CARNEGIE MELLON UNIVERSITY

School of Architecture
College of Fine Arts

Thesis

Submitted in Partial Fulfillment of the requirements for the degree of

Master of Science in Computational Design

TITLE:

Make Models Great Again:
Interactively generating simplified editable models from 3D-scanned dense meshes

AUTHOR:

Wumengjian Zhu

ACCEPTED BY ADVISORY COMMITTEE:

Daniel Cardoso Llach		May 15, 2018
Professor NAME	Principal Advisor	DATE

MAKE MODELS GREAT AGAIN:

Interactively generating simplified editable models from 3D-scanned dense meshes

Wumengjian Zhu

Advisory Committee: Daniel Cardoso Llach, PhD

Abstract

3D scanning is a technique that collects data from real-world objects to construct digital three-dimensional models. Following various cutting-edge applications in manufacturing and various industries, this reverse-engineering technique was creatively and famously adopted by architect Frank Gehry to digitize physical models to facilitate his further design.

Although integrating these techniques into the earlier phases of design has been a tempting topic in design practice, achieving it is not easy. In addition to generating a mesh using either professional scanners or easy-to-access photogrammetry software, the generated mesh must be simplified into a editable form that is friendly to human designers. While the first part is a relatively mature process, current methods for mesh simplification are either difficult to control during simplification, or generate hard-to-manipulate triangle meshes.

The purpose of this thesis is to present the world of design with a newly developed 3D reconstruction oriented design workflow. It's core contribution is implementing, optimizing, and evaluating a new interactive mesh simplification workflow based on Variational Shape Approximation, which generates easier-to-control designer-friendly simplifications. High performance parallel computing techniques (such as CUDA) were explored and used to improve the efficiency of this algorithm. A new modelling plug-in, iSimp for Autodesk Maya, is developed and evaluated by a group of designers in a workshop. It is found that the proposed mesh simplification workflow can greatly facilitate model digitization for creative designers, and more importantly, inspire new approaches in designing the from physical world. Digital models, which are once great in their physical forms, can be made great again.

Keywords

3D Scan, Mesh Simplification, Design Workflow, Photogrammetry

Table of Contents

1 Introduction

1.1 History of 3D Scanning

Measuring and reproducing drawings/models of existing objects has always been an important topic either from the point of view of design, construction, manufacture or preservation. In principle, 3D scanning is one of the many techniques developed for measurement, which can date back to the earliest civilizations of mankind, e.g. Ancient Egypt [1], Ancient China [2], etc.

The earliest 3D measurement system appear in the 1960s, when the very first Coordinate Measuring Machine (CMM) was invented by the Italian company DEA. Portable CMMs with measuring arms, first built in the 1980s, started a revolution on the traditional measurement process. With portable CMMs, it is now possible to take measurements directly in the field, instead of having to bring the object to a dedicated, controlled environment. During this time, CMMs are made with mechanical measuring arms, which causes them very sensitive to vibrations and instabilities induced by the environment. [3]

Later in the 1980s, laser was introduced in the development of 3D scanner. The first laser tracker was invented and put into market by Dr. Kam Lau in 1987. The use of laser has greatly improved precision and portability of 3D measurement systems and has enabled large-scale scans on objects such as aircraft wings, auto frames, etc. [3]

In addition to direct measurement technologies, photogrammetry, which is the science of making measurements from photographs, was first used in the 1840s shortly after the invention of photography. [4] This technology enables determination of the object's relative position in space from a series of photos taken from different viewing angles. As of today, photogrammetry and laser scanners are two of the most widely used 3D measurement techniques especially for large-scale objects. [3]

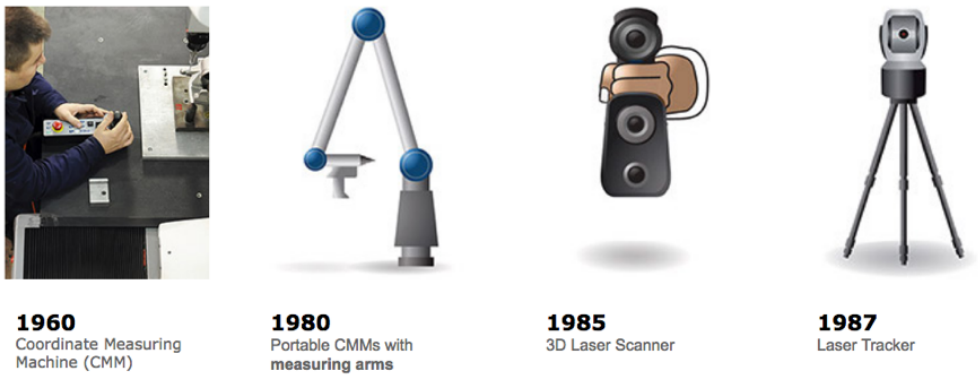

Figure 1 Illustrated history of 3D measurement [3]

1.2 3D Scanning for Reverse Engineering

Ever since its invention, 3D scanning has been playing an active part in both academia and industry. Reconstructing a physical object into its digital form is a very appealing topic and has been an essential step in various application scenarios ranging from manufacturing quality control [5], rapid prototyping [6], documentation [7] and visualization [8]. Many of these use cases are essentially reverse engineering of free-form shapes [9], which has been an active area of study in fields such as airplane industry, car industry, design practice, etc.

1.2.1 Applications in Academia

One of the first academic applications of reverse engineering took place in the University of Utah, where Professor Ivan Sutherland and his students digitize his Volkswagen Beetles, which was “something iconic to realistically render”. [10] The students used yardsticks to measure the x, y and z coordinates of the painted points on the car surface and used points and polygons to render it on a computer screen. Although tools were inaccurate and brought about huge measurement error, this computer-rendered image became one of the most symbolic piece of history in Computer Graphics.

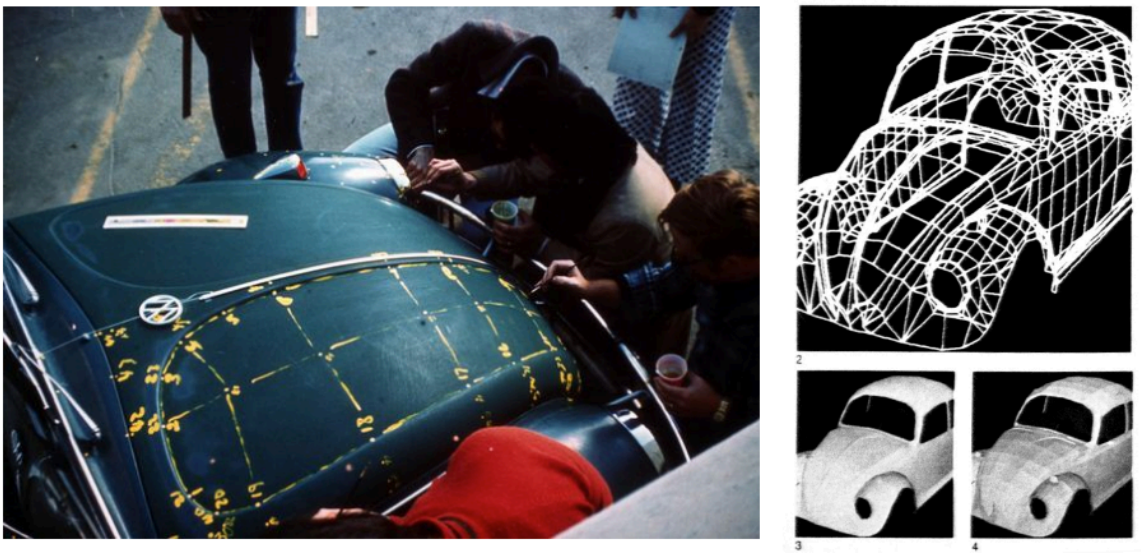

Figure 2 University of Utah students map Professor Ivan Sutherland’s Volkswagen car as a reference model for their graphics experiments, 1977 [11]

In current architectural studies, 3D scanning is being widely used in digitally reproducing historical buildings. Drawings can be generated directly from scanned point cloud data. As historical buildings tend to be complicated in shapes and details, the use of 3d scanning can be both efficient and precise.

The figure below shows the reconstruction of an ancient Chinese temple in Taiwan. Using 3D Scanning for historical building preservation. The team scanned a temple in Taiwan and printed scaled models of the model and some architectural parts.

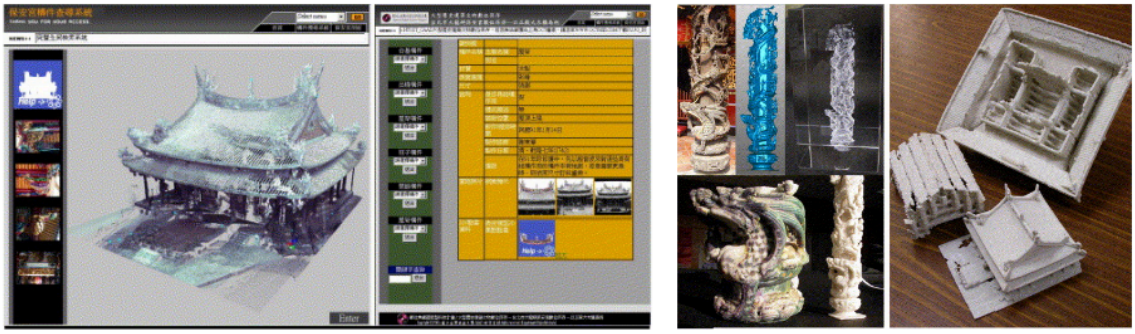

Figure 3 Reconstruction of an ancient Chinese temple [12]

1.2.2 Applications in Industry

The reverse engineering approach has been widely adopted by industrial designers. Otto et. al proposed a design methodology with three phases: reverse engineering, modelling & analysis and redesign [13]. This design approach, which begins from an existing product, seeks to develop from feedbacks such as user experience or market performance. It seeks to redesign based on thorough evaluations of the current product in order to improve product quality. Depending on the functionality of the product, the evaluation can be either by measurable metrics or aesthetic judgement [14].

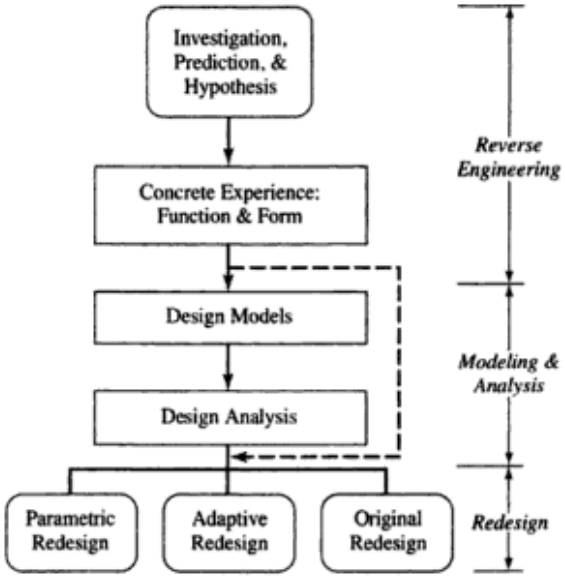

Figure 4 General reverse engineering and redesign methodology [13]

Manufactures in the automotive industry [15] has been using tools such as the Coordinate Measuring Machines to re-create CAD models of free-form surfaces, namely, car parts. This method can be very useful when re-designing upon a product which was not digitally designed in the first place.

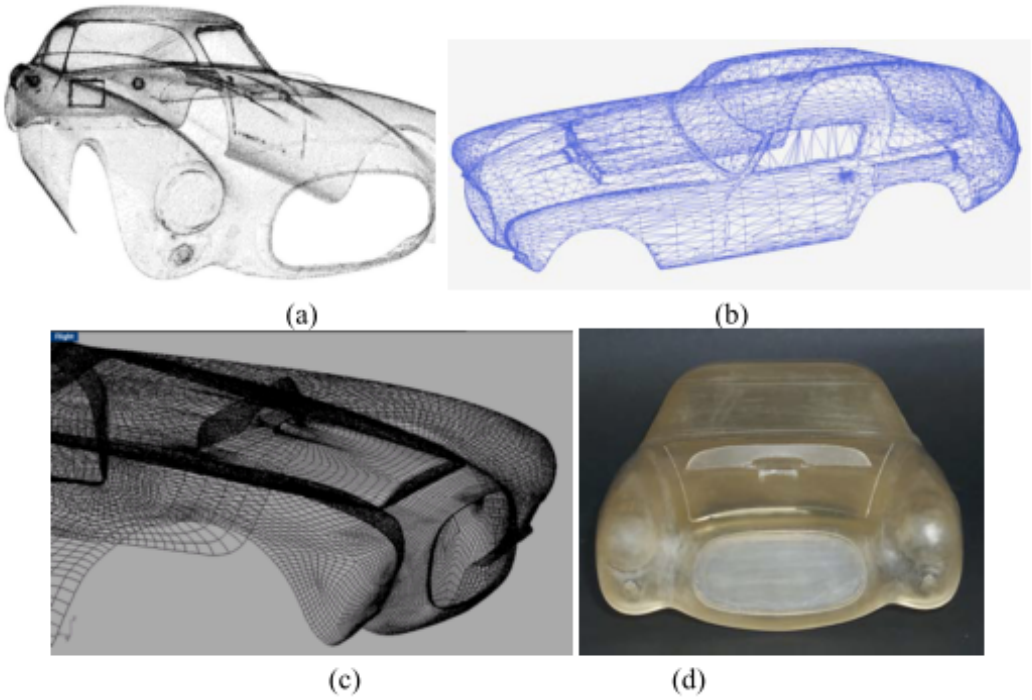

Figure 5 Example of full optical reverse engineering of the Ferrari 250MM historical racing car using OPL-3D. (a) Point cloud of the car body. (b) Triangle mesh. (c) CAD model. (d) Scaled prototype. [16]

It is notable there are tradeoffs regarding accuracy and time consumption of different measuring methods. Typically, a CMM based on physical probing is much more accurate than optical methods, but takes significantly more time[9]. The rapid development of 3D imagery technologies also contributes to the reduction of workload of reverse engineering. For application scenarios that does not need high precision, optical sensors and professional software such as PolyWorks can greatly simplify the design workflow.

In building construction process, there has been increasing use of 3D scanning technologies in recent years. The technology is being used to measure construction errors, as in manufacturing, and to measure and document old buildings for renovation purposes. The use of 3D scanning in these scenarios has greatly increased precision as well as reducing human labor involved.

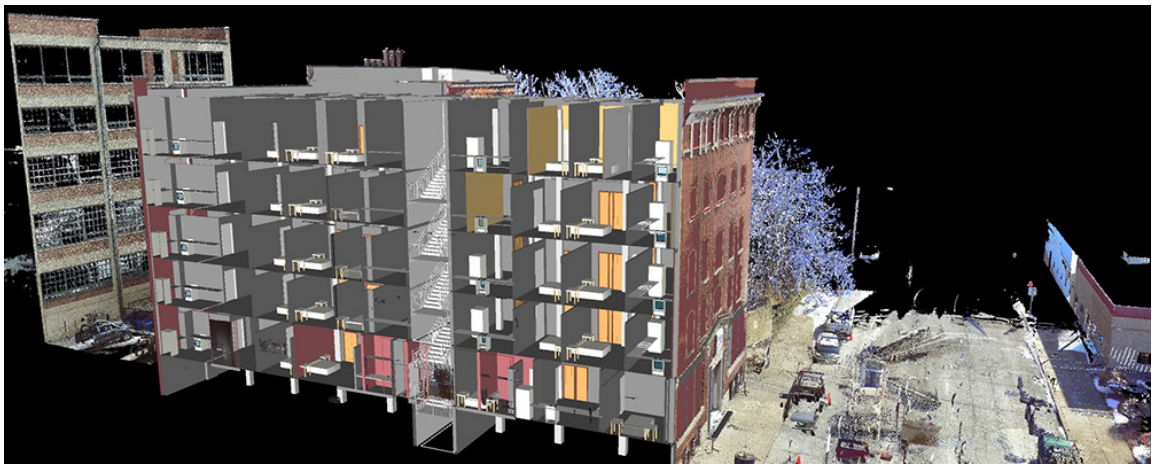

Figure 6 3D scan in building renovation [17]

1.2.3 Applications in Design

In architectural design, introduction of the reverse engineering design approach dates back to as early as 1991. The contact-based CMM system was creatively and famously adopted by architect Frank Gehry in his design of the Barcelona Fish [18], Walt Disney Concert Hall [19], and many other later projects.

According to Rick Smith, who was the technical consultant of Frank Ghery's, the introduction of digitizing arms and modelling software made it possible for building complex shapes that Gehry designed. Rick Smith recalled that unlike in aerospace or engineering industries, architects like Frank Gehry did not like designing on the computer. [20]

> *"Frank would work out his paper designs and then make a model. I would follow, digitizing his models to build a computer model. Then he would look at the computer model and the physical model he'd built to check for differences and to see whether we actually captured his shapes."* – *Rick Smith*

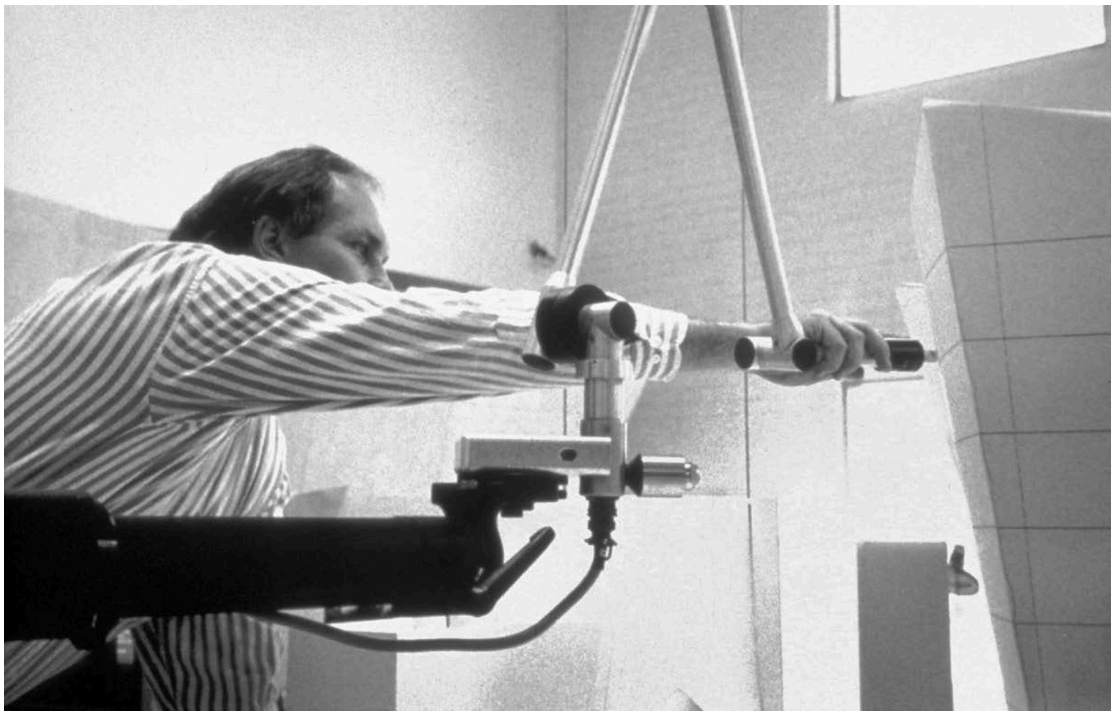

Figure 7 Rick Smith digitizing Frank Gehry's model of the Walt Disney Concert Hall with FaroArm, 1991 [20]

This design process adopted by Frank Gehry and Rick Smith is a perfect extension of Otto's redesign theory. Rather than gathering customer feedbacks from the market, the designer himself would offer feedbacks on the results from reverse engineering, and make adjustments based on feedbacks.

In this scenario, reverse engineering serves as an essential digitizing step in the design process for buildings that cannot be designed or built using traditional techniques. It bridges the gap between designers who are more comfortable with expressing ideas with physical models and computer-aided design methods that are crucial for designed complex systems.

1.3 Limitations in Current 3D Scanning Applications

Although the proposed design workflow seems promising, there are technical difficulties in its realization.

For most designers, it is hard to get access to a set of 3D digitizing equipment, nor do they have the resources for a time-consuming digitizing process using CMMs.

Another primary limitation is lack of software support in re-creating human editable models from 3D scan, aka mesh simplification. Currently reconstructed meshes from dense point clouds have the following problems:

1) *Reconstructed meshes typically consist of too many faces, which is not friendly to human designers.*

2) *The current solution for converting reconstructed mesh to NURBS CAD model still requires lots of manual adjustment, which is a very labor intensive type of work.*

The following figure shows a mesh decimation experiment in Autodesk Recap, a well-known commercial photogrammetry software. We can see that as we try to reduce the face count of the mesh, it will start to deform and lose high-frequency features. This decimation feature is only suited for reducing storage space, not for generating editable models.

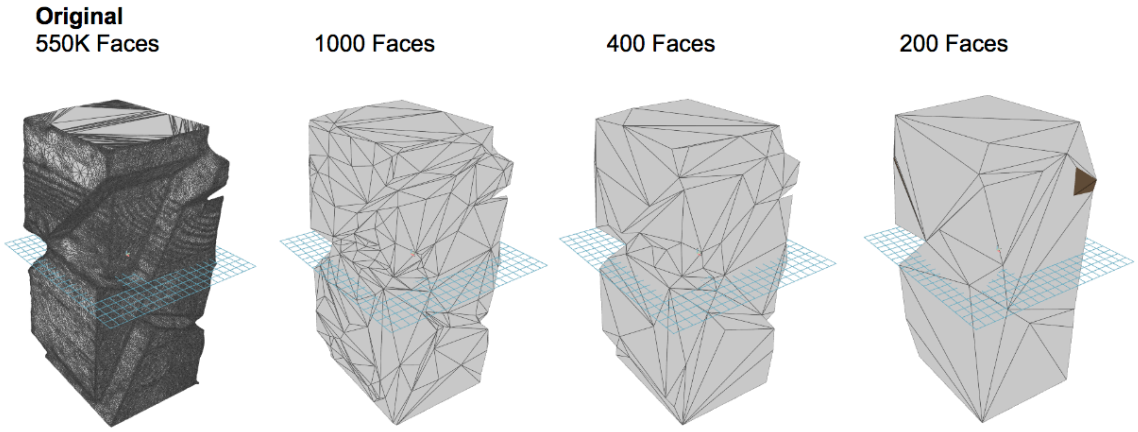

Figure 8 Mesh decimation experiment in Autodesk Recap

1.4 Vision: A New 3D Reconstruction Oriented Design Workflow

Reverse engineering practices set forth by engineering industry and pioneer architects are very inspiring. For designers looking for inspirations from real-life objects or working with physical models to test out concepts, it can be very tempting to adopt similar 3D digitizing techniques to move forward with the design process. Incorporating 3D scanning techniques into the earlier phases of design can open up great possibilities for creativity.

The broad utilization of 3D scanning technology benefits from a relative mature workflow. Currently, for 3D scanning and reconstruction, a typical workflow may appear as follows:

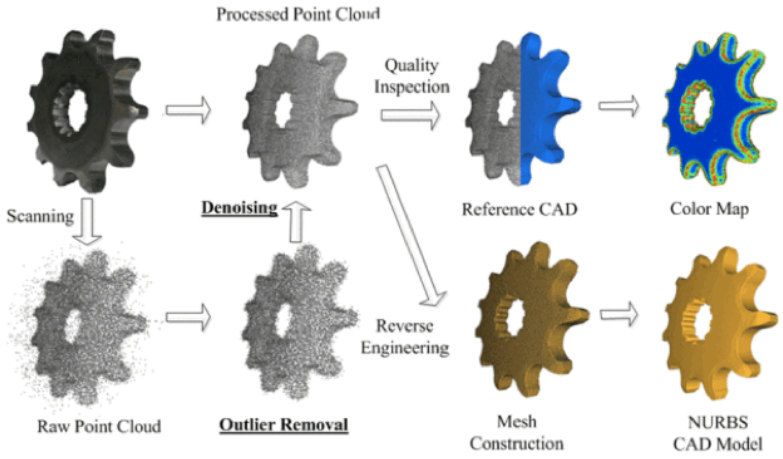

Figure 9 Typical 3D scan workflow [21]

Point clouds produced by a 3D scanning device will typically go through a de-noising phase. If it is used for quality inspection, it can be directly compared with the referenced CAD model. If the desired final output is a editable model, as used in Frank Gehry's design process, the point cloud will first be reconstructed to a mesh model, and then simplified for human editing.

In this thesis, we propose a simplification driven design workflow that is intended to be easily accessible to all designers with limited resources, who either do not have access to professional 3D scan equipment, or do not have enough time for labor intensive reverse engineering tasks. A designer may choose to incorporate the following procedures in a conceptual design phase:

1) *Get inspired by an object from nature or a handcrafted physical model.*
2) *Produce a digital representation (triangle mesh) of the object from step 1, using easy-to-access 3D scanning techniques such as photogrammetry.*
3) *Interactively simplify the dense mesh to an editable polygon mesh using the software tool developed by us.*
4) *Directly modify the simplified result to refine the shape, using commercial 3D modelling software.*
5) *Take the result to a next step, e.g., 3D printing, schematic design, etc.*

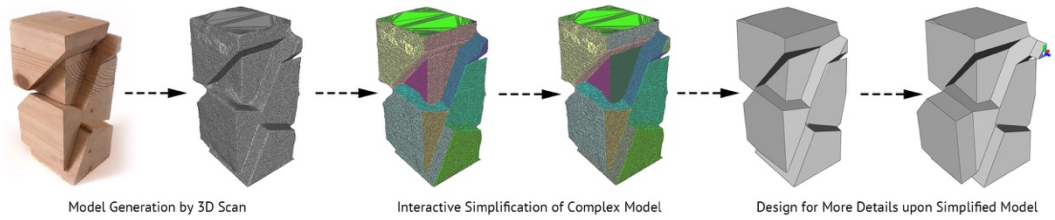

Figure 10 Step by step illustration of 3D scan oriented design workflow

The hypothesis is that this newly proposed workflow can help designers transfer smoothly from a physical model to designing in the digital world.

1.5 Summary of Work

Currently, certain amount of difficulty occurs if a designer wants to start a concept from digitizing a physical model. The goal of this thesis is to bridge the gap between the complex model generated by 3D scanning/photogrammetry software, and the simple polygon model liked by designers because of the edibility. Once eliminating this obstacle, new approaches will be inspired in designing directly from physical world.

In this thesis, the following contributions are made:

1) *A re-discovery of an existing mesh simplification algorithm was made to bridge the gap of processing 3D scanning results. The algorithm was then thoroughly implemented and tested for various edge cases.*
2) *A new plug-in for Autodesk Maya was developed to facilitate the proposed design workflow. It is the very first time that the algorithm mentioned above has been integrated in a widely used commercial modelling platform.*
3) *Optimizations to this existing mesh simplification algorithm were explored and achieved satisfactory results.*
4) *The newly proposed workflow were evaluated by design students and professionals in a design exploration workshop, and received very positive feedbacks.*

2 Review of Mesh Simplification Algorithms

In terms of the topology of outcome, there are three major categories of mesh processing algorithms that simplifies a mesh to triangle mesh, polygon mesh and free-form surfaces [22], [23]. In this thesis, a generalized concept of simplification is used, where it is defined as reducing the number of control units of a given complex digital form.

2.1 Triangle Oriented Methods

When targeting at a triangle output mesh, one of the most commonly used method is error-driven method. One of the early introduction of this method was by Hoppe et al. [24], where simplification is formulated as an optimization problem with an energy function. In 1997, a greedy error-driven mesh simplification algorithm was developed by Garland et al. [25]. They defined Quadric Error Metric as the minimizing target, and mesh decimation was performed in a greedy fashion where in each step, an edge with the least error value will be collapsed. This method was later widely used in many aspects of computer graphics and was further developed in various academic researches [26]–[28].

Apart from error-driven methods, simplification can also be achieve by a re-sampling routine which aims to build a simpler grid upon a given mesh. This can either be achieved by building a uniform grid [29] or some adaptive sampling method that either considering curvature [30], or direction fields [31].

Although the triangle oriented methods mentioned above can generate results that well approximate input meshes, it is very hard to gain direct control over the output of simplification algorithms. A majority of these algorithms focus on compression of an object for rendering or storage purposes, such that the downgrading of visual effect falls within a tolerable range. As a result, when used for design or modelling purposes, these algorithms either output denser results than human can handle, or cause significant deformation at small edge count.

Figure 11 Surface simplification using Quadric Error Metrics [25]

2.2 Polygon Oriented Methods

Re-sampling techniques can also generate evenly-spaced quad-based meshes. Dong et al. proposed a quadrilateral remeshing algorithm using harmonic functions [32]. The algorithm establishes an evenly-spaced grid on an arbitrary manifold that contains mostly quads. Iterative optimization algorithms such as Lloyd clustering [33] is also widely used in polygon-oriented simplification. Similar to the K-Means algorithm [34] which is heavily used in statistics, the most direct application of Lloyd clustering on meshes is to generate Centroidal Voronoi Tessellation (CVT) [35]. Given different distance metrics, different approximations of Voronoi Tessellation can be achieved: density-sampled [36], geodesic-based [37], surface-based [38], etc.

Using a different definition of energy function, Cohen-Steiner et al. proposed an error-driven simplification algorithm, Variational Shape Approximation, which uses normal metrics [39]. The algorithm casts the simplification problem as a variational partitioning problem where a set of planes (so-called proxies) are iteratively optimized using Lloyd's heuristic to minimize a predefined approximation error. With the distortion error defined as difference of face normal, the algorithm can generate simplification outputs that adhere to the geometric structure of the input shape. At lower face count, Lloyd's heuristic can ensure a much smaller deformation than greedy methods.

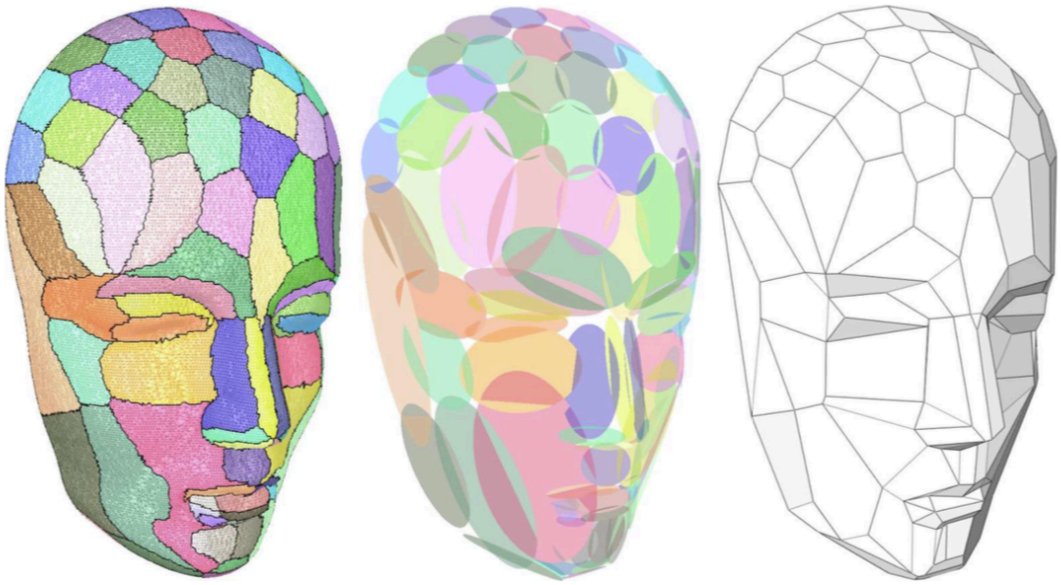

Figure 12 Variational Shape Approximation [39]

2.3 Free-form Oriented Methods

Apart from mesh oriented methods, another approach of 3D reconstruction that is widely

adopted by industry is the conversion from point cloud data to free-form surface representations such as NURBS [23].

The conversion typically involves two major steps, segmentation, and fitting. Given an arbitrary form as input, most algorithms first seek to segment it into a series shapes defined by four curved edges. For each segmented shape, NURBS fitting can be achieved with a two-step linear approach using simple least-squares fitting techniques [40]. Recent advancement in NURBS fitting includes fitting directly from 3D point clouds[41], increasing computation efficiency [42], improvement on fitting techniques [43], etc.

NURBS reconstruction of 3D scanned data can generally offer a good approximation of input via fewer number of faces, the consequence of which being increased number of control points associated with each face.

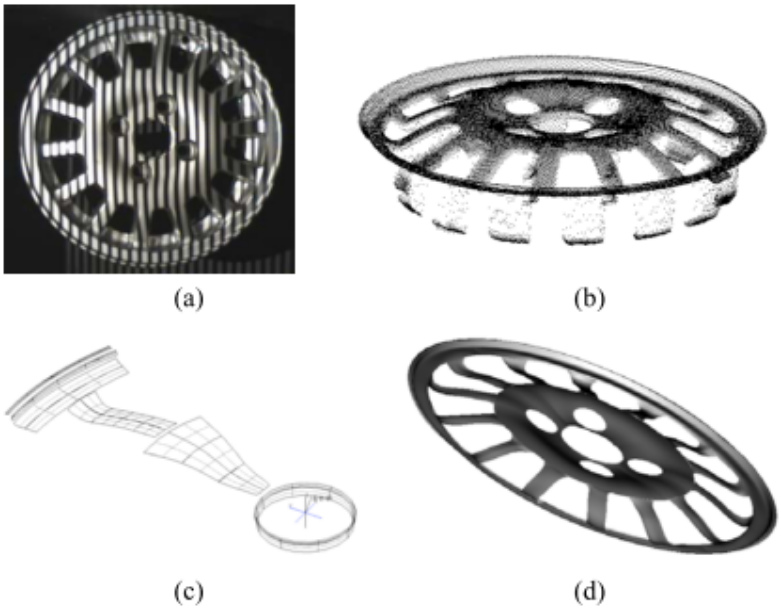

Figure 13 Process to create the CAD model of a free-form surface. (a) Optical acquisition of the wheel rim used as the target. (b) Range image of the top side of the wheel rim. (c) Generation of the paths for the digitization by using the CMM. (d) CAD model of the target. [9]

2.4 Commercially Available Solutions: Output and Editability

A preliminary research of existing modeling platform shows that most platforms use rectangular or polygon grid as the primary manipulation unit. This feature is favorable in the context of design, as it reduces the degree of manipulation. However, on most platforms, the provided simplification functions typically generate triangle based mesh, which is hard to manipulate for human because it either contains too many faces or suffers from severe deformation at lower face count. This gap between simplification outcome

and comfortable manipulation unit of human makes it very hard to directly model upon simplified 3D scanned models.

Table 1 Simplification Outcomes vs Comfortable Unit of Manipulation

Modelling Software	Manipulation Unit	Simplification Outcomes
SketchUp Pro 2018	Quadrilateral Faces	Triangle Mesh (Artisan Plugin)
Rhinoceros 6	Quadrilateral Control Grid	NURBS Surface and Triangle Mesh
Autodesk Maya 2018	Polygon Mesh	Triangle Mesh
Autodesk Revit 2018	Polygon Mesh	-
Autodesk Recap	-	Triangle Mesh
MeshLab 2016	-	Triangle Mesh
Geomagic Design X	Polygon Control Grid	NURBS Surface and Triangle Mesh
Polyworks 2017	Polygon Control Grid	NURBS Surface and Triangle Mesh

Mesh reduction is natively supported by various modelling software platforms. The following figure are screenshots of the mesh reduction function in Rhinoceros 6, a widely used modelling tool by designers. By reducing the dense input mesh to 500 polygons, it is still hard for users to manually manipulate on the simplified mesh.

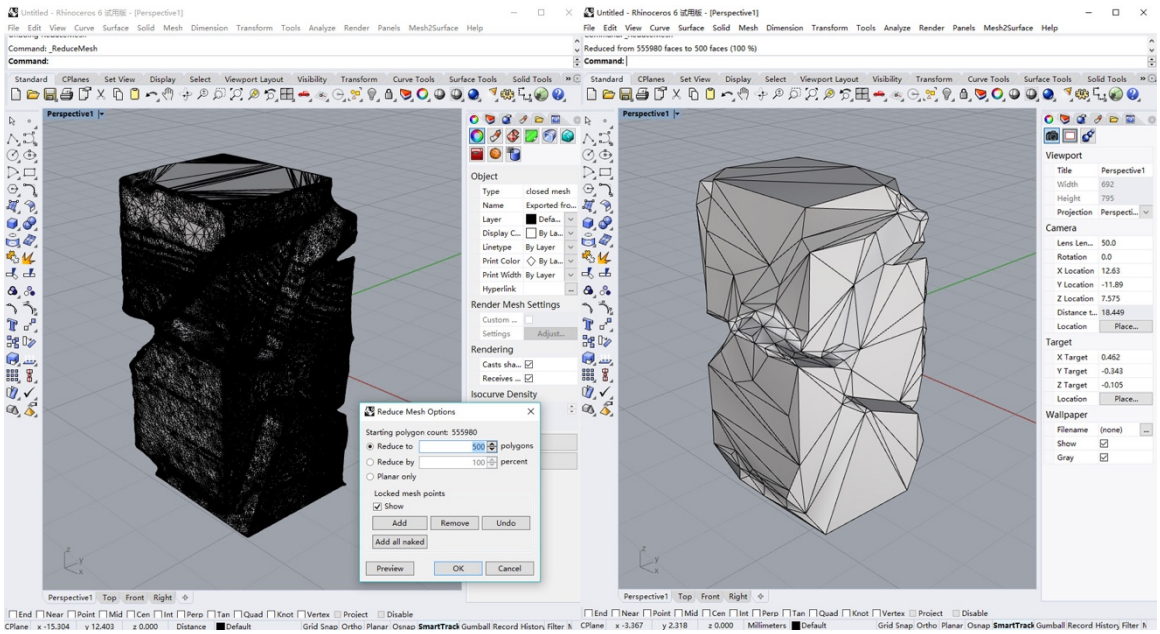

Figure 14 The native mesh reduction function in Rhino 6

Some platforms such as Polyworks, Geomagic Design X, and the mesh2surface plug-in for Rhino offer the option of converting dense input meshes to NURBS surfaces. As they are designed and optimized for free-form inputs, when tested against objects with linear contours, they are either labor intensive, or not following the geometric structure of the input.

The following figure shows in NURBS creating function of mesh2surface plug-in for Rhino. Although the plug-in outputs accurate NURBS surfaces, the creation process requires lots of manual adjustment. Surfaces has to be manually created one after another. The Polyworks software employs a similar workflow for NURBS creation.

Geomagic Design X is a powerful tool for reverse engineering on 3D scan data. It offers a variety of mesh decimation/segmentation approaches that are very handy. It takes a different approach in creating NURBS surfaces. The software automatically envelops the input mesh with a series of NURBS patches. However, the surfaces doesn't partition the inputs according to the edges of input objects, which makes them unsuitable for architectural design.

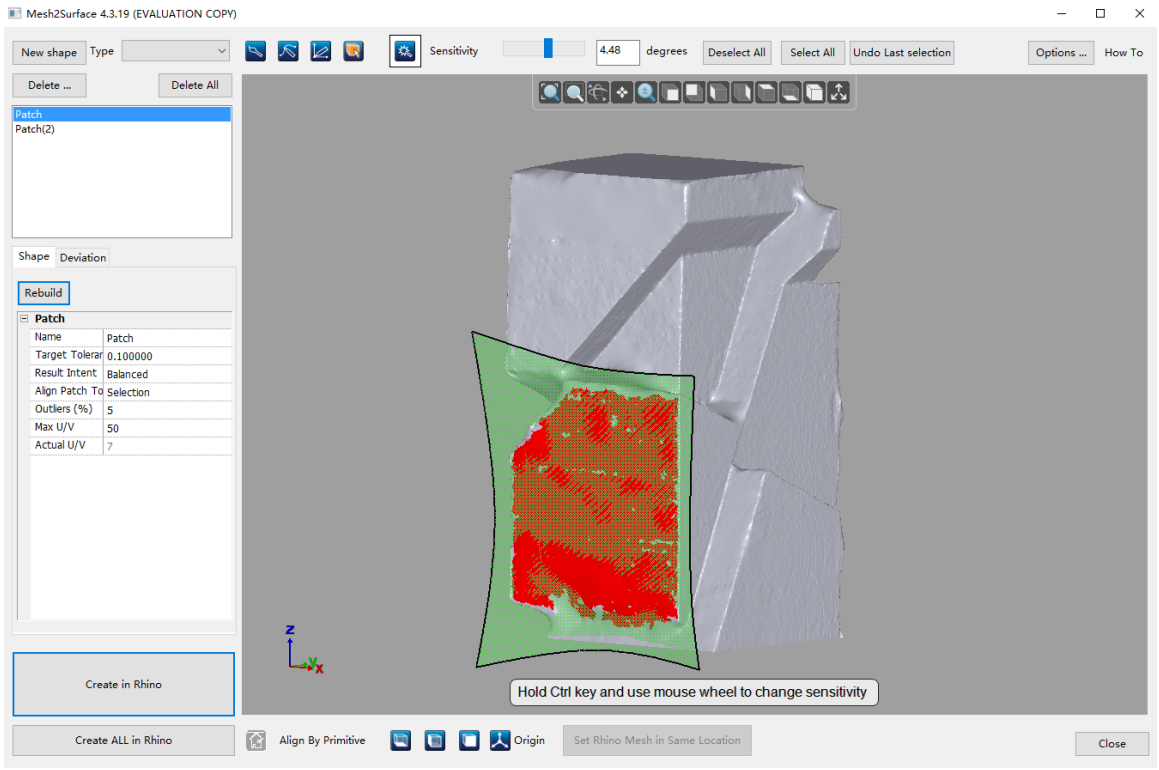

Figure 15 Surface creation in mesh2surface plug-in for Rhino

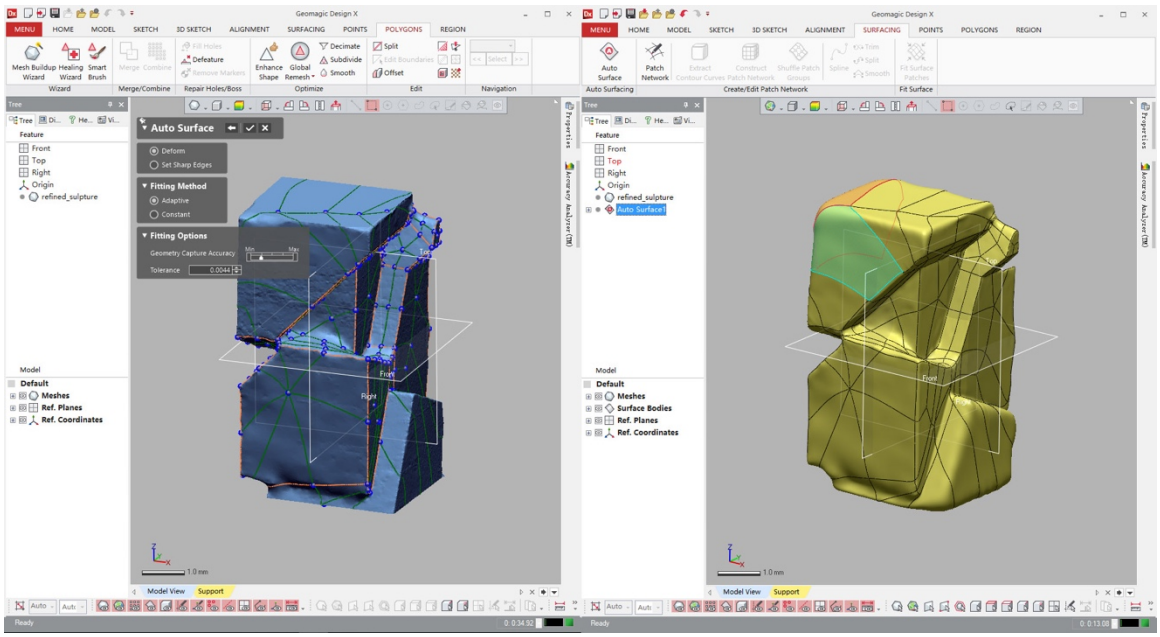

Figure 16 Automatic NURBS patching in Geomagic Design X

2.5 Variational Shape Approximation

Although frequently ignored by modelling software platforms, polygon oriented simplification methods are much more suitable for architectural design. Particularly, Variational Shape Approximation [39] has the most desirable features:

1) *It outputs polygon mesh, which is user-friendly to human editors.*
2) *It adheres to the geometric structure of the original object, using $L_{2,1}$ error metric.*
3) *It enables interactive simplification. User can point to parts of the original mesh to add/delete faces to/from the simplified result.*

2.5.1 Algorithm Overview

As described by Cohen-Steiner et al., the Variational Shape Approximation algorithm for simplifying meshes consists of two major steps: partitioning and meshing [39]. Given an input mesh, the algorithm will first partition it into regions (each represented by averaged centroid and normal, called proxy) and then generate a simplified mesh based on partitioning results. Partitioning is modeled as an optimization problem that seeks to minimize a defined error metric using the iterative Lloyd algorithm.

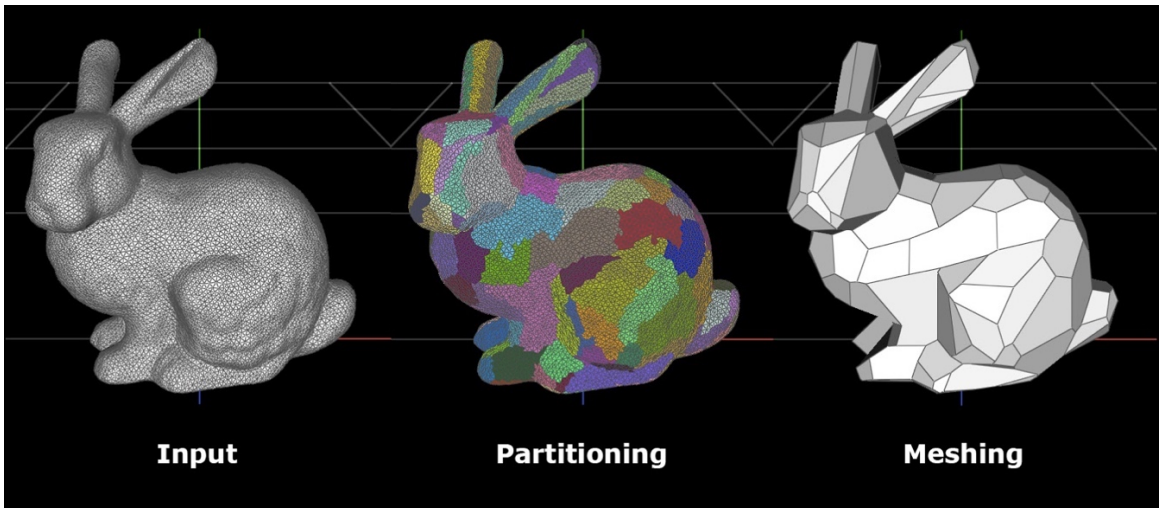

Figure 17 Steps of Variational Shape Approximation

2.5.2 Interactive Feature

The two-step approximation algorithm opens up possibilities of interaction. As an intermediate step, the partitioning result, which is represented as clusters of adjacent faces, gives a hint of how the final meshing result will be organized. Therefore, if a user want to have a finer grained level of control upon the final meshing result, the intermediate

partitioning step will be a perfect entry point for user interaction.

Cohen-Steiner et. al. described in their paper possible interactions between a user and the algorithm. Upon initialization, users are expected to input the desired number of regions (proxies) intended for a mesh. It is quite possible that a first choice of parameter is not a good one, or proxies get stuck in a local minimum. Cohen-Steiner et. al. introduced two interactive operations, add and delete based on certain heuristics. When told to add or delete a proxy, the algorithm automatically finds a face with largest distortion error and creates a new proxy or automatically deletes a proxy with minimum total distortion error. They claimed that with this level of interaction, it is very easy to obtain a good partition in just a few seconds. [39]

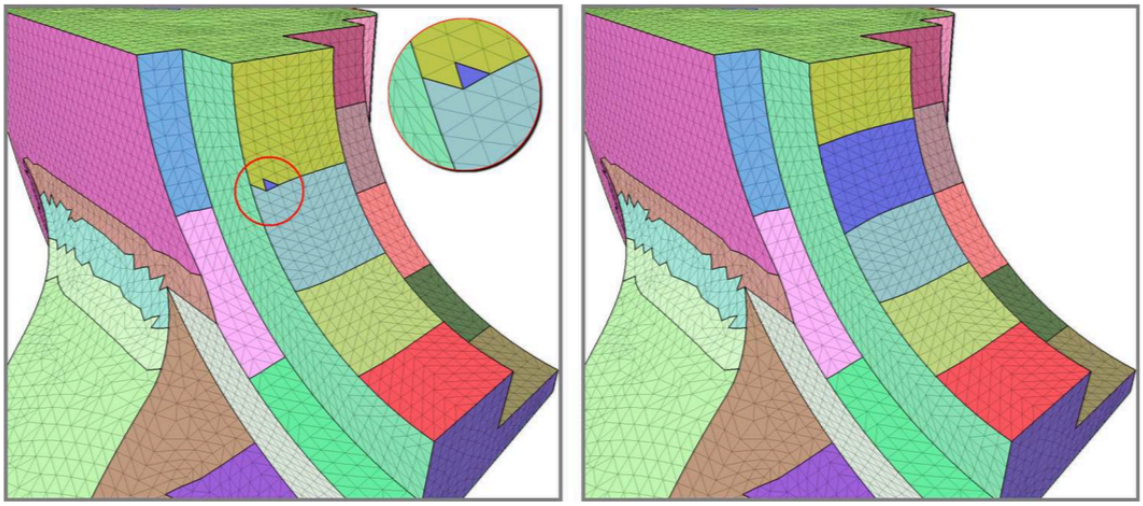

Figure 18 Interactive heuristics developed by Cohen-Steiner et.al. : “When the user interactively adds a proxy, a seed triangle is placed in the worst-approximated region (left), and the next iteration allows a new region to quickly grow (right).” [39]

2.5.3 Challenges in Application

Although Variational Shape Approximation seems to be an excellent solution of mesh simplification, current problems with this algorithm are that:

1) *There are very few modeling software that implements the Variational Shape Approximation algorithm.*

   Two open source implementation of this algorithm were found. MEPP, a 3D modeling platform that targets 3D developers [44]*, supports only the partitioning step of the algorithm. MeshLab* [45]*, a well-known mesh processing software has it included in the experiment folder unfinished.*

2) *There are hidden bugs and edge cases that have not been addressed either by the authors of this algorithm, or by developers who implemented this algorithm.*

3) *This algorithm is proved slower than other greedy mesh decimation algorithms because it involves multiple iterations, which indicates potential performance issue.*

   As Variational Shape Approximation is an algorithm based on iterative optimization, it can be much slower than other greedy mesh decimation algorithms. Although Cohen-Steiner et al. claimed their flooding algorithm to be efficient with O(NlogN) complexity, it can be much slower in practice compare to greedy algorithms such as edge collapse. In order to operate on large inputs and support interactive simplification with reasonable computation time, acceleration of the original algorithm is necessary.

4) *The level of interaction is still rather coarse since user cannot have direct control over the output.*

3 Methods

3.1 Implementing Variational Shape Approximation

The first part of building a mesh simplification tool is to implement the desired algorithm. As mentioned previously, there are very few open-source or commercial implementations for Variational Shape Approximation, implementing this algorithm is a major part of this thesis project.

3.1.1 Implementing the Partitioning Step

Given an input mesh, the algorithm first tries to partition it into different clusters. Each cluster will later be built into one single polygon in the meshing step. Partitions are determined by a very simple rule: ***adjacent faces that have similar orientation will be put into the same cluster.*** This rule guarantees that simplification outcome will best preserve the initial geometric structure of the input mesh.

The partitioning step contains three sub-routines: initialization, distortion minimizing flooding, and proxy fitting.

1) *Initialization*

   At first, once the algorithm is given the number of partitions (aka. proxies) desired, it seeks to randomly sample from all faces of the input mesh for a number of seed faces, one for each proxy. In this thesis project, random initialization is used. It should be noted that other initialization methods such as furthest point initialization can also be applied to improve performance. [39]

2) *Distortion Minimizing Flooding*

   The actual assignment of clusters to faces was achieved by a procedure called distortion minimizing flooding, where distortion is defined as the difference of normal vectors between a certain face to the seed face of some proxy. From each seed face, the face with least distortion error from all surrounding faces will be repeatedly added into the proxy represented by the seed face. The procedure is completed with a global priority queue data structure.

3) *Proxy Fitting*

   After all faces in the mesh have been assigned a proxy, the average normal of each proxy is calculated and used to update the choice of the seed face. For each proxy,

the face with the closest normal to the proxy's average normal will be chosen as the new seed face for this proxy.

After fitting all proxies, the algorithm will go back to a new iteration of flooding to reflect on the changes made by proxy fitting. The entire procedure will be repeated for several iterations until the algorithm converges or the limit for maximum number of iterations is hit.

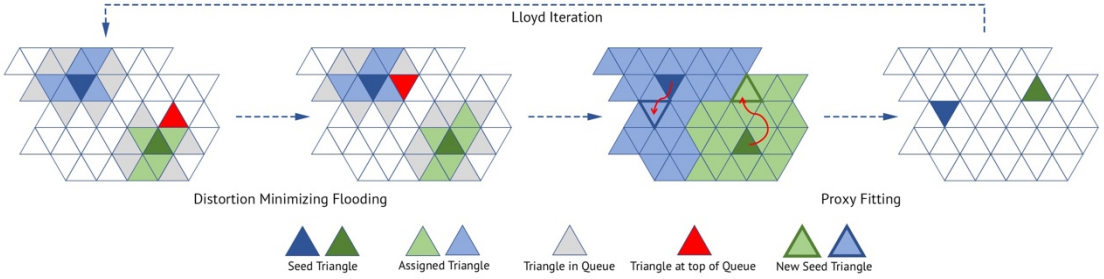

Figure 19 The partitioning procedure

```
VSA_Partitioning():
while not converge:
    Flooding()    // distortion minimizing flooding
    Fitting()     // proxy fitting and region updating

Flooding():
  define T[NUM_FACES] as an array of Triangle faces
  define R[NUM_REGIONS] as an array of Regions
  define Q<Triangle, Error> as the global priority queue
  for each Ri in R:
    Ri.seed = pick a unique Tj from T at random
    Tj.assignedRegion = Ri
    for each t in Tj.neighbors():
      t.potentialAssignment = Ri
      insert (t, distortionError(t, Tj)) to Q
  while Q is not empty:
    Tj = pop from Q with least distortion error
    if Tj.assignedRegion exists:
      continue
    else:
      r = Tj.potentialAssignment
      Tj.assignedRegion = r
      for each unassigned t in Tj. Neighbors():
        t.potentialAssignment =  r
      insert (t, distortionError(t, r.seed)) to Q

Fitting():
  for each Ri in R:
    T' = all t in T s.t. t.assignedRegion == Ri
    Ri.normal = area-weighted average normal of T'
    Ri.seed = pick t in T' with minimum ||t.normal - Ri.normal||
```

Figure 20 Pseudo-code for Sequential Partition in Variational Shape Approximation

3.1.2 Implementing Meshing Routines

Meshing step is computed upon partitioning result to produce editable meshes. Cohen-Steiner et al. proposed an outline of a 4-step meshing algorithm in [39]:

1) Create at least three anchor vertices for each proxy(region)

2) *Extract and subdivide edges*

3) *Constrained Delaunay Triangulation*

4) *Clean up useless edges and create polygons*

Because it is guaranteed that all faces are connected inside a proxy, each proxy can be transformed into a polygon face in the new mesh. Furthermore, there is no constraint set by the algorithm about how many vertices a polygon may have, the algorithm can output an arbitrary number of vertices for any given polygon face.

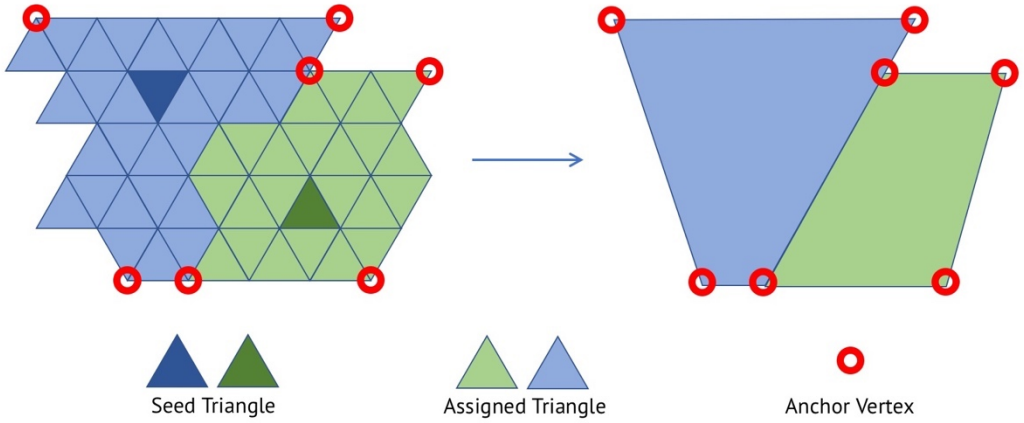

Figure 21 Meshing Step

3.1.3 Interactive Simplification

Cohen-Steiner et al. proposed an error-driven heuristic to allow interactive insertion and deletion of partitioned regions. When a user wants to adjust the number of partitions, the algorithm automatically inserts (or deletes) a region that has most (or least, respectively) distortion error. This method gives users a coarse level of control over simplification results.

Although the coarse level of interaction may be enough for academic researches, designers in production might prefer a more direct control over the process that enables creativity. In order to provide a fine-grained control over the simplification process, we further propose three types of manual manipulation over the partitioning process:

1) *Inserting a region: a new region is grown from the seed face selected by user, and inserted to the set of regions.*
2) *Deleting a region: user selects a face to delete its associated region from the set of regions.*
3) *Painting a region (to be supported in next release): a new region is directly created*

from a group of connected faces manually selected by user, and added to the set of regions.

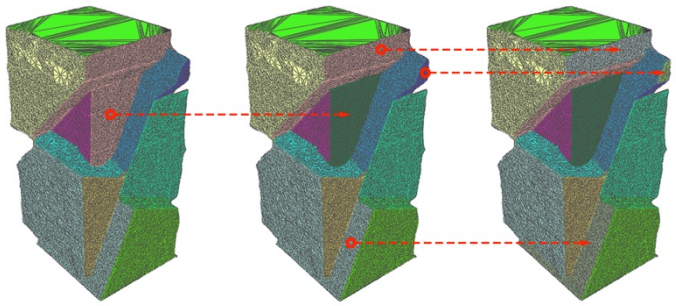

Figure 22 Region Insertion on a 3D scanned model

Each of the interactive manipulation above (except for region painting) requires at least one step of distortion minimizing flooding, which can greatly benefit from the acceleration techniques described above. Computation time between user actions can therefore be minimized to ensure good user experience.

3.2 iSimp Plug-in for Autodesk Maya

The very first prototype of the mesh simplification algorithm (based on Variational Shape Approximation) was implemented and tested on Scotty3D, an experimental modelling environment from CMU Computer Graphics course. Although it is a good platform for developing and testing graphics algorithms, it is not suitable for production. In the case of this thesis, as we are targeting the design community, a prototype on a test environment will not be enough for benefiting the design community. Therefore, as part of the thesis project, this simplification algorithm is further implemented on a commercial 3D modelling platform that was widely used by the design community: Autodesk Maya.

The iSimp (which stands for “interactive simplification”) plugin for Maya is an implementation for Variational Shape Approximation algorithm on the Maya platform. It is designed to be efficient, stable and user friendly.

3.2.1 Why Maya?

Autodesk Maya is a 3D computer animation software with powerful modeling tools for artists, modelers and animators. [46] It has a vast user community of users and designers and artists, who are targeted as the potential beneficiaries of this thesis. Despite its user community, the many internal features of Maya also make it the ideal platform to

implement the simplification routine:

1) *It has a comprehensive support of polygon meshes.*

   This is the most important feature for implementing Variational Shape Approximation. As the algorithm outputs polygon meshes, the modeling software must natively support polygon meshes. As in the case of Maya, meshes can be composed by polygons with arbitrary edges, while in other modelling software such as Rhino or SketchUp, only triangle or quad meshes are natively supported. Maya's native support of polygon mesh guarantees editability of meshing outputs.

2) *It has a well-documented API reference.*

   Maya plugins can be written in a variety of programming languages such as C++, Python and MEL (the Maya scripting language). For each language, Autodesk has prepared a well-documented API reference that can be very useful for developers.

3) *The Dependency Graph architecture enables parametric simplification.* [47]

   Maya's Dependency Graph architecture represents mesh modifying operations as a series of connected nodes. Each node can have a list of tweakable parameters that can be used to dynamically change modification results. This way this feature works is very similar to Grasshopper's programming batteries, which enables parametric simplification and design.

3.2.2 Core Functionalities

The iSimp plug-in for Maya implements all steps of Variational Shape Approximation, and interactive simplification features such as add and delete proxy. Painting proxies are currently not supported and is intended to be supported in future releases.

3.2.2.1 Partitioning

On a given mesh, partitioning step can invoked by the following steps on the Maya software:

1) *Select an arbitrary face from the model to be simplified.*

2) *Type the following command into the Maya command line interface:*

```
isimp init [-p <number of partitions>]
```

The desired number of partitions can either be specified using the -p flag from the command line interface, or changed later from the node attribute editor. After partitioning,

all faces of the mesh will be assigned different colors, where each color represents a different proxy.

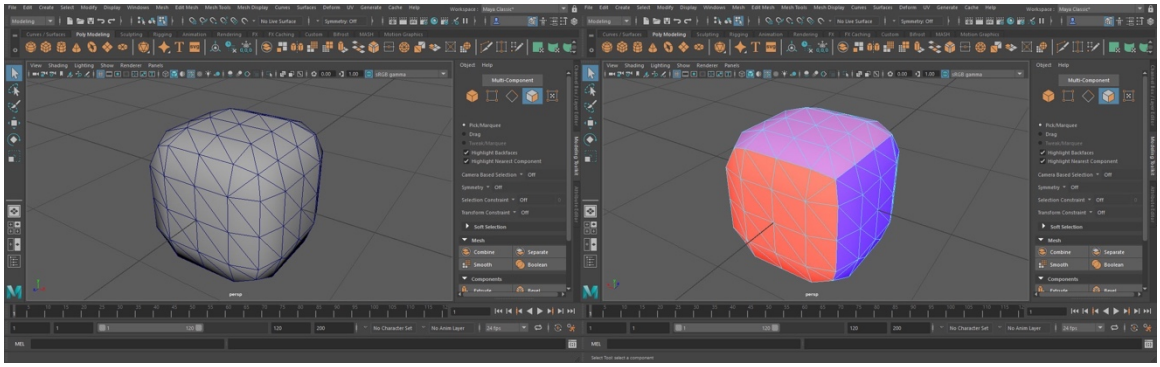

Figure 23 Partitioning a smoothened cube into 6 proxies. The result accurately reflects the six sides of the original cube.

3.2.2.2 Add/Delete Proxy

After an initial partitioning is computed on a given mesh, it is now up to the user to decide whether it is a satisfactory partitioning or not. The user may then decide to make further modifications directly on the partitioning result. Two interactive operations are provided: add a proxy and delete a proxy.

For add:

1) *Select a face that you wish to create a new proxy on.*
2) *Type the following command into the Maya command line interface:*

```
isimp add
```

For delete:

1) *Select a face in the proxy you wish to delete.*
2) *Type the following command into the Maya command line interface:*

```
isimp del
```

These two operations can be performed on the initial partitioning for an arbitrary number of times until the user is satisfied with the partitioning result.

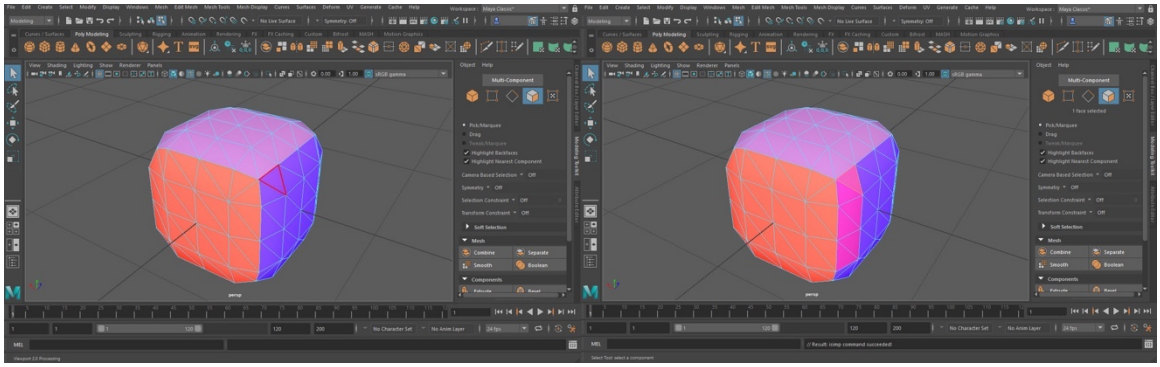

Figure 24 Interactively adding a proxy by selecting a face on the mesh and making it the seed face for the new proxy.

3.2.2.3 Meshing

Once the user is finished with the partitioning step, he/she can ask the plug-in to generate a simplified mesh based on the result of the partitioning step. To invoke the meshing step:

1) *Select an arbitrary face from the model to be simplified.*

2) *Type the following command into the Maya command line interface:*

```
isimp mesh
```

As we can see from the figure below, a triangle is inserted on the edge of the smoothened cube, where a pink proxy was previously located.

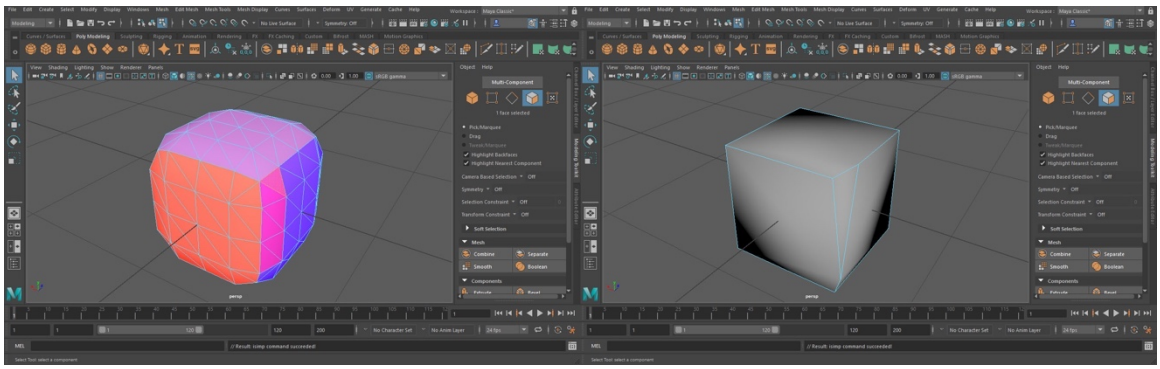

Figure 25 Meshing step: generate a simplified mesh from the partitioning result.

3.2.3 Plug-in Design

The iSimp plug-in implements Maya's `polyModifierNode` class, and is designed to be integrated into Maya's Dependency Graph architecture. Each of the functions will create a modifier node of its own type to process the input mesh.

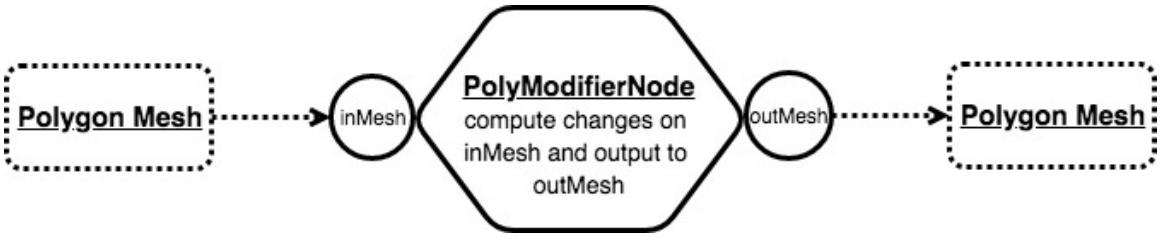

Figure 26 A generic polyModifierNode instance

Based on this node connection pattern, many instantiated nodes can connect together to perform a series of operations on an input mesh. In the case of iSimp plug-in, two or more nodes are needed to complete the entire simplification operation.

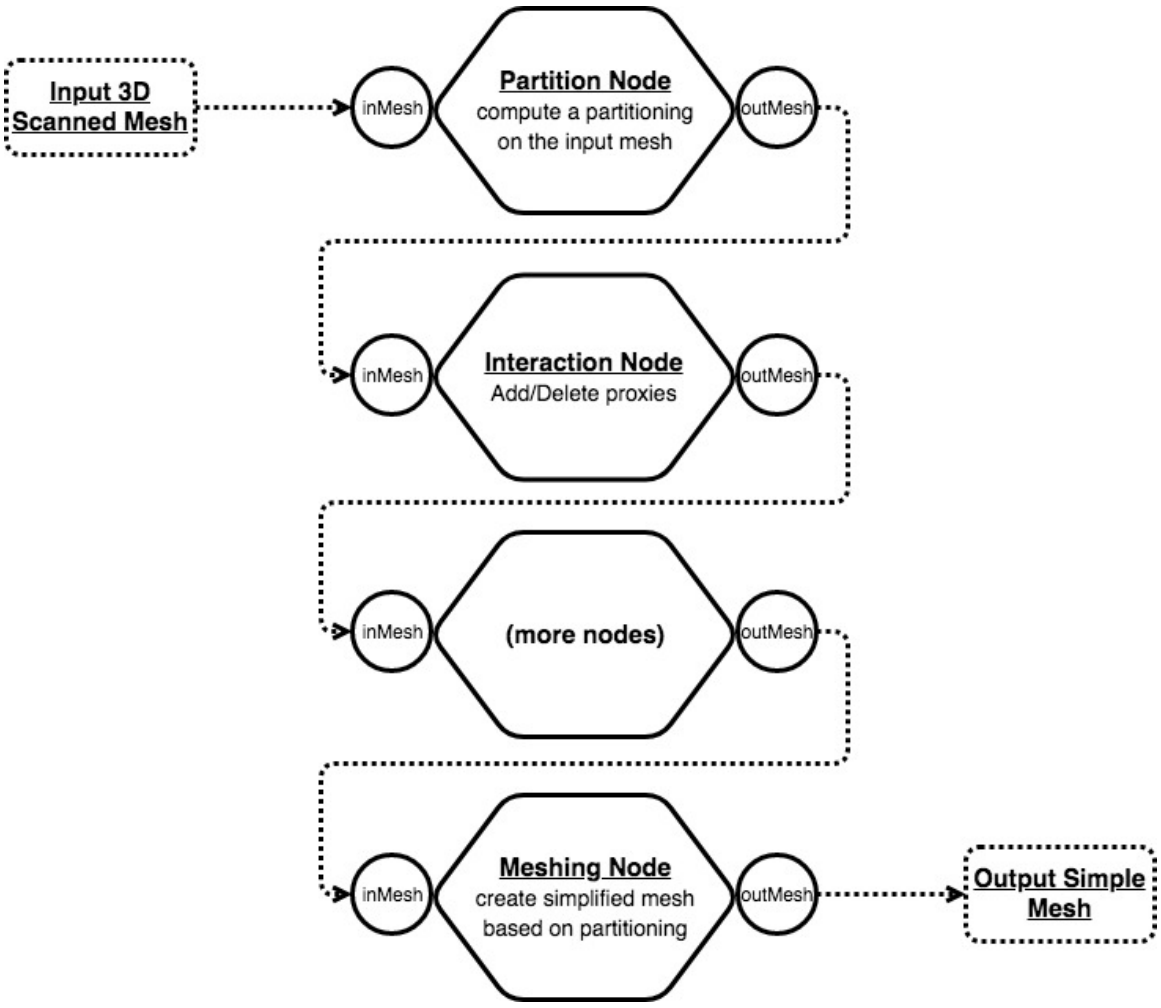

Figure 27 Typical node hierarchy for iSimp plug-in

Maya's Dependency Graph architecture allows dynamic changes on parameters of any intermediate node. Once the parameter of any node is changed, the chain of operations will be re-computed to let dynamically reflect the changes made.

With Maya's Dependency Graph architecture, it is easy to chain up different mesh modifications, dynamically change parameters, and support undo/redo operations, making the plug-in a user-friendly tool.

4 Results and Discussion

4.1 Application Scenario: Wooden Sculpture

In this section, a typical application case is presented as an example of the proposed 3D reconstruction oriented design workflow. We will show how to smoothly transfer from a physical model to a digital design space with the help of Autodesk Recap Photo for photogrammetry model generation and our newly developed iSimp plug-in for Maya for post-processing.

4.1.1 Creating Digital Model with Photogrammetry

As the very first step, we start from a physical (wooden) model and create a digital representation of the physical model. We took a series of photos of the model using a mobile phone in an ordinary living room and upload the photos to the Autodesk Recap Photo software for photogrammetry computing.

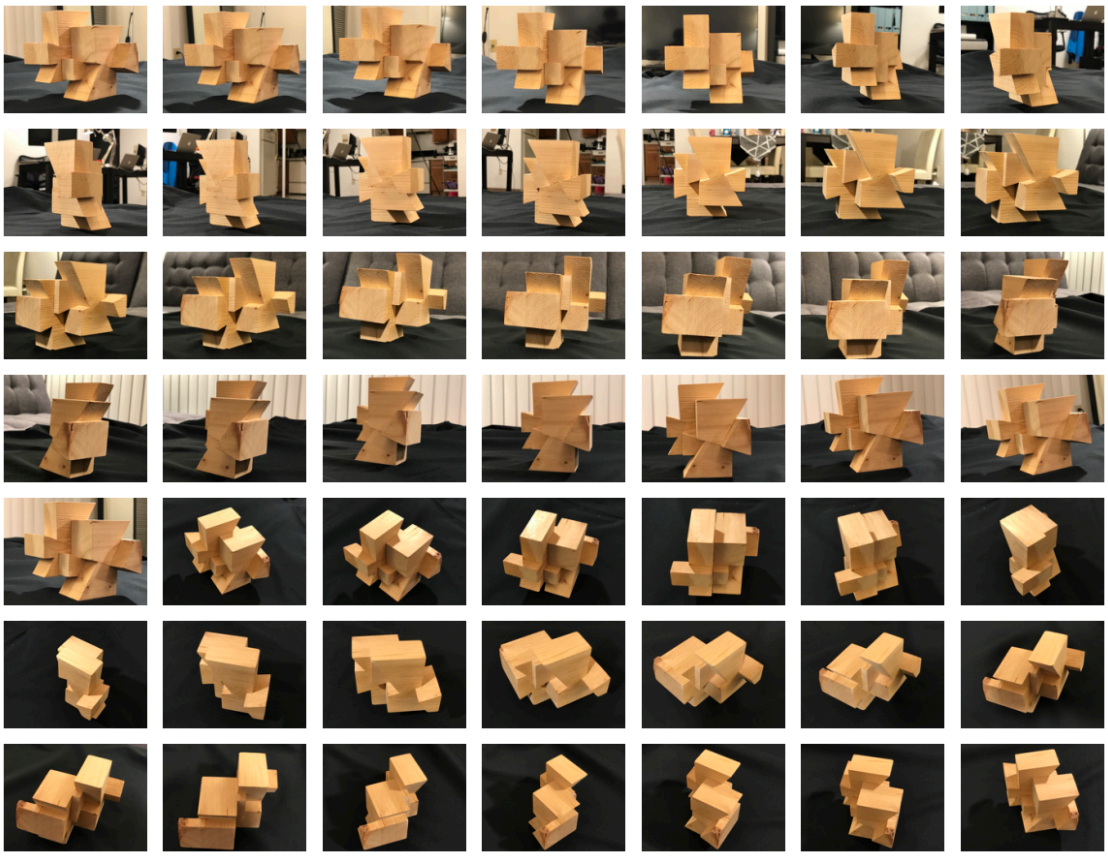

Figure 28 Photos taken of the physical model

Once the photogrammetry part is completed on the Autodesk Cloud Service, we download and open the project in Autodesk Recap Photo.

Figure 29 Downloaded photogrammetry project in Autodesk Recap Photo

As a guideline for iSimp plug-in, input meshes have to be connected, two-manifold and without self-intersection. A series of post processing steps are taken to prepare the model before importing it into Maya for interactive simplification and editing:

1) *Re-orient the model.*
2) *Remove unnecessary parts.*
3) *Slice and fill holes.*
4) *Auto detect model issues.*
5) *Export to universal 3D format, e.g. OBJ, DAE, etc.*

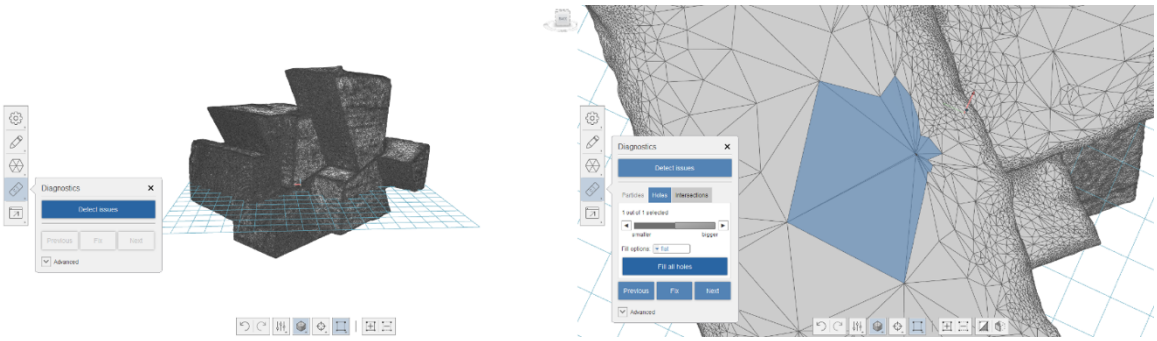

Figure 30 Auto-fixing model issues in Autodesk Recap Photo

4.1.2 Interactive Simplification and Further Editing

Once we get a connected, two-manifold mesh from the photogrammetry software, we can import the mesh into Autodesk Maya and use the iSimp plug-in to start the post-processing procedure.

In this example of interactive simplification, our goal is to reconstruct the model exactly as the physical one. We will use interactive operations to guide the algorithm to recognize as many flat faces as possible.

After getting a simplified model, we can utilize the various modelling functions Maya provides, such as bevel, extrusion, rotation, stretching, etc., to further modify the model.

The following figure shows the steps of interactive simplification and further editing. The modified model shown in Step 5 is created upon Step 4 by a participant from the 3D Scan Oriented Design Workshop.

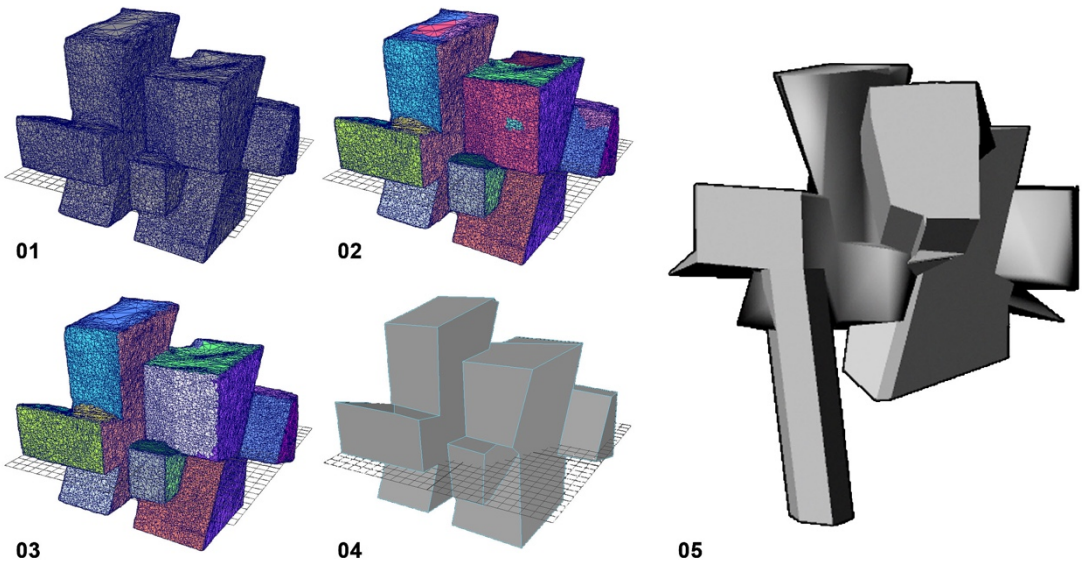

Figure 31 Steps of interactive simplification and editing. 01: Decimated input mesh. 02: Initial partitioning. 03: Refined partitioning through interactive simplification, notice the deletion of island proxies and two proxies sharing a planar face. 04: Simplified polygon mesh. 05: Further modification, design upon the simplified model.

4.2 User Evaluation

To further evaluate how this interactive model simplification tool can benefit the design community, a workshop is held to let interested designers try out this new Maya plug-in and offer feedbacks. Participants to this workshop are undergraduate and graduate students in the school of architecture. There were a total of 10 participants.

4.2.1 The 3D Scan Oriented Design Workshop

The 3D Scan Oriented Design Workshop is designed to let interested participants experience the entire 3D scan process using photogrammetry and a new interactive mesh simplification tool to create designer-friendly models. Participants gained hands-on experience working with 3D scan and interactive simplification of complex meshes, and had an opportunity to express their creativity by directly modify on the simplified digital model.

Anonymous feedbacks on the Maya iSimp plug-in were collected to further determine how this plug-in can help the architectural design process and how it can inspire design thinking.

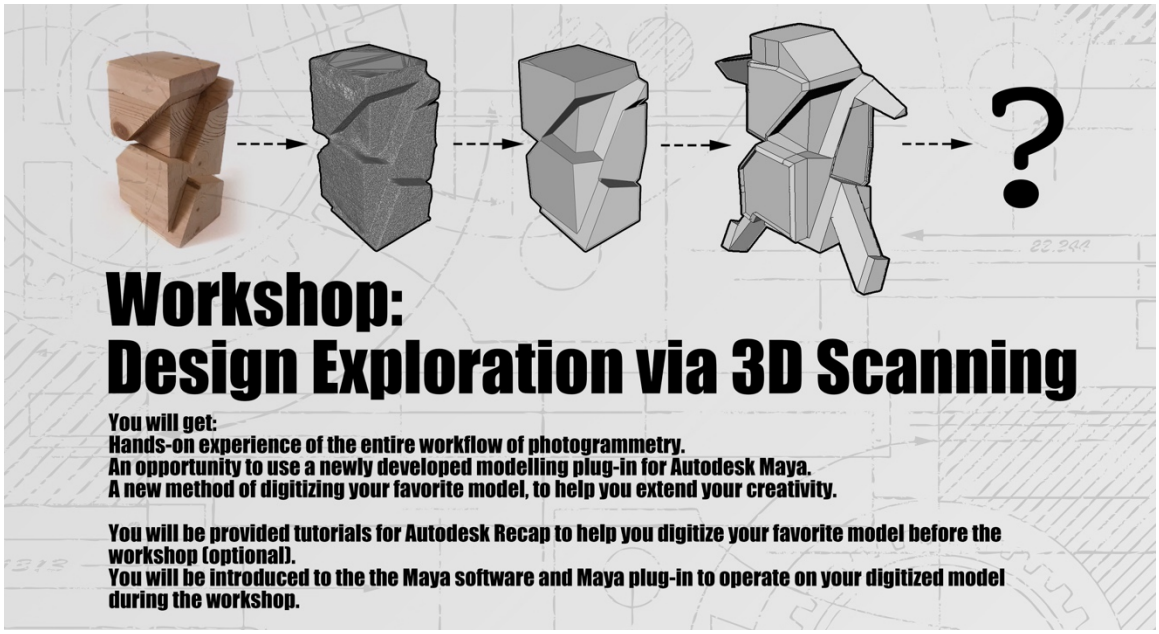

Figure 32 Workshop Poster

The workshop was divided into these steps:

1) *Tutorial of the plugin.*

A 10-minute demonstration of how to use the plugin to interactively simplify a 3D scanned dense mesh.

2) *Model simplification.*

   Participants were encouraged to use the plugin to simplify their own model file and generate an output.

3) *Get creative.*

   Participants were encouraged to be creative and add some modifications/details to the simplified model.

4) *Feedback gathering.*

   All participants filled out a short feedback form describing their experience with the entire design workflow.

4.2.2 Outcomes from the Workshop

Participants of the workshop, through interaction with their models and the software itself, discovered different potential application scenarios with the iSimp Maya plugin. Generally speaking, the plug-in has been used both as a mesh simplification tool and a tool for design exploration.

4.2.2.1 iSimp as a Mesh Simplification Tool

Just as any mesh decimation algorithm, Variation Shape Approximation has to the potential of being used as a mesh simplification tool.

One of the participants from the workshop, who has been working with creating 3D models of human characters with photogrammetry claimed that iSimp can be very helpful for the post-processing of her motion capture and rigging workflow. As most game engines, such as Unity3D, limit the total number of triangles for each import, the Maya plug-in can be an alternative approach to simplify 3D scanned meshes.

She was very excited to see the simplification outcome at both high and low proxy counts.

> *"It is exciting to still be able to recognize the human figure with so few number of polygons." – Workshop Participant*

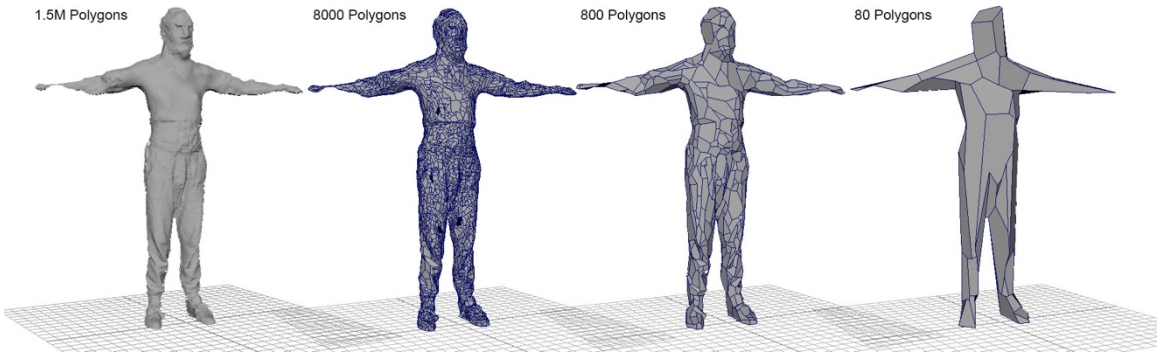

Figure 33 Simplification of human character by anonymous workshop participant

Another participant used iSimp on an urban scale 3D scan. Throughout his interaction with the iSimp plug-in, he was delighted to see how this tool is recognizing street façades as planar surfaces and simplifying them into single polygons. He claims this feature to be very desirable because most simplification tools will not generate results based on geometric structure of the city. While he is mostly using urban 3D scan data for representational purposes, he thinks that the iSimp plug-in can be of great help for increasing design possibilities of 3D scanning.

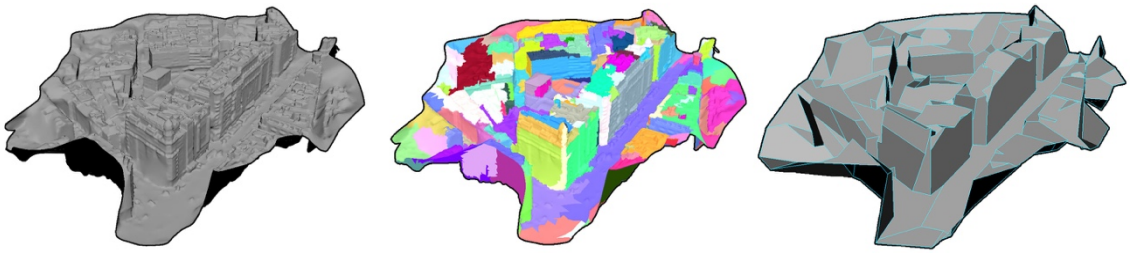

Figure 34 Simplification of urban 3D scan model by anonymous workshop participant

4.2.2.2 iSimp for Design Exploration

Apart from simplification applications, many participants of the workshop explored various interactive design approaches with Autodesk Maya and the iSimp plug-in.

For the interactive simplification part, many participants claimed that it is an efficient, accurate and intuitive process. They learned to use the iSimp plug-in in less than 10 minutes. Many participants liked the fact that the plug-in offers them more direct control over the simplification process.

"Considering that I have knowledge of 3D scanning, it is quite interesting how simply everything turns into surfaces." – Workshop Participant

Once mesh simplification is finished, participants explored many different approaches to further edit the model. One very popular design technique was to utilize the modelling functions provided by Maya, such as extrude, bevel and subdivide. Many participants incorporated this approach in their design exploration and produced very interesting results.

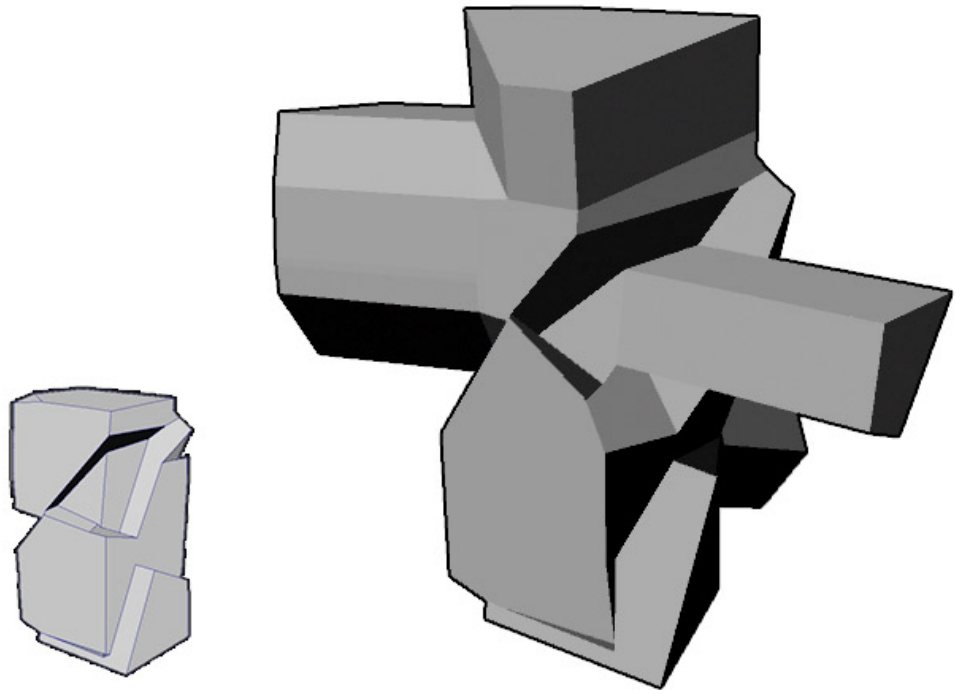

Figure 35 Straight-forward editing on simplified model by anonymous workshop participant

One participant, in particular, used a very different approach to explore the possibilities for creating space with iSimp plug-in and the Maya software. He started from stretching different faces of the simplified mesh so aggressively that defects started to appear. He then used the face deletion tool to delete the "bad" faces resulted from the stretching, trying to keep the mesh closed and without self-intersection. After repeating these steps for several times, he will then try to fill the various holes generated in the deletion using the Maya hole filling tool. It is very interesting that the result he produced looks astonishing and entirely different from the original model.

Designing through cleaning and deletion is the approach to design emerged from this participant's interaction with iSimp plug-in and the Maya software.

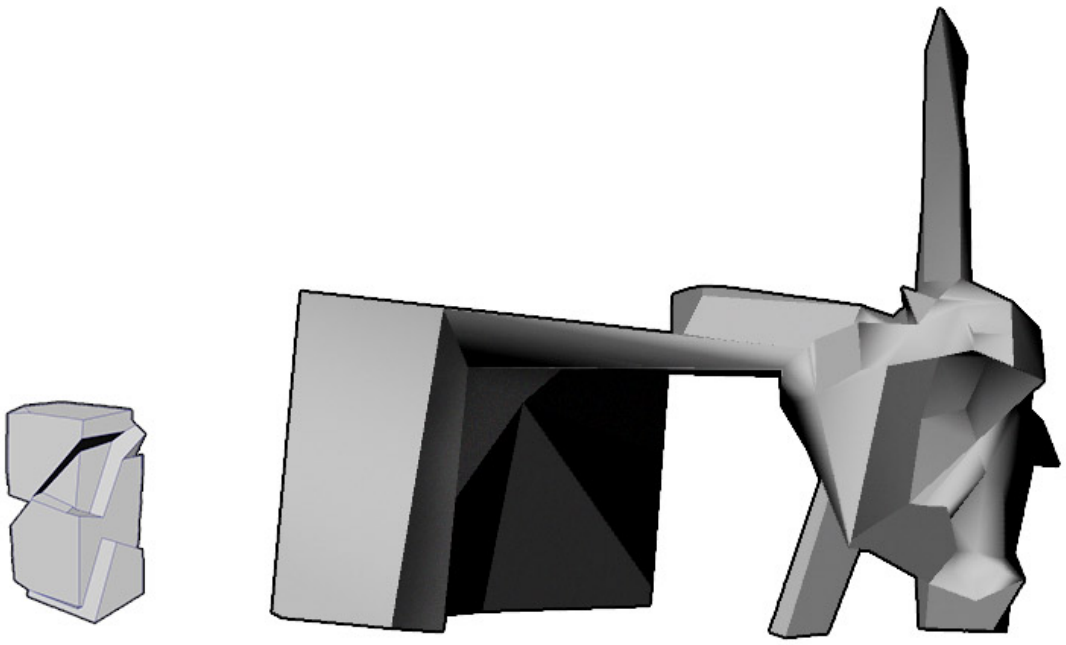

Figure 36 Model editing via deletion and hole-filling by anonymous workshop participant

4.2.3 User Feedbacks

In general, the iSimp plug-in for Maya is well welcomed by the sample of design students attending the workshop. It is well agreed that the iSimp plug-in can perform quick cleaning with high accuracy and without losing form controls. The fact that users can control polygon output of mesh simplification interactively and intuitively is appreciated by many workshop participants.

Many suggestions were given by workshop participants for future improvement needed by iSimp plug-in, including:

1) *Add graphical user interface.*

   The current command line interface is counter-intuitive and not user-friendly for designers. Despite buttons for interactive functions, the interface should also provide manipulation tools such as paint brush to allow direct user interaction.

2) *Increase interactivity.*

   In addition to the add/delete functions that are currently supported by iSimp Maya plug-in, more interactive operations should be incorporated to extend designer's control over the simplification process. Possible interactions include: weight painting

to allow non-uniform initialization, preview of meshing results to fit designer's intuition, etc.

3) *Add UV texture support.*

   In the case of animating 3D scanned human characters, texture is an essential part for simplification results.

4) *Automatic determination of number of proxies.*

   Some participants complained that it is difficult to determine the initial parameters, such as number of proxies, for the partitioning procedure. They would prefer automatic determination of number of proxies by the software.

5) *Adaptive simplification.*

   For some objects that have high frequency features, e.g., the face part on a human model, it would be better practice to develop an adaptive simplification feature to adaptively add more proxies to the high frequency area of the model.

6) *Higher robustness for trickier meshes.*

   Currently, the plug-in does not work well on meshes that have disconnected parts. It is desired that robustness should be increased on "trickier" disconnected meshes.

5 Conclusion

5.1 Contributions

In this thesis, we have proposed, developed and tested a 3D scan oriented workflow that aims to help designers smoothly transfer from designing based on physical models to designing in the digital world.

We have re-discovered Variational Shape Approximation, a mesh simplification algorithm that is capable of producing editable outputs from dense meshes. A thorough implementation of the algorithm is completed and various edge cases and bugs were addressed and solved for the first time in the open-source community. Three approaches were explored to accelerate this algorithm and achieved satisfactory results. (See Appendix)

iSimp, an interactive mesh simplification plug-in was developed for the Autodesk Maya modelling platform. It is the very first time that Variational Shape Approximation has been integrated in a widely used commercial modelling platform to benefit the design community.

A design workshop was held to let designers test this new mesh simplification plug-in and get first-hand user feedback. The interactive simplification functions are liked by all 10 participants, and interesting new design approaches have emerged from interaction with the iSimp plugin and Maya software.

The 3D scan oriented design workflow, which was proposed in this thesis, was brought to reality and proven beneficiary to the design community.

5.2 Limitations and Future Works

Despite the good feedbacks from user evaluation, the mesh simplification approach developed in this thesis is not a universal solution. Determined by the underlying algorithm, the interactive simplification approach provided by the iSimp Maya plug-in works best for polygonal shapes. Objects that contain linear components, such as flat surfaces, straight edges, are best suited for this algorithm. For curved surfaces, the algorithm will provide results with limited editability as a curved surface will be approximated by an arbitrary number of polygons. For simplification and reverse engineering of curved objects, it is better to utilize tools such as Mesh2Surface for Rhino, Polyworks or Geomagic Design X.

Although the 3D scan oriented design workflow has been developed and tested, it is still a preliminary prototype that needs improvements. Future works regarding this workflow should be focused on the following aspects:

1) *More applications is needed for this design workflow to thorough assess its contribution. Applications should come from different areas of design, e.g., industrial design, gaming, animation, etc. We should keep gathering user feedbacks especially those from users with a design background to optimize the plug-in for the design community.*

2) *The iSimp Maya plug-in should be continually developed to increase robustness, improve user experience and add more necessary functionality. The plug-in should also be released to the Maya Store for more publicity.*

3) *Optimization for Variational Shape Approximation should also be continually explored. The focus could be developing new approaches of parallelization on GPUs as well as on multi-core CPU platforms.*

6 Appendix: Algorithm Optimization

6.1 Optimizing Distortion Minimizing Flooding

A preliminary experiment on the original Variational Shape Approximation described in [39] shows that partitioning takes more than 95% of total computation time, while meshing only takes less than 5%.

Inside partitioning routine, flooding and fitting are very different steps that should be treated differently. Taking advantage of recent development of parallel computer architecture and general-purpose GPU computing, acceleration can be achieved by exploiting parallelism in the sequential algorithm.

The proxy fitting procedure of Variational Shape Approximation is very similar to the cluster update procedure of K-Means algorithm [34]. The type of computation involved exhibits great data parallel property so that it can be easily divided and distributed among a large number of processors. The problem and its solutions have been covered in previous researches such as [38], [48].

Distortion Minimizing Flooding is a more complicated case. Based on the original description by Cohen-Steiner et al., flooding is performed by constantly adding and removing faces from a global priority queue. The presence of the priority queue is crucial to ensuring convergence and preserving connectivity of regions. However, it is great obstacle to parallelization. High contention can be observed when scaling to multiple processors.

We will present two GPU parallel approaches that replace the sequential priority queue by introducing additional computation. We cannot expect the exact behavior of the sequential algorithm from a priority-queue free parallel approach. There are performance-accuracy tradeoffs in parallelizing distortion minimizing flooding.

6.1.1 Naïve Data-parallel Flooding on GPU

A first naïve approach to parallelizing the flooding routine is to eliminate the priority queue completely. We adopted a data-parallel approach to approximate the behavior of a priority queue in this scenario. Euclidean distance is used as an approximation of geodesic distance on mesh.

```
NaiveFlooding():
  define R[NUM_REGIONS] as an array of Regions
  define T[NUM_FACES] as an array of Triangle faces

  parallel for each Ri in R:
    Ri.seed = pick a unique Tj from T at random
  parallel for each Tj in T:
    for each Ri in all Regions:
      compute distance(Tj.centroid, Ri.centroid)
    R' = nearest M Regions of Tj
    Tj.assignedRegion = pick r in R' with minimum E(Tj, r)
```

Figure 37 Pseudo-code for Naïve Data-parallel Flooding

The diagram below illustrates the naïve data-parallel flooding procedure.

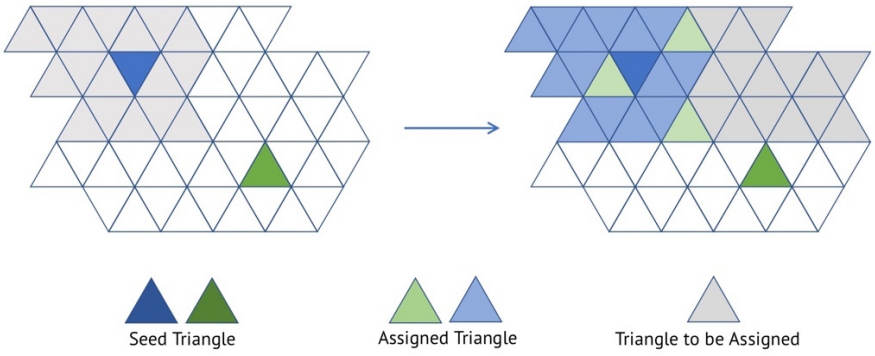

Figure 38 Illustration of Naïve Data-parallel Flooding procedure

This data-parallel procedure makes no guarantee on region connectivity. Therefore, to ensure connectivity of partitioning result, one iteration of sequential flooding is added to its end after the parallel partitioning converges. Considering each iteration, this algorithm has total work of O(N * P), where N is the number of triangles in the mesh and P is the number of regions desired. Based on implementation above, the span of it is O(P). Although computing on each triangle independently can result in good parallelism, a significant amount of extra work has been introduced. The algorithm may perform badly when the number of desired regions P become quite large.

6.1.2 Parallel Queued Flooding on GPU

To account for the connectivity requirement, we introduce a more sophisticated parallel flooding algorithm. Considering the sequential flooding algorithm where regions are

“grown” out of seed faces based on a global priority queue, it is natural to consider having multiple regions “grow” in parallel.

One important consideration for parallel flooding is to deal with scenarios where edges from different regions meet during the growing process. To deal with boundary conditions, we referred to the distance comparing procedure described by Fan et al.[49] and the cluster cleaning algorithm described by Valette et al.[50], and developed our parallel queued flooding algorithm upon these two solutions:

```
QueuedFlooding():
  define two queues Q1, Q2
  define R[NUM_REGIONS] as an array of Regions
  define T[NUM_FACES] as an array of Triangle faces

  parallel for each Ri in R:
    Ri.seed = pick a unique Tj from T at random
    insert Ri.seed to Q1

  while Q1 is not empty:
    parallel for each triangle Tj in Q1:
      find the region assignment of Tj's three neighbors
      R' = t.assignedRegion for all t in Tj.neighbors()
      Rj = pick r in R' with minimum E(Tj, r)
      if Rj == Tj.assignedRegion:
        continue
      else:
        Tj.assignedRegion = Rj
        insert t to Q2 for all t in Tj.neighbors()
    synchronize_thread()
    replace Q1 with Q2
    clear Q2
```

Figure 39 Pseudo-code for Parallel Queued Flooding

The diagram below illustrates the parallel queued flooding procedure.

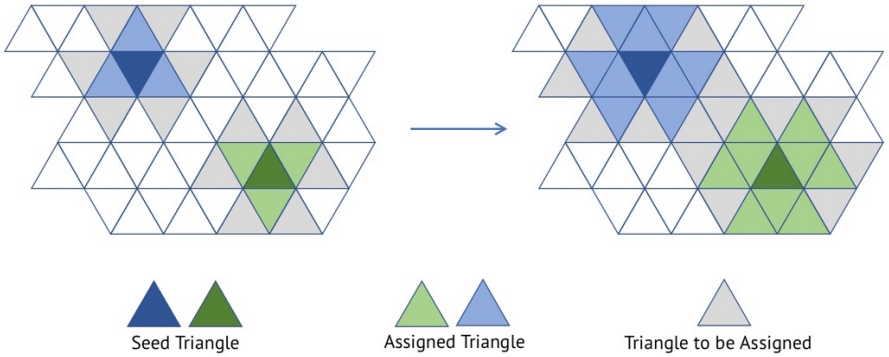

Figure 40 Illustration of Parallel Queued Flooding procedure

Specifically in the context above, queues are implemented using as an extension of line-of-sight problem using prefix sum described by Blelloch[51]. Detailed description is as follows:

1) *allocate* `output_array[NUM_FACES+1]` *and* `indicator_array[NUM_FACES+1]`*, initialized to zero*
2) *when adding a triangle* `Tj` *to queue, mark* `indicator_array[j]` *as 1*
3) *exclusive scan on* `indicator_array`*, store result in an array* `scan_result`
4) ***for all*** *index* `j` *of* `indicator_array`*:*

```
if indicator_array[j] == 1:
  output_array[scan_result[j]] = j
```

5) `output_array` *is now a queue with contiguous face indices*

We used the parallel exclusive scan (prefix sum) function provided in the CUDA thrust library[52]. Because of the use of exclusive scan, the total expected work in each iteration is O(NlogN * sqrt(N/P)). Based on the implementation above, the expected span is O(logN * sqrt(N/P)), as exclusive scan has the span of O(logN). From the preliminary analysis, Parallel Queued Flooding is arguably better than Naïve Data-parallel flooding due to the fact that it introduces less extra work to the sequential algorithm and that it enforces better connectivity constraints (there are much fewer disconnected regions).

6.1.3 Hybrid Edge-collapse Partitioning

Considering the fact that acceleration with GPU has hardware limitations, in that the underlying machine has to support compiling and running general-purpose GPU code,

GPU parallelization might not be a universally compatible solution to accelerating Variational Shape Approximation. Therefore, we propose a different acceleration approach that effectively combines the greedy simplification algorithm based on quadric error metrics[25] and Variational Shape Approximation[39].

The basic idea is that, when dealing with large input meshes, we first use the greedy method to decimate the original mesh to a certain extent that face count is low and the input shape is well preserved. After decimation, we run Variational Shape Approximation on the much simpler mesh for a user-friendly polygon output. It is noticeable that the most time consuming iterative optimization algorithm is now running on a much smaller input, therefore computation time can be significantly reduced for multiple iterations.

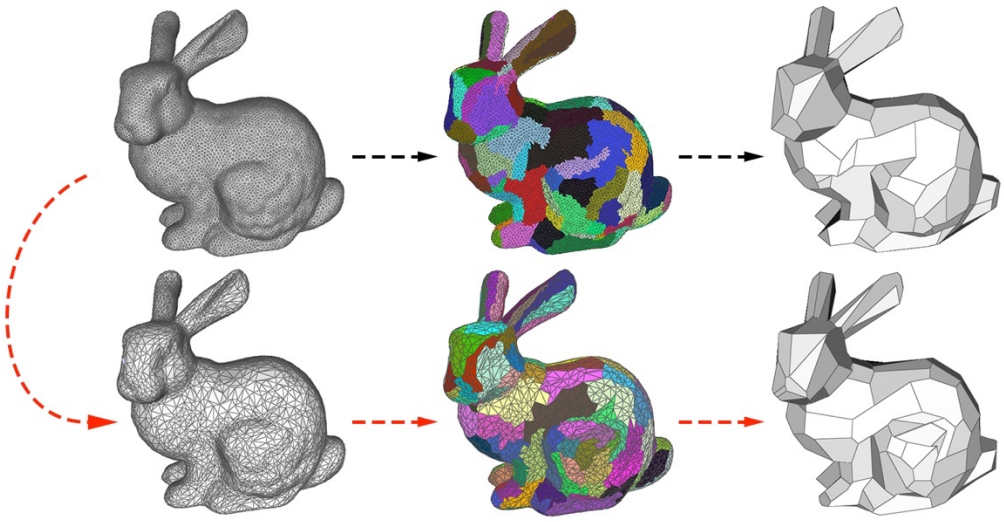

Figure 41 Illustration of Hybrid Edge-collapse Partitioning procedure

6.2 Experiments and Results

We tested the sequential flooding and the hybrid flooding algorithm on a Intel® Core™ i7-6700 Processor (8M Cache, 4.00 GHz), and the parallel algorithm on 2 different GPUs:

- NVidia GeForce GTX 1080 with 20 SMs
- NVidia GeForce GTX 1060 with 9 SMs

The algorithms are tested against three models of different face counts from the Stanford 3D Scanning Repository[53], and number of partitions are determined with respect to the complexity of model.

6.2.1 Parallel Speedup

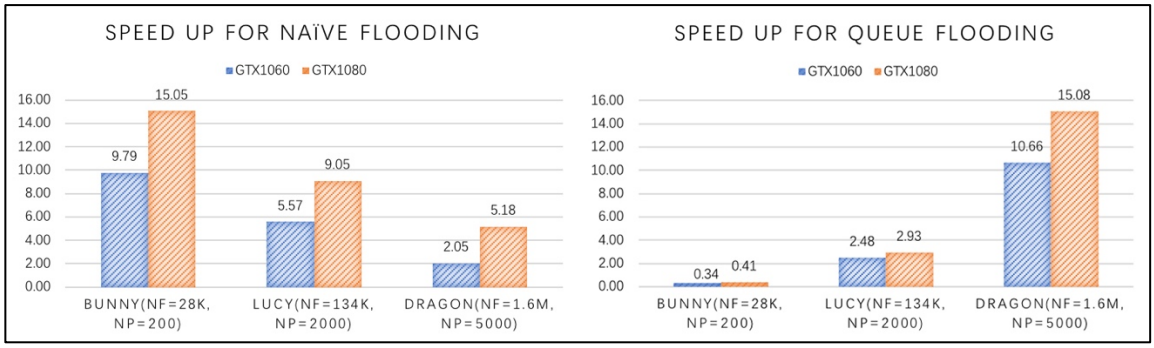

Figure 42 Parallel Speedup for models with different complexity

As seen from the speedup results, although Naive Data-parallel Flooding achieved a near-linear speedup across GPUs, its speedup with respect to the sequential algorithm decreases as the size of input increases. This is because the algorithm has introduced additional work that is linear with respect to P (the number of desired partitions).

Parallel Queued Flooding involves flooding in lock step with heavy synchronizations among threads. Therefore it cannot achieve a linear speedup across different GPUs. Its speedup increases as the size of input increases. From the expression of the span of the algorithm, O(logN * sqrt(N/P)), we can see that the amount of parallelism will increase as P increases. In practice, for large input, P is often set to a larger value compared to small inputs. Therefore, Parallel Queued VSA is more suitable for simplifying large meshes. For smaller inputs, this method actually performs worse than the sequential approach on the CPU.

6.2.2 Convergence for parallel methods

Based on Cohen-Steiner et al., the goal of the algorithm is to minimize total distortion error which is defined as:

$$\sum_i ||\boldsymbol{n}_i - \boldsymbol{N}_{Ri}||^2 \, |T_i|$$

where $|T_i|$ represents the area of the ith triangle, $\boldsymbol{n}_i$ represents the normal of ith triangle, and $\boldsymbol{N}_{Ri}$ represents the area weighted normal of the region this triangle belongs to.

Therefore, both the final value of the distortion error and how fast it will converge reflect the quality of the algorithm. Shown below are the convergence curve we obtained on three different inputs.

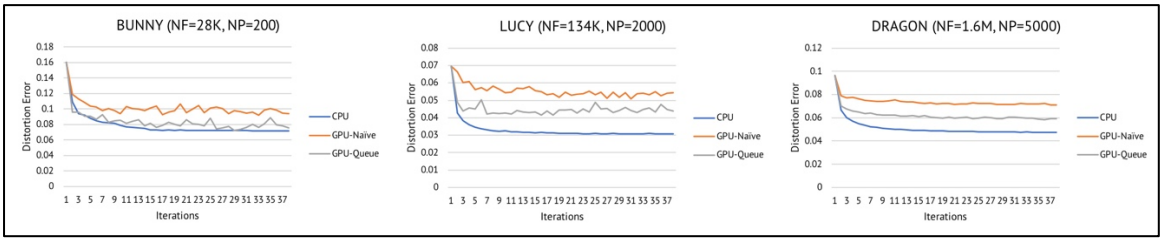

Figure 43 Convergence curve for models with different complexity

We can see that all three algorithms converge very fast in less than 10 iterations. Queued Flooding performs better than Naive Flooding in terms of minimizing distortion error. Both parallel approaches are worse than the sequential algorithm. The reason is that although parallel flooding also tries to minimize distortion error, it does not strictly guarantee connected output regions. In order to feed the output of the partitioning step to the meshing step, one iteration of sequential flooding is added after the parallel algorithm converges. Therefore, to ensure connectivity, the value of distortion error is increased by the final iteration of sequential flooding.

6.2.3 Hybrid Method Performance

As for the Hybrid Edge-collapse Partitioning, we first decimate the input mesh to reach a desired fraction (from 1/2 to 1/32) of face counts. Partitioning is computed using the sequential algorithm and record the proxy information, namely, seed faces and area weighted normal vectors. We then compute the distortion error on the original mesh with proxy information acquired from the previous step. For sequential partitioning, we set the number of iterations to 20 to match the previous parallel approaches and to let the algorithm converge.

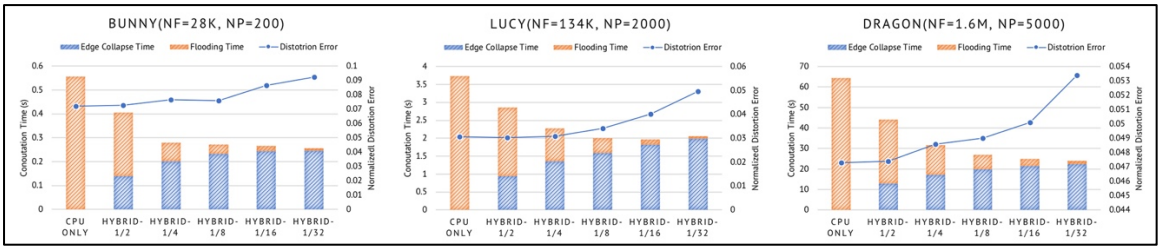

Figure 44 Performance curve for hybrid method

With a very straightforward implementation of Garland et al.'s simplification algorithm[25], we can achieve approximately 2X speedup of the partitioning process. Although this value seems not satisfying, it is noticeable that the actual computation time of flooding drops drastically with the increase of the number of faces decimated. Theoretically, as flooding is of O(NlogN) complexity, we can achieve a super linear speedup when we decrease the term N. In practice, it is very likely that the flooding procedure will be invoked multiple

times on the same input mesh for interactive simplification purposes. Therefore, huge performance gain can be expected from repeated calls to flooding on the same input.

In terms of output quality, we can conclude from the figures that it is safe to first decimate 90% to 95% of the faces before suffering from noticeable degraded output quality.

7 Acknowledgements

I would like to take this opportunity to express my most sincere gratitude to all the people who helped me, supported me and guided me throughout my graduate study.

First, I would like to thank my thesis advisor, Professor Daniel Cardoso Llach for all the guidance he provided during my entire graduate study. Thank you Daniel for all those great advice and support for my thesis work. Also, I'd like to thank Professor Keenan Crane for giving me great suggestions for the choice of the key algorithm.

In the meantime, special thanks to CMU 15-662 Computer Graphics and 15-618 Parallel Computer Architecture and Programming courses for teaching me all the necessary knowledge and providing test equipment.

I'd also like to acknowledge Min Park, my teammate from 15-618, for briefly joining in this project and contributed to the testing environment. Furthermore, to all anonymous participant of the 3D scan workshop, thank you for your interest and cooperation.

I want to offer my final acknowledgement to my girlfriend, my parents and all my friends. Thank you all for supporting me during this whole time!

Hope my thesis work can be an inspiration for future researches and benefit the computational design community.

8 References

[1] S. Clarke and R. Engelbach, *Ancient Egyptian construction and architecture*. Courier Corporation, 1930.

[2] 丘光明, 中国歷代度量衡考. 科学出版社, 1992.

[3] Creaform, “THE HISTORY OF METROLOGY FROM GALILEO TO OPTICAL SYSTEMS,” 2016. [Online]. Available: https://www.creaform3d.com/blog/2016/10/the-history-of-metrology-from-galileo-to-optical-systems/. [Accessed: 16-Apr-2018].

[4] Center for Photogrammetric Training, “HISTORY OF PHOTOGRAMMETRY,” 2008. [Online]. Available: http://wayback.archive-it.org/all/20090227061949/http://www.ferris.edu/faculty/burtchr/sure340/notes/History.pdf. [Accessed: 16-Apr-2018].

[5] S. H. Wang, C. J. Tay, C. Quan, H. M. Shang, and Z. F. Zhou, “Laser integrated measurement of surface roughness and micro-displacement,” *Measurement Science and Technology*, vol. 11, no. 5, pp. 454–458, May 2000.

[6] S. Filippi and B. Motyl, “Virtual and rapid prototyping by means of 3D optical acquisition and CAD modeling: application to cultural heritage and to the automotive domain,” in *Virtual Modelling and Rapid Manufacturing: Advanced Research in Virtual and Rapid Prototyping Proc. 2nd Int. Conf. on Advanced Research in Virtual and Rapid Prototyping, 28 Sep-1 Oct 2005, Leiria, Portugal*, 2005, p. 251.

[7] U. Buck, S. Naether, B. Räss, and M. J. Thali, “Accident or homicide – Virtual crime scene reconstruction using 3D methods,” *Forensic Science International*, vol. 225, no. 1–3, pp. 75–84, Feb. 2013.

[8] ArchDaily, “ICD-ITKE Research Pavilion 2013-14 / ICD-ITKE University of Stuttgart,” *ArchDaily*, 2016. [Online]. Available: http://www.archdaily.com/522408/icd-itke-research-pavilion-2015-icd-itke-university-of-stuttgart. [Accessed: 30-May-2016].

[9] G. Sansoni, M. Trebeschi, and F. Docchio, “State-of-The-Art and Applications of 3D Imaging Sensors in Industry, Cultural Heritage, Medicine, and Criminal Investigation,” *Sensors*, vol. 9, no. 1, pp. 568–601, Jan. 2009.

[10] Robert McDermott, “Robert Remembers: the VW Bug,” *The Utah Teapot*, 2003. [Online]. Available: https://www.cs.utah.edu/docs/misc/Uteapot03.pdf. [Accessed: 16-Apr-2018].

[11] School of Computing at the University of Utah, “Mapping Sutherland’s Volkswagen - CHM Revolution,” 1977. [Online]. Available: http://www.computerhistory.org/revolution/computer-graphics-music-and-art/15/206/560. [Accessed: 05-Mar-2018].

[12] CAAD LAB National Taiwan University of Science and Technology, “網站成果資源:大型歷史建築文物數位保存-台北市大龍峒保安宮數位保存-數位典藏與數位學習國家型科技計畫成果入口網.” [Online]. Available: http://digitalarchives.tw/site_detail.jsp?id=1304. [Accessed: 16-Apr-2018].

[13] K. N. Otto and others, *Product design: techniques in reverse engineering and new product development*. 清华大学出版社有限公司, 2003.

[14] L. C. Chen, “Surface digitization techniques for reverse engineering.” 2002.

[15] *Code for Design of Office Buildings JGJ67-2006*. Ministry of Housing and Urban-Rural Construction of the People’s Republic of China, 2006.

[16] G. Sansoni and F. Docchio, “Three-dimensional optical measurements and reverse engineering for automotive applications,” *Robotics and Computer-Integrated Manufacturing*, vol. 20, pp. 359–367, 2004.

[17] Shive-Hattery Architecture + Engineering, “Halligan Coffee Building Renovation 3D Scan, Davenport, Iowa,” 2018. [Online]. Available: http://www.shive-hattery.com/Services/Specialty-Services/3D-Scanning/Project/Halligan-Coffee-Building-Renovation-3D-Scan.aspx. [Accessed: 16-Apr-2018].

[18] “Fish, Gehry Partners | Barcelona | Spain | MIMOA.” [Online]. Available: https://www.mimoa.eu/projects/Spain/Barcelona/Fish/. [Accessed: 20-Feb-2018].

[19] P. Szalapaj, *CAD principles for architectural design*. Routledge, 2013.

[20] Greg Lynn, “A Fish Is Kind of Aerodynamic,” *Origins of the Digital*, 2016.

[21] Taylor Wang, “Mesh/Point Cloud Processing | My point cloud.” [Online]. Available: https://taylorwang.wordpress.com/category/meshpoint-cloud-processing/. [Accessed: 06-Nov-2017].

[22] P. Alliez, G. Ucelli, C. Gotsman, and M. Attene, “Recent Advances in Remeshing of Surfaces.”

[23] T. Várady, “Reverse engineering of geometric models—an introduction,” *Computer-Aided Design*, vol. 29, no. 4, pp. 255–268, Apr. 1997.

[24] H. Hoppe, T. Derose, T. Duchamp, J. Mcdonald, and W. Stuetzle, “Mesh Optimization.”

[25] M. Garland and P. S. Heckbert, “Surface simplification using quadric error

metrics," *Proceedings of the 24th annual conference on Computer graphics and interactive techniques - SIGGRAPH '97*, pp. 209–216, 1997.

[26] M. Garland and P. S. Heckbert, "Simplifying surfaces with color and texture using quadric error metrics," in *Proceedings Visualization '98 (Cat. No.98CB36276)*, p. 263–269,.

[27] H. Hoppe, "New Quadric Metric for Simplifying Meshes with Appearance Attributes."

[28] Y. Li and Q. Zhu, "A New Mesh Simplification Algorithm Based on Quadric Error Metrics," in *2008 International Conference on Advanced Computer Theory and Engineering*, 2008, pp. 528–532.

[29] P. Schroder and W. Sweldens, "Digital geometry processing," in *Sixth Annual Symposium on Frontiers of Engineering*, 2001, pp. 41–44.

[30] P. Alliez, M. Meyer, M. Desbrun, P. Alliez, M. Meyer, and M. Desbrun, "Interactive geometry remeshing," in *Proceedings of the 29th annual conference on Computer graphics and interactive techniques - SIGGRAPH '02*, 2002, vol. 21, no. 3, p. 347.

[31] P. Alliez *et al.*, "Anisotropic polygonal remeshing," in *ACM SIGGRAPH 2003 Papers on - SIGGRAPH '03*, 2003, vol. 22, no. 3, p. 485.

[32] S. Dong, S. Kircher, and M. Garland, "Harmonic functions for quadrilateral remeshing of arbitrary manifolds," *Computer Aided Geometric Design*, vol. 22, no. 5, pp. 392–423, Jul. 2005.

[33] S. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, Mar. 1982.

[34] J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A K-Means Clustering Algorithm," *Applied Statistics*, vol. 28, no. 1, p. 100, 1979.

[35] Q. Du, V. Faber, and M. Gunzburger, "Centroidal Voronoi Tessellations: Applications and Algorithms," *SIAM Review*, vol. 41, no. 4, pp. 637–676, Jan. 1999.

[36] P. Alliez, E. C. de Verdire, O. Devillers, and M. Isenburg, "Isotropic surface remeshing," in *2003 Shape Modeling International.*, pp. 49–58.

[37] G. Peyre and L. Cohen, "Surface segmentation using geodesic centroidal tesselation," in *Proceedings. 2nd International Symposium on 3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004.*, pp. 995–1002.

[38] G. Rong, Y. Liu, W. Wang, X. Yin, D. Gu, and X. Guo, "GPU-assisted computation of centroidal voronoi tessellation," *IEEE Transactions on Visualization and*

Computer Graphics, vol. 17, no. 3, pp. 345–356, Mar. 2011.

[39] D. Cohen-Steiner, P. Alliez, M. Desbrun, D. Cohen-Steiner, P. Alliez, and M. Desbrun, "Variational shape approximation," in *ACM SIGGRAPH 2004 Papers on - SIGGRAPH '04*, 2004, vol. 23, no. 3, p. 905.

[40] W. Ma and J.-P. Kruth, "NURBS curve and surface fitting for reverse engineering," *The International Journal of Advanced Manufacturing Technology*, vol. 14, no. 12, pp. 918–927, Dec. 1998.

[41] A. Dimitrov and M. Golparvar-Fard, "Robust NURBS Surface Fitting from Unorganized 3D Point Clouds for Infrastructure As-Built Modeling," in *Computing in Civil and Building Engineering (2014)*, 2014, pp. 81–88.

[42] D. Brujic, I. Ainsworth, and M. Ristic, "Fast and accurate NURBS fitting for reverse engineering," *The International Journal of Advanced Manufacturing Technology*, vol. 54, no. 5–8, pp. 691–700, May 2011.

[43] D. I. S. Adi, S. M. Shamsuddin, and A. Ali, "Particle Swarm Optimization for NURBS Curve Fitting," in *2009 Sixth International Conference on Computer Graphics, Imaging and Visualization*, 2009, pp. 259–263.

[44] G. Lavoué, M. Tola, F. Dupont, and G. Lavou, "MEPP - 3D mesh processing platform," *Proceedings of the International Conference on Computer Graphics Theory and Applications (GRAPP 2012) and International Conference on Information Visualization Theory and Applications (IVAPP 2012)*, pp. 206–210, 2012.

[45] P. Cignoni *et al.*, "MeshLab: an Open-Source Mesh Processing Tool," *Sixth Eurographics Italian Chapter Conference*, pp. 129–136, 2008.

[46] Autodesk Inc., "Maya | Computer Animation & Modeling Software | Autodesk." [Online]. Available: https://www.autodesk.com/products/maya/overview. [Accessed: 20-Apr-2018].

[47] Autodesk Inc., "Dependency graph | Maya | Autodesk Knowledge Network." [Online]. Available: https://knowledge.autodesk.com/support/maya/learn-explore/caas/CloudHelp/cloudhelp/2017/ENU/Maya/files/GUID-51096BC4-32B7-4391-BE39-21641B374745-htm.html. [Accessed: 20-Apr-2018].

[48] M. Zechner and M. Granitzer, "Accelerating k-means on the graphics processor via CUDA," in *Proceedings of the 1st International Conference on Intensive Applications and Services, INTENSIVE 2009*, 2009, pp. 7–15.

[49] F. Fan, F. (Frank) Cheng, C. Huang, Y. Li, J. Wang, and S. Lai, "Mesh clustering by approximating centroidal Voronoi tessellation," in *2009 SIAM/ACM Joint Conference on Geometric and Physical Modeling on - SPM '09*, 2009, p. 301.

[50] S. Valette, J.-M. Chassery, and R. Prost, “Generic Remeshing of 3D Triangular Meshes with Metric-Dependent Discrete Voronoi Diagrams,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 2, pp. 369–381, Mar. 2008.

[51] G. E. Blelloch, “Prefix Sums and Their Applications,” *Computer*, pp. 35–60, 1990.

[52] J. Hoberock and N. Bell, “Thrust - Parallel Algorithms Library.” [Online]. Available: https://thrust.github.io/. [Accessed: 25-Jan-2018].

[53] “The Stanford 3D Scanning Repository.” [Online]. Available: http://graphics.stanford.edu/data/3Dscanrep/. [Accessed: 06-Feb-2018].