Impact of Sparse Pixel Vectorization Algorithm Parameters on Line Segmentation Performance

Wenyin Liu[1,2], Xiaoyu Wang [2], Long Tang [2], and Dov Dori[3]

[1] Microsoft Research, Sigma Center,
#49 Zhichun Road, Beijing 100080, PR China
wyliu@microsoft.com
[2] Department of Computer Science and Technology, Tsinghua University,
Beijing 100084, PR China
[3] Faculty of Industrial Engineering and Management,
Technion – Israel Institute of Technology, Haifa 32000, Israel
dori@ie.technion.ac.il

Abstract. We present the impact of parameters in the Sparse Pixel Vectorization algorithm on the performance of the algorithm, which is evaluated using a pixel level performance evaluation protocol. Both theoretical and experimental analyses are presented, which show that the experimental results are well in accordance with the theoretical analyses. Both prove the robustness of the algorithm. Meanwhile the parameter values for the best performance of the algorithm are also found.

1 Introduction

Vectorization, also known as raster to vector conversion, is a process that extracts vectors—line segments—from raster images of line drawings. Vectorization is widely used in the area of Document Analysis and Recognition (DAR) as a preprocessing step for high-level object recognition, such as optical character recognition (OCR) and graphic objects recognition. Basic vectorization concerns grouping the pixels in the raster (binary) image into raw wires that are described by several attributes, such as characteristic points and line width. Advanced vectorization includes line fitting and extending, which yields fine wires. We refer to crude vectorization as the basic vectorization process that takes a raster image as input and yields coarse wire fragments, which may be bars (non-zero width line segments) or polylines (chains of bars linked end to end). Crude vectorization is a crucial preprocessing step in the interpretation process of line drawings in general and engineering drawings in particular, as the latter type of drawings requires maximal precision to avoid false alarms and misses in subsequent processing stages.

A typical vectorization algorithm requires the setting of a (sometimes significant) number of thresholds and other parameters. Examples include mesh size in the mesh pattern based algorithms (Lin et al. 1985) and scan line interval in the OZZ algorithm

Atul K. Chhabra and D. Dori (Eds.): GREC'99, LNCS 1941, pp. 335-344, 2000.

(Dori et al. 1993). As is the case with many other algorithms, setting the values of these parameters can have a profound effect on the algorithm performance.

Different drawings may have different parameter settings at which they exhibit peak performance. Based on theoretical analysis and supported by experiments, a subset of the parameters may be set to values that are near optimal for most drawings. To ensure algorithm robustness, the parameter settings should not affect the performance in a range near the default values to a significant extent.

For best performance, a trial-and-error process, which is part of the algorithm development, determines these parameters to some default values. In most cases, the performance is judged by the developers using human vision perception. Quantitative evaluation is seldom reported. Consequently, the values set for the parameter do not guarantee peak performance in terms of objective criteria.

The Sparse Pixel Vectorization (SPV) algorithm (Dori and Liu 1999) is a basic (crude) vectorization method that is fast due to accessing the pixels sparsely while preserves the original shape information to a large extent. Like most other algorithms, SPV features a number of parameters whose value settings may affect its performance. In this paper we examine the effect of these parameter values on SPV's performance, using the performance evaluation protocol we developed (Liu and Dori 1997).

We carry out a theoretical analysis for the parameter values for the best performance. We then determine a combination of these parameter values for which SPV operates at its best performance. The experimental results we present are in good accord with the theoretical analysis, and both prove the robustness of the SPV algorithm.

2 The Sparse Pixel Vectorization Algorithm

To familiarize the reader with SPV in the context of this research, we present the algorithm briefly with emphasis on its parameters and their settings. The SPV algorithm scans the entire image every several pixels. The *scan line interval* is the first parameter in the algorithm. When a scan line encounters a foreground pixel (e.g., on the boundary of a line area), a process starts aiming at finding the first reliable medial point (e.g., P_0 in **Figure 1**), which is the midpoint of both the horizontal and the vertical runs of the foreground area passing through it.

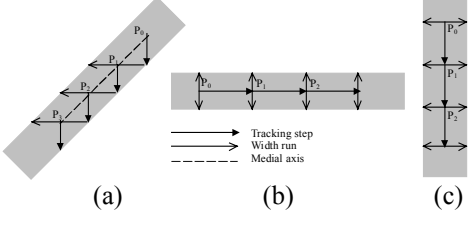

Figure 1. Illustration of the SPV general tracking process. (a) Slant segment. (b) Horizontal segment. (c) Vertical segment

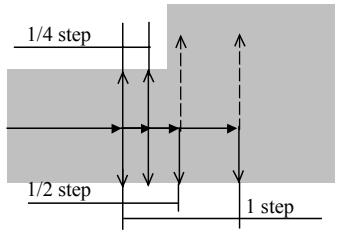

Figure 2. Illustration of the Junction Recovery Process

The longer of the two (horizontal and vertical) runs at that point is determined as the *tracking direction* or the *length direction* (the line's inclination, either horizontal or vertical) of the line area. The width direction is defined to be orthogonal to the length direction. Once the length (tracking) direction is determined, the *General Sparse Pixel Tracking* process starts from the reliable medial point along the length direction. The process takes a *tracking step* in the tracking direction. A tracking step is the distance from the current medial axis point to a *stop pixel* – a foreground pixel within the current line area. The tracking step should be less than a predefined *maximal tracking step*. This implies that the stop pixel, where the current tracking step stops, can be one of the following: (1) a pixel on the edge of the line area (the previous pixel of the encountered background pixel), as shown in **Figure 1**(a), (2) inside the line area, when the distance exceeds the maximal tracking step, as shown in **Figure 1**(b) and (c).

This *maximal tracking step* is the second SPV parameter that requiring setting. At each tracking step, we calculate the width run (which is the cross section of the line area parallel to the width direction at the stop pixel). We then take the width run midpoint at a new medial axis point for the next iteration of tracking if the following set of "next iteration conditions" is met. (1) Width continuity: The largest difference among line widths, represented by the width runs at the medial axis points found during tracking, along a neighborhood of some small number of adjacent medial axis points, is below a predefined threshold. We define this number as the *maximal width difference*, which is the third parameter in the SPV algorithm. (2) Sole occupancy:The medial point should not be occupied by another, already detected vector. (3) Direction preservation: The length direction at the stop pixel should be the same as the length direction at the last medial axis point during one tracking procedure. (4) Tracking length: The tracking step is non-zero.

If one or more of the conditions (1)-(3) of the next iteration conditions set is not met, an iterative Junction Recovery process, illustrated in **Figure 2**. Backtracking to the last medial axis point and going ahead with half of the normal tracking step, we may find that the violated conditions are met at the new stop pixel. If so, the tracking procedure continues from the new medial axis point with the normal tracking step. If the conditions are still not met, we backtrack again to the last medial axis point and cut the tracking step by half again. We iterate this "backtrack and cut by half" process until either we find a stop pixel that meets the conditions or the tracking step is reduced to zero. If the tracking step becomes zero, the tracking procedure for the current part of the line area stops and another tracking starts with a new foreground pixel found by the scan line.

The result of the line tracking procedure is a chain of point, called *polyline*, that lies on the approximate medial axis of the black area. Although the polyline can be considered as a vector representation of the corresponding black area, some of its detected intermediate medial axis points are redundant because they are (approximately) on the straight line segments formed by their neighbor points. To obtain the minimal description vector representation of this black area, these points should be removed from the polyline's point chain. This point removal process, called *Polygonalization*, should result in a polyline with the smallest number of edges (or vertices) that approximate the original line's black area shape.

Figure 3. Illustration of the polygonalization approximation

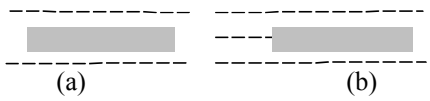

Figure 4. Illustration of the effect of the scan line interval parameter: (a) Larger one: line not detected. (b) Smaller one: line detected

The SPV Polygonalization process, shown in **Figure 3**, ultilizes the polygonalization algorithm developed by Sklansky and Gonzalez (1980). It tests the criticality of the points in the chain and removes non-critical points from the chain. The criticality of a point in a point chain is determined by its distance to the line joining the two neighboring critical points on both sides of the tested point. for three consecutive points. If the distance of the middle point to the segment formed by its two immediate neighbor is not greater than a predefined threshold ε, the middle point is non-critical and should be removed from the chain. This procedure is repeated until no more points can be removed from the chain. The predefined threshold ε is the fourth parameter in SPV and is referred to as the *maximal polygonalization error*.

3 Theoretical SPV Parameters Analysis

We have defined four parameters that may affect the vectorization performance: the *scan line interval*, the *maximal tracking step*, the *maximal width difference*, and the *maximal polygonalization error*. In order to see the impacts of the SPV parameter values on its performance, we define a set of quantitative objective performance indices.

3.1 The Performance Evaluation Protocol

We use the performance evaluation protocol for general line detection algorithms (Liu and Dori 1997). Since SPV is a vectorization algorithm, we require that it preserves the original image shape as much as possible. The pixel level performance indices (Liu and Dori 1997), explained below, are therefore used. The total Pixel Detection Rate for the entire image, D_p, is defined as the ratio of those foreground pixels covered by the detected vectors from the image to the total foreground pixels in the image. D_p represents the detection ability of the algorithm. The bigger D_p, the better the vectorization algorithm. The total Pixel False Alarm Rate for the entire image, F_p, is defined as the ratio of those background pixels covered by the detected vectors from the image to the total area covered by these detected vectors. It represents the error of the vectorization. The smaller F_p, the better the vectorization algorithm. A good vectorization algorithm requires a large D_p and a small F_p. Therefore the Pixel Recovery Index (PRI) is defined as PRI = $(D_p+1-F_p)/2$. PRI is a single index that represents the performance of the algorithm. The bigger the PRI, the better the vectorization algorithm.

3.2 The Scan Line Interval

The scan line interval is the distance in pixels between two adjacent scan lines. As **Figure 4** shows, the raster image is scanned line by line. Only when the scan line encounters a black pixel can a sparse pixel tracking procedure start. If no scan line encounters a pixel of a line black area, the whole line may not be detected. Therefore, as the scan line interval increases, more lines can be missed.

As the scan line width increases, D_p is expected to decrease. However, F_p is not directly affected, since no more false lines are detected. Overall, PRI decreases accordingly. This phenomenon, shown in **Figure 4**, often occurs when the method is applied on images with many thin, nearly horizontal lines. Theoretically, if the value of the scan line width is set to 1 pixel, all lines will be encountered by the scan lines and the PRI is therefore the biggest for all images. Nevertheless, there spare-pixel effect will be much smaller and the speed of the algorithm will be the slowest.

3.3 The Maximal Tracking Step and the Maximal Width Difference

As we can see from **Figure 5**, the maximal tracking step and the maximal width difference can jointly affect the performance of detecting some special areas.

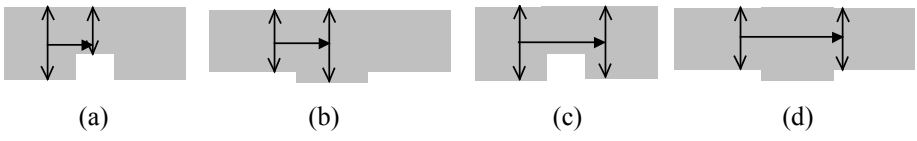

Figure 5. Illustration of the impacts of the maximal width difference and the maximal tracking step

There is a cavity in the line area in **Figure 5**(a) and a protrusion in **Figure 5**(b). Both may violate the condition of width continuity constrained by the parameter of the maximal width difference during the sparse pixel tracking process. When the condition is violated, the general tracking process pauses and a junction recovery process begins. As shown in Figure 5(a) and (b), if the maximal tracking step is smaller than the length of the cavity/protrusion area and the maximal width difference is smaller than the depth of the cavity/protrusion area, the whole area is detected as three distinct line segments. The middle of these is the cavity/protrusion area. On the other hand, if the maximal tracking step is bigger than the length of the cavity/protrusion area or the maximal width difference is bigger than the depth of the cavity/protrusion area, the area is detected as a single line. This is shown in **Figure 5**(c) and **Figure 5**(d).

D_p in **Figure 5**(c) is the same as that in **Figure 5**(a) while F_p in **Figure 5**(c) is higher than it is in **Figure 5**(a). D_p in **Figure 5**(d) is lower than that in **Figure 5**(b) and F_p in **Figure 5**(d) is the same as that in **Figure 5**(b). Therefore, in general, PRI will be lower if the junction is passed through. However, since these cavity and protrusion areas are much smaller than the whole line areas, their impact on the performance is small. Therefore the effect of the maximal width difference and the maximal tracking step on SPV performance is negligible. Therefore, this parameter

can be set to some value. We set 20 pixels for the maximal tracking step and 2 pixels for the maximal width difference.

3.4 The Maximal Polygonalization Error

The polygonal approximation result is affected by the parameter of the maximal polygonalization error. As the value increases, more points are removed from the polyline. As we can see from **Figure 3**, assuming that the polyline is exactly the same as it is in raster, after more points are removed, D_p is lower than it is originally, while F_p is higher. Therefore, the PRI decreses as the maximal polygonalization error increases. The impact of this parameter is remarkable especially when there are many circular arcs in the image. To preserve the original shape to the most extent, this parameter should be set to a value between 0.5 and 1.0 pixel.

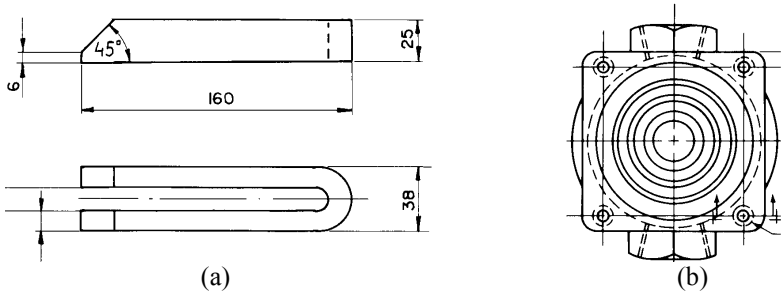

Figure 6. Real life drawing images used in the experiments. (a) horseshoe.tif. (b) ansidash.tif

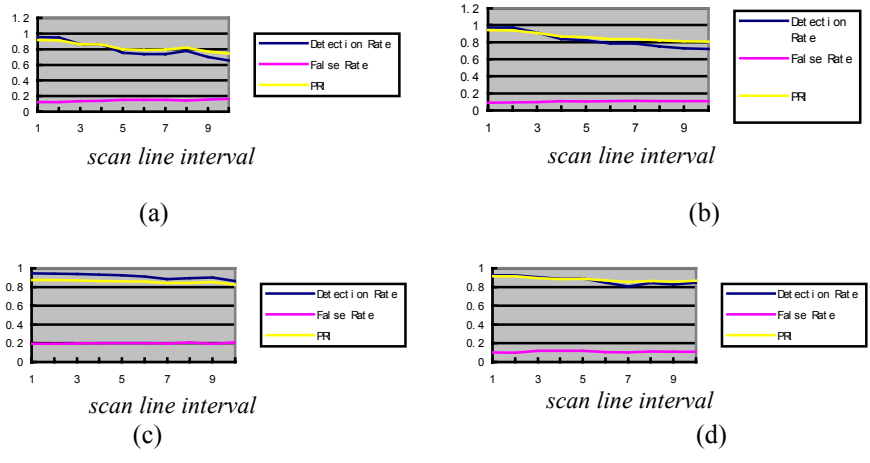

Figure 7. The changes of the pixel performance indices of SPV on four test images as the scan line interval changes from 1 to 10 pixels. (a) ds34.tif (b) ds33.tif (c) ansidash.tif (d) horseshoe.tif

4 Experiments and Discussion

We have evaluated the performance of SPV with different parameter settings using many different kinds of drawings. The experiments on four images are presented below, including two test images (ds34.tif and ds33.tif, which are referred to as contest test images thereafter) downloaded from the web page developed by Chhabra and Phillips (1997). The other two images displayed in **Figure 6** are real life drawing images.

4.1 The Experiments on the Scan Line Interval

We change the value of the scan line interval from 1 to 10 pixels and follow the changes in SPV performance, which are shown in **Figure 7**. The settings of other parameters are as follows. The maximal tracking step is 20, the maximal width difference is 2, and the maximal polygonalization error is 0.7. The performance index values on the four test images are presented in Tables 1-4 and the graphical illustration of the changes of these index values are shown in **Figure 7** (a-d). As we can see from **Figure 7**, the results are in accord with our theoretical analysis of the impact of the scan line interval to the SPV performance presented in Section 3.2. The detection rate and the PRI drop significantly as the value of the scan line interval parameter increases. The false alarm rate fluctuates slightly (the changes of the pixel false rate are less than 0.04 for the two contest images and less than 0.01 for the two real life drawing images). Therefore the scan line interval for best performance is 1. We can see from Table1~4 that the run time decreases dramatically when the scan line interval increases. This empiracally proves the analysis on the impact of the scan line interval to the vectorization speed in Section 3.2.

4.2 Maximal Tracking Step and Maximal Width Difference

We changed the value of the maximal tracking step from 8 to 25 pixels and show the changes of the SPV performance on the two real life drawing images in **Figure 8** and **Figure 9**. The values set for the other parameters are as follows. The scan line interval is set to 1 pixel, the maximal width difference is 2 pixels, and the maximal polygonalization error is 0.7. As we can see from **Figure 8** and **Figure 9**, the performance index values do not change almost at all (the change in **Figure 8** is less than 0.01), in accord with the theoretical analysis in Section 3.3. Experiments on other images also show that the changes of the performance index values are almost non-existent as the maximal tracking step varies, especially when the maximal tracking step is bigger than 20. Therefore the default value of the maximal tracking step in SPV is set to 20 pixels. Next, we change the value of the maximal width difference from 1.0 to 3.5 pixels and present the changes of the SPV performance on the two real life drawing images in **Figure 10** and **Figure 11**. The settings of other parameters are as follows. The scan line interval is set to 1 pixel, the maximal tracking step is 20 pixels, and the maximal polygonalization error is 0.7. As we can see from **Figure 10** and **Figure 11**, the performance index values remain almost constant as the maximal

width difference varies. This is also in accord with the theoretical analysis in Section 3.3. Experiments on other images also show the same conclusion: the impact of changing the maximal width difference on the SPV performance is almost unnoticeable. The default value of the maximal tracking step in SPV is set to 2.0 pixels.

4.3 Maximal Polygonalization Error

We change the value of the maximal polygonalization error from 0.1 to 3.5 pixels and present the changes of the SPV performance on the two real life drawing images in **Figure 12** and **Figure 13**. The settings of other parameters are as follows. The scan line interval is set to 1 pixel, the maximal width difference is 2 pixels, and the maximal width difference is 2.0.

Table 1. Performance indices of SPV on the image of ds34.tif

Scan Line Interval	1	2	3	4	5	6	7	8	9	10
Detection Rate	0.954	0.950	0.863	0.862	0.753	0.737	0.737	0.779	0.699	0.654
False Rate	0.122	0.124	0.136	0.139	0.153	0.153	0.151	0.142	0.155	0.162
PRI	0.916	0.913	0.864	0.861	0.800	0.792	0.793	0.818	0.772	0.746
Run Time (secs)	87.77	31.37	20	15.66	10.1	11.1	8.62	9.66		

Table 2. Performance indices of SPV on the image of ds33.tif

Scan Line Interval	1	2	3	4	5	6	7	8	9	10
Detection Rate	0.972	0.971	0.913	0.836	0.820	0.784	0.784	0.749	0.729	0.721
False Rate	0.091	0.092	0.097	0.107	0.105	0.107	0.112	0.107	0.109	0.110
PRI	0.941	0.939	0.908	0.865	0.857	0.838	0.836	0.821	0.810	0.805
Run Time (secs)	21.03	13.62	9.45	12.74	9.73	8.13	7.58	8.46		

Table 3. Performance indices of SPV on the image of ansidash.tif

Scan Line Interval	1	2	3	4	5	6	7	8	9	10
Detection Rate	0.945	0.942	0.939	0.932	0.926	0.911	0.884	0.894	0.902	0.862
False Rate	0.196	0.194	0.199	0.203	0.203	0.201	0.198	0.207	0.195	0.207
PRI	0.874	0.874	0.870	0.865	0.861	0.855	0.843	0.844	0.854	0.827
Run Time (secs)	1.78	0.99	0.72	0.6	0.49	0.44	0.38	0.33		

Table 4. Performance indices of SPV on the image of horseshoe.tif

Scan Line Interval	1	2	3	4	5	6	7	8	9	10
Detection Rate	0.926	0.925	0.903	0.888	0.891	0.848	0.806	0.842	0.826	0.846
False Rate	0.101	0.098	0.118	0.121	0.119	0.107	0.101	0.113	0.109	0.109
PRI	0.913	0.914	0.893	0.883	0.886	0.871	0.852	0.865	0.858	0.868
Run Time (secs)	1.15	0.61	0.49	0.39	0.33	0.28	0.22	0.27		

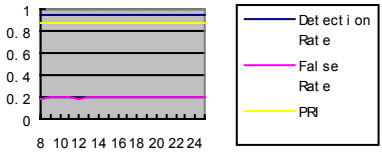

Figure 8. Variations in the pixel performance indices of SPV on ansidash.tif as the maximal tracking step varies from 8 to 25 pixels

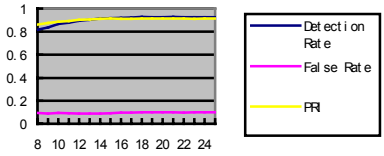

Figure 9. Variations in the pixel performance indices of SPV on horseshoe.tif as the maximal tracking step varies from 8 to 25 pixels.

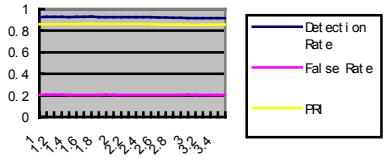

Figure 10. The changes of the pixel performance indices of SPV on ansidash.tif as the maximal width difference varies from 1.0 to 3.5 pixels

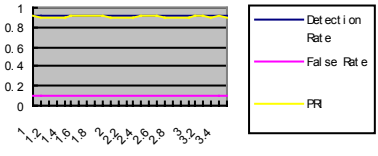

Figure 11. The changes of the pixel performance indices of SPV on horseshoe.tif as the maximal width difference varies from 1.0 to 3.5 pixels

As we can see, the performance index values decrease monotonically as the maximal polygonalization error increases. However, the speed of decreasing in **Figure 12** is bigger than that in **Figure 13**. This is because there are more circular arcs in **Figure 12** than in **Figure 13**, which can be seen from **Figure 6**. This result is also the same as the theoretical analysis in Section 3.4. From **Figure 12** and **Figure 13** we can also see that the changes of the performance index values remain almost unchanged when the maximal polygonalization error is less than 0.7. This is because these values of the parameter are precise enough and cannot affect the SPV performance. Therefore the default value of the maximal polygonalization error in SPV is set to 0.7 to obtain maximum speed of SPV while maintaining its accuracy.

5 Summary

We have presented a theoretical analysis and experimental results of the impact of the SPV algorithm parameters on its performance. Four parameters may affect SPV performance: scan line interval, maximal tracking step, maximal width difference, and maximal polygonalization error. Our experiments confirm the theoretical analysis, and both support the following findings. As the scan line interval increases, the detection rate and PRI drops while the false alarm rate is not affected since some horizontal thin lines may be missed and no more false line can be detected. Therefore the scan line interval for the best performance is 1 pixel. It should be born in mind

that this has a price in terms of processing speed, so if time is a major factor, one may wish to compromise this. The maximal tracking step and the maximal width difference have only little impact on SPV performance. The default settings of these two parameters, found by experience of trial and error, are 20 pixels and 2.0 pixels, respectively. The impact of the maximal polygonalization error is noticeable, especially if there are many circular arcs in the image. As the value of this parameter increases, the SPV performance becomes deteriorates. The default, optimal value for the maximal polygonalization error was found to be 0.7. This value obtains both best vectorization accuracy and processing speed. Our findings support the claim that SPV is a robust algorithm, since these parameter settings are suitable for almost all drawing images. SPV can therefore be used for vectorization of any class of line drawings, including drawings that contain a large amount of text.

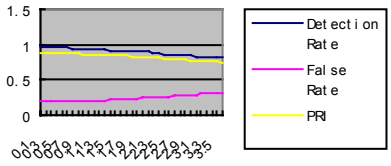

Figure 12. The changes of the pixel performance indices of SPV on ansidash.tif as the maximal polygonalization error varies 0.1-3.5 pixels

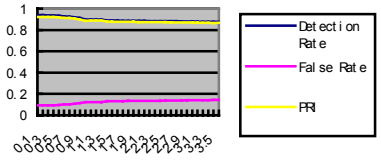

Figure 13. The changes of the pixel performance indices of SPV on horseshoe.tif as the maximal polygonalization error varies 0.1-3.5 pixels

References

1. Chhabra A and Phillips I (1997) Web pages for the Second International Graphics Recognition Contest. http://graphics.basit.com/iapr-tc10/contest.html
2. Dori D, Liang Y, Dowell J, Chai I (1993) Spare Pixel Recognition of Primitives in Engineering Drawings. Machine Vision and Applications, 6:79-82
3. Dori D and Liu W (1999) Sparse Pixel Vectorization: An Algorithm and Its Performance Evaluation. IEEE PAMI. 21(3):202-215.
4. Lin X et al. (1985) Efficient Diagram Understanding with Characteristic Pattern Detection. Computer Vision, Graphics and Image Processing 30:84-106
5. Liu W and Dori D (1997) A Protocol for Performance Evaluation of Line Detection Algorithms. Machine Vision Applications. 9:240-250.