EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling (2008)
C. Alvarado and M.- P. Cani (Editors)

Paper-based Scribble Simplification: Where Do We Stand?

A. Bartolo, K. P. Camilleri, S. G. Fabri[1] and J. C. Borg[2]

[1]Department of Systems and Control Engineering, University of Malta
[2]Department of Industrial and Manufacturing Engineering, University of Malta

Abstract

This paper presents a formal evaluation of the paper-based scribble simplification algorithm described in [BCFB07] and [BCFB08]. A comparative analysis of different aspects of the algorithm with other algorithms described in the literature such as Sparse Pixel Vectorization, spatial moving average filtering and Principal Component Analysis is performed, hence establishing the qualities of this paper-based scribble simplification algorithm. To quantify the performance of the algorithm, performance measures established in the literature, such as the Pixel Recovery Index are used when suitable. However, since there exists no quantitative measure which measures scribble simplification, this paper proposes a new methodology with which scribble simplification may be quantitatively assessed. Through the evaluation described in this paper, we will be able to determine remaining difficulties in the interpretation of paper-based scribbles and hence identify future research areas.

Categories and Subject Descriptors (according to ACM CCS): I.4.3 [Image Processing and Computer Vision]: Grayscale Manipulation C.4 [Performance of Systems]:

1. Introduction

Scribbled drawings such as those shown in Figure 1 are used by designers to make initial, rough representations of a form concept. By scribbling, the designer may represent concepts quickly, without paying undue attention to particular or unnecessary details. This gives the designer the possibility of exploring alternative solutions and therefore increasing the likelihood of innovation and creativity [DG96]. Paper scribbling is still used, particularly in the early-stage conceptual design process, despite the fact that many pen-based computer systems are available on the market. Paper is the medium that is preferred by designers because of its portability and simplicity, allowing a designer to represent his/her ideas without any distractions arising from the medium. Needless to say, interpreting paper-based scribbles is more complex than interpreting on-line scribbles, that is, those scribbles drawn using pen-based computer interfaces, mostly due to the fact that when drawing directly on a computer, it is possible to obtain each stroke as a distinct entity and each stroke is grouped in the order with which it is drawn, allowing the interpretation of the scribble to be performed incrementally. The majority of the sketch-based interfaces described in the literature are intended for on-line scribbles, with very little attention given to the development of algorithms that can interpret paper-based scribbles.

The scribble simplification algorithm described in [BCFB07, BCFB08] acts as a stepping stone in the interpretation of scribbled drawings, allowing paper-based scribbles to be successfully transformed into single-line vectors. In this paper, the qualities of this algorithm are formally evaluated, using established performance measures such as the Pixel Recovery Index (PRI) [LD97]. Furthermore, different aspects of the simplification algorithm are compared with established techniques such as the Hough Transform, Sparse Pixel Vectorization (SPV) [LD99] and moving average filtering in order to assess the advantages of this scribble simplification algorithm over other algorithms. This allows us to identify the challenges that still remain in the interpretation of paper-based scribbled drawings. The rest of the paper is organized as follows, Section 2 gives a brief overview of existing scribble simplification methods, Section 3 describes the simplification algorithm proposed in [BCFB07] and [BCFB08], Section 4 defines scribble roughness and proposes the use of the Roughness Index to measure scribble roughness, Section 5 gives the performance evaluation of the scribble simplification algorithm, Section 6 discusses the remaining issues in the

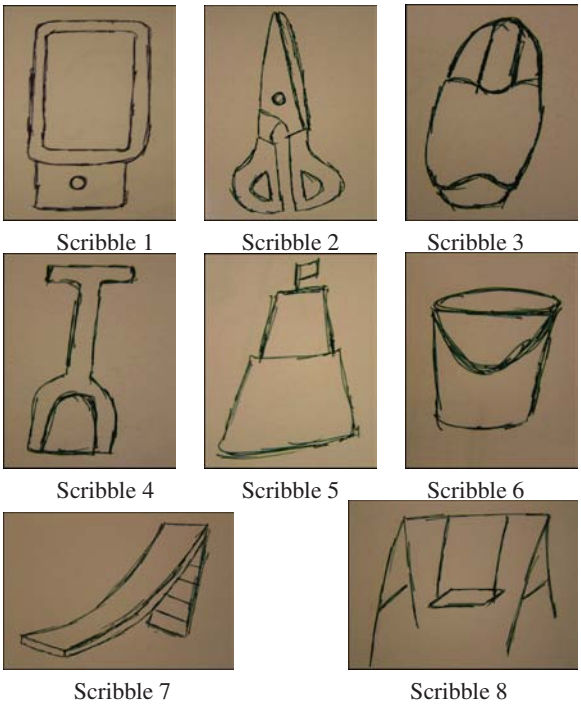

Figure 1: *Examples of the scribbles that benefit from the scribble simplification algorithm described in [BCFB07] and [BCFB08]*

interpretation of paper-based scribbling while Section 7 concludes this paper by discussing future research required to successfully interpret all paper-based scribbles.

2. On-line Scribble Simplification Algorithms

Simplification of on-line scribbled drawings is usually carried out by fitting mathematical models to the stroke segments as soon as the designer completes each segment. Using these methods, each stroke segment may be classified as either a new edge segment or a modification to an existing edge. Distinction between the two types of strokes is carried out either by using proximity and orientation thresholds [KQW06, FR02] or gestural commands that indicate grouping [KS06]. Since these methods interpret the scribbled strokes incrementally, these scribble simplification methods are not suitable for the simplification of paper-based scribbles. Simplification using point spread thinning, such as that described in [SD04] may be adapted for paper-based scribbles. However, this method requires the specification of tolerance regions which makes the amount of grouping possible dependent on the size of these regions. The stroke ordering algorithm described in [PSNW07] can also be, to some extent, adapted to scribbled drawings. This method uses Principal Component Analysis (PCA) to determine window regions in which the strokes have similar directionality and this may easily be adapted for paper-based scribbling. However the grouping of over-strokes within the windows requires that each stroke is represented by splines and this cannot be carried out in paper-based scribbled drawings since the individual strokes are not identified as separate entities.

3. A Paper-based Scribble Simplification Algorithm

The difficulty in interpreting paper-based scribbled drawings is mainly due to the fact that scribbles contain various degrees of roughness. Roughness is perceived when the edges of objects represented in the scribble do not consist of single strokes but are made up of a number of stroke segments separated from each other by gaps. Therefore, there seems to be a distinction between the gaps present in scribbles, namely, gaps which separate strokes that define the same object edge and gaps which separate strokes that define different edges. In this text, we denote as *edge groups* the group of strokes that form an object edge, as *intra-group* gaps those gaps that separate strokes within an edge-group and as *inter-group* gaps those gaps that separate different edge-groups. Scribbles are perceived as having different degrees of roughness since the size of the intra-group and inter-group gaps may change within the scribble such that the human perception of what constitutes an edge-group adjusts according to the different regions within the scribble. Traditional vectorization algorithms fail to distinguish between inter-group and intra-group gaps such that each stroke segment is represented by a line vector. This will result in a large number of vectors which must be re-grouped into their respective edge-groups. This makes the direct application of vectorization algorithms of limited use. Thus, in order to represent paper-based scribbles by vector data, prior processing must be carried out in order to groups the individual strokes into edge-groups.

Since the size of the inter-group and intra-group gaps are expected to vary within the scribble, creating visual patterns that have different frequencies, we chose to adopt pattern recognition algorithms, namely Gabor filtering to obtain the required stroke grouping [BCFB07]. The Gabor filter gives a mathematical model for the action of particular cells in the mammalian visual cortex which are frequency and orientation selective [JF90]. In [BCFB07] we describe a scribble simplification algorithm that uses two filter banks to group strokes that form an edge group into a single line while retaining sufficient distinction between different edge groups, particularly when these are separated by narrow inter-group gaps. Traditional vectorization algorithms may therefore be applied to this simplified, raster scribble. However, the Gabor grouping algorithm described in [BCFB07] provides additional information about the stroke groups in the form of quantized orientation estimates for each pixel in the edge group. For this reason, we chose to define a new path tracking algorithm that utilizes these orientation estimates to track the edge-groups in a piece-wise linear manner, using the edge boundaries to adjust the tracking in compensation for the quantization error of the orientation estimates as de-

scribed in [BCFB08]. However, since the edge-group boundaries may be of non-uniform thickness, the path tracking may deviate from the true medial paths. For this reason, a Kalman filtering post-processing step is introduced to reduce the effect of this noise [BCFB08].

The result of this processing is a sequence of points that lie on the medial axis and which may be joined in a piecewise linear manner, representing the scribble by line vector data. These data points may be used directly by CAD systems or other sketch-based interfaces such as [KS06, PSNW07, FR02] among others, to create 3D models from the paper-based scribble.

4. Measuring the Performance of the Scribble Simplification Algorithm

The purpose of the scribble simplification algorithm is to represent the users' intended shape by vector data. The performance of the scribble simplification may therefore be measured by determining how close the resulting vectors are to the intended object shape. The scribble simplification algorithm described in [BCFB07] and [BCFB08] consists of three steps, namely stroke groping, path tracking and Kalman smoothing such that in order to determine the performance of the scribble simplification algorithm, it is necessary to determine the performance of each step.

4.1. The stroke grouping step

The role of the stroke grouping step is to reduce the number of stroke segments and hence the number of grouping combinations that are obtained from scribble such that the algorithm will be considered effective only if the number of stroke segments are sufficiently reduced. This may be observed by comparing the number of stroke grouping combinations perceptual grouping algorithms such as that used in ScanScribe [SFLM02] would generate for the scribble and for the simplified drawing. The perceptual grouping algorithms used in ScanScribe are not well adapted for scribbles such that a large number of grouping combinations are expected from scribbles. These combinations should be reduced considerably if the stroke grouping algorithm is effective in grouping the individual stroke segments into stroke groups.

Measuring the extent of the stroke grouping algorithm from the number of segments alone is however insufficient as this does not determine whether the stroke groups actually correspond to the perceived stroke groups and hence reduce the scribble roughness. There is however no existing protocol that determines the roughness of a scribble such that it is necessary to formulate a measure of scribble roughness. In order to measure roughness, it will be necessary to compare the scribble to some ground truth. Unlike traditional vectorization algorithms, the ground truth drawing cannot be created first as this would restrict the users' drawing freedom by mentally pre-conditioning the user to draw strokes within the constraints of the ground truth image. For this reason, the user was asked to first draw the scribble, digitize it and then indicate the perceived or intended shape by digitally drawing over the digitized scribble, using the paint brush option in Paint Shop Pro®, changing the thickness of the paint brush to reflect the desired stroke width. The resulting digital strokes were subsequently used as the perceived ground truth drawing. The roughness of the scribble may be obtained by measuring the difference between the scribble and the ground truth drawing. This difference will be due to two main factors, namely the portion of the scribble that does not have matching ground truth lines, which we denote as N_g and the portion of the ground truth lines which do not have corresponding scribble strokes, which we denote as N_o. The difference between the two drawings may therefore be expressed as $\mathrm{RI} = N_g + N_o$

This roughness index may have a minimum value of $RI = 0$, obtained when the scribble is a perfect match of the ground truth and a maximum value of $RI = 2$ which is obtained when none of the scribble strokes match the ground truth lines. However, since the perceived ground truth is created by drawing the intended shape over the scribbled drawing, it is unlikely that none of the scribble strokes match the ground truth lines such that the roughness index is biased towards lower values of RI. To compensate for this bias, the Roughness Index may be redefined as $\mathrm{RI} = \log(4.5(N_g + N_o) + 1)$ such that the value of RI will vary between 0 and 1 and have a larger spread in the lower range of values for $N_g + N_o$. One may note that N_g gives a measure of the intra-group gaps present in the drawing, while N_o gives a measure of the number of scribble stroke segments that overshoot the intended line strokes such that expressing the difference between the scribble and the ground truth drawing in this manner allows us to determine the relative impact of the intra-group gaps and the overshooting segments on the perceived scribble roughness.

This roughness index value may be used to determine the performance of the stroke grouping algorithm by determining the reduction in the roughness achieved by the algorithm. Using this index it is also possible to compare the performance of the stroke grouping algorithm proposed in [BCFB07] with morphology operations which may be used to enhance images by filling in gaps in the image foreground due to noise [GW00].

4.2. The path tracking step

The line tracking algorithm described in [BCFB08] introduces the concepts of local saliency measures in making tracking decisions at junction regions. Using this local saliency measure, the path tracking algorithm may select paths that are perceived as being more salient than others, given the current tracking direction. It is therefore necessary to determine whether the local saliency measure is sufficient

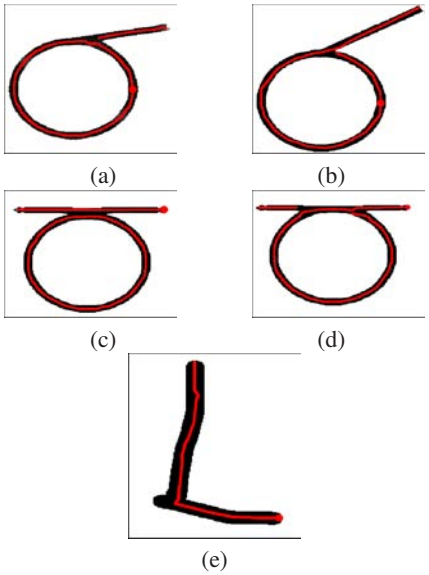

Figure 2: *Test segments used for qualitative analysis of the path tracking algorithm. Red curves indicate the tracking paths obtained by the tracking algorithm*

to allow for the selection of perceptually salient paths. To achieve this, several test strokes were digitally created to simulate different intersecting angles, including acute angles and tangents to circular arcs as shown in Figure 2. These test images were used to qualitatively determine the performance of the path tracking algorithm. By manually initializing the tracking algorithm from different starting points, indicated by the red spots in Figure 2, it is possible to observe the tracking decisions carried out by the algorithm when approaching junction regions from different directions.

4.3. The Kalman smoothing step

The Kalman smoothing step is required to reduce the effect of boundary noise on the medial points extracted by the path tracking algorithm when this is applied to the result of the stroke-grouping step. Since the action of this Kalman smoothing step is comparable to the spatial, moving average filter, it is necessary to compare the two filters in order to determine the advantages of the Kalman filter over the moving average filter. To do this, test images such as those shown in Figure 3 were created using the spline drawing tool provided by Paint Shop Pro®. These test images were created such that they consist of free-form curves, rigid shapes, sharp corners and tangential intersections, hence representing various geometric possibilities in real-world scribbles. The medial points of these images were extracted and since the test images do not have noisy boundaries, these medial paths may be used as ground truth data. Boundary noise was simulated by adding random positional noise having a uniform distribution whose range was varied between $[-5,5]$ and $[-20,20]$ to the ground truth paths hence displacing the

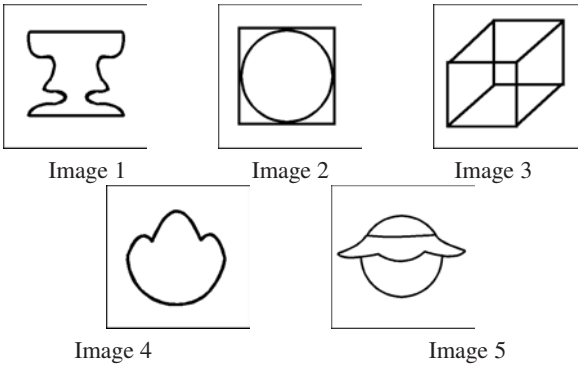

Figure 3: *Test images consisting of straight lines, curves, tangent junctions, rigid shapes and free-form shapes.*

medial points. The noisy paths were then smoothed using the proposed Kalman smoothing and the moving average filter, measuring the distance between the smoothed paths and the ground truth paths to compare the performance of the two filters.

4.4. The vectorization algorithm

The path tracking step and the Kalman smoothing step may be compared to the traditional vectorization algorithms such as the Sparse Pixel Vectorization (SPV) algorithm [LD99], using the Pixel Recovery Index (PRI) [LD97] as a performance measure. The PRI measures the quality of the extracted line paths by comparing them with the corresponding ground truth paths and is defined by Equation 1,

$$PRI = \gamma D_p + (1-\gamma)(1-F_p) \quad (1)$$

where D_p is the Pixel Detection Rate and F_p the Pixel False Alarm Rate and γ is the relative importance of the detection rate. In order to determine the benefits of the proposed vectorization algorithm over the SPV algorithm, the vectorization step was carried out on the simplified scribbles, using the indented ground truth obtained for these scribbles to obtain the respective PRI values.

However, the PRI does not capture the difference between continuous tracking and fragmented tracking. This has an effect on subsequent path beautification steps since it changes the number of path segmentations and path merges required by these algorithms in order to make the resulting vectors similar to the perceived strokes. In order to compare the beautification that would be required by the proposed line tracking algorithm and the SPV algorithm, manual segmentation and merges where applied to the line paths obtained by the two algorithms for each of the test images shown in Figure 3, until the resulting path segments correspond to the spline segments that were used to create the test images. This

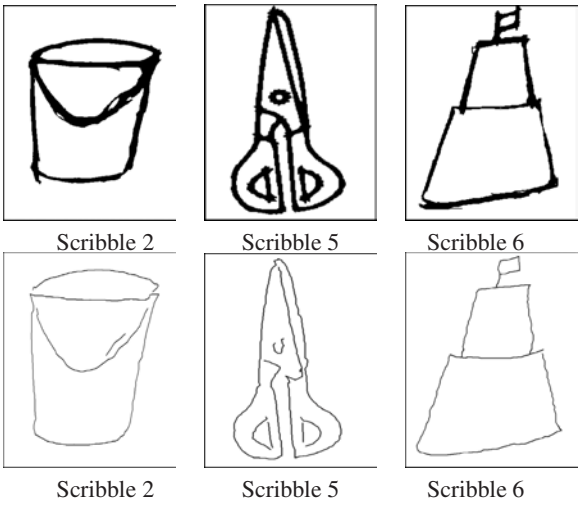

Figure 4: *A sample of the results obtained by the paper-based scribble simplification algorithm. The top row gives the result of the stroke grouping step and the bottom row the result of the path tracking and Kalman smoothing steps.*

will give an indication on the degree of fragmentation obtained by the proposed vectorization for different geometric shapes.

4.5. The paper-based scribble simplification

Besides assessing the performance of the individual steps in the proposed paper-based simplification algorithm, it is also necessary to determine the performance of the entire algorithm. This may be done by comparing the proposed scribble simplification with the simplification described in the online stroke grouping algorithm described in [PSNW07]. This algorithm may be adapted to paper-based scribbling by performing the Principal Component Analysis (PCA) on the binarised representation of the scribble, using the co-ordinates of all stroke pixels within the selected windows to form the required co-variance matrix. The eigen-vector corresponding to the largest eigen-value gives the main direction of the strokes and this direction is valid as long as there is a considerable difference between the two eigen-values determined by the PCA. Thus, scribbles such as those shown in Figure 1, were manually segmented into window regions consisting of approximately linear segments. This ensures that the principal component gives a suitable representation of the strokes within the window region and may therefore be used to replace the scribbled strokes by a single line vector. The PRI was then used to determine the error between this vector and the intended line strokes, comparing the result with that achieved by the proposed algorithms.

5. Results and Evaluation

This section presents the results obtained by the paper-based scribble simplification algorithm. As shown in Figure 4, the results obtained are a fair representation of the designer's intent indicating that the proposed algorithms are suitable in grouping the individual strokes into edge-groups from which piece-wise linear vectors have been accurately extracted. The results given in the following sections give numerical support to these visual results.

Scribble ID	$N_{GroundTruth}$	$N_{Scribble}$	N_{Gabor}
1	15	44	17
2	18	53	18
3	13	44	18
4	14	44	15
5	15	64	16
6	9	50	13
7	22	62	24
8	23	79	31

Table 1: *Listing the number of segments obtained by ScanScribe for the ground truth drawing ($N_{GroundTruth}$), the scribble ($N_{Scribble}$) and the simplified scribble (N_{Gabor}).*

5.1. The stroke grouping step

Table 1 lists the number of segments obtained by the ScanScribe software for the scribbles, the Gabor simplified scribbles and the perceived ground truth representations of the scribbles. As expected, the perceptual grouping algorithms used in ScanScribe give a large number of stroke combinations for the scribbled drawing. These combinations are significantly reduced when the ScanScribe software is used to segment the results of the stroke grouping algorithm. In fact the proposed Gabor grouping algorithm reduces the number of segment combinations by an average of 65% such that the number of segments obtained from the simplified scribble are similar to the number of segments obtained from the perceived ground truth representation of the scribbles. Furthermore, Table 2 shows that the proposed stroke grouping algorithm reduces the scribble roughness by an average of 79% in comparison to the 3% reduction in scribble roughness obtained by the moving average filter, hence indicating the effectiveness of the proposed Gabor grouping algorithm.

Moreover, morphology operations require the specification of a structuring element whose shape and size will be kept constant for all the image. Thus, unlike the Gabor filter which may close gaps of different widths, the morphology operation can only close gaps that are equal to or smaller than the size of the structuring element. Selection of small structuring elements will leave large gaps open whereas selection of large structuring elements would result in closure of inter-group gaps. Note that in order to obtain a fair comparison, the 'close' operation was performed using disk structuring elements of different radii and the results listed in Table 2 give the best RI values obtained together with the corresponding element radius.

Scribble ID	$RI_{Scribble}$	RI_{Gabor}	RI_{close} (disk radius)
1	0.67	0.07	0.64 (7)
2	0.66	0.19	0.63 (5)
3	0.58	0.04	0.55 (3)
4	0.65	0.19	0.59 (7)
5	0.70	0.10	0.63 (3)
6	0.74	0.16	0.65 (7)
7	0.68	0.21	0.62 (11)
8	0.70	0.16	0.62 (11)
μ	0.67	0.14	0.65
σ	0.04	0.06	0.10

Table 2: *Comparison of the Roughness Index for the scribbles shown in Figure 1. The last two rows give the mean μ and standard deviation σ for each column.*

5.2. The path tracking step

Analysis of tracking at different intersection angles has shown that the line tracking algorithm selects the path of smoothest continuation provided that the intersecting angle is smaller than or equal to the orientation resolution of the Gabor filter scheme. If the intersection angle is smaller than the orientation resolution of the filter bank, the Gabor grouping algorithm selects only one dominant orientation for the junction region, and this is not necessarily smoothly continuous with the tracking direction.

Line tracking through intersections formed by arcs and lines is dependent on the initial tracking direction as well as the angle of intersection. This happens because unlike the tracking of straight lines, curved paths require a change in direction at each tracking step. The initial tracking direction, which determines whether the curved path is traced in a clockwise or anti-clockwise direction will therefore change the approach to the junction region such that smooth continuation does not always guarantee that line tracking will proceed on the curved path. This may be observed in Figure 2 (a,b).

Straight line segments that intersect tangentially with circular arcs are also tracked differently, depending on the length of the junction as shown in Figure 2 (c,d). Longer junction regions allow the tracking algorithm to adjust the tracking such that the medial points determined by the tracking correspond to the midpoints of the junction region rather than the midpoints of the individual paths. Consequently, the tracking does not continue tracking along the circular path which may be the path that is perceived as having higher saliency.

The perceptual selection of the tracking direction is also useful when the intersecting lines form spurs as shown in Figure 2 (e). By tracking ahead in each of the orientations at the junction region, it is possible to determine when path segments from short spurs, allowing the tracking algorithm to change the tracking direction to follow the more salient line path.

Image ID	E_{input}	E_{KF}	E_{KS}	E_{MA} (W_{MA})
1	3.64	2.74	2.07	2.23 (5)
2	3.77	3.38	2.08	2.10 (5)
3	3.95	2.73	2.40	2.31 (7)
4	4.05	3.42	2.10	2.50 (5)
5	3.91	2.59	1.59	1.97 (3)

Table 3: *Comparing the performance of the Kalman filter to the moving average filter. Input and output errors are given in displacement in pixel-size per tracking point. E_{KF} refers to the error obtained by the Kalman filter forward estimation, E_{KS} is the error obtained by the Kalman smoothing, E_{MA} the error obtained by the moving average filter and W_{MA} is the window size of the moving average filter.*

5.3. The Kalman smoothing step

Table 3 lists the difference between the smoothed paths and the ground truth lines for all test images, subject to a uniform noise with range $[-5,5]$, giving the error in pixel-size displacement per track point. This comparison shows that the Kalman filtering results in a larger reduction in the path noise, resulting in smoother and better placed line paths than the moving average filter. Table 3 also compares the error obtained by the forward estimation and the smoothing steps of the proposed Kalman filtering. From this, one may be note that besides eliminating the causal nature of the Kalman filter, the backward estimations reduce further the noise in the medial line paths. Furthermore, Table 3 highlights the fact that the moving average filter requires different window sizes for the different test images in order to obtain an optimal smoothing that gives the lowest error between the smoothed path and the ground truth path. Thus, moving average filtering requires the selection of a suitable window size that balances the smoothing effect with the loss in path detail. In contrast, the Kalman filtering uses the noise covariance to perform smoothing and hence performs equally well for straight lines and curved paths.

5.4. The vectorization algorithm

Table 4 shows that the proposed line tracking and Kalman smoothing vectorization algorithm improves the pixel detection rate while retaining a similar false detection rate as the SPV algorithm, hence an overall improvement in the Pixel Recovery Index. Furthermore, the proposed line tracking algorithm obtains the medial paths in an average of 18 seconds for a simplified scribble in which 10% of the line strokes are part of the image foreground. This contrasts with the average time of 32 seconds required by the SPV algorithm for a similar line drawing. Hence, the proposed line tracking algorithm can reduce average computational time required to determine the medial line paths by 56.14%, obtaining line paths of better quality while reducing the computational time required to obtain the paths.

Furthermore, Table 5 shows that the proposed line track-

Scribble Id	SPV			Kalman Tracking		
	Dp	Fp	**PRI**	Dp	Fp	**PRI**
1	0.95	0.43	**0.84**	0.99	0.43	**0.87**
2	0.94	0.28	**0.87**	0.98	0.28	**0.90**
3	0.94	0.44	**0.82**	0.99	0.43	**0.86**
4	0.77	0.11	**0.81**	0.93	0.11	**0.92**
5	0.77	0.24	**0.76**	0.92	0.23	**0.88**
6	0.88	0.40	**0.80**	0.97	0.38	**0.87**
7	0.84	0.24	**0.82**	0.93	0.23	**0.88**
8	0.89	0.33	**0.82**	0.91	0.33	**0.84**
μ	0.88	0.32	**0.82**	0.95	0.32	**0.87**
σ	0.06	0.09	**0.03**	0.03	0.09	**0.02**

Table 4: *Comparison of the PRI obtained by the proposed line tracking algorithm and SPV.*

Image Id	SPV			Kalman Tracking		
	N_S	N_M	$\mathbf{N_{Total}}$	N_S	N_M	$\mathbf{N_{Total}}$
1	0	10	**10**	6	1	**7**
2	2	11	**13**	3	5	**8**
3	2	3	**5**	5	2	**7**
4	0	10	**10**	4	0	**4**
5	3	13	**16**	8	2	**10**

Table 5: *Compares the computational task in terms of number of merges and segmentations required by subsequent beautification algorithms*

ing algorithm requires fewer path adjustments than the SPV algorithm for 4 out of the 5 test images shown. This happens because the proposed line tracking adjusts the tracking direction to reflect the path curvature while the path tracking of the SPV algorithm is performed using either a horizontal or vertical path search. Although this would require fewer path adjustments in images dominated by straight lines, such a tracking would segment curve strokes. This is evident from the fact that for these test images, the SPV algorithm requires more path merges than path segmentations. In contrast, the proposed line tracking algorithm requires more path segmentations than merges. Since the proposed tracking algorithm retains the line orientations at each track point, it is possible to perform the required segmentations using these orientations such that the required segmentations should not cause a considerable increase in the computational load of the subsequent beautification step.

5.5. The paper-based scribble simplification

Table 6 compares the PRI values obtained for six window regions obtained from the scribbles shown in Figure 1. This comparison shows that the proposed line representation is more accurate than the principal component vector. Note that for all segments represented in Table 6, the principal eigenvalue was at least 40 times larger than the minor eigen-value which is larger than the threshold selected in [PSNW07], indicating that the window was adequately chosen and no further sub-divisions of the window region were necessary.

Segment Id	PCA			Kalman Tracking		
	Dp	Fp	**PRI**	Dp	Fp	**PRI**
1	0.18	0.75	**0.20**	0.93	0.56	**0.78**
2	0.25	0.64	**0.29**	0.87	0.59	**0.74**
3	0.38	0.59	**0.39**	0.94	0.56	**0.79**
4	0.57	0.42	**0.57**	1.00	0.57	**0.83**
5	0.53	0.42	**0.53**	0.96	0.59	**0.80**
6	0.17	0.82	**0.17**	0.94	0.59	**0.78**

Table 6: *Comparison of the PRI obtained by the proposed line tracking algorithm and the principal axis determined by the PCA.*

6. Discussion

The scribble simplification algorithm described in [BCFB07] and [BCFB08] introduces a feasible solution in the interpretation of paper-based scribbles. The evaluation performed highlights the advantages of this algorithm over other, generally on-line, scribble grouping techniques. This evaluation serves to indicate the limitations of existing paper-based scribble simplification techniques and hence the areas which require further research.

One requires for instance, a method with which the Gabor filter scheme may adapt itself to the different scribble resolutions that may exist concurrently in the scribbled drawing. One possible method for doing this is to cluster the different frequency bands of the scribble, hence determining the different spectral regions related to the scribble over-strokes that exist in the scribble. This may potentially provide a method to reduce the number of filters in the filter bank, creating a dynamic or adaptive filter bank rather than a fixed bank. This would in turn reduce the computational time required to group the scribbled strokes.

Another limitation of this simplification algorithm lies in the selection of salient paths at junction regions. The proposed simplification uses a preliminary tracking step in each direction in order to determine the most salient path. This however, is a local measure that has two main flaws. The first is that lines that are prolonged for more than one tracking step, but are still perceived as spurious lines, will not be identified as spurs, such that, if these are smoothly continuous with the current tracking direction, they will be selected over other paths that are more likely to form closed contours. Furthermore, due to the nature of the tracking itself, stroke segments that intersect tangentially, forming large junction regions, cannot be processed as separate lines as has been shown in Figure 2. Instead, the tracking algorithm, by taking the midpoint of the line boundaries, will retain a medial path made up of a set of points that belong to neither stroke. This is a difficulty encountered in paper-based vectorization

algorithms and which is often solved by applying a line fitting step, redefining the lines as necessary. Unlike the on-line counterpart, this line fitting step will be required to segment or re-group and re-use parts of the line segments since the paper-based line strokes will not be available as distinct entities as drawn by the designer.

The fact that in the interpretation of paper-based scribbling the line segments are not extracted as entire entities is a contributing factor to the perceptual selection difficulty mentioned earlier. Since the line strokes are progressively being discovered, the stroke saliency may only be measured on a local basis while tracking. In order to solve this problem, a mechanism that allows the estimation of a global saliency measure, while progressively tracking the line paths must be established such that the tracking decisions are performed according to the global perception of the line strokes.

7. Conclusion

In this paper, the paper-based scribble simplification algorithm described in [BCFB07] and [BCFB08] has been evaluated, comparing the results obtained by this algorithm with the results obtained by other related algorithms from the literature. This evaluation shows that the simplification algorithm reduces the complexity of scribbled drawings, enabling the conversion of the paper-based scribbles to vector data which may be exported directly to CAD applications. The evaluation shows that the quality of the vector data obtained from the scribble is similar to that obtained by established vectorization techniques such as SPV, allowing designers to obtain vector representations that would have been achieved from neater drawn drawings. This provides an essential, but often missing, link between the paper medium and sketch-based interfaces or CAD systems, allowing designers to use on-line sketch-based interfaces such as [ZNA07] and [FR02] among others, to exploit the easy deformations of virtual 3D prototypes while retaining the paper-based scribble which is the designers' preferred drawing medium in the initial conceptual design stages.

Acknowledgments

This research is funded by the University of Malta under the research grant IED 73-529 and is part of the project *Innovative 'Early Design' Product Prototyping (InPro)*

References

[BCFB07] BARTOLO A., CAMILLERI K. P., FABRI S. G., BORG J. C.: Scribbles to Vectors: Preparation of Scribble Drawings for CAD Interpretation. In *Proceedings of the Eurographics Workshop on Sketch-Based Interfaces and Modeling* (2007).

[BCFB08] BARTOLO A., CAMILLERI K. P., FABRI S. G., BORG J. C.: Line Tracking Algorithm for Scribbled. In *Proceedings of the International Symposium on Communications, Control and Signal Processing* (2008).

[DG96] DO E. Y., GROSS M. D.: Drawing as a Means to Design Reasoning. In *Artificial Intelligence in Design (AID) Ś96 Workshop on Visual Representation, Reasoning and Interaction in Design* (Palo Alto CA, 1996).

[FR02] FIORE F. D., REETH F. V.: A Multi-Level Skethcing Tool for "Pencil-and-Paper" Animation. In *AAAI Spring Symposium on Sketch Understanding* (2002), pp. 32 – 36.

[GW00] GONZALEZ E., WOODS P.: *'Digital Image Processing'*. Prentice Hall, 2000.

[JF90] JAIN A., FARROKHNIA F.: Unsupervised Texture Segmentation Using Gabor Filters. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics* (1990).

[KQW06] KU D. C., QIN S. F., WRIGHT D. K.: Interpretation of Overtracing Freehand Sketching for Geometric Shapes. In *Proceedings of the 14th International Conference on Computer Graphics, Visualization and Computer Vision* (January 2006), UNION Agency - Science Press, pp. 263–270.

[KS06] KARA L. B., SHIMADA K.: Sketch Based Design of 3D Geometry. In *Eurographics Workshop on Sketch Based Interfaces* (September 2006), Jorge J., Stahovich T., (Eds.), pp. 59–68.

[LD97] LIU W., DORI D.: A protocol for Performance Evaluation of Line Detection Algorithms. *Machine Vision Applications 9* (1997), 240 – 250.

[LD99] LIU W., DORI D.: Sparse Pixel Vectorisation: An Algorithm and Its Performance Evaluation. *IEEE Transactions of Pattern Analysis and Machine Intelligence 21*, 3 (1999), 202–215.

[PSNW07] PUSCH R., SAMAVATI F., NASRI A., WYVILL B.: Improving the Sketch-Based Interface, Forming Curves from Many Small Strokes. *The Visual Computer 23*, 9 (2007), 955–962.

[SD04] SEZGIN T. M., DAVIS R.: Handling Overtraced Strokes in Hand-Drawn Sketches. *Making Pen-Based Interaction Intelligent and Natural* (2004), 1–2.

[SFLM02] SAUND E., FLEET D., LARNER D., MAHONEY J.: Perceptually Supported Image Editing of Text and Graphics. In *AAAI Spring Symposium on Sketch Understanding* (2002), pp. 118–125.

[ZNA07] ZIMMERMANN J., NEALEN A., ALEXA M.: SilSketch: Automated Sketch-Based Editing of Surface Meshes. In *Eurographics Workshop on Sketch Based Interfaces* (August 2007).