Hierarchical Simplification of City Models to Maintain Urban Legibility

Remco Chang† Thomas Butkiewicz† Caroline Ziemkiewicz† Zachary Wartell†
Nancy Pollard‡ William Ribarsky†
University of North Carolina Charlotte† Carnegie Mellon University‡
{rchang, tjbutkie, caziemki, zwartell, ribarsky}@uncc.edu {nsp}@cs.cmu.edu

1 Introduction

For 3D global visualization systems such as Google Earth, it is important to be able to render city-sized collections of relatively simple building models at fast speeds without losing spatial coherence. Since traditional mesh simplification algorithms are not designed for collections of simple models, we introduce a method of simplification through merging of similar objects. We incorporate the concept of "urban legibility" from architecture and city-planning as a guideline for simplifying city models. Our algorithm can be broken down into five steps. Hierarchical clustering, cluster merging, polyline simplification, and hierarchical texturing are performed during pre-processing, while at runtime, the levels-of-detail (LOD) process selects the appropriate models to render.

It is our belief that many applications can benefit from our algorithm. Google Earth (and other 3D geographical information systems) as well as any spatial data visualization applications (including scatter plots) can all use logical, simplified clusters to represent large amounts of spatial information.

2 Algorithm

In city-planning, the five elements of urban legibility are classified by Lynch [1960] as *paths*, *edges*, *districts*, *nodes*, and *landmarks*. According to Lynch, these five elements are the principle cues that city inhabitants use to identify their environments. By designing the five steps of our algorithm to preserve these elements, we produce simplified models that are better understood by viewers. Figure (a) shows the original city model with 285,039 polygons, and (b) shows the simplified model with 53,020 polygons.

Hierarchical Clustering: We adapt a single-link hierarchical clustering algorithm that iteratively groups the two closest clusters in the entire data set into a new cluster. The distance between two clusters is defined as the Euclidean distance between the two closest buildings from the two clusters. Because single-link clustering is a greedy algorithm, the resulting clusters would contain buildings that are on the same side of the roads before including buildings that are at large distances away. Figure (f) shows that our approach creates clusters that obey *paths* and *edges*. The nature of hierarchical clustering produces a dendrogram (a binary tree) of (n-1) clusters, which we use as the LOD hierarchy.

Creating and Merging Hulls: When two clusters are clustered together in the previous process, we also merge their footprints into a Hull. Footprints are defined as the projected outline of each building onto ground plane. Figure (e) shows the result of merging the Hulls of two clusters (Figures (c) and (d)) while creating a logical *district*. When merging two Hulls together, "negative spaces" (the black regions inside the Hull) are introduced which define the amount of "error" in each Hull.

Polyline Simplification: The Hull for each cluster is defined by a polyline, and can often be very complex, containing as many edges as the original number of footprint edges of the buildings. To simplify the Hull, we develop a polyline simplification algorithm based on the principles of a convex hull. This algorithm iteratively removes vertices by connecting their neighbors. At each step, we remove the vertex that adds the smallest positive area to the footprint while maintaining a non-self-intersection constraint. With our simplification, *paths*, *edges*, *districts*, and *nodes* are preserved.

Mesh Generation and Hierarchical Textures: Given the simplified Hull of each cluster, we trivially protrude the polyline towards the sky and create a mesh model for the cluster (called a cluster mesh). The height of the cluster is defined by the weighted average of the heights of all buildings in the cluster (Figure (g) shows cluster meshes of the clusters in Figure (f)). The roof and each face of the cluster mesh is given a texture based on orthographic projections (Figure (h) shows textured cluster meshes from Figure (g)). Cluster meshes are sorted into log(n) bins based on their negative spaces such that each bin generates six texture files (one for side textures, and five for five angles of the roof textures).

Negative Spaces and Levels of Detail: During runtime, the LOD hierarchy is traversed to find the appropriate clusters to render. A cluster is said to be appropriate if its negative space projected onto screen space is smaller (in number of pixels) than a user-defined threshold ε. If a cluster's projected negative space is greater than ε, the cluster will not be rendered, but its two children will be checked recursively. Furthermore, because taller buildings (*landmarks*) have higher visual importance than shorter ones, if a building's height projected to screen space is greater than a user-defined threshold α, the building is rendered separately on top of the cluster mesh. By considering the building heights separately from negative spaces, the skyline of a city is better preserved.

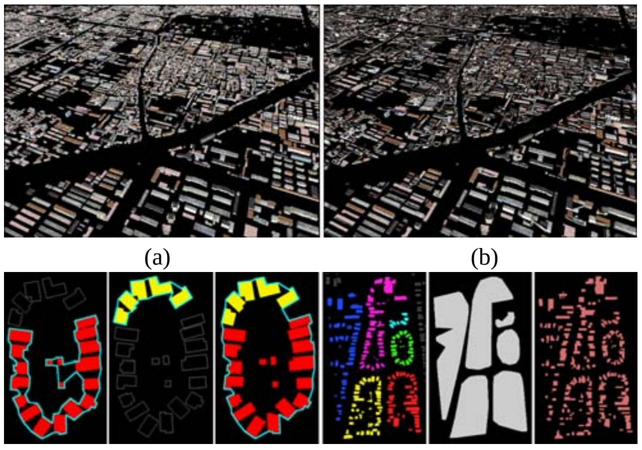

Figure 1: (a) Original model (285,039 polygons); (b) Simplified model $\varepsilon = 1000$ (53,020 polygons); (c)-(e) Merging the Hulls of (c) and (d) into (e); (f) Result of clustering; (g) Simplified cluster meshes from (f); (h) Applying textures to the meshes in (g).

References

Lynch, K. 1960. *The Image of the City*. The MIT Press.