ADAPTIVE BOUNDS FOR QUADRIC BASED GENERALIZATION

Abhinav Dayal

IDV Solutions Inc. 5913 Executive Drive, Suite 320, Lansing, MI 48911

1. INTRODUCTION

Generalization is a common practice in GIS applications in order to reduce detail for efficient visualization. It is the process of selecting and representing information of a map in a way that adapts to the scale of the display medium of the map. There are a variety of generalization algorithms proposed in the literature. Garland et. al. [2] describes line simplification method **GSlim** based on a quadric error metric that uses vertex edge removal and local vertex repositioning. They associate with each edge, a removal cost based on a quadric error metric, which for 2D geometry is the sum of squared perpendicular distance of its midpoint to the neighboring edges. They iteratively remove minimum cost edge by replacing two end points of that edge with a single vertex at the mid point. [2] show this method to be more efficient and accurate to the venerable DP algorithm [1]. Moreover, since cost is local to each edge, it allows for selective generalization. However this requires one to determine the final number of edges or a target minimum cost in the simplified geometry. In this paper we present a novel way to use the map extents, average edge length and a user defined factor between 0 and 1 to determine the desired level of simplification using quadric based simplification.

2. ADAPTIVE GENERALIZATION METHODOLOGY

We want to develop a simplification measure that adjusts itself automatically to the scale and extents of the map and the underlying geometry. Further, we also want the user to be able to fine tune the simplified geometry for best control over the desired output. We do this by combining following three factors.

Terminal Edge Collapse Cost: This measure indicates the maximum allowable cost of collapsing an edge within the given extents. In ***Error! Reference source not found.*** ABCD is the simple geometry in the extents. Mathematically, we can show the terminal edge collapse cost,

$$C_{terminal} = \frac{1}{2} \times \left[\left(\frac{east-west}{4} \right)^2 + \left(\frac{north-south}{2} \right)^2 \right].$$

Edge-Significance Ratio: This determines how significant edges are in the given geometry when viewed on the world map. If μ is mean edge length in the input geometry and $E_{world} = +180°, W_{world} = -180°, N_{world} = +90°,$ and $S_{world} = -90°$ are the world extents, we compute edge significance ratio,

$$R_{\mu D} = \frac{\mu}{\sqrt{(E_{world}-W_{world})^2+(N_{world}-S_{world})^2}}.$$

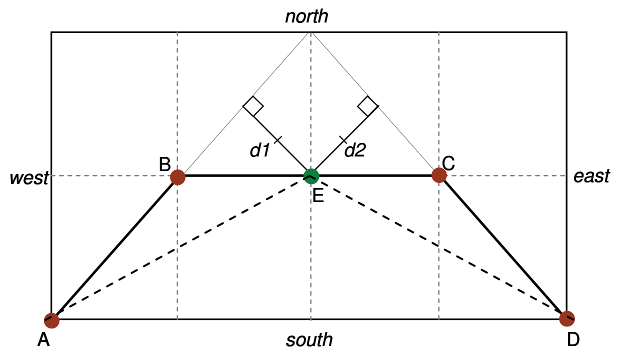

Figure 1: Computing Terminal Edge Collapse Cost.

User Defined Generalization Factor: This is a number (δ) between 0 [no generalization] and 1[maximum generalization in view]. We square this factor for a quadratic instead of a linear response.

We determine the ***Target minimum cost of edge collapse*** in simplified geometry by a simple product of the above three factors, i.e. target cost, $C_{target} = C_{terminal} \times R_{\mu D} \times \delta^2$. This is our final bound to limit the quadric line simplification. Algorithm 1 and 2 below show the stepwise procedure.

For each edge we set a collapse cost. And remove all edges whose collapse cost is below the target minimum. For a terminal edge, and edges that are on or adjacent to view boundary, we set collapse cost to a maximum value such that it never gets generalized. Finally, for edges outside the extents we downscale the cost by a constant factor of 0.01 giving them more priority to be generalized than the visible edges. Lowering collapse cost of outside view edges will push them towards top of the heap from where they are more likely to be removed.

Algorithm 1. Generalize

Input: EDGELIST: list of all edges, EXTENTS: view extents
Output: list of generalized edges

```
1. SET Heap ← empty set
2. SET Heap property to minimum cost at top.
3. For every edge $E_{BC}$ in input list
     a. Compute quardic matrix $Q_{BC}$
4. For every edge $E_{BC}$
     a. Set, $E_{BCcollapse}$ ← GetCollapseCost($E_{BC}$, EDGELIST, EXTENTS)
     b. Add edge to Heap and HEAPIFY
5. COMPUTE $C_{target}$. [section ]
6. While min cost in Heap > $C_{target}$
     a. EXTRACT min cost edge $E_{BC}$ from Heap
     b. EXTRACT left neighbor $E_{AB}$
     c. EXTRACT right neighbor $E_{CD}$
     d. Compute mid point E of $E_{BC}$
     e. Compute quadrics, $Q_{AE}$ and $Q_{ED}$
     f. INSERT $E_{AE}$ in Heap
     g. INSERT $E_{ED}$ in Heap
7. Form list of generalized edges from edges in the Heap.
```

Algorithm 2. GetCollapseCost

Input: E_{BC}: Edge to get collapse cost of, EDGELIST: list of all edges. EXTENTS: View extents
Output: $E_{BCcollapse}$: Collapse cost of edge, E_{BC}

```
1. If TerminalEdge($E_{BC}$)
     a. RETRUN MAX_COST
2. Get neighboring edges $E_{AB}$ and $E_{CD}$ of edge $E_{BC}$.
3. If OutsideExtents($E_{BC}$, EXTENTS) AND ( TerminalEdge($E_{AB}$) OR TerminalEdge($E_{CD}$))
     a. RETURN MAX_COST
4. SET $v$ ← [$x$ $y$ $1$], where ($x$,$y$) is mid point of edge, $E_{BC}$
5. SET Cost ← $Q_{AB}.v^T + Q_{CD}.v^T$
6. If OutsideExtents($E_{BC}$, EXTENTS)
     a. SET Cost ← 0.01 * Cost
7. RETURN Cost
```

3. RESULTS AND CONCLUSION

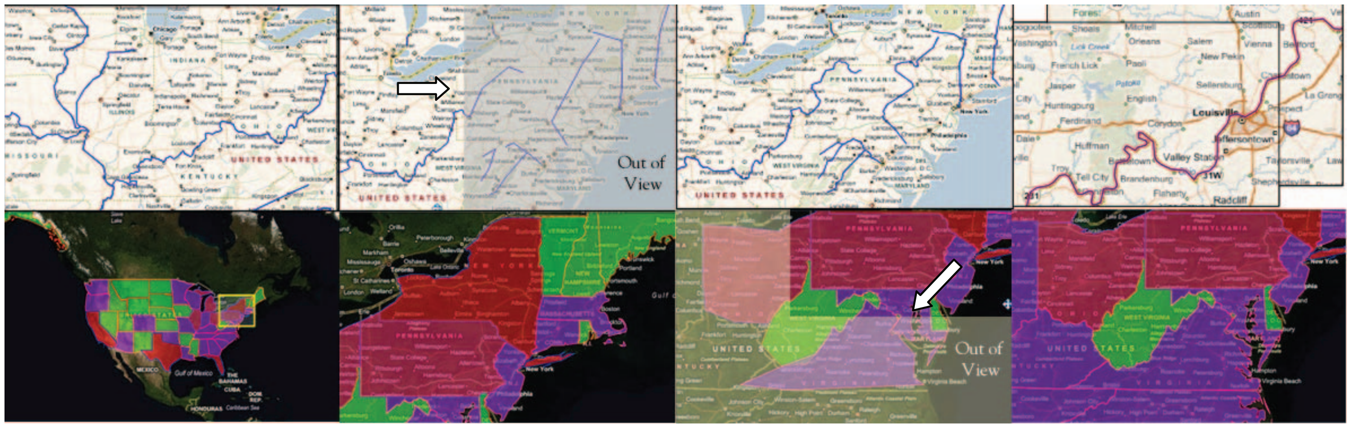

Figure 2. Results demonstration. Top row shows adaptive line generalization. Bottom row shows adaptive polygon generalization. See text for more details.

Figure 2 top row shows generalization on US rivers dataset with 5267 line segments. A zoomed in view renders only 444 edges with $\delta = 1$. The target cost maintains a necessary level of detail. Next slide shows out of view geometry to the right which is massively generalized because we reduced collapse cost of outside edges. Panning to right updates the view. The fourth slide on top row boundary edge continuity. It shows two frames in time. One below is a recent in time. Since the boundary edge is not collapsed, the continuity is maintained.

The bottom row shows similar characteristics with polygon dataset of the states in the US. Here $\delta = 1$ and in addition we also clip the geometry that lie completely outside the view extents.

In conclusion, this paper describes a novel way for automating the amount of generalization for the quadric based generalization method, providing a consistent view for any interactive mapping application. The suggested method is responsive to user defined parameters and adapts well to the view extents as does the Douglas Peucker algorithm [1], used most often in GIS applications. Providing such bounds for the GSlim method [2], allows for adaptive generalization of out of bounds geometry and efficient handling of borderline geometry.

11. REFERENCES

[1] Douglas, D.H. and T. K. Peucker. 1973. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. The Canadian Cartographer, 10(2): 112-122.

[2] Garland M. and Zhou Y. 2005. Quadric-based simplification in any dimension. ACM Transactions on Graphics Vol. 24(2): 209-239.