



Student Project (Studienarbeit)– Collaborative Topic : Series II

## **Abstracting Digital Ink Traces with the Douglas-Peucker Algorithm**

27 February 2004

### **Offered by:**

Khairael A. Mohamed  
khairael@informatik.uni-freiburg.de  
[<http://www.informatik.uni-freiburg.de/~khairael>]

### **Joint supervision by:**

Prof. Dr. Thomas Ottmann  
ottmann@informatik.uni-freiburg.de  
[<http://ad.informatik.uni-freiburg.de/~ottmann>]



### **Preamble**

Often a polyline curve has too much resolution for applications such as visual displays of freehand writings, representations of geographic map boundaries, or detailed animated figures in games or movies. The points on these polylines, corresponding to the object boundaries, are too close together for the resolution of the application. For example, in a computer display, successive vertices of a polyline may be rendered on the same screen pixel so that successive edge segments start, stay at, and end at the same displayed point, when a reduction of screen resolution is inevitable.

In such cases, drawing all degenerate lines proves inefficient, as we can achieve the same effect by simply drawing a single point. To achieve this, we need to reduce the vertices and edges of the polyline to just the essential ones for meeting predefined screen requirements. At the same time, we must ensure that all information represented by the polyline remain succinctly legible.

### **Digital Ink and the Douglas-Peucker (DP) Algorithm**

We will consider the DP algorithm for reducing points in a polyline to produce a *simplified polyline* that approximates the original within a specified tolerance. In this project, we refer a polyline to be that of a digital ink trace produced by freehand writings on a digital screen.

Compressing and vectorising the process of sampling ink traces are ways of attempting to remove the insignificant points derived from a resultant input freehand trace. We may also want to go further and test the feasibility of keeping only the crucial points and be able to describe mathematically the various freehand *strokes* and *glyphs* contained within these traces.

The classical DP approximation algorithm provides a straightforward compression technique that is extensively used for both computer graphics and geographic information systems. There are two variants of this algorithm; the original  $O(n^2)$  method [1], and a more recent  $O(n \log n)$  one by Hershberger and Snoeyink [2]. In the algorithm, two extreme endpoints of a polyline are connected with a straight line as the initial rough approximation of the polyline. Then, how well it approximates the whole polyline is determined by computing the distances from all intermediate polyline vertices to that (finite) line segment. If all these distances are less than the specified tolerance  $\epsilon$ , then the approximation is good, the endpoints are retained, and the other vertices are eliminated. However, if any of these distances exceeds the  $\epsilon$  tolerance, then the approximation is not good enough. In this case, we choose the point that is furthest away as a new vertex subdividing the original polyline into two (shorter) polylines. This procedure is repeated recursively on these two shorter polylines. If at any time, all of the intermediate distances are less than the  $\epsilon$  threshold, then all the intermediate points are eliminated. The routine continues until all possible points have been eliminated. See Figure 1 for illustration.

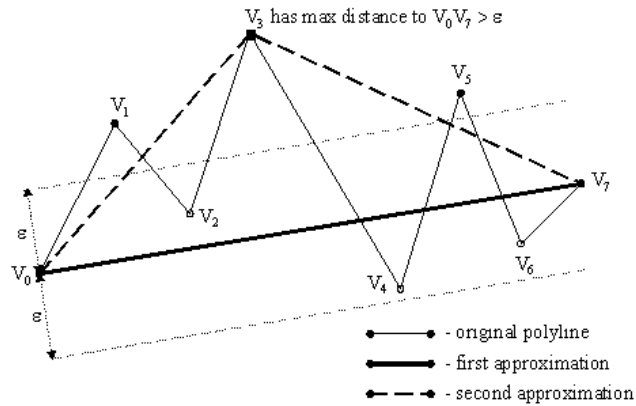


Figure 1: Step 2 of the DP algorithm

### Proposed agenda and guidelines

We will examine the effects the DP algorithm has on freehand ink traces, and then build on an online technique to abstract these traces in accordance a predefined set of criterions. The term of the project will be spent studying the:

- Literary groundworks on materials that are directly and indirectly related to fast and continuous polyline simplification algorithms.
- Standards and methodologies that involve the properties of digital freehand ink traces in accordance to the W3C InkML outline.
- Essentials that allow freehand ink traces to be dynamically scaled up (and down) without losing its rendering quality on screen of various resolutions.

Experiments carried out for this project are expected to run on the Java J2SE platform, accessing the extensive library of codes made available from the *uni.freiburg.mmg* package. We will work on:

- Developing a Java-based program (open-ended) to experiment with digital ink traces (parsed for InkML specifications) and implementing the DP algorithm on them.
- The comparative tasks of benchmarking the qualities between the original polylines against the simplified ones as a result of the DP algorithm.
- Proposing the set of criterions to be applied to a DP algorithm whenever digital freehand ink traces are involved.

### References

- [1] David Douglas and Thomas Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. In *The Canadian Cartographer* 10(2): 112 – 122, 1973.
- [2] John Hershberger and Jack Snoeyink. Speeding up the Douglas-Peucker line-simplification algorithm. In *Proceedings of the 5th Symposium on Data Handling*, pages 134 – 143, 1992.
- [3] Andrea Mantler and Jack Snoeyink. Safe sets for line simplification. In *Abstracts of the Tenth Annual Fall Workshop on Computation Geometry*, October 2000.