Report on paper “Surface Reconstruction from Unorganized Points”

Nicholas Wan

University of Richmond

**Abstract**

In this paper, I will discuss the great achievement presented by Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald and Werner Stuetzle in their paper “Surface Reconstruction from Unorganized Points”. In their paper, they describe and demonstrate an algorithm that takes as input an unorganized set of points {x1, x2 … xn} ⸦ R3 on or near an unknown manifold M, (Figure I) and produces as output a simplicial surface that approximates M. (Hoppe, 1992) (Figure II)

**Background and Algorithm Overview**

At the beginning of the paper, the authors go through a variety of scientific and engineering applications of their algorithm by solving related problems. The fields include medication, chemistry and architecture etc. Then, they compare their algorithm against previous ones regarding to the similar problems and to summarize their advantages:

1. Their algorithm requires only an unorganized collection of points on or near the surface. No additional normal information is needed. (Muraki 1991)
2. Unlike the parametric methods, (Sclaroff and Pentland 1991) their algorithm can reconstruct surfaces of arbitrary topology.
3. Unlike implicit methods, (Moore and Warren 1990) their algorithm deals with boundaries in a natural way, and it does not generate spurious surface components not supported by the data.

To provide and overview of their algorithm, we can divide the whole process into two stages: The first stage is to define a signed distance function f: D → R3 where D is a region near the data, such that f estimates the signed geometric distance to the unknown surface M. The second stage is use a contouring algorithm to approximate Z(f), which is the zero set of *f*, by a simplicial surface. Now, let’s dive into the details of their approach.

**Tangent Plane Estimation and Their Orientation**

The key ingredient to defining the signed distance function is to associate an oriented plane with each of the data points. These tangent planes serve as local linear approximations to the surface. This process can be split into two steps: the construction of the tangent planes, which is relatively simple, and the selection of their orientations to define a globally consistent orientation for the surface, which is the first major obstacle in their algorithm.

The tangent plane *Tp(****x****i)*associated with a data point **xi**is represented as a point **oi**, called the center, together with a unit normal vector **ni**. To find this plane, we need its k-Nearest Neighbors and calculate the following parameters:

1. This plane passes though the barycenter of the neighbors: .
2. The normal direction is given by an eigenvector of smallest eigenvalue of the covariance matrix: .

After we have the tangent plane for each data point, we need to select a consistent orientation for all nearby planes so that any two points (xi, xj) that are sufficiently close, are nearly parallel, meaning ni ∙ nj = 0. At this point, the problem we have can be described as a graph optimization problem where the graph contains one node Ni per tangent plane *Tp(****x****i),* with an edge (i, j) between Ni and Nj if the tangent plane centers oi and oj are sufficiently close. However, this problem has been shown to be NP-Hard via a reduction to MAX-CUT. (Garey and Johnson, 1982) To efficiently solve the problem we must therefore resort to an approximation algorithm.

**Approximation Algorithm on Riemannian Graph**

To begin with, we need some specifications on the graph. They start by constructing the Euclidean Minimum Spanning Tree while the edges are not sufficient enough for finding all tangent planes that are supposed to be nearly parallel. Therefore, they add more edges to the graph by adding edge (i, j) if oi and oj are within each other’s k-neighborhood. The resulting graph, called the Riemannian Graph, is thus constructed to be a connected graph that encodes geometric proximity of the tangent plane centers.

The approximation algorithm begins by simply arbitrarily choosing an orientation for some plane, then “propagate” the orientation to neighboring planes in the Riemannian Graph. However, if the order in which the orientation is propagated solely depends on geometric proximity, (Figure V) the result will be incorrect. The very smart propagation order they use is achieved by taking any route from *Tp(****x****i)* to *Tp(****x****j),* assigning the cost to each edge 1-|ni - nj|, then traversing the minimal spanning tree of the resulting graph. This order is advantageous because it tends to propagate orientation along directions of low curvature in the data, thereby largely avoiding ambiguous situations encountered when trying to propagate orientation across sharp edges.

The correct result of their oriented tangent planes is shown in Figure VI whereas the original mesh is shown in Figure IV.

**Signed Distance Function**

At this point, we have entered the last step of stage one, which is to assign the Signed distance function *f(p)* to an arbitrary point p∈R3. The signed distance is defined by the distance between p and the closest point z∈M, multiplied by plus or minus 1 depending on which side of surface p lies. To achieve this, these three steps are described by the authors in their algorithm:

1. Find the closest center to p in our set of oriented tangent planes.
2. Compute the signed distance to the plane.
3. If d(z, X) < ρ+δ, then f(p) = ((p-oi)∙ni)ni. Otherwise, f(p) is undefined. In the equation, ρ and δ are defined by the sample set X = {x1, x2 … xn}, which is a ρ-dense, δ-noisy sample of M.

There is one thing that needs attention though, the signed distance function is not continuous because the tangent planes are not continuous. However, it still provides a good approximation to reconstruct the surface. Furthermore, the contour tracing algorithm discussed in the next section will discretely sample the function f over a portion of a 3-dimentional grid near the data and reconstruct a continuous piecewise linear approximation of Z(f).

**Contour Tracing**

The authors chose to implement a variation of the marching cubes algorithm (Allgower and Schmidt, 1985) that samples the function at the vertices of a cubical lattice and finds the contour intersections within tetrahedral decompositions of the cubical cells. The detailed process of their algorithm can be summarized as follows:

1. Define a cube partition of the space. The edge of each cube should be less than ρ+δ.
2. Compute the signed distance function on the cube vertices.
3. Interpolate zero values (i.e. Surface intersections) at changing sign edges.
4. Find a triangulation with vertices at zero values.

This algorithm will finalize the whole algorithm and result in a figure shown in Figure III.

**Discussion and Reflection**

This paper in general provides an efficient algorithm in solving related problems. The complexity can be categorized into the following subproblems:

1. Construction of Euclidean Minimum Spanning Tree: O(n2).
2. K-nearest neighbors of given point: O(n+klogn).
3. Nearest tangent plane center to a given point: O(n).

There are a few issues I would like to address though. First, in choosing the number of neighbors in constructing the tangent planes, the paper claims that the value of k should be a parameter input by users. However, the value k is not a critical value especially in the case that the data contains little or no noise. Therefore, it would be better if we can come up a way to automatically generate the k-value based on the data. Also, the algorithm is meant to work in 3-dimention. I would like to know its performance in the context of 2-dimension and more importantly, higher dimension. If it is also efficient and generally correct in higher dimension, it could be very useful in the calculation of matrix.

**References**

Allgower, E. L., & Schmidt, P. H. (1985). An Algorithm for Piecewise-Linear Approximation of an Implicitly Defined Manifold. *SIAM Journal on Numerical Analysis,* *22*(2), 322-346. doi:10.1137/0722020

Hartmanis, J. (1982). Computers and Intractability: A Guide to the Theory of NP-Completeness (Michael R. Garey and David S. Johnson). *SIAM Review,* *24*(1), 90-91. doi:10.1137/1024022

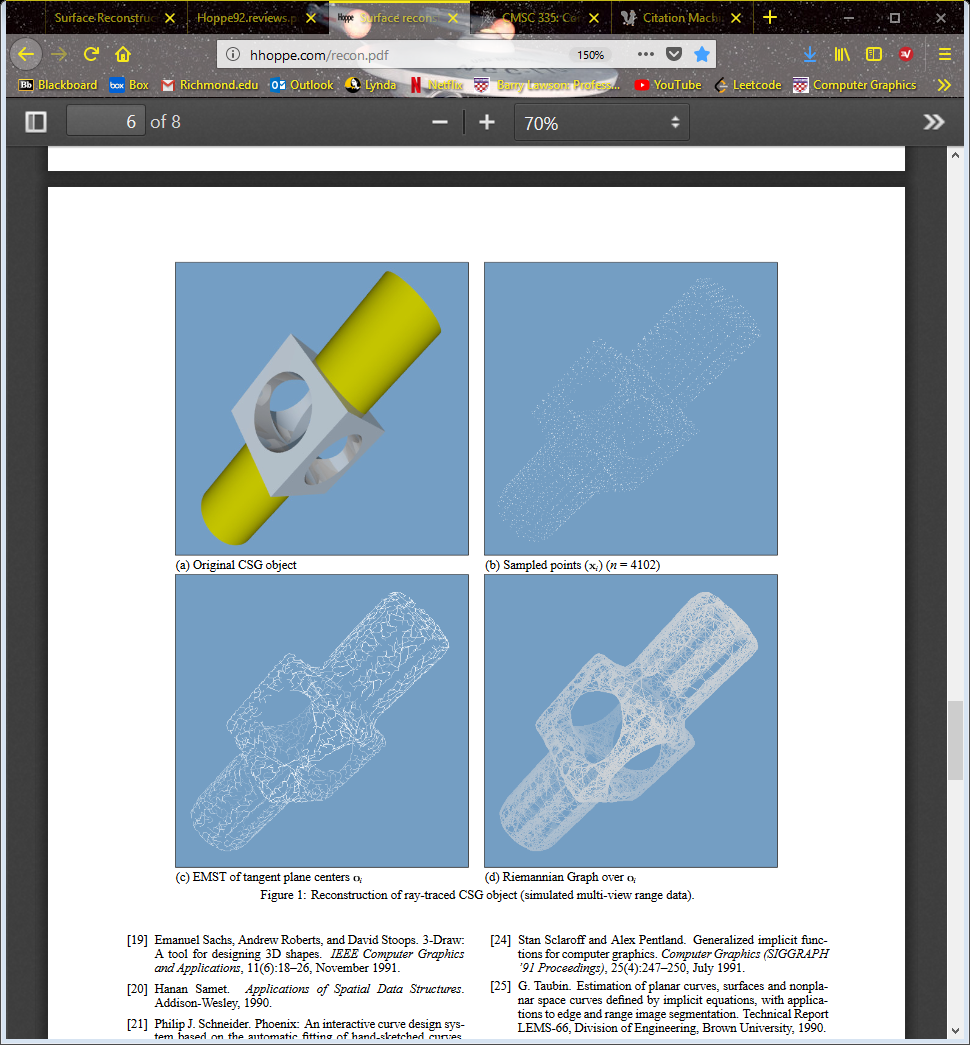
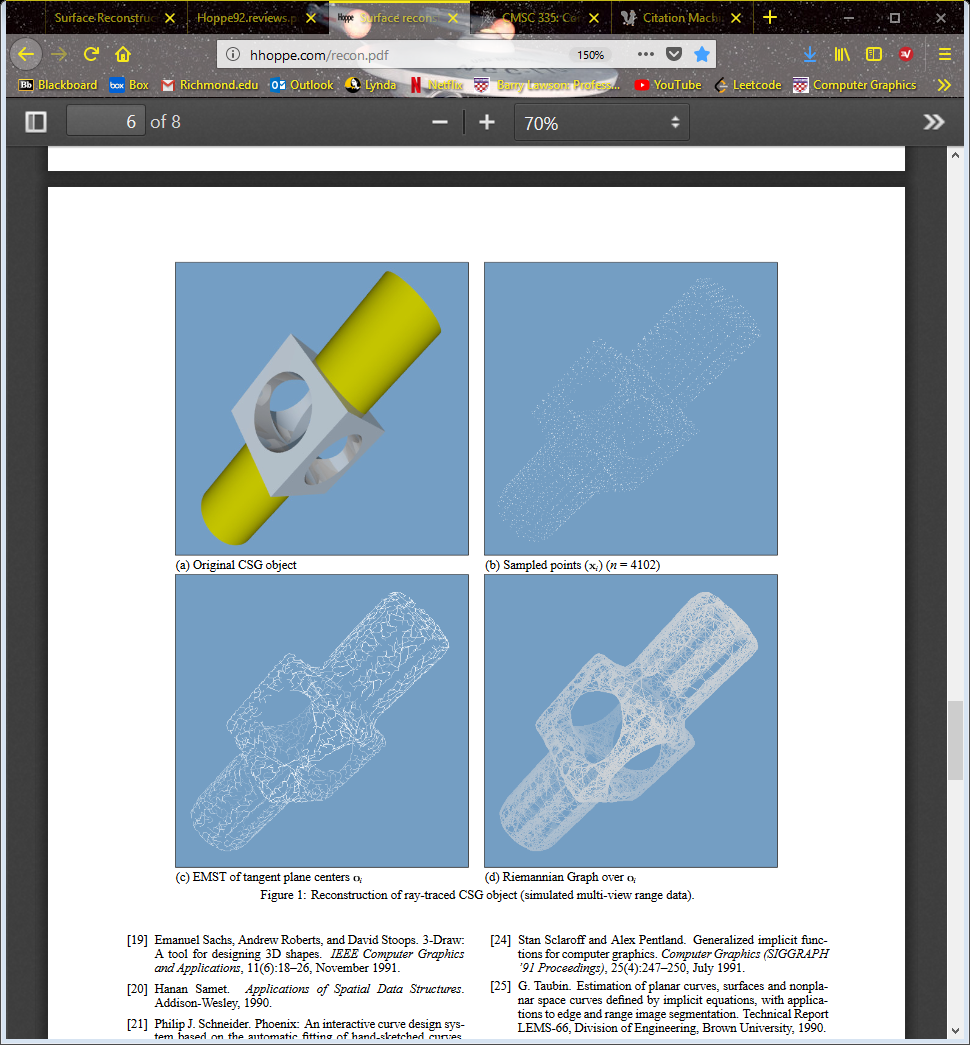
Hoppe, H., Derose, T., Duchamp, T., Mcdonald, J., & Stuetzle, W. (1992). Surface reconstruction from unorganized points. *Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques - SIGGRAPH 92*. doi:10.1145/133994.134011

Moore, D., & Warren, J. (n.d.). Approximation of dense scattered data using algebraic surfaces. *Proceedings of the Twenty-Fourth Annual Hawaii International Conference on System Sciences*. doi:10.1109/hicss.1991.183942

Muraki, S. (1991). Volumetric shape description of range data using “Blobby Model”. *Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques - SIGGRAPH 91*. doi:10.1145/122718.122743

Sclaroff, S., & Pentland, A. (1991). Generalized implicit functions for computer graphics. *Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques - SIGGRAPH 91*. doi:10.1145/122718.122745

Figures



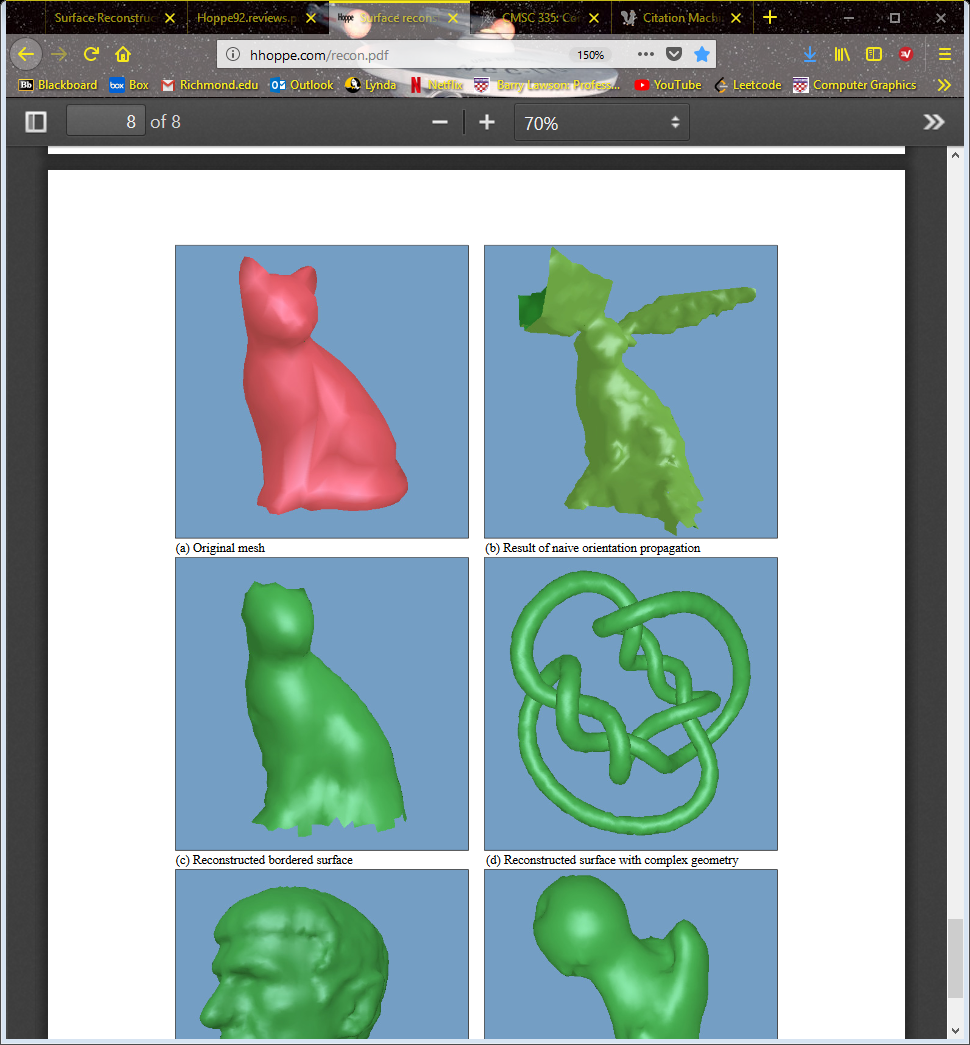
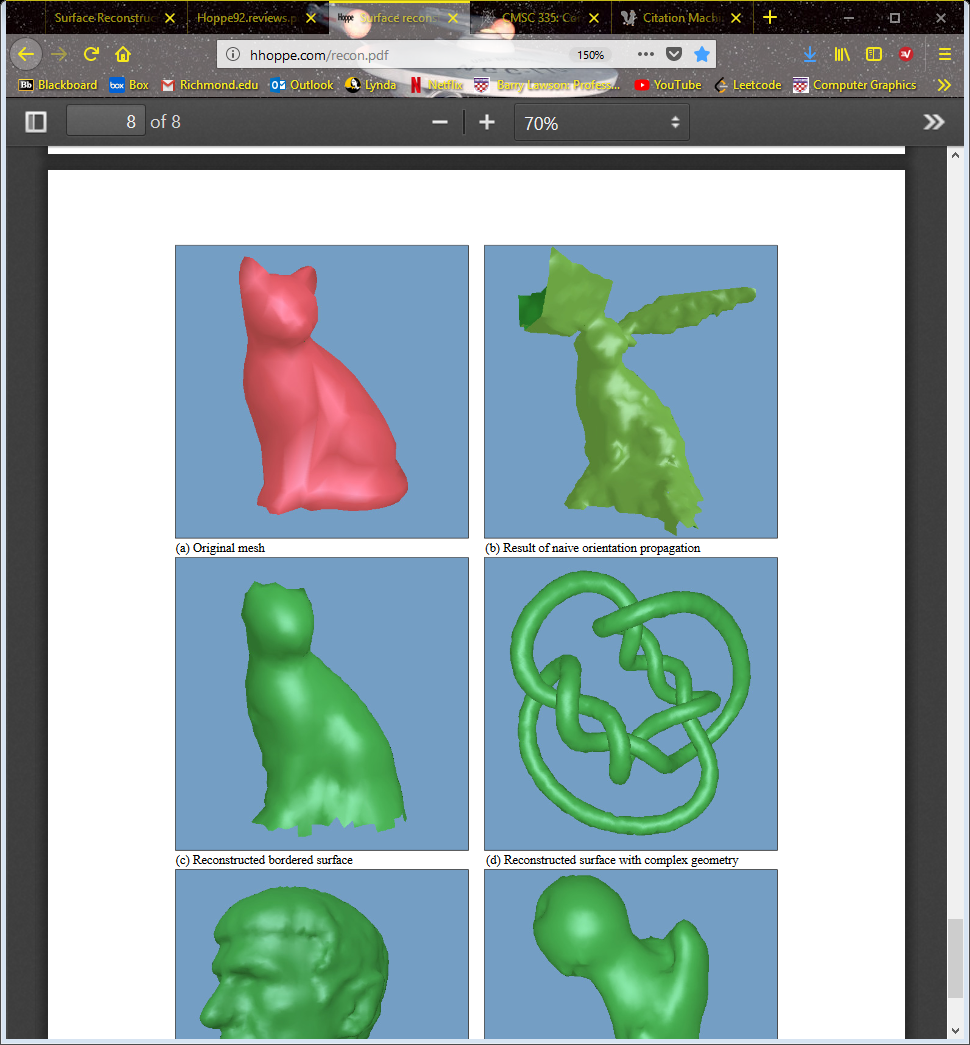
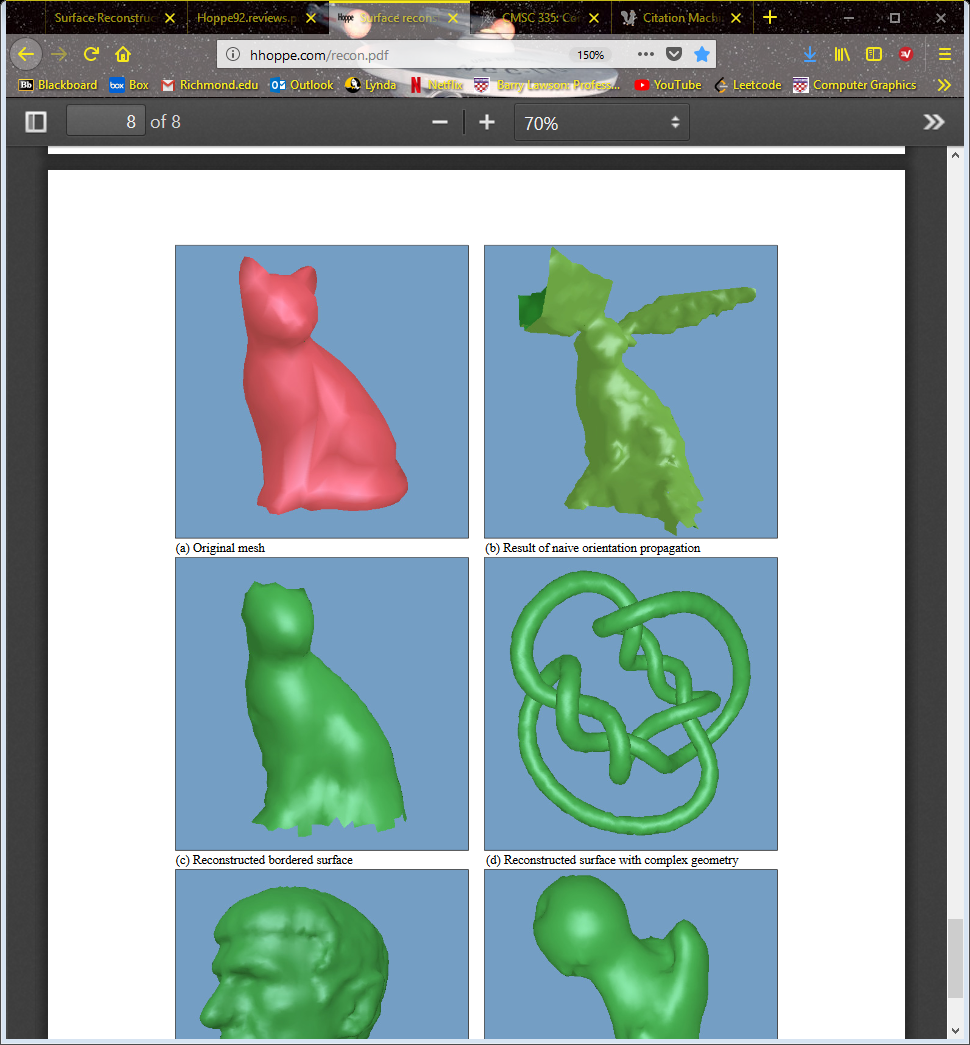
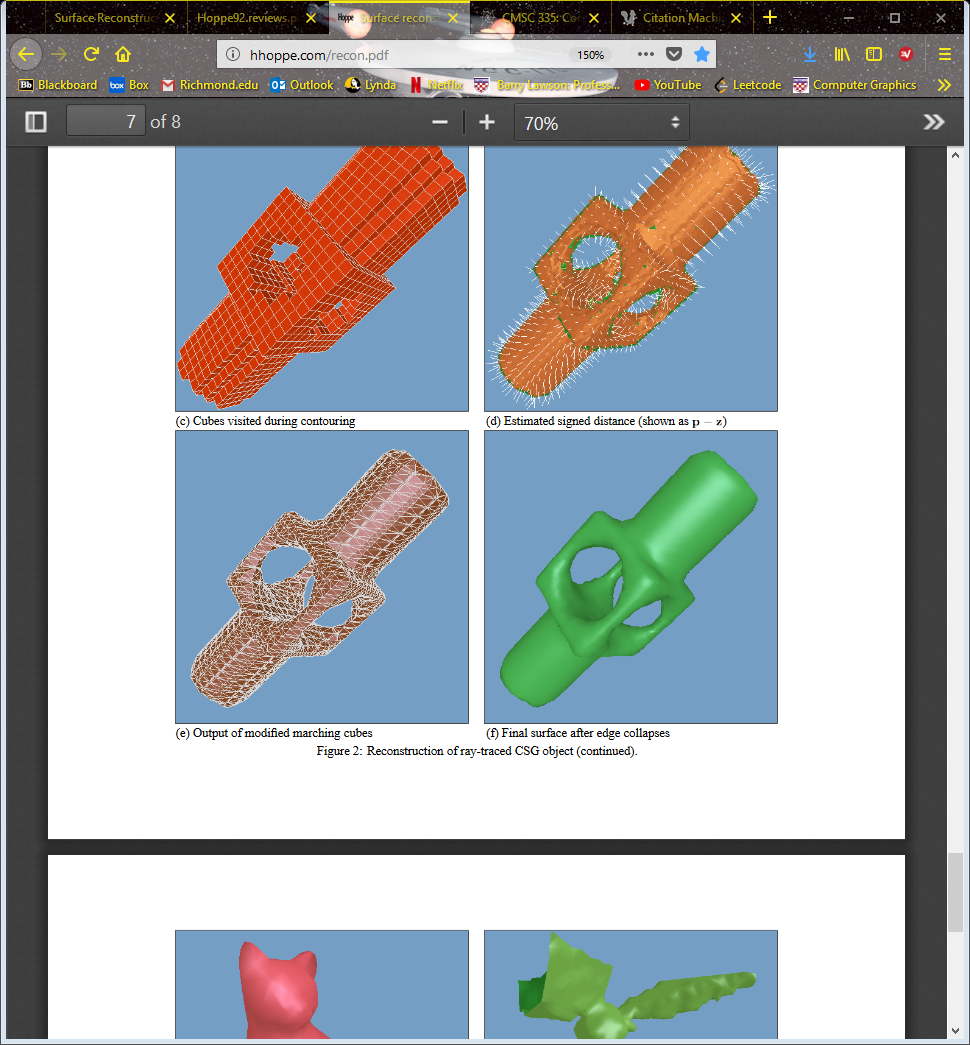


Figure VI

Figure V

Figure IV

Figure III

Figure II

Figure I