a) Original drawing (*L*=0), (b) Gray scale image with Gauss noise (*L* =8), (c) Binary noisy image (*L*=8), (d) Binary noisy image (*L*=5) , (c) Binary noisy image (*L*=2).

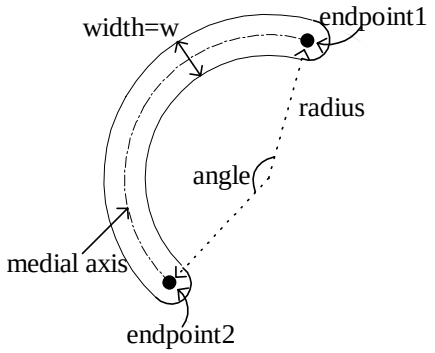


Figure 3. Illustration of arc-related geometric terms.

3.1.2 High frequency noise

In most engineering drawings, the boundary of the lines is often jagged because the paper is not quite flat, the ink is not uniform and the scanning adds more noise. The boundary where white and black areas meet is a high frequency area. We model this type of noise as diffusion of pixels through the boundary of the line using high frequency noise. Figure 3 illustrates arc-related terms used to explain our high frequency noise model.

Each black pixel in the arc is at a certain distance from the medial axis. The greater the distance, the more likely the pixel is to switch its color from black to white, such that near the boundary many pixels become white, making the edge rough. High frequency noise is simulated by a 2D-DFT operation in the frequency domain, which is then converted back the image pixel domain. As known

from convolution theory, $F[f_1(t) * f_2(t)] = F[f_1(t)] \bullet F[f_2(t)]$. Using this equation, instead of multiplying high frequency noise in the frequency domain, we use convolution to convert an image in the spatial domain. First, we create a randomly generated $L \times L$ bitmap file to be used as the convolution kernel. Since the size of this image affects the granularity of the processed picture, it is used to control the noise level of the high frequency subtype of noise. We then convolute this kernel bitmap with the original bitmap file of the synthetic drawing and get the high frequency test file with different noise levels, which are determined by *L*. After the convolution, each pixel value is between 0 and $L^2 * 256^2$. Since the recognition algorithms are designed to operate on binary files, we apply a 50% threshold to convert the convolution result into a binary TIFF file. Figure 4 presents the original image and the results of introducing high frequency noise with *L*= 1, 3, and 5.

(a) Original image (*L*=0) (b) *L* =1 (c) *L* =3 (d) *L* =5

Figure 4. Effects of high frequency noise

(a) Original Image (*T*=0) (b) *T* =1 (c) *T* =2 (d) *T* =4

Figure 5. Effects of hard pencil noise

3.1.3 Hard pencil simulation

Many hand drawings use hard pencil. Due to the paper's texture, when the pencil leaves its mark on the paper's surface, some places remain untouched, leaving tiny white lines across the black lines. We stimulate this phenomenon by the scan line method. We scan the picture using a line, whose expression is given by $x + y = k$, $k = 0,1,..., w + h$, where *w* is the width of picture and *h* is the height. Each scan line crosses white areas and black areas. When it crosses a white area, nothing is done, but when it reaches a black area, the hard pencil noise may be added to it. At each pixel in the original image, we generate a random number between 0 and 100 and compare it to a fixed threshold *T*, which is used to control the noise level. If it is less than *T*, a three-pixel-long white line is drawn at 45^0 on top of the original black area. If this white line extends out of the black area, its tail is cut off, effectively ignored. Hence, the threshold *T* is effective only if it is less than about 33. Longer simulated pen strokes decrease this number. For example, for a three-pixel-long white line, this number is 20. By properly choosing the threshold values *T*, different levels (*L*=*T*) of hard pencil noise can be generated. The larger the threshold value, the more white lines are possibly drawn on the black area, and thus the stronger the noise is. The effects of hard pencil noise are shown in Figure 5.

3.2 Vector Type Noise

For the pixel noise type, all the noise subtypes are added pixel by pixel, independently of the geometry of the shape. The vector noise takes into account the parameters of the shape's geometry. An arc is determined only when its center point, radius and width are fixed. In hand made drawings, the radius may vary for a variety of reasons, such as the use and quality of the compass and paper, as well as the paper's non-uniform stretching or shrinking due to dampness. The arc's radius, center and width may therefore slightly fluctuate around their original values. To account for these variations, we need to design the arcs with a non-zero variance for each of these three parameters. A simple method to add vector type noise is to multiply the central point, radius and line width by a random number around 1 (one) while using DDA or

Bresenham algorithms (Foley et al. 1995) to draw an arc. However, as Figure 6 shows, this method is not useful because independent random numbers at adjacent points along the arc may result in abrupt changes along the arc, such that the arc is broken into several arcs, each having its own center point, radius and line width.

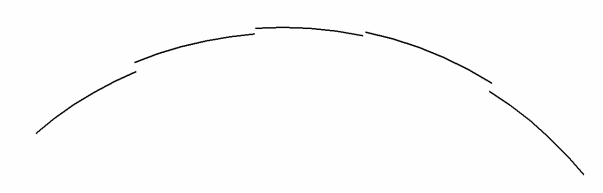


Figure 6. Broken arc due to independent random numbers used in radius variation

To solve this problem or modeling radius variation, we apply random walk using the Brown Movement stochastic process. We denote the arc radius at a point on the arc using Brown Movement $\{X(t), t \geq 0\}$, with t representing the central angle θ of the point to the X-axis. We divide the entire arc into several pieces with equal central angle $\Delta\theta$. The radius within each piece is fixed. Since in a digital image, the pixel is discrete, "continuous" means that there are no white pixels separating the continuity of the arc. We therefore restrict the radius variation between two adjacent arc pieces to one pixel. Brown Movement meets this condition. The noise level L is determined by $\Delta\theta$. The smaller $\Delta\theta$, the more pieces of arcs comprise the arc and thus the stronger the noise. We define $L = \ln(2\pi / \Delta\theta)$ to represent the vector noise level. Figure 7 shows a clean drawing (a) and distorted arcs at L=8 (b) and L=11 (c).

(a) Ground truth (b) L=8 (c) L=11

Figure 7. Effects of vector type noise

3.3 Context Type Noise

Arc segmentation is affected by the context in which the arc is situated in the drawing, i.e., other graphic objects in its neighborhood. For example, when a line and an arc are tangent, some algorithms mistake them for crossing lines or are otherwise fooled. To simulate basic context noise, we consider the interaction between an arc and a line segment and between an arc and another arc. As Figure 8 show, since both the line and the arc have non-zero widths, the tangent position is not uniquely defined. We define the tangency noise level L as the distance between the two lines' medial axes at the tangent point. L is then normalized to 0~10 by computing L=10 $|d_1 - d_2| / (W_2/2)$.

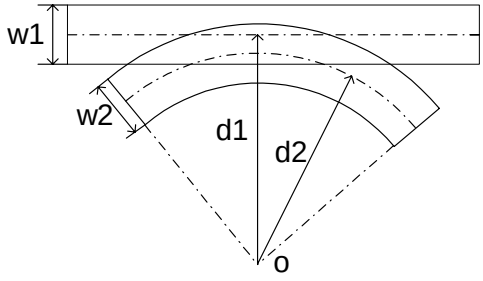


Figure 8. The tangent position of an arc and a line with non-zero line widths

4. Performance Evaluation of The IAS Algorithm

In this section, we test the IAS algorithm's performance using the metrics defined in Section 2.3 over the images generated in Section 3. We gradually increase the noise level of each noise type and examine the corresponding performance change of the algorithm. This way we can test how robust the algorithm is with respect to each type of noise.

4.1 Impact of Pixel Type Noise

The ground truth vector drawing we used in this test is shown in Figure 9 (Liu and Dori 2001).

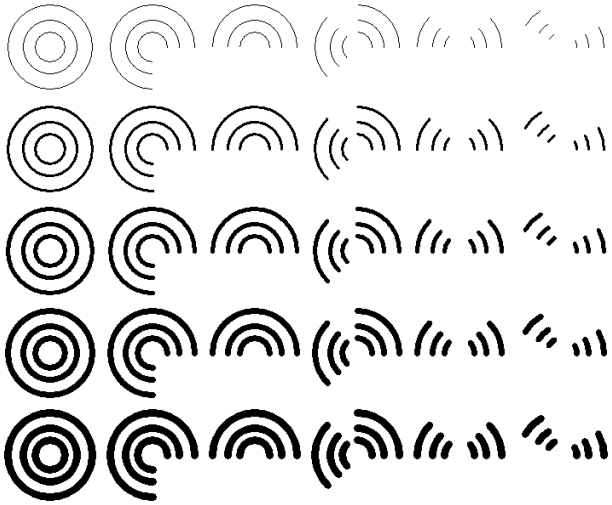


Figure 9. Illustration of The ground truth vector drawing used for testing performance impact of pixel type noise and vector type noise.

Table 1. IAS's performance metrics over Gaussian noisy images of levels from 0 to 8.

Level (*L*)	0	1	2	3	4	5	6	7	8
Dv	0.740	0.740	0.772	0.734	0.649	0.462	0.374	0.271	0.128
Fv	0.229	0.229	0.197	0.169	0.205	0.265	0.280	0.421	0.651
VRI	0.755	0.755	0.787	0.782	0.722	0.598	0.547	0.425	0.239

4.1.1 IAS' Performance with respect to Gaussian Noise

In this test, we obtain the IAS' performance metrics with respect to Gaussian noisy images with levels varying between 0 and 8. The results are presented in **Table 1** and the performance curve is shown in Figure 10.

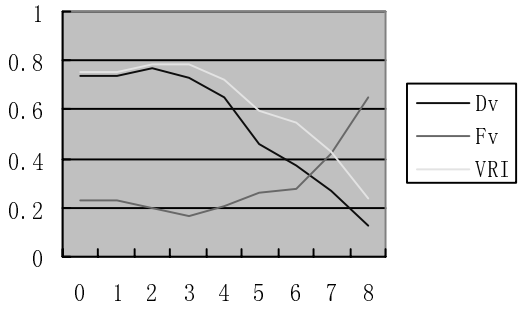


Figure 10. IAS' performance curves over Gaussian noisy images of levels from 0 to 8.

From Figure 10 we can see that, as expected, as the Gaussian noise level increases, the performance decreases. When *L* reaches 6, the performance decreases dramatically. Hence, we can say that the IAS algorithm performs well when the Gaussian noise level is less than 6 and is not suitable for the situation when *L* is larger than 6.

4.1.2 IAS' Performance with respect to High Frequency Noise

In this test, we obtain the IAS' performance metrics with respect to high frequency noisy images with levels between 0 and 22. The results are presented in Table 2 and the performance curve is shown in Figure 11.

From Figure 11 we can see that as the high frequency noise level increases, the performance decreases only slightly when the noise level is not very high. When *L* reaches 20, the performance decreases dramatically. Hence, we can say that IAS works fine when the high frequency noise level is less than 20 and is not suitable for the situation when *L* is larger than 20.

4.1.3 IAS' performance with respect to Hard Pencil Noise

In this test, we obtain the IAS' performance metrics over hard pencil noisy images of levels from 0 to 7. The results are presented in Table 3 and the performance curve is shown in Figure 12.

From Figure 12 we can see that as the hard pencil noise level increases, the performance decreases. When *L* reaches 3, the performance decreases dramatically. Hence, we can say that IAS works fine when the hard pencil noise level is less than 3 and is not suitable for the situation when *L* is larger than 3.

4.2 Impact of Vector Type Noise

In this section, we test the impact of vector type noise on the performance of the IAS algorithm. Using Figure 9 as the ground truth vector drawing image for this test, we vary the vector noise level from 0 to 11 and obtain corresponding performance metrics shown in Table 4. The performance curves are shown in **Figure 13**.

From **Figure 13** we can see that as the vector noise level increases, the performance decreases slowly and gracefully without any sudden breakdown point. Hence, we can say that IAS works fine when the vector noise is within reasonable level.

4.3 Impact of Context Type Noise

In this section, we test the impact of different levels of context type noise on the performance of the IAS algorithm. We use the arc and the interfering line shown in Figure 14 (Liu and Dori 2001) as the ground truth vector drawing for this test. By slightly moving the line away from the arc's medial axis, we vary the context noise level from 0 to 7 such that the line is kept "tangent" to the arc. The performance metrics are shown in Table 5 and the performance curves are shown in Figure 15. As the context noise level increases, the performance varies only

slightly at the high performance level. Hence, we can say that IAS works fine with tangency noise.

Table 2. IAS's performance metrics over high frequency noisy images of levels from 0 to 22.

L	0	2	4	6	8	10	12	14	16	18	20	22
Dv	0.740	0.742	0.740	0.709	0.726	0.711	0.691	0.664	0.642	0.624	0.583	0.387
Fv	0.229	0.157	0.126	0.148	0.130	0.110	0.121	0.129	0.134	0.151	0.163	0.275
VRI	0.755	0.793	0.807	0.781	0.798	0.801	0.767	0.749	0.744	0.737	0.710	0.556

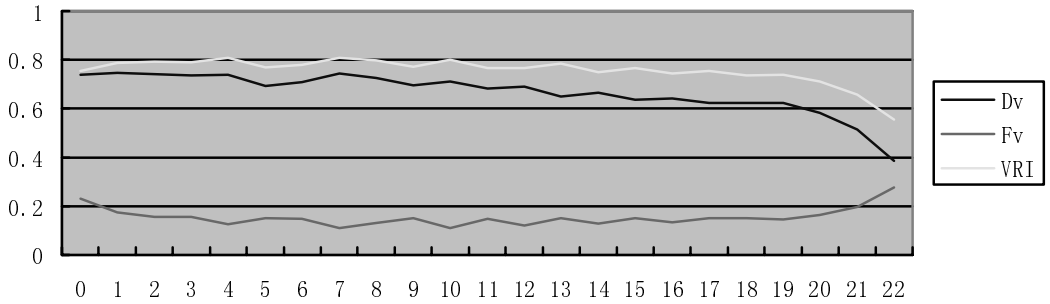


Figure 11. IAS' performances curve over high frequency noisy images of levels from 0 to 22.

Table 3. IAS' performance metrics over hard pencil noisy images of levels from 0 to 7.

L	0	1	2	3	4	5	6	7
Dv	0.740	0.616	0.603	0.289	0.160	0.151	0.112	0.105
Fv	0.229	0.177	0.205	0.373	0.586	0.611	0.711	0.728
VRI	0.755	0.719	0.699	0.458	0.287	0.270	0.200	0.188

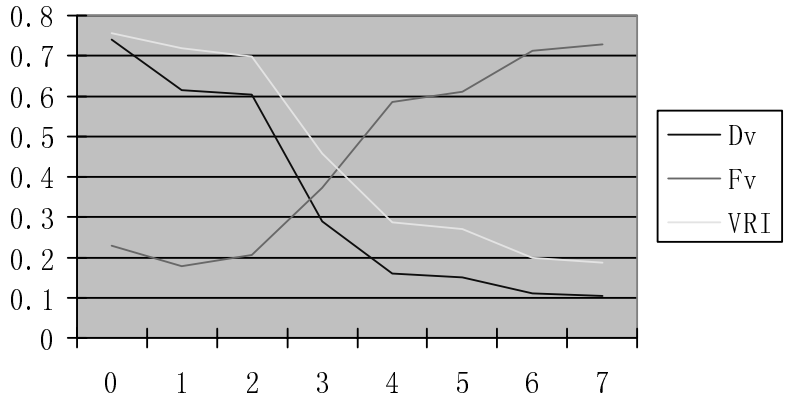


Figure 12. IAS' performance curves over hard pencil noisy images of levels from 0 to 7.

Table 4. IAS' performance metrics over vector noisy images of levels from 0 to 11.

L	0	1	2	3	4	5	6	7	8	9	10	11
Dv	0.740	0.744	0.751	0.682	0.727	0.701	0.690	0.665	0.645	0.620	0.575	0.600
Fv	0.229	0.224	0.203	0.258	0.230	0.238	0.245	0.244	0.291	0.305	0.355	0.308
VRI	0.755	0.760	0.774	0.712	0.749	0.732	0.723	0.711	0.677	0.658	0.610	0.646

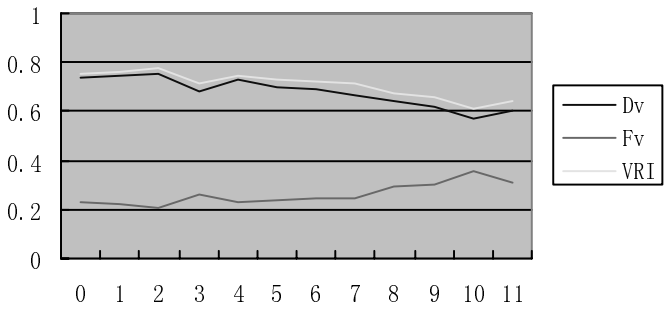


Figure 13. IAS' performance curves over vector noisy images of levels from 0 to 11.

Table 5. IAS' performance metrics over context noisy images of levels from 0 to 7.

L	0	1	2	3	4	5	6	7
Dv	0.863	0.940	0.902	0.748	0.927	0.888	0.811	0.778
Fv	0.153	0.073	0.113	0.084	0.086	0.125	0.086	0.109
VRI	0.855	0.933	0.894	0.832	0.920	0.882	0.863	0.834

Figure 14. Ground truth drawing for testing impact of context level noise

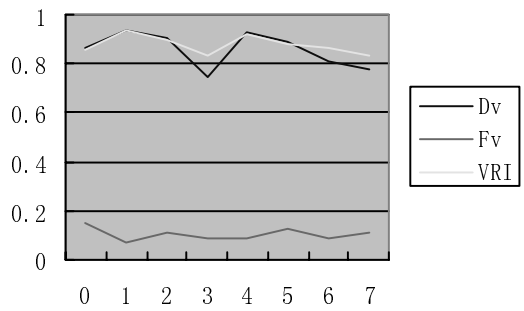


Figure 15. IAS' performance curves over context noisy images of levels from 0 to 7.

5. Application in the Arc Segmentation Contest

We have organized the Arc Segmentation Contest (Liu and Dori 2001) during the 4th IAPR Workshop on Graphics Recognition in Kingston, Canada, September 7-8, 2001, as the fourth contest in the series. More information about the contest and the series can be found at the contest website (http://research.microsoft.com/users/wyliu/arcsegcontest.htm). The proposed system has been effectively used as the evaluation system of the contest. Next, we briefly present the contest rules, test images, and the contest result.

The contest was for solid arc segmentation only. That is, we measured solid arcs only. The detection rate and false alarm rates were calculated according to the solid arcs in the ground truth files and the solid arcs recognized by the participant systems. They might yield other types of graphics than solid arcs, which they think may help improve the performance of solid arc segmentation. However, since other detection of dashed arcs, dashed

lines, text or symbols might produce false alarms, which are actually solid arcs, we did not encourage them to do so.

In total, we had used seven test images in the on-site contest, four of them are synthesized images and three are real scanned images. We had also provided many pre-contest images, which are only synthesized images with noises of various types and levels we defined in this paper. All are in binary TIFF format. All of them and their ground truth files are available now at the contest website. The participant systems are required to run as a black box with fixed parameters/configurations without any other human intervention except for specifying the input/output filenames. The overall average of the Vector Recovery Indices (defined in Section 2.3) of the seven images was used as the unique measure to judge which system is the best.

The four synthesized images that were finally used in the on-site contest were: one with Gaussian noise at Level 5, one with High-Frequency noise at Level 3, one with Hard-Pencil noise at Level 3, and one with Vector-Type (Geometry) noise at Level 3. The ground truth vector drawing for all of the four images is the same and similar to the one shown in Figure 9.

The ground truths for the three real scanned drawings were manually measured by us. One of them is shown in Figure 16. The other two are the same one as shown in Figure 17 but were scanned with different resolutions.

Initially, several groups had shown interests in participating in the contest. Finally, only two systems could finally participate: one from Dave Elliman (2001) and one from Xavier Hilaire (2002), and Elliman's system with a score of (0.681) won the First Place and Hilaire's with a score of (0.630) won the Honorable Mention Prize.

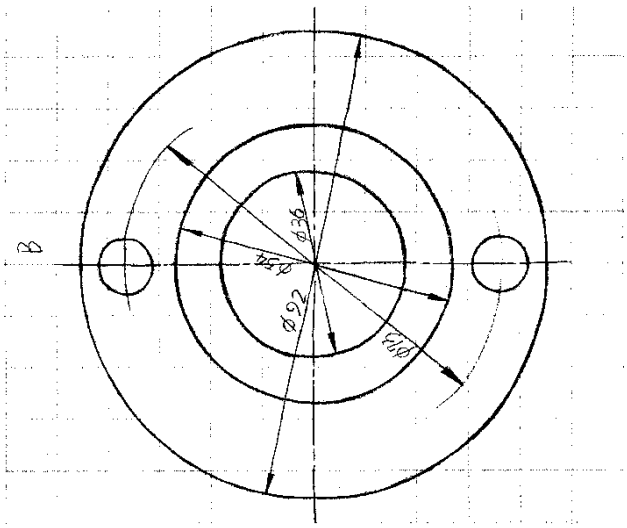


Figure 16. One real scanned images used in the contest.

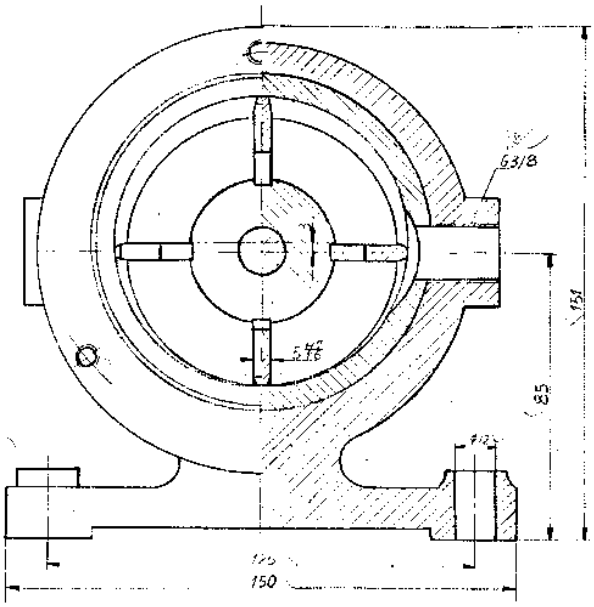


Figure 17. Another real scanned images used in the contest.

6. Concluding Remarks

An elaborate performance evaluation system that uses a multi-attribute protocol for arc segmentation algorithm has been presented. The protocol accounts for three types of noise: pixel, vector and context. Each noise type has several subtypes, and each subtype has a number of control variables. This rich testing environment provides for a sophisticated, accurate and flexible way of assessing various aspects of arc recognition quality. We have implemented the protocol and applied it to carry out a comprehensive evaluation of the Incremental Arc Segmentation (IAS) algorithm (Liu and Dori 1999). Experiments of the system have successfully proved that that the IAS algorithm is robust to various noise types at reasonable levels. The system has also effectively been used as the evaluation system in the Arc Segmentation Contest (Liu and Dori 2001) held during the 4th IAPR Workshop on Graphics Recognition, in Kingston, Canada, September 7-8, 2001.

Acknowledgement

The authors thank Professor Arnold Smeulders for his suggestion on stress testing.

References

[1] Baird HS (1990) Document image defect models. In: *Proc. of IAPR Workshop on Syntactic and Structural Pattern Recognition*, Murray Hill, NJ, pp. 38-46.

[2] Baird HS (1992) Document image defect models. In: *Structured Document Image Analysis*. Springer-Verlag, New York.

[3] Baird HS (1993) Calibration of Document Image Defect Models, in *Proc. of Second Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, Nevada, pp. 1-16

[4] Chhabra AK and Phillips IT (1998) The second international graphics recognition contest – raster to vector conversion: a report. In: K. Tombre, and A.K. Chhabra (eds). *Graphics Recognition - Algorithms and Systems*. Volume 1389 of Lecture Notes in Computer Science, Springer-Verlag, Berlin, Germany, pp. 390-410.

[5] Elliman D (2001) Arc Segmentation in Engineering Drawings. In: Proc. *of 4th IAPR Workshop on Graphics Recognition*, Kingston, Canada, September 7-8, 2001.

[6] Foley JD, van Dam A, Feiner SK, and Hughes JF (1995) *Computer Graphics: Principles and Practice (Second Edition in C)*. Addison Wesley.

[7] Haralick RM (1989) Performance Assessment of Near Perfect Machines. *Machine vision and applications* 2:1-16.

[8] Haralick RM (1992) Performance Characterization in Image Analysis—Thinning, a Case in Point, *Pattern Recognition Letters* 13, 5-12.

[9] Heaton JC (1993) The Interaction Between Time-Nonseparable Preferences and Time Aggregation. *Econometrica* 61:353-385.

[10] Hilaire X (2002) An Arc Segmentation Algorithm (tentative title). *To appear in the post-workshop proceedings of the 4th IAPR Workshop on Graphics Recognition* (in the series of *Lecture Notes in Computer Science).*

[11] Hori O and Doermann DS (1996) Quantitative Measurement of the Performance of Raster-to-Vector Conversion Algorithms. In: *Graphics Recognition—Methods and Applications (Lecture Notes in Computer Science, vol. 1072)*, R. Kasturi and K. Tombre (eds), Springer, Berlin, pp. 57-68.

[12] Kanungo T, Baird HS, and Haralick RM (1995) Estimation and Validation of Document Degradation Models. In: *Proc. of Fourth Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, Nevada.

[13] Kanungo T, Haralick RM, and Phillips I (1993) Global and local document degradation model. In: *Proc. International Conference on Document Analysis and Recognition*, pp. 730-734, Japan.

[14] Kanungo T, Haralick RM, and Phillips I (1994) Non-linear local and global document degradation models. *Int. Journal of Imaging Systems and Technology* 5(4).

[15] Kong B et al. (1996) A Benchmark: Performance Evaluation of Dashed-Line Detection Algorithms. In: *Graphics Recognition—Methods and Applications (Lecture Notes in Computer Science, vol. 1072)*, R. Kasturi and K. Tombre (eds), Springer, Berlin, pp. 270-285.

[16] Liu W and Dori D (1997) A Protocol for Performance Evaluation of Line Detection Algorithms. *Machine Vision and Applications* 9(5):240-250.

[17] Liu W and Dori D (1998) Incremental Arc Segmentation Algorithm and Its Evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20(4):424-431.

[18] Liu W and Dori D (1999) Principles of Constructing a Performance Evaluation Protocol for Graphics Recognition Algorithms. In: *Performance Characterization and Evaluation of Computer Vision Algorithms*, R. Klette, S. Stiehl, and M. Viergever (eds.), Kluwer, pp. 97-106.

[19] Liu W and Dori D (2001) The Arc Segmentation Contest. In: Proc. of 4th IAPR Workshop on Graphics Recognition, Kingston, Canada, September 7-8, 2001. http://research.microsoft.com/users/wyliu/arcsegcontest.htm

[20] Madej D and Sokolowski A (1993) Towards Automatic Evaluation of Drawing Analysis Performance: A Statistical Model of Cadastral Map. In: *Proc. of Int. Conf. on Document Analysis and Recognition*, Tsukuba, Japan, pp. 890-893.

[21] Phillips IT and Chhabra AK (1999) Empirical Performance Evaluation of Graphics Recognition Systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21(9): 849-870