Smoothing Via Iterative Averaging (SIA) A Basic Technique for Line Smoothing

Mohsen Mansouryar and Amin Hedayati, *Member, IACSIT*

***Abstract*—Line Smoothing is the process of curving lines to make them look smoother, better to say it is the representation of a polyline so that fewer points represent the caricature of the line. It also usually reduces the noise in a signal. This can apply to a vector, spline, or a list of points corresponding to a line or signal. Many algorithms are available for automated line smoothing that is commonly seen as a comparatively simple operation; however, instructions for using these algorithms are often complex. In this paper, we present a new method as a basic technique that can efficiently smooth a list of points. We focus on preserving characteristics of the line while avoiding any distortions. Our goal is to demonstrate a flexible method to preserve features of the input based on its characteristics with fewer constants. Since this technique can apply to both vectors and lists of points, it is also useful in map generalization. Selected test examples are illustrated and discussed, followed by an assessment of the models. Finally, results of the proposed method are examined, showing more stable preservation and better noise reduction compared to the available methods reported in the literature.**

***Index Terms*—Averaging, bezier curves, noise reduction, line generalization, line simplification, line smoothing**

I. Introduction

Cartographers produce manuscript lines that have a smooth 'flowing' appearance. In comparison, digital lines tend to be angular and non-aesthetically pleasing (particularly at large scales) - this is mostly due to the constraints of the digitizing grid. Unlike simplification [11], which endeavors to reduce detail, smoothing techniques [14] shift the position of points making up a line, in order to remove small perturbations and capture only the most significant trends of the line.

Line smoothing is a well-known theme in automated generalization [7], [10]. It is also a pre-processing part of most online OCR systems which is used for the purpose of noise reduction [12], [13]. There are various methods with different approaches such as McMaster's Distance Weighting Algorithm [4], Boyle's Forward-Looking algorithm and Chaiken's smoothing algorithm.

Our paper introduces a new line-smoothing algorithm, which is categorized as an averaging method. We claim our method have simple calculations while it has efficient results and less constant parameters, thus it has an effective speed while applying to vectors or lists of points. It does not need any segmentation in the line object since it considers all points.

The second section of this paper deals with an introductory to line smoothing routines and their properties. only a short description on other methods followed by Tab. 1, describing three smoothing routines and their properties is provided.

In the third section, we present the basic SIA model followed by some figures describing results for different parameter values. Mathematical formulas for the basic SIA model are also presented in subsection *B*. Subsections *C* and *D*, discuss parameters of the basic SIA model. Subsection *E* presents a helpful principle in setting appropriate values for parameters of the basic SIA model.

The fourth section of this paper discusses the weaknesses of the basic model. Subsection *A* presents an improvement which enables the algorithm to preserve characteristics of the original line object to a specific extent. In subsection *B*, results of the combined algorithms are presented and preservation of the improvement algorithm is discussed and compared to the basic model.

Finally, section V covers a brief comparison between improved SIA model and smoothing via Bezier curves based on their noise reduction and preservation capabilities. no detailed comparison between performance or efficiency of the two methods is made.

The algorithms of this method have been implemented and tested using Microsoft Visual Basic 8.

II. Variety of Line Smoothing Methods

In this section, we discuss the variety of line smoothing algorithms followed by an explanation of how SIA is categorized as an averaging method.

Three trends are stressed for line smoothing. We can cite smoothing methods based on averaging, convolution, or neighboring points. Averaging techniques [1, 4] smooth small details while preserving general shape. Algorithms considering convolution are more regular [2, 3]. They are used for smoothing details that have the same size. At the opposite, smoothing methods based on neighboring points [5, 6] have little influence on lines which are defined with a high density of points. There are various ways that smoothing algorithms can be classified, see Table I below for one such classification.

III. Basic SIA Model

The following section, we are about to present SIA as a basic technique for smoothing of curves represented as lists of points.

Manuscript received April 8, 2012; revised May 14, 2012.

The authors are with the Department of Mathematical Sciences at Sharif University of Technology, Tehran, Iran (e-mail: mansouryar@cs.sharif.edu, ahedayati@acm.org).

TABLE I :CHARACTERISTICS OF SMOOTHING ALGORITHMS (LEWIS 1990 IN MCMASTER & SHEA 1992)

Category	Description
Point averaging routines	a) Calculates an average based on the positions of existing coordinate pairs and neighbours. b) Only the end points remain the same. c) Maintains the same number of points as the original line. d) Each algorithm easily adapted for different smoothing conditions by choosing different tolerance parameters. e) All algorithms are local or extended local processors.
Mathematical curve fitting routines	a) Develops a mathematical function or series of functions that describe the geometrical nature of the line. b) The number of retained points is variable and user-controlled. c) Retention of end points and of points on the original line is dependent on choice of algorithm and tolerances. d) Once algorithm chosen, there is little flexibility allowed in changing the final shape of the smoothed line. e) Function parameters can be stored and used to later regenerate the line at the required point density. f) There are local, extended local and global processing routines.
Tolerancing routines	a) Each algorithm uses some geometrical relationship between the points and a user defined tolerance to smooth the cartographic line. b) End points are retained, but the number of points generated for the smoothed line and the number of interior points retained from the original data is algorithm-dependent. c) Ability to change the curve's final appearance is algorithm-dependent. d) There are local, extended local and global processing routines.

A. Input and Output Parameters

As mentioned before, the first input to this method that represents a line object, is a list of points which are relatively sorted. the second input is an integer, which we call *SS* i.e. an abbreviation for Smoothing Sensitivity. *SS* merely defines the level of smoothing which is proportional to its value. we will discuss how *SS* affects smoothing result in detail later in subsection *C*.

The other input that causes major differences is the iteration of the method. From now on, we call the iteration times *SI i.e*. an abbreviation for Smoothing Iteration.

B. The SIA Model

For line smoothing, we assume *PTS* as a list containing all points of a line object (assuming they are relatively sorted). We can cite many triangles in which *PTS(i)* is one of their vertexes and two other vertexes are from different sides of *PTS(i)*. When these two vertexes are *PTS(i-SS)* and *PTS(i+SS)* respectively, we call this triangle *TR*. Then we change the coordinates of *PTS(i)* to the coordinates of *TR*'s centroid. This is similar to a 3-point unweighted sliding-average smooth. This operation is formulated in

$$X_i = \frac{X_{i-SS} + X_i + X_{i+SS}}{3} \; ; \; Y_i = \frac{Y_{i-SS} + Y_i + Y_{i+SS}}{3} \qquad (1)$$

where:

$$SS = \begin{cases} i & ; \quad i < \alpha \\ \alpha & ; \quad \alpha \le i \le Ub \\ \left|PTS\right| - i \; ; & \quad i > Ub \end{cases}$$

$$Ub = \left|PTS\right| - SS \qquad (2)$$

α Is the user-defined value for SS

Therefore, SS indicates index of *TR*'s vertices from *PTS* list.

In this operation, *SS* is a constant integer, but in two occurrences, this value is obtained from (2). This is due to avoid any distortions in the line object.

By fixing the parameter *SS* = 1 and assuming *PTS* as the list of points corresponding to a vector's nodes, this method can also apply to vectors.

C. Smoothing Sensitivity

So far, we discoursed what *SS* is all about and what exactly it does in the smoothing process in subsections *A* and *B*. Here we are about to add a little more on how it affects the smoothing result.

In SIA and generally averaging methods of line smoothing, each point's coordinates will change in compare to its neighboring points since curvature – a line feature that is independent of the coordinate system – is common for a point and its neighboring points i.e. it changes slightly in the boundary points of each curve.

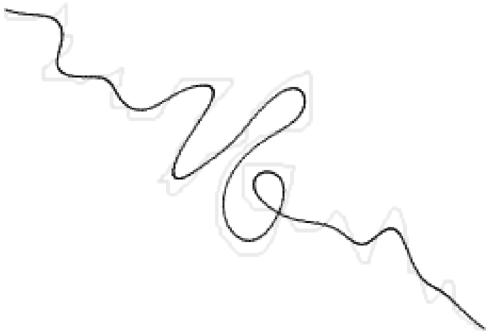


Fig. 1a. line object represented as a list of points, smoothed by basic SIA model. (*SS*=5, *SI*=10)

In our method, *SS* indicates two neighbors from both sides of the current point in process (in the same range). By setting a high value for *SS*, these points may get selected from different curves therefore we cannot call them neighboring points anymore. This can lead to distortions in smoothing result.

Consequently, it is obvious that setting appropriate values for *SS* directly depends on curvature but generally, we do not recommend high values for *SS*.

D. Smoothing Iteration

Iteration is an important parameter in SIA model since it leads to different levels of smoothing which totally affects the smoothing result. In other words, SIA needs to be applied more than once until obtaining a specific degree of smoothing and preservation as well.

E. Setting Appropriate Parameter Values

Setting appropriate values for *SS* and *SI* can be done based on experimental results; yet this is not efficient for automated line smoothing.

Our experimental results have shown that using (3) as a guiding principle can help to avoid any distortions and disorders that may happen in smoothing result.

$$\frac{SS}{SI} \leq 1 \rightarrow SS \leq SI \tag{3}$$

Although higher *SS* or *SI* will give a stronger degree of smoothing, it is strongly recommended that these values fit in (3). (See Fig. 1 and 2)

Note that some value pairs may not be acceptable under (3) while they are also fine values not leading to any distortions. This happens since *SS* is independent from line features; therefore it does not always lead to distortions in fewer iterations.

IV. Improved SIA Model

The evaluation of tests and observations for the basic SIA model reveal some weaknesses, which are discussed below.

A common fact between all averaging smoothing methods, is that they smooth small details while preserving the general shape. Although preserving characteristics of lines is one of our important goals, smoothing small details is an undeniable fact that occurs in our basic algorithm. Even though, how much we preserve the caricature depends on appropriate values for smoothing parameters, *SS* and *SI* are just two constant values that do not relate to the curvature.

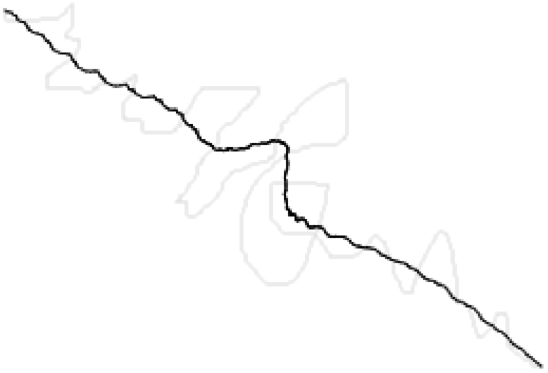


Fig. 2. Bad result caused by setting inappropriate value for *SS*. (*SS*=30, *SI*=5)

A. Solution for Preservation

In this section, a coefficient which indicates how strong we want to preserve the original line is presented. It does not lessen degree of smoothing but it will make the result more similar to the original line by preserving more details. Smoothing Preservation (*SP*) is what we call this variable. As a matter of fact, *SP* has no limit; however, it has an optimal value for each line depending on its features.

As previously discussed in subsection *B* of section III, let us assume *TR* as the triangle which is formed by *PTS(i – SS)*, *PTS(i)* and *PTS(i + SS)* respectively. What happens in the basic model is changing the coordinates of *PTS(i)* to *TR*'s centroid i.e. moving *PTS(i)* on one of *TR*'s medians until it gets to the centroid. Here, we increase the distance between *PTS(i)* and *TR*'s centroid by moving *PTS(i)* on the same median backwards, closer to its initial position or even outside of *TR*. Our new coefficient *SP* is the parameter which multiplies in the original distance and specifies the new distance between *PTS(i)* and *TR*'s centroid. This operation is formulated in (4).

The goal of this operation is to increase the distance between each point and its *TR*'s centroid to preserve the original distance when basic smoothing is applied. we combine this method with the basic SIA model to obtain a satisfying degree of preservation and smoothness in the same time. You can see that there is only a slight difference in the speed of smoothing process.

$$G = TR\text{ 's Centroid}$$
$$A = PTS(i)$$
$$A' = \text{New destination for } A$$
$$x_A = x_G \rightarrow \begin{cases} x_{A'} = x_A \\ y_{A'} = y_A + |AG|.SP & ; \quad y_A - y_G \geq 0 \\ y_{A'} = y_A - |AG|.SP & ; \quad y_A - y_G < 0 \end{cases}$$
$$x_A \neq x_G \rightarrow \begin{cases} \alpha = \dfrac{|AG|.SP}{\sqrt{(m_{AG}{}^2 + 1)}} \\ x_{A'} = x_A + \alpha & ; \quad x_A - x_G \geq 0 \\ x_{A'} = x_A - \alpha & ; \quad x_A - x_G < 0 \\ y_{A'} = m_{AG}.(x_{A'} - x_A) + y_A \end{cases}$$
$$\rightarrow |GA'| = |AG|.SP \tag{4}$$

Since this algorithm is combined with the basic model but behaving somehow opposite, the minimum value for *SI* should be 2, one for the basic algorithm and another for the preservation process; however, still there may be some distortions that can be repaired by applying more iterations of the basic method. (see Fig. 4)

The difference between basic model and the improved one lies in the first time smoothing process applies i.e. suppose that $SI = n$ for some natural number $n \geq 2$, then for the first time, operations formulated in (4) apply and for the next $n - 1$ iterations, only the basic algorithm applies.

B. Results for Combined Methods

Fig. 5 shows an example of a successful smoothing using both basic and improved SIA models. Notice the amount of preservation attained.

Due to the properties of neighboring points and the fact that both of these algorithms are based on these properties, we recommend limiting *SS* between 1 and 10. Since each point will get processed in both algorithms, changing *SS* does not have anything to do with the speed of smoothing process (smoothing speed mainly depends on *SI* and length of input list). The higher *SS* leads to a stronger smoothing in fewer iterations and vice versa.

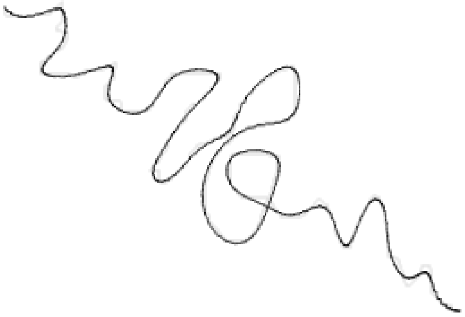


Fig. 3. Preservation of line characteristics for the improvement algorithm. The result is very close to original line while also nicely smoothed. (*SS*=5, *SI*=10, *SP*=15)

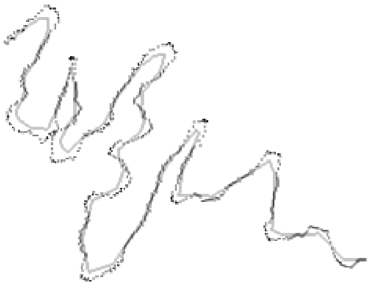


Fig. 4. Distortions happen by inappropriate number of iterations. Note that when applying the improved method, you need to find the minimum value for *SI* and it depends on the value used for *SP*. In this example 7 is the least value for *SI* that doesn't lead to any distortions. (*SS*=5, *SI*=2, *SP*=10).

V. SIA and Smoothing Via Bezier Curves

In this section, we have made a comparison between our presented method and Bezier Curve Smoothing. no comparison between speed of the algorithms is taken out since our focus is on resulting line and how much data has been preserved.

The application used for applying Bezier curve smoothing is the free demo program of simdesign's "Line smoothing using Bezier curves" by Nils Haeck M.Sc. which is available at http://www.simdesign.nl/bezier.html.

Smoothing using Bezier curves is a common solution to the problem of line smoothing. The method is capable of offering a wide range of smoothness. There are two tolerance parameters used in this method. First parameter is a number between 1 and 100 which indicates how hard or smooth the curves will be. The second parameter defines how precise the curve will match the user freeform, given in pixels.

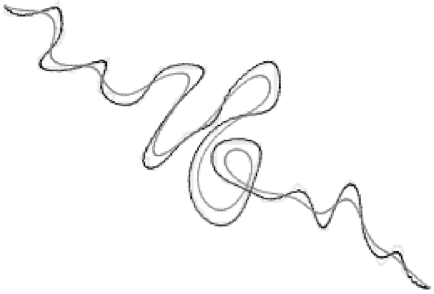


Fig. 5. Comparison of improved (Black line) and basic (dark gray line) methods. (*SS*=5, *SI*=10, *SP*=15)

After testing results of both methods on several inputs with different parameter values, we concluded that:

1) In case of noise reduction – the process of reducing unwanted perturbations in a line object – SIA offers better result since it's not sensitive to small variations of curvature (see Fig. 6)
2) Bezier curves tend to be sensitive about variation of tolerances especially precision. The method gives different results with small changes of precision. On the other hand, SIA slightly changes the overall shape and has a more stable manner. (see Fig. 7)

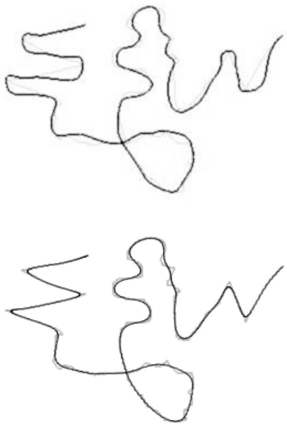


Fig. 6. Preserving the overall shape and sensitivity to slight variations of curvature. (left: bezier curves, right: SIA)

VI. Conclusion

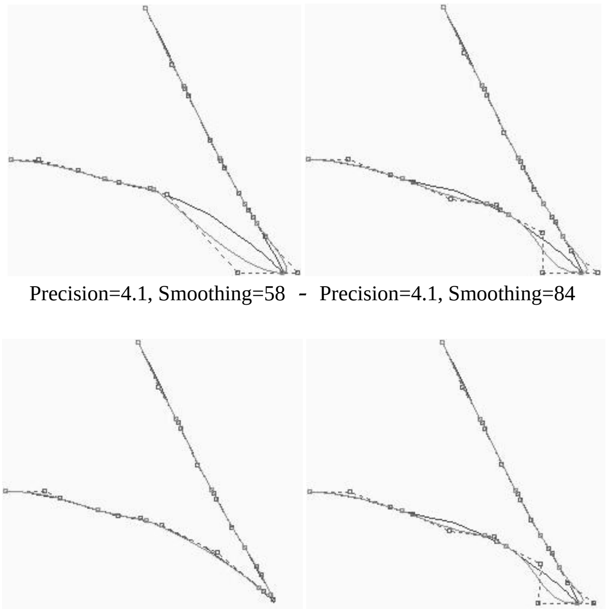
Smoothing of line objects can be done using various methods. What we are looking for however, is not just smoothing, but obtaining a sufficient amount of preservation.

Averaging methods usually small details which results in loss of characteristics in some conditions; however, we present some improvements which enable SIA to preserve original characteristics of a line object.

As a matter of memory usage, since the smoothing process is transition of points applying to the original list of points; no extra memory is needed.

The primary contribution of this paper is description of a novel and high-quality line smoothing algorithm with a working implementation available for practical use.

The smoothing algorithms presented in this paper are



Precision=4.1, Smoothing=58 - Precision=4.1, Smoothing=84

Precision=4.2, Smoothing=58 - Precision=4.2, Smoothing=88

Fig. 7. Sensitive behavior of bezier curve smoothing with slight variation of tolerances.

tested exhaustively and are recommended for commercial purposes due to promising results of our trials.

References

[1] M. Brophy, “An automated methodology for linear generalization in thematic cartography,” in *proc. of American congress of surveying and mappin*, pp. 300-314, 1973.

[2] J. Babauda, A. Witkin, M. Baudin, and R. Duda, “Uniqueness of the Gaussian kernel for scale-space filtering,” *IEEE Trans. On Pattern Analysis and Machine Intelligence*, vol. 8, no. 1, pp. 26-33, 1986.

[3] D. G. Lowe, “Organization of smooth image curves at multiple scales,” in *proc. of the second ICCV*, pp. 558-567, 1988.

[4] R. B. M. Master, “The integration of simplification and smoothing algorithms in line generalization,” *Cartographical*, vol. 26, no. 1, pp. 101-121, 1989.

[5] J. A. Dougenik. “Whirlpool; a program for polygon overlay,” in *proceedings of Auto-Carto 4*, pp. 304-311, 1980.

[6] E. V. Horn, “Generalizing cartographic data bases,” *Auto-Carto*, pp. 532-540, 1986.

[7] M. Harrower and M. Bloch, “MapShaper.org: a map generalization web service,” *IEEE Computer Graphics and Applications (CG&A) Special Issue on Geographic Visualization*, July / August 2006.

[8] B. R. McMaster and K. S. Shea, “Generalization in digital cartography,” *Assoc. of American Geographers*, Washington, D. C, 1992.

[9] M. Master and B. Robert, “Automated line generalization,” *Cartographica*, vol. 24, no. 2, pp. 74-lll,1987.

[10] G. Dutton, “Scale, sinuosity and point selection in digital line generalization,” *Cartography and Geographic Information Systems*, vol. 26, no. 1, pp. 33-54, 1999.

[11] A. Saalfeld, “Topologically consistent line simplification with the Douglas-Peucker algorithm,” *Cartography and Geographic Information Science*, vol .26, no. 1, pp. 7–18, 1999.

[12] M. S. Baghshah, S. B. Souraki, and S. Kasaei, “A novel fuzzy approach to recognition of online Persian handwriting,” ISDA’05, Wroclaw, Poland, September 2005.

[13] S. M. Razavi and E. Kabir, “A data base for online Persian handwritten recognition,” 6th Conference on Intelligent Systems, Kerman, Iran, 2004.

[14] D. Burghardt, “Controlled line smoothing by snakes,” GeoInformatica, vol . 9, no. 3, pp. 37–252, 2005.