A Benchmark: Performance Evaluation of Dashed-line Detection Algorithms

Bin Kong[1], Ihsin T. Phillips[2], Robert M. Haralick[1],
Arathi Prasad[3], and Rangachar Kasturi[3]

[1] Department of Electrical Engineering
University of Washington
Seattle, Washington 98195, USA

[2] Department of Computer Science
Seattle University
Seattle, Washington 98122, USA

[3] Department of Computer Science and Engineering
Pennsylvania State University
University Park, Pennsylvania 16802, USA

Abstract: This paper describes a protocol for systematically evaluating the performance of dashed-line detection algorithms. It includes a test image generator which creates random line patterns subject to prespecified constraints. The generator also outputs ground truth data for each line in the image. The output of the dashed line detection algorithm is then compared to these ground truths and evaluated using a set of criteria.

1 Introduction

Systems which convert existing paper-based engineering diagrams into electronic format are in demand and a few have been developed. However, the performance of these systems is either unknown, or only reported in a limited way by the system developers. A formal evaluation for these systems, or their subsystems, would contribute to the advancement of the field. Since many drawings include some form of dashed lines, we present a dashed-line generation protocol and a performance evaluation protocol for evaluating dashed-line detection algorithms [1]. The benchmark is designed to be used by the recognition system researchers and developers for testing and enhancing their dashed-line recognition algorithms. Figure 1 shows an object-process diagram of the benchmark.

Users of our benchmark first need to use the dashed-line generator to generate a set of test images of various complexities. The generator also produces the ground truth for each image it generates. The algorithm being evaluated operates on these images to perform detection and produces output in a pre-defined format. Next, the performance evaluator takes the detected lines produced by the algorithm and the corresponding ground truth lines produced by the generator and performs evaluation based on a set of criteria. The results of the evaluation are displayed in tabular form.

Note that the contents of these evaluation tables are computed facts. No scores are assigned to algorithms, since assignment of scores to algorithms requires a definition of weights associated with each of the different types of errors. The values of these weights depend on the particular application. In this paper we do not discuss these weights.

This paper is organized as follows: Section 2 contains the procedures for the generation of dashed-line test images. The performance evaluation protocol and its output, performance evaluation tables, are described in Section 3 followed by conclusions in Section 4. Several appendices describe in detail the conventions used and the specifications of various data sets and file formats.

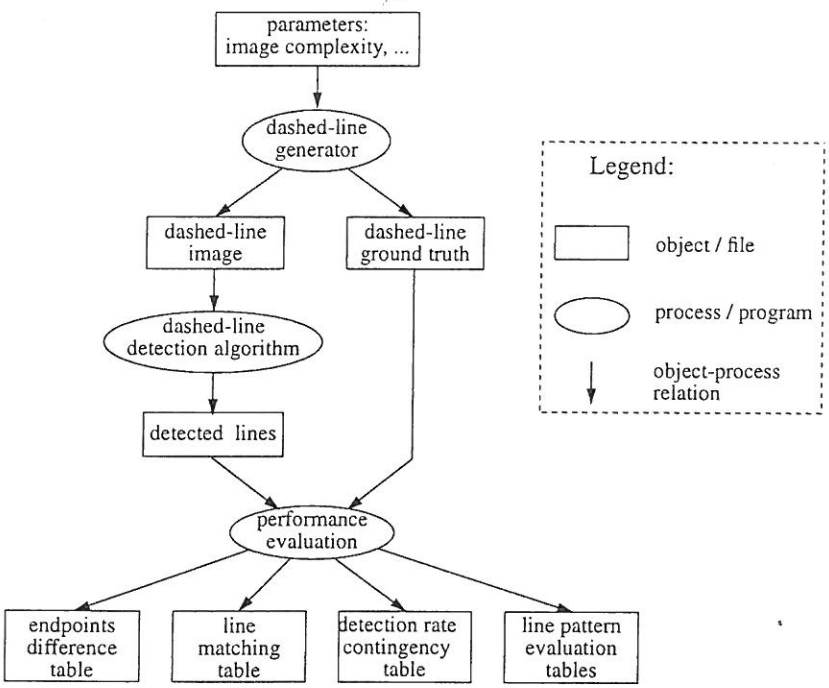
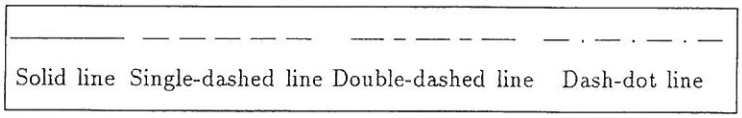


Fig. 1. The object-process diagram of the benchmark. The user supplies the dashed-line generator image complexity parameter. The generator produces a dashed-line image and the corresponding dashed-line ground truth. Detection algorithms perform detection of dashed lines in the image and produces detected lines. The performance evaluator takes the detected lines produced by the algorithm and the ground truth produced by the generator, and performs evaluation. The results of the evaluation are displayed in tabular form.

2 Dashed-line Test Image Generation

Dashed-line detection algorithms are tested using a set of images generated by the Dashed-line Test Image Generator described in this section. Standard graphics generation procedures are used by the generator to create a variety of dashed-lines in various positions and orientations in the image. The parameters controlling the generation procedures are randomly varied to create a rich variety of test images. There are four basic types of line patterns generated. These are shown below:



Lines within each test image vary in length, thickness and orientation. These variations as well as the composition of the test images are determined by the degree of complexity of the desired test image as follows:

- **Simple Test Image**

  A *simple* image includes only single-dashed lines in horizontal, vertical, (and possibly $\pm 45°$ diagonal) directions. The dashed-line segment and gap lengths may vary up to $\pm 10\%$ within each line. A typical *simple* image is shown in Figure 2. The size of the image is fixed at 1000×1000 pixels.

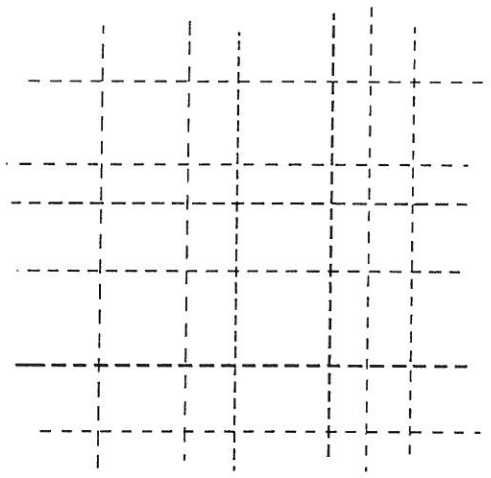


Fig. 2. An image with only horizontal, vertical, and (possibly) diagonal dashed lines representing the "simple" class of images.

- **Medium Test Image**

  In addition to the line patterns included in *simple* images, *medium* complexity images include lines with arbitrary orientation. All three types of dashed lines are included as well as polygons with and without hatching. A typical medium complexity image is shown in Figure 3. Maximum image size is 4000 x 4000 pixels. The segment and gap length parameters may vary by as much as ±40% of the nominal values.

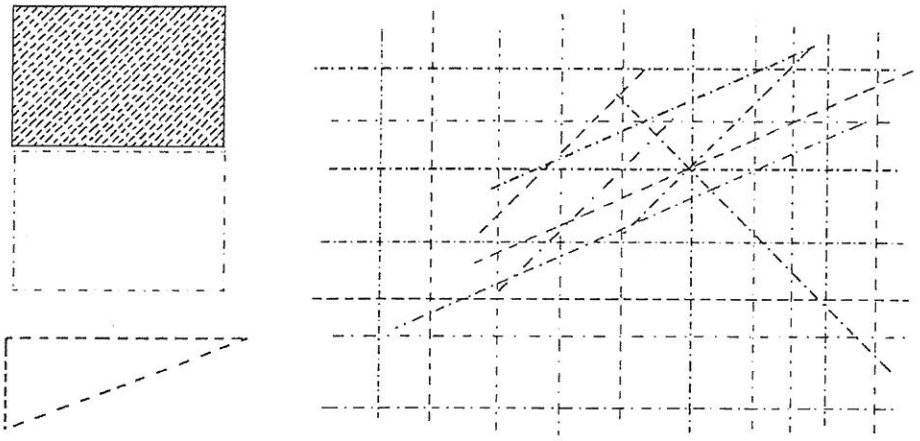


Fig. 3. An image with straight dashed lines and other objects representing the "medium" class of images.

- **Complex Test Image**

  These images include everything in a *medium* complexity image. In addition, curved lines and circles are added and annotations are superimposed on the graphics to further increase the complexity of the image. The length parameters could vary up to ±100% of their nominal values. A typical complex test image is shown in Figure 4.

For a complete specification of the three classes of test images see Appendix 1. All test images are saved by the generator in 8-bit binary TIFF format. More details on the dashed-line primitives generation are given in the following sections.

2.1 Generation of Straight Lines

Line orientation, line length, and the start coordinates of the line are chosen randomly based on the seed input by the user. Once the start and end coordinates of the line are found, the task is now one of determining the points along this

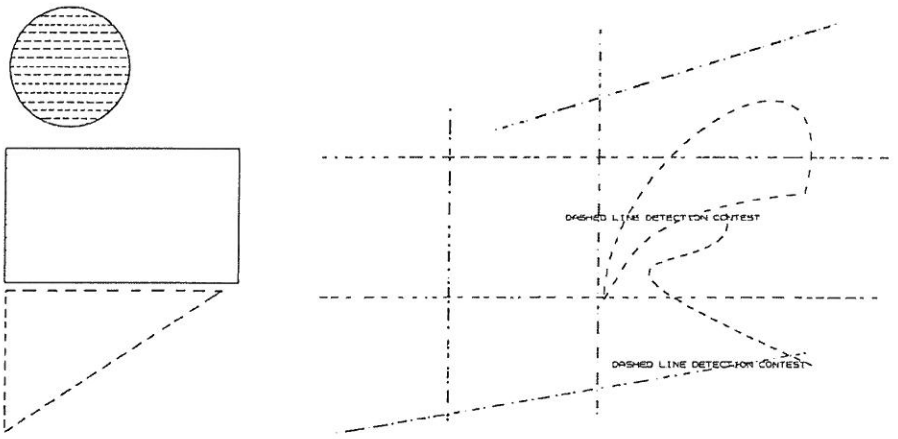


Fig. 4. An image with straight and curved dashed lines, other objects, continuous lines and lines of text representing the "complex" class of images.

line in terms of pixel coordinates and setting the pixels to the chosen intensity. The calculation of points along the line is done using the midpoint form of Bresenham's line algorithm [2]. In order to draw a dashed line, the number of pixels equal to a dash length are set to the foreground intensity while the next number of pixels equal to a gap length are set to the background intensity. This alternate turning "on" and "off" of the pixel intensity is repeated for an entire length of a complete dashed line.

After this stage, all the points along the dashed line are obtained. The next stage is thickening the line. The line is thickened by enclosing it in a rectangle and filling the area. The breadth of the rectangle is set to the line thickness. The rectangle is filled with the foreground intensity by a recursive seed fill algorithm [2]. The seed is taken to be any point central to the area to be filled.

2.2 Generation of Curves

A curved line is drawn as a Bezier curve [2]. The dashed nature of the curve is obtained by toggling the pixel intensities. Four initial control points are chosen randomly and the curve is approximated through these points. The iterative algorithm has the index varying between 0 and 1 in steps of a small, fixed fraction where each succeeding point is calculated as a weighted sum of Bernstein polynomials,

$$x = c_{0x}(1.0-t)^3 + 3c_{1x}t(1.0-t)^2 + 3c_{2x}t^2(1.0-t) + c_{3x}t^3 \qquad (1)$$

$$y = c_{0y}(1.0-t)^3 + 3c_{1y}t(1.0-t)^2 + 3c_{2y}t^2(1.0-t) + c_{3y}t^3 \qquad (2)$$

where x and y are the points on the curve; t is the index of the iteration; and c_0, c_1, c_2, and c_3 are the four control points.

2.3 Generation of Circles

The points on a circle are determined using Bresenham's circle algorithm [2]. Each iteration produces eight equidistant points along the circumference until the desired resolution is achieved. A second circle with the same center but a decreased radius is drawn and the area between the two circles is filled to represent the finite thickness.

3 Performance Evaluation Protocol

When the dashed-line test image generator produces a test image, it also produces, in a predefined format, the corresponding *ground truth* for each line it generates. We require that the output from a dashed-line detection algorithm conform to this predefined format. Detailed specifications of this format are given in Appendix 2. The performance evaluation steps are given below:

1. We assign unique IDs (from 1, 2,...) to the ground truth lines and the detected lines in the order in which they appear in the files.
2. We compute the *matches* between the ground truth lines and the detected lines regardless of their line types. If there is an offset (a horizontal shifting and/or a vertical shifting) from the detected lines to the ground truth lines, the detected lines are shifted accordingly and the matches are recomputed. The results of matching are output as several tables by the evaluation algorithm. The formats for these tables are described in sections 3.2 and 3.3.
3. For each detected line that matches a ground truth line, we check whether its given line type is correct. The set of detected lines with correct line types is collected for *line pattern* evaluation. We call this set of detected lines the *correctly-matched lines*. For each of the correctly-matched line pairs, we evaluate its line pattern and an evaluation table for each line category is computed (described in section 3.4). No further evaluation is given to any detected line having an incorrect line type.

3.1 Matching Criteria

Our performance evaluation is based on matching the output of the algorithm against the corresponding ground truth of the test image. We consider that a detected line d matches a ground truth line g if:

1. The *angle* between the detected line d and the ground truth line g is less than 3 degrees,
$$angle(d, g) \leq 3$$
2. The *distance* between lines d and g is less than 5 pixels,
$$llDist(d, g) \leq 5$$

3. The *Relative overlap* function of d and g with respect to the orientation of g is at least 0.8,

$$Reloverlap(d, g, \alpha_g) > 0.8, \text{where } \alpha_g = orient(g).$$

For a precise definition of these functions see Appendix 3. The thresholds were determined heuristically based on the properties of the test images such as dashed line thickness, line separation, minimum line length, etc.

A problem arises if a detected line matches multiple ground truth lines, and/or if there are other detected lines which also match the same ground truth line. In the first case, it is not correct to consider that the same detected line matches all ground truth lines. In this case, we only match the detected line to that ground truth line with which it has maximum *relative overlap*; other ground truth lines are considered as misdetections. In the second case, it is not correct to consider multiple detections of the same ground truth target as independent correct detections. Hence, only the detected line having the largest *relative overlap* with the ground truth line is considered matched with the ground truth; other detected-lines are considered as false alarms.

3.2 Offset Evaluation

During offset estimation, only matched detected-lines are considered. For each matched detected-line, the rows and the columns differences between its two endpoints and the two endpoints of its corresponding ground truth are computed. Note that there are four values here, i.e., a row and a column difference for each pair of matched endpoints. There will be two sets of the row differences and two sets of column differences after all matched lines are considered. The means and the variances for each of these four sets are estimated as follows:

1. Compute the initial mean and the initial variance for the set.
2. If a value within the set is greater than twice the estimated standard deviation of the set, the value is eliminated from the set. This step helps to group only those detected lines which are uniformly offset from their corresponding ground truth lines by avoiding the influence of outliers.
3. Compute the new mean and the new variance for the set.
4. Repeat steps 2 and 3 until the values become stable.

Finally, the offset estimation is computed using the four final means and the four final variances. Recall that there are two row means and two column means in the four sets. If the smaller of the two row variances is less than a predefined threshold value (we use 4.0 here), then the corresponding mean values are considered as the offset in row values. Otherwise, we consider that there is no offset. The column offset is computed in a similar fashion.

All this information is maintained in the *endpoints difference table*. The entries of the *endpoints difference table* give the differences, in term of columns and rows, between the endpoints of the *detected lines* and the endpoints of the

matched *ground truth lines*. The estimated mean and the estimated variance of the offsets are displayed in the last two rows of the *endpoints difference table*. If for some reason the *detected lines* are uniformly shifted either horizontally and/or vertically, the shift will be shown in the *endpoints difference table*, i.e., all column entries and/or all row entries will be identical. If a global offset is detected, the entire set of the detected lines are shifted, and the matches are recomputed.

3.3 Line Match Evaluation

A *match table* is also created during *line match evaluation*. A *match table* entry is either a 1 (a match) or a blank (no match). A 1 at the entry (i, j) indicates that the ith *detected line* matches the jth *ground truth line*. Within the *match table*, we also include misdetections and false-alarms. The entries for the ground truth misdetections are given in the last row of the *match table*. The entries for the falsely detected lines are given in the last column of the *match table*. A 1 on jth entry of the misdetection row indicates that the jth *ground truth line* was not detected by the algorithm. A 1 on ith entry of the false-alarm column indicates that the algorithm produced a line which is not a part of the ground truth.

Figure 5 contains templates of the two *contingency tables*. The top table contains the total number of *ground truth lines* which have been labeled as each of the line types: Solid line, Dashed line, Double-dashed line, and Dash-dotted line. The bottom table contains the correct-detection rate, the mis-label rate, the misdetection rate, and the false-alarm rate for each of the line types. These are calculated using the following equations:

$$P_{\text{correct}} = \frac{(A_1 + B_2 + C_3 + D_4)}{N_g} \tag{3}$$

$$P_{\text{mis-lab}} = \frac{\sum_i (a_i + b_i + c_i + d_i)}{N_g} \tag{4}$$

$$P_{\text{mis-detect}} = \frac{\sum_i M_i}{N_g} \tag{5}$$

$$P_{\text{false}} = \frac{\sum_i F_i}{N_d} \tag{6}$$

where the various quantities are shown in the top table of Figure 5 and the values N_g and N_d are the total number of ground truth lines and total number of detected lines, respectively.

3.4 Line Pattern Evaluation

Figure 6 contains a template of the *Single-dashed line pattern evaluation table*. The table displays the mean and variance of the dash length and the mean gap length for each matched pair of *ground truth* and *detected* lines. Chi-squared values ($\sum \frac{(D-G)^2}{G}$) are also computed. Similar tables are also created for other types of dashed lines.

Detected Lines

Ground Truth Lines	Solid	Double-dash	Single-dash	Dash-dot	Mis-detect
Solid	A_1	a_2	a_3	a_4	M_1
Double-dash	b_1	B_2	b_3	b_4	M_2
Single-dash	c_1	c_2	C_3	c_4	M_3
Dash-dot	d_1	d_2	d_3	D_4	M_4
False alarm	F_1	F_2	F_3	F_4	

A_1, a_i = number of ground truth Solid lines detected as Solid, Double-dash, etc.
B_2, b_i = number of ground truth Single-dash lines detected as Solid, Double-dash, etc.
C_3, c_i = number of ground truth Double-dash lines detected as Solid, Double-dash, etc.
D_4, d_i = number of ground truth Dash-dot lines detected as Solid, Double-dash, etc.
M_i = number of ground truth lines of Solid, Double-dash, etc. that were misdetected.
F_i = number of Solid lines, Double-dash lines, etc. that were falsely detected.

Detected Lines

Ground Truth	Line-type i	Not Line-type i	Not detected
Line-type i	P_{correct}	$P_{\text{mis-lab}}$	$P_{\text{mis-det}}$
Not ground truth	P_{false}		

Fig. 5. The top table contains the number of *ground truth lines* which are labeled as Solid lines, Dashed lines, Double-dashed lines, and Dash-dotted lines. The bottom table contains the correct-detection rate, the mis-label rate, the mis-detection rate, and the false-alarm rate.

4 Conclusions

A formal protocol for systematically evaluating the performance of dashed-line detection algorithms was described in this paper. A test image generator which generates a rich variety of test images for evaluating dashed-line detection algorithms was presented. The evaluation procedures as well as the test images, including source code, were made available for ftp well in advance of the contest. The participants were required to run their software for dashed-line detection on a new set of test images generated just before the contest. One of the limitations of this contest is that the testing was limited to synthetically generated images. This was done so that random images with accurate ground truth data could be

Single-dashed Line Pattern Evaluation Table

Index		Dashes				Gaps	
		Mean		Variance		Mean	
G	D	G	D	G	D	G	D
g_{21} g_{22} : : : g_{2b}	d_{21} d_{22} : : : d_{2b}						
Chi Square							

Fig. 6. The table displays the mean and the variance of the lengths of dashes and the mean of the lengths of the gaps for each matched pairs of the *ground truth line* and the *detected line*.

generated just before testing to ensure that the images used in testing would not be known to anyone. Testing on scanned paper documents would have required manual ground truthing prior to the contest. Clearly, for a more rigorous testing in a non-contest environment it is essential to test on scanned document pages from a variety of sources. Another limitation of the evaluation protocol is that it does not evaluate matching of dashed lines formed by circles and other curves. Such matches were subjectively evaluated.

Acknowledgements

We would like to acknowledge the help of David Kosiba for his help in organizing this contest. We would like to thank the graphics recognition community for its keen interest and its input on the format and organization of this contest. In particular, we are indebted to Karl Tombre for his enthusiasm, insight and support for this activity. We also thank the reviewers for many helpful suggestions in enhancing the readability of this paper.

Appendix 1: Complete Specifications of Test Image Classes

A1.1 Simple Test Image

a. The image size is 1000 x 1000 pixels.
b. Only *single-dashed lines* are used.
c. Only horizontal, vertical, and diagonal (±45%) dashed lines.
d. A minimum distance of 50 pixels between lines.
e. Minimum number of dashed lines is 10 while the maximum number is 20.
f. Segment length varies from 10 - 30 pixels.
g. Gap length varies from 1 - 10 pixels.
h. Segment length to gap length ratio varies from 0.8 - 2.0.
i. Thickness of a dashed line varies from 3 - 30 pixels.
j. The above parameters (f, g) have variations of the order of ±10%.
k. Minimum length of a dashed line is 50 pixels.

A1.2 Medium Test Image

a. The image size varies from 1000 x 1000 to 4000 x 4000.
b. All three kinds of dashed lines (*single-dashed, double-dashed, and dash-dot*) are present.
c. Lines can be in any random orientation, but only 4 orientations possible in an image (Note that all lines are straight lines).
d. Angles in an image have a minimum difference of 20 degrees between them.
e. Minimum number of dashed lines is 20 and the maximum number is 40.
f. The segment and gap-length parameters have variations of up to ±40%.
g. All other specifications are same as those for Simple images.

A1.3 Complex Test Image

a. The image size varies from 4000 x 4000 to 8000 x 8000.
b. In addition to the three kinds of dashed lines, solid lines and text may also be present.
c. Both straight and curved dashed lines are possible.
d. Minimum distance between lines equal to the thickness of lines.
e. Minimum number of dashed lines is 30 while the maximum number is 100.
f. Segment length to gap length ratio varies from 0.8 - 4.0.
g. The segment-length parameters could vary up to ±100% and missing segments are possible.
h. All other specifications are same as those for Medium images.

Appendix 2:
Output Specification for Dashed-line Detection Algorithms

The output of the dashed line detection algorithms should conform to the following specifications:

1. One text-line for each detected line.
2. For each detected line, the output text-line is of the format:

$$n\ c_1\ r_1\ c_2\ r_2[\textit{additional-parameters}]$$

   where n is 1 for solid line, 2 for single-dashed line, 3 for double-dashed line, and 4 for dash-dot line and c_1, r_1 and c_2, r_2 are the column and the row positions of the first and the second end points of the detected line. We require that $c_1 < c_2$. If $c_1 = c_2$, then $r_1 < r_2$.
3. For a Single-dashed line, the output text-line includes the following additional parameters: mean-dash-length, dash-variance and mean-gap-length. Note that, the *gap-length variance* will be identical to that of *dash-variance*, therefore, the *gap-length variance* is not required.
4. For a Double-dashed line, the output text-line includes the following additional parameters: *mean-dash1-length, dash1-variance, mean-dash2-length, dash2-variance, and mean-gap-length.* Length of dash-1 should be longer than that of dash-2.
5. For a Dash-dot line, the output text-line includes the following additional parameters: *mean-dash-length, dash-variance, mean-dot-width, width-variance, and mean-gap-length* where *mean-dot-width* is the average diameter of the dots.

Appendix 3: Conventions and Definitions

In this appendix, we give the conventions and definitions that are used in this work.

A3.1 Image Coordinate System

An image is given by columns and rows of pixels. In an 8-bit binary image, a foreground pixel has the value 255 and a background pixel has the value 0. We use the *column-row*[1] coordinate system, (c-coordinate, r-coordinate), to represent a pixel's position within an image. The origin of this system, $(0, 0)$, is at the top-left corner pixel of the image.

[1] We use the column-row notation to avoid the confusion caused by the orientation differences in the (x, y) Cartesian system notation and the $[i, j]$ image array notation.

A3.2 Definitions

- **Line orientation**
  The orientation of a line segment $l = (c_1, r_1, c_2, r_2)$, denoted as $orient(l)$, is the angle between l and the c-axis (see Figure 7). The range of $orient(l)$ is $(-90°, 90°]$. The function $orient(l)$ is

$$orient(l) = \begin{cases} \arctan\left(\dfrac{r_2 - r_1}{c_2 - c_1}\right), & \text{if } c_1 \neq c_2 \\ 90°, & \text{if } c_1 = c_2 \end{cases}$$

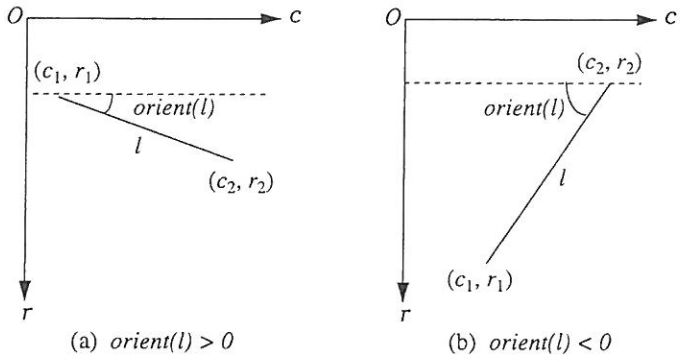


Fig. 7. The orientation of a line segment.

- **Angle between two line segments**
  The angle between two given line segments l_1 and l_2, is defined as the included angle between l_1 and l_2, and denoted as $angle(l_1, l_2)$ (see Figure 8). The range of $angle(l_1, l_2)$ is $[0°, 90°]$. The function $angle(l_1, l_2)$ is

$$angle(l_1, l_2) = \begin{cases} |orient(l_1) - orient(l_2)|, & \text{if } |orient(l_1) - orient(l_2)| \leq 90° \\ 180° - |orient(l_1) - orient(l_2)|, & \text{otherwise} \end{cases}$$

- **Point-line distance**
  The point-line distance between a point $p = (c, r)$ and a line segment $l = (c_1, r_1, c_2, r_2)$, is the orthogonal distance from p to l and denoted as $plDist(p, l)$ (see Figure 9). The point-line distance function is

$$plDist(p, l) = \frac{|(c_2 - c_1)r - (r_2 - r_1)c - (r_1 c_2 - r_2 c_1)|}{\sqrt{(c_2 - c_1)^2 + (r_2 - r_1)^2}}.$$

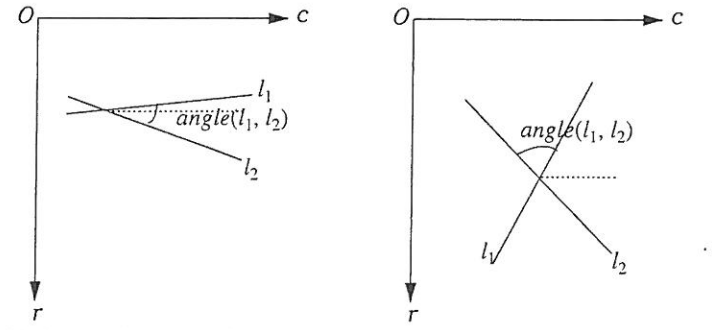


Fig. 8. The angle between two line segments.

- **Line-line distance**
  Let $l_1 = (c_{11}, r_{11}, c_{12}, r_{12})$ and $l_2 = (c_{21}, r_{21}, c_{22}, r_{22})$ be two line segments. The midpoints of l_1 and l_2 are m_1 and m_2, respectively. The line-line distance of l_1 and l_2, $llDist(l_1, l_2)$, is defined as the average of the point-line distance of m_1 to l_2 and that of m_2 to l_1 (see Figure 9). The line-line distance function is

$$llDist(l_1, l_2) = \frac{1}{2}\left(plDist(m_1, l_2) + plDist(m_2, l_1)\right).$$

  where

$$m_1 = \left(\frac{c_{11} + c_{12}}{2}, \frac{r_{11} + r_{12}}{2}\right) \text{ and } m_2 = \left(\frac{c_{21} + c_{22}}{2}, \frac{r_{21} + r_{22}}{2}\right).$$

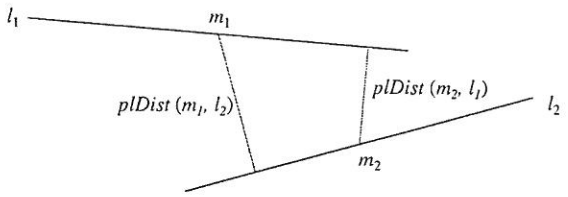


Fig. 9. The line-line distance between two line segments.

- **α-projection of a line segment**
  The α-projection of a line $l = (c_1, r_1, c_2, r_2)$ is the projection of l onto the given orientation $\alpha \in (-90°, 90°]$. The α-projection of l, $proj(l, \alpha)$, is also a line segment. Its two endpoints (c'_1, r'_1) and (c'_2, r'_2) are the projections of

(c_1, r_1) and (c_2, r_2) onto the orientation α, respectively (see Figure 10). The α- projection is given by

$$proj(l, \alpha) = \begin{cases} (c_1', r_1', c_2', r_2'), & \text{if } |\alpha - orient(l)| \leq 90° \\ (c_2', r_2', c_1', r_1'), & \text{otherwise} \end{cases}$$

where

$$\begin{aligned} c_1' &= \cos\alpha(c_1 \cos\alpha + r_1 \sin\alpha) \\ r_1' &= \sin\alpha(c_1 \cos\alpha + r_1 \sin\alpha) \\ c_2' &= \cos\alpha(c_2 \cos\alpha + r_2 \sin\alpha) \\ r_2' &= \sin\alpha(c_2 \cos\alpha + r_2 \sin\alpha). \end{aligned}$$

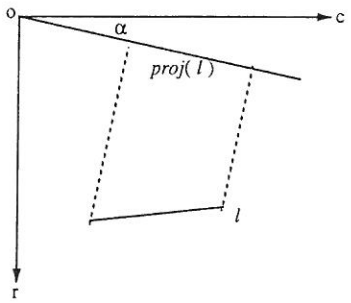


Fig. 10. The α-projection of a line segment.

- **Overlap between two line segments – a relationship function**
  The α-overlap between two line segment l_1 and l_2, $overlap(l_1, l_2, \alpha)$, is a relationship function of l_1 and l_2 with respect to a given orientation α. Suppose T_a and T_d are two given thresholds which are determined by the user based on the application in which the dashed line detection algorithm is used. T_a is a threshold for the angle between two line segments and T_d is a threshold for the line-line distance. If $angle(l_1, l_2)$ is not greater than T_a, and $llDist(l_1, l_2)$ is not greater than T_d, we say that l_1 and l_2 are *sufficiently close* to each other.
  The α-overlap of l_1 and l_2 is defined as the length of the common part of their α-projections if l_1 and l_2 are sufficiently close, and is defined as 0 otherwise. The function $overlap(l_1, l_2, \alpha)$ is

$$overlap(l_1, l_2, \alpha) = \begin{cases} length(proj(\ l_1, \alpha) \cap proj(l_2, \alpha)), \\ \qquad\qquad \text{if } angle(l_1, l_2) \leq T_a \text{ and } llDist(l_1, l_2) \leq T_d \\ 0, \qquad\qquad \text{otherwise} \end{cases}$$

- **Relative overlap**
  The relative overlap of two line segments l_1 and l_2 is defined as the ratio between the overlap function $overlap(l_1, l_2, \alpha)$ and the length of the longer segment:

$$reloverlap(l_1, l_2, \alpha) = \frac{overlap(l_1, l_2, \alpha)}{\max(length(l_1), length(l_2))}$$

References

1. R.M. Haralick, "Detection Performance Methodology" *Proceedings Image Understanding Workshop,* Palm Springs, California, February 1996, Vol II, pp. 981-983.
2. J.D. Foley and A. Van Dam, "Computer Graphics, Principles and Practice" Addison-Wesley, Reading, Massachusetts, 1991

This article was processed using the LaTeX macro package with LLNCS style